COMA++: Results for the Ontology Alignment Contest OAEI 2006

Sabine Massmann[1], Daniel Engmann, Erhard Rahm

University of Leipzig

[1]massmann@informatik.uni-leipzig.de

Abstract.
This paper summarizes the OAEI Contest 2006 results for the matching tool COMA++. The study shows that a generic schema matching system can also effectively solve complex ontology matching tasks.

1 Presentation of the system

COMA++ is an extension of our previous COMA prototype [1]. It is a customizable and generic tool for matching both schemas and ontologies specified in languages such as SQL, XML Schema or OWL [2]. COMA++ offers a GUI and supports the combined use of several match algorithms as well as the reuse of previously confirmed match results [6].
The COMA++ architecture is shown in figure 1. The *Repository* persistently stores all match-related data, the *Model* and *Mapping Pools* manage all schemas, ontologies, and mappings in memory, and the *Matching Engine* performs the match operations. The GUI provides access to these components and is used to visualize models, manage the match process and mappings. The Matching Engine contains different libraries that supports many *match algorithms* and *match strategies*. The similarity results of individual matchers are maintained and aggregated within a similarity matrix per match task [1]. Match strategies implement workflows to deal with complex match tasks and enable a reuse of previous results and the decomposition of larger match tasks into smaller ones [3].

1.1 State, purpose, general statement

COMA and COMA++ have proven to be very effective for matching database and XML schemas [1, 4, 6]. The main reason for this test was to see the effectiveness of a generic matching tool for dealing with ontologies.

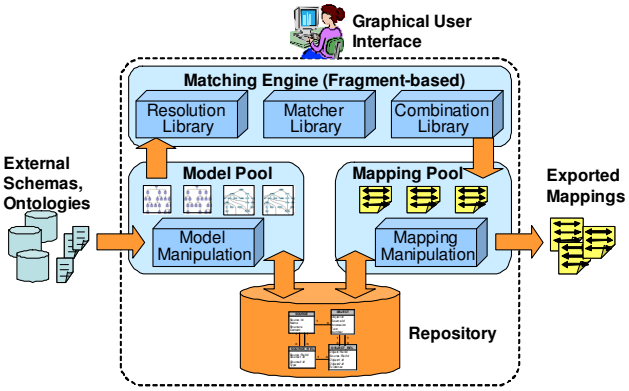

Figure 1.Architecture of COMA++

1.2 Specific techniques used

An automatic match process in COMA++ consists of several steps. In the first step the imported schemas and ontologies are transformed into a generic graph representation. The graph nodes represent schema/ontology components such as classes or properties and have attributes like name and data type. All relationships, e.g. aggregations and specializations, are uniformly represented by edges between nodes. In the next step graph nodes are matched with each other using a match strategy and matchers. There is no differentiation made between node types, so that for example classes and properties can be matched. The similarity values obtained by the individual matchers are aggregated according to a combination strategy (average, etc.). The match candidates are selected from the aggregated correspondences, e.g. based on a threshold criterion. Finally, the result mapping (RDF alignment) is generated.

In addition to the schema-based matchers we used an instance-level matcher which has recently been added to the COMA++ match library.

1.3 Adaptations made for the evaluation

In addition to the integration of an instance matcher only few changes to COMA++ were necessary to deal with specifics of the contest. As mentioned, the output mapping was translated into the predefined RDF alignment format. Furthermore the result of a matcher was ignored if it contained the same similarity value for all entities. This was a minor adaptation made because the same strategy had to be used for all tests.
Another change was the splitting of huge ontologies into several smaller ones. The results of the smaller match tasks were then merged. Another selection step was applied on the merged results to obtain the final result mapping.

To fit the rules of the contest the prototype is not using synonyms and abbreviations which can be given to the system. The specific creation of them was not allowed but would have been necessary because of the different domains.

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

At the following URL .zip archives of all the contest results are available. Furthermore the system with a parameters file can be downloaded.

http://dbs.uni-leipzig.de/Research/coma_oaei.html

2 Results

The results discussed here have been calculated with five matchers: NameType, Comment, Parents, Children and Instance. For the combination of the match results the average value has been computed and a selection has been made using, e.g. a threshold. The best setting has been determined by running different configurations on the benchmark and choosing the one with the highest f-measure. The exact parameters can be found in the appendix.

2.1 Benchmark

This test is a systematic benchmark test containing 50 tests which can be used for identifying the strengths and weaknesses of an algorithm.

The overall score of COMA++ for this task (except 102) is quite good:

	Precision	Recall	F-Measure	Time
Average	0.96	0.82	0.88	7.0 sec

2.1.1 Tests 101-104

The results for tests 101, 103 and 104 are perfect because the classes and properties have the same names, comments and instances. The language restriction and generalization have no influence.
The alignment for the irrelevant ontology 102 contains a few false matches that have similar names, e.g. “year – yearValue”. There are no matches expected for this test, thus precision and recall automatically are 0.0, so we left this value out at the average calculation.

	Precision	Recall	F-Measure	Time
Average	1.00	1.00	1.00	15.4 sec

2.1.2 Tests 201-247

The results of these tests differ depending on the given information because the chosen strategy uses names, data types, comments, structure and instance. If one or more of these information is missing only the remaining information can be used.

For the tasks 202, 209 and 210 the names and the comments differ so these information can't be used and the results have a lower recall.

For all other tests of this group the names, the comments or both contain useful information so the results are quite good.

The tests 221-247 even have the same names and comments, whereas the structure is different. Instances are similar but some ontologies don't contain them. The given information is enough to reach very good results.

	Precision	Recall	F-Measure	Time
Average	0.98	0.95	0.97	8.1 sec

2.1.3 Tests 248-266

In these tests the names have been substituted with random strings and there are no comments. The algorithm can thus only use the hierarchy and the instances, if given. Not for every class and property instances exist, so that information just helps to find corresponding entities. The results for these tests are therefore satisfactory.

	Precision	Recall	F-Measure	Time
Average	0.89	0.51	0.65	4.2 sec

2.1.4 Tests 301-304 (Real Ontologies)

The real-world ontologies have been a more difficult task for COMA++ because the ontologies are quite different compared with the 101 ontology. Three out of the four ontologies don't contain instances – only 304 does. 302 and 303 don't use comments, the structure is quite different and the names are often dissimilar, which the prototype could not find because the contest disallowed us to use auxiliary information.

	Precision	Recall	F-Measure	Time
Average	0.84	0.69	0.76	3.6 sec

2.2 Anatomy

For the anatomy task two large ontologies had to be aligned. Because of the huge size the matching task had to be splitted by our system into smaller ones. The part results were merged and then a variety has been selected. The selection was necessary because with the splitted matching more false matches have been found.

Another difficulty has been the fact that in the FMA ontology the id of classes look like “frame_92794” and “frame_51746” and the real information is in the label. Whereas the OpenGALEN ontology has meaningful ids and uses rarely labels. These labels or ids are made up of a lot of tokens and sometimes they differ only in a few letters, e.g. “fifth” instead of “first”. Therefore we expect that more false positives will be found than in the benchmark test.

2.3 Directory

For this test we matched 4640 pairs of ontologies.
To find out more about the quality of our strategy and that kind of test we also matched the 2265 ontology pairs of the contest 2005. We reached a recall around 0.32 what is as good as the best participants. Looking at the missing correspondences we couldn’t find any similarity of the names, e.g., “7/source.owl#Academic_Departments” and ”7/target.owl#United_Kingdom” and no comments or instances existed. That’s why we couldn’t figure out a way to improve our system.

2.4 Food

The food ontologies uses the different format SKOS. We transformed the given SKOS files into OWL format to be able to match them. These ontologies are quite large so the match process has to be splitted as well as in the anatomy test.

2.5 Conference

This task contains 10 ontologies that deal with conference organisation. The calculation of alignments between each of them was no problem because of the smaller size.

3 General comments

3.1 Comments on the results

Given that COMA and COMA++ were not specifically designed for matching ontologies and we invested only a small amount of time for the contest the overall results are surprisingly good. The new instance matcher proved to be effective especially for the tests where useful information was only provided by instance values.

The used parameters were selected for the whole set of tests. For individual match tasks better results than reported can be obtained by using tailored configuration parameters. Another point is that domain-specific abbreviations, synonyms and previous match results could not be utilized in order to conform with the contest rules.

3.2 Discussions on the way to improve the proposed system

The use of auxiliary information that is conforming to the rules, e.g. WordNet or UMLS, could improve the recall results. The addition of ontology-oriented matchers and the distinction between node and relationship types could also be helpful.

3.3 Comments on the OAEI 2006 procedure

This is our first participation in this Ontology Alignment Contest. Since we are not involved in the contest preparation we had no prior knowledge of most tasks and the regulations. We thus had comparatively little time (about 2 months) to deal with the details of six test series and technical problems caused by unknown formats and large files. Furthermore, we had to adapt the system to the contest rules and try to find the best strategy and configuration.

4 Conclusion

The presented contest results show that COMA++ is not only effective for schema matching but also for ontology matching. This underlines the viability of generic approaches for complex metadata management problems.

References

1. Do, H.H., E. Rahm: COMA - A System for Flexible Combination of Schema Matching Approach. Proc. Intl. Conf. Very Large Databases (VLDB), 2002
2. Aumüller, D., H.H. Do, S. Massmann, E. Rahm: Schema and Ontology Matching with COMA++ (Software Demonstration). Proc. 24. ACM SIGMOD Intl. Conf. Management of Data, 2005
3. Rahm, E., H.H. Do, S. Massmann: Matching Large XML Schemas. ACM SIGMOD Record 33(4), 2004
4. Do, H.H., S. Melnik, S., E. Rahm: Comparison of Schema Matching Evaluations. Proc. 2. Intl. Workshop Web and Databases, LNCS 2593, Springer Verlag, 2003
5. Rahm, E., P.A. Bernstein: A Survey of Approaches to Automatic Schema Matching. VLDB Journal, 10(4), 2001
6. Hong-Hai Do. Schema Matching and Mapping-based Data Integration, Dissertation, Department of Computer Science, Universität Leipzig, Germany, 2006

Appendix: Raw results

The following benchmark results have been computed with the following parameters:

- Strategie: NoContext
- Matcher: NameType, Comment, Instance, Parents, Children
- Combination: Average
- Selection: N=0, Delta=0.0001, Threshold=0.13; Direction=Both

The tests were run on a PC running Windows XP with an Intel Pentium 4 2.4 GHz processor and 512 MB memory.

Matrix of results

#	Name	Prec.	Rec.	Time (sec)
101	Reference alignment	1.00	1.00	15.9
102	Irrelevat ontology	0.00	0.00	5.8
103	Language generalization	1.00	1.00	16.5
104	Language restriction	1.00	1.00	13.7
201	No names	1.00	1.00	14.0
202	No names, no comments	0.90	0.68	12.1
203	No comments	1.00	1.00	12.5
204	Naming conventions	1.00	1.00	14.5
205	Synonyms	1.00	0.98	13.6
206	Translation	1.00	0.98	14.0
207		1.00	0.98	13.2
208		0.99	0.98	11.8
209		0.96	0.78	12.4
210		0.98	0.85	13.4
221	No specialisation	1.00	1.00	4.4
222	Flatenned hierachy	1.00	1.00	5.1
223	Expanded hierarchy	1.00	1.00	6.5
224	No instance	1.00	1.00	4.2
225	No restrictions	1.00	1.00	12.9
228	No properties	0.94	0.94	3.0
230	Flatenned classes	0.99	1.00	11.8
232		1.00	0.99	7.7
233		0.94	0.94	3.0
236		0.94	0.94	3.5
237		1.00	1.00	4.3
238		0.99	0.99	4.9
239		0.90	0.93	3.3
240		0.79	0.91	3.6
241		0.94	0.94	2.8
246		0.90	0.93	2.7
247		0.77	0.91	4.2

248		0.91	0.52	4.1
249		0.89	0.68	12.6
250		0.93	0.42	3.3
251		0.83	0.57	4.5
252		0.90	0.57	4.9
253		0.91	0.52	3.6
254		1.00	0.27	2.9
257		0.93	0.42	2.6
258		0.84	0.58	4.7
259		0.90	0.57	5.3
260		0.86	0.41	2.5
261		0.92	0.33	3.2
262		1.00	0.27	2.7
265		0.86	0.41	2.6
266		0.92	0.33	2.9
301	BibTeX/MIT	0.97	0.64	3.8
302	BibTeX/UMBC	0.78	0.44	2.4
303	Karlsruhe	0.62	0.65	4.0
304	INRIA	0.96	0.91	4.3