

AUTOMS: Automated Ontology Mapping through Synthesis of methods

Konstantinos Kotis, Alexandros Valarakos, George Vouros

AI Lab, Information and Communication Systems Eng. Department, University of the Aegean,
Samos, 83 200, Greece

kkot@aegean.gr, alexv@aegean.gr, georgev@aegean.gr

Abstract. AUTOMS is a tool for the automatic alignment of domain ontologies. To ensure high precision and recall with the minimum human involvement, AUTOMS integrates several matching methods. This paper presents the tool and the results obtained for the ontologies within the framework of the OAEI 2006 contest. Particularly, the synthesis of lexical, semantic and structural matching methods, together with the exploitation of concept instances resulted in a rather high recall with space of improvement, and a quite high precision that shows the accuracy of the individual methods, as well as of their synthesis.

1 Presentation of the system

1.1 State, purpose, general statement

In this paper we present the AUTOMS tool for the automatic alignment of ontologies. The proposed tool exploits the HCONE-merge [1] ontology mapping method, which is based on “uncovering” the informal intended meaning of concepts by mapping them to WordNet senses. Furthermore, AUTOMS integrates the HCONE-merge method with an innovative lexical matcher named COCLU (COmpression-based CLUstering) [2], as well as with matching heuristics that exploit structural features of the source ontologies. The synthesis of these methods contributes towards automating the mapping process of concepts and properties of OWL ontologies, by exploiting different features of them: lexical, structural and semantic features.

The WordNet lexicon and concept instances provide additional information towards unveiling mappings in cases where features such as labels and comments are missing or in cases where names are replaced by random strings. Structure matching heuristic rules are exploited to discover mappings in situations where lexical and semantic methods do not have enough information to proceed.

AUTOMS provides mappings between concept/property pairs with high precision. However, it must be stated that it does not achieve a satisfactory recall for the experiments contacted. This suggests that further improvements are necessary both to the individual methods as well as to the sophistication of the synthesis of results.

Since the execution times for obtaining these results using OAEI contest's benchmark ontologies were quite high, a trade-off between lower time and better results was to be made.

Finally, it must be stated that AUTOMS has been improved much due to the experience gained within the OAEI contest.

1.2 Specific techniques used

The methods integrated within AUTOMS run in a particular sequence: Mappings computed by a method are being exploited by subsequent methods so as new mappings to be produced. The following paragraphs present the individual methods in the sequence of their execution.

AUTOMS is mainly based on its lexical matching method, which is applied first in the sequence of the methods employed. COCLU exploits lexical information concerning names, labels and comments of ontologies' concepts and properties, in order to compute their similarity. Although labels are considered the most important, comments and names are also examined. COCLU was originally proposed as a method for discovering typographic similarities between strings, sequences of characters over an alphabet (ASCII or UTF character set), with the aim to reveal the similarity of concepts instances' lexicalizations during ontology population [2]. It is a partition-based clustering algorithm which divides data into clusters and searches the space of possible clusters using a greedy heuristic. Each cluster is represented by a model, rather than by the collection of data assigned to it. The cluster model is realized by a corresponding Huffman tree which is incrementally constructed as the algorithm dynamically generates and updates the clusters by processing one string (instance's surface appearance) at a time. The use of a model classifies the algorithm to the conceptual or model based learning algorithms. To decide whether a new string should be added in a cluster (and therefore, that it lexicalizes the same class/property as the other strings in the cluster do) the algorithm employs a score function that measures the compactness and homogeneity of a cluster. This score function, *Cluster Code Difference (CCDiff)*, is defined as the difference of the summed length of the coded string tokens that are members of the cluster, and the length of the cluster when it is updated with the candidate string. This score function groups together strings that contain the same set of frequent characters according to the model of a cluster (e.g. Pentium III and PIII). A string that lexicalizes an OWL class or property belongs in a particular cluster when its *CCDiff* is below a specific threshold and it is the smallest between the *CCDiff*'s of the given string and all existing clusters. Based on our experience with COCLU, the similarity threshold (ranging in [0,1]) was set to 0.986. A new cluster is created if the candidate string cannot be assigned to any of the existing clusters. As a result, it is possible to use the algorithm even when no initial clusters are available.

Next to the computation of the lexically matching pairs is the computation of the semantic morphism (s-morphism) which is the core technique behind the HCONE-merge method. Given two ontologies, the algorithm computes a morphism between each of these two ontologies and a "hidden intermediate" ontology. This morphism is computed by the Latent Semantic Indexing (LSI) method and associates ontology

concepts with WordNet senses. Latent Semantic Indexing (LSI) [4] is a vector space technique originally proposed for information retrieval and indexing. It assumes that there is an underlying latent semantic space that it estimates by means of statistical techniques using an association matrix ($n \times m$) of term-document data (WordNet senses in our case). It must be emphasized that although LSI exploits structural information of ontologies and WordNet, it ends up with semantic associations between terms. As it is specified in the HCONE-merge approach, WordNet is not considered to include any intermediate ontology, as this would be very restrictive for the specification of the original ontologies (i.e. the method would work only for those ontologies that preserve the inclusion relations among WordNet senses). Actually, it is assumed that the intermediate ontology is “hidden” and the method constructs this ontology while mapping concepts to the WordNet senses. We have used WordNet since it is a well-thought and widely available lexical resource with a large number of entries and semantic relations.

The mappings computed by the lexical and semantic matching methods are then used as input to a simple structural matching algorithm which exploits similarities in the vicinities of concepts/properties. Here, the vicinity of a concept/property includes only the subsumers and subsumees. The matching method has been implemented to improve the performance of the lexical and semantic matching methods by exploiting simple structural features. Consider the matching between two concepts c_1 and c_2 of source ontologies O_1 and O_2 , respectively. The heuristic is: “if at least two neighbor concepts of c_1 have already been (lexically or semantically) mapped to two neighbor concepts of c_2 such that the mappings respect the ontology axioms of inclusion and equivalence, i.e. a sub-concept of c_1 has been mapped to a sub-concept of c_2 , then c_1 and c_2 are considered to structurally match”. The threshold of two (2) neighbor concepts has been considered after conducting several experiments. It should be noticed that further in the alignment process, AUTOMS uses an enhanced structure matching method that runs iteratively using the mappings of all the methods. This method expands the vicinity of concepts to include object properties as well.

The fourth method in sequence utilizes concept instances (individuals). Particularly, for the concepts that have not been determined to be similar to any other concept, AUTOMS compares their individuals, if any. For those concept pairs that have at least one matching instance, AUTOMS discovers a possible mapping. The matching of concept instances is currently based on the similarity of their local names, which is their Uniform Resource Identifier (URI).

The fifth method utilizes information about properties. For the concepts that have not been determined to be similar to any other concept, AUTOMS compares their properties, if any. For those concept pairs that have at least two matching properties, AUTOMS identifies a possible mapping. The matching of object properties is based on the similarity of their property names, as well as on the similarities of their domain and range.

The final step in the alignment process is the execution of an enhanced iterative structure matching method. This method uses the proposed matching pairs from all the previous methods in order to compute mappings based on concepts’ enhanced vicinity: The enhanced vicinity of the concept includes all the concepts related to it. This method runs iteratively for 2 times, updating the list of proposed matching pairs with the pairs discovered in each iteration. It has been observed during the specific

benchmark experiments that although new mappings are discovered in each iteration, there is no change to the set of mappings after the second execution. Further experimentation and investigation is needed for improving this method.

1.3 Adaptations made for the evaluation

AUTOMS is an evolving tool that integrates new methods, which are being tested in new cases. The OAEI contest provides challenging cases that require the exploitation of special features for AUTOMS (and any other tool) to perform efficiently and effectively.

Specific adaptations have been made with respect to the utilization of ‘comments’, to the existence of the ‘lang’ property, and to the use of ‘random strings’ for concept/property names.

Implementation adjustments have also been made in order to be able to run large sets of ontologies in short time. Furthermore, AUTOMS has been modified in order to be able to produce alignments in the form that OAEI contest requires. The evaluation of the results however has been performed using organizers’ Alignment API (version 2.4)

It must be noticed that for the purposes of the contest the alignment output files for ontologies 302 and 303 need to be manually fixed: The <onto2> entry (the ontology file location) value must be replaced with <http://oaei.ontologymatching.org/2006/benchmarks/302/onto.rdf> and <http://oaei.ontologymatching.org/2006/benchmarks/303/onto.rdf>, respectively.

1.4 Link to the system and parameters file

http://www.icsd.aegean.gr/incosys_old/Projects/AUTOMS/OAEI/system/automs.zip

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

http://www.icsd.aegean.gr/incosys_old/Projects/AUTOMS/OAEI/results/automs.zip

2 Results

Results produced with AUTOMS for the 2006 OAEI contest are grouped and discussed below. These results were produced with a stand-alone Java version of AUTOMS on a standard Windows-based PC (2.4 GHz processor). Resulted alignments are sets of pairs of mappings, i.e. of equivalent (symb. =) concepts/properties.

2.1 Benchmark

2.1.1 Tests 101 to 104

Although these ontologies have no special features or difficulties for aligning them, AUTOMS loses in precision due to its lexical method. This is because COCLU compares first the labels of concepts/properties for their similarity, and finds 3 mappings: “number = numberOrVolume”, “collection = book”, “name = shortName”. As already pointed, although labels are considered the most important for COCLU, comments and names are also compared if labels’ similarity value is less than the specified threshold.

The semantic and structure matching methods return pairs that have already been computed by the lexical matching method, with no problems in precision. Instance and property matching methods do not contribute any mapping.

Language generalization and restriction features (103, and 104 ontologies) do not affect the results.

2.1.2 Test 201 to 210

Each case of this group of tests should be presented separately in order to thoroughly discuss the importance of each of the methods that AUTOMS integrates. We will point here to the most important issues and briefly comment each of them.

Ontologies with no names, which is the case where names have been replaced by random strings or synonyms or naming conventions or even foreign names, but with comments in place (ontologies 201, 204, 205, 206 and 207) do not cause serious problems in AUTOMS. In fact, the exploitation of the comments and the utilization of instances, as well as the mappings computed by the semantic and structure methods, result in recall that ranges from 0.66 to 1.00 and in precision that ranges from 0.94 to 0.97.

The alignment of ontologies with the above features and with no comments (ontologies 202, 209 and 210) resulted to low recall ranging from 0.10 to 0.33. The mappings were mainly contributed by the lexical method (ontologies 202, 209, and 210) and the instance matching method (ontology 202), and less by the semantic matching (ontology 209) and the enhanced structure matching method (ontology 209). Although we expected the semantic method with the use of WordNet 2.0 lexicon to unveil more mappings, it identified the pairs “Booklet = Brochure” and “Monograph = Monography”. This can be explained by the nature of most of the concept/property names (i.e. the use of compound terms or naming conventions), and by the amount/quality of information included in the vicinity of each concept/property.

Alignments of ontologies 203 and 208 have been easily produced by AUTOMS: the lack of comments did not affect much the performance. The use of labels and names, even with conventions, resulted to high precision and recall (1.00) for the ontology 203 and to precision 1.00 and recall 0.73 for the ontology 208. The

exploitation of instances and the enhanced structure matching method have revealed the mappings produced for the ontology 208.

2.1.3 Test 221 to 247

Names, labels, and comments in these ontologies have no special features that may distract the alignment: These ontologies resulted from modifications in structure and the addition of instances or/and properties. That is why the recall in all test cases is 1.00. Precision is influenced by some mistakenly returned (false positive) mappings. However it does not fall under 0.87 (247 ontology), with a maximum of 1.00 in some cases (228, 233, 236 and 241 ontologies). The worst case (247 ontology) is due to false positives returned from the lexical matching algorithm and the pair “Conference = Workshop” produced by the instance matching method.

2.1.4 Test 248 to 266

These are the most difficult tests for AUTOMS since names, labels, and comments have been removed or replaced by random strings. The lexical matching method contributed only one mapping, i.e. the “lastName = lastName”. Since only the instance matching method contributed in this set of alignments, the recall measure ranges from 0.10 to 0.31. The structure matching method did not contribute any mapping. The precision however is far more satisfactory, ranging from 0.82 to 1.00.

2.1.5 Test 301 to 304

Apart from ontology 304, for which the structure matching method contributed a significant number of mappings, all the mappings for these tests were computed by the lexical matching method. This fact has a negative impact to the recall measure; however precision was kept again above 0.86. The absence of concept instances, together with the fact that no semantic mappings have been computed, played a major role in the low recall measure.

2.2 Anatomy

We were not able to run this test due to the large size of ontology files.

2.4 Directory

We were able to run tests with the directory ontologies since they were given in OWL and they had a manageable size. For running this test we had to split the given ontology set in smaller sets since we experienced problems with the heap in Java (although we had used the Xmx1000M parameter). Minor adjustments to the code of AUTOMS had to be done since COCLU could not handle concept names of length 1

(e.g. 'A' or 'B') and names with numbers (e.g. '1990s'). The tool computed mappings between concept/property pairs, but since we had no expert mappings to evaluate our results, we can only wait for OAEI 2006 organizers' comments.

Our observations concerning the alignments computed by AUTOMS (randomly browsing some of the 4.639) are limited to the fact that these were mainly based on the lexical matching method, secondly on the semantic matching method (e.g. "Economics = Political_Economy", "Arts = Humanities") and less to the structure matching method (e.g. "Regional = By_region"). Since this test was mainly addressed to discover mappings by exploiting subsumption relations, AUTOMS should have integrated a more elaborated structure matching method. Furthermore, since ontologies have no concept instances and properties, the related methods returned no mappings.

2.5 Food

We were not able to run this test due to the large size of the ontology file, as well as due to the inability of AUTOMS to import ontologies in the SKOL language.

2.6 Conference

For the purpose of this test, we run 3 separate tests, aligning the three larger ontologies (Ekaw, Iasted and OpenConf) to each of the set. Equivalences between concepts/properties were identified. The instances matching method of AUTOMS did not return any mappings, since ontologies were not populated with concept instances.

The three different sets of outputs (total of 30 alignment files in OAEI format) were used to draw the following points:

1. A high number of mappings (14) were identified between Ekaw.owl and Conference.owl. Special features such as inverse compound names of concepts (e.g. PC_Member = Member_PC) have been tackled by the lexical matching method. Mappings such as "Document = Conference_Document" has been identified by the enhanced structure matching method. Pairs such as "Person = Human" and "Document = Article" between the Ekaw.owl and Confious.owl ontologies have been identified due to the semantic and enhanced structure matching methods, respectively. Incorrect mappings are also identified mainly due to the structure matching method (e.g. "Paper = review") when the Ekaw.owl ontology is mapped to itself.
2. Also 14 mappings were identified between iasted.owl and sigkdd.owl ontologies. The "Delegate=Conference_Participant" mappings is identified by the enhanced structure method for the ontologies iasted.owl and Ekaw.owl. Apart from this, the rest of the mappings identified were of no particular difficulty.
3. For the openConf.owl ontology and the rest of the ontologies, the mappings that AUTOMS computed, apart from the "surname = last_name" in Confious.owl, had no specific difficulty.

4. Although several alignments have been identified between the ontologies, it seems that OAEI participants will need to spend some time prior reaching a consensus on these alignments. More important, to be able to produce an alignment between all these ontologies and finally get the reference ontology (as a result of merging them) certainly needs more time and effort. It seems that although the domain of conferencing is a generally agreed and well understood context, the different types of conferences' organizers, participants, subjects, and reviewing systems drive quite divergent ontology specifications.

Since this test is a blind test, we expect organizers' feedback.

3 General comments

The participation in the OAEI contest has been rather valuable for improving our tool. Several minor code adjustments/improvements and other methodological amendments were made in order to be able to deliver the presented precision and recall in the OAEI 2006 tests. The experience gained from the contest is keeping AUTOMS in a continuous process of improvement. In future versions of AUTOMS several problematic cases that have been discussed in this paper will be addressed.

3.1 Comments on the results

As already mentioned and also implied by the results described in this paper, AUTOMS is mainly based on the lexical matching method. The synthesized methods' performance is rather satisfactory for ontologies that use labels and comments to the specification of concepts/properties. The approach works well with naming conventions and language variations (tested for English and French). The weakness to work with concept/property names that start with numbers ('1990s') or have length of one letter ('A') has been identified and tackled. AUTOMS results are significantly based on the exploitation of concept instances (in cases where this applies). The use of structural and semantic information did not contribute as much as it was expected in the final results. However the synthesis of all these matching methods has improved the overall recall and the precision of the results.

We have collected a large set of results that AUTOMS produced for the several OAEI experiments, using variations of the individual methods and of their synthesis. AUTOMS in its simple initial version returned for the benchmark ontologies an H-mean of 0.93 and 0.63 of precision and recall, respectively. During the enrichment of AUTOMS with new and/or more advanced methods, we managed to reach the H-mean of 0.97 for precision and 0.64 of recall. The final submitted results of 0.94 (precision) and 0.67 (recall) show a good tradeoff between precision and recall that our tool can deliver for the particular benchmark ontologies.

Running AUTOMS with last year's benchmark ontologies and the Alignment API 2.4, one can observe that our tool performs better from the rest of the tools as far as

the precision (0.94) is concerned, while the recall is not much lower from this of others' (0.71).

3.2 Discussions on the way to improve the proposed system

In future versions of our tool we should be experimenting with more advanced structure matching and with advanced semantic matching methods that already presented in other lines of related research [5].

Things need to be done also with the ability of AUTOMS to read and align large ontologies.

3.4 Comments on the OAEI 2006 test cases

This year OAEI contest has provided participants with harder tests and with more real ontologies. This is a good improvement on the test cases themselves. However, we think that more improvements are needed so as to avoid (at least in some of the cases) simplistic methods to dominate to more sophisticated ones: For instance, the heuristic of having only a common instance for a pair of concepts to match it seems too simplistic for contributing to the increase of methods' precision.

Furthermore, it would be more helpful to provide participants also with the reference alignments of the directory and conference tests so as to be possible for them to examine their tools, provide helpful insights to the research community and draw conclusions on real case ontologies.

3.5 Comments on the OAEI 2006 measures

Since there will always be a tradeoff between precision and recall, h-mean and/or ROC curves is a good alternative for giving participants a better view of their tool's performance. H-mean and/or ROC curves should be produced at least for each group separately since groups of tests have quite different design motivation. Also, it would be interesting to agree on the importance of precision against recall (or in the reverse order) for particular types of alignments depending on the context that they are performed. For instance, we can argue that there exist some alignments between real ontologies that are critical to being correct rather than being complete.

4 Conclusion

Our participation in the OAEI 2006 contest with the AUTOMS tool has been a significant experience. We have actually been able to identify cons and prons of our tool, and improve some points in its implementation. The organizers' feedback and the comparison with the other tools will also contribute to future improvements of the tool and of the approach in general

References

1. Kotis K., Vouros G., Stergiou K.: Towards Automatic Merging of Domain Ontologies: The HCONE-merge approach. Elsevier's Journal of Web Semantics (JWS), vol. 4:1, (2006) pp. 60-79
2. Valarakos A., Paliouras A., Karkaletsis G., Vouros G.: A name-Matching Algorithm for supporting Ontology Enrichment. In Proceedings of SETN'04, 3rd Hellenic Conference on Artificial Intelligence, Samos, Greece (2004)
3. Miller G., Beckwith R., Fellbaum D., et al.: Five papers on WordNet. CSL Report 43, Cognitive Science Laboratory, Princeton University (1995)
4. Deerwester S., Dumais S. T., Furnas G. W., Landauer T. K., Harshman R.: Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science (1990)
5. Vouros A. and Kotis K.: Extending HCONE-merge by approximating the intended interpretations of concepts iteratively. 2nd European Semantic Web Conference, Heraklion, Crete May 29 – June 1, 2005. Proceedings, Series: Lecture Notes in Computer Science, vol. 3532, Asunción Gómez-Pérez, Jérôme Euzenat (Eds.), Springer-Verlag

Appendix: Raw results

Matrix of results

#	Name	Prec.	Rec.	Time (sec)
101	Reference alignment	0.94	1.00	93
102	Irrelevant ontology	-	-	-
103	Language generalization	0.94	1.00	87
104	Language restriction	0.94	1.00	86
201	No names	0.94	0.95	86
202	No names, no comments	1.00	0.10	79
203	No comments	1.00	1.00	88
204	Naming conventions	0.94	1.00	89
205	Synonyms	0.94	0.99	89
206	Translation	0.97	0.66	84
207		0.97	0.66	84
208		1.00	0.73	82
209		1.00	0.33	81
210		1.00	0.28	78
221	No specialisation	0.94	1.00	71
222	Flatenned hierachy	0.93	1.00	81
223	Expanded hierarchy	0.89	1.00	180
224	No instance	0.94	1.00	92
225	No restrictions	0.94	1.00	64
228	No properties	1.00	1.00	44
230	Flatenned classes	0.89	1.00	80
231	Expanded classes	0.94	1.00	88
232		0.94	1.00	71
233		1.00	1.00	46
236		1.00	1.00	44
237		0.93	1.00	85
238		0.89	1.00	181
239		0.97	1.00	39
240		0.87	1.00	83
241		1.00	1.00	44
246		0.97	1.00	40
247		0.87	1.00	83
248		1.00	0.10	63
249		1.00	0.10	78
250		1.00	0.27	43
251		1.00	0.11	72
252		0.91	0.10	167
253		1.00	0.10	62
254		1.00	0.27	42
257		1.00	0.27	42
258		1.00	0.11	71
259		0.91	0.10	167
260		0.90	0.31	37
261		0.82	0.27	80
262		1.00	0.27	41
265		0.90	0.31	38
266		0.82	0.27	79
301	BibTeX/MIT	0.93	0.46	35
302	BibTeX/UMBC	1.00	0.58	24
303	Karlsruhe	0.93	0.78	141
304	INRIA	0.86	0.92	81
	H-mean	0.94	0.67	