Result of Ontology Alignment with RiMOM at OAEI'06

Yi Li, Juanzi Li, Duo Zhang, and Jie Tang
Department of Computer Science and Technology, Tsinghua University
{ly, ljz, zhangduo, tangjie}@keg.cs.tsinghua.edu.cn

Abstract. In this report, we briefly describe our system RiMOM and its underlying techniques. Given two ontologies, RiMOM intends to combine multiple strategies, aiming at finding the "optimal" alignments from the source ontology to the target one. RiMOM integrates multiple strategies: edit-distance based strategy, statistical-learning based strategy, and three similarity-propagation based strategies. Each strategy is defined based on one kind of ontological-information/approach. RiMOM conducts alignment finding as follows. It first estimates two factors respectively approximately representing the structure similarity and the label similarity of the two ontologies. The two factors are used in strategy selection to determine which strategies will be used in the alignment task. Then, we apply the selected strategies to find the alignment independently and combine the alignment results. Finally we employ the alignment refinement to prune "unbelievable" alignments. This report presents our results based on the evaluation. We also share our thoughts on the experiment design, showing specific strengths and weaknesses of our approach.

1. PRESENTATION OF THE SYSTEM

Ontology alignment is the key point to reach interoperability over ontologies. In semantic web environment, ontologies are usually distributed and heterogeneous and it is necessary to find the mapping between them before processing across them. In recent years, much research work has been conducted for finding the alignment of ontologies [1] [4].

RiMOM is a tool for ontology alignment by combining different strategies, aiming at finding the "optimal" alignment results [5]. Each strategy is defined based on one kind of information or one type of approach. In our current version, there are five strategies defined: edit-distance based strategy, statistical-learning based strategy, and three similarity-propagation based strategies (including concept-to-concept propagation strategy (CCP), property-to-property propagation strategy (PPP), and concept-to-property propagation strategy (CPP)).

1.1 State, purpose, general statement

We here define ontology alignment as a directional one. Given an alignment from ontology O_1 to O_2, we call ontology O_1 as source ontology and O_2 as target ontology. We call the process of finding the alignment from O_1 to O_2 as (Ontology) alignment discovery or alignment finding.

Challenges for automating ontology alignment include: 1) how to automatically find alignments of high quality; 2) how to find the alignments efficiently; 3) how to make full use of the user interaction, since entirely automatic alignment is usually not possible; 4) how to automatically adjust the strategies for finding the alignments in a specific task, since the characteristics of the ontologies to be aligned are different in different tasks; 5) how to ease parameterizing, as the accuracy of alignments may vary largely with different parameters.

In this campaign, we focus on dealing with the problems of 1), 2), and 4) with our system RiMOM.

1.2 Specific techniques used

There are six major steps in the alignment process of RiMOM:

1) Similarity factors estimation. Given two ontologies, it estimates two similarity factors, which respectively approximately represent the structure similarity and the label similarity of the two ontologies. The two factors are used in the next step of strategy selection.

2) Strategy selection. The basic idea of strategy selection is if two ontologies have high label similarity factor, then RiMOM will rely more on linguistic based strategies; while if the two ontologies have high structure similarity factor, then we will employ similarity-propagation based strategies on them. See Section 1.2.2 for details.

3) Strategy execution. We employ the selected strategies to find the alignment independently. Each strategy outputs an alignment result.

4) Alignment combination. It combines the alignment results obtained by the selected strategies. The combination is conducted by a linear-interpolation method.

5) Similarity propagation. If the two ontologies have high structure similarity factor, RiMOM employs an algorithm called similarity propagation to refine the found alignments and to find new alignments that can not be found using the other strategies. Similarity propagation makes use of structure information.

6) Alignment refinement. It refines the alignment results from the previous steps. We defined several heuristic rules to remove the "unbelievable" alignments.

1.2.1 Multiple strategies

The strategies defined in RiMOM can be classified into two categories: linguistic based strategies and structure based strategies.

1. Linguistic based strategies

RiMOM contains two kinds of linguistic based strategies: edit-distance based strategy and statistical-learning based strategy. In our current version of RiMOM, for the statistical-learning based strategy, we use the classification method of K-Nearest Neighbor (KNN). For facilitating the description, we hereafter write the two strategies as ED and KNN.

In ED, we calculate the edit distance between labels of two entities. In KNN, we formalize the problem of alignment as a problem of text classification. We view $e_2 \in O_2$ as a class and its label, comment, and instances as a 'document' of the class. The text in a 'document' is tokenized into words. Then we employ stemming and stop

words removing on the words and view the remains as features to train a text classification model. We also add some other general features which prove to be very helpful. For a concept, the features include: the number of its sub concepts, the number of properties it has, and the depth of the concept from "OWL:Thing".

For finding the alignment, we use the same method to generate a 'document' for a concept $e_1 \in O_1$ and also add the general features as that in building the classification model. Then we use the trained classification model to identify which class the document should be classified. In this way, we are able to find which entity in O_2 is the most possible one for an entity $e_1 \in O_1$ to be aligned.

The two strategies can be used for finding alignments independently. They can also be used together. In the latter case, we combine alignments of different strategies by:

$$Map(e_1, e_2) = \frac{\sum_{k=1...n} w_k \sigma\left(Map_k(e_1, e_2)\right)}{\sum_{k=1...n} w_k} \quad (1)$$

where $e_1 \in O_1$ and $e_2 \in O_2$; $Map_k(e_1,e_2)$ is the alignment score obtained by strategy k. w_k is the weight of strategy k. σ is a sigmoid function, which is defined as $\sigma(x) = 1/\left(1+e^{-5(x-\alpha)}\right)$, where α is tentatively set as 0.5.

2. Structure based strategies

The structure information in ontologies is useful for finding the alignments especially when two ontologies share the common/similar structure. According to the propagation theory [2], we define three structure based strategies in RiMOM, namely concept-to-concept propagation strategy (CCP), property-to-property propagation strategy (PPP), and concept-to-property propagation strategy (CPP).

Intuition of the propagation based method is that if two entities are aligned, their super-concepts may also be aligned. The basic idea of the method is to propagate the similarity of two entities to entity pairs with some kinds of relationship with them, for example, subClassOf, superClassOf, siblingClassOf, subPropertyOf, superPropertyOf, range, and domain (superClassOf is not defined in OWL, it is viewed as the converse relationship of subClassOf. Likewise for superPropertyOf. siblingClassOf is not defined also in OWL. It means that the two concepts have the same super concept). The idea is inspired by the algorithm of similarity flooding proposed for schema matching [3]. We extended the algorithm and adaptively used them in the three structure based strategies. Details of the method will be reported elsewhere.

In CCP, we propagate similarities of concepts pair across the concept hierarchical structure. In PPP, we propagate similarities of property pair across the property hierarchy. In CPP, we propagate similarities of concepts pair to their corresponding property pair, and vice versa.

The structure based strategies are employed after the linguistic based strategies. They can be used to adjust the alignments and find new alignments.

1.2.2 Similarity factors estimation

Our preliminary experiments show that the multi-strategy based alignment does not always outperform its single-strategy counterpart. We then consider three questions: (1) for a new, unseen mapping task, should we select a multi-strategy based solution or just one single-strategy based solution? (2) if the task is suitable to use multiple

strategies, then which strategies should be selected so as to obtain better alignment results? (3) the method for strategy selection needs to be efficient. This is important because for aligning large-scale ontologies, the efficiency may be a critical problem. We propose to deal with the problems by using similarity factors estimation.

Given two ontologies: source ontology O_1 and target ontology O_2, we calculate two approximate similarity factors: structure similarity factor and label similarity factor.

We define structure similarity factor as:

$$F\_SS = \frac{\#common\_concept}{\max(\#nonleaf\_c_1, \#nonleaf\_c_2)} \quad (2)$$

where $\#nonleaf\_c_1$ indicates the number of concepts in O_1 that has sub concepts. Likewise for $\#nonleaf\_c_2$. *#common_concept* is calculated as follows: if concepts $c_1 \in O_1$ and $c_2 \in O_2$ have the same number of sub concepts and they are in the same depth from the concept "owl:Thing", we add one to *#common_concept*. After enumerated all pair, we obtain the final score of *#common_concept*. Intuition of the factor is that the larger the structure similarity factor, the more similar the structures of the two ontologies are.

The label similarity factor is defined as:

$$F\_LS = \frac{\#same\_label}{\max(\#c_1, \#c_2)} \quad (3)$$

where $\#c_1$ and $\#c_2$ respectively represent the number of concepts in O_1 and O_2. *#same_label* represents the number of pairs of concepts $\{(c_1, c_2) | c_1 \in O_1 \text{ and } c_2 \in O_2\}$ that have the same label.

The two factors are defined simply and not used to accurately represent the real "similarities" of structures and labels. However, they can approximately indicate the characteristics of the two ontologies. Moreover, they can be calculated efficiently.

So far, we carried out the strategy selection by heuristic rules. For example, if the structure similarity factor *F_SS* is lower than 0.25, then RiMOM suppresses the CCP and PPP strategies. However, the CPP will always be used in the alignment process.

1.3 Adaptations made for the evaluation

No special adaptations have been made. However, some parameters are tuned and set in the experiments. For example, for strategies combination (cf. equation 1), we set the weight of ED as 0.5 and that of KNN as 1. For strategy selection, we define 0.25 as the threshold to determine whether CCP and PPP will be suppressed or not. We also define 0.2 as threshold to determine whether ED will be suppressed or not.

1.4 Link to the system, parameters file, and provided alignments

Our system RiMOM (including the parameters file) can be found at http://keg.cs.tsinghua.edu.cn/project/RiMOM/. For details of the approach, see [5].

The alignment results of the campaign are available at http://keg.cs.tsinghua.edu.cn/project/RiMOM/OAEI2006/.

2 Results

RiMOM has been implemented in Java. We use OWL-API to parse the RDF and OWL files. The experiments were carried out on a Server running Windows 2003 with two Dual-Core Intel Xeon processors (2.8 GHz) and 3-gigabyte memory. All the alignments outputted by RiMOM are based on the same parameters.

2.1 Benchmark

2.1.1 Tests 101-104

The tests 101, 103, and 104 are basic tests for ontology alignment. The source ontologies contain concepts and properties with the same names as those in the reference ontologies.

Both linguistic based strategies and structure based strategies were employed for finding the alignment (because both label similarity and structure similarity factors exceed the thresholds), however, as linguistic based strategies can easily find most of the alignments, the structure based strategies took little effect to the final results. In test 102, RiMOM outputs no alignment.

In the three tests (excluding test 102), both precision and recall are 1.0. The average time cost is 3.36s.

2.1.2 Tests 201-210

Tests 201 through 210 have high structure similarity factor (equal to 1.0) with the reference ontology. Some of the tests have high label similarity factor (e.g. test 203), some have synonym labels with the reference ontology (e.g. test 205 and 209), and some others have low label similarity factor (e.g. tests 201, 202, 206, 207, and 210).

Using the strategies selection method, we are able to apply different strategies in the different tests. For example, for test 201, where label of concepts and properties are replaced by a random ones, ED is suppressed and KNN and the structure based strategies are active. Using KNN, we can find some matched concept pairs and property pairs. Then based on the matched pairs, we utilize the structure based strategies to find the other alignments that cannot be found by KNN.

In the ten tests, precision ranges from 0.88 to 1.0 and recall stays between 0.82 and 1.0. The average time cost is 2.638s.

2.1.3 Tests 221~247

For most of these tests the structures are changed, which means that the structure similarity factors are low, however the label similarity factors are very high.

For tests that have low structure similarity factors, we suppress the structured based strategies, for example, tests 221, 232, 233, and 241. (Note: CPP is still active.) For tests that have both high label similarity factor and structure similarity factor, both linguistic based strategies and structure based strategies were employed, although structure based strategies made little contribution.

In these tests, precision ranges from 0.94 to 1.0 and recall equals to 1.0. The average time cost is 1.99s.

2.1.4 Tests 248~266

These tests were the most challenging ones to our approach. Labels and comments have been removed and structures have also been changed as well. In this case, both label similarity factor and structure similarity factor between the source ontologies and the reference ontology are low. For most of the tests, we found that KNN is the most useful one and the other strategies take little effects. In tests 249, 250, and 257, the structure based strategies took effect to help improve the final alignments.

In these tests, precision ranges from 0.73 to 1.0 and recall stays between 0.27 and 0.82. The average time cost is 1.59s.

2.1.5 Tests 301~304

In tests 301-304, the source ontologies are from real world, modeled by different institutions but for the same domain of bibliographic metadata. The real-world tests combine the difficulties of the previous tests.

In the tests, based on the strategy selection method, both linguistic based strategies and structure based strategies were employed except the test 301, where we only applied linguistic based strategies.

In these tests, precision ranges from 0.77 to 0.9 and recall stays between 0.69 and 0.97. The average time cost is 3.14s.

2.2 directory

The directory ontologies are organized as a taxonomy with sub-sumption hierarchies. We use two methods to obtain the alignment results. The first one was obtained by using RiMOM with the same set of parameters as the ones for benchmark test. Both linguistic based strategies and structure based strategies were employed in this task. The results seem to be not ideal.

The other alignment result was obtained by a specific version of RiMOM, called RiMOM-directory. In RiMOM-directory, except ED and KNN, we also integrate another strategy based on Wordnet, one of the most popular thesauruses (called as Wordnet hereafter). Because in directory alignment, there are many synonym words used in the labels, Wordnet is expected to be useful. We also made some other adaptation, for example, for structure based strategies we only use CCP, as there is no property and instances in the directory data (also in CCP, we only consider the relationship “OWL:subClassOf”).

We obtained three alignment results using RiMOM-directory with different strategies: 1) linguistic based strategies (including ED, KNN, and Wordnet) only. In this case, the precision, recall, and F1-measure are 0.36, 0.33, and 0.35 respectively; 2) both linguistic based strategies and the CCP strategy (with only one iteration of propagation). The precision, recall, and F1-measure are 0.39, 0.40, and 0.40 respectively; 3) same setting as that in 2) but with *n* iterations. The precision, recall, and F1-measure are 0.38, 0.40, and 0.39 respectively.

2.3 anatomy

RiMOM met problems in parsing the anatomy ontologies and finally outputs no alignments.

2.4 food

The ontologies in the food test are large and RiMOM suppressed the structure based strategies and applied only a simple version of the linguistic based strategies for finding the alignment.

3 General comments

3.1 Comments on the results

An objective and comprehensive comment on strengths or weakness requires the comparison with other participants, which are not available so far (will be available before the workshop). Here, we share some thoughts about the results.

Strengths

From experimental results, we see that RiMOM can achieve high performance when the ontologies to be aligned have similar linguistic information or similar structure information. Some concluding remarks are summarized as follows:

1) Linguistic information (including label of concepts and properties) is important and help to align most of the entities.

2) Structure information can be used to improve the alignments, in particular when linguistic information is missing.

3) Strategy selection is important. In different alignment tasks, the ontologies to be aligned have different characteristics, it would be particularly helpful to find the characteristics of the ontologies and apply correspondingly strategies on them.

4) Alignment refinement is helpful. In refinement, we removed the unbelievable alignments, which improves the precision in many tests.

5) RiMOM can find the alignment quickly. The time costs range from 0.69s to 6.70s in the benchmark tests.

Weakness

1) RiMOM cannot deal with large-scale ontologies. The biggest problem here is that our structure base strategies cannot efficiently do the propagation in the large graph (by viewing the ontology as a graph).

2) We met problems when dealing with the anatomy ontologies.

3) We note that parameter setting is very important. We have found that using different parameter settings, with the exactly same approach, the alignment results may differ largely. So far, we tuned the parameters manually. It is not adaptable in particular when the ontologies are very large, which means that tuning different parameters to find the best ones is not possible.

3.2 Discussions on the way to improve the proposed system

Possible improvements are corresponded to the related weaknesses in the previous section.

1) Our proposal is to partition the large ontologies into small slices and then employ the structure based strategies on the slices.

2) Efforts are being made to integrate a more powerful parser into our system.

3) Our thinking is to use a supervised machine learning method to find the optimal parameters based on some training data sets.

3.3 Comments on the OAEI 2006 test cases

The benchmark tests indicate very interesting general results on how the alignment approach behaves. These tests are really useful, as a good underlying test base, for evaluating and improving the alignment algorithm and system.

For future work, it might be interesting to add some tests to evaluate the time cost of systems, as for large-scale ontology alignment, the issue of efficiency may be critical.

4 Conclusion

In this report, we have briefly introduced our approach and the tool, that is called RiMOM, for finding ontology alignment. We have presented the alignment process of RiMOM and explained each step. We applied the tool to the test data and the experimental results show that our proposed approach can achieve high performance quickly. We summarized the strengths and the weaknesses of our proposed approach and gave possible improvement for the system in the future work.

References

[1] J. Euzenat. State of the art on ontology alignment. http://www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/. August, 2004.

[2] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. International Journal of Computer Vision, Vol. 70, No. 1, October 2006.

[3] S. Melnik, H. Garcia-Molina and E. Rahm: Similarity Flooding: a versatile graph matching algorithm and its application to schema matching. In Proc. of 18th ICDE. San Jose CA, Feb 2002. pp. 117-128

[4] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. The VLDB Journal, 2001, 10:334-350.

[5] J. Tang, J. Li, B. Liang, X. Huang, Y. Li, and K. Wang. Using Bayesian decision for ontology mapping. Journal of Web Semantics. To be published.

Appendix: Raw results

The following results were obtained in the evaluation runs.

Matrix of results

#	Name	Prec.	Rec.	Time
101	Reference alignment	1.00	1.00	4.72s
102	Irrelevat ontology	N/A	N/A	
103	Language generalization	1.00	1.00	2.84s
104	Language restriction	1.00	1.00	2.52s
201	No names	1.00	1.00	1.98s
202	No names, no comments	1.00	0.82	1.56s
203	No comments	1.00	1.00	2.53s
204	Naming conventions	1.00	1.00	2.72s
205	Synonyms	1.00	0.99	3.88s
206	Translation	1.00	0.99	5.16s
207		1.00	0.99	2.27s
208		0.98	0.98	2.48s
209		0.88	0.87	1.88s
210		0.99	0.89	1.92s
221	No specialisation	1.00	1.00	2.28s
222	Flatenned hierachy	1.00	1.00	2.58s
223	Expanded hierarchy	1.00	1.00	4.83s
224	No instance	1.00	1.00	2.75s
225	No restrictions	1.00	1.00	2.66s
228	No properties	1.00	1.00	1.00s
230	Flatenned classes	0.94	1.00	2.19s
231		1.00	1.00	3.00s
232		1.00	1.00	2.13s
233		1.00	1.00	0.77s
236		1.00	1.00	0.94s
237		0.99	1.00	2.42s
238		1.00	1.00	3.52s
239		0.97	1.00	1.08s
240		0.97	1.00	1.06s
241		1.00	1.00	0.69s
246		0.97	1.00	1.06s
247		0.97	1.00	1.02s
248		1.00	0.81	2.03s
249		1.00	0.82	1.74s
250		1.00	0.55	0.92s
251		0.74	0.59	1.80s
252		0.84	0.71	4.80s
253		1.00	0.81	1.61s
254		1.00	0.27	0.72s
257		1.00	0.55	0.89s
258		0.73	0.59	2.19s
259		0.84	0.71	2.17s
260		0.87	0.45	1.03s

261		0.82	0.27	1.33s
262		1.00	0.27	0.69s
265		0.87	0.45	1.03s
266		0.82	0.27	0.92s
301	BibTeX/MIT	0.82	0.74	1.78s
302	BibTeX/UMBC	0.77	0.69	1.73s
303	Karlsruhe	0.77	0.84	6.70s
304	INRIA	0.90	0.97	2.36s