Using NLP Techniques for Meaning Negotiation

Bernardo Magnini, Luciano Serafini, Manuela Speranza
{magnini,serafini,manspera}@itc.it
ITC-irst, Centro per la Ricerca Scientifica e Tecnologica,
Via Sommarive, 38050 POVO (TN), Italy

Abstract. This paper describes an automatic algorithm of meaning negotiation that enables semantic interoperability between local overlapping and heterogeneous ontologies. Rather than reconciling differences between heterogeneous ontologies, this algorithm searches for mappings between concepts of different ontologies. The algorithm is composed of three main steps: (i) computing the *linguistic meaning* of the label occurring in the ontologies via natural language processing, (ii) *contextualization* of such a linguistic meaning by considering the context, i.e. the ontologies, where a label occurs; (iii) comparing contextualized linguistic meaning of two ontologies in in order to find a possible matching between them.

1 Introduction

In the last decade, knowledge has been recognized as one of the most important assets of modern organizations [5]. Well-known theoretical work, such as [9] led many big corporations to start huge organizational and economic investments to improve their practices of knowledge management.

As a managerial practice, Knowledge Management (KM) can be described as a collection of methodologies and tools that provide support in: creating new knowledge within the organization (the learning organization), in particular by transforming tacit knowledge into explicit knowledge; coding such newly created knowledge into "objects" (e.g. documents, repositories, databases, procedures, forms) that can be stored in a sort of organizational memory.

Many researchers suggest, under different names, an approach that in [2] is called a distributed approach to KM, namely an approach that (i) starts from the recognition that there exist autonomous communities within an organization, (ii) that they are more an opportunity than a problem, and (iii) that technology should support knowledge exchange not by eliminating differences, but by designing systems that will *enable interoperability* (in particular, semantic interoperability) between autonomous communities.

Autonomous communities organize their knowledge, from now on local *knowledge*, according to a *local ontology*. A local ontology is a set of terms and relations between them used by the members of the autonomous community to classify, communicate, update, and, in general, to operate with local knowledge. Materializations of a local ontology can be, for instance, the logical organization of a web site used by the community to share information,

the directory structure of a shared file system, the schema of a database used to store common knowledge, the tag-structure of an XML schema document used to describe documents or services shared by the members of the community. In all these cases, we think that two of the main intuitions underlying local ontologies are the following:

1. Each community (team, group, and so on) within an organization has its own conceptualization of the world, which is partial (i.e., covers only a portion of the world), approximate (i.e., has a degree of granularity), and perspectival (i.e., reflects the community's viewpoint on the world - including the organization and its goals and processes);
2. There are possible mappings between different and autonomous conceptualizations. These mappings cannot be defined beforehand, as they presuppose a complete understanding of the two conceptualizations, which in general is not available. This means that these mappings are discovered dynamically via a process that we call *meaning negotiation*.

The goal of this paper is to outline an automatic algorithm of meaning negotiation that enables semantic interoperability between local overlapping and heterogeneous ontologies of different autonomous communities.

In the next section we define a theoretical framework, *context space*, were local ontologies and mappings between local ontologies are represented. A context space is composed of a set of *contexts* and a set of *mappings*. Contexts are the main data structure used to represent local knowledge, mappings represent the results of matching two (or in general many) contexts. From a theoretical point of view, the notion of context we use is derived from the notion described in formal papers like [7, 1, 3].

In the Section "Linguistic-based interpretation" we describe the computing of the *local semantics* of a context. Knowledge in a context is represented by structured labeled "small" linguistic expressions, as complex noun phrases, prepositional phrases, abbreviations, etc. The semantics of this structure is computed by combining the semantics of each single label. The semantics of a single label depends on two factors: the first is the *linguistic meaning* of the label (independent of the context where they appear), the second is the *contextualization* of such a linguistic meaning. The former is computed by accessing a natural language semantic repository such as the electronic lexical database WORDNET [6, 8]. The latter is computed by combining the linguistic meaning of a label with the linguistic meaning of (some of) the other labels in the context.

In the last section we describe how the local semantics of the labels of different contexts are compared in order to find possible overlaps and mappings between two structures and finally we draw some conclusions.

2 Context space

In an organization, different local ontologies coexist; they can be identified by a name, and are located in some place of a virtual space. Our proposal is to formalize each local ontology by a *context*, and the virtual space by a *context space*. A context, as described in [7], is a partial representation of the world. In general a context is an autonomous representation, but is not completely independent of what holds in other contexts. For example, if two contexts describe the same portion of the world from different points of view, there are some obvious constraints on what is true in the two contexts. Model-theoretically, this means that there is a relation between the "local models" of each context (i.e., not all local models of a context

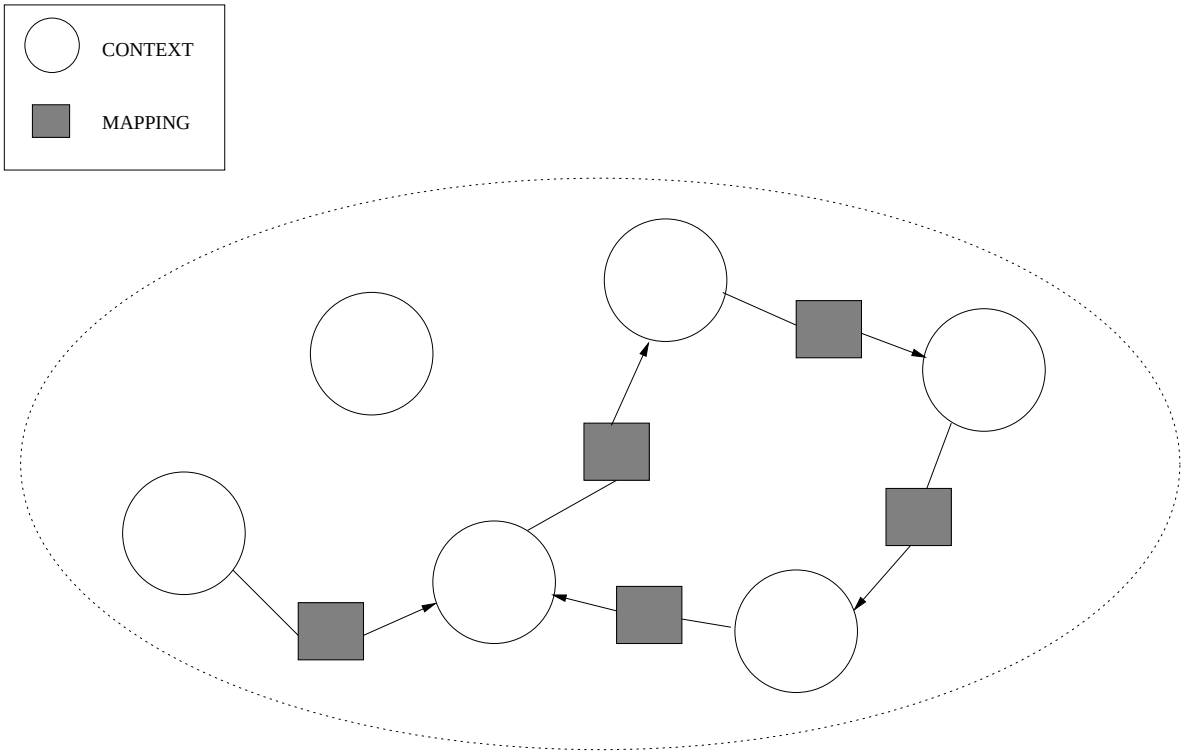


Figure 1: Context space

are compatible with the local models of the other one); proof-theoretically, this means that there are some "bridge rules" that allow us to infer new facts in a context from facts in the other. Intuitively bridge rules allow us to infer that a formula B, formalizing a certain state of the affairs s holds in a target context, if a formula A that formalizes the very same state of affairs s holds in a source context. When contexts are ontologies, we talk about concepts rather than properties. We therefore need mappings (bridges) between concepts in different ontologies. Context spaces therefore are also populated by *context mappings*. Like contexts, context mappings are created, changed, merged and combined with each other in a context space. A graphical representation of context space is given in Figure 1. Let us now define the components of a context space:

A context is a triple $\langle Cid, \mathcal{A}, \mathcal{R} \rangle$, where:

1. *Cid* is a unique identifier associated with a context.
2. $\mathcal{A}$ is a collection of explicit assumptions. Explicit assumptions are attributes (parameter/value pairs) that provide meta-information about the context (e.g., the owner of the context, or his history).
3. $\mathcal{R}$ is an explicit representation. The explicit representation is the real content of a context, namely a partial conceptualization. Possible reference models for context content can be based on first order logic, propositional logic, description logic, general graph structure (graph, acyclic graph, lattice, etc.), concept hierarchy [4]. Among these reference models, we chose concept hierarchy [4]

Concept hierarchies are built starting from a set L of labels composed of two disjoint subsets L_C and L_R, labels for concepts and labels for relations, respectively. L_R is composed of

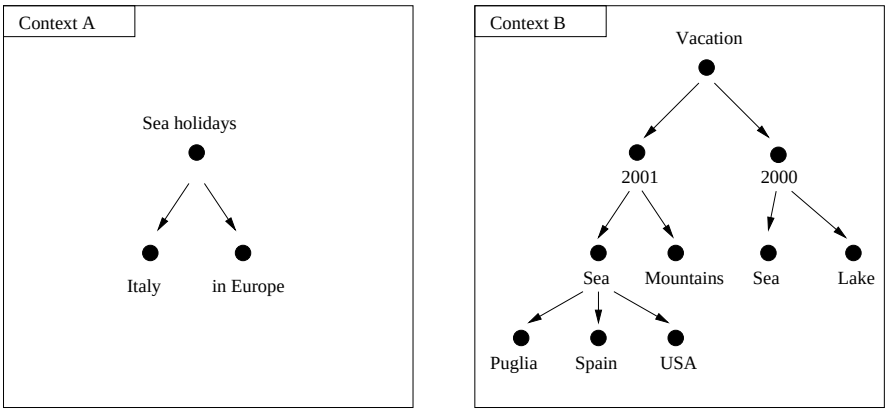


Figure 2: Two examples of concept hierarchies

two subsets L_H, labels for hierarchical relations, and L_G, labels for generic (non hierarchical) binary relations.

Definition 1 (concept hierarchy). A *concept hierarchy* is a graph $H = \langle C, E \rangle$ where C is a finite set of nodes and E a finite set of directed edges between nodes, and all the nodes and edges have a label, chosen in a set L of labels, such that the edges labeled with hierarchical labels form a tree.

At the present stage, we consider only concept hierarchies with a single hierarchical label, and no generic relational labels. In the future we will extend our algorithm to the case of multiple relations. Two examples of concept hierarchies are shown in Figure 2.

If two contexts in a context space conceptualize a common part of the world, we say that the two contexts *overlap*. For instance the context describing "car components" and the context describing "radio and hi-fi" overlap on "radio and hi-fi for cars". Despite this, it is not guaranteed that the common part about "radio and hi-fi for cars" is described in *the same way*. Contexts overlapping is represented by a structure called *context mapping*.

2.1 Mappings

A mapping is a relation between a context (called *source*) and another context (called *target*), with the following features:

- Context mapping is *directional*. We want to represent the situation where a context c_1 imports information from another context c_2 using a certain context mapping m, without forcing c_2 to use the same (or the inverse) mapping m to import the information for context c_1.

- Context mapping cannot limit itself to represent relations between equivalent concepts of two contexts. It should allow for the representation of relations between concepts at different abstract levels. For instance a context mapping should be able to represent the fact that the concept "printer" in a context is more general than the concept "laser printer" in some other context.

  A *context mapping* is a 5-tuple $\langle m, \mathcal{A}, \mathit{Cid}_s, \mathit{Cid}_t, \mathcal{M} \rangle$, where:

1. m is a unique identifier for the mapping (as for contexts);
2. $\mathcal{A}$ is a collection of explicit assumptions (as for contexts);
3. *Cid*$_s$ and *Cid*$_t$ are two different context identifiers, for the source context and the target context, respectively.
4. $\mathcal{M}$ is the real mapping, i.e. the actual relation between $\mathcal{R}_s$ and $\mathcal{R}_t$, the explicit representations of *Cid*$_s$ and *Cid*$_t$.

Since we assume that a context explicit representation is a concept hierarchy, $\mathcal{M}$ is a mapping from the set of concepts of the explicit representation of the source context, to the concepts of the explicit representation of the target context. More precisely:

Definition 2 (CH mapping). A *concept mapping* from a concept hierarchy $H_1 = \langle C_1, E_1 \rangle$, called *source*, to a context hierarchy $H_2 = \langle C_2, E_2 \rangle$, called *target*, is a set of 3-tuple $\langle c_i \ rel \ c_j \rangle$ where:

1. $rel \in \{\xrightarrow{\sqsubseteq}, \xrightarrow{\sqsupseteq}, \xrightarrow{*}, \xrightarrow{\perp}, \xrightarrow{\equiv}\}$;
2. $c_i \in C_1$;
3. $c_j \in C_2$.

Intuitively $c_1 \xrightarrow{\sqsupseteq} c_2$ means that c_1 is more general than c_2 (e.g., animal is more general than dog); $c_1 \xrightarrow{\sqsubseteq} c_2$ means that c_1 is less general than c_2; $c_1 \xrightarrow{\equiv} c_2$ means that $c_1 \xrightarrow{\sqsubseteq} c_2$ and $c_1 \xrightarrow{\sqsupseteq} c_2$; $c_1 \xrightarrow{\perp} c_2$ means that c_1 is disjoint from c_2 (e.g., mountain is disjoint from sea), $c_1 \xrightarrow{*} c_2$ means that c_1 is compatible with c_2 (e.g., cars are compatible with hi-fi, as there are hi-fi systems for cars).

A *context space* is a pair $\langle \mathbf{Ctx}, \mathbf{Map} \rangle$ where **Ctx** is a set of contexts and **Map** is a set of mappings between the contexts in **Ctx**.

Since the purpose of this document is to describe an algorithm that finds mappings from a source context to a target context, we develop all the concepts with respect to a context space composed of two contexts C_s and C_t, the source context and the target context, and a unique mapping from C_s to C_t.

3 Linguistic-Based Interpretation

Before starting the core matching process, source and target contexts must be pre-processed for linguistic disambiguation of labels. Once we have linguistically normalized contexts the matching process can start.

The main aim of linguistic normalization of contexts is the semantic interpretation of each concept label. Each label is associated with a set of senses, selected from WORDNET. Linguistic normalization is performed in two steps. In the first one we process each single label, independently of the context where it occurs, in the second one we refine the result of the first step, taking into account also the interaction between a label and all the other labels occurring in the context.

3.1 Meaningful labels

In principle, labels for concepts and relations of the context content can be any string. To allow semantic interpretation, however, we assume that most of the labels are linguistic expressions of some natural language. This language (e.g., "English", "Italian") is specified in one of the external parameters of the context. Furthermore, we suppose that a subset of the labels is taken from a specific domain vocabulary (e.g., "tourism", "electronic commerce", "agriculture", "law"). As for languages, domain is stored in a context external parameter.

Let us distinguish labels for concepts from labels for relations. We concentrate on labels for concepts, leaving out labels for relations.

3.2 Labels for concepts

Labels used in contexts allow a wide variety of linguistic expressions:

- simple labels (one-word labels): common nouns such as "vacation", proper nouns such as "Europe", abbreviations, such as "Mon." in place of "Monday", adjectives, pronouns, etc.;
- noun phrases (NP): "sea holidays", "the car", "sea holidays", "sea holidays car", "main seaside holidays", "old versions", "other things", "seaside holidays", "old versions", "other things", etc.;
- prepositional phrases (PP): "by train", "for Barbara", "for other things", "at the seaside", "for holidays," etc.;
- verb phrases (VP): "pay", "pay car insurance", etc.;
- complex labels: "books for selling" "holidays and trips", "paying and organizing", "paying holidays and organizing trips", "to pay and to organize", "paying holidays and organizing trips and other things", etc.

These linguistic expressions can combine with different kinds of separators (S), as in "Holidays.Spain" and "sea-holidays".

Table 1 shows the grammar used for analysisng context labels.

3.3 Labels for relations

In this first stage generic relations are not considered by the matching algorithm. This means that the matching algorithm behaves independently of the generic relations between concepts. In the rest of the paper we suppose therefore that $L_G = \emptyset$.

The set L_H of labels for hierarchical relations contains the three relations $\{$*isa*, *partof*, *instof*$\}$ with the following intuitive meaning:

isa representing the subclass relation: for instance, "Cat *isa* Animal", "Man *isa* Mortal".

partof representing the relation of being part of: for instance, "Leg *partof* Human body", "Tenor *partof* Choir".

L_C	::=	(S (C S)*)
S	::=	(NP | PP | VP | ((NP | PP)? P V))
VP	::=	(V (NP | PP)?)
NP	::=	(NP PP)
PP	::=	(P NP)
NP	::=	((D? A* (N | A) N* A*) | (Pr A*))
N	::=	common noun or proper noun
D	::=	article
A	::=	adjective
Pr	::=	pronoun
P	::=	simple preposition or complex preposition
V	::=	verb
C	::=	conjunction
S	::=	" " "⊥" "_" "." "," "(" ")"

Table 1: Grammar for conceptual context labels

.

instof representing that relation that a certain individual is an instance of a concept: for instance, "Paolo *instof* Man", "Michele *instof* Tenor".

Examples of concept hierarchies are shown in Figure 2. In this simplified example we have only one hierarchical relation (*isa*) and no generic (non hierarchical) relations.

3.4 Single label analysis

Linguistic and semantic analysis is performed taking into account each single concept of the context independently of the others.

Linguistic analysis. Linguistic analysis mainly consists of shallow parsing. A label is taken as input as it appears in the context; the output is a linguistic data structure providing the following information about each token contained in the input label:

- Identification number: a tokenizer identifies each single token within the label and provides it with an identification number.
- Lemma: a lemmatizer performs morphological analysis of each token, so as to find all possible normal forms and lexical categories of the token.
- Part of speech (PoS): a PoS tagger selects for each token the right lexical category among those proposed by the lemmatizer.
- Linguistic function: labels undergo functional decomposition. Complex noun phrases are decomposed into a head and a number of modifiers, while prepositional phrases are decomposed into a function word and a head.

For instance, the output of the linguistic analysis of the concept "Sea holidays" in context A (Figure 2) is represented by the following linguistic data structure:

Sea holidays		
ID	0	1
Token	Sea	holidays
Lemma	sea	holiday
PoS	N	N
Function	mod-1	head

Table 2: Output of the linguistic analysis of the concept "Sea holidays" (Context A in Figure 2)

.

Semantic analysis. Labels need to be interpreted according to world knowledge. As a repository of senses we use WORDNET [6], an electronic lexical database where the different senses of English words are grouped by synonymy. The sets of synonyms ("synsets") are organized hierarchically (i.e. each synset is connected to more general and more specific concepts) and other semantic relations (e.g. part-of relation, cause relation, etc.) are available so as to build a richer semantic net.

The algorithm takes each single lemma within a concept and checks whether it is contained in WORDNET. If a lemma has been found in WORDNET, the senses of that lemma are added to the linguistic data structure resulting from the previous phase.

Sea holidays		
ID	0	1
Token	Sea	holidays
Lemma	sea	holiday
PoS	N	N
Function	mod-1	head
W-senses	sea#1 sea#2 sea#3	holiday#1 holiday#2

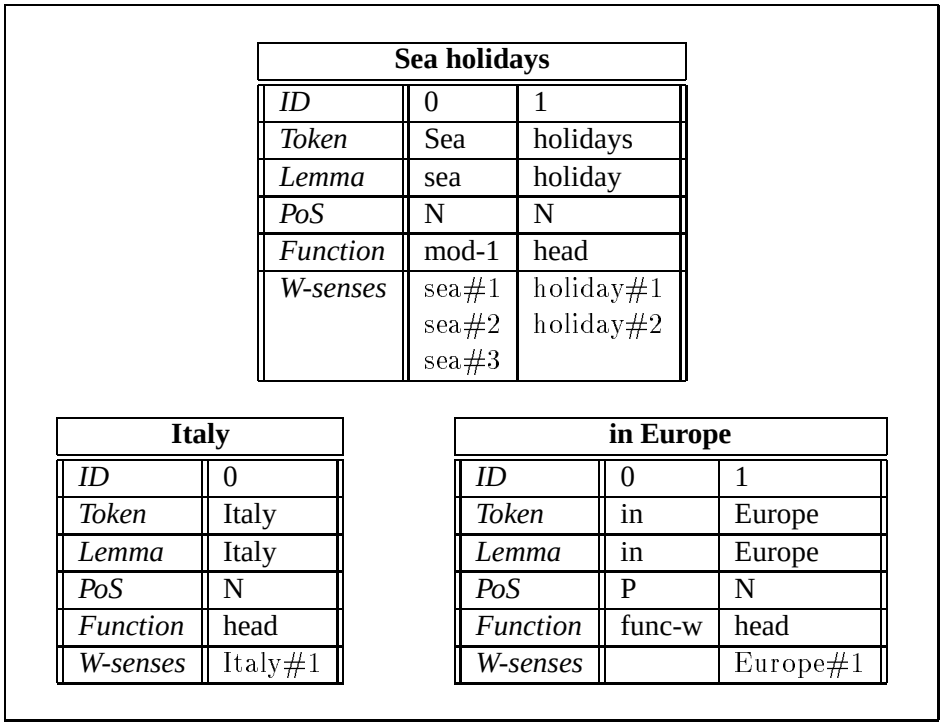
Table 3: Output of the linguistic and semantic analysis of the concept "Sea holidays" (Context A in Figure 2)

.

At this point we start to deal with linguistic concepts, i.e. WORDNET senses (hereafter "w-concepts"). The lemma "sea", for instance, can be found in WORDNET and therefore we have sea#1, sea#2 and sea#3, which means sense 1 ("sea" as "a division of an ocean"), sense 2 ("sea" as "anything apparently limitless"), and sense 3 ("sea" as "turbulent water") of "sea" as defined in WORDNET. Similarly, with "holiday" we have holiday#1 and holiday#2, which means senses 1 ("leisure time away from work") and 2 ("a day on which work is suspended") of the lemma "holiday". The next step has the main aim of eliminating the w-concepts that are not "compatible" with the other w-concepts that occur in the concept hierarchy.

3.5 Sense Refinement

Sense refinement consists of sense filtering and sense composition.



Sea holidays		
ID	0	1
Token	Sea	holidays
Lemma	sea	holiday
PoS	N	N
Function	mod-1	head
W-senses	sea#1 sea#2 sea#3	holiday#1 holiday#2

Italy	
ID	0
Token	Italy
Lemma	Italy
PoS	N
Function	head
W-senses	Italy#1

in Europe		
ID	0	1
Token	in	Europe
Lemma	in	Europe
PoS	P	N
Function	func-w	head
W-senses		Europe#1

Table 4: Output of the single label analysis applied to the focus shown on the left side of Figure 3.

Sense filtering. Sense filtering aims at eliminating the senses in disagreement with the other senses included in the hierarchy. For example, if there is a label "apple", which can denote either a computer brand or a fruit, and its parent node is labeled with "computer", it is clear that the sense "fruit" can be eliminated. Sense filtering can be performed by accessing the structural information about senses available in WORDNET. To represent relations between senses provided in WORDNET, we use the following notation, for any pair of w-senses $\mathrm{s\#k}$ and $\mathrm{t\#h}$:

- $\mathrm{s\#k} \leq_w \mathrm{t\#h}$ means that $\mathrm{s\#k}$ is either a hyponym or a meronym of $\mathrm{t\#h}$;
- $\mathrm{s\#k} -_w \mathrm{t\#h}$ means that $\mathrm{s\#k}$ belongs to the set of opposite meanings of $\mathrm{t\#h}$;
- $\mathrm{s\#k} \geq_w \mathrm{t\#h}$ means that $\mathrm{s\#k}$ is either a hypernym or a holonym of $\mathrm{t\#h}$;
- $\mathrm{s\#k} \equiv_w \mathrm{t\#h}$ means that $\mathrm{s\#k}$ and $\mathrm{t\#h}$ are synonyms.

Definition 3 (Sense elimination rules). Given a concept c and a list of associated senses $sense(c) = \{\mathrm{c\#1}, \ldots \mathrm{c\#n}\}$, the sense $\mathrm{c\#i}$ can be removed from the list by applying one of the following two rules:

R1 the following two conditions holds:

1. for some sense $\mathrm{c\#j}$ there is an ancestor c' of c and a sense $\mathrm{c'\#k}$, such that $\mathrm{c\#j} \geq_w \mathrm{c'\#k}$
2. there is no ancestor c' of c and no $\mathrm{c'\#k}$, such that $\mathrm{c\#i} \geq_w \mathrm{c'\#k}$.

R2 the following two conditions holds:

1. for some sense c#j there is a descendant c' of c and a sense c'#k, such that c#j $\leq_w$ c'#k
2. there is no descendant c' of c and no c'#k, such that c#i $\leq_w$ c'#k.

R3 there is a parent c' of c such that for all c'#j, c'#j$-_w$c#i.

Sense composition. Sense composition deals with the horizontal composition of senses, and it is based on the assumption that taxonomic relations within a context are inclusion relations interpreted over a domain of documents. Branches of a node N are *partitions* over the set of documents which can be classified under N. This means that the sets of documents classified under the children of N are disjoint.

For instance, in the case of "Europe" and "Italy" in Table 4, there are two different interpretations: from the point of view of the context structure the two concepts are disjoint (since they are branches of the same node); on the other hand, from the point of view of world knowledge, Italy#1 is part of Europe#1. This implies that the node labeled with "Europe" is intended to be "Europe except Italy".

The algorithm therefore checks the consistency between the world knowledge about a concept and the structural information coming from its context. When there is consistency the algorithm directly goes to the following step. On the other hand, if a concept within the path has a part relation or an inclusion relation with another concept on the same level, it is necessary to contextualize the meaning of those concepts by combining the two information sources.

By sense composition we incorporate these relations between sibling nodes and, for instance, the structure represented in Table 4 is modified as shown in Table 5.

Sea holidays

ID	0	1
Token	Sea	holidays
Lemma	sea	holiday
PoS	N	N
Function	mod-1	head
W-senses	sea#1 sea#2 sea#3	holiday#1 holiday#2

Italy

ID	0
Token	Italy
Lemma	Italy
PoS	N
Function	head
W-senses	Italy#1

in Europe

ID	0	1
Token	in	Europe
Lemma	in	Europe
PoS	P	N
Function	func-w	head
W-senses		Europe#1 – Italy#1

Table 5: Output of the sense compositions phase

The general rule for sense composition is described as follows:

Definition 4 (Sense composition rule). Let c and c' be two concepts, and let $\mathrm{c\#i}$ and $\mathrm{c'\#j}$ be two senses of c and c' respectively. We apply the following rule:

R4 replace the sense $\mathrm{c\#i}$ with $\mathrm{c\#i - c'\#j}$, if either $\mathrm{c'\#j} \leq_w \mathrm{c\#i}$ or $\mathrm{c\#i} \geq_w \mathrm{c\#j}$.

The result of this phase is a refinement of the result obtained in the previous step. The data structure associated with each node $c \in C_s \cup C_t$ is called the *refined context free meaning of* c.

4 Matching

The matching algorithm has the following three main steps:

1. Compute the focus $F(c_s)$ and $F(c_t)$ of the concepts c_s and c_t: Intuitively the focus $F(c)$ is the subset of a context C which are relevant for the meaning of c.
2. Refine the senses: A further step of sense refinement can be performed by considering the fact that we are focusing to a certain concept.
3. Compute the matching: We compute the matching of c_s and c_t by comparing their refined senses (in a matching matrix).

The input is a 4-tuple $\langle Ctx_s, Ctx_t, c_s, c_t \rangle$ containing the following elements:

1. a source concept hierarchy $H_s = \langle C_s, E_s \rangle$;
2. a target concept hierarchy $H_t = \langle C_t, E_t \rangle$;
3. a source concept $c_s \in C_s$;
4. a target concept $c_t \in C_t$.

The output is a relation between c_s and c_t, i.e. an expression of the form:

$$c_s \xrightarrow{\subseteq} c_t, c_s \xrightarrow{\supseteq} c_t, c_s \xrightarrow{\perp} c_t, c_s \xrightarrow{\equiv} c_t, c_s \xrightarrow{*} c_t$$

4.1 Concept focus

The *focus* $F(c)$ of a concept c in a context C is defined as follows:

Definition 5 (Focus). Given a context hierarchy H and a concept c of H, the *focus* of c is a subgraph $F(c) \subseteq H$ such that each element $f \in F(c)$ is either an ancestor of c or the child of an ancestor of c.

The focus considers only the hierarchical relations and forgets any general relations between the concepts. The focus of the node labeled with “in Europe’ and “Spain” of the concept hierarchies shown in Figure 2 is depicted in Figure 3.

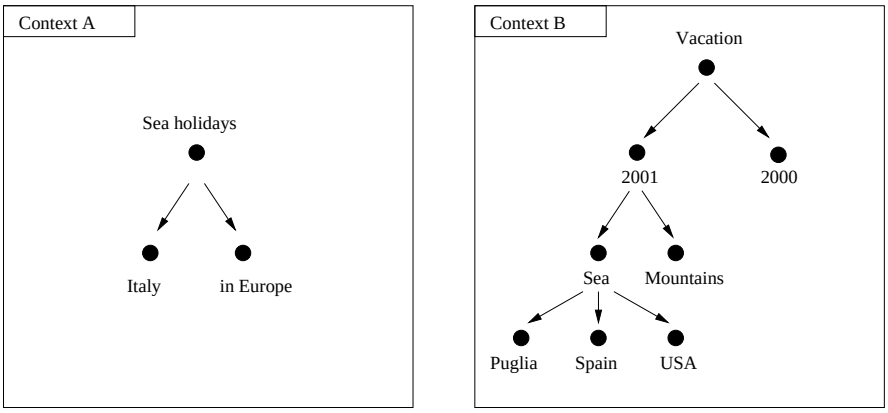


Figure 3: The focuses of ”in Europe” and ”Spain”.

sea#1 sea#2 sea#3	holiday#1 holiday#2	Europe#1	−Italy#1

Table 6: Contextual meaning of the concept labeled with “in Europe”

4.2 Sense refinement in focus

The main objective of this step is to refine the analysis performed during the pre-processing phase described previously. This is necessary to further disambiguate senses which cannot be disambiguated in the whole context. Consider for instance, a context hierarchy where a node labeled with “tree” has two branches, “apple” and “binary”. The sense of the label “tree” cannot be disambiguated if we consider the whole context. Indeed “tree” might refer both to a tall perennial woody plant, and a diagram. However, if we focus on, e.g. “apple”, “tree” can be disambiguated.

4.3 Semantic Matching

In this phase we compute the semantic relation between the concepts c_s and c_t, starting from their meaning in the source and target contexts Let us first define the *contextual meaning* of a concept.

Definition 6 (Contextual meaning). The *contextual meaning* of a concept c is the ordered list of the refined context free meanings of the nodes contained in the branch from the root to c.

The contextual meaning of the concept labeled with “in Europe”, and the contextual meaning of the concept labeled with “Spain” are shown in Table 6 and 7 respectively.

4.4 Matching matrix

The contextual meaning of c_s and c_t is used to fill in a so-called *matching matrix*, which has the components of the contextual meaning of c_s on the lines, and the components of the contextual meaning of c_t on the columns. Each filtered context free meaning of the nodes

vacation#1 vacation#2	2001	Spain#1

Table 7: Contextual meaning of the concept labeled with “Spain”

	holiday#1 holiday#2	sea#1 sea#2 sea#3	Europe#1	−Italy#1
vacation#1 vacation#2				
2001				
sea#1 sea#2 sea#3				
Spain#1				

Table 8: Matching matrix for the concepts “in Europe” and “Spain”

in the path of a concept goes in the matrix, with the only exception being function-words (e.g. articles, prepositions, etc.). As far as the order is concerned, the root of the path is in the first place, followed by its child, the child of its child, etc. If the meaning of a concept is represented by two or more concepts (i.e. the head word and one or more modifiers), the modifiers will follow the head word.

An example of matching matrix for the concepts “in Europe” and “Spain” is given in Table 8.

	holiday#1	sea#1	Europe#1	−Italy#1
vacation#1	$\equiv$			
2001				
sea#1		$\equiv$		
Spain#1			$\subseteq$	$-$

Table 9: One of the possible matrices resulting from the matrix represented in Table 8

4.5 Filling the matching matrix

Let $s = \{\mathrm{s\#1}, \ldots, \mathrm{s\#n}\}$ $t = \{\mathrm{t\#1}, \ldots, \mathrm{t\#m}\}$, be the senses associated to the s row and the t column of a matching matrix M, labeled with their filtered context free senses. The values of $M(s,t)$ is determined by applying one of the following rules to any pair of $\mathrm{s\#k}$ and $\mathrm{t\#h}$. In this step, we ignore if a sense $\mathrm{s\#k}$ or $\mathrm{t\#h}$ occurs with a “$-$” sign in front.

M0 If $\mathrm{t\#k} =_w \mathrm{s\#h}$, then remove all the senses different from $\mathrm{s\#k}$ from s and $\mathrm{t\#h}$ from t, and set $M(s,t) =$ “ $\equiv$ ”.

	A_1	A_2	...	A_n
B_1	M_{11}	M_{12}	...	M_{1n}
B_2	M_{21}	M_{22}	...	M_{2n}
$\vdots$				
B_m	M_{m1}	M_{m2}	...	M_{mn}

Table 10: Matching matrix M

M1 If $\mathrm{t}\#\mathrm{k} \leq_w \mathrm{s}\#\mathrm{h}$, then remove all the senses different from $\mathrm{s}\#\mathrm{k}$ from s and $\mathrm{t}\#\mathrm{h}$ from t, and set $M(s,t) =$ " $\subseteq$ ".

M2 If $\mathrm{t}\#\mathrm{k} \geq_w \mathrm{s}\#\mathrm{h}$, then remove all the senses different from $\mathrm{s}\#\mathrm{k}$ from s and $\mathrm{t}\#\mathrm{h}$ from t, and set $M(s,t) =$ " $\supseteq$ ".

M3 If $\mathrm{t}\#\mathrm{k} -_w \mathrm{s}\#\mathrm{h}$, then remove all the senses different from $\mathrm{s}\#\mathrm{k}$ from s and $\mathrm{t}\#\mathrm{h}$ from t, and set $M(s,t) =$ " $-$ ".

Notice that more than one rule (to more than one sense) can be applied at the same time. The choice of one rule w.r.t. another is a question of heuristics. If a choice is made which does not lead to a satisfactory result, backtracking should be possible. For instance, one of the three possible results of "sea" in the matching matrix in Table 8, depending on the choice of the sense, is represented in Table 9 (the other two are the same but with the other senses of "sea").

4.6 Computing the matching via Sat

To compute the final result we reason as follows. Let M be the matching matrix in Table 10, where M_{ij} is either empty or an element of the set $\{-, \equiv, \subseteq, \supseteq\}$.

Interpreting each A_i and B_i as a set (of documents), we have that the set of documents that are classifiable under the node with contextual meaning $A_1, A_2, \ldots A_n$ is the set

$$A'_1 \cap A'_2 \cap \ldots \cap A'_n$$

where $A'_k = \neg A_k$, if A_k in the matching matrix is prefixed by the "$-$" symbol, and A_k, otherwise. The same reasoning can be done for the B_i's. We have therefore to find the *best* set-theoretical relation between

$$A'_1 \cap A'_2 \cap \ldots \cap A'_n \text{ and } B'_1 \cap B'_2 \cap \ldots \cap B'_m$$

starting from the relations M_{ij} between A_i and B_j represented in M. The *best* is expressed with respect to the partial order that states that " $\equiv$ " and "$-$" are better than " $\subseteq$ " and " $\supseteq$ ".

The computation of the matrix can be rephrased in terms of a satisfiability problem.

For each non empty M_{ij} generate the following propositional formula:

$$\begin{array}{rcl} B_i \subseteq A_j & \Longrightarrow & B_i \rightarrow A_j \\ B_i \supseteq A_j & \Longrightarrow & A_j \rightarrow B_i \\ B_i \equiv A_j & \Longrightarrow & A_j \equiv B_i \\ B_i - A_i & \Longrightarrow & \neg(B_i \wedge A_j) \end{array}$$

Add to the set of clauses generated as above, the clause

$$\neg(A'_1 \wedge A'_2 \wedge \ldots \wedge A'_n \equiv B'_1 \wedge B'_2 \wedge \ldots \wedge B'_m)$$

and check for satisfiability. If the check fails, this means that

$$A'_1 \cap A'_2 \cap \ldots \cap A'_n \equiv B'_1 \cap B'_2 \cap \ldots \cap B'_m$$

A similar procedure can be followed for $-$ and $\subseteq$ and $\supseteq$.

5 Conclusions

We have presented an automatic algorithm of meaning negotiation that enables semantic interoperability between local overlapping and heterogeneous ontologies of different autonomous communities.

References

[1] M. Benerecetti, P. Bouquet, and C. Ghidini. Contextual Reasoning Distilled. *Journal of Theoretical and Experimental Artificial Intelligence*, 12(3):279–305, 2000.

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

[3] A. Borgida and L. Serafini. Distributed description logics. In Ian Horrocks and S. Tessaris, editors, *Proceedings of the 2002 Intl. Workshop on Description Logics (DL2002)*, Toulouse, April 2002. CEUR-WS. ONLINE: http://CEUR-WS.org/Vol-53/, ARCHIVE: ftp://SunSITE.Informatik.RWTH-Aachen.DE/pub/publications/CEUR-WS/Vol-53.tar.gz.

[4] 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.

[5] P.F. Drucker. *Post Capitalist Society*. Cambridge University Press, 1994.

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

[7] F. Giunchiglia. Contextual reasoning. *Epistemologia, special issue on I Linguaggi e le Macchine*, 16:279–305, 1993.

[8] G. Miller. An on-line lexical database. *International Journal of Lexicography*, 13(4), 1990.

[9] I. Nonaka and H. Takuechi. *The Knowledge Creating Company*. Oxford University Press, 1990.