H-MATCH: an Algorithm for Dynamically Matching Ontologies in Peer-based Systems *

S. Castano, A. Ferrara, and S. Montanelli

Università degli Studi di Milano
DICO - Via Comelico, 39, 20135 Milano - Italy
{castano,ferrara,montanelli}@dico.unimi.it

Abstract. In this paper, we present H-MATCH, an algorithm for dynamically matching distributed ontologies. By exploiting ontology knowledge descriptions, H-MATCH can be used to dynamically perform ontology matching at different levels of depth, with different degrees of flexibility and accuracy. H-MATCH has been developed in the HELIOS framework, conceived for supporting knowledge sharing and ontology-addressable content retrieval in peer-based systems.

1 Introduction

Ontologies are generally recognized as an essential tool for allowing communication and knowledge sharing among distributed users and applications, by providing a common understanding of a domain of interest. Due to the vision of the Semantic Web, a large body of research is being moving around ontologies, and contributions have been produced regarding methods and tools for covering the entire ontology life cycle, from design to deployment and reuse [8], and ontology languages, such as OIL [9] or OWL [18]. As a matter of fact, when considering distributed contexts, the knowledge of interest is generally provided by many different ontologies. For instance, the vision of the Semantic Web envisages the Web enriched with several domain ontologies, which specify formal semantics of data, for different intelligent services for information sharing, search, retrieval, and transformation [3, 11]. As another example, the problem of distributed knowledge sharing is eminent in the P2P area and is receiving a lot of attention in the research community [10, 15]. Basically, peers need to perform content retrieval by interacting with other peers of the network, and queries have to be routed and resolved based on knowledge descriptions available at the peers. To enable information processing and content retrieval in distributed contexts with a multitude of autonomous ontologies, appropriate *matching techniques* are required to determine semantic mappings between concepts of different ontologies that are semantically related [2, 7, 17]. Some research work on this topic has recently appeared. We review such work in the related work section of the paper.

* This paper has been partially funded by "Wide-scalE, Broadband, MIddleware for Network Distributed Services (WEB-MINDS)" FIRB Project funded by the Italian Ministry of Education, University, and Research.

An important requirement to be considered in developing ontology matching techniques for distributed contexts, such as the P2P, is related to the inherent dynamicity of the context, and to the need of matching techniques that are conceived to operate in a dynamic fashion. In this paper, we present H-MATCH, an algorithm for dynamically matching distributed ontologies. H-MATCH has been developed in the framework of HELIOS, the infrastructure we have conceived for supporting knowledge sharing and ontology-addressable content retrieval in peer-based systems [5, 6]. After introducing the reference architecture of a HELIOS peer ontology, we show how the ontology knowledge description can be exploited to perform dynamic ontology matching at different levels of depth, with different degrees of flexibility and accuracy.
The paper is organized as follows. In Section 2, we provide the main motivations of our work. In Section 3, we present the HELIOS ontology model for knowledge representation. In Section 4, we describe the foundations of our approach for ontology matching. In Section 5, we present the H-MATCH algorithm for semantic affinity evaluation. In Sections 6 and 7, we compare our approach with other recent approaches for distributed ontology matching, by showing the original contribution of our work. Finally, in Section 8, we give our concluding remarks.

2 Motivating scenario

To address the requirements of knowledge sharing and ontology matching in distributed systems, we consider a typical P2P scenario, where a number of peers can acquire or extend their knowledge by interacting with other peers of the network. As shown in Figure 1, we suppose that the peer A wants to enlarge its knowledge about the concept of Book by learning which nodes own concepts with semantic affinity with it. This requires capability to describe the knowledge owned by a peer and to match an incoming request against the knowledge of a peer, to find semantically related information to be returned to the requesting peer. The HELIOS (Helios Evolving Interaction-based Ontology knowledge Sharing) framework has been conceived to enable knowledge sharing and evolution considering a P2P system where nodes are equipotential in terms of functionalities and capabilities. The knowledge sharing and evolution processes in HELIOS are based on *peer ontologies*, describing the knowledge of each peer, and on *interactions among peers*, allowing information search and knowledge acquisition/extension, according to pre-defined *query models* and semantic techniques for ontology matching. Each peer has a different amount of knowledge, that depends on the interactions it has performed in the network. Each peer can acquire new knowledge and/or extend his knowledge only by querying peers which have this information. *Probe queries* are sent by a peer interested in extending its knowledge of the network. Each peer having concepts matching the target concept(s) of a probe query can answer to the requesting peer. When the peer A asks for semantically related contents about the target Book concept, peer B and peer C evaluate the semantic affinity between Book and the concepts contained in their respective peer ontology. The semantic affinity evaluation pro-

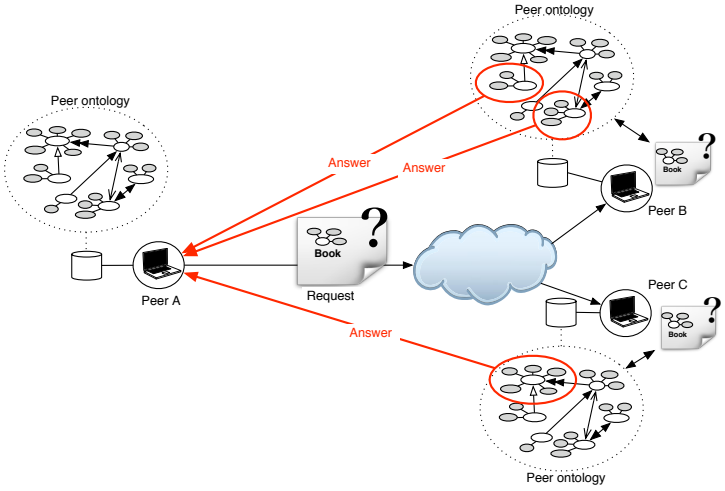


Fig. 1. Example of request/answer in the HELIOS network

cedure is based on the execution of the H-MATCH algorithm which determines the level of affinity of each concept in the peer ontology of peer B and peer C and the Book concept. Concepts having a high affinity value with Book are finally returned by peer B and peer C to the requesting peer A.
In the remainder of the paper, we focus on the formalization of the peer ontology knowledge model and on the H-MATCH algorithm for ontology matching.

3 Peer ontology representation

In this section, we provide a description of the architecture of a peer ontology and we formalize the peer ontology model adopted in HELIOS.

3.1 Ontology architecture

The ontology of a HELIOS peer is organized as a two-layer ontology, where the upper layer represents the *content knowledge* and the lower layer represents the *network knowledge* (See Figure 2).
The **Content Knowledge Layer** describes the knowledge of a peer, namely the knowledge a peer brings to the network and the knowledge the peer has of the network contents. We conceptualize the content knowledge layer as a network of *content concepts*, where each content concept is characterized by a set of *properties* and a set of *semantic relations* with other content concepts. A generic peer P can increase its content knowledge by adding new content concepts and/or by enriching existing content concept descriptions in terms of

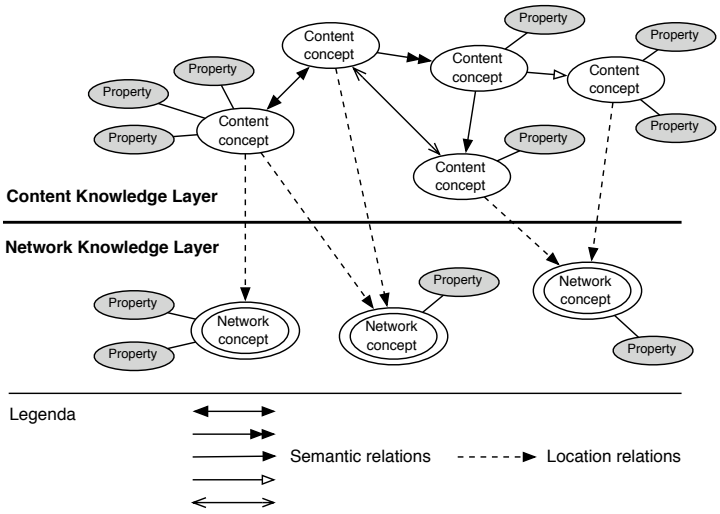


Fig. 2. Architecture of a peer ontology of a generic peer P

new properties and/or of new semantic relations, based on the answers acquired by other peers.
The **Network Knowledge Layer** describes the knowledge that a generic peer P has of other peers of the network it has interacted with. When a peer P receives a content concept from another peer P1, it stores in the network knowledge layer a description of the peer P1. Peer descriptions are given in form of *network concepts*, characterized by a set of properties describing the network features of a peer (e.g., IP-address).
An inter-layer relation, called *location relation* associates a content concept *cc* in the content knowledge layer with all network concept(s) describing peers storing concepts having semantic affinity with *cc*.

3.2 Peer ontology model

The peer ontology model organizes ontology knowledge in terms of *concepts*, *properties*, *semantic relations* and *location relations*, and is formally defined as follows.

Definition 1 (Peer Ontology). *A peer ontology PO is a 4-tuple of the form* $PO = (C, P, SR, LR)$*, where:*

- $C = CC \cup NC$ *is a set of concepts of PO, where CC is a set of* content concepts *of the content knowledge layer, and NC is a set of* network concepts *of the network knowledge layer.*

– *P is a set of concept properties. A property $p \in P$ is defined as a unary relation of the form $p(c)$, where $c \in C$ is the concept associated to the property p.*
– *$SR = \{$same-as, kind-of, part-of, contains, associates$\}$ is a set of semantic relations between content concepts. A semantic relation $sr \in SR$ is defined as a binary relation of the form $sr(c, c')$, where c and $c' \in CC$ are the content concepts related through sr.*
– *LR is a set of location relations between content concepts and network concepts. A location relation $lr \in LR$ is defined as a binary relation of the form $lr(c, c')$, where $c \in CC$ is a content concept in the content knowledge layer and $c' \in NC$ is a network concept in the network knowledge layer, respectively.*

To obtain a semantically rich and expressive representation of the knowledge in a peer ontology, we introduce the following semantic relations [1]:

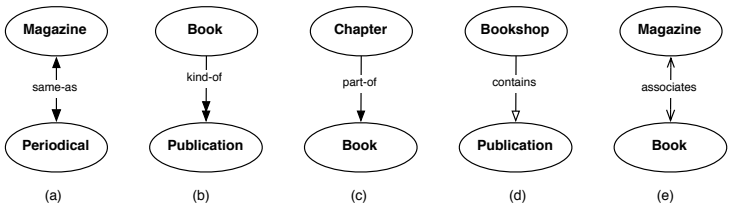


Fig. 3. Examples of semantic relations in a HELIOS peer ontology

– Same-as. The same-as relation is defined between two concepts c and c' which are considered semantically equivalent, that is, which denote the same real world entity or have the same meaning. As an example, we have Same-as(Periodical, Magazine) shown in Figure 3(a), referring to a peer ontology describing knowledge on publications.
– Kind-of. The kind-of relation defined between two concepts c and c' states that the concept c is a specialization of the concept c'. As an example, consider the case of Kind-of(Book, Publication) in Figure 3(b).
– Part-of. The part-of relation defined between two concepts c and c' states that the concept c represents a component of the concept c' as in the case of Part-of(Chapter, Book) shown in Figure 3(c).
– Contains. The contains relation defined between two concepts c and c' states that the concept c contains the concept c' as in the case of Contains(Bookshop, Publication) shown in Figure 3(d).

[1] The set SR of semantic relations has been defined according to relation classifications in ontology modelling [14] and metadata management [16] literature.

– Associates. The associates relation defined between two concepts c and c' states that a generic positive association is defined between c and c' . We use this relation when no other semantic relations hold between two concepts. As an example, consider the case of Associates(Magazine, Book) in Figure 3(e).

4 Foundations of ontology matching in HELIOS

The general goal of ontology matching techniques is to find concepts that have a semantic affinity with a target concept [2]. In this section, we propose an algorithm, called H-MATCH, for evaluating semantic affinity between concepts of different ontologies. In the context of HELIOS, we are interested in matching a target concept described in a query against a peer ontology (knowledge sharing), or in assimilating new concepts returned as the answer to probe queries into a peer ontology (knowledge evolution). H-MATCH grounds on the techniques developed in the ARTEMIS tool environment [1, 4] for the integration of heterogeneous data sources. In ARTEMIS, the semantic affinity evaluation is performed in the context of the schema matching process, in order to find mappings among elements of different source schemas that are semantically related for subsequent unification. In HELIOS, we extend and enrich the ARTEMIS techniques to address the typical problems of the ontology matching. In particular, the H-MATCH algorithm is based on the idea of considering both the linguistic features of concepts as well as the semantic relations among concepts in a peer ontology. Linguistic features are constituted by the semantic content of terms used as names of concepts and properties. The meaning of concepts is not established according to a given definition, but depends on the network of relations holding among terms (i.e., terminological relationships) and among concepts (i.e., semantic relations), respectively. Based on these considerations, the evaluation of the linguistic features is not based on a dictionary, where the meaning of each term depends on its definition, but on a thesaurus, where the meaning of each term is represented by the set of terminological relationships that it has with other terms in the thesaurus. Following the same approach, we assume that the meaning of a concept depends not only on its name, but also on its properties and on its semantic relations with other concepts in the ontology. To this purpose, the H-MATCH algorithm explicitly considers the *context* of each concept given by the set of its properties and of its adjacents (i.e., concepts which have a semantic relation with the considered concept), allowing a deep evaluation of semantic affinity between ontology concepts.

4.1 Linguistic interpretation

To capture the meaning of terms used as names of concepts and properties in a peer ontology, we exploit the terminological relationships among terms. In HELIOS, the network of terminological relationships is represented by a thesaurus,

[2] When speaking of concepts for matching, we refer to content concepts although not explicitly specified.

which is built by exploiting WordNet [13] as a source of lexical information, which can be possibly enriched by the ontology designer, if required. In particular, we consider a subset of the relations provided by WordNet represented by the following terminological relationships: {SYN (Synonym-of), BT/NT (Broader/Narrower Terms), RT (Related Terms)}, where the SYN relationship corresponds to the Synonym relation of WordNet, the BT/NT relationships correspond to the Hypernym/Hyponym relations of WordNet, and the RT relationship corresponds to the Meronym relation of WordNet, respectively. In the following, we denote by TR the set of terminological relationships in the HELIOS thesaurus.

4.2 Context interpretation

The H-MATCH algorithm evaluates the semantic affinity between two concepts by taking into account the affinity between their contexts. Given a concept $c \in CC$, we denote by $P(c) = \{p_i \mid p_i(c)\}$ the set of properties of c, and by $SR(c) = \{c_j \mid sr_j(c, c_j)\}$ the set of adjacents of c, namely all concepts c_j which have a semantic relation sr_j with c. The context of a concept is defined as follows:

Definition 2 (Concept context). *The context $Ctx(c)$ of a concept $c \in CC$ is defined as the union of the properties and of the adjacents of c, that is, $Ctx(c) = P(c) \cup SR(c)$.*

An example of concept context for the Volume concept is shown in Figure 4, where content concepts are represented as white ovals, properties are represented as grey ovals, and relations as arrows, respectively.

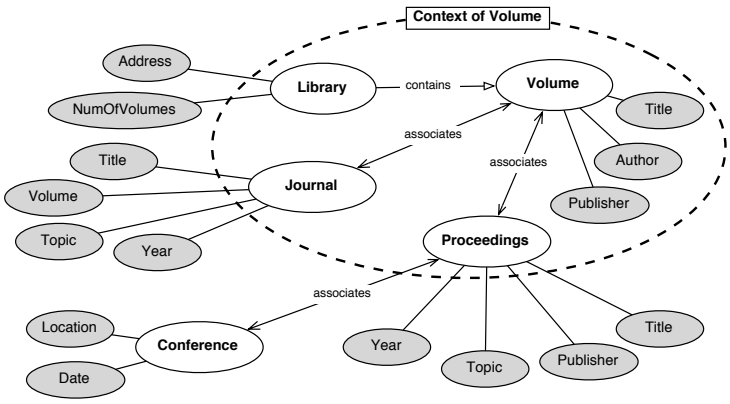


Fig. 4. Example of context for the Volume concept in a peer ontology

5 The H-MATCH algorithm

The semantic affinity between two ontology concepts c and c' is evaluated in HELIOS by weighting both the terminological relationships in the thesaurus and the semantic relations in the contexts of c and c', respectively. In Table 1, we report the weights associated which each kind of terminological relationship and semantic relation, respectively. The weights associated with the terminological

	Relation	**Weight**
Linguistic interpretation	SYN	1.0
	BT/NT	0.8
	RT	0.5
Context interpretation	property	1.0
	same-as	1.0
	kind-of	0.8
	part-of	0.7
	contains	0.5
	associates	0.3

Table 1. Weights associated with terminological and semantic relations

relationships are taken from ARTEMIS, where they have been tested on several real integration cases. The weights associated with semantic relations have been defined in HELIOS to express a measure of the strength of the concept connection posed by each relation for semantic affinity evaluation purposes. The higher is the weight associated with a semantic relation, the higher is the strength of the semantic connection between concepts. Furthermore, we associate the weight 1.0 with properties since they are strongly related to a concept and provide its structural description. The weight associated with the terminological relationships are exploited for performing linguistic affinity evaluation, while the weights associated with properties and semantic relations are exploited for performing contextual affinity evaluation, respectively.

5.1 Linguistic affinity

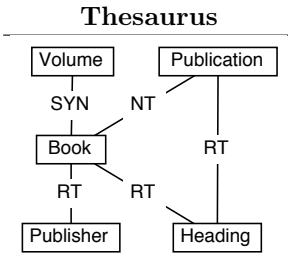
The aim of the linguistic affinity is to evaluate the semantic affinity between two concepts by considering the semantic contents of their names as terms in the thesaurus. An affinity function $LA(t, t')$ is defined to evaluate the affinity between two terms t and t', as shown in Figure 5. The affinity $LA(t, t')$ of two terms t and t' is equal to the highest-strength path of terminological relationships between them in the thesaurus, if at least one path exists, and is zero otherwise. Given t and t' and a path of terminological relationships between them, the strength of

function $LA(t, t')$
input two terms t and t'
output linguistic affinity value between t and t'
begin function
  def $x = 0$, $y = 1$;
  if exists a path P of terminological relationships $tr_i \in TR$ between t and t'
    /* σ_{tr_i} is the weight associated with each $tr_i \in$ P */
    for each P
      $y = 1$;
      for each $tr_i \in P$
        $y = y \cdot \sigma_{tr_i}$;
      if $y \geq x$
        $x = y$;
  return x;
end function

Fig. 5. The $LA()$ function for linguistic affinity evaluation

this path is computed by multiplying the weights of all terminological relationships forming the path.

Example 1. As an example of linguistic affinity evaluation, we consider the portion of thesaurus shown in Figure 6. Suppose we are interested in the linguistic affinity of concepts Book and Publication. Two paths exist between Book and Publication in the thesaurus. The first path P1 is {NT(Book,Publication)}. The second path P2 is composed by {RT(Book,Heading), RT(Heading,Publication)}. A graphical representation of the thesaurus graph and of the results of the linguistic affinity evaluation are shown in Figure 6. The linguistic affinity of Book



LA(Book,Publication)

Path	Path composition	Path evaluation	Result
P1	[NT]	0.8	0.8
P2	[RT, RT]	$0.5 \cdot 0.5$	0.25

Fig. 6. Example of linguistic affinity evaluation between the Book and Publication

and Publication is 0.8, obtained by considering the path P1.

5.2 Contextual affinity

The aim of the contextual affinity is to calculate a measure of affinity between concepts based on their contexts. To this purpose, we evaluate the linguistic affinity of properties and adjacents, as well as the degree of closeness between the semantic relations that are involved in concept contexts.

Relation affinity function. The aim of the relation affinity function is to calculate a measure of closeness between two semantic relations or between a semantic relation and a property, based on their associated weights (see Table 1). Function $RA(r, r')$ is defined to evaluate the affinity between r and r', where r and r' are either two semantic relations or a semantic relation and a property, respectively. The relation affinity function $RA(r, r')$ is reported in Figure 7. The relation

function $RA(r, r')$
input relations r and r'
output relational affinity value between r and r'
begin function
  def σ_r, $\sigma_{r'}$ as the weights associated with r and r', respectively
  def $x = 0$;
  $x = 1 - \mid \sigma_r - \sigma_{r'} \mid$;
  return x;
end function

Fig. 7. The $RA()$ function for relational affinity evaluation

affinity is a value in the range [0,1] and is proportional to the level of closeness of the considered relations. The highest value (i.e., 1.0) is obtained when r and r' have the same weight. The higher the difference between the weights associated with the relations, the lower the relation affinity value.

Evaluation of the contextual affinity. The contextual affinity evaluation is performed by exploiting a function $CA(CV(c), CV(c'))$ on the contexts of two concepts c and c'. In this function, context $Ctx(c)$ of a concept c is represented through a context vector $CV(c) = (cv_1, ..., cv_n)$, where $\forall i \in (1, ..., n), cv_i = (f_i, r_i)$, where f_i denotes either a property or an adjacent concept of c, and r_i denotes the semantic relation between c and f_i. The contextual affinity function is defined as shown in Figure 8.

Based on some experimental results, we noted that in the contextual affinity evaluation the impact of the concepts with low affinity is stronger than the impact of the concepts with a high affinity, thus originating biased measures. For this reason, a control factor F_k has been introduced for refining the results of the contextual affinity evaluation. In particular, in presence of very low affinity values, F_k proportionally increases them, in order to better balance all affinity values in the context and avoid too large gaps between affinity results.

function $CA(CV(c), CV(c'))$
input the context vectors $CV(c)$ and $CV(c')$ representing the contexts of the concepts c and c', respectively
output contextual affinity value of c and c'
begin function
  def $x = 0, y = 0, z = 0$;
  foreach $cv \in CV(c) \mid cv = (f, r)$;
    foreach $cv' \in CV(c') \mid cv' = (f', r')$;
      $y = LA(f, f') \cdot RA(r, r')$;
      $z = z + y$;
  $z = z \div (\mathbf{length}(CV(c)) \cdot \mathbf{length}(CV(c')))$;
  /* $F_k = 1 + (1 - z)$ is a control factor */
  $x = z \cdot F_k$;
  return x;
end function

Fig. 8. The $CA()$ function for contextual affinity evaluation

Example 2. As an example of the contextual affinity evaluation, we consider the concepts Book and Volume shown in Figure 9, with their respective contexts:

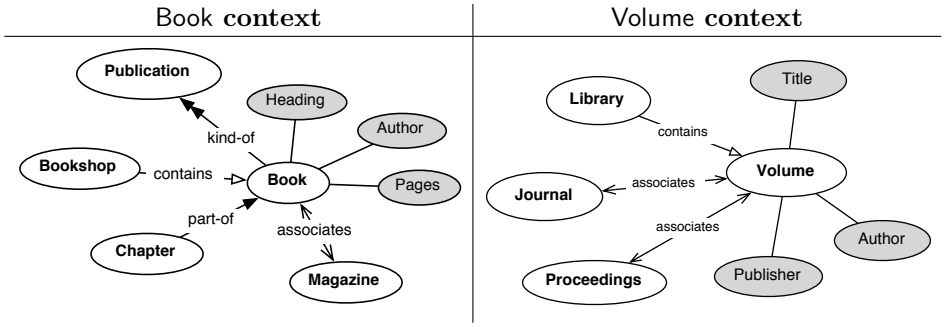


Fig. 9. The contexts of the Book and Volume concepts

$CV(\text{Book}) = [(\text{Heading, property}), (\text{Author, property}), (\text{Pages, property}), (\text{Magazine, associates}), (\text{Chapter, part-of}), (\text{Bookshop, contains}), (\text{Publication, kind-of})]$

$CV(\text{Volume}) = [(\text{Title, property}), (\text{Author, property}), (\text{Publisher, property}), (\text{Proceedings, associates}), (\text{Journal, associates}), (\text{Library, contains})]$

The linguistic affinity and the relation affinity are evaluated as shown in Table 2. The contextual affinity $CA(CV(\text{Book}), CV(\text{Volume}))$ is evaluated by exploiting

Linguistic affinity (CV(Book), CV(Volume))							
$LA()$	Heading	Author	Pages	Magazine	Chapter	Bookshop	Publication
Title	0.5	0.25	0.25	0.25	0.25	0.25	0.4
Author	0.25	1.0	0.25	0.25	0.25	0.25	0.4
Publisher	0.25	0.0	0.25	0.5	0.25	0.0	0.5
Proceedings	0.25	0.25	0.0	0.64	0.25	0.0	0.8
Journal	0.25	0.25	0.0	0.64	0.25	0.0	0.8
Library	0.25	0.25	0.0	0.5	0.25	0.5	0.5

Relation affinity (CV(Book), CV(Volume))							
$RA()$	property	property	property	associates	part-of	contains	kind-of
property	1.0	1.0	1.0	0.3	0.7	0.5	0.8
property	1.0	1.0	1.0	0.3	0.7	0.5	0.8
property	1.0	1.0	1.0	0.3	0.7	0.5	0.8
associates	0.3	0.3	0.3	1.0	0.6	0.8	0.5
associates	0.3	0.3	0.3	1.0	0.6	0.8	0.5
contains	0.5	0.5	0.5	0.8	0.8	1.0	0.7

Table 2. Linguistic and relation affinity evaluation for the contexts of Book and Volume

the $LA()$ and $RA()$ results, according to the function definition shown in Figure 8:

$$CA(CV(\text{Book}), CV(\text{Volume})) = (9.4 \ / \ 42) \cdot 1.78 = 0.40$$

5.3 The H-MATCH algorithm

The H-MATCH algorithm evaluates the semantic affinity between two concepts by considering both their linguistic and contextual affinity. H-MATCH can be configured for differently evaluating concept semantic affinity, by setting the impact of the linguistic and the contextual affinity, and by choosing dynamically which part of concept context has to be considered in the matching process. This flexibility of H-MATCH has the aim of facing two different requirement of the ontology matching process. The first requirement regards the balance between the linguistic and the contextual features of concepts in a peer ontology. The meaning of the peer ontology concepts depends basically on the terms used for their definition and on the relations they have with other concepts in the ontology. In HELIOS, we are interested in addressing the fact that those features can have a different impact in different ontology structures. A second requirement regards the context evaluation, in which we distinguish between properties and concepts. The role of the properties in the concept definition might have a different relevance in different peer ontologies. As an example, if a peer ontology

is defined describing high structured data sources (e.g., relational databases), the properties which describe the structure of each concept have a high impact on the concept meaning evaluation. Furthermore, the composition of the context and its extension in terms of number of adjacents have an impact on the matching quality and on its performance. The aim of H-MATCH is to allow a dynamic choice of the kind of features to be considered in the semantic affinity evaluation.

Matching models. In order to address these requirements, three different matching models are proposed in HELIOS to configure H-MATCH.

- *Shallow matching.* The shallow matching is performed by considering only the linguistic information provided by the concept names and by the reference thesaurus. The precision of the semantic affinity evaluation depends on the choice of the concept names in the ontology definition. Meaningful and precise names will guarantee more appropriate results. Being based only on linguistic information, the shallow matching guarantees a high performance since requires less computation than the other two models, and is recommended when only concept names are specified in a query.
- *Intermediate matching.* The intermediate matching is performed by considering concept names and also concept properties. With this model, we want a more accurate level of matching by taking into account the property part of the concept context.
- *Deep matching.* The deep matching model considers concept names and the whole context of concepts. The deep matching requires a complete description of target concept in the query and guarantees the highest level of precision in the semantic affinity evaluation. As such, it requires more computation than the other two, and is recommended when the accuracy is more important than the response time.

Linguistic and contextual information balancing. The problem of dynamically setting the balance between the linguistic and the contextual features of a peer ontology in the matching process is addressed in HELIOS by setting a weight $W_{LA} \in [0,1]$ which measures the degree of the impact of the linguistic affinity in the semantic affinity evaluation process.

H-MATCH *algorithm.* The input of the H-MATCH algorithm is constituted by: two concepts c and c'; the matching model; the value of the weight W_{LA}. Deep and 0.5 are the default values for the matching model and W_{LA}, respectively. W_{LA} =0.5 ensures that the linguistic affinity and the contextual affinity have the same impact in the semantic affinity evaluation. The output of H-MATCH is the semantic affinity value of c and c', calculated as the weighted sum of their linguistic affinity and contextual affinity. The H-MATCH algorithm is shown in Figure 10. The algorithm exploits the $LA()$ and $CA()$ functions for evaluating the linguistic and the contextual affinity values, respectively. The choice of the matching model determines the composition of the context vectors used for the

algorithm H-MATCH($c, c', model =$ "deep", $W_{LA} = 0.5$)
input the concepts c and c', the matching model $\in$ [shallow; intermediate; deep], and the weight $W_{LA} \in$ [0,1]
output the semantic affinity value between c and c'
begin algorithm
  def t, t' as the names of c and c', respectively;
  def $CV(c) = [], CV(c') = []$ as the context vectors for c and c', respectively;
  def $context\_item = []$ as a pair of the form (f, r), where f is a name associated with a property or a concept, and $r \in$ {property; same-as; kind-of; part-of; contains; associates};
  def $x = 0, y = 0, semantic\_affinity = 0$;
  $x = LA(t, t')$;
  switch $model$
    case "shallow" :
      $W_{LA} = 1$;
    case "intermediate" :
      foreach property $p(c) \in Ctx(c)$
        $context\_item = [p(c),$"property"$]$;
        append $context\_item$ **to** $CV(c)$;
      foreach property $p(c') \in Ctx(c')$
        $context\_item = [p(c'),$"property"$]$;
        append $context\_item$ **to** $CV(c')$;
    case "deep" :
      foreach property $p(c) \in Ctx(c)$
        $context\_item = [p(c),$"property"$]$;
        append $context\_item$ **to** $CV(c)$;
      foreach concept $c_i \in Ctx(c)$
        /* $sr(c, c_i)$ is the semantic relation between c and c_i */
        $context\_item = [c_i, sr(c, c_i)]$;
        append $context\_item$ **to** $CV(c)$;
      foreach property $p(c') \in Ctx(c')$
        $context\_item = [p(c'),$"property"$]$;
        append $context\_item$ **to** $CV(c')$;
      foreach concept $c_j \in Ctx(c')$
        /* $sr(c', c_j)$ is the semantic relation between c' and c_j */
        $context\_item = [c_j, sr(c', c_j)]$;
        append $context\_item$ **to** $CV(c')$;
  $y = CA(CV(c), CV(c'))$;
  $semantic\_affinity = W_{LA} \cdot x + (1 - W_{LA}) \cdot y$;
  return $semantic\_affinity$;
end algorithm

Fig. 10. The H-MATCH algorithm

contextual affinity evaluation. If the shallow model is chosen, W_{LA} is set to 1, and only the linguistic affinity is considered. Otherwise, W_{LA} is exploited in order to correctly combine the linguistic affinity value with the contextual affinity value.

Example 3. Consider the concepts of Book and Volume of Example 2. Below, we report the semantic affinity of Book and Volume obtained by exploiting the H-MATCH algorithm according to the three different matching models, with $W_{LA} = 0.5$.

- Shallow matching. The shallow matching returns a semantic affinity value which coincides with the linguistic affinity value, that is:

$$\text{H-MATCH}(\text{Book},\text{Volume},\text{“shallow”},0.5) = 1$$

- Intermediate matching. The intermediate matching evaluates the linguistic and the contextual affinity, by considering only the properties in the contexts of Book and Volume, that is:

$$\text{H-MATCH}(\text{Book},\text{Volume},\text{“intermediate”},0.5) = 0.5 \cdot 1 + 0.5 \cdot 0.55 = 0.78$$

- Deep matching. The deep matching evaluates semantic affinity by considering the whole contexts of Book and Volume, that is:

$$\text{H-MATCH}(\text{Book},\text{Volume},\text{“deep”},0.5) = 0.5 \cdot 1 + 0.5 \cdot 0.40 = 0.7$$

Considerations. We note that in our example the deeper is the matching model used for semantic affinity evaluation, the lower is the semantic affinity returned for Book and Volume. It depends on the fact that considering the context of the concepts to be matched, H-MATCH is able to capture more precisely the differences between them than considering only the linguistic affinity. In particular, H-MATCH is useful in order to address the fact that the same concept can have a different meaning if used in different contexts. In our example, the Book and the Volume concepts, which are synonyms from a linguistic point of view, are used in a bookstore context and in a library context, respectively. The differences between the kind of publications contained in the bookstore context and in the library context are the reason of the decreasing value of semantic affinity when applying the deep match.

6 Related work

In this section, we overview the main approaches for ontology matching in distributed systems.

Edamok [17] is a research project focused on semantic interoperability issues in P2P systems. The project implements the KEx (Knowledge Exchange) P2P system which aims to realize knowledge sharing among peer communities of interest (called federations). The system is based on the concept of context of a peer, to represent the interests of the peer. KEx implements specific tools (e.g., context

editors, context extractors) to extract the context of a peer from the peer knowledge (e.g., file system, mail messages). In order to point out semantic mapping between concepts stored in distinct peers, the system uses the CTX-MATCH algorithm. This algorithm compares the knowledge contained in different contexts looking for semantic mappings denoting peers interested in similar concepts. These mappings are stored in order to assist the query resolution components to direct queries to peers which store relevant information. The CTX-MATCH is based on a *semantic explicitation* phase where concepts are associated with the correct meaning with respect to their context and on a *semantic comparison* phase where concepts are translated in logical axioms and matched. The algorithm implements a description logic approach: mapping discovering is reduced to the problem of checking a set of logical relations.

Chatty web [2] represents a novel approach for obtaining semantic interoperability among data sources in a semi-automatic manner. This approach applies to any system which provides a communication infrastructure (e.g., decentralized systems, P2P systems) and offers the opportunity to study semantic interoperability as a global phenomenon in a network of information sharing communities. Each peer offers data which are organized according to some schema expressed in a data model (e.g., relational, XML, RDF). Semantic interoperability is accomplished by assuming the existence of local agreements provided as mappings between different schemas. Peers introduce their own schemas and exchanging translations between them; then peers can incrementally come up with an implicit "consensus schema" which gradually improves the global search capabilities of the system. The paper identifies different methods that can be applied to establish global forms of agreement starting from a graph of local mappings among schemas and presents the *gossiping* algorithm which is used to identify the sufficiently large set of peers capable of rendering meaningful results on a specified query.

GLUE [7] is a system that employs machine learning techniques to find semantic mappings between concepts stored in distinct and autonomous ontologies. Given two distinct ontologies, the mapping discovery process between their concepts is based on the measure of similarity which is defined through the *joint probability distribution*. GLUE follows a probabilistic approach: the measure of similarity between the concepts A and B is computed as the likelihood that an instance belongs to both the concepts ($P(A \cap B)$). According to these probabilistic measurements, two base learning techniques are applied in order to build a similarity matrix expressing the prediction of semantic affinity between concepts. A *relaxation labeling* procedure is performed in order to improve the matching accuracy of the affinity predictions. Domain-independent and domain-dependent constraints are introduced to evaluate such kind of refinement process.

KAON [14] is an ontology and Semantic Web tool suite. In [14], the authors discuss the problem of ontology representation and querying for semantics-driven applications, describing a prototype implementation within the KAON system. In particular, the paper presents the mathematical definition of the KAON modeling language, and the denotational semantics for it. The ontology structure is

presented as a view of a general model, called OI-model, which consists of entities and may include a set of other OI-models. The ontology structure contains definitions specifying how instances should be constructed, and is composed by concepts and properties. The properties can have domain concepts, and relational properties can have range concepts. Relational properties may be marked as transitive and/or symmetric and it is possible to define inverse properties for each relational property. The emphasis of this system is on ontology definition and on formal properties for correctness and completeness.
The original contribution of our ontology matching techniques, with respect to these approaches, is the use of combined semantic affinity evaluation strategies to obtain a flexible and dynamic algorithm. The H-MATCH algorithm is able to discover the location of semantically related concepts to a target argument without requiring a complete description and matching procedure between independent ontologies. In the next section, we deeply compare H-MATCH with the approach adopted in Edamok, by discussing our contribution in more detail.

7 Applicability issues

We made a comparison of H-MATCH and the matching techniques developed in the Edamok [17] project, which are more strictly related to our approach. In particular, the aim of the comparison is to verify which mappings are discovered by the two techniques for a given concept, by considering as the reference case study the Art domain concept hierarchies of Google[3] and Yahoo[4] shown in Figure 11. In particular, we are interested in discovering which concepts of the Yahoo hierarchy match the Art history concept of the Google hierarchy. In [17], the following relations are discovered for the Art history concept:

$$\text{Arts/Art history} \equiv \text{Arts \& Humanities/Art History}$$
$$\text{Arts/Art history} \sqsupseteq \text{Arts \& Humanities/Design Art/Architecture/History}$$

In HELIOS, the H-MATCH algorithm is exploited to discover the semantic affinity value between the Art history concept and each concept of the Yahoo hierarchy. In this example, we set H-MATCH by choosing the deep matching model and $W_{LA} = 0.5$. In order to address the fact that the concept hierarchies are resource directories in Google and Yahoo, in HELIOS, we represent the is-a relations by means of the contains semantic relation. The linguistic and contextual affinity are evaluated as described in Section 4. In particular, the H-MATCH algorithm is performed by considering the context of the concept Art history and the contexts of the concepts in the Yahoo hierarchy, as shown in Figure 12. The results obtained with H-MATCH are the following:

[3] www.google.com
[4] www.yahoo.com

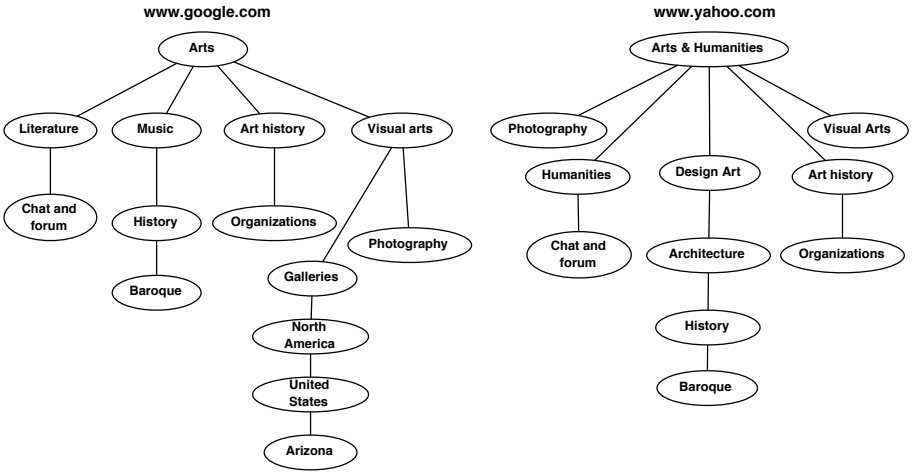


Fig. 11. The Art domain concept hierarchies of Google and Yahoo

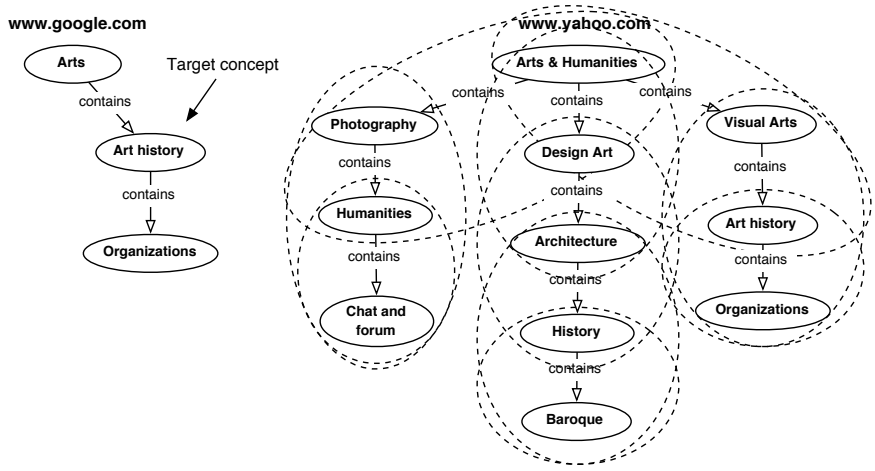


Fig. 12. Concept contexts involved in the semantic affinity evaluation between Art history of Google and the Yahoo concept hierarchy

H-MATCH(Art history, Art history) = 1
H-MATCH(Art history, History) = 0.72
H-MATCH(Art history, Photography) = 0.57
H-MATCH(Art history, Visual arts) = 0.57
H-MATCH(Art history, Design art) = 0.55
H-MATCH(Art history, Arts & humanities) = 0.54
H-MATCH(Art history, Humanities) = 0.54
H-MATCH(Art history, Architecture) = 0.5
H-MATCH(Art history, Baroque) = 0.47
H-MATCH(Art history, Organizations) = 0.22
H-MATCH(Art history, Chat & Forum) = 0.21

A full comparison between our results an those discussed in [17] is not possible, because the H-MATCH algorithm results cannot be interpreted as semantic relations among the considered concepts. An interesting point about the comparison, is the fact that the concepts having the highest semantic affinity value with Art history in the H-MATCH results (i.e., Art history and History) are the same concepts discovered by the CTX-MATCH algorithm presented in [17]. In conclusion, the H-MATCH algorithm is a valid support for discovering, given a concept ontology, a set of corresponding concepts in another ontology. The main contribution of our techniques is the fact that H-MATCH gives a measure of correspondence in terms of semantic affinity among concepts. On these measures, a set of different interpretations are possible in order to define mappings between the considered concept ontologies. For instance, when using H-MATCH for query resolution a threshold is used in order to select the concepts which have the highest semantic affinity with the target concept in the query.

8 Concluding remarks

In this paper, we have presented the H-MATCH algorithm for dynamic distributed ontology matching. Considering the linguistic affinity evaluation as an atomic step, H-MATCH has a complexity of $O(N^2)$, being N the number of elements in the contexts of the concept to be matched. We are working in the direction of testing the algorithm on real matching cases in the context of HELIOS, to evaluate and experiment performance and scalability issues posed by ontology-based query resolution considering large ontologies.
The accuracy of the matching results depends on the thesaurus (e.g., WordNet) which may not be sufficient to precisely evaluate semantic similarity between two terms in different vocabularies. Such vocabulary heterogeneity may cause a loss of information. This problem has been discussed in [12], where measures and metrics are used to select results having the desired quality of information. To this end, future work will be devoted to the extension of our techniques taking into account aspects related to the quality of information.

References

1. The ARTEMIS project web site. http://islab.dico.unimi.it/artemis/d2i/.

2. K. Aberer, P. Cudrè-Mauroux, and M. Hauswirth. The chatty web: Emergent semantics through gossiping. In *Proc. of the Twelfth International World Wide Web Conference, (WWW2003)*, Budapest, Hungary, May 2003.
3. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.
4. S. Castano, V. De Antonellis, and S. De Capitani di Vimercati. Global viewing of heterogeneous data sources. *IEEE Transactions on Data and Knowledge Engineering*, 13(2):277–297, 2001.
5. S. Castano, A. Ferrara, S. Montanelli, E. Pagani, and G.P. Rossi. Ontology-addressable contents in P2P networks. In *Proc. of WWW'03 1st SemPGRID Workshop*, Budapest, May 2003. http://www.isi.edu/ stefan/SemPGRID/proceedings/proceedings.pdf.
6. S. Castano, A. Ferrara, S. Montanelli, and D. Zucchelli. HELIOS: a general framework for ontology-based knowledge sharing and evolution in P2P systems. In *Proc. of DEXA'03 2nd Web Semantics Workshop*, Prague, Czech Republic, September 2003.
7. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proc. of the Eleventh International World Wide Web Conference, (WWW2002)*, Honolulu, Hawaii, USA, May 2002.
8. D. Fensel. *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag, Berlin, 2001.
9. D. Fensel, I. Horrocks, F. van Harmelen, S. Decker, M. Erdmann, and M. Klein. OIL in a nutshell. In *In Knowledge Acquisition, Modeling, and Management, Proceedings of the European Knowledge Acquisition Conference (EKAW-2000)*, pages 1–16, Juan-les-Pins, France, October 2000. Springer-Verlag.
10. A. Halevy, Z. Ives, D. Suciu, and I. Tatarinov. Schema mediation in peer data management systems. In *Proc. of ICDE'03*, Bangalore, India, March 2003.
11. J.Heflin and J.Hendler. A portrait of the Semantic Web in action. *IEEE Intelligent System*, 16:54–59, May 2001.
12. E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. Imprecise answers on highly open and distributed environments: An approach based on information loss for multi-ontology based query processing. *International Journal of Cooperative Information Systems (IJCIS)*, 9(4):403–425, December 2000.
13. A.G. Miller. WordNet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
14. B. Motik, A. Maedche, and R. Volz. A conceptual modeling approach for semantics-driven enterprise applications. In *Proc. of the First International Conference on Ontologies, Databases and Application of Semantics (ODBASE-2002)*, 2002.
15. Nejdl et al. EDUTELLA: a P2P networking infrastructure based on RDF. In *Proc. of the Eleventh International World Wide Web Conference, WWW2002*, Honolulu, Hawaii, USA, May 2002.
16. R.A. Pottinger and P.A. Bernstein. Merging models based on given correspondences. Technical report, University of Washington, February 2003. Available at ftp://ftp.cs.washington.edu/tr/2003/02/UW-CSE-03-02-03.pdf.
17. L. Serafini, P. Bouquet, B. Magnini, and S. Zanobini. An algorithm for matching contextualized schemas via SAT. Technical report, DIT University of trento, Italy, January 2003. Available at http://eprints.biblio.unitn.it/archive/00000348/.
18. F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, and L.A. Stein. OWL (march 2003) reference description. http://www.w3.org/TR/2003/WD-owl-ref-20030331/.