A Tool for Evaluating Ontology Alignment Strategies

Patrick Lambrix and He Tan
Department of Computer and Information Science
Linköpings universitet, Sweden
{patla,hetan}@ida.liu.se

This is a pre-print version of the article published in the Journal of Data Semantics VIII:182-202, 2007. LNCS 4380.

Abstract

Ontologies are an important technology for the Semantic Web. In different areas ontologies have already been developed and many of these ontologies contain overlapping information. Often we would therefore want to be able to use multiple ontologies. To obtain good results, we need to find the relationships between terms in the different ontologies, i.e. we need to align them. Currently, there exist a number of systems that support users in aligning ontologies, but not many comparative evaluations have been performed and there exists little support to perform such evaluations. However, the study of the properties, the evaluation and comparison of the alignment strategies and their combinations, would give us valuable insight in how the strategies could be used in the best way. In this paper we propose the KitAMO framework for comparative evaluation of ontology alignment strategies and their combinations and present our current implementation. We evaluate the implementation with respect to performance. We also illustrate how the system can be used to evaluate and compare alignment strategies and their combinations in terms of performance and quality of the proposed alignments. Further, we show how the results can be analyzed to obtain deeper insights into the properties of the strategies.

1 Introduction

Intuitively, ontologies (e.g. [22, 14]) can be seen as defining the basic terms and relations of a domain of interest, as well as the rules for combining these terms and relations. They are considered to be an important technology for the Semantic Web. Ontologies are used for communication between people and organizations by providing a common terminology over a domain. They provide the basis for interoperability between systems. They can be used for making

the content in information sources explicit and serve as an index to a repository of information. Further, they can be used as a basis for integration of information sources and as a query model for information sources. They also support clearly separating domain knowledge from application-based knowledge as well as validation of data sources. The benefits of using ontologies include reuse, sharing and portability of knowledge across platforms, and improved maintainability, documentation, maintenance, and reliability. Overall, ontologies lead to a better understanding of a field and to more effective and efficient handling of information in that field. In the field of bioinformatics, for instance, the work on ontologies is recognized as essential in some of the grand challenges of genomics research [4] and there is much international research cooperation for the development of ontologies (e.g. the Gene Ontology (GO) [13] and Open Biomedical Ontologies (OBO) [33] efforts) and the use of ontologies for the Semantic Web (e.g. the EU Network of Excellence REWERSE [36, 37]).

Many ontologies have already been developed and many of these ontologies contain overlapping information. Often we would therefore want to be able to use multiple ontologies. For instance, companies may want to use community standard ontologies and use them together with company-specific ontologies. Applications may need to use ontologies from different areas or from different views on one area. Ontology builders may want to use already existing ontologies as the basis for the creation of new ontologies by extending the existing ontologies or by combining knowledge from different smaller ontologies. In each of these cases it is important to know the relationships between the terms in the different ontologies. We say that we align two ontologies when we define the relationships between terms in the different ontologies. We merge two ontologies when we, based on the alignment relations between the ontologies, create a new ontology containing the knowledge included in the source ontologies.

Ontology alignment and merging is recognized as an important step in ontology engineering that needs more extensive research (e.g. [34]). Currently, there exist a number of systems that support users in aligning or merging ontologies in the same domain. These systems use different techniques, but it is not clear how well these techniques perform for different types of ontologies. Also, it is not clear whether and how different techniques could be combined to provide better alignments. The study of the properties, the evaluation and comparison of the alignment strategies and their combinations, would give us valuable insight in how the strategies could be used in the best way. It would also lead to recommendations on how to improve the alignment techniques. However, relatively few comparative evaluations on ontology merging and alignment have been performed [23, 24, 25, 26, 34, 10, 17] and no advanced tools for supporting these kinds of evaluations exist yet [20]. To be able to study the properties of the alignment techniques and their combinations and to compare them, we need tools that allow us to evaluate them in different settings. Such tools should allow us to apply the techniques and different combinations of techniques to different types of ontologies. The tools should also support evaluation and comparison of the techniques and their combinations in terms of e.g. performance and quality of the alignment. Further, we need support to analyze the evaluation results in

different ways.

In this paper we propose a tool for evaluating ontology alignment strategies and their combinations. The tool covers the non-interactive part of the general framework for aligning ontologies as described in [25]. In section 3 we first describe the KitAMO[1] framework for evaluating ontology alignment strategies and then describe the current implementation. In section 4 the implementation is evaluated with respect to performance. We also show how the tool can be used to evaluate and compare strategies and their combinations in terms of performance and quality of the proposed alignment relationships. Further, we show how the results can be analyzed to examine the advantages and disadvantages of the strategies in more details. Related work is discussed in section 5 and the paper concludes in section 6. In the next section we provide some background on (biomedical) ontologies, ontology alignment systems and evaluations of ontology alignment strategies.

2 Background

2.1 Ontologies

Ontologies differ regarding the kind of information they can represent. From a knowledge representation point of view ontologies can have the following components (e.g. [22, 38]). Concepts represent sets or classes of entities in a domain. Instances represent the actual entities. They are, however, often not represented in ontologies. Further, there are many types of relations. Finally, axioms represent facts that are always true in the topic area of the ontology. These can be such things as domain restrictions, cardinality restrictions or disjointness restrictions. Depending on which of the components are represented and the kind of information that can be represented, we can distinguish between different kinds of ontologies such as controlled vocabularies, taxonomies, thesauri, data models, frame-based ontologies and knowledge-based ontologies. These different types of ontologies can be represented in a spectrum of representation formalisms ranging from very informal to strictly formal. For instance, some of the most expressive representation formalisms in use for ontologies are description logic-based languages such as DAML+OIL and OWL.

2.2 Biomedical ontologies

In this paper we have chosen to use test cases based on biomedical ontologies (e.g. [22]). There are several reasons for this. Research in biomedical ontologies is recognized as essential in some of the grand challenges of genomics research [4]. The field has also matured enough to develop standardization efforts. An example of this is the organization of the first conference on Standards and Ontologies for Functional Genomics (SOFG) in 2002 and the development of the SOFG resource on ontologies. Further, there exist ontologies that have reached

[1] toolKit for Aligning and Merging Ontologies.

the status of de facto standard and are being used extensively for annotation of databases. Also, OBO was started as an umbrella web address for ontologies for use within the genomics and proteomics domains. Many biomedical ontologies are already available via OBO. There are also many overlapping ontologies available in the field.

The ontologies that we use in this paper are GO ontologies, Signal-Ontology (SigO) [43], Medical Subject Headings (MeSH) [28] and the Adult Mouse Anatomical Dictionary (MA) [16]. The GO Consortium is a joint project which goal is to produce a structured, precisely defined, common and dynamic controlled vocabulary that describes the roles of genes and proteins in all organisms. Currently, there are three independent ontologies publicly available over the Internet: biological process, molecular function and cellular component. The GO ontologies are a de facto standard and many different bio-databases are today annotated with GO terms. The terms in GO are arranged as nodes in a directed acyclic graph, where multiple inheritance is allowed. The purpose of the SigO project is to extract common features of cell signaling in the model organisms, try to understand what cell signaling is and how cell signaling systems can be modeled. SigO is based on the knowledge of the Cell Signaling Networks data source [41] and treats complex knowledge of living cells such as pathways, networks and causal relationships among molecules. The ontology consists of a flow diagram of signal transduction and a conceptual hierarchy of biochemical attributes of signaling molecules. MeSH is a controlled vocabulary produced by the American National Library of Medicine and used for indexing, cataloging, and searching for biomedical and health-related information and documents. It consists of sets of terms naming descriptors in a hierarchical structure. These descriptors are organized in 15 categories, such as the category for anatomic terms, which is the category we use. The purpose of MA is to provide an ontology for annotating and integrating different types of data pertinent to anatomy. It is based on the Mouse Embryo Anatomy Nomenclature Database [2] and will be integrated with the Anatomical Dictionary for Mouse Development to generate an anatomy ontology covering the entire lifespan of the laboratory mouse. The ontology contains more than 2400 anatomical terms. They are structured as directed acyclic graphs across *is-a* and *part-of* relationships. The hierarchy of the ontology is organized in both spatial and functional ways.

2.3 Ontology alignment systems

There exist a number of ontology alignment systems that support users in finding inter-ontology relationships. Some of these systems are also ontology merge systems. Many ontology alignment systems can be described as instantiations of the general framework defined in [25, 26] (figure 1).

In our framework an alignment system receives as input two source ontologies. The system can be seen as being composed of two major parts. The first part (*I* in figure 1) computes alignment suggestions. The second part (*II*) interacts with the user to decide on the final alignments.

An alignment system can include several matchers. These matchers calcu-

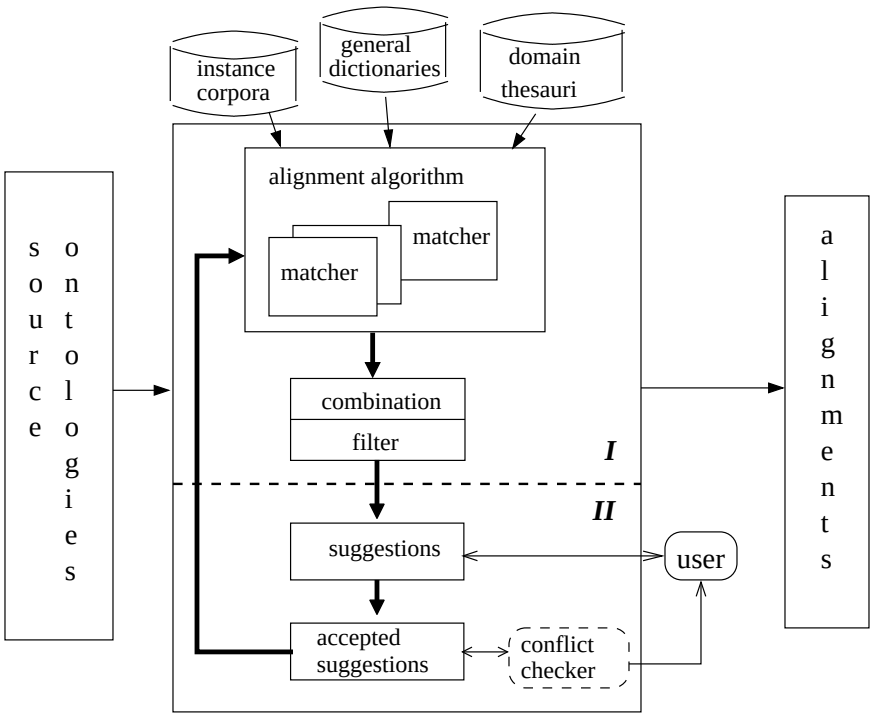

Figure 1: A general alignment strategy [25].

late similarities between the terms from the different source ontologies. The matchers can implement strategies based on linguistic matching, structure-based strategies, constraint-based approaches, instance-based strategies, strategies that use auxiliary information or a combination of these. Strategies based on linguistic matching make use of textual descriptions of the concepts and relations such as names, synonyms and definitions. The similarity measure between concepts is based on comparisons of the textual descriptions. Structure-based strategies use the structure of the ontologies to provide suggestions. For instance, previously accepted alignments can be used to influence the similarity values between the sub- and super-concepts of already aligned concepts. Other approaches use paths between already aligned concepts to generate new alignment suggestions. In the constraint-based approaches the axioms are used to provide suggestions. For instance, knowing that the range and domain of two relations are the same, may be an indication that there is a relationship between the relations. On their own these approaches may not be sufficient to provide high quality suggestions, but they may complement other approaches to reduce the number of irrelevant suggestions. In some cases instances are available directly or can be obtained. When instances are available, they may be used in defining similarities between concepts. Further, dictionaries and thesauri representing general or domain knowledge, or intermediate ontologies may be used to enhance the alignment process. Table 1 gives an overview of the used strategies per alignment system. For more information we refer to [26].

Alignment suggestions are then determined by combining and filtering the results generated by one or more matchers. Although most systems combine different approaches, not much research is done on the applicability and performance of these combinations. In the current systems similarity values are often combined using a weighted sum. In most systems the filtering consists of retaining the pairs of terms with a similarity value above a certain threshold as alignment suggestions. Recently, some more advanced filtering methods are proposed, such as in [3] where the structure of the ontologies is used to filter out alignment suggestions. By using different matchers and combining them and filtering in different ways we obtain different alignment strategies.

The interactive component of the alignment system presents the suggestions to the user who accepts or rejects them. The acceptance and rejection of a suggestion may influence further suggestions. Also, some matchers (e.g. some structural matchers as in [32, 24]) require as input already accepted suggestions. Further, a conflict checker is used to avoid conflicts introduced by the alignment relationships. The output of the alignment algorithm is a set of alignment relationships between terms from the source ontologies.

Figure 2 shows a simple merging algorithm. A new ontology is computed from the source ontologies and their identified alignment. The checker is used to avoid conflicts as well as to detect unsatisfiable concepts and, if so desired by the user, to remove redundancy.

	linguistic	structure	constraints	instances	auxiliary
ArtGen [30]	name	parents, children		domain-specific documents	WordNet
ASCO [27]	name, label, description	parents, children, siblings, path from root			WordNet
Chimaera [29]	name	parents, children			
FCA-Merge [39]	name			domain-specific documents	
FOAM [12, 6]	name, label	parents, children	equivalence		
GLUE [5]	name	neighborhood		instances	
HCONE [21]	name	parents, children			WordNet
IF-Map [19]				instances	a reference ontology
iMapper [40]		leaf, non-leaf, children, related node	domain, range	instances	WordNet
OntoMapper [35]	name	parents, children		documents	
(Anchor-) PROMPT [32]	name	direct graphs			
SAMBO [24, 25, 26]	name, synonym	is-a and part-of, descendants and ancestors		domain-specific documents	WordNet, UMLS
S-Match [15]	label	path from root	semantic relations codified in labels		WordNet

Table 1: Strategies used by alignment systems [26].

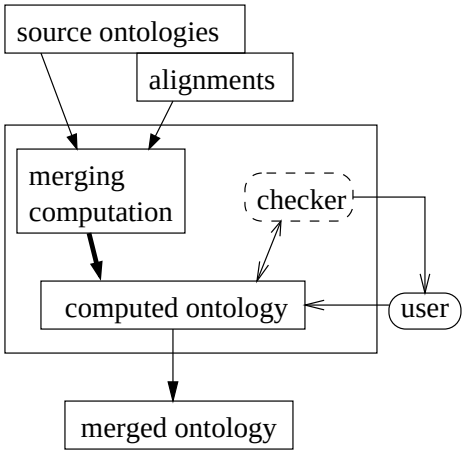

Figure 2: A general merging algorithm [25].

2.4 Evaluation of ontology alignment strategies

To date comparative evaluations of ontology alignment and merge systems have been performed by some groups ([34], [23, 24, 25, 26] and the EON and I3CON contests). The EU OntoWeb project [34] evaluated the systems PROMPT [31] based on Protégé (with extension Anchor-PROMPT [32]), Chimaera [29] (described, not evaluated), FCA-Merge [39] and ODEMerge. This evaluation focused on such things as functionality, interoperability and visualization, but did not include tests on the quality of the alignment. In [23, 24, 26] PROMPT, Chimaera, FOAM and an early version of SAMBO were evaluated in terms of the quality of the alignment as well as the time it takes to align ontologies with these tools. Different alignment algorithms and their combinations were evaluated in [25, 26]. The test cases were biomedical ontologies and ontologies about academia.

In 2004, two different experiments for the evaluation of the alignment tools were launched: the ontology alignment contest held by the International Workshop on the Evaluation of Ontology-based Tools (EON) [10] and the evaluation of ontology alignment tools organized by the Information Interpretation and Integration Conference (I3CON) [17]. Their main goals were to show how it is possible to evaluate ontology alignment tools and provide a framework for the evaluation of the alignment tools. In 2005 EON and I3CON organized a unique evaluation campaign. Its outcome is presented in [9]. In this experiment there were 7 participants. The participants were provided pairs of ontologies (OWL) and their expected results (RDF/XML). The participants submitted to the organizers their best alignment results which were generated under the same set of parameters. The alignment algorithms were to be performed without intervention. The test cases were from three topics, including bibliographic ontologies,

ontologies constructed from Google, Yahoo and Looksmart web directories, and anatomy models FMA and OpenGalen. Not all participants finished all these tests. The organizers evaluated the results submitted by the participants and compared them. The evaluation measures were precision and recall.

3 KitAMO

In this section we present the KitAMO framework for evaluating ontology alignment strategies and present the current implementation. KitAMO supports the study, evaluation and comparison of alignment strategies and their combinations based on their performance and the quality of their alignments on test cases. This corresponds to the evaluation of the non-interactive alignment components (part *I* in figure 1) in an ontology alignment system. KitAMO also provides support for the analysis of the evaluation results.

3.1 Framework

Figure 3 illustrates the KitAMO framework for comparative evaluation of the different alignment components. KitAMO receives as input different alignment components that we want to evaluate, e.g. various matchers, filters and combination algorithms. KitAMO contains a database of evaluation cases which is built in advance. Each case consists of two ontologies and their expected alignments produced by experts on the topic area of the ontologies. The alignment components are evaluated using these cases.

The evaluation tool in the framework provides the wrapper which allows the alignment components to work on the ontologies in the database of evaluation cases, and provides the interface where the user can decide, e.g. which evaluation cases are used, and how these alignment components cooperate. The evaluation tool also has the responsibility to save the similarity values generated by the alignment components to the similarity database, and retrieves these similarity values from the database when required by the analysis tool.

The analysis tool receives as input data from the database of evaluation cases, similarity values retrieved by the evaluation tool from the similarity database, and possibly previously generated data from the analysis database. The analysis tool allows a user to analyze different properties of the evaluated alignment components and their combinations. For instance, it is possible to analyze such things as the similarity values between terms from different matchers, the performance of the matchers, and the quality of the alignment suggestions generated by different matchers and their combinations with different filters. Through the analysis tool the user can also save the evaluation results into the analysis database and produce an evaluation report.

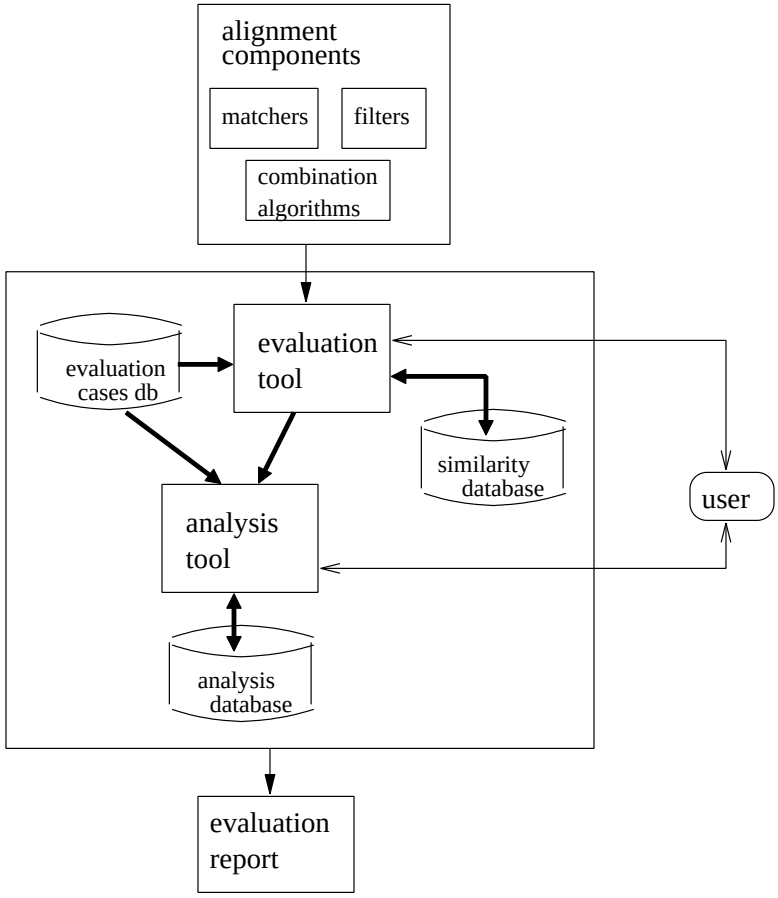

Figure 3: The KitAMO framework.

3.2 Implementation

In the current implementation of KitAMO we have focused on the evaluation of matchers. Instead of allowing different combination and filtering strategies as input, currently we implemented the most used strategies in KitAMO, i.e. a weighted sum as combination strategy and filtering based on a threshold value.

The matchers are added to KitAMO as plug-ins. Each matcher needs to implement the plug-in interface where similarity values between terms in ontologies are computed. When new matchers are added, the system is restarted in order to pick up the new plug-ins, and to take the change in configuration into account.

The current database of evaluation cases consists of five test cases based on two groups of biomedical ontologies. These cases were previously used in the evaluations in [25, 26, 42]. For the first two cases we use a part of a GO ontology together with a part of SigO. The first case, *behavior* (B), contains 57 terms from GO and 10 terms from SigO. The second case, *immune defense* (ID), contains 73 terms from GO and 15 terms from SigO. We used more terms from GO than from SigO because the granularity of GO is higher than the granularity of SigO for these topics. The other cases are based on MeSH (anatomy category) and MA. The three cases used in our test are: *nose* (containing 15 terms from MeSH and 18 terms from MA), *ear* (containing 39 terms from MeSH and 77 terms from MA), and *eye* (containing 45 terms from MeSH and 112 terms from MA). We translated the ontologies from the GO flat file format to OWL retaining identifiers, names, synonyms, definitions and is-a and part-of relationships. The alignments for these test cases were provided to us by biologists. In this implementation of the database we only considered equivalence and is-a relations between terms as alignment relationships. For each case we also stored the expected suggestions and the inferred suggestions. The expected suggestions is the minimal set of alignment suggestions that matchers are expected to generate for a perfect recall. This set does not include the inferred suggestions. Inferred suggestions can be inferred by a merging algorithm. An example of an inferred suggestion is that incus is-a ear ossicle. In this case we know that auditory bone (MA) is equivalent to ear ossicle (MeSH), and incus is-a auditory bone in MA. Then the system should derive that incus is-a ear ossicle.

The user starts the evaluation process by choosing an evaluation case. Then the user decides which matchers should be used in the evaluation from the list of matcher plug-ins configured in KitAMO. For instance, figure 4 shows that we have 4 matcher plug-ins (UMLSKSearch, TermWN, TermBasic and BayesLearning) and that we decided to perform the evaluations on the first two matchers. The selected matchers calculate similarity values between the terms in the chosen evaluation case, and the results are written to the similarity database. For the combination each matcher can be assigned a weight (weight in figure 5). The similarity values generated by the combination, i.e. the weighted sum, can also be saved to the similarity database by the user. For the filter the user can assign threshold values for individual matchers and the combination (threshold in figure 5).

Figure 4: The list of matcher plug-ins.

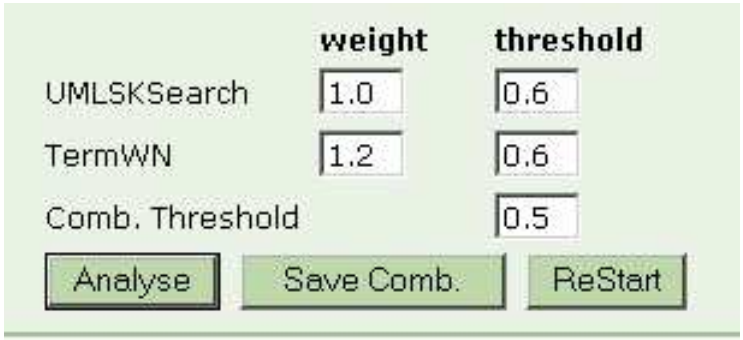

Figure 5: The weights and thresholds assignment.

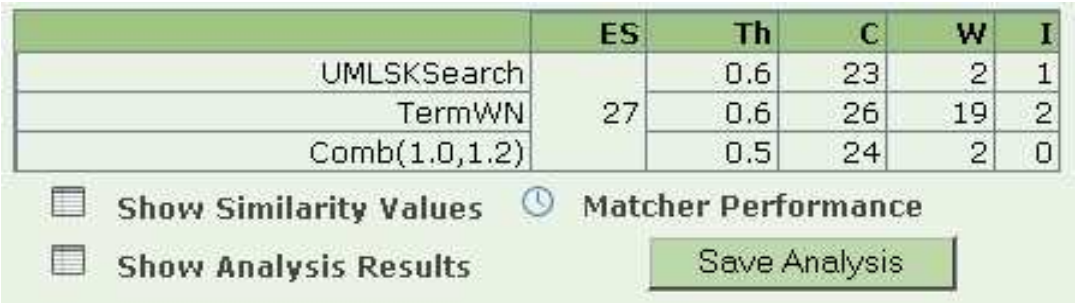

Figure 6: The analysis result.

KitAMO shows the result of an evaluation for a group of weights and thresholds in the form of a table as illustrated in figure 6. In the example we see that the number of expected suggestions (ES) is 27 for the evaluation case. UMLSKSearch found 23 correct alignments, 2 wrong suggestions and 1 inferred suggestion for the threshold 0.6. For the combination with weight 1.0 for UMLSKSearch and weight 1.2 for TermWN we found 24 correct suggestions, 2 wrong suggestions and no inferred suggestions for threshold 0.5. The user can save this data to the analysis database and at any time the user can look at previously saved data (figure 7). The table with previously saved data can be sorted according to each column. The user can also look at the actual similarity values between the pairs of terms in the ontologies of the evaluation case. For instance, figure 8 shows a table with the terms in the ontologies together with the similarity values generated by the analyzed matchers and combinations. It also shows whether the pair is a correct alignment, an inferred suggestion or a wrong suggestion. The table can be sorted according to each column. Further, the user can look at the time needed by the matchers for the computation of similarity values as illustrated in figure 9.

The user can always restart the evaluation process with a different group of matchers or with different combinations and thresholds. Finally, the user can export the similarity and analysis data to Excel files.

KitAMO is implemented in Java. It relies on the Jena ontology API [18] for parsing OWL files. MySQL is used for the databases in KitAMO.

matcher	Th	C	W	I
(1.0UM,1.0TW)	0.50	23	2	0
(1.0UM,1.2TW)	0.50	24	2	0
TermWN	0.40	26	110	19
TermWN	0.50	26	65	8
TermWN	0.60	26	19	2
UMLSKSearch	0.40	23	2	1
UMLSKSearch	0.50	23	2	1
UMLSKSearch	0.60	23	2	1

Figure 7: The previously saved analysis results.

4 Evaluation and discussion

In this section we evaluate the performance of the system using our test cases. This gives us an indication of the extra amount of time that KitAMO needs to process the alignment evaluations. This extra amount of time should be compared to the time it takes to manually analyze the similarity results generated by different matchers. We also give an example use of the system.

MA	MeSH	UMLSKSearch	TermWN	(1.0UM,1.2TW)	Sug
basilar membrane	basilar membrane	1.0000	1.0000	1.0000	C
tectorial membrane	tectorial membrane	1.0000	1.0000	1.0000	C
stapedius	stapedius	1.0000	1.0000	1.0000	C
scala tympani	scala tympani	1.0000	1.0000	1.0000	C
vestibular aqueduct	vestibular aqueduct	1.0000	1.0000	1.0000	C
utricle	saccule and utricle	1.0000	1.0000	1.0000	W
tensor tympani	tensor tympani	1.0000	1.0000	1.0000	C
middle ear	middle ear	1.0000	1.0000	1.0000	C
ear	ear	1.0000	1.0000	1.0000	C
spiral organ	organ of corti	1.0000	1.0000	1.0000	C
tympanic membrane	tympanic membrane	1.0000	1.0000	1.0000	C
auditory bone	ear ossicle	1.0000	1.0000	1.0000	C
cochlea	cochlea	1.0000	1.0000	1.0000	C
saccule	saccule and utricle	1.0000	1.0000	1.0000	W
incus	incus	1.0000	1.0000	1.0000	C

Figure 8: The similarity table.

matchers	Performance (s)
TermWN	41.156
UMLSKSearch	137.798

Figure 9: The performance table.

Case	I/O	Setup	Analysis
B	2.4	2.4	0.3
ID	2.6	4.4	0.3
nose	2.9	1.7	0.4
ear	3.0	8.3	0.7
eye	3.4	14.5	1.1

Table 2: Time for evaluation (in seconds).

Case	TermBasic	TermWN	UMLSKSearch	BayesLearning
B	0.7	11.0	33.6	50.9
ID	3.0	37.0	37.6	90.6
nose	0.6	7.3	44.1	24.5
ear	3.8	37.9	105.8	114.2
eye	8.0	60.4	132.2	173.1

Table 3: Average time for computation of similarity values based on 5 runs (in seconds).

4.1 Performance of the system

We have run KitAMO using our test cases on a PC with 128Mb memory and an AMD Athlon 64 processor. We divide the time needed for an evaluation task into four parts. The first part includes the time for loading the ontologies and for generating the final evaluation reports as output (I/O in table 2). The second part is the time needed by each matcher to calculate the similarity values (table 3). This is an inherent property of the matchers and is outside KitAMO's control. However, we note that KitAMO actually measures this time as part of an evaluation (e.g. figure 9). The third part is the time necessary to set up the evaluation (Setup in table 2). This includes the creation and initial set-up of the similarity database, the insertion of the similarity values into the database, and the creation of the analysis database. The fourth part is the time needed by the analysis tool for one evaluation given the selected matchers, weights and threshold (Analysis in table 2). The first, second and third parts are done only once per evaluation. The fourth part is repeated for each new analysis.

As expected, the larger the ontologies, the more time the evaluation takes. With respect to the parts under KitAMO's control, the initial set-up takes the most time. Also this is expected as a number of databases needs to be created. However, the part that usually takes the most time is outside KitAMO's control, i.e. the calculation of the similarity values by the matchers. Both the initial set-up and the running of the matchers is performed only once. The actual analysis is fast and can be repeated to create new analysis results.

In the past we have run analysis experiments on the implemented test cases using the matchers in SAMBO (e.g. [25, 26, 42]). The time needed by each

matcher to calculate the similarity values was similar to the time it takes in KitAMO. The analysis process was done manually and partly using Excel. This process needed to be repeated for each new analysis. While KitAMO generates the analysis results in seconds, this process was previously time-consuming and error-prone.

4.2 Example use

In this part we show how we can use KitAMO for evaluating matchers and analyzing the results. We use the ear case to evaluate two matcher plug-ins TermWN and UMLSKSearch. The experiments are similar to the ones in [26].

TermWN [26] is a terminological matcher combined with the general thesaurus WordNet. The terminological matcher is a combination matcher based on the textual descriptions (names and synonyms) of concepts and relations. In the current implementation, the matcher combines two approximate string matching algorithms (n-gram and edit distance) and a linguistic algorithm. A n-gram is a set of n consecutive characters extracted from a string. Similar strings will have a high proportion of n-grams in common. Edit distance is defined as the number of deletions, insertions, or substitutions required to transform one string into the other. The greater the edit distance, the more different the strings are. The linguistic algorithm computes the similarity of the terms by comparing the lists of words of which the terms are composed. Similar terms have a high proportion of words in common in the lists. A Porter stemming algorithm is employed to each word. Further, the similarity measure is enhanced by looking up the hypernym relationships of the pairs of words in the terms in WordNet [45].

UMLSKSearch [26] uses domain knowledge. We utilize the Metathesaurus in the Unified Medical Language System (UMLS) [44] which contains more than 100 biomedical and health-related vocabularies. The Metathesaurus is organized using concepts. The concepts may have synonyms which are the terms in the different vocabularies in the Metathesaurus that have the same intended meaning. The similarity of two terms in the source ontologies is determined by their relationship in UMLS. In our experiments we use the UMLS Knowledge Source Server to query the Metathesaurus with source ontology terms. As a result we obtain concepts that have the source ontology term as their synonym. We assign a similarity value of 1 for exact matches of query results for the two terms, 0.6 if the source ontology terms are synonyms of the same concept and 0 otherwise.

We decide to experiment with thresholds 0.4, 0.5, 0.6, 0.7 and 0.8 for the two individual matchers, and different weights for the combination for the threshold 0.5. The analysis results are shown in figure 10. We have sorted the results according to the matchers and their thresholds. This allows us to analyze the influence of the thresholds for the matchers. For TermWN we see that the quality of the results differs significantly for the different thresholds. Although the number of correct suggestions is almost the same (25 or 26), the number of wrong suggestions goes from 3 to 8, 19, 65 and 110 when the threshold decreases.

matcher	Th	C	W	I
(1.0UM,1.0TW)	0.50	23	2	0
(1.0UM,1.2TW)	0.50	24	2	0
(1.0UM,1.4TW)	0.50	25	2	0
(1.0UM,1.6TW)	0.50	26	3	0
(1.0UM,1.8TW)	0.50	26	3	0
(1.0UM,2.0TW)	0.50	26	3	0
(1.0UM,3.0TW)	0.50	26	13	2
(1.0UM,5.0TW)	0.50	26	19	2
(1.2UM,2.0TW)	0.50	26	3	0
(1.2UM,3.0TW)	0.50	26	8	0
(1.2UM,5.0TW)	0.50	26	17	2
(1.4UM,2.0TW)	0.50	26	2	0
(1.4UM,3.0TW)	0.50	26	3	0
(1.4UM,5.0TW)	0.50	26	14	2
TermWN	0.40	26	110	19
TermWN	0.50	26	65	8
TermWN	0.60	26	19	2
TermWN	0.70	26	8	0
TermWN	0.80	25	3	0
UMLSKSearch	0.40	23	2	1
UMLSKSearch	0.50	23	2	1
UMLSKSearch	0.60	23	2	1
UMLSKSearch	0.70	22	2	0
UMLSKSearch	0.80	22	2	0

Figure 10: The analysis results for the ear case.

Also the number of inferred suggestions increases when the threshold decreases. This would suggest to use a high threshold for TermWN for this case. For UMLSKSearch the quality of results stays similar when the threshold changes.

For the combination the threshold is the same, but we have varied the weights for the matchers in the combination. In addition to comparing the different combinations to each other (e.g. the combinations with weights (1,1.4) and (1,1.6) give good results), we can also compare the combinations with the individual matchers. We note, for instance, that TermWN finds the correct suggestions that the combinations find. However, the combination finds fewer wrong suggestions. In the combination UMLSKSearch can be seen as the contributing factor to filter out the wrong and inferred suggestions. This is reasonable since the similarity values from UMLSKSearch can only be 1, 0.6 and 0. It also suggests that the available domain knowledge in UMLS has good quality.

We can also sort the table with respect to the threshold. This allows us to compare the influence of the threshold between the different matchers. We can also sort the table with respect to the number of correct suggestions. In the best case this gives us the best alignment situation. Otherwise, when there are also many wrong suggestions, it may give a good starting point for combining with other algorithms (as TermWN in the example) or for applying a more advanced filtering technique as in [3].

To examine the matchers in more detail we can use the similarity table as in figure 8. By sorting the table with respect to TermWN and looking at the pairs with similarity values above a certain threshold we can analyze the properties of TermWN. For instance, we observe that TermWN finds suggestions where the names of terms are slightly different, e.g. (stapes, stape). As the test ontologies contain a large number of synonyms, also suggestions where the names of terms are completely different can be found, e.g. (inner ear, labyrinth), where inner ear has labyrinth as synonym. By using WordNet, TermWN finds suggestions such as (perilymphatic channel, cochlear aqueduct) where cochlear aqueduct has perilymphatic duct as synonym, and duct is a synonym of channel in WordNet. On the other hand, since endothelium is a kind of epithelium in WordNet, TermWN generates a wrong suggestion (corneal endothelium, corneal epithelium). Sorting the table with respect to UMLSKSearch we can analyze the properties of that matcher. As the similarity values assigned by UMLSKSearch can only be 1, 0.6 and 0, we obtain good results for the threshold 0.6. (This was already clear from the table in figure 10.) The matcher finds suggestions of which the terms have completely different names and synonyms, or have no synonyms at all, e.g. (external acoustic meatus, ear canal). The matcher works for some terms with slightly different names, e.g. (optic disc, optic disk), which are mapped to the concept optic disc in UMLS, but does not work for others, e.g. (stapes, stape), which are mapped to different concepts in UMLS.

The number of expected suggestions for the ear case is 27 (see figure 6). To find out the expected suggestion that is not found by any of the matchers we can check the similarity table as in figure 8. By sorting the similarity table according to the similarity values of a matcher, and looking at the values below the thresholds we will easily find that the only pair marked with 'C' in the

'Sug' column is (auricle, ear cartilage). This pair receives a very low similarity value from TermWN as the strings are very different and also the synonyms in WordNet are very different. We can also see that the terms are not synonyms in UMLS.

The similarity table can also be sorted with respect to the terms in the first ontology or the terms in the second ontology. This allows for checking for a term in one ontology which term in the other ontology is closest related according to the different matchers.

An advantage of using a system like KitAMO is that we can experiment with different (combinations of) strategies and different (combinations of) types of ontologies. For instance, the evaluation in our example may give an indication about what (combinations of) strategies may work well for aligning ontologies with similar properties as our test ontologies. However, when choosing a strategy other factors, such as time, may also play a role. For instance, KitAMO shows that UMLSKSearch is more time consuming than TermWN.

5 Related work

The experiments for EON and I3CON used tools in the evaluations [11, 1]. An API for ontology alignment for EON is described in [11, 7]. In the API the interface *AlignmentProcess* provides the method *align* which needs to be implemented to perform the computation of the alignments. The alignment algorithms should not require user intervention. In the API there are several linguistic-based alignment algorithms implemented that compute similarity values between terms. The different components of ontologies, e.g. concepts, instances and relations can be aligned. The API also allows to choose a filter method out of a few predefined methods, such as threshold-based filtering or retaining the n % pairs with highest similarity values. The *evaluator* is the interface for the evaluation of two alignment results. In the API two evaluators are implemented. One computes the precision, recall, fallout and f-measure of an alignment result. The other produces a weighted symmetric difference between two alignments. The API supports source ontologies in OWL, and expected alignments which are represented in RDF/XML. The alignment results can be output as RDF, OWL, XSLT, SWRL and COWL files. The evaluation results are reported in a RDF/XML file.

OLA [8] is a GUI application implemented on top of the API. OLA supports ontologies represented in OWL-Lite. The ontologies can be visualized as graphs. After loading two ontologies, choosing an alignment algorithm, and specifying the parameters for alignment (e.g. a threshold), the system runs the alignment algorithm. After the computation the alignments and their similarity values can be presented in a table and output as an XML file. Further, OLA provides a tool for alignment comparison. After loading two alignments in the form of XML files which were the results of the alignment tool, the precision, recall, fallout and f-measure of the alignments are computed. The results are displayed and the user can compare them. The evaluation results can also be saved as an XML

file.

Both the EON tools and KitAMO focus on the non-interactive part of the alignment framework. KitAMO provides an integrated system for comparative evaluation of alignment strategies and their combinations. In KitAMO after the computation of the similarity values, the evaluations can be performed for different alignment algorithms with different thresholds, and also for different combinations with different algorithms, weights and thresholds. The EON tools do not support the evaluation of the combination of different alignment algorithms. Also, to evaluate different alignment algorithms and different thresholds, batch programs in Java based on the API need to be implemented. OLA can also only compare two alignment results. While OLA presents the alignment results to the user, KitAMO presents the alignment results as well as the similarity values for all pairs of terms. KitAMO also allows to sort the table according to the different columns which gives the user the opportunity to analyze the properties of the alignment strategies. In OLA the tool for alignment comparison computes the precision, recall, fallout and f-measure of the alignments, while KitAMO presents the number of the correct, wrong and inferred suggestions to the user in a table. The measures presented by OLA can be easily computed and we intend to add these to the interface. KitAMO also allows to store the evaluation results from different matchers and combinations, and with different thresholds. This allows for a deeper comparison of the strategies. Further, KitAMO computes the performance of the strategies.

6 Conclusions

In this paper we proposed the KitAMO framework for comparative evaluation of the non-interactive alignment components, including alignment algorithms, combination algorithms and filters. We presented our current implementation of the framework. In this implementation we focused on the evaluation of different alignment algorithms and implemented the most used combination and filter methods. We evaluated the implementation with respect to performance. We also showed how the system can be used to evaluate and compare alignment algorithms and their combinations in terms of performance and quality of the proposed alignments and how these results lead to deeper insights into the properties of the strategies.

In the future we will test the scalability of KitAMO. We will also further develop different aspects of KitAMO. First, we will provide support for the evaluation of combination and filter methods. We will also use the framework as a basis for implementing and testing new alignment components. For the evaluation cases ontologies from different topic areas and with different representational complexity should be included. The current test cases are small pieces from larger ontologies. Although the expected alignments for large 'real life' ontologies are hard to obtain, they are necessary for better evaluations. Further, we will add different ways of visualizing the alignment and evaluation results.

Acknowledgments

We acknowledge the financial support of the Swedish Research Council (Vetenskapsrådet), the Center for Industrial Information Technology (CENIIT), the Swedish national graduate school in computer science (CUGS), and the EU Network of Excellence REWERSE (Sixth Framework Programme project 506779).

References

[1] Ashpole B (2004) Ontology translation protocol (ontrapro). *Proceedings of the Performance Metrics for Intelligent Systems Workshop.*

[2] Bard JL, Kaufman MH, Dubreuil C, Brune RM, Burger A, Baldock RA, Davidson DR (1998) An internet-accessible database of mouse developmental anatomy based on a systematic nomenclature. *Mechanisms of Development*, 74:111-120.

[3] Chen B, Tan H, Lambrix P (2006) Structure-based filtering for ontology alignment. *Proceedings of the IEEE WETICE Workshop on Semantic Technologies in Collaborative Applications.*

[4] Collins F, Green E, Guttmacher A, Guyer M (2003) A vision for the future of genomics research. *Nature*, 422:835-847.

[5] Doan A, Madhavan J, Domingos P, Halevy A (2003) Ontology matching: A machine learning approach. Staab, Studer (eds) *Handbook on Ontologies in Information Systems*, pp 397-416, Springer.

[6] Ehrig M, Haase P, Stojanovic N, Hefke M (2005) Similarity for Ontologies - A Comprehensive Framework. *Proceedings of the 13th European Conference on Information Systems.*

[7] Euzenat J (2005) An API for Ontology alignment (version 1.3).

[8] Euzenat J, Loup D, Touzani D, Valtchev D (2004) Ontology Alignment with OLA. *Proceedings of the 3rd International Workshop on Evaluation of Ontology-based Tools.*

[9] Euzenat J, Stuckenschmidt H, Yatskevich M (2005) Introduction to the Ontology Alignment Evaluation 2005. *Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies.*

[10] Euzenat J (2004) Introduction to the EON ontology alignment context. *Proceedings of the 3rd International Workshop on the Evaluation of Ontology-based Tools.*

[11] Euzenat J (2004) An API for ontology alignment. *Proceedings of the 3rd International Semantic Web Conference*, pp 698-712.

[12] FOAM. http://www.aifb.uni-karlsruhe.de/WBS/meh/foam/

[13] The Gene Ontology Consortium (2000) Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25-29. http://www.geneontology.org/.

[14] Gómez-Pérez A (1999) Ontological Engineering: A state of the Art. *Expert Update*, 2(3):33-43.

[15] Giunchiglia F, Shvaiko P, Yatskevich M (2004) S-Match: an algorithm and an implementation of semantic matching. *Proceedings of the European Semantic Web Symposium*, LNCS 3053, pp 61-75.

[16] Hayamizu TF, Mangan M, Corradi JP, Kadin JA, Ringwald M (2005) The Adult Mouse Anatomical Dictionary: a tool for annotating and integrating data. *Genome Biology*, 6(3):R29

[17] I3CON (2004) http://www.atl.lmco.com/projects/ontology/i3con.html

[18] Jena - A Semantic Web Framework for Java. http://jena.sourceforge.net/

[19] Kalfoglou Y, Schorlemmer M (2003) IF-Map: an ontology mapping method based on information flow theory. *Journal on Data Semantics*, 1:98-127.

[20] KnowledgeWeb Consortium (2004) Deliverable 2.2.4 (Specification of a methodology, general criteria, and benchmark suites for benchmarking ontology tools). http://knowledgeweb.semanticweb.org/

[21] Kotis K, Vouros GA (2004) The HCONE Approach to Ontology Merging. *Proceedings of the European Semantic Web Symposium*, LNCS 3053, pp 137-151.

[22] Lambrix P (2004) Ontologies in Bioinformatics and Systems Biology. Chapter 8 in Dubitzky W, Azuaje F (eds) *Artificial Intelligence Methods and Tools for Systems Biology*, pp 129-146, Springer.

[23] Lambrix P, Edberg A (2003) Evaluation of ontology merging tools in bioinformatics. *Proceedings of the Pacific Symposium on Biocomputing*, 8:589-600.

[24] Lambrix P, Tan H (2005) Merging DAML+OIL Ontologies. Barzdins, Caplinskas (eds) *Databases and Information Systems*, pp 249-258, IOS Press.

[25] Lambrix P, Tan H (2005) A Framework for Aligning Ontologies. *Proceedings of the 3rd Workshop on Principles and Practice of Semantic Web Reasoning*, LNCS 3703, pp 17-31.

[26] Lambrix P, Tan H (2006) SAMBO - A System for Aligning and Merging Biomedical Ontologies. *Journal of Web Semantics, special issue on Semantic Web for the Life Sciences.*

[27] Le BT, Dieng-Kuntz R, Gandon F (2004) On ontology matching problem (for building a corporate semantic web in a multi-communities organization). *Proceedings of 6th International Conference on Enterprise Information Systems*.

[28] Medical Subject Headings. http://www.nlm.nih.gov/mesh/

[29] McGuinness D, Fikes R, Rice J, Wilder S (2000) An Environment for Merging and Testing Large Ontologies. *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning*, pp 483-493.

[30] Mitra P, Wiederhold G (2002) Resolving terminological heterogeneity in ontologies. *Proceedings of the ECAI Workshop on Ontologies and Semantic Interoperability*.

[31] Noy NF, Musen M (2000) PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. *Proceedings of Seventeenth National Conference on Artificial Intelligence*, pp 450-455.

[32] Noy NF, Musen M (2001) Anchor-PROMPT: Using Non-Local Context for Semantic Matching. *Proceedings of the IJCAI Workshop on Ontologies and Information Sharing*, pp 63-70.

[33] OBO - Open Biomedical Ontologies. http://obo.sourceforge.net/

[34] OntoWeb Consortium (2002) Deliverables 1.3 (A survey on ontology tools) and 1.4 (A survey on methodologies for developing, maintaining, evaluating and reengineering ontologies). http://www.ontoweb.org

[35] Prasad S, Peng Y, Finin T (2002) Using Explicit Information To Map Between Two Ontologies, *Proceedings of the AAMAS Workshop on Ontologies in Agent Systems*.

[36] REWERSE. Backofen R, Badea M, Burger A, Fages F, Lambrix P, Nutt W, Schroeder M, Soliman S, Will S (2004) State-of-the-art in Bioinformatics. REWERSE Deliverable A2-D1.

[37] REWERSE. Backofen R, Badea M, Barahona P, Burger A, Dawelbait G, Doms A, Fages F, Hotaran A, Jakonienė V, Krippahl L, Lambrix P, McLeod K, Möller S, Nutt W, Olsson B, Schroeder M, Soliman S, Tan H, Tilivea D, Will S (2005) Usage of bioinformatics tools and identification of information sources. REWERSE Deliverable A2-D2.

[38] Stevens R, Goble C, Bechhofer S (2000) Ontology-based knowledge representation for bioinformatics. *Briefings in Bioinformatics*, 1(4):398-414.

[39] Stumme G, Mädche A (2001) FCA-Merge: Bottom-up merging of ontologies. *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pp 225-230.

[40] Su XM, Hakkarainen S, Brasethvik T (2004) Semantic enrichment for improving systems interoperability. *Proceedings of the ACM Symposium on Applied Computing*, pp 1634-1641.

[41] Takai-Igarashi T, Nadaoka Y, Kaminuma T (1998) A Database for Cell Signaling Networks. *Journal of Computational Biology*, 5(4):747-754.

[42] Tan H, Jakonienė V, Lambrix P, Aberg J, Shahmehri S (2006) Alignment of biomedical ontologies using life science literature, *Proceedings of the International Workshop on Knowledge Discovery in Life Science Literature*, LNBI 3886, pp 1-17.

[43] Takai-Igarashi T and Takagi T (2000), SIGNAL-ONTOLOGY: Ontology for Cell Signalling. *Genome Informatics*, 11:440-441.

[44] UMLS. http://www.nlm.nih.gov/research/umls/about_umls.html

[45] WordNet. http://wordnet.princeton.edu/