Working notes of the ISWC-04 Workshop on

Meaning Coordination and Negotiation (MCN-04)

held in conjunction with

3rd International Semantic Web Conference (ISWC-2004)

8 November 2004
Hiroshima Prince Hotel
Hiroshima, Japan

Paolo Bouquet & Luciano Serafini (Eds.)

Meaning Coordination and Negotiation (MCN-04)

Paolo Bouquet and Luciano Serafini

[1] Dept. of Information and Communication Technology – University of Trento (Italy)
[2] ITC-IRST – Trento (Italy)
bouquet@dit.unitn.it, serafini@itc.it

A short introduction

One of the key challenges in the development of open distributed systems – like the Semantic Web – is enabling the exchange of meaningful information across applications which may use autonomously developed models/schemas for organizing locally available data, and need to interact/collaborate to achieve their users' goals. Typical examples are databases using different schemas, document repositories using different classification structures or annotated with respect to different ontologies, file systems, poorly annotated multimedia content.

One possible approach to this problem is that of creating global schemas (or shared models) onto which local schemas are mapped and thus interoperated. This "centralized" approach may work in restricted environments, like a small corporate Intranet. However, in open environments (like the Web), it does not seem a viable solution, as it can be very difficult to reconciliate/integrate schemas/models that suit different needs in a single shared model; in addition, it would be almost impossible to maintain such a shared model in a highly dynamic environment.

The aim of this workshop on **Meaning Coordination and Negotiation** (MCN-04)is to investigate an alternative approach to semantic interoperation, namely an approach in which no global schemas are presupposed, and schemas/models are directly mapped onto each other in a "peer-to-peer" spirit. A requirement of the proposed approach is that it must be applicable to scenarios where peers that cannot assess semantic problems by "looking into each other's head", like humans or software agents (what we call semantically autonomous entities).

In such a scenario, it is possible to distinguish between two different processes:

- a process of *meaning coordination*, through which the involved parties try to establish mappings between the meaning of a collection of expressions. Such an agreement could result, for example, in a collection of mappings between their ontologies/schemas;
- a process of *meaning negotiation*, namely the process of solving semantic conflicts among parties when a direct mapping is not possible (e.g., different parties adopt with different ontological assumptions, and this makes impossible for them to find a correspondence between the meaning of what they say).

In game theoretic terms, the first is a coordination problem, as (i) all parties have a common interest in achieving such an agreement, but (ii) there are many possible solutions to the problem, and thus the selection of one of these solutions can be problematic;

the second is a negotiation problem, as (i) an agreement is valuable for all parties, but (ii) parties may have conflicting preferences over which solution should be selected, so that every agreement implies that at least someone has to concede to some extent to other party.

The problem of meaning coordination and negotiation can be addressed from many different perspectives, using different conceptual and technological tools, and with different motivations in mind. So we expect that the workshop will attract people from very different fields, such as knowledge representation, ontology engineering, agents, databases, natural language processing, machine learning, game theory, philosophy of language, cognitive linguistics. The papers presented at this workshop provide a significant selection of approaches and methods to achieve meaning coordination and negotiation, and at the same time illustrate how challenging such a problem is and how stimulating from a research point of view.

Background information

MCN-04 was held in conjunction with the 3rd International Semantic Web Conference (ISWC-04), Hiroshima (Japan), 7–11 November 2004. It follows the workshop on Meaning Negotiation (MeaN-02)[1], held in conjuction with the Eighteenth National Conference on Artificial Intelligence (AAAI-02), Edmonton, Alberta, Canada (July 28 – August1, 2002).

Workshop committee

Paolo Bouquet	University of Trento (Italy)	General Chair
Luciano Serafini	ITC-IRST (Italy)	Co-Chair
Ludger van Elst	DFKI (Germany)	
Nicola Guarino	CNR Lab. of Applied Ontology (Italy)	
Ramanathan V. Guha	IBM Research (USA)	
Yiannis Kompatsiaris	ITI (Greece)	
Stefano Zanobini	University of Trento (Italy)	Publicity Chair

Sponsors

The workshop was sponsored by the EU Network of Excellence *Knowledge Web*[2] and by the EU integrated project *VIKEF*[3].

[1] `http://dit.unitn.it/ bouquet/AAAI-02-MN/`
[2] `http://kw.dia.fi.upm.es/`
[3] `http://www.vikef.net/`

Table of Contents

Semantic coordination: a new approximation method and its application in the music domain

Zharko Aleksovski[1, 2], Warner ten Kate[1], Frank van Harmelen[2]

1, Philips Research, Eindhoven
2, Vrije Universiteit, Amsterdam
July 2004

zharko@cs.vu.nl

Abstract: We address the problem of semantic coordination, namely finding an agreement between the meanings of heterogeneous semantic models. We propose a new approximation method to discover and assess the "strength" (preciseness) of semantic mappings between concepts from different concept hierarchies. We apply this method in the music domain. We present the results of preliminary tests on mapping two music concept hierarchies from actual sites on the Internet.

1. Introduction

The progress of information technology has made it possible to store and access large amounts of data. However, since people think in different ways and use different terminologies to store information, it becomes hard to search each other's data stores. With the advent of the Internet, which has enabled the integrated access of an ever-increasing number of such data stores, the problem becomes even more serious. The music domain is no exception. (We restrict to legal distribution.) The variety and size of offered content makes it difficult to find music of interest. It is often cumbersome to retrieve even a known piece of music.

Our ultimate goal is to improve this Internet music search. We aim to use semantics in the retrieval process, which is conveyed in the Semantic Web. In this context we study the problem of semantic integration over different music provider's schemas. More specific, the problem is to find pairs of concepts (genres, styles, classes...) from different metadata schemas that have an equivalent meaning. It is not sufficient to use the concept labels only, since, for example, their position in the schemas influences their meaning as well. Figure 1 illustrates with an example from existing music schemas. Although the labels are equivalent ("Experimental"), they represent different classes.

The problem of finding the right music that fits a user's preferences is similar to the problem of matching the schemas of two different providers. In the latter case we need to find the pairs of concepts that have equivalent meaning. In the first case, we

Submitted to the ISWC-04 Workshop on "Meaning Coordination and Negotiation", Hiroshima, Japan 2004

can treat the user's preferences as concepts themselves, and the problem is to match the preferred concept with those in the provider's terminology.

Being able to search for matches at the level of concepts (without using instances) is important. The search may use instances (artists, releases, tracks), but when it comes to user preferences we think that we cannot rely on instances solely, in contrast to the recommender systems, which appeared to rely on instances only [Dublin].

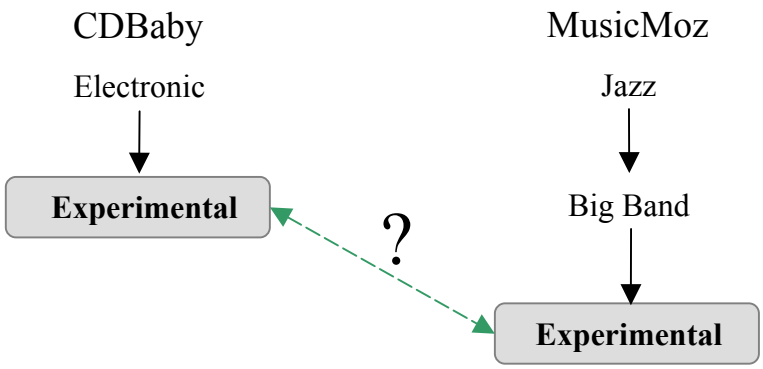


Figure 1 Two music genres. Although the labels are equivalent, they represent different classes.

In this paper we address the problem of matching between two different music metadata schemas. We base ourselves on the approach proposed by Bouquet *et al.* (2003), see the next section. Our main contributions are the following:

- We propose a new method to approximate mappings between concepts from two different Concept Hierarchies. Given two concepts from different Concept Hierarchies, our method checks whether the first concept is a subconcept of the second [Trento], but in addition, when that's not the case, it calculates "how strongly" the first concept is subconcept of the second. This is indicated by a value that we call sloppiness between 0.0 and 1.0 for each pair of concepts. The sloppiness indicates the error in the subsumption relation between the two concepts. Closer to 0.0 means that most of the (semantic) content of the first concept is also present in the second concept, while values closer to 1.0 indicate that there is no subsumption relation.

- We present first results from an analysis of the approximation method. We conducted experiments on actual schemas as used by music sites on the Internet. We extracted the music metadata schemas, which were underlying the navigation paths at the provider sites. We applied our approximation method on those schemas and compared them with the matches based on the actual instances (music artists) in the classes. We discuss the problems we encountered in applying our method and the level of correspondence observed between concepts and instances.

In Section 2 we discuss the approach of Bouquet *et al.* (2003) [Trento], with limitation to our scope. In the section 3 we discuss the present situation with the music metadata schema on the Internet. In section 4 we introduce and explain our idea of approximate matching. In section 5 we present results from applying our

approximate method. In section 6 we identify the future possible improvements on this work. Section 7 gives brief conclusion of the work presented in this paper.

2. Semantic coordination

We have taken the approach of Bouquet *et al.* (2003) [Trento]. We briefly summarize their approach as relevant to our contribution.

The goal is to find mappings between the concepts of two *Concept Hierarchies*. For the current discussion a **Concept Hierarchy** can be thought of as a rooted tree where each node and each edge has a label. It has the explicit purpose to provide an object classification.

Next to the label of the nodes, the method accounts for the position of the nodes in the hierarchy. The method works in three main phases:

- *Linguistic interpretation:* The senses that WordNet [Wordnet] returns on the node's label are combined as propositional terms in a logical formula. The formula represents all the possible linguistic interpretations of the label.

- *Contextualization:* The position of the nodes in the hierarchy is encoded in the logical formula. Each node's formula is considered in conjunction with its ancestor's formula, i.e. each node is assumed to be in the intersection with its ancestor. This makes sense because we expect a superclass to contain everything that the subclasses contain. In a similar way the disjointness of siblings in the hierarchy is encoded into the formula.

- *Semantic comparison:* The so-obtained formulas from both hierarchies are evaluated for relationships. This is done by pair-wise encoding of the formulas in the relationship to be tested. And by testing the satisfiability of the obtained logical relation. Five relationships are considered: (i) they are equivalent, (ii) they are disjoint, (iii) they are not in a subconcept relation but have a non-empty intersection, (iv) the first is a subclass of the second and (v) the second is a subclass of the first.

For more details see Bouquet *et al.* (2003) [Trento].

3. Internet Music Schemas

On the Internet, music metadata schemas mostly exist in the form of a navigation path through the music offered. A metadata schema isn't always offered next to the music, but a visitor can interactively navigate through different pages that list the music offered. We consider this structure of navigation paths together with the labeling on the links and pages as the metadata schema of that provider.

After considering several music provision sites, we selected seven of them and extracted the schema (navigation path): CDNOW (Amazon.com) [CDNow], MusicMoz [MM], CD Baby [CDBaby], ARTIST direct Network [ADN], allmusic

[AMG], LAUNCH cast on Yahoo [Yahoo] and ArtistGigs.com [ArtistGigs]. Commercial providers often include classes whose meaning lies outside the music styles domain (Example: "Music Accessories"). Most of the schemas have some peculiarities that are typical or unique for that schema. After the extraction we applied some simplifying modifications to the data. In the first place, we normalized the labels in order to make the data more suitable for our experiments. This included the correction of typing mistakes and the removal of abbreviations and similar peculiarities.

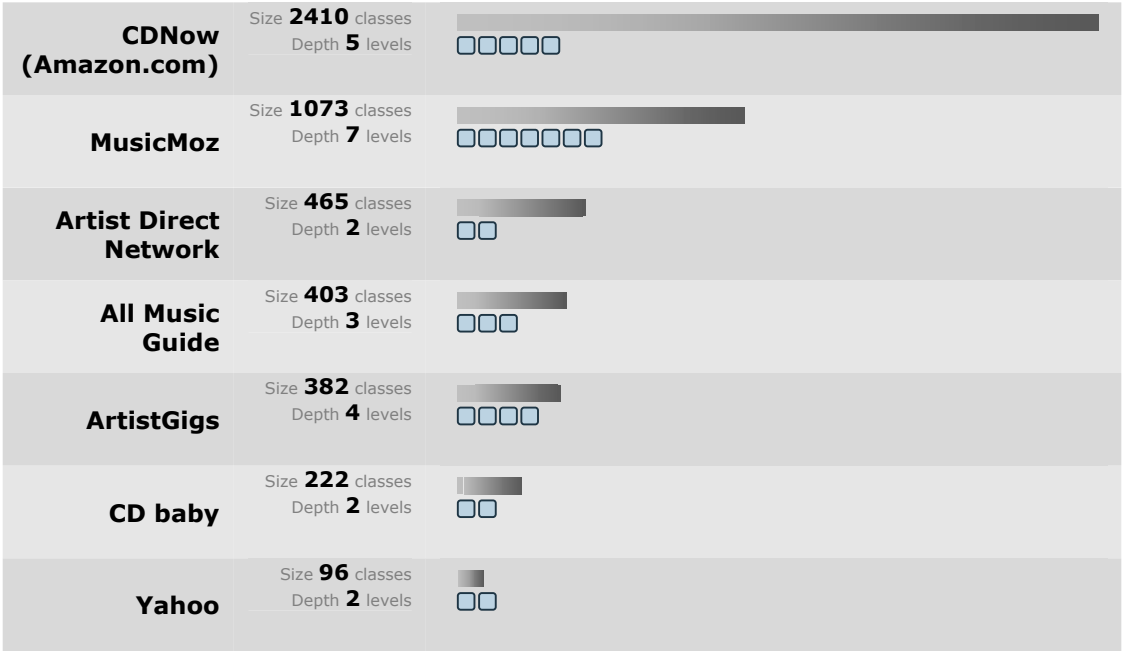


Figure 2 The extracted schemas.

In Figure 2 we present a general overview of the data.

Discussion on the criteria used to create the classes in the schemas

Most of the labels in the ontologies appeared to be of one of the following kinds: *style of music* (the genre of the music), *geographic region with music style* (region where the music originates from) and *time or historical period when the music was created* (decades like "90's", named periods like "baroque"…). All of the schemas are concept hierarchies that only used the subconcept relationship. Sibling subclasses often have overlap (they are not disjoint).

The nodes are often named with more then one word. These words either denote the intersection of the terms they express (example: Chicago Blues), or they are a multiword (example: "New Zealand Rock", New Zealand is a term and should not be considered as separate words). The first case happens most often.

Fuzziness in music classification

There are no objective criteria that sharply define music classes. Genre is not precisely defined. As a result, different providers often classify the same music entities (artists, albums, songs...) differently. Widely used terms like Pop and Rock do not denote the same sets of artists at different portals. That is also the case for even more specific styles of music like Speed Metal.

In our experiments when testing with instance data, we restricted to the artists shared by Music Moz and Artist Direct Network, i.e. artists that are present and classified in both portals. In the sequel we refer to them as MM and ADN, respectively. From the class named *Rock* (including its subclasses) in MM there are 471 shared classified artists, in ADN there are 245, and 196 shared artists are classified under *Rock* in both of them. Hence, from all the artists classified under Rock in at least one of the two portals, only about 38% is classified under Rock in both portals. This example shows that there is a high degree of fuzziness present in the music domain. Therefore we expect that exact reasoning methods to create matching are not useful, and approximate methods would be more useful.

4. Approximate matching

We follow the approach of semantic coordination to find subclass relations between music classes. In this section we explain how we extend the approach with a form of approximation to handle the impreciseness occurring in the actual data. First the (propositional) formulas representing the classes in the Concept Hierarchies are rewritten in normal forms.

We want to check whether a left-hand formula is a subclass of a right-hand formula. The left-hand formula is transformed into *disjunctive normal form* and the right-hand side into *conjunctive normal form*. In this way, the subclass check can be split into a set of subproblems, each checking if one (left) disjunct is a subclass of a (right) conjunct. If all the subproblems are satisfied, the original problem is satisfied. In our approximation, we allow a few of the subproblems to be unsatisfiable, while still declaring the original problem satisfiable. The (relative) number of satisfiable subproblems is a measure of **how strongly** the subclass relation between the two given formulas hold.

Below, we explain the approach in a more formal way. In our notion we confuse logical notion and interpretation: disjunct (logical or) to union, conjunct by intersection, and implication by subset.

Normal Forms

Given the two propositional logic formulas A and B, the problem is to check whether the relation

$$A \subseteq B \qquad \textbf{(1)}$$

holds. We transform A into *disjunctive normal form* and B into *conjunctive normal form*.

The ***Disjunctive Normal Form*** **(DNF)** has the following form:

$$A = \left(A_1^1 \cap A_1^2 \cap \ldots \cap A_1^{n_1}\right) \cup \left(A_2^1 \cap A_2^2 \cap \ldots \cap A_2^{n_2}\right) \cup \ldots \cup \left(A_I^1 \cap A_I^2 \cap \ldots \cap A_I^{n_I}\right)$$

where each A_i^n is an atomic concept. Briefly it can be written as $A = A_1 \cup A_2 \cup \ldots \cup A_I$ where $A_i = \left(A_i^1 \cap A_i^2 \cap \ldots \cap A_i^{n_i}\right)$ for each i from 1 to I. The short form A_i is called a *disjunct*.

The ***Conjunctive Normal Form*** **(CNF)** has the following form:

$$B = \left(B_1^1 \cup B_1^2 \cup \ldots \cup B_1^{m_1}\right) \cap \left(B_2^1 \cup B_2^2 \cup \ldots \cup B_2^{m_2}\right) \cap \ldots \cap \left(B_J^1 \cup B_J^2 \cup \ldots \cup B_J^{m_{Jk}}\right)$$

where each B_i^m is an atomic concept Briefly it can be written as $B = B_1 \bigcap B_2 \bigcap ... \bigcap B_J$ where $B_j = \left(B_j^1 \cup B_j^2 \cup ... \cup B_j^{m_j}\right)$ for each j from 1 to J. The short form B_j is called a *conjunct*.

Now, the problem to check whether $A \subseteq B$ can be written as

$$A_1 \bigcup A_2 \bigcup ... \bigcup A_I \subseteq B_1 \bigcap B_2 \bigcap ... \bigcap B_J$$

This relation holds if and only if (iff) anything that belongs to some of the A_i disjuncts on the left side also belongs to all of the B_j conjuncts on the right side. Written formally it looks like this:

$$A_1 \bigcup A_2 \bigcup ... \bigcup A_I \subseteq B_1 \bigcap B_2 \bigcap ... \bigcap B_J \text{ iff } \left(\forall i = 1..I, j = 1..J\right)\left(A_i \subseteq B_j\right).$$

Now the problem to check whether $A \subseteq B$ is transformed into IxJ subproblems:

$$\left(\forall i, j\right)\left(A_i \subseteq B_j\right) \qquad \textbf{(2)}$$

Now we introduce the **idea of approximation**: The relation **(1)** holds iff for all disjunct-conjunct pairs the subclass relation **(2)** holds. If for only a few of the subproblems the relation **(2)** doesn't hold, we may say that the relation **(1)** $A \subseteq B$ *almost holds*. Even more, we can express the strength at which the relation **(1)** holds as the ratio between the number of false disjunct-conjunct pairs (pairs that do not satisfy the subclass relation) and the total number of pairs. We call this ratio the **sloppiness** and we use the letter s to denote its value. Formally:

$$s\left(A \subseteq B\right) = \frac{\left|\left\{\left(i, j\right) : A_i \not\subset B_j\right\}\right|}{I \times J}$$

Here $\left|\left\{\left(i, j\right) : A_i \not\subset B_j\right\}\right|$ denotes the number of disjunct-conjunct pairs that do not satisfy the subclass relation, I is the number of disjuncts in the DNF form of the formula A and J is the number of conjuncts in the CNF form of the formula B.

Note that this method works on the concepts level and can be applied when no information about the instances is available.

5. Experiment: Approximate matching on ADN and MM

5.1 Approximate matching

In this section we summarize the results from some preliminary experiments we conducted on the approximate matching method. We used the metadata schema as extracted from ADN and MM.

The linguistic interpretation (building of the formulas from the labels of the nodes) was done using simple techniques. For example, "Alternative Rock" was transformed into the following formula:

$$\left(Alternative \cap Rock\right) \cup Alternative\_Rock$$

Special characters "&" and "/" were treated as logical union, For example: "Pop & Rock" was transformed into the following formula:

$$Pop \cup Rock$$

No background knowledge was used. When using background knowledge, each of the atomic concepts (*Alternative*, *Rock*, *Alternative_Rock*) should be replaced with union of the different senses for that concept.

We made the assumption that concepts with the same label have the same meaning. When comparing the disjunct-conjunct relations we simplify a little: we considered a disjunct to be a subclass of a conjunct when at least one part in the disjunct (which is an intersection of literals) is found in the conjunct (which is an union of literals). So, given a disjunct-conjunct pair:

$$A_i = \left(A_i^1 \cap A_i^2 \cap ... \cap A_i^{n_i}\right),\ B_j = \left(B_j^1 \cup B_j^2 \cup ... \cup B_j^{m_j}\right)$$

we say that $A_i \subseteq B_j$ if the class names of A_i^n and B_j^m are equal for some n and m. If none such pair is found, it was considered that the disjunct A_i is not a subclass of conjunct B_j. In principal, this simplification may reject correct subclass relationships, however.

Now we explain the process of approximate inferring an equivalence relation in detail. For the sake of the explanation we have chosen an example that produces simple formulas, however in the reality these can grow bigger and complex.

In our example, we considered the relation between two styles from ADN and MM that are named "Glam Rock" on both of the portals. See Figure 3.

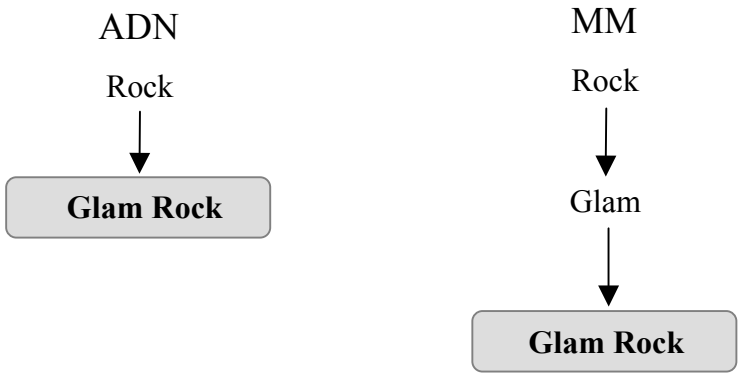


Figure 3 *Glam Rock* style from the schemas of ADN and MM

The first step is to transform the concepts into formulas. We first transform the "Glam Rock" style from ADN. First thing to consider is that "Glam Rock" is a substyle of "Rock". Next, as "Glam Rock" consists of two words, we have to account for the separate meanings of the two member part words (that is the intersection of their meanings), as well as considering it as a multiword (as the case of "New Zealand"). Therefore, the formula representing the meaning of "Glam Rock" from ADN is the following:

Glam_Rock_A = **Rock** ∩ ((**Glam** ∩ **Rock**) ∪ **Glam_Rock**)

The normal forms of the formula are the following:

Glam_Rock_DNF_A = (**Glam** ∩ **Rock**) ∪ (**Glam_Rock** ∩ **Rock**) **(3)**

Glam_Rock_CNF_A = (**Rock**) ∩ (**Glam** ∪ **Glam_Rock**) **(4)**

Following the same way of construction, the "Glam Rock" style from MM is transformed into the formula:

Glam_Rock_B =

Rock ∩ **Glam** ∩ ((**Glam** ∩ **Rock**) ∪ **Glam_Rock**) =

Rock ∩ **Glam**

The literal Glam_Rock in the formula is discarded because of the absorption.
The normal forms of the formula are the following:

Glam_Rock_DNF_B = (**Glam** ∩ **Rock**) **(5)**

Glam_Rock_CNF_B = (**Rock**) ∩ (**Glam**) **(6)**

Now, we use these formulas to test for the equivalence relation between the two concepts Glam_Rock_A and Glam_Rock_B. Therefore we have to check whether the subsumption relation holds between them in both directions.

In the case of the subsumption Glam_Rock_B ⊆ Glam_Rock_A we need the formulas **(4)** and **(5)**. Glam_Rock_B consists of only one disjunct, and Glam_Rock_A consists of two conjuncts. So we have to check for two pairs, namely:

(**Glam** ∩ **Rock**) ⊆ (**Rock**) - *is true* (Rock is on both sides)

(**Glam** ∩ **Rock**) ⊆ (**Glam** ∪ **Glam_Rock**) - *is true* (Glam is on both sides)

Both pairs satisfy the relation, so Glam_Rock_B ⊆ Glam_Rock_A holds with sloppiness 0%.

In the case of the subsumption Glam_Rock_A ⊆ Glam_Rock_B we need the formulas **(3)** and **(6)**. Glam_Rock_A consists of two disjunct, and Glam_Rock_B consists of two conjuncts. So we have to check for four pairs, namely:

(**Glam** ∩ **Rock**) ⊆ (**Rock**) - *true* (Rock is on both sides)

(**Glam** ∩ **Rock**) ⊆ (**Glam**) - *true* (Glam is on both sides)

(**Glam_Rock** ∩ **Rock**) ⊆ (**Rock**) - *true* (Rock is on both sides)

(**Glam_Rock** ∩ **Rock**) ⊄ (**Glam**) - ***false***

Three out of four disjunct-conjunct pairs satisfy the relation, but one, the last pair does not. That makes 25% of the pairs wrong, and therefore the relation Glam_Rock_A ⊆ Glam_Rock_B holds with sloppiness 25%.

When assessing the sloppiness in the equivalence relation between Glam_Rock_A and Glam_Rock_B, we take the maximum of the sloppiness calculated in the two subsumptions. That makes the equivalence relation between Glam_Rock_A and Glam_Rock_B inferred with sloppiness 25%.

5.2 Comparison with instance data

For our experiments we extracted real data from the Internet, see section 3. In the sequel, we present some results that we obtained on these data sets: MM and ADN, Figure 4.

Most of the shared classified artists are classified under “Rock”-related classes (Alternative Rock, Glam Rock, Heavy Metal...). A significant limitation of our dataset is that the number of instances is of the same order as the number of classes.

Name	Number of classes	Number of artists	Number of classified artists	Number of shared classified artists
ArtistDirectNetwork	465	16072	16072	**1183**
MusicMoz	1073	6415	2356	

Figure 4 Size of the data in ArtistDirectNetwork and MusicMoz

We tested for equivalence matchings between the classes in both hierarchies, i.e. whether each is a subclass of the other. We varied the sloppiness measure in the tests. In order to assess the success of the matching we introduce a value called *significance*, which we defined as the cardinality ratio between the intersection and the union of the two classes. Here, by class we mean set of artists. Formally:

$$significance.(A \Leftrightarrow B) = \frac{|A \cap B|}{|A \cup B|}$$

The significance is close to 0 when the two classes have no big overlap, i.e. a relatively small amount of the instances belong to their intersection. When the value is close to 1.0 (or 100%) then the two classes denote almost the same set of instances.

Figure 5 shows the results. We distinguished four different intervals for sloppiness values, as shown on the left column. In the right column is the average of the significance of these equivalences, inferred with sloppiness value from the interval denoted in the left column.

Sloppiness	Average significance
0%-30%	**29.055**%
30%-45%	**26.778**%
45%-55%	**11.439**%
55%-100%	less then **6.7**%

Figure 5 Equivalence testing between ADN and MM

We considered only the results from the equivalences where both of the classes had at least 10 instances. The relations inferred on sloppiness less then 30% are mostly "true" matches, i.e. matches that one would expect to be found with a sloppiness of 0%. However, it appeared that most of them remained undiscovered when the sloppiness was set to 0%. In the case of Figure 3 the equivalence relation was discovered with a sloppiness of 25%.

The relatively low value of the average significance revealed in the performed tests, is a notification that people do not agree on the meaning of the music styles names. While given the same names, the actual songs are stored in different ways.

Figure 6 shows the number of equivalence relations inferred, in dependence of the value for the sloppiness parameter. As the sloppiness is increased the number of inferences increases very slowly in the beginning. That is reasonable since a relatively small amount of pairs of classes from different sources should be considered as equivalent or "almost" equivalent. In general most of the pairs of classes are not related at all, and sloppiness should not change that. From 50% toward the end, the number of inferences increases more rapidly. At the end there is a "cliff" at the sloppiness of 100%, because on sloppiness 100% all classes are equivalent. The statistic of Figure 6 gives confidence in the method.

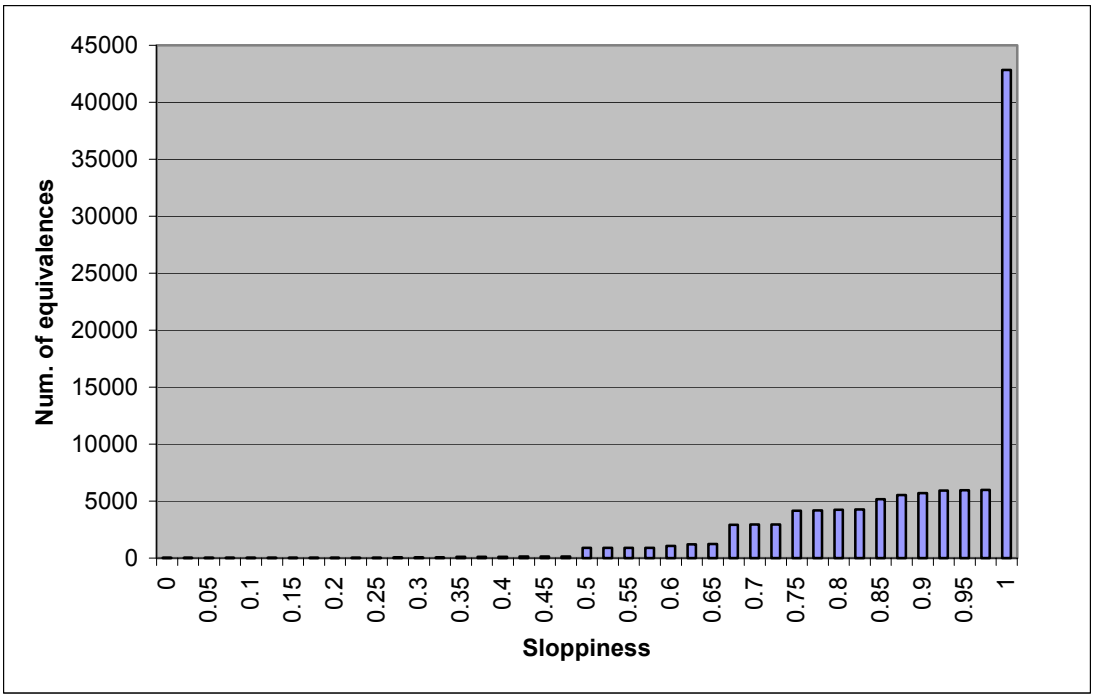


Figure 6 Number of equivalence relations inferred between ADN and MM using different sloppiness parameter.

6. Future work

This general scheme of approximation can be improved in several directions. For example, not all disjunct-conjunct pairs are equally important in their contribution to the tested formulas. Disjuncts and conjuncts can have a different size, i.e. the number of literals they consist of. Literals may also have different size when it comes to the sets of instances they denote. Accounting for these differences may result in a more accurate sloppiness measure.

We expect that a bigger size of the disjuncts and conjuncts should have less impact on the result. The intuition is that more parts in the disjunct make it a class of smaller scope, since it is an intersection, so its contribution to the total class (disjunction) is smaller. A similar intuition holds for the conjuncts, more parts in the conjunct make it a class of bigger scope since it is a union.

A final heuristic is to assign more weight to "rare" classes. The intuition is that the more general concept names have wider use, and therefore matching them only provides general confidence. When matching "Rock" between two classes, or when matching "Cajun", it is natural to expect a stronger relation in the second case.

7. Conclusion

In this paper we have presented a new method to do approximate matching between concept hierarchies. We have discussed the situation and the present problems in the music artist classifications on the Internet. Fuzziness is highly present and the

approximate matching method that we proposed may help to deal with this problem. First results from applying the approximate matching method in the music domain were presented.

This is a preliminary work; we still need to do thorough testing and to implement the improvements of this method. We also need to test against other approximate matching methods.

Acknowledgements

Thanks to Heiner Stuckenschmidt for his useful feedback and the fruitful discussions we had on the early version of the paper.

References

[Trento]
P. Bouquet, L. Serafini, and S. Zanobini,
Semantic Coordination: A New Approach and an Application,
Proc. ISWC2003, Springer, LNCS 2870, 130-145, 2003.

[Wordnet]
G.A. Miller et al.,
WordNet - A Lexical Database for the English Language,
Communications of the ACM, November 1995 / Vol. 38, No. 11
"http://www.cogsci.princeton.edu/~wn/".

[Mendelson]
E. Mendelson: *Introduction to Mathematical Logic*,
Chapman & Hall/CRC, 1997.

[Dublin]
C. Hayes, P. Cunningham: *Context Boosting Collaborative Recommendations*
Trinity College, Dublin, 2003

J. Aucouturier, F. Pachet: *Representing Musical Genre: A State of the Art*
Journal of New Music Research, © Swets & Zeitlinger Vol. 32, No. 1, pp 83-93, 2003

F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schneider (editors),
The Description Logic Handbook,
Cambridge University Press, 2003.

[CDNow]
Online CD store.
http://www.cdnow.com/

[ADN]
MP3 download, music chat, online CD store, videos.
http://artistdirect.com/

[MM]
A directory of all music things.
http://musicmoz.org/

[CDBaby]
A CD store for new independent music.
http://www.cdbaby.com/

[Yahoo]
Broadcasting Internet radio stations.
http://launch.yahoo.com/

[AMG]
Comprehensive music resources.
http://www.allmusic.com/

[ArtistGigs]
ArtistGigs.com, Independent Musical Artists.
http://www.artistgigs.com/

Soundness of Semantic Methods for Schema Matching*

M. Benerecetti[1], P. Bouquet[2], S. Zanobini[2]

[1] Department of Physical Science – University of Naples – Federico II
Via Cintia, Complesso Monte S. Angelo, I-80126 Napoli (Italy)
[2]Department of Information and Communication Technology – University of Trento
Via Sommarive, 10 – 38050 Trento (Italy)
bene@na.infn.it, bouquet@dit.unitn.it, zanobini@dit.unitn.it

Abstract. One of the key challenges in the development of open semantic-based systems is enabling the exchange of meaningful information across applications which may use autonomously developed schemata. Semantic coordination is the problem of discovering mappings across schemata and schema matching is one of the proposed approaches. In this paper we provide a preliminary investigation on the notion of correctness of semantic methods for schema matching. We define a first notion of semantic soundness (and completeness), but immediately show that this notion is not appropriate to capture the intuitive notion of correctness for a method. We then introduce the idea of pragmatic soundness, and argue that it corresponds to what we intuitively expect, but that it can't be directly computed. Finally, we discuss some preliminary conditions under which a semantically sound method can guarantee pragmatic soundness as well, which is – in our opinion – the best we can get from semantic methods.

1 Introduction

One of the key challenges in the development of open semantic-based systems is enabling the exchange of meaningful information across applications which may use autonomously developed schemata (database schemata, classifications, even directory trees on file systems in peer-to-peer applications) for organizing locally available data. As in open system a beforehand agreement on the meaning of schemata seems impossible in practice, a large number of methods and systems have been proposed to automatically match schemata[1]. The resulting mappings are then used as the basis for a runtime semantic-based coordination of such a network of autonomous applications.

Methods may differ along many dimensions: the type of structures to which they can be applied (e.g., trees, directed acyclic graphs, graphs); the type of result they return (e.g., similarity measures, model-theoretic relations, fuzzy relations); the resources they use to compute such a relation (e.g. external lexical resources, ontologies, string manipulators, graph matching techniques, instance-based techniques). In this paper, for

* The work presented in this paper was done as part of the EU funded project VIKEF (Virtual Information and Knowledge Environment Framework), contract n. 507173.

[1] A very partial list includes [14, 13, 11, 10, 4, 6, 9, 2, 5, 8, 3]. A detailed description of these methods is out of the scope of this paper.

reasons that will be explained in detail, we are mostly concerned with a class of methods that we call *semantic methods*. The general intuition underlying semantic methods is that they aim at discovering relations between (pairs of) entities belonging to different schemata *based on the meaning of the two entities*. However, beyond this point, there is a significant disagreement on what characterizes a semantic method from a non semantic method. For example, a recent paper by Giunchiglia and Schvaiko [7] proposes to include among semantic methods only those methods that directly return a semantic relation (e.g., material implication or logical equivalence), namely a relation with a well-defined model-theoretic interpretation. This analysis is far from being shared in the community, as other people feel that a method is semantic if it uses semantic information to return its results, or if there is a principled way to assign an indirect semantics to its results (e.g., mapping numerical values on semantic relations through the definition of suitable thresholds).

In such a situation, it is not surprising that we still lack a clear definition of the conditions under which a semantic method can be said to work "correctly". Suppose, for example, that we have a method α that takes in input two nodes n_A and n_B from two schemata S_A and S_B respectively and returns `True` if the two nodes represent equivalent concepts, `False` otherwise. Now, imagine that α is fed with the categories `/IMAGES/TUSCANY/FLORENCE` and `/PHOTOS/ITALY/FLORENCE`[2] belonging to two classification schemata, and that it returns `True`. Is the result "correct"? Why? And what if the result were `False`? Under what conditions would we accept this result as "correct"?

This paper aims at answering this kind of questions. We start by providing a precise characterization of schema matching for a special (but interesting) case of schemata, namely hierarchical classifications. Then we propose a characterization of semantic-based methods based on the idea that they must at least provide an explicit and formal interpretation of the entities they compare, and of the resulting relation. Finally, for this class of methods, we define the notions of semantic soundness and completeness, but immediately show that this notion is not appropriate to capture the intuitive notion of correctness for a method. We then introduce the idea of pragmatic soundness, and argue that it corresponds to what we intuitively expect, but that it can't be directly computed. Finally, we discuss some preliminary conditions under which a semantically sound method can guarantee pragmatic soundness as well, which is – in our opinion – the best we can get from a semantic method for schema matching.

2 The problem of schema matching

Schema is a broad term, that applies to different kinds of structures. In [3], it was argued that it makes no much sense to speak about schema matching in general, and that the analysis should be done case by case along the dimension of the intended use of a schema. Accordingly, in this paper we restrict our attention to a special kind of schemata, *hierarchical classifications*, whose explicit purpose is to classify objects

[2] Throughout the paper we will use the notation `X/.../Y` to refer to a path in schema in analogy with the notation for paths in a file system. If the schema is a tree, then `/` represent the root of the schema and `/X/.../Y` a path from the root to `Y` through `X`.

(e.g., documents). This restriction does not affect the generality of our investigation, as the method of analysis can be applied to study the problem of matching other types of schema, such as database schemata, service descriptions, datatypes.

We start with a few definitions that characterize the kind of schemata we deal with, namely topic hierarchies used as classification schemata.

Definition 1 (Topic hierarchy). *Let Λ be a set of labels (e.g., words in natural language). A* topic hierarchy *$\mathcal{S} = \langle K, E, l \rangle$ is a triple where K is a finite set of nodes, E is a set of arcs on K, such that $\langle K, E \rangle$ is a rooted tree, and l is a function from K to Λ.*

Two simple examples of topic hierarchies are depicted in Figure 1.

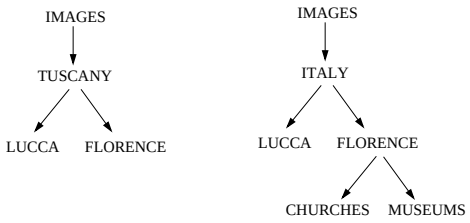


Fig. 1. Two simple topic hierarchies

A possible use for topic hierarchies is to classify documents. To express this formally, we introduce the notion of classification function.

Definition 2 (Classification function). *Let D be a set of documents and $\mathcal{S}$ a topic hierarchy $\langle K, E, l \rangle$. A classification function over $\mathcal{S}$ is a function $\tau : D \rightarrow K$ from documents to nodes of $\mathcal{S}$.*

A classification function places a document under a node in a topic hierarchy. We associate to each classification function a *retrieval function*, which is a function from nodes to the sets of documents attached to them in a topic hierarchy. It essentially plays the inverse rôle of the classification function.

Definition 3 (Retrieval function). *Let D be a set of documents, $\mathcal{S} = \langle K, E, l \rangle$ a topic hierarchy, and τ a classification function over $\mathcal{S}$. The* retrieval function *of τ over $\mathcal{S}$ is a function $\mu_\tau : K \rightarrow 2^D$ satisfying the following condition:*

$$\textit{for every } d \in D,\ d \in \mu_\tau(\tau(d))$$

Finally, we want to formalize the intuition of a classification being a hierarchical classification (hereafter HC). Intuitively, it must satisfy the following requirement: documents classified under a node `Z` along a path `X/.../Y/Z` could be also classified under the ancestor nodes of the same path (though with a lower degree of precision) if `Z` were removed from the topic hierarchy:

Definition 4 (Hierarchical classification). *Given a set of documents D, a* hierarchical classifi cation $\mathcal{H} = \langle \mathcal{S}, \tau \rangle$ *is a pair where $\mathcal{S}$ is a topic hierarchy and τ is a classification function over $\mathcal{S}$ which satisfies the following property: if τ classifies a document d under a node Z along a path X/.../Y/Z, and we remove Z from the path, then τ would assign d to the node Y of the path X/.../Y.*

Schema matching can be defi ned as the problem of computing relations between pairs of nodes belonging to different HCs. Let $\Re$ be a set of relations that may hold between two nodes belonging to two distinct schemata $\mathcal{S}_A$ and $\mathcal{S}_B$. Then a mapping is defi ned as follows:

Definition 5 (Mapping). *A* mapping $\mathcal{M}_{A \to B}$ *between two HCs $\mathcal{H}_A = \langle \mathcal{S}_A, \tau_A \rangle$ and $\mathcal{H}_B = \langle \mathcal{S}_B, \tau_B \rangle$ is a set of triples $\langle n_A, n_B, r \rangle$, where:*

- *n_A and n_B are two nodes belonging to $\mathcal{S}_A$ and $\mathcal{S}_B$, respectively;*
- *$r \in \Re$ is a relation between n_A and n_B.*

Each triple $\langle n_A, n_B, r \rangle$ belonging to a mapping is called a *mapping element*.

Finally, as our goal is to discuss properties of schema matching methods, we formally defi ne a method as a function which returns true when a given relation holds between two elements of different schemata, false otherwise:

Definition 6 (Schema Matching Method). *Let $\mathcal{M}_{A \to B}$ be a mapping between two HCs $\mathcal{H}_A$ and $\mathcal{H}_B$. A* schema matching method $\alpha : \mathcal{M}_{A \to B} \to \{T, F\}$ *is a function from mapping elements to boolean values.*

Of course, it is more natural to view a method as a function which takes two nodes as input and returns a relation as output. Here we adopt this more abstract (but after all equivalent) characterization as it is more appropriate for our analysis.

3 Semantic methods for schema matching

In the previous section, we deliberately left the defi nition of schema matching methods quite vague, as we wanted to characterize the problem of schema matching in a very general form. Here we provide a precise characterization of semantic methods in a precise way through two general principles that, in our opinion, distinguish semantic methods from non semantic methods.

A method for schema matching is a semantic methods if it satisfi es the two following principles:

Explicit representation of meaning: a semantic method must match schema elements on the basis of an explicit representation of their meaning, where meaning is a formal object of a logical type which corresponds to the type intended by the schema designer. Notationally, if n is a node of a HC, then $\mathcal{R}(n)$ is the formal representation of its meaning;

Computation of relations based on meaning: given two nodes n and m belonging to different HCs, a semantic method must return a relation which connects the meanings of the schema elements under comparison. Such a relation must in turn have an interpretation defi ned over the meaning of the compared elements.

So, according to the first principle, a semantic method should explicitly interpret the elements of a HC as concepts, and provide a corresponding formal representation of type concept (e.g., as terms in some Description Logic system [1]). For example, the meaning of the nodes FLORENCE belonging to the right hand side schema and CHURCHES belonging to the left hand side of schema of Figure 1 approximately corresponds to the two concepts "Images of Florence in Tuscany" and "Images of churches in Florence, in Italy". Notice that a schema describing how a web service works (basically, a finite state automaton) should be interpreted in a completely different way, as nodes would represent states that can be reached through actions associated to arcs.

Then, according to the second principle, when matching HCs, a semantic method should return a relation between concepts (e.g., subsumption, equivalence, and so on). Notice that here we will privilege classical model-theoretic relations, though it is possible to work with fuzzy-theoretic relations between concepts (see e.g. the common framework for ontology alignment produced as part of the EU network of excellence called Knowledge Web). Going back to the example of Figure 1, the relation between the two nodes FLORENCE and CHURCHES (interpreted as we said above) is that the first is more general than the second. We note that, in this case, determining the relation between the two concepts intuitively requires to use knowledge that was not extracted from the two schemata, namely that Tuscany is in Italy. In the following, we will refer to this external knowledge as the *ontology* associated to a method. In analogy to what we said above, a relation between elements of two service description schemata would be completely different.

We are now ready to start our discussion about soundness of semantic methods.

4 Semantic soundness and completeness

Given the two principles discussed above, a semantic method α is defined by: (i) a language $\mathcal{L}$ suitable to explicitly represent the meaning of each schema element, (ii) a procedure for extracting the meaning of each element n ($\mathcal{R}(n)$), (iii) a (possibly empty) ontology $\mathcal{O}$, and (iv) a set of relations $\Re$ that it can compute. The 4-tuple $\langle \mathcal{L}, \mathcal{O}, \mathcal{R}(), \Re \rangle$ is what we call the *semantic frame* of the method.

We now propose a notion of semantic soundness and completeness with respect to a semantic frame F. The intuition is the following: a method is *semantically sound* w.r.t. F if, whenever it computes a relation between two elements of distinct schemata, the relation follows from what the method knows about the meaning associated to the two elements; and is *semantically complete* if, whenever one of the relations in $\Re$ between the meaning of two nodes follows from what the method knows, then the method actually returns that relation. More formally:

Definition 7 (Semantic Soundness). *Let $F = \langle \mathcal{L}, \mathcal{O}, \mathcal{R}(), \Re \rangle$ be the semantic frame of a method α and $\mathcal{H}_A$ and $\mathcal{H}_B$ be two HCs. Then α is* semantically sound *w.r.t. F if and only if for any mapping element $\langle n_A, n_B, r \rangle$ the following holds:*

$$\textit{if } \alpha(\langle n_A, n_B, r \rangle) = T, \textit{ then } \mathcal{O} \models_{\mathcal{L}} \mathcal{R}(n_A)\ r\ \mathcal{R}(n_B)$$

Definition 8 (Semantic Completeness). *Let $F = \langle \mathcal{L}, \mathcal{O}, \mathcal{R}(), \Re \rangle$ be the semantic frame of a method α and $\mathcal{H}_A$ and $\mathcal{H}_B$ be two HCs. Then α is* semantically complete *w.r.t. F if and only if for any two nodes n_A and n_b the following holds:*

$$\textit{if } \mathcal{O} \models_{\mathcal{L}} \mathcal{R}(n_A)\ r\ \mathcal{R}(n_B),\ \textit{then } \alpha(\langle n_A, n_B, r \rangle) = T$$

Though this notion of semantic soundness and completeness seems reasonable, it should be quite evident that it does not capture the intuition of a correct method. Indeed, what we want to say is that a method is sound when it computes the "right" relation between two elements, namely the relation that follows from the "correct" interpretation of the schemata and from the use of the "right" background knowledge. Instead, what the definitions above says is only that, given an ontology and a formal representation of the meaning of two nodes, then a semantic method is sound if and only if it derives only relations that logically follows from the background knowledge provided by its ontology. But this is tantamount as saying that a semantic method is sound if and only if the reasoner used to compute the relation between meanings is sound, which is a very trivial result (similarly for completeness). Indeed, imagine a dummy method that associate the same concept k to all the elements of two HCs, and always returns the equivalence relation for any pair of nodes (for all $k \in \mathcal{S}$ and $k' \in \mathcal{S}'$, $\alpha(k, k', \equiv) = T$). Since any concept is always equivalent to itself, then this method is semantically sound. But is this method of any interest?

Intuitively, the problem is that semantic soundness as we defined it (and a similar argument can be done for completeness) does not say anything on the appropriateness of the meaning explicitation performed by the method and on the relation between the meaning of nodes and the available ontology. In short, semantic soundness is a necessary but not sufficient condition to capture the intuitions we have about the correctness of a method. What we need as a sufficient condition is a way for excluding dummy methods like the one described above, namely methods that build arbitrary interpretations and use pertinent knowledge about the meaning of schema elements.

However, this is an extremely tough problem not only in schema matching, but in general for any semantic theory based on formal logic. Indeed, as we know from classical results (see e.g. the model-theoretic argument discussed by the philosopher H. Putnam in [12]), there's nothing we can do to prevent unintended interpretations of a formal language. The form in which Putnam discusses this problem is the following: even if two agents agreed on the truth value of all the sentences of a language L (including modal propositions on the necessity of propositions), this would not be sufficient to fix the interpretations of the terms they use, which means that they may still be talking about different things. Our version of this argument is that, even if we can guarantee that a method is semantically sound and complete, there's nothing which guarantees that the two elements were correctly interpreted, and therefore that the relation between the two nodes is the one we would expect.

To get around this problem, there are basically two approaches available for semantic methods. The first one, which we will call the *linguistic approach*, is to exploit the fact that almost invariably the labels of schemata are meaningful expressions of natural language, e.g. English. If this is the case, then not every interpretation is acceptable, though ambiguity is still possible. For example, the word "bank" can mean "depository

financial institution" or "the slope beside a body of water" (Wordnet 1.7), but cannot mean "animal with four legs", unless we relax the assumptions that labels are taken from English[3]. The second approach, which we call *instance-based approach*, is to exploit the data attached to a schema (e.g., the documents attached to the categories of a HC) to guess or refine the interpretation of the category itself. The idea is that we can determine the meaning of an element in a schema by processing the documents associated to that element, e.g. by analyzing the number of times a word occur in a document, or the co-occurrence of words in the same document. Both approaches, however, have their drawbacks: the linguistic approach suffers from the "ambiguity problem", namely there is still the possibility that we haven't caught the intended interpretation of a schema element (as natural languages are ambiguous in different respects); instance-based methods suffers from the "contingency problem", namely the actual set of documents attached to an element may be insufficient to capture the intended concept (let alone the fact that there may be schemata not populated with documents at all).

To sum up, linguistic and instance-based approaches improve the situation of semantic methods, but do not provide the sufficient conditions we are looking for. Is there another way of defining the correctness of a method which does not refer to the explicitation of meanings? In the next section we propose a possible answer.

5 Pragmatic Soundness

The main reason why people develop schemata is to provide a suitable organization of a body of relevant data, e.g. records in a database, file in a file system, documents in a classification schema. Schema matching methods should allow applications to exchange data in a meaningful way through the exploitation of mappings across schemata. For example, when we match two HCs, the goal is to find mappings that allow us to retrieve documents on a given topic which are classified under (possibly different) categories in different HCs. From this perspective, if a semantically sound method derives that two nodes are equivalent, then one would expect that a user would classify the same documents under those two nodes; if the methods derives that a node is more general than another one, then one would expect that the documents classifiable under the first node are a superset of those classifiable under the second; and so on. If this holds, then the method is intuitively correct, as it "does the right thing" for its users. This notion of correctness, based on data rather than on the meaning of schema elements, is pragmatic, as it refers to how people use schemata, and not (directly) to how they interpret it. Let us try to make this intuition more precise.

Preliminary, we introduce the notation to refer to all documents classified in a subtree of a topic hierarchy, and not only in a single node. The reason is the following. Consider the two HCs of Figure 1. Note that the node `FLORENCE` in the right hand side topic hierarchy has two children (`CHURCHES` and `MUSEUMS`), whereas the left one has no children. This means that the documents that in the right hand side topic hierarchy are classified under `CHURCHES`, would be probably classified under the node `FLORENCE` in the left hand side topic hierarchy. Therefore, we introduce the notation

[3] This is the basis, for example, of what in [3] is called *semantic coordination.*

$\mu_\tau(n\downarrow)$ to denote the set of documents classified under a subtree rooted at the node n. More formally, let $n\downarrow = \{k \in K \mid k \text{ is a descendant of } n\}$ denote the set of nodes in the subtree rooted at n, then $\mu_\tau(n\downarrow) = \bigcup_{m \in n\downarrow} \mu_\tau(m)$.

Let D be a set of documents and $\Re$ a set of relations between sets of documents (for example, $\Re = \{=, \subseteq, \supseteq, \perp\}$, where $\perp$ means disjoint). Furthermore, imagine that someone classifies all documents of D in two different HCs ($\mathcal{H}_A$ and $\mathcal{H}_B$). Then a first tentative definition of pragmatic soundness is the following:

Definition 9 (Strong Pragmatic Soundness). *Let $\mathcal{H}_A$ and $\mathcal{H}_B$ be two HCs and α a semantic method. Then α is* strongly pragmatically sound *if for any mapping element $\langle n_A, n_B, r\rangle$ (with $r \in \Re$) the following holds:*

$$\textit{if } \alpha(\langle n_A, n_B, r\rangle) = T \quad \textit{then} \quad \mu_\tau(n_A\downarrow)\ r\ \mu_\tau(n_B\downarrow)$$

Intuitively, this means that if a semantic method α discovers a relation r between two nodes n_A and n_B, then the corresponding set-theoretic relation r also holds between the sets of documents classified by the function τ in the subtree rooted at the nodes n_A and n_B[4].

However, this definition presupposes two very strong assumptions. First, D must be the set of all possible documents of the universe; otherwise, it may happen that an actual set of documents is not sufficient to discriminate between some set-theoretical relations, such as $\subset$ and $=$ (it may happen that no document which would be associated to n_A but not to n_B belongs to D, and therefore the two sets would be contingently the same, whereas they would not if we had had more documents available).

But even more important, it presupposes that each document can be classified in a unique way. Of course, this is not the case in general, as documents are typically *rich objects*, and can be classified under different categories, depending on what aspect of the document is taken as dominant. For example, this paper could be classified under different categories (e.g. `SEMANTIC INTEROPERABILITY`, `ONTOLOGY INTEGRATION`, `SCHEMA MATCHING`, `FORMAL MODELS`, ...), and each of these categories would reflect a legitimate point of view on the paper. Therefore, even if two categories in two different HCs – populated by the same classifier – are semantically related, we can't guarantee that the sets of documents classified under those two categories will be in the same relation.

The considerations above suggest that we need a weaker notion of pragmatic soundness, which can take into account the possibility that a classifier (human or automatic) can legitimately classify the same document under different categories. In this situation, the question arises of whether there can be a reasonable notion of correctness. Intuitively, we propose a *counterfactual* notion of correctness: a method is correct if a classifier would not disagree with the answers produced by the method; in other words

[4] To keep the formalism simple, we are abusing our notation by using the symbol r to refer both to the (semantic) relation computed by a semantic method and the relation which holds between sets of documents. In fact, we rely on the intuitive mapping between semantic relations (say, subsumption between concepts) and set-theoretic relations between their interpretation (for subsumption, it would be set inclusion). To be precise, such a mapping should be explicitly defined.

if, no matter what his/her actual classification is, the classifier could have classified the documents according to the relation discovered by the method.

To capture this intuition, we first introduce the following notion of classifier:

Definition 10 (Classifier). *A* classifier C *is a pair* $\langle\{\tau_i\}, F\rangle$*, where* $\{\tau_i\}$ *is a set of classification functions, and* $F = \langle \mathcal{L}^C, \mathcal{O}^C, \mathcal{R}^C(), \Re\rangle$ *is a semantic frame.*

Associating a set of classification functions to a classifier allows us to capture the fact that s/he can classify the same set of document in different ways. Therefore, when populating a topic hierarchy, we allow classifiers to employ any of their classification functions. Intuitively, the set $\{\tau_i\}$ can be seen as a set of "acceptable" classification functions, in the sense that the classifier will be prepared to accept classifying a document under a given node if there is a classification function belonging to $\{\tau_i\}$ which would classify that document under the same node.

But this is not enough. Indeed, we also expect that there is a rationale behind the classification tasks of any "reasonable" classifier. In other words, we expect that classifiers perform their task based on their knowledge about the documents to be classified and about the available categories. Here is where the ontology $\mathcal{O}^C$ and the interpretation function $\mathcal{R}^C()$ associated to a classifier come into play. Intuitively, the ontology and the interpretation function represents the knowledge classifiers use to understand the meaning of a node and, consequently, for classifying a document. Classifiers are then called *pragmatically adequate* if they act consistently with their knowledge. We capture this by imposing the following condition: when a classifier C recognizes a relation holding between (the meaning of) two nodes n_A and n_B of two HCs $\mathcal{H}_A$ and $\mathcal{H}_B$, then – whatever classification function she is actually using – if this function classifies under n_A and n_B the sets X and Y of documents, then there must be a (possibly distinct) acceptable classification function which would classify under n_A a set X' of documents in such a way that the corresponding set-theoretic relation holds between the sets X' and Y. The following definition formalizes this intuition.

Definition 11 (Compatible classification functions). *Let* C *be a classifier,* τ_1 *and* τ_2 *two classification functions of* C*,* r *any relation in* $\Re$*, and* n_A *in* $\mathcal{H}_A$ *and* n_B *in* $\mathcal{H}_B$ *two nodes. We say that* τ_1 *is* compatible with τ_2 w.r.t. n_A and n_B *if the following holds:*

$$\textit{if } \mathcal{O}^C \models \mathcal{R}^C(n_A)\ r\ \mathcal{R}^C(n_B),\ \textit{then } \mu_{\tau_1}(n_A{\downarrow})\ r\ \mu_{\tau_2}(n_B{\downarrow})$$

Intuitively, two classification functions of a classifier are compatible w.r.t. two nodes if their respective way of classifying documents under the two nodes preserve the relation the classifier recognizes between the meanings of the two nodes. We can now formalize the notion of *pragmatically adequate* classifier.

Definition 12 (Pragmatic adequacy). *Let* $C = \langle\{\tau_i\}, F\rangle$ *be a classifier. Then* C *is* pragmatically adequate *if, given any two nodes* n_A *in* $\mathcal{H}_A$ *and* n_B *in* $\mathcal{H}_B$*, and any classification function* $\tau_1 \in \{\tau_i\}$*, there is another* $\tau_2 \in \{\tau_i\}$ *which is compatible with* τ_1 *w.r.t.* n_A *and* n_B*.*

This definition simply says that if a classifier C associate a set of documents to some node n_A, and n_A is in a certain relation r with a second node n_B, then C must

be prepared, possibly by employing some other acceptable classification function of his (namely, a compatible classification function), to classify under the n_A a set of documents holding the same relation r with the set of documents attached to n_B.

Based on the definitions above, we can now attempt a second definition of *pragmatic soundness* which, we believe, is the best we can expect from a schema matching method. Intuitively, we say that a schema matching method is *pragmatically sound* if whenever it derives a relation r between two nodes n_A and n_B, a pragmatically adequate classifier would consider this result as "acceptable" according to the possible ways he could classify a set of documents. By "acceptable" here we mean that whatever set of documents C has actually placed under n_A and n_B (using one of his classification functions), C could have placed under n_A, using a possibly different admissible classification function, a set of documents in the same relation r with the set of documents actually placed under n_B. This intuition is captured by the following definition:

Definition 13 (Pragmatic Soundness). *Let $C = \langle\{\tau_i\}, F\rangle$ be a pragmatically adequate classifier, and $\mathcal{H}_A$ and $\mathcal{H}_B$ be two HCs. A method α is* pragmatically sound *w.r.t. C if, for any mapping element $\langle n_A, n_B, r\rangle$, the following holds: if $\alpha(\langle n_A, n_B, r\rangle) = T$, then for any classification $\tau_2 \in \{\tau_i\}$ there is a classification $\tau_1 \in \{\tau_i\}$ such that, for any $r \in \Re$, $\mu_{\tau_1}(n_A\downarrow)\ r\ \mu_{\tau_2}(n_B\downarrow)$.*

6 Can semantic methods be pragmatically sound?

Assume now we have a semantically sound matching method α which can answer whether a semantic relation between two nodes holds or not. In this section we try to answer the question of what can condition can guarantee that a sound semantic method α is also pragmatically sound[5]. We can state the following proposition:

Proposition 1. *Let $F = \langle\mathcal{L}, \mathcal{O}, \mathcal{R}(), \Re\rangle$ be a semantic frame, α a method semantically sound w.r.t. F, and $C = \langle\{\tau_i\}, F^C\rangle$ a pragmatically adequate classifier (where $F^C = \langle\mathcal{L}^C, \mathcal{O}^C, \mathcal{R}^C(), \Re\rangle$). If $\mathcal{O}^C \sqsubseteq \mathcal{O}$ and $\mathcal{R}^C(m) = \mathcal{R}(m)$, then α is pragmatically sound. Moreover, if $|\{\tau_i\}| = 1$, then α is strongly pragmatically sound.*

The proposition states that if (i) α is semantically sound, (ii) the ontology used by α is subsumed by a pragmatically competent classifier knowledge (i.e., it is a sound but not necessarily complete representation of the classifier knowledge), and (iii) the meaning assigned to the nodes by $\mathcal{R}^C(m)$ and $\mathcal{R}(m)$ is the same (namely, $\models \mathcal{R}^C(m) \equiv \mathcal{R}(m)$), then α is also pragmatically sound. If, in addition, (iv) the classifier always uses the same classification function, then clearly α is also strongly pragmatically sound.

The first part of the proposition immediately follows from Definitions 7, 12 and 13. The second part descends from Definitions 7, 12 and 9. A sketch of proof follows. Since any relation between concepts that can be deduced from a less specific ontology ($\mathcal{O}$) can also be deduced by a more specific one ($\mathcal{O}^C$), Condition (i) together with Condition (ii) ensure that any relation discovered by the method α would also be inferred by any

[5] The problem of pragmatic completeness is significantly harder and out of the scope of this paper. We will not discuss it here.

classifier. Moreover, if C is a pragmatically adequate classifier, whatever classification function τ he has used to place documents under node n_A and n_B, by Definition12 there must be another acceptable classification function τ' of C using which C would have placed under n_A a set of documents holding the relation r with those placed by τ under n_B. Hence pragmatic soundness follows. Adding the additional constraint that the classifier only allows for a single classification function, immediately leads to strong pragmatic soundness.

Let us now briefly comment on the conditions we needed to guarantee pragmatic soundness of a semantic matching method. Condition (i) is quite easy to ensure, as we already pointed out in Section 3. A logic framework powerful enough to express the desired semantic relations between the concepts of interest, for which decidability is guaranteed will suffice. Condition (ii) seems to be a relatively weak requirement. This is an important observation, since providing a method with complete knowledge with respect to a classifier is likely to be a very hard task, let alone the problem of providing complete knowledge with respect to *any* classifier. Even though the first two conditions seem to be reasonably easy to satisfy, Condition (iii) turns out to be quite strong. Notice that weakening the condition on the semantic explicitation functions is problematic. Soundness with respect to any set of semantic relations can indeed be ensured only if the matching method and the classifier both assign the same interpretation (in terms of concepts) to all the nodes of the HCs. Nevertheless, soundness can still be retained on some specific sets of semantic relations, depending on the cases. For instance, if we consider the set of relations $\{\sqsubseteq, \bot\}$, then any semantically sound method employing a more specific semantic explicitation function than that of the classifier (namely, $\models \mathcal{R}^C(m) \sqsupseteq \mathcal{R}(m)$) will still be pragmatically sound. Unfortunately, soundness with respect to none of the other relations we have been considering in the paper would be preserved in this case. Similarly, any semantically sound method employing a less specific semantic explicitation function than that of the user (namely, $\models \mathcal{R}(m) \sqsupseteq \mathcal{R}^C(m)$) will still be pragmatically sound only with respect to the relation $\sqsupseteq$.

7 Conclusions

The consequence of Proposition 1 is that semantic methods can be guaranteed to obtain pragmatically correct results under conditions (i)–(iii) (also (iv) if we want strong pragmatic soundness). As condition (i) is quite trivial, we can conclude that the roadmap to correct semantic methods is quite clear: (a) we need to build ontology which reflect the classifier's (or the user's) point of view on the world ($\mathcal{O}^C \sqsubseteq \mathcal{O}$) and (b) we need to design tools that interpret a schema element as the user interprets it. These two problems are not trivial, but they can be addressed with well-known methods belonging to disciplines like ontology engineering and knowledge representation. Ontology engineering can help us to design better ontologies, e.g. ontologies that appropriately represent what am individual or a community knows on a given domain; knowledge representation gives us methods for representing the meaning of different types of schemata, beyond classifications.

To conclude, we see our work as a small step towards a much more general goal, namely the construction of a theory which explains how semantically autonomous en-

tities (agents) can communicate without presupposing a beforehand agreement on how things should be represented. In other words, a theory of the role of meaning coordination in a theory of (inter)action. A lot remains to be done, but this goes beyond the scope of this paper.

References

1. Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook. Theory, Implementation and Applications.* Cambridge University Press, January 2003.
2. Sonia Bergamaschi, Silvana Castano, and Maurizio Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.
3. P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: a new approach and an application. In K. Sycara, editor, *Second International Semantic Web Conference (ISWC-03)*, Lecture Notes in Computer Science (LNCS), Sanibel Island (Florida, USA), October 2003.
4. Jeremy Carroll and Hewlett-Packard. Matching rdf graphs. In *Proc. in the fi rst International Semantic Web Conference - ISWC 2002*, pages 5–15, 2002.
5. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proceedings of WWW-2002, 11th International WWW Conference, Hawaii*, 2002.
6. J. Euzenat and P. Valtchev. An integrativive proximity measure for ontology alignment. *Proceedings of the workshop on Semantic Integration*, October 2003.
7. P. Shvaiko F. Giunchiglia. Semantic matching. *Proceedings of the workshop on Semantic Integration*, October 2003.
8. Ryutaro Ichisem, Hiedeaki Takeda, and Shinichi Honiden. Integrating multiple internet directories by instance–base learning. In *AI AND DATA INTEGRATION*, pages 22–28, 2003.
9. Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *The VLDB Journal*, pages 49–58, 2001.
10. Tova Milo and Sagit Zohar. Using schema matching to simplify heterogeneous data translation. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 122–133, 24–27 1998.
11. Marcello Pelillo, Kaleem Siddiqi, and Steven W. Zucker. Matching hierarchical structures using association graphs. *Lecture Notes in Computer Science*, 1407:3–??, 1998.
12. H. Putnam. *Reason, Truth, and History*. CUP, 1981.
13. Jason Tsong-Li Wang, Kaizhong Zhang, Karpjoo Jeong, and Dennis Shasha. A system for approximate tree matching. *Knowledge and Data Engineering*, 6(4):559–571, 1994.
14. K. Zhang, J. T. L. Wang, and D. Shasha. On the editing distance between undirected acyclic graphs and related problems. In Z. Galil and E. Ukkonen, editors, *Proceedings of the 6th Annual Symposium on Combinatorial Pattern Matching*, volume 937, pages 395–407, Espoo, Finland, 1995. Springer-Verlag, Berlin.

Asking and answering semantic queries*

P. Bouquet, G. Kuper, M. Scoz, S. Zanobini

Department of Information and Communication Technology – University of Trento
Via Sommarive, 10 – 38050 Trento (Italy)
{bouquet, kuper, scoz, zanobini}@dit.unitn.it

Abstract. One of the main issues in the development of the Semantic Web is the design and implementation of query languages that allow users to retrieve information from semantically annotated sources. In this paper, we describe a general methodology for querying a distributed collection of semantically heterogeneous resources, linked to each others through a collection of semantic mappings. The main contribution of this paper is the definition of *semantic query*, namely a query which enables users to tune a collection of semantic parameters to formulate the intended request. We show why this is different from what is typically done in data integration and peer-to-peer query reformulation.

1 Introduction

One of the main issues in the development of the Semantic Web is the design and implementation of query languages that allow users to retrieve information from semantically annotated sources. This problem, and several proposals have been put forward.

This problem has two fundamental dimensions. The first, which we call the *local dimension*, has to do with the problem of querying a single knowledge source (for example, an RDF [12] or an OWL [9] knowledge base) whose structure is known *a priori* and semantic heterogeneity is not a serious issue. A solution to this problem essentially amounts to proposing a query language (or family of languages) that does for Semantic Web languages what SQL does for relational databases. This problem neglects a crucial aspect of the Semantic Web, namely that in most real situations information will be distributed over a collection of distributed resources. This introduces the second dimension of the problem, called the *distributed dimension*, namely the problem of querying a collection of knowledge sources whose structure is not known *a priori* and where the degree of semantic heterogeneity can be quite high. Our work is focused on this second dimension. Relevant work in this area can be divided into two classes: global schema and peer-to-peer approaches. The first includes approaches based on some form of global schema. The idea is that a solution to the distributed dimensions of the querying problem requires the construction of a global schema which is then used to reformulate queries, either in a local as view (LAV) or global as view (GAV) architecture [13]. The second class includes peer-to-peer approaches, namely approaches in which the solution to the query problem is based on "horizontal" mappings across local

* The work presented in this paper was done as part of the EU funded project VIKEF (Virtual Information and Knowledge Environment Framework), contract n. 507173.

schemas. In such a scenario, a query can be thought of as a request formulated on a local schema to find semantically related data/information from a collection of remote schemas.

It was suggested (e.g. in [4]) that the problem of querying distributed and heterogeneous structures on a peer-to-peer basis can be divided into two main sub-problems: (i) the problem of discovering mappings across heterogeneous schemas (the *mapping problem*); and (ii) the problem of using a pre-existing collection of mappings to rewrite/ reformulate queries (the *query rewriting problem*). The idea is that first one needs to discover the (semantic) relation between two or more schemas; mappings are then used to answer queries over heterogeneous schemas, e.g. by reformulating a query written on a local schema into one or more queries on remote schemas.

In this paper, we argue that there is a further level, which we call the problem of *asking and answering semantic queries*. Indeed, the query rewriting problem can be restated as the problem of using semantic information (i.e., the available mappings) to reformulate "syntactic queries", namely queries that dig into the data associated to a schema by exploiting its structural properties (for XML-based languages, an example could be the rewriting of XPath expressions). But, in our view, a query is a semantic query only when the parameters used in its formulation are intrinsically semantic, namely are intended to refine the expression of a user's intended meaning. In other words, a semantic query is one whose result depends on parameters that are semantic in nature. Of course, an important question is what counts as a semantic parameter. In this paper we do not provide a general answer. However, since we will be mainly concerned with the problem of querying heterogeneous classifications, the relevant semantic parameters we will consider are: (i) the *type of relation*; (ii) the *ontological distance*; and (iii) the *lexical distance*.

2 The problem

Imagine that John is trying to find images for a book that he is writing about his holiday in Tuscany. Two web sites (say `PICS1` and `PICS2`) provide multi-media content. Figure 1 depicts a tiny portion of the structures they use to classify images. Suppose now that John is navigating the structure in `PICS1` and is interested in finding more images of Tuscany. John would like to ask something like: "Get me more documents which are related to what on this site is classified under `IMAGES/TUSCANY`".

Following what we said in the introduction, this request can be addressed at three different levels:

1. the first level corresponds to what in the introduction we called the mapping problem. It has to do with the discovery of the semantic relations between the categories of `PICS1` and `PICS2`. In Figure 1 we reported some possible mappings, for example that the category `IMAGES/TUSCANY` in `PICS1` is more specific than `PHOTOS/ITALY` in `PICS2`. At this level, one can say that there are many categories in `PICS2` semantically related to the category `IMAGES/TUSCANY` in `PICS1`, and that there are different possible relations (e.g. more general categories, or equivalent categories);

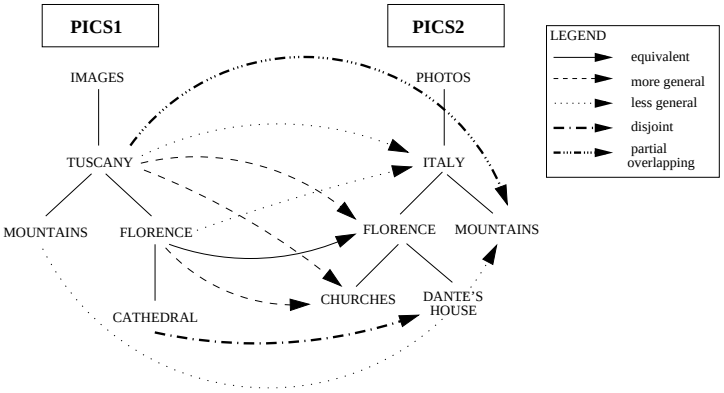


Fig. 1. Semantic mapping across classifications

2. the second level corresponds to the query rewriting problem. It refers to the fact that each semantic relation (i.e., each arrow from `IMAGES/TUSCANY` in `PICS1` to a category in `PICS2`) can be used to rewrite a query like `IMAGES/TUSCANY` on the schema `PICS1` into some query on the schema `PICS2`;
3. the third level corresponds to what we called the semantic query problem. It has to do with the fact that a query like `IMAGES/TUSCANY` on the schema `PICS1` does not provide enough information on what John may have in mind. For example: is he interested only in images which are classified under nodes that are semantically equivalent to the node `TUSCANY`? Is he willing to accept also images from more specific categories? If so, to what extent? Is a photo of Florence acceptable? And a photo of Dante's house in Florence? Adding this information to a standard query (or to any reformulation of a query) is what we define as asking a semantic query, and is the main focus of this paper.

The problem of allowing semantic queries can be divided into two sub-problems, which we call the *How-To-Ask* and the *How-To-Answer* problems respectively.

How-To-ask. The first class of problems is related to the parameters that John should be able to "tune" to specify his request. The parameters we are interested in are semantic parameters, namely parameters that can be used to refine the interpretation of John's request. The three parameters we take into account in this paper are: (i) *type of relation*, which is used to restrict the query to categories which are in a specific semantic relation with the category of the source schema; (ii) the *ontological distance*, which is used to specify the acceptable distance from the category in the source schema and categories in other schemas (with respect to some reference ontology); and (iii) the *lexical distance*, which is used to tune the distance between the linguistic formulation of the category in the source schema and the linguistic formulation of categories in other schemas.

How-To-answer. Once a semantic query is formulated, there is the problem of answering it appropriately. In this paper, we will ignore the details of solving the "structural" part of the query, instead, we will focus on the semantic part, namely the resolution of

semantic constraints specified by users. To do this, we will assume that a collection of semantic relations across the different structures has already been computed by some matching algorithm[1], and show how these mappings can be used to answer a query which specifies the values of these semantic parameters. The "How-To-Answer" part of the problem is non-trivial, as it requires one to take into account the fact that the evaluation of semantic parameters depends on what knowledge is used. For example, two concepts that are ontologically very close for John might be very distant for Mary, in particular if they use different background ontologies to evaluate such a distance. Therefore we need to provide a solution in which it is clear whose knowledge is being used.

Returning to the example, John can ask the following semantic query: "Get me documents classified in categories which are equivalent or more specific than the category `IMAGES/TUSCANY` in `PICS1`, where the ontological distance is less or equal to n and the linguistic distance is unbounded". Intuitively, if we assume that at `PICS2` an ontology is available according to which Dante's house is in Florence, and Florence is in Tuscany, then we can deduce that the node `FLORENCE` in `PICS2` is ontologically less distant from the node `IMAGES/TUSCANY` in `PICS1` than the node `DANTE'S HOUSE`, even though they are both related to the source category by the same semantic relation (i.e. less general).

3 Semantic queries over distributed classifications

Considering a collection $S_1, \ldots, S_n$ of semantically heterogeneous structures and a set of mappings $\mathcal{M}_1, \ldots, \mathcal{M}_j$ across them, a distributed query is a request, posed on one of the structures, to retrieve data from the other structures. Such a query is a semantic query when the answer is based on the satisfaction of a collection of semantic parameters.

In this section we focus in particular on a specific scenario, where $S_1, \ldots, S_n$ are hierarchical classifications, such as Web directories or catalogs. For such an application, we now propose a precise notion of semantic query, in which semantic parameters are explicitly listed, and then define the notion of semantically appropriate answer. In the following section, we show that this notion of semantic query can be easily implemented on top of CTXMATCH, an algorithm presented in [3] which automatically generates mappings across hierarchical classifications.

3.1 Choosing the mapping

As stated in the introduction, the problem of semantic queries is different from the problem of discovering mappings across structures, and that semantic queries use pre-existing mappings. We now discuss what types of mappings are needed to support semantic queries.

Generally speaking, a mapping between two schemas S_1 and S_2 (including taxonomies, ontologies, catalogs) can be thought of as a triple $\langle n_1, n_2, R \rangle$, where n_1 is

[1] Section 4 describes one possible method for computing these relations, based on the work presented in [3].

a node of S_1, n_2 is a node of S_2, and R is a relation between the two nodes. These mappings are calculated in many different ways and the existing approaches can be classified in two main categories, depending on the nature of the relation they compute:

- methods that return numerical values (typically between 0 and 1), whose intended meaning is the semantic proximity between the two nodes. Examples include CUPID [14], MOMIS [1], and GLUE [6];
- methods that return semantic relations, i.e., a relation with a clear model-theoretic interpretation (e.g., logical equivalence or subsumption). Examples include CTX-MATCH [3], S-MATCH [10].

From the perspective of semantic queries, the problem with the first category is that the interpretation of the result is unclear. In fact, for example, could be difficult to interpret in the right way a 0.9 similarity? These questions are important if we want to allow such logical relation between concepts as semantic parameters. For this reason, we assume in this paper that the available mappings are a collection of coordination rules, defined as follows.

Definition 1 (Coordination rule). *A* coordination rule *from a structure $\mathcal{S}_A$ to a $\mathcal{S}_B$ is a quadruple $\langle id, m, n, r \rangle$, where:*

- *id is a unique identifier for the rule;*
- *$m \in \mathcal{S}_A$ and $n \in \mathcal{S}_B$ are nodes in the corresponding structures;*
- *r is the semantic relation holding between m and n.*

In [3], it is argued that, when the structures are classifications, the following set $\Re$ of semantic relations must be considered: $\equiv$ (equivalence), $\sqsubset$ (the first is strictly less general than the second), $\sqsupset$ (the first is strictly more general than the second), $*$ (partial overlapping), $\perp$ (exclusion). Relations are interpreted in terms of documents that would be classified under the two categories. Given a collection of documents D, $\equiv$ means that the same subset of D would be classified under the two categories, $\sqsubset$ means that all documents classified under the category in the source structure would be classified also under the category of the target structure (and similarly for $\sqsupset$), $*$ means that there is a possible intersection between the sets of documents classified under the two categories, $\perp$ means that no document can be classified under both categories.

A mapping is defined as a set of coordination rules:

Definition 2 (Mapping). *A* mapping *$\mathcal{M}_{A \to B}$ between two structures $\mathcal{S}_A$ and $\mathcal{S}_B$ is a pair $\langle id, \mathcal{CR} \rangle$, where id is a unique identifier for the mapping and $\mathcal{CR}$ is a set of coordination rules from nodes of $\mathcal{S}_A$ to nodes of $\mathcal{S}_B$.*

3.2 Choosing the relevant semantic parameters

We now discuss the types of parameters we consider. They may depend on the types of structures that are queried, or on the specific application. We propose a list of semantic parameters that, in our opinion, are among the most important for the specification of semantic queries.

In [3], it is argued that computing semantic mappings across hierarchical classifications depends on three different types of knowledge:

Ontological Knowledge. Ontological Knowledge ($\mathcal{O}$) represents what is known about a given domain, or about the world in general.[2]. Intuitively, $\mathcal{O}$ can be thought of as the set of 'objects', or concepts, that an agent has knowledge about together with some relations among them. Facts in the $\mathcal{O}$ used in our example include the fact that Florence is located in Tuscany, that Tuscany is part of Italy, that Italy is in Europe, and that Europe is a continent.

Lexical knowledge. Lexical knowledge represents knowledge about the relationship between the concepts of an ontology $\mathcal{O}$ and their encoding into the language that is d used to communicate with other agents. One of the best-known instances of lexical knowledge is WORDNET [7], but note that WORDNET also includes part of what we call ontological knowledge.

Structural knowledge. Structural knowledge refers to the fact that a classification typically classifies documents under categories that correspond to concepts which are not directly defined in the ontology, but are obtained from the "composition" of concepts defined in one or more ontologies. For example, the category 'photos of Italian mountains' from the schema `PICS2` in Figure 1 is obtained by combining the concepts photo, Italy, and mountains. This knowledge is called structural, as it is used to build the structure of the classification.

If we assume that a mapping is used as a way of rewriting queries for different schemas, then the parameters associated to a semantic query should be used to filtering out some of these rewritings. To make this possible, we introduce parameters that are related to the way that each of the three kinds of knowledge described above are used to compute each coordination rule in a mapping. The result is the following list of parameters:

1. Ontological distance. This parameter encodes the 'ontological effort' that is required for determining the semantic relation between two concepts. As an example, we show that the ontological distance between `PHOTOS/ITALY/FLORENCE` in `PICS2` and `IMAGES/TUSCANY` in `PICS1` is smaller that the ontological distance between `PHOTOS/ITALY/FLORENCE/DANTE'S HOUSE` in `PICS2` and `IMAGES/TUSCANY` in `PICS1`. Both pairs of nodes are connected via two coordination rules which contain the same relation ($\supset$). However, the computation of the second rule requires the use of more ontological knowledge, as it depends on at least two facts: that Florence is in Tuscany, and that Dante's house is in Florence. The computation of the first rule depends only on the first fact. This observation can be used to conclude that, given an ontology which contains these two facts, the derivation of the second coordination rule requires a greater "ontological effort" and therefore that the ontological distance is higher[3].

2. Lexical distance. This parameter represents the 'lexical effort' needed to determine if two words denote the same concept. The prototypical example of lexical effort is the substitution of a word with a synonym. However, other (not strictly semantic) techniques can be used to force words occurring in different schemas to refer to the same

[2] Here, we use the word 'ontology' in the broad sense of an explicit and formal conceptualization of the world. Indeed, at this level, we do not need to distinguish between types of ontologies, e.g. top level ontologies, domain ontologies, application ontologies, etc.

[3] We would like to stress the fact that the ontological distance does not express a structural distance between nodes, but only refers to how far a relation is from another w.r.t. the ontology.

concept. These include string manipulation, lemmatizers, *ad hoc* thesauri, etc. Lexical distance allows us to say, for example, that the concept `IMAGES/ITALY` is closer to `IMAGES/TUSCANY` in `PICS1` than to the concept `PHOTOS/ITALY` in `PICS2`. Indeed, even though one may argue that `IMAGES/ITALY` and `PHOTOS/ITALY` are the same concept, the computation of the coordination rule which determines their semantic relation with `IMAGES/TUSCANY` in `PICS1` requires a greater lexical effort, namely the use of the piece of lexical knowledge saying that, at least in one possible sense, the word 'PHOTO' and the work 'IMAGES' are synonyms. In this paper, we shall only consider only synonymy, in which case the lexical distance is a Boolean parameter with values 0 or 1. In general, however, it could be used as a real distance, with sophisticated techniques to introduce finer grained measures, where the similarity between words could be expressed as a real number between 0 and 1.

3. Type of relation. Each coordination rule represents a semantic relation between two complex concepts, i.e., concepts that are built from simple concepts defined in some ontology and organized in a classification structure. As stated above, there is may be more than one possible relation between two such concepts. This parameter is used to select the relation of interest for a given query. For example, it allows John to say that he wants only images that are classified under categories that are equivalent to the category `IMAGES/TUSCANY` in `PICS1`; in our example this would return the empty set.

3.3 Semantic queries

We can now define formally the notion of a semantic query.

Definition 3 (Semantic Query). *A semantic query $\mathcal{Q}$ is a 5-tuple $\langle \mathcal{S}, m, r_{\mathcal{M}}, \Delta_o, \Delta_l \rangle$, where:*

- *$\mathcal{S}$ is a structure;*
- *m is a node in $\mathcal{S}$;*
- *$r_{\mathcal{M}} \in \Re$ is a semantic relation;*
- *Δ_o is the ontological distance;*
- *Δ_l is the lexical distance;*

A *semantically appropriate answer* to a semantic query $\mathcal{Q}$ can be defined as follows:

Definition 4 (Semantically Appropriate Answer). *Let $\mathcal{M}$ be a mapping between a source structure $\mathcal{S}_A$ and a target structure $\mathcal{S}_B$, and let $\mathcal{Q}$ be a query. The* semantically appropriate answer *to $\mathcal{Q}$ is the set of nodes $n \in \mathcal{S}_B$ such that n is related to m through the mapping $r_{\mathcal{M}}$, i.e., $\langle id, m, n, r_{\mathcal{M}} \rangle \in \mathcal{M}$, for some values of id. Furthermore, n must be at the appropriate ontological and lexical distance from m.*

4 An example

We illustrate our general framework by showing how a semantic query engine can implemented on top of CTXMATCH, the algorithm for discovering semantic mappings across heterogeneous structures described in [3].

The input to CTXMATCH consists of two structures, and the result is a mapping between them. This mapping is computed in two main steps: (i) semantic explicitation, in which the meaning of each node of the two structures is made explicit and is encoded as a set of logical formulas; and (ii) semantic comparison, in which the problem of discovering the semantic relation between two nodes is now encoded as a relatively simple problem of logical deduction. Then, determining whether there is an equivalence relation between two nodes becomes a problem of testing whether the formulas associated to two nodes are logically equivalent, w.r.t. the appropriate axioms. Consequently it's used a standard SAT solver to checks the relations. Table 1 summarizes the satisfiability problems associated to each relation, where ϕ represents the meaning associated to a node in the source structure and ψ represents the meaning associated to a node in the target structure. The tests are performed in the order listed in this table, and the relation that is returned corresponds to the first positive answer[4].

	TEST	RELATION RETURNED
1	$\Theta \models \neg(\phi \wedge \psi)$	$\perp$
2	$\Theta \models \phi \equiv \psi$	$\equiv$
3	$\Theta \models \phi \rightarrow \psi$	$\subset$
4	$\Theta \models \psi \rightarrow \phi$	$\supset$
5	default	$*$

Table 1. Set of SAT tests

In the current version of CTXMATCH, ontological knowledge $\mathcal{O}$ is represented as a directed acyclic graph where nodes represent concepts and arcs represent roles.

Definition 5 (Ontological Knowledge). *Let C be a set of concepts, and R a set of roles.* Ontological knowledge *(denoted by $\mathcal{O}$) is a quadruple $\langle N, E, l, l' \rangle$ where N is a finite set of nodes, $E \subseteq N \times N$ is the set of arcs on N, $l : N \rightarrow C$ is a bijective function from the set N of nodes to the set C of concepts and $l' : E \rightarrow R$ is a function from the set E of arcs to the set R of roles.*

Lexical knowledge is a function which assigns sets of concepts to each word of a lexicon[5], where a lexicon is the set of words that are used to describe the concepts. Formally, let C be a set of concepts and L a set of lemmas. Then lexical knowledge is defined as follows:

[4] Note that the relation returned by tests 3 and 4 is strict containment, since the system performs test 3 only if a negative result was returned by test 2. If a positive answer is returned by 3, it means that $\Theta \models \phi \rightarrow \psi \wedge \neg(\phi \equiv \psi)$, which corresponds to the '$\subset$' relation. A similar explanation applies to the $*$ relation, which the default case.

[5] We allow sets of concepts in the lexical function since in most human languages the same lemma can express more that one concept (polysemy). In an ideal language, where no polysemy is possible, the sets L and C would be isomorphic and the lexicon function would be a bijective function.

Definition 6 (Lexical knowledge). Lexical knowledge *is a function* $\mathcal{L} : L \rightarrow 2^C$ *from lemmas to sets of concepts.*

The coordination rules returned by CTXMATCH already contain information about the semantic relations between pairs of nodes of two classifications. However, to implement the mechanism of semantic query on top of CTXMATCH, we also need to compute the lexical and ontological distance associated to each rule. In this paper we decide to precompute the values when creating the mapping, adding this information to the existing coordination rules [6].

The modified version of CTXMATCH therefore returns *extended coordination rules*:

Definition 7 (Extended Coordination Rule). *An* Extended Coordination Rule *from a structure* $\mathcal{S}_A$ *to a structure* $\mathcal{S}_B$ *is a 6-tuple* $\langle id, m, n, r, d, ld \rangle$*, where:*

- id, $m \in \mathcal{S}_A$, $n \in \mathcal{S}_B$ *and* r *are as in Definition 1;*
- d *is the ontological distance of the coordination rule;*
- ld *is the lexical distance of the coordination rule.*

Accordingly, we extend the definition of mapping as follows:

Definition 8 (Extended Mapping). *A* extended mapping $\mathcal{M}_{A \rightarrow B}$ *between two structures* $\mathcal{S}_A$ *and* $\mathcal{S}_B$ *is a set of Extended Coordination Rules.*

We now discuss how CTXMATCH actually computes the ontological and lexical distance above.

Let $\mathcal{O}$ be an ontology as defined in Definition 5, and let c and c' be two concepts in $\mathcal{O}$. We say that two concepts c and c' are *related* iff there is at least one path on the graph that connects the corresponding nodes $l^{-1}(c)$ and $l^{-1}(c')$. The ontological distance between c and c' is then defined as follows.

Definition 9 (Ontological Distance between simple concepts). *The* Ontological Distance *between* c *and* c'*, written* $D_s(c, c')$*, is the length of the minimal path between the nodes corresponding to* c *and* c' *in* $\mathcal{O}$*, if such a path exists, and is* 0 *otherwise.*

For example, if 'Florence $\xrightarrow{\text{Part-Of}}$ Tuscany $\xrightarrow{\text{Part-Of}}$ Italy' is the minimal path in $\mathcal{O}$ between the simple concepts 'Italy' and 'Florence', then the ontological distance D_s(Italy, Florence) is 2 (two arcs).

However, in general, we are interested in calculating the distance between two complex concepts. To define this distance, we introduce the following definitions.

Definition 10 (Ontological Distance between sets of simple concepts). *Let* A *and* B *be two sets of simple concepts. The* ontological distance *between the sets* A *and* B*,* $D_c(A, B)$*, is*

$$\Sigma_{c \in A,\, c' \in B} D(c, c')$$

[6] This fact does not increase the complexity of CTXMATCH.

The ontological distance between sets of simple concepts is the sum of the ontological distances of all the possible pairs of simple concepts in the two sets.

This definition involves some redundancy, and we therefore introduce the notion of normalized set of simple concepts.

Definition 11 (Normalized set of simple concepts). *Let K be the set of simple concepts occurring in a complex concept α. A* normalized set of simple concepts $K' \subseteq K = \{c \in K \mid$ *there is no path from c' to c in $\mathcal{O}$ for some $c' \in K\}$.*

For example, $K = \{\mathrm{Photos}, \mathrm{Italy}, \mathrm{Florence}\}$ is the set of simple concepts associated to the complex concept 'Images of Florence in Italy'. Then the normalized set K' is $\{\mathrm{Images}, \mathrm{Florence}\}$, as the presence of the *Part-Of* relation between 'Florence' and 'Italy' in the ontology $\mathcal{O}$ allows us to delete 'Italy' from the set.

The ontological distance between complex concepts can now be defined as follows.

Definition 12 (Ontological Distance between complex concepts). *Let A and B be the set of simple concepts occurring in complex concepts ϕ and ψ respectively. The* Ontological Distance between the complex concepts *ϕ and ψ is defined as $D_c(A', B')$, where A' and B' are the normalized sets of simple concepts for A and B respectively.*

For lexical distance, we introduce the notion of translation clause as follows.

Definition 13 (Translation clause). *Let k be a node in a structure $\mathcal{S}$ and ϕ the associated complex concept. Then a* translation clause *C_j for ϕ is a set of pairs $\langle w, \mathcal{L}(w)\rangle$, where w is a word occurring in one of the labels of the nodes lying in the path from root to k, and $\mathcal{L}(w) = \langle s_1, \ldots, s_n\rangle$ is the set of possible concepts denoted by the word w w.r.t. a lexicon $\mathcal{L}$.*

Consider the node TUSCANY of the right structure depicted in Figure 1. The translation clause for this node w.r.t. WORDNET (used as lexicon) is the set

$$\{\langle image, \langle image\#1, \ldots, image\#7\rangle\rangle, \langle Tuscany, \langle tuscany\#1\rangle\rangle\}$$

Definition 14 (Lexical Distance between complex concepts). *Let ϕ and ψ complex concepts and C_s and C_t be the 'translation clauses' for ϕ and ψ respectively. The* Lexical Distance *between ϕ and ψ is 1 if $C_t \subseteq C_s$, and is 0 otherwise.*

In this framework, the definition of *semantically appropriate answer* can be restated as follows:

Definition 15 (Semantically Appropriate Answer Specialized). *Let $\mathcal{M}$ be an extended mapping between a source structure $\mathcal{S}_A$ and a target structure $\mathcal{S}_B$, and let $\mathcal{Q}$ be a query. The* semantically appropriate answer *to $\mathcal{Q}$ is the set of nodes $n \in \mathcal{S}_B$ such that n is related to m through the mapping $r_{\mathcal{M}}$, i.e., $\langle id, m, n, r_{\mathcal{M}}, d, ld\rangle \in \mathcal{M}$, for some values of id. Furthermore, n is at the appropriate lexical distance, i.e. $ld \leq \Delta_l$, and ontological distance, i.e. $d \leq \Delta_o$, from m.*

We illustrate this definition with a simple example. We use a syntax is based on XPath, extended to allow the specification of semantic parameters. The notation for the semantic parameters is similar to XPath qualifiers, but using angle brackets to make the distinction clear. We can therefore qualify a node by using: (i) $\langle rel = r \rangle$, where $r \in \Re$ (i.e., $\equiv$, $\subset$, etc.); (ii) $\langle od \leq k \rangle$, to restrict the ontological distance to be less that of equal to k (strict equality can also be used); (iii) $\langle ld = 0 \rangle$ to restrict the lexical distance ($ld = 1$ could be used, but would be redundant).

Consider Figure 1. The query '$/\texttt{IMAGES}/\texttt{TUSCANY}\langle r_{\mathcal{M}} = \supset \rangle \langle \Delta_o \leq 1 \rangle \langle \Delta_l = 0 \rangle'$, expressed using a XPath expression on the source HC (on the left), specifies that the user wants documents that are contained in nodes that are semantically related to the path `IMAGES/TUSCANY` by means of a semantic relation $\supset$, but that are ontologically distant up to 1, and whose concepts are lexically equivalent.

Considering the mapping, we know that the path `IMAGES/TUSCANY` is related, in some way, to six elements. The first constraint is represented by the restriction on the semantic relation '$\supset$'. We can see that only the paths `PHOTOS/ITALY/FLORENCE/CHURCHES` and `PHOTOS/ITALY/FLORENCE/DANTE'S HOUSE` satisfy this constraint. The second constraint is that on the ontological distance, which is satisfied only be `PHOTOS/ITALY/FLORENCE/CHURCHES` respects the constraint. Indeed, the path `PHOTOS/ITALY/FLORENCE/CHURCHES` is at ontological distance 2 from `IMAGES/TUSCANY`, since 'Dante's house *Part-Of* Florence', and 'Florence *Part-Of* Tuscany'. The third constraint is that on the lexical distance. This is satisfied by none of the paths, as, the lexical distance between `IMAGES/TUSCANY` and `PHOTOS/ITALY/FLORENCE/CHURCHES` is 1, since the translation clause $\langle image, \langle s_1, \ldots, s_n \rangle \rangle$ is in C_t but not in C_s.

5 Related Work

We present here roadmap to relevant work in the area of querying the Semantic Web.

We first distinguish approaches that address the problem of querying a single knowledge source from those that address the issue of distributing queries across multiple (heterogeneous) sources. In the first group we find RQL [12, 11], XML-QL [5], and XQuery [2]). We also include DQL [8] and OWL-QL [9], even though they are designed for a distributed environment. Indeed, they address the problem of a client-server interaction, but, to the best of our knowledge, not the problem of querying semantically heterogeneous resources.

In the other group, there are two main approaches, those that use a global schema (GAV and LAV), and and P2P approaches. Among the P2P approaches, we classify the relevant work according to the three levels discussed in the introduction: the mapping level, the query rewriting level, and the semantic query level.

We have explained why the mapping problem is different from the problem of using mappings to answer queries. The approach that is closest to ours is that of [4]. In this paper, the two levels are (i) a particular kind of mapping between the exported fragments V_l and V_r of a knowledge base from the local and the remote peers, and (ii) the query problem, regarded as the problem of rewriting a query on the local structure into another query on the remote structure using the ontological knowledge encoded in the mapping.

These two levels are different from our *mapping* and the *semantic query* levels, and both of the problems of [4] are addressed at the mapping level. A set of semantic relations relating a path m in a structure $\mathcal{S}_A$ with a set of paths $n_1, \ldots, n_k$ in another structure $\mathcal{S}_B$ represents the set of all the possible rewritings in $\mathcal{S}_B$ of a query q on m, so that the 'syntactical rewriting' of the query q would be redundant. Furthermore, our semantic query level adds a further level called the *asking and answering problem*.

References

1. Sonia Bergamaschi, Silvana Castano, and Maurizio Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.
2. S. Boag, D. Chamberlin, M. Fernandez, D. Florescu, J. Robie, and J. Simeon. XQuery 1.0: An XML query language. Technical report, W3C, November 2003.
3. P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: a new approach and an application. In K. Sycara, editor, *Second International Semantic Web Conference (ISWC-03)*, Lecture Notes in Computer Science, Sanibel Island (Florida, USA), October 2003.
4. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. What to ask to a peer: ontology-based query reformulation. In *9th International Conference on Principles of Knowledge Representation and Reasoning (KR-2004)*, 2004.
5. A. Deutsch, M. Fernandez, A. Levy, and D. Suciu. XML-QL: A query language for XML. Technical report, W3C, August 1998.
6. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *11th Int. WWW Conf., Hawaii*, 2002.
7. Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database.* The MIT Press, Cambridge, US, 1998.
8. R. Fikes, P. Hayes, and I. Horrocks. DQL - a query language for the semantic web. Technical report, Knowledge Systems Laboratory, 2002.
9. R. Fikes, P. Hayes, and I. Horrocks. OWL-QL - a language for deductive query answering on the semantic web. Technical report, Knowledge Systems Laboratory, Stanford, CA, 2003.
10. F. Giunchiglia and P. Shvaiko. Semantic matching. *Proceedings of the workshop on Semantic Integration*, October 2003.
11. G. Karvounarakis. The RDF query language (RQL). Technical report, Institute of Computer Science, Foundation of Research Technology, 2003.
12. G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, and M. Scholl. RQL: A declarative query language for RDF. In *Proc. of the eleventh international world wide web conference*, Honolulu, Hawaii, USA, May 2002.
13. A. Calì, D. Calvanese, G. De Giacomo, and M. Lenzerini, Data integration under integrity constraints, In *Information Systems*, pages 147–163, 2004.
14. Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *The VLDB Journal*, pages 49–58, 2001.

Element Level Semantic Matching

Fausto Guinchiglia, Mikalai Yatskevich

Dept. of Information and Communication Technology
University of Trento,
38050 Povo, Trento, Italy
{fausto, yatskevi}@dit.unitn.it

Abstract. We think of Match as an operator which takes two graph-like structures and produces a mapping between semantically related nodes. The matching process is essentially divided into two steps: element level and structure level. Element level matchers consider only labels of nodes, while structure level matchers start from this information to consider the full graph. In this paper we present various element level semantic matchers, and discuss their implementation within the S-Match system. The main novelty of our approach is in that element level semantic matchers return semantic relations (=, ⊑, ⊒, ⊥) between concepts rather than similarity coefficients between labels in the [0, 1] range.

1. Introduction

We think of matching as the task of finding semantic correspondences between elements of two graph-like structures (e.g., conceptual hierarchies, database schemas or ontologies). Matching has been successfully applied to many well-known application domains, such as schema/ontology integration, data warehouses, and XML message mapping.

Semantic matching, as introduced in [4, 6], and its implementation within the S-Match system [7] are based on the intuition that mappings should be calculated between the concepts (but not labels) assigned to nodes. Thus, for instance, two concepts can be equivalent; one can be more general than the other, and so on. As from [6], all previous approaches are classified as syntactic matching. These approaches, though implicitly or explicitly exploiting the semantic information codified in graphs, differ substantially from our approach in that, instead of computing semantic relations between nodes, they compute syntactic "similarity" coefficients between labels, in the [0,1] range (see [6] for an in depth discussion about syntactic and semantic matching).

The system we have developed, called S-Match, takes two trees, and for any pair of nodes from these two trees, it computes the strongest semantic relation holding between them. In order to perform this, the matching task is articulated into two basic steps, namely element and structure level matching (See [7] for details). Element level matchers consider only the information at the atomic level (e. g., the information contained in elements of the schemas), while structure level matchers consider also the information about the structural properties of the schemas.

Our goal in this paper is to describe a set of element level semantic matchers, as they have been implemented within S-Match. In order to satisfy the input requirements of the structure level matchers the element level matchers return semantic relations (=,⊒, ⊑, ⊥). Some matchers are modifications of previously developed syntactic matchers. The main novelty is the output returned. However, we have introduced the new method for determining semantic words relatedness namely *semantic gloss comparison*.

The rest of the paper is organized as follows. Section 2 provides an overview of S-Match. Section 3 is dedicated to semantic element level matchers. *String based* matchers are discussed in Section 4, while *sense* and *gloss based* matchers are described in Sections 5 and 6, respectively. The descriptions of matchers are structured as follows. First, we give the overview of the matcher under consideration. Afterwards, we provide some examples of the matcher results with execution times (all tests were performed on a P4 computer with 512 Mb RAM installed). Finally, we discuss the results obtained.

2. S-Match: Algorithm and Implementation

According to [6] possible semantic relations returned by element level matchers are: *equivalence* (=); *more general* (⊒); *less general* (⊑); *mismatch* (⊥); *overlapping* (∩). They are ordered according to decreasing binding strength, e.g., from the strongest (=) to the weakest (∩). When no relations are found the special *Idk* (I don't know) relation returned.

As from [7], the S-Match algorithm is organized in the following four macro steps:

- *Step 1*: for all labels in the two trees, compute concepts denoted by labels
- *Step 2*: for all nodes in the two trees, compute concepts at nodes
- *Step 3*: for all pairs of labels in the two trees, compute semantic relations among concepts denoted by labels
- *Step 4*: for all pairs of nodes in the two trees, compute semantic relations among concepts at nodes

Let us consider, for instance, the two trees depicted in Figure 1a.

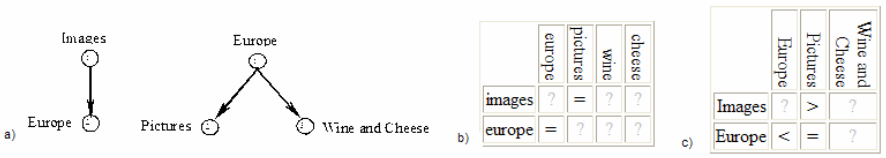


Fig. 1. Simple schemas (a). Matrices of relations between labels (b) and concepts at nodes (c).

During Step 1 we try to capture the meaning of the labels in the trees. In order to perform this we first tokenize the complex labels. Then for instance "*Wine and Cheese*" from Figure 1 becomes <Wine, and, Cheese>. Then, we lemmatize tokens; and "*Images*" becomes "*image*". Then, an Oracle (at the moment we use WordNet 2.0 as an Oracle) is queried in order to obtain the senses of the lemmatized tokens. Afterwards, these senses are attached to the atomic concepts. Finally, the complex concepts

are built from the atomic ones. Thus, the concept of label *Wine and Cheese*, $C_{Wine\ and\ Cheese}$ is computed as $C_{Wine\ and\ Cheese}$ = <*wine*, *{senses*$_{WN}$*#4}*>&<*cheese*, *{senses*$_{WN}$*#4}*>, where <*cheese*, *{senses*$_{WN}$*#4}*> is taken to be the union of the four WordNet senses.

Step 2 takes into account structural schema properties. The logical formula for a concept at node is constructed, most often as the conjunction of the formulas in the concept path to the root (see [7] for more details).

Element level semantic matchers are applied during Step 3 while determining the semantic relations between labels. For example, we can derive from WordNet the information that *image* and *picture* are synonyms ($C_{Images} = C_{Pictures}$). The relations between the atomic concepts in our example in Figure 1a are depicted in Figure 1b. Element level semantic matchers provide the input to the structure level matcher, which is applied on the Step 4 and produces the matching result, which is depicted in Figure 1c.

The pseudo code of Step 3 which contains the calls of element level semantic matchers is provided in Figure 2. **`getRelation`** takes two concept labels (*`sLabel`*, *`tLabel`*) and their WordNet senses (arrays *`sSenses`*, *`tSenses`*) as input, and produces a semantic relation between these two labels (*`rel`*). First, it tries to obtain the relation from WordNet (line 2). If it does not succeed (the result is equal to *Idk*) the *string, sense, and gloss based* matchers are executed in sequential order (line 4).

```
1.String getRelation(String[] sSenses,String[] tSenses,
                              String sLabel,String tLabel)
2.  String rel=getWordNetRel (sSenses,tSenses);
3.  if (rel==”Idk”)
4.    rel=getMLibRel(sLabel,tLabel,sSenses,tSenses);
5.  return rel;
```

Fig. 2. Pseudo code of weak semantic matchers library.

`getWordNetRel` takes two arrays of WordNet senses and produces the strongest semantic relation between any two senses. In order to perform this, it triple loops on relations, source, and target senses. If there no semantic relations are found, it returns *Idk* .

`getMLibRel` takes two labels and two arrays of WordNet senses and returns a semantic relation between them. In order to perform this, it sequentially executes different *string, sense, and gloss based* matchers. *String based* matchers are executed once for each pair of input labels. *Sense* and *gloss based* matchers are executed for each pair of concept senses and each possible semantic relation between them. If the matchers fail to determine the relation, *Idk* is returned.

Notice that element level semantic matchers are executed only in the case we cannot obtain the necessary information from *WordNet* which is the only element level matcher whose result is guaranteed to be correct.

3. Element level semantic matchers

S-Match is implemented in Java 1.4. The current version contains 13 semantic element level matchers listed in Table 1.

Table 1. Element level semantic matchers implemented so far

Matcher name	Execution Order	Approximation level	Matcher type	Schema info
Prefix	2	2	String based	Labels
Suffix	3	2		
Edit distance	4	2		
Ngram	5	2		
Text Corpus	13	3		Labels + Corpus
WordNet	1	1	Sense based	WordNet senses
Hierarchy distance	6	3		
WordNet Gloss	7	3	Gloss based	WordNet senses
Extended WordNet Gloss	8	3		
Gloss Comparison	9	3		
Extended Gloss Comparison	10	3		
Semantic Gloss Comparison	11	3		
Extended semantic gloss comparison	12	3		

The first column lists the matcher names. The second column lists the order in which they are called. The third column introduces the notion of approximation level. The relation produced by matcher with first approximation level is always considered to be correct (e. g., *auto=car* returned by *WordNet*). The relations of second approximation level matcher is likely to be correct (e.g., *net=network* but *hot=hotel* by *Suffix*). The third approximation level relations are fuzzier in the sense that they depend on the context of the matching task (e.g., *cat* can be considered equivalent to *dog* by *Extended Gloss Comparison* in the sense they are both pets). It can be notified that matchers are executed following the order of increasing approximation. The fourth column reports the matcher type. The fifth column reports the matchers' input. At the moment we have three main categories of matchers. *String based* matchers have two labels as input (with exception of the *Text Corpus* which takes also a text corpus). *Sense based* matchers have two WordNet senses in input. *Gloss based* matchers also have two WordNet senses as input and produce relations exploiting gloss comparison techniques.

For lack of space in this paper we describe only some of the matchers. We do not consider: *Suffix*, *Ngram* and *Semantic Gloss Comparison*. A full description is reported in the technical report version of this paper.

4. String based matchers

Approximate string matching techniques [10] are widely used in various schema matching systems [5, 13]. Our *String based* matchers are modifications of well known element level syntactic matchers, and produce an equivalence relation if the input labels satisfy the given criteria, which are specific for each matcher. Otherwise, *Idk* is returned.

4.1 Prefix

Prefix checks whether one input label starts with the other. It returns an equivalence relation in this case, and *Idk* otherwise. The examples of relations *Prefix* produced and the time it needs to compute them are summarized in Table 2.

Prefix is efficient in matching cognate words and similar acronyms (e.g., *RDF* and *RDFS*) but often syntactic similarity does not imply semantic relatedness. Consider the examples in Table 2. The matcher returns equality for *hot* and *hotel* which is wrong but it recognizes the right relations in the case of the pairs *net*, *network* and *cat*, *core*.

Table 2. The relations produced by the prefix matcher and its execution time

Source label	Target label	Relation	Time, ms
net	*network*	=	0.00006
hot	*hotel*	=	0.00006
cat	*core*	*Idk*	0.00005

4.2 Edit Distance

Edit distance calculates the edit distance measure between two labels. The calculation includes counting the number of the simple editing operations (delete, insert and replace) needed to convert one label into another and dividing the obtained number of operations with *max(length(label1),length(label2))*. The result is a value in [0..1]. If the value exceeds a given threshold (0.6 by default) the equivalence relation is returned, otherwise, *Idk* is produced.

Edit Distance is useful with some unknowns to WordNet labels. For example, it can easily match labels *street1, street2, street3, street4* to *street* (edit distance measure is 0.86). In the case of matching *proper* with *propel* the edit distance similarity measure has 0.83 value, but equivalence is obviously the wrong output.

Table 3. The relations produced by the edit distance matcher and its execution time

Source label	Target label	Relation	Time, ms
street	*street1*	=	0.019
proper	*propel*	=	0.016
owe	*woe*	*Idk*	0.007

4.3 Text Corpus

Corpus based matchers exploit natural language processing and sense disambiguation techniques [2, 3, 12, 16]. However, with the noticeable exception of [15], they have never been used in the schema matching/ontology alignment context.

Corpus based matchers find occurrences of the first label in the second label immediate vicinity (or text window, whose size typically varies from a few to several thousand characters) in a corpus. *Text Corpus* has in input two labels and a text corpus and produces a relation between the labels. If a sufficient number of labels co-

occurrences is found, then *Text Corpus* returns the equivalence relation. With these matchers the major problem is the choice of the right corpus. For example, using the Genesis, from The King James Holy Bible as corpus we can infer (if we have the text window of size at least 3 words ahead) that *darkness* is related to *night* because of the following sentence.

5: *And God called the light Day, and the darkness he called Night. And the evening and the morning were the first day.*

But using the same example (and window size of 4 words ahead) we can infer that *God* is related to *Day* which is wrong. Table 7 illustrates the example.

At the moment we use a very simple version of this matcher, which calculates the label co-occurrences within a given text window in a given corpus. If these co-occurrences exceed a given threshold (at the moment this value depends on the corpus size) the equivalence relation is produced. Otherwise, the matcher returns *Idk*. A possible solution for the "right" corpus selection strategy is to query Internet search engines as sources of relevant corpuses [15].

Table 7. The relations produced by the text corpus matcher and its execution time

Source label	Target label	Relation	Time, ms
Night	*darkness*	=	0.008
God	*Day*	=	0.008
light	*first*	*Idk*	0.009

5. Sense based matchers

Sense based matchers take in input two WordNet senses and exploit the structural properties of WordNet hierarchies.

WordNet [14] is a lexical database which is available online [20] and provides a large repository of English lexical items. WordNet contains synsets (or senses), structures containing sets of terms with synonymous meanings. Each synset has a gloss that defines the concept that it represents. For example the words *night, nighttime* and *dark* constitute a single synset that has the following gloss: *the time after sunset and before sunrise while it is dark outside*. Synsets are connected to one another through explicit semantic relations. Some of these relations (hypernymy, hyponymy for nouns and hypernymy and troponymy for verbs) constitute *kind-of* and *part-of* (holonymy and meronymy for nouns) hierarchies. In example, *tree* is a kind of *plant*, *tree* is hyponym of *plant* and *plant* is hypernym of *tree*. Analogously from *trunk* is a part of *tree* we have that *trunk* is meronym of *tree* and *tree* is holonym of *trunk*.

We translate the relations provided by WordNet to semantic relations according to the following rules:

- A $\sqsubseteq$ B if A is a hyponym, meronym or troponym of B;
- A $\sqsupseteq$ B if A is a hypernym or holonym of B;
- A = B if they are connected by synonymy relation or they belong to one synset (*night* and *nighttime* from abovementioned example);
- A $\perp$ B if they are connected by antonymy relation or they are the siblings in the *part of* hierarchy

Further, we use the notion of extended gloss [3]. An extended gloss is a text corpus obtained by concatenating the glosses of synsets known to be related, via a WordNet hierarchy, with a given WordNet synset. For example, two extended glosses can be built for the concept *tree*. The first consists the glosses of the less general synsets (*trunk, oak*, etc). The second is obtained from the glosses of the more general synsets (*plant, object,* etc.).

5.1 WordNet

WordNet is an exact element level semantic matcher. It provides if it exists, a relation between two input senses and *Idk* otherwise. For example, *car*, according to WordNet, is more general than *minivan*. Thus, we return $\sqsupseteq$ relation (See Table 8). On the other hand, *red* and *pink* are not connected by any of WordNet relations and we return *Idk*. The results depend heavily on the content of WordNet. Our work with this matcher is basically a reimplementation of the work described in [4]. An interesting extension of this work is the possibility of extending WordNet with domain specific information.

Table 8. The relations produced by WordNet matcher and its execution time

Source label	Target label	Relation	Time, ms
car	*minivan*	$\sqsupseteq$	2,3
car	*auto*	=	0.6
tail	*dog*	$\sqsubseteq$	0.2
red	*pink*	*Idk*	0.4

5.2 Hierarchy distance

Hierarchy based matchers measure the distance between two concepts in a given input hierarchy. Several semantic word relatedness measures have been proposed. See, for instance [1, 8, 17, 19]. At the moment we use a very simple hierarchy distance measure, which is a slight modification of the method used in [17]. In particular, *Hierarchy distance* returns the equivalence relation if the distance between two input senses in a WordNet hierarchy is less than a given threshold value and *Idk* otherwise. The number of less general and more general arcs is also considered. In the case of equivalence they must be nearly of the same number.

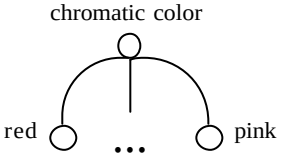


Fig. 3. The immediate vicinity of *red* and *pink* concepts in WordNet is-a hierarchy

Consider the example in Fig. 3.It can be noticed that, there is no direct relation between *red* and *pink*. However, the distance between these concepts is 2 (1 more gen-

eral link and 1 less general). Thus, we can infer that *red* and *pink* are close in their meaning and return the equivalence relation. On the other hand, synsets of *catalog* and *classification* are not connected through a WordNet hierarchy (e.g., they have different top level ancestors). Thus, *Idk* is returned. Table 9 illustrates these examples.

Table 4. The relations produced by hierarchy distance matcher and its execution time

Source label	Target label	Relation	Time, ms
red	*pink*	=	0.159
catalog	*classification*	*Idk*	0.203

This matcher has several modifications regarding the way the distance between the concepts is calculated. In the current implementation we use a simple hierarchy distance measure. We count the number of arcs in a *is-a* hierarchy and if this value is less than a given threshold the equivalence relation is returned.

Hierarchy distance works relatively fast and provides a good approximation of the concepts similarity. The major drawback of this matcher is the strong dependence on the concepts vicinity structure in WordNet. The hierarchy based word relatedness measures from [19, 9, 1] can also be adapted to an matching application.

6. Gloss based matchers

Gloss based matchers, similarly to *sense based* matchers, have in input two WordNet senses and return the semantic relation holding between them. However, *gloss based* matchers differ in that they use the information contained in WordNet glosses. Many of them exploit techniques from natural language processing [2, 3, 12, 16].

The majority of *gloss based* matchers use corpus based and corpus comparison techniques. As a result, two questions arise: which corpuses should be compared and how comparison should be performed? Concerning the second problem, at the moment we use two methods.

According to the first method, we calculate the number of occurrences of the same words in two corpuses. If the number exceeds a given threshold the relation is produced. We call this method *syntactic corpus comparison*. The second method is based on the calculation not only of the number of occurrences, but also of the number of synonyms, and more and less general words between corpuses. We call this method as *semantic corpus comparison*. Table 10 reports what is compared and how comparison is performed in the *gloss based* matchers implemented so far.

Table 10. Sense based matchers: what is compared and how comparison is performed

Matcher name	What is compared?	Comparison method
WordNet gloss	labels and gloss	syntactic
WordNet extended gloss	labels and extended gloss	syntactic
Gloss comparison	gloss and gloss	syntactic
Extended gloss comparison	gloss and extended gloss	syntactic
Semantic gloss comparison	gloss and gloss	semantic
Extended semantic gloss comparison	gloss and extended gloss	semantic

6.1 WordNet gloss

WordNet gloss compares the labels of the first input sense with the WordNet gloss of the second. First, it extracts the labels of the first input sense from WordNet. Then, it computes the number of their occurrences in the second gloss. If this number exceeds a given threshold, $\sqsubseteq$ is returned. Otherwise, *Idk* is produced.

The reason why the less general relation is returned comes from the lexical structure of the WordNet gloss. Very often the meaning of the index words is explained through a specification of the more general concept. In the following example, *hound* (*any of several breeds of dog used for hunting typically having large drooping ears*) *hound* is described through the specification of the more general concept *dog*. In this example *hound* is a *dog* with special properties (*large drooping ears, used for hunting*).

Counting the label occurrences in the gloss does not give a strong evidence of what relation holds between concepts. For example, *WordNet gloss* returns the less general relation for *hound* and *ear* in the abovementioned example, which is clearly wrong. Table 11 illustrates the example.

Table 11. The relations produced by the WordNet gloss matcher and its execution time

Source label	Target label	Relation	Time, ms
hound	*dog*	$\sqsubseteq$	0.031
hound	*ear*	$\sqsubseteq$	0.014
dog	*cat*	*Idk*	0.033

This matcher implements the ideas developed in [12, 2]. The main difference is that the matcher returns a semantic relation rather than a numerical similarity coefficient.

6.2 WordNet extended gloss

WordNet extended gloss compares the labels of the first input sense with the extended gloss of the second. This extended gloss is obtained from the input sense descendants (ancestors) descriptions in the *is-a* (*part-of*) WordNet hierarchy. A given threshold determines the maximum allowed distance between these descriptions and the input sense in the WordNet hierarchy. By default, only direct descendants (ancestors) are considered.

The idea of using extended gloss originates from [3]. Unlike [3], we do not calculate the *extended gloss overlaps measure*, but count the number of first input sense labels occurrences in the extended gloss of the second input sense. If this number exceeds a given threshold, a semantic relation is produced. Otherwise, *Idk* is returned. The type of relation produced depends on the glosses we use to build the extended gloss. If the extended gloss is built from descendant (ancestor) glosses, then the $\sqsupseteq$ ($\sqsubseteq$) relation is produced.

For example, the relation holding between the words *dog* and *breed* can be easily found by this matcher. These concepts are not related in WordNet, but the word *breed* occurs very often in the *dog's* descendant glosses. Table 12 illustrates the example.

Table 12. The relations produced by the WordNet extended gloss matcher and its execution time

Source label	Target label	Relation	Time, ms
dog	*breed*	$\sqsupset$	0.268
dog	*cat*	*Idk*	4.3
wheel	*machinery*	$\sqsubseteq$	2.6

6.3 Gloss comparison

Within *Gloss comparison* the number of the same words occurring in the two input glosses increases the similarity value. The equivalence relation is returned if the resulting similarity value exceeds a given threshold. *Idk* is produced otherwise.

Let us try to find the relation holding, for example, between *Afghan hound* and *Maltese dog* using gloss comparison strategy. These two concepts are breeds of dog, but unfortunately WordNet does not have explicit relation between them. However, the glosses of both concepts are very similar. Let us compare:

Maltese dog is a breed of toy dogs having a long straight silky white coat; and:
Afghan hound is a tall graceful breed of hound with a long silky coat; native to the Near East.

There are 4 shared words in both glosses (*breed, long, silky, coat*). Hence, the two concepts are taken to be equivalent. Table 13 illustrates the example.

Table 13. The relations produced by the gloss comparison matcher and its execution time

Source label	Target label	Relation	Time, ms
Afghan hound	*Maltese dog*	=	0,074
dog	*cat*	*Idk*	0,019

Several modifications of this matcher exist. One can assign a higher weight to the phrases or particular parts of speech than single words [16]. In the current implementation we have exploited the approach used in [16], but changed the output to be a semantic relation.

6.4 Extended Gloss comparison

Extended gloss comparison compares two extended glosses built from the input senses. Thus, if the first gloss has a lot of words in common with descendant glosses of the second then the first sense is more general than the second and vice versa. If the corpuses (extended glosses) formed from descendant (ancestor) glosses of both labels have a lot of words in common (this value is controlled by a given threshold) then the equivalence relation is returned.

For example, *dog* and *cat* are not connected by any relation in WordNet. Comparing the corpuses obtained from descendants glosses of both concepts we can find a lot of words in common (*breed, coat*, etc). Thus, we can infer that *dog* and *cat* are related

(they are both pets), and return the equivalence relation. The relations produced by the matcher and its execution time are summarized in Table 14.

Table 14. The relations produced by the extended gloss comparison matcher and its execution time

Source label	Target label	Relation	Time, ms
dog	*cat*	=	4.3
house	*animal*	*Idk*	79.8

The idea of this matcher originates from the *extended gloss overlaps measure*, as described in [3]. Unlike [3], we do not calculate the extended gloss overlaps measure, but return semantic relations produced by the rules stated above.

6.5 Semantic Gloss comparison

Semantic Gloss comparison is based on a new method for determining semantic words relatedness. This method extends the work of [16]. The key idea is to maintain statistics not only for the same words in the input senses glosses (like in *Gloss comparison*) but also for words which are connected through *is-a* (*part-of*) relationships in WordNet. This can help finding the gloss relevance not only at the syntactic but also at the semantic level. In *Semantic Gloss Comparison* we consider synonyms, less general and more general concepts which (hopefully) lead to better results.

In the first step the glosses of both senses are obtained. Then, they are compared by checking which relations hold in WordNet between the words of both glosses. If there is a sufficient amount (in the current implementation this value is controlled by a threshold) of synonyms the equivalence relation is returned. In the case of a large amount of more (less) general words, the output is $\sqsupseteq$ ($\sqsubseteq$) correspondingly. *Idk* is returned if we have a nearly equal amount of more and less general words in the glosses or there are no relations between words in glosses. Table 15 contains the results produced by semantic gloss comparison matcher.

Table 15. The relations produced by the semantic gloss comparison matcher and its execution time

Source label	Target label	Relation	Time, ms
dog	*breed*	$\sqsupseteq$	6.7
cat	*dog*	*Idk*	10.9
wheel	*machinery*	$\sqsubseteq$	1419

7. Conclusion

In this paper we have presented the library of element level semantic matchers implemented within the S-Match system. We have implemented three kinds of matchers: *string*, *sense* and *gloss based*.

A lot of work still needs to be done in order to identify their performance and how they can be successfully exploited in the many, very diverse, matching tasks.

References

1. E. Agirre and G. Rigau. Word sense disambiguation using conceptual density. In Proceedings of the 16th International Conference on Computational Linguistics, Copenhagen, 1996.
2. S. Banerjee and T. Pedersen. An adapted Lesk algorithm for word sense disambiguation using WordNet. In Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, 2002.
3. S. Banerjee and T. Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In Proceedings of the 18th International Conference on Artificial Intelligence (IJCAI-03), Acapulco, Mexico, 2003.
4. P. Bouquet, L. Serafini, S. Zanobini. Semantic Coordination: A new approach and an application. In Proceedings of ISWC 2003.
5. H. Do, E. Rahm. COMA - A system for Flexible Combination of Schema Matching Approaches, In Proceedings of VLDB 2002
6. F. Giunchiglia, P. Shvaiko. Semantic Matching. In The Knowledge Engineering Review Journal, 18(3) 2004.
7. F. Giunchiglia, P. Shvaiko, M. Yatskevich. S-Match: An algorithm and an implementation of semantic matching. In Proceedings of ESWS'04.
8. F. Giunchiglia, I. Zaihrayeu. Making peer databases interact - a vision for an architecture supporting data coordination. In Proceedings of the Conference on Information Agents, Madrid, September (2002)
9. J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings on International Conference on Research in Computational Linguistics, Taiwan, 1997.
10. P. Hall, G. Dowling. Approximate String Matching. In the Computing Survey 12: 4, 1980
11. M. Hearst. "Automated Discovery of WordNet Relations" in WordNet: An Electronic Lexical Database, Christiane Fellbaum (ed.), MIT Press, 1998.
12. M. Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. In Proceedings of SIGDOC 1986.
13. J. Madhavan, P.A. Bernstein, E. Rahm. Generic Schema Matching with Cupid. VLDB 2001
14. A. Miller. Wordnet: A lexical database for english. In Communications of the ACM, number 38(11) 1995.
15. P. Mitra, G. Wiederhold. Resolving Terminological Heterogeneity in Ontologies Proceedings ECAI-02 Workshop on Ontologies and Semantic Interoperability, 2002
16. S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In Proceedings of the 4th International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, 2003.
17. R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. In IEEE Transactions on Systems, Man and Cybernetics, 19(1), 1989.
18. E. Rahm, P.A. Bernstein. A Survey of Approaches to Automatic Schema Matching. In VLDB Journal 10: 4, 2001
19. P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, August 1995.
20. WordNet a lexical database for the English language. http://www.cogsci.princeton.edu/~wn

Communicating References
(Very Preliminary Report)

R.Guha

IBM Research

1 Introduction and Framework

Information Theory [2] has focussed on the problem of a receiver reproducing a sequence of symbols sent by a transmitter. Often the transmitter intends these symbols to *refer* to some entities. The problem of the receiver reproducing the intended references of the symbols, or more generally, the meaning of the message, has been given little attention.[1] In this paper we propose a mathematical framework for addressing the problem of communicating references.

We assume that there is an underlying world or structure that the messages pertain to. Parts of the structure are visible to the transmitter and receiver. Each message describes a portion of this structure. We say that the transmitter has communicated the meaning of a message when the receiver can identify the portion of the structure described by it. This gets complicated when transmitter and receiver don't share a vocabulary and grammar. In this case, the transmitter and receiver can guess some or even all of each others vocabularies by exploiting regularities in the structure. Our goal is to establish the relationship between the properties of the structure and what the receiver and transmitter need to a priori share to enable the communication of references.

The example in figure 1 illustrates our framework. There is an underlying predicate logic structure that the message describes. Some of this structure is visible to the transmitter and receiver. The transmitter and receiver each has a representation of the portions of the structure visible to her/him, each her/his own vocabulary, which we will assume are disjoint. In this example the transmitter intends the message ($P1(A, D) \wedge P2(A, C)$) to communicate a portion of this structure. The transmitter encodes the message in her vocabulary. We consider the receiver to have understood the meaning if he can identify the portion of the structure that the transmitter encoded in the message, that is also visible to him. In other words, the receiver should be able to re-encode the contents of the message in his own vocabulary, i.e., as ($Q1(S, V) \wedge Q2(S, U)$).

The meaning of the message is carried in part by the mapping of symbols to objects/relations and in part by the correspondence between grammatical relations between symbols in the message and relations between the denoted objects in the structure. We begin by assuming that the latter is simple (as it is in the case of predicate

[1] In fact, in [4] Shannon explicitly writes: "The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point. Frequently the messages have meaning; that is they refer to or are correlated according to some system with certain physical or conceptual entities. These semantic aspects of communication are irrelevant to the engineering problem."

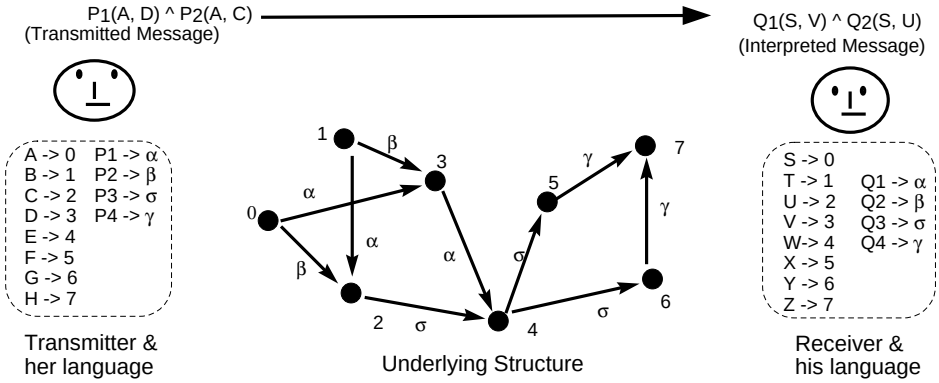


Fig. 1. Framework for communicating references

logics) and a priori shared by the receiver and transmitter. So, for example, we will assume that for both the transmitter and receiver, a string such as $P1(A, D)$ will be interpreted as representing that the objects denoted by A and D are in the relation denoted by $P1$. On the other hand, we do not make this assumption about the symbols. That is, the transmitter and receiver cannot be assumed to know all of the other's symbol to object/relation mappings. Therefore, to reconstruct the meaning of the message, the receiver has to construct a mapping between the symbols used by him and the transmitter. The transmitter and receiver may however a priori have the mapping for the symbols corresponding to some (possibly zero) of the objects. A symbol for which the transmitter and receiver a priori have the mapping is called a *Shared Symbol*. Though the transmitter and receiver may begin by sharing only some (possibly zero) symbols, by exploiting the messages and the underlying structure visible to both, they can bootstrap to obtain the mapping for all of the symbols.

We restrict our attention to a limited class of messages. In particular, we only consider declarative messages. Other speech acts such as questions and imperatives are ignored. We also only look at messages that do not rely on any contextual constructs (such as indexicals) to communicate their meaning.

A mathematical framework for studying the problem of communicating meaning is likely to have many applications. For example, problems in data integration, such as mapping between different data sources and problems in data privacy, such as hiding the identity of an individual can be cast in this framework.

2 Approach

We will use the example in figure 2 to illustrate our approach. Here, both parties observe the same structure and the messages are restricted to ground atomic formulas. Using standard predicate logic prefix notation (which is shared), the transmitter sends the message $P1(A, D) \wedge P2(A, C)$. Given the underlying structure, since there are only

2 objects in the structure that have 2 arcs with different labels going out, this message can be interpreted by the receiver only as $Q1(S,V) \wedge Q2(S,U)$ or $Q2(S,V) \wedge Q1(S,U)$ or $Q1(T,U) \wedge Q2(T,V)$ or $Q2(T,U) \wedge Q1(T,V)$. The shape of the substructure communicated by the message rules out certain interpretations. However, without more information, it is not possible to resolve the ambiguity between these interpretations. To resolve this ambiguity, the transmitter must provide the additional information that disambiguates between $0/1$, $2/3$ and α/β. An examination of the structure reveals that it is only necessary to provide the information that disambiguates between 2 and 3. Once this is done, there is only one possible interpretation. The disambiguating property of 3 is that it is the only object that has two arcs, one coming into it and one leaving it, with the same label. That is, 3 is the only object O satisfying the description $(\exists(p,y,z)(p(y,O),p(O,z)))$. By adding this *Disambiguating Description*, in her own vocabulary, the transmitter can unambiguously communicate her meaning. Since the message already contains $P1(A,D)$, she only needs to augment it with $P1(D,E)$. In general, our approach is to augment messages with disambiguating descriptions so as to make them unambiguous.

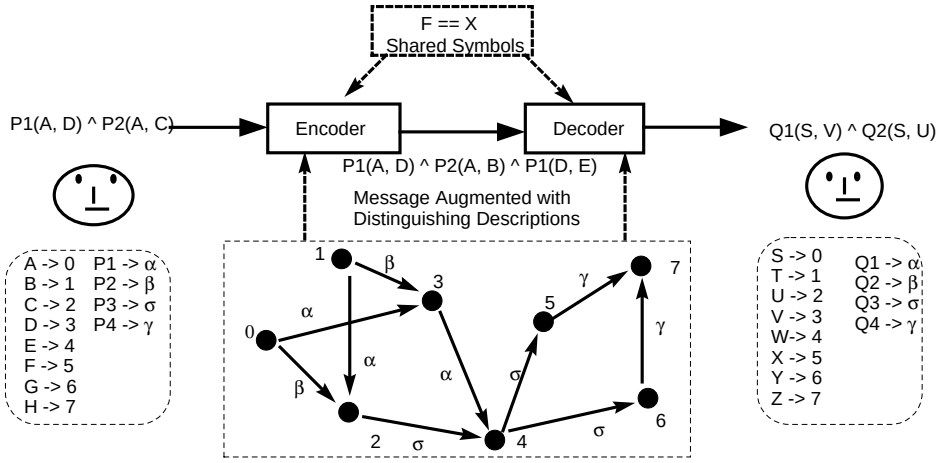


Fig. 2. Example of disambiguating descriptions

Now consider the objects 5 and 6. There is nothing in the structure that disambiguates between them. To disambiguate between these two, the receiver needs to a priori know the transmitter's symbol for either 5 or 6. In other words, either F or G needs to be a shared symbol. We would have a similar problem disambiguating between 0 and 1 if we imposed the constraint that disambiguating description could contain at most 1 literal.

In this example, we used ground atomic formulas involving an object to disambiguate it from others. Richer languages (such as those that allow quantification and

counting) allow for other kinds of descriptions. For example, to disambiguate between β and α, we could exploit the fact that they occur different number of times in the overall structure. More generally, descriptions of objects can include relations to other objects, statistical properties of the object/relation, etc. In this paper, we restrict our attention to descriptions using ground atomic formulae.

The structure used in the this example was very small. In practice, we are interested in very large structures. In any large enough structure, there will likely be objects that are structurally indistinguishable. This is especially the case if we impose restrictions on the size of the descriptions we can use. If we restrict the description to be of size M, there are at most $O(2^{\binom{M}{2}})$ distinct structural shapes of that size. If the size of the structure is larger than this, or if the structure does not contain a large enough variety of shapes, there will be structural isomorphisms. In such cases, where the shape of the substructure around an object by itself does not provide a unique reference mechanism, we rely on a combination of shapes and shared symbols to create the mappings.

To summarize, the shared structure and shared symbols impose constraints on the interpretation of a symbol in the message. When these constraints are strong enough to uniquely identify a particular object, they enable the transmitter to communicate a reference to the object. The transmitter can augment messages by adding disambiguating descriptions of the objects mentioned in the message so as to ensure proper interpretation of the message.

3 Parameterizations of the Framework

The framework be parameterized along various dimensions.

1. Overlap in the subsets of the underlying structure as observed by the transmitter and receiver: In the simplest case, they both observe the same subset. In the most complex case, the subsets they observe have no overlap, but contain the ‘same’ relations (i.e., different subsets of the same set of tuples). Most cases are in between, where the transmitter and receiver have some overlap. These are also the most interesting cases since it allows the transmitter to tell the receiver something he didn’t know and relate it to things he already knows.
2. The transmitter’s knowledge of the what the receiver can see: In the simplest case, the transmitter precisely which subset of the structure the receiver can see. At the other extreme, the transmitter has no knowledge about the receiver. In the middle we have cases where the transmitter has some, often statistical, model of what the receiver can observe.
3. Differences in the transmitter’s and receiver’s observation of the underlying structure: In the simple case, both correctly observe the structure. In the more general case, there are differences between the observations. E.g., consider an object in the structure that is a person with weight as one his attributes. The transmitter and receiver might have different values for his weight. If the transmitter uses his weight as part of a disambiguating description for him, the receiver might not recognize him or worse, mistake him for someone else.

4. (Non)-isomorphism of the representations used by the transmitter and receiver: In the simple case, we assume that both use representations that are isomorphic to the underlying structure. That is, the relation between the vocabulary used by the transmitter/receiver and the underlying structure is the inverse of the definition of satisfaction as conventionally defined in predicate logic. The more general case allows for the transmitter and receiver to have different representations of the same underlying structure.
   In other words, the grammars used by the transmitter and receiver could be much more complex than that of predicate logic and also different from each other. This is the case with natural languages, database schemas, etc.
5. The language used for the messages: In the simple case, the messages are just conjunctions of ground atomic formulas, i.e., we disallow quantifiers, negations and disjunctions. More complex cases allow some or all of these.
6. Class of structures: Though the framework discussed in this paper applies generally to all predicate logic structures and its derivatives (such as trees and XML), in this paper, we restrict our attention to finite directed labelled graphs, i.e., finite structures with no functions and only binary relations. Further, in most of this paper, unless the extension to multiple relations is not straightforward, we will assume that there is only a single relation.
7. Memory: If the transmitter and receiver can incrementally bootstrap from some to more shared symbols over a sequence of messages, fewer shared symbols may be required. However, this does impose additional computational resources. Arbitrary amounts of memory is essentially equivalent to arbitrarily long descriptions. In practice, allowing limited amounts of memory is useful.
8. Feedback: If the receiver can provide the transmitter feedback, they can exchange a set of back and forth messages to resolve ambiguities, thereby opening the potential for meaning negotiation. This is especially helpful if the receiver and transmitter observe different subsets of the structure.

The simplest model, which we used to illustrate our approach, assumes that the transmitter and receiver both correctly observe the same subset of the underlying structure, know this, use isomorphic representations, restrict messages to conjunctions of ground atomic formulas and don't have memory of previous messages. Most of this paper is restricted to this case.

4 Goals

Our goal is to reliably communicate references at the lowest possible cost. The biggest cost is associated with setting up the mappings for shared symbols. Ideally, we would like to have as few shared symbols as possible. If we allow for longer descriptions (i.e., higher M), we are likely to encounter a greater variety of shapes in the structure and might require fewer shared constants. However, longer descriptions also require more computational resources to generate and decode. Given a structure S and a maximum description length M, we want to compute the minimal set of symbols that needs to be shared in order to uniquely identify, with disambiguating descriptions of size less than M, all the non-shared objects and relations in the structure.

The number shared symbols required is a function of the shape of the structure and the size limit M on descriptions. There are structures that don't require any shared symbols — structures in which every object has a description that is structurally distinct from all other descriptions. These however can only be small. In the best case, if a structure with N objects has S_M different shapes with M objects each, we will need only $N/(MS_M)$ shared symbols. This is rarely achieved. At the other extreme, there are structures (such as cliques) where every description is structurally the same as every other description, for which every object needs to be shared.

Often, the structure has objects such as the integers or strings, which have natural/canonical representations that can be assumed to be shared. Given such a set of shared symbols, we are interested in determining the fraction of objects that can be mapped. We are interested in this question in the context of large (though finite) structures, where the transmitter sends a large number of messages to the receiver.

Fig. 3 provides a way of visualizing the problem of determining the minimum set of shared symbols. Given a set of D descriptions over N symbols, imagine if we replaced every symbol occurrance with a blank. Most of the descriptions will look isomorphic under these conditions, with the only differences coming from the shape. That is, there is a many to one mapping between the D descriptions to each of the isomorphism classes obtained by blanking the symbols. Now imagine randomly picking a symbol and unblanking all occurances of the symbol. This corresponds to sharing that symbol. Some of the isomorphism classes will split up to give more isomorphism classes. As more symbols are randomly chosen and shared, the number of distinct looking descriptions (or isomorphism classes) will increase. At some point one or more of these isomorphism classes will contain only one element, i.e., it will be unambiguous. Then, we reach a critical point at which each of the symbols that is not shared has a unambiguous description. As the number of symbols grows further, the number of ambiguous descriptions drops and finally, as it approaches approaches N, the number of isomorphism classes becomes equal to D. The sequence in which the symbols are shared determines the path taken from A to B. We want to find the sequence that reaches the critical point fastest.

We can approach this questions from combinatorial perspective or from a statistical perspective. In the combinatorial setting, we are interested in computing the minimum number of shared symbols required for a particular structure. The combinatorial approach is less applicable as we move away from the simplest model. For such cases, we use an approach where we characterize the structure statistically and relate it to the size of descriptions, number of unambiguous descriptions, number of shared symbols, etc. There is a related class of problems, namely that of generating and decoding disambiguating descriptions. Since these reduce to well known, traditional database query answering problems, we don't discuss them in this paper.

5 Complexity of MS and DISTINCT DESCRIPTION

The MSS problem is that of determining the minimum number of shared symbols. The DISTINCT DESCRIPTIONS problem is that of identifying the minimum number of shared symbols so that every description is distinct from every other description. When

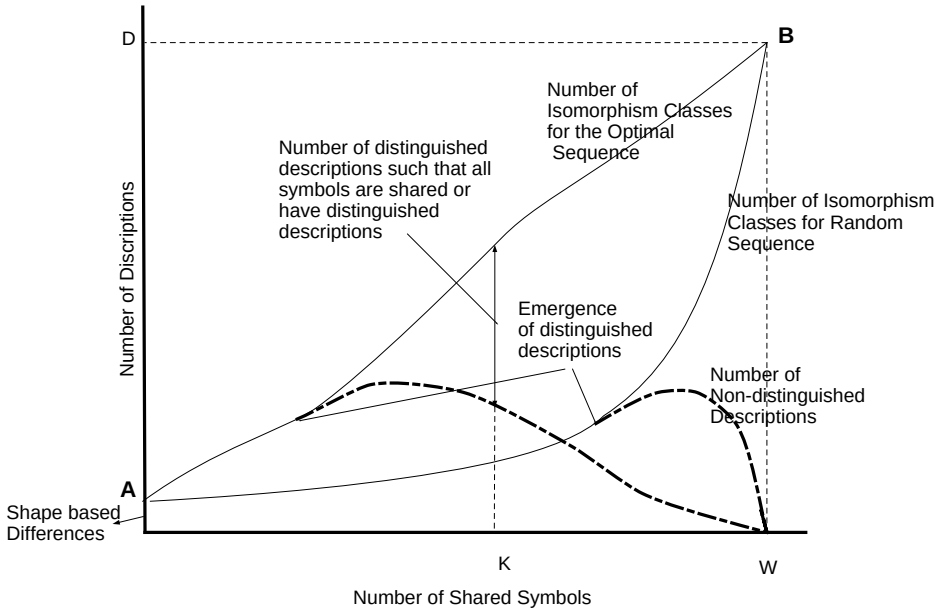


Fig. 3. Visualizing the sequence of sharing symbols

this is true, every object also has a unambiguous description, but since there can be many more descriptions than objects, a solution to the DISTINCT DESCRIPTIONS problem may require many more shared symbols than one for the MSS problem. We state some preliminary complexity results for the DISTINCT DESCRIPTION and MSS problems here.

Both DISTINCT DESCRIPTIONS and MSS are NP-Hard in the case where M is not a fixed parameter. We can show this by reducing the Feature Selection Problem [1] to DISTINCT DESCRIPTIONS in the special case where the structure is an unlabelled directed bipartite graph.

In the case where M is a fixed parameter, DISTINCT DESCRIPTIONS can be reduced to set cover with the size constraint M on each subset and therefore is at worst APX-Complete. An approach to MSS, which works when the smallest number of shared symbols also obtains the smallest (or close to smallest) number of unambiguous descriptions, is to first construct an integer program to compute the minimum number of unambiguous descriptions required and from the results of this program, construct another integer program that gives us the minimum number of shared symbols that disambiguates these descriptions.

6 Knowledge about the structure

The combinatorial approach is adequate when the simple model applies. In more complex cases, such as when the transmitter and receiver observe different but overlapping

parts of the structure, or when there may be errors in the observation, the combinatorial approach does not apply. In such cases, even though the transmitter and receiver cannot observe the entire structure, they might have some knowledge about the structure that enables them to communicate references.

For example, consider a structure about a set of students that specifies each student's first name, university and parents. It is very rare that two siblings to have the same first name. So, even if the transmitter and receiver don't see the entire structure, a description of the form 'Child of C_1 with the first name C_2' is likely to provide an unambiguous description. Even though the transmitter cannot verify that this is an unambiguous description, by using the knowledge that two siblings rarely have the same first name, she can guess that this is an unambiguous description. In this case, the transmitter and receiver have access to a portion of the structure and to the knowledge about the overall structure. We assume that in addition to the shared symbols, they also share this knowledge.

There are two ways of encoding this knowledge about the structure: axiomatic and statistical. In the axiomatic approach, we use a set of axioms to express the knowledge about the structure. Given a set of axioms Δ about a structure, a description $\beta(x)$ is unambiguous iff $\Delta \models (\forall(x, y)\beta(x) \wedge \beta(y) \rightarrow (x = y)$.

In the statistical characterization of the structure, β is considered unambiguous iff the probability of two objects being the same, conditioned on their both satisfying β, is sufficiently close to 1. In the next section, we present a statistical characterization of structures.

6.1 Partial Observation and Observation Differences

We require longer descriptions (or more shared symbols) when the transmitter and receiver cannot observe the entire structure or when there are differences in their observations.

When the transmitter picks a description to refer to an object, she needs to ensure that it is very unlikely that the receiver can see some different object that also satisfies this description. So, the transmitter needs to not only ensure that the description is unambiguous in the portion of the structure she can see, but also that it is very likely to be unambiguous in the overall underlying structure. Similarly, when the receiver dereferences the description, he uses the knowledge to ensure that it is very unlikely that the transmitter was referring to some other object satisfying the description. We need longer descriptions and more shared symbols when the receiver and transmitter observe only parts of the structure.

In the earlier example of students, consider the case where a very small number siblings have the same first name. If the complete structure were visible to both, the transmitter can use just the parent and first name for all but these few cases and use longer descriptions for these few cases. However, if they can't observe the structure, they don't know which students correspond to the rare case and have to use longer descriptions all the time.

Similarly, in the case where the transmitter and receiver observe certain certain parts of the structure differently, we longer descriptions. For example, consider the case where for a small number of students, the transmitter and receiver have different values

for the first name and/or university. Even in such cases, if this fraction of students is small enough and there are alternative unambiguous descriptions, by combining some of these alternative descriptions, the transmitter can construct a robust unambiguous description that correctly refers even in the presence of such differences.

7 Statistical Formulation

We first provide a general treatment for an arbitrary sample space of structures and then consider a special but common case where all the structures in the sample space exhibit certain common features.

7.1 General Formulation

There $O(2^N)$ possible structures involving N objects. The statistical characterization assigns each of these structures a probability of being the real underlying structure that the transmitter and receiver are observing. Given a set of K shared symbols, in each of these structures we will have a set (possibly empty) of unambiguous descriptions. The probability P_{D_i} of a description D_i being unambiguous is the sum of probabilities of all the structures in which it is unambiguous. The expected number of unambiguous descriptions (E_{UD}) is the sum of P_{D_i} over all descriptions. Similarly, the probability (P_{S_i}) of an unshared symbol S_i being communicable is the sum of probabilities of the structures in which there is an unambiguous description containing S_i. The expected number of symbols that can be communicated is the sum of P_{S_i} over all the unshared symbols.

The general case does not make any assumptions about the distribution. So, for example, the distribution could assign non-zero probabilities to a set of structures that are very dissimilar from each other. We are interested in the case where there is some known regularity in the underlying structure. In the example of the previous section, we are likely to find parents with one, two, three or four children, less likely to find parents with ten children and extremely unlikely to find parents with 20 children. Indeed, it is this regularity that is exploited by the transmitter and receiver to estimate the number of objects satisfying a description. The distribution is supposed to capture this regularity. Structures that exhibit this regularity should be assigned higher probabilities than those that don't. To do this we use a statistical model that operates at a finer granularity than the overall structure. In the next section, we present our statistical model of structures.

7.2 Ergodic Random Structures

Starting with Erdos and Renyi there has been considerable work on various models of random graphs [3]. The two basic models are $G(p, N)$ in which we have a graph with N nodes where the probability of finding an arc between any two nodes is p and $G(m, N)$ in which we have a set of equiprobable graphs with N nodes and m arcs. More recently many researchers have considered random graph models corresponding to a particular degree distribution. We generalize this to define our statistical model of structures.

The concept of the degree of a node can be extended to a directed labelled graph by associating in and out degrees with each arc label. The degree of a node is a very local description of the neighbourhood of a node. We can further generalize the notion of degree distribution to M-neighbourhood distribution. Let the different possible shapes of structures of radius M with less than N_{max} nodes be $(S_{M1}, S_{M2}, ..., S_{Ml})$. Given a random node, let the probability of the M-neighbourhood of the node being S_{Mi} be P_{Mi}. When $M = 1$, this reduces to the degree distribution of the structure. The M-neighborhood of a node is just the set of descriptions of length M that the node appears in. The joint M-neighbourhood of 2,3,... nodes is the set of descriptions of size M containing the 2,3,... nodes. As with simple M-neighbourhoods, we can characterize a structure by providing a distribution over joint M-neighbourhoods. Of particular interest is joint M-neighbourhood of M nodes. This gives us a precise characterization of the full set of descriptions. If the simple M-neighbourhoods of different nodes are independent of each other, the probabilities associated with the various joint neighbourhoods can be derived from the simple neighbourhood probabilities without any additional information. We will make this assumption in the rest of this paper. Let the number of descriptions in the neighbourhood S_{Mi} be C_{Mi}. A node with the neighbourhood S_{Mi} appears in C_{Mi} descriptions.

Traditional information theory largely restricts its attention to ergodic sequences. An ergodic sequence is one where the emperical distribution of the symbols in the sequence tends to its expected value as the sequence grows arbitrarily large. Informally, ergodic sequences are those for which there is some granularity above which they are uniform everywhere. By analogy, we define M-ergodic structures as those whose emperical distribution over M-neighbourhoods approaches its expected value as the structure grows arbitrarily large. This has the interesting consequence that the number of different types of neighbourhoods of size M cannot grow as the size of the structure grows. Further this restricts the number of nodes that can have very large in or out degrees of any node in the structure. In this paper, we restrict our attention to ergodic structures.

An important consequence of the ergodic assumption is that the laws of large numbers can be used. In particular, given a sufficiently large structure with N nodes with the neighbourhood distribution $(P_{M1}, P_{M2}, ..., P_{Ml})$, we can expect NP_{M1} nodes with the neighbourhood S_{M1}, NP_{M2} nodes with the neighbourhood S_{M2} and so on. Since a node with the neighbourhood S_{Mi} appears in C_{Mi} descriptions, the total number of descriptions is $N_D = \sum_{i=0}^{l} NPMiC_{Mi}/M$. In other words, the probability of finding a node j which has the neighbourhood S_{Mi} in a randomly chosen description is $P_j = C_{Mi}/N_D$. We will refer to this as the probability of the node/symbol.

7.3 Types of Errors

There are two types of errors that might occur when the transmitter and receiver don't both correctly observe the same structure. Detectable errors are those where the receiver can detect that there is a problem. These arise when the receiver detects more or less than one object satisfying the description. If the transmitter and receiver don't both observe enough of the structure, we can have a situation where the receiver cannot resolve the description to any object. In other cases, there may be multiple objects

satisfying the description. Undetectable errors are harmless since they don't lead to miscommunicated references.

Undetectable errors are those where the receiver resolves the description to a different object than the one that was intended by the transmitter. There will almost always be some fraction of communications that have undetectable errors. By using longer descriptions (more shared symbols), we can reduce the likelihood of error. The number of shared symbols is a function of the tolerable error which we denote by ϵ_D.

Given a certain number of shared symbols and fixed size of description per object, the likelihood of errors can be reduced by constructing longer descriptions that communicate references for multiple objects. In the limit, ϵ_D can be made arbitrarily small for a fixed overhead in size of descriptions (number of shared symbols). We do not discuss this approach here.

7.4 Computing the Shared Symbols

We now compute the expected number of required shared symbols for the three cases: when the transmitter and receiver both observe the same portion of the structure, when they observe different but overlapping parts of the structure and when there are observation differences. As mentioned earlier, we need longer descriptions and more shared symbols in the case where the transmitter and receiver cannot observe the entire structure or when there are observation differences.

Given a set of $G < M$ shared symbols with probabilities $P_1, P_2, ..., P_G$, the probability of having an unambiguous description containing these G (and no other) shared symbols is $P_{1,2,...G} = P(1-P)^{N_D-1}$ where $P = (\Pi_{i=1}^{G} P_i) \times \Pi_{j=G+1}^{K}(1-P_j) \times \frac{G!}{M!}$.

If the entire structure is observable, the transmitter can emperically verify whether a given description is unambiguous and hence the expected number of unambiguous descriptions with G shared symbols each (call this UD_G) for a set of K shared symbols is the sum of $P_{1,2,...,G}$ over all $\binom{K}{G}$ combinations of symbols. If the entire structure is not observable, the transmitter can use a particular description, whose probability of being unabiguous is $P_{1,2,...G}$ only if $P_{1,2,...G} > 1-\epsilon_D$ where ϵ_D is the allowable error. In this case, the expected number of unambiguous descriptions for a set of K shared symbols is the sum of $(P_{1,2,...,G} \quad if \quad P_{1,2,...,G} > 1-\epsilon; \quad 0 \quad otherwise)$ over all $\binom{K}{G}$ combinations of shared symbols. In the case where there may be observation differences, an unambiguous description is usable if all portions of it are observed identically by the transmitter and receiver. So, the probability of having an unambiguous description containing these G (and no other) shared symbols is $P_{1,2,...G} = P(1-P)^{N_D-1} \times PC_{1,2,...G}$ where $PC_{1,2,...G}$ is the probability of the description being observed identically.

The number of unshared symbols occurrances in these UD_G unambiguous descriptions is $UD_G \times (M-G)$. The total number of unshared symbol occurrances is obtained by summing over the different values of G, i.e., $N_{USP} = \sum_{G=1}^{M-1} UD_G \times (M-G)$. Any unshared symbol that occurs in one of these N_{USP} positions can be communicated using one of the unambiguous descriptions it occurs in. The expected number of distinct unshared symbols that occur in these N_{USP} positions is $N_{US} = \sum_{i=K+1}^{N} 1 - (1-(P_i/M))^{N_{USP}}$. When $N_{US} = N-K$, all the unshared symbols have unambiguous descriptions. So, given an M-neighbourhood distribution we can compute the number of unshared symbols that can be communicated using a given set of K shared symbols.

8 Related Work and Conclusion

There has been substantial work in many communities in related areas such as data base schema mapping, privacy and object tracking. It is beyond the scope of this paper to provide surveys of the extant work in those areas. However, it does seem that this work is the first to combine predicate logic models and information theoretic concepts to provide a first principles framework for this problem. The preliminary work reported in this paper are of course just the first tentative steps in the direction of what we hope will be a theory of communicating meaning.

References

1. M. Charikar, V. Guruswami, R. Kumar, S. Rajagopalan, and A. Sahai. Combinatorial feature selection problems. In *FOCS*, pages 631–640, 2000.
2. T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, 1991.
3. M. Newman. The structure and function of networks. *Computer Physics Communications*, 147:40–45, 2002.
4. C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, pages 379–423, 1948.

Reconciling Heterogeneous Information Sources

Kendall Lister and Leon Sterling

Intelligent Agent Laboratory
Department of Computer Science and Software Engineering
The University of Melbourne, Australia
krl,leon@cs.mu.oz.au

Abstract. This paper discusses the results of experiments in automatic ontological reconciliation using AReXS, an implementation of the Example-Based Frame Mapping algorithm. Modifications and extensions to the base algorithm are discussed in the context of comparing heterogeneous data from multiple on-line information sources. Consideration is given to the types of data encountered and their impact on the task of reconciling meaning. Other less automatic approaches to meaning negotiation are discussed and an argument is made for combining automatic and manual reconciliation techniques.

1 Introduction

AReXS (Automatic Reconciliation of XML Structures) is an application that reconciles heterogeneous data sources presented in XML documents. It aligns data sources according to their implicit ontological structure. It is able to reconcile differences of expression and representation across XML documents from heterogeneous sources without any predefined knowledge or human intervention [6, 4]. It achieves this by identifying XML elements whose meanings are similar enough to be considered equivalent. AReXS uses Example-Based Frame Mapping (EBFM) [5] and requires no knowledge or experience of the domain in which it works - the algorithm is completely domain independent. By requiring no domain knowledge, AReXS is suitable for application to any field; its success relies on its ability to identify and resolve the differences in representation that result from sourcing data from a heterogeneous environment. The AReXS implementation of the EBFM algorithm is described in more detail in [7].

AReXS works by analysing two XML structures and identifying matching elements, generating a map of equivalence between concepts represented in the two documents. It is important to note that no formal representation of such concepts is attempted - rather, it is assumed that elements represent concepts, and that the equivalence of two elements can be deduced based on similarities between instances of both elements. As a metaphor, AReXS works in much the same way that two people who do not share a common language might teach each other by pointing at objects and saying the names that each person's language gives to that object.

Identification of conceptual equivalence is based on a consideration of lexicographical similarity between both the names and the contents of XML tags in each document. Matches are then assessed to deduce structural similarities between documents from different sources. By repeating this search for semiotic correspondence across other pairs of elements generated from the contents of the XML documents under consideration, AReXS is able to build a local context for data and then use this context to reconcile the ontological differences between XML documents.

2 Data collection and preparation

In order to explore the possibilities for automatic reconciliation of data of a type commonly encountered on the Internet, the domain of second-hand car advertisements was chosen for a series of experiments. Classified advertisements are one of the more popular and successful use that people have found for the Internet so far, probably due to the fact that they leverage the Internet's strengths, being convenient communication and information retrieval, while avoiding it's weaknesses by conducting actual transactions off-line - although auction sites are also popular on the Internet, for high-value items such as cars, classified advertisement sites tend to act more like match-makers than brokers, bringing together interested sellers and buyers who then meet and make their own arrangements. Five popular web sites were chosen:

- Autotrader (*www.autotrader.com.au*)
- Autoweb (*www.autoweb.com.au*)
- Carsales (*www.carsales.com.au*)
- Drive (*www.drive.com.au*)
- Ebay (*www.ebay.com.au*)

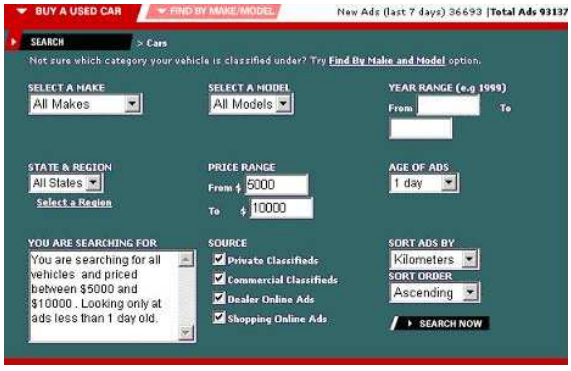


Fig. 1. Search interface for Autotrader

The search function of each site, an example of which is shown in Figure 1, was used to collect 100 current entries, creating a database of 500 car descriptions. The web sites typically presented the car descriptions in the form of tables, like that shown in Figure 2. This data was then parsed by custom written information agents to create pseudo-XML documents (it was not necessary to conform strictly to the XML standard for our purposes). The descriptions of cars were thereby transformed into documents of the form:

```
<Autotrader>
    <car>
        <make_model>Toyota Corolla SECA</make_model>
        <yr>1993</yr>
        <kms>140,000</kms>
        <price>$9,995</price>
        <state>WA</state>
        <media>?</media>
        <dealer></dealer>
    </car>
</Autotrader>
```

It is important to note that no two documents shared the same XML schema - the XML documents created naturally reflected the same variety of fields and names as the original web sites. As a contrasting example, another document appeared thus:

```
<Ebay>
    <vehicle>
        <bids>19</bids>
        <car>1993 XG LONGREACH UTE</car>
        <price>AU $2,600.00</price>
        <time_left>$7d 03h 10m</time_left>
    </vehicle>
</Ebay>
```

Variations in field names, missing fields and extraneous data were not the only differences among the information sources sampled. The data presented by the classified advertisement web sites was not always even in a tabular form - Figures 3, 4 and 5 show different styles of presentation. Regardless, the use of task-oriented information agents to extract the required data from the web pages made collecting the data into a consistent form quite simple. Attempting to create a single information agent to parse all five web sites would have taken significantly longer than this more direct approach. When creating the XML documents to be processed by AReXS, whereever possible the markup tag names were taken directly from the original source data.

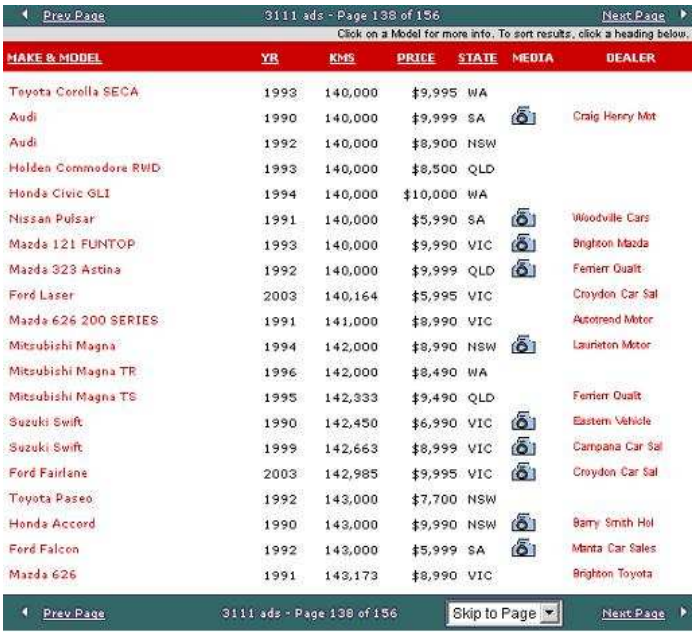


Fig. 2. Car descriptions presented by Autotrader

Fig. 3. Car descriptions presented by Ebay

Fig. 4. Car descriptions presented by Carsales

Fig. 5. Car descriptions presented by Autoweb

3 Reconciliation

Each of the five sets of car descriptions were paired and given to AReXS to reconcile, producing a collection of mappings between fields based on any evidence that AReXS could find to claim conceptual correspondences between the sets. For example, the data from Autotrader was compared with with that from Autoweb:

Source Autotrader.xml	*Source Autoweb.xml*
<make_model>Subaru Liberty GX</make_model>	<year>1989</year>
<yr>1993</yr>	<description>FORD FALCON</description>
<kms></kms>	<price>$5,990</price>
<price>$9,990</price>	<body_type>WAGON</body_type>
<state>VIC</state>	<colour>YELLOW</colour>
<dealer>Eastern Vehicle</dealer>	<location>OAKLEIGH, VIC</location>

The original form of the data for these two web sites can be seen in Figures 2 and 5. These two representations are strictly tabular and it is very easy to read a schema or ontology for the data directly from the column headings. Other web sites did not present such a straight forward format, notably those in Figures 4 and 3.

For a person, a quick glance at the sample records from Autotrader and Autoweb is enough to identify two direct field matches, *yr* with *year* and *price* with *price*. Relying solely on the field names, *make_model* might not be expected to encode the same meaning as *description*, but the sample contents of these two fields suggest that there may in fact be a correlation. AReXS proposed the following mapping between these two data sources:

Slot 0 [*year*]	↔ Slot 1 [*yr*]	: 0.99965
Slot 1 [*description*]	↔ Slot 0 [*make_model*]	: 0.99996
Slot 2 [*price*]	↔ Slot 2 [*kms*]	: 0.81160
Slot 2 [*price*]	↔ Slot 3 [*price*]	: 0.99962
Slot 3 [*body_type*]	↔ Slot 0 [*make_model*]	: 0.41018
Slot 5 [*location*]	↔ Slot 0 [*make_model*]	: 0.02155

Comparing the automated results with our expectations, the correlation between *year* and *yr* has been convincingly detected, as has that between *price* and *price*. Further, as predicted based on the field content, a strong correspondence has also been identified between *description* and *make_model*. These are the three strong content-based concept matches. However, a number of other weaker correspondences are suggested, and these are just as interesting as the strong matches. Based on the example data, there is a moderate correlation between *price* values and *kms* values. This is understandable, as AReXS used the Character-Based Best Match string comparison algorithm [10], which focuses on pairs of characters within the strings being compared. Since both *price* and *kms* are actually numbers, there is naturally a high coincidental correspondence, as

numbers have a small alphabet and so exhibit much less variation than alphanumeric strings. The CBBM algorithm was further confused by the fact that for second-hand cars, the distance travelled will generally be in the order of 100,000 kilometres and the asked sale price will be in the order of 10,000 dollars, thus there is almost no difference in the lengths of the strings that represent these numbers. It seems very likely that the only reason that the correspondence between *price* and *kms* was not stronger is the '$' dollar sign present in the *price* fields.

AReXS also reported a weak correspondence between *body_type* and *make_model*. Examining the sample date reveals that the people who entered the data occasionally include words such as 'sedan' or 'wagon' in the *make_model* field, which seems natural enough. This highlights the fact that even in natural language communication between people, ontological differences arise - some car owners obviously believe that the concept of a car's make and model includes its body type, whereas others believe that that information belongs elsewhere. Some probably would have included it in a separate field if one had been offered, but lacking such a field decided to attach the information to the most appropriate available field. We also believe that the lexicographic nature of the CBBM comparison algorithm contributed to the weak correspondence between *body_type* and *make_model*. These fields contain natural language words which inherently contain common substrings - for example, the words 'rather', 'catcher' and 'therapy' would be considered quite similar by the CBBM algorithm, for the purely coincidental reason that the substrings 'at' and 'ther' happen to occur frequently in English, although there is no semantic reason. Similarly, we also explain the slight correspondence between *location* and *make_model* as merely the result of linguistic coincidence.

In many cases, the unexpected results were as interesting as the expected ones, and revealed potential methods for improving the AReXS functionality. For example, both the Autotrader site and the Drive site included in their car descriptions fields labelled *colour*. Typical contents of these fields were *white*, *yellow* and so on. Intuitively, these fields should have been among the easiest for AReXS to match, and yet when comparing 100 records from each site AReXS was only confident enough to rate them with a correspondence of 0.71911. It seems likely that a string comparison algorithm working on a word basis rather than a character-pair basis as the CBBM algorithm does would be more appropriate for such data, yet it is not clear how an information agent should make a decision like this. Perhaps colour should be regarded as a basic concept that all information agents should know (general knowledge, in knowledge classification scheme implemented in CASA), or at least ones dealing with cars (domain knowledge, as per CASA). Similarly, although prices and numbers look similar, they generally shouldn't be considered for conceptual equivalence just because they both consist mainly of digits. AReXS copes quite well with them, but tends to allow too much correspondence between them - some basic knowledge about types of data would surely improve its results.

4 Modifications to the AReXS algorithm

Two noteworthy modifications were made to the AReXS implementation of the EBFM algorithm, after running the initial experiments. The first of these was to remove uniqueness screening for fields in the input XML documents (referred to as slots and frames respectively by [5]). The EBFM algorithm screens input data records according to their uniqueness, that is, how different they are to all other input data. This was found to significantly reduce the ability identify semantic matches, as even what appeared to be strong correspondences were heavily penalised. In effect, fields such as vehicle prices had very little chance of being successfully reconciled because there was relatively little variance in the actual prices. We felt that similarity within a field in a single information source shouldn't reduce the value of matches between fields from different sources, and when we removed this restriction we found the results to be more in line with our expectations, and thus much more useful.

The second modification that we made was similar. We made the reward for matching contents of fields independent of the uniqueness of the match. The original EBFM algorithm devalues matches if they are common, whereas we found that this reduced the effectiveness of the reconciliation process. An example of the effect of removing this restriction was the following improvement in reconciling two information sources:

Reward tied to uniqueness of match:

Slot 0 [*make_model*] ↔ Slot 1 [*description*] : 0.57109
Slot 3 [*price*] ↔ Slot 2 [*price*] : 0.07830

Reward tied only to similarity of contents:

Slot 0 [*make_model*] ↔ Slot 1 [*description*] : 0.99997
Slot 0 [*make_model*] ↔ Slot 3 [*body_type*] : 0.41019
Slot 0 [*make_model*] ↔ Slot 5 [*location*] : 0.02156
Slot 1 [*yr*] ↔ Slot 0 [*year*] : 0.99965
Slot 2 [*kms*] ↔ Slot 2 [*price*] : 0.81161
Slot 3 [*price*] ↔ Slot 2 [*price*] : 0.99962

Certainly the number of false positive results has been increased, but they are well below the cut-off level at which a correspondence would be considered to be a semantic match (0.9 or greater seems to be a reasonable limit). The only real concern is the moderate correspondence between *kms* and *price*, but even this is not rated highly enough that presents a risk of being confused with other stronger matches. More importantly, the confidence of the matches between *make_model* and *description* and *price* and *price* has increased dramatically, and a further very strong correspondence has been identified between *yr* and *year*, bringing the overall result much closer to what we had expected based on our own understanding of the sample data.

5 Conclusions and future directions

Generally, it is our opinion that the techniques for enabling semantic interoperability used in AReXS complement well other techniques such as explicit ontology-based approaches. Combining a variety of techniques should lead to a synergy that provides even better results. As identified earlier, seeding AReXS's reconciliation attempt with small domain- or task-specific ontologies should enhance its results. Other domain or general knowledge such as types of data would allow AReXS to explore the most likely matches first, increasing its efficiency greatly. Although we have not as yet conducted a comprehensive analysis, experiments with smaller sample sets (50, 25 and 10 examples per information source) have produced less consistent results as individual variations in the sample data have greater impact on the reconciliation process. We intend to look further for a correlation between the sample size and the accuracy of AReXS' results. Also, some data are clearly more easily reconciled than others; what is not yet clear and deserves further attention is how to identify how useful given data will be prior to reconciling.

It seems intuitive that, for example, the uninformed reconciliation done by AReXS could be significantly enhanced by using the knowledge contained in any supplied ontology, schema or DTD as an advantageous starting point for the reconciliation effort. Thesaurus projects such as WordNet [2] can provide ready-made synonym tables, enabling the space of possible concept matches that must be searched to be greatly reduced, or at least for the most interesting and profitable areas to be searched first. Basic grammatical or formatting knowledge could be similarly employed, as could lists or ontologies of manufacturers and products. Similarly, the instance-based approach employed by AReXS could augment current approaches to merging and reconciling structured ontologies, particularly if instances or exemplars are include along with concepts, as is the case in the popular ontology editing tool Protégé [3], or as a supporting method for semi-automated reconciliation solutions such as PROMPT [9] or Chimaera [8]. Finally, general knowledge databases such as Cyc [1] should allow smarter elimination of inappropriate matches. We intend to investigate these methods further in the future.

Acknowledgements

This paper was supported by an ARC Discovery Grant, Multi-Ontologies meet UML: Improving the Software Engineering of Multi-Agent Systems. Thanks to the members of the Intelligent Agent Laboratory at the University of Melbourne for many useful discussions.

References

1. Cycorp, Inc. *www.cyc.com.*

2. Fellbaum, C. (ed.) *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, America, 1998.
3. Gennari, J., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., Noy, N. F., Tu, S. W. *The Evolution of Protégé: An Environment for Knowledge-Based Systems Development*. Stanford University, Technical Report SMI-2002-0943, 2002.
4. Hou, D. *Automatic Reconciliation of XML Structures*. Honours thesis, Department of Computer Science and Software Engineering, The University of Melbourne, 2001.
5. Ikeda, Y., Itoh, F., and Ueda, T. *Example-based Frame Mapping for Heterogeneous Information Agents*. In Proceedings of the International Conference on Multi-Agent Systems, Paris, France, 1998.
6. Lister, K., Sterling, L. *Agents in a Multi-Cultural World: Towards Ontological Reconciliation*. In M. Stumptner, D. Corbett, M. Brooks (eds), Advances in Artificial Intelligence, Proceedings of the Fourteenth Australian Joint Conference on Artificial Intelligence, Springer-Verlag Lecture Notes in Artificial Intelligence, Vol. 2256, pp 321-332, 2001.
7. Lister, K., Sterling, L. *Reconciling Ontological Differences for Intelligent Agents*. In P. Bouquet (ed), Meaning Negotiation, AAAI Technical Report WS-02-09, Eighteenth National Conference on Artificial Intelligence, Edmonton, Canada, 2002.
8. McGuinness, D., Fikes, R., Rice, J., Wilder, S. *An Environment for Merging and Testing Large Ontologies*. In Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning, Breckenridge, America, 2000.
9. Noy, F., Musen, N. *An Algorithm for Merging and Aligning Ontologies: Automation and Tool Support*. In Proceedings of the Workshop on Ontology Management at the Sixteenth National Conference on Artificial Intelligence, Orlando, America, 1999.
10. Sato, S. *CTM: An example-based translation aid system*. In Proceedings of the Fifteenth International Conference on Computational Linguistics, Nantes, France, 1992.

OMEN: A Probabilistic Ontology Mapping Tool

Prasenjit Mitra[1], Natalya F. Noy[2], Anuj R. Jaiswal[1]

[1] The Pennsylvania State University, University Park, PA 16802, U.S.A.,
{pmitra,ajaiswal}@ist.psu.edu
[2] Stanford University, Stanford, CA 94305, U.S.A.,
noy@smi.stanford.edu

Abstract. Most existing ontology mapping tools do not provide exact mappings. Rather, there is usually some degree of uncertainty. We describe a framework to improve existing ontology mappings using a Bayesian Network. OMEN, an Ontology Mapping ENhancer uses a set of meta-rules that capture the influence of the ontology structure and the semantics of ontology relations and matches nodes that are neighbors of already matched nodes in the two ontologies. We have implemented a protype ontology matcher that can enhance existing matches between ontology concepts. Preliminary experiments demonstrate that OMEN successfully identifies and enhances ontology mappings.

1 Introduction

Information sources, even those from the same domain, are heterogeneous in nature. The semantics of the information in one source differs from that in another. In order to enable interoperation among heterogeneous information sources or to compose information from multiple sources, we often need to establish mappings between database schemas or between ontologies. These mappings capture the semantic correspondence between concepts in schemas or ontologies.

In recent years, researchers have developed a number of tools for finding these mappings in a semi-automated fashion (see Section 7 for a brief overview). In addition, there are interactive tools that enable experts to specify the mappings themselves. However, in most cases, the mappings produced are imprecise. For instance, automatic ontology-mapping tools can rank possible matches, with the ones that are more likely to be correct getting higher rankings. Most automatic ontology-mapping tools use heuristics or machine-learning techniques, which are imprecise by their very nature. Even experts sometimes could be unsure about the exact match between concepts and typically assign some certainty rating to a match. Once a particular set of mappings is established (by an expert or a tool), we can analyze the structure of ontologies in the neighborhood of these mappings to produce additional mappings.

Our main premise in this work is the following: if we know a mapping between two concepts from the source ontologies (i.e., they *match*), we can use the mapping to infer mappings between related concepts. For example, if two properties and their domains match, then we can infer (with some certainty) that

their ranges may be related as well. We build a Bayesian Net with the concept mappings. The Bayesian Net uses a set of *meta-rules* based on the semantics of the ontology relations that expresses how each mapping affects other related mappings. We can use existing automatic and semi-automatic tools to come up with initial probability distributions for mappings. Next, we use this probability distribution to infer probability distributions for other mappings.

We have implemented a tool called OMEN (Ontology Mapping ENhancer). OMEN uses a Bayesian Net and enhances existing ontology mappings by deriving missed matches and invalidating existing false matches. Our preliminary results show that by using OMEN we can enhance the quality of existing mappings between concepts across ontologies.

The primary contributions of this paper are as follows:

1. We introduce a probabilistic method of enhancing existing ontology mappings by using a Bayesian Net to represent the influences between potential concept mappings across ontologies.
2. In OMEN, we provide an implemented framework where domain knowledge of mapping influences can be input easily using simple meta-rules.
3. We demonstrate the effectiveness of OMEN in our preliminary experiments.

To the best of our knowledge, no existing work has extensively used a probabilistic representation of ontology mapping rules and probabilistic inference to improve the quality of existing ontology mappings.

2 Knowledge Model

We assume a simple ontology model (similar to RDF Schema). We use the following components to express ontologies:

Classes Classes are concepts in a domain, organized in a subclass–superclass hierarchy with multiple inheritance.

Properties Properties describe attributes of classes and relationships between classes. Properties have one or more **domains**, which are classes to which the property can be applied; and one or more **ranges**, which restrict the classes for the values of property.

We use the following notation conventions through the rest of this paper:

- all concepts from O have no prime (’); all concepts from O' have a prime (’);
- upper-case C with or without a subscript is a class;
- lower-case q with or without a subscript is a property;
- $P(C_1\ \theta\ C_2, x)$ indicates that the probability of the match $(C_1\ \theta\ C_2)$ is x.

3 Construction of the Bayesian Net

We now discuss how the Bayesian Net is constructed.

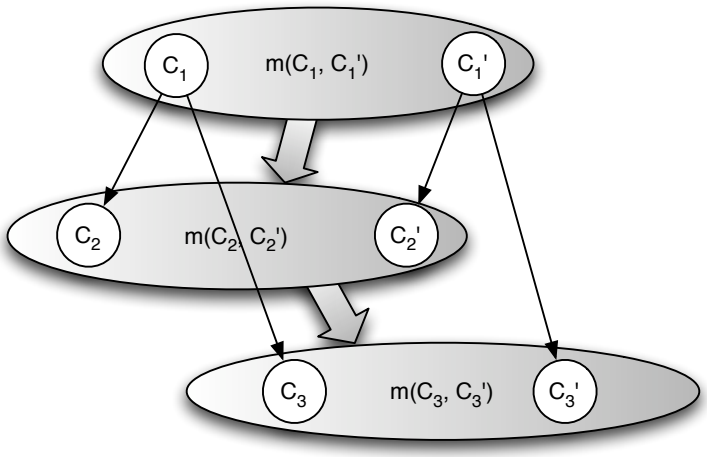


Fig. 1. Subgraphs representing some concepts in ontologies O and O' (small circles) and relations between them (thin arrows). The large gray ovals and solid arrows represent a snippet of the BN graph with nodes corresponding to matches and arrows corresponding to influences in the BN graph.

3.1 The BN-Graph

Nodes in the BN-graph represent individual pairs of matches. Consider Figure 1. This figure represents some classes in ontology O in the left-hand tree and some classes in ontology O' in the right-hand tree. The thin arrows in the figure are relationships between classes in the ontology, such as subclass–superclass relationships. The gray nodes and arrows represent the BN graph superimposed on the graphs representing ontologies. Nodes in the BN graph are matches between pairs of classes or properties from different ontologies. Arrows in the BN graph (the solid gray arrows in Figure 1) represent the influences between the nodes in the BN graph. The Conditional Probability Tables (CPTs) represent how a probability distribution in one node in the BN graphs affects the probability distribution in another node downstream from it. For example, in Figure 1, the mapping between concepts C_1 and C_1' affects the mapping between concepts C_2 and C_2', which in turn affects the mapping between C_3 and C_3'

A Scalable Selection of Nodes If we create a node for all possible pairings of concepts in two ontologies, the number of nodes in the BN-graph grows quadratically with respect to the number of nodes in the source ontologies. For example, if the sizes of the ontologies are 100 nodes each, the BN-graph will have 10,000 nodes. However, most of these nodes will express matches that are extremely unlikely to hold, because an evidence node will not influence a node that is distant from an evidence node significantly. Therefore, for performance, reasons, it

makes sense to prune the BN-graph. We generate all possible nodes in the BN-graph that are at a maximum distance of k from an evidence node. The value of k is tuneable by the expert running the system, but empirically, we found a small value like $k = 1$ or $k = 2$ suffices. Larger values of k make very little difference to the result but increase the size of the Bayesian Net significantly.

Anther factor that effects the size of the BN-graph is the number of parents (i.e., nodes that influence the match) that each node has. For example, if a concept C has 5 parents, and C' has 8 parents, the node representing a match between C and C' would have 40 parent nodes in the BN-graph. As we discuss in the next section, the size of a CPT is exponential with respect to the number of parents of a node. Therefore, generating the CPT would cost 2^40 units of computation. Even if the computation is very small, this number is exceedingly large and very soon makes the Bayesian Net unweildy. Thus, we restrict the maximum number of parent nodes for a single node to 10. We choose these 10 parents by selecting the top 5 parents with the maximum *a priori* probability and the top 5 parents with the minimum *a priori* probability.

If the Bayesian Net is constructed by adding edges such that matching ancestor nodes in an ontology influence the children nodes, we refer to this method as the "down-flow" method. A method where the Bayesian Net edges are constructe such that matching descendant nodes influence their ancestors, we call the method a "top-down" method. In case the ontologies contain cycles and this introduces cycles in the BN-graph, the algorithm breaks cycles in the BN-graph by rejecting the edges from the parents whose matching information is minimum (confidence score near 0.5).

3.2 Evidence and CPTs

In order to run a Bayesian Net we need to provide it with two types of information: (1) evidence (obtained from the initial probabilities) describing what we already know with high confidence, and (2) Conditional Probability Tables, describing how the parent nodes influence the children in the BN-graph.

The input to the OMEN algorithm consists not only of the two source ontologies to be matched, but also, of the initial probability distributions on the "root" nodes (nodes with no parents) in the BN graph. Note that our definition of mapping allows for inputs that are themselves imprecise and contain some probability values. For instance, if there is an ontology matcher that produces a set of pairs of matches ordered according to the algorithm's certainty about the match we can translate that into specific values for each $m(C_n, C'_k)$ where the probability value for "=" is less than 1 and diminishes as we go further down in the ranked list of the external matcher's result.

The final missing piece are the CPTs. The CPTs describe how a match between two classes affects other matches (these are the solid gray arrows in Figure 1). For example, a match between two classes from the source ontologies affects the match between their superclasses. Or a match between properties affects the match between their domains. These rules depend on the knowledge model and semantics of the relationships (such as subclass or domain) defined

in the knowledge model. Therefore, we have developed a set of generic *meta-rules* that enable us to generate CPTs for each particular pair of ontologies automatically. In fact, our implementation is parameterized with respect to the meta-rules, and we can add or remove meta-rules to evaluate which ones work best for a particular knowledge model.

We present some of the meta-rules that we used in the next Section.

The following summarizes the OMEN algorithm:

- Input: source ontologies O and O', initial probability distribution for matches
- Steps:
  1. If initial probability of a match is above a given threshold, create a node representing the match and mark it as evidence node.
  2. Create nodes in the BN graph representing each pair of concepts (C, C') , such that $C \in O$ and $C' \in O'$ as a node in the graph and the nodes are within a distance k of an evidence node
  3. Create edges between the added nodes
  4. Use the meta-rules to generate CPTs for the BN
  5. Run the BN
- Output: a new set of matches

In the next section we discuss how Step 1 above can be modified to prune out unnecessary nodes.

4 Meta-rules for Generating New Probability Distributions

In this section, we show examples of meta-rules that are used to match the ontologies and discuss how the algorithm generates new probability distributions depending upon the existing ones.

4.1 Examples of Meta-rules

The following is one of the basic meta-rules we used in our implementation: if two concepts C_1 and C_1' match, and there is a relationship q between C_1 and C_2 in O and a matching relationship q' between C_1' and C_2' in O', then we can increase the probability of match between C_2 and C_2'. Informally, if two nodes in an ontology graph match and so do two arrows coming out of these nodes, then the probability that nodes at the other end of the arrows match as well is increased. In the formal rule below we generalize this meta-rule to any relationship θ between C_1 and C_1', not just match.

$$P(C_1\ \theta\ C_1', x) \wedge P(q = q', 1) \wedge q(C_1, C_2) \wedge q'(C_1', C_2') \Rightarrow P(C_2\ \theta\ C_2', min(1, x + \delta) \quad (1)$$

where δ is an expert-provided constant less than 1. We use a similar meta-rule for the case where relationships q and q' do not match (i.e., for arbitrary pair of outgoing edges), but subtract *delta* from x in the consequent.

While not in our initial implementation, other meta-rules rely more heavily on the semantics of the components in the ontology language. Below are some informal examples of such rules.

Mappings between properties and ranges of properties: If two properties match, and each of them has a single range, we can increase the probability of match between the classes representing the range. Similarly if two properties q and q' match and the range of q is a union of classes C_1 and C_2, and the range of q' is a class C', then the tool can increase the probability that C_1 is a specialization of C' and C_2 is a specialization of C'.

Mappings between superclasses and all but one sibling: In this case, we say that the existing matches between the superclasses and the matched siblings result in the remaining siblings matching with high probability.

We experimented with three different ways of generating the CPTs for the nodes in a BN graph:

1. Fixed Influence Method (FI): The meta-rules state that the probability of the children matching depends upon whether the parents match and is given by a set of constants. An example of such a rule is:

$$P[C_p = C'_p, x] \wedge x > t_{max} \wedge q(C_p, C_c) \wedge q(C'_p, C'_c) \Rightarrow P[C_c = C'_c, 0.9] \quad (2)$$

   where t_{max} is an expert-defined threshold value. There are similar rules for the other cases.
2. Initial Probability Method (AP): The meta-rules state that the probability distribution of a child node is affected depending upon the probability distribution of the parent node by a set of constants. An example of this class of meta-rules is:

   $P(C_1\ \theta\ C'_1, x) \wedge P(q = q', 1) \wedge q(C_2, C_1) \wedge q'(C'_2, C'_1) \wedge P[C_2\ \theta C'_2, y] \wedge (y > t_{max})$
   $\Rightarrow P(C_1\ \theta\ C'_1, min(1, (x + \delta))$

   where t_{max} and δ are expert-provided constants less than 1.
3. Parent Probability Method (PP): The meta-rules state that the probability distribution of the child node is derived from the probability distribution of the parent node using a set of constants. Rule 1 is an example of a meta-rule used in this method.

The algorithm must combine probabilitic influences of different rules and determine the probability distribution of a mappings. For example, consider a pair of classes, C and C' (Figure 2). In the example in the figure, the following mappings can affect the probability that they match (depending on a specific set of meta-rules used):

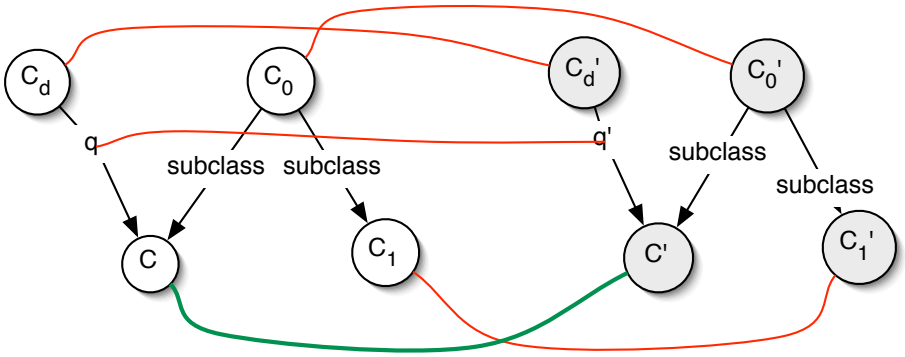


Fig. 2. The probability distribution for the mapping between C and C' is affected by the mappings between their superclasses, siblings, and domains of the properties q and q' for which C and C' are ranges.

- A mapping between superclasses of C and C'
- Mappings between the siblings of C and C'
- A mapping between properties q and q' ($P(q = q', 1)$) for which C and C' are ranges respectively, and mappings between domains of q and q' ($P(C_d = C'_d, z)$).

In this work, we combine probabilitistic influences as follows. If a child in the Bayesian Net (not the ontologies) has two parents, we combine the conditional probability distributions of the child on each parent using the assumption that the two parents are independent. That is,

$$P[C|A, B] = P[C|A] \times P[C|B] \tag{3}$$

In cases, where the match of two parents influence each other, this assumption is not true. However, empirically, even with this simplifying assumption we have obtained encouraging results. A more sophisticated method of combining influences is left for future work.

5 Experimentation and Results

OMEN uses BNJ, Version 3.0 pre-Alpha 3 [1] as its probabilistic inference engine.

We used two ontologies obtained from the Knowledge Representation and Reasoning group at the Vrije University.[3] The ontologies are expressed in RDF using RDF-Schema. They contain concepts related to university departments and students, staff and faculty of the departments.

[3] `http://wbkr.cs.vu.nl/`

For our experiments, we extracted portions of the ontologies manually to make sure that they have at least some overlap. Because matching predicates is beyond the scope of this tool, we matched predicates across the ontologies by manual examination. When we decided that two predicates represented the same relationship, the names of one predicate was replaced by the names of the matching predicate in the other.

If the generated Bayesian Net contained a cycle, we manually weeded out one edge choosing one at random from those edges that are between nodes that are farthest from the root.

For this preliminiary experimentation, instead of using several values as cited in Section 2 that a node in the Bayesian Net can have, we just assigned two values "true" and "false" to the nodes. A value of "true" represents that the concepts represented by the node match and vice-versa.

We generated initial probability distribution for matches using a simple script using string-edit distance. Recall that in practice these probability distributions can come from any other tool. To generate probability distributions for the experiment, we manually identified i) the nodes across the ontolgies that the matched and ii) the nodes that surely did not match. The matching nodes were assigned a random probability between 0.7 to 0.9 and the nodes that did not match assigned a random number between 0.1 to 0.4 with the rest of the nodes lying in between 0.4 to 0.7.

We fixed the threshold values to 0.85 and 0.15. That is, if the probability of a match is determined by our methods or by the previous method to be greater than 0.85, then we determine that the concepts match and if the probability is less than 0.15, then we declare that the concepts do not match. Such matches and mismatches are used as evidence to the Bayesian NetḞurthermore, the same threshold value is used to determine a match from the posterier probability generated by OMEN. In some cases, the threshold was taken to be too stringent and resulted in lower recall. As future work, we intend to look into dynamically selecting proper thresholds by clustering.

We experimented with two sets of ontology graphs. In the first set, both graphs had 11 nodes each and in the second case both had 19 nodes. The preliminary results that we obtained are given in the two tables below:

Table 1. Summary of results for the smaller ontologies

Case No.	CPT-Method	Precision	Recall	F-measure
1	FI	0.75	0.375	0.5
	AP	1.0	0.5	0.67
2	FI	1.0	0.5	0.67
	AP	1.0	0.875	0.933
3	AP	1.0	0.75	0.85
4	AP	1.0	0.125	0.22

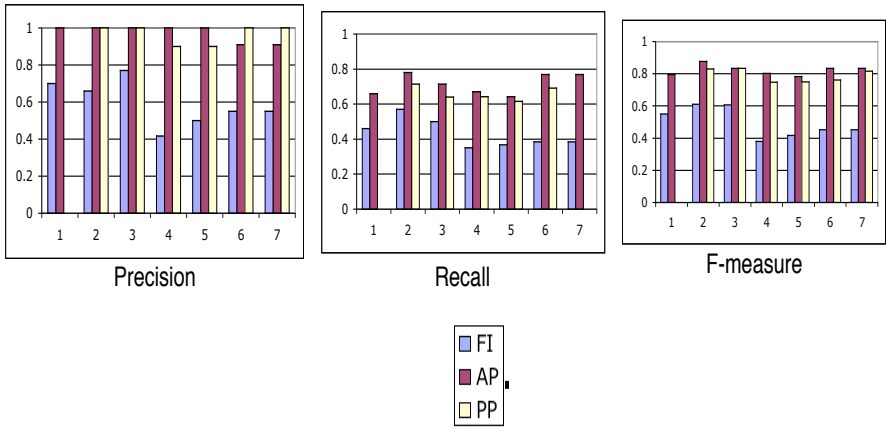


Fig. 3. Results for the ontologies of size 19 nodes

Table 1 lists the results for the case where the source ontologies were of size 11 nodes each. In this case, we specified three matching nodes as positive evidence and four pairs of nodes that do not match as negative evidence. The precision, recall and f-measure are calculated in the usual Information Retrieval (IR) sense using both the positive match and the negative match results together. In case 1, the evidence was introduced at random points. In case 2, the evidence was introduced at or very near the leaf nodes. The results show that introducing the evidence at or near the leaf nodes increased the performance of the algorithm. Case 3 is very similar to Case 2, but with false evidence introduced. Case 4 shows the effect of introducing drastic errors in the initial probabilities. Since the CPTs in the AP method depend directly on the quality of the initial probabilities, when we assigned the initial probabilities at random intentionally, the quality of the results deteroriate.

Overall, we see that the AP method outperforms the FI method in both cases. We also see that by giving only 3 matches out of 11, we could generate upto 7 of the missing matches. This implies that the method can be very useful even when the results of the previous matcher is not very good as not as it is totally random.

We show the results for the case where the source ontologies contained 19 nodes each in Figure 3. The figure shows 7 different test cases for the three different CPT-generating methods. The FI method is shown first, followed by the AP method, followed by the PP method for each test case.

The evidence provided in the cases above were as follows: For case 1, we provided positive evidence of 4 matches at or near the leaf nodes. For cases 2, 3, and 4, we provided positive evidence of 5 matches and negative evidence of 4 matches. For cases 5, 6, and 7 we provided positive evidence of 6 matches and negative evidence of 4 matches. In cases 2, and 3 the evidence was also provided

at or near the leaf nodes. In cases 4, and 5, the evidence was provided at or near the root nodes. In cases 6 and 7, the evidence was provided at randomly selected nodes. In cases 3, 5, and 7, wrong evidence was introduced.

Not surprisingly, we see that both the AP and the PP methods outperforms the FI method of constructing CPTs and provide good precision and recall values. The AP method slightly outperforms the PP method in general. However, the PP method is more stable, that is, it recovers from a few wrong evidences better than the AP method. In this case, the place where the evidence was introduced did not matter much for the AP and PP methods.

6 Future Work

In the future, we intend to perform experiments to determine whether a system based on up-flow rules outperform one based on down-flow rules and to identify the scenarios when one outperforms the other. Furthermore, we will extend our algorithm to perform multiple iterations on the data. For example, we can employ alternate iterations using down-flow and up-flow rules for a fixed number of iterations or until the results converge. Designing better CPTs using more of the semantics of the ontology relationships, and emperically evaluating them and experimenting with large ontologies and coupling our matcher with various external matchers are charted for as future work.

7 Related Work

Two research directions are related to our work: automatic or semi-automatic discovery of ontology mappings and the use of uncertainty in knowledge-based systems.

7.1 Automatic ontology mapping

Over the past decade, researchers have actively worked on developing methods for discovering mappings between ontologies or database schemas. These method employ a slew of different techniques. For example, Similarity Flooding [8] and AnchorPrompt [10] algorithms *compare graphs* representing the ontologies or schemas, looking for similarities in the graph structure. GLUE [3] is an example of a system that employs *machine-learning techniques* to find mappings. GLUE uses multiple learners exploiting information in concept instances and taxonomic structure of ontologies. GLUE uses a probabilistic model to combine results of different learners. Hovy [5] describes a set of heuristics that researchersat ISI/USC used for semi-automatic alignment of domain ontologiesto a large central ontology. Their techniques are based mainly on*linguistic analysis* of concept names and natural-languagedefinitions of concepts. A number of researchers propose *similarity metrics* between concepts in different ontologies based on their relations to other concepts. For example, a similarity metric between concepts in

OWL ontologies developed by Euzenat and Volchev [4] is a weighted combination of similarities of various features in OWL concept definitions: their labels, domains and ranges of properties, restrictions on properties (such as cardinality restrictions), types of concepts, subclasses and superclasses, and so on. Finally, approaches such as ONION [9] and Prompt [11] use a combination of *interactive specifications* of mappings and *heuristics* to propose potential mappings.

The approach that we describe in this paper is complementary to the techniques for automatic or semi-automatic ontology mapping. Many of the methods above produced pairs of matching terms with some degree of certainty. We can use these results as input to our network and run our algorithm to improve the matches produced by others or to suggest additional matches. In other words, our work complements and extends the work by other researchers in this area.

7.2 Probabilistic knowledge-base systems

Several researchers have explored the benefits of bringing together Nayes Nets an knowledge-based systems and ontologies. For instance, Koller and Pfeffer [7] developed a "probabilistic frame-based system," which allows annotation of frames in a knowledge base with a probability model. This probability model is a Bayesian Net representing a distribution over the possible values of slots in a frame. In another example, Koller and colleagues [6] have proposed probabilistic extensions to description logics based on Bayesean Networks.

In the context of the Semantic Web, Ding and Peng [2] have proposed probabilistic extensions for OWL. In this model, the OWL language is extended to allow probabilistic specification of class descriptions. The authors then build a Bayesean Network based on this specification, which models whether or not an individual matches a class description and hence belongs to a particular class in the ontology.

Researchers in machine learning have employed probabilistic techniques to find ontology mappings. For example, the GLUE system mentioned earlier [3], uses a Bayes classifier as part of its integrated approach. Similarly, Prasad and colleagues [12] use a Bayesean approach to find mappings between classes based on text documents classified as exemplars of these classes. These approaches, however, consider instances of classes in their analysis and not relations between classes, as we do. As with other approaches to ontology mapping, our work can be viewed as complementary to the work done by others.

8 Conclusion

We have outlined the design and implementation of Omen, an ontology match enhancer tool, that improves existing ontology matches based on a probabilistic inference. This tool is dependent upon a set of meta-rules which express the influences of matching nodes on the existence of other matches across concepts in source ontologies that are located in the proximity of the matching nodes. We described how we implemented a simple first version of the matching tool and

discussed our preliminary results. We have also outlined several improvements that can be made to the tool and identified several open questions that if resolved can make the performance of the tool even better.

References

1. Bayesian network tools in java(bnj), version 2.0, July 2004.
2. Z. Ding and Y. Peng. A probabilistic extension to ontology language owl. In *37th Hawaii International Conference On System Sciences (HICSS-37)*, Big Island, Hawai, 2004.
3. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *The Eleventh International WWW Conference*, Hawaii, US, 2002.
4. J. Euzenat and P. Valtchev. Similarity-based ontology alignment in OWL-Lite. In *The 16th European Conference on Artificial Intelligence (ECAI-04)*, Valencia, Spain, 2004.
5. E. Hovy. Combining and standardizing largescale, practical ontologies for machine translation and other uses. In *The First International Conference on Language Resources and Evaluation (LREC)*, pages 535–542, Granada, Spain, 1998.
6. D. Koller, A. Levy, and A. Pfeffer. P-Classic: a tractable probabilistic description logic. In *14th National Conference on Artificial Intelligence (AAAI-97)*, 1997.
7. D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin, 1998. AAAI Press.
8. S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *18th International Conference on Data Engineering (ICDE-2002)*, San Jose, California, 2002. IEEE Computing Society.
9. P. Mitra, G. Wiederhold, and S. Decker. A scalable framework for interoperation of information sources. In *The 1st International Semantic Web Working Symposium (SWWS'01)*, Stanford University, Stanford, CA, 2001.
10. N. F. Noy and M. A. Musen. Anchor-PROMPT: Using non-local context for semantic matching. In *Workshop on Ontologies and Information Sharing at the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)*, Seattle, WA, 2001.
11. N. F. Noy and M. A. Musen. The PROMPT suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003.
12. S. Prasad, Y. Peng, and T. Finin. A tool for mapping between two ontologies using explicit information. In *AAMAS 2002 Workshop on Ontologies and Agent Systems*, Bologna, Italy, 2002.

XeOML: An XML-based extensible Ontology Mapping Language*

Maria Teresa Pazienza[1], Armando Stellato[1], Michele Vindigni[1],
Fabio Massimo Zanzotto[2]

[1] DISP- University of Rome Tor Vergata
Via del Politecnico, 1 – 00133 Roma ITALY
{pazienza, stellato, vindigni, zanzotto}@info.uniroma2.it
[2] DISCO - University of Milano Bicocca
P.zza Ateneo Nuovo 1 – 20126 Milano ITALY
zanzotto@disco.unimb.it

Abstract. Semantic Interoperability is a crucial issue in the Semantic Web context: web services and portals, providing real-time access to widely distributed information sources, need to overcome problems due to heterogeneities in the use of distinct locales, languages and idioms. Though standardization efforts for semantic content representation are converging toward some concrete standards, still a lot of work is required to achieve real interoperability over knowledge content as managed by communities of autonomous and distributed individuals. An unavoidable trade-off between coverage of heterogeneous information sources and achievement of common semantics for accessing their content emerges. In this framework, we have carried out our research to develop an extensible language (XeOML) for describing mappings between domain ontologies, in which knowledge representation formalisms and similarity measures can be dynamically added according to community needs and intent.

1 Introduction

The goal of granting semantic accessibility to the web content pursued by the Semantic Web [6] can be achieved through ontologies, as they play a crucial role in supporting the exchange of data, in providing a formal vocabulary for the information and in unifying different views of a domain in a safe cognitive approach [9].

Despite all the active researches on ontology reuse, the inherently decentralized nature of the WWW pushes for a multitude of autonomously conceived choices, where different communities adopt their locally developed ontologies to represent their own knowledge. Different solutions to the knowledge sharing issue have been proposed and experimented: each of them is related to a different framework, thus suggesting a coarse classification with respect to the constraints they impose on the way knowledge must be represented and structured. Many existing information systems exploited both as research results (TSIMMIS [1], Information Manifold [2], Infomaster

* This work has been partially funded by the EU project MOSES IST-2001-37244.

[3], MOMIS [7]) or industrial solutions (Xyleme[1]) proposed centralized systems of mediation between users and distributed data sources, which exploit mappings between a single mediated schema and the local schemas. Other approaches (such as Mafra [8]) followed a more flexible solution based on distributed mediation systems. These systems generally include static representations of the relationships that bind different and distributed knowledge resources, or in some cases rely on the behavior of underlying communities of software agents [9,10,12] to dynamically negotiate the meaning of both concepts and relations from the different ontologies.

We will not analyze here the pros and cons of such approaches, being out of the scope of this work; by looking at the scenarios depicted above, we will highlight that, whichever the situation to be considered, either sharing knowledge between myriads of (couples of) independent ontologies or linking every local ontology to a centralized one, a key issue for the completion of real knowledge interoperability is represented by the definition of appropriate mappings between ontologies. We will focus on such a vital problem for any application scenario.

We refer to the ontology mediation activity as the process of reconciling differences among different information sources (and their schemas), to achieve interoperability between several applications and their underlying annotated data. This activity includes "discovery" of ontology mappings, that is, of declarative specifications of the semantic overlap between two (or more) ontologies. The mappings can broadly vary depending on the tasks they will support: different scenarios could require either *injective (*specifying how to go from a source to a target ontology) or *bijective* (stating equivalences among concepts and relationships in both ontologies) correspondences, different accuracy in establishing semantic similarities, and different levels of coverage of the mapped information sources. These differences are often underestimated in literature where most researches are devoted to define languages that completely state correspondences between entities, mixing together declarative and operational aspects of this task.

We argue instead that several factors interact in real world scenarios: complex mappings and reasoning capabilities are both necessary for comparing and combining ontologies and for integrating the data they describe. A big effort has been devoted in defining knowledge representation languages powerful enough to express the different views of a domain. OWL [14] has recently been accepted as a W3C recommendation for the representation of ontologies on the Web, and this represents an important step towards the realization of the Semantic Web vision. Far beyond the standardization of knowledge representation however, still remains the problem of reaching semantic consensus at content level. In the context of an open environment, what really happens is that many different heterogeneous ontologies with overlapping domains exist and may be shared by several partners of the communication.

To cope with such heterogeneity there is a need for tools and languages to formally and explicitly specify ontology mappings in order to achieve the desired interoperability. As pointed out in [15], OWL offers limited support for these mappings through the import statement that is used to import an ontology into another one: after importing, relationships among concepts in the different ontologies can be specified through equivalence and subsumption axioms.

[1] http://www.xyleme.com/

However, the mechanisms provided by OWL can reveal unsatisfactory for mapping specification even in the general case. OWL promotes a tight coupling between ontologies, as it makes dependent the importing ontology on the imported one. In a dynamic scenario, where ontologies should be updated to reflect changes in the domain, this kind of dependence does not allow for a flexible knowledge organization, and can result in a severe loss of consistency that usually is very difficult to be amended. Moreover, in [15] is also argued an epistemological inadequacy of OWL as a mapping language, because Description Logic constructs in OWL are useful for describing merged ontologies while general ontological mappings are not supported.

In [11] a significant extension to the OWL model has been proposed in the form of C-OWL, a language which allows for the representation of "contextual ontologies", intended as OWL ontologies embedded in a space of other OWL ontologies and related to them via context mappings [5]. Inside that work, five bridging rules, which account for four levels of similarity (identity, generalization/specification, compatibility and orthogonality) are defined to map concepts between different ontologies. The above rules do not take into account, however, of the really complex relationships which may hold between ontological entities of different types or even involving complex structures of entities from any of the mapped resources. A trade-off between generality and adaptivity of the proposed mapping model on the one side, and accurateness and completeness towards every possible contingency is however hard to balance. To this end, a deeper introspection inside the knowledge representation models which are mostly adopted nowadays and the factorization and formalization of the recurring constructs is throughout necessary, in order to obtain a language which can be tuned to different situations, still maintaining the integrity of its underlying fabric.

In the remainder of this paper, we present a novel mapping specification language, XeOML, that bases on a layered approach to define a well formed declarative representation of complex structural correspondences between the entities involved in a mapping process. Formal semantics of the core layer of the language is intentionally omitted, as we aim to keep it separated from the more structural aspects of the problem. The syntax of the language has in fact been chosen to make possible an incremental specification of most accurate correspondences, in order to leave such details to the specific situations arising in application dependant contexts (where the actual reasoning capabilities and the expressiveness of the formalisms used to represent the knowledge become clear, making easier to define more effective ways to deal with such aspects).

2 XeOML: An XML-based extensible Ontology Mapping Language

XeOML is an extensible language for describing ontology mappings, developed at the University of Roma Tor Vergata and adopted by first for the ontology mapping task inside the European project Moses[2]. As its acronym suggests, it is based on XML

[2] Examples in the rest of the paper refer to MOSES project that deals with a distributed multilingual Q/A system (based on ontological knowledge) accessing to different university web sites.

syntax, taking advantage from its expressive power to offer a core language characterized by easy machine-readability and high extensibility.

XeOML is defined by an XML schema[3], *AbstractMapping*, which provides information for describing mappings between ontologies, detailing the structure of a mapping document and defining the set of elements that populate an ontology.

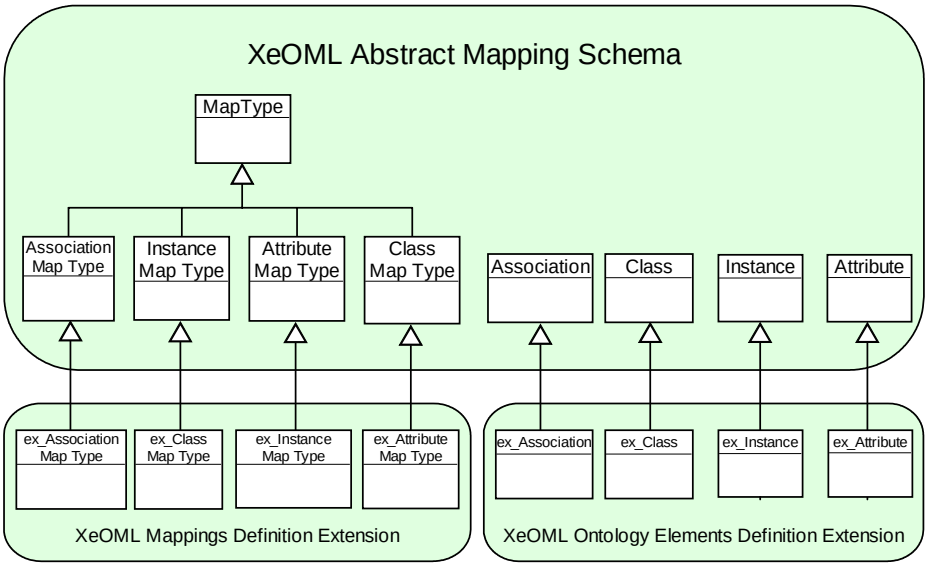


Fig. 1 An overview of the Abstract Mapping Schema and its Extensions

The *AbstractMapping* schema (Fig. 1) thus defines a core language for ontology mapping representation, voluntarily ignoring details on different levels of mapping relationships which may be considered among ontology elements and on the semantics associated to them: what is clearly asserted and organized in this schema, is the declaration and classification of typical *mapping patterns* that may involve complex structures of entities from the ontologies to be mapped. More semantically declarative information may thus be plugged to the main schema in the form of XML Schema extensions, which reflect different perspectives and approaches to the mapping process and/or heterogeneous knowledge representation styles; the core schema, together with its extensions, forms a complete mapping document definition.
The two extensions which need to be provided are:

- an *Ontology Elements Definition* schema extension, which accounts for specification of ontological elements according to a given representation language
- a *Mappings Definition* schema extension, where ad-hoc descriptions of the level of mappings that may be considered and agreed inside a particular framework may be specified.

[3] XeOML Abstract Mapping Schema and its extensions are freely available for download at: http://ai-nlp.info.uniroma2.it/xeoml

Thanks to this approach, the agents that want to exchange knowledge inside a distributed framework may rely on the same basic functionalities for interpreting the core language (thus favoring reuse of existent technologies) and need only to be "tuned" to the extensions adopted inside their community, in order to capture and exploit the committed semantics for both recognizing and ranking ontology mappings.

In Fig. **1** a partial overview of the *XeOML Abstract Mapping* Schema is given, reporting only the XML element types which are defined as abstract, thus needing to be implemented in the two schema extensions.

In the next paragraphs, a brief description of the structure of a mapping document, as implied by the *AbstractMapping* schema, will be given.

Mapping Terminology

As an ontology mapping language, XeOML foresees the presence of two target ontologies that need to be mapped. In the rest of the paper we will address these two ontologies as *Left Ontology* and *Right Ontology* (*LO* and *RO*, respectively). The *AbstractMapping* schema defines an ontology as composed of four different ontological entities: *instances*, *classes*, *properties* and *associations*; these elements reflect most of the more common ontological definitions, like those proposed by OWL [14], OKBC [4] or Topic Maps [13]. The syntax definitions for these four elements will be described in the *Ontology Elements Definition* schema, according to the model adopted to represent the knowledge content.

Two types of mappings are defined inside the *AbstractMapping* schema:

- *Simple Mappings* (or, simply, *mappings*), i.e. one-to-one relations between ontology elements of the same type.
- *Complex Mappings*, i.e. mappings involving even more than one element from one or both the ontologies; different ontology element types may be correlated into heterogeneous combinations, depending on the specific mapping relation.

We will use the term *Mapper*, indistinctly referring either to an automatic process for both producing ontology mappings or managing a meaning negotiation activity, or to a human annotator who will produce a manual mapping between the two ontologies

Mapping Structure

The structure of the mapping task is very complex in its nature. To obtain a uniform management of mappings between elements from the two ontologies, every ontology element from both *LO* and *RO* must always be included in a (simple) *mapping*, as this kind of mapping can be characterized by one the following:

- a *one-to-one* correspondence between two ontology elements
- a single element from one of the two ontologies and a reference to a complex map, meaning that the given element is involved in that complex mapping

An automatic process willing to know how an ontology element is mapped, only needs to inspect (in a uniform way) simple mappings, and, where necessary, be redirected towards a complex map; see **Ex. 1** where the participation of the "Professor" Class from *LO* in a complex map is reported.

```
<mapping xsi:type="absm:ClassMap" ID="c2">
    <MapRank xsi:type="map:ClassMapType">ExtensionalEquivalence</MapRank>
    <LeftMapped>
        <Class xsi:type="oed:OWLClass" ID="Professor"/>
    </LeftMapped>
    <RightMapped>
        <participationInMapping xlink:href="#cc1"/>
    </RightMapped>
</mapping>
```

Ex. 1 a class participating in a complex map

Notice the element inside the LeftMapped tag: it is a class as defined in OWL, because the *Ontology Elements Definition* schema extensions which implements OWL definitions has been adopted to represent this class in the example.

The Abstract Mapping Schema defines an abstract ComplexMapType (so that even this aspect can be extended to meet specific requirements) and its subclasses with some concrete types for diverse kind of Complex Mappings. We analyze here some of these types:

AttributeAggregationMap: it represents a map between one attribute from one of the two ontologies and more than one attribute from the other ontology.

```
<complex_mapping xsi:type="absm:AttributeAggregationMap" ID="cc2">
    <MapRank xsi:type="map:AttributeMapType">RangeEquivalence</MapRank>
    <MappingFuctional xsi:type="map:AttrAggMapType">StringConcat</MapRank>
    <LeftMapped xsi:type="oed:OWLDatatypeProperty" ID="name"/>
    <RightMappedAggregation>
        <Attribute xsi:type="oed:OWLDatatypeProperty" ID="name">
            <label xml:lang="en">first name</label>
        </Attribute>
        <Attribute xsi:type="oed:OWLDatatypeProperty" ID="surname">
            <label xml:lang="en">last name</label>
        </Attribute>
    </RightMappedAggregation>
</complex_mapping>
```

Ex. 2 aggregation of attribute values

An example of this map is given in **Ex. 2**, reporting string concatenation of more attributes into one, e.g. attribute *name* from a *LO* is mapped to the concatenation of *name* and *surname* from *RO*. Other cases could include attributes from one ontology whose ranges correspond to the union of the ranges of different attributes from the other. All these cases should be shown in the Mapping Definition extension and explicit semantics for handling them should be captured by agents responsible for ontology mediation activity.

ClassAggregationMap: in **Ex. 3** two classes coming from *LO*, "man" and "woman" are mapped to the class "human" from *RO*. This assertion implies that the classes "man" and "woman" can be considered as complete partitions of the class "human". ClassAggregation Complex Mappings should deal with this sort of relationship.

```
< complex_mapping xsi:type="absm:ClassAggregationMap" ID="cc1">
    <MapRank xsi:type="map:ClassMapType">ExtensionalEquivalence</MapRank>
    <LeftMappedAggregation>
        <Class xsi:type="oed:OWLClass" ID="woman"/>
        <Class xsi:type="oed:OWLClass" ID="man"/>
    </LeftMappedAggregation>
    <RightMapped xsi:type="oed:OWLClass" ID="human"/>
</complex_mapping>
```

Ex. 3 aggregation of Classes

Istance-ClassMap: very often, depending on the conceptualization of the world and on the objectives that lies behind the development of an ontology, the same concepts appears either in the form of a class or of an instance. Theorically, a class is a "set of instances" and could never be compared to an instance, as clearly motivated in [14]. On the other hand, this is not in line with several typical ontology modeling approaches where a concept is conceived as an instance or a class depending either on the given level of abstraction or on the task the ontology is thought for.

```
<complex_mapping xsi:type="absm:ClassWRestr-ClassMap" ID="crc1">
    <MapRank xsi:type="map:ClassMapType">ExtensionalEquivalence</MapRank>
    <LeftMappedClass xsi:type="oed:OWLClass" ID="automobile_rossa">
        <label xml:lang="en">red car</label>
    </LeftMappedClass>
    <RightMappedClassWithRestrictions>
        <Class xsi:type="oed:OWLClass" ID="car"/>
        <AttributeRestriction>
            <Attribute xsi:type="oed:OWLDatatypeProperty" ID="color"/>
            <Restriction>red</Restriction>
        </AttributeRestriction>
    </RightMappedClassWithRestrictions>
</complex_mapping>
```

Ex. 4 A class mapped to another class with a value restriction on one of its attributes

ClassWithRestrictions-Class and ClassWithRestrictions-Instance: these two kind of mappings deal with conceptual equivalences between classes (instances) and partitions of classes which depend on restrictions over the range of one or more of their attributes. In **Ex. 4**, the class "red_car" is extensionally equivalent (in the sense of: share the same instances) to the class car with a restriction on the range of its "color" attribute set to "red".

3 Extending the XeOML Schema: a case study

We hereafter describe, as an example of possible extensions, two schemas for the XeOML language, providing respectively:

- definitions for OWL ontology elements
- enumerated descriptions of possible distinct levels of conceptual similarity, classified depending on mapping type (i.e., the type of elements involved in a mapping)

OntologyElementsDefinition Schema: implementation for OWL

As a first possible extension to the XeOML language, we provided implementations for all of the XeOML abstract Ontology Elements in the form of OWL data types. All defined elements contain an ID attribute for specifying the ID of the concept from the ontologies, and an optional number of labels to represent these concepts in different languages, as defined for almost all OWL categories. We stress here that it has not always been a straight 1-1 mapping between XeOML abstract types and elements from the implemented model: in the OWL case, both OWL DataType properties and OWL Object properties have been mapped as XeOML Attribute types.

On the contrary, being Associations not explicated in OWL, they are represented in XeOML by normal OWL Classes (many knowledge representation languages do not allow for associations, being them mimed by classes, with attributes acting as roles of the association: this way of modeling is typically indicated as Association Classing).

The idea behind the extensible definitions of element types to different representation formalisms, is that a mediation activity involving two agents, requires them to be only proficient about the knowledge model adopted to express their underlying ontological resource while not necessarily being able to understand the model owned by the interlocutor. This way every agent could fully exploit the detailed semantics of its knowledge model, and leave as meaningless strings the concepts expressed for the other ontology in the mapping document, as they need only to be used as a transaction mechanism inside the mediation activity. The choice of allowing for detailed and language dependent descriptions of the ontological elements instead of neutral IDs (which could be of help in retrieving the same information from the source ontologies), may be questionable. However, although it is introducing redundancy inside the mapping document, it is indeed true that an agent exploiting this so-defined IDs would need the capability of matching them with elements from the source ontology (this may not be always trivial). Moreover, there could be many reasons to introduce more information in the ontology elements definition schema, which could be useful for making fast inference over large data from the mapping document as a whole, without the need for explicit reference to the source ontologies.

MappingsDefinition Schema

The *Abstract Mapping Schema* declares four types of mappings, related to the four basic ontology elements types: *InstanceMapType*, *ClassMapType*, *AttributeMapType* and *AssociationMapType*. A *MappingsDefinition* Schema Implementation should of-

fer enumerated restrictions to these types, assigning specialized semantics to the level of similarity between elements of the mapped ontologies. We have produced a *Default MappingDefinition* Schema extension, providing a few examples of possible level of mappings which could be reported in a mapping document. The intent of these mapping types is to specify at what extent the knowledge data (classes, attributes and instances) that is available in an ontology can be augmented with the foreign data contained in other ones.

DefaultInstanceMapType is a restriction of the generic *InstanceMapType* and foresees the following levels of similarity between ontology instances:

1. *Equivalence*: two instances are equivalent if they refer to the same object of the world. For example "President of USA Bush" and "George W. Bush" are, apart from their different surface forms, probably referring to the same person, intended as a unique individual (in the hypothesis we are not speaking of another person with the same name as the U.S. President!).
2. *Similarity*: the two instances represent very similar concepts, though cannot be considered, under all aspects, as totally overlapping.

DefaultClassMapType is a restriction of the generic *ClassMapType* and foresees four levels of similarity between ontology instances:

1. *ExtensionalEquivalence*: it is hard to tell if two concepts are totally equivalent; many a knowledge theory should even confute the notion of equivalence between concepts coming from two different agents (either automatic agents accessing ontologies to convey the meaning of their knowledge, or humans involving in discourse). Nevertheless, there are objects and individuals in the world which may be considered unique and to which unambiguously refer. If we consider the extension of a class as the list of objects/instances referred by it, then we may say that two classes share a *Extensional Equivalence* if they describe the same instances of the world. What can be inferred from such a mapping type is that, given two classes A and B, extensionally equivalent, every instance of A may be considered an instance of B too, and vice versa. In some cases, these instances may result to be super specified or under specified, depending on the intensional similarity of the two classes: if A has some attributes which have no equivalent in B, instances of B would be under specified wrt these characteristics; this is indeed a partial lack of knowledge that cannot be filled otherwise.
2. *IntensionalSimilarity*: this level of class-similarity holds when it is not sure two given classes share exactly the same instances, but indeed they share deep intensional similarity, expressed through different aspects (terminological affinity, structural similarity, common instances and so on…). This kind of similarity does not guarantee any strong semantic implication, though could be useful in some contexts: a query to a QA system could benefit of "similar" matches, as the retrieved information may then judged by the human receiving the answer.
3. *SuperClass-Of (SubClass-Of)*: it holds when the class from *LO* represents a more general (specific) concept than the one from *RO*. The term SuperClass-Of (SubClass-Of) indicates that the class from *LO* could be ideally considered as a SuperClass (SubClass) of the one from *RO*, should the two ontologies be merged. The semantics follows as for ExtensionalEquivalence: If a class A is SuperClass

of a class B, it is possible to consider each instance of B as an instance of A, though, in this case, the converse is not true.

DefaultAttributeMapType: it restricts the abstract *AttributeMapType* with the similarity cases between ontology attributes defined below:

1. *RangeEquivalence*: should the range of two considered attributes (in their original definition, not considering restrictions applied to them when they are attached to different classes) be covering elements which are themselves mapped as equivalent, then a *RangeEquivalence* is considered for them.
2. *RangeSimilarity*: should the range of two considered attributes (in their original definition, not considering restrictions applied to them when they are attached to different classes) be covering elements which are themselves mapped as similar, then a RangeSimilarity holds between them.
3. *RangeTypeMismatch*: should the range of two considered attributes (in their original definition, not considering restrictions applied to them when they are attached to different classes) be covering elements which present a mapping mismatch of any kind, then a RangeTypeMismatch holds between them.
4. *RoleInversionRangeTypeMismatch, RoleInversionRangeSimilarity, RoleInversionRangeEquivalence*: these three types of matching are under all aspects equivalent to the (corresponding) first three ones, with the exception that Range and Domain and inverted. A match of this kind should be reported if a direct match (i.e. a match from one attribute to another attribute with proper Domain and Range equivalences) is not available for the desired attribute.

In this schema, we limited specification of attribute similarities only to range similarity, as diverse knowledge representation languages do not allow for explicit specification of attributes' domain. Association similarity levels are still under study.

Manual Mappings between Federated Ontologies: some considerations

Mapping procedure, as it has been previously described in details, may be carried on also by humans. Mappings should be defined at the best of *Mapper*'s knowledge; this implies that humans producing a map should base on their personal knowledge and perspective of the world, together with observations on structural similarities between the two given ontologies. An *Automatic Mapper* could instead exploit some external linguistic/ontological resources to obtain a wider knowledge in judging mappings between the domain ontologies.

The two main considerations to be clearly assumed when producing a map are:

1. *avoid complex mappings wherever simple mappings are available*
2. *avoid "low-rank" mapping types when "high-rank" mappings are available*

Regarding the first assertion., it is important to clarify that, while complex mappings are a valuable mean to represent complex conceptual anchors between elements from different ontologies, they should only be used to fill "holes" in the mapping document that could never be bridged otherwise.

The same considerations hold for the second statement, as it is best to map a class/instance/attribute/association with its best matching counterpart and omit the (many) other degraded mappings that may involve the given ontology element. So, for example, if a given class C_L from *LO* is extensionally equivalent (the highest rank

of available class mappings) to a class C_R fro *RO*, C_L and C_R must be mapped using the proper map type without caring about other classes from *RO* that may be "more or less" similar to C_L; for the same reason, if an attribute A_L from *LO* is in *RangeEquivalence* with an attribute A_R from *RO* and if there exists an attribute I_R in *RO* which is the inverse role of A_R, there is no need of stating the *RoleInversionRangeEquivalence* between A_L and I_R: such a strategy is necessary to prevent an exponential growth in the range of possible relations that should be instantiated for every given ontological entity. If an element has no direct counterpart, it is up to the mediating agents to choose the best strategy for navigating their own ontologies and look for conceptual correspondences with the other ones: this is a fundamental difference wrt MAFRA [8], where a mapping ontology is built to bridge every (mappable) element from the two ontologies, or C-OWL [11], where the provided examples show complete mappings using the given primitives. Our future research work will go in the direction of formalizing conditions for completeness and compactness of a mapping document, to enhance the quality of automatically inferred mappings.

4 Conclusions

Ontologies are very often considered as a mean to conceptualize and express (by means of concepts and relationships) all of the knowledge relevant to a given domain, thus supporting automatic reasoning.

Nowadays, although still retaining their original role, Semantic Web has pushed forward the idea of ontologies as a mean for enabling knowledge sharing and reuse. Semantic Interoperability is thus a crucial issue in scenarios where different and distributed resources need to be reconciled, overcoming heterogeneities rising from distinct locales, languages, and, at a deeper analysis, different ways of structuring and organizing the information knowledge.

Following this trend, we are now facing a growing need for tools, languages and formalisms that should support the sharing of domain knowledge in a wide variety of different situations. XeOML, with its layered approach, tries to suggest a new direction for the representation of mappings between ontologies. This language presents a simple unifying view over the kind of elements that should be considered as relevant in every ontological framework, along with a classification of the diverse relationships that may occur between them in unforeseen mapping scenarios. Several extensions can be specified over the core language, providing local and more specific descriptions of the mapped ontological elements and detailing with the desired accuracy the relationships they are involved in.

With approaches like that, the unavoidable trade-off laying between coverage of heterogeneous information sources and preservation of common semantics to access the related content can thus be coped with: under a social perspective, communities which aim to reach knowledge sharing and services interoperability, can rely on a well assessed formalism for mapping their knowledge, only needing to commit to the agreed semantics for qualifying and ranking knowledge mappings (XeOML Mapping Rank Language Extension); at the same time, if we consider the technological aspect of the approach, the distributed agents which are meant to exchange knowledge may rely on the same basic functionalities for interpreting the core language and need only

to be “tuned” to the community, in order to capture and exploit the committed semantics for recognizing and ranking ontology mappings.

Acknowledgements This work has been partially funded by the EU project MOSES IST-2001-37244.

References

1. Hammer, J., Garcia-Molina, H., Ireland, K., Papakonstantinou, Y., Ullman, J., and Widom, J.: Information Translation, Mediation, and Mosaic-Based Browsing in the TSIMMIS System. In Exhibits Program of the Proceedings of the ACM SIGMOD International Conference on Management of Data, page 483, San Jose, California (June 1995).
2. Levy, A. Y., Rajaraman, A. and Ordille, J. J.: Querying Heterogeneous Information Sources Using Source Descriptions. In Proceedings of VLDB-96 (1996)
3. Genesereth, M. R., Keller, A. M. and Duschka, O.: Infomaster: An Information Integration System. In proceedings of 1997 ACM SIGMOD Conference (May 1997).
4. Chaudhri, V. K., Farquhar, A., Fikes, R., Karp, P.D. and Rice, J. P.: OKBC: A programmatic foundation for knowledge base interoperability. In Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98), pages 600-607, Madison, Wisconsin, USA, MIT Press (1998).
5. C. Ghidini and F. Giunchiglia. Local models semantics, or contextual reasoning = locality + compatibility. Artificial Intelligence, 127(2):221–259, April 2001
6. Berners-Lee, T., Hendler, J., and Lassila, O.: The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American. (May, 2001).
7. Beneventano, D., Bergamaschi, S., Guerra, F., Vincini, M.: The MOMIS approach to Information Integration. IEEE and AAAI International Conference on Enterprise Information Systems (ICEIS01), Setúbal, Portugal (7-10 July, 2001).
8. Maedche, A., Motik, B., Silva, N. and Volz, R.: Mafra a mapping framework for distributed ontologies. In Proceedings of the 13th European Conference on Knowledge Engineering and Knowledge Management EKAW-2002, Madrid Spain (2002).
9. Pazienza, M.T., Vindigni M.: Language-based agent communication. 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02), Sigueza, Spain (2002)
10. Pazienza, M.T., Stellato, A., Vindigni, M.: ALINAs: un'architettura multilayer ad agenti per il supporto alla comunicazione linguistica. Dagli oggetti agli agenti (WOA2002), Milano, Italy (2002)
11. Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L. and Stuckenschmidt, H.: C-OWL: Contextualizing Ontologies, ISWC'2003, LNCS 2870, Sept. 2003, pp. 164--179
12. Pazienza, M.T., Vindigni, M: Agent based Ontological Mediation in IE Systems. In “Information Extraction in the Web Era: Natural Language Communication for Knowledge Acquisition and Intelligent Information Agents” LNAI 2700, Springer 2003
13. Garshol, L. G. and Moore, G.: Topic Maps — XML Syntax. (16 March 2004)
14. Dean, M. and Schreiber, G. editors.: OWL Web Ontology Language Guide. 2004. W3C Recommendation (10 February 2004).
15. De Bruijn, J. and Polleres, A.: Towards an Ontology Mapping Specification Language for the Semantic Web. DERI Technical Report (30 June 2004).

Ontologies with Vocabulary Consent Relationship

Yuzhong Qu, Zhiqiang Gao, Yuqing Zhai, Jianming Deng

{yzqu, zqgao, yqzhai, jmdeng}@seu.edu.cn
Dept. of Computer Science and Engineering
Southeast University, Nanjing 210096, P. R. China

Abstract. In this paper, two kinds of vocabulary consent relations are proposed as a mechanism for the partial reuse and integration of ontologies, and then a formal semantics of consent relationships is given to support such a mechanism. Particularly, some semantic conditions are proposed to guarantee the expected consequences of vocabulary consent relationships. The issue about reasoning with the given semantic framework is also briefly discussed.

1 Introduction

Web ontology languages play a crucial role in the emerging Semantic Web [1], as they provide mechanisms to represent and reason with term vocabularies and the relationships between entities in these vocabularies. Among them, the OWL Web Ontology Language [2] and RDF Schema [3, 4] are two promising ones. OWL provides more expressive power than RDF Schema, while RDF Schema defines basic ontological modeling primitives on top of RDF [5]. To make the Semantic Web vision become reality, we are challenged by the URI meaning issues arising from integrating and reusing ontologies distributed on the Web. As we know, URI references (or names) are used to denote some resources, and the characteristics of the denoted resources are depicted by the ontology using the URI references. In the situation where different ontologies use one URI reference, some problems occur. For example, should the meaning of a URI reference be global or local? Does the use of a URI reference from an existing ontology in current ontology constitute some kind of an assent to the meaning specified by the original ontology? Can one ontology reuses or shares some part of another ontology?

As we know, a URI reference (or a name in short) alone is meaningless for machine without context, while the meaning of a set of related names comes mainly from the relations among them gained from some context. In the case of a single ontology, the meaning of the names used in the ontology comes from the axioms and facts asserted by the ontology, and usually, can be given in terms of the possible interpretations prescribed in an agreed-up formalism, such as RDF Semantics [4], OWL Semantics and Abstract Syntax [2], etc. In the case of distributed ontologies, OWL has a mechanism to import an OWL ontology into another OWL ontology, more concretely, an owl:imports statement refers another OWL ontology containing the descriptions of the resources denoted by some names, whose meaning is considered to be a part of the meaning of the importing ontology. However, the

import mechanism does not deal with the situations where the partial reuse is needed or where some different but compatible perspectives do exist. Recently, Paolo Bouquet et al. [6] gave an extension to OWL with context, named with C-OWL (Context OWL), which is based on local models semantics [7, 8] and distributed description logics [9]. C-OWL uses explicit mappings (bridge rules) as the context. However, the vocabularies of local ontologies in C-OWL are supposed to be pair-wise disjointed, and the globalization can only be obtained by using explicit mappings.

In this paper, two kinds of vocabulary consent relations are proposed as a mechanism for partial reuse and integration of ontologies on the Web, and then a formal semantics of vocabulary consent relationships is presented to support such a mechanism. In particularly, some semantic constraints are given to guarantee the expected consequences of these relationships. The rest of this paper is organized as follows: Some notations for distributed ontologies used in this paper as well as the possible interpretations of distributed ontologies are introduced in section 2. The concept of vocabulary consent relationships and their semantic conditions are formally given in section 3, and the issue of reasoning with consent relationships is also discussed in that section. Finally, section 4 gives some comparison of the presented approach with other related approached, and sketches out some future works.

2 Distributed Ontologies

To simplify the presentation of our formalism, we mainly discuss OWL Lite ontologies distributed on the Web. Datatypes, values as well as datatype properties are ignored for simplicity. The OWL Lite abstract syntax and the corresponding Description Logic syntax are used interchangeably throughout this paper, as shown in Figure 1 and Figure 2, which are the OWL Lite fragments of the ones in [10]. The specific meaning given to OWL Lite atomic classes and restriction classes is shown in the last column of Figure 1, where Δ^{ι} is the domain of individuals in an interpretation. As usual, the meaning of assertions (axioms and facts) is given in terms of constraints on possible interpretations, as shown in the last column of Figure 2.

The terminologies for relations are given here. Let $\Re_1$ be a relation from U_1 to U_2 (a subset of $U_1 \times U_2$), $\Re_2$ be a relation from U_2 to U_3, and S be a subset of U_1. We have the following definitions.

$$\Re_1(S) \triangleq \{y \mid \exists x.\ x \in S \text{ and } \langle x, y\rangle \in \Re_1\};$$
$$\Re_1^{-1} \triangleq \{\langle y, x\rangle \mid \langle x, y\rangle \in \Re_1\};$$
$$\Re_2 \circ \Re_1 \triangleq \{\langle x, z\rangle \mid \exists y.\ \langle x, y\rangle \in \Re_1 \text{ and } \langle y, z\rangle \in \Re_2\};$$
$$I_{U1} \triangleq \{\langle x, x\rangle \mid x \in U_1\}.$$

Obviously, $\Re_1$ is transitive iff $\Re_1 \circ \Re_1 \subseteq \Re_1$; symmetric iff $\Re_1 = \Re_1^{-1}$; functional (i.e. a partial function) iff $\Re_1 \circ \Re_1^{-1} \subseteq I_{U2}$; reverse functional (i.e.one-to-one) iff $\Re_1^{-1} \circ \Re_1 \subseteq I_{U1}$; onto iff $I_{U2} \subseteq \Re_1 \circ \Re_1^{-1}$; etc.

No.	Abstract Syntax	DL Syntax	Semantics
	Concept Formation		
CF1	A (URI reference)	A	$A^I \subseteq \Delta^I$
CF2	owl:Thing	$\top$	$(owl:Thing)^I = \Delta^I$
CF3	owl:Nothing	$\bot$	$(owl:Nothing)^I = \{\}$
CF4	restriction(R someValuesFrom(A))	$\exists R.A$	$\{x \in \Delta^I \mid \exists y.<x,y> \in R^I \wedge y \in A^I\}$
CF5	restriction(R allValuesFrom(A))	$\forall R.A$	$\{x \in \Delta^I \mid \forall y.<x,y> \in R^I \rightarrow y \in A^I\}$
CF6	restriction(R minCardinality(0))	≥ 0 R (i.e. $\top$)	Δ^I
CF7	restriction(R minCardinality(1))	≥ 1 R (i.e. $\exists R.\top$)	$\{x \in \Delta^I \mid \exists y.<x, y> \in R^I\}$
CF8	restriction(R maxCardinality(0))	≤ 0 R (i.e. $\forall R.\bot$)	$\{x \in \Delta^I \mid !(\exists y.<x, y> \in R^I)\}$
CF9	restriction(R maxCardinality(1))	≤ 1 R	$\{x \in \Delta^I \mid \forall y.\forall z.$ $<x,y> \in R^I \wedge <x,z> \in R^I \rightarrow y=z\}$
	Property Formation		
PF1	R (URI reference)	R	$R^I \subseteq \Delta^I \times \Delta^I$
		R^-	$(R^-)^I = (R^I)^{-1}$
	Individual Formation		
IF1	o (URI reference)	o	$o^I \in \Delta^I$

Fig. 1. OWL Lite Concepts, Properties and Individuals.

No.	Abstract Syntax	DL Syntax	Semantics
AF1	Class(A partial C_1 ...C_n)	$A \sqsubseteq C_1 \sqcap ... \sqcap C_n$	$A^I \subseteq C_1^I \cap \ldots \cap C_n^I$
AF2	Class(A complete C_1 ...C_n)	$A = C_1 \sqcap ... \sqcap C_n$	$A^I = C_1^I \cap \ldots \cap C_n^I$
AF3	EquivalentClasses(A_1 ...A_n)	$A_1 = \ldots = A_n$	$A_1^I = \ldots = A_n^I$
AF4	SubPropertyOf(R_1 R_2)	$R_1 \sqsubseteq R_2$	$R_1^I \subseteq R_2^I$
AF5	EquivalentProperties(R_1 ...R_n)	$R_1 = \ldots = R_n$	$R_1^I = \ldots = R_n^I$
AF6	ObjectProperty(R super(R_1)...super(R_n)	$R \sqsubseteq R_i$	$R^I \subseteq R_i^I$
	domain(A_1) ...domain(A_m)	$\exists R.\top \sqsubseteq A_i$	$R^I \subseteq A_i^I \times \Delta^I$
	range(A_1) ...range(A_k)	$\top \sqsubseteq \forall R.A_i$	$R^I \subseteq \Delta^I \times A_i^I$
	[inverseOf(R_0)]	$R = (R_0^-)$	$R^I = ((R_0)^I)^{-1}$
	[Symmetric]	$R = (R^-)$	$R^I = (R^I)^{-1}$
	[Functional]	$\top \sqsubseteq \leq 1R$	R^I is functional
	[InverseFunctional]	$\top \sqsubseteq \leq 1(R^-)$	R^I is reverse functional
	[Transitive])	Tr(R)	R^I is transitive
AF7	SameIndividual(o_1 ... o_n)	$o_1 = \ldots = o_n$	$o_1^I = \ldots = o_n^I$
AF8	Individual(o type(C_1) ...type(C_n)	$C_i(o)$	$o^I \in C_i^I$
	value(R_1 o_1)...value(R_n o_n))	$R_i(o, o_i)$	$<o^I, o_i^I> \in R_i^I$
AF9	DifferentIndividuals(o_1 ... o_n)	$o_i \neq o_j$, $i \neq j$	$o_i^I \neq o_j^I$ $i \neq j$

Fig. 2. OWL Lite Assertions (Axioms and Facts)

Some other terminologies are as follows: The vocabulary of an ontology is the set of names (URI references) that occur in the ontology as individuals, classes and properties, except for the ones of built-in. **We use Σ to denote a distributed ontology, i.e. a set of ontologies $\{O_i\}_{i\in I}$, where I is a set of indexes (for the sake of convenience, we will also use Σ to denote a set of ontologies with some identified context in other places)**. We use $V(O_i)$ to denote the vocabulary of O_i, and use $V_I(O_i)$, $V_C(O_i)$ and $V_P(O_i)$ to denote the individual, class, and property vocabulary of O_i, respectively. Following the notion of distributed description logics [9], the formal semantics of distributed ontologies is defined in terms of its possible interpretations.

Definition 1. (Interpretations of distributed ontologies) Let $\Sigma=\{O_i\}_{i\in I}$ be a set of ontologies. A possible interpretation of Σ, denoted by ι, is composed of a pair $<\{\iota_i\}_{i\in I},\{r_{ij}\}_{i,j\in I}>$ such that

1. ι_i is an OWL Lite interpretation of O_i (satisfying all of the axioms in O_i) with a local domain $\Delta^{\iota i}$.
2. r_{ij} is a domain relation from i to j, i.e. a subset of $\Delta^{\iota i} \times \Delta^{\iota j}$. (For every $i\in I$, r_{ii} is always assumed to be the identity relation on $\Delta^{\iota i}$.)

Notice that the domain relation r_{ij} is a relation from $\Delta^{\iota i}$ to $\Delta^{\iota j}$, it's not necessarily to be a function from $\Delta^{\iota i}$ to $\Delta^{\iota j}$, i.e. there may be no corresponding element in $\Delta^{\iota j}$ for some elements in $\Delta^{\iota i}$ via r_{ij} (e.g. something observed by agent *i* is neglected by agent *j*) and there may be more than one corresponding elements in $\Delta^{\iota j}$ for one element in $\Delta^{\iota i}$ via r_{ij} (e.g. one thing observed by agent *i* is perceived as many things by agent *j* due to different properties concerned).

Example 1. Consider a set of two ontologies $\{O_1, O_2\}$ as the following (using DL syntax to present axioms, where A, B, and C stands for class names):

$$O_1: A \sqsubseteq B; B \sqsubseteq C$$

$$O_2: C \sqsubseteq D; D \sqsubseteq A$$

How to interpret the integration of the above two ontologies? We know that A is a sub-class of C from perspective of O_1 (i.e. $A^{\iota 1}\subseteq C^{\iota 1}$) and C is a sub-class of A from perspective of O_2 (i.e. $C^{\iota 2}\subseteq A^{\iota 2}$). There are many possibilities for other implicit information, and more context information should be given to gain a more clear understanding.

Example 2. Consider an ontology about people[1], called *example*, it consists of at least following assertions:

$$\text{Person} \sqsubseteq \text{Animal}; \qquad \text{(A1)}$$

$$\text{Driver} = \text{Person} \sqcap \exists \text{drives. Vehicle}; \qquad \text{(A2)}$$

[1] http://owl.man.ac.uk/2003/why/latest/ontology.rdf

$$\text{Man} = \text{Male} \sqcap \text{Adult} \sqcap \text{Person}; \quad \text{(A3)}$$

$$\text{Driver} \sqsubseteq \text{Adult}; \quad \text{(A4)}$$

……

Suppose some application developers want to reuse the above ontology when they design their application-specific ontology, called *myOntology*. However, they do not like some vocabulary of this ontology, e.g. Adult, and somewhat consent to some other vocabulary, e.g. Man and Animal, and do consent to some other vocabulary, e.g. Person, Driver, Vehicle and drives. In this case, the import mechanism provided by OWL is clearly not applicable. The vocabulary consent relationship presented in next section will give a formalism to address this kind of situation.

Before presenting the formalism, we need to define the requirements on the expected consequences of consent relations. In this paper, two kinds of consent relations are identified, namely weak consent and consent relations, and the expected consequences of these relationships are roughly identified as follows.

(1) Most of the simple OWL Lite assertions with the concerned vocabulary keep true via weak consent relation. For example, in *example* ontology, a man must be an animal, i.e. Man $\sqsubseteq$ Animal, this is an implicit assertion derived from (A1) and (A3). If *myOntology* weakly consents to *example* with the vocabulary Man and Animal, then Man $\sqsubseteq$ Animal should keeps true in *myOntology*.

(2) Most of the OWL Lite assertions with the concerned vocabulary keep true via consent relation. For example, If *myOntology* consents to *example* with the vocabulary Person, Driver, Vehicle and drives, then (A2), which is explicitly asserted in *example*, should keeps true in *myOntology*.

The identified requirements listed above are primary and somewhat rough. This paper doesn't try to nail down the commonly agreed requirements on the consent relations, because it needs the Semantic Web community to do so. Instead, this paper proposes two kinds of vocabulary consent relations as a mechanism for partial reuse and integration of ontologies, and presents a formalism to support such a mechanism, especially, some semantic constraints are given to guarantee the expected consequences of these relationships. Whenever some kinds of consent relations and their expected consequences of these relationships are identified by the Semantic Web community, the presented approach can be used to formalize these commonly agreed consent relations.

3 Consent Relationship

In this section, consent relationships are proposed as key component of context for distributed ontologies, and the formal semantics of consent relationships is presented, and the reasoning along with consent relationships is also discussed.

Definition 2 (Consent relation). Let $\Sigma=\{O_i\}_{i\in I}$ be a set of local ontologies. Syntactically, a consent relation on $\{O_i\}_{i\in I}$ is an indexed family of sets $\{\mathfrak{C}_{ij}\}_{i,j\in I}$, usually denote by $\mathfrak{C}\equiv\{\mathfrak{C}_{ij}\}_{i,j\in I}$, where $\mathfrak{C}_{ij}$ is a subset of $V(O_i)\cap V(O_j)$. We say O_j consents to O_i with ν so long as $\nu\in\mathfrak{C}_{ij}$, and O_j consents to O_i so long as $V(O_i)=\mathfrak{C}_{ij}$. The syntax form of a weak consent relation is defined analogously, denoted by $\mathfrak{C}^w\equiv\{\mathfrak{C}_{ij}^w\}_{i,j\in I}$.

From now on, we use $\Sigma=(\{O_i\}_{i\in I}, \mathfrak{C}^w, \mathfrak{C})$ to denote a set of local ontologies with consent relations (a weak consent relation $\mathfrak{C}^w$ and a consent relation $\mathfrak{C}$). It's not hard to extend OWL Syntax to represent consent relations, but the details about the syntax of such an extension are beyond the scope of this paper. The semantics of consent relationships is given in terms of constraints (i.e. semantic conditions) on possible interpretations of the distributed ontology concerned.

Definition 3 (Semantic condition for consent relationship). Let $\Sigma=(\{O_i\}_{i\in I}, \mathfrak{C}^w, \mathfrak{C})$ be a distributed ontology with consent relations. A possible interpretation of Σ, $<\{\iota_i\}_{i\in I},\{r_{ij}\}_{i,j\in I}>$, is an interpretation of $\{O_i\}_{i\in I}$ satisfying the following semantic conditions for consent relations.

(1). If $A\in\mathfrak{C}^w_{ij}\cap V_C(\Sigma)$, then (SCC−C1) holds;

$$A^{\iota_j} = r_{ij}(A^{\iota_i}) \qquad \text{(SCC−C1)}$$

(1'). If $A\in\mathfrak{C}_{ij}\cap V_C(\Sigma)$, then (SCC−C1) and (SCC−C2) holds;

$$r_{ij}^{-1}(r_{ij}(A^{\iota_i})) \subseteq A^{\iota_i} \qquad \text{(SCC−C2)}$$

(2). If $R\in\mathfrak{C}^w_{ij}\cap V_P(\Sigma)$, then (SCC−P1) holds;

$$R^{\iota_j} = r_{ij}\circ R^{\iota_i}\circ r_{ij}^{-1} \qquad \text{(SCC−P1)}$$

(2'). If $R\in\mathfrak{C}_{ij}\cap V_P(\Sigma)$, then (SCC−P1), (SCC−P2) and (SCC−P3) holds;

$$R^{\iota_j}\circ r_{ij} \subseteq r_{ij}\circ R^{\iota_i} \qquad \text{(SCC−P2)}$$

$$R^{\iota_i}\circ r_{ij}^{-1} \subseteq r_{ij}^{-1}\circ R^{\iota_j} \qquad \text{(SCC−P3)}$$

(3). If $o\in\mathfrak{C}^w_{ij}\cap V_I(\Sigma)$, then (SCC−I1) holds;

$$\{o^{\iota_j}\} = r_{ij}(\{o^{\iota_i}\}) \qquad \text{(SCC−I1)}$$

(3'). If $o\in\mathfrak{C}_{ij}\cap V_I(\Sigma)$, then (SCC−I1) and (SCC−I2) holds.

$$r_{ij}^{-1}(r_{ij}(\{o^{\iota_i}\})) = \{o^{\iota_i}\} \qquad \text{(SCC−I2)}$$

Note. The conditions (SCC−C1), (SCC−P1) and (SCC−I1) are given to ensure the expected consequence of weak consent relationship (as roughly described at the end of section 2), while the more conditions (SCC−C2), (SCC−P2), (SCC−P3) and (SCC−I2) are given to ensure the expected consequence of consent relationship. When one consent relationship is to be used, the corresponding conditions need to be checked, which may involve meaning coordination and negotiation. For example, suppose *myOntology* is prepared to partially reuse the *example* ontology and consent to it with the vocabulary *drives* (a property). Before doing so, the ontology designer need to check whether or not for every possible interpretation of the *example* ontology and his/her intended interpretation of *myOntology*, there exists a domain

relation r_{ij} such that the conditions (SCC−P1), (SCC−P2) and (SCC−P3) hold for r_{ij} and R (here R is *drives*).

Example 3. (Revisit example 1) If we further identify the context that O_2 weakly consents to O_1 with class vocabulary A and C, i.e. $\{A, C\} = \mathfrak{C}^w_{12}$, then the following constraints hold.

$$\begin{aligned} A^{\iota 2} &= r_{12}(A^{\iota 1}) && \text{by } A \in \mathfrak{C}^w_{12} \text{ and (SCC−C1)} \\ &\subseteq r_{12}(C^{\iota 1}) && \text{(Due to the fact } A^{\iota 1} \subseteq C^{\iota 1}) \\ &= C^{\iota 2} && \text{by } C \in \mathfrak{C}^w_{12} \text{ and (SCC−C1)} \end{aligned}$$

Together with $C^{\iota 2} \subseteq A^{\iota 2}$, we derive $C^{\iota 2} = A^{\iota 2}$, i.e. A is equivalent to C in O_2. It is worthy of noticing that this example allows O_2 not consent O_1 with class B (O_2 may not agree with what O_1 says about class B). **In fact, this case illustrates a scenario of partial reuse of vocabulary by using weak consent relationship. This is different from the owl:import mechanism, where the importing ontology must agree with all of the what the imported ontology says.** This example also motivates us to think about the entailment relation between distributed assertions.

Definition 4. (Entailment between distributed assertions) Let $\Sigma = (\{O_i\}_{i \in I}, \mathfrak{C}^w, \mathfrak{C})$ a distributed ontology. We say O_{i1}: assertion$_1$, …, O_{in}: assertion$_n$ entail O_j: assertion$_j$, written by

$$O_{i1}\text{: assertion}_1, \ldots, O_{in}\text{: assertion}_n \models_\Sigma O_j\text{: assertion}_j,$$

iff for each intepretation $<\{\iota_i\}_{i \in I}, \{r_{ij}\}_{i,j \in I}>$ of Σ, if ι_{i1} satisfies assertion$_1$, …, and ι_{in} satisfies assertion$_n$, then ι_j must satisfy assertion$_j$.

Notice that we usually consider whether or not an assertion (explicitly asserted or implicitly deduced) in a reused ontology keeps true in the ontology reusing vocabulary.

Proposition 1. Let $\Sigma = (\{O_i\}_{i \in I}, \mathfrak{C}^w)$ be a distributed ontology with a weak consent relation, and $\{\upsilon_1, \upsilon_2, o_1, o_2\} \in \mathfrak{C}^w_{ij}$. Then every form of assertions in the following is true in O_j under a given interpretation of Σ if it is true in O_i under the given interpretation of Σ.

(1). $\upsilon_1 \sqsubseteq \upsilon_2$, where both of υ_1 and υ_2 are either class names or property names.
(2). $\upsilon_1(o_1)$, where υ_1 is a class name while o_1 is an individual name.
(3). $o_1 = o_2$, where both of o_1 and o_2 are individual names.
(4). $\upsilon_1(o_1, o_2)$, where υ_1 is a property name while both of o_1 and o_2 are individual names.
(5). $\upsilon_1 = \upsilon_2^-$, where both of υ_1 and υ_2 are property names.

Proof. We just give the proof of (1) in the case of property. Let $<\{\iota_i\}_{i \in I}, \{r_{ij}\}_{i,j \in I}>$, be an interpretation of Σ. We have

$$\upsilon_1^{\iota j} = r_{ij} \circ \upsilon_1^{\iota i} \circ r_{ij}^{-1} \subseteq r_{ij} \circ \upsilon_2^{\iota i} \circ r_{ij}^{-1} = \upsilon_2^{\iota j}$$

So, $\upsilon_1 \sqsubseteq \upsilon_2$ is also true in O_j.
The proof for others is similar to the above. □

Corollary 1. Every form of assertions in the following keeps true from O_i to O_j whenever the involved names belong to $\mathfrak{C}^{w}_{ij}$, under the distributed interpretations satisfying semantic condition for weak consent relation.

(1). (AF3), (AF4), (AF5) and (AF7);
(2). (AF1) and (AF8) with C_i restricted to atomic concepts;
(3). (AF6) without [Functional], [InverseFunctional] and [Transitive].

Notice that the assertions can be explicitly asserted axioms and facts in O_i or implicit assertions (derived within O_i), e.g. Man $\sqsubseteq$ Animal is an implicit assertion derived from (A1) and (A3) in the *example* ontology. On the other hand, the assertions taking form of (AF2) doesn't keep true in general due to the fact that $r_{ij}(C_1^{\iota_i}) \cap r_{ij}(C_2^{\iota_i})$ and $r_{ij}(C_1^{\iota_i} \cap C_2^{\iota_i})$ are not necessary to be equal, even when C_1 and C_2 are atomic concepts, while (AF9) doesn't keep true in general due to the fact that r_{ij} is not necessary to be one-to-one.

Roughly speaking, most of the simple assertions of OWL Lite (including all of the ones borrowed from RDF Schema) keep true via the weak consent relation. It will be proved that almost all assertion of OWL Lite keeps true via consent relation.

Lemma 1. Let $\Re$ be a relation from U_1 to U_2, and S_i (i=1…n) be subsets of U_1. If for i=1…n, $\Re^{-1}(\Re(S_i)) \subseteq S_i$ holds, then $\Re(S_1 \cap \ldots \cap S_2) = \Re(S_1) \cap \ldots \cap \Re(S_n)$. (The proof is omitted due to the space limitation)

Proposition 2. Let $\Sigma = (\{O_i\}_{i \in I}, \mathfrak{C})$ be a distributed ontology with consent relation, if $\{R, A, A_1, \ldots, A_n, o_1, o_2\} \in \mathfrak{C}_{ij}$, and $\langle \{\iota_i\}_{i \in I}, \{r_{ij}\}_{i,j \in I} \rangle$ be an interpretation of Σ, then,

(1). $r_{ij}((\exists R.A)^{\iota_i}) = (\exists R.A)^{\iota_j}$
(2). $r_{ij}((\forall R.A)^{\iota_i}) = r_{ij}(\top^{\iota_i}) \cap (\forall R.A)^{\iota_j}$
(3). $r_{ij}((A_1 \sqcap \ldots \sqcap A_n)^{\iota_i}) = (A_1 \sqcap \ldots \sqcap A_n)^{\iota_j}$
(4). R^{ι_j} is transitive if R^{ι_i} is transitive. (i.e. $R^{\iota_j} \circ R^{\iota_j} \subseteq R^{\iota_j}$ so long as $R^{\iota_i} \circ R^{\iota_i} \subseteq R^{\iota_i}$)
(5). R^{ι_j} is functional if both of R^{ι_i} and r_{ij} are functional
(6). $o_1^{\iota_j} \neq o_2^{\iota_j}$ if $o_1^{\iota_i} \neq o_2^{\iota_i}$.

Proof. (1). Suppose $y \in r_{ij}((\exists R.A)^{\iota_i})$, then there exists an $x \in (\exists R.A)^{\iota_i}$ such that $\langle x, y \rangle \in r_{ij}$. According to the interpretation of $(\exists R.A)^{\iota_i}$, there exists an $x_1 \in A^{\iota_i}$ such that $\langle x, x_1 \rangle \in R^{\iota_i}$. So we have $\langle y, x_1 \rangle \in R^{\iota_i} \circ r_{ij}^{-1}$. By (SCC−P3), we have $\langle y, x_1 \rangle \in r_{ij}^{-1} \circ R^{\iota_j}$, i.e. there exists $y_1 \in \Delta^{\iota_j}$ such that $\langle y, y_1 \rangle \in R^{\iota_j}$ and $\langle y_1, x_1 \rangle \in r_{ij}^{-1}$. Since $\langle x_1, y_1 \rangle \in r_{ij}$ and $x_1 \in A^{\iota_i}$, we have $y_1 \in r_{ij}(A^{\iota_i}) = A^{\iota_j}$ (by (SCC−C1)), and then $y \in (\exists R.A)^{\iota_j}$. Therefore, $r_{ij}((\exists R.A)^{\iota_i}) \subseteq (\exists R.A)^{\iota_j}$.

As for the reverse direction, suppose $y \in (\exists R.A)^{\iota_j}$, then there exists a $y_1 \in A^{\iota_j}$ such that $\langle y, y_1 \rangle \in R^{\iota_j}$. By (SCC−P1), there exist x and x_1 such that $\langle x, y \rangle \in r_{ij}$, $\langle x, x_1 \rangle \in R^{\iota_i}$,

and $<x_1,y_1>\in r_{ij}$. By (SCC−C1, C2) and the facts that $<x_1, y_1>\in r_{ij}$ and $y_1\in A^{\iota j}$, we have $x_1\in A^{\iota i}$, and then $x\in(\exists R.A)^{\iota i}$. It leads to $y\in r_{ij}((\exists R.A)^{\iota i})$. Thus, $(\exists R.A)^{\iota j}\subseteq r_{ij}((\exists R.A)^{\iota i})$.

We omit the proof of (2) – (6) due to the space limitation. □

Proposition 3. Let $\Sigma=(\{O_i\}_{i\in I}, \mathfrak{C})$ be a distributed ontology with consent relation, if $\{R, A\}\in\mathfrak{C}_{ij}$, and $<\{\iota_i\}_{i\in I},\{r_{ij}\}_{i,j\in I}>$ be an interpretation of Σ, then the following holds.

(1). $r_{ij}(C^{\iota i}) = C^{\iota j}$ and $r_{ij}^{-1}(r_{ij}(C^{\iota i})) \subseteq C^{\iota i}$ for C taking the form of A, $\exists R.A$, $\bot$, $\exists R.\top$.
(2). $r_{ij}(C^{\iota i}) = r_{ij}(\top^{\iota i})\cap C^{\iota j}$ and $r_{ij}^{-1}(r_{ij}(C^{\iota i})) \subseteq C^{\iota i}$ for C taking the form of $\forall R.A$, $\top$, $\forall R.\bot$.
(3). Suppose C takes the form of ≤ 1 R, i.e. restriction(R maxCardinality(1)). If r_{ij} is functional, then $r_{ij}(C^{\iota i})\subseteq C^{\iota j}$. If r_{ij} is one-to-one, then $r_{ij}^{-1}(C^{\iota j}) \subseteq C^{\iota i}$ and $r_{ij}(\top^{\iota i})\cap C^{\iota j}\subseteq r_{ij}(C^{\iota i})$. (Namely, (2) also holds for C taking the form of ≤ 1 R when r_{ij} is required to be one-to-one and functional).

(The proof is similar to the proposition 2, and is omitted due to the space limitation.)

Corollary 2. Every form of OWL Lite assertions, except for the following forms of assertions, keeps true from O_i to O_j whenever the involved names belong to $\mathfrak{C}_{ij}$, under the distributed interpretations satisfying semantic condition for consent relation.

(1). ObjectProperty(R Functional) or ObjectProperty(R InverseFunctional);
(2). Assertions involving maxCardinality=1 (or cardinality=1);
(3). Class(A complete $C_1...C_n$) with each C_i being a value restriction ($n\geqslant 2$).

Proof. In the case of an assertion with one of the forms (AF3), (AF4), (AF5) and (AF7), it comes from proposition 1 and corollary 1. We just need to give the proof of following forms (Notice that C_i doesn't take the form of CF9, i.e. ≤ 1 R):

(AF1). Class(A partial C_1 ...C_n)	$A\sqsubseteq C_1\sqcap...\sqcap C_n$
(AF2). Class(A complete $C_1...C_n$) except each C_i is value restriction	$A= C_1\sqcap...\sqcap C_n$
(AF6). ObjectProperty(R Transitive)	Tr(R)
(AF8). Individual(o type(C_1) ...type(C_n))	$C_i(o)$
(AF9). DifferentIndividuals(o_1 ... o_n)	$o_i\neq o_j$, $i\neq j$

In the case of (AF1), we just need to illustrate the case C_1. Suppose $A\sqsubseteq C_1$ is true in O_i under $<\{\iota_i\}_{i\in I},\{r_{ij}\}_{i,j\in I}>$ satisfying semantic condition for consent relation. We have $A^{\iota i}\subseteq C_1^{\iota i}$, and then $r_{ij}(A^{\iota i})\subseteq r_{ij}(C_1^{\iota i})\subseteq C_1^{\iota j}$(By Proposition 3). So $A^{\iota j}= r_{ij}(A^{\iota i})\subseteq C_1^{\iota j}$. It means $A\sqsubseteq C_1$ is also true in O_j under $<\{\iota_i\}_{i\in I},\{r_{ij}\}_{i,j\in I}>$.

In the case of (AF2), we suppose C_1 is an atomic concept or an existential qualification without the loss of the generality. So we have $C_1^{\iota j}=r_{ij}(C_1^{\iota i})\subseteq r_{ij}(\top^{\iota i})$. Therefore

$$\begin{aligned}(C_1\sqcap...\sqcap C_n)^{\iota j} &= C_1^{\iota j}\cap C_2^{\iota j}\cap\ldots\cap C_n^{\iota j}\\ &= C_1^{\iota j}\cap(r_{ij}(\top^{\iota i})\cap C_2^{\iota j}))\cap\ldots\cap(r_{ij}(\top^{\iota i})\cap C_n^{\iota j})\\ &= r_{ij}(C_1^{\iota i})\cap r_{ij}(C_2^{\iota i})\cap\ldots\cap r_{ij}(C_n^{\iota i})\ \text{(By Proposition 3)}\end{aligned}$$

$= r_{ij}(C_1^{ui} \cap C_2^{ui} \cap \ldots \cap C_n^{ui}))$ (By lemma 1, Proposition 3)
$= r_{ij}((C_1 \sqcap ... \sqcap C_n)^{ui}) = r_{ij}(A^{ui}) = A^{ij}$

It leads to the conclusion. If each C_i is a value restriction, then we couldn't get the above result in general, due to the fact $(C_1 \sqcap ... \sqcap C_n)^{ij}$ may exceed $r_{ij}(\top^{ui})$.

As for (AF6), it comes from (4) of Proposition 2. As for (AF8), it comes from (1) and (2) of Proposition 2. As for (AF9), it comes from (6) of Proposition 2. □

Notice that, by the (3) of proposition 3, if r_{ij} is required to be one-to-one and functional, then every OWL Lite assertion keeps true via consent relation with only one exception, which occurs only when a complete class axiom Class(A complete $C_1 ... C_n$) (**n⩾2**) does have each C_i taking the form of value restriction ($\forall$R.A and $\forall$R.$\bot$) or maxCardinality=1 ($\leq$1 R). The existence of such an exception ascribe to the possibility of that $r_{ij}(\top^{ui}) \subsetneq \top^{ij}$. (The universe or thinking space of agent j is strictly larger than the one of agent i.) When $r_{ij}(\top^{ui}) \subsetneq \top^{ij}$, in the case of C taking one form of $\forall$R.A, $\forall$R.$\bot$, and $\leq$1 R, C^{ij} may contain some element exceeding the scope of $r_{ij}(\top^{ui})$ due to the fact that the interpretations of these concept constructs do not assume the existence of the property value. It leads to that $r_{ij}(C^{ui}) \subsetneq C^{ij}$. So, this exception, the case (3) of Corollary 2, is technically apprehensible.

Example 4. (Revisit example 2) Suppose *myOntology* consents to *example* with the vocabulary Person, Driver, Vehicle and drives, and weakly consents to *example* with Man and Animal. The simple assertion Man $\sqsubseteq$ Animal (taking the form AF1 as in Fig.2 with an atomic class as super-class) is derived from (A1) and (A3) in the *example* ontology, and then it is true in *myOntology* by Corollary 1. The complex assertion (A2) Driver = Person $\sqcap$ $\exists$drives. Vehicle (taking the form AF2 as in Fig 2.) is asserted in the *example* ontology, and then it is true in *myOntology* by Corollary 2.

In the case that O_j consents to O_i (as a whole), it implicitly requires that $V(O_i) \subseteq V(O_j)$. This case is close to the case that O_j owl:imports O_i, but with some difference as indicated in Corollary 2. If we introduce a new kind of consent relationship at ontology level (not at vocabulary level), called complete consent, with additional semantic condition that the corresponding domain relation be one-to-one and functional, then the complete consent relation is closely similar to owl:imports mechanism with only one difference as indicated in the case (3) of Corollary 2, which is technically apprehensible. Generally speaking, consent relationship (as well as weak consent relationship) is at vocabulary level, and it facilitates the meaning sharing of logically relevant part of ontology by specifying with which names an ontology (target ontology) consents to another one (source ontology). Here, the logically relevant part of an ontology means some forms of explicit and implicit assertions in the ontology which involve no other names than the ones specified. In other words, the meaning of relevant names is inherited from source ontology into the target ontology. As to the owl:imports mechanism, it's at ontology level, and its ability of combining models is restricted to the import of complete model of source ontology.

By Corollary 1 and Corollary 2, we can roughly say, most of the simple OWL Lite assertions (including all of the ones borrowed from RDF Schema) keep true via weak consent relation, while every OWL Lite assertion, except those indicated in Corollary 2, keeps true via consent relation. From viewpoint of semantics and reasoning, the consent relations should have some transitivity. If O_j consents to O_k with a set of vocabulary and O_k consents to O_i with the same set of vocabulary, then O_j would consents to O_i with the same set of vocabulary. However, if O_j consents to O_k with υ_1 and υ_2 while O_k consents to O_i with υ_2 and υ_3, then machines may not obtain any meaningful result directly from O_i to O_j in general due to the fact that O_j consents to O_i with only one name can not bring any useful thing with reasoning.

Based on above properties observed, we can conduct distributed reasoning along with consent relations. Roughly speaking, If O_j consents to O_{ik} (k=1,…n) with vocabulary set V_k and there are some explicit or implicit assertions (involving no other vocabulary than the ones in V_k) in O_{ik}, then we can conclude that these assertions are also true in O_j. Certainly, the reasoning should be goal oriented, and more research should be taken to make it practical. In addition, various consent relations among distributed ontologies can be identified and established according to the corresponding semantic conditions. Certainly, some techniques, such as multi-agent and context capturing, should be incorporated in order to identify consent relations among ontologies distributed on the Web.

4 Conclusion

In this paper, vocabulary consent relations have been proposed as a mechanism for partial reuse and integration of ontologies, and the formalism for such a mechanism has been given to underpin the idea of vocabulary consent relations. As compared to the import mechanism built in OWL, consent relation (as well as weak consent relation) supports the partial reuse at vocabulary level, and it provides a mechanism to inherit the meaning of relevant vocabularies from source ontology into target ontology. The presented approach is fine-grained and flexible, while the owl:import mechanism is at ontology level, and its ability of combining models is restricted to the import of complete model of source ontology. As compared to the C-OWL, our approach uses vocabulary consent relationships as context for possible interpretations of ontologies, while C-OWL uses bridge rules as context [6]. Both of them are based on the local models semantics [7, 8] and distributed description logics [9].

In addition, some semantic conditions (in definition 3) are given to ensure the expected consequences of consent relationships. On one hand, when one consent relationship is to be used, the corresponding conditions need to be checked, which may involve meaning coordination and negotiation. On the other hand, these semantic conditions can help agents to conduct distributed reasoning along with consent relations. Furthermore, the semantic conditions proposed in this paper can help people (and then agents) identify some consent relations among ontologies to be integrated. At least, these semantic conditions can be used as criteria for vocabulary reuse as well as vocabulary mapping.

The research work given in this paper is primary in some aspects. As pointed out at the end of section 2, the requirement on vocabulary consent relations needs to be elicited, and the expected consequence of different kind of consent relations needs further exploration. In addition, the research work needs to be extended to cope with OWL DL constructs. Other interesting research related to this paper includes context capturing and distributed reasoning among multiple agents with their own ontologies.

5 Acknowledgments

This paper is jointly supported by NSFC with project no. 60173036, 973 Program with project no. 2003CB316904, JSNSF with project no. BK2003001 and Hwa-Ying Culture and Education Foundation. The authors are also grateful to the anonymous referees for their comments and suggestions.

Reference

1. Tim Berners-Lee, James H, Ora Lassila. The Semantic Web. Scientific American. May, 2001.
2. Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks, eds. OWL Web Ontology Language Semantics and Abstract Syntax. W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/owl-semantics/
3. Dan Brickley and R.V. Guha, eds. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/rdf-schema/
4. Patrick Hayes, ed. RDF Semantics. W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/rdf-mt/
5. Dave Beckett, ed. RDF/XML Syntax Specification (Revised). W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/rdf-syntax-grammar/
6. Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, et al. C-OWL: Contextualizing Ontologies, 2nd International Semantic Web Conference (ISWC'2003), October 2003, Sanibel Island, Florida, (USA). Lecture Notes in Computer Science Volume 2870, September 2003, Pages: 164 – 179.
7. F. Giunchiglia and L. Serafini. Multilanguage hierarchical logics (or: how we can do without modal logics). Artificial Intelligence, 65:29–70, 1994.
8. C. Ghidini, F. Giunchiglia. Local Models Semantics, or Contextual Reasoning = Locality + Compatibility. Artificial Intelligence, 127(2):221–259, 2001.
9. A. Borgida and L. Serafini. Distributed description logics: Directed domain correspondences in federated information sources. In R. Meersman and Z. Tari, editors, On The Move to Meaningful Internet Systems 2002: CoopIS, Doa, and ODBase, volume 2519 of LNCS, pages 36 – 53. Springer Verlag, 2002.
10. Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics, 1(1):7-26, 2003.

A Channel-Theoretic Foundation for Ontology Coordination

Marco Schorlemmer[1] and Yannis Kalfoglou[2]

[1] Escola Universitària de Tecnologies d'Informació i Comunicació
Universitat Internacional de Catalunya
marco@cir.unica.edu

[2] School of Electronics and Computer Science
University of Southampton
y.kalfoglou@ecs.soton.ac.uk

Abstract. We address the mathematical foundations of the ontology coordination problem and investigate to which extend the Barwise-Seligman theory of information flow may provide a faithful theoretical description of the problem. We give a formalisation of the coordination of populated ontologies based on instance exchange that captures progressive partial semantic integration. We also discuss the insights that the Barwise-Seligman theory provides to the general ontology coordination problem.

1 Introduction

For two systems to interoperate, exchanging syntax is insufficient, because systems also need to agree upon the meaning of the communicated syntactic constructs. Separate applications, though, are most often engineered assuming different, sometimes even incompatible, conceptualisations. Ontologies have been advocated as a solution to this semantic heterogeneity: separate applications would need to match their own conceptualisations against a common ontology of the application domain, so that all communication is done according to the constraints derived from the ontology.

Although the use of ontologies may indeed favour semantic interoperability, it relies on the existence of agreed domain ontologies in the first place. Furthermore, these ontologies will have to be as complete and as stable for a domain as possible, because different versions only introduce more semantic heterogeneity. Thus, semantic-integration approaches based on *a priori* common domain ontologies may be useful for clearly delimited and stable domains, but they are untenable and even undesirable in highly distributed and dynamic environments such as the Web. In such an environment, it is more realistic to progressively achieve certain levels of semantic interoperability by coordinating and negotiating the meaning attached to syntactic constructs on the fly. Although we are skeptical that *meaning* as such can ever be coordinated or negotiated in a way such that all systems share the understanding of a communicated concept, we do argue that communication between separate systems will hardly ever be achieved if we lack the necessary commodity for meaning to be coordinated and negotiated in the first place: information.

This puts us within the philosophical tradition put forth by Dretske [3], which sees information as prior to meaning, namely as an interpretation-independent objective commodity that can be studied by its own right. Consequently, we believe that any satisfactory formalisation of semantic interoperability needs to be built upon a mathematical theory capable of describing under which circumstances information flow occurs. We shall use Barwise and Seligman's channel theory for this purpose [1]. It constitutes a general mathematical theory that aims at describing the information flow in any kind of distributed system.

In our previous work we have been starting from the Barwise-Seligman theory of information flow in order to formalise and automate semantic interoperability [5, 6]. In this paper we investigate the ways in which the Barwise-Seligman theory applies to the problem of ontology coordination. We do not present a fully-fledged theory for ontology coordination, nor do we provide an ontology coordination methodology or procedure. Instead, our aim here is to explore if the insights about information and its flow provided by the Barwise-Seligman theory translate to the ontology coordination problem.

2 Ontology Coordination

Before applying all the channel-theoretic machinery to the ontology coordination problem, we first need to delimit the problem and state the assumptions upon which we build the theoretical framework.

We assume a scenario in which two agents A_1 and A_2 want to interoperate, but in which each agent A_i has its knowledge represented according to its own conceptualisation, which we assume to be explicitly specified according to its own ontology O_i. By this we mean a concept of O_1 will always be considered semantically distinct *a priori* from any concept of O_2, even if they happen to be syntactically equal, unless there is sufficient semantic evidence that it means the same to A_1 as it does to A_2. Furthermore, we assume that the agent's ontologies are not open to other agents for inspection, so that semantic heterogeneity can not be solved by "looking into each agents' head." Hence, an agent may learn about the ontology of another agent only through interaction. Thus, following an approach similar to that of Wang and Gasser described in [10], if A_1 wants to explain A_2 the meaning of a concept, it can use an instance classified under this concept as a representation of it.

Take, for example, the issues one has to take into account when attempting to align the English concepts river and stream of O_1 with the French concepts of fleuve and rivière of O_2. According to Sowa,

> In English, size is the feature that distinguishes river from stream; in French, a fleuve is a river that flows into the sea, and a rivière is either a river or a stream that runs into another river. [9]

Given these distinct conceptualisations, A_1 may explain to A_2 what a river is by informing A_2 that Ohio is a river. In principle, agents may handle different instance sets as A_1 may be situated in the context of the North-American geography (Mississippi, Ohio, Captina) while A_2 may be situated in the context of the French geography (Rhône,

Saône, Roubion). But for any successful explanation of foreign concepts by exchanging information about instances, one needs to assume that A_2 will be able to identify instances of A_1 (e.g., Ohio) as belonging to the same *domain of discourse* D as its own instances (Rhône, Saône, Roubion)—which, for this particular scenario, consists of all water-flowing entities—and that it will be able to classify any new elements of D according to its own ontology.

In fact, by lacking any *a priori* domain ontology about water-flowing entities, it is hard to see how agents A_1 and A_2 could coordinate their respective ontologies O_1 and O_2 in another way. It is the assumption that A_1's and A_2's instances belong to a common domain of discourse which makes our approach to ontology coordination possible. Ontology coordination is then the progressive sharing of instances of this domain of discourse and the subsequent communication about how they are classified according to each ontology.

3 Channel-Theoretic Preliminaries

We introduce briefly the main channel-theoretic constructs needed for our foundation for ontology coordination. As we proceed, we shall hint at the intuitions lying behind them, but for a proper in-depth understanding of the theory we refer the interested reader to [1]. In the remainder of the paper we use the prefix 'IF' (information flow) in front of some of the channel-theoretic terminology to distinguish it from their usual meaning.

3.1 IF Classification, Infomorphism, and Channel

In channel theory, each component (or context) of a distributed system is modelled by means of an *IF classification*. The system itself is described by the way IF classifications are connected with each other through *infomorphisms*.

Definition 1. *An* IF classification $\mathbf{A} = \langle tok(\mathbf{A}), typ(\mathbf{A}), \models_{\mathbf{A}} \rangle$*, consists of a set of* tokens $tok(\mathbf{A})$*, a set of* types $typ(\mathbf{A})$ *and a* classification relation $\models_{\mathbf{A}} \subseteq tok(\mathbf{A}) \times typ(\mathbf{A})$ *that classifies tokens to types.*

Definition 2. *An infomorphism* $f = \langle f^{\wedge}, f^{\vee} \rangle : \mathbf{A} \rightarrow \mathbf{B}$ *from IF classifications* $\mathbf{A}$ *to* $\mathbf{B}$ *is a contravariant pair of functions* $f^{\wedge} : typ(\mathbf{A}) \rightarrow typ(\mathbf{B})$ *and* $f^{\vee} : tok(\mathbf{B}) \rightarrow tok(\mathbf{A})$ *satisfying the following fundamental property, for each type* $\alpha \in typ(\mathbf{A})$ *and token* $b \in tok(\mathbf{B})$*:*

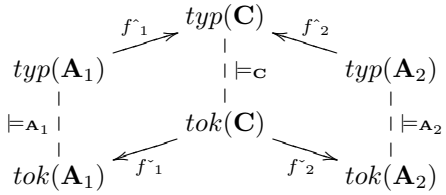
$$\begin{array}{ccc} \alpha & \vdash\!\!\!\xrightarrow{f^{\wedge}} & f^{\wedge}(\alpha) \\ \models_{\mathbf{A}} \mid & & \mid \models_{\mathbf{B}} \\ f^{\vee}(b) & \xleftarrow[f^{\vee}]{} \!\!\!\dashv & b \end{array}$$

$$f^{\vee}(b) \models_{\mathbf{A}} \alpha \quad \textit{iff} \quad b \models_{\mathbf{B}} f^{\wedge}(\alpha)$$

Definition 3. *A* distributed IF system $\mathcal{A}$ *consists of an indexed family* $cla(\mathcal{A}) = \{\mathbf{A}_i\}_{i \in I}$ *of IF classifications together with a set* $inf(\mathcal{A})$ *of infomorphisms all having both domain and codomain in* $cla(\mathcal{A})$.

The basic construct of channel theory is that of an *IF channel* between two IF classifications. It models the information flow between components:

Definition 4. *An* IF channel *consists of two IF classifications* $\mathbf{A}_1$ *and* $\mathbf{A}_2$ *connected through a core IF classification* $\mathbf{C}$ *via two infomorphisms* f_1 *and* f_2*:*



3.2 IF Theory and Logic

Channel theory is based on the understanding that the flow of information is a result from the regularities of a distributed system. These regularities are implicit in the representation of the system as a distributed IF system of connected IF classifications, but we can make them explicit in a logical fashion by means of IF theories and IF logics:

Definition 5. *An* IF theory $T = \langle typ(T), \vdash \rangle$ *consists of a set* $typ(T)$ *of types, and a binary relation* $\vdash$ *between subsets of* $typ(T)$*. Pairs* $\langle \Gamma, \Delta \rangle$ *of subsets of* $typ(T)$ *are called* sequents*. If* $\Gamma \vdash \Delta$*, for* $\Gamma, \Delta \subseteq typ(T)$*, then the sequent* $\Gamma \vdash \Delta$ *is called a* constraint*.* T *is* regular *if for all* $\alpha \in typ(T)$ *and all sets* $\Gamma, \Gamma', \Delta, \Delta', \Sigma', \Sigma_0, \Sigma_1$ *of types:*

1. Identity: $\alpha \vdash \alpha$
2. Weakening: *If* $\Gamma \vdash \Delta$*, then* $\Gamma, \Gamma' \vdash \Delta, \Delta'$
3. Global Cut: *If* $\Gamma, \Sigma_0 \vdash \Delta, \Sigma_1$ *for each partition* $\langle \Sigma_0, \Sigma_1 \rangle$ *of* Σ'*, then* $\Gamma \vdash \Delta$.[3]

Definition 6. *An* IF logic $\mathfrak{L} = \langle tok(\mathfrak{L}), typ(\mathfrak{L}), \models_{\mathfrak{L}}, \vdash_{\mathfrak{L}}, N_{\mathfrak{L}} \rangle$ *consists of an IF classification* $cla(\mathfrak{L}) = \langle tok(\mathfrak{L}), typ(\mathfrak{L}), \models_{\mathfrak{L}} \rangle$*, a regular IF theory* $th(\mathfrak{L}) = \langle typ(\mathfrak{L}), \vdash_{\mathfrak{L}} \rangle$ *and a subset of* $N_{\mathfrak{L}} \subseteq tok(\mathfrak{L})$ *of* normal tokens*, which satisfy all the constraints of* $th(\mathfrak{L})$*; a token* $a \in tok(\mathfrak{L})$ *satisfies a constraint* $\Gamma \vdash \Delta$ *of* $th(\mathfrak{L})$ *if, when* a *is of all types in* Γ*,* a *is of some type in* Δ*. An IF logic* $\mathfrak{L}$ *is* sound *if* $N_{\mathfrak{L}} = tok(\mathfrak{L})$*.*

Regularity arises from the observation that, given any classification of tokens to types, the set of all sequents that are satisfied by all tokens always fulfill Identity, Weakening, and Global Cut.

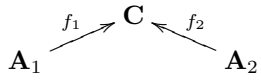
Every classification determines a *natural IF logic*, which captures the regularities of the classification in a logical fashion.

Definition 7. *The* natural IF logic *is the IF logic* $Log(\mathbf{C})$ *generated from an IF classification* $\mathbf{C}$*, and has as classification* $\mathbf{C}$*, as regular theory the theory whose constraints are the sequents satisfied by all tokens, and whose tokens are all normal.*

[3] A partition of Σ' is a pair $\langle \Sigma_0, \Sigma_1 \rangle$ of subsets of Σ', such that $\Sigma_0 \cup \Sigma_1 = \Sigma'$ and $\Sigma_0 \cap \Sigma_1 = \emptyset$; Σ_0 and Σ_1 may themselves be empty (hence it is actually a quasi-partition).

3.3 Distributed IF Logic

The key channel-theoretic construct we shall use in order model the semantic interoperability between agents with different ontologies is that of a *distributed IF logic*, which is the logic that represents the flow of information occurring in a distributed system. Semantic interoperability between agents A_1 and A_2 is then described by the IF theory of the distributed IF logic of IF channel



representing the information flow between $\mathbf{A}_1$ and $\mathbf{A}_2$, and which describes how the different types from $\mathbf{A}_1$ and $\mathbf{A}_2$ are logically related to each other, both respecting the local IF classification systems of each agent and interrelating types whenever there is a similar semantic pattern (i.e., a similar way communities classify related tokens). The distributed IF logic is defined by *moving* an IF logic on the core $\mathbf{C}$ of the channel to the sum of components $\mathbf{A}_1 + \mathbf{A}_2$.

Definition 8. *Given an infomorphism $f : \mathbf{A} \rightarrow \mathbf{B}$ and an IF logic $\mathfrak{L}$ on $\mathbf{B}$, the* inverse image $f^{-1}[\mathfrak{L}]$ *of $\mathfrak{L}$ under f is the IF logic on $\mathbf{A}$, whose theory is such that $\Gamma \vdash \Delta$ is a constraint of $th(f^{-1}[\mathfrak{L}])$ iff $f^{\wedge}[\Gamma] \vdash f^{\wedge}[\Delta]$ is a constraint of $th(\mathfrak{L})$, and whose normal tokens are $N_{f^{-1}[\mathfrak{L}]} = \{a \in tok(\mathbf{A}) \mid a = f^{\vee}(b)$ for some $b \in N_{\mathfrak{L}}\}$. If $f^{\vee}$ is surjective on tokens and $\mathfrak{L}$ is sound, then $f^{-1}[\mathfrak{L}]$ is sound.*

Definition 9. *Given an IF channel $\mathcal{C} = \{f_{1,2} : \mathbf{A}_{1,2} \rightarrow \mathbf{C}\}$ and an IF logic $\mathfrak{L}$ on its core $\mathbf{C}$, the* distributed IF logic $DLog_{\mathcal{C}}(\mathfrak{L})$ *is the inverse image of $\mathfrak{L}$ under the sum infomorphisms $f_1 + f_2 : \mathbf{A}_1 + \mathbf{A}_2 \rightarrow \mathbf{C}$. This sum is defined as follows: $\mathbf{A}_1 + \mathbf{A}_2$ has as set of tokens the Cartesian product of $tok(\mathbf{A}_1)$ and $tok(\mathbf{A}_2)$ and as set of types the disjoint union of $typ(\mathbf{A}_1)$ and $typ(\mathbf{A}_2)$, such that for $\alpha \in typ(\mathbf{A}_1)$ and $\beta \in typ(\mathbf{A}_2)$, $\langle a, b\rangle \models_{\mathbf{A}_1+\mathbf{A}_2} \alpha$ iff $a \models_{\mathbf{A}_1} \alpha$, and $\langle a, b\rangle \models_{\mathbf{A}_1+\mathbf{A}_2} \beta$ iff $b \models_{\mathbf{A}_2} \beta$. Given two infomorphisms $f_{1,2} : \mathbf{A}_{1,2} \rightarrow \mathbf{C}$, the sum $f_1 + f_2 : \mathbf{A}_1 + \mathbf{A}_2 \rightarrow \mathbf{C}$ is defined by $(f_1 + f_2)^{\wedge}(\alpha) = f_i(\alpha)$ if $\alpha \in \mathbf{A}_i$ and $(f_1 + f_2)^{\vee}(c) = \langle f^{\vee}{}_1(c), f^{\vee}{}_2(c)\rangle$, for $c \in tok(\mathbf{C})$.*

3.4 Ontologies in Channel Theory

For the purposes of ontology coordination described in this paper, we adopt a definition of ontology that includes some of its core components: *Concepts*, organised in an *is-a hierarchy*, and notions of *disjointness* of two concepts—when no instance can be considered of both concepts—and *coverage* of two concepts—when all instances are covered by two concepts.[4] Disjointness and coverage are typically specified by means of ontological axioms. In this paper we take these kind of axioms into account including disjointness and coverage into the hierarchy of concepts by means of two binary relations '$\perp$' and '$|$', respectively. In [5] we included also *relations* over concepts in our core treatment of ontologies. We have left them out here for the ease of presentation.

[4] Both disjointness and coverage can easily be extended to more than two concepts.

Definition 10. *An* ontology *is a tuple* $\mathcal{O} = (C, \leqslant, \perp, |)$ *where*

1. C *is a finite set of concept symbols;*
2. $\leqslant$ *is a reflexive, transitive and anti-symmetric relation on* C *(a partial order);*
3. $\perp$ *is a symmetric and irreflexive relation on* C *(disjointness);*
4. $|$ *is a symmetric relation on* C *(coverage); and*

When an ontology $\mathcal{O} = (C, \leqslant, \perp, |)$ is used in some particular application domain, we need to populate it with instances. First, we will have to classify objects of a set X according to the concept symbols in C by defining a binary classification relation $\models_{\mathbf{C}}$. This determines an IF classification $\mathbf{C} = (X, C, \models_{\mathbf{C}})$, where $X = tok(\mathbf{C})$ and $C = typ(\mathbf{C})$. The classification relation $\models_{\mathbf{C}}$ will have to be defined in such a way that the partial order $\leqslant$, the disjointness $\perp$, and the coverage $|$ are respected:

Definition 11. *A* populated ontology *is a tuple* $\widetilde{\mathcal{O}} = (\mathbf{C}, \leqslant, \perp, |)$ *such that* $\mathbf{C} = (X, C, \models_{\mathbf{C}})$ *is an IF classification, and* $\mathcal{O} = (C, \leqslant, \perp, |)$ *is an ontology, and for all* $x \in X$ *and* $c, d \in C$*,*

1. *if* $x \models_{\mathbf{C}} c$ *and* $c \leqslant d$*, then* $x \models_{\mathbf{C}} d$*;*
2. *if* $x \models_{\mathbf{C}} c$ *and* $c \perp d$*, then* $x \not\models_{\mathbf{C}} d$*;*
3. *if* $c \mid d$*, then* $x \models_{\mathbf{C}} c$ *or* $x \models_{\mathbf{C}} d$*.*

Our approach to ontology coordination uses the fact that, in the context of channel theory, a populated ontology $\widetilde{\mathcal{O}} = (\mathbf{C}, \leqslant, \perp, |)$—with $\mathbf{C} = (X, C, \models_{\mathbf{C}})$—determines a local logic $\mathcal{L} = (X, C, \models_{\mathbf{C}}, \vdash)$ whose theory $(C, \vdash)$ is given by the smallest regular consequence relation (i.e., the smallest relation closed under Identity, Weakening, and Global Cut) such that, for all $c, d \in C$,

$$c \vdash d \quad \text{iff} \quad c \leqslant d \qquad\qquad c, d \vdash \quad \text{iff} \quad c \perp d \qquad\qquad \vdash c, d \quad \text{iff} \quad c \mid d$$

4 Progressive Semantic Integration

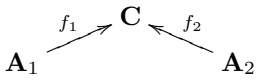
In order to formalise the semantic integration of a collection of agents via the precise mathematical construct of an IF channel, in [6] we articulated the following four steps:

1. Modelling the populated ontologies of agents by means of IF classifications.
2. Defining an IF channel—its core and infomorphisms—connecting the agents' IF classifications.
3. Defining an IF logic on the core of the IF channel representing the information flow between agents.
4. Distributing the IF logic to the sum of agent IF classifications to obtain the IF theory that describes the desired semantic interoperability.

These steps need to be understood in the context of a theoretical exercise, and hence will hardly be implemented directly as engineering steps in actual interoperability scenarios. In particular, the definition of an IF channel and an IF logic on the core of this channel representing the information flow between agents (steps 2 and 3) requires a global view of all involved parties, which we seldom will possess in general. On the contrary, we started from the assumption that the agents' ontologies are not open to other agents for inspection, and that an agent learns about the ontology of another agent only through interaction.

4.1 The Global Ontology

The four steps above determine what we will call the *global ontology* of two semantically integrated agents A_1 and A_2. It is the distributed logic of an IF channel $\mathcal{C}$ connecting IF classifications $\mathbf{A_1}$ and $\mathbf{A_2}$ modelling the agents' populated ontologies $\widetilde{O}_1$ and $\widetilde{O}_2$ respectively:



At the core of IF channel $\mathcal{C}$, $typ(\mathbf{C})$ covers $typ(\mathbf{A}_1)$ and $typ(\mathbf{A}_2)$, while the elements of $tok(\mathbf{C})$ connect tokens from $tok(\mathbf{A}_1)$ with tokens from $tok(\mathbf{A}_2)$. By defining an IF logic on the core of the channel and distributing it to the sum of IF classifications $\mathbf{A_1} + \mathbf{A_2}$ we get the *global ontology* that captures the overall semantic integration of the scenario.

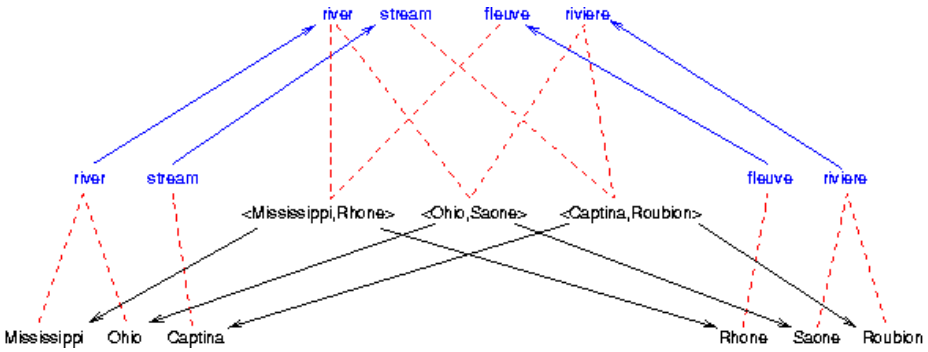


Fig. 1. Aligning ontologies through a pair of maps

For example, an IF channel for the English-French river alignment scenario of Section 2 is shown in Figure 1. At the core of this channel the connections $\langle$Mississippi, Rhône$\rangle$, $\langle$Ohio, Saône$\rangle$, and $\langle$Captina, Roubion$\rangle$ link particular instances of type river or stream together with particular instances of type fleuve or rivière in such a way that their resulting classification into the four concepts river, stream, fleuve, and rivière, determines an IF theory about how these concepts are semantically related. This theory is given by the distributed IF logic of the natural IF logic of the core classification: $DLog_{\mathcal{C}}(Log(\mathbf{C}))$. It includes among its constraints:

$$\vdash \text{river}, \text{rivière} \qquad \text{fleuve} \vdash \text{river}$$
$$\text{stream} \vdash \text{rivière} \qquad \text{fleuve}, \text{stream} \vdash$$

i.e., that river $\mid$ rivière, fleuve $\leqslant$ river, stream $\leqslant$ rivière, and fleuve $\perp$ stream. Other IF channels modelling a different semantic integration are possible in principle, although we defined this one with the particular relationship in mind linking together

big rivers flowing into the sea (Mississippi and Rhône), rivers flowing into other rivers (Ohio and Saône), and streams flowing into other rivers (Captina and Roubion).

In ontology coordination scenarios we cannot assume that we will be able to define a global IF channel that connects $\mathbf{A_1}$ and $\mathbf{A_2}$ directly, capturing thus their semantic integration. In the channel of Figure 1, for example, it is not clear from where we would gain the additional understanding that allowed us to link tokens in the way we did. Nor can we assume that we ever will be able to define such a channel completely, linking all tokens and defining an IF theory on the union of all types. Therefore, the global IF channel is not appropriate as a mathematical model for describing the process of ontology coordination.

4.2 The Coordinated Channel

We shall model ontology coordination with a *coordinated channel* instead, an IF channel that captures how $\widetilde{O}_1$ and $\widetilde{O}_2$ are progressively coordinated, and which captures the semantic integration achieved through interaction between A_1 and A_2. As we have described in Section 2, if A_1 wants to explain A_2 the meaning of a concept, it can do so using an instance classified under this concept as a representation of it.

The coordinated channel is a mathematical model of this coordination that captures the *degree of participation* of an agent A_i at any stage of the coordination process. This degree is determined both, at the type and at the token level, since

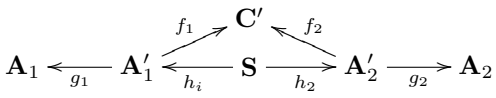
- an agent A_i will have attempted to explain a subset of its concepts to other agents, and
- other agents will have shared with agent A_i some of its instances, incrementing in this way the instance set managed originally by agent A_i.

This degree of participation can easily be captured with an infomorphism $g_i : \mathbf{A}'_i \rightarrow \mathbf{A}_i$, for which functions $\hat{g}_i$ and $\check{g}_i$ are the inclusions $typ(\mathbf{A}'_i) \subseteq typ(\mathbf{A}_i)$ and $tok(\mathbf{A}_i) \subseteq tok(\mathbf{A}'_i)$, respectively. The coordination is then established not between the original IF classifications $\mathbf{A}_i$, but between the *subclassifications* $\mathbf{A}'_i$ that result from the interaction carried out so far:

$$\mathbf{A}_1 \xleftarrow[g_1]{} \mathbf{A}'_1 \xrightarrow{f_1} \mathbf{C}' \xleftarrow{f_2} \mathbf{A}'_2 \xrightarrow[g_2]{} \mathbf{A}_2$$

In Section 2 we argued that although agents may handle different instance sets, any successful explanation of foreign concepts by exchanging information about instances will need to assume that A_2 is able to identify instances of A_1 as belonging to a theoretically domain of discourse D common to its own instances, and that it will be able to classify, in theory, any element of D according to its own ontology. We also assumed disjoint sets of concepts among agents. These assumptions ultimately determine the coordinated channel $\mathcal{C}'$; this is mathematically captured by an IF classification $\mathbf{S}$ with no concepts, $typ(\mathbf{S}) = \emptyset$, the domain of discourse as its instance set, $tok(\mathbf{S}) = D$, and empty classification relation.

The coordinated IF channel that captures the semantic integration achieved by the agents is mathematically defined by taking the category-theoretical *colimit* (see, e.g., [7]) $\mathcal{C}' = colim\{\mathbf{A}'_1 \leftarrow \mathbf{S} \rightarrow \mathbf{A}'_2\}$ of the diagram linking the IF subclassifications that model each agent's participation through the assumptions of the scenario:



4.3 Partial Semantic Integration

The diagram above is a general model of the coordinated channel between to agents, and it faithfully captures the semantic integration between them, according to the Barwise-Seligman theory of information flow. Initially, when the agents have not yet coordinated themselves, the IF classifications modelling the agents' participation have no concepts since none of them have been communicated yet, and the instance set of the core of the coordinated channel is empty (as no instances have been shared yet):

$$typ(\mathbf{A}'_i) = \emptyset \qquad typ(\mathbf{C}') = \emptyset$$
$$tok(\mathbf{A}'_i) = tok(\mathbf{A}_i) \qquad tok(\mathbf{C}') = \emptyset$$

After A_1 told A_2 that $Ohio \models$ river and A_2 told A_1 that $Ohio \models$ rivière, A_1 participates in the coordinated channel with concept river and A_2 participates in the coordinated channel with concept rivière. Furthermore A_2 will have extended its instance set with the shared instance $Ohio$, resulting in the coordinated channel of Figure 2.

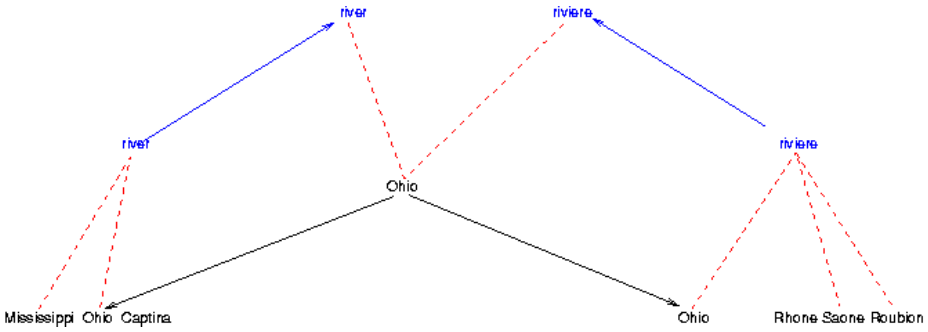


Fig. 2. Partially coordinated channel

Furthermore, after A_2 told A_1 that $Roubion \models$ rivière and A_1 told A_2 that $Roubion \models$ stream, new concepts participate in the ontology coordination, and new instances are shared, resulting in the newly coordinated channel of Figure 3.

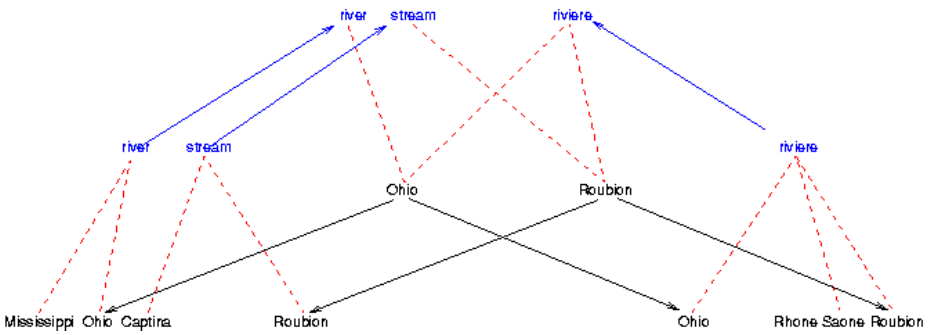


Fig. 3. Partially coordinated channel

At each stage a new coordinated channel arises. The distributed IF logic of the natural logic determined by the core of each new channel **captures the semantic integration achieved so far**. For instance, for this last coordinated channel the theory of the distributed IF logic $DLog_{\mathcal{C}'}(Log((\mathbf{C}')))$ would include among its constraints:

$$\vdash \text{rivière} \qquad \vdash \text{river}, \text{stream} \qquad \text{river}, \text{stream} \vdash$$

4.4 Complete Semantic Integration

In the optimal limit case, all concepts would be eventually communicated and all instances shared, which would yield a situation of complete semantic integration in which the IF classifications modelling the agents' participation in the coordination would include each agent's concepts and would have the domain of discourse as their instance set:

$$typ(\mathbf{A}'_i) = typ(\mathbf{A}_i) \qquad typ(\mathbf{C}') = \bigcup_i typ(\mathbf{A}_i)$$
$$tok(\mathbf{A}'_i) = D \qquad tok(\mathbf{C}') = D$$

This is an ideal scenario, in which agents would have exchanged their entire IF classification (all tokens, all types, and the entire classification relation). In our example, complete semantic integration would have been achieved with the coordinated channel shown in Figure 4. The distributed IF logic of this channel is equivalent to the global ontology discussed above.

Because in practice complete semantic integration will seldom be achieved (e.g., because it would be computationally too expensive) the ontology coordination process will usually yield only a partial semantic integration involving a fraction of communicated types and shared instances. In these cases it is important to have a faithful formalisation of the resulting situation, which we believe is achieved with its modelling as a coordinated IF channel.

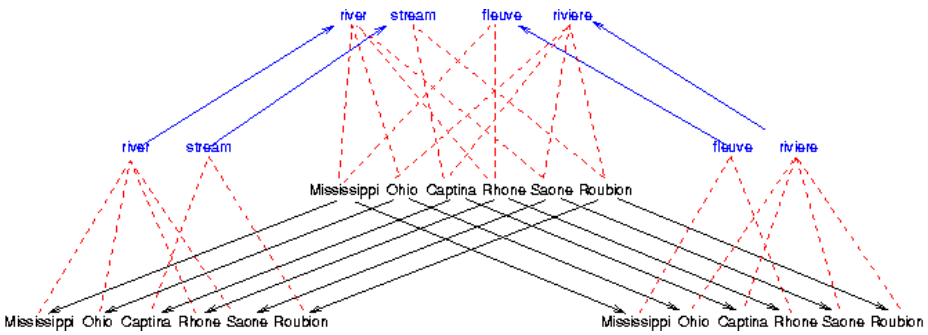


Fig. 4. Completely coordinated channel

5 Concluding Discussion

Channel theory emphasises that, since information is carried by particular tokens, information flow crucially involves both types and tokens. Barwise and Seligman realised the fundamental duality between types and tokens, which is central to all channel-theoretic constructions. Thus, although ontology coordination is usually thought of as a process during which concepts of separate ontologies are being aligned at the type-level, the logical relationship between concepts arises when tokens are being connected by means of an IF channel. Knowing what these connections at the token-level are is therefore fundamental for determining the semantic integration of ontologies at the type-level.

In this paper, we have been formalising an ontology coordination approach in which token connection is the result of instance passing between agents. But the general formalisation based on channel theory presented here provides a wide view about what we can consider to be a *token* and a *connection between tokens*. This allows for accommodating different understandings of semantics—depending on the particularities of the interoperability scenario—whilst retaining the core aspect that will allow coordination among agents: connections through their tokens. Schorlemmer showed in [8] how the type-token duality helps to pin down some of the reasons why ontologies appear to be insufficient in certain interoperability scenarios for which a common verified ontology is not enough for knowledge sharing [2]. Depending on the scenario being analysed, the role of tokens is taken either by instances, model-theoretic structures, or even proof-theoretic derivations. In [6], for example, we showed how the coordination of various UK and US government ministries can be derived from a partial alignment of ministerial responsibilities, which take the role of connected tokens for that particular scenario.

An information-theoretic analysis of ontology coordination based on channel theory highlights the fact that a coordination process can hardly be absolute. On the contrary, not only is it relative to the respective ontologies being coordinated, but also

1. to the way ontologies are actually used in the context of specific application domains (what we have been calling the populated ontologies);

2. to the way ontologies are characterised as IF logics: the particular understanding of semantics of the interoperability scenario is relative to our choice of types and tokens and its classification relation; (this is closely related to what Farrugia calls the *logical setup*, and which he claims needs to be established first before any meaning negotiation between agents can start [4];)
3. to the way ontologies are linked together via connected tokens: as discussed in [8] reliable semantic integration is only guaranteed on connected tokens, which nicely includes into the framework the unavoidable imperfections of most ontology coordination processes, unless complete semantic integration is achieved.

It would be interesting, for instance, to explore the channel-theoretical notion of *induced IF logic* in the ontology coordination context. This logic characterises how an agent extends its own ontology with the understanding it has gained of other agents' ontologies *relative to the coordinated channel*. This logic is defined by moving the distributed IF logic of the coordinated channel to its restriction to one particular agent's IF classification. It turns out that the resulting induced IF logic is only sound and complete when the infomorphisms constituting the coordinated channel are surjective on tokens (see Definition 8). Such a particular case is when we achieve complete semantic integration, but it would be desirable to find conditions for ontology coordination processes that, without obtaining complete semantic integration, lead to coordinated channels for which sound and complete induced IF logics exist.

Acknowledgments. M. Schorlemmer is supported by a *Ramón y Cajal* Research Fellowship from Spain's Ministry of Science and Technology. Y. Kalfoglou is supported by the Advanced Knowledge Technologies (AKT) Interdisciplinary Research Collaboration (IRC), which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. The AKT IRC comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University.

References

1. J. Barwise and J. Seligman. *Information Flow*. Cambridge University Press, 1997.
2. F. Corrêa da Silva et al. On the insufficiency of ontologies: Problems in knowledge sharing and alternative solutions. *Knowledge-Based Systems*, 15(3):147–167, 2002.
3. F. Dretske. *Knowledge and the Flow of Information*. MIT Press, 1981.
4. J. Farrugia. Logical systems: Towards protocols for Web-based meaning negotiation. In *Meaning Negotiation, Papers form the AAAI Workshop*, pages 56–59. The AAAI Press, 2002.
5. Y. Kalfoglou and M. Schorlemmer. IF-Map: An ontology-mapping method based on information-flow theory. In *Journal on Data Semantics I*, LNCS 2800, pages 98–127. Springer, 2003.
6. Y. Kalfoglou and M. Schorlemmer. Formal support for representing and automating semantic interoperability. In *The Semantic Web: Research and Applications. ESWS 2004. Proceedings*, LNCS 3053, pages 45–60, Heraklion, Crete, Greece, May 2004. Springer.
7. B. Pierce. *Basic Category Theory for Computer Scientists*. MIT Press, 1991.
8. M. Schorlemmer. Duality in knowledge sharing. In *7th International Symposium on Artificial Intelligence and Mathematics*, Ft. Lauderdale, Florida, USA, 2002.
9. J. Sowa. *Knowledge Representation and Reasoning*. Brooks/Cole, 2000.
10. J. Wang and L. Gasser. Mutual online ontology alignment. In *OAS'02 Ontologies in Agent Systems, Proc. of the AAMAS 2002 Workshop*, CEUR-WS 66, 2002.

A Classification of Schema-Based Matching Approaches

Pavel Shvaiko

University of Trento, Povo, Trento, Italy
pavel@dit.unitn.it

Abstract. Schema/ontology matching is a critical problem in many application domains, such as, Semantic Web, schema/ontology integration, data warehouses, e-commerce, catalog matching, etc. Many diverse solutions to the matching problem have been proposed so far. In this paper we present a taxonomy of schema-based matching techniques that builds on the previous work on classifying schema matching approaches. Some innovations are in introducing new criteria which distinguish between matching techniques relying on diverse semantic clues. In particular, we distinguish between heuristic and formal techniques at schema-level; and implicit and explicit techniques at element- and structure-level. Based on the classification proposed we overview some of the recent schema/ontology matching systems pointing which part of the solution space they cover.

1 Introduction

Match is a critical operator in many well-known application domains, such as, Semantic Web, schema/ontology integration, data warehouses, e-commerce, XML message mapping, catalog matching, etc. Many solutions to the matching problem include identifying terms in one information source that "match" terms in another information source. The applications can be viewed as graph-like structures containing terms and their inter-relationships. These might be database schemas, taxonomies, or ontologies, for example [14], etc. Match operator takes two graph-like structures as input and produces a mapping between the nodes of the graphs that correspond semantically to each other as output.

Many diverse solutions to the matching problem have been proposed so far, for example [19, 15, 8, 21, 32, 1, 17, 23, 26, 20], etc. In this paper we focus only on schema-based solutions, e.g., matching systems exploiting only intensional information, not instance data. Although, there is a difference between schema and ontology matching (alignment) problems (see next section for details), we believe that techniques developed for each of them can be of a mutual benefit, therefore we discuss schema and ontology matching referring as to the one problem.

With the emergence and proliferation of the Semantic Web, the semantics captured in schemas/ontologies should be also handled at different levels of details. Therefore, there is a need in distinguishing between schema/ontology matching techniques relying on diverse semantic clues. In this paper we present

Fig. 1. Two XML schemas

a taxonomy of schema-based matching techniques that builds on the previous work of E. Rahm and P. Bernstein on classifying schema matching approaches [28]. Some innovations are in introducing new criteria which distinguish between schema/ontology matching techniques relying on diverse semantic clues. In particular, we distinguish between heuristic and formal techniques at schema-level; and implicit and explicit techniques at element- and structure-level.

The rest of the paper is organized as follows. Section 2 provides, via an example, the basic motivations to the schema/ontology matching problem. Section 3 introduces the classification of schema-based approaches and discusses in details possible alternatives. Section 4 overviews some of the recent schema/ontology matching solutions in light of the classification proposed pointing which part of the solution space they cover. Section 5 reports some conclusions.

2 The Matching Problem

2.1 Motivating Example

To motivate the matching problem, let us use two simple XML schemas that are shown in Figure 1 and exemplify one of the possible situations which arise, for example, when resolving a schema integration task.

Suppose an e-commerce company A1 needs to finalize a corporate acquisition of another company A2. To complete the acquisition we have to integrate databases of the two companies. The documents of both companies are stored according to XML schemas A1 and A2 respectively. Numbers in boxes are the unique identifiers of the nodes (sometimes in the following we refer to nodes as elements). A first step in integrating the schemas is to identify candidates to be merged or to have taxonomic relationships under an integrated schema. This step refers to a process of schema matching. For example, the nodes with labels $Office\_Products$ in A1 and in A2 are the candidates to be merged, while the node with label $Digital\_Cameras$ in A2 should be subsumed by the node with label $Photo\_and\_Cameras$ in A1.

2.2 Matching: Syntactic vs. Semantic

In this paper we discuss the problem of matching schemas and ontologies from the generic perspective i.e., we analyze information which is exploited by matching systems in order to produce mappings. In this respect, ontology matching differs substantially from schema matching in the following two (among the others, see [25]) areas:

- Database schemas often do not provide explicit semantics for their data. Semantics is usually specified explicitly at design-time, and frequently is not becoming a part of a database specification, therefore it is not available. Ontologies are logical systems that themselves incorporate semantics (intuitive or formal). For example, in the case of formal semantics we can interpret ontology definitions as a set of logical axioms.
- Ontology data models are richer (the number of primitives is higher, and they are more complex) then schema data models. For example, OWL [30] allows defining inverse properties, transitive properties; disjoint classes, new classes as unions or intersections of other classes, etc.

However, ontologies can be viewed as schemas for knowledge bases. Having defined classes and slots in the ontology, we populate the knowledge base with instance data [25]. Thus, techniques developed for each separate problem can be of interest to each other. On the one side, schema matching is usually performed with the help of heuristic techniques trying to guess semantics encoded in the schemas. On the other side, ontology matching systems (primarily) try to exploit knowledge explicitly encoded in the ontologies. In real-world applications, schemas/ontologies usually have both well defined and obscure labels (terms), and contexts they occur, therefore, solutions from both problems would be mutually beneficial.

Apart from the information that matching systems exploit, the other important dimension of schema/ontology matching is a form of the result they produce. Based on these criteria, following the proposal first introduced in [11], schema/ontology matching systems can be viewed as *syntactic* and *semantic* matching systems. Syntactic matching approaches do not analyze term meaning, and thus semantics, directly. In these approaches semantic correspondences are determined using (i) syntactic similarity measures, usually in [0,1] range, for example, with the help of similarity coefficients [19, 10] or confidence measures [32]; and (ii) syntax driven techniques, for instance techniques, which consider labels as strings, etc., see [21, 19, 15]. The first key distinction of the semantic matching approaches is that mappings are calculated between schema/ontology elements by computing *semantic relations* (for example, equivalent (=) or subsuming elements ($\sqsubseteq, \sqsupseteq$), etc., see for details [12]). The second key distinction is that semantic relations are determined by analyzing *meaning* (concepts, not labels as in syntactic matching) which is codified in the elements and the structure of schemas/ontologies. These ideas are schematically represented in Figure 2.

Let us define the matching problem in terms of graphs [11]. A *mapping element* is a 4-tuple $< ID_{ij},\ n1_i,\ n2_j,\ R >$, i=1,...,N1; j=1,...,N2; where ID_{ij}

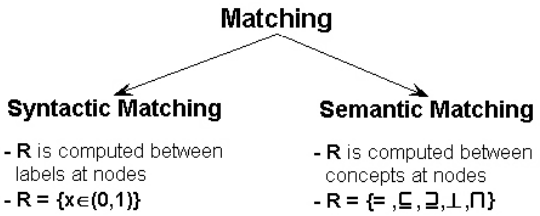


Fig. 2. Matching: Syntactic vs. Semantic

is a unique identifier of the given mapping element; $n1_i$ is the i-th node of the first graph, N1 is the number of nodes in the first graph; $n2_j$ is the j-th node of the second graph, N2 is the number of nodes in the second graph; and R specifies a *similarity relation* (a coefficient in [0,1] range or a semantic relation) holding between the nodes $n1_i$ and $n2_j$. For instance, based on linguistic and structure analysis, the similarity coefficient between nodes with labels $Photo\_and\_Cameras$ in A1 and $Cameras\_and\_Photo$ in A2 in Figure 1 could be 0.67. Thus, the corresponding mapping element is $< ID_{54}, n1_5, n2_4, 0.67 >$. A *mapping* is a set of mapping elements. *Matching*: given two graphs G1 and G2, for any node $n1_i \in$ G1, find a similarity relation R holding with node $n2_j \in$G2.

3 Classification of schema-based matching approaches

At present, there exists a line of semi-automated schema/ontology matching systems, see for instance [19, 15, 8, 21, 32, 1, 17, 23, 26, 20], etc. Good surveys are provided in [28, 31, 16]. The classification of [28] distinguishes between *individual* implementations of match and *combinations* of matchers. Individual matchers comprise *instance-based* and *schema-based*, *element-* and *structure-level*, *linguistic-* and *constrained-based* matching techniques. Also *cardinality* and *auxiliary information* (e.g., dictionaries, global schemas, etc.) can be taken into account. Individual matchers can be used in different ways: directly (*hybrid* matchers), see [19, 1] or combining the results of independently executed matchers (*composite* matchers), see for instance [15, 8, 9].

We focus only on schema-based approaches, and therefore consider only schema/ontology information, not instance data [1]. There are two levels of granularity while performing schema-based matching: element-level and structure-level. Element-level matching techniques compute mapping elements by analyzing individual labels/concepts at nodes; structure-level techniques compute mapping elements by analyzing also subgraphs.

With the emergence and proliferation of the Semantic Web, the semantics captured in schemas/ontologies should be also handled at different levels of details. Therefore, there is a need in distinguishing between schema/ontology

[1] Prominent solutions of instance-based schema/ontology matching as well as possible extensions of the instance-based part of the classification of [28] can be found in [8] and [17] correspondingly.

matching techniques relying on diverse semantic clues. We introduce for individual matchers the following classification cretiria:

- *Heuristic vs formal.* Matching techniques can have either heuristic or formal ground. The key characteristic of the heuristic techniques is that they try to guess relations which may hold between similar labels or graph structures. The key characteristic of the formal techniques is that they have model-theoretic semantics which is used to justify their results.
- *Implicit vs explicit.* These matching techniques rely either on implicitly or explicitly codified semantic information. Implicit techniques are syntax driven techniques: examples are techniques, which consider labels as strings, or analyze data types, or soundex of schema/ontology elements. Explicit techniques exploit the semantics of labels. These techniques are based on the use of tools, which explicitly codify semantic information, e.g., thesauruses, ontologies, etc.

To make the distinctions between the categories proposed more clear, we revised a schema-based part of the classification of matching techniques by E. Rahm and P. Bernstein [28], see Figure 3. All the innovations are marked in bold type. Let us discuss the main alternatives (also indicating in which matching systems they were exploited) according to the above classification criteria in more detail. We omit in our further discussions heuristic element-level implicit techniques as well as heuristic structure-level implicit constrained-based techniques because they appear in a revised classification without changes in respect to the original publication. We also renamed linguistic techniques into string-based techniques, to discard from this category thesaurus look-up methods (they appear in the other category) and methods that perform morphological analysis of strings, which we view only as a preprocessing part, for example, for matching techniques based on lexicons, etc.

3.1 Heuristic techniques

Element-level explicit techniques

- *Precompiled thesaurus and domain ontologies.* A precompiled thesaurus usually stores domain knowledge as entries with synonym, hypernym and other relations. For example, in Figure 1 elements *NKN* in A1 and *Nikon* in A2 are treated by a matcher as synonyms from the thesaurus look up: syn key - "NKN:Nikon = syn", see, for instance [19]. In some cases domain ontologies (notice only those, which fall into this category, with intuitive semantics) also can be used as a source of auxiliary information, see, for example [23].
- *Lexicons.* The approach is to use lexicons to obtain meaning of terms used in schemas/ontologies. For example, WordNet [24] is an electronic lexical database for English (and other languages), where various *senses* (possible meanings of a word or expression) of words are put together into sets of synonyms. Relations between schema/ontology elements can be computed in terms of bindings between WordNet senses, see, for instance [12,

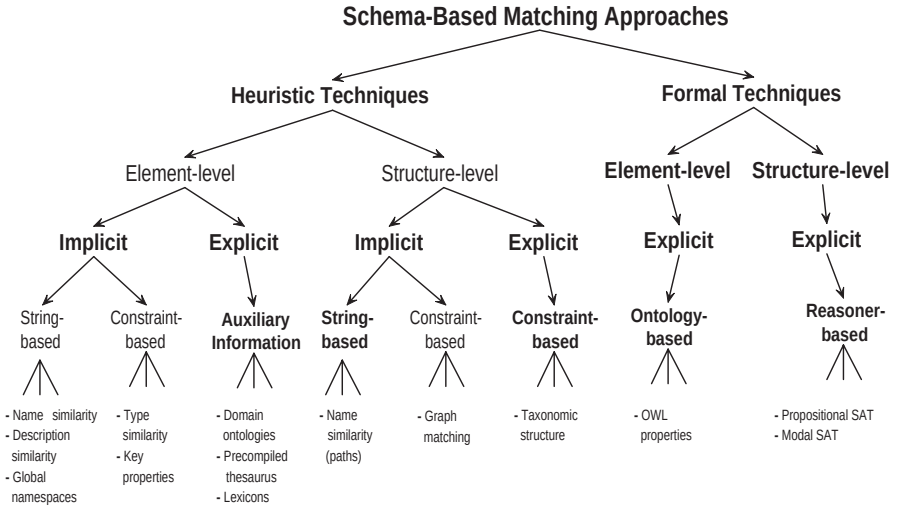


Fig. 3. A revised classification of schema-based matching approaches

4]. For example, in Figure 1 a matcher may learn from WordNet (with a prior morphological preprocessing of labels performed) that "Camera" in A1 is a hypernym for "Digital Camera" in A2, and, therefore conclude that element $Digital\_Cameras$ in A2 should be subsumed by the element $Photo\_and\_Cameras$ in A1.

Structure-level implicit techniques

- *Name similarity (paths).* These matchers build long labels by concatenating all labels at nodes in a path to a single string. For example, in Figure 1 the node with label *Cases* in A1 can be codified as $Sale.Consumer\_Electronics.Cell\_Phones.Accessories.Cases$, while the node with label *Cases* in A2 can be codified as $Consumer\_Electronics.Sale.Cameras\_and\_Photos.Accessories.Cases$. Then string-based techniques (e.g., affix, n-gram, edit distance, etc.) can be used by a matcher to compute similarity between the two long strings, see, for instance [15].

Structure-level explicit techniques

- *Taxonomic structure.* These matchers analyze and compare positions of terms (labels) within taxonomies. For example, they take two paths with links between classes defined by the hierarchical relations or by slots and their domains and ranges, compare terms and their positions along these paths, and identify similar terms, see, for instance [26]. The intuition behind taxonomic structure methods is that *is-a* links connect terms that are already similar (being a subset or superset of each other), therefore their neighbors may be also somehow similar. For example, in Figure 1 given

that element *Digital_Cameras* in A2 should be subsumed by the element *Photo_and_Cameras* in A1, a matcher would suggest *FJFLM* in A1 and *FujiFilm* in A2 as an appropriate match.

3.2 Formal techniques

Element-level explicit techniques

- *OWL properties.* OWL [30] is ontology web language with clear, model-theoretic semantics, and hence methods exploiting its constructors are formal element-level methods. For instance, *sameClassAs* constructor explicitly states that one class is equivalent to the other, see for a particular implementation [9]. For example, in Figure 1 one of the possible OWL encodings could specify semantics of the element *Digital_Cameras* in A2 as follows: $Digital\_Cameras = Camera \sqcap DigitalPhoto\_Producer$. An intuitive reading of the above statement is that digital camera means the same thing as a camera, which encodes and stores images digitally. Then, a matcher would determine that the node 7 in A2 has to be subsumed by the node 5 in A1. Possible extensions to the given category would also exploit other OWL constructors: class properties (e.g., enumeration, disjointness), object properties (inverse-of, symmetric, transitive), etc.

Structure-level explicit techniques

- *Propositional satisfiability (SAT).* As from [11, 4] the approach is to translate the matching problem, namely the two graphs (trees) and mapping queries into a propositional formula and then to check it for its validity. By a mapping query we mean here the pair of nodes and a possible semantic relation between them. Notice that SAT deciders are correct and complete decision procedures for propositional satisfiability, and therefore will exhaustively check for all possible mappings.
- *Modal SAT.* As from [29] the approach is to delimit propositional SAT which allows handling only unary predicates (e.g., classes) by admitting binary predicates (e.g., attributes). The key idea is to enhance propositional logics with modal logic (or ALC description logics) operators. Therefore, the matching problem is translated into a modal logic formula which is further checked for its validity using sound and complete satisfiability search procedures.

4 Prototype Matchers

We now look at some recent schema-based state of the art matching systems in light of the classification presented in Figure 3. We also indicate how systems combine individual matchers in their implementations, e.g., in a hybrid or composite manner.

Similarity Flooding (SF). The SF [21] approach as implemented in Rondo [22] utilizes a hybrid matching algorithm based on the ideas of similarity propagation. Schemas are presented as directed labeled graphs; the algorithm manipulates them in an iterative fix-point computation to produce mapping between the nodes of the input graphs. The technique starts from string-based comparison (common prefixes, suffixes tests) of the vertice's labels to obtain an initial mapping which is refined within the fix-point computation. The basic concept behind the SF algorithm is the similarity spreading from similar nodes to the adjacent neighbors through propagation coefficients. From iteration to iteration the spreading depth and a similarity measure are increasing till the fix-point is reached. The result of this step is a refined mapping which is further filtered to finalize the matching process.

Artemis. Artemis (Analysis of Requirements: Tool Environment for Multiple Information Systems) [5] was designed as a module of MOMIS mediator system [1] for creating global views. It performs affinity-based analysis and hierarchical clustering of source schemas elements. Affinity-based analysis represents the matching step: in a hybrid manner it calculates the name, structural and global affinity coefficients exploiting a common thesaurus. The common thesaurus is built with the help of ODB-Tools, WordNet or manual input. It represents a set of intensional and extensional relationships which depict intra- and inter-schema knowledge about classes and attributes of the input schemas. Based on global affinity coefficients, a hierarchical clustering technique categorizes classes into groups at different levels of affinity. For each cluster it creates a set of global attributes - global class. Logical correspondence between the attributes of a global class and source schema's attributes is determined through a mapping table.

Cupid. Cupid [19] implements a hybrid matching algorithm comprising linguistic and structural schema matching techniques, and computes similarity coefficients with the assistance of a precompiled thesaurus. Input schemas are encoded as graphs. Nodes represent schema elements and are traversed in a combined bottom-up and top-down manner. Matching algorithm consists of three phases and operates only with tree-structures to which no-tree cases are reduced. The first phase (linguistic matching) computes linguistic similarity coefficients between schema element names (labels) based on morphological normalization, categorization, string-based techniques (common prefixes, suffixes tests) and a thesaurus look-up. The second phase (structural matching) computes structural similarity coefficients weighted by leaves which measure the similarity between contexts in which individual schema elements occur. The third phase (mapping generation) computes weighted similarity coefficients and generates final mappings by choosing pairs of schema elements with weighted similarity coefficients which are higher then a threshold. Referring to [19], Cupid performs somewhat better overall, then the other hybrid matchers: Dike [27] and Artemis [5].

COMA. COMA (COmbination of MAtching algorithms) [15] is a composite schema matching tool. It provides an extensible library of matching algorithms; a framework for combining obtained results, and a platform for the evaluation of the effectiveness of the different matchers. Matching library is extensible,

and as from [15] it contains 6 individual matchers, 5 hybrid matches, and one reuse-oriented matcher. Most of them implement string-based techniques (affix, n-gram, edit distance, etc.) as a background idea; others share techniques with Cupid (thesaurus look-up, etc.); and reuse-oriented is a completely novel matcher, which tries to reuse previously obtained results for entire new schemas or for its fragments. Schemas are internally encoded as DAGs, where elements are the paths. This fact aims at capturing contexts in which the elements occur. Distinct features of the COMA tool in respect to Cupid, are a more flexible architecture and a possibility of performing iterations in the matching process. Based on the comparative evaluations conducted in [6], COMA dominates Autoplex[2] and Automatch [3]; LSD [7] and GLUE [8]; SF [21], and SemInt [18] matching tools.

QOM. QQM (Quick Ontology Mapping) system [9] adopts the idea of composite matching from COMA [15] to the ontology matching domain. The approach claims that the loss of quality in matching algorithms is marginal (to a standard baseline), however improvement in efficiency can be tremendous. This fact allows QOM producing mappings fast, even for large-size ontologies. Some other innovations in respect to COMA, are in the set of individual matchers based on rules, exploiting explicitly codified knowledge in ontologies, such as information about super- and sub-concepts, super- and sub-properties, etc. Rules are introduced referring to layers of Tim Berners-Lee's Semantic Web "layer cake". At present the system supports 17 rules. Most of the novel individual matchers reside on the *description logics* (for example, rule#5 (R5): if superconcepts are the same, the actual concepts are similar to each other, etc.) and *restrictions* (for example, R15: two entities are the same if they are binded by *sameClassAs* OWL property) layers. QOM also exploits a set of instance-based techniques, this topic is beyond scope of the paper.

Anchor-PROMPT. Anchor-PROMPT [26] (an extension of PROMPT, also formerly known as SMART) is an ontology merging and alignment tool with a sophisticated prompt mechanism for possible matching terms. The anchor-PROMPT is a hybrid alignment algorithm which takes as input two ontologies, (internally represented as graphs) and a set of anchors-pairs of related terms, which are identified with the help of string-based techniques (edit-distance test), or defined by a user, or another matcher computing linguistic similarity, for example [20]. Then the algorithm refines them by analyzing the paths of the input ontologies limited by the anchors in order to determine terms frequently appearing in similar positions on similar paths. Finally, based on the frequencies and a user feedback, the algorithm determines matching candidates. at

S-Match. S-Match [11, 12] is a schema-based schema/ontology matching system implementing semantic matching approach. It takes two graph-like structures (e.g., database schemas or ontologies) as input and returns semantic relations between the nodes of the graphs that correspond semantically to each other as output. Possible semantic relations are: equivalence ($=$), more general ($\sqsupseteq$), less general ($\sqsubseteq$), mismatch ($\perp$), and overlapping ($\sqcap$). The current version of S-Match is a rationalized re-implementation of the CTXmatch system [4] with a

Fig. 4. Characteristics of state of the art matching approaches

			SF [21], [22]	Artemis [5]	Cupid [19]	COMA [15]	QOM [9]	Anchor-Prompt [26]	S-Match [12]
Heuristic	Element-level	Implicit	string-based (2), data types, key properties	domain compatibility	string-based (2), data types, key properties	string-based (4), data types	string-based (1), domains and ranges	string-based (1), domains and ranges	string-based (5)
		Explicit	-	common thesaurus (CT): synonyms, broader terms, related terms	auxiliary thesaurus (synonyms, hypernyms, hyponyms, abbreviations)	auxiliary thesaurus (synonyms, hypernyms, hyponyms, abbreviations)	application-specific vocabulary	-	WordNet: sense-based (2) gloss-based (6)
	Structure-level	Implicit	iterative fix-point computation	-	tree matching weighted by leaves	string-based (1), DAG (tree) matching with a bias towards leaf or children structures (2)	matching of neighbors (2)	bounded paths matching (arbitrary links)	-
		Explicit	-	matching of neighbors via CT	-	-	taxonomic structure (4)	bounded paths matching (processing *is-a* links separately)	-
Formal	Element-level	Explicit	-	-	-	-	OWL properties (1)	-	-
	Structure-level	Explicit	-	-	-	-	-	-	propositional SAT (2)

few added functionalities. S-Match was designed and developed as a platform for semantic matching, namely a highly modular system with the core of computing semantic relations where single components can be plugged, unplugged or suitably customized. It is a hybrid system performing composition of element level techniques. At present, S-Match libraries contain 13 element-level matchers, see [13], and 2 structure-level (JSAT and SAT4J) matchers.

Notice that from the discussed systems, only S-Match returns as output semantic relations, while all the other systems return coefficients rating match quality in [0,1] range. Although, almost all the matching systems analyze term meaning, for example, with the help of a thesaurus, however when they produce a mapping, they encode it in [0,1] range, therefore loosing some information. For example, from a similarity coefficient with value of 0.7 we can not say if the elements it binds are more or less general. We only may conclude that they are similar, and the probability of their equality is 70%. Therefore, only the S-Match system can be considered as a semantic matching system, all other systems, in this sense, are syntactic systems.

Figure 4 briefly summarizes how the matching systems cover the solution space in terms of the proposed classification. Numbers in brackets specify how many matchers of a particular type a system supports. For example, S-Match supports 5 string-based heuristic element-level implicit matchers (prefix, suffix, edit distance, n-gram, and text corpus, see [13]). Figure 4 also testifies that schema/ontology matching research was mainly focused on heuristic techniques so far. Formal element-level and structure-level techniques have been exploited only by two systems: QOM [9] and S-Match [12] correspondingly.

5 Conclusions

This paper presents the taxonomy of schema-based matching approaches, which builds on the previous work by E. Rahm and P. Bernstein on classifying schema matching approaches. We have introduced new criteria which distinguish between schema/ontology matching techniques relying on diverse semantic clues. In particular, we distinguish between heuristic and formal techniques at schema-level; and implicit and explicit techniques at element- and structure-level. We reviewed some of the recent schema/ontology matchers in light of the classification proposed pointing which part of the solution space they cover. Analysis of state of the art systems discussed has shown, that most of them exploit only heuristic techniques, and only a few utilize formal techniques. However, the category of formal techniques was identified only recently as a part of the solution space; its methods provide sound and complete results, and, hence it represents a wide area for the future investigations.

Acknowledgements: This work has been partly supported by the European Knowledge Web network of excellence (IST-2004-507482). Thanks to Fausto Giunchiglia, Paolo Bouquet, and Jerome Euzenat for their insightful comments and suggestions.

References

1. S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. In *SIGMOD Record*, 28(1), pages 54–59, 1999.
2. J. Berlin and A. Motro. Autoplex: Automated discovery of content for virtual databases. In *Proceedings of CoopIS*, pages 108–122, 2001.
3. J. Berlin and A. Motro. Database schema matching using machine learning with feature selection. In *Proceedings of CAiSE*, pages 452–466, 2002.
4. P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: A new approach and an application. In *Proceedings of ISWC*, pages 130–145, 2003.
5. S. Castano, V. De Antonellis, and S. De Capitani di Vimercati. Global viewing of heterogeneous data sources. In *IEEE Transactions on Knowledge and Data Engineering*, number 13(2), pages 277–297, 2001.
6. H.H. Do, S. Melnik, and E.Rahm. Comparison of schema matching evaluations. In *Proceedings of workshop on Web and Databases*, 2002.
7. A. Doan, P. Domingos, and A. Halvey. Reconciling schemas of disparate data sources: A machine-learning approach. In *Proceedings of SIGMOD*, pages 509–520, 2001.
8. A. Doan, J. Madhavan, R. Dhamankar, P. Domingos, and A. Halvey. Learning to map ontologies on the semantic web. In *Very Large Databases Journal, Special Issue on the Semantic Web*, 2003. (to appear).
9. M. Ehrig and Y. Sure. Ontology mapping - an integrated approach. In *Proceedings of ESWS*, pages 76–91, 2004.
10. J. Euzenat and P. Valtchev. An integrative proximity measure for ontology alignment. In *Proceedings of Semantic Integration workshop at ISWC*, 2003.
11. F. Giunchiglia and P. Shvaiko. Semantic matching. In *The Knowledge Engineering Review Journal*, number 18(3), pages 265–280, 2004.

12. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: an algorithm and an implementation of semantic matching. In *Proceedings of ESWS*, pages 61–75, 2004.
13. F. Giunchiglia and M. Yatskevich. Element level semantic matching. In *To appear in Proceedings of Meaning Coordination and Negotiation workshop at ISWC*, 2004.
14. F. Giunchiglia and I. Zaihrayeu. Making peer databases interact - a vision for an architecture supporting data coordination. In *Proceedings of international workshop on Cooperative Information Agents*, pages 18–35, 2002.
15. H.H.Do and E. Rahm. Coma - a system for flexible combination of schema matching approaches. In *Proceedings of VLDB*, pages 610–621, 2001.
16. Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. In *The Knowledge Engineering Review Journal*, number 18(1), pages 1–31, 2003.
17. J. Kang and J.F. Naughton. On schema matching with opaque column names and data values. In *Proceedings of SIGMOD*, pages 205–216, 2003.
18. W.S. Li and C. Clifton. Semantic integration in heterogeneous databases using neural networks. In *Proceedings of VLDB*, pages 1–12, 1994.
19. J. Madhavan, P. Bernstein, and E. Rahm. Generic schema matching with cupid. In *Proceedings of VLDB*, pages 49–58, 2001.
20. D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder. An environment for merging and testing large ontologies. In *Proceedings of KR*, pages 483–493, 2000.
21. S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm. In *Proceedings of ICDE*, pages 117–128, 2002.
22. S. Melnik, E. Rahm, and P. Bernstein. Rondo: A programming platform for generic model management. In *Proceedings of SIGMOD*, pages 193–204, 2003.
23. E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. Observer: An approach for query processing in global information systems based on interoperability between pre-existing ontologies. In *Proceedings of CoopIS*, pages 14–25, 1996.
24. A.G. Miller. Wordnet: A lexical database for english. In *Communications of the ACM*, number 38(11), pages 39–41, 1995.
25. N. Noy and M. Klein. Ontology evolution: Not the same as schema evolution. In *Knowledge and Information Systems*, in press, 2002.
26. N. Noy and M. A. Musen. Anchor-prompt: Using non-local context for semantic matching. In *Proceedings of IJCAI workshop on Ontologies and Information Sharing*, pages 63–70, 2001.
27. L. Palopoli, G. Terracina, and D. Ursino. The system dike: Towards the semi-automatic synthesis of cooperative information systems and data warehouses. In *ADBIS-DASFAA, Matfyzpress*, pages 108–117, 2000.
28. E. Rahm and P. Bernstein. A survey of approaches to automatic schema matching. In *Very Large Databases Journal*, number 10(4), pages 334–350, 2001.
29. P. Shvaiko. Iterative schema-based semantic matching. Technical Report DIT-04-020, University of Trento, 2004.
30. M.K. Smith, C. Welty, and D.L. McGuinness. Owl web ontology language guide. Technical report, World Wide Web Consortium (W3C), http://www.w3.org/TR/2004/REC-owl-guide-20040210/, February 10 2004.
31. H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Huebner. Ontology-based integration of information - a survey of existing approaches. In *Proceedings of IJCAI workshop on Ontologies and Information Sharing*, pages 108–117, 2001.
32. L. Xu and D.W. Embley. Using domain ontologies to discover direct and indirect matches for schema elements. In *Proceedings of Semantic Integration workshop at ISWC*, 2003.

Toward a Flexible Human-Agent Collaboration Framework with Mediating Domain Ontologies for the Semantic Web

Yuri A. Tijerino[1], Muhammed Al-Muhammed[2], and David W. Embley[2]

[1] Utah Valley State College
Computing and Networking Sciences Department
MS129 800 W. University Parkway, Orem UT 84058
tijeriyu@uvsc.edu, http://www.uvsc.edu/cns/
[2] Brigham Young University, Computer Science Department
TMCB 2228, Provo UT 84602
{mja47, embley}@cs.byu.edu, http://www.deg.byu.edu/

Abstract. This paper introduces a human-agent interaction framework that allows human and computational mechanisms to interact in a seamless manner to collaboratively perform problem solving tasks in the Semantic Web. The framework is based on a combination of services supported by intermediate domain ontologies and flexible mapping mechanisms that map agent and human internal representations and external communication protocols. The framework is based on a solid conceptual foundation and can be extended to also incorporate other service-based computational mechanisms such as web services. The central focus of the framework is to encourage contribution by anyone or anything that might possess useful information, knowledge or expertise for the successful completion of collaborative problem solving tasks. Yet, the problem solving participants are not required to go through a rigorous ontological commitment or have a common communication protocol in order to collaborate.

Key words: Human-Agent Interaction, Semantic Interoperability, Ontology Mapping

1 Introduction

Some researchers have recognized the need for interaction between Semantic Web agents and end-users and are proposing novel and unique approaches to allow human users to interact with semantically annotated web resources. For example, the CS AKTiveSpace system [1] allows users to "explore" information about computer science research in the UK. The system is very sophisticated, yet very elegant in the way it visually presents information to users by allowing them to navigate pages annotated with about 10 million RDF triplets.

Another area of potential collaboration between humans and computational mechanisms, is in the creation of the Semantic Web itself. A good example of this

is an approach proposed by the Mangrove system [2]. This system deals directly with the problem of populating the Semantic Web with useful ontology-based annotations by non-technical people. The approach is interesting because not only does it facilitate annotation of HTML pages by "ordinary" people, but it also allows the people to immediately benefit from the annotations. However, the system doesn't go far enough to promote a post-annotation, two-way interaction mechanism in which both humans and computational mechanisms such as agents collaborate in problem solving efforts. This paper proposes a framework that addresses this particular issue by creating intermediate ontologies on the fly that bridge the gap between human and agent internal representations.

The remainder of this paper is organized as follows. The next section describes the conceptual and philosophical motivation for making possible knowledge sharing between humans and computational mechanism. Section 3 describes the proposed framework for allowing human-agent interoperability in the Semantic Web. Section 4 illustrates the most important parts of the framework through a simple example. Section 5 presents some concluding remarks.

2 Knowledge Sharing Between Humans and Machines

Knowledge does not exist in an isolated form. It exists in an Economy of Knowledge[3]. In early work on so-called expert systems [4], it was noted that knowledge was not something that a particular individual had in a closed system, but instead existed in a distributed manner among many individuals. The task of a so-called "expert" was not only to retrieve pre-stored internal representations of knowledge, but also to collaborate with other experts to create that knowledge through practice. This paper expands this view to include other computational mechanisms such as software agents and web services[4] in the list of experts. In addition, it proposes that by expanding this view, not only can people and machines collaborate in the context of explicit knowledge, but also to some degree in the context of tacit knowledge.

According to Paul Romer [3] knowledge has become the third factor of production in leading world economies, with labor and capital being the first two[5]. If knowledge is a commodity, then it really doesn't matter whether the producer or consumer is a person or a machine. It could be refined, stored, retrieved, shipped or even recycled by either a person or a machine. As in the industrial revolution when machines played an important role in changing the balance of power of the World's economies, machines once more can play an important role as they provide an efficient way to process knowledge for the benefit of human kind. In this analogy, the Web currently represents the pre-industrial revolution

[3] According to Paul Romer [3] knowledge has become the third factor of production in leading world economies, with labor and capital being the first two.

[4] Although the readily could support web services because the framework is built on web service technology, we will focus on human and agent interoperability.

[5] For the past two hundred years labor and capital were recognized by neo-classical economists as the only two factors of production

phase when people were the central producers and consumers of knowledge with all the limitations that that entails. The Semantic Web, however, promises to bring us full throttle into the "Knowledge Revolution", or the equivalent of the Industrial Revolution, when machines make the production of knowledge as a commodity more efficient for people, institutions and governments.

So why is this all so important? Because the ultimate goal of the Semantic Web is to allow machines to find, share, combine and understand knowledge in the Web way, i.e. without central authority, with few basic rules, in a scalable, adaptable, extensible manner capable of supporting both explicit and tacit knowledge [5, 6].

Nevertheless, to truly take advantage of tacit knowledge available to humans and the information in the Web, in combination with the explicit knowledge in the Semantic Web, available to machines in the form of ontologically annotated resources, we must devise mechanisms that allow humans to collaborate with computational mechanisms in a symbiotic partnership.

3 Human-Agent Interoperability Framework

It is obvious from the discussion that there needs to be a transparent framework that allows interaction between agents and agents and between humans and agents. In order for the framework to be transparent we impose two constraints: The framework should not require a strong ontological commitment nor should it require a strong pre-agreement between agent and agents or agents and humans on the communication format.

By "no strong ontological commitment" we mean that the agents and humans do not need to agree on a shared ontology and could have their own internal representation (e.g., a local ontology), which does not need to be the same shared representation. However, they need to be able to readily map their internal representations to a mediating ontology. What makes this a weak ontological commitment, as opposed to a strong one, is that the agents or humans do not need to know *a priori* what the mediating representation (e.g., the global ontology) is.

By "no pre-agreement on communication format" we mean that both the agents and the humans do not need to compose messages in a way that is not natural to the way they do things already. However, in order for this to be possible there needs to be a mechanism that translates the messages between agents and agents and humans an agents in a transparent manner by using the mediating ontology during an initialization phase to understand their message format.

Fulfilling these two requirements would allow for agent-agent and human-agent interoperability on the fly. Here we not only attempt to allow agent-agent interoperability in that manner, but also human-agent interoperability, which is also necessary to bring about the Berners-Lee[7] original vision of the Semantic Web. We do not claim that the framework presented in this paper meets these two requirements in full, but that it does it well within its limitations.

One limitation of this framework is that it only allows for interoperability of humans with computational mechanisms. In other words, the computational mechanisms such as agents are the target of the interaction as opposed to direct interaction of humans with the semantically annotated pages on the Semantic Web, which is the approach of Shadbolt et al [1] and McDowell et al [2] in the systems we described earlier. Interaction between humans and Semantic Web based computational mechanisms, however, is very important and more natural than direct interaction of humans with semantically annotated pages. This is because the Semantic Web is originally intended for computational mechanisms and not for humans. On the other hand, if humans can interact more readily with those computational mechanisms, it should be possible for them to performs tasks such as mining the web for tacit or explicit knowledge in the form of specific answers to questions or solutions to real-world problems as opposed to finding information that can be used to obtain that knowledge as is the case of today's Web. For instance, instead of finding multiple semantically annotated web sites where one can book a flight, buy a book or configure and buy a computer, through a "Semantic Web Explorer" such as the CS AKTiveSpace system, it should be possible instead for users to interact with agents that can find those sites and then perform those transactions directly on behalf of and through interaction with the human user.

The framework described in this section meets these challenges head on and consists of four main components: 1) a directory service that allows seamless registration and search of human and agents, 2) a message mapping infrastructure that translates between internal representations of both agents and humans, 3) a communication service that handles agent-agent and agent-human communication, and 4) a trust and security infrastructure in which the agent-agent and user-agent interactions takes place.

3.1 Directory Services

The framework adapts the matchmaking algorithm based on OWL-S[6] with UDDI as proposed by Paolucci et al [9], to allow registration and search of services provided not only by web services, but also to include services provided by agents and human users. However, since this architecture is fairly well-understood on the context of web services, we will focus here on how to discover, register and search services provided by human users and agents. For instance, an agent whose task is to shop for products on the Internet can be registered in the directory services and advertise that it is a shopping agent. A user who interacts with the agent to buy a product, say a PC, is registered temporarily in the directory as one who can disambiguate the request should the agent find them ambiguous.

To allow agents and human users to interoperate on the fly with other agents or human users, they first need to be registered in the UDDI directory. The process is described as follows.

[6] OWL-S evolved from DAML-S and is now the basis for SWSL, or the Semantic Web Services Language[8].

Agent Registration: Al-Muhammed [10] presents an agent interoperation method that does not require a shared ontology or pre-agreed message format to allow communication between agents. This approach consists of defining local ontologies for the agent based on an extended data extraction methodology that constructs the local ontology from clues provided in the agent code and the code comments. This methodology has worked fairly well in previous efforts to extract data from structured information sources [11, 12] and to dynamically construct ontologies from tables [13]. After the local ontology is constructed for each agent, the task of making agents interoperable is then simplified to translating the local ontologies to a domain-specific global ontology.

Conceptually, the approach of building local ontologies from agent code works because agent code can also be considered to be a structured information source and is treated in a similar way to other structured information sources such as tables or lists. This approach has its limitations since it requires access to the agent's code, which is not always available. It also assumes that the code is well formed, documented and written with meaningful names for class names, methods and variables, which is not always the case either. Nevertheless, experiments showed that, if the assumptions hold true, this approach is viable. We use this approach to first extract the agent's ontology. Then we use the agent's ontology to discover the services and map them to OWL-S. Once the agent's ontology and the agent's services have been discovered they are registered in the UDDI directory for other service requesters to see, including human users. This approach is an extension of that developed by Al-Muhammed [10], but in his approach no central service directory is proposed.

Rather than requiring agents to share ontologies, we provide our framework with automated mapping to agent-independent, domain-specific ontologies such as those readily available in the web in the form of DAML, DAML+OIL or OWL. To accomplish this, we utilize the same mapping techniques described by Al-Muhammed [10] to map the agent's services to OWL-S and make them available through the UDDI directory in a similar way as that described by Paolucci [9] for web services. When an agent makes a request in its native communication language, for instance a Java method call, it is translated using the particular domain ontology, say one written in DAML, and then a service matching the request is found using OWL-S or SWSL service descriptions in the UDDI directory. If the service is that of another agent, then the request is translated to its local ontology and a response is sent back to the requesting agent following the same translation process.

In order to generate local ontologies for the agents, an *Agent Ontology Extraction Engine* uses pre-configured recognizers modeled after data frames [14], which are snippets of knowledge as to how to recognize instances of specific concepts in an ontology. This process is illustrated in Figure 1. This agent ontology includes the names of concepts the agent uses such as class name, parameter names, variable names and the data types of the concepts. To compensate for not having a shared ontology, the framework maps the the agent ontologies of all

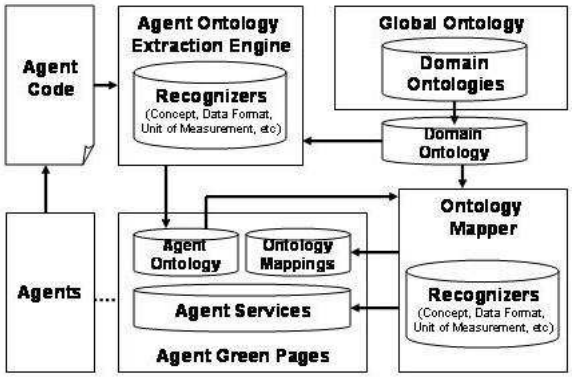


Fig. 1. Agent registration using an intermediate domain ontology and general-purpose concept recognizers.

registered agents to one of the various domain ontologies the framework maintains[7]. This process is described further in Section 3.2.

User Registration: User registration can occur in two different ways. A user might choose to enter information in the UDDI directory and following the OWL-S format to also advertise the basic services provided. Of course this might be a bit unrealistic for most users including experienced ones. A second way is to temporarily register user when she sends a service request that might require interaction with the service provider. We will go in more detail on this process in a later example.

3.2 Ontology Mapping Services

The domain ontologies we propose in our framework consist of two components: a conceptual model of the domain, which describes the domain in terms of concepts, relationships, constraints and axioms, and recognizers[8] that currently are based on regular expressions[9] to help us recognize concepts, data formats and units of measure from the agent's code. We have experimented with this type of recognizers using various domains with very promising results [12, 15, 16].

[7] We emphasize that there is a major difference between our approach and a shared ontology approach, because an agent's developer needs to know nothing about any other agent's ontology, nor do they need to know anything about the domain ontology. It is the ontology mapper that does the work.

[8] Recognizers are currently modeled after Data Frames [14] which original purpose is to allow the extraction of concept instances found in structured or unstructured content.

[9] Although data frames are currently based on regular expressions there is ongoing research to make them smarter by incorporating other recognition techniques such as decision trees, neural networks and even latent semantic indexing.

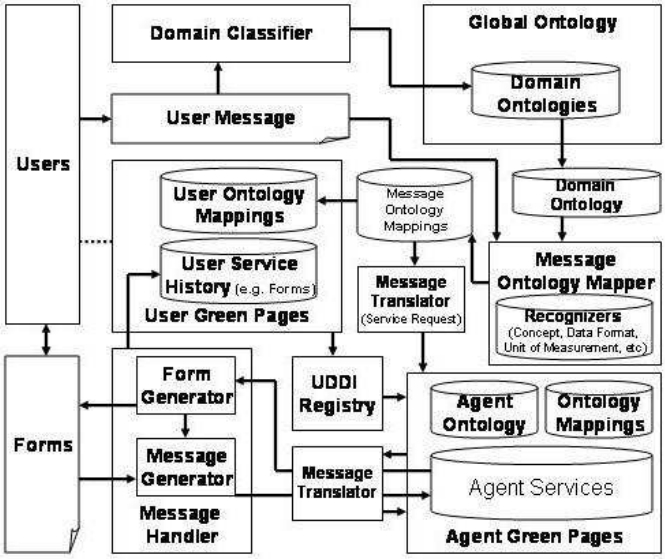


Fig. 2. Message registration and user-agent interaction.

Domain ontologies can either be manually constructed based on ontology engineering techniques [17, 18] or on automatic ontology generation techniques being developed by the authors [13, 19].

Agent Ontology Mapping: As Figure 1 illustrates, the *Agent Ontology Extraction Engine* parses the agent code, finds its services, and expresses them in an agent-independent way, as proposed by Al-Muhammed [10], in the green page corresponding to the agent.

The *Ontology Mapper* then uses the *domain ontologies* to translate internal representations of both agents and users into local representations. It uses recognizers to handle the mapping between agent internal representations and human generated messages. In this framework it is not required that all agents and messages be described in terms of the domain ontology, but instead depend on the recognizers to conciliate the differences between their representations by recognizing concept equivalences and instances. Concept equivalences deal with equivalent concepts in different ontologies, in this case the domain-specific ontology and the agent-specific ontology. Concept instances are related to information found in messages that can be thought of as instances of a particular concept in the domain specific ontology.

User Message Ontology Mapping: When a user sends a message and the framework has identified a domain for the message, an appropriate domain ontol-

ogy from the *Global Ontology Repository* is used to allow the *Message Ontology Mapper* to employ recognizers associated with the particular domain ontology to establish message ontology mappings. This results in several mappings of the message to the ontology to occur and be placed in the *Message Ontology Mappings*. Thus, both the user messages and the agent services are described in terms of the intermediate domain ontology.

Once the message is parsed and the mappings are placed in the *Message Ontology Mappings* repository, two things need to happen: 1) the *Message Ontology Mappings* are placed in the *User Ontology Mappings* repository in the corresponding *User Green pages*, and 2) the *Message Ontology Mappings* are transferred to the *Service Request Generator* which generates a service request for the UDDI registry as shown in Figure 2.

3.3 Communication Services

Rather than having agents deal directly with incoming messages, our framework provides for automatic mapping of incoming messages, either agent or user generated, to an appropriate service in the UDDI registry. Then, a *Message Translator* makes a mapping between the service and incoming messages by 1) parsing a message and identifying its type and its input and output parameters, and 2) matching the type of the input and output parameters of the message with those in a service provided by an agent or user.

User-agent communication is not much unlike agent-agent communication. The main difference is that there is one more formating step necessary so that human users can generate and reply to agent requests. Figure 2 illustrates this kind of user-agent interaction.

If a user sends a message, it is first parsed and mapped to the appropriate domain ontology and stored in the *User Ontology Mappings* as described in Section 3.2. Then the message is translated and an appropriate agent which can handle the request, is identified through the UDDI register. The message is then sent to the agent, which processes the message with the appropriate service. If the agent needs additional information from the user, the agent then composes a message and sends it to the message translator, which translates it and then converts it into a human-readable form using the *Message Handler*'s *Form Generator*[10]. A form is then presented to the user, who takes the necessary action and sends it back to the *Message Handler*. The *Message Handler* then converts the form again into a message, which is translated by the *Message Translator* and sent back to the original agent which made the request.

[10] At the BYU Data Extraction Group [20] with which the authors are affiliated, we have experimented with ontology-driven forms with much success [12, 21, 22]. Here we plan to use the results of that research to enable the interaction of humans with the agents through dynamically generated forms.

3.4 Trust and Security Services

There are many trust and security issues that come to mind in regards to agent-agent and human-agent interactions. These issues deal with network security, message content, authentication and authorization. These issues are very common in any distributed system, however, what makes them unique in this framework of agent and human interactions is that there is no predetermined communication between a server and a client because this framework proposes a highly distributed architecture with no pre-arranged relationships between service providers and requesters. What is needed, therefore, is a trust and security infrastructure that can be customized to this type of distributed architecture.

There are at least three systems that come to mind that are originally intended for supporting distributed agent interactions in the Semantic Web. The first and most favored by the authors is the one proposed by Gavriloaie et al [23], because it does not require registration of the agents or the users in order to interact and build trust. In addition, it is based on declarative policies which can be easily maintained as an additional service in the UDDI registry.

The second approach is proposed by Kagal et al [24] and deals with annotating distributed Semantic Web sources with policies. Although, this could work with Semantic Web pages and even agents, it would be difficult to maintain for human users. Yet, the third approach is based on Semantic Web languages for policy representation and reasoning as described by Tonti et al [25]. Although these policy based approaches could in theory work with the UDDI registry, they are not flexible enough to allow users and agents to develop the policies on the fly as is the case of the approach proposed by Gavriloaie et al [23].

Although, we identify trust and security services as service necessary for our framework, we have yet to do much work in this area. We leave these issues open for future work as there still remains much to be studied and done to understand this area further.

4 Experimental Results

In this section we illustrate the four services described in sections 3.1, 3.2 and 3.3, but not 3.4, since it not in the main scope of this paper.

We have experimented with translation of agent requests and agent-to-agent service matching. We have also performed experiments in which we have provided simple interfaces for human users to interact with agents in collaborative tasks. Because of space limitation, here we describe only one of these experiments, a computer shopping application, and the results briefly.

First, we constructed a global ontology manually[11] from 8 Web sites (*amazon.com, cdw.com, dell.com, half.ebay.com, gateway.com, plasmakings,com, price.com and ubid.com*). To construct this mediating ontology, we collected concepts for

[11] Eventually, we will use results from our semi-automated ontology generation approach reported in [13]

each part/attribute of a computer, the units of measurement, and the data formats. For each concept, we then created a recognizer for the concept using data frames. In addition, we also created recognizers for each unit of measurement found in each of the 8 sites.

Based on the forms found in each of the Web sites, we manually identified interesting services generally common in the Web sites through input forms. We then, for each site, implemented those services in a seller agent (one per site) generating service signatures by determining service names, input parameters and return types based on information extracted from the forms in the site. For instance, to determine data type for a particular form, we chose two possible types: *double* if the allowable input contains a decimal point and *String* otherwise. Service names were assigned by choosing the form label as the name for the service for each of the Web sites. We then registered the services in the UDDI directory. We also implemented a buyer agent to interact with the seller agent. In turn, the user can either interact directly with the buyer, or the service agent by interacting with forms associated with each agent to request or provide information.

The bulk of our current experiments include communication between the buyer agent and the seller agent. The buyer agent communicates with the seller by making requests indirectly through the matchmaking system reported in [10]. To evaluate the performance in matching requests[12], we chose 9 test sites (different from the 8 sites we used to build the global ontology), one for a buyer agent and the rest for the seller agents. We measured the system performance in mapping concepts, units of measurement, and data formats used by agents to the global ontology. The agents' code included 104 concepts, which the system needed to map to the global concepts. The system was successful in generating 94 mapping pairs of the form (*Local*, *Global*), of which 91 were correct, yielding (91/104) or 88% recall and (91/94) or 97% precision. The units of measurement, in the agents' code, that the system needed to recognize were currencies, processor/hard-drive speed units, and memory/hard-disk capacity units. The currency types in the 9 test sites were US$, GBP (Great Britain Pound), and EUR (Euro). There were 9 currency instances that the system needed to recognize. The system recognized 9, all of which were correct. The number of processor/hard-drive speed units and memory/hard-disk capacity units was 23. The system recognized 25, of which 23 were correctly associated with their global counterparts. Altogether there were 32 unit instances; the system recognized 34, of which 32 were correct, yielding 100% recall and 94% precision.

These results demonstrate that it is possible to create agents with local ontologies that can communicate with each other by translating service requests via a mediating (global) ontology. It also demonstrates that users can interact with agents to perform simple tasks collaborative such as the one described in this experiment. Although, we have yet to perform more sophisticated experiments in cross-domain tasks such as the one described by Berners-Lee [26], this

[12] Although we used only agent generated requests in the experiment, we found that user generated requests are not much different

experiment demonstrates that our approach can help exploit the distributed and ad-hoc nature of the Semantic Web by allowing ontologists to develop decentralized Semantic Web ontologies, developers to develop domain-independent service agents and users to communicate and guide agents to collaboratively accomplish complex tasks.

This experiment also highlights some of the limitations of our approach. Mainly, our approach works well if the agents expose code that can be mapped to an ontology via our concept-to-instance mapping mechanism. In our experiment, we generated the code based on the Web site forms, thus the code was available to us. However, it might be unreasonable to expect all developers to make their code available for scrutinity. Even if the source code was available, it is not guaranteed that the mapping from its local ontology to the global ontology might occur smoothly, especially if the agent is automatically generated or poorly coded.

5 Concluding Remarks

We have described a framework for agent-agent and human-agent interaction in the Semantic Web. The most interesting and unique contribution of the paper is that no-matter what or who the problem solving participants are, they are not required to go through a rigorous ontological commitment or have a common communication protocol in order to collaborate. We perceive this to be the main problem that overshadows the successful proliferation of the Semantic Web and have tackled it head on. Although, there is still much work to be done to expand the vision of this framework beyond its current form, we are confident that it will make it possible for "ordinary", non-technical people to reap the full benefits of the Semantic Web to come.

References

1. Shadbolt, N., Schraefel, M., Gibbins, N., Glaser, H., Harris, S.: Cs aktivespace: Representing computer science in the semantic web. In: Proceedings of the 13th International World Wide Web Conference (WWW2004), New York, NY, ACM (2004)
2. McDowell, L., Etzioni, O., Gribble, S., Halevy, A., Levy, H., Pentney, W., Verma, D., Vlasseva, S.: Mangrove: Enticing ordinary people onto the semantic web via instant gratification. In: Proceedings of the 2nd International Semantic Web Conference (ISWC03), Sanibel Island, Florida, USA (2003)
3. Romer, P.M.: Endogenous technological change. Journal of Political Economy (1990) S71–S102
4. Tijerino, Y.A.: A Task Analysis Interview System Based on Two-Level Task Ontologies. PhD thesis, Osaka University, Osaka Japan (1993)
5. Polanyi, M.: The Tacit Dimension. Routledge & Kegan Paul Ltd, London (1966)
6. Nonaka, I., Takeuchi, H.: The Knowledge-Creating Company. Oxford University Press, Oxford (1995)
7. Berners-Lee, T.: Information management: A proposal (the original www proposal) (1989) http://www.w3.org/History/1989/proposal.html.

8. : Semantic web services language (2004) http://www.daml.org/services/swsl/.
9. Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: Semantic matching of web services capabilities. In: Proceedings of the First International Semantic Web Conference (ISWC02). (2002) 333–347
10. Al-Muhammed, M.: Dynamic matchmaking between messages and services in multi-agent systems. Technical report, Brigham Young University, (Provo, Utah)
11. Embley, D., Tao, C., Liddle, S.: Automating the extraction of data from tables with unknown structure. Data & Knowledge Engineering (2004) (to appear).
12. Chen, X.: Query rewriting for extracting data behind html forms. Technical report, Brigham Young University, Provo, Utah (2004) Currently at www.deg.byu.edu/proposals/index.html).
13. Tijerino, Y., Embley, D., Lonsdale, D., Nagy, G.: Ontology generation from tables. In: Proceedings of the 4th International Conference on Web Information Systems Engineering, Rome, Italy (2003) 242–249.
14. Embley, D.: Programming with data frames for everyday data items. In: Proceedings of the 1980 National Computer Conference, Anaheim, California (1980) 301–305
15. Embley, D., Tao, C., Liddle, S.: Automatically extracting ontologically specified data from HTML tables with unknown structure. In: Proceedings of the 21st International Conference on Conceptual Modeling (ER'02), Tampere, Finland (2002) 322–327
16. Chartrand, T.: Ontology-based extraction of RDF data from the world wide web. Master's thesis, Brigham Young University, Provo, Utah (2003)
17. Mizoguchi, R., Tijerino, Y.A., Ikeda, M.: Task analysis interview based on task ontology. Expert Systems with Applications **9** (1995) 15–25
18. Mizoguchi, R., Ikeda, M.: Towards ontology engineering. In: proceedings of the Joint 1997 Pacific Asian Conference on Expert Systems / Singapore International Conference on Intelligent Systems, Singapore (1997) 259–266
19. Tijerino, Y., Embley, D., Lonsdale, D., Nagy, G.: Ontology generation from tables. (Journal of World Wide Web Internet and Web Information Systems) Submitted.
20. : Homepage for BYU data extraction research group (2004) URL: http://osm7.cs.byu.edu/deg/index.html.
21. Liddle, S., D.W. Embley, D.S., Yau, S.: Extracting data behind web forms. In: Proceedings of the Joint Workshop on Conceptual Modeling Approaches for E-business: A Web Service Perspective (eCOMO 2002), Tampere, Finland (2002) 38–49
22. Yau, S.: Automating the extraction of data behind web forms. Technical report, Brigham Young University, Provo, Utah (2001) http://www.deg.byu.edu.
23. Gavriloaie, R., Nejdl, W., Olmedilla, D., Seamons, K.E., Winslett, M.: No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In: 1st European Semantic Web Symposium, Heraklion, Greece (2004)
24. Kagal, L., Finin, T., Joshi, A.: A policy based approach to security for the semantic web. In: Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida, USA (2003)
25. Tonti, G., Bradshaw, J., Jeffers, R., Montanari, R., Suri, N., Uszok, A.: Semantic web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In: Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida, USA (2003) 419–437
26. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American **36** (2001)