Optimizing the Computation of Overriding

Piero A. Bonatti, Iliana M. Petrova, and Luigi Sauro(✉)

Dip. Ing. Elet. e Tecnologie dell'Informazione,
Università di Napoli Federico II, Naples, Italy
luigi.sauro74@gmail.com

Abstract. We introduce optimization techniques for reasoning in $\mathcal{DL}^{\mathrm{N}}$– a recently introduced family of nonmonotonic description logics whose characterizing features appear well-suited to model the examples naturally arising in biomedical domains and semantic web access control policies. Such optimizations are validated experimentally on large KBs with more than 30K axioms. Speedups exceed 1 order of magnitude. For the first time, response times compatible with real-time reasoning are obtained with nonmonotonic KBs of this size.

1 Introduction

Recently, a new family of nonmonotonic Description Logics (DLs), called $\mathcal{DL}^{\mathrm{N}}$, has been introduced [8]. It supports *normality concepts* NC to denote the normal/standard/ prototypical instances of a concept C, and prioritized *defeasible inclusions* (DIs) $C \sqsubseteq_n D$ with the following meaning: "*by default, the instances of* C *satisfy* D*, unless stated otherwise*", that is, unless some higher priority axioms entail $C \sqcap \neg D$; in that case, $C \sqsubseteq_n D$ is *overridden*. The normal/standard/prototypical instances of C are required to satisfy all the DIs that are not overridden in C.

Given the negligible number of applications based on nonmonotonic logics deployed so far, $\mathcal{DL}^{\mathrm{N}}$ has been designed to address real-world problems and concrete knowledge engineering needs. In this regard, the literature provides clear and articulated discussions of how nonmonotonic reasoning can be of help in important contexts related to the semantic web, such as biomedical ontologies [25,28] (with several applications, such as literature search) and (semantic web) policy formulation [29]. These and other applications are extensively discussed in [8].

The distinguishing features in $\mathcal{DL}^{\mathrm{N}}$'s design are: (*i*) $\mathcal{DL}^{\mathrm{N}}$ adopts the simplest possible criterion for overriding, that is, inconsistency with higher priority axioms; (*ii*) all the normal instances of a concept C conform to the same set of default properties, also called *prototype* in the following; (*iii*) the conflicts between DIs that cannot be resolved with priorities are regarded as knowledge representation errors and are to be fixed by the knowledge engineer (typically, by adding specific DIs). No traditional nonmonotonic logic satisfies (*i*), and very few satisfy (*ii*) or (*iii*). $\mathcal{DL}^{\mathrm{N}}$ behaves very well on application examples due to the following consequences of (*i*)–(*iii*) (a comparison with other nonmonotonic DLs with respect to these features is summarized in Table 1):

M. Arenas et al. (Eds.): ISWC 2015, Part I, LNCS 9366, pp. 356–372, 2015.
DOI: 10.1007/978-3-319-25007-6_21

No inheritance blocking: In several nonmonotonic logics a concept with exceptional properties inherits *none* of the default properties of its superclasses. This undesirable phenomenon is known as *inheritance blocking*.

No undesired closed-world assumption (CWA) effects: In some nonmonotonic DLs, an exceptional concept is shrinked to the individuals that explicitly belong to it, if any; hence, it may become inconsistent.

Control on role ranges: Unlike most nonmonotonic DLs, $\mathcal{DL}^{\mathrm{N}}$ axioms can specify whether a role should range only over normal individuals or not.

Detect inconsistent prototypes: $\mathcal{DL}^{\mathrm{N}}$ facilitates the identification of all conflicts that cannot be resolved with priorities (via consistency checks over normality concepts), because their correct resolution is application dependent and should require human intervention (cf. [8, Sec. 1] and Example 1 below).

Tractability: $\mathcal{DL}^{\mathrm{N}}$ is currently the only nonmonotonic DL known to preserve the tractability of all low-complexity DLs, including $\mathcal{EL}^{++}$ and *DL-lite* (that underly the OWL2-EL and OWL2-QL profiles). This opens the way to processing very large nonmonotonic KBs within these fragments.

Table 1. Partial comparison with other nonmonotonic DLs, cf.[8], where CIRC, DEF, AEL, TYP, RAT, PR stand, respectively, for Circumscribed DLs, Default DLs, Autoepistemic DLs, DLs with Typicality, DLs with Rational Closure, and Probabilistic DLs.

Features	CIRC [5,6]	DEF [1,2]	AEL [13]	TYP [17,18]	RAT [10,11]	RAT [12]	PR [22]	$\mathcal{DL}^{\mathrm{N}}$
no inheritance blocking	✓	✓	✓			✓	✓	✓
no CWA effects		✓	✓		✓	✓		✓
fine-grained control on role ranges				sometimes				✓
detects inconsistent prototypes				sometimes			✓	✓
preserves tractability								✓(*)

(*) It holds for subsumption, assertion checking, concept consistency, KB consistency.

The performance of $\mathcal{DL}^{\mathrm{N}}$ inference has been experimentally analyzed on large KBs (with more than 20K concept names and over 30K inclusions). The results are promising; still, as defeasible inclusions approach 25% of the KB, query response time slows down enough to call for improvements. In this paper, we study two optimization techniques to improve $\mathcal{DL}^{\mathrm{N}}$ query response time:

1. Many of the axioms in a large KB are expected to be irrelevant to the given query. We investigate the use of *module extractors* [24,27] to focus reasoning on relevant axioms only. The approach is not trivial (module extractors are unsound for most nonmonotonic logics, including circumscription, default and autoepistemic logics) and requires an articulated correctness proof.
2. We introduce a new algorithm for query answering, that is expected to exploit incremental reasoners at their best. Incremental reasoning is crucial as $\mathcal{DL}^{\mathrm{N}}$'s reasoning method iterates consistency tests on a set of KBs with large intersections. While the assertion of new axioms is processed very

efficiently, the computational cost of axiom deletion is generally not negligible. We introduce an *optimistic reasoning method* that is expected to reduce the number of deletions.

Both optimizations are validated experimentally. Speedups exceed 1 order of magnitude. To the best of our knowledge, this is the first time that response times compatible with real-time reasoning are obtained with nonmonotonic KBs of this size.

The paper is organized as follows: Sec. 2 provides the basics of $\mathcal{DL}^{\mathrm{N}}$ and illustrates its inferences with examples. Sections 3 and 4 introduce the two optimization methods, respectively, and prove their correctness. Their experimental assessment is in Sec. 5. Proofs have been omitted due to space limitations. They can be found in [7], together with further explanations and examples. We assume the reader to be familiar with description logics, see [15] for all details. The code and test suites are available at: http://goo.gl/KnMO9l.

2 Preliminaries

Let $\mathcal{DL}$ be any classical description logic language (see [15] for definitions), and let $\mathcal{DL}^{\mathrm{N}}$ be the extension of $\mathcal{DL}$ with a new concept name $\mathrm{N}C$ for each $\mathcal{DL}$ concept C. The new concepts are called *normality concepts*.

A $\mathcal{DL}^{\mathrm{N}}$ *knowledge base* is a disjoint union $\mathcal{KB} = \mathcal{S} \cup \mathcal{D}$ where $\mathcal{S}$ is a finite set of $\mathcal{DL}^{\mathrm{N}}$ inclusions and assertions (called *strong* or classical axioms) and $\mathcal{D}$ is a finite set of *defeasible inclusions* (DIs, for short) that are expressions $C \sqsubseteq_n D$ where C is a $\mathcal{DL}$ concept and D a $\mathcal{DL}^{\mathrm{N}}$ concept. If $\delta = (C \sqsubseteq_n D)$, then $\mathsf{pre}(\delta)$ and $\mathsf{con}(\delta)$ denote C and D, respectively. Informally speaking, the set of DIs satisfied by all the instances of a normality concept $\mathrm{N}C$ constitute the *prototype* associated to C.

DIs are prioritized by a strict partial order $\prec$. If $\delta_1 \prec \delta_2$, then δ_1 has higher priority than δ_2. $\mathcal{DL}^{\mathrm{N}}$ solves automatically only the conflicts that can be settled using $\prec$; any other conflict shall be resolved by the knowledge engineer (typically by adding suitable DIs). Two priority relations have been investigated so far. Both are based on *specificity*: the specific default properties of a concept C have higher priority than the more generic properties of its superconcepts (i.e. those that subsume C). The priority relation used in most of [8]'s examples identifies those superconcepts with strong axioms only:

$$\delta_1 \prec \delta_2 \text{ iff } \mathsf{pre}(\delta_1) \sqsubseteq_{\mathcal{S}} \mathsf{pre}(\delta_2) \text{ and } \mathsf{pre}(\delta_2) \not\sqsubseteq_{\mathcal{S}} \mathsf{pre}(\delta_1)\,.^{1} \tag{1}$$

The second priority relation investigated in [8] is

$$\delta_1 \prec \delta_2 \text{ iff } \mathit{rank}(\delta_1) > \mathit{rank}(\delta_2), \tag{2}$$

where $\mathit{rank}(\cdot)$ is shown in Algorithm 1 and corresponds to the ranking function of rational closure [11,12]. This relation uses also DIs to determine superconcepts,

[1] As usual, $C \sqsubseteq_{\mathcal{S}} D$ means that $\mathcal{S} \models C \sqsubseteq D$.

Algorithm 1. Ranking function

Input: Ontology $\mathcal{KB} = \mathcal{S} \cup \mathcal{D}$
Output: the function $rank(\cdot)$

$i := -1$; $\mathcal{E}_0 := \{C \sqsubseteq D \mid C \sqsubseteq_n D \in \mathcal{D}\}$
repeat
  $i := i + 1$
  $\mathcal{E}_{i+1} := \{C \sqsubseteq D \in \mathcal{E}_i \mid \mathcal{S} \cup \mathcal{E}_i \models C \sqsubseteq \bot\}$
  forall $C \sqsubseteq_n D$ *s.t.* $C \sqsubseteq D \in \mathcal{E}_i \setminus \mathcal{E}_{i+1}$ **do**
    assign $rank(C \sqsubseteq_n D) := i$
until $\mathcal{E}_{i+1} = \mathcal{E}_i$
forall $C \sqsubseteq_n D \in \mathcal{E}_{i+1}$ **do** assign $rank(C \sqsubseteq_n D) := \infty$
return $rank(\cdot)$

so (roughly speaking) a DI $C \sqsubseteq_n D$—besides defining a default property for C—gives the specific default properties of C higher priority than those of D. The advantage of this priority relation is that it resolves more conflicts than (1); the main advantage of (1) is predictability; e.g. the effects of adding default properties to an existing, classical KB are more predictable, as the hierarchy used for determining specificity and resolving conflicts is the original, validated one, and is not affected by the new DIs (see also the related discussion in [3,4], that adopt (1)).

The expression $\mathcal{KB} \mid\!\!\approx \alpha$ means that α is a $\mathcal{DL}^{\mathrm{N}}$ *consequence* of $\mathcal{KB}$. Due to space limitations, we do not report the model-theoretic definition of $\mid\!\!\approx$ and present only its reduction to classical reasoning [8]. For all subsumptions and assertions α, $\mathcal{KB} \mid\!\!\approx \alpha$ holds iff $\mathcal{KB}^{\Sigma} \models \alpha$, where Σ is the set of normality concepts that explicitly occur in $\mathcal{KB} \cup \{\alpha\}$, and $\mathcal{KB}^{\Sigma}$ is a classical knowledge base obtained as follows (recall that $\mathcal{KB} = \mathcal{S} \cup \mathcal{D}$):

First, for all DIs $\delta \in \mathcal{D}$ and all $\mathrm{N}C \in \Sigma$, let:

$$\delta^{\mathrm{N}C} = \left(\mathrm{N}C \sqcap \mathsf{pre}(\delta) \sqsubseteq \mathsf{con}(\delta)\right). \tag{3}$$

The informal meaning of $\delta^{\mathrm{N}C}$ is: "$\mathrm{N}C$'s instances satisfy δ".

Second, let $\mathcal{S}' \downarrow_{\prec\delta}$ denote the result of removing from the axiom set $\mathcal{S}'$ all the $\delta_0^{\mathrm{N}C}$ such that $\delta_0 \not\prec \delta$:

$$\mathcal{S}' \downarrow_{\prec\delta} = \mathcal{S}' \setminus \{\delta_0^{\mathrm{N}C} \mid \mathrm{N}C \in \Sigma \wedge \delta_0 \not\prec \delta\}.$$

Third, let $\delta_1, \ldots, \delta_{|\mathcal{D}|}$ be any *linearization* of $(\mathcal{D}, \prec)$.[2]

Finally, let $\mathcal{KB}^{\Sigma} = \mathcal{KB}^{\Sigma}_{|\mathcal{D}|}$, where the sequence $\mathcal{KB}^{\Sigma}_i$ $(i = 1, 2, \ldots, |\mathcal{D}|)$ is inductively defined as follows:

$$\mathcal{KB}^{\Sigma}_0 = \mathcal{S} \cup \left\{\mathrm{N}C \sqsubseteq C \mid \mathrm{N}C \in \Sigma\right\} \tag{4}$$

$$\mathcal{KB}^{\Sigma}_i = \mathcal{KB}^{\Sigma}_{i-1} \cup \left\{\delta_i^{\mathrm{N}C} \mid \delta_i \in \mathcal{KB}, \mathrm{N}C \in \Sigma, \text{ and } \mathcal{KB}^{\Sigma}_{i-1} \downarrow_{\prec\delta_i} \cup \{\delta_i^{\mathrm{N}C}\} \not\models \mathrm{N}C \sqsubseteq \bot\right\}. \tag{5}$$

[2] That is, $\{\delta_1, \ldots, \delta_{|\mathcal{D}|}\} = \mathcal{D}$ and for all $i, j = 1, \ldots, |\mathcal{D}|$, if $\delta_i \prec \delta_j$ then $i < j$.

In other words, the above sequence starts with $\mathcal{KB}$'s strong axioms extended with the inclusions $\mathrm{N}C \sqsubseteq C$, then processes the DIs δ_i in non-increasing priority order. If δ_i can be consistently added to C's prototype, given all higher priority DIs selected so far (which is verified by checking that $\mathrm{N}C \not\sqsubseteq \bot$ in line (5)), then its translation $\delta_i^{\mathrm{N}C}$ is included in $\mathcal{KB}^{\Sigma}$ (i.e. δ_i enters C's prototype), otherwise δ_i is discarded, and we say that δ_i is *overridden in* $\mathrm{N}C$.

2.1 Examples

We start with a brief discussion of $\mathcal{DL}^{\mathrm{N}}$'s conflict handling. Most other logics silently neutralize the conflicts between nonmonotonic axioms with the same (or incomparable) priorities by computing the inferences that are invariant across all possible ways of resolving the conflict. A knowledge engineer might solve it in favor of *some* of its possible resolutions, instead; however, if the logic silently neutralizes the conflict, then missing knowledge may remain undetected and unfixed. This approach may cause serious problems in the policy domain:

Example 1. Suppose that project coordinators are both administrative staff and research staff. By default, administrative staff are allowed to sign payments, while research staff are not. A conflict arises since both of these default policies apply to project coordinators. Formally, $\mathcal{KB}$ can be formalized with:

$$\texttt{Admin} \sqsubseteq_{\mathrm{n}} \exists \texttt{has\_right.Sign} \quad (6)$$

$$\texttt{Research} \sqsubseteq_{\mathrm{n}} \neg\exists \texttt{has\_right.Sign} \quad (7)$$

$$\texttt{PrjCrd} \sqsubseteq \texttt{Admin} \sqcap \texttt{Research} \quad (8)$$

Leaving the conflict unresolved may cause a variety of security problems. If project coordinators should *not* sign payments, and the default policy is *open* (authorizations are granted by default), then failing to infer $\neg\exists\texttt{has\_right.Sign}$ would improperly authorize the signing operation. Conversely, if the authorization is to be granted, then failing to prove $\exists\texttt{has\_right.Sign}$ causes a *denial of service* (the user is unable to complete a legal operation). To prevent these problems, $\mathcal{DL}^{\mathrm{N}}$ makes the conflict visible by inferring $\mathcal{KB} \approx \mathrm{N}\,\texttt{PrjCrd} \sqsubseteq \bot$ (showing that `PrjCrd`'s prototype is inconsistent). This can be proved by checking that $\mathcal{KB}^{\Sigma} \models \mathrm{N}\,\texttt{PrjCrd} \sqsubseteq \bot$, where $\Sigma = \{\mathrm{N}\,\texttt{PrjCrd}\}$. Then $\mathcal{KB}^{\Sigma}$ consists of (8), $\mathrm{N}\,\texttt{PrjCrd} \sqsubseteq \texttt{PrjCrd}$, and the translation of (6) *and* (7) (none overrides the other because none is more specific under any of the two priorities):

$$\mathrm{N}\,\texttt{PrjCrd} \sqcap \texttt{Admin} \sqsubseteq \exists\texttt{has\_right.Sign},$$
$$\mathrm{N}\,\texttt{PrjCrd} \sqcap \texttt{Research} \sqsubseteq \neg\exists\texttt{has\_right.Sign}. \qquad \square$$

Here is another application example from the semantic policy domain, showing $\mathcal{DL}^{\mathrm{N}}$'s behavior on multiple exception levels.

Example 2. We are going to axiomatize the following natural language policy: *"In general, users cannot access confidential files; Staff can read confidential files; Blacklisted users are not granted any access. This directive cannot be overridden."* Note

that each of the above directives contradicts (and is supposed to override) its predecessor in some particular case. Authorizations can be reified as objects with attributes *subject* (the access requestor), *target* (the file to be accessed), and *privilege* (such as *read* and *write*). Then the above policy can be encoded as follows:

$$\texttt{Staff} \sqsubseteq \texttt{User} \tag{9}$$
$$\texttt{Blklst} \sqsubseteq \texttt{Staff} \tag{10}$$
$$\texttt{UserReqst} \sqsubseteq_n \neg\exists\texttt{privilege} \tag{11}$$
$$\texttt{StaffReqst} \sqsubseteq_n \exists\texttt{privilege.Read} \tag{12}$$
$$\texttt{BlkReq} \sqsubseteq \neg\exists\texttt{privilege} \tag{13}$$

where $\texttt{BlkReq} \doteq \exists\texttt{subj.Blklst}$, $\texttt{StaffReqst} \doteq \exists\texttt{subj.Staff}$, and $\texttt{UserReqst} \doteq \exists\texttt{subj.User}$. By (9), both the specifity relations (1) and (2) yield (12) $\prec$ (11), that is, (12) has higher priority than (11). Let $\Sigma = \{\text{N}\texttt{StaffReqst}\}$; (12) overrides (11) in N`StaffReqst` (under (1) as well as (2)), so $\mathcal{KB}^{\Sigma}$ consists of: (9), (10), (13), plus

$$\text{N}\texttt{StaffReqst} \sqsubseteq \texttt{StaffReqst}$$
$$\text{N}\texttt{StaffReqst} \sqcap \texttt{StaffReqst} \sqsubseteq \exists\texttt{privilege.Read}$$

Consequently, $\mathcal{KB} \mid\!\approx \text{N}\texttt{StaffReqst} \sqsubseteq \exists\texttt{privilege.Read}$. Similarly, it can be verified that:

1. Normally, access requests involving confidential files are rejected, if they come from generic users: $\mathcal{KB} \mid\!\approx \text{N}\texttt{UserReqst} \sqsubseteq \neg\exists\texttt{privilege}$;
2. Blacklisted users cannot do anything by (13), so, in particular: $\mathcal{KB} \mid\!\approx \text{N}\texttt{BlkReq} \sqsubseteq \neg\exists\texttt{privilege}$. □

Some application examples from the biomedical domain can be found in [8] (see Examples 3, 4, 10, 12, and the drug contraindication example in Appendix C). Like the above examples, they are all correctly solved by $\mathcal{DL}^{\text{N}}$ with both priority notions. Applicative examples hardly exhibit the complicated networks of dependencies between conflicting defaults that occur in artificial examples. Nonetheless, we briefly discuss the artificial examples, too, as a means of comparing $\mathcal{DL}^{\text{N}}$ with other logics such as [5,12,26].

In several cases, e.g. examples B.4 and B.5 in [26], $\mathcal{DL}^{\text{N}}$ agrees with [5,12,26] under both priority relations. Due to space limitations, we illustrate only B.4.

Example 3 (Juvenile offender). Let $\mathcal{KB}$ consist of axioms (14)–(18) where `J`, `G`, `M`, `P` abbreviate `JuvenileOffender`, `GuiltyOfCrime`, `IsMinor` and `ToBePunished`, respectively.

$$\texttt{J} \sqsubseteq \texttt{G} \tag{14}$$
$$\texttt{J} \sqsubseteq \texttt{M} \tag{15}$$
$$\texttt{M} \sqcap \texttt{G} \sqsubseteq_n \neg\texttt{P} \tag{16}$$
$$\texttt{M} \sqsubseteq_n \neg\texttt{P} \tag{17}$$
$$\texttt{G} \sqsubseteq_n \texttt{P} \tag{18}$$

$$\texttt{J} \sqsubseteq \texttt{G} \tag{19}$$
$$\texttt{J} \sqsubseteq \texttt{M} \tag{20}$$
$$\text{N}\texttt{J} \sqsubseteq \texttt{J} \tag{21}$$
$$\text{N}\texttt{J} \sqcap \texttt{M} \sqcap \texttt{G} \sqsubseteq \neg\texttt{P} \tag{22}$$
$$\text{N}\texttt{J} \sqcap \texttt{M} \sqsubseteq \neg\texttt{P} \tag{23}$$

On one hand, criminals have to be punished and, on the other hand, minors cannot be punished. So, what about juvenile offenders? The defeasible inclusion (16) breaks the tie in favor of their being underage, hence not punishable. By setting $\Sigma = \{\mathtt{NJ}\}$, priorities (1) and (2) both return axioms (19)–(23) as $\mathcal{KB}^{\Sigma}$. Then, clearly, $\mathcal{KB}^{\Sigma} \models \mathtt{NJ} \sqsubseteq \neg\mathtt{P}$ which is $\mathcal{DL}^{\mathrm{N}}$'s analogue of the inferences of [5,12,26].

In other cases (e.g. example B.1 in [26]) $\mathcal{DL}^{\mathrm{N}}$ finds the same conflicts as [5,12,26]. However, $\mathcal{DL}^{\mathrm{N}}$'s semantics signals these conflicts to the knowledge engineer whereas in [5,12,26] they are silently neutralized.

Example 4 (Double Diamond). Let $\mathcal{KB}$ be the following set of axioms:

$$\mathtt{A} \sqsubseteq_n \mathtt{T} \quad (24) \qquad \mathtt{S} \sqsubseteq_n \mathtt{R} \quad (28)$$
$$\mathtt{A} \sqsubseteq_n \mathtt{P} \quad (25) \qquad \mathtt{P} \sqsubseteq_n \mathtt{Q} \quad (29)$$
$$\mathtt{T} \sqsubseteq_n \mathtt{S} \quad (26) \qquad \mathtt{Q} \sqsubseteq_n \neg\mathtt{R} \quad (30)$$
$$\mathtt{P} \sqsubseteq_n \neg\mathtt{S} \quad (27)$$

DIs (26) and (27) have incomparable priority under (1) and (2). Consequently, it is easy to see that $\mathtt{NA} \sqsubseteq \mathtt{S}$ and $\mathtt{NA} \sqsubseteq \neg\mathtt{S}$ are both implied by $\mathcal{KB}^{\Sigma}$ and hence the knowledge engineer is warned that NA is inconsistent. The same conflict is silently neutralized in [5,12,26] (A's instances are subsumed by neither S nor $\neg$S and no inconsistency arises). Similarly for the incomparable DIs (28) and (30) and the related conflict.

The third category of examples (e.g. B.2 and B.3 in [26]) presents a more variegated behavior. In particular, [12] and $\mathcal{DL}^{\mathrm{N}}$ with priority (2) solve all conflicts and infer the same consequences; [26] solves only some conflicts; [5] is not able to solve any conflict and yet it does not raise any inconsistency warning; $\mathcal{DL}^{\mathrm{N}}$ with priority (1) cannot solve the conflicts but raises an inconsistency warning. Here, for the sake of simplicity, we discuss in detail a shorter example which has all relevant ingredients.

Example 5. Let $\mathcal{KB}$ be the following defeasible knowledge base:

$$\mathtt{A} \sqsubseteq_n \mathtt{B} \quad (31) \qquad \mathtt{A} \sqsubseteq_n \mathtt{C} \quad (32) \qquad \mathtt{B} \sqsubseteq_n \neg\mathtt{C} \quad (33)$$

According to priority (1) all DIs are incomparable. Therefore, $\mathcal{DL}^{\mathrm{N}}$ warns (by inferring $\mathtt{NA} \sqsubseteq \bot$) that the conflict between $\mathtt{NA} \sqsubseteq \mathtt{C}$ and $\mathtt{NA} \sqsubseteq \neg\mathtt{C}$ cannot be solved. Note that [5] adopts priority (1), too, however according to circumscription, any interpretation where A's instances are either in $\neg\mathtt{C} \sqcap \mathtt{B}$ or in C is a model, so A is satisfiable (the conflict is silently neutralized). Under priority (2), instead, axiom (31) gives (31) and (32) higher priority than (33). Consequently, $\mathtt{NA} \sqsubseteq \mathtt{C}$ prevails over $\mathtt{NA} \sqsubseteq \neg\mathtt{C}$. In this case, $\mathcal{DL}^{\mathrm{N}}$ and rational closure infer the same consequences.

3 Relevance and Modularity

The naive construction of $\mathcal{KB}^{\Sigma}$ must process all the axioms in $\mathcal{KB}^{\Sigma}_{all} = \mathcal{KB}^{\Sigma}_{0} \cup \{\delta^{\mathrm{NC}} \mid \delta \in \mathcal{D},\ \mathrm{N}C \in \Sigma\}$. Here we optimize $\mathcal{DL}^{\mathrm{N}}$ inference by quickly discarding some of the irrelevant axioms in $\mathcal{KB}^{\Sigma}_{all}$ using modularization techniques.

Roughly speaking, the problem of module extraction can be expressed as follows: given a reference vocabulary *Sig*, a module is a (possibly minimal) subset $\mathcal{M} \subseteq \mathcal{KB}$ that is relevant for *Sig* in the sense that it preserves the consequences of $\mathcal{KB}$ that contain only terms in *Sig*.

The interest in module extraction techniques is motivated by several ontology engineering needs. We are interested in modularization as an optimization technique for querying large ontologies: the query is evaluated on a (hopefully much smaller) module of the ontology that preserves the query result (as well as any inference whose signature is contained in the query's signature).

However, the problem of deciding whether two knowledge bases entail the same axioms over a given signature is usually harder than standard reasoning tasks. Consequently deciding whether $\mathcal{KB}'$ is a module of $\mathcal{KB}$ (for *Sig*) is computationally expensive in general. For example, $DL\text{–}Lite_{horn}$ complexity grows from PTIME to coNP-TIME-complete [21]; for $\mathcal{ALC}$, complexity is one exponential harder [16], while for $\mathcal{ALCQIO}$ the problem becomes even undecidable [23].

In order to achieve a practical solution, a syntactic approximation has been adopted in [19,27]. The corrisponding algorithm $\top\bot^*$-Mod$(Sig, \mathcal{KB})$ is defined in [27, Def. 4] and reported in Algorithm 2 below. It is based on the property of $\bot$-locality and $\top$-locality of single axioms (line 15). An axiom is local w.r.t. *Sig* if the substitution of all non-*Sig* terms with $\bot$ (resp. $\top$) turns it into a tautology.

The module extractor identifies a subset $\mathcal{M} \subseteq \mathcal{KB}$ of the knowledge base and a signature *Sig* (containing all symbols of interest) such that all axioms in $\mathcal{KB} \setminus \mathcal{M}$ are local w.r.t. *Sig*. This guarantees that every model of $\mathcal{M}$ can be extended to a model of $\mathcal{KB}$ by setting each non-*Sig* term to either $\bot$ or $\top$. In turn, this property guarantees that any query whose signature is contained in *Sig* has the same answer in $\mathcal{M}$ and $\mathcal{KB}$.

The function x-Mod$(Sig, \mathcal{KB})$ (lines 9-19), where x stands for $\top$ or $\bot$, describes the procedure for constructing modules of a knowledge base $\mathcal{KB}$ for each notion of locality. Starting with an empty set of axioms (line 11), iteratively, the axioms α that are non-local are added to the module (line 16) and, in order to preserve soundness, the signature against which locality is checked is extended with the terms in α (line 15). Iteration stops when a fixpoint is reached.

Modules based on a single syntactic locality can be further shrinked by iteratively nesting $\top$-extraction into $\bot$-extraction, thus obtaining $\top\bot^*$-Mod$(Sig, \mathcal{KB})$ modules (lines 1-8).

The notions of module and locality must be extended to handle DIs, before we can apply them to $\mathcal{DL}^{\mathrm{N}}$. Definition 1 generalizes the substitutions operated by the module extraction algorithm, abstracting away procedural details. As in [27], both $\widetilde{X}$ and sig(X) denote the signature of X.

[2] For efficiency, this test is approximated by a matching with a small set of templates.

Algorithm 2. $\top\bot^*$-Mod$(Sig, \mathcal{KB})$

Input: Ontology $\mathcal{KB}$, signature Sig
Output: $\top\bot^*$-module $\mathcal{M}$ of $\mathcal{KB}$ w.r.t. Sig

```
   // main
1  begin
2  |  M := KB
3  |  repeat
4  |  |  M' := M
5  |  |  M := ⊤-Mod(⊥-Mod(M, Sig), Sig)
6  |  until M ≠ M'
7  |  return M
8  end
9  function x-Mod(KB,Sig)  //   x ∈ {⊥, ⊤}
10 begin
11 |  M := ∅,  T := KB
12 |  repeat
13 |  |  changed = false
14 |  |  forall α ∈ T do
15 |  |  |  if  α is not x-local w.r.t. Sig ∪ M~  then
16 |  |  |  |  M := M ∪ {α}
17 |  |  |  |  T := T \ {α}
18 |  |  |  |  changed = true
19 |  until changed = false
20 |  return M
21 end
```

Definition 1. ***(Module, locality)*** *A* $\top\bot^*$-substitution *for* $\mathcal{KB}$ *and a signature* Sig *is a substitution* σ *over* $\widetilde{\mathcal{KB}} \setminus Sig$ *that maps each concept name on* $\top$ *or* $\bot$*, and every role name on the universal role or the empty role. A strong axiom* α *is* σ-local *iff* $\sigma(\alpha)$ *is a tautology. A DI* $C \sqsubseteq_n D$ *is* σ*-local iff* $C \sqsubseteq D$ *is* σ*-local. A set of axioms is* σ*-local if all of its members are. We say that an axiom* α *is* $\top$*-local (resp.* $\bot$*-local) if* α *is* σ*-local where the substitution* σ *uniformly maps concept names to* $\top$ *(resp.* $\bot$*).*

A (syntactic) module *of* $\mathcal{KB}$ *with respect to* Sig *is a set* $\mathcal{M} \subseteq \mathcal{KB}$ *such that* $\mathcal{KB} \setminus \mathcal{M}$ *is* σ*-local for some* $\top\bot^*$*-substitution* σ *for* $\mathcal{KB}$ *and* $\widetilde{\mathcal{M}} \cup Sig$*.*

Let $\text{Mod}_{\text{DI}}(Sig, \mathcal{KB})$ be the variant of $\top\bot^*$-Mod$(Sig, \mathcal{KB})$ where the test in line 2 is replaced with (the complement of) the $\top$ or $\bot$-locality condition of Def. 1 (that covers DIs, too). Using the original correctness argument for $\top\bot^*$-Mod$(Sig, \mathcal{KB})$ cf. [19, Prop.42], it is easy to see that $\text{Mod}_{\text{DI}}(Sig, \mathcal{KB})$ returns a syntactic module of $\mathcal{KB}$ w.r.t. Sig according to Def. 1. If $\mathcal{KB}$ contains no DIs (i.e. it is classical), then Def. 1 is essentially a rephrasing of standard syntactic notions of modules and locality,[3] so

[3] Informally, $\top\bot^*$-Mod's greedy strategy tends to find small Def. 1's modules.

$$\text{for all queries } \alpha \text{ such that } \widetilde{\alpha} \subseteq \mathit{Sig},\ \mathcal{M} \models \alpha \text{ iff } \mathcal{KB} \models \alpha. \tag{34}$$

However, proving that $\top\bot^*$-Mod$_{\mathrm{DI}}(\mathit{Sig}, \mathcal{KB})$ is correct for *full* $\mathcal{DL}^{\mathrm{N}}$ is far from obvious: removing axioms from $\mathcal{KB}$ using module extractors is incorrect under most nonmonotonic semantics (including circumscription, default logic and autoepistemic logic). The reason is that nonmonotonic inferences are more powerful than classical inferences, and the syntactic locality criterions illustrated above fail to capture some of the dependencies between different symbols.

Example 6. Given the knowledge base $\{\top \sqsubseteq A \sqcup B\}$ and $\mathit{Sig} = \{A\}$, the module extractor returns an empty module (because by setting $B = \top$ the only axiom in the KB becomes a tautology). The circumscription of this KB, assuming that both A and B are minimized, does not entail $A \sqsubseteq \bot$, while the circumscription of the empty module entails it.

Now we illustrate the correct way of applying $\top\bot^*$-Mod$_{\mathrm{DI}}$ to a $\mathcal{DL}^{\mathrm{N}}$ $\mathcal{KB} = \mathcal{S} \cup \mathcal{D}$ and a query α (subsumption or assertion). Let Σ be the union of $\widetilde{\alpha}$ and the set of normality concepts occurring in $\mathcal{KB}$. Let

$$\mathcal{M}_0 = \mathrm{Mod}_{\mathrm{DI}}(\Sigma, \mathcal{KB} \cup \mathrm{N}\Sigma)\,,$$

where $\mathrm{N}\Sigma$ abbreviates $\{\mathrm{N}C \sqsubseteq C \mid \mathrm{N}C \in \Sigma\}$.

Example 7. Let $\mathcal{KB}$ be the knowledge base:

$$A \sqsubseteq B \tag{35}$$
$$A \sqsubseteq_n D \sqcap E \tag{36}$$
$$B \sqcap C \sqsubseteq A \tag{37}$$
$$F \sqsubseteq_n A \tag{38}$$

and α the query $\mathrm{N}A \sqsubseteq D$. $\mathcal{M}_0$ is calculated as follows: first, since no normality concept occurs in $\mathcal{KB}$, Σ is equal to the signature $\widetilde{\alpha} = \{\mathrm{N}A, D\}$.

Algorithm 2 calls first the function $\bot$-Mod$(\mathcal{KB} \cup \mathrm{N}\Sigma, \Sigma)$. Notice that by replacing C and F with $\bot$, axioms (37) and (38) become tautologies. Consequently, it is easy to see that the returned knowledge base is $\mathcal{KB}' = \{(35), (36), \mathrm{N}A \sqsubseteq A\}$.

Then, $\top$-Mod is called on $\mathcal{KB}'$ and Σ. Now, replacing B with $\top$ makes $A \sqsubseteq B$ a tautology, so the resulting knowledge base is $\mathcal{KB}'' = \{(36), \mathrm{N}A \sqsubseteq A\}$. It is easy to see that a fix point is reached and hence $\mathcal{KB}''$ is returned.

We shall prove that $(\mathcal{KB} \cap \mathcal{M}_0)^{\Sigma}$ can be used in place of $\mathcal{KB}^{\Sigma}$ to answer query α. This saves the cost of processing $\mathcal{KB}^{\Sigma}_{all} \setminus \mathcal{M}$, where

$$\mathcal{M} = (\mathcal{KB}^{\Sigma}_0 \cap \mathcal{M}_0) \cup \{\delta^{\mathrm{NC}} \mid \delta \in \mathcal{D} \cap \mathcal{M}_0,\ \mathrm{N}C \in \Sigma\}.$$

Note that $\mathcal{KB}^{\Sigma}_{all} \setminus \mathcal{M}$ is usually even larger than $\mathcal{KB} \setminus \mathcal{M}_0$ because for each DI $\delta \notin \mathcal{M}_0$, all its translations δ^{NC} ($\mathrm{N}C \in \Sigma$) are removed from $\mathcal{M}$.

Lemma 1. *$\mathcal{M}$ is a module of $\mathcal{KB}^{\Sigma}_{all}$ w.r.t. Σ.*

Lemma 2. *If $\mathcal{M}$ is a module of $\mathcal{KB}$ w.r.t. a signature Sig and $\mathcal{KB}' \subseteq \mathcal{KB}$, then $\mathcal{KB}' \cap \mathcal{M}$ is a module of $\mathcal{KB}'$ w.r.t. Sig.*

The relationship between $(\mathcal{KB} \cap \mathcal{M}_0)^\Sigma$ and $\mathcal{KB}^\Sigma$ is:

Lemma 3. $\mathcal{KB}^\Sigma \cap \mathcal{M} \subseteq (\mathcal{KB} \cap \mathcal{M}_0)^\Sigma \subseteq \mathcal{KB}^\Sigma$.

As a consequence, the modularized construction is correct:

Theorem 1. $(\mathcal{KB} \cap \mathcal{M}_0)^\Sigma \models \alpha$ *iff* $\mathcal{KB}^\Sigma \models \alpha$.

Proof. By Lemmas 1 and 2, and (34), $\mathcal{KB}^\Sigma \models \alpha$ iff $\mathcal{KB}^\Sigma \cap \mathcal{M} \models \alpha$. The Theorem then follows by Lemma 3. □

4 Optimistic Computation

The construction of $\mathcal{KB}^\Sigma$ repeats the concept consistency check (5) over a sequence of knowledge bases $(\mathcal{KB}^\Sigma_{i-1} \downarrow_{\prec\delta_i} \cup \{\delta_i^{\mathrm{NC}}\})$ that share a (possibly large) common part $\mathcal{KB}^\Sigma_0$, so incremental reasoning mechanisms help by avoiding multiple computations of the consequences of $\mathcal{KB}^\Sigma_0$. On the contrary, the set of δ_j^{NC} may change significantly at each step due to the filtering $\downarrow_{\prec\delta_i}$. This operation requires many axiom deletions, which as already highlighted in [20], are less efficient than monotonically increasing changes. The optimistic algorithm introduced here (Algorithm 3) computes a knowledge base $\mathcal{KB}^*$ equivalent to $\mathcal{KB}^\Sigma$ in a way that tends to reduce the number of deletions, as it will be assessed in Sec. 5.

Phase 1 optimistically assumes that the DIs with the same priority as δ_i^{NC} do not contribute to entailing $\mathrm{N}C \sqsubseteq \bot$ in (5), so they are not filtered with $\downarrow_{\delta_i}$ in line 3. Phase 2 checks whether the DIs discarded during Phase 1 are actually overridden by applying $\downarrow_{\delta_i}$ (lines 14 and 21). DIs are processed in non-increasing priority order as much as possible (cf. line 19) so as to exploit monotonic incremental classifications.

The following theorem shows the correctness of Alghorithm 3 in case the normality concepts do not occur in $\mathcal{KB}$, but only in the queries. We call such knowledge bases *N-free*. It is worth noting that the optimistic method is not generally correct when $\mathcal{KB}$ is not N-free and $|\Sigma| > 1$, yet it may still be applicable after the module extractor if the latter removes all normality concepts from $\mathcal{KB}$.

Theorem 2. *If $\mathcal{KB}$ is N-free, then Algorithm 3's output is equivalent to $\mathcal{KB}^\Sigma$.*

5 Experimental Assessment

Currently there are no "real" KBs encoded in a nonmonotonic DL, because standard DL technology does not support nonmonotonic reasoning. The nonmonotonic KBs encoded in the hybrid rule+DL system DLV-Hex [14] are not suited to our purposes because they do not feature default inheritance due to a restriction of the language: DL predicates cannot occur in rule heads, so rules cannot be used for encoding default inheritance. Consequently, synthetic test cases are the only choice for evaluating our algorithms. We start with the two

Algorithm 3. Optimistic-Method

Input: $\mathcal{KB} = \mathcal{S} \cup \mathcal{D}$, Σ
Output: a knowledge base $\mathcal{KB}^*$ such that $\mathcal{KB}^* \equiv \mathcal{KB}^\Sigma$

```
// Phase 1
compute a linearization δ_1, ..., δ_|D| of D
Π := ∅  // Π collects the prototypes
Δ := ∅  // ordered list of all discarded δ_i^NC
for i = 1, 2, ..., |D| do
    for NC ∈ Σ do
        Π' := Π ∪ {δ_i^NC}
        if KB_0^Σ ∪ Π' ⊭ NC ⊑ ⊥ then
            Π := Π'
        else
            append δ_i^NC to Δ
// Phase 2
KB* = KB_0^Σ ∪ Π
while Δ ≠ ∅ do
    extract from Δ its first element δ_i^NC
    if (KB_0^Σ ∪ Π) ↓_{≺δ_i} ∪{δ_i^NC} ⊭ NC ⊑ ⊥ then
        KB* := KB* ∪ {NC ⊑ ⊥}
        extract all δ_k^NE with E = C from Δ
    else
        // δ_i^NC is actually overridden
        δ := δ_i
        while Δ contains some δ_j^ND such that δ ≺ δ_j do
            extract from Δ the first such δ_j^ND
            if (KB_0^Σ ∪ Π) ↓_{≺δ_j} ∪{δ_j^ND} ⊭ ND ⊑ ⊥ then
                KB* := KB* ∪ {ND ⊑ ⊥}
                extract all δ_k^NE with E = D from Δ
                δ := δ_j
```

test suites introduced in [8] as they have been proved to be nontrivial w.r.t. a number of structural parameters, including nonclassical features like exception levels and the amount of overriding. The two test suites are obtained by modifying the popular Gene Ontology (GO)[4], which contains 20465 atomic concepts and 28896 concept inclusions. In one test suite, randomly selected axioms of GO are turned into DIs, while in the second suite random synthetic DIs are injected in GO. The amount of strong axioms transformed into DIs is controlled by *CI-to-DI-rate*, expressed as the percentage of transformed axioms w.r.t. |GO| while the amount of additional synthetic DIs is controlled by *Synthetic-DI-rate*, i.e. the ratio $|\mathcal{D}|/|\mathrm{GO}|$. The number of conflicts between DIs can be increased by adding

[4] http://www.geneontology.org

an amount of random disjointness axioms specified by parameter *DA-rate* (see [8] for further details).

The experiments were performed on an Intel Core i7 2,5GHz laptop with 16 GB RAM and OS X 10.10.1, using Java 1.7 configured with 8 GB RAM and 3 GB stack space. Each reported value is the average execution time over ten nonmonotonic ontologies and fifty queries on each ontology. For each parameter setting, we report the execution time of: (*i*) the naive $\mathcal{DL}^{\mathrm{N}}$ reasoner of [8]; (*ii*) the optimistic method introduced in Sec. 4 (Opt); (*iii*) the module extraction method of Sec. 3 (Mod) using the module extraction facility of the OWLAPI; (*iv*) the sequential execution of Mod and Opt, i.e. Algorithm 3 is applied to $\mathcal{KB} \cap \mathcal{M}_0$. This combined method is correct by Theorem 2 and Theorem 1.

Table 2. Impact of $|\mathcal{D}|$ on performance (sec) – DA rate = 15% – priority (1)

CI-to-DI	naive	opt	mod	mod+opt
05%	12.91	05.93	00.30	00.25
10%	22.37	11.13	00.32	00.27
15%	31.50	15.90	00.37	00.32
20%	42.97	20.67	00.40	00.33
25%	55.22	25.17	00.44	00.36

Synth DIs	naive	opt	mod	mod+opt
05%	11.64	06.94	0.41	0.42
10%	21.66	11.21	0.62	0.67
15%	32.80	14.90	1.11	1.64
20%	41.51	18.82	2.01	1.42
25%	51.85	22.33	3.05	2.09

Table 2 shows the impact of the number of DIs on response time for the two test suites, as DI rate ranges from 5% to 25%. The methods Mod and Mod+Opt are slightly less effective in the second suite probably because random defaults connect unrelated parts of the ontology, thereby hindering module extraction. In both suites, Opt's speedup factor (w.r.t. the naive method) is about two, while on average Mod is approximately 87 times faster in the first test suite (max. speedup 125), and 28 times faster in the second (max. speedup 35). On average, the combined method yields a further 13% improvement over Mod alone; the maximum reduction is 31% (2^{nd} suite, Synthetic-DI-rate=25%, DA-rate=15%). The additional conflicts induced by injected disjointness axioms have moderate effects on response time (cf. Table 3). Mod+Opt's average response time across both test suites is 0.7 sec., and the longest Mod+Opt response time has been 2.09 sec. As a term of comparison, a single classification of the original GO takes approximately 0.4 seconds.

Table 4 is the analogue of Table 2 given priority (2). With respect to priority (1), the computation time for $\mathcal{KB}^{\Sigma}$ and query answering in the first test suite grows faster for the naive algorithm, while there are smaller differences for the optimized approaches (the reponse times of the combined approach are almost identical). In the second test suite, the performance of the naive algorithms decreases less dramatically, while the optimized methods seem slightly less effective than in the first test suite. In all cases, the speedups of Mod and Mod-Opt remain well above one order of magnitude. The performance as DAs grow has similar features (see Table 5).

Table 3. Impact of DAs on performance (sec) – DI rate = 15% – priority (1)

Test suite 1 (CI-to-DI)

DA	naive	opt	mod	mod+opt
05%	29.88	13.21	0.36	0.31
10%	32.96	14.08	0.37	0.32
15%	31.50	15.90	0.37	0.32
20%	34.23	16.23	0.39	0.33
25%	36.47	17.80	0.40	0.34
30%	37.71	18.09	0.40	0.34

Test suite 2 (Synth. DIs)

DA	naive	opt	mod	mod+opt
05%	28.20	12.63	0.99	0.84
10%	30.18	13.68	1.04	0.97
15%	32.80	14.90	1.11	1.06
20%	35.68	16.29	1.18	1.10
25%	37.46	17.02	1.25	1.15
30%	38.37	18.79	1.36	1.23

Table 4. Impact of $|\mathcal{D}|$ on performance (sec) – DA rate = 15% – priority (2)

CI-to-DI	naive	opt	mod	mod+opt
05%	22.01	05.74	00.30	00.25
10%	52.82	11.48	00.32	00.28
15%	81.84	16.56	00.34	00.31
20%	133.62	20.51	00.38	00.33
25%	193.27	26.42	00.41	00.36

Synth DIs	naive	opt	mod	mod+opt
05%	12.76	07.21	0.45	0.46
10%	23.72	14.44	0.81	0.86
15%	34.53	17.05	1.57	1.21
20%	44.92	21.77	2.67	1.96
25%	55.92	25.77	3.87	2.46

Table 5. Impact of DAs on performance (sec) – DI rate = 15% – priority (2)

Test suite 1 (CI-to-DI)

DA	naive	opt	mod	mod+opt
05%	84.53	15.02	0.34	0.29
10%	90.38	16.12	0.35	0.30
15%	91.84	16.56	0.35	0.31
20%	92.93	16.67	0.36	0.31
25%	93.54	17.76	0.37	0.32
30%	96.37	19.49	0.38	0.33

Test suite 2 (Synth. DIs)

DA	naive	opt	mod	mod+opt
05%	29.55	14.93	1.28	1.07
10%	30.81	15.82	1.41	1.15
15%	34.54	17.05	1.57	1.21
20%	36.79	16.93	1.62	1.27
25%	40.86	17.90	1.78	1.36
30%	43.35	18.74	1.79	1.34

The above test sets are N-free. We carried out a new set of experiments by randomly introducing normality concepts in DIs, within the scope of quantifiers.[5] Specifically, $\exists R.C$ is transformed into $\exists R.\mathrm{N}\, C$. The response times of the naive algorithm and Mod[6] under priority (1) are listed in Table 6 for increasing values of $|\Sigma|$ (that is directly related to the amount of normality concepts occurring in $\mathcal{KB}$). We estimate that the values of $|\Sigma|$ considered here are larger than what should be expected in practice, given the specific role of explicit normality concepts, cf. footnote 5. Such values are also much larger than in N-free

[5] So far, all the application examples that are not N-free satisfy this restriction, as apparently the only purpose of explicit normality concepts is restricting default role ranges to normal individuals, cf. Ex. 12 and the nomonotonic design pattern in [8, Sec. 3.3].

[6] In this setting Opt and Mod+Opt are not applicable, in general.

Table 6. Impact of normal roles values (sec) – DI rate = 25% DA rate = 15%

$\|\Sigma\|$	50	100	150	200	250
Test suite 1					
naive	1794.37	>30 min.	>30 min.	>30 min.	>30 min.
mod	2.31	7.26	14.77	25.32	39.22
Test suite 2					
naive	>30 min.	>30 min.	>30 min.	>30 min.	>30 min.
mod	103.4	211.5	327.4	459.2	586.7

experiments, where $|\Sigma|$ is bounded by the query size. Response times increase accordingly. In most cases, the naive algorithm exceeded the timeout. In the first test suite, Mod remains well below 1 minute; in the second suite it ranges between 100 seconds and 10 minutes. The reason of the higher computation times in the second suite is that the extracted modules are significantly larger, probably due to the random dependencies between concept names introduced by fully synthetic DIs.

6 Conclusions

The module-based and optimistic optimizations introduced here are sound and complete, where the later applies only if the knowledge base is N-free. In our experiments, the combined method (when applicable) and the module-based method make $\mathcal{DL}^{\mathrm{N}}$ reasoning at least one order of magnitude faster (and up to ∼780 times faster in some case). In most cases, optimized reasoning is compatible with real time $\mathcal{DL}^{\mathrm{N}}$ reasoning. This is the first time such performance is reached over nonmonotonic KBs of this size: more than 20K concept names and over 30K inclusions.[7] Our approach brings technology closer to practical nonmonotonic reasoning with very large KBs. Only the random dependencies introduced by synthetic DIs, combined with numerous restrictions of role ranges to normal individuals, can raise response time over 40 seconds; in most of the other cases, computation time remains below 2 seconds.

We are currently exploring a more aggressive module extraction approach, capable of eliminating some of the normality concepts in Σ and related axioms. Besides improving performance over non-N-free KBs, a more powerful module extractor might enable the application of the combined Mod+Opt method to non-N-free $\mathcal{DL}^{\mathrm{N}}$ knowledge bases, by removing all normality concepts from $\mathcal{KB}$ before Opt is applied.

We are also planning to adopt a different module extractor [24] that is promising to be faster than the OWLAPI implementation.

Last but not least, we are progressively extending the set of experiments by covering the missing cases and by widening the benchmark set, using real ontologies different from GO as well as thoroughly synthetic ontologies.

[7] Good results have been obtained also for KBs with ∼5200 inclusions under rational closure semantics [9,10].

Acknowledgments. The authors would like to thank the reviewers for their valuable comments and suggestions. This work has been partially supported by the PRIN project Security Horizons.

References

1. Baader, F., Hollunder, B.: Embedding defaults into terminological knowledge representation formalisms. J. Autom. Reasoning **14**(1), 149–180 (1995)
2. Baader, F., Hollunder, B.: Priorities on defaults with prerequisites, and their application in treating specificity in terminological default logic. J. Autom. Reasoning **15**(1), 41–68 (1995)
3. Bonatti, P.A., Faella, M., Sauro, L.: $\mathcal{EL}$ with default attributes and overriding. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 64–79. Springer, Heidelberg (2010)
4. Bonatti, P.A., Faella, M., Sauro, L.: Adding default attributes to EL++. In: Burgard, W., Roth, D. (eds.) AAAI. AAAI Press (2011)
5. Bonatti, P.A., Faella, M., Sauro, L.: Defeasible inclusions in low-complexity DLs. J. Artif. Intell. Res. (JAIR) **42**, 719–764 (2011)
6. Bonatti, P.A., Lutz, C., Wolter, F.: The complexity of circumscription in DLs. J. Artif. Intell. Res. (JAIR) **35**, 717–773 (2009)
7. Bonatti, P.A., Petrova, I.M., Sauro, L.: Optimizing the computation of overriding. ArXiv e-prints, July 2015
8. Bonatti, P.A., Petrova, I.M., Sauro, L.: A new semantics for overriding in description logics. Artif. Intell. **222**, 1–48 (2015). http://www.sciencedirect.com/science/article/pii/S0004370215000028
9. Casini, G., Meyer, T., Moodley, K., Nortjé, R.: Relevant closure: a new form of defeasible reasoning for description logics. In: Fermé, E., Leite, J. (eds.) JELIA 2014. LNCS, vol. 8761, pp. 92–106. Springer, Heidelberg (2014)
10. Casini, G., Meyer, T., Moodley, K., Varzinczak, I.J.: Towards practical defeasible reasoning for description logics. In: Eiter, T., Glimm, B., Kazakov, Y., Krötzsch, M. (eds.) Description Logics. CEUR Workshop Proceedings, vol. 1014, pp. 587–599. CEUR-WS.org (2013)
11. Casini, G., Straccia, U.: Rational closure for defeasible description logics. In: Janhunen, T., Niemelä, I. (eds.) JELIA 2010. LNCS, vol. 6341, pp. 77–90. Springer, Heidelberg (2010)
12. Casini, G., Straccia, U.: Defeasible inheritance-based description logics. J. Artif. Intell. Res. (JAIR) **48**, 415–473 (2013)
13. Donini, F.M., Nardi, D., Rosati, R.: Description logics of minimal knowledge and negation as failure. ACM Trans. Comput. Log. **3**(2), 177–225 (2002)
14. Drabent, W., Eiter, T., Ianni, G., Krennwallner, T., Lukasiewicz, T., Małuszyński, J.: Hybrid reasoning with rules and ontologies. In: Bry, F., Małuszyński, J. (eds.) Semantic Techniques for the Web. LNCS, vol. 5500, pp. 1–49. Springer, Heidelberg (2009)
15. Baader, F., Calvanese, D., Mcguinness D., Nardi, D., Patel Schneider, P.: The description logic handbook, theory, implementation, and applications (2nd edition). In: The Description Logic Handbook, pp. 555–555. Cambridge University Press, Cambridge (2010)

16. Ghilardi, S., Lutz, C., Wolter, F.: Did I damage my ontology? a case for conservative extensions in description logics. In: Doherty, P., Mylopoulos, J., Welty, C. (eds.) Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2006), pp. 187–197. AAAI Press (2006)
17. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.L.: A non-monotonic description logic for reasoning about typicality. Artif. Intell. **195**, 165–202 (2013)
18. Giordano, L., Olivetti, N., Gliozzi, V., Pozzato, G.L.: ALC + T: a preferential extension of description logics. Fundam. Inform. **96**(3), 341–372 (2009)
19. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. J. Artif. Intell. Res. (JAIR) **31**, 273–318 (2008)
20. Kazakov, Y., Klinov, P.: Incremental reasoning in EL+ without bookkeeping. In: Description Logics, pp. 294–315 (2013)
21. Kontchakov, R., Wolter, F., Zakharyaschev, M.: Can you tell the difference between dl-lite ontologies? In Brewka, G., Lang, J. (eds.) Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, 16–19 September 2008, pp. 285–295. AAAI Press (2008)
22. Lukasiewicz, T.: Expressive probabilistic description logics. Artif. Intell. **172**(6–7), 852–883 (2008)
23. Lutz, C., Walther, D., Wolter, F.: Conservative extensions in expressive description logics. In: Veloso, M.M. (ed.) Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI 2007, Hyderabad, India, 6–12 January 2007, pp. 453–458 (2007)
24. Martin-Recuerda, F., Walther, D.: Axiom dependency hypergraphs for fast modularisation and atomic decomposition. In: Bienvenu, M., Ortiz, M., Rosati, R., Simkus, M. (eds.) Proceedings of the 27th International Workshop on Description Logics (DL 2014). CEUR Workshop Proceedings, vol. 1193, pp. 299–310 (2014)
25. Rector, A.L.: Defaults, context, and knowledge: Alternatives for OWL-indexed knowledge bases. In: Pacific Symposium on Biocomputing, pp. 226–237. World Scientific (2004)
26. Sandewall, E.: Defeasible inheritance with doubt index and its axiomatic characterization. Artif. Intell. **174**(18), 1431–1459 (2010)
27. Sattler, U., Schneider, T., Zakharyaschev, M.: Which kind of module should I extract? In: Grau, B.C., Horrocks, I., Motik, B., Sattler, U. (eds.) Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK, 27–30 July 2009, vol. 477. CEUR Workshop Proceedings. CEUR-WS.org (2009)
28. Stevens, R., Aranguren, M.E., Wolstencroft, K., Sattler, U., Drummond, N., Horridge, M., Rector, A.L.: Using OWL to model biological knowledge. Int. J. Man Mach. Stud. **65**(7), 583–594 (2007)
29. Woo, T.Y.C., Lam, S.S.: Authorizations in distributed systems: A new approach. J. Comput. Secur. **2**(2–3), 107–136 (1993)