Mix'n'Match: An Alternative Approach for Combining Ontology Matchers (Short Paper)

Simon Steyskal[1,2] and Axel Polleres[1]

[1] Siemens AG, Siemensstrasse 90, 1210 Vienna, Austria
[2] Vienna University of Technology, Favoritenstraße 9, 1040 Vienna, Austria

Abstract. The existence of a standardized ontology alignment format promoted by the Ontology Alignment Evaluation Initiative (OAEI) potentially enables different ontology matchers to be combined and used together. Along these lines, we present a novel architecture for combining ontology matchers based on iterative calls of off-the-shelf matchers that exchange information in the form of reference mappings in this standard alignment format. However, we argue that only a few of the matchers contesting in the past years' OAEI campaigns actually allow the provision of reference alignments to support the matching process. We bypass this lacking functionality by introducing an alternative approach for aligning results of different ontology matchers using simple URI replacement in the aligned ontologies. We experimentally prove that our iterative approach benefits from this emulation of reference alignments.

1 Introduction

Mapping between the internal models of software systems is a crucial task in almost all data and system integration scenarios, keeping computer scientists busy for the last decades with still no "silver bullet" in sight.

In quest for the said "silver bullet" [9] to solve such alignment problems, ontologies and ontology matching seem to be a good starting point particularly for for aligning object-oriented models, since ontologies can represent (and be extracted from) object-oriented models fairly naturally, where the hope is that the increasing number of off-the-shelf ontology matching tools can be used "out of the box" to propose reference alignments that can be tailored in a semi-automatic process. In the past decade, the field of ontology matching has matured with many different ontology matchers having participated in the last years' OAEI [1] campaigns, exposing different strengths and weaknesses. The different tracks provided by the OAEI target different matching problems and since every matching tool has its specific special techniques and features, good performance in a specific sub-track often comes in hand with a bad performance in another one. So, intuitively, combining the results of a heterogeneous set of ontology matchers,

[1] `http://oaei.ontologymatching.org/`

R. Meersman et al. (Eds.): OTM 2013, LNCS 8185, pp. 555–563, 2013.

taking the "best off-the-shelf matcher for the problem at hand" seems to be a promising approach for a "universal matcher" without the need for designing a new matcher from scratch.

Unfortunately, (i) neither deciding which matcher is best suitable for a set of given ontologies nor (ii) using ontology matchers in a semi-automatic human-guided matching process is trivial: for one, selecting the most suitable matcher requires some preparatory work like interviews or tests [18] or background knowledge in terms of training sets for learning a classifier [7,8].

On the other hand, for an interactive matching process, one needs to be able to guide a matcher by providing (or confirming) reference alignments, either provided by a domain expert directly or iteratively added by confirming matching results from an ontology matcher to be used as reference for supporting another call of the same or another ontology matcher.

In the present paper, we demonstrate the feasibility of a system that can support such a process in one go; that is, we present a combined ontology matcher which performs better on average than a single matcher on a number of heterogeneous ontology matching problems (from the OAEI campaign), without the need to choose a single matcher, but by iteratively combining results of different matchers, and feeding them as support into subsequent runs of those matchers.

Structure of the Paper. We start our paper with section 2 presenting a framework, which defines an architecture for a combined iterative matcher, that allows to plug existing matchers flexibly together and iteratively combines their results. Preliminary evaluation results presented in Section 3 are very encouraging showing that our approach performs very stable, outperform each single matcher in some cases, or scoring comparably well to the best single matchers in other cases. We discuss ideas for future improvements in Section 4 before we conclude in Section 5.

2 The Mix'n'Match Framework

In this section, we present our combined approach to ontology matching, which we call "Mix'n'Match". Our goal is an approach that combines the above existing matchers with all their strengths and weaknesses in a unified manner. Instead of finding the one "best off-the-shelf matcher for the problem at hand" we aim at combining matcher results of different matchers. Intuitively, the idea of Mix'n'Match is very simple: starting from an empty set of alignments, we aim at iteratively supporting in each round, matchers with the combined results of other matchers found in previous rounds, somehow aggregating the results of a heterogeneous set of ontology matchers.

Two main obstacles need to be overcome to make this approach feasible.

Result Aggregation: In each round we need to decide which alignments to keep and which ones to ignore from the union of new alignments found by all considered matchers.

Reference Alignments: as mentioned above only one of the considered ontology matchers supports reference alignments as inputs, so we need to find another way to "inject" alignments in a non-obtrusive way[2] into existing matchers.

As for result aggregation, for the moment, we have chosen a straightforward strategy here, based on a simple majority vote, that is, all alignments confirmed by a majority of matchers above a certain threshold prevail. Adding support for reference alignments (without touching the matchers' source code) was not that easy. Our first idea was to add reference alignments explicitly in the form of OWL equivalences, using properties `owl:equivalentClass` (for matching concepts), `owl:equivalentProperty` (for matching properties), or `owl:sameAs` (for matching individuals). However, this approach to "emulating" reference alignments proved unsatisfactory, since we would be referring in $O1$ to entities from $O2$ and vice versa. First, such referring would need additional definitions (`owl:Class`, `owl:DatatypeProperty`, or `owl:ObjectProperty` of the referred entities in the respective other ontology of all aligned entities, since otherwise the such enriched ontology would no longer satisfy the syntactic restrictions of OWL DL: obviously, this is inconvenient, because it would lead to significant overhead essentially redefining already existing entities. Somewhat surprisingly, an alternative, much simpler approach to emulating reference alignments proved to be much more effective than adding OWL equivalence axioms: instead of "axiomatizing" alignments using OWL, we just created for each reference alignment a combined new URI for the matched entities, which was then replaced within both $O1$ and $O2$. Let us illustrate this replacement in a short example.

Example 1. Based on found alignments we modify both ontologies by replacing the URIs of the two classes *Teacher* and *Professor* by a unified one *<http://example.org/MixMatch/Teacher___Professor>*, cf. Listings 1+2.

Listing 1. Ontology $O1$ after enrichment

```
@prefix mm: <ex.org/MMatch
    />.
...
mm:Teacher_Professor a owl:
    Class.

ont1:name a owl:
    DatatypeProperty;
    rdfs:domain mm:
        Teacher_Professor;
        rdfs:range xsd:string
...
```

Listing 2. Ontology $O2$ after enrichment

```
@prefix mm: <ex.org/MMatch
    />.
...
mm:Teacher_Professor a owl:
    Class.

ont2:nameIs a owl:
    DatatypeProperty;
    rdfs:domain mm:
        Teacher_Professor;
        rdfs:range xsd:string
...
```

A schematic overview of the overall Mix'n'Match framework is shown in Figure 1; we briefly explain each step in the following.

[2] In order to keep our framework general and extensible, we do not want to modify the code of the ontology matchers or interfere in some other tool-specific way with the matchers used in our framework.

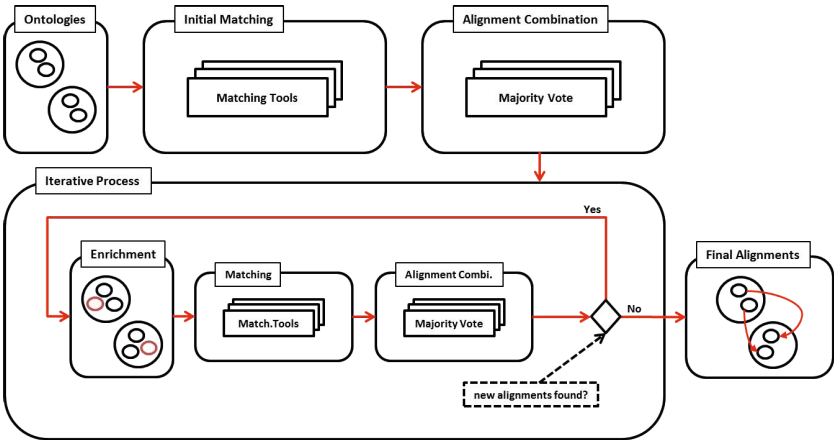

Fig. 1. Framework of Mix'n'Match

Initial Matching: In the initial matching step all participating ontology matchers try to find alignments of the two unaltered base ontologies.

Alignment Combination: The combination of the alignments, especially the choice of those which are used for the enrichment step is – as mentioned above – based on majority votes. By only accepting alignments which were found by a large number of heterogeneous matching tools we aim to ensure a high precision of the found alignments and therefore emulate reference alignments provided by a human domain expert. Although Mix'n'Match would support the definition of an alignment confidence threshold as additional parameter (i.e, only allowing alignments over a specific threshold to pass) we set this threshold per default to 0 in our experiments: since the calculation of confidence values is not standardised across matchers and some matchers only produce boolean confidence values. Other result aggregation methods may be conceivable here and will need more detailed investigations in future versions of Mix'n'Match.

Enrichment: After mixing of the alignments, enrichment of the ontologies takes place, implementing the simple URI replacement sketched above: for every pair of matched entities in the set of aggregated alignments, a merged entity URI is created and will replace every occurrence of the matched entities in both ontologies (cf. Example 1). This approach is motivated by the assumption that if two entities were stated as equal by the majority of ontology matchers, their URI can be replaced by an unified URI, stating them as equal in the sense of URIs as global identifiers. Note that, despite the above-mentioned fact that most matchers seem to ignore URIs as unique identifiers of entities, our experiments showed that URI replacement was effective in boosting the confidence in such asserted alignments

in almost all considered matchers.[3] Again, other enrichment techniques – as mentioned before, for instance, for matchers supporting OWL, reference alignments could be encoded by enrichment with respective OWL axioms – could be conceivable and would fit into the generic framework of Mix'n'Match. Note also that in the current version of Mix'n'Match only one-to-one and equality alignments were considered.

Iteration: After enrichment (which imitates the behavior of reference alignments) we start a new matching round, i.e., we restart the individual matchers on the enriched ontologies, gathering new reference alignments, followed by another alignment combination step, and so forth, until a fix point is reached, in the sense that the alignment combination step produces no further new alignments.

Implementation: We have implemented Mix'n'Match using Java for the general framework, and integrated the existing matchers either by importing the respective Java sources, if available, or including .jar files (some of which were available through the SEAL[4] project).

3 Evaluation

We evaluated our framework using datasets provided by the ontology alignment evaluation initiative. In the following sections we show, that our approach is either able to outperform current ontology matching tools or enhance the results of the in the Mix'n'Match process participated ones.

As evaluation system we used an Intel Core Duo 3GHz with 4GB RAM and Windows 7 64 Bit as OS.

3.1 Evaluation Datasets

Our initial intention was to compare Mix'n'Matchs results to those retrieved by a machine learning approach of combining ontology matchers. [7] Therefore we based our evaluations on the same datasets, i.e. #301 to #304 of the benchmark track consisting of four real world ontologies which were matched against a base ontology (#101), all describing a bibliographic domain and using a subset of the OAEIs conference track, but in contrast to the evaluation of the machine learning approach we used the updated reference alignment set *ra1* which consists of seven ontologies namely *cmt, conference, ekaw, edas, conference, iasted* and *sigkdd*. In contrast to the benchmark track, all seven ontologies were matched against each other resulting in 21 possible combinations and for which reference alignments were available. Regardless their respective participation on either the benchmark or the conference track, we included *AROMA [5], AnchorFlood [11], Eff2Match [1], FalconAO [14], HotMatch [4]* and *Hertuda [12]* for the

[3] Deliberately leaving out internal details of the respective matchers, we may only conjecture here that other features like string-matching were triggered after reference alignments being "asserted" by URI replacement, which consequently lead to further new alignments being found in subsequent calls.

[4] `http://www.seals-project.eu/`

final Mix'n'Match process. Based on performance issues we excluded *Lily [21]* from our matching process and therefore were able to decrease the computation time of Mix'n'Match about more than 50%, two matching tools where excluded because of compatibility problems (*AUTOMSv2 [16], ServOMap2 [22]*). To prove the robustness of our approach, *LogMap2 [15] and YAM++ [19]* were evaluated for comparison purposes but didn't participate in the Mix'n'Match process.

3.2 Evaluation Results

Regarding the Benchmark track, Mix'n'Match was able to outperform all tested ontology matchers although it needed more than twice as much time to complete than the second slowest matcher *Lily*. Matchers marked with † were not able to retrieve alignments for testcase 303 in our test setting. As stated in Table 1 Mix'n'Match was able to increase the F-Measure value of the best used matcher by 2.72%. Nevertheless LogMap2 and Yam++ retrieved better results, but since our approach is primarily based on using majority vote for choosing alignments, single outstanding ontology matchers don't significantly affect the results if we considering their alignments for Mix'n'Match.

Table 1. Aggregated results of the Benchmark and Conference Tracks

	Benchmark			Conference		
Matcher	Prec./Rec.(%)	F-Meas.(%)	Time(ms)	Prec./Rec.(%)	F-Meas.(%)	Time(ms)
Mix'n'Match	**89,70/76,40**	**82,51**	**229795**	**68,78/57,50**	**62,64**	**1215747**
Eff2Match	86,97/74,26	80,12	35123	34,43/64,59	44,91	90649
FalconAO	87,24/73,42	79,73	10424	56,32/57,46	56,89	10176
YAM++	89,86/70,01	78,70	51057	77,29/68,95	72,88	394041
Anchor-Flood†	68,86/57,46	62,64	1021	45,16/57,73	50,68	2451
AUTOv2†	68,50/48,40	56,73	12751	76,90/45,13	56,88	30654
Aroma†	59,06/50,15	54,24	1723	30,85/45,97	36,92	5667
Lily†	54,57/51,68	53,09	86390	35,72/46,03	40,23	337861
Hertuda	56,95/42,90	48,94	1077	72,62/51,01	59,92	1953
LogMap2	83,62/ 30,41	44,60	7255	78,81/57,91	66,76	22072
HotMatch	61,58/33,94	43,76	2553	69,66/51,60	59,29	8994

Remark. We recognized slight differences between the evaluation values measured by the OAEI and those measured by us. But since our approach relies on the results of other matchers, the relative improvement of the values should stay the same.

4 Discussion

Related Work: The approach of using an iterative process which reruns matching techniques till no further alignment pairs were found is not new per se. Tools like Anchor-Flood [11] or ASMOV [13] are based on such a framework but in contrast to our approach neither one of them use off-the-shelf matchers for retrieving alignments. The benefit of using whole tools instead of specific matching techniques is obvious, on the one hand we can highly increase the flexibility of Mix'n'Match since single off-the-shelf matcher can easily be integrated and segregated from the matching process and on the other hand since matching tools

evolve over time and perform better and better by themselves, Mix'n'Match will also benefit from this evolution based on its framework. Besides of combining ontology matchers during the matching step, some approaches only focus on the combination of alignment sets of already matched ontologies. But these approaches doesn't rerun the matching process with the gathered additional knowledge again, although they received very good results especially by using machine learning techniques for the combination of the alignments [6,7,17]. Furthermore there exist some other approaches than a majority voite for combining the results of different ontology matching techniques which could be fairly easily adopted to work with our approach like in [10] where the authors propose an automated approach for weighting individual ontology matchers based on their importance on different matching tasks or in [2] were an automatic alignment evaluation by using quality measures is described. In [3] the authors provide an approach for automatically configuring the parameters of off-the-shelf matchers and therefore optimize their performance; together with the approach proposed in [20] where unsupervised learning techniques are used to find similarity parameters these approaches could be used to improve the performance and flexibility of Mix'n'Match.

Future Work: Several topics will be part of future investigations like using other combination approaches than majority vote or the distributed execution of the used ontology matchers. As alternative combination approach, an advanced machine learning technique like discussed in [7] could be used. Therefore we could increase the *recall* of the reference alignments generated in each round and also remove the common drawbacks of majority votes like ignoring single good results. Furthermore a self-learning classifier must be developed to ensure the full automation of our approach.

5 Conclusions

In this paper we have presented an alternative approach for combining ontology matchers, using their alignment results for iteratively including the knowledge these alignments provide into the ontologies and then rerun the matching process. We have shown that it is basically possible to emulate the additional knowledge eventual input alignments could provide and furthermore eliminate the common drawbacks like loss of automation, which usually comes in hand with the use of reference alignments as support for matching processes.

Nevertheless we discovered that not all ontology matchers consider entities with the exact same URI as 100% identical, but since string similarity measures seems to be part of nearly every investigated ontology matcher it may helped to exceed some internal confidence plateaus of found alignments and therefore accept them as correct alignments. We also pointed out that the more information for the matching process is available, either in terms of additional axioms in the ontology or by providing reference alignments, the better results can be obtained.

References

1. Wei Khong Chua, W., Kim, J.-J.: Eff2match results for oaei 2010. In: OM 2010. CEUR Workshop Proceedings (2010)
2. Cruz, I., Palandri Antonelli, F., Stroe, C.: Efficient selection of mappings and automatic quality-driven combination of matching methods. In: OM 2009. CEUR Workshop Proceedings (2009)
3. Cruz, I.F., Fabiani, A., Caimi, F., Stroe, C., Palmonari, M.: Automatic configuration selection using ontology matching task profiling. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 179–194. Springer, Heidelberg (2012)
4. Dang, T.T., et al.: Hotmatch results for oeai 2012. In: OM 2012. CEUR Workshop Proceedings (2012)
5. David, J., Guillet, F., Briand, H.: Matching directories and owl ontologies with aroma. In: CIKM 2006, pp. 830–831. ACM, New York (2006)
6. Doan, A., Madhavan, J., Dhamankar, R., Domingos, P., Halevy, A.Y.: Learning to match ontologies on the semantic web. VLDB J. 12(4), 303–319 (2003)
7. Eckert, K., Meilicke, C., Stuckenschmidt, H.: Improving ontology matching using meta-level learning. In: Aroyo, L., et al. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 158–172. Springer, Heidelberg (2009)
8. Ehrig, M., Staab, S., Sure, Y.: Bootstrapping ontology alignment methods with APFEL. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 186–200. Springer, Heidelberg (2005)
9. Fensel, D.: Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce. Springer (2001)
10. Gulić, M., Magdalenić, I., Vrdoljak, B.: Automated weighted aggregation in an ontology matching system. Int'l Journal of Metadata, Semantics and Ontologies 7(1), 55–64 (2012)
11. Seddiqui Hanif, M., Aono, M.: An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size. J. Web Sem. 7(4), 344–356 (2009)
12. Hertling, S.: Hertuda results for oeai 2012. In: OM 2012. CEUR Workshop Proceedings (2012)
13. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology Matching with Semantic Verification. J. Web Sem. 7(3), 235–251 (2009)
14. Jian, N., Hu, W., Cheng, G., Qu, Y.: Falcon-ao: Aligning ontologies with falcon. In: K-Cap 2005 Workshop on Integrating Ontologies, pp. 87–93 (2005)
15. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y.: Logmap 2.0: towards logic-based, scalable and interactive ontology matching. In: 4th Int'l Workshop on Semantic Web Applications and Tools for the Life Sciences, SWAT4LS 2011, pp. 45–46. ACM (2012)
16. Kotis, K., Katasonov, A., Leino, J.: Automsv2 results for oaei 2012. In: OM 2012. CEUR Workshop Proceedings (2012)
17. Maio, P., Bettencourt, N., Silva, N., Rocha, J.: Evaluating a confidence value for ontology alignment. In: Proceedings of the 2nd International Workshop on Ontology Matching (OM 2007), Busan, Korea, November 11. CEUR Workshop Proceedings, vol. 304. CEUR-WS.org (2007)
18. Mochol, M., Jentzsch, A.: Towards a rule-based matcher selection. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 109–119. Springer, Heidelberg (2008)

19. Ngo, D.H., Bellahsene, Z.: YAM++: (not) Yet Another Matcher for Ontology Matching Task. In: Bases de Données Avancées, France, p. 5 (2012)
20. Nikolov, A., d'Aquin, M., Motta, E.: Unsupervised learning of link discovery configuration. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 119–133. Springer, Heidelberg (2012)
21. Wang, P., Xu, B.: Lily: Ontology alignment results for oaei 2009. In: OM 2009. CEUR Workshop Proceedings (2009)
22. Wang, Z., Wang, Y., Zhang, S., Shen, G., Du, T.: Matching large scale ontology effectively. In: Mizoguchi, R., Shi, Z.-Z., Giunchiglia, F. (eds.) ASWC 2006. LNCS, vol. 4185, pp. 99–105. Springer, Heidelberg (2006)