Falcon-AO: Aligning Ontologies with Falcon

Ningsheng Jian, Wei Hu, Gong Cheng, Yuzhong Qu

Department of Computer Science and Engineering
Southeast University
Nanjing 210096, P. R. China

{nsjian, whu, gcheng, yzqu}@seu.edu.cn

ABSTRACT

Falcon-AO is an automatic tool for aligning ontologies. There are two matchers integrated in Falcon-AO: one is a matcher based on linguistic matching for ontologies, called LMO; the other is a matcher based on graph matching for ontologies, called GMO. In Falcon-AO, GMO takes the alignments generated by LMO as external input and outputs additional alignments. Reliable alignments are gained through LMO as well as GMO according to the concept of reliability. The reliability is obtained by observing the linguistic comparability and structural comparability of the two ontologies being compared. We have performed Falcon-AO on tests provided by OAEI 2005 campaign and got some preliminary results. In this paper, we describe the architecture and techniques of Falcon-AO in brief and present our results in more details. Finally, comments about test cases and lessons learnt from the campaign will be presented.

Categories and Subject Descriptors

D.2.12 [**Software**]: Interoperability; I.2.6 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods; I.5.3 [**Pattern Recognition**]: Clustering—*Similarity measures*

General Terms

Experimentation, Measurement

Keywords

Semantic Web, Ontology Alignment, Mapping, Matching, Similarity Measurement

1. PRESENTATION OF THE SYSTEM

As an infrastructure for semantic web applications, Falcon is a vision of our research group. It will provide enabling technologies for finding, aligning and learning ontologies, and ultimately for capturing knowledge by an ontology-driven approach. It is still under development in our group. As a component of Falcon, Falcon-AO is an automatic tool for aligning ontologies. It is dedicated to aligning web ontologies expressed in OWL DL [5]. There are two matchers integrated in current version of Falcon-AO (version 0.3). One is a matcher based on linguistic matching for ontologies, called LMO, and the other one is a matcher based on graph matching for ontologies, called GMO.

1.1 Linguistic Matching for Ontologies

As is known, linguistic matching plays an important role in matching process. Generally, linguistic similarity between two entities relies on their names, labels, comments and some other descriptions.

LMO combines two different approaches to gain linguistic similarities: one is based on lexical comparison; the other is based on statistic analysis.

In lexical comparison, we calculate the edit distance [4] between names of two entities and use the following function to capture the string similarity (denoted by SS):

$$SS = 1/e^{\frac{ed}{|s1.len+s2.len-ed|}} \quad (1)$$

Where ed denotes the edit distance between $s1$ and $s2$; $s1.len$ and $s2.len$ denote the length of the input strings $s1$ and $s2$, respectively.

In statistic analysis, we use the algorithm of VSM [6] (Vector Space Model) in our implementation. Given a collection of documents, we denote N the number of unique terms in the collection. In VSM, we represent each document as a vector in an N-dimensional space. The components of the vector are the term weights assigned to that document by the term weighting function for each of the N unique terms. Clearly, most of these

are going to be 0, since only a few of the N terms actually appear in any given document. In our scenario, we construct a virtual document for each of the ontology entities (classes, properties and instances). The virtual document of an entity consists of "bag of terms" extracted from the entity's names, labels and comments as well as the ones from all neighbors of this entity. The term weighting functions are defined as follows:

$$TermWeighting = TF * IDF \quad (2)$$

$$TF = \frac{t}{T} \quad (3)$$

$$IDF = \frac{1}{2} * (1 + log_2 \frac{D}{d}) \quad (4)$$

In equation (3), t denotes the number of times where one term occurs in a given document and T denotes the maximum number of times. In equation (4), D denotes the number of documents in collection and d denotes the number of documents where the given term occurs at least once.

We can gain the cosine similarity between documents (denoted by DS) by taking the vectors' dot product:

$$DS = N \cdot N^t \quad (5)$$

It is worthy of note that there are several preparing steps before calculating term weights, such as splitting words, stemming and removing stop words.

The two methods described above will both take effect in ontology matching. In our implementation, we combine them together, and use the following equation to calculate the final linguistic similarity. Please note that the parameters in the equation comes from our experience:

$$LinguisticSimilarity = 0.8 * DS + 0.2 * SS \quad (6)$$

Currently, we do not use any lexicons in LMO and it is certain that the use of lexicons may bring some benefits for matching. We plan to take into account using some lexicons in later versions.

1.2 Graph Matching for Ontologies

Another important component in Falcon-AO is GMO, which is based on a graph matching approach for ontologies. It uses directed bipartite graphs to represent ontologies and measures the structural similarity between graphs by a new measurement. Details of the approach are described in another paper [3] also presented in the K-Cap 2005 Workshop on Integrating Ontologies [1].

The main idea of GMO is as follows. Similarity of two entities from two ontologies comes from the accumulation of similarities of involved statements (triples) taking the two entities as the same role (subject, predicate, object) in the triples, while the similarity of two statements comes from the accumulation of similarities of involved entities of the same role in the two statements being compared.

Usually, GMO takes a set of matched entity pairs, which are typically found previously by other approaches, as external mapping input in the matching process, and outputs additional matched entity pairs by comparing the structural similarity.

Our previous experiments showed that GMO were irreplaceable when there was little gain from lexical comparison. In addition, GMO can be integrated with other matchers. While using GMO approach to align ontologies, there should be another component to evaluate reliability of alignments generated by GMO.

1.3 Linguistic vs. Structural Comparability

Given two ontologies to be aligned, GMO always tries to find all the possible matched entity pairs. However, how to evaluate the reliability of these matched entity pairs is still a problem. As mentioned above, another component is needed to select more reliable matched entity pairs by using other information. In Falcon-AO, we use a simple approach to observe the reliability of matched entity pairs output by GMO, and select more reliable matched entity pairs to the users. The approach is based on the measure of linguistic comparability (LC) and structural comparability (SC) of two ontologies to be aligned.

Given two ontologies O_1, O_2 to be aligned, the linguistic comparability *(LC)* of O_1 and O_2 is defined as follows:

$$LC = \frac{M}{\sqrt{N_{O_1} * N_{O_2}}} \quad (7)$$

Where M denotes the number of entity pairs with similarity larger than c and c is an experience value; N_{O_1} and N_{O_2} represent the number of named entities in O_1 and O_2, respectively.

The structural comparability is determined through comparing the occurrences of built-in properties used in the two ontologies to be aligned. The built-in properties are RDF [2], RDFS [1] and OWL [5] built-in vocabularies used as properties in triples (e.g. rdf:type, rdfs:subClassOf and owl:onProperty).

[1] http://km.aifb.uni-karlsruhe.de/ws/intont2005

We use VSM method to observe the structural comparability. The vectors V_1, V_2 represent the frequency of built-in properties used in O_1 and O_2 and the element v_{ij} denotes the number of occurrence of built-in property p_j in O_i. The structural comparability of O_1 and O_2 is the cosine similarity [7] of V_1 and V_2:

$$SC = \frac{V_1 \cdot V_2}{\|V_1\| \ \|V_2\|} = \frac{\sum_{j=1}^{n} v_{1j} * v_{2j}}{\sqrt{\sum_{j=1}^{n} v_{1j} * v_{1j}} \ \sqrt{\sum_{j=1}^{n} v_{2j} * v_{2j}}} \quad (8)$$

1.4 Implementation

LMO and GMO are integrated in Falcon-AO. Alignments output by Falcon-AO come from the integration of alignments generated by LMO and GMO. The architecture of Falcon-AO is shown in Figure. 1.

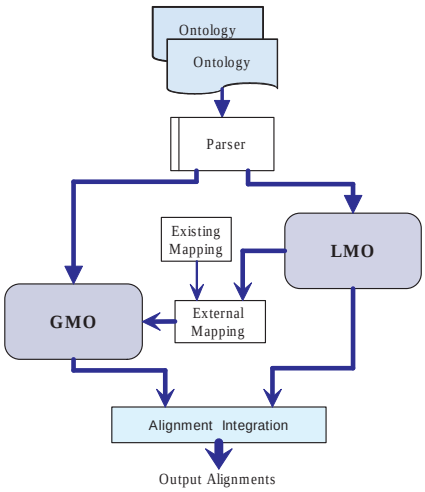

Figure 1: System Architecture

Due to heterogeneous ways in expressing semantics and the inference capability brought from ontology languages, two ontologies being matched may need to be coordinated by removing some redundant axioms from it or adding some inferred axioms. So coordination actions should be taken before using GMO approach. We have integrated several coordination rules in Falcon-AO. Our *Parser* component based on Jena [2] has the functionality of coordinating ontology models.

As is known, given external mapping as input, GMO can find additional mapping. The external mapping is made of two parts: one is the existing mapping preassigned by the system; the other comes from another matcher. The existing mapping is the mapping between built-in vocabularies of web ontology languages, datatypes, data literals and URIs used in both ontologies. And in Falcon-AO we take the alignments generated by LMO as the other part of external mapping. Entities involved in the alignments generated by LMO are set to be external entities and GMO will just output mapping between internal entities.

When the alignments generated by LMO and GMO are obtained, Falcon-AO will integrate these alignments by observing the linguistic comparability and structural comparability, following the rules below:

1. We take that linguistic similarity is somewhat more reliable than structural similarity, and that the alignments generated by LMO are always accepted by Falcon-AO.
2. When the linguistic comparability is high and the structural comparability is low, only alignments generated by GMO with high similarity are reliable and accepted by Falcon-AO.
3. If the linguistic comparability is low, all of the alignments generated by GMO are accepted by Falcon-AO. In this case, there is no enough information to measure these alignments and we can only assume that they are reliable.

Falcon-AO is implemented in Java. The implemented process can be outlined as follows:

1. Input two ontologies and parse them.
2. Run LMO and obtain matched entity pairs.
3. Calculate linguistic comparability and structural comparability.
4. In the case that linguistic comparability is below a very low threshold (e.g. 0.01) and the structural comparability of them is also low, we take that these ontologies are quite different and Falcon-AO exits with no alignment.
5. Set external entities of the ontologies according to the matched entity pairs generated by LMO.
6. Input matched entity pairs generated by LMO into GMO and form external mapping for GMO. In the current version of Falcon-AO, all the individuals of ontologies are specified as external entities and their similarities are computed by LMO.
7. Run GMO and obtain matched entity pairs.
8. Integrate the alignments generated by LMO and GMO following the rules described above.
9. Exit with alignments as output.

[2] http://jena.sourceforge.net/

1.5 Adaptations Made for the Contest

For anatomy test, FMA [3] ontology and OpenGALEN [4] ontology are not OWL DL. In order to make effective use of descriptions of entities, we have manually found some annotation properties and inputted them into LMO. With the help of these annotation properties, Falcon-AO can find about 500 more matched entity pairs in addition to other 2000 matched entity pairs found by a simple version of LMO.

2. RESULTS

In this section we will present the results of alignment experiments on OAEI 2005 campaign. All the alignments output by Falcon-AO are based on the same parameters.

2.1 Systematic Benchmark Test

We divide all the benchmark tests [5] into five groups: test 101-104, test 201-210, test 221-247, test 248-266 and test 301-304. We will report the results of alignment experiments on these five groups in correspondence. The full results on all tests are listed in **section** 6.3.

2.1.1 Test 101–104

In tests 101, 103 and 104, the source ontologies contain classes and properties with exactly the same names as those in the reference ontologies. LMO can easily get all the matched entity pairs, and GMO takes little effect.

In test 102, the linguistic comparability of the two ontologies is nearly zero and the structural comparability is low as well. So it could be concluded that the two ontologies to be aligned are quite different. Falcon-AO exits with no alignment.

The average performance on test 101-104 is shown below:

	Precision	Recall	F-Measure	Time
Average	1.0	1.0	1.0	5s

2.1.2 Test 201–210

We find that each pair of ontologies of these ten tests has high structural comparability, which means that each pair of the ontologies to be aligned is quite similar in structure. Our previous experiments showed that GMO performed well on these tests even without any additional external mapping input. In most tests, LMO just finds a small part of all the matched entity pairs, the rest are generated by GMO. Since GMO runs slower than LMO, it takes Falcon-AO more time to find all matched entity pairs.

[3]http://sig.biostr.washington.edu/projects/fm/
[4]http://www.opengalen.org/
[5]http://oaei.inrialpes.fr/2005/benchmarks/

For test 201, where each of the local name of class and property is replaced by a random one, LMO can still find some matched classes and properties due to the sameness of their labels or comments. With these matched entity pairs as feed, GMO performs well.

In test 202, names of classes and properties are disturbed and their comments are suppressed. LMO can only find little mapping. Meanwhile, Falcon-AO still performs not bad by running GMO. In this test, we find that it is hard to distinguish many properties purely by the structure of the ontology, since they have the same domains and ranges, and never used in other part of the ontologies. Falcon-AO failed to find correct mapping of these properties, which makes the result not so well as test 201.

In test 203, LMO is able to find all the matched entity pairs. Therefore, it just takes Falcon-AO several seconds to find all alignments.

For tests 204 and 208 with naming conventions, both the linguistic comparability and structural comparability are high. The outputs of the integration of LMO and GMO are well.

For the synonym tests 205 and 209, due to the fact that no thesaurus is used in our tool, LMO performs not so well. There are some errors in the outputs of LMO. With these errors feed to GMO, GMO failed to perform well. As a result, the outputs of Falcon-AO may be weaker than the outputs of using GMO independently.

In tests 206, 207 and 210, ontologies to be aligned are expressed in different languages. Falcon-AO does not have a specific matcher that uses a dictionary for word translation. However, because of their high structural comparability, GMO in Falcon-AO performs not bad on these tests.

The average performance on test 201-210 is described below:

	Precision	Recall	F-Measure	Time
Average	0.96	0.95	0.95	63s

2.1.3 Test 221–247

In these tests, the linguistic comparability of each pair of ontologies to be aligned is very high. Most of the alignments are found by LMO and GMO takes little effect. So, it only takes Falcon-AO a few seconds to align them.

As is shown below, the average performance on these tests are perfect.

	Precision	Recall	F-Measure	Time
Average	0.99	1.0	0.99	4s

2.1.4 Test 248–266

These fifteen tests are the most difficult ones in all benchmark tests, since both their linguistic comparability and structural comparability are low. In the case that the linguistic comparability between two given ontologies is very low, Falcon-AO would not call any matchers. However, in these tests, there are still some individuals with the same names, which increase the linguistic comparability. So Falcon-AO will still run GMO integrated with LMO.

Since the ontology pairs to be aligned are quite different both in linguistics and in structure, our outputs are not good (with average F-Measure 0.63). Indeed, in some cases, it is really hard to determine the exact mapping. For these tests, the time for aligning relies on the size of two ontologies.

	Precision	Recall	F-Measure	Time
Average	0.71	0.60	0.63	60s

2.1.5 Real Ontologies Test 301–304

In these tests, each pair of ontologies has high linguistic comparability but low structural comparability. This indicates that the outputs of Falcon-AO mainly come from the outputs of LMO. Alignments with high similarity generated by GMO matcher are also reliable and these matched entity pairs should also be output by Falcon-AO. The average performance on these four tests is presented below:

	Precision	Recall	F-Measure	Time
Average	0.93	0.81	0.86	20s

2.2 Blind Tests

Blind tests consist of two groups: directory test [6] and anatomy test [7], and they are all real world cases.

Directory

We have got the alignment results on directory test by using the same set of parameters as the ones for benchmark test.

Anatomy

Falcon-AO detects that the FMA ontology and OpenGALEN ontology in anatomy test are so large that our GMO could not process them. Therefore, our alignment result of anatomy test comes only from a simple version of LMO.

[6]http://oaei.inrialpes.fr/2005/directory/
[7]http://oaei.inrialpes.fr/2005/anatomy/

3. GENERAL COMMENTS

In this section, we will summarize some features of Falcon-AO and the improvement in our future work, some comments about test cases will also be presented.

3.1 Comments on the Results

Our Falcon-AO performs well on benchmark tests 101-104, 201-210 and 221-247, and the results of test 301-304 are moderate, but on test 248-266, Falcon-AO doesn't perform so well. According to the results on these test cases, we can see the strengths and weaknesses of Falcon-AO:

Strengths

According to the experimental results, Falcon-AO performs well when the structures of the ontologies to be aligned are similar to each other or there is much lexical similarity between the two ontologies. Particularly, Falcon-AO performs well when the two ontologies have very little lexical similarity but high structural comparability.

Weaknesses

When there is little common vocabulary between the ontologies and in the meanwhile the structures of the ontologies are quite different, Falcon-AO can hardly find the exact mapping. Furthermore, GMO could not process very large ontologies, which means that while aligning very large ontologies, Falcon-AO cannot use their structural information.

3.2 Improvement of Falcon-AO

From the experiments we have learnt some lessons and plan to make improvements in the later versions. The following three improvements should be taken into account.

1. While expressing the same thing, people may use synonyms and even different languages. Therefore, it is necessary to use lexicons to match ontologies.
2. The current version of Falcon-AO did not support many-to-many mapping. The functionality of finding many-to-many mapping will be included in the later version of Falcon-AO.
3. Currently, the measure of linguistic comparability and structural comparability of ontologies are still simple and an improvement will be considered.

3.3 Comments on the Test Cases

The proposed test cases covered a large portion of discrepancies occurring of ontologies while aligning ontologies. Doing experiments on these test cases is helpful to improving the alignment algorithm and system. However, there are few tests on real world ontologies in benchmark tests.

4. CONCLUSION

While aligning real ontologies, linguistic matching plays an important role in matching process. Therefore, we integrate our GMO with LMO in Falcon-AO. From the experiments, we found that, Falcon-AO performed well on most of benchmark tests. It is also worthy of note that most of benchmark tests came from artificially altered ontologies, and more real world ontologies are expected to be included in benchmark tests.

Acknowledgments

This work is supported in part by National Key Basic Research and Development Program of China under Grant 2003CB317004, and in part by the NSF of Jiangsu Province, China, under Grant BK2003001. We are grateful to other members in our team for their discussion and comments that helped strengthen our work. We also would like to thank OAEI 2005 campaign for providing test cases.

5. REFERENCES

[1] D. Brickley and R. Guha (eds). RDF Vocabulary Description Language 1.0: RDF Schema. *W3C Recommendation, 10 February 2004. Latest version available at http://www.w3.org/TR/rdf-schema/*

[2] P. Hayes (ed.). RDF Semantics. *W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/rdf-mt/*

[3] W. Hu, N. Jian, Y. Qu and Y. Wang. GMO: A Graph Matching for Ontologies. *Submitted to K-Cap workshop on Integrating Ontologies (2005). Now is available at http://xobjects.seu.edu.cn/project/falcon/GMO.pdf*

[4] V. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics - Doklady 10 (1966) 707-710*

[5] P. Patel-Schneider, P. Hayes, I. Horrocks (eds.). OWL Web Ontology Language Semantics and Abstract Syntax. *W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/owl-semantics/*

[6] V. Raghavan and S. Wong. A Critical Analysis of Vector Space Model for Information Retrieval. *Journal of the American Society for Information Science 37(5) (1986) 279-287.*

[7] G. Salton. Automatic Text Processing. *Addison-Wesley Longman Publishing Co., Inc. (1989) 613-620.*

6. RAW RESULTS

Information about our project can be found at http://xobjects.seu.edu.cn/project/falcon/falcon.html, and our tool is available now.

6.1 Link to the System and Parameters File

Falcon-AO can be found at http://xobjects.seu.edu.cn/project/falcon/download.html.

6.2 Link to the Set of Provided Alignments

Results presented in this paper are available at http://xobjects.seu.edu.cn/project/falcon/results/falcon.zip.

6.3 Matrix of Results

Runtime Environment: Tests were run on a PC running Windows XP with an Intel Pentium 4 2.4 GHz processor and 512M memory.

No.	Precision	Recall	Time
101	1.0	1.0	4s
102	NaN	NaN	6s
103	1.0	1.0	4s
104	1.0	1.0	4s
201	0.98	0.98	105s
202	0.87	0.87	140s
203	1.0	1.0	4s
204	1.0	1.0	22s
205	0.88	0.87	55s
206	1.0	0.99	51s
207	1.0	0.99	51s
208	1.0	1.0	34s
209	0.86	0.86	102s
210	0.97	0.96	68s
221	1.0	1.0	4s
222	1.0	1.0	4s
223	1.0	1.0	4s
224	1.0	1.0	4s
225	1.0	1.0	4s
228	1.0	1.0	3s
230	0.94	1.0	4s
231	1.0	1.0	4s
232	1.0	1.0	4s
233	1.0	1.0	3s
236	1.0	1.0	3s
237	1.0	1.0	4s
238	0.99	0.99	4s
239	0.97	1.0	3s
240	0.97	1.0	4s
241	1.0	1.0	3s
246	0.97	1.0	3s
247	0.94	0.97	3s
248	0.84	0.82	100s
249	0.86	0.86	114s
250	0.77	0.70	7s
251	0.69	0.69	166s
252	0.67	0.67	119s
253	0.86	0.85	80s
254	1.0	0.27	4s
257	0.70	0.64	4s
258	0.70	0.70	162s
259	0.68	0.68	113s
260	0.52	0.48	7s
261	0.50	0.48	8s
262	0.89	0.24	4s
265	0.48	0.45	7s
266	0.50	0.48	8s
301	0.96	0.80	18s
302	0.97	0.67	3s
303	0.80	0.82	39s
304	0.97	0.96	18s