Towards Distributed Information Retrieval in the Semantic Web: Query Reformulation Using the oMAP Framework[*]

Umberto Straccia[1] and Raphaël Troncy[2]

[1] ISTI-CNR, Via G. Moruzzi 1, 56124 Pisa, Italy
straccia@isti.cnr.it
[2] CWI Amsterdam, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
raphael.troncy@cwi.nl

Abstract. This paper introduces a general methodology for performing distributed search in the Semantic Web. We propose to define this task as a three steps process, namely *resource selection*, *query reformulation/ontology alignment* and *rank aggregation/data fusion*. For the second problem, we have implemented *oMAP*, a formal framework for automatically aligning OWL ontologies. In oMAP, different components are combined for finding suitable mapping candidates (together with their weights), and the set of rules with maximum matching probability is selected. Among these components, traditional terminological-based classifiers, machine learning-based classifiers and a new classifier using the structure and the semantics of the OWL ontologies are proposed. oMAP has been evaluated on international test sets.

1 Introduction

Information Retrieval (IR) studies the problem of finding a (ranked) set of documents that are relevant for a specific information need of a user. One of the premises of the Semantic Web is that it provides the means to use metadata that help determining which documents are relevant. In a Semantic Web-based version of IR, not only the sheer amount of data, but also the differences among the local metadata vocabularies, call for a distributed approach. In this paper, we propose a three-step framework for distributed, Semantic Web-enabled Information Retrieval. The first step is *resource selection*, because on the Semantic Web it is unlikely that for any given query the full Web has to be queried. The second step, *query reformulation and ontology alignment* deals with the differences in the vocabularies used by the user and the selected information resources. The third and last step, *aggregation and data fusion* integrates the ranked results from the individual resources into a single ranked result list. In this paper, we focus on the second step for which we describe an efficient model that is compared with other approaches using the independent OAEI[1] benchmarks.

[*] This work was carried out during the tenure of an ERCIM fellowship.
[1] http://oaei.inrialpes.fr.

The paper is organized as follows. We briefly present in the next section our main problem: distributed search over the Semantic Web. Then, we introduce in Section 3 *oMAP*, a framework whose goal is to automatically align all the entities defined in two OWL ontologies. These mappings are then used for the query reformulation process. The mappings are obtained by combining the prediction of different classifiers. We describe the set of classifiers used: terminological, machine learning-based and we present a new one, based on the structure and the semantics of the OWL axioms. We have evaluated oMAP on an independent test set provided by an international ontology alignment contest and we show our results with respect to the other competitors in Section 4. Finally, we provide some related work and give our conclusions and future work in Section 5.

2 Motivating Problem

In Information Retrieval the task of Distributed IR (DIR) [3] is the task, given an information need, of accessing (and retrieving from) in an effective way distributed information resources [2]. DIR has been proposed to overcome the difficulties of centralized approaches. For instance, information resources become more and more "proprietary" and not crawl-able. That is, more and more the content information resources (e.g. Web repositories, Digital Libraries) cannot be crawled anymore and, thus, indexed by a centralized Web retrieval engine. Documents may be accessed by issuing a specific query to the information resource only and remain mostly hidden to Web search engines. DIR is an effective solution to this problem as it aims at, given an information request, to discover the relevant information resources and to query them directly. So, in DIR we do not require to crawl and index documents, but just to select relevant resources and submitting appropriately a query to them. In the following, we show how DIR can be reformulated in the context of the Semantic Web.

2.1 Towards Distributed Search in the Semantic Web

In order to effectively cope with very large amounts of knowledge, the task of distributed search in the Semantic Web may be defined in terms of three different sub-tasks. Let us assume that an agent A has to satisfy an information need Q_A expressed in a query language $\mathcal{Q}_A$, whose basic terms belong to an ontology O_A, defined using the ontology language $\mathcal{O}_A$. Let us assume also that there are a large number of ontology-based Web resources $\mathscr{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_n\}$ accessible to A, where each Web resource $\mathcal{S}_i$ provides access to its Web pages by having its own ontology O_i, ontology language $\mathcal{O}_i$ and query language $\mathcal{Q}_i$ (see Figure 1).

Then, the agent should perform the following three steps to satisfy its information need:

1. **Resource selection:** The agent has to *select* a subset of some *relevant* resources $\mathscr{S}' \subseteq \mathscr{S}$, since it is not reasonable to assume that it will access and query all the resources;

[2] The techniques of DIR are also applied in so-called *Metasearch engines* [27]

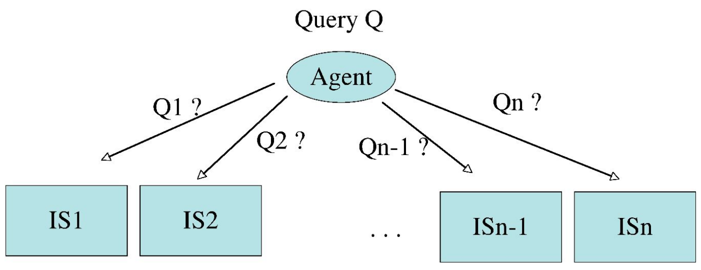

Fig. 1. Distributed Information Retrieval

2. **Query reformulation:** For every selected resource $\mathcal{S}_i \in \mathscr{S}'$ the agent has to *re-formulate* its information need Q_A into the query language $\mathcal{L}_i$ provided by the resource;
3. **Data fusion and rank aggregation:** The results from the selected resources have to finally be merged together.

That is, an agent must know *where* to search, *how* to query, and *how* to combine the information and ranked lists provided back from querying different and heterogeneous resources. As information resources continue to proliferate, these problems become major obstacles to information access. This is an ineffective manual task for which accurate automated tools are desired.

As noted previously, the problem of DIR has already been addressed in the context of textual IR. Our approach to DIR in the Semantic Web is incremental and tries to follow the way IR addressed the issue. The tasks of automated resource selection and the one of query reformulation seem to be the more problematic ones, while the data fusion and rank aggregation issue may be solved apparently by applying directly existing techniques [19]. Therefore, the latter will not be discussed further in this paper.

In IR, both the automated resource selection and the query-reformulation tasks are fully automatic and do not require human intervention. In order to make resource selection effective and automatic, in DIR, an agent has to compute an approximation of the content of each information resource. Based on this approximation, the agent is then able to select resources and perform query reformulation effectively. The approximation is computed by relying on the so-called *query-based resource sampling methodology* (see, e.g. [4]). This method consists of computing automatically an approximation of the content of a resource, relying on a sampling technique. Roughly, it consists of a series of quasi-random queries submitted to the information resource. In the context of textual IR, it has been shown that the retrieval of a few documents is a sufficient representation of the content of information resource [4]. In automated resource selection, this approximation is then used to decide whether a resource may contain relevant information with respect to the agents' information need [3]. For ontology-based information resources such an approximation may contain the ontology the information resource relies on and some annotated documents (called instances) retrieved using quasi-random queries.

For the query reformation task, the agent relies on so-called transformation rules, which dictates how to translate concepts and terms of the agent's vocabulary into the vocabulary of the information resource. Once the set of rules is given, the query transformation is relatively easy. What is difficult is to learn these rules automatically. In the context of the Semantic Web, these rules are essentially rules which map an entity (concept or property) in the agent's ontology into one or several entities of the information resource's ontology. Therefore, the major difficulty is in learning these *ontology mappings* automatically. Again, to do this, we may rely on the approximation computed so far through query-based sampling. The ontology of the agent, the ontology of the information resource and some annotated documents will allow the agent to learn these mappings automatically. This task is called *Ontology Alignment* in the Semantic Web. Furthermore, from a DIR perspective, these mappings are often established only to a degree of probability to which the mapping is true. [16] also shows that this degree cannot be neglected during the DIR process without loosing in retrieval effectiveness.

In summary, while numerous works deal with one of the three sub-tasks described above for distributed textual IR, to the best of our knowledge, very few works address the issue of distributed search in the context of the Semantic Web, where documents are well-structured and annotated semantically using terms belonging to a (formal) ontology. In this paper we tackle the second task, namely the problem of query reformulation in general and automatically learning mapping rules in particular, in the context of OWL-annotated resources. The first task (resource selection) will be addressed in future work.

2.2 Query Reformulation

In the context of database schema matching, [16] proposes to rely on *Probabilistic Datalog* (pDatalog for short) [12] to express mapping rules and use it for query reformulation. We show that we can use it in our context as well. pDatalog, for which an effective implementation exists, is an extension to Datalog, a variant of predicate logic based on function-free Horn clauses. Queries and mapping rules are probabilistic rules (see examples below). The mapping rules we consider are of the form

$$\alpha_{i,j}\ T_j(x) \leftarrow S_i(x)$$

stating that the source entity S_i may be aligned to the target entity T_j with the probability $\alpha_{j,i}$. For instance, by relying on Figure 2, we may establish the mappings:

$$\begin{array}{lll} 0.78\ \texttt{Creator}(d,x) & \leftarrow & \texttt{Author}(d,x) \\ 0.22\ \texttt{Creator}(d,x) & \leftarrow & \texttt{Editor}(d,x) \\ 0.90\ \texttt{Journal}(y) & \leftarrow & \texttt{Periodical}(y)\ . \end{array} \qquad (1)$$

The first rule establishes that the probability that the creator x of document d is the also the author of d is 78%, while in the remaining 22% the creator is the editor. The third rule is similar. So, for instance, for the agent's ontology ("Ontology2") and the resource's ontology ("Ontology1"), if the agent's information need is "find a periodical paper whose author is Straccia and the document is about IR", then this request can be represented by the agent by means of the pDatalog rule [3]

$$1.00\ \texttt{Query}(d) \leftarrow \texttt{Periodical}(d), \texttt{Author}(d, \text{“}\texttt{Straccia}''), \texttt{KeyWordSearch}(d, \text{“}\texttt{IR}'')$$

The reformulation of the query with respect to the information resource based on "Ontology1", using the mapping rules, gives us the query:

$$0.702\ \texttt{Query}(d) \leftarrow \texttt{Journal}(d), \texttt{Creator}(d, \text{“}\texttt{Straccia}''), \texttt{KeyWordSearch}(d, \text{“}\texttt{IR}'')$$

(where $0.702 = 0.78 \cdot 0.9$), i.e. find all journal papers created by Straccia and about IR. This is exactly the query to be submitted to the information resource based on "Ontology1". Once the query is submitted to the information resource, for instance using PIRE [16], the degree of relevance of a retrieved document is multiplied with the degree of the corresponding rule. The documents are ranked then according to this final score. Note that without the weight of the rules, we would erroneously give the same preference to documents authored or edited by Straccia, i.e. the weights give more importance to documents matching "author" than those matching "editor" As the example above shows and as already stated previously, once the weighted mappings are established the query reformulation is rather easy. In the following, we will describe how to establish the mappings automatically.

3 oMAP: An Implemented Framework for Automatically Aligning OWL Ontologies

oMAP [23] is a framework whose goal is to automatically align two OWL ontologies, finding the best mappings (together with their weights) between the entities defined in these ontologies. Our approach is inspired by the data exchange problem [11] and borrows from others, like GLUE [6], the idea of using several specialized components for finding the best set of mappings.

We draw in section 3.1 the general picture of our approach. Then, we detail several classifiers used to predict the *weight* of a possible mapping between two entities. These classifiers are terminological (section 3.2) or machine learning-based (section 3.3). Finally, we propose a classifier working on the structure and the formal semantics of the OWL constructs, thus using the meaning of the entities defined in the ontology (section 3.4).

3.1 Overall Strategy

Our goal is to automatically determine "similarity" relationships between classes and properties of two ontologies. For instance, given the ontologies in Figure 2, we would like to determine that an instance of the class `Conference` is likely an instance of the class `Congress`, that the property `creator` should subsume the property `author`, or that the class `Journal` is disjoint from the class `Directions`.

[3] The predicate $\texttt{KeywordSearch}(d, x)$ performs a key word search of key x in document d and gives back the probability that document d is about x.

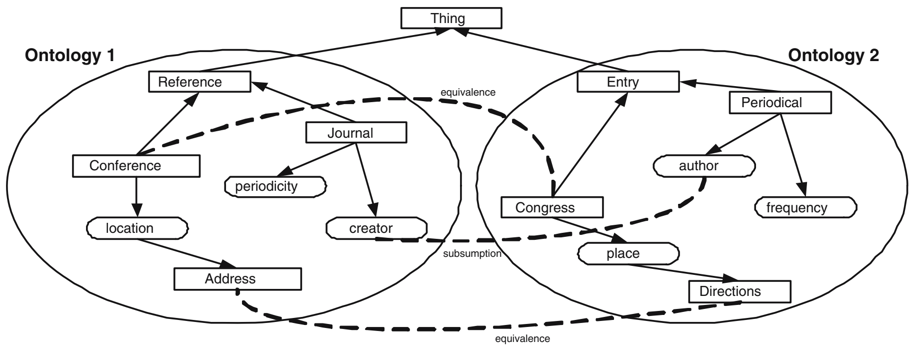

Fig. 2. Excerpt of two bibliographic ontologies and their mappings

Theoretically, an ontology *mapping* is a triple $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ and $\mathbf{T}$ are respectively the source and target ontologies, and Σ is a finite set of *mapping constraints* of the form:

$$\alpha_{i,j}\ T_j \leftarrow S_i$$

where S_i and T_j are respectively the source and target entities. The intended meaning of this rule is that the entity S_i of the source ontology is mapped onto the entity T_j of the target ontology, and the confident measure associated with this mapping is $\alpha_{i,j}$. Note that a source entity may be mapped onto several target entities and conversely. But, we do not require that we have a mapping for every target entity.

Aligning two ontologies in *oMap* consists of three steps:

1. We form a possible Σ, and estimate its quality based on the quality measures for its mapping rules;
2. For each mapping rule $T_j \leftarrow S_i$, we estimate its quality $\alpha_{i,j}$, which also depends on the Σ it belongs to, i.e. $\alpha_{i,j} = w(S_i, T_j, \Sigma)$;
3. As we cannot compute all possible Σ (there are exponentially many) and then choose the best one, we rather build iteratively our final set of mappings Σ using heuristics.

Similar to GLUE [6], we estimate the weight $w(S_i, T_j, \Sigma)$ of a mapping $T_j \leftarrow S_i$ by using different classifiers $CL_1, \ldots, CL_n$. Each classifier CL_k computes a weight $w(S_i, T_j, CL_k)$, which is the classifier's approximation of the rule $T_j \leftarrow S_i$. For each target entity T_j, CL_k provides a rank of the plausible source entities S_{i_k}. Then we rely on a priority list on the classifiers, $CL_1 \prec CL_2 \prec \ldots \prec CL_n$ and proceed as follows: for a given target entity T_j, select the top-ranked mapping of CL_1 if the weight is non-zero. Otherwise, select the top-ranked mapping provided by CL_2 if non-zero, and so on.

In the following we present several classifiers that are currently used in our framework. It is worth noting that some of the classifiers consider the terminological part of the ontologies only, while others are based on their instances (i.e.

the values of the individuals). Finally, we end this section by introducing a new classifier that fully uses the structure and the semantics of ontology definitions and axioms.

3.2 Terminological Classifiers

The terminological classifiers work on the name of the entities (class or property) defined in the ontologies. In OWL, each resource is identified by a URI, and can have some annotation properties attached. Among others, the `rdfs:label` property may be used to provide a human-readable version of a resource's name. Furthermore, multilingual labels are supported using the language tagging facility of RDF literals. In the following, we consider that the name of an entity is given by the value of the `rdfs:label` property or by the URI fragment if this property is not specified.

Same entity names. This binary classifier CL_{SN} returns a weight of 1 if and only if the two classes (or properties) have the same name, and 0 otherwise:

$$w(S_i, T_j, CL_{SN}) = \begin{cases} 1 & \text{if } S_i, T_j \text{ have same name,} \\ 0 & \text{otherwise} \end{cases}$$

Same entity name stems. This binary classifier CL_{SS} returns a weight of 1 if and only if the two classes (or properties) have the same *stem*[4] (for the English text, we use the Porter stemming algorithm [18]), and 0 otherwise:

$$w(S_i, T_j, CL_{SS}) = \begin{cases} 1 & \text{if } S_i, T_j \text{ have same } \textit{stem}, \\ 0 & \text{otherwise} \end{cases}$$

String distance name. This classifier CL_{ED} computes some similarity measures between the entity names (once downcased) such that the Levenshtein distance [15] (or edit distance), which is given by the smallest number of insertions, deletions, and substitutions required to transform one string into the other. The prediction is then computed as:

$$w(S_i, T_j, CL_{ED_1}) = 1 - \frac{dist_{Levenshtein}(S_i, T_j)}{\max(length(S_i), length(T_j))}$$

Another possible variant is:

$$w(S_i, T_j, CL_{ED_2}) = 1/\exp\left(\frac{dist_{Levenshtein}(S_i, T_j)}{|length(S_i) + length(T_j)|}\right)$$

We can then threshold this measure and consider only the mappings $T_j \leftarrow S_i$ such that $w(S_i, T_j, CL_{ED}) \geq 0.9$.

[4] The root of the terms without its prefixes and suffixes.

Iterative substring matching. This classifier CL_{IS} proposed by [22] also considers the commonalities and differences between the two strings but in a more stable and discriminating way. The prediction is computed as:

$$w(S_i, T_j, CL_{IS}) = Comm(S_i, T_j) - Diff(S_i, T_j) + winkler(S_i, T_j)$$

where

- $Comm(S_i, T_j) = \frac{2 \times \sum_i length(maxComSubString_i)}{length(S_i) + length(T_j)}$,
- $Diff(S_i, T_j) = \frac{uLen_{S_i} \times uLen_{T_j}}{p + (1-p) \times (uLen_{S_i} + uLen_{T_j} - uLen_{S_i} \times uLen_{T_j})}$ with $p^5 \in [0, \infty)$, and $uLen_{S_i}$, $uLen_{T_j}$ represents the length of the unmatched substring from the initial strings S_i and T_j scaled with the string length, respectively,
- $winkler(S_i, T_j)$ stands for the improvement of the result using the method introduced by Winkler in [26].

WordNet distance name. This classifier CL_{WN} computes another similarity measure between the entity names using the WordNet®[6] relational dictionary. The prediction is obtained by[7]:

$$w(S_i, T_j, CL_{WN}) = \begin{cases} 1 & \text{if } S_i, T_j \text{ are synonyms,} \\ \max\left(sim\ , \frac{2 * lcs}{length(S_i) + length(T_j)}\right) & \text{otherwise} \end{cases}$$

where

- lcs is the longest common substring between S_i and T_j (also named "substring similarity" in [9]),
- $sim = \frac{|synonym(S_i)| \cap |synonym(T_j)|}{|synonym(S_i)| \cup |synonym(T_j)|}$ where $|synonym(S_i)|$ is the cardinality of the set of all synonyms of S_i.

3.3 Machine Learning-Based Classifiers

An ontology often contains some individuals. It is then possible to use machine learning-based classifiers to predict the weight of a mapping between two entities. In the following, we define, for each instance of an OWL ontology, u as a set of strings obtained by gathering: (*i*) the label for the named individuals, (*ii*) the data value for the datatype properties and (*iii*) the type for the anonymous individuals and the range of the object properties.

For example, using the abstract syntax of [14], let us consider the following individuals :

[5] The parameter p can be adjusted, but the experiments reported in [22] show that the value 0.6 tends to give the best results.

[6] WordNet: `http://wordnet.princeton.edu/`.

[7] Of course, many other WordNet based classifiers exist (or new ones can be developed). Anyway, they can easily be added to oMAP. Their effectiveness will be evaluated in future work.

```
Individual (x1 type (Conference)
      value (label "3rd European Semantic Web Conference")
      value (location x2))
Individual (x2 type (Address)
      value (city "Budva") value (country "Montenegro"))
```

Then, the text gathered u_1 for the named individual x_1 will be (`"3rd European Semantic Web Conference", "Address"`) while u_2 for the anonymous individual x_2 will be (`"Address", "Budva", "Montenegro"`).

We describe in the following typical and well-known classifiers that we used in oMAP: the kNN classifier and the Naive Bayes [20].

kNN classifier. The algorithm of the k-nearest neighbors is based on the calculus of the distances between an unknown form and all the forms of a reference base. It is particularly popular for text classification [20]. In our CL_{kNN} classifier, each class (or property) S_i acts as a category, and training sets are formed from the instances x (which have u as value) of S_i:

$$\mathit{Train} = \bigcup_{i=1}^{s} \{(S_i, x, u) \colon (x, u) \in S_i\}$$

For every instance $y \in T_j$ and its value v, the k-nearest neighbors TOP_k have to be found by ranking the values $(S_i, x, u) \in \mathit{Train}$ according to their similarity $RSV^8(u, v)$. The prediction weights are then computed by summing up the similarity values for all x which are built from S_i, and by averaging these weights $\tilde{w}(y, v, S_i)$ over all instances $y \in T_j$:

$$\begin{aligned}
w(S_i, T_j, CL_{kNN}) &= \frac{1}{|T_j|} \cdot \sum_{(y,v) \in T_j} \tilde{w}(y, v, S_i) \ , \\
\tilde{w}(y, v, S_i) &= \sum_{(S_l,x,u) \in \mathit{TOP}_k, S_i = S_l} RSV(u, v) \ , \\
RSV(u, v) &= \sum_{m \in u \cap v} Pr(m|u) \cdot Pr(m|v) \ , \\
Pr(m|u) &= \frac{\mathit{tf}(m, u)}{\sum_{m' \in u} \mathit{tf}(m', u)} \ , \\
Pr(m|v) &= \frac{\mathit{tf}(w, v)}{\sum_{m' \in v} \mathit{tf}(m', v)}
\end{aligned}$$

Here, $\mathit{tf}(m, u)$ (*resp.* $\mathit{tf}(m, v)$) denotes the number of times the word m appears in the string u (seen as a bag of words).

Naive Bayes text classifier. The classifier CL_{NB} uses a Naive Bayes text classifier [20] for text content. Like the previous one, each class (or property) S_i acts as a category, and training sets are formed from the instances x (which have u as value) of S_i:

[8] The *Retrieval Status Value* is the similarity among two vectors of word, i.e. the sum is the scalar product among the two vectors u and v, i.e. the cosine of the angle among the two vectors.

$$\mathit{Train} = \bigcup_{i=1}^{s} \{(S_i, x, u)\colon (x, u) \in S_i\}$$

For example, the triple $(\texttt{Conference}, x_1, u_1)$ will be considered, where x_1 and u_1 are defined above.

For each $(y, v) \in T_j$, the probability $Pr(S_i|v)$ that the value v should be mapped onto S_i is computed. In a second step, these probabilities are combined by:

$$w(S_i, T_j, CL_{NB}) = \sum_{(y,v) \in T_j} Pr(S_i|v) \cdot Pr(v)$$

Again, we consider the values as bags of words. With $Pr(S_i)$ we denote the probability that a randomly chosen value in $\bigcup_k S_k$ is a value in S_i. If we assume independence of the words in a value, then we obtain:

$$Pr(S_i|v) = Pr(v|S_i) \cdot \frac{Pr(S_i)}{Pr(v)} = \frac{Pr(S_i)}{Pr(v)} \cdot \prod_{m \in v} Pr(m|S_i)$$

Together, the final formula is:

$$w(S_i, T_j, CL_{NB}) = Pr(S_i) \cdot \sum_{(y,v) \in T_j} \prod_{m \in v} Pr(m|S_i)$$

If a word does not appear in the content for any individual in S_i ($Pr(m|S_i) = 0$), we assume a small value to avoid a product of zero.

3.4 A Structural Classifier

Besides these well-known algorithm in information retrieval and text classification, we introduce a new classifier, CL_{Sem}, which is able to use the semantics of the OWL definitions while being guided by their syntax. In other words, this classifier computes similarities between OWL entities by comparing their syntactical definitions. It is used in the framework *a posteriori*. Indeed, we rely on the classifier preference relation $CL_{SN} \prec CL_{SS} \prec CL_{ED_1} \prec CL_{ED_2} \prec CL_{IS} \prec CL_{NB} \prec CL_{kNN}$. According to this preference relation, a set Σ' of mappings is determined. This set is given as input to the structural classifier. Then the structural classifier tries out all alternative ways to extend Σ' by adding some $T_j \leftarrow S_i$ if no mapping related to T_j is present in Σ'. Any extension of Σ' is denoted below by Σ ($\Sigma' \subseteq \Sigma$).

In the following, we note with $w'(S_i, T_j, \Sigma)$ the weight of the mapping $T_j \leftarrow S_i$ estimated by the classifiers of the previous sections, where S_i (*resp.* T_j) is a concept or property name of the source (*resp.* target) ontology. Note that in case the structural classifier is used alone, we set: $w'(S_i, T_j, \Sigma) = 1$. The formal recursive definition of CL_{Sem} is then given by:

1. If S_i and T_j are property names:

$$w(S_i, T_j, \Sigma) = \begin{cases} 0 & \text{if } T_j \leftarrow S_i \notin \Sigma \\ w'(S_i, T_j, \Sigma) & \text{otherwise} \end{cases}$$

2. If S_i and T_j are concept names: let assume that their definitions are $S_i \sqsubseteq C_1 \ldots and \ldots C_m$ and $T_j \sqsubseteq D_1 \ldots and \ldots D_n$, and we note $\mathcal{D} = \mathcal{D}(S_i) \times \mathcal{D}(T_j)$[9], then:

$$w(S_i, T_j, \Sigma) = \begin{cases} 0 & \text{if } T_j \leftarrow S_i \notin \Sigma \\ w'(S_i, T_j, \Sigma) & \text{if } |\mathcal{D}| = 0 \text{ and } T_j \leftarrow S_i \in \Sigma \\ \frac{1}{(|Set|+1)} \cdot \left(w'(S_i, T_j, \Sigma) + \max\limits_{Set} \left(\sum\limits_{(C_i, D_j) \in Set} w(C_i, D_j, \Sigma) \right) \right) & \text{otherwise} \end{cases}$$

3. Let $C_S = (QR.C)$ and $D_T = (Q'R'.D)$, where Q, Q' are quantifiers $\forall$ or $\exists$ or cardinality restrictions, R, R' are property names and C, D are concept expressions, then:

$$w(C_S, D_T, \Sigma) = w_Q(Q, Q') \cdot w(R, R', \Sigma) \cdot w(C, D, \Sigma)$$

4. Let $C_S = (op\ C_1 \ldots C_m)$ and $D_T = (op'\ D_1 \ldots D_m)$, where the concept constructors op, op' in the concepts C_S, D_T are in prefix notation, op, op' are the concept constructors among $\sqcap, \sqcup, \neg$ and $n, m \geq 1$, then:

$$w(C_S, D_T, \Sigma) = w_{op}(op, op') \cdot \frac{\max\limits_{Set} \left(\sum\limits_{(C_i, D_j) \in Set} w(C_i, D_j, \Sigma) \right)}{\min(m, n)}$$

where:

– $Set \subseteq \{C_1 \ldots C_m\} \times \{D_1 \ldots D_n\}$ and $|Set| = \min(m, n)$,
– $(C, D) \in Set, (C', D') \in Set \Rightarrow C \neq C', D \neq D'$.

We give in the Table 1 the values for w_Q and w_{op} we used.

Table 1. Possible values for w_{op} and w_Q weights

w_{op} is given by:

	$\sqcap$	$\sqcup$	$\neg$
$\sqcap$	1	1/4	0
$\sqcup$		1	0
$\neg$			1

w_Q is given by:

	$\exists$	$\forall$
$\exists$	1	1/4
$\forall$		1

	$\leq n$	$\geq n$
$\leq m$	1	1/3
$\geq m$		1

4 Evaluation

The problem of aligning ontologies has already produced some interesting works. However, it is difficult to compare theoretically the various approaches proposed since they base on different techniques. Hence, it is necessary to compare them on common tests. This is the goal of the Ontology Alignment Evaluation Initiative (OAEI[10]) since two years, who set up contests and benchmark tests for assessing

[9] $\mathcal{D}(S_i)$ represents the set of direct (immediate) parent concepts of S_i.
[10] `http://oaei.inrialpes.fr`

the strengths and weakness of the available tools. We have thoroughly evaluated *oMAP* with the data of the OAEI 2005 campaign [24]. We present below the updated results of our new approach with respect to the other competitors of this contest for two different scenarios: systematic benchmark tests based on bibliography data (section 4.1), and the alignment of three large web directories (Google, Looksmart and Yahoo) which fits perfectly with our web distributed search scenario (section 4.2).

oMAP is freely available at: `http://homepages.cwi.nl/~troncy/oMAP` and all these results can be reproduced.

4.1 Aligning Bibliographic Data: The OAEI Benchmarks

The *benchmarks tests* are systematic benchmarks series produced for identifying the areas in which each alignment algorithm is strong and weak. Taking back the tests of the 2004 contest [25] and extending them, there are based on one particular ontology dedicated to the very narrow domain of bibliography and a number of alternative ontologies of the same domain for which alignments are provided. The overall score of *oMAP* for this task is quite good (see Table 2, the results of the other competitors are based on [1]).

Table 2. Overall results for the OAEI benchmark tests, oMAP is given in the last column

algo	**edna**		**Falcon**		**FOAM**		**ctxMatch**		**Dublin20**		**CMS**		**OLA**		**oMAP**	
Test	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
1xx	0.96	1.00	1.00	1.00	0.98	0.65	0.87	0.34	1.00	0.99	0.74	0.20	1.00	1.00	**1.00**	**1.00**
2xx	0.41	0.56	0.90	0.89	0.89	0.69	0.72	0.19	0.94	0.71	0.81	0.18	0.80	0.73	**0.80**	**0.63**
3xx	0.47	0.82	0.93	0.83	0.92	0.69	0.37	0.06	0.67	0.60	0.93	0.18	0.50	0.48	**0.93**	**0.65**
H-mean	0.45	0.61	0.91	0.89	0.90	0.69	0.72	0.20	0.92	0.72	0.81	0.18	0.80	0.74	**0.85**	**0.68**

However, *oMAP* has poor performance for the tests 25x and 26x where all labels are replaced with random strings. Actually, the terminological and machine-learning based classifiers give wrong input to our structural classifier. This classifier is then not able to counterbalance this effect and give also wrong alignments. It is the typical case where the other classifiers should be turn off and the structural classifier should work alone. In this specific case, the computing time increases but the performances are much better.

4.2 Aligning Web Categories

The *directory real world case* consists of aligning web sites directory using the large dataset developing in [2]. These tests are blind in the sense that the expected alignments are not known in advance. As explained in [1], only recall results are available. The results for web directory matching task are presented on the Table 3 (the results of the other competitors are based on [1]). The web directories matching task is a very hard one, since the best systems found

Table 3. Overall results for the web categories alignment, oMAP is given in the last column

Falcon	**FOAM**	**ctxMatch**	**Dublin20**	**CMS**	**OLA**	**oMAP**
31.17%	11.88%	9.36%	26.53%	14.08%	31.96%	**34.43**

about 30% of mappings form the dataset (i.e. have Recall about 30%). *oMAP* gives already good results but a complete analysis of them should provide some improvements in a very near future.

5 Related Work and Conclusion

In this paper, we have proposed a three-step framework for distributed, Semantic Web-enabled Information Retrieval. For the second step, namely *query reformulation and ontology alignment*, we have described *oMAP*, an efficient tool for automatically aligning OWL ontologies, whereas the first step, namely *resource selection*, will be addressed in future work. *oMAP* uses different classifiers to estimate the quality of a mapping. Novel are the use of machine learning-based classifiers and a classifier which uses the structure of the OWL constructs and thus the semantics of the entities defined in the ontologies. We have implemented the whole framework and evaluated it on the OAEI benchmark tests with respect to the other competitors.

The alignment problem for ontologies, as well as the matching problem for schemas, has been addressed by many researchers so far and are strictly related. Some of the techniques applied in schema matching can be applied to ontology alignment as well, taking additionally into account the formal semantics carried out by the taxonomies of concepts and properties and the axioms of the ontology. Among the works related to ontology alignment, FOAM [7, 8] propose to combine different similarity measures from pre-existing hand-established mapping rules. Besides the validity of these rules could be generally put into question, this method suffers from not being fully automatic. [17] has developed an interesting approach: from anchor-pairs of concepts that seem to be close (discovered automatically or proposed manually), their *hors-context* similarity are computed analyzing the paths in the taxonomy that link the pairs of concepts. This method has been implemented into the Anchor-Prompt tool which has, until now, one of the best performance. [10] have adapted works on similarity calculus for object-based knowledge representation languages to the Semantic Web languages. A global similarity measure taking into account all the features of the OWL-Lite language has been proposed, capable to treat both the circular definitions and the collections. For a complete state of the art on the numerous ontology alignment approaches proposed, see [5, 21].

Our future work will concentrate on the major issue left out so far, namely automated resource selection. To this end, we plan to extend methods for query-based sampling and automated resource selection from the textual IR resources to ontology-based information resources. Furthermore, the *oMAP* framework could still be improved. The combination of a rule-based language with an expressive ontology language like OWL has attracted the attention of many researchers [13] and is considered now as an important requirement. Taking into account this additional semantics of the ontologies appear thus necessary. Additional classifiers using more terminological resources can be included in the framework, while the effectiveness of the machine learning part could be improved using other measures like the KL-distance. While to fit new classifiers into our model is straightforward theoretically, practically finding out the most appropriate one or a combination of them is quite more difficult. In the future, more variants should be developed and evaluated to improve the overall quality of *oMAP*.

Acknowledgments

The authors wish to thank in particular Jacco van Ossenbruggen and Lynda Hardman for insightful discussions and helpful comments.

References

1. B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors. *K-CAP 2005 Workshop on Integrating Ontologies (IntOnt'05)*, Banff, Canada, 2005.
2. P. Avesani, F. Giunchiglia, and M. Yatskevich. A Large Scale Taxonomy Mapping Evaluation. In *4th International Semantic Web Conference (ISWC'05)*, pages 67–81, Galway, Ireland, 2005.
3. J. Callan. Distributed Information Retrieval. In W.B. Croft, editor, *Advances in Information Retrieval*, pages 127–150. Kluwer Academic, 2000.
4. J. Callan and M. Connell. Query-Based Sampling of Text Databases. *ACM Transactions on Information Systems*, 19(2):97–130, 2001.
5. KW Consortium. State of the Art on Ontology Alignment. Deliverable Knowledge Web 2.2.3, FP6-507482, 2004.
6. A. Doan, J. Madhavan, R. Dhamankar, P. Domingos, and A. Halevy. Learning to Match Ontologies on the Semantic Web. *The VLDB Journal*, 12(4):303–319, 2003.
7. M. Ehrig and S. Staab. QOM - quick ontology mapping. In *3rd International Semantic Web Conference (ISWC'04)*, pages 683–697, Hiroshima, Japon, 2004.
8. M. Ehrig, S. Staab, and Y. Sure. Bootstrapping Ontology Alignment Methods with APFEL. In *4th International Semantic Web Conference (ISWC'05)*, pages 186–200, Galway, Ireland, 2005.
9. J. Euzenat. An API for ontology alignment. In *3rd International Semantic Web Conference (ISWC'04)*, pages 698–712, Hiroshima, Japon, 2004.
10. J. Euzenat and P. Valtchev. Similarity-based ontology alignment in OWL-Lite. In *15th European Conference on Artificial Intelligence (ECAI'04)*, pages 333–337, Valence, Spain, 2004.
11. R. Fagin, P.G. Kolaitis, R.J. Miler, and L. Popa. Data Exchange: Semantics and Query Answering. In *9th International Conference on Database Theory (ICDT'03)*, pages 207–224, Sienne, Italie, 2003.
12. N. Fuhr. Probabilistic Datalog: Implementing Logical Information Retrieval for Advanced Applications. *Journal of the American Society for Information Science*, 51(2):95–110, 2000.

13. I. Horrocks and P.F. Patel-Schneider. A proposal for an OWL rules language. In *13th International World Wide Web Conference (WWW'04)*, pages 723–731, 2004.
14. I. Horrocks, P.F. Patel-Schneider, and F. van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.
15. V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals (Russian). *Doklady Akademii Nauk SSSR*, 163(4):845–848, 1965. English translation in Soviet Physics Doklady, **10**(8):707–710, 1966.
16. H. Nottelmann and U. Straccia. sPLMap: A probabilistic approach to schema matching. In *27th European Conference on Information Retrieval (ECIR'05)*, pages 81–95, Santiago de Compostela, Spain, 2005.
17. N.F. Noy and M.A. Musen. Anchor-PROMPT: Using non-local context for semantic matching. In *Workshop on Ontologies and Information Sharing at IJCAI'01*, Seattle, Washington, USA, 2001.
18. M.F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
19. M.E. Renda and U. Straccia. Web Metasearch: Rank vs. Score Based Rank Aggregation Methods. In *18th Annual ACM Symposium on Applied Computing (SAC'03)*, pages 841–846, Melbourne, Florida, USA, 2003.
20. F. Sebastiani. Machine learning in automated text categorization. *ACM Comuting Surveys*, 34(1):1–47, 2002.
21. P. Shvaiko and J. Euzenat. A Survey of Shema-based Matching Approaches. *Journal on Data Semantics (JoDS)*, 2005.
22. G. Stoilos, G. Stamou, and S. Kollias. A String Metric for Ontology Alignment. In *4th International Semantic Web Conference (ISWC'05)*, pages 624–637, Galway, Ireland, 2005.
23. U. Straccia and R. Troncy. oMAP: Combining Classifiers for Aligning Automatically OWL Ontologies. In *6th International Conference on Web Information Systems Engineering (WISE'05)*, pages 133–147, New York City, New York, USA, November, 20-22 2005.
24. U. Straccia and R. Troncy. oMAP: Results of the Ontology Alignment Contest. In *Workshop on Integrating Ontologies*, pages 92–96, Banff, Canada, 2005.
25. Y. Sure, O. Corcho, J. Euzenat, and T. Hughes, editors. *3rd International Workshop on Evaluation of Ontology-based Tools (EON'04)*, Hiroshima, Japon, 2004.
26. W. Winkler. The state record linkage and current research problems. Technical report, Statistics of Income Division, Internal Revenue Service Publication, 1999.
27. C. Yu, W. Meng, King-Lup, W. Wu, and N. Rishe. Efficient and Effective Metasearch for a Large Number of Text Databases. In *8th International Conference on Information and Knowledge Management (CIKM'99)*, pages 217–224, Kansas City, Missouri, USA, 1999.