A SAT-Based Algorithm for Context Matching

Paolo Bouquet[1,2], Bernardo Magnini[2], Luciano Serafini[2], and Stefano Zanobini[1]

[1] Department of Information and Communication Technologies – University of Trento
Via Inama, 5 – 38100 Trento (Italy) {bouquet,zanobini}@dit.unitn.it
[2] ITC-IRST – Istituto per la Ricerca Scientifica e Tecnologica
Via Sommarive, 14 – 38050 Trento (Italy)
{magnini,serafini}@itc.it

Abstract. The development of more and more complex distributed applications over large networks of computers has raised the problem of *semantic interoperability* across applications based on local and autonomous semantic schemas (e.g., concept hierarchies, taxonomies, ontologies). In this paper we propose to view each semantic schema as a context (in the sense defined in [1]), and propose an algorithm for automatically discovering relations across contexts (where relations are defined in the sense of [7]). The main feature of the algorithm is that the problem of finding relationships between contexts is encoded as a problem of logical satisfiability, and so the discovered mappings have a well–defined semantic. The algorithm we describe has been implemented as part of a peer-to-peer system for Distributed Knowledge Management, and tested on significant cases.

1 Introduction

The development of more and more complex distributed applications over large networks of computers has created a whole new class of conceptual, technical, and organizational problems. Among them, one of the most challenging one is the problem of *semantic interoperability*, namely the problem of allowing the exchange of meaningful information/knowledge across applications which (i) use autonomously developed conceptualizations of their domain, and (ii) need to collaborate to achieve their users' goals.

Essentially, there are two main approaches for solving the problem of semantic interoperability. The first is based on the availability of shared semantic structures (e.g., ontologies, global schemas) onto which local representations can be totally or partially mapped. The second is based on the creation of a global representation which integrates local representations. Both approaches do not seem suitable in scenarios where: (i) local representations are updated and changed very frequently, (ii) each local representation is managed in full autonomy w.r.t. the other ones, (iii) local representations may appear and disappear at any time, (iv) the discovery of semantic relation across different representations can be driven by a user's query, and thus cannot be computed beforehand (runtime discovery) nor take advantage of human intervention (automatic discovery).

P. Blackburn et al. (Eds.): CONTEXT 2003, LNAI 2680, pp. 66–79, 2003.

In this paper we propose an approach in which local schemas are viewed as contexts, namely as partial and approximate representations of the world from an individual's or a group's perspective [1] (two simple examples of schemas are the two directory structures from Google and Yahoo in Figure 1). This approach, which is motivated by the work on Distributed Knowledge Management (DKM) [4,3], is based on the assumption that a successful knowledge–based application should not "force" people to change their way of looking at things (encoded, for example, in a database schema or in the classification of a document management system), as the imposed schema would be perceived "either as oppressive or irrelevant" [13]. Thus, from our perspective, local schemas play the role of a lens through which people look at the world and make sense of it. In a word, a schema is the context in which facts are taken as true, decisions are made, objects are classified, relations among things are asserted and understood.

The problem of such a vision is that communication across different local schemas (contexts) becomes difficult. The algorithm we present in this paper is precisely a first solution to the problem of runtime and automatic discovery of semantic relations across autonomous contexts. More specifically, we start from a broad family of schemas (called concept hierarchies), and present a method for discovering the type of relation existing between two nodes (each representing a concept) belonging to different schemas. The main feature of the algorithm is that the problem of finding relations between concepts in different contexts is encoded as a problem of logical satisfiability of a set of formulae. This allows us to assign a precise semantic to each discovered mapping. In particular, we claim that the correct semantic for a mapping between concepts of different contexts is in terms of a compatibility relation (as defined in [7]), namely as a constraint on the local interpretations of the two contexts that are compatible with each others. In this sense, the algorithm we present is a first attempt to discover (rather than assume) relations over local models of two or more contexts (which, from a proof–theoretical point of view, corresponds to discover "bridge rules" [8] across contexts).

The paper goes as follows. First, we characterize the scenarios that motivate our approach, and explain why we use the theory of context as a theoretical background of the algorithm. Then, we describe the macro–blocks of the algorithm, namely semantic explicitation and context mapping via SAT. Finally, we describe the results of our preliminary tests and briefly compare our algorithm with some other proposals in the literature.

2 Motivating Scenarios

The work on the algorithm was originally motivated by a research on Distributed Knowledge Management [4], namely a distributed approach to managing corporate knowledge in which users (or groups of users, e.g. communities) are allowed to organize their knowledge using autonomously developed schemas (e.g., directories, taxonomies, corporate ontologies), and are then supported in finding relevant knowledge in other local schemas available in the corporate network.

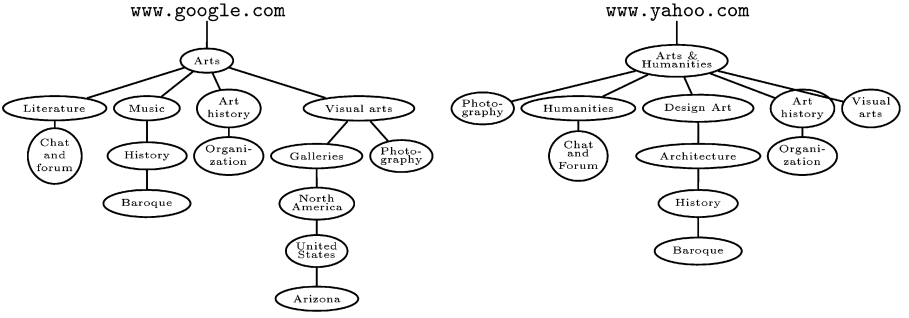

Fig. 1. Examples of concept hierarchies (source: Google and Yahoo)

In this scenario, the algorithm we present aims at solving the following problem. Let s (the *source schema*) and t (the *target schema*) be two autonomous schemas that different users (or groups) use to organize and access a local body of data. Given a concept k_s in s, and a concept k_t in t, what is the semantic relations between k_s and k_t? For example, are the two concepts equivalent? Or one is more (less) general than the other one? In addressing this problem, it is assumed that the basic elements of each schema are described using words and phrases from natural language (e.g., English, Italian); this reflects the intuition that schemas encode a lot of implicit knowledge, which can be made explicit only if one has access to the meaning of the words that people use to denote concepts in the schema.

Scenarios with similar features can be found in other important application domains, such as the semantic web (where each site can have a semantic description of its contents and services), marketplaces (where every participating company may have a different catalog, and every marketplace may adopt a different standard for cataloging products); search engines (some of them , e.g. Google and Yahoo, provide heterogeneous classifications of web pages in web directories); the file system on the PCs of different users (where each user stores documents in different directory structures). So the class of applications in which our algorithm can be applied is quite broad.

3 Local Schemas as Contexts

In many interesting applications, schemas are directed graphs, whose nodes and edges are labeled with terms or phrases from natural language. A typical example is depicted in Figure 1, whose structures are taken from the Google and Yahoo directories. In this section, we briefly argue why we interpret these schemas as contexts in the sense of [1] (see [7] for a formalization).

In schemas like the ones in the figure, the meaning of a label depends not only on its linguistic meaning (what a dictionary or thesaurus would say about

that word or phrase), but also on the context in which it occurs: first, it depends on the position in the schema (e.g., the documents we as humans expect to find under the concept labeled *Baroque* in the two structures in Figure 1 are quite different, even if the label is the same, and is used in the same linguistic sense); second, it depends on background knowledge about the schema itself (e.g., that there are chat and forums about literature helps in understanding the implicit relation between these two concepts in the left hand side schema). These contextual aspects of meaning are distinct (though related) to purely linguistic meaning, and we want to take them into account in our algorithm.

For this purpose, the algorithm we present in this paper is applied to contexts rather than to schemas directly. In [1], a context is viewed as a box, whose content is an explicit (partial, approximate) representation of some domain, and whose boundaries are defined by a collection of assumptions which hold about the explicit representation. The notion of context we use in this paper is an special case of the notion above. A context is defined as a pair $c = \langle R_c, A_c \rangle$, where:

1. R_c is a graph, whose nodes and edges can be labeled with expressions from natural language;
2. A_c is a collection of explicit assumptions, namely attributes (parameter/value pairs) that provide meta-information about the content of the context.

In the current version of the algorithm, we restrict ourselves to the case in which R_c is a concept hierarchy (see Def. 1), and the explicit assumptions A_c are only three: the *id* of the natural language in which labels are expressed (e.g., English, Italian), the reference structure R_c of the explicit representation (the only accepted value, at the moment, is "concept hierarchy", but in general other values will be allowed, e.g., taxonomy, ontology, semantic network, frame), and the domain theory (see below for an explanation of this parameter). Their role will become apparent in the description of the algorithm.

A concept hierarchy is defined as follows:

Definition 1 (Concept hierarchy). *A* concept hierarchy *is a triple* $H = \langle K, E, l \rangle$ *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* $K \cup E$ *to a set* L *of strings.*

Definition 2 (Hierarchical classification). *A* hierarchical classification *of a set of documents* D *in a concept hierarchy* $H = \langle K, E, l \rangle$ *is a function* $\mu : K \rightarrow 2^D$.

μ satisfies the following *specificity principle*: a user classifies a document d under a concept k, if d is about k (according to the user) and there isn't a more specific concept k' under which d could be classified[1].

[1] See Yahoo instruction for "Finding an appropriate Category" at http://docs.yahoo.com/info/suggest/appropriate.html.

Mappings between contexts are defined as follows:

Definition 3 (Mapping function). *A* mapping function *M from $H = \langle K, E, l\rangle$ to $H' = \langle K', E', l'\rangle$ is a function $M : K \times K' \rightarrow rel$, where rel is a set of symbols, called the* possible mappings*.*

The set *rel* of possible mappings we consider in this paper contains the following: $k_s \xrightarrow{\supseteq} k_t$, for k_s is more general than k_t; $k_s \xrightarrow{\subseteq} k_t$ for k_s is less general than k_t; $k_s \xrightarrow{*} k_t$ for k_s is compatible with k_t; $k_s \xrightarrow{\perp} k_t$ for k_s is disjoint from k_t; $k_s \xrightarrow{\equiv} k_t$ for k_s is equivalent to k_t. The formal semantics of these expressions is given in terms of compatibility between document classifications of H_s and H_t:

Definition 4. *A mapping function M from H_s to H_t is* extensionally correct *with respect to two hierarchical classifications μ_s and μ_t of the same set of documents D in H_s and H_t, respectively, if the following conditions hold for any $k_s \in K_s$ and $k_t \in K_t$:*

$$
\begin{aligned}
&k_s \xrightarrow{\supseteq} k_t \Rightarrow \mu_s(k_s{\downarrow}) \supseteq \mu_t(k_t{\downarrow})\\
&k_s \xrightarrow{\subseteq} k_t \Rightarrow \mu_s(k_s{\downarrow}) \subseteq \mu_t(k_t{\downarrow})\\
&k_s \xrightarrow{\perp} k_t \Rightarrow \mu_s(k_s{\downarrow}) \cap \mu_t(k_t{\downarrow}) = \emptyset\\
&k_s \xrightarrow{\equiv} k_t \Rightarrow \mu_s(k_s{\downarrow}) = \mu_t(k_t{\downarrow})\\
&k_s \xrightarrow{*} k_t \Rightarrow \mu_s(k_s{\downarrow}) \cap \mu_t(k_t{\downarrow}) \neq \emptyset
\end{aligned}
$$

where $\mu(c{\downarrow})$ is the union of $\mu(d)$ for any d in the subtree rooted at c.

The semantics introduced in Definition 4 can be viewed as an instance of the compatibility relation between contexts as defined in Local Models Semantics [7, 5]. Indeed, suppose we take a set of documents D as the domain of interpretation of the local models of two contexts c_1 and c_2, and each concept as a unary predicate. If we see the documents associated to a concept as the interpretation of a predicate in a local model, then the relation we discover between concepts of different contexts can be viewed as a compatibility constraint between the local models of the two concepts. For example, if the algorithm returns an equivalence between the concepts k_1 and k_2 in the contexts c_1 and c_2, then it can be interpreted as the following constraint: if a local model of c_1 associates a document d to k_1, then any compatible model of c_2 must associate d to k_2 (and vice versa); analogously for the other relations.

4 The Matching Algorithm

The algorithm has two main phases:

Semantic explicitation. In the schema level, a lot of information is implicit in the labels, and in the structure. The objective of this first phase is to

make it as explicit as possible by associating to each node (and edge) k a logical formula $w(k)$ that encodes this information. Intuitively, $w(k)$ is an approximation of the human interpretation.

Semantic comparison. We encode the problem of finding mappings between two concepts k and k', whose explicit meaning is $w(k)$ and $w(k')$, into a problem of satisfiability, which is then solved by a SAT solver in a logic W (i.e., the logic in which $w(c)$ and $w(c')$ are expressed). Domain knowledge is also encoded as a set of formulas of W.

Since here we are mainly focussed on the second phase, we only provide a short description of semantic explicitation (details can be found in [10]), and then move to the SAT encoding.

4.1 Semantic Explicitation

The goal of the first phase is to make explicit all the semantic information which can be fruitfully used to define the SAT problem in a rich way. The main intuition is that any schema is interpreted (by its users) using two main sources of information: lexical information, which tells us that a word (or a phrase) can have multiple senses, synonyms, and so on; and a background theory, which provides extra-linguistic information about the concepts in the schema, and about their relations. For example, lexical information about the word "Arizona" tells us that it can mean "a state in southwestern United States" or a "glossy snake". The fact that snakes are animals (reptiles), that snakes are poisonous, and so can be very dangerous, and so on, are part of a background theory which one has in mind when using the word "Arizona" to mean a snake[2]. In the version of the algorithm we present here, we use WordNet as a source both of lexical and background information about the labels in the schema. However, we'd like to stress the fact that the algorithm does not depend on the choice of any particular dictionary or theory (i.e., does not depend on WordNet). Moreover, we do not assume that the same dictionary and background theory are used to explicit the semantic of the two contexts to be matched.

Semantic explicitation is made in two main steps: *linguistic interpretation* and *contextualization*.

Linguistic interpretation. Let $H = \langle K, E, l \rangle$ be a concept hierarchy and L_H the set of labels associated to the nodes and edges of a hierarchy H by the function l. In this phase we associate to each label $s \in L_H$ a logical formula representing the interpretation of that label w.r.t. the background theory we use.

[2] We are not saying here that there is only one background theory. On the contrary, theories tend to differ a lot from individual to individual, and this is part of the reason why communication can fail. What we are saying is that, to understand what "Arizona" means in a schema (such as the concept hierarchy in the left hand side of Figure 1), one must have a theory in mind.

Definition 5 (Label interpretation). *Given a logic W, a* label interpretation *in W is a function $I : L_H \rightarrow wff(W)$, where $wff(W)$ is the set of well formed formulas of W.*

The choice of W depends on the external assumptions of the context containing H. For concept hierarchies, we adopted a description logic W with $\sqcup$, $\sqcap$ and $\neg$, whose primitive concepts are the synsets of WordNet that we associate to each label (with a suitable interpretation of conjunctions, disjunctions, multiwords, punctuation, and parenthesis). For example, WordNet provides 2 senses for the label *Arizona* in Figure 1, denoted by `#1` and `#2`; in this case, the output of the linguistic analysis is the following formula in W: `Arizona#1` $\sqcup$ `Arizona#2`

Contextualization. Linguistic analysis of labels is definitely not enough. The phase of contextualization aims at pruning or enriching the synsets associated to a label in the previous phase by using the context in which this label occurs. In particular, we introduce the concept of *focus* of a concept k, namely the smallest subset of H which we need to consider to determine the meaning of k. What is in the focus of a concept depends on the structure of the explicit representation. For concept hierarchies, we use the following definition:

Definition 6 (Focus). *The* focus *of a concept $k \in K$ in a concept hierarchy $H = \langle K, E, l \rangle$, is a finite concept hierarchy $f(k, H) = \langle K', E', l' \rangle$ such that: $K' \subseteq K$ contains k, its ancestors, and their direct descendants; $E' \subseteq E$ is the set of edges between the concepts of K'; l' is the restriction of l on K'.*

The *contextualization* of the interpretation of concept k of a context c is formula $w(k)$, called *contextualized interpretation* of k, which is computed by combining the linguistic interpretations associated to each concept h in the focus of k. The two main operations performed to compute $w(k)$ are sense filtering and sense composition.

Sense filtering uses NL techniques to discard synsets that are not likely to be correct for a label in a given focus. For example, the sense of *Arizona* as a snake can be discarded as it does not bear any explicit relation with the synsets of the other labels in the focus (e.g., with the synsets of *United States*), whereas it bears a part-of relation with `United States#1` (analogously, we can remove synsets of *United States*).

Sense composition enriches the meaning of a concept in a context by combining in linguistic interpretation with structural information and background theory. For concept hierarchies, we adopted the default rule that the contextual meaning of a concept k is formalized as the conjunction of the senses associated to all its ancestors. Furthermore, some interesting exceptions are handled. An example: in the Yahoo Directory, *Visual arts* and *Photography* are sibling nodes under *Arts & Humanities*; since in WordNet photography is in a is–a relationship with visual art, the node *Visual arts* is re-interpreted as visual arts minus photography, and is then formalized in description logic as: `visual art#1` $\sqcup \neg$ `photography#1`

4.2 Computing Relations between Concepts via SAT

In the second phase of the algorithm, the problem of discovering the relationship between a concept k in a context c and a concept k' in a context c' is reduced to the problem of checking, via SAT, a set of logical relations between the formulas $w(k)$ and $w(k')$ associated to k and k'. The SAT problem is built in two steps. First, we select the portion T of the background theory relevant to the contextualized interpretation $w(k)$ and $w(k')$, then we compute the logical relation between $w(k)$ and $w(k')$ which are implied by T.

Definition 7. *Let $\phi = w(k)$ and $\psi = w(k')$ be the contextualized interpretation of two concepts k and k' of two contexts c and c', respectively. Let B be a theory (= logically closed set of axioms) in the logic where ϕ and ψ are expressed. The* portion of B relevant to ϕ and ψ*, is a subset T of B such that T contains all the axioms of B containing some concept occurring in ϕ or ψ.*

Clearly different contexts can be associated to different background theories, which encodes general and domain specific information. This information is stored in the context external assumptions under the field "domain". Furthermore, when we determine the mapping between two contexts c_s and c_t we can take the perspective (i.e., the background theory) of the source or that of the target. The two perspectives indeed might not coincide. This justifies the introduction of directionality in the mapping. I.e. $c_s \xrightarrow{\subseteq} c_t$ means that c_s is more general than c_t according to the target perspective; while the relation $c_t \xrightarrow{\supseteq} c_s$ represent the fact that c_s is more general that c_t according to the source perspective.

In the first version of our matching algorithm we consider a background theory B determined by transforming the WordNet relations in a set of axioms in description logic, as shown in Table 1. In this table we introduce the notation $\equiv_w$, $\leq_w$, $\geq_w$, and $\perp_w$ to represent the following relation between senses stored in WordNet.

1. `s#k` $\equiv_w$ `t#h`: `s#k` and `t#h` are synonyms (i.e., they are in the same synset);
2. `s#k` $\leq_w$ `t#h`: `s#k` is either a hyponym or a meronym of `t#h`;
3. `s#k` $\geq_w$ `t#h`: `s#k` is either a hypernym or a holonym of `t#h`;
4. `s#k` $\perp_w$ `t#h`: `s#k` belongs to the set of opposite meanings of `t#h` (if `s#k` and `t#h` are adjectives) or, in case of nouns, that `s#k` and `t#h` are different hyponyms of the same synset.

In the extraction of the theory B from WordNet we adopt a certain heuristic which turns out to perform satisfactory (see section on experimentation and evaluation). However, different sources as, specific domain ontologies, domain taxonomies, etc. and different heuristics can be used to build the theory B, from which T is extracted.

Going back to how we build the theory B, suppose, for example, that we want to discover the relation between *Chat and Forum* in the Google directory

Table 1. Encoding WordNet relations in T-Box axioms

WordNet relation	Domain axiom
$\texttt{t\#k} =_w \texttt{s\#h}$	$\texttt{t\#k} \equiv \texttt{s\#h}$
$\texttt{t\#k} \leq_w \texttt{s\#h}$	$\texttt{t\#k} \sqsubseteq \texttt{s\#h}$
$\texttt{t\#k} \geq_w \texttt{s\#h}$	$\texttt{t\#k} \sqsupseteq \texttt{s\#h}$
$\texttt{t\#k} \perp_w \texttt{s\#h}$	$\neg\texttt{t\#k} \sqsubseteq \texttt{s\#h}$

Table 2. Verifying relations as a SAT problem

relation	SAT Problem
$k_s \xrightarrow{\supseteq} k'_t$	$T_t \models w(k_t) \sqsubseteq w(k_s)$
$k_s \xrightarrow{\subseteq} k_t$	$T_t \models w(k_s) \sqsubseteq w(k_t)$
$k_s \xrightarrow{\perp} k_t$	$T_t \models w(k_s) \sqcap w(k_t) \sqsubseteq \perp$
$k_s \xrightarrow{\equiv} k_t$	$T_t \models w(k_t) \sqsubseteq w(k_s)$ and $T_t \models w(k_s) \sqsubseteq w(k_t)$
$k_s \xrightarrow{*} k_t$	$w(k_s) \sqcap w(k_t)$ is consistent in T_t

and *Chat and Forum* in the Yahoo directory in Figure 1. From WordNet we can extract the following relevant axioms:

$$\texttt{art\#1} \sqsubseteq \texttt{humanities\#1}$$

(the sense 1 of 'art' is an hyponym of the sense 1 of 'humanities'), and

$$\texttt{humanities\#1} \sqsupseteq \texttt{literature\#2}$$

(the sense 1 of 'humanities' is an hyperonym of the sense 2 of 'literature').

The axioms extracted from WordNet can now be used to check what mapping (if any) exists between k and k' looking at their contextualized interpretation. But which are the logical relations of $w(k)$ and $w(k')$ that encode a mapping function between k and k' as given in Definition 3? Again, the encoding of the mapping into a logical relation is a matter of heuristics. Here we propose the translation described in Table 2. In this table T_t is the portion of the background theory of c_t relevant to k_s and k_t. The idea under this translation is to see WordNet senses (contained in $w(k)$ and $w(k')$) as sets of documents. For instance the concept `art#i`, corresponding to the first WordNet sense of art, is though as the set of documents speaking about art in the first sense. Using the set theoretic interpretation of mapping given in definition 4, we have that mapping can be translated in terms of subsumption of $w(k)$ and $w(k')$. Indeed subsumption relation semantically corresponds to the subset relation.

So, the problem of checking whether *Chat and Forum* in Google is, say, less general than *Chat and Forum* in Yahoo amounts to a problem of satisfiability on the following formula:

$$\texttt{art\#1} \sqsubseteq \texttt{humanities\#1} \tag{1}$$

$$\texttt{humanities\#1} \sqsupseteq \texttt{literature\#2} \tag{2}$$

$$(\texttt{art\#1} \sqcap \texttt{literature\#2} \sqcap (\texttt{chat\#1} \sqcup \texttt{forum\#1})) \tag{3}$$

$$(\texttt{art\#1} \sqcup \texttt{humanities\#1}) \sqcap \texttt{humanities\#1} \sqcap (\texttt{chat\#1} \sqcup \texttt{forum\#1}) \tag{4}$$

It is easy to see that from the above axioms we can infer (3) $\sqsubseteq$ (4).

To each relation it is possible to associate also a quantitative measure. For instance the relation "c is compatible with d" can be associated with a degree, representing the percentage of models that satisfy $\phi \sqcap \psi$ on the models that satisfy $\phi \sqcup \psi$. Another example is the measure that can be associated to the relation "c is more general than d" which is the percentage of the models of that satisfy ϕ on the models that satisfy ψ. This measure give a first estimation on how much ψ is a generalization of ϕ, the lower percentage, the higher generalization.

5 Testing the Algorithm

In this section we briefly report from [11] the results of the first tests of the algorithm. We observe that the tests are performed on real schemas (i.e., pre-existing schemas that we found in real applications), and not on schemas created *ad hoc*.

5.1 Experiment 1: Generating Google's Links

The first test uses the Google web directory. It can be viewed as a concept hierarchy in which some paths in the hierarchical structure are linked to other paths (links are marked by the @-sign in the Google web page), a mechanism that allows "jumping" from a path to another in the hierarchy (a sort of symbolic link in a Unix file system). Our hypothesis is that these links can be viewed as human-defined relations between concepts, and thus can be used to validate the results of running our algorithm between concepts of the Google directory as if they were concepts of different contexts.

Since the Google directory is very large, the test was performed on the *News* sub–hierarchy, as it is relatively small and well covered by WordNet. The result of computing 1740^2 (about 3,000,000) mappings are summarized and compared with Google's mappings in the following table:

Description	**exact links found**	**%**	**wrong links found**
Equivalence	7	5%	4
More + less general	3+81	56%	688
Total	91	61%	692

The table must be read as follows: the links provided by Google are on the whole 151, and the 61% has been found by the algorithm, while 60 links (39%) have been not found. Regarding the wrong links found, we can say that in the four cases the algorithm found an equivalence between concepts that were not linked in Google; we manually checked these cases, and concluded that the results of the algorithm were extremely plausible, and that the two concepts could be correctly linked in Google. For example, the algorithm found that the concept *News/Media/Media Producers/Television* is equivalent to *News/Media/Media Producers/Video*, based on the fact that one of the senses of television in WordNet has video among its synonyms. The algorithm was not very accurate for the other two relations (precision = 11%), even though a manual verification of the "false positives" led us to conclude that in most cases they could be valuable suggestions for new Google links.

5.2 Experiment 2: Matching Google with Yahoo!

The aim of this experiment was to evaluate the CtxMatch algorithm over pairs of overlapping structures from Google and Yahoo!. The test was performed on two pairs, those with root 'Architecture' and 'Medicine'. The results, expressed in terms of precision and recall, are reported in the following table:

	Architecture	**Medicine**
Relations	Pre. Rec.	Pre. Rec.
equivalence $\xrightarrow{\equiv}$	.71 .10	.78 .13
less general than $\xrightarrow{\subseteq}$	.85 .49	.88 .46
more general than $\xrightarrow{\supseteq}$	.51 .91	.60 .78

We observe that a content–based interpretation of contextual knowledge allows the discovery of non trivial mappings. For example, an inclusion mapping was computed between *Architecture/History/Periods_and_Styles/Gothic/Gargoyles* and *Architecture/History/ Medieval* as a consequence of the relation between *Medieval* and *Gothic* that can be found in WordNet.

5.3 Experiment 3: Product Re-classification

The third test was in the domain of e–commerce. In the framework of a collaboration with a worldwide telecommunication company, the matching algorithm was applied to re-classify the catalog of the office equipment and accessories (used to classify company suppliers) into UNSPSC[3] (version 5.0.2). The validity of the relations found by the algorithm, shown in the following table, were double-checked manually.

[3] UNSPSC (Universal Standard Products and Services Classification) is an open global coding system that classifies products and services. UNSPSC is extensively used around the world for electronic catalogs, search engines, e–procurement applications and accounting systems.

	automatic classification[4]		after manual revision[5]	
Total items found	324	100%	324	100%
Rightly classified	197	60%	246	76%
Wrongly classified	67	21%	17	5%
Non classified	60	19%	61	19%

In particular, the automatic classification percentages are computed comparing the algorithm results with the pre-existent mappings. After manual review, the mappings automatically discovered by the algorithm improved the manual ones.

6 Related Work

Rahm and Bernstein [12] suggest that there are three general strategies for matching schemas: *instance based* (using similarity between the objects (e.g., documents) associated to the schema to infer the relationship between the concepts); *schema–based* (determining the relationships between concepts analyzing the structure of a hierarchy and the meanings of the labels); and *hybrid* (a combination of the two strategies above). Our algorithm falls in the second group. In this section, we briefly compare our method with some of the most promising schema–based methods recently proposed, namely MOMIS [2] a schema based semi automatic matcher, CUPID [9] a schema based automatic matcher and GLUE [6] an instance based automatic matcher.

The MOMIS (Mediator envirOnment for Multiple Information Sources) [2]) is a framework to perform information extraction and integration from both structured and semistructured data sources. It takes a global–as–view approach by defining a global integrated schema, starting from a set of sources schema. In one of the first phases of the integration, MOMIS supports the discovery of overlapping (relations) between the different source schema. This is done by exploiting the knowledge in a Common Thesaurus with a combination of clustering techniques and Description Logics. Another difference between the matching algorithm implemented in MOMIS and CtxMatch is that MOMIS includes an interactive process as a step of the integration procedure, and thus does not support a fully automatic and run-time generation of mappings.

More similar to CtxMatch is the algorithm proposed in [9], called CUPID. This is an algorithm for generic schema matching, based on a weighted combination of names, data types, constraints and structural matching. This algorithm uses a limited amount of linguistic knowledge, as it associates a thesaurus to each schema. However, unlike CtxMatch, it does not exploit the whole power of a linguistic resource like WordNet. Another difference between CUPID and CtxMatch is that CUPID discovers relations between two schemas S and T only when S and the embedding of S in T are structurally isomorphic. As a

[4] Manually verified by ourselves.
[5] Manually verified by Alessandro Cederle Managing Director of Kompass Italia

consequence, CUPID cannot deal with concepts that are intuitively equivalent, but are represented as non isomorphic schemas.

A different approach to ontology matching has been proposed in [6]. Although the aim of the work (i.e. establishing mappings among concepts of overlapping ontologies) is in many respects similar to ours, the methodologies are significantly different. A major difference is that the GLUE system builds mappings taking advantage of information contained in instances, while the current version of the CtxMatch algorithm completely ignores them. This makes CtxMatch more appealing, since most ontologies currently available in the Semantic Web do not contain a significant collection of instances. A second difference concerns the use of domain-dependent constraints, which, in case of the GLUE system, need to be provided manually by domain experts, while in CtxMatch they are automatically extracted from an already existing resource (i.e. WordNet). Finally, CtxMatch provides a qualitative characterization of mappings in terms of the relation between two concepts, a feature which is not considered in GLUE. Even though a comparison with the results reported in [6] is rather difficult, the accuracy achieved by CtxMatch can be roughly compared with the accuracy of the GLUE module which uses less information (i.e., the "name learner").

7 Conclusions

In the paper, we presented a first version of an algorithm for matching semantic schemas – viewed as contexts – via SAT.

We believe that this work can have a significant impact from a theoretical point of view. Indeed, the scientific challenge behind the algorithm is to determine what is the minimal common ground to enable communication between entities that do not share common meanings (at least, not in the sense of the approaches that assume the necessity of a shared ontology to enable communication). As a consequence, the relations discovered by the algorithm are always directional (from a concept in a context to concept in another context, but not vice versa), and this reflects the idea that what is a good mapping from the point of view encoded in a context might not be acceptable from the point of view encoded in the other context.

Of course, a lot of work remains to be done, and in particular: generalizing the types of structures we can match (beyond concept hierarchies); taking into account a larger collection of explicit assumptions; going beyond WordNet as a source of linguistic and domain knowledge.

References

1. M. Benerecetti, P. Bouquet, and C. Ghidini. Contextual Reasoning Distilled. *Journal of Theoretical and Experimental Artificial Intelligence*, 12(3):279–305, July 2000.
2. Sonia Bergamaschi, Silvana Castano, and Maurizio Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.

3. M. Bonifacio, P. Bouquet, and R. Cuel. Knowledge Nodes: the Building Blocks of a Distributed Approach to Knowledge Management. In *To be appear in a special issue of the J.UCS journal and in the Proceedings of I-KNOW '02 - 2nd International Conference on Knowledge Management.*, Gratz - Austria, July 11–12, 2002. Springer Pub. & Co.
4. M. Bonifacio, P. Bouquet, and P. Traverso. Enabling distributed knowledge management. managerial and technological implications. *Novatica and Informatik/Informatique*, III(1), 2002.
5. 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.
6. 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*, page ....., 2002.
7. C. Ghidini and F. Giunchiglia. Local models semantics, or contextual reasoning = locality + compatibility. *Artificial Intelligence*, 127(2):221–259, April 2001.
8. F. Giunchiglia and L. Serafini. Multilanguage hierarchical logics, or: how we can do without modal logics. *Artificial Intelligence*, 65(1):29–70, 1994. Also IRST-Technical Report 9110-07, IRST, Trento, Italy.
9. Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *The VLDB Journal*, pages 49–58, 2001.
10. B. Magnini, L. Serafini, and M. Speranza. Linguistic based matching of local ontologies. In P. Bouquet, editor, *Working Notes of the AAAI-02 workshop on Meaning Negotiation. Edmonton (Canada)*, Edmonton, Alberta, Canada, July 2002. AAAI, AAAI Press.
11. B. M. Magnini, L. Serafini, A. Doná, L. Gatti, C. Girardi, and M. Speranza. Large–scale evaluation of context matching. Technical Report 0301–07, ITC–IRST, January 2003.
12. Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal: Very Large Data Bases*, 10(4):334–350, 2001.
13. S. L. Star. Working together: Symbolic interactionism, activity theory, and iformation systems. In *Communication and Cognition at Work*, pages 296–318. Cambridge University Press, 1997.