ITC irst

CENTRO PER LA RICERCA
SCIENTIFICA E TECNOLOGICA

38050 Povo (Trento), Italy
Tel.: +39 0461 314312
Fax: +39 0461 302040
e−mail: prdoc@itc.it − url: http://www.itc.it

Peer to Peer Semantic Coordination

Bouquet P., Serafini L., Zanobini S.

February 2004

Technical Report # T04−02−04

Peer-to-Peer Semantic Coordination

P. Bouquet[1,2], L. Serafini[2] and S. Zanobini[1]

[1] Department of Information and Communication Technology – University of Trento

Via Sommarive, 10 – 38050 Trento (Italy)

[2]ITC-IRST – Istituto per la Ricerca Scientifica e Tecnologica

Via Sommarive, 14 – 38050 Trento (Italy)

bouquet@dit.unitn.it serafini@itc.it zanobini@dit.unitn.it

February 6, 2004

Abstract

The problem of computing/discovering mappings across heterogeneous schemas (e.g., classifications, taxonomies, catalogs, data types definitions) is one of the key issues in the development of the Semantic Web. In this paper, we argue that this problem can be viewed as a problem of semantic coordination (namely, as a problem of coordinating the meaning of portions of schemas through a collection of semantic mappings), and propose a new method for discovering semantic mappings called CTXMATCH. This approach shifts the problem of semantic coordination from the problem of computing linguistic or structural similarities (what most other proposed approaches do) to the problem of deducing relations between sets of logical formulae that represent the meaning of elements belonging to different schema. We show how to apply the method to an interesting family of schemas (namely hierarchical classifications), and present the results of preliminary tests on two types of hierarchical classifications, web directories and catalogs. Finally, we argue why this is a significant improvement on previous approaches.

1 Introduction

One of the key issues in the development of the Semantic Web is the problem of enabling machines to exchange meaningful information/knowledge across applications which (i) may use autonomously developed schemas (e.g. taxonomies, classifications, database schemas, data types) for organizing locally available data, and (ii) need to discover relations between schemas to achieve their users' goals. This problem can be viewed as a problem of coordination, defined as follows: (i) all parties have an interest in finding an agreement on how to map their schemas onto each others, but (ii) there are many possible/plausible solutions (many alternative mappings across local schemas) among which they need to select the right, or at least a sufficiently good, one. For this reason, we see this as a problem of *semantic coordination*[1].

[1]See the introduction of [4] for this notion, and its relation with the notion of *meaning negotiation*.

In environments with more or less well-defined boundaries, like a corporate Intranet, the problem of semantic coordination can be addressed *a priori* by defining and using shared schemas (e.g. ontologies) throughout the entire organization[2]. However, in open environments, like the Semantic Web, this "centralized" approach to semantic coordination is not viable for several reasons, such as the difficulty of "negotiating" a shared model that suits the needs of all parties involved, the practical impossibility of maintaining such a model in a highly dynamic environment, the problem of finding a satisfactory mapping of pre-existing local schemas onto such a global model. In such a scenario, the problem of exchanging meaningful information across locally defined schemas (each possibly presupposing heterogeneous semantic models) seems particularly tough, as we cannot assume an *a priori* agreement, and therefore its solution requires a more dynamic and flexible form of coordination, which we call "peer-to-peer" semantic coordination.

In this paper, we address an important instance of the problem of peer-to-peer semantic coordination, namely the problem of coordinating hierarchical classifications (HCs). HCs are structures having the *explicit* purpose of organizing/classifying some kind of data (such as documents, goods, activities, services). The problem of coordinating HCs is significant for at least two main reasons:

- first, HCs are widely used in many applications[3]. Examples are: web directories (see e.g. the Google™ Directory or the Yahoo!™Directory), content management tools and portals (which often use hierarchical classifications to organize documents and web pages), service registry (web services are typically classified in a hierarchical form, e.g. in UDDI), marketplaces (goods are classified in hierarchical catalogs), PC's file systems (where files are typically classified in hierarchical folder structures);

- second, it is an empirical fact that most actual HCs (as most concrete instances of models available on the Semantic Web) are built using structures whose labels are expressions from the language spoken by the community of their users (including technical words, neologisms, proper names, abbreviations, acronyms, whose meaning is shared in that community). In our opinion, recognizing this fact is crucial to go beyond the use of syntactic (or weakly semantic) techniques, as it gives us the chance of exploiting the complex degree of semantic coordination implicit in the way a community uses the language from which the labels of a HC are taken.

The main technical contribution of this part is an algorithm, called CTXMATCH, which takes in input two HCs H and H' and, for each pair of concepts $k \in H$ and $k' \in H'$, returns their semantic relation (called a semantic mapping). The idea is that mappings across semantic models can then be used by other application to answer queries (e.g., by finding documents classified under an unknown category in another HC) or more in general to provide services which require an agreement on the meaning of terms.

[2]But see [3] for a discussion of the drawbacks of this approach from the standpoint of Knowledge Management applications.

[3]For an interesting discussion of the central role of classification in human cognition see, e.g., [12, 5].

With respect to other approaches to semantic coordination proposed in the literature (often under different "headings", such as schema matching, ontology mapping, semantic integration; see Section 6 for references and a detailed discussion of some of them), our approach is innovative in three main aspects: (1) we introduce a new method for making explicit the meaning of nodes in a HC (and in general, in structured semantic models) by combining three different types of knowledge, each of which has a specific role; (2) the result of applying this method is that we are able to produce a new representation of a HC, in which all relevant knowledge about the nodes (including their meaning in that specific HC) is encoded as a set of logical formulae; (3) mappings across nodes of two HCs are then deduced via logical reasoning, rather then derived through some more or less complex heuristic procedure, and thus can be assigned a clearly defined model-theoretic semantics. As we will show, this leads to a major conceptual shift, as the problem of semantic coordination between HCs is no longer tackled as a problem of computing linguistic or structural similarities (possibly with the help of a thesaurus and of other information about the type of arcs between nodes), but rather as a problem of deducing relations between (the models of) formulae representing the meaning of nodes in a given HC (namely, the concept expressed by that node in that HC). This explains, for example, why our approach performs much better than other ones when two concepts are intuitively equivalent, but occur in structurally very different HCs.

The paper goes as follows. In Section 2 we introduce the main conceptual assumptions of our approach. In Section 3 we show how this approach is instantiated to the problem of coordinating HCs. Then we present the main features of CTXMATCH, the proposed algorithm for coordinating HCs (Section 4). In the final part of the paper, we sum-up the results of testing the algorithm on web directories and catalogs (Section 5) and compare our approach with other proposed approaches for matching schemas (Section 6).

2 Our approach

The method we propose assumes that we deal with a network of physically connected entities which can autonomously decide how to organize locally available data; we call these entities "semantic peers". Peers organize their data using one or more schemas (e.g., database schemas, directories in a file system, classification schemas, taxonomies, and so on); as we said, in this paper we focus on classifications. Different peers may use different schemas to classify the same collection of documents/data, and conversely the same schemas can be used to organize different collections of documents/data.

We also assume that semantic peers need to exchange data (in our scenario, this means documents classified under categories belonging to distinct classification schemas). To do this, each semantic peer needs to discover "mappings" between its local classification schema and other peers' schemas. Intuitively, a mapping can be viewed as a set of pairwise relations between elements of two distinct classification schemas.

The first idea behind our approach is that *mappings must represent semantic relations*, namely relations with a well-defined model-theoretic interpretation. This is an important difference with respect to approaches based on matching techniques, where

a mapping is a measure of (linguistic, structural, . . .) similarity between schemas (e.g., a real number between 0 and 1). The main problem with the latter techniques is that the interpretation of their results is an open problem. For example, how should we interpret a 0.9 similarity? Does it mean that one concept is slightly more general than the other one? Or maybe slightly less general? Or that their meaning 90% overlaps (whatever that means)? Instead, our method returns semantic relations, e.g. that the two concepts are (logically) equivalent, or that one is (logically) more/less general, or that they are mutually exclusive. As we will argue, this gives us many advantages, essentially related to the consequences we can infer from the discovery of such a relation[4].

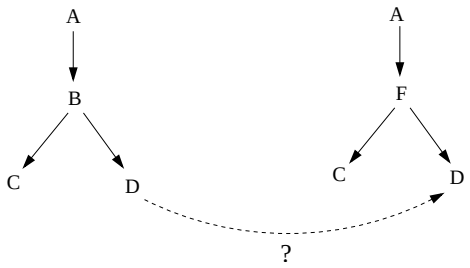

Figure 1: Mapping abstract structures

The second idea is that, to discover semantic relations, one must make explicit the meaning implicit in each element of a schema. The claim is that making explicit the meaning of elements is the necessary premise for computing semantic relations between elements of distinct schemas, and that this can be done only for schemas in which meaningful labels are used, where "meaningful" means that their interpretation is not arbitrary, but is contrained by the conventions of some community of speakers/users[5].

To illustrate this idea, we discuss the difference between the problem of mapping abstract schemas (like those in Figure 1) and the problem of mapping schemas with meaningful labels (like those in Figure 2). Nodes in abstract schemas do not have an implicit meaning, and therefore *any technique we may use to map pairs of nodes* (e.g., the two nodes `D` in the two schemas) *will return a relation which depends only on the abstract form of the two schemas and on syntactic features of labels*. The situation is completely different for schemas with meaningful labels. Consider for example the two pairs of structures depicted in Figure 2. Both are structurally equivalent to the pair of abstract schemas depicted in Figure 1. However, despite this similarity, we can easily understand that the relation between the two nodes `MOUNTAIN` is 'less than', while the relation between the two nodes `FLORENCE` is 'equivalent'. Indeed, for the first pair of nodes, the set of documents we would classify under the node `MOUNTAIN` on the left hand side is a subset of the documents we would classify under the node `MOUNTAIN`

[4]For a more detailed discussion of the distinction between syntactic and semantic methods, see [9].

[5]Notice that these conventions is 'codified' in artifacts (e.g., dictionaries, but today also ontologies and other formalized models), which provide senses for words (and also for more complex expressions), relations between senses, and other important knowledge about them. Our aim is to exploit these artifacts as an essential source of constraints on possible/acceptable mappings across structures.

on the right; whereas the set of documents which we would classify under the node FLORENCE in the left schema is exactly the same as the set of documents we would classify under the node FLORENCE on the right hand side. The reason of this difference resides in the presence of meaningful labels which allow us to make explicit a lot of information that we have about the terms which appear in the graph, and their relations (e.g., that Tuscany is part of Italy, that Florence is in Tuscany, and so on). It's only this information which allows us to understand why the semantic relation between the two nodes MOUNTAIN and the two nodes FLORENCE is different.

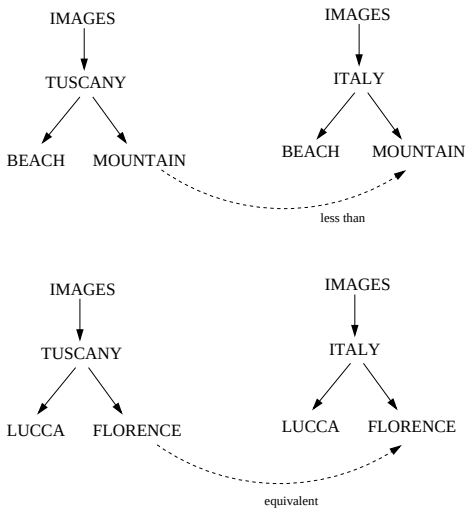

Figure 2: Mapping schemas with meaningful labels

This approach gives us the chance of exploiting the complex degree of semantic coordination implicit in the way a community uses the language from which the labels are taken. The method is based on the explicitation of the meaning associated to each node in a schema (notice that schemas such as the two classifications in Figure 2 *are not* semantic models themselves, as they do not have the purpose of defining the meaning of terms they contain; however, they *presuppose* a semantic model, and indeed that's the only reason why we humans can read them quite easily). The explicitation process uses three different levels of knowledge:

Lexical knowledge: knowledge about the words used in the labels. For example, the fact that the word 'Florence' can be used to indicate 'a city in Italy' or 'a town in northeast South Carolina', and to handle the synonymy;

World knowledge: knowledge about the relation between the concepts expressed by words. For example, the fact that Tuscany is part of Italy, or that Florence is in Italy;

Structural knowledge: knowledge deriving from how labeled nodes are arranged in a given schema. For example, the fact that the node labeled MOUNTAIN is below a node IMAGES tells us that it classifies images of mountains, and not, say, books about mountains.

As an example of how the three levels are used, consider again the mapping between the two nodes MOUNTAIN of Figure 2. Lexical knowledge is used to determine what concepts can be expressed by each label, e.g. that the word 'Images' can denote the concept 'a visual representation produced on a surface'. World knowledge tells us, among other things, that Tuscany is part of Italy. Finally, structural knowledge tells us that the intended meanings of the two nodes MOUNTAIN is 'images of Tuscan mountains' on the left hand side, and 'images of Italian mountains' on the right hand side. Using this information, human reasoners (i) understand the meaning expressed by the left hand node, ('images of Tuscan mountains', denoted by P), (ii) understand the meaning expressed by the right hand node ('images of Italian mountains', denoted by P'), and finally (iii) understand the semantic relation between the meaning of the two nodes, namely that $P \subseteq P'$.

This analysis of meaning has an important consequence on our approach to semantic coordination. Indeed, unlike all other approaches we know of, we do not use lexical knowledge (and, in our case, world knowledge) to improve the results of structural matching (e.g., by adding synonyms for labels, or expanding acronyms). Instead, we combine knowledge from all three levels to build a new representation of the problem, where the meaning of each node is encoded as a logical formula, and relevant world knowledge and structural relations between nodes are added to nodes as sets of axioms that capture background knowledge about them.

This, in turn, introduces another feature of our approach. Indeed, once the meaning of each node, together with all relevant domain and structural knowledge, is encoded as a set of logical formulae, the problem of discovering the semantic relation between two nodes can be stated not as a matching problem, but as a relatively simple problem of logical deduction. Intuitively, as we will say in a more technical form in Section 4, determining whether there is an equivalence relation between the meaning of two nodes becomes a problem of testing whether the first implies the second and vice versa (given a suitable collection of axioms, which acts as a sort of background theory); and determining whether one is less general than the other one amounts to testing if the first implies the second. In the current version of the algorithm we encode this reasoning problem as a problem of logical satisfiability, and then compute mappings by feeding the problem to a standard SAT solver.

3 P2P coordination of hierarchical classifications

In this section we show how to apply the general approach described in the previous section to the problem of coordinating HCs. Intuitively, a classification is a grouping of things into classes or categories. When categories are arranged into a hierarchical structure, we have a hierarchical classification. Formally, the hierarchical structures we use to build HCs are *concept hierarchies*, defined as follows in [6]:

Definition 1 (Concept hierarchy) *A* concept hierarchy *is a triple $S = \langle N, E, l \rangle$ where N is a finite set of nodes, E is a set of arcs on N, such that $\langle N, E \rangle$ is a rooted tree, and l is a function from N to a set L of labels.*

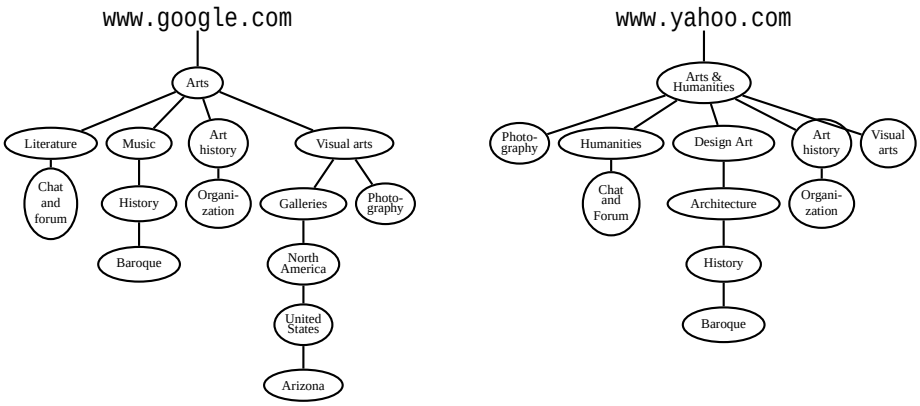

Figure 3: Examples of concept hierarchies (source: Google!Directory and Yahoo!Directory)

Essentially, a concept hierarchy is a rooted tree where the categories are the nodes of the tree[6]. Given a concept hierarchy S, a classification can be defined as follows:

Definition 2 (Classification) *A* classification *of a set of objects D in a concept hierarchy $S = \langle N, E, l \rangle$ is a function $\mu : N \rightarrow 2^D$.*

Prototypical examples of HCs are the web directories of many search engines, as for example the Google™ Directory, the Yahoo!™ Directory, or the Looksmart™ web directory, or a file system. A tiny fraction of the HCs corresponding to the Google™ Directory™ and to the Yahoo!™Directory is depicted in Figure 3.

Intuitively, the problem of semantic coordination arises when one needs to find semantic mappings between nodes belonging to distinct (and thus typically heterogeneous) HCs. Formally, we define a semantic mapping between two nodes belonging to distinct classifications S and S' as follows:

Definition 3 (Mapping) *A* mapping *M from $S = \langle N, E, l \rangle$ to $S' = \langle N', E', l' \rangle$ is a function $M : N \times N' \rightarrow rel$, where rel is a set of names of possible relations.*

The set *rel* of possible (semantic) relations depends on the intended use of the structures we want to map. Indeed, in our experience, the intended use of a structure (e.g., classifying objects) is semantically much more relevant than the type of abstract structures involved to determine how a structure should be interpreted. As the purpose of mapping HCs is to discover relations between nodes (concepts) that are used to classify objects, five semantic relations can hold between two nodes m and n belonging to different HCs: $m \xrightarrow{\supseteq} n$ (m is more general than n); $m \xrightarrow{\subseteq} n$ (m is less general than n); $m \xrightarrow{\equiv} n$ (m is equivalent to n); $m \xrightarrow{*} n$ (m is compatible with n); $m \xrightarrow{\perp} n$ (m is disjoint from n).

[6]Hereafter node, category and classes are used in the same sense.

4 The algorithm: CTXMATCH

CTXMATCH (see Algorithm 1) takes two HCs as input and returns a mapping between the nodes of the two classifications as output (in what follows, we assume that the two HCs used in our examples are those depicted in the lower part of Figure 2).

The algorithm has essentially two main macro steps:

Semantic Explicitation: in this phase, the meaning of each node m and n in two classifications S and S' is made explicit by using knowledge extracted by a lexicon L (lexical knowledge) and from an ontology O (world knowledge). The output is a pair $\langle \phi, \Theta \rangle$, where ϕ is a logical formula which approximates the meaning expressed by the node m (n) in S (in S'), and Θ is a set of relevant axioms extracted from O (see Section 4.1 for details and examples). As a result, each node in S and S' is associated to what we call a *contextualized concept*, which formalizes the implicit meaning each node in the structure it belongs to.

Semantic comparison: in this phase the problem of finding the semantic relation between two nodes m and n is encoded as the problem of finding the semantic relation holding between the contextualized concepts, $\langle \phi, \Theta \rangle$ and $\langle \psi, \Upsilon \rangle$ associated to m and n respectively. The outcome of this phase is one of the admissible semantic relations, e.g. an equivalence relation between the two nodes FLORENCE in Figure 2.

Algorithm 1 CTXMATCH(S,$S'L$,O)
- ▷ *Hierarchical Classifications* S, S'
- ▷ *Lexicon* L
- ▷ *World knowledge* O
- **VarDeclaration:**
  - contextualized concept $\langle \phi, \Theta \rangle, \langle \psi, \Upsilon \rangle$
  - relation R
  - mapping M

1 **for** each pair of nodes $m \in S$ and $n \in S'$ **do**
2 $\quad \langle \phi, \Theta \rangle \leftarrow$ SEMANTIC−EXPLICITATION(m, S, L, O);
3 $\quad \langle \psi, \Upsilon \rangle \leftarrow$ SEMANTIC−EXPLICITATION(n, S', L, O);
4 $\quad R \leftarrow$ SEMANTIC−COMPARISON$(\langle \phi, \Theta \rangle, \langle \psi, \Upsilon \rangle, O)$;
5 $\quad M \leftarrow M \cup \langle m, n, R \rangle$;
6 **return** M;

The two following sections describe in detail these two top-level operations, implemented by the functions SEMANTIC−EXPLICITATION and SEMANTIC−COMPARISON.

4.1 Semantic explicitation

In this phase we make explicit in a logical formula the meaning of a labeled node into a structure, by means of a lexical and a world knowledge. For practical reasons, the current version of the algorithm uses WORDNET [10] as a source of both lexical and domain knowledge. However, we stress that WORDNET could be replaced by

other combinations of a linguistic resource and a world knowledge resource (in particular, we are currently working on using generic OWL ontologies as a source of world knowledge).

The resulting formula is written in some logical language $\mathcal{L}$. The choice of $\mathcal{L}$ depends on the degree of expressiveness required to encode the meaning of a schema's elements, and on the complexity of the NLP techniques used to process labels. For simple concept hierarchies, we adopted a simple propositional encoding, where each propositional letter corresponds to a concept ("synset") extracted from WORDNET.

The function EXTRACT–LOCAL–AXIOMS exploits lexical knowledge to associate to each word occurring in the label of a node the set of concepts possibly denoted by that word. For example, the label 'Florence' is associated with two WORDNET synsets, namely 'a city in central Italy on the Arno' (`florence#1`) and a 'a town in northeast South Carolina' (`florence#2`). To maximize the possibility of finding an entry into the Lexicon, we use both a POS tagger and a lemmatizator. For complex labels (i.e., labels composed by two or more words), we check whether there is a corresponding multiword synset and the corresponding sense is selected; otherwise, we extract all possible senses for each word.

Algorithm 2 SEMANTIC–EXPLICITATION(t,S,L,O)

▷ *t is a node in S*
▷ *structure S*
▷ *lexicon L*
▷ *world knowledge O*

VarDeclaration:
single concept $con[][]$
set of axioms Σ
set of simple concepts Γ
formula δ

1 **for** each node n in S **do**
2 $con[n] \leftarrow$ EXTRACT–CANDIDATE–CONCEPTS(n,L);
3 $\Sigma \leftarrow$ EXTRACT–LOCAL-AXIOMS$(t,S,con[],O)$;
4 $con[] \leftarrow$ FILTER–CONCEPTS$(S,\Sigma,con[])$;
5 **for** each node n in S **do**
6 $\Gamma \leftarrow$ BUILD–SIMPLE–CONCEPT$(con[],n)$;
7 $\delta \leftarrow$ BUILD–COMPLEX–CONCEPT$(t,S,con[])$;
8 **return** $\langle \delta, \Sigma \rangle$;

The function EXTRACT–LOCAL–AXIOMS exploits world knowledge to find relevant ontological relations existing between concepts in a structure. Actually, the function do not try to find ontological relations between a node n and any other node in the structure, but only between n and the nodes belonging to what we call its *focus*. Intuitively, the focus of a node n in a classification S, denoted by $f(n,S)$, is the smallest sub-tree of S that one should take into account to determine the meaning of n in S. In CTXMATCH, the focus is defined as follows:

Definition 4 (Focus) *The* focus *of a node n in a classification $S = \langle N,E,l \rangle$, is a finite concept hierarchy $f(n,S) = \langle N',E',l' \rangle$ such that: $N' \subseteq N$, and N' contains exactly n,*

its ancestors, and their children; $E' \subseteq E$ is the set of edges between the concepts of N'; l' is the restriction of l on N'.

Less formally, $f(n,S)$ includes all the nodes on the path from n to the root of the classification, and – for each of them – their siblings[7]. We notice that this definition is appropriate only for HCs. For structures having different purposes, different definitions of focus may be needed. For example, for a data type definition (e.g., an XML Schema complex type definition), the meaning an element is defined also by its sub-elements, so a more suitable definition of focus would include the sub-tree rooted at n.

As an example, consider again the left structure in Figure 2. Imagine that the concept 'a region in central Italy' (`tuscany#1`) has been associated to the node `TUSCANY`. The function EXTRACT–LOCAL–AXIOMS checks if there exists some relation between the concept `tuscany#1` and the concepts associated to other nodes, namely `florence#1` and `florence#2` (associated to node `FLORENCE`), `images#1`, ..., `images#8` (associated to node `IMAGE`), and `lucca#1` (associated to node `LUCCA`). We would discover that '`florence#1` is part of `tuscany#1`', i.e. that there exists a 'part of' relation between the first sense of 'Florence' and the first sense of 'Tuscany'.

World knowledge relations are translated into logical axioms, according to Table 1. So, the relation '`florence#1` PartOf `tuscany#1`' is encoded as '$\texttt{florence\#1} \rightarrow \texttt{tuscany\#1}$'.

WORDNET relation	axiom
s#k synonym t#h	$\texttt{s\#k} \equiv \texttt{t\#h}$
s#k { hyponym | PartOf }t#h	$\texttt{s\#k} \rightarrow \texttt{t\#h}$
s#k { hypernym | HasPart }t#h	$\texttt{t\#h} \rightarrow \texttt{s\#k}$
s#k contradiction t#h	$\neg(\texttt{t\#k} \wedge \texttt{s\#h})$

Table 1: WORDNET relations and their axioms.

The function FILTER–CONCEPTS filters out unlikely senses associated to a node in the previous phases. Going back to the previous example, the sense 2 of 'Florence' ('a town in northeast South Carolina', `florence#2`), can be intuitively discarded on the basis that the node `FLORENCE` occurs as a child of the node `Tuscany`. In CTXMATCH, this result is achieved automatically by analyzing the extracted local axioms: the presence of an axiom such as '$\texttt{florence\#1} \rightarrow \texttt{tuscany\#1}$' is used to make the conjecture that the contextually relevant sense of Florence is the city in Tuscany, and not the city in USA. When ambiguity persists (axioms related to different senses or no axioms at all), all the possible senses are left.

The formula which approximates the meaning of an element in a structure is built in two steps. First, the function BUILD–SIMPLE–CONCEPT builds a formula which represents the interpretation of the meaning expressed by (the labels of) the nodes independently from the position in which they occur in the structure. Starting from simple

[7] This definition is motivated by observations on how we humans use HCs. When searching for documents in a HC, we incrementally construct the meaning of a node n by navigating the classification from the root to n. During this navigation, we have access to the labels of the ancestors of n, and also to the labels of their siblings. This information is used at each stage to build the meaning expressed by a node in a structure.

concepts, the function BUILD–COMPLEX–CONCEPT builds the formula approximating the meaning expressed by a node into a structure. Let us see how this happens in detail.

For each node $n \in S$, BUILD–SIMPLE–CONCEPT takes the list of concepts which passed the filter of FILTER–CONCEPTS and builds a logical formula (a *simple concept*) which represents the admissible linguistic interpretations of the label of n.

In Figure 2, the simple concept approximating the meaning expressed by the node IMAGES is the formula image#1 $\vee \ldots \vee$ image#8 (the eight senses provided by WORDNET), the meaning expressed by the node TUSCANY is the atom tuscany#1 (the only sense provided by WORDNET), while the meaning of the node FLORENCE is approximated by the atom florence#1 (one of the two senses provided by WORDNET and not discarded by the filtering). But, of course, more complicated cases can be handled. Consider the two classifications depicted in Figure 3. The following *simple concepts* can be built:

Google

- simple concept expressed by node Baroque = baroque#1, the unique sense of 'Baroque' presents in WORDNET;
- simple concept expressed by node Chat and Forum = chat#1 $\vee$ chat#2 $\vee$ chat#3 $\vee$ forum#1 $\vee$ forum#2 $\vee$ forum#3 i.e. the disjunction of the meaning of 'chat' and 'forum' taken separately (both 'chat' and 'forum' have tree senses in WORDNET);
- simple concept expressed by node North America = (north america#1 $\vee$ north america#2) the senses associated to the multiword 'North America' in Lexicon (WORDNET).

Yahoo

- simple concept expressed by node Visual Arts = visual art#1 $\wedge \neg$ photography#1: both Visual Arts and Photography are sibling nodes under Arts & Humanities; since in WORDNET the concept photography#1 is in a *IsA* relationship with the concept visual art#1, the node Visual arts is re-interpreted as visual arts with the exception of photography.

BUILD–COMPLEX–CONCEPT uses the simple concepts produced by BUILD–SIMPLE–CONCEPT to build a formula which approximates the meaning expressed by a node in a structure (the *complex concept* expressed by a node). In the current version of the algorithm, the complex concept expressed by a node n is built as the conjunction of the simple concepts associated to all its ancestors (i.e., the path from root to n). So, the complex concept associated to the node FLORENCE is (image#1 $\vee \ldots \vee$ image#8) $\wedge$ tuscany#1 $\wedge$ florence#1.

It is important to observe that this particular encoding depends on the fact that the current version of CTXMATCH deals with HCs. Indeed, in a classification, it is always the case that documents classified under a node n could be classified under any ancestor of n as well. For example, images classified under the category IMAGES.TUSCANY.FLORENCE could be classified also under the node TUSCANY (if we

didn't have the node FLORENCE. This means that the formula ϕ associated to a node *n* (e.g., to FLORENCE)ϕ must logically imply the formula ψ associated to a node *m* ancestor of *n* (e.g., TUSCANY). In other words, it must be the case that $\phi \rightarrow \psi$. In our current encoding, this is exactly what happens. For example, if the complex concept associated to the node TUSCANY is $(\texttt{image\#1} \vee \ldots \vee \texttt{image\#8}) \wedge \texttt{tuscany\#1}$ (say, ψ) and the complex concept associated to the node FLORENCE is $(\texttt{image\#1} \vee \ldots \vee \texttt{image\#8}) \wedge \texttt{tuscany\#1} \wedge \texttt{florence\#1}$ (say, ϕ), then we have that the first entails the second.

Finally, for each node in a HC *S*, the phase of semantic explicitation returns a *contextualized concept*, namely a pair in which we have a complex concept and the set of local axioms returned by EXTRACT–LOCAL–AXIOMS (step 3).

4.2 Semantic comparison

After semantic explicitation is over, the problem of discovering semantic relations between two nodes *m* and *n* in two HCs can be reduced to the problem of checking if a logical relation holds between two formulas ϕ and ψ: this is checked as a problem of propositional satisfiability (SAT), and then computed via a standard SAT solver.

Algorithm 3 SEMANTIC–COMPARISON($\langle \phi, \Theta \rangle, \langle \psi, \Upsilon \rangle, O$)
▷ *contextualized concept* $\langle \phi, \Theta \rangle$
▷ *contextualized concept* $\langle \psi, \Upsilon \rangle$
▷ *world knowledge O*

VarDeclaration:
set of formulas Γ
relation *R*

1 $\Gamma \leftarrow$ EXTRACT–RELATIONAL–AXIOMS(ϕ, ψ, O);
2 $R \leftarrow$ FIND–SEMANTIC–RELATION($\langle \phi, \Theta \rangle$, $\langle \psi, \Upsilon \rangle$, Γ)
3 **return** *R*;

First, the function EXTRACT–RELATIONAL–AXIOMS takes as input two contextualized concepts $\langle \phi, \Theta \rangle$ (from *S*) and $\langle \psi, \Upsilon \rangle$ (from a HC S') and tries to extract from *O* new axioms which can connect complex concepts belonging to different HCs (called *relational axioms*). The procedure is analogous to that of function EXTRACT–LOCAL–AXIOMS described above. Consider, for example, the senses italy#1 and tuscany#1 associated respectively to nodes ITALY and TUSCANY of Figure 2: the relational axioms express the fact that, for example, 'Tuscany is part of Italy', and is translated into the axiom $\texttt{tuscany\#1} \rightarrow \texttt{italy\#1}$, according to Table 1.

The problem of finding the semantic relation between two nodes *n* and *m* (line 2) is encoded as a satisfiability problem involving both the contextualized concepts associated to the two nodes and the relational axioms extracted in the previous phase. This function is done by the function FIND–SEMANTIC–RELATION.

Five possible semantic relations are allowed: disjoint ($\perp$), equivalent ($\equiv$), less than ($\subseteq$), more than ($\supseteq$) and compatible ($\cap$). Algorithm 4 checks which relation holds by running the SAT solver on five satisfiability problems (lines 1 to 5), where the sets of formulas Θ, Δ, Γ represent the local axioms of the source node, the local axioms of the target node and the relational axioms respectively, and ϕ and ψ represent the *complex*

concepts approximating the meaning expressed by the source node and the target node respectively. Note that the compatibility relation is leaved as default case (line 5).

Algorithm 4 FIND−SEMANTIC−RELATION$(\langle \phi, \Theta \rangle, \langle \psi, \Delta \rangle, \Gamma)$

▷ *contextualized concept* $\langle \phi, \Theta \rangle, \langle \psi, \Delta \rangle$

▷ *set of formulas* Γ

VarDeclaration:
semantic relation R

1 **if** $\Theta, \Delta, \Gamma \models \neg(\phi \wedge \psi)$ **then** $R \leftarrow \perp$;
2 **else if** $\Theta, \Delta, \Gamma \models (\phi \equiv \psi)$ **then** $R \leftarrow \equiv$;
3 **else if** $\Theta, \Delta, \Gamma \models (\phi \rightarrow \psi)$ **then** $R \leftarrow \subseteq$;
4 **else if** $\Theta, \Delta, \Gamma \models (\psi \rightarrow \phi)$ **then** $R \leftarrow \supseteq$;
5 **else** $R \leftarrow \cap$;
6 **return** R;

Going back to our main example, to prove whether the two nodes labeled FLORENCE in Figure 2 are equivalent, we check the logical equivalence between the formulas approximating the meaning of the two nodes, given the local and the relational axioms. Formally, we have the following satisfiability problem:

Θ	$\texttt{florence\#1} \rightarrow \texttt{tuscany\#1}$
ϕ	$(\texttt{image\#1} \vee \ldots \vee \texttt{image\#8}) \wedge \texttt{tuscany\#1} \wedge \texttt{florence\#1}$
Δ	$\texttt{florence\#1} \rightarrow \texttt{italy\#1}$
ψ	$(\texttt{image\#1} \vee \ldots \vee \texttt{image\#8}) \wedge \texttt{italy\#1} \wedge \texttt{florence\#1}$
Γ	$\texttt{tuscany\#1} \rightarrow \texttt{italy\#1}$

It is simple to see that the returned relation is '$\equiv$'. Note that the satisfiability problem for finding the semantic relation between the nodes MOUNTAIN of Figure 2 is the following:

Θ	$\emptyset$
ϕ	$(\texttt{image\#1} \vee \ldots \vee \texttt{image\#8}) \wedge \texttt{tuscany\#1} \wedge \texttt{mountain\#1}$
Δ	$\emptyset$
ψ	$(\texttt{image\#1} \vee \ldots \vee \texttt{image\#8}) \wedge \texttt{italy\#1} \wedge \texttt{mountain\#1}$
Γ	$\texttt{tuscany\#1} \rightarrow \texttt{italy\#1}$

The returned relation is '$\subseteq$'.

5 Testing the algorithm

In this section, we report from [14] some results of the first test on CTXMATCH on real HCs (i.e., pre-existing classifications used in real applications). The first example we discuss is quite simple, and concerns the nodes of the portion of the Google and Yahoo classifications depicted in Figure 3; the second is a much more complex and articulated example of product reclassification with real catalogs.

5.1 Matching portions of Google and Yahoo

The table below reports some of the mappings discovered by CTXMATCH running on the portion of the Google and Yahoo classifications depicted in Figure 3:

Google node	Yahoo node	semantic relation founded
`Baroque`	`Baroque`	Disjoint ($\perp$)
`Visual Arts`	`Visual Arts`	More general than ($\supseteq$)
`Photography`	`Photography`	Equivalent ($\equiv$)
`Chat and Forum`	`Chat and Forum`	Less general than ($\equiv$)

In the first example, CTXMATCH returns a 'disjoint' relation between the two nodes `Baroque`: the presence of two different ancestors (`Music` and `Architecture`) and the related world knowledge 'Music is disjoint with Architecture' allow us to derive the right semantic relation. In the second example, CTXMATCH returns 'more general than' relation between the nodes `Visual Arts`. This is a very sophisticated result: as we said before, world knowledge provides the information that 'photography *IsA* visual art' (`photography#1` $\rightarrow$ `visual art#1`). From structural knowledge, we can deduce that, while in the left structure the node `Visual Arts` denotes the whole concept (in fact `photography` is one of its child), in the right structure the node `Visual Arts` denotes the concept 'visual arts except photography' (in fact `photography` is one of its siblings). Given this information, it easy to deduce that, despite the two nodes lying in the same path, they have different meanings,

The third example shows how the right relation holding between nodes `Photography` is returned ('equivalence'), despite the presence of different paths, because of the presence of the world knowledge axiom `photography#1` $\rightarrow$ `visual art#1`.

Finally, between the nodes `Chat and Forum` a 'less general than' relation is founded because of the presence of the world knowledge axiom 'literature *IsA* humanities'.

5.2 Use case Product Re-classification

In order to centrally manage all the company acquisition processes, the headquarter of a well known worldwide telecommunication company had realized an e-procurement system[8], which all the company branch-quarters were required to join. Each single office was also required to migrate from the product catalogue they used to manage, to this new one managed within the platform. This catalogue is extracted from the Universal Standard Products and Services Classification (UNSPSC), which is an open global coding system that classifies products and services. The UNSPSC is used extensively around the world in the electronic catalogues, search engines, procurement application systems and accounting systems. UNSPSC is a four-level hierarchical classification; an extract is reported in the following table:

Level 1 **Furniture and Furnishings**
Level 2 **Accommodation furniture**

[8] An e-procurement system is a technological platform which supports a company in managing its procurement processes and, more in general, the re-organization of the value chain on the supply side.

Level 3	**Furniture**
Level 4	**Stands**
Level 4	**Sofas**
Level 4	**Coat racks**

The Italian office asked us to apply the matching algorithm to re-classify into UNSPSC (version 5.0.2) the catalogue of office equipment and accessories used to classify company suppliers. The result of running CTXMATCH over UNSPSC and the catalogue can be cleary interpreted in terms of re-classification: if the algorithm returns that the item *i* of the catalogue is equivalent to, or more specific than, the node c_{UNSPSC} of UNSPSC, then *i* can be classified under c_{UNSPSC} of UNSPSC.

The items to be re-classified are mainly labeled with Italian phrases, but labels also contain abbreviations, acronyms, proper names, some English phrases and some typing errors. The English translation of an extract of this list is reported in the following table. The italic parts were contained in the original labels.

Code	Description
ENT.21.13	cartridge *hp desk jet 2000c*
ENR.00.20	magnetic tape cassette *exatape 160m xl 7,0gb*
ESA.11.52	*hybrid roller pentel* red
EVM.00.40	safety scissors, length 25 cm

The item list was matched with two UNSPSC's-segments, namely: *Office Equipment and Accessories and Supplies* (segment 44) and *Paper Materials and Products* (segment 14).

Notice that the company item catalogue we had to deal with was a plain list of items, each identified with a numerical code composed of two numbers, the first referring to a set of more general categories. For example, the number 21 at the beginning of *21.13-cartridge hp desk jet 2000c* corresponds to *printer tapes, cartridge and toner*. We first normalized and matched the plain list against UNSPSC. This did not lead us to a satisfactory result. The algorithm performed much better when we made the hierarchical classification contained in the item codes explicit. This was done by substituting the first numerical code of each item with their textual description provided by experts of the company.

After running CTXMATCH, the validation phase of our results was made by comparing them with the results of a simple keyword-based algorithm. Obviously, in order to establish the correctness of results in terms of precision and recall we have to compare them with a correct and complete matching list. Not having such a list, we asked a domain expert, Alessandro Cederle, Managing Director of Kompass Italia[9] to manually validate them.

Results

This section presents the results of the re-classification phase. Consider first the baseline matching process. The baseline was performed by a simple keyword-based match-

[9] Kompass (www.kompass.com) is a company which provides product information, contacts and other information about 1.8 million companies worldwide. All companies are classified under the Kompass Product Classification with more than 52,000 products and services.

ing that worked according to the following rule:

> *for each item description (made up of one or more words) return the set of nodes, and their paths, which maximize the occurrences of the item words*

The following tables summarizes the results of the baseline matching:

	Baseline classification	
Total items	194	100%
Rightly classified	75	39%
Wrongly classified	92	47%
Non classified	27	14%

Given the 193 items to be re-classified, the baseline process found 1945 possible nodes in UNSPSC. This means that for each item the baseline found an average of 6 possible classifications. What is crucial is that only 75 out of the 1945 proposed nodes are correct. The baseline, being a simple string matching, is able to capture a certain number of re-classifications. However the percentage of error is quite high (47%) with respect to the one of correctness (39%). The results of the matching algorithm are reported in the following table:

	CTXMATCH **classification**	
Total items	194	100%
Rightly classified	136	70%
Wrongly classified	16	8%
Non classified	42	22%

In this case, the percentage of success is sensibly higher (70%) and, even more relevant, the percentage of error is minimal (8%)[10]. This is also confirmed by the values of precision and recall, computed with respect to the validated list:

	Total matches	**Precision**	**Recall**
Baseline	1945	4%	39%
CTXMATCH	641	21%	70%

The baseline precision level is quite small, while the matching one is not excellent, but definitely better. The same observations can be made also for the recall values.

Table 2 reports some examples where the algorithm found a correct item for re-classification, while the baseline did not.

If there are not enough information to infer semantic relation, CTXMATCH returns a percentage, which is intended to represent the degree of compatibility between the two elements. Degree of compatibility is computed on the basis of a linguistic co-occurrence measures. Examples of compatibility relations are contained in the last four rows of Table 2.

As far as the Non Classified items, notice that:

[10] Notice that the algorithm did not take into account just the UNSPSC level 4 category, since in some cases catalogues items can be matched with UNSPSC level 3 category nodes.

Some examples of matching found by CTX-MATCH *and not found by the baseline. Items (depicted in columns) are matched against* UNSPSC *classes (depicted in raws).* $\supseteq$ *stands for "more general than", and percentages represent the compatibility degree.*	drill/ drill with 2-4 holes	printing tape, toner, cartridge , printing head	lampostil pen, marker, highlighter/ highlighter	ball-point pen, pen	double ruler / double ruler in white plastic	cleaning kit/ PC cleaning kit	cleaning kit/ cleaning kit for exatape 4 mm tape heads
Offi ce machines, materials and accessories/ Supplies for offi ce / Supplies for writing-desks/ Paper staplers	$\subseteq$						
Offi ce machines, materials and accessories/ Supplies of printing, telecoping-machine e coping-machine/ Ink cartridges		$\subseteq$					
Offi ce machines, materials and accessories Printing forniture, telecoping-machine e coping-machine/ Toner		$\subseteq$					
Supplies for offi ce/ Tools for writing/ Highlighters			$\subseteq$				
Supplies for offi ce/ Tools for writing/ Assortment of pens and penciles				70%			
Accessories for the offi ce and the writing desk/ Accessories for drawing					71%		
Offi ce machines, materials and accessories/ Accessories for offi ce machines/ computer cleaning kit						80%	
Materials for offi ce / Offi ce machines, materials and accessories/ Accessories for offi ce machines/ Cleaners of tapes							72%

Table 2:

- In some cases, the item to be re-classified were not correctly classified in the company catalogue. Therefore, CTXMATCH could not compute the relations with the node and its father node, in the right way. Examples are: *ashtray* was classified under *tape dispenser*; *wrapping paper* was classified under *adhesive labels*.

- In other cases, semantic coordination was not discovered due to a lack of domain knowledge. For instance to match *paper for hp* with UNSPSC class of printer paper it would have been necessary to know that *hp* stands for Helwett Packard, and that it is a company which produces printers.

In a further experiment, we run CTXMATCH between the company catalogue (in italian) and the English version of UNSPSC. This was possible because the matching is computed on the basis of the WORDNET sense IDs, and in the version of WORDNET we used, wordnet-senses ID of italian and english words are aligned (i.e., the wordnet-sense ID associated to word and its translation in the other language is the same). This experiment allows us to find more semantic matches.

More in general, this way allows us to approach and manage multilanguage environments and to exploit the richness which typically characterizes the English version of linguistic resources[11].

6 Related work

CTXMATCH shifts the problem of semantic coordination from the problem of matching (in a more or less sophisticated way) semantic structures (e.g., schemas) to the problem of deducing semantic relations between sets of logical formulae. Under this respect, to the best of our knowledge, there are no other works to which we can compare ours.

However, it is important to see how CTXMATCH compares with the performance of techniques based on different approaches to semantic coordination. There are four other families of approaches that we will consider: graph matching, automatic schema matching, semi-automatic schema matching, and instance based matching. For each of them, we will discuss the proposal that, in our opinion, is more significant. The comparison is based on the following five dimensions: (1) if and how structural knowledge is used; (2) if and how lexical knowledge is used; (3) if and how domain knowledge is used; (4) if instances are considered; (5) the type of result returned. The general results of our comparison are reported in Table 3.

In graph matching techniques, a concept hierarchy is viewed as a tree of labelled nodes, but the semantic information associated to labels is substantially ignored. In this approach, matching two graphs G_1 and G_2 means finding a sub-graph of G_2 which is isomorphic to G_2 and report as a result the mapping of nodes of G_1 into the nodes of G_2. These approaches consider only structural knowledge and completely ignore lexical and domain knowledge. Some examples of this approach are described in [18, 17, 16, 15, 11].

[11] The results of this experiment is not reported as they are not comparable with our simple keyword-based baseline, which makes no sense with multiple languages.

	graph matching	CUPID	MOMIS	GLUE	CTXMATCH
Structural knowledge	•	•	•		•
Lexical knowledge		•	•	•	•
Domain knowledge				•	•
Instance-based knowledge				•	
Type of result	Pairs of nodes	Similarity measure $\in [0..1]$ between pairs of nodes	Similarity measure $\in [0..1]$ between pairs of nodes	Similarity measure $\in [0..1]$ between pairs of nodes	Semantic relations between pairs of nodes

Table 3: Comparing CTXMATCH with other methods

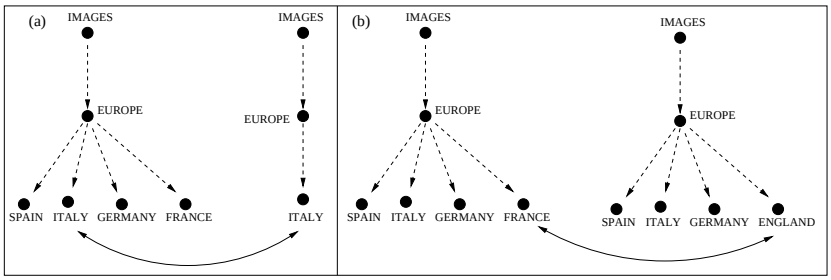

Figure 4: Example of right and wrong mapping

CUPID [13] is a completely automatic algorithm for schema matching. Lexical knowledge is exploited for discovering linguistic similarity between labels (e.g., using synonyms), while the schema structure is used as a matching constraint. That is, the more the structure of the subtree of a node s is similar to the structure of a subtree of a node t, the more s is similar to t. For this reason CUPID is more effective in matching concept hierarchies that represent data types rather than hierarchical classifications. With hierarchical classifications, there are cases of equivalent concepts occurring in completely different structures, and completely independent concepts that belong to isomorphic structures. Two simple examples are depicted in Figure 4. In case (a), CUPID does not match the two nodes labelled with `ITALY`; in case (b) CUPID finds a match between the node labelled with `FRANCE` and `ENGLAND`. The reason is that CUPID combines in an additive way lexical and structural information, so when structural similarity is very strong (for example, all neighbor nodes do match), then a relation between nodes is inferred without considering labels. So, for example, `FRANCE` and `ENGLAND` match because the structural similarity of the neighbor nodes is so strong that labels are ignored.

MOMIS (Mediator envirOnment for Multiple Information Sources) [1] is a set of tools for information integration of (semi-)structured data sources, whose main objective is to define a global schema that allow an uniform and transparent access to the data stored in a set of semantically heterogeneous sources. One of the key steps

of MOMIS is the discovery of overlappings (relations) between the different source schemas. This is done by exploiting knowledge in a Common Thesaurus together with a combination of clustering techniques and Description Logics. The approach is very similar to CUPID and presents the same drawbacks in matching hierarchical classifications. Furthermore, MOMIS includes an interactive process as a step of the integration procedure, and thus, unlike CTXMATCH, it does not support a fully automatic and run-time generation of mappings.

GLUE [8] is a taxonomy matcher that builds mappings taking advantage of information contained in instances, using machine learning techniques and domain-dependent constraints, manually provided by domain experts. GLUE represents an approach complementary to CTXMATCH. GLUE is more effective when a large amount of data is available, while CTXMATCH is more performant when less data are available, or the application requires a quick, on-the-fly mapping between structures. So, for instance, in case of product classification such as UNSPSC or Eclss (which are pure hierarchies of concepts with no data attached), GLUE cannot be applied. Combining the two approaches is a challenging research topic, which can probably lead to a more precise and effective methodology for semantic coordination.

7 Conclusions

In this paper we presented a new approach to semantic coordination in open and distributed environments, and an algorithm (called CTXMATCH) that implements this method for hierarchical classifications. The algorithm has already been used in a peer-to-peer application for distributed knowledge management (the application is described in [2]), and is going to be applied in a peer-to-peer wireless system for ambient intelligence [7].

An important lesson we learned from this work is that methods for semantic coordinations should not be grouped together on the basis of the type of abstract structure they aim at coordinating (e.g., graphs, trees), but on the basis of the intended use of the structures under consideration. In this paper, we addressed the problem of coordinating concept hierarchies when used to build hierarchical classifications. Other possible uses of structures are: conceptualizing some domain (ontologies), describing services (automata), describing data types (schemas). This "pragmatic" level (i.e., the use) is essential to provide the correct interpretation of a structure, and thus to discover the correct mappings with other structures.

The importance we assign to the fact that HCs are labelled with meaningful expressions does not mean that we see the problem of semantic coordination as a problem of natural language processing (NLP). On the contrary, the solution we provided is mostly based on knowledge representation and automated reasoning techniques. However, the problem of semantic coordination is a fertile field for collaboration between researchers in knowledge representation and in NLP. Indeed, if in describing the general approach one can assume that some linguistic meaning analysis for labels is available and ready to use, we must be very clear about the fact that real applications (like the one we described in Section 4) require a massive use of techniques and tools from NLP, as a good automatic analysis of labels from a linguistic point of view is a necessary precondition

for applying the algorithm to HC in local applications, and for the quality of mappings resulting from the application of the algorithm.

The work we presented in this paper is only the first step of a very ambitious scientific challenge, namely to investigate what is the minimal common ground needed to enable communication between autonomous entities (e.g., agents) that cannot look into each others head, and thus can achieve some degree of semantic coordination only through other means, like exchanging examples, pointing to things, remembering past interactions, generalizing from past communications, and so on. To this end, a lot of work remains to be done. On our side, the next steps will be: generalizing the types of structures we can match (for example, structures with non hierarchical relations, e.g. roles, which may require the use of more complex logical languages, e.g. description logic); going beyond WORDNET as a source of lexical and domain knowledge (e.g. testing other lexical sources, or using general purpose ontologies for extracting work knowledge); allowing different lexical and/or domain knowledge sources for each of the local structures to be coordinated (which means the computation of possibly asymmetric mappings from a node *n* to a mode *m* and vice versa). This last problem is perhaps the most challenging one, as it introduces a situation is which the space of 'senses' is not necessarily shared, and thus we cannot rely on that information for inferring a semantic relation between labels of distinct structures.

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] M. Bonifacio, P. Bouquet, G. Mameli, and M. Nori. Kex: a peer-to-peer solution for distributed knowledge management. In D. Karagiannis and U. Reimer, editors, *Fourth International Conference on Practical Aspects of Knowledge Management (PAKM-2002)*, Vienna (Austria), 2002.

[3] M. Bonifacio, P. Bouquet, and P. Traverso. Enabling distributed knowledge management. managerial and technological implications. *Novatica and Informatik/Informatique*, III(1), 2002.

[4] P. Bouquet, editor. *AAAI-02 Workshop on Meaning Negotiation*, Edmonton, Canada, July 2002. AAAI, AAAI Press.

[5] G. C. Bowker and S. L. Star. *Sorting things out: classification and its consequences*. MIT Press., 1999.

[6] A. Büchner, M. Ranta, J. Hughes, and M. Mäntylä. Semantic information mediation among multiple product ontologies. In *Proc. 4th World Conference on Integrated Design & Process Technology*, 1999.

[7] P. Busetta, P. Bouquet, G. Adami, M. Bonifacio, and F. Palmieri. K-Trek: An approach to context awareness in large environments. Technical report, Istituto

per la Ricerca Scientifi ca e Tecnologica (ITC-IRST), Trento (Italy), April 2003. Submitted to UbiComp'2003.

[8] 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.

[9] P. Shvaiko F. Giunchiglia. Semantic matching. *Proceedings of the workshop on Semantic Integration*, October 2003.

[10] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, US, 1998.

[11] Jeremy Carroll Hewlett-Packard. Matching rdf graphs. In *Proc. in the first International Semantic Web Conference - ISWC 2002*, pages 5–15, 2002.

[12] G. Lakoff. *Women, Fire, and Dangerous Things*. Chicago Unversity Press, 1987.

[13] Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *The VLDB Journal*, pages 49–58, 2001.

[14] B. M. Magnini, L. Serafi ni, A. Doná, L. Gatti, C. Girardi, , and M. Speranza. Large–scale evaluation of context matching. Technical Report 0301–07, ITC–IRST, Trento, Italy, 2003.

[15] 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.

[16] Marcello Pelillo, Kaleem Siddiqi, and Steven W. Zucker. Matching hierarchical structures using association graphs. *Lecture Notes in Computer Science*, 1407:3–??, 1998.

[17] 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.

[18] 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.