RiMOM Results for OAEI 2009

Xiao Zhang, Qian Zhong, Feng Shi, Juanzi Li and Jie Tang

Department of Computer Science and Technology, Tsinghua University, Beijing, China
zhangxiao,zhongqian,shifeng,ljz,tangjie@keg.cs.tsinghua.edu.cn

Abstract. In this report, we give a brief explanation of how RiMOM obtains the results at OAEI 2009 Campaign, especially in the new Instance Matching track. At first, we show the basic alignment process of RiMOM and different alignment strategies in RiMOM. Then we give new features in instance matching compared with traditional ontology matching (schema matching) and introduce the specific techniques we used for the 3 different subtracks of Instance Matching Track. At last we give some comments on our results and discuss some future work about RiMOM.

1 Presentation of the system

Ontology matching is the key technology to reach interoperability over ontologies. In recent years, much research work has been conducted for finding the alignment of ontologies[1]. Many automatic matching algorithms achieves good results in real world data. With the development of Linked Data[2], huge amount of semantic data are available through the web. Thus instance matching, a special branch of ontology matching, draws lots of research interest recent years.

RiMOM is a multiple strategy dynamic ontology matching system implemented in Java [3]. In RiMOM, we implement several different matching strategies. Each strategy is defined based on one kind of ontological information. Moreover, we investigate the differences between the strategies and compare the performances of different strategies on different matching tasks. We propose a mechanism in RiMOM to choose appropriate strategies (or strategy combination) according to the features and the information of the ontologies. RiMOM can deal with unbalanced ontology matching [4]. We also try to bring user interaction into RIMOM [5]. This year We modified the RiMOM system to make it with better support for instance matching.

1.1 State, purpose, general statement

RiMOM is a framework for ontology matching. Different kinds of alignment strategies can be added into RiMOM. Based on the features of the input ontology and the defined rules, appropriate strategies are chosen to apply for the matching task. The basic matching process of RiMOM is shown in figure 1.

There are six major steps in a general alignment process of RiMOM.

– Ontology Preprocessing and Feature Factors Estimation. The input ontologies are loaded into the memory and the ontology graph is constructed. Some redundant or useless information are removed. Then the three ontology feature factors used in strategy selection are estimated.

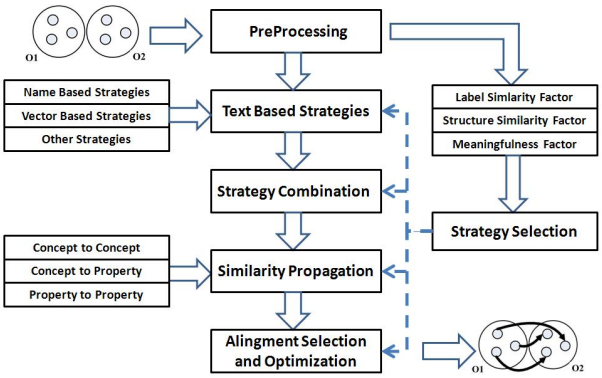

Fig. 1. The Alignment Process of RiMOM

- Strategy Selection. The basic idea of strategy selection is that if two ontologies have some same feature, then strategies based on these feature information are employed with high weight; and if some feature factors are two low, then these strategies may be not employed. For example, the string based strategy will be used when the label Similarity factor is high while the WordNet [6] based stratey will not be used when the label meaningful factor is low.
- Single strategy execution. We get the selected strategies to find the alignment independently. Each strategy outputs an alignment result.
- Alignment combination. In this phase RiMOM combines the alignment results obtained by the selected strategies. The combination is conducted by a linear-interpolation method.
- Similarity propagation(Optional). If the two ontologies have high structure similarity factor, RiMOM employs a similarity propagation process to refine the found alignment and to find new alignment according to the structural information.
- Alignment refinement. It refines the alignment results from the previous steps. We defined several heuristic rules to remove the ”unreliable” alignments.

1.2 Specific techniques used

This year we participate four tracks of the campaign: Benchmark, Anatomy, Oriented Matching and Instance Matching. The Benchmark and Anatomy dataset is almost the same as last year. The Oriented Matching dataset is very similar to the Benchmark one. We focused on the new and challenging Instance Matching Track.

Benchmark and Anatomy Track The strategy we use for Benchmark and Anatomy track is almost the same for OAEI 2008, more detailed explanation of the strategies used could be found in [7] [8].

Oriented Matching Track The dataset of oriented matching track is derived from the benchmark dataset. Naturally, we combined the methods we use in the benchmark track and the sub relation in the ontology graph. Since RiMOM's performance for the Benchmark Track is fairly good, the result shows that the combination also works for the Oriented Matching Track. This technique is also applied for the schema matching phase of the Instance Matching Track.

Instance Matching Track The Instance Matching Track is introduced into the campaign this year. The traditional ontology matching focus on the schema matching and the ontology may contain no individual. If there are small amount of individuals, the alignment of individuals are usually used to enhance the alignment of concepts and properties. Previous OAEI campaigns also evaluated the performance of Matching Systems according to the schema matching results. By analyzing the datasets, we found some differences between the traditional ontology matching and instance matching. We summarize the differences as following:

- Ontology is used as a formal, explicit specification of a shared conceptualization[9]. It defines the concepts of the domain and the relation between the concepts. That is to say, it describe the domain in the concept layer. However, the instance is the instantiation of the ontology, it is composed of concrete values of the domain, and has rich practical semantic information. As we observe, some attribute values may clearly different from others. How to find the key attributes and key values of instances to facilitate the process of ontology matching is a very challenging issue.
- The ontology can be viewed as a whole ontology graph and some graph based algorithms are employed in ontology matching. However, a concept may have lots of instances and all the instances are with almost the same structure. The graph algorithm with the whole ontology graph is not suitable for the instance matching task. For a given instance, some other instances related to it may contain information about it through object properties. Moreover, the information in the instance may be not symmetric as in the ontology. For instance, an instance A of the "Author" concept is "list_author_in" an instance B of the "Scientific_Publication" concept. However, the statement is only written in B's description. How to find the complete information of a given instance is also very important.
- The scale of instance matching is usually much more larger than ontology matching. The sweto data and the dbpedia data file both contain more than 800,000 instances which seems impossible in ontology matching. As a result, the efficiency of the instance matching strategies becomes a major concern. Some complicate algorithm can not be employed.
- The schema ontology for most instance files are available. So the result of ontology matching is a very good background knowledge for instance matching. Instinctively, the instance pair of not matched concept pair have no chance to be matched. So with the ontology alignment, the number of instance matching candidates can be pruned greatly. However, sometimes the ontology itself is not very well defined. For example, The DBpedia ontology does not cover all the instances in the instance file, so the ontology itself should be enriched with instance type information.

With regard to the different characteristics of the three different subtracks of Instance Matching (We do not take part in the vlcr subtrack), we employ some different strategies to solve the three tasks.

The A-R-S benchmark includes three datasets containing instances from the domain of scientific publications. The three data file are quite different in size, especially the DBLP file is really large. So we use some light-weight method in this subtrack. On the other hand, all the three data files are mainly in the scientific publication domain. At first, we choose some data-type properties as the key attributes carefully. These properties are of two types. The first type is the "sufficient" property group: if the values of the properties between two instances are matched respectively, the two instances are considered as matched. The second type is the "necessary" property group: if two instances are marked as matched, the corresponding values should be matched. The "sufficeint" properties, such as {foaf:name} are employed in a edit-distance strategy to find the initial alignment. The "necessary" properties, such as the opus:year, are employed to refine the initial alignment. In the second step, a structure based matching method is used to propagate the similarities among the instances according to the object properties. For example, we can refine the "person" matches with the "document" alignments in terms of the {opus:author} property.

Compared with the A-R-S benchmark which is restricted to the scientific publication domain, the T-S-D benchmark covers much more wider domains. The DBpedia data is encyclopedia-like knowledge base. This makes it difficult to find particular attributes and values, so we take another strategy. First of all, we compute the schema matching results with RiMOM, check the incorrect alignments and add the missing ones carefully. Then for every instance, we generate a vector to describe the information contained in the instance. The vector contains labels of the instance, data-type property values of the instance, labels and property values of the instances related to the underlying one through object-type properties. Then the similarity of the instance pair of matched concept pair is calculated by a vector based algorithm. The weight of respective element of the vector is designated by heuristic rules defined based on the structure of the instance. The instance pairs of non-matched concept pairs are discarded directly. In the DBpedia data file, there are some instances missing the rdf:type information. We try to match these instances with all source instances in the reference file.

The IIMB benchmark, on the other hand, is systematically generated. The dataset is constituted using one data file by modifying it according to various criteria. The variation can be sorted into three types: value transformation, structural transformation and logical transformation. The purpose of value transformation is to simulate typographical errors, so edit-distance strategy employed on the relevant property values between instances is effective enough. In structural transformation, some data-type properties may transformed in the form of object-type property. We design an property value passing approach to cope with this kind of modification. The data-type property value of the instance are passed to the instances connected with it through a object-type property. We also consider structural information when matching instances. If two instances have more property values on the same properties, they will be considered more similar. In logical transformation, the TBox is modified, so we matching the TBox first to find the find the relations between concepts in the TBoxes, then we try to match instances

according to the type information. In addition, some instance pairs with very similar property values but with non-matched concepts are checked. If they can match each other, then we consider their concepts are matched to enhance the TBox alignment. When the two-direction matching process convergence to a stable matching result, we take it as the final output.

1.3 Adaptations made for the evaluation

To reduce the number of matching candidate in T-S-D benchmark subtrack of Instance Matching Track, the schema matching alignments is refined manually by correcting some incorrect alignments and adding missing ones.

1.4 Link to the system and parameters file

The RiMOM System can be found at `http://keg.cs.tsinghua.edu.cn/project/RiMOM/`

1.5 Link to the set of provided alignments (in align format)

The results for OAEI 2009 Campaign are available at `http://keg.cs.tsinghua.edu.cn/project/RiMOM/OAEI2009/`

2 Results

As mentioned above, RiMOM takes part in four tracks of campaign in OAEI 2009. Normally RiMOM uses OWL-API[10] to parse RDF and OWL Files. RiMOM also uses Jena API[11] to convert N3 format files into RDF files and to deal with some large scale instance files. The Benchmark, Oriented Matching and IIMB matching tasks are carried out on a PC running Window XP with AMD Athlon 64 X2 4200+ processor(2.19GHz) and 2GB memory. To run the large scale matching tasks, Anatomy, A-R-S benchmark and T-S-D benchmark, the experiments are carried out on a server running Ubuntu Server 8.10 with two 4-core Intel Xeon E5440 processors(2.83GHz) and 32GB memory.

2.1 Benchmark

There are in total 111 alignment tasks defined on the benchmark data set. RiMOM takes exactly the general process of matching. However, on the tasks where the labels are absolutely random strings, the WordNet based strategy and edit-distance based strategy are not used. The vector-similarity based strategy is always employed. RiMOM maintains the high performance on benchmark as previous years.

2.2 Anatomy

The anatomy data set contains two large scale anatomy ontologies. RiMOM first extract the labels of all concepts from rdfs:label and oboInOwl:Synonym property. The match process first employs edit-distance based strategy on labels to get the initial mapping, then RiMOM propagates the similarity on both the concept hierarchy ant the object property "UNDEFINED_part_of" to get the alignments which cannot be extracted by just comparing the labels simply. Since the structure of the two ontologies is somehow not that similar, we restricted the propagation for every concept locally.

2.3 Oriented Matching

Because our strategy in oriented matching is the combination of the strategy in the Benchmark dataset and structure based stratedy by using the rdfs:subclass property. The result relies heavily on the Benchmark strategy and shows the same characteristics as in the Benchmark dataset. Except in the files the name of the entities are totally random string and nearly no other information are available, RiMOM achieves satisfying results.

2.4 Instance Matching

The result for A-R-S benchmark is as Table 1 shows. RiMOM produces alignments all with an F-Measure in the range of about 0.75+. The result relate to eprints data have both high precision than the rexa-dblp one.

Table 1. Result of A-R-S Benchmark

Data	Precision	Recall	F-Measure
eprints-rexa	0.928	0.699	0.797
eprints-dblp	0.930	0.671	0.780
rexa-dblp	0.805	0.725	0.763

The T-S-D benchmark is a blind test, so we do not know the final results for it now. According to our observation on our alignment, about 30% to 50% of the instances in the reference files are matched. It seems indeed that most of instances in the reference can not find a correspondence. Since we choose a relatively high threshold in the final alignment extraction, we believe the result is of high precision.

Except the dataset 028 which seems missing some correct alignments and the dataset 029 which do not contain a reference alignment, the result for IIMB benchmark is as Table 2 shows. We can see that RiMOM can achieve perfect alignment in more than half of the dataset. Only for the dataset 017 in which the information is severely suppressed, RiMOM can only get an F-Measure less than 0.90. RiMOM is quite successful in IIMB dataset.

Table 2. Result of A-R-S Benchmark

Dataset	Precision	Recall	F-Measure
001 - 014	1.0	1.0	1.0
015	1.0	0.991	0.995
016	1.0	0.910	0.953
017	0.993	0.626	0.768
018	1.0	0.986	0.993
019	1.0	0.883	0.938
020 - 027	1.0	1.0	1.0
030	1.0	1.0	1.0
031	1.0	0.892	0.943
032 - 037	1.0	1.0	1.0

3 General comments

Except for performing the ontology matching tasks like the previous years, this year we concentrate on the new and interesting Instance Matching Track. We first modify the infrastructure of RiMOM to make it to support instance matching naturally. The results shows that now RiMOM can handle instance matching tasks with good performance. But there are still many future works to do:

- Although instance matching is regarded as a subtask of ontology matching, the model of instance matching is different from traditional schema matching to some extent. Some algorithms in schema matching can not be imported into instance matching directly. In addition, instance matching seems more close to practical use than schema matching. This makes it a very attractive research topic.
- The scalability problem is very critical in instance matching. The scale of instance files are greatly larger than the schema files and the execution times and memory needs grows very fast as the input scale increases. For example, our strategy for A-R-S benchmark consumes more than 36 hours to generate the alignment on our 8-core server while the strategy itself is not that complicated. How to solve this problem is a big challenge. We may try to introduce the database-like techniques into RiMOM to make it support the large scale instance data better.
- Because the instance data are retrieved from the real web data, so it usually contains more semantic information than the theoretically designed schemas. However, most of our approaches are string based comparisons and so on. How to dig the deeperthe semantics in the instance is another work.

4 Conclusion

In this report, we give aa briefly explanation of how we employed RiMOM to obtain the alignment results in OAEI 2009 Campaign. We have presented the alignment process of RiMOM and explained the strategy defined in RiMOM. We focus on the Instance Matching Track, analyzing the feature of instance matching and introduce the strategies

we use in this track. The experiments illustrates that our system RiMOM can achieve good results in both schema matching and instance matching tracks. We also discuss the future work we will do to improve our system.

Acknowledgement

The work is supported by the National Natural Science Foundation of China (No. 60973102 and No. 60703059), the National Basic Research Program of China (973 Program) (No. 2007CB310803), the National High-tech R&D Program (No. 2009AA01Z138), and the Chinese Young Faculty Research Fund (No. 20070003093).It is also supported by IBM SUR joint project.

References

1. J. Euzenat and P. Shivako. *Ontology Matching*. Springer-Verlag, Berlin Heidelberg (DE), 2007.
2. `http://linkeddata.org/`.
3. J. Li, J. Tang, Y. Li, and Q. Luo. RiMOM: A dynamic multi-strategy ontology alignment framework. *IEEE Transaction on Knowledge and Data Engineering*, 21(8):1218–1232, Aug 2009.
4. Q. Zhong, H. Li, J. Li, G. Xie, and J. Tang. A Gauss Function based approach for unbalanced ontology matching. In *Proc. of the 2009 ACM SIGMOD international conference on Management of data (SIGMOD'2009)*, Jul 2009.
5. F. Shi, J. Li, and J. Tang. Actively learning ontology matching via user interaction. In *Proc. of the 8th International Conference of Semantic Web (ISWC'2009)*, Oct 2009.
6. `http://wordnet.princeton.edu/`.
7. Y. Li, J. Li, D. Zhang, and J. Tang. Results of ontology alignment with RiMOM. In *Proc. of the Second International Workshop on Ontology Matching (OM'07)*, Nov 2007.
8. X. Zhang, Q. Zhong, J. Li, J. Tang, G. Xie, and H. Li. RiMOM results for OAEI 2008. In *Proc. of the Third International Workshop on Ontology Matching (OM'08)*, 2008.
9. T. R. Grubber. A translation approach to portable ontology specification. *Knowledge Acquisition*, 5:199–200, 1993.
10. `http://owlapi.sourceforge.net/`.
11. `http://jena.sourceforge.net/`.