NoHR: Querying $\mathcal{EL}$ with Non-monotonic rules

Vadim Ivanov[1,2], Matthias Knorr[1], and João Leite[1]

[1] CENTRIA & Departamento de Informática, Universidade Nova de Lisboa, Portugal
[2] Department of Computing Mathematics and Cybernetics, Ufa State Aviation Technical University, Russia

Abstract. We present NoHR, a Protégé plug-in that allows the user to take an $\mathcal{EL}^+_\bot$ ontology, add a set of non-monotonic (logic programming) rules – suitable e.g. to express defaults and exceptions – and query the combined knowledge base. Provided the given ontology alone is consistent, the system is capable of dealing with potential inconsistencies between the ontology and the rules, and, after an initial brief pre-processing period utilizing OWL 2 EL reasoner ELK, returns answers to queries at an interactive response time by means of XSB Prolog.

1 Introduction

Ontology languages have become widely used to represent and reason over taxonomic knowledge, and often such knowledge bases are expressed within the language of the OWL 2 profile OWL 2 EL.[1] For example, the clinical health care terminology SNOMED CT,[2], arguably the most prominent example in the area of medicine and currently used for electronic health record systems, clinical decision support systems, or remote intensive care monitoring, to name only a few, builds on a fragment of OWL 2 EL and its underlying description logic (DL) $\mathcal{EL}^{++}$ [2].

Since OWL and its profiles are based on DLs [3], hence monotonic by nature, which means that once drawn conclusions persist when adopting new additional information, the ability to model defaults and exceptions with a closed-world view is frequently requested as a missing feature. For example, in clinical health care terminology, it would be advantageous to be able to express directly that normally the heart is on the left side of the body unless the person is a dextrocardiac, which matters when applying ECG or defibrillation to a patient.

In recent years, there has been a considerable amount of effort devoted to extending DLs with non-monotonic features – see, e.g., related work in [8] – many of the existing approaches focusing on combining DLs and non-monotonic rules. The latter are one of the most well studied formalisms (in the area of Logic Programming) that admit expressing defaults, exceptions, and also integrity constraints in a declarative way. As such, they are part of the RIF,[3] the other language for the Semantic Web whose standardization is driven by the W3C.[4]

[1] http://www.w3.org/TR/owl2-profiles/
[2] http://www.ihtsdo.org/snomed-ct/
[3] http://www.w3.org/TR/rif-overview/
[4] http://www.w3.org

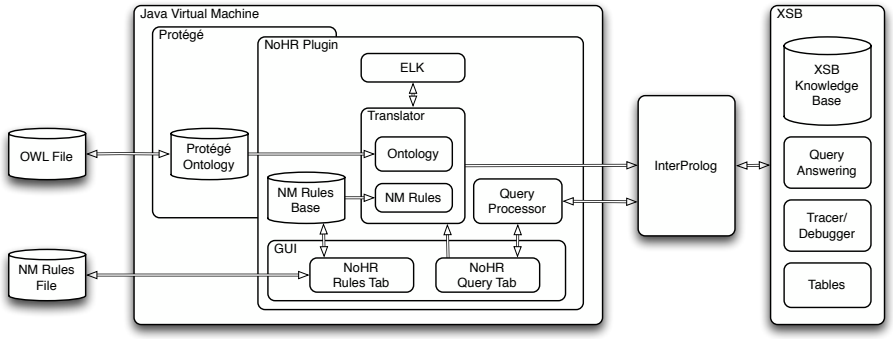

Fig. 1. System Architecture of NoHR

Here, we focus on Hybrid MKNF under the well-founded semantics [7] combining ontologies and such rules, because, as argued for the preceding semantics in [8], the overall framework is very general and flexible, and unlike [8], [7] has a polynomial data complexity and admits top-down query-answering based only on the information relevant for the query, and without computing the entire model – no doubt a crucial feature when dealing with large ontologies such as SNOMED with over 300,000 classes.

In our ISWC 2013 Research Track paper [5], we describe a system, realized as a plug-in for the ontology editor Protégé 4.X,[5] that allows the user to query combinations of $\mathcal{EL}^+_\bot$ ontologies and non-monotonic rules in a top-down manner. To the best of our knowledge, it is the first Protégé plug-in to integrate non-monotonic rules and top-down queries. Our approach is theoretically founded on the abstract procedure **SLG**($\mathcal{O}$) [1] and developed upon the usage of the consequence-driven, concurrent $\mathcal{EL}$ reasoner ELK [6] to classify the ontology part, whose result is translated into rules which, together with the non-monotonic rules, subsequently serve as input for the top-down query engine XSB Prolog.[6] Additional features of the plug-in include: the possibility to load and edit rule bases, and define predicates with arbitrary arity; guaranteed termination of query answering, with a choice between one/many answers; robustness w.r.t. potential inconsistencies between the ontology and the rules in case the $\mathcal{EL}^+_\bot$ ontology contains $DisjointWith$ axioms; leveraging of XSB tabling mechanisms to improve performance, and trace/debug features, e.g., to provide explanations.

2 System Description

In this Section, we briefly describe the architecture of NoHR, our plug-in for Protégé, as shown in Fig. 1 and discuss some features of our implementation and querying in XSB. For the technical details and the evaluation of our approach, we refer to [5].

The input for our plug-in consists of an OWL file, which can be manipulated as usual in Protégé, and a rule file. For the latter, we provide a tab called NoHR Rules that

[5] http://protege.stanford.edu
[6] http://xsb.sourceforge.net

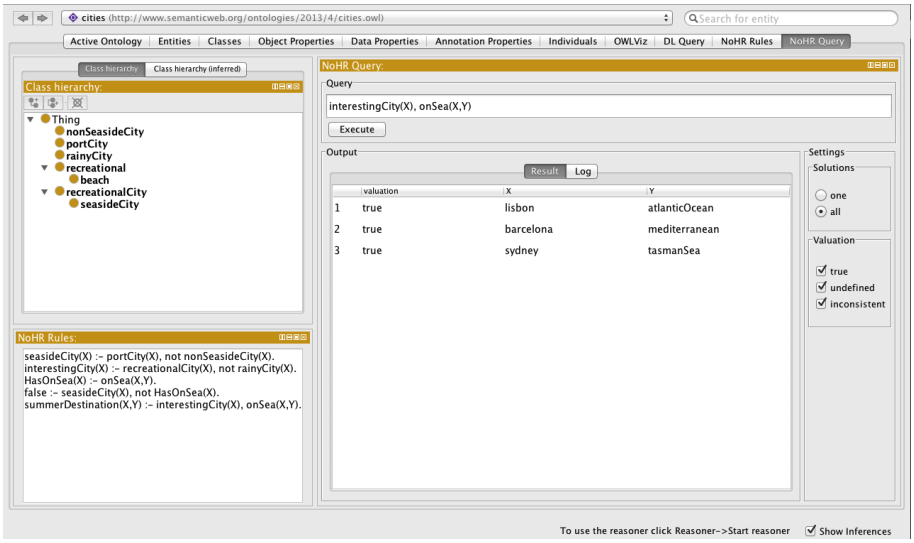

Fig. 2. NoHR Query Tab with a query $interestingCity(X), onSea(X,Y)$

allows the user to load, save and edit rule files in a text panel. The syntax follows Prolog conventions, so that one rule from Ex. 2 in [5] can be represented, e.g., by

```
SeaSideCity(X) :- PortCity(X), not NonSeaSideCity(X).
```

The NoHR Query tab as shown in Fig. 2 also allows for the visualization of the rules (in the lower left corner), but its main purpose is to provide an interface for querying the combined KB. Whenever the first query is posed by pushing "Execute", the translator is started, initiating the ELK reasoner to classify the ontology and return the result to the translator. It is verified whether $DisjointWith$ axioms appear in $\mathcal{O}$ which determines whether the transformation into rules has to contain means to check for potential inconsistencies or not. Then, accordingly, a joint (non-monotonic) rule set is created in which predicates and constants, i.e., all terms, are encoded using MD5. This requires the user to write case-sensitive rules (w.r.t. to the ontology), but ensures full compatibility with XSB Prolog's more restrictive admitted input syntax. The resulting program is transfered to XSB via InterProlog [4], which is an open-source Java front-end that provides the ability to communicate between Java and a Prolog engine.

Next, the query can be sent via InterProlog to XSB, and answers are returned to the query processor, which collects them and sets up a table showing for which variable substitutions we obtain *true*, *undefined*, or *inconsistent* valuations (or just shows the truth value for a ground query). The table itself is shown in the Result tab of the Output panel (see Fig. 2), while the Log tab shows measured times and system messages, including those from XSB via InterProlog. XSB not only answers queries very efficiently in a top-down manner, with tabling, it also avoids infinite loops.

Once the query has been answered, the user may pose other queries, and the system will simply send them to XSB directly without any repeated preprocessing. If the user changes data in the ontology or in the rules, then the system offers the option to recompile, but always restricted to the part that actually changed.

During the demo exhibition, we take a given/chosen ontology loaded into Protégé, and we show, first how to edit, load, and save rules, and subsequently, run queries on the combined knowledge base. In particular, we interactively demonstrate how changing the ontology and the rules affects the query results, also in the presence of inconsistencies between the ontology and the rule set. For that purpose, we use data sets of two kinds, namely toy examples for which the query result can be verified right away and some of the real world ontologies, utilized already during testing in [5] (cf. Fig.3), to which we add non-monotonic rules.

Our plug-in is under active development and the most recent version is available at https://code.google.com/p/nohr-reasoner/. The example file sets for testing can also be found on this web page.

Acknowledgments. We would like to thank Miguel Calejo for his help with InterProlog, Pavel Klinov for his help with ELK, Terry Swift for his help with XSB, and Gonca Güllü for her collaboration. Vadim Ivanov was partially supported by a MULTIC – Erasmus Mundus Action 2 grant. Matthias Knorr and João Leite were partially supported by FCT funded project ERRO – Efficient Reasoning with Rules and Ontologies (PTDC/EIA-CCO/121823/2010) and Matthias Knorr also by FCT grant SFRH/BPD/86970/2012.

References

1. Alferes, J.J., Knorr, M., Swift, T.: Query-driven procedures for hybrid MKNF knowledge bases. ACM TOCL 14(2) (2013)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: IJCAI'05: 19th Int. Joint Conf. on Artificial Intelligence. pp. 364–369. Morgan Kaufmann (2005)
3. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 3rd edn. (2010)
4. Calejo, M.: Interprolog: Towards a declarative embedding of logic programming in java. In: Alferes, J.J., Leite, J.A. (eds.) JELIA. Lecture Notes in Computer Science, vol. 3229, pp. 714–717. Springer (2004)
5. Ivanov, V., Knorr, M., Leite, J.: A query tool for $\mathcal{EL}$ with non-monotonic rules. In: ISWC 2013. Springer (2013), to appear
6. Kazakov, Y., Krötzsch, M., Simančík, F.: Concurrent classification of $\mathcal{EL}$ ontologies. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) Proceedings of the 10th International Semantic Web Conference (ISWC'11). LNCS, vol. 7032. Springer (2011)
7. Knorr, M., Alferes, J.J., Hitzler, P.: Local closed world reasoning with description logics under the well-founded semantics. Artif. Intell. 175(9–10), 1528–1554 (2011)
8. Motik, B., Rosati, R.: Reconciling description logics and rules. J. ACM 57(5) (2010)