A SAT-BASED ALGORITHM FOR CONTEXT MATCHING

Paolo Bouquet, Bernardo Magnini, Luciano Serafini
and Stefano Zanobini

January 2003

Technical Report # DIT-03-005
A SAT–based algorithm for context matching

P. Bouquet¹,², B. Magnini², L. Serafini², and S. Zanobini¹

¹Department of Information and Communication Technologies – University of Trento
Via Inama, 5 – 38100 Trento (Italy)
²ITC-IRST – Istituto per la Ricerca Scientifica e Tecnologica
Via Sommarive, 14 – 38050 Trento (Italy)
bouquet@dit.unitm.it magnini@itc.it seraﬁni@itc.it
zanobini@dit.unitm.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 [2]). 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 meaningful information/knowledge across applications which (i) use autonomously developed conceptualizations of their domain, and (ii) need to collaborate to achieve their users’ goals.

Two are the main approaches proposed 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).

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” [14]. 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 [8]), 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” [9] 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., di-
rectories, taxonomies, corporate ontologies), and are then supported in finding relevant knowledge in other local schemas available in the corporate network.

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., the Google and the 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 [8] 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.

To this end, 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 \( c = (R_c, A_c) \), 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 context of the content.

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 \( \mu : K \to 2^D \).
\( \mu \) satisfies the following \textit{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\(^1\).

Mappings between contexts are defined as follows:

\textbf{Definition 3 (Mapping function).} A mapping function \( M \) from \( H = (K, E, I) \) to \( H' = (K', E', I') \) is a function \( M : K \times K' \rightarrow \text{rel} \), where \text{rel} is set of symbols, called \textbf{the possible mappings}.

The set \text{rel} of possible mappings we consider in this paper contains the following: \( k_s \xrightarrow{\supseteq} k_s \) for \( k_s \) is more general than \( k_t \); \( k_s \xrightarrow{=} k_t \) for \( k_s \) is less general than \( k_t \); \( k_s \xrightarrow{\equiv} k_t \) 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 \( H_t \):

\textbf{Definition 4.} A \textbf{mapping function} \( M \) from \( H_s \) to \( H_t \) is \textbf{extensionally correct with respect to two hierarchical classifications} \( \mu_s \) and \( \mu_t \) 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{align*}
&k_s \xrightarrow{\supseteq} k_t \Rightarrow \mu_s(k_s) \supseteq \mu_t(k_t) \\
&k_s \xrightarrow{=} k_t \Rightarrow \mu_s(k_s) \subseteq \mu_t(k_t) \\
&k_s \xrightarrow{\equiv} k_t \Rightarrow \mu_s(k_s) \cap \mu_t(k_t) = \emptyset \\
&k_s \xrightarrow{\perp} k_t \Rightarrow \mu_s(k_s) = \mu_t(k_t) \\
&k_s \xrightarrow{=\equiv} k_t \Rightarrow \mu_s(k_s) \cap \mu_t(k_t) \neq \emptyset
\end{align*}

where \( \mu(c_d) \) 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 [8, 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.

\(^1\)See Yahoo\ instruction for “Finding an appropriate Category” at http://docs.yahoo.com/info/suggest/appropriate.html.
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 \( w(d') \) 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 [11]), 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.

\(^2\) 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.
Linguistic interpretation Let \( H = \langle K, E, I \rangle \) be a concept hierarchy and \( L_H \) the set of labels associated to the nodes and edges of a hierarchy \( H \) by the function \( I \). 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.

**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 \( \sqcap, \sqcup \) and \( \neg \), whose primitive concepts are the synsets of WORDNET that we associate to each label (with a suitable interpretation of conjunctions, disjunctions, multi-words, 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 \sqcap 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 \( H = \langle K, E, I \rangle \), is a finite concept hierarchy \( f(k, H) = \langle K', E', I' \rangle \) 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 \( K' \); \( I' \) is the restriction of \( I \) 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
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 \( \phi \) and \( \psi \) are expressed. The portion of \( B \) relevant to \( \phi \) and \( \psi \), is a subset \( T \) of \( B \) such that \( T \) contains all the axioms of \( B \) containing some concept occurring in \( \phi \) or \( \psi \).

Clearly different contexts can be associated to different background theories, which encode 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 justify the introduction of directionality in the mapping i.e. \( c_s \xrightarrow{\leq} c_t \) means that \( c_s \) is more general than \( c_t \) according to the target perspective, while the relation \( c_t \xrightarrow{\geq} c_s \) 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 one a background theory \( B \) determined by transforming the \textsc{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 \textsc{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 \textsc{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
and Chat and Forum in the Yahoo directory in Figure 1. From WordNet we can
extract the following relevant axioms:

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

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

$$\text{humanities#1} \sqsubseteq \text{literature#2}$$

(the sense 1 of ‘humanities’ is an hypernym 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 interpre-
tation. But which are the logical relations of $w(k)$ and $w(k’)$ that encodes a
mapping function between $k$ and $k’$ as given in Definition 3? Again, the en-
coding of the mapping into a logical relation is a matter of heuristics. Here we
propose the translation described in Table 2. In this table $T_i$ is the portion of the
background theory of $c_i$ relevant to $k_i$ and $k$. 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#1, 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:

$$\text{art#1} \sqsubseteq \text{humanities#1}$$ \hspace{1cm} (1)
$$\text{humanities#1} \sqsubseteq \text{literature#2}$$ \hspace{1cm} (2)
$$\text{(art#1} \sqsubseteq \text{literature#2}) \cap$$ \hspace{1cm} (3)
$$\text{(chat#1} \sqsubseteq \text{forum#1})$$
$$\text{(art#1} \sqsubseteq \text{humanities#1}) \cap$$ \hspace{1cm} (4)
$$\text{humanities#1} \sqsubseteq \text{(chat#1} \sqsubseteq \text{forum#1})$$
<table>
<thead>
<tr>
<th>relation</th>
<th>SAT Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_1 \rightarrow k_2$</td>
<td>$T_i \models w(k_1) \subseteq w(k_2)$</td>
</tr>
<tr>
<td>$k_1 \subseteq k_2$</td>
<td>$T_i \models w(k_1) \subseteq w(k_2)$</td>
</tr>
<tr>
<td>$k_1 \rightarrow k_2$</td>
<td>$T_i \models w(k_1) \cap w(k_2) \subseteq \bot$</td>
</tr>
<tr>
<td>$k_1 \rightarrow k_2$</td>
<td>$T_i \models w(k_1) \subseteq w(k_2)$ and $T_i \models w(k_2) \subseteq w(k_1)$</td>
</tr>
<tr>
<td>$k_1 \rightarrow k_2$</td>
<td>$w(k_2) \cap w(k_1)$ is consistent in $T_i$</td>
</tr>
</tbody>
</table>

Table 2. Verifying relations as a SAT problem

It is easy to see that from the above axioms we can infer (3) $\subseteq$ (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 $\phi \cap \psi$ 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 $\phi$ on the models that satisfy $\psi$. This measure gives a first estimation on how much $\psi$ is a generalization of $\phi$, the lower percentage, the higher generalization.

5 Testing the algorithm

In this section we briefly report from [12] 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:
An interesting observation is 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:

<table>
<thead>
<tr>
<th>Relations</th>
<th>Architecture</th>
<th>Medicine</th>
</tr>
</thead>
<tbody>
<tr>
<td>equivalence</td>
<td>.71 .10</td>
<td>.78 .13</td>
</tr>
<tr>
<td>less general</td>
<td>.85 .49</td>
<td>.88 .46</td>
</tr>
<tr>
<td>more general</td>
<td>.51 .91</td>
<td>.60 .78</td>
</tr>
</tbody>
</table>

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 into UNSPSC\(^3\) (version 5.0.2) the catalog of the office equipment and accessories used to classify company suppliers. The validity of the relations found by the algorithm, shown in the following table, were double-checked manually.

<table>
<thead>
<tr>
<th></th>
<th>automatic classification(^4)</th>
<th>after manual revision(^5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total items</td>
<td>324</td>
<td>324</td>
</tr>
<tr>
<td>Rightly classified</td>
<td>197</td>
<td>245</td>
</tr>
<tr>
<td>Wrongly classified</td>
<td>67</td>
<td>17</td>
</tr>
<tr>
<td>Non classified</td>
<td>60</td>
<td>61</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>100%</th>
<th>76%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>21%</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>19%</td>
<td>19%</td>
</tr>
</tbody>
</table>

6 Related work

Rahm and Bernstein\(^1\) 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\(^10,6\) a schema based automatic matcher and GLUE\(^7\) an instance based automatic matcher.

The MOMIS (Meditator Environment for Multiple Information Sources)\(^2\) is a framework to perform information extraction and integration from both structured and semi-structured 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\(^10\), 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

\(^3\) 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.

\(^4\) Manually verified by ourselves.

\(^5\) Manually verified by Alessandro Cederle Managing Director of Kompass Italia.
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
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 [7]. Al-
though the aim of the work (i.e. establishing mappings among concepts of over-
lapping ontologies) is in many respects similar to ours, the methodologies are
significantly different. A major difference is that the GLUE system builds map-
pings taking advantage of information contained in instances, while the current
version of the CTXMATCH algorithm completely ignores them. This makes CTX-
MATCH more appealing, since most ontologies currently available in the Seman-
tic 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. Word-
Net). 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 [7] 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 deter-
mine 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 communi-
cation). 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


