# NoHR: A Protégé Plugin for Polynomial Querying of Ontologies and Non-Monotonic Rules

Matthias Knorr and João Leite

NOVA LINCS, Departamento de Informática, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa

**Abstract.** Ontology languages and non-monotonic rule languages are both well-known formalisms for knowledge representation, each with its own distinct benefits and features, often orthogonal to each other. Both appear in the Semantic Web stack in distinct standards – OWL and RIF. Over the last decade, a considerable research effort has been put into trying to provide a framework that combines the two. Yet, the considerable number of theoretical solutions was accompanied by very few practical applications employing hybrid knowledge bases composed of combinations of ontologies and rules. One of the reasons may be the lack of viable practical tools to reason with such hybrid knowledge.

In this paper, we present NoHR (Nova Hybrid Reasoner), a plug-in for the ontology editor Protégé – the first of its kind – that allows its users to query knowledge bases composed of both an ontology and a set of non-monotonic rules, based on a semantics which ensures that reasoning over these combined knowledge bases is polynomial. Using a top-down reasoning approach, which ensures that only the part of the ontology and rules that is relevant for the query is actually evaluated, NoHR combines the capabilities of ELK and a dedicated direct translation with the rule engine XSB Prolog to deliver very fast interactive response times. Test results performed with large hybrid knowledge bases show that NoHR is able to scale, hence offering a viable solution as the underlying reasoner for applications using hybrid knowledge bases.

## 1 Introduction

Ontology languages in the form of Description Logics (DLs) [4] and non-monotonic rule languages as known from Logic Programming (LP) [6] are both well-known formalisms in knowledge representation and reasoning (KRR) each with its own distinct benefits and features. This is also witnessed by the emergence of the Web Ontology Language (OWL) [13] and the Rule Interchange Format (RIF) [18] in the ongoing standardization of the Semantic Web driven by the W3C. [1]

On the one hand, ontology languages have become widely used to represent and reason over taxonomic knowledge. Since DLs are (usually) decidable fragments of first-order logic, they are monotonic by nature, which means that drawn conclusions persist when adopting new additional information. Furthermore, they allow reasoning on abstract information, such as relations between classes of objects, even without knowing

[1] http://www.w3.org

any concrete instances. The balance between expressiveness and complexity of reasoning with ontology languages, inherited from DLs, is witnessed by the fact that the very expressive general language OWL 2, with its high worst-case complexity, includes three tractable (polynomial) profiles [22] each with a different application purpose in mind.

On the other hand, non-monotonic rules explicitly represent inference, from premises to conclusions, focusing on reasoning over instances. They commonly employ the Closed World Assumption (CWA), i.e., the absence of a piece of information suffices to derive it being false, until new information to the contrary is provided, hence being non-monotonic. This permits to declaratively model defaults and exceptions, in the sense that the absence of an exceptional feature can be used to derive that the (more) common case applies, and also integrity constraints, which can be used to ensure that the data under consideration is conform to the desired specifications.

Combining both formalisms has been frequently requested by applications [24, 25, 1, 26]. For example, in clinical health care, large ontologies such as SNOMED CT,[2] that are captured by the OWL 2 profile OWL 2 EL and its underlying description logic (DL) $\mathcal{EL}^{++}$ [5], are used for electronic health record systems, clinical decision support systems, or remote intensive care monitoring, to name only a few. Yet, expressing conditions such as dextrocardia, i.e., that the heart is exceptionally on the right side of the body, is not possible and requires non-monotonic rules. Another example can be found in [24], where modeling pharmacy data of patients with the closed-world assumption would have been preferred in the study to match patient records with clinical trials criteria, because usually it can be assumed that a patient is not under a specific medication unless explicitly known. In [1] it is shown that in Legal Reasoning, besides the well known need for default reasoning afforded by non-monotonic rules, it is also necessary to reason in the absence of concrete known individuals (instances), hence requiring features found in ontology languages such as DL.

Finding such a combination is a non-trivial problem due to the mismatch between semantic assumptions of the two formalisms, and the considerable differences as to how decidability is ensured in each of them, where a naive combination can easily become undecidable. In recent years, there has been a considerable amount of effort devoted to combining DLs with non-monotonic rules – see, e.g., related work in [10, 23] – but this has not been accompanied by similar variety of reasoners and applications.

In this paper, we describe NoHR[3] (Nova Hybrid Reasoner), a plug-in for the ontology editor Protégé 5.0,[4] that allows the user to query combinations of $\mathcal{EL}^{+}_{\bot}$ or $DL\text{-}Lite_R$ ontologies and non-monotonic rules in a top-down manner.

NoHR is theoretically founded on the formalism of Hybrid MKNF under the well-founded semantics [19] which comes with two main arguments in its favor. First, the overall approach, which was introduced in [23] and is based on the logic of minimal knowledge and negation as failure (MKNF) [21], provides a very general and flexible framework for combining DL ontologies and non-monotonic rules (see [23]). Second, [19], which is a variant of [23] based on the well-founded semantics [11] for logic programs, has a lower data complexity than the former – it is polynomial for polynomial

[2] http://www.ihtsdo.org/snomed-ct/
[3] http://centria.di.fct.unl.pt/nohr/
[4] http://protege.stanford.edu

DLs – and is amenable for applying top-down query procedures, such as **SLG**($\mathcal{O}$) [2], to answer queries 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 and huge amounts of data.

NoHR – the first Protégé plug-in to integrate non-monotonic rules and top-down queries – is implemented in a way that combines the capabilities of the DL reasoner ELK [17] (for $\mathcal{EL}_{\bot}^{+}$) and a dedicated direct translation (for $DL\text{-}Lite_R$) with the rule engine XSB Prolog,[5] exhibiting the following additional features:

- rule editor within Protégé;
- possibility to define predicates with arbitrary arity;
- guaranteed termination of query answering;
- choice between one/many answers;
- robustness w.r.t. inconsistencies between the ontology and the rules;
- scalable fast interactive response times.

## 2 Preliminaries

We start with some preliminary notions to illustrate the kind of hybrid knowledge bases considered by NoHR. We begin with some notation on the description logics underlying the OWL 2 profiles, EL and QL, referring to [4] for a more general and thorough introduction to DLs. Then we present hybrid MKNF knowledge bases, followed by a basic description on how to perform query answering.

### 2.1 Description Logics

The language of $\mathcal{EL}_{\bot}^{+}$, a large fragment of $\mathcal{EL}^{++}$ [5], the DL underlying the tractable profile OWL 2 EL [22], is defined over countably infinite sets of *concept names* $\mathsf{N_C}$, *role names* $\mathsf{N_R}$, and *individual names* $\mathsf{N_I}$ as shown in the upper part of Table 1. Building on these, *complex concepts* are introduced in the middle part of Table 1, which, together with atomic concepts, form the set of *concepts*. We conveniently denote individuals by $a$ and $b$, (atomic) roles by $R$ and $S$, atomic concepts by $A$, and concepts by $B$, $C$ and $D$. All expressions in the lower part of Table 1 are *axioms*. A *concept equivalence* $C \equiv D$ is an abbreviation for $C \sqsubseteq D$ and $D \sqsubseteq C$. Concept and role assertions are *ABox axioms* and all other axioms *TBox axioms*, and an *ontology* is a finite set of axioms.

The semantics of $\mathcal{EL}_{\bot}^{+}$ is defined in terms of an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a non-empty domain $\Delta^{\mathcal{I}}$ and an *interpretation function* $\cdot^{\mathcal{I}}$. The latter is defined for (arbitrary) concepts, roles, and individuals as in Table 1. Moreover, an interpretation $\mathcal{I}$ *satisfies* an axiom $\alpha$, written $\mathcal{I} \models \alpha$, if the corresponding condition in Table 1 holds. If $\mathcal{I}$ satisfies all axioms occurring in an ontology $\mathcal{O}$, then $\mathcal{I}$ is a *model* of $\mathcal{O}$, written $\mathcal{I} \models \mathcal{O}$. If $\mathcal{O}$ has at least one model, then it is called *consistent*, otherwise *inconsistent*. Also, $\mathcal{O}$ *entails* axiom $\alpha$, written $\mathcal{O} \models \alpha$, if every model of $\mathcal{O}$ satisfies $\alpha$. *Classification* requires to compute all concept inclusions between atomic concepts entailed by $\mathcal{O}$.

[5] http://xsb.sourceforge.net

| | Syntax | Semantics |
|---|---|---|
| atomic concept | $A \in \mathsf{N_C}$ | $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ |
| atomic role | $R \in \mathsf{N_R}$ | $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ |
| individual | $a \in \mathsf{N_I}$ | $a^\mathcal{I} \in \Delta^\mathcal{I}$ |
| top | $\top$ | $\Delta^\mathcal{I}$ |
| bottom | $\bot$ | $\emptyset$ |
| conjunction | $C \sqcap D$ | $C^\mathcal{I} \cap D^\mathcal{I}$ |
| existential restriction | $\exists R.C$ | $\{x \in \Delta^\mathcal{I} \mid \exists y \in \Delta^\mathcal{I} : (x, y) \in R^\mathcal{I} \wedge y \in C^\mathcal{I}\}$ |
| concept inclusion | $C \sqsubseteq D$ | $C^\mathcal{I} \subseteq D^\mathcal{I}$ |
| role inclusion | $R \sqsubseteq S$ | $R^\mathcal{I} \subseteq S^\mathcal{I}$ |
| role composition | $R_1 \circ \dots \circ R_k \sqsubseteq S$ | $(x_1, x_2) \in R_1^\mathcal{I} \wedge \ldots \wedge (x_k, y) \in R_k^\mathcal{I} \to (x_1, y) \in S^\mathcal{I}$ |
| concept assertion | $A(a)$ | $a^\mathcal{I} \in A^\mathcal{I}$ |
| role assertion | $R(a, b)$ | $(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$ |

**Table 1.** Syntax and semantics of $\mathcal{EL}^+_\bot$.

In $DL\text{-}Lite_R$ one language of the $DL\text{-}Lite$ family [7, 3] and underlying OWL 2 QL, complex concepts and roles can be formed according to the following grammar

$$B \to A \mid \exists Q \qquad C \to B \mid \neg B \qquad Q \to R \mid R^- \qquad P \to Q \mid \neg Q$$

where, in addition to the already mentioned, $R^-$ is the inverse of the (atomic) role $R$.

A $DL\text{-}Lite_R$ TBox contains concept inclusions of the form $B \sqsubseteq C$ and role inclusions of the form $Q \sqsubseteq P$, and a $DL\text{-}Lite_R$ ABox contains assertions of the form $A(a)$ and $R(a, b)$, with $A$, $B$, $C$, $Q$, $P$, and $R$ defined as above.

The semantics of $DL\text{-}Lite_R$ is also based on interpretations $\mathcal{I}$ with the difference that $\cdot^\mathcal{I}$ assigns to each individual $a$ a distinct[6] element $a^\mathcal{I}$ of $\Delta^\mathcal{I}$ and the following extensions to constructors not shown in Table 1:

$$(P^-)^\mathcal{I} = \{(i_2, i_1) \mid (i_1, i_2) \in P^\mathcal{I}\} \qquad (\neg B)^\mathcal{I} = \Delta^\mathcal{I} \setminus B^\mathcal{I}$$
$$(\exists Q)^\mathcal{I} = \{i \mid (i, i') \in Q^\mathcal{I}\} \qquad (\neg Q)^\mathcal{I} = \Delta^\mathcal{I} \times \Delta^\mathcal{I} \setminus Q^\mathcal{I}$$

The notions of model, (in-)consistency, and entailment coincide with those for $\mathcal{EL}^+_\bot$.

$\mathcal{EL}^+_\bot$ is tailored towards reasoning with large conceptual models, i.e., large TBoxes, while $DL\text{-}Lite_R$ focuses on answering queries over huge amount of data, i.e., large ABoxes. Standard reasoning tasks for both DLs are polynomial, in particular, classification for $\mathcal{EL}^+_\bot$ is PTime-complete, and query-answering for $DL\text{-}Lite_R$ even in $AC^0$.

## 2.2 MKNF Knowledge Bases

MKNF knowledge bases (KBs) build on the logic of minimal knowledge and negation as failure (MKNF) [21]. Two main different semantics have been defined [23, 19], and

[6] Hence, the unique name assumption is applied and, as shown in [3], dropping it would increase significantly the computational complexity of $DL\text{-}Lite_\mathcal{R}$.

we focus on the well-founded version [19], due to its lower computational complexity and amenability to top-down querying without computing the entire model. Here, we only point out important notions following [14], and refer to [19] and [2] for the details.

We start by recalling MKNF knowledge bases as presented in [2] to combine an ontology and a set of non-monotonic rules (similar to a normal logic program).

**Definition 1.** *Let $\mathcal{O}$ be an ontology. A function-free first-order* atom $P(t_1, \ldots, t_n)$ *s.t. $P$ occurs in $\mathcal{O}$ is called* DL-atom*; otherwise* non-DL-atom*. A* rule *$r$ is of the form*

$$H \leftarrow A_1, \ldots, A_n, \mathbf{not}\ B_1, \ldots, \mathbf{not}\ B_m. \tag{1}$$

*where the* head *of $r$, $H$, and all $A_i$ with $1 \leq i \leq n$ and $B_j$ with $1 \leq j \leq m$ in the* body *of $r$ are atoms. A* program *$\mathcal{P}$ is a finite set of rules, and an* MKNF knowledge base *$\mathcal{K}$ is a pair $(\mathcal{O}, \mathcal{P})$. A rule $r$ is* DL-safe *if all its variables occur in at least one non-DL-atom $A_i$ with $1 \leq i \leq n$, and $\mathcal{K}$ is* DL-safe *if all its rules are DL-safe.*

DL-safety ensures decidability of reasoning with MKNF knowledge bases and can be achieved by introducing a new predicate $o$, adding $o(i)$ to $\mathcal{P}$ for all constants $i$ appearing in $\mathcal{K}$ and, for each rule $r \in \mathcal{P}$, adding $o(X)$ for each variable $X$ appearing in $r$ to the body of $r$. Therefore, we only consider DL-safe MKNF knowledge bases.

*Example 2.* Consider an MKNF knowledge base for recommending vacation destinations taken from [23] (with a few modifications). We denote DL-atoms and constants with upper-case names and non-DL-atoms and variables with lower-case names.[7]

$$\begin{aligned}
&\mathit{PortCity}(\mathit{Barcelona}) && \mathit{OnSea}(\mathit{Barcelona}, \mathit{Mediterranean})\\
&\mathit{PortCity}(\mathit{Hamburg}) && \mathit{NonSeaSideCity}(\mathit{Hamburg})\\
&\mathit{RainyCity}(\mathit{Manchester}) && \mathit{Has}(\mathit{Manchester}, \mathit{AquaticsCenter})\\
&\mathit{Recreational}(\mathit{AquaticsCenter}) &&
\end{aligned}$$

$$\begin{aligned}
\mathit{SeaSideCity} &\sqsubseteq \exists \mathit{Has}.\mathit{Beach}\\
\mathit{Beach} &\sqsubseteq \mathit{Recreational}\\
\exists \mathit{Has}.\mathit{Recreational} &\sqsubseteq \mathit{RecreationalCity}\\
\mathit{SeaSideCity}(x) &\leftarrow \mathit{PortCity}(x), \mathbf{not}\ \mathit{NonSeaSideCity}(x)\\
\mathit{interestingCity}(x) &\leftarrow \mathit{RecreationalCity}(x), \mathbf{not}\ \mathit{RainyCity}(x)\\
\mathit{hasOnSea}(x) &\leftarrow \mathit{OnSea}(x, y)\\
\mathit{false} &\leftarrow \mathit{SeaSideCity}(x), \mathbf{not}\ \mathit{hasOnSea}(x)\\
\mathit{summerDestination}(x) &\leftarrow \mathit{interestingCity}(x), \mathit{OnSea}(x, y)
\end{aligned}$$

This example shows that we can seamlessly express defaults and exceptions, such as every port city normally being a seaside city, integrity constraints, such as requiring to know for every seaside city on which sea it lies, and at the same time taxonomic/ontological knowledge including information over unknown individuals, such as a seaside city being recreational even if we do not know the specific name of the beach. Note that, unlike [23], the rule with head *false* is not a true integrity constraint in our

[7] To ease readability, we omit the auxiliary atoms that ensure DL-safety and leave them implicit.

case. Rather, whenever the keyword $false$ would be derivable, we know that there is at least one seaside city for which we do not know on which sea it lies.

The semantics of MKNF knowledge bases $\mathcal{K}$ is usually given by a translation $\pi$ into an MKNF formula $\pi(\mathcal{K})$, i.e., a formula over first-order logic extended with two modal operators $\mathbf{K}$ and $\mathbf{not}$. Namely, every rule of the form (1) is translated into a rule of the form $\mathbf{K}H \leftarrow \mathbf{K}A_1, \ldots, \mathbf{K}A_n, \mathbf{not}\ B_1, \ldots, \mathbf{not}\ B_m$, $\pi(\mathcal{P})$ is the conjunction of the translations of its rules, and $\pi(\mathcal{K}) = \mathbf{K}\pi(\mathcal{O}) \wedge \pi(\mathcal{P})$ where $\pi(\mathcal{O})$ is the first-order translation of $\mathcal{O}$. Reasoning with such MKNF formulas is then commonly achieved using a partition of *modal atoms*, i.e., all expressions of the form $\mathbf{K}\varphi$ for each $\mathbf{K}\varphi$ or $\mathbf{not}\ \varphi$ occurring in $\pi(\mathcal{K})$. For [19], such a partition assigns *true*, *false*, or *undefined* to (modal) atoms, and can be effectively computed in polynomial time. If $\mathcal{K}$ is *MKNF-consistent*, then this partition does correspond to the unique model of $\mathcal{K}$ [19], and, like in [2], we call the partition the *well-founded MKNF model* $\mathsf{M}_{\mathsf{wf}}(\mathcal{K})$. Here, $\mathcal{K}$ may indeed not be MKNF-consistent if the ontology alone is unsatisfiable, or by the combination of appropriate axioms in $\mathcal{O}$ and rules in $\mathcal{P}$, e.g., axiom $A \sqsubseteq \neg B$ in $\mathcal{O}$, and facts $A(a)$ and $B(a)$ in $\mathcal{P}$. In the former case, we argue that the ontology alone should be consistent and be repaired if necessary before combining it with non-monotonic rules. Thus, we assume that $\mathcal{O}$ occurring in $\mathcal{K}$ is consistent, which does not truly constitute a restriction as we can always turn the ABox into rules without any effect on $\mathsf{M}_{\mathsf{wf}}(\mathcal{K})$.

## 2.3 Querying in MKNF Knowledge Bases

In [2], a procedure, called **SLG($\mathcal{O}$)**, is defined for querying MKNF knowledge bases under the well-founded MKNF semantics. This procedure extends SLG resolution with tabling [8] with an *oracle* to $\mathcal{O}$ that handles ground queries to the DL-part of $\mathcal{K}$ by returning (possibly empty) sets of atoms that, together with $\mathcal{O}$ and information already proven true, allows us to derive the queried atom. We refer to [2] for the full account of **SLG($\mathcal{O}$)**, and only discuss central ideas here.

**SLG($\mathcal{O}$)** is based on creating top-down derivation trees with the aim of answering *(DL-safe) conjunctive queries* $Q = q(\boldsymbol{X}) \leftarrow A_1, \ldots, A_n, \mathbf{not}\ B_1, \ldots, \mathbf{not}\ B_m$, where each variable in $Q$ occurs in at least one non-DL atom in $Q$, and where $\boldsymbol{X}$ is the (possibly empty) set of requested variables appearing in the body. Query processing involving an oractle to the ontology is intuitively handled as described next.

*Example 3.* Recall $\mathcal{K}$ in Ex. 2, and consider $q = interestingCity(Manchester)$. We find a rule whose head unifies with $q$, and new queries $RecreationalCity(Manchester)$ and $\mathbf{not}\ RainyCity(Manchester)$ are obtained. There is no rule whose head matches the former, but we can query the ontology and the answer is yes together with an empty set of atoms, i.e., $RecreationalCity(Manchester)$ can be proven from $\mathcal{O}$ alone. Now we handle $\mathbf{not}\ RainyCity(Manchester)$, so we query $RainyCity(Manchester)$ which can also be proven by $\mathcal{O}$ alone. Consequently, $\mathbf{not}\ RainyCity(Manchester)$ fails, so $q$ is false.

Now, consider $q_1 = interestingCity(Barcelona)$. We obtain again two new queries, $q_2 = RecreationalCity(Barcelona)$ and $q_3 = \mathbf{not}\ RainyCity(Barcelona)$. In this case, $q_2 = RecreationalCity(Barcelona)$ cannot be proven from $\mathcal{O}$ alone, but the oracle could return $Has(Barcelona, X)$ and $Recreational(X)$, which, if we would find

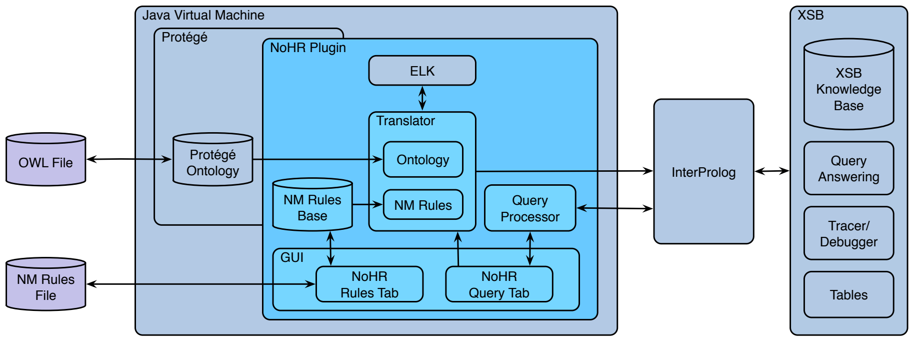

**Fig. 1.** System architecture of NoHR

a value for $X$, would allow us to derive $q_2$. However, neither of the two atoms appear in a rule head in $\mathcal{P}$, so we will never be able to derive it from $\mathcal{P}$. In fact, the only proper answer the oracle may return is $q_4 = SeaSideCity(Barcelona)$. From the corresponding rule in $\mathcal{P}$ we obtain two new queries $q_5 = PortCity(Barcelona)$ and $q_6 = \textbf{not}\ NonSeaSideCity(Barcelona)$. Then, $q_5$ can be derived from $\mathcal{O}$ alone, and $q_6$ succeeds, because $NonSeaSideCity(Barcelona)$ fails. So $q_4$ succeeds, and therefore also $q_2$. Finally $q_3$ succeeds since $RainyCity(Barcelona)$ fails, so $q_1$ is true.

## 3 System Description

In this section, we briefly describe the architecture of the plug-in for Protégé as shown in Fig. 1 and discuss some features of the implementation and how querying is realized.

The input for the plug-in consists of an OWL file in the DL $\mathcal{EL}_{\perp}^{+}$ or $DL\text{-}Lite_R$ as described in Sect. 2.1, which can be manipulated as usual in Protégé, and a rule file. For the latter, we provide a tab called NoHR Rules that allows us to load, save and edit rule files in a text panel following standard Prolog conventions.

The NoHR Query tab also allows for the visualization of the rules, but its main purpose is to provide an interface for querying the combined KB. Whenever the first query is posed by pushing “Execute”, a switch determines the profile of the ontology, upon which the translator is started. For $\mathcal{EL}_{\perp}^{+}$, the ontology reasoner ELK [17], tailored for $\mathcal{EL}_{\perp}^{+}$ and considerably faster than other reasoners when comparing classification time, is used to classify the ontology $\mathcal{O}$. The inferred axioms together with $\mathcal{O}$ are translated discarding certain axioms which are irrelevant for answering ground queries. For $DL\text{-}Lite_R$, a dedicated direct translation without prior classification is used, introducing some auxiliary predicates instead to compensate for the missing inferred axioms (see [14] and [9] for the respective details on both approaches). In both cases, the translation result is joined with the given non-monotonic rules in $\mathcal{P}$, which is further transformed if inconsistency detection is required (in the presence of certain DL constructs in the ontology, such as DisjointWith axioms).

The result is used as input for the top-down query engine XSB Prolog which realizes the well-founded semantics for logic programs [11], and the transfer to XSB is realized

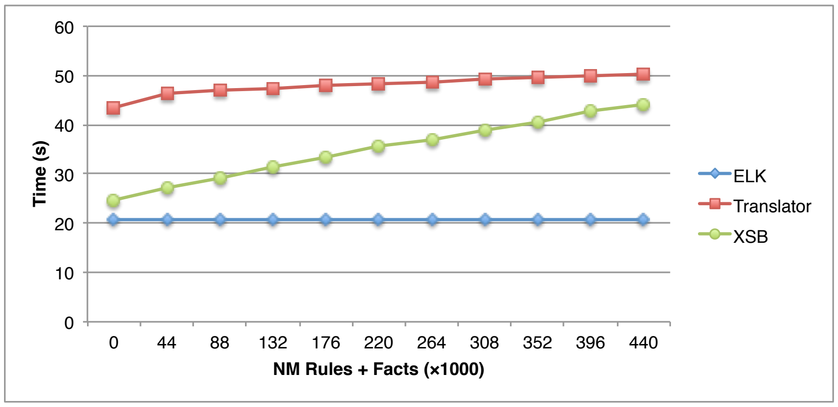

**Fig. 2.** Preprocessing time for SNOMED with a varying number of rules

via InterProlog,[8] which is an open-source Java front-end allowing the communication between Java and a Prolog engine.

Next, the query is 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). XSB itself 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 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.

## 4 Evaluation

Tests on the $\mathcal{EL}_{\perp}^{+}$ component alone already have shown that a) different $\mathcal{EL}$ ontologies can be preprocessed for querying in a short period of time (around one minute for SNOMED CT with over 300,000 concepts), b) adding rules increases the time of the translation only linearly, and c) querying time is in general neglectable, in comparison to a) and b) [14]. Fig. 2 visualizes the results from b) showing that the times for processing of files and ELK is basically independent of the number of added rules, and that the time of translator and XSB only grows linearly on the number of rules, with a small degree.

In subsequent tests on improved versions of both components (for $\mathcal{EL}_{\perp}^{+}$ and $DL\text{-}Lite_R$) [9], we have shown that i) our system scales reasonably well for OWL QL query answering without non-monotonic rules (only slowing down for memory-intensive cases), ii) preprocessing is even faster when compared to NoHR's previous version using a classifier (for EL), which was already capable of preprocessing large ontologies in a short

[8] http://interprolog.com/java-bridge/

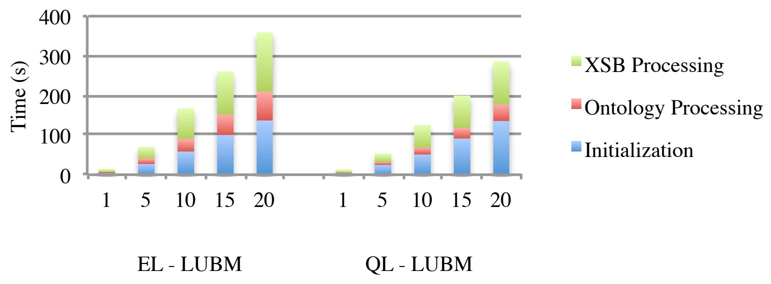

**Fig. 3.** Preprocessing time for LUBM for the two translation modes

period of time, iii) querying scales well, even for over a million facts/assertions in the ABox, despite being slightly slower on average in comparison to EL, and iv) adding rules scales linearly for pre-processing and querying, even for an ontology with many negative inclusions (for $DL\text{-}Lite_R$).

Fig. 3 shows the results for ii) where we considered LUBM[9] [12], a standard benchmark for evaluating queries over a large data set, which also includes a given set of standard queries. We created instances of LUBM$_n$ with $n = 1, 5, 10, 15, 20$ using the provided generator, and a restricted version of LUBM which fits both OWL EL and QL (thus only rendering a few of the standard queries meaningless), with the number of assertions ranging from roughly 100,000 to over 2,700,000. Note that "Initialization" includes loading the ontology and for EL also classifying it, "Ontology Processing" includes the actual translation, and "XSB Processing" the writing of the rule file and loading it in XSB. We observe that QL is considerably faster, indeed up to 80s for LUBM$_{20}$, which is to a considerable extent due to avoiding classification and a smaller rule file being created. This is compensated when querying as the $DL\text{-}Lite_R$ approach is slightly slower on average, and it thus seems that deciding which of the two forms of translation performs better depends on the kind (and number) of queries we pose.

## 5 Conclusions

The Protégé plugin NoHR – also distributed as an API – affords us the possibility to query knowledge bases composed of both an ontology in OWL 2 EL or QL and a set of non-monotonic rules, using a top-down reasoning approach, which means that only the part of the ontology and rules that is relevant for the query is actually evaluated. Its sound theoretical foundation together with the fast interactive response times make NoHR a truly one-of-a-kind reasoner.

Extending NoHR to the third profile – OWL 2 RL – seems an obvious next step, although developing an alternative for OWL 2 QL using the classifier integrated in ontop [20] or even the general reasoner Konclude [27], could shed more light on whether

[9] http://swat.cse.lehigh.edu/projects/lubm/

classification or direct translation fares better for proper OWL 2 QL ontologies. The efficiency of the latter reasoner also motivates looking into non-polynomial DLs, with possible influences from recent work on rewriting disjunctive datalog programs [15]. Adjusting NoHR to the paraconsistent semantics for MKNF knowledge bases of [16] would provide better support to the already observed paraconsistent behavior.

**Acknowledgments.** We would like to acknowledge the valuable contribution of both Nuno Costa and Vadim Ivanov to the development of NoHR. This work was partially supported by Fundação para a Ciência e a Tecnologia (FCT) under strategic project PEst/UID/CEC/04516/2013, and grant SFRH/BPD/86970/2012 (M. Knorr).

## References

1. Alberti, M., Knorr, M., Gomes, A.S., Leite, J., Gonçalves, R., Slota, M.: Normative systems require hybrid knowledge bases. In: Procs. of AAMAS. pp. 1425–1426. IFAAMAS (2012)
2. Alferes, J.J., Knorr, M., Swift, T.: Query-driven procedures for hybrid MKNF knowledge bases. ACM Trans. Comput. Log. 14(2), 1–43 (2013)
3. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The $DL\text{-}Lite$ family and relations. J. Artif. Intell. Res. (JAIR) 36, 1–69 (2009)
4. 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)
5. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Kaelbling, L.P., Saffiotti, A. (eds.) Procs. of IJCAI. pp. 364–369. Professional Book Center (2005)
6. Baral, C., Gelfond, M.: Logic programming and knowledge representation. J. Log. Program. 19/20, 73–148 (1994)
7. Calvanese, D., de Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The $DL\text{-}Lite$ family. Journal of Automated Reasoning 39(3), 385–429 (2007)
8. Chen, W., Warren, D.S.: Tabled Evaluation with Delaying for General Logic Programs. J. ACM 43(1), 20–74 (1996)
9. Costa, N., Knorr, M., Leite, J.: Next step for NoHR: OWL 2 QL. In: Arenas, M., Corcho, O., et al. (eds.) Procs. of ISWC. LNCS, vol. 9366, pp. 569–586. Springer (2015)
10. Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R., Tompits, H.: Combining answer set programming with description logics for the semantic web. Artif. Intell. 172(12-13), 1495–1539 (2008)
11. Gelder, A.V., Ross, K.A., Schlipf, J.S.: The well-founded semantics for general logic programs. J. ACM 38(3), 620–650 (1991)
12. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmark for OWL knowledge base systems. J. Web Sem. 3(2-3), 158–182 (2005)
13. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S. (eds.): OWL 2 Web Ontology Language: Primer (Second Edition). W3C Recommendation 11 December 2012 (2012), available from http://www.w3.org/TR/owl2-primer/
14. Ivanov, V., Knorr, M., Leite, J.: A query tool for $\mathcal{EL}$ with non-monotonic rules. In: Alani, H., et al. (eds.) Procs. of ISWC. LNCS, vol. 8218, pp. 216–231. Springer (2013)
15. Kaminski, M., Nenov, Y., Grau, B.C.: Datalog rewritability of disjunctive datalog programs and its applications to ontology reasoning. In: Brodley, C.E., Stone, P. (eds.) Procs. of AAAI. pp. 1077–1083. AAAI Press (2014)

16. Kaminski, T., Knorr, M., Leite, J.: Efficient paraconsistent reasoning with ontologies and rules. In: Yang, Q., Wooldridge, M. (eds.) Procs. of IJCAI. IJCAI/AAAI (2015)
17. Kazakov, Y., Krötzsch, M., Simančík, F.: The incredible ELK: From polynomial procedures to efficient reasoning with $\mathcal{EL}$ ontologies. Journal of Automated Reasoning 53, 1–61 (2013)
18. Kifer, M., Boley, H. (eds.): RIF Overview (Second Edition). W3C Working Group Note 5 February 2013 (2013), available at http://www.w3.org/TR/rif-overview/
19. 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)
20. Kontchakov, R., Rezk, M., Rodriguez-Muro, M., Xiao, G., Zakharyaschev, M.: Answering SPARQL queries over databases under OWL 2 QL entailment regime. In: Mika, P., et al. (eds.) Procs. of ISWC. LNCS, vol. 8796, pp. 552–567. Springer (2014)
21. Lifschitz, V.: Nonmonotonic databases and epistemic queries. In: Mylopoulos, J., Reiter, R. (eds.) Procs. of IJCAI. pp. 381–386. Morgan Kaufmann (1991)
22. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C. (eds.): OWL 2 Web Ontology Language: Profiles (Second Edition). W3C Recommendation 11 December 2012 (2012), available at http://www.w3.org/TR/owl2-profiles/
23. Motik, B., Rosati, R.: Reconciling description logics and rules. J. ACM 57(5), 93–154 (2010)
24. Patel, C., Cimino, J.J., Dolby, J., Fokoue, A., Kalyanpur, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: Matching patient records to clinical trials using ontologies. In: Aberer, K., et al. (eds.) Procs. of ISWC. LNCS, vol. 4825, pp. 816–829. Springer (2007)
25. Slota, M., Leite, J., Swift, T.: Splitting and updating hybrid knowledge bases. TPLP 11(4-5), 801–819 (2011)
26. Slota, M., Leite, J., Swift, T.: On updates of hybrid knowledge bases composed of ontologies and rules. Artif. Intell. 229, 33–104 (2015)
27. Steigmiller, A., Liebig, T., Glimm, B.: Konclude: System description. J. Web Sem. 27, 78–85 (2014)