Reasoning With Bounded Self-Reference Using Logical Interpreters

Daniel Gorín, Lutz Schröder, and Thorsten Wißmann

Dept. of Comput. Sci., Friedrich-Alexander-Universität Erlangen-Nürnberg

Abstract. Self-referential concepts in description logic may formally be viewed as concepts that at some point mention an individual previously encountered in the evaluation; this corresponds to a restricted form of the down-arrow binder known from hybrid logic. Typical examples include being a *narcissist*, (someone who loves himself); a *celebrity* (someone who is known by everyone she meets) or a *grant application reviewer* (who can only review proposals of researchers with whom she did not collaborate in the past). While reasoning in $\mathcal{ALC}$ plus self-reference is known to be undecidable, it has recently been shown that one can often safely restrict self-reference to at most two indirections (allowing the definition of narcissists and celebrities, but not of grant reviewers). In particular, the extension of $\mathcal{ALCQ}$ with this construct has an expressive power roughly equivalent to $\mathcal{ALCHIQ}$ plus safe Boolean combinations of roles. Here, we discuss a translation-based approach to reasoning in this logic. The translation goes into $\mathcal{ALCHIQ}$ (without Boolean roles) and therefore can be used in combination with off-the-shelf reasoners for OWL. The translation strategy encodes the semantics of the source axiomatically in the target logic. We use an implementation of this type of translations to evaluate the feasibility of the whole approach.

1 Introduction

Modern description logics reasoners accept definitions and queries in very expressive logical languages such as $\mathcal{SROIQ}$, which contains nominals, complex role inclusions, disjoint roles, universal roles, etc. among its features [5]. Yet, it is not unusual to run into settings that on the one hand make use of only some of the features available, and on the other hand require other features that are not supported. Moreover, the latter may be known not to interact well with other constructs, so it may not be clear whether a given reasoner can be extended easily or at all in order to add support for them. Role intersections is a typical example that comes to mind.

We suggest a translation-based methodology as a cost-effective alternative for such scenarios. Satisfiability-preserving translations are a common tool in computational logic, usually as a means of establishing complexity bounds (but not always, see e.g. [6,7,10] for practical examples); here, however, we want to go beyond satisfiability preservation. We look for translations taking a knowledge-base Γ in a source logic L to a knowledge-base Γ' in a target logic L' (for which reasoners exist) such that common queries on Γ including satisfiability, but also concept classification or instance retrieval, can be translated to analogous queries on Γ'. That is, modulo a thin layer of translations, we would like to use a reasoner for L' as if it were a reasoner for L.

More concretely, we employ what we shall call *logical interpreters*. Extending the example above, we represent complex L-concepts with atomic L'-concepts and encode the semantics of L (restricted to a finite set of L-concepts) using a set of L'-axioms, so that we can say that we are *interpreting* the semantics of L in L'. This is reminiscent of works such as [9], where modal formulas are expressed as ground first-order terms and modal axiomatic systems encoded as Π_1^0-formulas (then fed to an ATP); but notice that here i) we deal with semantics and not with provability, and ii) we expect L and L' to be "semantically close", so that the cost of interpretation is kept relatively low.

We showcase this technique by building a logical interpreter in $\mathcal{ALCHIQ}$ for the logic $\mathcal{ALCQ}\mathsf{me}_2$, the extension of $\mathcal{ALCQ}$ with Marx's I-me operators [8] under the restriction that there can be at most two qualified number restrictions between an I and the me it binds. Intuitively, an individual a is a member of the $\mathsf{I}.C$ concept whenever a is a member of C under the assumption that me denotes the singleton concept $\{a\}$. In $\mathcal{ALCQ}\mathsf{me}_2$ one can naturally define some self-referential concepts such as "a narcissist is someone who loves himself" ($\mathsf{Narcissist} \equiv \mathsf{I}.\exists \mathsf{loves}.\mathsf{me}$) or "a celebrity is a person that is known by everyone she meets" ($\mathsf{Celebrity} \equiv \mathsf{Person} \sqcap \mathsf{I}.\forall \mathsf{meet}.\exists \mathsf{knows}.\mathsf{me}$).

In $\mathcal{ALCQ}\mathsf{me}_2$ one can define inverses, role hierarchies and safe Boolean combination of roles (where full role complementation is replaced by role subtraction $R - S$). In fact it has been shown that $\mathcal{ALCQ}\mathsf{me}_2$ is expressively equivalent to the extension of $\mathcal{ALCHIQ}$ with safe Boolean combination of roles and the $\exists R.\mathsf{Self}$ operator of $\mathcal{SROIQ}$ (but $\mathcal{ALCQ}\mathsf{me}_2$ is conjectured exponentially more succinct) [4]. The good computational behavior of $\mathcal{ALCQ}\mathsf{me}_2$ does not transfer to other logics with bounded self-reference: already $\mathcal{ALC}\mathsf{me}_3$ has an undecidable satisfiability problem [3]. It is not yet known if nominals and/or transitive roles can be safely added to the mix.

We formally introduce $\mathcal{ALCQ}\mathsf{me}_2$ in Sec. 2. The general idea of logical interpreters is first explained in Sec. 3 by describing a trivial self-interpreter for $\mathcal{ALCHIQ}$. We then extend this construction to a logical interpreter for $\mathcal{ALCQ}\mathsf{me}_2$ in Sec. 4. Finally we present in Sec. 5 preliminary results on empirical evaluation of the approach.

2 Preliminaries

The DL $\mathcal{ALCQ}\mathsf{me}_2$ extends $\mathcal{ALCQ}$ with the I-me construct originally proposed in [8], in which me is a nominal that can be bound by I, just like the $\downarrow$-binder of hybrid logics would do [1], but under the restriction that me is never separated from its binding I by more than two qualified number restrictions. Formally, let $\langle \mathsf{N_C}, \mathsf{N_R} \rangle$ be a vocabulary where $\mathsf{N_C} = \{A_1, A_2, \ldots\}$ and $\mathsf{N_R} = \{R_1, R_2, \ldots\}$ are disjoint sets of *atomic* concepts and roles, respectively. The grammar for complex $\mathcal{ALC}\mathsf{me}_2$-concepts is given by C in:

$$\begin{aligned} C &:= \top \mid \bot \mid \mathsf{me} \mid \neg\mathsf{me} \mid A_i \mid \neg A_i \mid C \sqcap C \mid C \sqcup C \mid \mathsf{I}.Q_n R_i.\, C \mid Q_n R_i.\, D \\ D &:= \top \mid \bot \mid \mathsf{me} \mid \neg\mathsf{me} \mid A_i \mid \neg A_i \mid D \sqcap D \mid D \sqcup D \mid \mathsf{I}.Q_n R_i.\, C \end{aligned} \tag{1}$$

for $Q_n \in \{\geq_{n+1}, \leq_n\}$. Here, $A_i \in \mathsf{N_C}$ and $R_i \in \mathsf{N_R}$, and n is a non-negative integer. We will say that me occurs *free* in a concept C if it is not under the scope of an I. Notice that according to (1), $\geq_n R.\geq_m S.\mathsf{me}$ is not well-formed, even though $\mathsf{I}.\geq_n R.\geq_m S.\mathsf{me}$ is so. Hence, *we shall not consider* $Q_n R.C$ *a sub-concept of* $\mathsf{I}.Q_n R.C$. For convenience, we have defined concepts in *negation normal form* (NNF), i.e., negation only occurs in front of atoms. We denote with $\overline{C}$ the negation normal form of $\neg C$. For conciseness, we will sometimes employ standard abbreviations: $\exists R.C := \geq_1 R.C$, $\forall R.C := \leq_0 R.C$, $C \to D := C \sqcap D$ and $C \leftrightarrow D := (C \to D) \sqcap (D \to C)$.

An *interpretation* or *model* is an structure $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where $\Delta^{\mathcal{I}}$ is a non-empty set (the *domain* of $\mathcal{I}$); $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ for each $A \in \mathsf{N_C}$; and, for $R_i \in \mathsf{N_R}$, $R_i^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. For a set N_i of *individual names* and an individual $i \in N_i$, we also expect $i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Now, for every $a \in \Delta^{\mathcal{I}}$, we use $C_a^{\mathcal{I}}$ to denote the extension of an $\mathcal{ALCQ}\mathsf{me}_2$-concept C under the assumption that me stands for a. This is formally defined as:

$$
\begin{aligned}
A_a^{\mathcal{I}} &= A^{\mathcal{I}} & \mathsf{me}_a^{\mathcal{I}} &= \{a\} \\
(\neg C)_a^{\mathcal{I}} &= \Delta^{\mathcal{I}} - C_a^{\mathcal{I}} & (C \sqcap D)_a^{\mathcal{I}} &= C_a^{\mathcal{I}} \cap D_a^{\mathcal{I}} \\
(\mathsf{I}.{\geq_n} R.C)_a^{\mathcal{I}} &= \{b : b \in ({\geq_n} R.C)_b^{\mathcal{I}}\} & ({\geq_n} R.C)_a^{\mathcal{I}} &= \{b : |R^{\mathcal{I}}(b) \cap C_a^{\mathcal{I}}| \geq n\}
\end{aligned}
$$

A concept C is said *closed* if it has no free occurrences of me. It is easy to prove that:

Proposition 1. *If C is a closed concept, then $C_a^{\mathcal{I}} = C_b^{\mathcal{I}}$ for all $a, b \in \Delta^{\mathcal{I}}$.*

We thus denote the extension of a closed concept C just by $C^{\mathcal{I}}$. From Proposition 1 it is clear that $\mathcal{ALCQ}\mathsf{me}_2$ is a conservative extension of $\mathcal{ALCQ}$.

An $\mathcal{ALCQ}\mathsf{me}_2$ knowledge-base (KB) $\mathcal{T}$ is a pair $\langle \mathsf{T}, \mathsf{A} \rangle$ —a TBox and an ABox— with the provision that *every concept occurring in it must be closed*. Formally, T is a set of concept equivalence axioms of the form $C \equiv D$ and A is a set of assertions of the form $R_i(a, b)$ or $a : C$ with $a, b \in N_i$ (individual names). We say that $\mathcal{I}$ *satisfies* $\mathcal{T}$, notation $\mathcal{I} \models \mathcal{T}$, whenever: i) $C^{\mathcal{I}} = D^{\mathcal{I}}$ for all $C \equiv D \in \mathsf{T}$; ii) $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all $a : C \in \mathsf{A}$. iii) $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R_i^{\mathcal{I}}$ for all $R(a, b) \in \mathsf{A}$. We say that $\mathcal{T}$ is *consistent* if it is satisfied by some model. We may also write axioms of the form $C \sqsubseteq D$ in TBoxes, which are short for $C \sqcup A \equiv D$, for some fresh atomic concept A.

We assume the reader to be familiar with the DL $\mathcal{ALCHIQ}$ (see e.g. [2]). We shall employ R_i^- as the role constructor for *inverses* and assume $\mathcal{ALCHIQ}$-KB to contain also an *RBox* R, i.e., a set of role-inclusion axioms of the form $R \sqsubseteq S$.

3 Logical (self-)interpreters

We now introduce the notion of *logical interpreters* via a toy example: a self-interpreter for $\mathcal{ALCHIQ}$. This will be later turned into an $\mathcal{ALCHIQ}$-interpreter for $\mathcal{ALCQ}\mathsf{me}_2$.

When discussing logical interpreters, one needs to clearly distinguish the language to be interpreted from the language in which the interpreting axioms are written; we shall refer to the latter as the *host* language. In the example of this section, $\mathcal{ALCHIQ}$ is both the host and the interpreted language. The main idea is then simple to describe: we represent each concept of the interpreted language as an atomic concept of the host language, called the *host representation*, and describe its semantics by suitable axioms.

To illustrate this, consider a concept $\geq_5 R.C$ of the interpreted language. We associate to it an atomic concept $H_{\geq_5 R.C}$ and, if the host language includes qualified number restrictions among its features, we can describe its meaning with the TBox axiom $H_{\geq_5 R.C} \equiv \geq_5 R.H_C$, where H_C is the host representation of the concept C, which in turn needs to be suitably axiomatized. As this example already suggests, in order to axiomatize the semantics of a concept C, we need to provide axiomatizations for the host representations of the subconcepts of C (this, of course, is a consequence of the fact that Tarski-style semantics are compositional in nature).

Formally, assume then for the rest of the section that the interpreted language is $\mathcal{ALCHIQ}$ over a vocabulary $\langle N_C^i, N_R^i\rangle$. For a finite set Σ of concepts of this language, we let $\langle N_C^h(\Sigma), N_R^h(\Sigma)\rangle$ denote the vocabulary of $\mathcal{ALCHIQ}$ as host language, and put:

$$N_R^h(\Sigma) := N_R^i \qquad N_C^h(\Sigma) := \{H_C \mid C \text{ or } \overline{C} \text{ are subconcepts of a } D \in \Sigma\}.$$

For convenience we shall usually identify $A \in N_C^i$ with H_A and thus assume that $N_C^i \subseteq N_C^h(\Sigma)$. We shall also identify $H_\top$ with $\top$ and $H_\bot$ with $\bot$. We denote by $S_0(\Sigma)$ the set of axioms used to interpret the concepts in $N_C^h(\Sigma)$; $S_0(\Sigma)$ contains axioms

$$\begin{array}{lll} H_{C\sqcup D} \equiv H_C \sqcup H_D & H_{C\sqcap D} \equiv H_C \sqcap H_D & H_{\neg A} \equiv \neg H_A \\ H_{\geq_n R.C} \equiv \geq_n R.H_C & H_{\leq_n R.C} \equiv \leq_n R.H_C & \end{array}$$

whenever the index of the left hand side is in Σ. Notation involving Σ is extended to a KB $\mathcal{T}$ by considering the set of all subconcepts of concepts occurring in $\mathcal{T}$.

Let $\mathcal{T} = \langle \mathsf{T}^i, \mathsf{A}^i, \mathsf{R}^i\rangle$ be a KB to be interpreted. Our logical interpreter maps $\mathcal{T}$ to the KB $I_0(\mathcal{T}) = \langle \mathsf{T}^h, \mathsf{A}^h, \mathsf{R}^h\rangle$, where $\mathsf{T}^h := S_0(\mathcal{T}) \cup \{H_C \equiv H_D \mid C \equiv D \in \mathsf{T}^i\}$, $\mathsf{A}^h := \{a : H_C \mid a : C \in \mathsf{A}^i\}$ and $\mathsf{R}^h := \mathsf{R}^i$.

One can now make the relation between a KB $\mathcal{T}$ and its interpretation $I_0(\mathcal{T})$ precise. It suffices to observe how models for one are to be converted into models for the other, and vice versa. In general, we need to define two functions on models: $\mathcal{I}^\uparrow$ taking a model of the interpreted language to one in the host language, and $\mathcal{I}^\downarrow$ in the converse direction. In the case of the self-interpreter under consideration, we have that $\mathcal{I}^\downarrow$ is the restriction of $\mathcal{I}$ to the signature $N_C^i \subseteq N_C^h(\Sigma)$, while $\mathcal{I}^\uparrow$ interprets each $H_C \in N_C^h(\Sigma) - N_C^i$ as ${H_C}^{\mathcal{I}^\uparrow} := C^{\mathcal{I}}$. It is then straightforward to show the following:

Proposition 2. *For all $\mathcal{T}$, $\mathcal{I}$ and $\mathcal{J}$, we have that $\mathcal{I} \models \mathcal{T} \iff \mathcal{I}^\uparrow \models I_0(\mathcal{T})$ and $\mathcal{J}^\downarrow \models \mathcal{T} \iff \mathcal{J} \models I_0(\mathcal{T})$.*

This proposition shows that I_0 is satisfiability-preserving in a strong sense; in particular, individuals mentioned in the ABox are preserved both by $\mathcal{I}^\uparrow$ and $\mathcal{J}^\downarrow$, so that it is safe to map even ABox-queries on $\mathcal{T}$ to analogous queries in $I_0(\mathcal{T}')$, where $\mathcal{T}'$ is a suitable extension of $\mathcal{T}$. For example, for any given a concept C, it is easy to see that the instance retrieval problem for C on $\mathcal{T} = \langle \mathsf{T}, \mathsf{A}, \mathsf{R}\rangle$ corresponds to the instance retrieval problem for a fresh atomic concept A_0 on $I_0(\mathcal{T}')$ where $\mathcal{T}' = \langle \mathsf{T} \cup \{A_0 \equiv C\}, \mathsf{A}, \mathsf{R}\rangle$.

Restricting the interpreter to tree-models. The interpreter for $\mathcal{ALCQ}\mathsf{me}_2$ to be defined in Sec. 4 exploits heavily a tree-like model property of $\mathcal{ALCQ}\mathsf{me}_2$. This means that we will want to restrict our attention to models of the host language that are *trees* (or rather forests). We now show how to enforce this, and discuss some associated pitfalls.

Definition 3. We say that a binary relation $R \subseteq X^2$ is a *tree* whenever there exists $r \in X$, the *root* of R, such that there is exactly one directed path in (X, R) from r to any $x \in X$. We say that R is a *forest* if it is isomorphic to a disjoint union of trees. We say that $\mathcal{I}$ is a *tree model* whenever $U = \bigcup_R R^{\mathcal{I}}$ is a forest.

Observation 4. If R is such that R^{-1} is a partial function and $R^{-1}(r)$ is undefined, r is the root of a tree obtained by restricting R to the connected component of r in R.

The observation serves as a guiding principle in refining I_0. For $\mathcal{T} = \langle \mathsf{T}^i, \emptyset, \mathsf{R}^i \rangle$, define $I_0^t(\mathcal{T}) := \langle \mathsf{T}^{ht}, \mathsf{A}_\emptyset^{ht}, \mathsf{R}^{ht} \rangle$, where for a fresh role symbol f and a fresh individual $i_\emptyset$, $\mathsf{T}^{ht} := \mathsf{T}^h \cup \{\top \sqsubseteq \leq_1 f.\top\}$, $\mathsf{A}_\emptyset^{ht} := \{i_\emptyset : \forall f.\bot\}$ and $\mathsf{R}^{ht} := \mathsf{R}^h \cup \bigcup_{R \in N_R^i} \{R^- \sqsubseteq f\}$.

Clearly, any model $\mathcal{I}$ for $I_0^t(\mathcal{T})$ is turned into a tree model $T(\mathcal{I})$ for $I_0^t(\mathcal{T})$ by restricting it to only the connected components of elements satisfying $\forall f.\bot$, like $i_\emptyset{}^{\mathcal{I}}$. In the presence of non-empty ABoxes we have to be more careful since it is easy to have one that forces non-tree models. We will consider this when interpreting $\mathcal{ALCQ}\mathsf{me}_2$.

4 An Interpretation of $\mathcal{ALCQ}\mathsf{me}_2$ in $\mathcal{ALCHIQ}$

Self-interpretation as was done in Sec. 3 is an extreme case: since the host and interpreted languages coincide, the interpreter is trivial. We are now faced with the problem of interpreting in $\mathcal{ALCHIQ}$ a radically different language, namely, $\mathcal{ALCQ}\mathsf{me}_2$. We shall overcome here several challenges.

First and foremost, there is a radical mismatch between the semantics of $\mathcal{ALCQ}\mathsf{me}_2$ and that of $\mathcal{ALCHIQ}$: the extension of concepts in the latter corresponds to subsets of the domain, while in the former they induce a binary relation (i.e., we need two points of evaluation: the *current* one and that denoted by me). To circumvent this problem, we observe that $\mathcal{ALCQ}\mathsf{me}_2$ possesses a tree-like model property (cf. Theorem 6 below); i.e., for purposes of reasoning we can in general restrict our attention to tree-like models. Now, it turns out that if we intend to decide whether $x \in C_y^{\mathcal{I}}$ on a tree-like model $\mathcal{I}$ where the distance between x and y is at most 1 (i.e., they either coincide or one is the father of the other), we will only need to consider subproblems of the form $z \in D_w^{\mathcal{I}}$, where the distance between z and w is 1 as well (for closed concepts, in particular, we can assume distance 0). In the end, there is a finite combination of relevant cases that we will be able to encode in the host representation.

Definition 5. A relation $S \subseteq X^2$ is a quasi-tree with root r if there is a tree R with root r and $R \subseteq S \subseteq R \cup R^{-1} \cup \mathrm{Id}_X$. Such R is unique and called the *underlying tree* of S.

Terms *quasi-forest* and *quasi-tree model* are used in analogy to Def. 3. The following is proved with an unraveling similar to the one in the analogous proof for $\mathcal{ALC}\mathsf{me}_2$ [3].

Theorem 6 (Quasi-tree model property). *If C is a satisfiable closed $\mathcal{ALCQ}\mathsf{me}_2$-concept, then there exists a quasi-tree model that satisfies C at its root.*

We have seen in the previous section how to enforce tree models in a $\mathcal{ALCHIQ}$-KB with an empty ABox. We use f throughout this section to denote the "father-of" relation (partial function) of this encoding. In a quasi-tree, one can also have self-loops and links going back to the father of a node. We can simulate them in the interpreter by adding atomic concepts $\circlearrowleft_R$ and $\uparrow_R$, for each $R \in N_R^i$, to the host language; the former represents a self-loop for role R, the latter an up-link. In particular, $\uparrow_R$ cannot hold at a root, so $\uparrow_R \sqsubseteq \exists f.\top$ will have to be an axiom of the interpreter.

In the self-interpreter of Sec. 3, a concept C of the interpreted language was mapped to the atomic concept H_C of the host language. We could think of H_C as meaning "satisfies concept C in the interpreted language". In order to account for the extension of me we will need a richer set of atomic concepts in the host language. We will use atomic concept $H_{**:C}$ to describe "those elements that satisfy C in the interpreted language when they are also the value of me". Similarly, $H_{f*:C}$ shall represent "those elements that satisfy C in the interpreted language when their father is the value of me". It will be crucial to consider a third family, namely $H_{*f:C}$, which is to be read as "those elements *whose father* satisfies C under the assumption that they are the value of me". Although dispensable, we find it conceptually clearer to include an additional family, $H_{*:C}$, where C is a closed concept, for "those elements that satisfy C regardless the value of me". We refer to $L := \{*, **, f*, *f\}$ as the set of possible contexts.

For the case where C has $\sqcap$ or $\sqcup$ as top-level operator, we can mimic the corresponding axioms from Sec. 3. For atomic negation we observe some differences in contexts involving the father: by definition, elements satisfying $H_{f*:\neg A}$ or $H_{*f:\neg A}$ must have a father, so the corresponding axioms need to be $H_{*f:\neg A} \equiv \exists f.H_{*:\neg A}$ and $H_{f*:\neg A} \equiv H_{*:\neg A} \sqcap \exists f.\top$. In fact, we will generalize these axioms to any closed concept (not just negated atoms) and also have $H_{**:C} \equiv H_{*:C}$ when C is closed.

The last missing ingredient of the interpreter is the handling of concepts of the form $(\mathsf{I}.)\geq_n R.C$ and $(\mathsf{I}.)\leq_n R.C$, that is, those whose top-level operator is a qualified number restriction or an I binder in front of one. This requires some effort since we need to take into account the "virtual" part of roles (i.e., the $\uparrow_R$ and $\circlearrowleft_R$) in the counting. The general idea in handling this is to interpret $H_{\ell:\geq_n R.C}$ by way of a case-analysis: depending on which of $\circlearrowleft_R$ and $\uparrow_R$ hold, and whether C holds at the current state and/or at the father, we pick a suitable discount $0 \leq d \leq 2$ and then test if $\geq_{n-d} R.C$ holds. For example, a discount value of 1 indicates that either the current node or its father (but not both) is one of the n successors required, so it suffices to find $n-1$ more. The axioms for contexts other than $*f$ are given in terms of the following helper construct:

$$\xi_R^{\geq_n}(C_1, C_2, C_3) := \bigsqcap \begin{pmatrix} (\uparrow_R \sqcap C_1) \sqcap \quad (\circlearrowleft_R \sqcap C_2) \rightarrow \geq_{n-2} R.C_3 \\ (\uparrow_R \sqcap C_1) \leftrightarrow \neg(\circlearrowleft_R \sqcap C_2) \rightarrow \geq_{n-1} R.C_3 \\ \neg(\uparrow_R \sqcap C_1) \sqcap \neg(\circlearrowleft_R \sqcap C_2) \rightarrow \geq_{n-0} R.C_3 \end{pmatrix} \qquad (2)$$

$\xi_R^{\leq_n}$ is defined using $\geq$ instead of $\leq$. Observe that whenever $m > n$, then $\geq_{n-m} R.C_3$ (resp. $\leq_{n-m} R.C_3$) shall represent $\top$ (resp. $\bot$). The intuition is that $\xi_R^{\geq_n}(C_1, C_2, C_3)$ is discounting 2 when both $\circlearrowleft_R$ and $\uparrow_R$ hold and C_1, a suitable modification of C, holds "at the father" while C_2, another suitable modification of C, holds at the current node; it discounts 1 when only one of these conditions hold; and nothing otherwise.

Now, why do we need concepts C_1, C_2 and C_3 in (2) instead of just one concept C? There are two reasons for this: i) adjusting the context and ii) dealing with free occurrences of me in C. This will become clear with an example. Assume we need to interpret $H_{*:\mathsf{I}.\geq_n R.C}$. In order to see if the father node (if any) needs to be discounted from the n successors, we need to know whether C holds at the father *under the assumption that the current node stands for* me. So, in principle, we would like to use $H_{*f:C}$ for C_1. But we know more: if we are evaluating this formula on a tree, then the current node and its father will be distinct nodes, so evaluating C at the father will be equivalent to evaluating C' at the father, where C' is obtained from replacing every top-level occurrence of me in C by $\bot$. Observe that the fact that no me appearing under the scope of a qualified number restriction is replaced by $\bot$ is critical for the correctness of the argument. The following piece of notation will be handy:

Definition 7. For concepts C and D, $C(D)$ is the concept obtained by replacing every free occurrence of me *outside the scope of a qualified number restriction* in C by D.

Now we can formalize the axioms for $H_{*:(\mathsf{l}.)\geq_n R.C}$, $H_{**:\geq_n R.C}$ and $H_{f*:\geq_n R.C}$:

$$H_{*:(\mathsf{l}.)\geq_n R.C} \equiv \xi_R^{\geq_n}(H_{*f:C(\bot)}, H_{**:C(\top)}, H_{f*:C(\bot)}) \tag{3}$$

$$H_{**:\geq_n R.C} \equiv \xi_R^{\geq_n}(\exists f.H_{*:C(\bot)}, H_{*:C(\top)}, H_{*:C(\bot)}) \tag{4}$$

$$H_{f*:\geq_n R.C} \equiv \xi_R^{\geq_n}(\exists f.H_{*:C(\top)}, H_{*:C(\bot)}, H_{*:C(\bot)}) \sqcap \exists f.\top \tag{5}$$

Three things are worth observing. First, notice the relation between the context of the interpreted concept and the assignments of $\top$ and $\bot$. Second, (3) is the first axiom we discussed so far that contains a concept with context other than $*$ that does not occur at the left of the $\equiv$. In fact, this and the dual axiom for $(\mathsf{l}.)\leq_n R.C$ concepts will be the only such axioms. From this observation and because in $\mathcal{ALCQ}\mathsf{me}_2$ there are at most two qualified number restrictions between l and me, it is clear that C in (4) and (5) can be assumed to have all occurrences of me at the top-level, so that $C(\bot)$ and $C(\top)$ are closed concepts. Finally, notice that unlike the self-interpreter of Sec. 3, the set of relevant concepts needs to be closed by replacements of top-level me by $\top$ and $\bot$ (recall, also, from Sec. 2, that $\geq_n R.C$ as not a subconcept of $\mathsf{l}.\geq_n R.C$).

The only kind of axiom we are missing is that for $H_{*f:\geq_n R.C}$. Because $\geq_n R.C$ is expected to hold at the father of the current node, we will need to use $\exists f.\xi^{\geq_{n+c}}(\ldots)$, which will take care of the father's virtual relations, if any, picking also a correction factor $-1 \leq c \leq 1$ that will compensate for "interferences" of the current node in the counting. Formally, the required axiom is $H_{*f:\geq_n R.C} \equiv \vartheta_R^{\geq_n}(C) \sqcap \exists f.\top$, where:

$$\vartheta_R^{\geq_n}(C) := \bigsqcap \begin{pmatrix} (\exists R^-.\top \sqcap \quad (H_{*:C(\top)} \sqcap \neg H_{*:C(\bot)})) \rightarrow \exists f.\xi_{\uparrow R}^{\geq_{n-1}}(C) \\ (\neg\exists R^-.\top \sqcup \quad (H_{*:C(\top)} \leftrightarrow \quad H_{*:C(\bot)})) \rightarrow \exists f.\xi_{\uparrow R}^{\geq_{n+0}}(C) \\ (\exists R^-.\top \sqcap (\neg H_{*:C(\top)} \sqcap \quad H_{*:C(\bot)})) \rightarrow \exists f.\xi_{\uparrow R}^{\geq_{n+1}}(C) \end{pmatrix} \tag{6}$$

Here $\xi_{\uparrow R}^{\geq_n}(C)$ stands for $\xi_R^{\geq_n}(\exists f.H_{*:C(\bot)}, H_{*:C(\bot)}, H_{*:C(\bot)})$. Firstly, it is helpful to compare the latter with the ξ-expressions of (4) and (5) and observe that me is not being replaced by $\top$ anywhere. In particular, if the current state satisfies $H_{*:C(\bot)}$ it may get counted when it should not be. More precisely, in order to have a non-zero correction factor, we need the current state to be an R-successor of its father. If it additionally satisfies $H_{*:C(\top)}$ then we need to discount it from the expected R-successors of the father satisfying C ($c = 1$), except when $H_{*:C(\bot)}$ holds as well, since then it will be incorrectly be counted as one of the R-successors of its father that satisfies C under the assumption that me is *not* current node ($c = 0$). The case where $c = -1$ is analogous.

To formalize the whole procedure, we assume the vocabulary of $\mathcal{ALCQ}\mathsf{me}_2$ to be $\langle N_C^i, N_R^i \rangle$ and fix a set of concepts Σ. Let $\Sigma' \supseteq \Sigma$ be the smallest set closed by

$H_{\ell:C\sqcup D} \equiv H_{\ell:C} \sqcup H_{\ell:D}$	$H_{\ell:C\sqcap D} \equiv H_{\ell:C} \sqcap H_{\ell:D}$
$H_{*:\neg A} \equiv \neg H_{*:A}$	$H_{*:(\mathsf{l}.)Q_nR.C} \equiv \xi_R^{Q_n}(H_{*f:C(\bot)}, H_{**:C(\top)}, H_{f*:C(\bot)})$
$H_{**:C^0} \equiv H_{*:C^0}$	$H_{**:Q_nR.C} \equiv \xi_R^{Q_n}(\exists f.H_{*:C(\bot)}, H_{*:C(\top)}, H_{*:C(\bot)})$
$H_{f*:C^0} \equiv H_{*:C^0} \sqcap \exists f.\top$	$H_{f*:Q_nR.C} \equiv \xi_R^{Q_n}(\exists f.H_{*:C(\top)}, H_{*:C(\bot)}, H_{*:C(\bot)}) \sqcap \exists f.\top$
$H_{*f:C^0} \equiv \exists f.H_{*:C^0}$	$H_{*f:Q_nR.C} \equiv \vartheta_R^{Q_n}(C) \sqcap \exists f.\top$

Table 1: $S_1(\Sigma)$ contains one of these axioms for each concept in $N_C^h(\Sigma)$. Notation: $\ell \in L$, C^0 is a *closed* concept but *neither a conjunction nor a disjunction*; $Q_n \in \{\geq_n, \leq_n\}$.

subconcepts and such that if $C \in \Sigma'$ then $C(\bot), C(\top) \in \Sigma'$ as well. We then have:

$$N_C^h(\Sigma) := \{H_{\ell:C} \mid \ell \in \{**, *f, *f\}, C \in \Sigma'\} \cup \{H_{*:C} \mid C \in \Sigma_r^s, C \text{ is closed}\} \ .$$

We identify $H_{*:\top}$ with $\top$, $H_{*:\bot}$ with $\bot$ and $H_{*:A}$ with A, for $A \in N_C^i$. We then define the atomic roles of the host language with $N_R^h(\Sigma) := N_R^i \cup \{f\} \cup \{\circlearrowleft_R, \uparrow_R \mid R \in N_R^i\}$. The set $S_1(\Sigma)$ of axioms needed to interpret relevant concepts in Σ is listed in Table 1. Again, we extend the notation from sets of concepts to knowledge-bases as expected.

For simplicity, we shall first discuss the interpretation of $\mathcal{ALCQ}\mathsf{me}_2$-KBs with empty ABoxes. So let $\mathcal{T} = \langle \mathsf{T}^i, \emptyset \rangle$ be such a knowledge-base. A logical interpreter for $\mathcal{T}$ is the $\mathcal{ALCHIQ}$-KB $I_1(\mathcal{T}) = \langle \mathsf{T}^h, \mathsf{A}_\emptyset^h, \mathsf{R}^h \rangle$ where $\mathsf{R}^h := \bigcup_{R \in N_R^i} \{\uparrow_R \sqsubseteq \exists f.\top\}$, $\mathsf{A}_\emptyset^h := \{i_\emptyset : \forall f.\bot\}$ and $\mathsf{T}^h := S_1(\mathcal{T}) \cup \{\top \sqsubseteq \leq_1 f.\top\} \cup \{\uparrow_R \sqsubseteq \exists f.\top \mid R \in N_R^i\}$.

We now look at how quasi-tree models for $\mathcal{ALCQ}\mathsf{me}_2$ are turned into tree models of the host logic and vice versa. We first take a quasi-tree model $\mathcal{I}$ and observe that since every rooted quasi-tree induces a *unique* tree, $\mathcal{I}$ induces some tree model $\mathcal{I}'$ (which only depends on the elements picked as roots). Then $\mathcal{I}'$ is extended to a model $\mathcal{I}^\uparrow$ in the signature of the host language by interpreting $\circlearrowleft_R$ and $\uparrow_R$ in the obvious way (e.g., ${\circlearrowleft_R}^{\mathcal{I}'} := \{x \mid (x, x) \in R^{\mathcal{I}}\}$); making $f^{\mathcal{I}'}$ the union of the converse of every role in $\mathcal{I}'$, and interpreting all the other concepts in the expected way, for instance, one defines $(H_{f*:C})^{\mathcal{I}^\uparrow} = \{x \mid \exists y : (x, y) \in f^{\mathcal{I}'} \text{ and } x \in C_y^{\mathcal{I}})\}$. Also, i_0 denotes some root of $\mathcal{I}^\uparrow$. On the other direction, for a model $\mathcal{J}$ of the host language we let $\mathcal{J}'$ be the restriction of the domain of $\mathcal{J}$ to all elements the connected components that include an "f-root", i.e., an element that has no f-predecessor (or $\mathcal{J}' = \mathcal{J}$ when no such element exist). We then define $\mathcal{J}^\downarrow$ as the restriction of $\mathcal{J}'$ to the vocabulary of the interpreted language, plus $R^{\mathcal{J}^\downarrow} := R^{\mathcal{J}'} \cup \{(x, x) \mid x \in \circlearrowleft_R^{\mathcal{J}'}\} \cup \{(x, y) \mid \exists y : (y, x) \in f^{\mathcal{J}'} \text{ and } x \in {\uparrow_R}^{\mathcal{J}'}\}$.

Proposition 8. *Fix an $\mathcal{ALCQ}\mathsf{me}_2$-KB $\mathcal{T}$. If $\mathcal{I}$ is a quasi-tree model, then $\mathcal{I} \models \mathcal{T} \iff \mathcal{I}^\uparrow \models I_1(\mathcal{T})$. If $\mathcal{J} \models I_1(\mathcal{T})$, then $\mathcal{J}^\downarrow$ is a quasi-tree model and $\mathcal{J}^\downarrow \models \mathcal{T}$.*

Example 9. Let $\mathcal{T} = \langle \{\top \equiv \mathsf{l}.\geq_2 R.\mathsf{me}\}, \emptyset \rangle$ and let $\mathcal{T}' = I_1(\mathcal{T})$. We want to verify that $\mathcal{T}'$ is unsatisfiable (we use $\models$ for the global entailment relation). Clearly, we have that $\mathcal{T}' \models H_{*:\mathsf{l}.\geq_2.\mathsf{me}}$, and because of (3), we get $\mathcal{T}' \models \xi_R^{\geq_2}(H_{*f:\bot}, H_{**:\top}, H_{f*:\bot})$ as well. Now, we know $\mathcal{T}' \models (H_{*f:\bot} \leftrightarrow \exists f.H_{*:\bot}) \sqcap (H_{**:\top} \leftrightarrow H_{*:\top})$ and $H_{*:\bot}$ and

$H_{*:\top}$ were identified with $\bot$ and $\top$, respectively. Hence, $\mathcal{T}' \models \xi_R^{\geq_2}(\bot, \top, H_{f*:\bot})$. Simplifying in (2), we conclude that $\mathcal{T}' \models (\circlearrowleft_R \rightarrow \geq_1 R.H_{f*:\bot}) \sqcap (\neg \circlearrowleft_R \rightarrow \geq_2 R.H_{f*:\bot})$. However, $\mathcal{T}' \models H_{f*:\bot} \leftrightarrow \bot \sqcap \exists f.\top$, so in particular $\mathcal{T}' \models H_{f*:\bot} \rightarrow \bot$. Therefore, we have $\mathcal{T}' \models (\circlearrowleft_R \rightarrow \geq_1 R.\bot) \sqcap (\neg \circlearrowleft_R \rightarrow \geq_2 R.\bot)$, so $\mathcal{T}'$ must be unsatisfiable.

We shall end this section by briefly discussing the handling of ABoxes. The first thing to observe is that for ABox reasoning there is no tree-like model property one can use: that is, it is trivial to build a consistent $\mathcal{ALC}$ ABox without tree models. That models in the host language are trees is a crucial part of the interpreter (it is what allows us to safely replace me by $\bot$ in the axiom for $H_{*f:\geq_n R.C}$), so we cannot let go this property. The upshot is: we can only handle ABoxes whose graph can be embedded in a quasi-forest.

We assume that the so called Unique Name Assumption (UNA) does not hold, that is, different individual names may denote the same individual (this corresponds with the OWL standard and, moreover, the case where the UNA holds is somewhat simpler). The following tells us how to eliminate all role assertions from such an ABox:

Proposition 10. *Let $\mathcal{T} = \langle \mathsf{T}, \mathsf{A} \rangle$ be an $\mathcal{ALCQ}\mathsf{me}_2$-KB and let I be the set of individual names in A. If the graph of A can be embedded in a quasi-tree, then for each $i \in I$ there is a C_i such that $\mathcal{T}$ is consistent iff $\mathcal{T} = \langle \mathsf{T}, \bigcup_{i \in I} \{C_i\} \rangle$ is consistent.*

Proof (sketch). Because individual names cannot occur in concepts we are free in principle to replace an assertion $R(i, j) \in \mathsf{A}$ by an assertion $R(i, j')$, where j' is a fresh individual that realizes the same formulas as j. Moreover, $\mathcal{ALCQ}\mathsf{me}_2$ is expressive enough to describe such an individual without needing a fresh individual name. We only need a fresh atomic concept in combination with two fresh role symbols U and $\overline{U}$.

Formally, fix $i \in I$ and let A_i be the ABox obtained by renaming every $j \in I - \{i\}$ by j_i. Moreover, let A_{j_i} denote a fresh concept for each such j. We assume i to be the root of A_i from which the underlying tree structure is uniquely determined. C_i is then the conjunction of i) all concepts C such that $i : C \in \mathsf{A}$; ii) $\mathsf{I}.R.\mathsf{me}$ for each $R(i, i) \in \mathsf{A}$; and iii) for each j_i that is an immediate successor of i in A_i we add $\leq_1 U.A_{j_i}$ plus:

$$\mathsf{I}.\exists U.(A_{j_i} \sqcap C_{j_i} \sqcap \exists S_1.\mathsf{me} \sqcap \ldots \exists S_n.\mathsf{me} \sqcap \exists \overline{U}.\mathsf{me} \sqcap \forall \overline{U}.\mathsf{me} \sqcap \mathsf{I}.\forall \overline{U}.(\exists R_1.\mathsf{me} \sqcap \ldots \exists R_m.\mathsf{me}))$$

where $R_1(i, j_i) \ldots R_m(i, j_i)$ and $S_1(j_i, i) \ldots S_m(j_i, i)$ are all such role assertions in A_i. For C_{j_i} we iterate conditions i-iii; the tree guarantees the recursion to terminate.

The proof of the left-to-right implication is then straightforward; for the converse direction one needs to reassign the interpreted individuals to make sure that those that were in the same connected component in A are in the same connected component of the model, the fact that A was a quasi-tree guarantees that this can be done.

5 Feasibility of Logical Interpreters: Preliminary Results

The interpreters we have considered are, of course, more complex than the knowledge-bases they interpret; yet, it is not hard to see that the blow-up (measured in number of axioms and assertions, and size of all complex concepts mentioned) is only linear – or $O(n \log n)$ if one considers the representation cost of fresh symbols. Although the translation is in principle small, it is not clear a priori how it is to interact in practice with off-the-shelve DL reasoners. On the other hand, these interpreters are not hard to implement, so it is worth experimenting. We report here some preliminary results.

We implemented a translation from $\mathcal{ALCQ}\mathsf{me}_2$ to $\mathcal{ALCHIQ}$[1] following the interpreter approach of Sect. 4. The translator takes OWL files (where we expect certain role-name and certain atomic concept to stand for I and me respectively) and outputs OWL files. This means that we can use them in combination with standard tools. We included only very basic optimizations in order to reduce the size of the interpreter, the most important one is simplifying complex concepts using Boolean identities such as $\bot \sqcup C \mapsto C$. This is important since these type of concepts are generated often when replacing top-level occurrences of me by $\top$ and $\bot$.

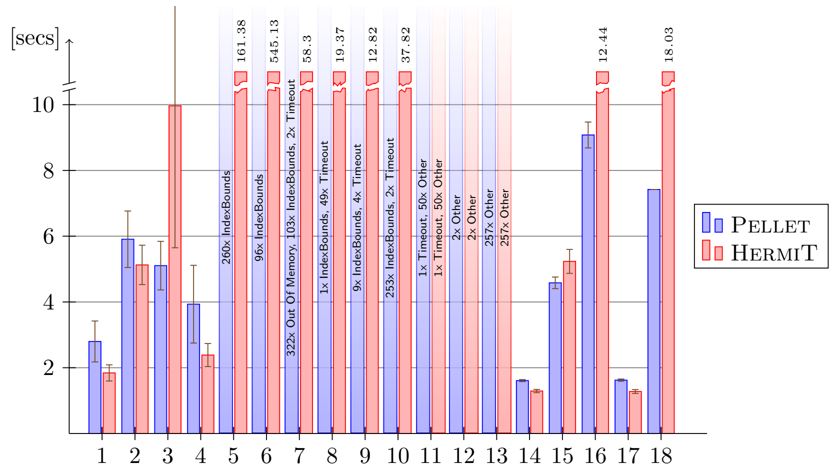

Fig. 1: Average runtime for the interpreter in seconds, with standard devation.

Since there are of course no real examples of $\mathcal{ALCQ}\mathsf{me}_2$ ontologies, we have to rely on artificial examples. We downloaded from the TONES ONTOLOGY REPOSITORY[2] ontologies that corresponded to fragments of $\mathcal{ALCHIQ}$ and turned them into $\mathcal{ALCQ}\mathsf{me}_2$ ontologies by interpreting some atomic concept A as me and, $\geq_n R$ as I, for some atomic role R. Care was put to delete role inclusion axioms mentioning R and ignoring occurrences of A that were free or not under at most two qualified number restrictions from the binding point. We actually did this conversion for up to 400 hundred combinations per ontology, which in most of the cases was exhaustive. Each of the resulting ontologies was translated to $\mathcal{ALCHIQ}$ using our tool and checked for consistency with PELLET [12] and HERMIT [11], versions 2.3.0 and 1.3.6.1054, respectively, on a LINUX box with a 2.4GHz INTEL XEON CPU and 24Gb RAM.

[1] Available at `http://sourceforge.net/p/imeowl`.
[2] Online resource: `http://owl.cs.manchester.ac.uk/repository/`.

We measured the time to required for each ontology (*usertime*) and, for those derived from the same TONES ontology, took the average time and the standard deviation (only for those that ended before the timeout of 600 seconds). The results are summarized in Fig. 1 (the raw data can be found on Table 4, in the Appendix).

We are including in Fig. 1 also ontologies which consistently failed either by out-of-memory errors, triggered what seems like an internal bug in PELLET ("ArrayIndexOutOfBoundsError") or simply could not be parsed by the reasoners, even though they were generated using the OWLAPI (indicated by "Other").

One can clearly see that most of the cases that could be handled end within a minute, regardless of the reasoner (although HERMIT seems to be able to handle more cases). In principle, the results are positive, although one needs to be cautious here: because of the artificial nature of the examples, it is possible that they are being solved simply because they are trivial (e.g., if there are no interactions between the chosen I and me).

Instead of performing a case-by-case analysis, we decided to conduct a quick experiment that could give us an idea of the cost of interpretation. We extended our tool to turn it into a very simple self-interpreter generator, trivially extending the one of Sect. 3 to $\mathcal{SHIQ}$. We downloaded from TONES all the ontologies for fragments of $\mathcal{SHIQ}$, which made for a total of 45 ontologies of diverse complexity. We had to discard 17 of them because our OWL API-based tool failed to parse them. Of the remaining 62%, we rejected three more ontologies since they required (undeclared) features beyond $\mathcal{SHIQ}$. For the 25 surviving ontologies, we ran a consistency check, both on the original and the translated ontologies and measured the runtime (same setup as before).

These results are summarized in Fig. 2 (the raw data can be found in Table 3 in the Appendix). We plotted, for each ontology, the quotient between the runtime for the translated and for the original version. Even though the sample is small, it is still noticeable that PELLET exhibits a relatively consistent behavior, staying in general within a factor of 2, which is more than acceptable. HERMIT on the other hand shows a more erratic behavior (sample 16 is actually way off the scale). At the moment we were unable to find any noticeable correlation between file size, number of axioms in the original ontology or language, and the performance of the reasoner on the interpreter.

6 Conclusions

We have described a translation-based approach to reasoning in the context of Description Logics, based on the idea of axiomatizing the semantics of the original logic – what we have called a *logical interpreter*. We illustrated the idea with a trivial example, a self-interpreter, and then constructed a non-trivial one for the case of $\mathcal{ALCQ}\mathsf{me}_2$: a language with a binder construct, capable to express features such as role-intersections. We also report preliminary results on the behavior of off-the-shelf DL reasoners on both on an interpreter for $\mathcal{ALCQ}\mathsf{me}_2$ and several self-interpreters. The results are suggestive that the logical-interpreter approach may be effective in some real scenarios. Building a logical interpreter is relatively cheap, so it appears as a cost-effective alternative.

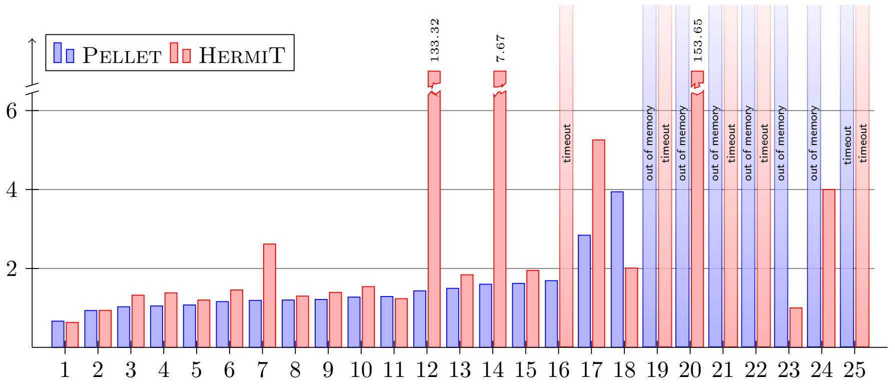

Fig. 2: Runtime for the interpreter divided by the runtime for the original KB.

References

1. Areces, C., ten Cate, B.: Hybrid logics. In: Blackburn, P., van Benthem, J., Wolter, F. (eds.) Handbook of Modal Logic, pp. 821–868. Elsevier (2007)
2. 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, 2nd edn. (2010)
3. Gorín, D., Schröder, L.: Narcissists are easy, stepmothers are hard. In: Foundations of Software Science and Computation Structures, FoSSaCS 2012. LNCS, vol. 7213, pp. 240–254. Springer (2012)
4. Gorín, D., Schröder, L.: Extending $\mathcal{ALCQ}$ with bounded self-reference. In: Advances in Modal Logic, AiML 2012. pp. 300–316. College Publications (2012)
5. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible SROIQ. In: Knowledge Representation and Reasoning, KR 2006. pp. 57–67. AAAI Press (2006)
6. Hustadt, U., Schmidt, R.A.: An empirical analysis of modal theorem provers. Journal of Applied Non-Classical Logics 9(4), 479–522 (1999)
7. Kazakov, Y., Motik, B.: A resolution-based decision procedure for $\mathcal{SHOIQ}$. Journal of Automated Reasoning 40(2–3), 89–116 (2008)
8. Marx, M.: Narcissists, stepmothers and spies. In: Proc. of the 2002 International Workshop on Description Logics (DL'02). CEUR, vol. 53 (2002)
9. Rabe, F., Pudlák, P., Sutcliffe, G., Shen, W.: Solving the \$100 modal logic challenge. Journal of Applied Logic 7(1), 113 – 130 (2009), special Issue: Empirically Successful Computerized Reasoning
10. Sebastiani, R., Vescovi, M.: Automated reasoning in modal and description logics via SAT encoding: the case study of $K_m/\mathcal{ALC}$-satisfiability. Journal of Artificial Intelligence Research 35, 343–389 (2009)
11. Shearer, R., Motik, B., Horrocks, I.: HermiT: A Highly-Efficient OWL Reasoner. In: Ruttenberg, A., Sattler, U., Dolbear, C. (eds.) Proc. of the 5th Int. Workshop on OWL: Experiences and Directions (OWLED 2008 EU). Karlsruhe, Germany (October 26–27 2008)
12. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Web Semantics: Science, Services and Agents on the World Wide Web 5(2), 51–53 (Jun 2007)

A Test data

From all ontologies which were considered because of their claimed expressitivity, the ontologies listed in Table 2 had to be discarded. The remaining ontologies are listed by Table 3 – together with the exact data measured.

Ontology	DL	Problem
action	$\mathcal{ALCHIN}$	Syntax
atom-complex-proton-2.0	$\mathcal{ALEQ}$	Syntax
biochemistry-complex	$\mathcal{ALC}$	Syntax
chemistry-complex	$\mathcal{ALCHQ}$	Syntax
expression	$\mathcal{ALCHI}$	Syntax
heart	$\mathcal{SHI}$	Syntax
legal-action	$\mathcal{ALC}$	Syntax
legal-role	$\mathcal{ALC}$	Syntax
lkif-rules	$\mathcal{ALCH}$	Syntax
mereology	$\mathcal{SHIN}$	Syntax
molecule-complex	$\mathcal{ALCH}$	Syntax
norm	$\mathcal{SHI}$	Syntax
process	$\mathcal{ALCHI}$	Syntax
relative-places	$\mathcal{SHIF}$	Syntax
role	$\mathcal{ALCI}$	Syntax
time	$\mathcal{SHI}$	Syntax
yowl-complex	$\mathcal{ALCH}$	Syntax
tambis-full	$\mathcal{SHIN}$	Nominals
tambis-patched	$\mathcal{SHIN}$	Nominals
biochemical-reaction-complex	$\mathcal{ALCR}$	SubObjectPropertyOf

Table 2: Ontologies that could not be tested. The problem can be a syntactical reason or of language features not supported by our implentation (e.g. nominals).

Nr.	Name	DL	Filesize		Axiom count		HERMIT user-time		PELLET user-time	
			before	after	before	after	before	after	before	after
1	atom-complex-disjoint.owl	$\mathcal{ALC}$	15363	85697	1	252	2.939	1.858	3.094	2.056
2	mob.owl	$\mathcal{ALI}$	4604	5379	10	14	0.843	0.79	1.384	1.291
3	download.owl	$\mathcal{ALCHQ}$	9458	22874	24	74	0.889	1.176	1.436	1.476
4	ribosome.owl	$\mathcal{ALCF}$	4356	24105	21	87	0.785	1.084	1.36	1.428
5	atom-common.owl	$\mathcal{ALCHI}$	19106	43613	89	124	1.037	1.243	1.604	1.724
6	unnamed.owl	$\mathcal{ALCF}$	4343	24092	21	87	0.794	1.155	1.375	1.594
7	physics-complex.owl	$\mathcal{ALEQ}$	3909	115457	8	330	2.155	5.642	2.631	3.133
8	subatomic-particle-complex.owl	$\mathcal{ALC}$	21142	121651	75	335	2.817	3.66	3.394	4.07
9	atom-primitive.owl	$\mathcal{ALH}$	40273	109830	136	383	1.144	1.594	1.745	2.117
10	bfo.owl	$\mathcal{ALC}$	48992	82021	88	212	1.342	2.065	2.092	2.665
11	brokenPizza.owl	$\mathcal{ALCN}$	35188	80534	137	245	1.816	2.239	1.561	2.011
12	pharmacogenomics-complex.owl	$\mathcal{ALC}$	76031	287734	256	706	2.126	283.431	2.543	3.643
13	periodic-table-complex.owl	$\mathcal{ALU}$	32596	256468	58	572	1.52	2.794	2.145	3.209
14	Ontology1191594278.owl	$\mathcal{ALCN}$	33870	119110	140	379	2.492	19.111	1.596	2.553
15	substance.owl	$\mathcal{ALCF}$	109294	221817	458	703	1.279	2.494	1.847	2.994
16	particle.owl	$\mathcal{ALCQ}$	39897	259670	87	529	1.363	T	1.741	2.943
17	Movie.owl	$\mathcal{ALCN}$	28384	117745	140	379	2.377	12.498	1.417	4.027
18	UnsatCook.owl	$\mathcal{ALCF}$	14321	60378	38	156	0.958	1.925	1.239	4.885
19	reaction.owl	$\mathcal{ALCHIQ}$	46705	269999	88	553	1.625	T	M	M
20	cton.owl	$\mathcal{SHF}$	23193568	24064647	33207	87285	11.389	1749.941	11.808	M
21	organic-compound-complex.owl	$\mathcal{ALCI}$	11836	298344	25	671	1.58	T	2.141	M
22	so-xp.obo.owl		1133677	2123808	1943	7039	3.287	T	3.472	M
23	ontology.owl	$\mathcal{ALCHIF}$	443918	1473601	1960	4019	3.135	T	3.777	M
24	DOLCE_Lite_397.owl	$\mathcal{SHIF}$	105716	148804	351	523	1.173	4.692	1.954	M
25	organic-functional-group-complex.owl	$\mathcal{ALCQ}$	16249	225042	30	436	1.985	T	2.628	T

Table 3: Raw test data: File sizes are given in bytes, durations in seconds, M marks an OutOfMemoryException by the reasoner, T marks that reasoning was stopped after a timeout of 300 seconds.

Nr.	Name		HermiT user-time:				Pellet user-time:			
		Axiom count	Average time	Standard deviation	Passed tests	Errors Time/Memory/Other	Average time	Standard deviation	Passed tests	Error Time/Memory/Other
1	atom-common	89	1.84	0.25	70	0/0/0	2.8	0.62	70	0/0/0
2	atom-primitive	136	5.13	0.6	197	1/0/0	5.91	0.86	198	0/0/0
3	brokenPizza	137	9.96	4.32	76	2/0/0	5.1	0.74	15	0/0/63
4	download	24	2.39	0.35	39	0/0/0	3.93	1.18	34	0/0/5
5	Movie	140	161.38	104.06	11	249/0/0	–	–	0	0/0/260
6	Ontology1191594278	140	545.13	0.0	1	95/0/0	–	–	0	0/0/96
7	ontology	1960	58.3	11.29	378	49/0/0	–	–	0	2/322/103
8	organic-compound-complex	25	19.37	2.1	50	0/0/0	–	–	0	49/0/1
9	organic-functional-group-complex	30	12.82	2.09	13	0/0/0	–	–	0	4/0/9
10	particle	87	37.82	17.03	54	201/0/0	–	–	0	2/0/253
11	pharmacogenomics-complex	256	–	–	0	1/0/50	–	–	0	1/0/50
12	physics-complex	8	–	–	0	0/0/2	–	–	0	0/0/2
13	reaction	88	–	–	0	0/0/257	–	–	0	0/0/257
14	ribosome	21	1.29	5.0e-2	11	0/0/0	1.61	3.0e-2	11	0/0/0
15	subatomic-particle-complex	75	5.23	0.36	52	0/0/0	4.58	0.18	52	0/0/0
16	substance	458	12.44	1.92	240	0/0/0	9.08	0.39	25	0/0/215
17	unnamed	21	1.28	6.0e-2	11	0/0/0	1.62	3.0e-2	11	0/0/0
18	UnsatCook	38	18.03	11.59	132	0/0/0	7.42	0.0	1	0/0/131

Table 4: Raw test data: Each row describes a summary of all $\mathcal{ALCQ}\mathsf{me}_2$ ontologies generated from it. Other exceptions include the number of cases where pellet. Times and standard deviations are given in seconds.