On the Feasibility of Using OWL 2 DL Reasoners for Ontology Matching Problems

Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, and Ian Horrocks

Department of Computer Science, University of Oxford
{ernesto,berg,ian.horrocks}@cs.ox.ac.uk

Abstract. In this paper we discuss the feasibility of using OWL 2 DL reasoners to diagnose the integration of large-scale ontologies via mappings. To this end, we have extended our ontology matching system LOGMAP with complete OWL 2 DL reasoning and diagnosis capabilities. We have evaluated the new system, which we call LOGMAP-FULL, with the largest matching problems of the Ontology Alignment Evaluation Initiative, and we have compared its performance with LOGMAP, which currently relies on a highly-scalable (but incomplete) reasoner.

1 Introduction

The Ontology Alignment Evaluation Initiative(OAEI) is an international campaign for the systematic evaluation of ontology matching systems —software programs capable of finding simple correspondences (called *mappings*) between the vocabularies of a given set of input ontologies [25, 5, 6, 26]. The matching problems in OAEI are organised in several tracks, with each track involving different kinds of test ontologies [5]; for example, the ontologies in the largest tracks of OAEI 2011.5 are the Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT), the Foundational Model of Anatomy (FMA), the National Cancer Institute Thesaurus (NCI), and the Adult Mouse Anatomical Dictionary (MOUSE) —all of which are semantically rich bio-medical ontologies with thousands of classes.

Ontology mappings are commonly represented as OWL subclass or equivalence axioms. Hence, the ontology $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ resulting from the integration of the input ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ via the mappings $\mathcal{M}$ automatically computed by a matching system may entail axioms that do not follow from $\mathcal{O}_1$, $\mathcal{O}_2$, or $\mathcal{M}$ alone. Many such entailments correspond to logical inconsistencies caused by either erroneous mappings in $\mathcal{M}$, or by inherent disagreements between $\mathcal{O}_1$ and $\mathcal{O}_2$. Recent work has shown that even the integration of ontologies via manually curated mappings can lead to thousands of such inconsistencies [11, 12, 10].

Most matching systems do not implement any kind of reasoning technique, and hence are unable to detect and repair such inconsistencies. In recent years, however, there has been a growing interest in the development of "built-in" reasoning and diagnosis algorithms for ontology matching systems. Systems like CODI [22, 9] and LOGMAP [14, 10, 13] implement efficient techniques that substantially reduce the number of inconsistencies derived from $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$. To achieve favourable scalability behaviour, however, the reasoning algorithms in CODI and LOGMAP are *incomplete*, and hence cannot guarantee that the output mappings will not lead to logical inconsistencies.

In this paper we evaluate the feasibility of integrating a fully-fledged OWL 2 DL reasoner in the ontology matching system LOGMAP, and hence of guaranteeing that the set of output mappings will not lead to unsatisfiable classes; we call the new system LOGMAP-FULL. Although current reasoners can easily cope with the aforementioned OAEI ontologies individually, the diagnosis of ontology mappings poses very interesting challenges for the evaluation of OWL 2 DL reasoners.

2 LOGMAP in a nutshell

LOGMAP is a highly scalable ontology matching system with "built-in" reasoning and diagnosis capabilities. The latest version of LOGMAP's algorithm [13] can be roughly divided into the stages briefly described next.

I. Computation of candidate mappings. LOGMAP efficiently computes a set of candidate mappings $\mathcal{M}$ using lexical techniques only. Those candidate mappings $\mathcal{M}_r$ involving classes that are lexically very similar are additionally identified as *reliable*.

II. Reasoning-based diagnosis of reliable mappings. The input ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ and the reliable mappings $\mathcal{M}_r$ are encoded in Horn propositional logic. This encoding is sound (but incomplete) for checking the unsatisfiability of each class with respect to $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}_r$. LOGMAP's propositional reasoner implements the well-known Dowling-Gallier algorithm [4, 7] and extends it to record all *conflicting* mappings that may be involved in each unsatisfiability. LOGMAP then implements a greedy diagnosis algorithm that tries to delete as few such recorded mappings as possible in order to resolve the identified unsatisfiabilities.

III. Pruning non-reliable mappings. LOGMAP efficiently indexes the propositional representations of $\mathcal{O}_1$, $\mathcal{O}_2$ and $\mathcal{M}_r$ using an interval labelling schema [1]. This type of semantic index has shown to significantly reduce the cost of computing taxonomic queries over large class hierarchies [3, 21]. LOGMAP exploits this semantic index to efficiently discard those *conflicting* candidate mappings that, if added to the reliable mappings (after diagnosis), will make some class unsatisfiable.

IV. Final diagnosis. LOGMAP uses the same technique as in Step **II** to perform diagnosis over all the remaining candidate mappings (reliable and non-reliable). The resulting set of mappings $\mathcal{M}_{out}$ is returned as output.

3 Reasoning and diagnosis in LOGMAP-FULL

As already mentioned, LOGMAP implements a sound but *incomplete* reasoning algorithm for checking class unsatisfiability. Consequently, there is no guarantee that all classes in $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}_{out}$ will be satisfiable. Furthermore, LOGMAP might fail to detect conflicting candidate mappings in Steps **II-IV**, which might lead to incorrect choices when discarding mappings.

These limitations stem from the fact that reasoning in LOGMAP is incomplete, and hence they could be addressed by integrating a complete OWL 2 DL reasoner R into LOGMAP. A straightforward possibility is to extend LOGMAP's algorithm with a final

Input: $\mathcal{O}_1$, $\mathcal{O}_2$: input ontologies. $\mathcal{M}$: set of mappings.
Output: $\mathcal{M}$: set of mappings

1: **repeat**
2: $Unsat :=$ unsatisfiable classes w.r.t. $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$
3: $Just := \emptyset$
4: **for each** $C \in Unsat$ **do**
5: $Just := Just \cup \mathsf{SingleJustification}(C, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M})$
6: **end for**
7: $\mathcal{M} := \mathcal{M} \setminus \mathsf{ConflictiveMappings}(Just)$
8: **until** $Unsat \neq \emptyset$
9: **return** $\mathcal{M}$

Table 1. Diagnosis in LOGMAP-FULL.

step in which R is used to check the satisfiability of each class w.r.t. $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}_{out}$. Standard justification-based diagnosis techniques (e.g., [16, 15, 23, 8, 28]) can then exploited to fix the identified inconsistencies.[1] This approach, which guarantees a "clean" output, is adopted by systems such as CONTENTMAP [11] and ALCOMO [19]; however, the detection of conflicting mappings in Steps **II-IV** would still rely on an incomplete reasoner.

In LOGMAP-FULL we have implemented a different approach, where each call to the Dowling and Gallier algorithm in Steps **II** and **IV** and to the semantic index in Step **III** is replaced with a call to the complete reasoner R. The (obvious) technical issue with this approach is scalability, with the main scalability bottleneck being not so much in the detection of unsatisfiable classes, but rather in performing diagnosis using justification-based technologies. For example, when running LOGMAP-FULL with FMA and SNOMED as input ontologies, we obtain 3,351 unsatisfiable classes in Step **II**; computing all justifications for each unsatisfiable class required, on average, more than 9 minutes,[2] which implies that LOGMAP-FULL would need more than 3 weeks to complete Step **II** (when LOGMAP does it in under 82 seconds).

To address these scalability issues, LOGMAP-FULL implements the "greedy" diagnosis algorithm in Table 1, which avoids computing all justifications for each unsatisfiable class. The algorithm uses R to check for unsatisfiable classes (Line 2) and to compute a single justification for each unsatisfiable class (Line 5); this is feasible since computing only one justification is much easier in practice than computing all of them [16, 15]. In Line 7, the algorithm heuristically selects a set of mappings $\mathsf{ConflictiveMappings}(Just)$ containing at least one mapping per justification in $Just$. A single iteration of this process does not guarantee that all unsatisfiabilities will be resolved, so the process needs to be repeated until no more unsatisfiable classes can be found. This algorithm is quite different from the one used in LOGMAP, where the computation of justifications was not an issue (see [10] for details).

[1] Given a class A that is unsatisfiable w.r.t. an ontology $\mathcal{O}$, a justification $\mathcal{O}'_A$ is a subset of $\mathcal{O}$ such that *(i)* A is unsatisfiable w.r.t. $\mathcal{O}'_A$ and *(ii)* A is satisfiable w.r.t. each strict subset of $\mathcal{O}'_A$.
[2] Using HermiT reasoner [24, 20] and the optimisation proposed in [28].

Table 2. LOGMAP and LOGMAP-FULL diagnosis times (s)

Diagnosis of MOUSE-NCI Anatomy			
System	**Step II**	**Step III**	**Step IV**
LOGMAP	0.7	0.3	0.2
LOGMAP-FULL $_{\text{HermiT}}$	10.6	1.8	2.0
LOGMAP-FULL $_{\text{Pellet}}$	7.7	0.4	0.2
LOGMAP-FULL $_{\text{FaCT++}}$	16.4	0.6	1.9
Diagnosis of FMA-NCI			
System	**Step II**	**Step III**	**Step IV**
LOGMAP	14.6	3.4	11.6
LOGMAP-FULL $_{\text{HermiT}}$	469.7	54.3	1,550
LOGMAP-FULL $_{\text{Pellet}}$	392.5	25.8	2,787
Diagnosis of FMA-SNOMED			
System	**Step II**	**Step III**	**Step IV**
LOGMAP	81.4	21.3	87.7
LOGMAP-FULL $_{\text{HermiT}}$	2,628	479.6	11,018
LOGMAP-FULL $_{\text{Pellet}}$	21,477	1,351	$>10^5$
Diagnosis of SNOMED-NCI			
System	**Step II**	**Step III**	**Step IV**
LOGMAP	182.9	142.7	237

4 Evaluation

We have tested LOGMAP and LOGMAP-FULL with the largest ontologies of the OAEI 2011.5 campaign: SNOMED CT (306,591 classes), NCI (66,724 classes), FMA (78,989 classes), MOUSE (2,744 classes), and the anatomy fragment of NCI (3,304 classes). For the experiments we have used a high performance server with 15 Gb of RAM. Table 2 summarises the computation times for the Steps **II**-**IV** in LOGMAP and LOGMAP-FULL. LOGMAP-FULL has been tested with three well-known OWL 2 DL reasoners:[3] Pellet [27], FaCT++ [29] and HermiT [24, 20].

LOGMAP-FULL was able to efficiently handle MOUSE and NCI Anatomy for each of the three tested reasoners and reported times in line with LOGMAP. LOGMAP-FULL, however, did not terminate when given SNOMED and NCI as input for any of the evaluated reasoners. Furthermore, FaCT++ could not process any input involving FMA, and hence failed to produce an output for FMA-NCI and FMA-SNOMED.

When using HermiT and Pellet, LOGMAP-FULL did successfully compute output mappings for FMA-NCI and the computation times, although much higher than those reported by LOGMAP, were in line with many of the tools participating in the OAEI 2011.5 campaign.[4] Note that many of these matching tools do not perform any kind of reasoning, and hence LOGMAP-FULL's computation times are surprisingly good. Times for FMA-SNOMED, however, increased dramatically (especially when using Pel-

[3] LOGMAP-FULL was not tested with the OWL 2 EL reasoner ELK [17, 18] because it does not implement yet the axiom pinpointing service.

[4] `http://www.cs.ox.ac.uk/isg/projects/SEALS/oaei/index.html`

let, where Step **II** required almost 6 hours and Step **IV** did not finish after 4 days); these results are in contrast to the low computation times obtained by LOGMAP.

Finally, it is worth mentioning that LOGMAP output mappings only led to two unsatisfiable classes for the FMA-NCI case.[5] As expected, LOGMAP-FULL produced a clean output for all cases it could successfully process.

5 Conclusions

In this paper we have evaluated the feasibility of using full OWL 2 DL reasoning capabilities for "on-the-fly" mapping diagnosis. For this purpose, we have developed LOGMAP-FULL as an extension of our ontology matching systems LOGMAP.

Our empirical results suggest that the use of LOGMAP-FULL is feasible for medium-sized ontologies such as MOUSE and NCI Anatomy. For larger and semantically richer ontologies, however, computation times increase considerably; thus, LOGMAP seems to be a better choice than LOGMAP-FULL for applications with strict scalability demands (i.e., applications where user intervention is required to obtain high precision mappings); note, however, that LOGMAP-FULL's computation times are still competitive with (and in many cases faster than) most existing matching tools.

Finally, we have shown that reasoning with the integration of large-scale ontologies via mappings still poses serious problems to current OWL 2 DL reasoners. Hence, these integrated ontologies seem ideal as reasoning benchmarks.

Acknowledgements

This work was supported by the Royal Society, the EU FP7 project SEALS and by the EPSRC projects ConDOR, ExODA and LogMap. We also thank the organisers of the SEALS and OAEI evaluation campaigns for providing test data and infrastructure.

References

1. Agrawal, R., Borgida, A., Jagadish, H.V.: Efficient management of transitive relationships in large data and knowledge bases. SIGMOD Rec. 18, 253–262 (1989)
2. Armas Romero, A., Cuenca Grau, B., Horrocks, I.: Modular combination of reasoners for ontology classification. In: Proc. of the 25th Description Logics workshop (2012)
3. Christophides, V., Plexousakis, D., Scholl, M., Tourtounis, S.: On labeling schemes for the Semantic Web. In: Proc. of WWW. pp. 544–555. ACM (2003)
4. Dowling, W.F., Gallier, J.H.: Linear-time algorithms for testing the satisfiability of propositional Horn formulae. J. Log. Program. pp. 267–284 (1984)
5. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology Alignment Evaluation Initiative: six years of experience. J Data Semantics (2011)

[5] To the best of our knowledge, no OWL 2 DL reasoner can cope with SNOMED-NCI (not even with the optimization proposed in [2]) and thus, we were not able to check if LOGMAP's output was 'clean" for this case. The OWL 2 EL reasoner ELK could classify SNOMED-NCI and reported no unsatisfiable classes; the version of NCI we are using, however, is not in OWL 2 EL and hence ELK might be incomplete.

6. Euzenat, J., Ferrara, A., van Hage, W.R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Sváb-Zamazal, O., Trojahn dos Santos, C.: Results of the Ontology Alignment Evaluation Initiative 2011. 6th OM workshop (2011)
7. Gallo, G., Urbani, G.: Algorithms for testing the satisfiability of propositional formulae. The Journal of Logic Programming 7(1), 45 – 61 (1989)
8. Horridge, M., Parsia, B., Sattler, U.: Laconic and precise justifications in OWL. In: Proc. of International Semantic Web Conference. pp. 323–338 (2008)
9. Huber, J., Sztyler, T., Nößner, J., Meilicke, C.: CODI: Combinatorial optimization for data integration: results for OAEI 2011. In: Proc. of the 6th OM Workshop (2011)
10. Jiménez-Ruiz, E., Cuenca Grau, B.: LogMap: Logic-based and Scalable Ontology Matching. In: Proc. of the 10th International Semantic Web Conference (ISWC). pp. 273–288 (2011)
11. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Ontology integration using mappings: Towards getting the right logical consequences. In: Proc. of ESWC (2009)
12. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Logic-based assessment of the compatibility of UMLS ontology sources. J Biomed. Sem. 2 (2011)
13. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: Proc. of ECAI (2012)
14. Jiménez-Ruiz, E., Morant, A., Cuenca Grau, B.: LogMap results for OAEI 2011. In: Proc. of the 6th OM Workshop (2011)
15. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of OWL DL entailments. In: ISWC 2007. pp. 267–280 (2007)
16. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.A.: Debugging unsatisfiable classes in OWL ontologies. J. Web Sem. 3(4), 268–293 (2005)
17. Kazakov, Y., Krötzsch, M., Simancik, F.: Concurrent classification of EL ontologies. In: Proceedings of the 10th International Semantic Web Conference (ISWC). pp. 305–320 (2011)
18. Kazakov, Y., Krötzsch, M., Simancik, F.: ELK reasoner: Architecture and evaluation. In: Proceedings of the 1st International OWL Reasoner Evaluation Workshop (ORE) (2012)
19. Meilicke, C.: Alignment Incoherence in Ontology Matching. Ph.D. thesis, University of Mannheim, Chair of Artificial Intelligence (2011)
20. Motik, B., Shearer, R., Horrocks, I.: Hypertableau Reasoning for Description Logics. Journal of Artificial Intelligence Research 36, 165–228 (2009)
21. Nebot, V., Berlanga, R.: Efficient retrieval of ontology fragments using an interval labeling scheme. Inf. Sci. 179(24), 4151–4173 (2009)
22. Niepert, M., Meilicke, C., Stuckenschmidt, H.: A probabilistic-logical framework for ontology matching. In: Proc. of AAAI (2010)
23. Schlobach, S., Huang, Z., Cornet, R., van Harmelen, F.: Debugging incoherent terminologies. J. Autom. Reasoning 39(3), 317–349 (2007)
24. Shearer, R., Motik, B., Horrocks, I.: HermiT: A highly-efficient OWL reasoner. In: Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions (2008)
25. Shvaiko, P., Euzenat, J.: Ten challenges for ontology matching. In: On the Move to Meaningful Internet Systems (OTM Conferences) (2008)
26. Shvaiko, P., Euzenat, J.: Ontology matching: State of the art and future challenges. IEEE Trans. Knowl. Data Eng. 99 (2011)
27. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. J. Web Sem. 5(2), 51–53 (2007)
28. Suntisrivaraporn, B., Qi, G., Ji, Q., Haase, P.: A modularization-based approach to finding all justifications for OWL DL entailments. In: 3rd Asian Semantic Web Conference (2008)
29. Tsarkov, D., Horrocks, I.: FaCT++ Description Logic Reasoner: System Description. In: Third International Joint Conference on Automated Reasoning, IJCAR. pp. 292–297 (2006)