A method for recommending ontology alignment strategies

He Tan and Patrick Lambrix

Department of Computer and Information Science
Linköpings universitet, Sweden

This is a pre-print version of the article published in the Proceedings of the International Semantic Web Conference, 2007.

Abstract. In different areas ontologies have 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 already exist a number of different alignment strategies. However, it is usually difficult for a user that needs to align two ontologies to decide which of the different available strategies are the most suitable. In this paper we propose a method that provides recommendations on alignment strategies for a given alignment problem. The method is based on the evaluation of the different available alignment strategies on several small selected pieces from the ontologies, and uses the evaluation results to provide recommendations. In the paper we give the basic steps of the method, and then illustrate and discuss the method in the setting of an alignment problem with two well-known biomedical ontologies. We also experiment with different implementations of the steps in the method.

1 Introduction

In recent years many ontologies have been developed. The benefits of using ontologies include reuse, sharing and portability of knowledge across platforms, and improved documentation, maintenance, and reliability (e.g. [13]). Ontologies lead to a better understanding of a field and to more effective and efficient handling of information in that field. Many of the currently developed ontologies contain overlapping information. For instance, Open Biomedical Ontologies (OBO, http://obo.sourceforge.net/) lists 18 different anatomy ontologies (January 2007), some of which are deprecated (e.g. Arabidopsis anatomy and Cereal anatomy) and have been replaced by a larger ontology (e.g Plant anatomy) when the large amount of overlap was realized.

Often we would 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. Further, the data in different data sources in the same domain may have been annotated with different but similar ontologies. Knowledge of the inter-ontology relationships would in this case lead to improvements in search, integration and analysis of data. It has been realized that this is a major issue and some organizations have started to deal with it. For instance, in the area of anatomy the SOFG (http://www.sofg.org/) has developed the SOFG Anatomy Entry List and an NCBO anatomy workshop was organized to start the development of the Common Anatomy Reference Ontology (http://www.bioontology.org/wiki/index.php/CARO:Main_Page).

In the remainder of this paper we say that we align two ontologies when we define the relationships between terms in the different ontologies. Currently, there exist a number of ontology alignment systems that support the user to find inter-ontology relationships. For overviews we refer to, e.g., [4, 11, 6] and the Ontology Matching website (http://www.ontologymatching.org/). These systems use different techniques, but it is not clear how well these techniques perform for different types of ontologies. Relatively few comparative evaluations on ontology alignment systems and algorithms have been performed. It is therefore difficult for a user to decide, among the different alignment strategies, which strategy or combination of strategies is best to use for aligning given ontologies.

In this paper we tackle this problem by proposing a method that provides recommendations on alignment strategies for a given alignment problem. As not much information is available on which strategies work best in which situations, we use information inherent in the actual ontologies to align. We base our method on the evaluation of the different available alignment strategies on several small selected pieces from the ontologies. These evaluation results are then used to provide recommendations. The method defines different steps: segment pair selection, segment pair alignment generation, evaluation and recommendation. The method and different steps are presented in section 3 and illustrated in the setting of an alignment problem with two well-known biomedical ontologies in section 4. Each step in the method can be instantiated by different algorithms. In section 4 we also discuss experiments with different algorithms for different steps of the method and discuss their influence on the recommendations. We conclude the paper with a conclusion and discussion of future work. In the next section we give some background and related work.

2 Background

Ontology alignment Many of the current systems are based on the computation of similarity values between terms in the source ontologies, and can be seen as instantiations of the framework defined in [6]. This framework is shown in figure 1. It consists of two 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. Some systems may not have the second part. An alignment algo-

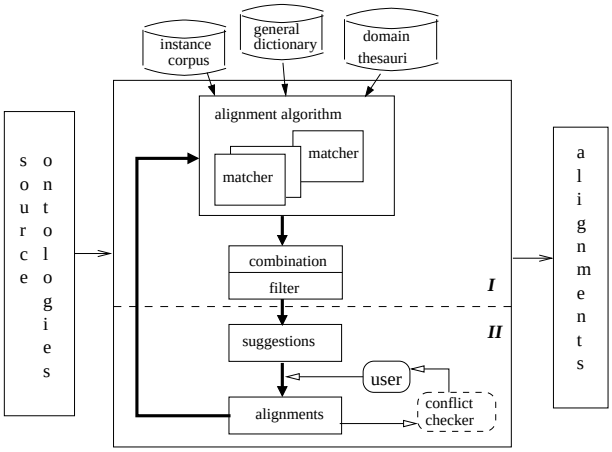

Fig. 1. Alignment framework [6].

rithm receives as input two source ontologies. The algorithm can include several matchers. 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. Each matcher utilizes knowledge from one or multiple sources. The matchers calculate similarities between the terms from the different source ontologies. Alignment suggestions are then determined by combining and filtering the results generated by one or more matchers. By using different matchers and combining and filtering the results in different ways we obtain different alignment strategies. The suggestions are then presented to the user who accepts or rejects them. The acceptance and rejection of a suggestion may influence further 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.

Evaluation of alignment strategies Currently, we do not have much knowledge about how well the different alignment strategies perform for different kinds of ontologies. Comparative evaluations of ontology alignment systems have been performed by some groups. The EU OntoWeb project [10] evaluated the systems PROMPT based on Protégé (with extension Anchor-PROMPT), Chimaera (described, not evaluated), FCA-Merge 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 [5, 6] PROMPT, Chimaera, FOAM and 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 with different threshold values for filtering in [6]. Further, there are evaluations connected to the Ontology Alignment Evaluation Initiative (OAEI, http://oaei.ontologymatching.org/). The 2006 campaign

consisted of 4 tracks: a comparison track, an expressive ontologies track, a directories and thesauri track, and a consensus workshop. Each track has a different evaluation purpose. The comparison track used precision and recall as evaluation measures.

It is realized that the study of the properties, and the evaluation and comparison of the alignment strategies and their combinations, give us valuable insight in how the strategies could be used in the best way. Recently, some tools have been developed for evaluating and comparing the non-interactive part of alignment algorithms. The OAEI describes an API [3] that could be used by systems participating in the initiative. Evaluators are implemented. They compute the precision, recall, fallout and f-measure of an alignment result and a weighted symmetric difference between two alignments. KitAMO [7] provides an integrated system for comparative evaluation and analysis of alignment strategies and their combinations. KitAMO reports on similarity values, allows evaluations based on precision and recall for different combinations of different algorithms, combination weights and thresholds, as well as computes the performance of the strategies. Further, an environment is provided for analyzing the available data.

Selecting the best alignment strategies The problem of selecting the best alignment strategy is tackled by [9] and [2]. In [9] it is argued that finding appropriate alignment strategies should be based on knowledge about the strategies and their previous use. As a first step a number of factors (related to input, output, approach, usage, cost and documentation) were identified that are relevant when selecting an alignment strategy. The relevant data is collected by questionnaires. The Analytic Hierarchy Process is used to detect suitable alignment approaches. In [2], APFEL, a machine learning approach to optimize alignment strategies is proposed. In APFEL a set of feature parameters are declared for the source ontologies, the similarity assessment, and the different matchers, combination and filter algorithms. To generate training data, an existing parametrization is used and alignment suggestions are generated. These suggestions need to be validated by the user. A machine learning approach is then used to learn an optimal parametrization. In the next section we propose another technique. As not much knowledge is available yet about the suitability of alignment strategies for different alignment problems (as is required by the method in [9]), we have chosen to use information about the actual ontologies to be aligned. This information is in the form of alignments between small pieces of the ontologies, which can be used to compute how well the available alignment strategies perform for these small pieces. In contrast to [2], it gives us complete information on the alignments for smaller areas in the ontologies, while [2] can only assume full knowledge about their initially generated alignment suggestions.

The problem of finding the best approach for aligning ontologies can also be seen as a variant of the general tuning problem for schema matching systems as defined in [8]. However, the scenario that is discussed in [8] is a different instance of the general tuning problem than what is tackled here. They consider the problem of finding the best approaches for matching a given (relational)

schema with all other future schemas, while we tackle the problem of finding the best approaches for aligning two given ontologies.

3 Recommendation Method

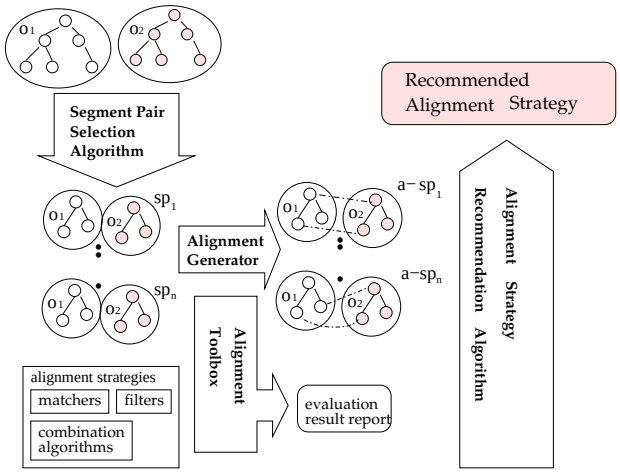

Fig. 2. Recommendation method.

As a first step in our method (figure 2), the segment pair selection algorithm selects pairs of small pieces of the ontologies, called segments. For these segment pairs expected alignments need to be generated. In the alignment toolbox the available alignment strategies align the segment pairs, and reports on the alignment results are generated. Based on these reports, the recommendation algorithm gives recommendations on the strategies for aligning the two given ontologies. In the rest of this section we present each step in more detail.

Segment Pair Selection Algorithm A segment of an ontology is a portion of the ontology. It represents a piece of the knowledge that the ontology represents and can be viewed as an ontology itself, usually with similar characteristics as the original ontology. The selection algorithm may use already available segment pairs (with or without their alignments). Further, the segment pairs could be manually selected by domain experts or ontology experts. Also (semi-)automatic means may be used.

Several factors regarding the selection of the segment pairs may have an influence on the final recommendation, such as, for instance, the overlap between the segments (e.g. the number of terms occurring in more than one segment), the number of segment pairs, and the number of elements included in a segment. Some recommendation algorithms may also require a certain (minimum) number

of segment pairs. For those alignment strategies utilizing the structure of the ontologies and the constraint knowledge, the distribution of the segment pairs over the ontologies and the granularity within the segments may also influence the evaluation results.

Segment Pair Alignment Generator For the segment pairs, expected alignments need to be generated. In some cases, alignments may already be available. The expected alignments can also be specified manually by domain experts. As the segments are only small parts of the original ontologies, the effort and complexity related to this task is much smaller than for the whole ontologies. In both these cases, there is an assumption that we have full information about the alignments, although, in general, this is not the case. Domain experts may not always agree with each other. In the case where no alignments or domain experts are readily available, it may still be possible to obtain alignments by using established bodies of domain knowledge (e.g. in the form of other ontologies and alignments) as oracles.

Alignment Toolbox In the alignment toolbox the alignment strategies including the different matchers, filtering and combination algorithms are applied to align the segment pairs. A report on the alignment results is generated and given to the recommendation algorithm. The report may contain information about, for instance, the similarity values between the terms generated by the different matchers and their combinations, the alignment suggestions that are filtered out by the filtering algorithms, and the execution time for the strategies.

Alignment Strategy Recommendation Algorithm The main purpose of the recommendation algorithm is to recommend one or more alignment strategies. The algorithm can return the best strategies according to a certain performance measure, or the top n strategies, or the best $m\%$ of the strategies. For a particular alignment problem, several alignment strategies could be suitable and even strategies with a slightly lower performance may work well for the whole ontologies. The performance measure may be based on such things as the quality of the alignment suggestions (e.g. in terms of precision, recall, f-measure) or the execution performance. The different components in the performance measure may have different degrees of importance. For instance, the quality of the suggestions may be more important than the execution time.

4 Experiments and Illustration

In this section we illustrate our method for recommending alignment strategies using two biomedical ontologies and 6 available matchers with 5 different thresholds each. We describe implementations for the different steps and discuss the results. Further, we experimented with 2 segment pair selection algorithms, different numbers of segment pairs, and 3 recommendation measures, and discuss their influence on the recommendation results.

4.1 Experiment Case

Ontologies In the experiments we use two well-known biomedical ontologies. The NCI thesaurus (http://ncicb.nci.nih.gov/infrastructure/cacore_overview/-vocabulary) is a reference terminology produced by the National Cancer Institute's Center for Bioinformatics (NCICB). The thesaurus includes broad coverage of the cancer domain, including cancer-related diseases, drugs and chemicals, genes and gene products, and anatomy. Around 34,000 terms are hierarchically organized and partitioned into 20 kinds. In our experiment we use the anatomy kind which contains 3495 terms. Within the NCICB Core Infrastructure, the NCI thesaurus together with the NCI Metathesaurus provides the semantic base for different projects. Medical Subject Headings (MeSH, http://www.nlm.nih.gov/-mesh/) is a controlled vocabulary published by the American National Library of Medicine (NLM). It consists of sets of terms naming descriptors in a hierarchical structure. These descriptors are organized in 16 categories. The category A for anatomy terms used in the experiment includes 1391 terms. MeSH is used for indexing, cataloging, and searching for biomedical and health-related literature in MEDLINE/PubMed.

We used the Unified Medical Language System (UMLS, http://www.nlm.-nih.gov/research/umls/about_umls.html) as an oracle during the generation of expected alignments. The Metathesaurus in UMLS contains more than 100 biomedical and health-related vocabularies, among which the NCI thesaurus and MeSH. It 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. This means that the knowledge represented in UMLS can be used as an approximation of domain expert knowledge and UMLS can be used as an oracle in its domain. As the NCI thesaurus and MeSH are included in the Metathesaurus of UMLS, alignments are available. According to UMLS there are 919 expected alignments for the two ontologies.

Alignment Strategies We experiment with four linguistic matchers, a weighted sum combination algorithm and a threshold filter. The n-gram (NG) and edit-distance (ED) matchers use approximate string matching algorithms. An 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 two other matchers (WL and WN) compute the similarity between two terms by comparing the lists of words of which the terms are composed. Similar terms have a high proportion of words in common. Both matchers use a Porter stemming algorithm. The more advanced (WN) of the two also uses WordNet (http://wordnet.princeton.edu/), which has a good coverage of anatomy [1], during the computation of the similarity values, by using the hypernym relationships in WordNet. All matchers compute similarity values in [0..1], where the higher the value the more similar two terms. The combination algorithm is a weighted sum, $Sim(t_1, t_2) = \sum_{k=1}^{n} w_k \cdot sim_k(t_1, t_2)$, where n is the number of the combined matchers and sim_k and w_k represent the similarity

values and weights, respectively, for the different matchers. We also require that $w_k \in [0, 1]$ and $\sum_{k=1}^{n} w_k = 1$. In the experiment we evaluate two combinations. The first combination (C1) includes n-gram, edit-distance and the first word list matcher. The second combination (C2) includes n-gram, edit-distance and the word list matcher with WordNet. In both combinations all weights are set to $\frac{1}{3}$. The threshold filter allows only term pairs with similarity values higher than or equal to the threshold value as alignment suggestions. We use thresholds 0.4, 0.5, 0.6, 0.7 and 0.8. This means that we have 30 available strategies in total (6 matchers with 5 thresholds each).

4.2 Algorithms in the recommendation process

Segment Pair Selection Algorithms For this experiment we developed two algorithms that select segment pairs. The first algorithm (SubG) collects the pairs of terms in the two ontologies which have the same name (case-insensitive string matching). The pairs of sub-graphs of the ontologies rooted at these terms with respect to the *is-a* and *part-of* hierarchies are candidate segment pairs. The segment pairs are randomly chosen from the candidate segment pairs, with the restriction that the segments are pairwise disjoint. We also required that the number of terms in a segment is strictly between 1 and 60. This avoids leaves in the ontologies as well as too large segments.

In the second algorithm (Clust) the terms in the ontologies are first partitioned into clusters. We use a variant of the algorithm proposed in [12], where a dependency and the strength of the dependency between two terms is defined based on the *is-a* and *part-of* hierarchies of the ontology. The clusters satisfy the intuition that the dependency between any two terms in a cluster is stronger than the dependency between a term in the cluster and a term that is not in the cluster. Further, we require that the number of terms in a cluster is at least 5. The candidate segment pairs are the pairs of clusters from the two ontologies including at least one pair of terms with the same name (case-insensitive string matching). The segment pairs are then randomly chosen from the candidate segment pairs.

In the experiment we generate 5 segment pairs per trial. We experimented with 3 different generated segment pair sets per segment pair selection algorithm, but also used their combinations. This means that for each segment pair selection algorithm, we have 3 sets with 5 segment pairs, 3 sets with 10 segment pairs and 1 set with 15 segment pairs.

Segment Pair Alignment Generator We use UMLS as alignment generator. We query the ontology terms in the UMLS Metathesaurus using their names. If the queries for two terms in different ontologies return the same UMLS concept, we consider the pair of terms as an expected alignment.

Alignment Toolbox We use the current implementation of KitAMO [7] as the alignment toolbox. We input the segments, the results from the alignment generator and the matchers, combination algorithm and filter method into KitAMO. KitAMO runs the alignment strategies and produces reports on the

similarity values generated by the different matchers, their execution time, and the number of correct, wrong and redundant suggestions for different thresholds.

Alignment Strategy Recommendation Algorithm In this experiment we calculate a recommendation score for the alignment strategies as a weighted sum of property measures, $\sum_{i=1}^{m} w_i \cdot p_i$, where w_i is the weight of the property, and p_i is the score of the property. The higher the recommendation score, the more preferred the alignment strategy is. Our algorithm returns a ranking of the available strategies based on the recommendation score. This ranking can then be used to return the best, top n, or best $m\%$ of the strategies.

We used two properties. The first property is the quality of the alignment suggestions. It is measured as $\frac{1}{m} \cdot \sum_{s=1}^{m} f_s$, where m is the number of segment pairs, and f_s the f-measure value. The f-measure integrates precision and recall. It is calculated as $\frac{P \cdot R}{(1-\alpha) \cdot P + \alpha \cdot R}$, in which the P is the precision (the number of correct suggestions divided by the number of suggestions), R the recall (the number of correct suggestions divided by the number of expected alignments) and α a number between 0 and 1. The higher the value for α, the more important precision is with respect to recall. The value of α is 0.5 in the experiment, i.e. we use the harmonic mean of precision and recall. The second property is the execution performance of an alignment strategy. It is measured as $-\frac{1}{m} \cdot \sum_{s=1}^{m} \frac{t_s}{n_s}$, where m is the number of the segment pairs, t_s is the execution time the alignment strategy needs to calculate similarity values for a segment pair, and n_s is the number of term pairs in the segment pair. The execution time in this experiment is calculated using the KitAMO system.

We experimented with three different ways to compute a recommendation score using the two properties. In the first case, we only consider the quality of the alignment suggestions (F). In the second case we give equal weight to the quality of the suggestions and the execution performance (F+E). In the last case, the quality is weighted ten times higher than the execution performance (10F+E).

Expected Recommendations Table 1 represents the top 3 (or top 10%) alignment strategies per measure for the whole ontologies. A perfect recommendation for the experiment would thus be the simple word list matcher with threshold 0.8 (WL,0.8) for measures F (f-measure) and 10F+E (10 times f-measure and execution performance), and edit distance with threshold 0.8 (ED,0.8) for measure F+E (f-measure and execution performance).

F	F+E	10F+E
1. (WL,0.8)	1. (ED,0.8)	1. (WL,0.8)
2. (C1,0.8)	2. (WL,0.8)	2. (C1,0.8)
3. (C2,0.8)	3. (NG,0.7)	3. (WL,0.7)

Table 1. Top 3 alignment strategies per measure.

4.3 Results

Tables 2 and 3 present the top 3 recommendations given in the experiment for measure F and segment pair selection algorithms SubG and Clust, respectively. The 'SPS's represent the different generated segment pair sets. SPS A(1+2) is the segment pair set including both SPS A1 and SPS A2, and similarly for the other combination segment pair sets. For instance, table 2 shows that using A2, our algorithm recommends the simple word list matcher with thresholds 0.8 (WL,0.8) and 0.7 (WL,0.7) as best, respectively second best strategy, while the matcher with WordNet and threshold 0.7 (WN,0.7) is the third choice. Figure 3 shows the recommendation scores for the different matchers for the measure F and the SPSs containing 5 elements.

There is no overlap between the segment pair sets in the experiments for SubG (A1, A2 and A3), i.e. no term occurs in more than one segment. Similarly, there is no overlap between the segment pair sets in the experiments for Clust (B1, B2 and B3). In the experiments for SubG, the number of terms in a segment ranges from 2 to 34, and the levels in the *is-a* and *part-of* hierarchies in the segments range from 2 to 6. The number of expected alignments between segments ranges from 1 to 4 for A1, 1 to 23 for A2, and 1 to 5 for A3. In the experiments for Clust, the number of terms in a segment ranges from 5 to 14, and the levels in the *is-a* and *part-of* hierarchies in the segments range from 2 to 3. The number of expected alignments between segments ranges from 1 to 4 for B1, 1 to 3 for B2, and 1 to 3 for B3.

Table 4 shows the top 3 recommendations for measures F+E and 10F+E for the segment pair selection algorithms SubG and Clust. We only show the results for the segment pair sets with 5 elements.

SPS A1	SPS A2	SPS A3	SPS A(1+2)	SPS A(1+3)	SPS A(2+3)	SPS A(1+2+3)
1. (WL,0.8)	1. (WL,0.8)	1. (WL,0.8)	1.(WL,0.8)	1.(WL,0.8)	1.(WL,0.8)	1. (WL,0.8)
1. (WL,0.7)	2. (WL,0.7)	1. (WL,0.7)	2.(WL,0.7)	1.(WL,0.7)	2.(WL,0.7)	2. (WL,0.7)
1. (C1,0.8)	3. (WN,0.7)	1. (C1,0,8)	3.(C2,0.8)	1.(C2,0.8)	3.(C2,0.8)	3. (C2,0.8)
1. (C2,0.8)		1. (C2,0.8)	3.(C1,0.8)	1.(C1,0.8)	3.(C1,0.8)	3. (C1,0.8)

Table 2. Top 3 recommendations per segment pair set for segment pair selection algorithm SubG and measure F.

4.4 Discussion

Table 2 shows that the quality of the recommendations for measure F provided by the SPSs which are generated by the algorithm SubG is excellent. The best strategy for this alignment problem ((WL,0.8), see table 1) has always rank 1. The other top 3 strategies ((C1,0.8) and (C2,0.8)) are also often included in the highly recommended strategies. They have both rank 1 for A1 and A3 and

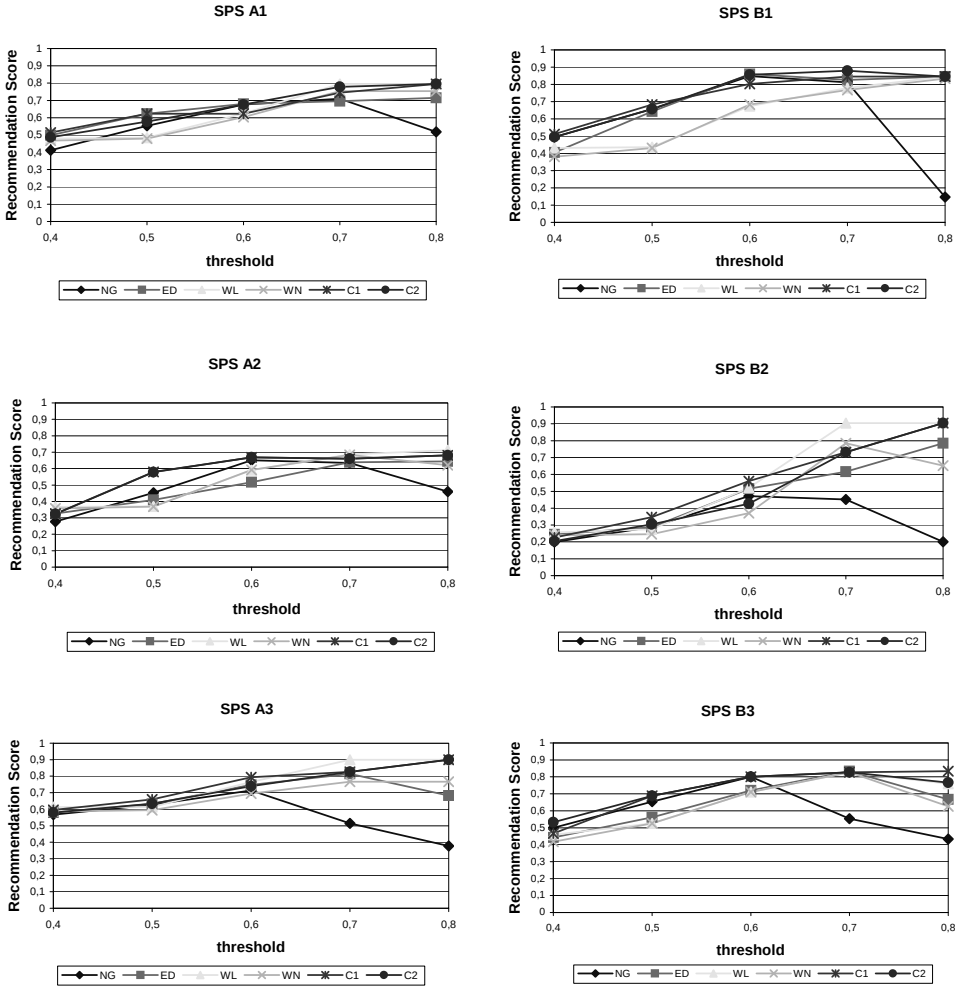

Fig. 3. Recommendation scores for the different matchers per segment pair set for the measure F. (A1, A2 and A3 for algorithm SubG. B1, B2 and B3 for algorithm Clust.)

SPS B1	SPS B2	SPS B3	SPS B(1+2)	SPS B(1+3)	SPS B(2+3)	SPS B(1+2+3)
1. (C2,0.7)	1. (WL,0.8)	1. (C1,0.8)	1. (WL,0.8)	1. (C2,0.7)	1. (C1,0.8)	1. (C1,0.8)
2. (ED,0.6)	1. (WL,0.7)	1. (ED,0.7)	1. (C1,0.8)	2. (C1,0.8)	2. (WL,0.7)	2. (C2,0.8)
3. (C2,0.6)	1. (C1,0.8)	3. (C1,0,7)	1. (C2,0.8)	3. (C1,0.7)	3. (C2,0.8)	3. (WL,0.8)
	1. (C2,0.8)	3. (C2,0.7)				3. (WL,0.7)
		3. (WL,0.7)				
		3. (WN,0.7)				

Table 3. Top 3 recommendations per segment pair set for segment pair selection algorithm Clust and measure F.

	SPS A1	SPS A2	SPS A3	SPS B1	SPS B2	SPS B3
F+E	1. (WL,0.8)	1. (WL,0.8)	1. (WL,0.8)	1. (C2,0.7)	1.(WL,0.8)	1.(ED,0.7)
	1. (WL,0.7)	2. (WL,0.7)	1. (WL,0.7)	2. (ED,0.6)	1.(WL,0.7)	2.(C1,0.8)
	3. (C1,0.8)	3. (C1,0.8)	3. (C1,0.8)	3. (C2,0.6)	3.(C1,0.8)	3.(WL,0.7)
10F+E	1. (WL,0.8)	1. (WL,0.8)	1. (WL,0.8)	1. (C2,0.7)	1.(WL,0.8)	1.(ED,0.7)
	1. (WL,0.7)	2. (WL,0.7)	1. (WL,0.7)	2. (ED,0.6)	1.(WL,0.7)	2.(C1,0.8)
	3. (C1,0.8)	3. (WN,0.7)	3. (C1,0.8)	3. (C2,0.6)	3.(C1,0.8)	3.(WL,0.7)

Table 4. Top 3 recommendations per segment pair set for the recommendation measures F+E and 10F+E. (A1, A2 and A3 for algorithm SubG. B1, B2 and B3 for algorithm Clust.)

(not shown in table) both rank 4 for A2. Regarding the other strategies that are recommended, (WL,0.7) also performs well for the whole ontologies (rank 4), while (WN,0.7) has rank 16.

Table 3 shows that the quality of the recommendations given by the SPSs which are generated by Clust is also good. For B2 we obtain all top 3 strategies. For B3 one of the top 3 strategies (C1,0.8) is recommended, while the others have rank 10 and 11. For B1 the top 3 have all recommendation rank 5. Regarding the other strategies that are recommended, (WL,0.7), (C1,0.7) and (C2,0.7) also perform well for the whole ontologies (ranks 4, 5 and 6), while (ED,0.6), (ED,0.7), (C2.0.6) and (WN,0.7) have ranks between 9 and 16.

The complete results for SubG and Clust with SPS containing 5 elements and measure F are shown in figure 3. For several matchers we can observe a certain trend in the behavior. Although the scores are different, often they behave in a similar way across the sets. This may be useful information to prune some of the computations.

From tables 2 and 3, we also observe that the combination of the results from different segment pair sets often provides higher quality recommendations. For instance, for SubG, the best strategy (WL,0.8) is often ranked alone as number 1 in the combinations (A(1+2),A(2+3),A(1+2+3)), while the other top 3 strategies are also top 3 recommendations. Also, the second recommendation (ED,0.6) for B1, which is ranked 17th for the whole ontologies, has dropped to

rank 13 for B(1+2+3). This is in line with the intuition that more information may give better alignment results.

As shown in tables 2 and 3, the recommendations based on the SPSs generated by SubG are usually better than the ones from Clust.

In total the number of expected alignments in SPSs is larger for SubG than for Clust. However, this does not seem to influence the recommendation results. For example, SPS A2 gives the worst recommendations for SubG, although it contains the highest number of expected alignments in the three SPSs from SubG. SPS B2 is the only SPS for Clust that recommends the top 3 strategies, although it has the smallest number of alignments.

The results from the experiments with 10F+E are similar to the results of the experiments with F. This is not surprising as the f-measure is by large the most important component in 10F+E. The differences are that the matchers using WordNet are not as highly ranked anymore. When F+E is the measurement, the quality of the recommendations given by the SPSs from both SubG and Clust is not as good. The best strategy (ED,0.8) (see table 1) is ranked between 5 (for B1 and B2) and 11 (for A3). Also the third best strategy (NG,0.7) has relatively low ranks (between 7 and 21). The second best strategy (WL,0.8) has rank 1 for all the SubG SPS, and ranks 7, 1, and 9 for B1, B2 and B3, respectively. Regarding the other recommended strategies, (ED,0.7), (C1,0.8), and (WL,0.7) also perform well for the whole ontologies (ranks 4, 5 and 6), while (ED,0.6), (C2,0.6) and (C2,0.7) have ranks between 11 and 20. The method for calculating the property measures may influence the results. For instance, in the experiment the execution time tends to increase from NG to ED to WL to WN. However, in practice there is an influence from the run-time environment which, for instance, gave very different execution times of 0.001 and 0.289 for WL with similar segment pairs. The value for F is between 0 and 1, and thus F+E is sensitive to the variation. Taking an average over several runs, may alleviate the problem in this case.

5 Conclusion

In this paper we have tackled the problem of deciding on which strategy to use for a particular alignment problem. We have done this by proposing a method for recommending alignment strategies. As we do not have much knowledge yet regarding the suitability of the current algorithms, we have proposed a method that uses evaluation results from the strategies on small pieces of the ontologies to be aligned. This requires a small effort of the user and as the original ontologies are used, we implicitly use knowledge about these ontologies. We have illustrated the method for an alignment problem using two anatomy ontologies and different alignment strategies. We also experimented with different segment pair selection algorithms, different numbers of segment pairs, and different recommendation measures, and discussed their influence on the recommendations. We also showed that for this alignment problem good results were obtained even with reasonably simple strategies.

There are several issues for future work. Even though we have shown the feasibility of our method and have obtained good results for the alignment problem in the experiment, it is necessary to perform more experiments with different kinds of ontologies and alignment strategies. We also want to further investigate the influence of the different choices in the different steps of the method. This includes investigating other segment pair selection strategies, recommendation measures and recommendation algorithms. We intend to develop a tool that supports these investigations by extending the KitAMO system. It will also be interesting to look at how the ideas from [9] and [2] can be used to augment our approach. For instance, when more knowledge is obtained regarding the different strategies and their previous use (as in [9]), this knowledge could be used as a first step to filter the available strategies and it can be used by the recommendation strategy. Also the optimization approach in [2] may be useful for finding better combinations as well as within the recommendation step. Finally, we intend to extend the SAMBO ontology alignment tool [6] with a recommendation component.

Acknowledgements. We thank Bo Servenius for recommending us to use UMLS as an oracle. We acknowledge the financial support of the Swedish Research Council, the Center for Industrial Information Technology , the Swedish national graduate school in computer science, and the EU NoE REWERSE (FP6 project 506779).

References

1. Bodenreider O, Burgun A, Characterizing the Definitions of Anatomical Concepts in WordNet and Specialized Sources, *Proc. of the First Global WordNet Conference*, 223-230, 2002.
2. Ehrig M, Staab S, Sure Y, Bootstrapping Ontology Alignment Methods with APFEL, *Proc. of the Int. Semantic Web Conference*, 186-200, 2005.
3. Euzenat J, An API for ontology alignment, version 2.1, *Ontology Alignment Evaluation Initiative*, 2006.
4. Kalfoglou Y, Schorlemmer M, Ontology mapping: the state of the art, *The Knowledge Engineering Review*, 18(1):1-31, 2003.
5. Lambrix P, Edberg A, Evaluation of ontology merging tools in bioinformatics, *Proc. of the Pacific Symposium on Biocomputing*, 8:589-600, 2003.
6. Lambrix P, Tan H, SAMBO - A System for Aligning and Merging Biomedical Ontologies, *Journal of Web Semantics*, 4(3):196-206, 2006.
7. Lambrix P, Tan H, A Tool for Evaluating Ontology Alignment Strategies, *Journal on Data Semantics*, VIII:182-202, 2007.
8. Lee Y, Sayyadian M, Doan A, Rosenthal A, eTuner: tuning schema matching software using synthetic scenarios, *VLDB Journal*, 16(1)::97-122, 2007.
9. Mochol M, Jentzsch A, Euzenat J, Applying an Analytic Method for Matching Approach Selection, *Proc. of the Int. Workshop on Ontology Matching*, 2006.
10. OntoWeb Consortium, A survey on ontology tools, Deliverable 1.3, 2002.
11. Shvaiko P, Euzenat J, A Survey of Schema-Based Matching Approaches, *Journal on Data Semantics*, IV:146-171, 2005.

12. Stuckenschmidt H, Klein M, Structure-based partitioning of large concept hierarchies, *Proc. of the 3rd Int. Semantic Web Conference*, 289-303, 2004.
13. Uschold M, Jasper R, A Framework for Understanding and Classifying Ontology Applications, *Proc. of the IJCAI Workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends*, 1999.