Efficient OWL Reasoning with Logic Programs – Evaluations[⋆]

Sebastian Rudolph[1], Markus Krötzsch[1], Pascal Hitzler[1], Michael Sintek[2], and Denny Vrandecic[1]

[1] Institute AIFB, Universität Karlsruhe, Germany
[2] DFKI Kaiserslautern, Germany

Abstract. We report on efficiency evaluations concerning two different approaches to using logic programming for OWL [1] reasoning and show, how the two approaches can be combined.

Introduction. Scalability of reasoning remains one of the major obstacles in leveraging the full power of the Web Ontology Language OWL [1] for practical applications. Among the many possible approaches to address scalability, one of them concerns the use of logic programming for this purpose. It was recently shown that reasoning in Horn-$\mathcal{SHIQ}$ [2–4] can be realised by invoking Prolog systems on the output of the KAON2-transformations [5]. Still, performance experiments had not been reported yet.

An entirely different effort to leveraging logic programming for OWL reasoning rests on the idea of approximate reasoning, by allowing some incorrect inferences in order to speed up the reasoning. First experiments with an implementation – called Screech [6], have been encouraging.

In this paper, we report on evaluations concerning the feasibility of the two mentioned approaches. We performed corresponding experiments using the ontologies GALEN, DOLCE, WINE and SEMINTEC.

The KAON2-Transformation. Reasoning with KAON2 is based on special-purpose algorithms which have been designed for dealing with large ABoxes, detailed in [2]. The KAON2 approach transforms OWL DL ontologies to disjunctive datalog, and applies established algorithms for dealing with this formalism. The program returned by the transformation algorithm is in general not logically equivalent to the input TBox, but equisatisfiable, which is sufficient for most reasoning problems.

Convenient access to the KAON2 transformation algorithm is given by means of the KAON2 OWL Tool[3] dlpconvert,[4] which can also produce F-Logic serialisations which can be used with F-Logic engines like OntoBroker.

⋆ This work is partially supported by the German Federal Ministry of Education and Research (BMBF) under the SmartWeb project (grant 01 IMD01), by the Deutsche Forschungsgemeinschaft (DFG) under the ReaSem project and by the EU in the IST project NeOn (IST-2006-027595), http://www.neon-project.org/.

[3] http://owltools.ontoware.org/

[4] http://logic.aifb.uni-karlsruhe.de/wiki/Dlpconvert

Approximate OWL-Reasoning with Screech. Screech uses a modified notion of *split program* [7] to deal with disjunctive datalog: for any rule $H_1 \vee \cdots \vee H_m \leftarrow A_1, \ldots, A_k$, from the output of the KAON2 transformation algorithm, the *derived split rules* are defined as: $H_1 \leftarrow A_1, \ldots, A_k \quad \ldots \quad H_m \leftarrow A_1, \ldots, A_k$. The *split program* P' of a given program P, obtained by splitting all its rules, is complete but may be unsound wrt. instance retrieval tasks. Note that the data complexity for this is polynomial since P' is (non-disjunctive) datalog. A prototype implementation of our approach is available as the Screech OWL approximate reasoner.[5]

Experiments and Evaluation. An approximate reasoning procedure needs to be evaluated on real data from practical applications. So we evaluated the following popular publicly available ontologies: the GALEN Upper Ontology[6], DOLCE[7], the WINE ontology[8], and SEMINTEC[9]. For each of these ontologies, we measured the time and precision for retrieving the extensions of all named classes. The results of our evaluations are summarized in the table below, where also the fraction of disjunctive rules in the KAON2 output can be found for each ontology.

ontology	time saved	correct instances	correct class extensions	disjunctive rules
GALEN	38.0%	91.8%	138/175	54/1449
DOLCE	29.1%	62.1%	93/123	47/1797
WINE	34.5%	95.8%	131/140	26/559
SEMINTEC	67.3%	100%	59/59	0/221

The Data-Tractable OWL Fragment Horn-$\mathcal{SHIQ}$. Horn-$\mathcal{SHIQ}$ is defined as the fragment of OWL DL for which the disjunctive datalog program obtained from the KAON2 transformation is in fact non-disjunctive, i.e. Horn [2, 3]. A direct definition using a grammar is due to [4]. Horn-$\mathcal{SHIQ}$'s data complexity is polynomial, qualifying it as a tractable description logic [8]. Its combined complexity is still exponential [4].

In [5] was shown that using off-the shelf Prolog implementations for reasoning with Horn-$\mathcal{SHIQ}$ after the KAON2-transformation is possible in principle by using Prolog with tabling, as implemented e.g. in the XSB system.

It turns out, however, that tabling is too expensive for our test ontologies. For none of our test ontologies, XSB with tabling was able to produce answers to the queries, which shows that it cannot be used naively on realistic data.

Using OntoBroker 5.0 build 690, the results were similar. OntoBroker could be used with our test ontologies only with bottom-up reasoning, which is the least efficient of the reasoning strategies. Consequently, performance was much worse if compared with Screech on the KAON2 datalog engine, with about factor 20 for SEMINTEC and factor 100 for WINE. For the GALEN ontology, however, OntoBroker performed drastically better than the KAON2 datalog engine: querying for the extensions of all classes, on average, OntoBroker performed better than the KAON2 datalog engine on the screeched

[5] http://logic.aifb.uni-karlsruhe.de/screech
[6] http://www.cs.man.ac.uk/~rector/ontologies/simple-top-bio/
[7] http://www.loa-cnr.it/DOLCE.html
[8] http://www.schemaweb.info/schema/SchemaDetails.aspx?id=62
[9] http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm

version, by a factor of 3.9. In this experiment, the speedup by SCREECH compared to the unscreeched version on KAON2 was at a factor of 11.4. Overall, using a combination of SCREECH and OntoBroker we obtained a speedup of factor 44.7 compared to using KAON2 on the unscreeched version.

Discussion. The two approaches which we presented can be combined in a straightforward manner, namely by first screeching a given TBox and then performing the subsequent reasoning on a logic programming engine. The results presented for the GALEN ontology indicate that a significant speedup is possible, in this case by an overall factor of 44.7 (i.e. 97.8% time saved), while 91.8% of the retrieved instances are correct.

The SCREECH part of the performance improvement is stable over all tested ontologies. The gain varied between 29.1 and 67.3 %, the amount of correctly retrieved instances was above 91.8% for all but one of the ontologies. It is encouraging that the approach appears to be feasible even for the sophisticated WINE ontology.

Concerning the use of logic programming systems for improving Horn-$\mathcal{SHIQ}$ performance, the results were mostly discouraging, because a naive application of such systems was not possible. However, the drastic speed-up of factor 11.4 (i.e. 91.2% time saved) compared to SCREECH obtained with OntoBroker in bottom-up setting on the GALEN ontology indicates that a special-purpose logic programming system should frequently be able to outperform KAON2 on Horn-$\mathcal{SHIQ}$. But specialised implementations may be needed for this purpose as the off-the-shelf systems are currently not applicable in general.

References

1. McGuinness, D.L., van Harmelen, F.: OWL web ontology language overview (February 2004) http://www.w3.org/TR/owl-features/.
2. Motik, B.: Reasoning in Description Logics using Resolution and Deductive Databases. PhD thesis, Universität Karlsruhe (2006)
3. Hustadt, U., Motik, B., Sattler, U.: Data complexity of reasoning in very expressive description logics. In Kaelbling, L.P., Saffiotti, A., eds.: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland. (2005) 466–471
4. Krötzsch, M., Rudolph, S., Hitzler, P.: On the complexity of Horn description logics. In Grau, B.C., Hitzler, P., Shankey, C., Wallace, E., eds.: Proceedings of the 2nd Workshop on OWL: Experiences and Directions. Volume 216 of CEUR Workshop Proceedings. (November 2006)
5. Krötzsch, M., Hitzler, P., Vrandecic, D., Sintek, M.: How to reason with OWL in a logic programming system. In Eiter, T., Franconi, E., Hodgson, R., Stephens, S., eds.: Proceedings of the Second International Conference on Rules and Rule Markup Languages for the Semantic Web, RuleML2006, Athens, Georgia, IEEE Computer Society (2006) 17–26
6. Hitzler, P., Vrandecic, D.: Resolution-based approximate reasoning for OWL DL. In Gil, Y., et al., eds.: Proceedings of the 4th International Semantic Web Conference, Galway, Ireland, November 2005. Volume 3729 of Lecture Notes in Computer Science., Springer, Berlin (2005) 383–397
7. Sakama, C., Inoue, K.: An alternative approach to the semantics of disjunctive logic programs and deductive databases. Journal of Automated Reasoning **13** (1994) 145–172
8. Grau, B.C.: OWL 1.1 web ontology language tractable fragments (November 2004) http://owl1-1.cs.manchester.ac.uk/tractable.html.