

Enhancing Communication inside Multi-Agent Systems^{*}

An Approach based on Alignment via Upper Ontologies

Viviana Mascardi[†], Paolo Rosso[‡], and Valentina Cordi[†]

[†]DISI, Università degli Studi di Genova, Via Dodecaneso 35, 16146, Genova, Italy

E-mail: mascardi@disi.unige.it, valentina.cordi@gmail.com

[‡]DSIC, Universidad Politécnic de Valencia, Camino de Vera s/n, 46022, Valencia Spain

E-mail: proso@dsic.upv.es

Abstract. This paper deals with a theoretical issue related to multi-agent system development and deployment, namely the need of a mechanism for aligning ontologies owned by agents, in order to allow them to communicate in a profitable way. Our approach exploits upper ontologies, i.e., ontologies which describe very general concepts that are the same across all domains, as a “lingua franca” among agents. This approach may overcome some problems that arise in various real scenarios, such as the impossibility for (or the lack of will of) an agent to disclose its own entire ontology to another agent, despite the need to communicate with it. In this paper we propose a comparison of seven existing upper ontologies, and an algorithm for aligning any two (or more) ontologies by exploiting an upper ontology as a bridge.

1 Introduction

In a paper that dates back to 2001, James A. Hendler predicted that

in the next few years virtually every company, university, government agency or ad hoc interest group will want their web resources linked to ontological content - because of the many powerful tools that will be available for using it. [...] On top of this infrastructure, agent-based computing [...] will be a primary means of computation in the not-so-distant future. [9]

Hendler’s vision has found a partial realisation: ontologies, web services, and the combination of both, i.e., semantic web services, are more and more exploited to share knowledge within and outside the boundaries of companies and other organisations. Intelligent software agents are recognised by both researchers and practitioners from the industry as one of the most suitable means for mediating among the heterogeneity of applications working within open, distributed, concurrent systems, and for this reason find application in many commercial projects [14]. Proposals of integrating intelligent agents, web services, and ontologies, thus realising “agent-based semantic web services”, are already around [6,20,11]. However, although it is probably true that almost every company, university, government agency *would want* their web resources linked

^{*} This work was partially supported by the research projects TIN2006-15265-C06-04 and “Iniziativa Software” CINI-FINMECCANICA.

to ontological content, and made available by exploiting an infrastructure based on software agents, it is also true that only a small subset of them *has* already implemented this vision.

One of the reasons for this delay with respect to Hendler’s predictions, is that linking the organisation’s web resources, that in most cases will contain knowledge represented by some domain dependent, ad-hoc ontology O_{Org} , to some reference ontological content O_{Ref} , requires to find mappings between the concepts of O_{Org} and O_{Ref} . Without an automatic, agent-driven means for finding these mappings, linking web resources to ontological contents becomes a very difficult and time-consuming activity, not only because of the time and human resources needed for finding them for the first time, but also for maintaining them, as the organisation’s ontology O_{Org} will evolve during time, and this will require continuous updates to the mappings with O_{Ref} .

Finding formal statements that assert the semantic relation between two entities belonging to different ontologies (namely, finding an *alignment* of the two ontologies, [4,2]) is thus a key problem in many scenarios such as enterprise information integration, querying and indexing the deep web, merchant catalog mapping, etc. [8]. In all of these scenarios, one organisation wants to align its own entire ontology O with the entire ontology O' of another organisation. This is the most classical instance of the alignment problem, that requires that both ontologies O and O' are given in input to the alignment algorithm, and that are entirely available. The two organisations whose ontologies are involved in the alignment process, must have neither restrictions due to privacy issues, nor restrictions due to limited space or time resources that prevent them from sharing their whole ontology. There are also scenarios, however, where two organisations might want to perform an alignment of their ontologies in order to be able to interact, but either limiting the reciprocal disclosure of the ontologies to the minimum required for understanding each other, or disclosing portions of the ontologies in an incremental way. These scenarios are almost common inside multi-agent systems. In this paper, we face a theoretical issue strongly relevant for developing and deploying multi-agent systems (MASs) in the above scenarios: that of finding and automatic and incremental alignment of the agents’ ontologies by exploiting “Upper Ontologies”, namely ontologies which describes very general concepts that are the same across all domains.

The paper is organised in the following way: Section 2 describes two scenarios that motivate our approach, Section 3 provides some background on upper ontologies and ontology alignment, Section 4 discusses the algorithm that we propose, and Section 5 draws some conclusions and outlines the future directions of our work.

2 Motivation

In many applicative scenarios, the alignment of two ontologies may be performed directly, namely without any other ontology serving as a bridge, and offline, namely with both ontologies entirely known in advance. We will refer to this scenario as the “*classical*” one. However, there are situations where the assumptions made in the “*classical*” scenario, do not hold. Before considering two of these situations, we observe that ontologies may be developed and owned by a single human developer, by an institution

like a company, or by a software agent equipped with knowledge and algorithms suitable for managing an ontology in a (semi-)automatic way. In this paper, we attribute the capability of developing and possessing an ontology to human beings, software agents, and institutions indifferently. Also, coherently with the “strong definition” of agents [22], we attribute intentions, goals, and beliefs to them.

Personalised content provider. In this scenario, a provider of personalised content (news, commercial advertisements, etc.) is a software agent that provides content to the agents that access it, be them human or software, in a personalised way. The most accepted personalisation approach is the one where the user’s interests and habits are learnt just by watching his/her behaviour, without any explicit request apart from request on feedback on the provided content. The MAS composed by the personalised content provider agent (PCPA) and its users is highly open and dynamic.

The PCPA is strongly motivated to make a good job because it usually has economic advantages in increasing the number of users that access its services. Also, it is motivated to make its users faithful to it, since a user that accesses the service on a regular basis can be better profiled and the PCPA can deliver most appropriate contents to him/her. The user will then be more satisfied with the offered service, and will tend to go on using it. As far as ontologies are concerned, it is very likely that the PCPA possesses a private ontology O_{PCPA} for categorising the contents that it manages and delivers.

On the other hand, the user does not want to spend time in training the PCPA, and s/he probably does not even want to reveal extra information to it, besides that concerning his/her interests which can be extracted from the queries s/he poses to the PCPA. The user may or may not have an explicit ontology defining the concepts that appear in the queries made to the PCPA. In case s/he (or it, if we consider software users acting on behalf of humans) has an ontology, s/he might not want to share it with the PCPA.

- **Alignment problem.** The alignment problem in this scenario is asymmetric: the PCPA would like to know all the ontologies that its users use, and align them with its own in order to provide the best answers to the users’ queries. Instead, most of the users either do not have an explicit ontology (this means that they *do* have a reference ontology, but it is “hard-wired” inside their knowledge, or inside their code, or inside their brain), or they have it but are not interested in sharing it with the PCPA.
- **Proposed solution: incremental alignment.** What the PCPA may do to meet its objectives, is to consider the concepts that appear in queries issued by $User$ as a subset of the concepts of $User$ ’s ontology, O_{User} . The PCPA can make a partial alignment of the sub-ontology composed only by those concepts (without any relation) and O_{PCPA} , use this partial alignment to provide answers, and refine it as soon as new queries from $User$ (and thus, new concepts of O_{User}) arrive. In other words, the PCPA may perform a *partial, incremental, on-line alignment* between an (unknown and undisclosed) ontology O_{User} and O_{PCPA} . The alignment is partial because only a portion of O_{User} can be elicited from the users’ queries. It is incremental, because as new portions of O_{User} arrive, the alignment is updated and

enriched with new information. Finally, it is an on-line process, made of successive refinements performed at any interaction between *PCPA* and *User*.

Virtual Enterprises. A virtual enterprise (VE) is a temporary consortium of autonomous, diverse and possibly geographically dispersed organizations that pool their resources to meet short-term objectives and exploit fast-changing market trends [3].

Due to the autonomy, distribution and heterogeneity of the enterprises belonging to a VE, it can be suitably conceptualised as a MAS. This MAS is neither open nor dynamic since the enterprises belonging to it are known in advance. It is reasonable to assume that, at least within the information and communication technologies domain, most or all of them possess a private ontology and are strongly motivated in sharing and aligning it with the ontologies of the other enterprises in the VE in order to improve the benefits of the VE as a whole.

This scenario would be the right one for adopting a classical approach to alignment, if there were no privacy restrictions on the ontologies of each individual enterprise. Unfortunately, this is not often the case. What happens in the real case, is that the enterprises want and need to exchange useful information among them, but they also want and need to protect (part of) their enterprise information, represented by the enterprise ontology. In fact, the VE is just a temporary consortium where each component also runs its own business, often in concurrence with other enterprises within the same VE.

The disclosure of the entire ontology to all the partners belonging to the VE, has economic and commercial advantages for the VE, but may be a serious disadvantage for the single enterprise.

- **Alignment problem.** The alignment problem in this scenario is symmetric: all the enterprises would like to have the means for interacting with each other in the best possible way, but they also would like to avoid disclosing their own ontologies.
- **Proposed solution: alignment via upper ontologies.** The solution to this instance of the alignment problem may come from the use of upper ontologies: each enterprise may align its own ontology with an upper ontology upon which all the enterprises agree. Then, each enterprise becomes able to communicate with any other enterprise by means of the upper ontology. This gives two advantages to both the individual enterprises and the VE:
 1. no enterprise has to disclose its own ontology: it may only disclose the corresponding portion of the upper ontology obtained by alignment;
 2. if there are n enterprises within the VE, the alignments required to allow any enterprise to interact with any other are only n (one alignment between each private ontology and the upper ontology), instead of the n^2 that would be required if any enterprise had to align its own ontology with all the other ones.

The analysis of the state of the art discussed in Section 3 shows that there are very few proposals of performing an incremental alignment of ontologies, and no implemented systems that exploit upper ontologies in the alignment process. However, the two motivating scenarios that we have identified, demonstrate that both incremental and upper ontology-based approaches would be extremely useful in many agent systems.

3 Background

In this section we introduce upper ontologies and a systematic comparison of seven of them, and we discuss the state-of-the-art of alignment techniques. To the best of our knowledge, there are no comparisons that consider all the upper ontologies discussed in Section 3.1. Thus, this section represents an original contribution of our work. For space constraints, we only report the synthesis of our comparison in form of tables. More information can be found in [13].

3.1 Upper Ontologies

Upper ontologies are quickly becoming a key technology for integrating heterogeneous knowledge coming from different sources. In fact, they may be used by different parties involved in a knowledge integration and exchange process as a reference, common model of the reality.

The definition of upper ontology (also named top-level ontology, or foundation ontology) given by Wikipedia [21] is “*an attempt to create an ontology which describes very general concepts that are the same across all domains. The aim is to have a large number on ontologies accessible under this upper ontology*”.

In this section, we review the state-of-the-art in the field of upper ontologies by comparing seven of them based on dimension, implementation language(s), modularity, alignment with the WordNet lexical resource, and licensing. These software engineering criteria may prove useful for the developer of a knowledge-based system that has to choose the most suitable upper ontology for his/her needs, among a set of existing ones. The choice of the upper ontologies we describe and compare, namely BFO, Cyc, DOLCE, GFO, PROTON, Sowa’s ontology, and SUMO, is based on how much they are visible and used inside the research community. In fact we have discussed all the upper ontologies referenced by Wikipedia, apart from WordNet that we consider a lexical resource rather than an upper ontology, and from the Global Justice XML Data and National Information Exchange Models, that address the specific application domain of justice and public safety. Moreover, we have added PROTON and Sowa’s ontology to those considered by Wikipedia.

The methodology followed to draw our comparison consisted in checking the existing literature, producing a first draft of the comparison based on the retrieved literature, submitting it to the attention of the developers of all the seven upper ontologies under comparison, and integrating the obtained answers and suggestions.

Table 1 provides a short description of the seven upper ontologies, while Tables 2 and 3 provide an handy way for comparing them from a software engineering viewpoint. Few other comparisons among upper ontologies exist, and they just consider subsets of the upper ontologies that we have treated in this section. Most of these existing comparisons, such as Pease’s comparison of DOLCE and SUMO [16,17], Onto-Med’s comparison of GFO, DOLCE, and Sowa’s ontology [10], and Grenon’s comparison of DOLCE and BFO [7], take a philosophical perspective, and thus complement our work that is much more application-oriented. MITRE’s comparison of SUMO, Upper Cyc, and DOLCE [18], compares the three upper ontologies according to a subset of our criteria.

BFO	BFO (http://www.ifomis.org/bfo) consists in two sub-ontologies: SNAP – a series of snapshot ontologies (O_{ti}), indexed by times – and SPAN – a single videoscopic ontology (O_v). An O_{ti} is an inventory of all entities existing at a time, while an O_v is an inventory of all processes unfolding through time. Both types of ontology serve as basis for a series of sub-ontologies, each of which can be conceived as a window on a certain portion of reality at a given level of granularity. It finds application mainly in the biomedical domain.
Cyc	The Cyc Knowledge Base KB (http://www.cyc.com/) is a formalised representation of facts, rules of thumb, and heuristics for reasoning about the objects and events of everyday life. The KB consists of terms and assertions which relate those terms. These assertions include both simple ground assertions and rules. The Cyc KB is divided into thousands of “microtheories” focused on a particular domain of knowledge, a particular level of detail, a particular interval in time, etc. It finds application in natural language processing, network risk assessment, terrorism management.
DOLCE	DOLCE (http://www.loa-cnr.it/DOLCE.html) captures the ontological categories underlying natural language and human commonsense. According to DOLCE, different entities can be co-located in the same space-time. DOLCE is an “ontology of particulars”, i.e. an ontology of instances, rather than an ontology of universals or properties. DOLCE-Lite+ (http://wiki.loa-cnr.it/index.php/LoaWiki:Ontologies#Modules_of_the_DOLite.2B_Library) encodes the basic DOLCE ontology into OWL-DL and adds eight pluggable modules, including collections, social objects, plans, spatial and temporal relations, to it. DOLCE is used for multilingual information retrieval, web-based systems and services, e-learning.
GFO	GFO (http://www.onto-med.de/ontologies/gfo.html) includes elaborations of categories like objects, processes, time and space, properties, relations, roles, functions, facts, and situations. Work is in progress on an integration with the notion of levels of reality in order to more appropriately capture entities in the material, mental, and social areas. It is used in the biomedical domain.
PROTON	PROTON (PROTo ONtology, http://proton.semanticweb.org/) is a basic upper-level ontology providing coverage of the general concepts necessary for a wide range of tasks. The design principles are (i) domain-independence; (ii) light-weight logical definitions; (iii) consistence with popular standards; (iv) good coverage of named entities and concrete domains (i.e., people, organizations, locations, numbers, dates, addresses). It is used for semantic annotation, knowledge management systems in legal and telecomm. domains, business data ontology for semantic web services.
Sowa’s	Sowa’s ontology (http://www.jfsowa.com/ontology/) is based on [19]. The basic categories and distinctions have been derived from a variety of sources in logic, linguistics, philosophy, and artificial intelligence. Sowa’s ontology is not based on a fixed hierarchy of categories, but on a framework of distinctions, from which the hierarchy is generated automatically. For any particular application, the categories are not defined by selecting an appropriate set of distinctions. No documented applications have been developed, but Sowa’s ontology inspired the creation of many implemented upper ontologies.
SUMO	SUMO (http://www.ontologyportal.org/) and its domain ontologies [15] form one of the largest formal public ontology in existence today. SUMO is extended with many domain ontologies and a complete set of links to WordNet, and is freely available. It finds application in linguistics, knowledge representation, reasoning.

Table 1. Short description of the upper ontologies

	Developers	Dimensions	Language(s)
BFO	Smith, Grenon, Stenzhorn, Spear (IFOMIS)	36 classes related via the is_a relation	OWL
Cyc	Cycorp	About 300,000 concepts, 3,000,000 facts and rules, 15,000 relations (including microtheories)	CycL, OWL
DOLCE	Guarino and other researchers of the LOA	About 100 concepts and 100 axioms	First Order Logic, KIF, OWL
GFO	The Onto-Med Research Group	79 classes, 97 subclass-relations, 67 properties	First Order Logic and KIF (forthcoming); OWL
PROTON	Ontotext Lab, Sirma	300 concepts and 100 properties	OWL Lite
Sowa's	Sowa	30 classes, 5 relationships, 30 axioms	First Order Modal Language, KIF
SUMO	Niles, Pease, and Menzel	20,000 terms and 60,000 axioms (including domain ontologies)	SUO-KIF, OWL

Table 2. Comparison, Part I

	Modularity	Alignment with WordNet	Licensing
BFO	SNAP and SPAN modules	Not supported	Freely available
Cyc	"Microtheory" modules	Mapped to about 12,000 WordNet synsets	Commercial product; ResearchCyc and OpenCyc are freely available but are more limited than the commercial version
DOLCE	DOLCE is not divided into modules, while DOLCE-Lite+ is	Aligned with about 100 WordNet synsets	Freely available
GFO	Abstract top level, abstract core level, basic level	Not supported	Released under the modified BSD Licence
PROTON	Three levels including four modules	Not supported	Freely available
Sowa's	Not divided into modules	Not supported	Freely available
SUMO	Divided into SUMO itself, MILO, and domain ontologies	Mapped to all of WordNet v2.1 by hand	Freely available

Table 3. Comparison, Part II

3.2 Ontology Alignment

In [2], an alignment is described as

“a set of mappings expressing the correspondence between two entities of different ontologies through their relation and a trust assessment. The relation can be equivalence as well as specialisation/generalisation or any other kind of relation. The trust assessment can be boolean as well as given by other measures (e.g., probabilistic or symbolic measures)”.

Intuitively, a mapping can be described as a 5-tuple $\langle id, e, e', n, R \rangle$ where:

- id is a unique identifier of the given mapping element;
- e and e' are the entities (e.g. tables, XML elements, properties, classes) of the first and the second ontology respectively;
- n is a confidence measure (typically in the [0,1] range) holding for the correspondence between the entities e and e' ;
- R is a relation such as equivalence, more general, disjointness, overlapping, holding between the entities e and e' .

In [5], ontology alignment approaches are classified into:

Local Methods — The main issue in aligning consists of finding to which entity or expression in one ontology corresponds another one in the other ontology. Local methods are the basic methods which enable to measure this correspondence at a local level, i.e., only comparing one element with another and not working at the global scale of ontologies. Very often, this amounts to measuring a pair-wise similarity between entities (which can be as reduced as an equality predicate) and computing the best match between them. Local methods exploit the definitions of similarity and of distance. In [5] such definitions are provided, as well as a detailed classification of local methods.

Global Methods — Once the local methods for determining the similarity are available, the alignment must be computed. This involves some kind of more global treatments, including:

- aggregating the results of local methods in order to compute the similarity between compound entities;
- developing a strategy for computing these similarities in spite of cycles and non linearity in the constraints governing similarities;
- organising the combination of various similarity algorithms;
- involving the user in the loop;
- finally extracting the alignments from the resulting similarity: indeed, different alignments with different characteristics can be extracted from the same similarity.

Since global methods are based upon local ones, in the following paragraph we briefly discuss local methods.

Local Methods.

Definition 1. (Similarity). A similarity $\sigma : O \times O \rightarrow \mathbb{R}$ is a function from a pair of entities to a real number expressing the similarity between two objects such that:

$$\begin{aligned}\forall x, y \in O, \sigma(x, y) &\geq 0 \quad (\text{positiveness}) \\ \forall x \in O, \forall y, z \in O, \sigma(x, x) &\geq \sigma(y, z) \quad (\text{maximality}) \\ \forall x, y \in O, \sigma(x, y) &= \sigma(y, x) \quad (\text{symmetry})\end{aligned}$$

The dissimilarity is the dual operation of the similarity.

Definition 2. (Distance). A distance (or metrics) $\delta : O \times O \rightarrow \mathbb{R}$ is a dissimilarity function satisfying the definiteness and triangular inequality:

$$\begin{aligned}\forall x, y \in O, \delta(x, y) &= 0 \text{ iff } x = y \quad (\text{definiteness}) \\ \forall x, y, z \in O, \delta(x, y) + \delta(y, z) &\geq \delta(x, z) \quad (\text{triangular inequality})\end{aligned}$$

A (dis)similarity is said to be normalised if it ranges over the unit interval of real numbers [0 1]. Local methods introduced in the following classification use normalised measures.

1. **Terminological methods** compare strings. They can be applied to the name, the label or the comments concerning entities in order to find those which are similar. Terminological methods are further divided into string-based methods, that take advantage of the structure of the string as a sequence of letter, and language-based methods, that rely on using natural language processing (NLP) techniques to find associations between instances of concepts or classes.
2. **Structural methods** compare the structure of the entities. This comparison may either be a comparison of the internal structure of an entity (i.e., its attributes) or a comparison of the entity with other entities to which it is related.
3. **Extensional methods** compare the extension of classes, i.e., the set of their instances rather than their interpretation.
4. **Semantic methods** have model-theoretic semantics which is used to justify their results, and thus they are deductive methods. Examples are propositional satisfiability (SAT) and modal SAT techniques or description logic based techniques.

While the literature discusses many local and global “off-line” alignment methods, very few attempts have been made to propose “incremental” alignment methods [1,12], and none exploits upper ontologies for this purpose. The next section discusses our algorithm for incremental alignment based on upper ontologies. Even if the algorithm has not been implemented yet, our approach may be a preliminary contribution to a research field that is still unexplored.

4 Algorithm

The algorithm that we propose is based on two functions, $Align(O_1, O_2)$, where O_1 and O_2 are two ontologies, and $Merge(Al_1, Al_2)$, where Al_1 and Al_2 are two alignments. Both functions return an alignment. The algorithm consists of three steps: the concepts of the two ontologies to align are first “tagged” with concepts of a reference upper ontology by exploiting the $Align$ function (first two steps), and the two alignments obtained in this way are merged. The two agents that want to align their ontologies may exploit only the merged alignment, thus avoiding the disclosure of their private ontologies, and this process may take place following incremental steps, for coping with all those situations where one of the ontologies is not fully known in advance. We consider alignment of ontologies based on the equivalence relation R , which is thus dropped from the 5-tuple that represents the mapping element.

$Align(O_1, O_2)$ just computes an alignment between O_1 and O_2 by exploiting one (or a combination of) standard local method(s) among those introduced in Section 3.2.

Given two alignments $Al_1 = Align(O_1, O_{Bridge})$ and $Al_2 = Align(O_2, O_{Bridge})$, $Merge(Al_1, Al_2)$ computes the alignment Al between O_1 and O_2 starting from Al_1 and Al_2 . A mapping element $\langle id, C_1, C_2, Conf \rangle$ belongs to Al iff $\exists C_{Bridge} \in O_{Bridge}$ such that $\langle id_1, C_1, C_{Bridge}, Conf_1 \rangle \in Al_1$, $\langle id_2, C_2, C_{Bridge}, Conf_2 \rangle \in Al_2$ and $Conf_1 * Conf_2 \geq Threshold$, where $Threshold$ is a configurable threshold.

Our algorithm consists of the following steps:

- Al(O_1, O_{Bridge})** Each concept $C_1 \in O_1$ is “tagged” with one or more concepts $\{C_{Bridge_1}, \dots, C_{Bridge_k}\} \in O_{Bridge}$ by exploiting traditional mapping techniques based on a combination of string comparison, natural language processing techniques, and exploitation of linguistic resources such as common knowledge or domain specific thesauri. Namely, an alignment $Al(O_1, O_{Bridge})$ between O_1 and O_{Bridge} is computed. The tagging is represented as a set of mapping elements $\{\langle id_1, C_1, C_{Bridge_1}, Conf_1 \rangle, \dots, \langle id_k, C_1, C_{Bridge_k}, Conf_k \rangle\}$.
- Al(O_2, O_{Bridge})** In the same way, an alignment $Al(O_2, O_{Bridge})$ between O_2 and O_{Bridge} is computed, resulting into a “tagging” of concepts of O_2 with concepts of O_{Bridge} .
- Merge(Al_1, Al_2)** The alignment between O_1 and O_2 , namely $Merge(O_1, O_2)$, is computed.

This algorithm may be used in three different ways, depending on the usage scenario and just skipping some of its steps:

Usage 1 - Classical way: This usage corresponds to the situation where two agents Ag_1 and Ag_2 agree to share their ontologies O_1 and O_2 , to give them in input to the alignment function, and to take advantage of the computed output. Once both agents know the computed alignment $Align(O_1, O_2)$, they may communicate either using O_1 as their reference ontology, or using O_2 , indifferently. In this case, only the first step of the algorithm is performed, with O_2 used instead of O_{Bridge} .

Usage 2 - Incremental way: This is the usage foreseen within the personalised content provider scenario. The PCPA Ag_1 computes $Al_1 = Align(O_1, O_{Upper})$, where

O_{Upper} is an upper ontology; when Ag_1 receives a message containing the concepts $\{C_{2_1}, \dots, C_{2_n}\} \in O_2$ from the user agent Ag_2 , it interprets these concepts as a sub-ontology $O_{2_{small}}$ of the implicit ontology O_2 used by Ag_2 . $O_{2_{small}}$ is a degenerated ontology, since it has only concepts with no relations among them. Although degenerated, this is the only ontology that the PCPA Ag_1 may elicit from the user agent Ag_2 . Ag_1 may then call $Al_2 = Align(O_{2_{small}}, O_{Upper}) \forall C_{2_i}$, and merge the obtained partial alignment Al_2 with Al_1 by calling the function $Merge(Al_1, Al_2)$. As new messages from the user agent Ag_2 arrive, containing new concepts of the ontology used by Ag_2 , the PCPA reiterates the computation of the alignment, and integrates the new alignment with the previously obtained one by making a union of the tuples belonging to the old and new alignments, just omitting those tuples that appear more than one time, with different identifier and confidence. In this case, the tuple with the best confidence may be kept, and the other one(s) may be skipped.

In this scenario, exploiting an upper ontology is very important for providing the right content to the user agent. In fact, the user agent might require something that is not included in the content provider's ontology. To make a trivial example, the user might look for content dealing with "marsupials", and the content provider might only possess the "mammal" concept in its ontology, with no sub-concepts. If the concepts in the user's message are directly aligned with those in the content provider's ontology, the mapping between marsupial and mammal is lost, since there is no way for the content provider to know what a marsupial is. Instead, if the reference upper ontology contains the information that a marsupial is a mammal (like SUMO does, <http://ontology.teknowledge.com/sumo-1.36classes.pdf>), a mapping between the marsupial concept belonging to the user agent's ontology O_2 and the mammal concept belonging to the PCPA's ontology O_1 , may be found with a confidence lower than 1 (the two concepts are not equivalent).

Usage 3 - Upper ontology way: This is the usage foreseen within the Virtual Enterprise scenario. All the agents belonging to the VE, agree to use one upper ontology O_{Upper} as their "lingua franca" for knowledge exchange. Each agent Ag having a private ontology O calls $Align(O, O_{Upper})$. O_{Upper} is then used as the reference ontology for exchanging information among agents, without requiring to any agent to disclose its own private ontology. In this case, the first two steps of the algorithm are repeated n times - instead of twice - where n is the number of agents in the VE, and the last step of the algorithm that merges two alignments for finding a third one, is omitted.

4.1 Complexity

We evaluate the algorithm complexity when used in the three different ways identified in the previous paragraph.

Assumption. Given a concept $C_1 \in O_1$, finding a mapping between C_1 and any concept in O_2 by exploiting techniques based on a combination of string comparison, natural language processing techniques, and exploitation of linguistic resources, is in $\mathcal{O}(V_2)$, where V_2 is the number of concepts of O_2 .

We obtain this result because we may consider that string comparison, NLP techniques and exploitation of a fixed-size thesaurus require the same amount of time $T_{string,NLP,thesaurus}$ whatever the two concepts to be mapped.

Usage 1 - Classical way: The time required for aligning O_1 with V_1 vertices and O_2 with V_2 vertices without exploiting an upper ontology is the time required by performing only the first step of the algorithm, namely $\mathcal{O}(V_1 * V_2)$.

Usage 2 - Incremental way: O_1 is the ontology owned by Ag_1 that wants to elicit O_2 owned by Ag_2 , and align it with O_1 . O_{Upper} is the reference upper ontology. The alignment between O_{Upper} and O_1 (first step of the algorithm) is done once and for all, and costs $\mathcal{O}(V_{Upper} * V_1)$. We do not count this time as part of the online incremental alignment process, since it is done offline, and only once.

Assuming that for any concept C_1 in O_1 there are at most k concepts in O_{Upper} that have a mapping with C_1 with a confidence greater than a given threshold, with k fixed, the concepts of O_{Upper} to which at least some concept in O_1 maps are at most in $k * V_1$, and identify a sub-ontology $O_{Upper}(O_1)$ of O_{Upper} . When j new concepts of O_2 arrive as part of a message from Ag_2 to Ag_1 , they may be aligned only with the concepts in $O_{Upper}(O_1)$ (second step of the algorithm). Since the concepts in $O_{Upper}(O_1)$ are $k * V_1$, aligning j concepts of O_2 with them requires $\mathcal{O}(j * V_1)$ (k is a constant, thus we drop it from the time complexity).

After this alignment has been computed, it must be merged with the one between O_{Upper} and O_1 . The merge step requires $\mathcal{O}(j * V_1)$.

To make a comparison with the “classical” approach, let us suppose that a message containing all the concepts of O_2 arrives from Ag_2 to Ag_1 . This will require that all the V_2 concepts of O_2 are aligned with those in O_1 . The complexity of finding this alignment is $\mathcal{O}(V_1 * V_2)$ (second step of the algorithm). The last step of the algorithm requires once again $\mathcal{O}(V_1 * V_2)$ resulting into a $\mathcal{O}(V_1 * V_2)$ overall time complexity. If we exclude the first step of the algorithm, the time required for aligning O_1 and O_2 by exploiting O_{Upper} as a bridge is the same required for aligning O_1 and O_2 without using O_{Upper} .

Upper ontology way: If V is the number of concepts of O , and V_{Upper} is the number of concepts of O_{Upper} , finding an alignment between O and $Upper$ requires $\mathcal{O}(V * V_{Upper})$.

In the Virtual Enterprise scenario, finding an alignment between any two ontologies O_1, \dots, O_n via O_{Upper} requires $\mathcal{O}(n * V_{max} * V_{Upper})$ where V_{max} is the number of concepts belonging to the largest ontology in the set $\{O_1, \dots, O_n\}$. In fact, the alignment step of the algorithm must be performed not only twice, but n times, and the merge step is not needed, since all virtual enterprises will communicate using concepts from O_{Upper} . We may use V_{max} as an upper bound for the number of concepts of O_1, \dots, O_n , and consider that any alignment between one ontology in this set, and O_{Upper} requires at most $\mathcal{O}(V_{max} * V_{Upper})$ time. Since the alignments to compute are n , we obtain the stated complexity result.

If we did not exploit O_{Upper} as a bridge among O_1, \dots, O_n , alignments between any two couples of ontologies O_1, \dots, O_n should be computed to allow virtual enterprises to communicate, and this would require to compute $n * (n - 1)$ alignments, instead of n .

The choice of using O_{Upper} in order to obtain a gain in complexity depends on the size of O_{Upper} . If the size of all the ontologies in O_1, \dots, O_n is comparable to that of O_{Upper} , let us name this size S , the complexity of the algorithm that uses O_{Upper} as a bridge is in $\mathcal{O}(n * S^2)$, while the complexity of the algorithm that does not use O_{Upper} as a bridge is in $\mathcal{O}(n^2 * S^2)$. If O_{Upper} is much larger than O_1, \dots, O_n , it might be better to compute the $n * (n - 1)$ small alignments between each couple of ontologies, instead of computing n very large alignments between any ontology and O_{Upper} .

In this second case, however, there is a real gain only if the system is closed and no new ontologies will be aligned in subsequent moments. In fact, if a new ontology O_{new} needs to be aligned, and no upper ontology O_{Upper} has been used, O_{new} must be aligned with n ontologies instead than with only one. In an open system, such an approach may become soon unacceptable.

4.2 Implementation

The implementation of the algorithm and its testing are under way. In this section we describe the approach that we are following; changes are still possible according to the experimental results that we will obtain and that will drive our future work.

We have chosen to exploit an API for ontology alignment developed by J. Euzenat, with contributions from other researchers.¹ The API provides services for

- storing, finding, and sharing alignments;
- piping alignment algorithms;
- manipulating alignments;
- comparing alignments.

The Alignment interface provides the following methods:

NameEqAlignment compares the equality of class and property names and aligns those objects with the same name;

EditDistNameAlignment uses an editing (or Levenshtein) distance between entity names. It builds a distance matrix and chooses the alignment based on that distance;

SubsDistNameAlignment computes a substring distance on the entity name;

StrucSubsDistNameAlignment computes a substring distance on the entity names and aggregates this distance with the symmetric difference of properties in classes.

The pseudo-code for the *align* and *merge* functions is given below. Ontologies O_1 and O_2 are loaded from two URIs (the *loadOntology* method is provided by Euzenat's API). *AlignmentMethod1 ... AlignmentMethodN* will be chosen among the alignment methods provided by the API and listed above, and *combine* will combine the results of the alignments performed with different methods, for example by pipelining them. The *merge* function implements the algorithm introduced in the beginning of Section 4, in an iterative way.

¹ <http://alignapi.gforge.inria.fr/>

```

public Alignment align (URI uri1 , URI uri2) {
    OWLOntology O1 = loadOntology (uri1);
    OWLOntology O2 = loadOntology (uri2);

    AlignmentProcess A1 = new AlignmentMethod1 (O1, O2);
    ...
    AlignmentProcess AN = new AlignmentMethodN (O1, O2);

    Alignment A = combine (A1, A2, ... , AN);

    return A;}

public Alignment merge (Alignment A, Alignment B){
    Alignment M = null;

    for each mapping m in A
        for each mapping n in B
            if (m.C2 is equal to n.C2)
                then
                    add (<id , m.C1 , n.C2 , m.Conf*n.Conf>) to M;

    return M;}

```

The output file would consist of XML structures like the one shown below.

```

<Alignment>
  <map>
    <Cell>
      <identifier
        rdf:datatype=' http://www.w3.org/2001/XMLSchema#integer '>
        Id </identifier>
      <entity1 rdf:resource='O1#Concept1' />
      <entity2 rdf:resource='O2#Concept2' />
      <measure
        rdf:datatype=' http://www.w3.org/2001/XMLSchema#float '>
        Conf </measure>
      </Cell>
    </map>
    ...
  </Alignment>

```

Each correspondence (`map`) is made of an integer identifier, two references to the aligned entities, and a confidence measure ([0,1]) in this correspondence.

5 Conclusions

The work proposed in this paper originates from the observation that, in many situations, agents need to “learn” new knowledge that must be understood and added to the already possessed knowledge on-the-fly, in order to communicate in a profitable way with other agents in an open, dynamic system. The approaches for coping with this integration need to approximate and optimize the newly acquired knowledge. In

fact, the new knowledge cannot be “precise” since it comes from heterogeneous agents with which agreement neither on the terms to be used in the communication, nor on their meaning, had been made before. The common knowledge shared among agents, that is often implicit even to the agents themselves (in case of human agents), must be constructed via interaction, in an incremental way.

Starting from these considerations, we have designed an algorithm for enhancing communication among heterogeneous agents via incremental ontology alignment and exploitation of upper ontologies. The agents that may benefit from implementing such an algorithm are “semantic web agents” equipped with an ontology and able to communicate via the net, be it an intranet, like in the Virtual Enterprise scenario, or the Internet, like in the Personalised Content Provider scenario. Both issues that characterise our approach, i.e., incremental alignment and use of upper ontologies, are original contributions, as well as the comparison among BFO, Cyc, DOLCE, GFO, PROTON, Sowa’s ontology, and SUMO, that we have drawn.

At the time of writing, implementation and testing of our algorithm are still under way, but we will soon receive feedback from the implementation results, and this will give us an important help in understanding under which conditions the exploitation of upper ontologies is feasible, and which upper ontologies are better for being used as a bridge in the alignment process. Our current work is entirely aimed at completing the implementation of the algorithm and systematically describing our experimental results. Afterwards, one extension of our approach that might be interesting to explore is whether the seven upper ontologies that we have described in our paper may be themselves aligned into one “upper-upper ontology”.

Acknowledgments

We want to acknowledge all the researchers that helped in drawing the comparison of Section 3.1 with their constructive comments and useful advices. In particular, many thanks go to J. Euzenat, A. Kiryakov, L. Lefkowitz, F. Loebe, A. Pease, J. Schoening, P. Shvaiko, and H. Stenzhorn. We also thank A. Locoro for her thoughtful advices.

References

1. P. A. Bernstein, S. Melnik, and J. E. Churchill. Incremental schema matching. In *VLDB’2006, 32nd International Conference on Very Large Data Bases, Proceedings*, pages 1167–1170. VLDB Endowment, 2006.
2. P. Bouquet, J. Euzenat, E. Franconi, L. Serafini, G. Stamou, and S. Tessaris. Specification of a common framework for characterizing alignment. Technical Report D2.2.1, NoE Knowledge Web project, 2004.
3. H. Davulcu, M. Kifer, L. R. Pokorny, C. R. Ramakrishnan, I. V. Ramakrishnan, and S. Dawson. Modeling and analysis of interactions in virtual enterprises. In *RIDE-VE’99, 9th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises, Proceedings*, pages 12–18. IEEE Computer Society, 1999.
4. J. Euzenat. An API for ontology alignment. In S. A. McIlraith, D. Plexousakis, and F. van Harmelen, editors, *International Semantic Web Conference, Proceedings*, volume 3298 of *Lecture Notes in Computer Science*, pages 698–712. Springer, 2004.

5. J. Euzenat, T. Le Bach, J. Barrasa, and P. Bouquet et al. State of the art on ontology alignment. Technical Report D2.2.3, NoE Knowledge Web project, 2005.
6. N. Gibbins, S. Harris, and N. Shadbolt. Agent-based semantic web services. In *WWW '03, 12th International Conference on World Wide Web, Proceedings*, pages 710–717. ACM Press, 2003.
7. P. Grenon. BFO in a nutshell: A bi-categorial axiomatization of BFO and comparison with DOLCE. Technical Report 06/2003, IFOMIS, University of Leipzig, 2003.
8. A. Y. Halevy. Why your data won't mix. *ACM Queue*, 3(8):50–58, 2005.
9. J. A. Hendler. Agents and the semantic web. *IEEE Intelligent Systems*, 16(2):30–37, 2001.
10. H. Herre, B. Heller, P. Burek, R. Hoehndorf, F. Loebe, and H. Michalek. General formal ontology (GFO) – part I: Basic principles. Technical Report 8, Onto-Med, University of Leipzig, Germany, 2006.
11. L. Kerschberg, M. Chowdhury, A. Damiano, H. Jeong, S. Mitchell, J. Si, and S. Smith. Knowledge sifter: Agent-based ontology-driven search over heterogeneous databases using semantic web services. In M. Bouzeghoub, C. A. Goble, V. Kashyap, and S. Spaccapietra, editors, *Semantics for Grid Databases, First International IFIP Conference on Semantics of a Networked World: ICSNW 2004, Revised Selected Papers*, volume 3226 of *Lecture Notes in Computer Science*, pages 278–295. Springer, 2004.
12. J-S. Lee and K-H. Lee. XML schema matching based on incremental ontology update. In X. Zhou, S. Y. W. Su, M. P. Papazoglou, M. E. Orlowska, and K. G. Jeffery, editors, *Web Information Systems - WISE 2004, 5th International Conference on Web Information Systems Engineering, Proceedings*, volume 3306 of *Lecture Notes in Computer Science*, pages 608–618. Springer, 2004.
13. V. Mascardi, V. Cordí and P. Rosso. A comparison of upper ontologies. Technical Report DISI-TR-06-21, University of Genoa, 2006.
14. S. Munroe, T. Miller, R. A. Belecheanu, M. Pechoucek, P. McBurney, and M. Luck. Crossing the agent technology chasm: Lessons, experiences and challenges in commercial applications of agents. *The Knowledge Engineering Review*, 21(4):345 – 392, 2006. Cambridge University Press.
15. I. Niles and A. Pease. Towards a standard upper ontology. In C. Welty and B. Smith, editors, *FOIS 2001, 2nd International Conference on Formal Ontology in Information Systems, Proceedings*, pages 2–9. ACM Press, 2001.
16. A. Pease. Formal representation of concepts: The Suggested Upper Merged Ontology and its use in linguistics. In A. C. Schalley and D. Zaefferer, editors, *Ontolinguistics. How Ontological Status Shapes the Linguistic Coding of Concepts*. Mouton de Gruyter, 2006.
17. A. Pease and C. Fellbaum. Formal ontology as interlingua: The SUMO and WordNet linking project and GlobalWordNet. To appear.
18. S. K. Semy, M. K. Pulvermacher, and L. J. Obrst. Toward the use of an upper ontology for U.S. government and U.S. military domains: An evaluation. Technical Report MTR 04B0000063, The MITRE Corporation, 2004.
19. J. F. Sowa. In *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing, 1999.
20. K. P. Sycara, M. Paolucci, J. Soudry, and N. Srinivasan. Dynamic discovery and coordination of agent-based semantic web services. *IEEE Internet Computing*, 8(3):66–73, 2004.
21. Wikipedia. Upper ontology – Wikipedia, the Free Encyclopedia, 2006. [Online; accessed 31-July-2007].
22. M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.