VLDB Journal (2005) 14: 112–136 / Digital Object Identifier (DOI) 10.1007/s00778-003-0119-8

Mediators over taxonomy-based information sources

Yannis Tzitzikas[1,⋆], **Nicolas Spyratos**[2], **Panos Constantopoulos**[1]

[1] Department of Computer Science, University of Crete, Greece and Institute of Computer Science, ICS-FORTH;
e-mail: {tzitzik|panos}@csi.forth.gr
[2] Laboratoire de Recherche en Informatique, Université de Paris-Sud, France;
e-mail: spyratos@lri.fr

Edited by M. Missikoff. Received: November 29, 2001 / Accepted: October 30, 2003

Abstract. We propose a mediator model for providing integrated and unified access to multiple taxonomy-based sources. Each source comprises a taxonomy and a database that indexes objects under the terms of the taxonomy. A mediator comprises a taxonomy and a set of relations between the mediator's and the sources' terms, called articulations. By combining different modes of query evaluation at the sources and the mediator and different types of query translation, a flexible, efficient scheme of mediator operation is obtained that can accommodate various application needs and levels of answer quality. We adopt a simple conceptual modeling approach (taxonomies and inter-taxonomy mappings) and we illustrate its advantages in terms of ease of use, uniformity, scalability, and efficiency. These characteristics make this proposal appropriate for a large-scale network of sources and mediators.

Keywords: Mediators – Taxonomies – Approximate query translation – Information integration

1 Introduction

The need for integrated and unified access to multiple information sources has stimulated research on *mediators* (initially proposed in [78]). Roughly, a mediator is a secondary information source aiming at providing a uniform interface to a number of underlying sources (which may be primary or secondary). Users submit queries to the mediator. Upon receiving a user query, the mediator queries the underlying sources. This involves selecting the sources to be queried and formulating the query to be sent to each source. These tasks are accomplished based on what the mediator "knows" about the underlying sources. Finally, the mediator appropriately combines the returned results and delivers the final answer to the user.

In this paper we consider information sources over a common domain consisting of a denumerable set of objects. For example, in the environment of the Web, the domain could be the set of all Web pages, specifically, the set of all pointers to Web pages. Each source has a *taxonomy*, i.e., a structured set of names, or *terms*, that are familiar to the users of the source. In particular, the taxonomies considered in this paper consist of a set of terms structured by a subsumption relation. In addition, each source maintains a database storing objects that are of interest to its users. Specifically, each object in the database of a source is indexed under one or more terms of the taxonomy of that source. In quest for objects of interest, a user can browse the source taxonomy until he reaches the desired terms, or he can query the source by submitting a boolean expression of terms. The source will then return the appropriate set of objects. In the environment of the Web, general-purpose catalogs, such as Yahoo! or Open Directory,[1] domain-specific catalogs/gateways (e.g., for medicine, physics, tourism), as well as personal bookmarks of Web browsers can be considered as examples of such sources.

However, although several sources may carry information about the same domain, they usually employ different taxonomies, with terms that correspond to different natural languages, or different levels of granularity. For example, consider two sources S_1 and S_2 that both provide access to electronic products as shown in Figs. 1a and 1b. Each source consists of a taxonomy plus a database that indexes objects under the terms of that taxonomy. However, the two sources provide *different* information about electronic products, as seen in the figures. Suppose now that we want to provide unified access to these two sources through a single taxonomy that is familiar to a specific group of users. An example of such a unifying taxonomy is shown in Fig. 1c and constitutes part of what we call a "mediator".

A *mediator* is a secondary source that can bridge the heterogeneities that may exist between two or more sources in order to provide unified access to those sources. Specifically, a mediator has a taxonomy with terminology and structuring that reflects the needs of its potential users but does *not* maintain a database of objects. Instead, the mediator maintains a number of *articulations* to the sources. An articulation to a source is a set of relationships between the terms of the mediator and the terms of that source. These relationships are defined by the designer of the mediator at design time and are

---

[⋆] Current address: Istituto di Scienza e Tecnologie dell' Informazione, ISTI-CNR, Pisa, Italy

[1] http://dmoz.org

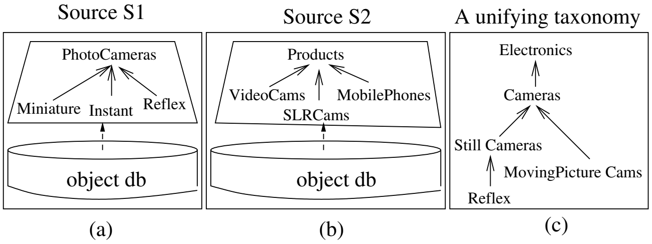


Fig. 1. Two sources providing access to electronic products

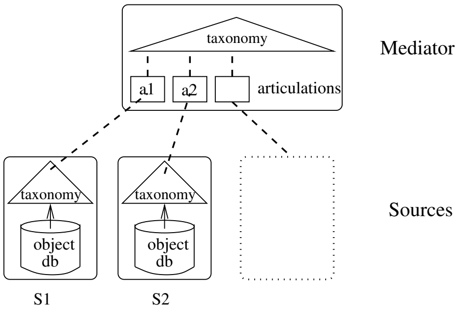


Fig. 2. Mediator architecture

stored at the mediator. Figure 2 shows the general architecture of a mediator.

Users formulate queries over the taxonomy of the mediator, and it is the task of the mediator to choose the sources to be queried and to formulate the query to be sent to each source. To this end, the mediator uses the articulations to translate queries over its own taxonomy to queries over the taxonomies of the articulated sources. Then it is again the task of the mediator to combine the results returned by the sources appropriately to produce the final answer.

An essential feature that distinguishes our work is that we adopt a simple conceptual modeling approach for both sources and mediators. This conceptual modeling approach has the following advantages: (a) it is very easy to create the conceptual model of a source or a mediator, and (b) the integration of information from multiple sources can be done very easily. Indeed, as we shall see, the articulations offer a *uniform* and easy-to-use method to bridge naming, contextual, and granularity heterogeneities between the conceptual models of the sources. Given this conceptual modeling approach, the mediator does not have to tackle complex structural differences between the sources (as happens in mediators for relational databases).

Another essential feature that distinguishes our approach is that a source can provide two types of answer to a given query, namely, a *sure* answer or a *possible* answer. The first type of answer is appropriate for a user who does not want to retrieve objects that are not relevant to his information need, while the second is for a user who does not want to miss objects that are relevant to his information need. Moreover, as exact translation of user queries is not always possible, a user query to the mediator admits two types of approximation – *lower* or *upper* translation.

What kind of translation will be used at the mediator level and what kind of answer will be requested at the source level is decided by the mediator designer at design time and/or the mediator user at query time. Therefore, a prominent feature of our approach is that sources and mediators can operate in a variety of modes according to specific application needs. As a consequence, our mediators are quite flexible and can adapt to a variety of situations.

A main objective of this paper is to prescribe easy-to-use and formally sound methods for building mediators. In the context of the Web, our mediators can be used for providing unified access to multiple Web catalogs. An advantage of our approach is that a mediator can be constructed quite easily; therefore ordinary Web users can use it to define their own mediators. In this sense, this approach can be used for personalizing existing Web catalogs. Furthermore, it can be used for building mediators over XFML [1] information bases (XFML aims at applying the faceted classification paradigm in the context of the Web).

The remainder of the paper is organized as follows. Section 2 describes the information sources and the query answering process at a single source. Section 3 defines the architecture of a mediator over a set of sources and the different modes in which a mediator can operate. Section 4 discusses query evaluation, and Sect. 5 discusses enhancements of the query answering process. Section 6 discusses various extensions of our model. Section 7 discusses related work and, finally, Sect. 8 concludes the paper and discusses further research. All proofs are given in the Appendix.

2 The sources

Why taxonomies

Taxonomies are probably the oldest and most widely used conceptual modeling tool. Nevertheless, it is a powerful tool still used in Web directories (e.g., Google and Yahoo!), content management (hierarchical structures are used to classify documents), Web publishing (many authoring tools require one to organize the contents of portals according to some hierarchical structure), Web services (services are typically classified in a hierarchical form), marketplaces (goods are classified in hierarchical catalogs), personal file systems, personal bookmarks for the Web, libraries (e.g., Thesauri [40]), and in very large collections of objects (e.g., see [61]). Although more sophisticated conceptual models (including concepts, attributes, relations, and axioms) have emerged and have recently been employed even for metatagging on the Web [48,75], almost all of them have a backbone consisting of a subsumption hierarchy, i.e., a taxonomy.

Furthermore, a taxonomy-based conceptual modeling approach has several advantages in large and open domains. In a very broad domain, such as the set of all Web pages, it is not easy to identify the classes of the domain because the domain is too wide and different users, or applications, conceptualize it differently, e.g., one class of the conceptual model according to one user may correspond to a value of an attribute of a class of the conceptual model according to another user. For example, Fig. 3 shows two different conceptual models for the same domain. We consider only two objects of the domain, denoted by the natural numbers 1 and 2.

The conceptual model of Fig. 3a is appropriate for building an information system for a furniture store, while the conceptual model of Fig. 3b is appropriate for building an information

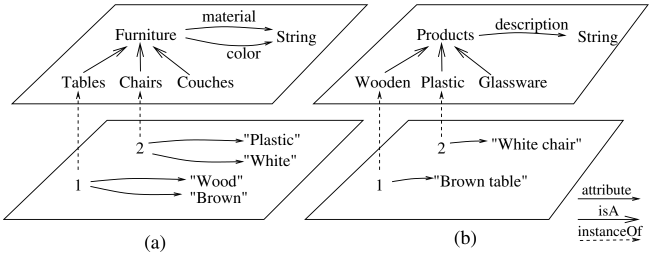


Fig. 3. Two different conceptual models for the same domain

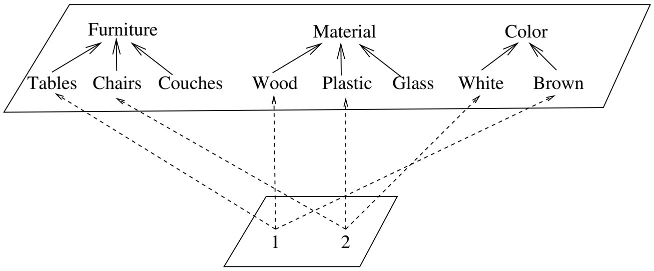


Fig. 4. A conceptual model that consists of terms and subsumption links only

system for a department store. The classes of model a, i.e., the classes `Tables`, `Chairs`, and `Couches`, have been defined so as to distinguish the objects of the domain according to their *use*. On the other hand, the classes of model b, i.e., the classes `Wooden`, `Plastic`, and `Glassware`, have been defined so as to distinguish the objects of the domain according to their *material*. This kind of distinction is useful for a department store, as it determines (to some degree) the placement of the objects in the various departments of the store. Figure 4 shows a conceptual model for the same domain that consists of terms and subsumption links only, i.e., a *taxonomy*. This conceptual modeling approach seems to be more application independent. All criteria (characteristics) for distinguishing the objects are equally "honored".

A simple conceptual modeling approach, where each conceptual model is a taxonomy, has three main advantages. The first is that it is very easy to create the conceptual model of a source or a mediator. Even ordinary Web users can design this kind of conceptual models. Besides, the queries submitted by ordinary users are mostly bags of words, not structured queries. Furthermore, the design can be done more systematically if done following a faceted approach (e.g., see [56,58]). In addition, thanks to techniques that have emerged recently [71], taxonomies of compound terms can also be defined in a flexible and systematic manner.

The second advantage is that the simplicity and modeling uniformity of taxonomies allow for integrating the contents of several sources without having to tackle complex structural differences. Indeed, as will be seen in subsequent sections, intertaxonomy mappings offer a *uniform* method of bridging *naming, contextual,* and *granularity* heterogeneities among the taxonomies of the sources. Given this conceptual modeling approach, a mediator does not have to tackle complex structural differences between the sources, as happens with relational mediators (e.g., see [36,46]) and description logics-based mediators (e.g., see [13,42]). Moreover, it allows the integration of *schema* and *data* in a uniform manner. Another advantage of this conceptual modeling approach is that query evaluation in taxonomy-based sources and mediators can be done efficiently (polynomial time).

The third advantage is that this conceptual modeling approach makes the automatic construction of mappings possible [68]. This is also the major drawback of the current more expressive Web annotation languages.

Due to the above benefits (conceptual modeling simplicity, integration flexibility, efficient query evaluation), taxonomies are worthy of further investigation. The only assumption that we make is that the domain is a set of objects that we want to index and subsequently retrieve without concerning ourselves with the relationships that may hold between the objects of the domain.

Defining a source

Let Obj denote the set of all objects of a domain common to several information sources. A typical example of such a domain is the set of all pointers to Web pages. We assume that each source has a *taxonomy*, defined as follows:

Definition 1 A taxonomy is a pair $(T, \preceq)$, where T is a *terminology*, i.e., a set of names, or *terms*, and $\preceq$ is a *subsumption* relation over T, which is a reflexive and transitive relation over T.

If a and b are terms of T, we say that a is *subsumed* by b if $a \preceq b$; we also say that b *subsumes* a; for example, `Databases` $\preceq$ `Informatics`, `Canaries` $\preceq$ `Birds`. We say that two terms a and b are *equivalent*, and write $a \sim b$, if both $a \preceq b$ and $b \preceq a$ hold, e.g., `Computer Science` $\sim$ `Informatics`. Note that the subsumption relation is a preorder over T and that $\sim$ is an equivalence relation over the terms of T. Moreover, $\preceq$ is a partial order over the equivalence classes of terms.

We assume that, in addition to its taxonomy, each source has a stored *interpretation* I of its terminology, i.e., a function $I : T \to 2^{Obj}$ that associates each term of T with a set of objects. Here we use the symbol 2^{Obj} to denote the powerset of Obj. Figure 5 shows an example of a source.

In this and subsequent figures, the objects are represented by natural numbers and membership of objects in the interpretation of a term is indicated by a dotted arrow from the object to that term. For example, objects 1 and 3 in Fig. 5 are members of the interpretation of the term `JournalArticle` as these objects are connected to `JournalArticle` with dotted arrows. Moreover, as these are the only objects connected to `JournalArticle` by dotted arrows, they make up the interpretation of `JournalArticle`, i.e., $I(\texttt{JournalArticle}) = \{1, 3\}$.

Subsumption of terms is indicated by a continuous-line arrow from the subsumed term to the subsuming term. For example, the term `RDB` in Fig. 5 is subsumed by `DB` as there is a continuous-line arrow going from `RDB` to `DB`; this arrow indicates that `RDB` $\preceq$ `DB`.

Note that we do not represent the entire subsumption relation but rather a subset of it sufficient to generate the entire relation. In particular, we do not represent the reflexive or the transitive arrows of the subsumption relation.

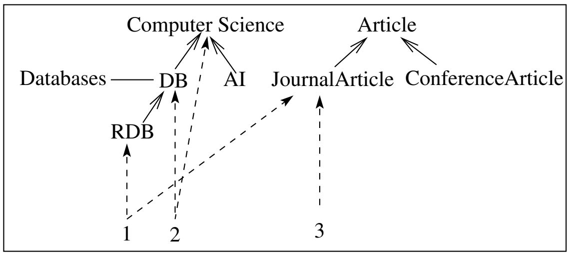


Fig. 5. Graphical representation of a source

Equivalence of terms is indicated by a continuous nonoriented line connecting the terms that are equivalent. For example, the term `Databases` is equivalent to the term `DB` since these two terms are connected by a continuous nonoriented line. Note that equivalence captures the notion of synonymy and that each equivalence class simply contains alternative terms for naming a given set of objects.

For technical reasons that will become clear shortly, we assume that every terminology T contains two special terms, the *top term*, denoted by $\top$, and the *bottom term*, denoted by $\bot$. The top term subsumes every other term t, i.e., $t \preceq \top$. The bottom term is strictly subsumed by every other term t different than top and bottom, i.e., $\bot \preceq \bot$, $\bot \preceq \top$, and $\bot \prec t$, for every t such that $t \neq \top$ and $t \neq \bot$. Moreover, we assume that every interpretation I of T satisfies the condition $I(\bot) = \emptyset$.

Querying a source

Each source responds to queries over its own terminology. A query is either a term or a combination of terms using the usual connectives $\wedge$, $\vee$, $\neg$, and (). For technical reasons that will become clear shortly, we shall also use the concept of *empty query*, denoted by ϵ. More formally, a query is defined as follows:

Definition 2 Let T be a terminology. A *query* over T is any string derived by the following grammar, where t is a term of T:

$$q ::= \; t \mid q \wedge q' \mid q \vee q' \mid q \wedge \neg q' \mid (q) \mid \epsilon.$$

Note that our use of negation corresponds to domain-restricted negation.

In what follows, given a query q we define two answers of q, which we call the *sure* and *possible* answer. To this end, we need some preliminary definitions and notations.

The set of interpretations of a given terminology T can be ordered using pointwise set inclusion.

Definition 3 Given two interpretations I, I' of T, we call I less than or equal to I', and we write $I \sqsubseteq I'$, if $I(t) \subseteq I'(t)$ for each term $t \in T$.

Note that $\sqsubseteq$ is a partial order over interpretations.

A source answers queries based on the stored interpretation of its terminology. However, in order for query answering to make sense, the interpretation that a source uses for answering queries must respect the structure of the source's taxonomy (i.e., the relation $\preceq$) in the following sense: if $t \preceq t'$, then $I(t) \subseteq I(t')$. For example, consider a source whose taxonomy contains only three terms: `DB`, `AI`, and `Computer Science`, where $\mathtt{DB} \preceq \mathtt{Computer\ Science}$ and $\mathtt{AI} \preceq \mathtt{Computer\ Science}$. Assume that in the stored interpretation I of the source we have: $I(\mathtt{DB}) \neq \emptyset$, $I(\mathtt{AI}) \neq \emptyset$, and $I(\mathtt{ComputerScience}) = \emptyset$. Clearly, I does not respect the structure of the taxonomy, as $\mathtt{DB} \preceq \mathtt{Computer\ Science}$, and yet $I(\mathtt{DB}) \not\subseteq I(\mathtt{ComputerScience})$. However, I is acceptable as we can "augment" it to a new interpretation I' that *does* respect the structure of the taxonomy. The interpretation I' is defined as follows: $I'(\mathtt{DB}) = I(\mathtt{DB})$, $I'(\mathtt{AI}) = I(\mathtt{AI})$, $I'(\mathtt{ComputerScience}) = I(\mathtt{ComputerScience}) \cup I(\mathtt{DB}) \cup I(\mathtt{AI})$. An interpretation such as I' that respects the structure of a taxonomy is what we call a model of that taxonomy.

Definition 4 An interpretation I is a *model* of a taxonomy $(T, \preceq)$ if for all t, t' in T, if $t \preceq t'$, then $I(t) \subseteq I(t')$.

For brevity hereafter we shall sometimes write T instead of $(T, \preceq)$ whenever no confusion is possible.

Now, as there may be several models of T in general, we assume that each source answers queries from one or more *designated* models induced by its stored interpretation. In this paper we will use two specific models for answering queries, the *sure model* and the *possible model*. To define these models formally, we need to introduce the notions of *tail* and *head* of a term.

Definition 5 Given a term $t \in T$ we define

$$\mathit{tail}(t) = \{s \in T \mid s \preceq t\} \;\; \mathit{and} \;\; \mathit{head}(t) = \{u \in T \mid t \preceq u\}.$$

Note that t and all terms that are equivalent to t belong to both $\mathit{tail}(t)$ and $\mathit{head}(t)$. Also note that $\mathit{tail}(t)$ always contains the bottom term $\bot$ and $\mathit{head}(t)$ always contains the top term $\top$.

Definition 6 Given an interpretation I of T, we define the *sure model* of T generated by I, denoted by I^-, as follows:

$$I^-(t) = \bigcup \{I(s) \mid s \in \mathit{tail}(t)\}.$$

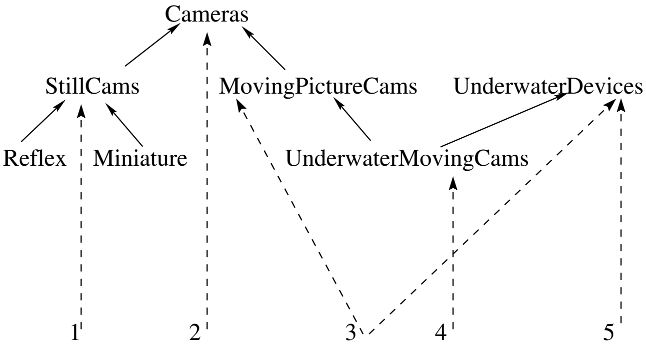
Intuitively, the stored set $I(t)$ consists of the objects that are known to be indexed under t. The set $I^-(t)$, on the other hand, consists of the objects known to be indexed under t plus the objects that are known to be indexed under terms subsumed by t. Therefore, $I^-(t)$ consists of all objects that are *surely* indexed under t with respect to I and $\preceq$. Figure 6 shows an example of a source and its sure model I^-.

Proposition 1 If I is an interpretation of T, then I^- is the unique minimal model of T that is greater than or equal to I.

Definition 7 Given a taxonomy T and interpretation I, we define the *possible model* of T generated by I, denoted by I^+, as follows:

$$I^+(t) = \bigcap \{I^-(u) \mid u \in \mathit{head}(t) \;\; \mathit{and} \;\; u \not\preceq t\}.$$

As is clear from its definition, the set $I^+(t)$ consists of the objects known to be indexed under each term strictly subsuming t. Therefore, $I^+(t)$ consists of all objects that are *possibly* indexed under t with respect to I and $\preceq$. An example of the



Term	I	I^-	I^+
$\bot$	$\emptyset$	$\emptyset$	$\emptyset$
$\top$	$\emptyset$	$\{\mathbf{1,2,3,4,5}\}$	$\{1,2,3,4,5\}$
Cameras	$\{2\}$	$\{\mathbf{1},2,\mathbf{3,4}\}$	$\{1,2,3,4,\mathbf{5}\}$
StillCams	$\{1\}$	$\{1\}$	$\{1,\mathbf{2,3,4}\}$
Reflex	$\emptyset$	$\emptyset$	$\{\mathbf{1}\}$
Miniature	$\emptyset$	$\emptyset$	$\{\mathbf{1}\}$
MovingPictureCams	$\{3\}$	$\{3,\mathbf{4}\}$	$\{\mathbf{1,2},3,4\}$
UnderwaterMovingCams	$\{4\}$	$\{4\}$	$\{\mathbf{3},4\}$
UnderwaterDevices	$\{3,5\}$	$\{3,\mathbf{4},5\}$	$\{\mathbf{1,2},3,4,5\}$

Fig. 6. Graphical representation of a source

possible model of a source is given in Fig. 6. In this example we have

$$I^+(\texttt{Reflex}) = \{1\}$$
$$I^+(\texttt{UnderwaterMovingCams}) = \{3,4\}.$$

Note that the possible interpretations of the terms `Cameras` and `UnderwaterDevices` are the set of *all* stored objects. This is so because the head of each of these terms contains only the term itself and the top term $\top$; thus we have

$$I^+(\texttt{Cameras}) = I^+(\texttt{UnderwaterDevices}) = I^-(\top) = \\ = \bigcup\{I(s) \mid s \preceq \top\}.$$

Note that since $head(\top) = \{\top\}$, the set $\{\ u \in head(\top) \mid u \not\sim \top\}$ is empty. This means that $I^+(\top)$, i.e., $\bigcap\{I^-(u) \mid u \in head(\top) \text{ and } \ u \not\sim \top\}$ is actually the intersection of an empty family of subsets of Obj. However, according to the Zermelo axioms of set theory[2] (see [10] for an overview), the intersection of an empty family of subsets of a universe is equal to the universe. In our case, the universe is the set of all objects known to the source, i.e., the set $I^-(\top)$; thus we conclude that $I^+(\top) = I^-(\top)$.

Proposition 2 If I is an interpretation of T, then I^+ is a model of T and $I \sqsubseteq I^- \sqsubseteq I^+$.

It follows from the above proposition that for every term t we have $I^-(t) \subseteq I^+(t)$.

We view the stored interpretation I as the result of indexing. However, although we may assume that indexing is done correctly, certain objects may not have been indexed under all terms that could apply to them. For example, object 1 in Fig. 6 is indexed under `StillCams` but not under `Cameras`, and object 3 is indexed under `MovingPictureCams` and `UnderwaterDevices` but not under `UnderwaterMovingCams`. Note that object 3 could in fact be an `UnderwaterMovingCamera` but was not indexed under this term because either the indexer did not use this term or the term `UnderwaterMovingCamera` was defined after the indexing of object 3 was performed.

Consequently, given a query that consists of a single term t, we may want to answer it in either of two ways: (a) by including in the answer only objects that are known to be indexed under t or (b) by including in the answer objects that are possibly indexed under t. In the first case, the answer is the set $I^-(t)$, while in the second it is the set $I^+(t)$.

Remark. If we consider that each term corresponds to a property or characteristic of the objects of a domain, then $t \preceq t'$ means that if an object has the property t, then it also has the property t'. In this view, $I(t)$ consists of the objects where each has the set of properties $head(t)$, $I^-(t)$ consists of the objects where each has at least the set of properties $head(t)$, i.e., some of the objects in $I^-(t)$ have one or more properties t' such that $t' \preceq t$, and, finally, $I^+(t)$ consists of the objects where each has at least the set of properties $head(t) \setminus \{t\}$. $\diamond$

Referring to Definition 2, let us now define query answering for a general query q.

Definition 8 Let q be a query over a terminology T and let I be an interpretation of T.
(a) The *sure answer* of q, denoted by $I^-(q)$, is a set of objects defined as follows:

$$\begin{array}{lcl} I^-(t) & = & \bigcup\{I(s) \mid s \in tail(t)\} \\ I^-(q \wedge q') & = & I^-(q) \cap I^-(q') \\ I^-(q \vee q') & = & I^-(q) \cup I^-(q') \\ I^-(q \wedge \neg q') & = & I^-(q) \setminus I^-(q') \\ I^-(\epsilon) & = & \emptyset. \end{array}$$

(b) The *possible answer* of q, denoted by $I^+(q)$, is a set of objects defined as follows:

$$\begin{array}{lcl} I^+(t) & = & \bigcap\{I^-(u) \mid u \in head(t) \text{ and } \ u \not\sim t\} \\ I^+(q \wedge q') & = & I^+(q) \cap I^+(q') \\ I^+(q \vee q') & = & I^+(q) \cup I^+(q') \\ I^+(q \wedge \neg q') & = & I^+(q) \setminus I^-(q') \\ I^+(\epsilon) & = & \emptyset. \end{array}$$

It follows easily from the above definition that for every query q we have $I^-(q) \subseteq I^+(q)$. This means that the sure answer of a query q is always included in the possible answer of q.

Note that we interpret $I^+(q \wedge \neg q')$ by $I^+(q) \setminus I^-(q')$ and *not* by $I^+(q) \setminus I^+(q')$. This is because if we had interpreted $I^+(q \wedge \neg q')$ by $I^+(q) \setminus I^+(q')$, then we could have found queries q for which $I^-(q) \supset I^+(q)$, contrary to intuition. For example, consider a terminology T with three terms a, b, and c such that $c \preceq b \preceq a$ and an interpretation I such that $I(c) = \emptyset$, $I(b) = \{1\}$, and $I(a) = \{2\}$.

Then for $q = a \wedge \neg c$ we would have had $I^-(q) = I^-(a) \setminus I^-(c) = \{1,2\}$ and $I^+(q) = I^+(a) \setminus I^+(c) = \{2\}$, i.e., $I^-(q) \supset I^+(q)$. However, with our definition we have $I^+(a \wedge \neg c) = I^+(a) \setminus I^-(c) = \{1,2\}$, i.e., the relation $I^- \sqsubseteq I^+$ is preserved.

---

[2] We do not mean here the Zermelo-Fraenkel axioms.

User interaction with a source consists in submitting a query q plus the nature of the desired answer (sure or possible). The source then responds by computing $I^-(q)$ or $I^+(q)$ according to the user's desire. The possibility of providing two types of answer to a query can enhance the quality of user interaction with the source. For example, the user may submit a query and require a sure answer. If the sure answer is empty, this may mean either that no object has been indexed under the user's query or that the objects have been indexed at a coarser level. So if the sure answer turns out to be empty, then the user can ask for the possible answer to his query. In the possible answer, the user can see objects related to, but not necessarily indexed under, his query. Another possibility is that the sure answer to the query is not empty but the user just likes to see more objects related to his query, just at a coarser level. In this case, again, the user can ask for a possible answer to his query.

A source can be implemented using any of a number of data models. For example, using the relational model [18], a source can be implemented as a database schema consisting of three tables, one for storing the terminology, one for storing the subsumption relation, and one for storing the interpretation I.

TERMINOLOGY(term-id:`Int`, term-name:`Str`),
SUBSUMPTION(term1:`Int`, term2:`Int`),
INTERPRETATION(term-id:`Int`, obj:`Int`).

Note that each term of the terminology is stored in the form of a pair <term-id, term-name>, where "term-id" is an internal identifier.

Concerning query evaluation at a source, there are basically two approaches. The first approach consists in computing and storing the models I^- and I^+ and then using these stored models for computing answers to queries. This can be done using algorithms that follow easily from Definition 8. The advantage of this approach is that answers can be computed in a straightforward manner from the stored models. The disadvantage is increased space requirements as well as increased maintenance costs for the stored models. Indeed, whenever the taxonomy or the interpretation I changes, I^- and I^+ must be updated appropriately. This requires an efficient method for handling updates since recomputing I^- and I^+ from scratch would be inefficient.

The second approach consists of storing only the interpretation I and, whenever a query q is submitted, computing the appropriate answer, $I^-(q)$ or $I^+(q)$, using I. The computation of $I^-(q)$ can be done in a straightforward manner following Definition 8a.

The computation of $I^+(q)$ can be done again following Definition 8b but requires the previous computation of $I^-(t)$ for all terms t that subsume terms appearing in the query. The advantage of this approach is that we have no additional space requirements and no additional maintenance costs. The disadvantage is increased time cost for the computation of the answers.

The relative merits of the two approaches depend on the application at hand as well as on the frequency by which the taxonomy and/or the stored interpretation of the source are updated. In both approaches we need algorithms for computing the head and the tail of a term. However, if we compute the transitive closure of the subsumption relation by one of the existing algorithms (e.g., see [54]), then the algorithms for computing the head and tail of a term follow immediately from Definition 5. The complexity of evaluating the transitive closure of $\preceq$ is polynomial.

For instance, the time complexity of the Floyd-Warshall algorithm is cubic in the number of terms, and the space used is at most quadratic in the number of terms. If the entire subsumption relation $\preceq$ is stored, i.e., if the transitive links are stored, then the computation of $head(t)$ and $tail(t)$ can be done in $O(|\preceq|)$ time. If only the interpretation I is stored, then the computation of $I^-(t)$ requires taking the union of at most $|T|$ subsets of Obj. If U denotes the set of objects that are stored in the source,[3] then the union of two subsets of Obj can be computed in $O(|U|)$ time. Thus the computation of $I^-(t)$ can be done in $O(|T| * |U|)$ time. [4] If the sure model I^- is stored, then the computation of $I^+(t)$ requires taking the intersection of at most $|T|$ subsets of Obj. Thus the computation of $I^+(t)$ can be done in $O(|T| * |U|)$ time. If only the interpretation I is stored, then the computation of $I^+(t)$ can be done as follows:

$$I^+(t) = \bigcap_{u \succ t} \left(\bigcup \{ I(s) \mid s \preceq u \} \right) .$$

This computation can be done in $O(|T|^2 * |U|)$ time.

3 The mediator

So far we have seen that an information source over an underlying set of objects Obj consists of

(1) a taxonomy $(T, \preceq)$ and
(2) a stored interpretation I of T.

The terminology T contains terms that are familiar to users of the source, the subsumption relation $\preceq$ contains relationships between terms of T, and the stored interpretation I associates each term t with the objects that are indexed under t (by the indexer).

Consider now a set of sources $S_1, ..., S_k$ over the *same* underlying set of objects Obj. In general, two different sources may have different terminologies either because the users of the two sources are familiar with different sets of terms or because one source indexes objects at a different level of granularity than the other. The two sources may also have different subsumption relations as the relationships between any two given terms may be perceived differently in the two sources. Finally, two different sources may have different stored interpretations; for example, some objects may have been indexed by one source but not by the other.

Clearly if one wants to combine or *integrate* information coming from different sources, one has to cope with the above heterogeneities. One way of rendering all these heterogeneities transparent to users is through the use of *mediators* (initially proposed in [78]).

The problem of information integration has attracted considerable attention in the last few years, especially in the area of databases (see [31] for a comprehensive overview). The main

---

[3] Specifically, $U = \{o \in Obj \mid \exists\, t \in T \text{ s.t. } o \in I(t)\}$.

[4] Note that here we express the execution time with respect to two parameters: the size of the terminology and the number of the stored objects.

idea is to have users access information sources through a common schema that reflects their needs. Two main approaches seem to have emerged, namely, the virtual view approach and the materialized view approach. In the first, only the common schema is stored (but no data), while in the second (also called the warehouse approach) both the common schema and data over that schema are stored. Our approach is similar in spirit to the virtual view approach.

In our approach, a mediator M has a taxonomy $(T, \preceq)$ that reflects the needs of its potential users but has *no* stored objects. Instead, each term at the mediator is related directly or indirectly to terms in the underlying sources. More formally, a mediator is defined as follows:

Definition 9 A mediator M over k sources $S_1 = \langle (T_1, \preceq_1), I_1 \rangle$,...,$S_k = \langle (T_k, \preceq_k), I_k \rangle$ consists of:
(1) a taxonomy $(T, \preceq)$ and
(2) a set of *articulations* a_i, one for each source S_i; each articulation a_i is a subsumption relation over $T \cup T_i$.

Roughly speaking, a mediator is just like a source but with an important difference: there is no interpretation stored at the mediator. What is stored at the mediator, instead, is the set of articulations a_i, one for each source S_i. For example, suppose that we want to integrate two Web catalogs that provide access to pages about electronic products. In particular, consider the sources S_1 and S_2 shown in Fig. 7 and assume that we want to provide access to these sources through a mediator M as shown in that figure. To achieve integration, we enrich the mediator with articulations, i.e., with relationships that relate the terms of the mediator to the terms of the sources, as shown in Fig. 7. The articulations a_1 and a_2 shown in Fig. 7 are the following sets of subsumption relationships:

$$\begin{aligned} a_1 = \{ & \texttt{PhotoCameras} \preceq \texttt{Cameras}, \\ & \texttt{StillCameras} \preceq \texttt{PhotoCameras}, \\ & \texttt{Miniature} \preceq \texttt{StillCameras}, \\ & \texttt{Instant} \preceq \texttt{StillCameras},\ \texttt{Reflex}_1 \preceq \texttt{StillCameras}, \\ & \texttt{Reflex}_1 \preceq \texttt{Reflex},\ \texttt{Reflex} \preceq \texttt{Reflex}_1 \} \\ a_2 = \{ & \texttt{Products} \preceq \texttt{Electronics},\ \texttt{SLRCams} \preceq \texttt{Reflex}, \\ & \texttt{VideoCams} \preceq \texttt{MovingPictureCams}, \\ & \texttt{MovingPictureCams} \preceq \texttt{VideoCams} \} \end{aligned}$$

Note that a_1 is a subsumption relation over $T \cup T_1$ and a_2 is a subsumption relation over $T \cup T_2$, as required by the definition of an articulation (Def. 9).

Figure 8 shows another example of a mediator over three sources. These three sources provide access to tourist information, and the information is organized by location.

Now, in the presence of several sources, one and the same term may appear in two or more sources. If the same term appears in two different sources, then we consider the two appearances as two different terms. This is denoted here by subscripting each term of a source S_i by the subscript i and can be implemented in practice by, say, prefixing each term by the name of the source in which the term appears. Take, for example, the term DB and suppose that it appears in sources S_i and S_j. Then, from the mediator's point of view there are two distinct terms: the term $\texttt{DB}_i$ in source S_i and the term $\texttt{DB}_j$ in source S_j. This is reasonable as the same term can have different interpretations (meanings) in different sources. Thus for every $i \neq j$ we assume $T_i \cap T_j = \emptyset$, and for every i we assume $T \cap T_i = \emptyset$. In this way we overcome the problems of homonyms. Under these assumptions, two terms are considered equivalent, e.g., $DB_i \sim DB_j$, only if they can be shown to be equivalent through the articulations a_i and a_j, e.g., DB_i and DB_j are equivalent if there is a term t in T such that $t \sim_{a_i} DB_i$ and $t \sim_{a_j} DB_j$.

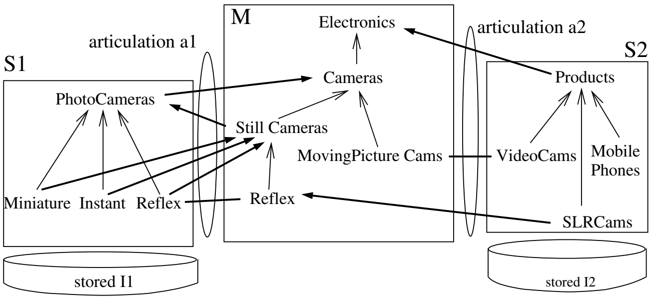


Fig. 7. A mediator over two catalogs of electronic products

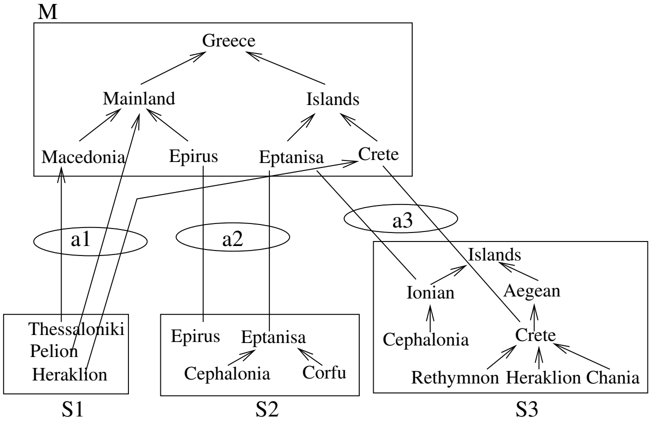


Fig. 8. A mediator over three catalogs of tourist information

Integrating objects from several sources often requires *restoring the context* of these objects, i.e., adding information that is missing from the original representation of the objects that concerns the context of the objects. Consider, for example, a mediator that provides access to electronic products according to the *type* of the products and according to the *location* of the stores that sell these products. Suppose that the mediator has two underlying sources S_1 and S_2, as shown in Fig. 9. Assume that S_1 is the source of a store located in Heraklion, while S_2 is the source of a store located in Paris. The context of the objects of each source, here the location of the store that sells each product, can be restored by adding appropriate relationships to the articulations. Specifically, for defining that all `PhotoCameras` of the source S_1 are available through a store located in `Heraklion`, it suffices to put in the articulation a_1 the relationship

$$\texttt{PhotoCameras}_1 \preceq \texttt{Heraklion},$$

while for defining that all products of the source S_2 are available through a store located in `Paris`, it suffices to put in the articulation a_2 the following relationship:

$$\top_2 \preceq \texttt{Paris}.$$

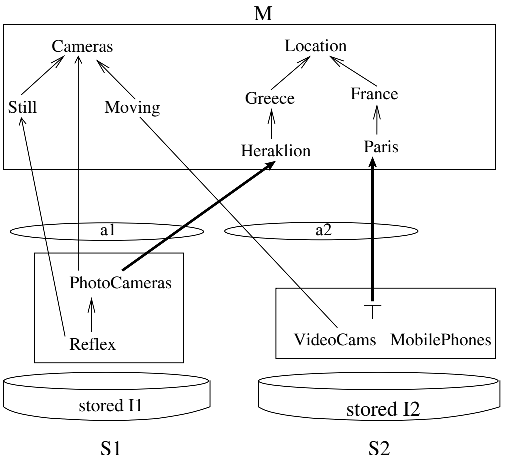


Fig. 9. Using articulations to restore the context of source objects

This example demonstrates how the articulations of the mediator can restore the context of the objects.

Turning now to query answering, we recall that the mediator receives queries over its own terminology T. Now, as the mediator has no stored interpretation of T, the only way to obtain one is by *querying* the underlying sources. However, as the mediator and the sources have different terminologies, to compute the interpretation of a term $t \in T$, the mediator sends to each source S_i a *translation* of t, i.e., a query that can be answered by the source, and then takes the *union* of the answers returned by the sources. The definition of translations is based on the articulations of the mediator.

Thus we will actually define an interpretation I of the mediator terminology based on the interpretations I_i stored at the sources, on the one hand, and on the articulations a_i, $i = 1, .., k$, on the other. Conceptually, once the interpretation I of the mediator is defined, the mediator can answer queries just like any other source does, i.e., from its sure model I^- and from its possible model I^+.

To define the mediator interpretation I we proceed as follows. For every term t of the mediator terminology T:

1. First, we define a translation t^i of t in a_i in the form of a query to source S_i, $i = 1, .., k$.
2. Then, we evaluate the query t^i at source S_i, $i = 1, .., k$.
3. Finally, we define $I(t)$ by taking the union of the answers to the queries t^i returned by the sources.

Now, there are two ways to translate t using the articulation a_i; we shall call these the *upper approximation* of t and the *lower approximation* of t in a_i. Roughly speaking, the upper approximation of t in a_i is the conjunction of all terms of T_i that subsume t in a_i, and the lower approximation of t in a_i is the disjunction of all terms of T_i that t subsumes in a_i. To define these notions formally, we need the notions of *tail* and *head* of a term *relative* to an articulation:

Definition 10 Given a term $t \in T$ and articulation a_i, we define

$$tail_i(t) = \{s \in T_i \mid s a_i t\} \text{ and } head_i(t) = \{u \in T_i \mid t a_i u\}.$$

Definition 11 Let $M = (T, \preceq, a_1, ..., a_k)$ be a mediator over sources $S_1, .., S_k$. If t is a term of T, then

- the *lower approximation* of t with respect to a_i, denoted by t^i_l, is defined by

$$t^i_l = \bigvee tail_i(t);$$

- the *upper approximation* of t with respect to a_i, denoted by t^i_u, is defined by

$$t^i_u = \begin{cases} \bigwedge head_i(t), & \text{if } head_i(t) \neq \emptyset \\ t^i_l, & \text{otherwise} \end{cases}$$

Note that if $head_i(t) = \emptyset$, then we consider that $t^i_u = t^i_l = \bigvee tail_i(t)$. The reason behind this choice is that we want the interpretation obtained by using lower approximation to be less than or equal to ($\sqsubseteq$) the interpretation obtained by using the upper approximation.

Here are some examples of approximations for the mediator shown in Fig. 7:

$\texttt{StillCameras}^1_l = \texttt{Miniature} \vee \texttt{Instant} \vee \texttt{Reflex}$
$\texttt{StillCameras}^1_u = \texttt{PhotoCameras}$
$\texttt{Reflex}^1_l = \texttt{Reflex}$
$\texttt{Reflex}^1_u = \texttt{Reflex} \wedge \texttt{PhotoCameras}$
$\texttt{Reflex}^2_l = \texttt{SLRCams}$
$\texttt{Cameras}^1_l = \texttt{PhotoCameras} \vee \texttt{Miniature} \vee \texttt{Instant} \vee \texttt{Reflex}$
$\texttt{Cameras}^1_u = \texttt{PhotoCameras} \vee \texttt{Miniature} \vee \texttt{Instant} \vee \texttt{Reflex}$
$\texttt{MovingPictureCams}^1_u = \texttt{MovingPictureCams}^1_l = \epsilon.$

Note that for a given term $t \in T$ the evaluation of t^i_u requires the previous evaluation of $head_i(t)$, and the evaluation of t^i_l requires the previous evaluation of $tail_i(t)$. However, if we compute the transitive closure of a_i, then the evaluation of $head_i(t)$ and $tail_i(t)$ is straightforward.

Now, the approximations t^i_u and t^i_l of t are actually queries to the source S_i, and as such each can have a sure answer and a possible answer (Sect. 2). As a consequence, we can define at least four different interpretations I for the mediator. Assuming for simplicity that *all* sources respond in the same manner, i.e., either all give a sure answer or all give a possible answer, we can define exactly four interpretations for the mediator that we shall denote by I_{l-}, I_{l+}, I_{u-}, I_{u+}. These interpretations are defined as follows:

1 Lower approximation of t at mediator and sure answer from sources:
$I_{l-}(t) = \bigcup_{i=1}^{k} I_i^-(t^i_l)$.
2 Lower approximation of t at mediator and possible answer from sources:
$I_{l+}(t) = \bigcup_{i=1}^{k} I_i^+(t^i_l)$.
3 Upper approximation of t at mediator and sure answer from sources:
$I_{u-}(t) = \bigcup_{i=1}^{k} I_i^-(t^i_u)$.
4 Upper approximation of t at mediator and possible answer from sources:
$I_{u+}(t) = \bigcup_{i=1}^{k} I_i^+(t^i_u)$.

So, the mediator can answer queries submitted by its users based on any of the four interpretations above. Moreover, for any of these four interpretations, the mediator can give either a sure answer or a possible answer – just like any source can (Sect. 2). By consequence, we can distinguish eight possible modes in which a mediator can operate. Each mode essentially

Table 1. Modes in which a mediator can operate

Oper. mode at med.	Term approx. at med.	Query eval. at source	Query eval. at med.	Answer model of med.
1	lower	sure	sure	I^-_{l-}
2	lower	possible	sure	I^-_{l+}
3	upper	sure	sure	I^-_{u-}
4	upper	possible	sure	I^-_{u+}
5	lower	sure	possible	I^+_{l-}
6	lower	possible	possible	I^+_{l+}
7	upper	sure	possible	I^+_{u-}
8	upper	possible	possible	I^+_{u+}

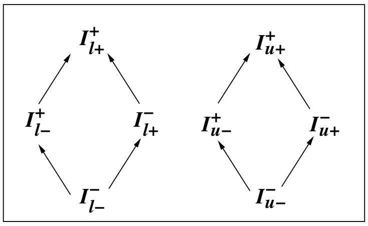


Fig. 10. Ordering ($\sqsubseteq$) of eight answer models of the mediator

corresponds to a different answer model of the mediator. The operation modes of a mediator and the corresponding answer models are summarized in Table 1.

Very roughly speaking, as we go down the table (from mode 1 to 8) the answer to the same user query is more likely to contain objects that are not "relevant" to the query. This is described more precisely in the following proposition.

Proposition 3 The answer models of the mediator are ordered as follows:

(a) $I^-_{l-} \sqsubseteq I^-_{l+}$
(b) $I^-_{u-} \sqsubseteq I^-_{u+}$
(c) $I^+_{l-} \sqsubseteq I^+_{l+}$
(d) $I^+_{u-} \sqsubseteq I^+_{u+}$
(e) $I^-_{l-} \sqsubseteq I^+_{l-}$
(f) $I^-_{l+} \sqsubseteq I^+_{l+}$
(g) $I^-_{u-} \sqsubseteq I^+_{u-}$
(h) $I^-_{u+} \sqsubseteq I^+_{u+}$. ◇

Figure 10 shows graphically the orderings of the above proposition. The nodes represent the answer models shown in Table 1. An arrow from node m to a node n means that $m \sqsubseteq n$.

For example, the interpretation of the term `RelationalDatabases` of the mediator shown in Fig. 11, in each of the models $I^-_{l-}, I^-_{l+}, I^-_{u-}, I^-_{u+}$, is as follows:

$$I^-_{l-}(\texttt{RelationalDatabases}) = \{1\}$$
$$I^-_{l+}(\texttt{RelationalDatabases}) = \{1, 2\}$$
$$I^-_{u-}(\texttt{RelationalDatabases}) = \{1, 2, 3\}$$
$$I^-_{u+}(\texttt{RelationalDatabases}) = \{1, 2, 3, 4\}.$$

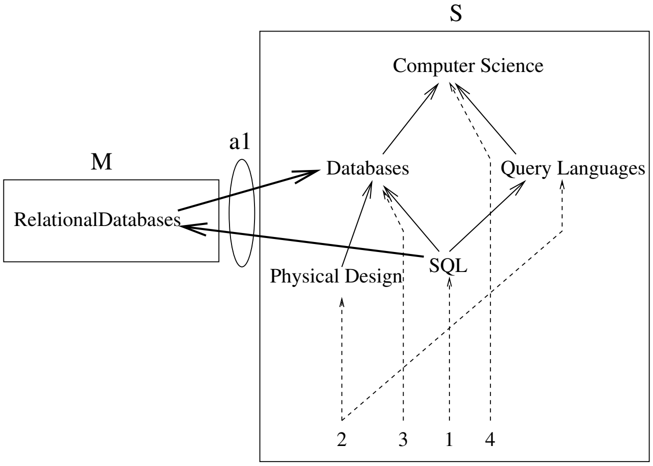
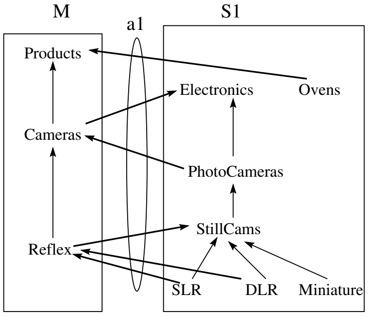


Fig. 11. A mediator over one source



a

T_1	I_1	I_1^-	I_1^+
$\bot$	$\emptyset$	$\emptyset$	$\emptyset$
$\top$	$\emptyset$	{0,1,2,3,4,5,6}	{0,1,2,3,4,5,6}
Ovens	{6}	{6}	{0,1,2,3,4,5,6}
Electronics	{5}	{0,1,2,3,4,5}	{0,1,2,3,4,5,6}
PhotoCameras	{4}	{0,1,2,3,4}	{0,1,2,3,4,5}
StillCams	{3}	{0,1,2,3}	{0,1,2,3,4}
SLR	{2}	{2}	{0,1,2,3}
DLR	{1}	{1}	{0,1,2,3}
Miniature	{0}	{0}	{0,1,2,3}

Q	1:I^-_{l-}	2,3: I^-_{l+}, I^-_{u-}	4 :I^-_{u+}
Products	{0,1,2,3,4,6}	{0,1,2,3,4,5,6}	{0,1,2,3,4,5,6}
Cameras	{0,1,2,3,4}	{0,1,2,3,4,5}	{0,1,2,3,4,5,6}
Reflex	{1,2}	{0,1,2,3}	{0,1,2,3,4}

b

Fig. 12. A mediator with one articulation to a source S_1

Another example of mediator operation is given in Fig. 12. Figure 12a shows a mediator having an articulation to a source S_1 and Fig. 12b shows two tables. The table at the upper part of the figure shows the interpretation I_1 of source S_1 and the corresponding (sure and possible) models. The first column of the table at the bottom shows three queries that are actually the three terms of T. The subsequent columns show what the mediator returns in each of the first four operation modes.

The operation modes of the mediator can either be decided (and fixed) by the mediator designer at design time or indicated by the mediator users at query time. We can distinguish at least three approaches:

- *Fixed approach.* The mediator designer selects and fixes one of the eight possible modes of operation for the mediator and the sources, and users simply submit their queries to the mediator without any further indication.
- *Variable approach.* The mediator users submit their queries along with a specification for the query evaluation mode they wish. This is done by providing values to the mediator for selecting one of the eight operation modes from Table 1. For example, the following user specification selects operation mode number 3 from Table 1:

  Term approximation at mediator = `upper`,
  Query evaluation at source = `sure`,
  Query evaluation at mediator = `sure`.
- *Mixed approach.* The mediator designer selects and fixes some of the attributes of Table 1, and the user provides the remaining ones. For example, the designer may select and fix the query evaluation mode at the source (i.e., sure or possible) and the kind of term approximation at the mediator (i.e., lower or upper approxiamtion) during design time, while the users select the query evaluation mode at the mediator during query time.

Clearly, the selection of one of the above approaches depends on several factors, such as the reliability of the sources, the level of expertise of the users, and so on. One can even think of more sophisticated modes of mediator operation than those presented in Table 1. For example, the mediator designer may assign a degree of reliability to each source and then ask sources to evaluate queries in a mode depending on their degree of reliability. In this paper, however, we do not pursue this idea any further.

We have seen so far how the mediator communicates with the sources through the articulations. In fact, the articulations are the *only* means of communication between the sources and the mediator. Now, certain kinds of articulation are better than others. One kind of articulation that is of particular interest are those that ensure what we call "compatibility" between the sources and the mediator.

Definition 12 A source S_i is *compatible* with the mediator M if for any terms s, t in T_i; if $s a_i t$, then $s \preceq_i t$.

That is, S_i is compatible with the mediator whenever the following condition holds: for all terms s and t in T_i, if s is subsumed by t in the articulation a_i, then s is also subsumed by t in $\preceq_i$.

For example, the source S_1 of Fig. 7 is compatible with the mediator since we have `Miniature` a_1 `PhotoCameras` and `Miniature` $\preceq_1$ `PhotoCameras`, `Instant` a_1 `PhotoCameras` and `Instant` $\preceq_1$ `PhotoCameras`, `Reflex` a_1 `PhotoCameras` and `Reflex` $\preceq_1$ `PhotoCameras`.

An interesting consequence of compatibility is that if a source S_i is compatible with the mediator, then in every model I_i of S_i the following condition holds: $I_i(t_l^i) \subseteq I_i(t_u^i)$ for each mediator term t, where t_l^i is the lower approximation of t and t_u^i is the upper approximation of t. From this property we infer that if all sources are compatible with the mediator, then the ordering relation over the eight answer models of the mediator (Fig. 10) is enriched as stated by the following proposition.

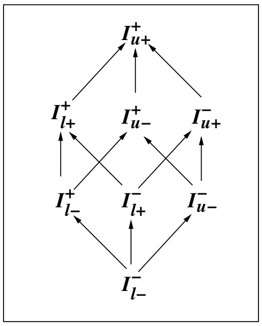


Fig. 13. Ordering ($\sqsubseteq$) of the eight answer models of the mediator in the case where all sources are compatible with the mediator

Proposition 4 If all sources are compatible with the mediator, then

(1) $I^-_{l-} \sqsubseteq I^-_{u-}$
(2) $I^-_{l+} \sqsubseteq I^-_{u+}$
(3) $I^+_{l-} \sqsubseteq I^+_{u-}$
(4) $I^+_{l+} \sqsubseteq I^+_{u+}$

As a result, the two diagrams of Fig. 10 are now connected in a single diagram, as shown in Fig. 13.

Note that the above ordering relationships do not necessarily hold if the sources are not compatible with the mediator. For example, consider a source S_1 with terminology $T_1 = \{b, b'\}$ and no subsumption relationships. Suppose that the source has a stored interpretation I_1 defined as follows: $I_1(b) = \{1\}$ and $I_1(b') = \{2\}$. Now consider a mediator connected to source S_1 through the articulation $a_1 = \{b \preceq t, t \preceq b'\}$, where t is a term of the mediator. Notice that S_1 is not compatible with the mediator because b is subsumed by b' in a_1 while b is not subsumed by b' in $\preceq_1$, i.e., $b a_1 b'$ and $b \not\preceq_1 b'$. Here we have $t_l^1 = b$ and $t_u^1 = b'$, thus $I_1^-(t_l^1) = \{1\}$ and $I_1^-(t_u^1) = \{2\}$. It follows that $I_1^-(t_l^1) \not\subseteq I_1^-(t_u^1)$, which implies $I^-_{l-}(t) \not\subseteq I^-_{u-}(t)$. From this example we see that if the underlying sources are not compatible with the mediator, then $I^-_{l-} \sqsubseteq I^-_{u-}$ does not hold.

Another interesting implication of compatibility concerns the efficiency of query evaluation. Let s, t be two terms in T_i that are known to the mediator (through a_i) and assume that the mediator knows that source S_i is compatible. In this case, if $s a_i t$, then $s \preceq_i t$. From this knowledge the mediator can conclude that $I_i(s) \subseteq I_i(t)$, in every model I_i of T_i, and thus $I_i(s) \cap I_i(t) = I_i(s)$ and $I_i(s) \cup I_i(t) = I_i(t)$. This means that the mediator can retain only the minimal elements of the set $head_i(t)$ and still obtain the same answer for the query t_u^i from source S_i. Therefore, if the mediator knows that source S_i is compatible, then instead of sending to source S_i the query $\bigwedge head_i(t)$, the mediator can send the query $\bigwedge min(\ head_i(t)\ )$. Similarly, in the set $tail_i(t)$, the mediator can retain only the maximal elements and still obtain the same answer for the query t_l^i from source S_i, i.e., instead of sending the query $\bigvee tail_i(t)$ to source S_i, the mediator can send the query $\bigvee max(\ tail_i(t)\ )$.

For example, in Fig. 7, as source S_1 is compatible with the mediator, the lower approximation of the term `Camera` is the term `PhotoCameras`. If S_1 were not compatible, then the lower approximation of `Camera` would be the disjunction `PhotoCameras` $\vee$ `Miniature` $\vee$ `Instant` $\vee$ `Reflex`.

Thus if S_i is compatible, then $t^i_u = \bigwedge min(head_i(t))$ and $t^i_l = \bigvee max(tail_i(t))$. In this case, the evaluation of t^i_u and t^i_l can be done more efficiently without having to compute the transitive closure of a_i. Specifically, for evaluating $max(tail_i(t))$ we traverse in a depth-first-search the relation a_i starting from the term t. If an element t' of T_i is reached, then this term is "collected" and the algorithm does not traverse any other element subsumed by t' (in a_i). All elements of T_i that were collected during the traversal are then returned. We can evaluate $min(head_i(t))$ analogously. We conclude that if a source is compatible, then the approximation of a term for that source can be done more efficiently, especially when the articulation to that source is big. Moreover, the resulting approximations are shorter, which implies that their transmission requires less time and the underlying source can evaluate these queries more efficiently.

Note that maintaining compatibility is not an easy task. Of course, the designer of the mediator can initially design articulations such that the underlying sources are compatible. However, an update at a source S_i or at the mediator (changing either T or a_i) may destroy compatibility. Therefore, the mediator should (periodically) check the compatibility of its sources, e.g., by submitting to them queries that allow one to check whether $t \preceq_i t'$.

4 Query evaluation at the mediator

We have seen how the two possible approximations at the mediator (lower or upper) and the two possible query evaluation modes at the sources (sure or possible) give rise to four possible interpretations at the mediator: I_{l-}, I_{l+}, I_{u-}, and I_{u+}. If these four interpretations were stored at the mediator, then the interaction between a user and the mediator would be straightforward, i.e.,

- The user submits a query to the mediator (as if it were a usual source),
- The mediator and/or the user specifies the answer model to be used
- The mediator uses the specified model to provide a sure or possible answer to the query (as is done in a usual source).

However, there is *no* interpretation actually stored at the mediator, so to answer queries the mediator has to call on the underlying sources, submit to them appropriate queries, and then merge the results to produce the final answer for the user. Therefore, the crucial tasks for the evaluation of user queries at the mediator can be summarized as follows:

- Translate the user's query into a set of queries to the underlying sources, i.e., determine *what* queries to send to *which* sources.
- Merge the results returned by the sources in order to produce the answer to the user's query.

Clearly, the complexity of these tasks depends on the nature of the user query, i.e.,

- The form of the query (single term, disjunction of terms, etc.),
- The answer model used by the mediator.

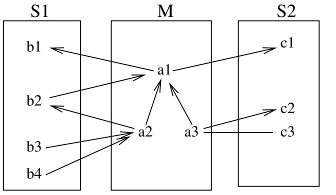


Fig. 14. A mediator over two sources

In the discussion that follows, we analyze the complexity of query evaluation at the mediator with respect to the form that a user query can have and the answer model used by the mediator for evaluating the query.

The complexity measure that we use in our analysis is the *number of queries* that the mediator sends to the sources to answer the user's query and the *execution time* expressed in terms of several parameters, such as size of the mediator terminology, size of the articulations, number of sources, length of the query, and size of the domain. However, we believe that the number of queries that the mediator needs to send is the most important measure, as the mediator spends a lot of time waiting for the answers from the sources.

We are aware that, in doing so, we do not take into account the complexity of query evaluation at each source. However, the mediator has little or no control over how queries are evaluated at individual sources. This is especially true for the applications that we have in mind (Web environment), where the mediator is set up by individual users who have no control over the underlying sources (which are Web catalogs).

In the complexity analysis that follows, we consider a mediator over k sources, $S_1, ..., S_k$. Note that we write I_l instead of I_{l-} or I_{l+}, and I_u instead of I_{u-} or I_{u+}, since the translation and the evaluation of queries at the mediator does not depend on the evaluation of queries at the underlying sources. First, we describe the evaluation of queries in the sure models of the mediator, i.e., in the models I^-_l and I^-_u.

It is interesting to note that the mediator *will not* necessarily query all sources. A source is queried only if the evaluation of the answer requires sending a subquery to that source; otherwise the source is not queried. Thus query translation also determines the selection of the sources.

We study separately the following forms of queries: *single-term queries*, *disjunctive queries*, *conjunctive queries*, *CNF queries*, and *DNF queries*.

- *Single-term queries*

Proposition 5 If the query is a single term, i.e., $q = t \in T$, then $I^-_l(t)$ and $I^-_u(t)$ can be evaluated as follows:

$$I^-_l(t) = \bigcup_{i=1..k} I_i(q^i_l(t)), \text{ where } q^i_l(t) = \bigvee\{s^i_l \mid s \preceq t\}$$

$$I^-_u(t) = \bigcup_{i=1..k} I_i(q^i_u(t)), \text{ where } q^i_u(t) = \bigvee\{s^i_u \mid s \preceq t\}.$$

$\diamond$

This means that the mediator M can evaluate the query by sending at most one query to each source. Thus M will send at most k queries. Note that if $q^i_l(t) = \epsilon$ (or $q^i_u(t) = \epsilon$), then M does not have to send any query to source S_i.

For example, consider a mediator over two sources, as shown in Fig. 14. The answer in I^-_l of the query $q = a_1$

can be evaluated as follows:

$$I_l^-(\mathtt{a}_1) = I_1(\, q_l^1(\mathtt{a}_1)\,) \;\cup\; I_2(\, q_l^2(\mathtt{a}_1)\,), \text{ where}$$
$$q_l^1(\mathtt{a}_1) = \mathtt{b}_2 \vee (\mathtt{b}_3 \vee \mathtt{b}_4)$$
$$q_l^2(\mathtt{a}_1) = \mathtt{c}_3,$$

while the answer I_u^- can be evaluated as follows:

$$I_u^-(\mathtt{a}_1) = I_1(\, q_u^1(\mathtt{a}_1)\,) \;\cup\; I_2(\, q_u^2(\mathtt{a}_1)\,), \text{ where}$$
$$q_u^1(\mathtt{a}_1) = \mathtt{b}_1 \vee \mathtt{b}_2$$
$$q_u^2(\mathtt{a}_1) = \mathtt{c}_1 \vee (\mathtt{c}_2 \wedge \mathtt{c}_3).$$

If the mediator knows that a source S_i is *compatible*, then the mediator can set

$$q_l^i(t) = \bigvee \max\left(\bigcup\{tail_i(s) \mid s \preceq t\}\right).$$

Note that if the entire articulation a_i is stored (including the transitive links), then the computation of t_l^i can be done in $O(|a_i|)$ time. The same holds for t_u^i. Thus the computation of $q_l^i(t)$ can be done in $O(|T| * |a_i|)$ time. The same holds for $q_u^i(t)$. This means that the computation of all $q_l^i(t)$, or $q_u^i(t)$, for $i = 1..k$ can be done in $O(|T| * |a|)$, where a denotes the union of all articulations, i.e., $a = a_1 \cup ... \cup a_k$.
Now, the set operations over the answers returned by the sources that are needed for computing $I_l^-(t)$ can be performed in $O(k * U)$ time.
Thus the total computation needed by the mediator can be done in $O(|T| * |a| + k * U)$ time.

- *Disjunctive queries*
  If the query is a disjunction of terms, i.e., $q = t_1 \vee ... \vee t_n$, then
  $$I_l^-(t_1 \vee ... \vee t_n) = \bigcup_{i=1..k} I_i(q_l^i(t_1) \vee ... \vee q_l^i(t_n))$$
  $$I_u^-(t_1 \vee ... \vee t_n) = \bigcup_{i=1..k} I_i(q_u^i(t_1) \vee ... \vee q_u^i(t_n)).$$
  Again, the mediator can evaluate the query by sending at most one query to each source.
  If, furthermore, a source S_i is *compatible*, then the mediator can send to S_i the query
  $$\bigvee \max\left(\bigcup_{j=1..n} (\cup\{tail_i(s) \mid s \preceq t_j\})\right).$$
  Clearly, the computation of each $q_l^i(t_1) \vee ... \vee q_l^i(t_n)$ can be done in $O(|T| * |a_i| * n)$ time. Thus the computation of all $q_l^i(t_1) \vee ... \vee q_l^i(t_n)$ for $i = 1..k$ can be done in $O(|T| * |a| * n)$ time.
  The set operations for computing $I_l^-(t)$ can be performed in $O(k * U)$ time.
  Thus the total computation needed by the mediator can be done in $O(|T| * |a| * n + k * U)$ time.
- *Conjunctive queries*
  If the query is a conjunction of terms, i.e., $q = t_1 \wedge ... \wedge t_n$, then
  $$I_l^-(t_1 \wedge ... \wedge t_n) = \bigcap_{j=1..n}\left(\bigcup_{i=1..k} I_i(q_l^i(t_j))\right)$$
  $$I_u^-(t_1 \wedge ... \wedge t_n) = \bigcap_{j=1..n}\left(\bigcup_{i=1..k} I_i(q_u^i(t_j)).\right)$$
  Thus for evaluating the query, the mediator has to send at most one query to each source for each term that appears in the conjunction. This means that the mediator will send at most $k * n$ queries.
  The computation of all $q_l^i(t_j)$ for $j = 1..n$ can be done in $O(|T| * |a| * n)$ time.
  The set operations for computing $I_l^-(t)$ can be performed in $O(k * n * U)$ time.
  Thus the total computation needed by the mediator can be done in $O(|T| * |a| * n + k * U * n)$ time.
- *Conjunctive normal form queries (CNF Queries)*
  A CNF query is a conjunction of maxterms where each maxterm is either a single term or a disjunction of distinct terms ([29]), i.e., $q = d_1 \wedge ... \wedge d_m$ where $d_j = t_{j1} \vee ... \vee t_{jn_j},\ j = 1..m, n_j \le |T|$. In this case,
  $$I_l^-(q) = \bigcap_{j=1..m} (\bigcup_{i=1..k} I_i(\, q_l^i(t_{j1}) \vee ... \vee q_l^i(t_{jn_j})\,)\,)$$
  $$I_u^-(q) = \bigcap_{j=1..m} (\bigcup_{i=1..k} I_i(\, q_u^i(t_{j1}) \vee ... \vee q_u^i(t_{jn_j})\,)\,).$$
  The mediator first evaluates each maxterm (disjunction) by sending at most one query to each source and then takes the intersection of the returned results. This means that the mediator will send at most $k * m$ queries, where m is the number of maxterms.
  Let l be the length of the query, that is, the number of term appearances in the query, i.e., $l = \sum_{j=1..m} n_j$. The computation of $q_l^i(t), i = 1..k$ for all t that appear in q can be done in $O(|T| * |a| * l)$ time.
  The set operations for computing $I_l^-(t)$ can be performed in $O(k * m * U)$ time.
  Thus the total computation needed by the mediator can be done in $O(|T| * |a| * l + k * m * U)$ time.
- *Disjunctive normal form queries (DNF Queries)*
  A DNF query is a disjunction of minterms where a minterm is either a single term or a conjunction of distinct terms, i.e., $q = c_1 \vee ... \vee c_m$, where $c_j = t_{j1} \wedge ... \wedge t_{jn_j},\ j = 1..m, n_j \le |T|$. In this case,
  $$I_l^-(q) = \bigcup_{j=1..m}\left(\bigcap_{h=1..n_j} (\bigcup_{i=1..k} I_i(q_l^i(t_{jh})))\right)$$
  $$I_u^-(q) = \bigcup_{j=1..m}\left(\bigcap_{h=1..n_j} (\bigcup_{i=1..k} I_i(q_u^i(t_{jh}))).\right)$$
  Thus M will send at most $k * l$ queries, where l is the length of the query.
  The computation of all $q_l^i(t)$ for $i = 1..k$ for all t that appear in q can be done in $O(|T| * |a| * l)$ time.
  The set operations for computing $I_l^-(t)$ can be performed in $O(k * l * U)$ time.
  Thus the total computation needed by the mediator can be done in $O(|T| * |a| * l + k * l * U)$ time.

Table 2 summarizes the *number of calls* complexity and Table 3 the time complexity. Note that any query that contains the logical connectives $\wedge$ and $\vee$ can be converted to DNF or CNF by using one of the existing algorithms (e.g., see [29]). In our case, CNF is preferred to DNF since the evaluation of a query in CNF requires sending a smaller number of queries to

Table 2. *Number of calls* complexity of query evaluation at mediator (for sure model) assuming k sources

	Query form	Max. no. of calls
Single term	t	k
Disjunction	$t_1 \vee ... \vee t_n$	k
Conjunction	$t_1 \wedge ... \wedge t_n$	$k * n$
CNF	$d_1 \wedge ... \wedge d_m$ where $d_j = t_{j1} \vee ... \vee t_{jn_j}$	$k * m$
DNF	$c_1 \vee ... \vee c_m$ where $c_j = t_{j1} \wedge ... \wedge t_{jn_j}$	$k * \sum_{j=1..m} n_j$

Table 3. Time complexity of query evaluation at mediator (for sure model)

Query form	Time complexity (wrt $\lvert T\rvert, \lvert a\rvert, k, U$)
t	$O(\lvert T\rvert * \lvert a\rvert + k * U)$
s $t_1 \vee ... \vee t_n$	$O(\lvert T\rvert * \lvert a\rvert * n + k * U)$
$t_1 \wedge ... \wedge t_n$	$O(\lvert T\rvert * \lvert a\rvert * n + k * U * n)$
$d_1 \wedge ... \wedge d_m$ where $d_j = t_{j1} \vee ... \vee t_{jn_j}$	$O(\lvert T\rvert * \lvert a\rvert * l + k * m * U)$
$c_1 \vee ... \vee c_m$ where $c_j = t_{j1} \wedge ... \wedge t_{jn_j}$	$O(\lvert T\rvert * \lvert a\rvert * l + k * l * U)$

the sources. For this reason, the mediator first converts the user query in CNF and then evaluates the CNF query by sending queries to the sources.

We conclude this section by describing the evaluation of queries in the possible models of the mediator, i.e., in the models I_l^+ and I_u^+. The evaluation of a single-term query in I_l^+ or I_u^+ is done by evaluating a conjunction of terms in I_l^- or I_u^-, respectively:

$$I^+(t) = \bigcap\{I^-(u) \mid u \in head(t) \text{ and } u \not\sim t\}$$
$$= I^-(\bigwedge\{u \mid u \in head(t) \text{ and } u \not\sim t\}),$$

where $I^+(t)$ stands for $I_l^+(t)$ or $I_u^+(t)$, and I^- stands for I_l^- or I_u^-, respectively. Therefore, the complexity analysis of evaluating $I^+(t)$ can be done using Tables 2 and 3. Finally, the evaluation of a disjunction in I^+ is done by evaluating a DNF query in I^-, and the evaluation of a conjunction in I^+ is done by evaluating a conjunction in I^-.

5 Enhancing the quality of answers with object descriptions

We have just seen how to compute several kinds of answers at the mediator. We shall now see how to improve the "quality" of the answers by providing additional information on the objects returned.

First, let us see an example in the context of a single source. Consider a source S that contains an object 1 indexed under two terms, `Cameras` and `Underwater`, and an object 2 also indexed under two terms, `Cameras` and `Miniature`. Next, assume that S receives the query q = `Cameras` and is asked to return both the sure and the possible answer to that query. Clearly, in both cases S will return the set $\{1, 2\}$. However, instead of just returning the set $\{1, 2\}$, the source could return the following set:

$$\{(1, \{\texttt{Cameras}, \texttt{Underwater}\}),$$
$$(2, \{\texttt{Cameras}, \texttt{Miniature}\})\}$$

In this set, each object is accompanied by the set of *all* terms under which the object is indexed. This information could provide valuable help to the user. Indeed, the user of our example may have actually been looking for miniature cameras, but he only used the term `Cameras` in his query for one of several reasons. For example,

- The user may have forgotten to use the term `Miniature`;
- Or the user did not know that the term `Miniature` was included in the terminology of the source;
- Or the user did not know that the objects of the source were indexed in such specificity.

We believe that including in the answer all terms under which each object returned is indexed might aid the user in selecting the objects that are most relevant to his information need. In addition, such terms could aid the user in getting better acquainted with the taxonomy of the source. Indeed, more often than not users are not familiar with the source taxonomy and know little about its specificity and coverage (see [50]). As a result, user queries are often imprecise and do not reflect real user needs. We believe that familiarity with the source taxonomy is essential for a precise formulation of user queries. Therefore, we extend the notion of answer to be a set of objects each accompanied by its *index*, i.e., by the set of all terms under which the object is indexed.

Definition 13 The *index* of an object o with respect to an interpretation I, denoted by $D_I(o)$, is the set of all terms that contain o in their interpretation, i.e., $D_I(o) = \{t \in T \mid o \in I(t)\}$.

For brevity, henceforth we shall sometimes write $D(o)$ instead of $D_I(o)$, $D^-(o)$ instead of $D_{I^-}(o)$, and $D^+(o)$ instead of $D_{I^+}(o)$ when the interpretation I is clear from the context. Clearly, the index of an object depends on the interpretation I, so the same object can have different indexes under different interpretations. Here are some examples of indexes in the source shown in Fig. 6:

$$\begin{aligned}
D(1) &= \{\texttt{StillCams}\}\\
D^-(1) &= \{\texttt{StillCams}, \texttt{Cameras}\}\\
D^+(1) &= \{\texttt{StillCams}, \texttt{Cameras}, \texttt{Reflex}, \texttt{Miniature},\\
&\quad\ \texttt{MovingPictureCams}, \texttt{UnderwaterDevices}\}\\
D(2) &= \{\texttt{Cameras}\}\\
D^-(2) &= \{\texttt{Cameras}\}\\
D^+(2) &= \{\texttt{Cameras}, \texttt{StillCams}, \texttt{MovingPictureCams},\\
&\quad\ \texttt{UnderwaterDevices}\}.
\end{aligned}$$

We have seen that the user of a source can submit a query and ask for a sure or a possible answer. Following our discussion on indexes, the user can now also ask for the sure or possible index for each object in the answer.

Table 4. Answers to a query

	Object set	Object index	Answer returned
1	Sure	Sure	$\{ (o, D^-(o)) \mid o \in I^-(q)\}$
2	Sure	Possible	$\{ (o, D^+(o)) \mid o \in I^-(q)\}$
3	Possible	Sure	$\{ (o, D^-(o)) \mid o \in I^+(q)\}$
4	Possible	Possible	$\{ (o, D^+(o)) \mid o \in I^+(q)\}$

This means that the answer returned by the source to a given query q can have one of the forms shown in Table 4. It is up to the user to specify the desired form of the answer.

Note that if I^- is stored at the source, then the evaluation of D^- for an object o is straightforward. If, however, only the interpretation I is stored at the source, then we can compute $D^-(o)$ as follows:

Proposition 6 $D^-(o) = \bigcup\{head(t) \mid o \in I(t)\}$, or equivalently, $D^-(o) = \bigcup\{head(t) \mid t \in D(o)\}$.

If we have computed $D^-(o)$, then we can compute $D^+(o)$ as follows:

Proposition 7 $D^+(o) = \{t \mid head(t) \setminus \{t' | t' \sim t\} \subseteq D^-(o)\}$.

By analogy to the single-source case, a mediator can return answers consisting of objects that are accompanied by their indexes. In other words, a mediator can return a set of pairs $(o, D_I(o))$, where I is the model used by the mediator for answering queries. For example, consider two sources, S_1 and S_2, providing information about animals (e.g., photos), as shown in Fig. 15. The terms of source S_1 are in English, while the terms of source S_2 are in French. Moreover, a mediator M integrates the information of the two sources and provides a unified access through a taxonomy with English terms. Assume now that the mediator receives the query $q = \texttt{Animal}$, in which case the mediator sends the query $q_1 = \texttt{Animal} \vee \texttt{Dog}$ to source S_1 and the query $q_2 = \texttt{Mammifère} \vee \texttt{Chat}$ to source S_2. Moreover, assume that the sources S_1 and S_2 return objects accompanied by their sure indexes. Then, the source S_1 will return the answer

$\{ (1, \{\texttt{Dog}, \texttt{Animal}\}),\ (2, \{\texttt{Canis}, \texttt{Dog}, \texttt{Animal}\})\ \}$

and the source S_2 the answer

$\{ (1, \{\texttt{Mammifère}\}),\ (3, \{\texttt{Chat}, \texttt{Mammifère}\})\ \}$.

Next, assume that the mediator operates under operation mode 1 (Table 1), that is, the mediator uses the model I^-_{l-} for answering queries. Moreover, assume that the mediator returns objects accompanied by their sure indexes (in I^-_{l-}). In this case, the mediator will return the following answer:
$\{\ (1, \{\texttt{Dog}, \texttt{Mammal}, \texttt{Animal}\}),\ (2, \{\texttt{Dog}, \texttt{Mammal}, \texttt{Animal}\}),\ (3, \{\texttt{Mammal}, \texttt{Animal}).\ \}$

Let I denote any of the four interpretations I_{l-}, I_{l+}, I_{u-} and I_{u+} of the mediator, and assume that we want to compute $D^-(o)$, i.e. the sure index of some object o, at the mediator. Since the interpretation I is not stored at the mediator, we cannot compute $D^-(o)$ like we do for a source (Proposition 6 above). Instead, we must exploit the articulations a_i and the indexes $D_i(o)$ returned by the sources. Specifically, the mediator can compute $D_l(o)$ (i.e., the index of o with respect to I_l) and $D_u(o)$ (i.e., the index of o with respect to I_u) as stated by the following proposition. Note that again we write I_l instead of I_{l-} or I_{l+}, and I_u instead of I_{u-} or I_{u+}, since the computation of the object indexes at the mediator does not depend on the evaluation of queries at the underlying sources.

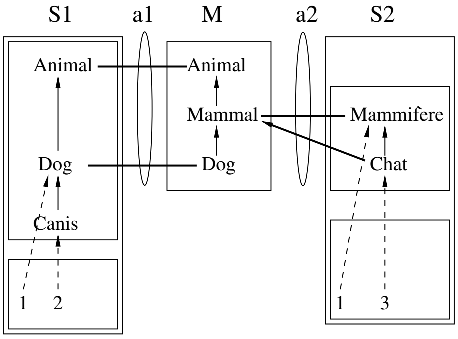


Fig. 15. A mediator over two sources

Proposition 8

$$\begin{aligned}
D_l(o) &= \bigcup_{i=1..k} D^i_l(o), \quad \text{where} \\
D^i_l(o) &= \{t \in T \mid t_i \in D_i(o) \text{ and } t_i\ a_i\ t\} \\
D_u(o) &= \bigcup_{i=1..k} D^i_u(o), \quad \text{where} \\
D^i_u(o) &= \{t \in T \mid (head_i(t) \neq \emptyset \text{ and } head_i(t) \subseteq D_i(o)) \text{ or} \\
&\qquad (head_i(t) = \emptyset \text{ and } t_i \in D_i(o) \text{ and } t_i\ a_i\ t).\}
\end{aligned}$$

Now, $D^-_l(o)$ and $D^-_u(o)$ can be computed by applying Proposition 6 to $D_l(o)$ and $D_u(o)$, respectively. Similarly, $D^+_l(o)$ and $D^+_u(o)$ can be computed by applying Proposition 7 to $D^-_l(o)$ and $D^-_u(o)$, respectively.

6 Extending our model

In this section we discuss various extensions of our model. Specifically, in Sect. 6.1 we extend the form of our articulations, in Sect. 6.2 we discuss mediators that also have a stored interpretation of their terminology, in Sect. 6.3 we describe how our mediators can be combined with information retrieval systems, and in Sect. 6.4 we discuss how our approach can lead to a network of articulated sources.

6.1 Extending the form of articulations

According to Sect. 3, an articulation a_i consists of subsumption relationships between terms only. However, we can extend the definition of an articulation to include subsumption relationships between terms and queries as well. This extension is useful because now the designer of the mediator can define articulations containing more complex relationships, as in the following examples:

- $\texttt{Electronics}_M \succeq (\texttt{TV}_i \vee \texttt{Mobiles}_i \vee \texttt{Radios}_i)$,
- $\texttt{DBArticles}_M \sim (\texttt{Databases}_i \wedge \texttt{Articles}_i)$.

In the first example, the users of the mediator can use the term `Electronics` instead of a long disjunction of terms at source S_i (benefit: brevity), while in the second they can use the term `DBArticles` instead of the conjunction of two terms at source S_i (note, however, that this is useful only if S_i supports multiple classification).

Definition 14 Let $(T, \preceq)$ be the taxonomy of a mediator and $(T_i, \preceq_i)$ the taxonomy of source S_i. An *articulation* a_i is a subsumption relation over $T \cup Q_{T_i}$, where Q_{T_i} is the set of all queries over T_i.

Let us now discuss the consequences of this extension with regard to the functionality of the mediators. For each term $t \in T$ the *tail* and *head* of t with respect to a_i can be defined as follows:

Definition 15 Given a term $t \in T$ and articulation a_i, we define

$$tail_i(t) = \{s \in Q_{T_i} |\ s a_i t\} \text{ and } head_i(t) = \{u \in Q_{T_i} |\ t a_i u\}.$$

Note that now the tail and head of a term are not sets of terms of T_i, but sets of *queries* over T_i.

The *lower* and *upper* approximations of t with respect to a_i are defined as in Sect. 3. The four interpretations and the eight answer models of the mediator are defined in the same way, too.

In this framework, the concept of compatibility is now redefined as follows:

Definition 16 A source S_i is *compatible* with the mediator M if for any queries s, t in Q_{T_i}, if $s a_i t$, then $s \preceq_i t$.

As mentioned earlier, maintaining compatibility is not an easy task. The mediator should (periodically) check the compatibility of its sources, e.g., by submitting to them queries that allow one to check whether $t \preceq_i t'$. However, now t and t' are queries; thus the sources should support subsumption checking over queries.

In general, an articulation may contain relationships between terms and arbitrary queries. For example, consider a source S_i implemented in the relational model (as described in Sect. 2) and suppose that this source can answer only pure SQL queries. In this case, the articulation a_i may contain relationships of the form `Cameras` $\preceq_{a_i} q_i$, where

$$q_i = \Pi_{object}(\sigma_{term-name="Cameras"} (INTERPRETATION \bowtie TERMINOLOGY)).$$

6.2 Mediators with stored interpretations

We can easily extend a mediator so as to *also* store an interpretation of its terminology T. Figure 16 shows graphically the architecture of a mediator of this kind. Such an extension can prove quite useful in the context of the Web: a Web user can define his own mediator consisting of a taxonomy that is familiar to him, a set of articulations to other Web catalogs, *and* a stored interpretation of the mediator's taxonomy. Note that the taxonomy of the mediator and its stored interpretation resemble the bookmark facility of Web browsers. However, the addition of articulations now allows the user to browse and query remote catalogs.

Let I_M denote the stored interpretation of T. When a user sends a query to the mediator, he has three choices:

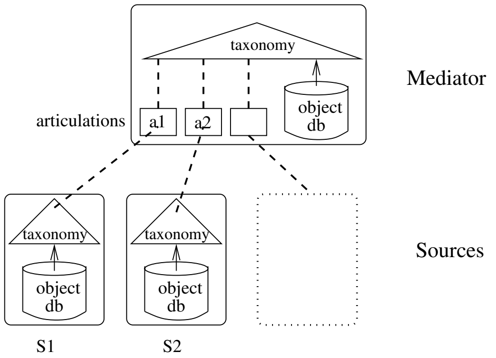


Fig. 16. Architecture of a mediator with a stored interpretation

(a) He can ask for an answer derived from I_M,
(b) He can ask for an answer derived from the interpretations of the remote sources, or
(c) He can ask for an answer derived from both I_M and the interpretations of the remote sources.

In the first case, the mediator operates as a source (Sect. 2), in the second case it operates like the mediators described earlier, while in the third case it again operates like the mediators described earlier but with one difference: the interpretations I_{l-}, I_{l+}, I_{u-}, I_{u+} are now defined by taking the union of I_M and the interpretations of the sources. For instance, the interpretation I_{l-} is now defined as

$$I_{l-}(t) = I_M(t) \cup \left(\bigcup_{i=1}^{k} I_i^-(t_l^i) \right).$$

In other words, in the third case, the mediator operates as usual, except that now, in addition to the k external sources $S_1, ..., S_k$, we have the mediator's own source $S_M = \langle (T, \preceq), I_M \rangle$ acting as a $(k+1)$-th source.

In the case where the mediator also stores an interpretation I_M of T, the mediator's ability to "translate" the descriptions of the objects returned by the underlying sources leads to an interesting scenario for the Web. Consider a user who has submitted a query to the mediator and assume that the mediator has returned a set of objects to the user. If some of these objects are of real interest to the user (e.g., a set of beautiful images, good papers, etc.), he user can store these objects in the database of the mediator. These objects will be stored under terms of the mediator's taxonomy, i.e., in the interpretation I_M of T.

For example, consider the mediator shown in Fig. 15. The mediator can store objects 1 and 2 under the terms `Dog`, `Mammal`, and `Animal` and object 3 under the terms `Mammal` and `Animal`.

However, one can easily see that it suffices to store objects 1 and 2 under the term `Dog` and object 3 under the term `Mammal`. More formally, to store an object o in I_M, the mediator associates this object with the following terms of T:

$$min_{\preceq_M} D_l(o) \quad \text{or} \quad min_{\preceq_M} D_u(o).$$

6.3 Mediators over hybrid sources

Let us use the term *free retrieval source* to refer to a source that indexes objects of interest using an *uncontrolled* vocabulary.

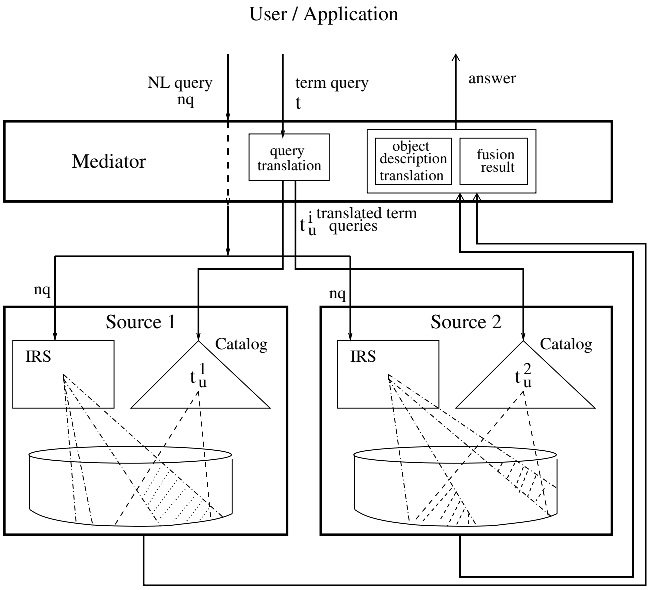


Fig. 17. Building mediators over hybrid sources

In this case, the objects of the domain have textual content and the vocabulary used for indexing them consists of those words that appear in the objects. These sources usually accept natural language queries and return a set of objects ordered according to their relevance to the query. Text retrieval systems (the typical case of "information retrieval systems"), as well as the search engines of the Web, fall into this category.

We can now use the term *hybrid source* to refer to a source that is both taxonomy-based and a free retrieval source. A hybrid source accepts *two* kinds of queries: queries over a controlled vocabulary and natural language queries. A source whose functionality is moving in this direction is Google. Using Google, one can first select a category, e.g., `Sciences/CS/DataStructures`, from the taxonomy of Open Directory and then submit a natural language query, e.g., "`Tree`". The search engine will compute the degree of relevance with respect to the natural language query, "`Tree`", only of those pages that fall in the category `Sciences/CS/DataStructures` in the catalog of Open Directory. Clearly, this enhances the precision of the retrieval and is computationally more economical.

Our approach can be used for building mediators over hybrid sources whose functionality extends the functionality offered by the existing metasearchers of the Web (e.g., MetaCrawler [63], SavvySearch [38], Profusion [27]). The user of a hybrid mediator can use the taxonomy of the mediator to browse or query those parts of the sources that are of interest to him. Moreover, he is able to query the databases of these sources using natural language queries. This implies that the mediator will send two kinds of queries to the sources: (a) queries that are evaluated based on indexing of the objects with respect to the taxonomy of the source and (b) queries that are evaluated based on the contents of the objects (pages). Figure 17 describes this architecture graphically.

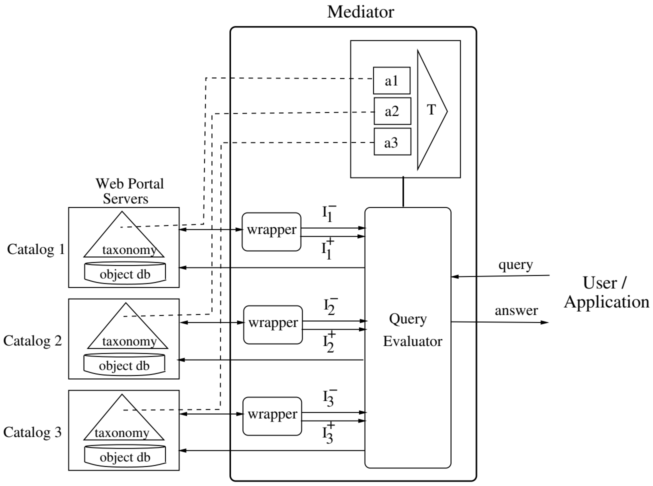


Fig. 18. An architecture for implementing our mediators over the catalogs of the Web

The functionality of the mediators described in this paper presumes that each source can provide a sure answer and a possible answer. However, the taxonomy-based sources that can be found on the Web, e.g., Yahoo! or ODP, do not currently provide such answers. This means that the functionality of our mediators cannot be implemented straightforwardly. Nevertheless, we can implement the functionality of our mediators over such sources by employing appropriate *wrappers*.

First note that the taxonomy and the interpretation of a Web catalog are published as a set of Web pages. For each term t of the taxonomy there is a separate Web page. This page contains the name of the term and links pointing to pages that correspond to the terms subsumed by t. In addition, the page contains links pointing to the objects, here Web pages, that have been indexed under the term t. However, we can employ a wrapper to parse each such page and extract the name of the term, the subsumed terms, and the indexed objects.

Now, the architecture for implementing our mediators over Web catalogs is shown in Fig. 18. The key point is that the interpretation of a term t of a source S_i in the sure model I_i^- and in the possible model I_i^+ can be computed on the mediator side. This can be achieved by building an appropriate wrapper for that source. In particular, to compute $I_i^+(t)$, the wrapper will fetch the pages of all terms t' such that $t \preceq_i t'$, and then it will derive $I_i^+(t)$ by computing the intersection $\cap\{I_i^-(t') | t \preceq t'\}$. According to this architecture, our mediators can be implemented by using the standard HTTP protocol. A prototype version of our mediators over the Web has already been implemented at Université de Paris-Sud.

6.4 Networks of articulated sources

One can easily see how our approach can be used for creating a complex information network, comprising sources and mediators, in a natural and straightforward manner. Indeed,

- To add a mediator to such a network one has to (a) design the mediator taxonomy $(T, \preceq)$ based on the domain of interest, (b) select the sources to be mediated, and (c) design the articulations a_i based on the known/observed relations between terms of the mediator and terms of the selected sources;
- To remove a mediator from the network, one just has to disconnect the mediator from the network;

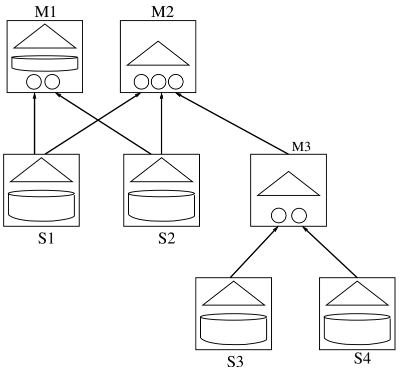


Fig. 19. A network consisting of primary and secondary taxonomy-based sources

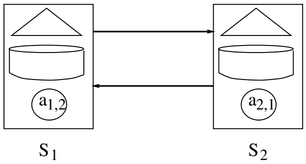


Fig. 20. Mutually articulated sources

- Moreover, to add a source to the information network, all one has to do is (a) select one or more mediators in the network and (b) design an articulation between the source and each mediator;
- Finally, to remove a source from the network, one simply has to remove the corresponding articulation from the mediator(s) to which the source is connected and disconnect the source. Note that as each mediator has one articulation for each underlying source, the deletion of an articulation does not affect the rest of the articulations.

A significant consequence of this approach is that network evolution can be *incremental*. Indeed, new relationships between terms of the mediator and terms of the sources can be added with minimum effort as soon as they are observed, and relationships that are seen to be no more valid can be removed just as easily: simply add/remove the relationships at the appropriate articulation in the mediator database storing the articulation.

For example, Fig. 19 shows a network consisting of four primary sources S_1, S_2, S_3, S_4 and three mediators M_1, M_2, and M_3. A line segment connecting a mediator M to a source or to a mediator S means that M is a mediator over S, and circles denote articulations. For example, M_2 is a mediator over the sources S_1 and S_2 and the mediator M_3. Note that the mediator $M1$ can also be considered as a primary source because it has a stored interpretation.

Also note that our approach allows *mutually articulated* sources, as shown in Fig. 20. In this case, we can no longer divide sources into primary and secondary.

Query evaluation and updating in a network of articulated sources raises several interesting questions. For example, a query to a source may trigger an infinite number of calls between the sources if the network is cyclic (e.g., Fig. 20). This and other related problems go beyond the scope of this paper and are treated in [69,72].

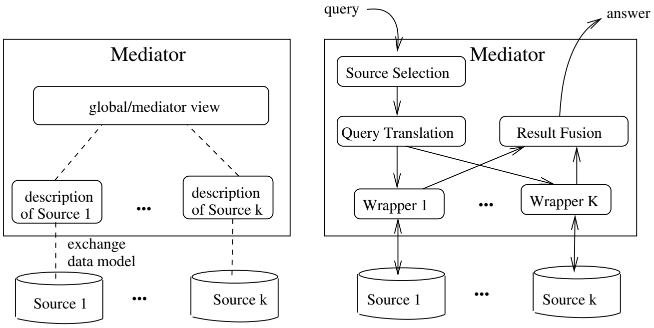


Fig. 21. Architecture and functional overview of the mediator

7 Related work

The need for integrated and unified access to multiple information sources has stimulated research on *mediators*. The concept of mediator was initially proposed by Wiederhold [78]. Since then it has been applied and specialized for several kinds of source and application needs. Nevertheless, in every instance we can identify a number of basic architectural components. Specifically, in most cases the mediator architecture consists of a mediator's view (usually in the form of a conceptual model), *source descriptions*, and *wrappers* that describe the contents and/or the querying capabilities of each source with respect to the mediator's view, and an *exchange data model*, which is used to convey information between the mediator and the sources. The mediator accepts queries expressed in the mediator view. Upon receiving a user query, the mediator selects the sources to be queried and formulates the query to be sent to each of them. These tasks are accomplished based on the source descriptions that encode what the mediator "knows" about the underlying sources. Finally, the mediator appropriately combines the returned results and delivers the final answer to the user. Figure 21 shows the general architecture and the functional overview of the mediator.

In this section we compare our approach with other existing mediator approaches. Our objective is to identify the basic differences and analogies and identify issues that are worthy of further research, rather than presenting a complete survey of this very broad area. Figure 22 shows a rough taxonomy of the mediator approaches according to two criteria: (a) the kind of underlying sources and (b) the kind of mediator view. According to the first, we can divide sources into two broad categories: *information retrieval systems* and *information bases*. The former provide content-based access to a set of (text) documents, while the latter store structured data. We will use this dichotomy in our subsequent discussion.

7.1 Mediators over information retrieval systems

Information retrieval systems (IRS) provide content-based access to a set of (text) documents. The content of the documents (as well as the user queries) is described using an "indexing language" that can be either

(a) a "free" vocabulary consisting of the words that appear in the documents of the collection, excluding those words that carry no information (such as articles) and reducing words to their grammatical root (a task called "stemming"), or

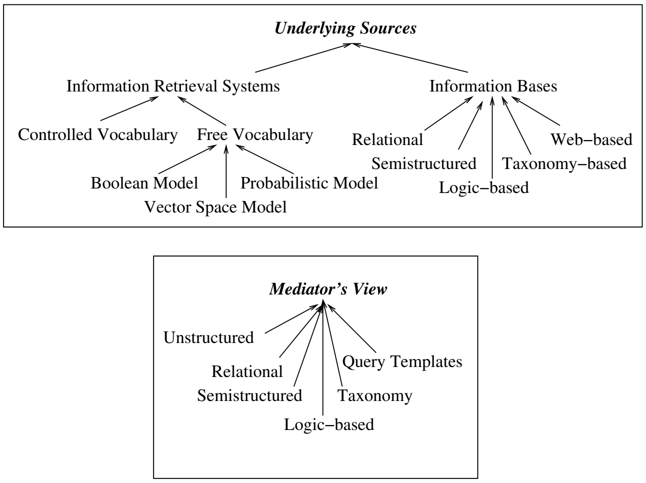


Fig. 22. Rough taxonomy of the mediator approaches

(b) a "controlled" vocabulary that may be different from the set of words that appear in the documents. This vocabulary may be structured by a small set of relations like hyponymy and synonymy.

For the relative merits of each of these approaches, see [21, 62]. One could say that taxonomy-based sources resemble those IRSs that employ the boolean retrieval model (see [5] for a review) and exploit lexical ontologies or word-based thesauri [40] (like WordNet [57] and Roget's thesaurus) for query expansion, i.e., for expanding queries with synonyms, hyponyms, and related terms to improve recall (e.g., see [35, 49,50,55]). However, note that the IRS techniques are applicable only if the objects of the domain have a textual content (not a prerequisite of our approach). Another remarkable difference with our sources is that the taxonomies employed by IRSs usually do not accept the semantic interpretation described in this paper. Lexical ontologies like WordNet [57] are structured using lexical relations (synonymy, hyponymy, antonymy) that 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. The justification of the possible answer (in sources) and the eight answer models (in mediators) does not apply to such ontologies (for more about this problem, see [34]). Instead, techniques like spreading activation [55] are more appropriate if lexical ontologies are employed.

By consequence, our approach is quite different from the mediator approaches that have emerged within the IR community. Specifically, a mediator over IRSs that employ free vocabularies does not have to translate the mediator queries, as each source accepts the same set of queries, i.e., natural language queries. As a consequence, mediators over such sources mainly focus on issues like *source selection* and *result fusion* (metaranking) (e.g., see [6,11,20,28,33,67,76,77]). On the other hand, mediators over systems that employ controlled and structured vocabularies have not received adequate attention until now. To the best of our knowledge, all of the existing approaches focus on ontology merging and not on ontology articulation. Moreover, as they mainly employ lexical ontologies, the mappings between two ontologies consist of lexical relationships, too (in many cases, one term is associated with a set of terms of the other ontology [3]).

Although the controlled indexing languages that are used for information retrieval usually consist of a set of terms structured by a small number of relations (such as hyponymy and synonymy), there are cases where the indexing of the objects is done (especially in the case of a manual indexing process) with respect to more expressive conceptual models representing domain knowledge in a more detailed and precise manner. Such conceptual models can be represented using logic-based languages, and the corresponding reasoning mechanisms can be exploited for retrieving objects. There are several studies that take this conceptual modeling and reasoning approach to information retrieval (e.g., relevance terminological logics [51], four-valued logics [59]). This conceptual modeling approach is useful and effective if the domain is narrow. If the domain is too wide (e.g., the set of all Web pages), then the problem is that it is hard to conceptualize the domain; actually, there are many different ways to conceptualize it, so it is hard to reach a conceptual model of wide acceptance. Thus a mediator over such sources has to tackle complex structural differences (recall the example of Sect. 2). For this purpose, even today ontologies that have a simple structure, like the one we consider, are usually employed for retrieving objects from large collections of objects ([61]).

7.2 Mediators over information bases

We use the term *information bases* to refer to sources that store structured data, not documents. Relational, semistructured, logic-based, and Web-based sources belong to this category. Indeed, there are several approaches to building mediators over relational databases (e.g., see [30,31,47,79]), SGML documents (e.g., see [17]), and Web-based sources (e.g., see [4, 14,15]). We include this discussion here because our sources can be considered as information bases as we do not presuppose that the domain objects have a textual content.

Concerning the kind of mediator view, several approaches have been proposed (as the rightmost taxonomy of Fig. 22 illustrates). Indeed, we have seen mediators whose unified view has the form of a relational schema (e.g., Infomaster [25, 32]), a semantic network (e.g., SIMS [43]), an F-logic schema (e.g., OntoBroker [7,22]), a description logics schema (e.g., Information Manifold [47], OBSERVER [41,42,52], PICSEL [45]), a set of query templates (e.g., TSIMMIS [16,30], HERMES [66]). Furthermore, several data models have been used for conveying information between the mediator and sources (as shown in Fig. 21) including relational tuples (e.g., in Infomaster, SIMS, Information Manifold, OBSERVER), tuples that encode graph data structures (like the OEM in TSIMMIS, or the YAT in [17]), and HTML pages (in Web-oriented approaches).

To identify similarities, differences, and analogies between the above approaches and the one presented in this paper, we first describe a set of layers from which we can view an information base, then we use these layers to discuss the kinds of heterogeneity that may exist between two information bases, and, finally, we provide several observations.

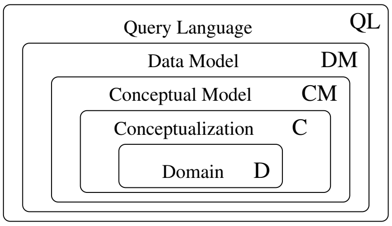
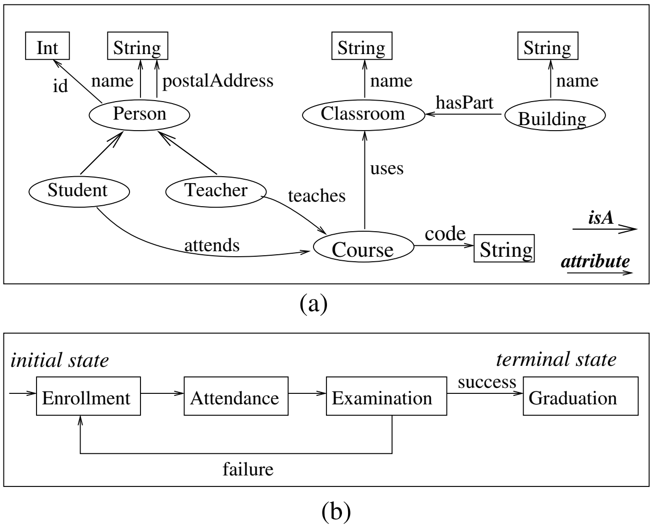


Fig. 23. Layers of a source

A layered view of an information base

We could view a source at five different layers: the *domain*, the *conceptualization*, the *conceptual model*, the *data model*, and the *query language*. We are aware that these distinctions are not crystal clear or widely accepted; however, they enable us to discuss systematically a number of issues and draw analogies. There are dependencies among these layers, as shown in Fig. 23, e.g., the query language layer of a source depends on the data model layer of the source, and so on.

Each source stores information about a part of the real world that we call the *domain* layer of the source. For example, the domain of a source can be the set of all URLs, or the set of all universities, or the set of Greek universities, or the Computer Science Department of the University of Crete (which we call *CSD domain* in the discussion below). The *conceptualization* of a domain is the intellectual lens through which the domain is viewed. For example, one conceptualization of the CSD domain may describe its *static* aspects, i.e., what entities or things exist (e.g., persons, buildings, classrooms, computers), their attributes, and their interrelationships. Another conceptualization may describe its *dynamic* aspects in terms of states, state transitions, and processes (e.g., enrollments, graduations, attendances, teaching). A *conceptual model* is used to describe a particular conceptualization of a domain in terms of a set of (widely accepted) structuring mechanisms that are appropriate for the conceptualization. For example, a conceptual model that describes the static aspects of the CSD domain, using generalization and attribution, is shown in Fig. 24a, while a conceptual model that describes the dynamic aspects of the CSD domain, using states and state transitions, is shown in Fig. 24b. The representation of a conceptual model in a computer is done according to a specific *data model* (e.g., relational, object-oriented, semantic network-based, semistructured). For example, the class `Person` of the conceptual model of Fig. 24 can be represented in the relational model by a relation scheme as follows: `Person(id:Int, name:Str, postalAddress:Str)`. Alternatively, in a different source, it could also be represented using two relation schemes: `PERSON(id:Int, name:Str, addressId:Int)` and `POSTALADDRESS(id:Int, address:Str)`. However, there are also some data models that allow a straightforward representation of the conceptual model, e.g., the semantic network-based data model of SIS-Telos [19,39]. Finally, each source can answer queries expressed in a particular query language. For example, a source may respond to Datalog queries, while another may respond only to SQL queries. In this case, we say that the *query language layers* of these sources are different.



(a)

(b)

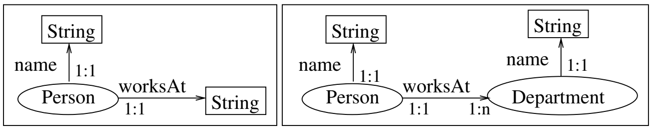
Fig. 24. Two conceptual models of the CSD domain: one for the *static* and one for the *dynamic* aspects

Kinds of heterogeneity

Given a source S_i, we will use D_i to denote the domain, C_i the conceptualization, CM_i the conceptual model, DM_i the data model, and QL_i the query language layer of S_i. Consider now two sources S_1 and S_2. We may have several forms of heterogeneity between these sources, specifically, there are $2^5 = 32$ different cases (due to the five layers). For example, the case $D_1 = D_2, C_1 = C_2, CM_1 \neq CM_2$ means that S_1 and S_2 have the same conceptualization of the (same) domain, but they employ different conceptual models. Even if the conceptual models are expressed using the same structuring mechanisms (e.g., generalization, attribution), they may differ due to:

- *Different naming conventions* (also called naming conflicts). A frequent phenomenon is the presence of homonyms and synonyms.
- *Different scaling schemes*. These occur when different reference systems are used to measure a value, e.g., 1 ft. vs 0.304 m, 23°C vs. 73°F.
- *Different levels of granularity*. For example, CM_1 may contain only a class `Cameras`, while CM_2 may contain the classes `StillCameras` and `MovingPictureCameras`.
- *Structural differences*. For example, CM_1 may contain a class `Person` having an attribute `owns` whose range is a class `ArtificialObject`, and a class `Car` defined as a specialization of `ArtificalObject`, while CM_2 may contain a class `Car` having an attribute `owner` whose range is the class `Person`.

As another example, the case $D_1 = D_2, C_1 = C_2, CM_1 = CM_2, DM_1 \neq DM_2$ means that S_1 and S_2 have same conceptual model, but these models are represented differently in the data model layer. Note that, even if S_1 and S_2 employ the same data model, e.g., the relational, DM_1 and DM_2 may differ in that they may represent the conceptual model differently.



(a) (b)

Fig. 25. Two conceptual models of the CSD domain

Remarks and analogies

Let us now give some remarks and discuss some analogies between our approach and other mediator approaches that have emerged.

- An important remark is that, given an existing source, we usually have at our disposal only its data model and query language layer, and more often than not from these two layers we cannot infer the conceptual model or the conceptualization layer of the source. For example, consider the following relation scheme: `PERSON(name:Str, worksAt:Str)`. The underlying conceptual model could be any of those shown in Fig. 25, as the translations of both a and b to the relational model (by using an algorithm such as the one described in [9]) are identical. Note that, according to a, the domain consists of entities of one kind, i.e., persons, while according to b, the domain consists of two kinds of entities: persons and departments. Moreover, although two sources may have the same conceptual model, e.g., conceptual model a, their representation in the data model may differ. For example, the conceptual model a could be represented in the relational model by one relation scheme (as we saw before) or by the following two relational schemes: `PERSON(name:Str, worksAt:Int)` and `DEP(depId:Int, name:Str)`. We believe that this is the basic reason why information integration is a difficult and laborious task.
- According to the layered view described above, the sources of our mediators (a) may have different domains (i.e., may index different sets of objects), (b) conceptualize their domains similarly (i.e., all C_i are denumerable sets of objects), (c) may have different conceptual models (i.e., different taxonomies), (d) may have different query languages (recall the remark at the end of Sect. 6.1).
- In relational mediators (see [31] for a review), the mediator view is represented as a relational database schema. Relational mediators have some critical differences with our mediators. Relational mediators and their sources are *schema-based*, while our mediators and their sources are *taxonomy-based*. Also, recall that the relational model is value-based, not object-based. This implies that the conceptualization and the conceptual model of a relational source is hidden, or unclear. Therefore, mediators over such sources "work" on the data model layer. Instead, we propose a totally different conceptual modeling approach for both sources and mediators.

  Concerning source descriptions, we can distinguish the *local-as-view* (LAV) and the *global-as-view* (GAV) approaches (see [13,46] for a comparison). In the LAV approach, the source relations are defined as relational views over the mediator's relations, while in the GAV approach the mediator relations are defined as views of the source relations. The former approach offers flexibility in representing the contents of the sources, but query answering is "hard" because this requires answering queries using views [24,36,74]. On the other hand, the GAV approach offers easy query answering (expansion of queries until arriving at the source relations), but the addition/deletion of a source implies updating the mediator view, i.e., the definition of the mediator relations. It is worth mentioning here that as our articulations contain relationships between single terms, these kinds of mappings enjoy the benefits of both GAV and LAV approaches, i.e., they have (a) the query processing simplicity of the GAV approach, as query processing basically reduces to unfolding the query using the definitions specified in the mapping so as to translate the query in terms of accesses (i.e., queries) to the sources, and (b) the modeling scalability of the LAV approach, i.e., the addition of a new underlying source does not require changing the previous mappings. On the other hand, term-to-query articulations (presented in Sect. 6.1) resemble the GAV approach.

  Concerning the translation facilities, relational mediators attempt to construct *exact translations* of SQL queries, while our mediators allow *approximate translations* of boolean expressions through their articulations. We might say that the answers returned by a relational mediator correspond to the answers returned by a taxonomy-based mediator in the I^{-}_{l-} model.

  Moreover, in several approaches (e.g., in Infomaster) a predicate corresponding to a source relation can appear only in the head or in the tail of a rule. This means that granularity heterogeneities cannot be tackled easily.
- A different approach to mediators can be found in [12], which presents the fundamental features of a declarative approach to information integration based on description logics. The authors describe a methodology for integrating relational sources, and they resort to very expressive logics to bridge the heterogeneities between the unified view of the mediator and the source views. However, the reasoning services for supporting translations have exponential complexity, as opposed to the complexity of our mediators, which is polynomial. In addition, the eight possible answers of our approach allow one to provide a novel query relaxation facility.
- One difference between our approach and the system OBSERVER [41,42,52] is that OBSERVER requires merging the ontologies of all underlying sources. Instead, we just articulate the taxonomies of the sources with the taxonomy of the mediator. Moreover, the compatibility condition introduced here allows the mediator to draw conclusions about the structure of a source taxonomy without having to store that taxonomy.
- In the approximate query mapping approach of [14,15], the translated queries minimally subsume the original ones. However, the functionality offered by our mediators is different, first because we support negation while they do not, and second because our mediators support multiple operation modes, one of which is the case where the translated queries subsume the original ones.
- An alternative solution to the problem of query relaxation in mediators is *query repairing* described in [8]. If the sub-

mitted query yields no answer, then the mediator provides to the user an answer to a “similar” query. The selection of this query is based on a measure of similarity between the concepts and the predicates, which is based on the taxonomic structure of the mediator’s ontology. In our opinion, the eight answer models of our mediators offer a better founded approach to query relaxation.

8 Concluding remarks

We have presented an approach to providing uniform access to multiple taxonomy-based sources through mediators that render the heterogeneities (naming, contextual, granularity) of the sources transparent to users. This paper integrates and extends the work presented in [70] and [73] and was inspired by the approach presented in [65].

A user of the mediator, apart from being able to pose queries in terms of a taxonomy that was not used to index the objects of the sources being searched, gets an answer comprised of objects that are accompanied by descriptions over the mediator’s taxonomy. A mediator is seen as just another source, but *without* a stored interpretation. An interpretation for the mediator is defined based on the interpretations stored at the sources and on the *articulations* between the mediator and the sources; and in fact, we have seen *eight* different ways of defining a mediator interpretation depending on the nature of the answers that the mediator provides to its users (Table 1). Since the resulting mediator models are ordered, they can be used to support a form of *query relaxation*.

Articulations can be defined by humans, but they can also be constructed automatically or semiautomatically in some specific cases, following a model-driven approach (e.g., [2, 53,64]) or a data-driven approach (e.g., [3,23,37,44,60,68]).

The distinctive features of our approach are the following:

- We assume that all sources have the same *domain* and the *same* conceptualization of that domain. The intended domain is the Web, and each source views the Web as a set of objects Obj (URLs) and stores information about a subset of it (i.e., $O_i \subseteq Obj$). This means that each object has a *unique identity* over all sources. From this point of view, our mediators may be called object-oriented, as opposed to mediators over relational sources, which may be called value-oriented.
- We consider that the conceptual layer of each source is a triple $(T, \preceq, I)$. This conceptual modeling approach has two main advantages: (a) It is easy to create the conceptual model of a source or a mediator, and (b) the integration of information from multiple sources can be done easily. Indeed, articulations offer a *uniform* method of bridging naming, contextual, and granularity heterogeneities between the conceptual models of the sources. Given this conceptual modeling approach, the mediator does not have to tackle complex structural differences between the sources (as happens with relational mediators). Moreover, it allows the integration of *schema* and *data* in a uniform manner. For example, consider a source S having the conceptual model shown in Fig. 3a and a source S' having the conceptual model shown in Fig. 3b, and suppose that both sources are implemented in the relational model. In source S, the concept `wood` would be represented at the data level (it would be an element of the domain of an attribute), while in S', it would be a relation. Furthermore, this approach makes the automatic construction of articulations feasible [68].

Summarizing, the taxonomy-based mediation approach presented here offers the following advantages:

- *Easy construction of mediators*
  A mediator can be easily constructed even by ordinary Web users. Indeed, the simple conceptual modeling approach that we adopt makes the definition of the mediator’s taxonomy and articulations very easy.
- *Query relaxation*
  Often a query to a mediator yields no answer. The sure and the possible answers of sources, as well as a mediator’s several modes of operation, offer a solution to this problem.
- *Efficient query evaluation*
  The time complexity of query translation at the mediator is linear with respect to the size of the subsumption relations of the mediator.
- *Scalability*
  Articulation (instead of merging) enables a natural, incremental evolution of a network of sources. The taxonomies employed by Web catalogs contain very large numbers of terms (e.g., the taxonomy of Open Directory contains 450,000 terms). Therefore, the *articulation* of taxonomies has several advantages compared to taxonomy *merging*. First, merging would introduce storage and performance overheads. Second, full merging is a laborious task that in many cases does not pay off because the integrated taxonomy becomes obsolete when the taxonomies involved change. Another problem with full merging is that it usually requires full consistency, which may be hard to achieve in practice, while articulation can work on locally consistent parts of the taxonomies involved. However, note that the taxonomies considered here present no consistency problems. There may only be long cycles of subsumption relationships that induce big classes of equivalent terms.
- *Applicability*
  The taxonomy-based approach presented provides a flexible and formal framework for integrating data from several sources and/or for personalizing the contents of one or more sources. The taxonomies considered fit quite well with the content-based organizational structure of Web catalogs (e.g., Yahoo!, Open Directory), keyword hierarchies (e.g., ACM’s thesaurus), XFML [1] taxonomies, and personal bookmarks. By defining a mediator, the user can employ his own terminology to access and query several Web catalogs, specifically those parts of the catalogs that are of interest to him.
  Moreover, as a mediator can also have a stored interpretation, our approach can lead to a network of articulated sources. Recall that a mediator can translate the descriptions of the objects returned by the underlying sources. This implies that all (or some) of these objects can be straightforwardly stored in the mediator base (under terms of the mediator taxonomy).

An interesting line of research would be to investigate query evaluation and updating in a network of articulated sources. Another interesting line would be to investigate how the mediator can exploit the object indexes that are returned by a compatible source in order to check whether that source remains compatible. If we consider sources that answer queries by returning an ordered set of objects, the mediator should also return ordered sets of objects. It would then be interesting to investigate whether the work presented in this paper can be integrated with the work presented in [67] and [26].

Appendix: Proofs

Proposition 1. If I is an interpretation of T, then I^- is the unique minimal model of T that is greater than or equal to I.

Proof. (I^- is a model of T)
$t \preceq t' \Rightarrow tail(t) \subseteq tail(t') \Rightarrow \bigcup\{I(s) \mid s \in tail(t)\} \subseteq \bigcup\{I(s) \mid s \in tail(t')\} \Rightarrow I^-(t) \subseteq I^-(t')$. Thus I^- is a model of T.

(I^- is the unique minimal model of T which is greater than I.)
Let I' be a model of T that is larger than I. Below we prove that $I^- \sqsubseteq I'$. By the definition of $I^-(t)$, if $o \in I^-(t)$, then either $o \in I(t)$ or $o \in I(s)$ for a term s such that $s \preceq t$. However, if $o \in I(t)$, then $o \in I'(t)$ too because I' is larger than I, and if $o \in I(s)$ for a term s such that $s \preceq t$, then $o \in I'(t)$ too because I' is a model of T. We conclude that for every $o \in I^-(t)$ it holds that $o \in I'(t)$. Thus I^- is the unique minimal model T that is larger than I. ◇

Proposition 2. If I is an interpretation of T, then I^+ is a model of T and $I \sqsubseteq I^- \sqsubseteq I^+$.

Proof. (I^+ is a model of T)
$t \preceq t' \Rightarrow \{u \mid u \in head(t)\} \supseteq \{u \mid u \in head(t')\} \Rightarrow$
$\{u \mid u \in head(t) \text{ and } u \not\sim t\} \supseteq$
$\{u \mid u \in head(t') \text{ and } u \not\sim t'\} \Rightarrow$
$\bigcap\{I(u) \mid u \in head(t) \text{ and } u \not\sim t\} \subseteq$
$\bigcap\{I(u) \mid u \in head(t') \text{ and } u \not\sim t'\} \Rightarrow I^+(t) \subseteq I^+(t')$.

($I^- \sqsubseteq I^+$)
Clearly, if $t \in T$, $u \in head(t)$ and $u \not\sim t$, then in every model I of T we have $I(t) \subseteq I(u)$. Thus this also holds in the model I^-, i.e., $I^-(t) \subseteq I^-(u)$. From this we conclude that for every $t \in T$:
$I^-(t) \subseteq \bigcap\{I^-(u) \mid u \in head(t) \text{ and } u \not\sim t\} = I^+(t)$. Thus $I^- \sqsubseteq I^+$. ◇

Proposition 3. The answer models of the mediator are ordered as follows:

(a) $I^-_{l-} \sqsubseteq I^-_{l+}$
(b) $I^-_{u-} \sqsubseteq I^-_{u+}$
(c) $I^+_{l-} \sqsubseteq I^+_{l+}$
(d) $I^+_{u-} \sqsubseteq I^+_{u+}$
(e) $I^-_{l-} \sqsubseteq I^+_{l-}$
(f) $I^-_{l+} \sqsubseteq I^+_{l+}$
(g) $I^-_{u-} \sqsubseteq I^+_{u-}$
(h) $I^-_{u+} \sqsubseteq I^+_{u+}$

Proof. The proofs of propositions (a)–(d) derive easily from the fact that in every model I_i of a source S_i it holds that $I^-_i \sqsubseteq I^+_i$.

The proofs of propositions (e)–(h) derive easily from the fact that in every model I of the mediator it holds that $I^- \sqsubseteq I^+$. ◇

Proposition 4. If all sources are compatible with the mediator, then

(1) $I^-_{l-} \sqsubseteq I^-_{u-}$
(2) $I^-_{l+} \sqsubseteq I^-_{u+}$
(3) $I^+_{l-} \sqsubseteq I^+_{u-}$
(4) $I^+_{l+} \sqsubseteq I^+_{u+}$

Proof. Let t be a term of T. Clearly, for every $s \in tail_i(t)$ and $u \in head_i(t)$ it holds that $s a_i u$ (because $s a_i t$ and $t a_i u$). Since the source S_i is compatible, we know that $s a_i u \Rightarrow s \preceq_i u$. This implies that in every model I_i of T_i it holds that

$$\bigcup\{I_i(s) \mid s a_i t\} \subseteq \bigcap\{I_i(u) \mid t a_i u\} \Leftrightarrow$$
$$I_i(t^i_l) \subseteq I_i(t^i_u).$$

From $I_i(t^i_l) \subseteq I_i(t^i_u)$ we infer that $I^-_i(t^i_l) \subseteq I^-_i(t^i_u)$ and $I^+_i(t^i_l) \subseteq I^+_i(t^i_u)$. From this we obtain propositions (1)–(4).

For example, the proof of proposition (1) i.e., $I^-_{l-} \sqsubseteq I^-_{u-}$, and proposition (3) i.e., $I^+_{l-} \sqsubseteq I^+_{u-}$, is derived as follows: Since $\forall\ t \in T$ and $\forall\ i = 1..k$ it holds that $I^-_i(t^i_l) \subseteq I^-_i(t^i_u)$, we conclude that

$$\bigcup_{i=1..k} I^-_i(t^i_l) \subseteq \bigcup_{i=1..k} I^-_i(t^i_u) \Leftrightarrow$$
$$I_{l-}(t) \subseteq I_{u-}(t) \Rightarrow \begin{cases} I^-_{l-} \sqsubseteq I^-_{u-}\ (I_1 \sqsubseteq I_3) \\ I^+_{l-} \sqsubseteq I^+_{u-}\ (I_5 \sqsubseteq I_7). \end{cases}$$

Proposition 5. If $q = t \in T$, then $I^-_l(t)$ and $I^-_u(t)$ can be evaluated as follows:

$$I^-_l(t) = \bigcup_{i=1..k} I_i(q^i_l(t)) \ \text{ where }\ q^i_l(t) = \bigvee\{s^i_l \mid s \preceq t\}$$
$$I^-_u(t) = \bigcup_{i=1..k} I_i(q^i_u(t)) \ \text{ where }\ q^i_u(t) = \bigvee\{s^i_u \mid s \preceq t\}.$$

Proof.

$$\begin{aligned} I^-_l(t) &= \bigcup\{I_l(s) \ \mid\ s \in tail(t)\} = \bigcup\{I_l(s) \ \mid\ s \preceq t\} \\ &= \bigcup\{\cup_{i=1..k} I_i(s^i_l) \ \mid\ s \preceq t\} \\ &= \bigcup_{i=1..k} \{\cup I_i(s^i_l) \ \mid\ s \preceq t\} \\ &= \bigcup_{i=1..k} I_i(\bigvee\{s^i_l \ \mid\ s \preceq t\}) = \bigcup_{i=1..k} I_i(q^i_l(t)). \end{aligned}$$

Analogously, we prove that $I^-_u(t) = \bigcup_{i=1..k} I_i(q^i_u(t))$. ◇

Proposition 6. $D^-(o) = \bigcup\{head(t) \mid o \in I(t)\}$

Proof.

$$\begin{aligned} D^-(o) &= \{t \in T \mid o \in I^-(t)\} \\ &= \{t \in T \mid o \in \cup\{I(s) \mid s \preceq t\}\} \\ &= \{t \in T \mid o \in I(s) \text{ and } s \preceq t\} \\ &= \bigcup\{head(s) \mid o \in I(s)\}. \end{aligned}$$

Proposition 7. $D^+(o) = \{t \mid head(t) \setminus \{t' | t' \sim t\} \subseteq D^-(o)\}$.

Proof. $t \in D^+(o) \Leftrightarrow o \in I^+(t) \Leftrightarrow o \in \bigcap\{I^-(u) \mid u \in head(t) \text{ and } u \not\sim t\} \Leftrightarrow$
$o \in I^-(u)\ \forall\ u \in head(t)$ s.t. $u \not\sim t \Leftrightarrow u \in D^-(o)\ \forall\ u \in head(t)$ s.t. $u \not\sim t \Leftrightarrow head(t) \setminus \{t'|t' \sim t\} \subseteq D^-(o)$. ◇

Proposition 8.

$$D_l(o) = \bigcup_{i=1..k} D_l^i(o), \quad \text{where}$$

$$D_l^i(o) = \{t \in T \mid t_i \in D_i(o) \text{ and } t_i\ a_i\ t\}$$

$$D_u(o) = \bigcup_{i=1..k} D_u^i(o), \quad \text{where}$$

$$D_u^i(o) = \{t \in T \mid (head_i(t) \neq \emptyset \text{ and } head_i(t) \subseteq D_i(o)) \text{ or } (head_i(t) = \emptyset \text{ and } t_i \in D_i(o) \text{ and } t_i\ a_i\ t)\}.$$

Proof. Consider a mediator over a single source S_i and let o be an object stored at that source. Let $t_i \in D_i(o)$, where I_i is the answer model of the source S_i that is used by the mediator. If $\exists\ t \in T$ such that $t_i\ a_i\ t$, then certainly $o \in I_l(t)$ (since $I_l(t) = \bigcup\{I(t_i) \mid t_i\ a_i\ t\}$), thus $t \in D_l(o)$. Hence $D_l(o) = \{t \in T \mid t_i \in D_i(o) \text{ and } t_i\ a_i\ t\}$. However, since there are many sources, we denote the right part of the above formula by $D_l^i(o)$, and since an object may belong to more than one source, we arrive at the following: $D_l(o) = \bigcup_{i=1..k} D_l^i(o)$.

Consider again a mediator over a single source S_i and let o be an object stored at that source. If $\exists\ t \in T$ such that $head_i(t) \neq \emptyset$ and $head_i(t) \subseteq D_i(o)$, then certainly $o \in I_u(t)$ (since $I_u(t) = \bigcap\{I(t_i) \mid t\ a_i\ t_i\}$), thus certainly $t \in D_u(o)$. If $\exists\ t \in T$ such that $head_i(t) = \emptyset$ and there is a $t_i \in D_i(o)$ and $t_i\ a_i\ t$, then certainly $o \in I_u(t)$ (since in this case $I_u(t) = \bigcup\{I(t_i) \mid t_i\ a_i\ t\}$). Hence $D_u(o) = \{t \in T \mid (head_i(t) \neq \emptyset \text{ and } head_i(t) \subseteq D_i(o)) \text{ or } (head_i(t) = \emptyset \text{ and } t_i \in D_i(o) \text{ and } t_i\ a_i\ t)\}$. However, since there are many sources, we denote the right part of the above formula by $D_u^i(o)$, and since an object may belong to more than one source, we arrive at the following: $D_u(o) = \bigcup_{i=1..k} D_u^i(o)$. ◇

Acknowledgements. Part of this work was conducted while the first two authors were visiting Meme Media Laboratory, Hokkaido University, Sapporo, Japan. Many thanks to Professor Tanaka, director of the Meme Media Laboratory, and to the University of Hokkaido for their hospitality. We also want to thank the anonymous referees for their comments, which improved this paper.

References

1. "XFML: eXchangeable Faceted Metadata Language". http://www.xfml.org.
2. Amann B, Fundulaki I (1999) Integrating ontologies and thesauri to build RDF schemas. In: Proceedings of the 3rd European conference for digital libraries ECDL'99, Paris, France, 22 September 1999, pp 234–253
3. Amba S (1996) Automatic linking of thesauri. In: Proceedings of SIGIR'96, Zurich, Switzerland, 18–22 August 1996, pp 181–186. ACM Press, New York
4. Ambite JL, Ashish N, Barish G, Knoblock CA, Minton S, Modi PJ, Muslea I, Philpot A, Tejada S (1998) Ariadne: a system for constructing mediators for Internet sources. In: Proceedings of the ACM SIGMOD international conference on management of data, Seattle, 2–4 June, 1998, pp 561–563
5. Baeza-Yates R, Ribeiro-Neto B (1999) Modern information retrieval. ACM Press/Addison-Wesley, Reading, MA
6. Baumgarten C (1999) Probabilistic information retrieval in a distributed heterogeneous environment. PhD thesis, Technical University of Dresden, Dresden, Germany
7. Benjamins VR, Fensel D (1998) Community is knowledge! in (KA)^2. In: Proceedings of KAW'98, Alberta, Canada, 18–23 April, 1998
8. Bidault A, Froidevaux C, Safar B (2000) Repairing queries in a mediator approach. In: Proceedings of the ECAI'00, Berlin, 20–25 August 2000, pp 406-410
9. Boman M, Bubenko JA, Johannesson P, Wangler B (1997) Conceptual modelling. Prentice-Hall, Upper Saddle River, NJ
10. Boolos G (1998) Logic, logic and logic. Harvard University Press, Cambridge, MA
11. Callan JP, Lu Z, Croft WB (1995) Searching distributed collections with inference networks. In: Proceedings of the 18th international conference on research and development in information retrieval, Seattle, 9–13 July, 1995, pp 21–18
12. Calvanese D, de Giacomo G, Lenzerini M, Nardi D, Rosati R (1998) Description logic framework for information integration. In: Proceedings of the 6th international conference on the principles of knowledge representation and reasoning (KR-98), Trento, Italy, 2–5 June 1998, pp 2–13
13. Calvanese D, de Giacomo G, Lenzerini M (2001) A framework for ontology integration. In: Proceedings of the 2001 international Semantic Web working symposium (SWWS 2001), Stanford, CA, 30 July–1 August 2001, pp 303–316
14. Chang C-CK, García-Molina H (1999) Mind your vocabulary: query mapping across heterogeneous information sources. In: Proceedings of the ACM SIGMOD, Philadelphia, 1–3 June 1999, pp 335–346
15. Chang C-CK, García-Molina H (2001) Approximate query mapping: accounting for translation closeness. J Very Large Databases 10(2–3):155–181
16. Chawathe S, Garcia-Molina H, Hammer J, Ireland K, Papakonstantinou Y, Ullman J, Widom J (1994) The TSIMMIS project: integration of heterogeneous information sources. In: Proceedings of IPSJ, Tokyo, October 1994, pp 7–18
17. Cluet S, Delobel C, Siméon J, Smaga K (1998) Your mediators need data conversion! In: Proceedings of the ACM SIGMOD international conference on management of data, Seattle, 2–4 June 1998, pp 177–188
18. Codd EF (1970) A relational model of data for large shared data banks. Commun ACM 13(6):377–387
19. Constantopoulos P, Doerr M, Vassiliou Y (1993) Repositories for software reuse: the software information base. In: Proceedings IFIP WG 8.1 conference on information system development process, Como, Italy, September 1993, pp 285–307
20. Craswell N, Hawking D, Thistlewaite P (1999) Merging results from isolated search engines. In: Proceedings of the 10th Australasian database conference, Auckland, New Zealand, 18–21 January 1999, pp 189–200
21. Croft B (1993) Knowledge-based and statistical approaches to text retrieval. IEEE Expert 9:8–12
22. Decker S, Erdmann M, Fensel D, Studer R (1999) Ontobroker: ontology based access to distributed and semi-structured information. In: Semantic issues in multimedia systems. Kluwer, Dordrecht
23. Doan A, Madhavan J, Domingos P, Halevy A (2002) Learning to map between ontologies on the Semantic Web. In: Proceedings of the World Wide Web Conference (WWW-2002), Honolulu, 7–11 May 2002, pp 662–673

24. Duschka OM, Genesereth MR (1997a) Answering recursive queries using views. In: Proceedings of PODS 1997, Tucson, AZ, 12–14 May 1997, pp 109–116
25. Duschka OM, Genesereth MR (1997b) Query planning in infomaster. In: Proceedings of the 12th annual ACM symposium on applied computing, SAC'97, San Jose, February 1997, pp 109–111
26. Fagin R (1999) Combining fuzzy information from multiple systems. J Comput Sys Sci 58(1):83–99
27. Fan Y, Gauch S (1999) Adaptive agents for information gathering from multiple, distributed information sources. In: Proceedings of the 1999 AAAI symposium on intelligent agents in cyberspace, Stanford, CA, March 1999, pp 40–46
28. Fuhr N (1999) A decision-theoretic approach to database selection in networked IR. ACM Trans Inf Sys 17(3):229–249
29. Galton A (1990) Logic for information technology. Wiley, New York
30. Garcia-Molina H, Papakonstantinou Y, Quass D, Rajaraman A, Sagiv Y, Ullman J, Vassalos V, Widom J (1994) The TSIMMIS approach to mediation: data models and languages. In: Proceedings of IPSJ, Tokyo, October 1994, pp 7–18
31. Garcia-Molina H, Ullman JD, Widom J (2000) Database system implementation, chap 11. Prentice-Hall, Upper Saddle River, NJ
32. Genesareth MR, Keller AM, Duschka O (1997) Infomaster: an information integration system. In: Proceedings of 1997 ACM SIGMOD, Tucson, AZ, May 1997, pp 539–542
33. Gravano L, Garcia-Molina H (1995) Generalizing GlOSS to vector-space databases and broker hierarchies. In: Proceedings of the 21st conference on very large databases, Zurich, Switzerland, 11–15 September 1995, pp 78–89
34. Guarino N (1998) Some ontological principles for designing upper level lexical resources. In: Proceedings of the 1st international conference on language resources and evaluation, Granada, Spain, May 1998, pp 527–534
35. Guarino N, Masolo C, Vetere G (1999) OntoSeek: content-based access to the Web. IEEE Intell Sys 14(3):70–80
36. Halevy AY (2001) Answering queries using views: a survey. J Very Large Databases 10(4):270–294
37. Helleg H, Krause J, Mandl T, Marx J, Muller M, Mutschke P, Strogen R (2001) Treatment of semantic heterogeneity in information retrieval. Technical Report 23, Social Science Information Centre, Köln, Germany, May 2001. http://www.gesis.org/en/publications/reports/iz_working_papers/
38. Howe A, Dreilinger D (1997) SavvySearch: a metasearch engine that learns which search engines to query. AI Mag 18(2):19–25
39. Information Systems Laboratory. The Semantic Index System (SIS). Institute of Computer Science Foundation for Research and Technology Hellas. http://zeus.ics.forth.gr/forth/ics/isl/r-d-activities/semantic_index_system.html
40. International Organization for Standardization (1986) Documentation – guidelines for the establishment and development of monolingual thesauri. Ref. No ISO 2788-1986
41. Kashyap V, Sheth A (1996) Semantic and schematic similarities between database objects: a context-based approach. J Very Large Databases 5(4):276–304
42. Kashyap V, Sheth A (1998) Semantic heterogeneity in global information systems: the role of metadata, context and ontologies. In: Cooperative information systems: trends and directions. Academic, San Diego
43. Knoblock C, Arens Y, Hsu C-N (1994) Cooperating Agents for Information Retrieval. In: Procedings of the 2nd international conference on cooperative information systems, Toronto, 17–20 May 1994, pp 122–133
44. Lacher M, Groh G (2001) Facilitating the exchange of explicit knowledge through ontology mappings. In: Proceedings of the 14th international FLAIRS conference, Key West, FL, 21–23 May 2001, pp 305–309
45. Lattes V, Rousset M-C (1998) The use of CARIN language and algorithms for information integration: the PISCEL project. In: Proceedings of the 2nd international and workshop on intelligent information integration, Brighton Centre, Brighton, interdisciplinary UK, August 1998
46. Lenzerini M (2002) Data integration: a theoretical perspective. In: Proceedings of ACM PODS 2002, Madison, WI, June 2002, pp 233–246
47. Levy AY, Srivastava D, Kirk T (Data model and query evaluation in global information systems. J Intell Inf Sys 5(2):121–143
48. Luke S, Spector L, Rager D, Hendler J (1997) Ontology-based Web Agents. In: Proceedings of the 1st international conference on autonomous agents, Marina del Rey, CA, 5–8 February 1997, pp 59–66. http://www.cs.umd.edu/projects/plus/SHOE/
49. Mazur Z (1994) Models of a distributed information retrieval system based on thesauri with weights. Inf Process Manage 30(1):61–77
50. McGuinness DL (1998) Ontological issues for knowledge-enhanced search. In: Proceedings of FOIS'98, Trento, Italy, June 1998. IOS Press, Amsterdam
51. Meghini C, Straccia U (1996) A relevance terminological logic for information retrieval. In: Proceedings of SIGIR'96, Zurich, Switzerland, August 1996, pp 197–205
52. Mena E, Kashyap V, Sheth A, Illarramendi A (1996) OBSERVER: an approach for query processing in global information systems based on interoperation across preexisting ontologies. In: Proceedings of the 1st IFCIS international conference on cooperative information systems (CoopIS'96), Brussels, Belgium, June 1996, pp 14–25. IEEE Press, New York
53. Mitra P, Wiederhold G, Jannink J (1999) Semi-automatic integration of knowledge sources. In: Proceedings of the 2nd international conference on information fusion, Sunnyvale, CA, July 1999
54. Nuutila E (1995) Efficient transitive closure computation in large digraphs. PhD thesis, Acta Polytechnica Scandinavica, Helsinki, 1995. http://www.cs.hut.fi/∼enu/thesis.html
55. Paice C (1991) A thesaural model of information retrieval. Inf Process Manage 27(5):433–447
56. Prieto-Diaz R (1991) Implementing faceted classification for software reuse. Commun ACM 34(5):88–97
57. Princeton University Cognitive Science Laboratory. WordNet: a lexical database for the English language. http://www.cogsci.princeton.edu/∼wn
58. Ranganathan SR (1965) The colon classification. In: Artandi S (ed) Rutgers series on systems for the intellectual organization of information, vol IV. Graduate School of Library Science, Rutgers University, New Brunswick, NJ
59. Rolleke T, Fuhr N (1996) Retrieval of complex objects using a four-valued logic. In: Proceedings of SIGIR'96, Zurich, Switzerland, August 1996, pp 206–214
60. Ryutaro I, Hideaki T, Shinichi H (2001) Rule induction for concept hierarchy alignment. In: Proceedings of the 2nd workshop on ontology learning at the 17th international conference on AI (IJCAI), Seattle, 4 August 2001.
61. Sacco GM (2000) Dynamic taxonomies: a model for large information bases. IEEE Trans Knowl Data Eng 12(3):468–479
62. Salton G (1983) Introduction to modern information retrieval. McGraw-Hill, New York
63. Selberg E, Etzioni O (1995) Multi-service search and comparison using the MetaCrawler. In: Proceedings of the 1995 World Wide Web conference, Boston, December 1995

64. Sintichakis M, Constantopoulos P (1997) A method for monolingual thesauri merging. In: Proceedings of the 20th international conference on research and development in information retrieval, ACM SIGIR'97, Philadelphia, July 1997, pp 129–138
65. Spyratos N (1987) The partition model: a deductive database model. ACM Trans Database Sys 12(1):1–37
66. Subrahmanian VS, Adah S, Brink A, Emery R, Rajput A, Ross R, Rogers T, Ward C (1996) HERMES: a heterogeneous reasoning and mediator system. http://www.cs.umd.edu/projects/hermes/overview/paper
67. Tzitzikas Y (2001) Democratic data fusion for information retrieval mediators. In: Proceedings of the ACS/IEEE international conference on computer systems and applications, Beirut, Lebanon, June 2001
68. Tzitzikas Y, Meghini C (2003a) Ostensive automatic schema mapping for taxonomy-based peer-to-peer systems. In: Proceedings of the 7th international workshop on cooperative information agents, CIA-2003, Helsinki, Finland, August 2003. Lecture notes on artificial intelligence, vol 2782, pp 78–92. (Best Paper Award)
69. Tzitzikas Y, Meghini C (2003b) Query evaluation in peer-to-peer networks of taxonomy-based sources. In: Proceedings of the 19th international conference on cooperative information systems, CoopIS'2003, Catania, Sicily, Italy, November 2003
70. Tzitzikas Y, Spyratos N, Constantopoulos P (2001) Mediators over ontology-based information sources. In: Proceedings of the 2nd international conference on Web information systems engineering, WISE 2001, Kyoto, Japan, December 2001, pp 31–40
71. Tzitzikas Y, Analyti A, Spyratos N, Constantopoulos P (2003a) An algebraic approach for specifying compound terms in faceted taxonomies. In: Proceedings of the 13th European-Japanese conference on information modelling and knowledge bases, Kitakyushu, Japan, June 2003
72. Tzitzikas Y, Meghini C, Spyratos N (2003b) Taxonomy-based conceptual modeling for peer-to-peer networks. In: Proceedings of the 22nd international conference on conceptual modeling, ER'2003, Chicago, October 2003, pp 446–460
73. Tzitzikas Y, Spyratos N, Constantopoulos P (2002) Query evaluation for mediators over Web catalogs. Int J Inf Theories Appl 9(2)
74. Ullman JD (1997) Information integration using logical views. In: Proceedings of the 6th international conference on database theory (ICDT-97), Delphi, Greece, 8–10 January 1997, pp 19–40
75. Van Harmelen F, Fensel D (1999) Practical knowledge representation for the Web. In: Proceedings of the workshop on intelligent information integration, IJCAI'99, Stockholm, Sweden, 31 July 1999
76. Vorhees E, Gupta N, Johnson-Laird B (1995) The collection fusion problem. In: Proceedings of the 3rd text retrieval conference (TREC-3), Gaithersburg, MD, November 1995.
77. Vorhees E (1997) Multiple search engines in database merging. In: Proceedings of the 2nd ACM international conference on digital libraries, 25–28 July 1997, Philadelphia, pp 93–102
78. Wiederhold G (1992) Mediators in the architecture of future information systems. IEEE Comput 25:38–49
79. Yerneni R, Li C, Garcia-Molina H, Ullman J (1999) Computing capabilities of mediators. In: Proceedings of ACM SIGMOD'99, Philadelphia, 1–3 June 1999, pp 443-454