Towards Practical Uniform Interpolation and Forgetting for $\mathcal{ALC}$ TBoxes

Michel Ludwig[1,2] and Boris Konev[1]

[1] Department of Computer Science, University of Liverpool, United Kingdom
Konev@liverpool.ac.uk

[2] Institute for Theoretical Computer Science, TU Dresden, Germany
Center for Advancing Electronics Dresden
michel@tcs.inf.tu-dresden.de

Abstract We develop a clausal resolution-based approach for computing uniform interpolants of TBoxes formulated in the description logic $\mathcal{ALC}$ when such uniform interpolants exist. We also present an experimental evaluation of our approach and its applications to concept forgetting, ontology obfuscation and logical difference on real-life $\mathcal{ALC}$ ontologies. Our results indicate that in many practical cases a uniform interpolant exists and can be computed with the presented algorithm.

1 Introduction

Ontologies or *TBoxes* expressed in Description Logics (DL) provide a common vocabulary for a domain of interest together with a description of the meaning of the terms built from the vocabulary and of the relationships between them. Modern applications of ontologies, especially in the biological, medical, or healthcare domain, often demand large and complex ontologies; for example, the National Cancer Institute ontology (NCI) consists of more than 60 000 term definitions. For developing, maintaining, and deploying such large-scale ontologies it can be advantageous for ontology engineers to concentrate on specific parts of an ontology and ignore or *forget* the rest, for example, in the following scenarios. *Exhibiting hidden relations*: in addition to the explicitly stated relations between terms, additional relations can also be derived from ontologies with the help of reasoners. Such inferred relations are often harder to understand or debug. By forgetting everything but a handful of terms of interest, it then becomes possible to exhibit inferred relations that are hidden initially, which can simplify the understanding of the ontology structure. *Ontology obfuscation*: in software engineering, obfuscation [4] transforms a given program into a functionally equivalent one that is more difficult to read and understand for humans for the purpose of preventing reverse engineering. Forgetting can provide a similar function in the context of ontology engineering. Terms are often defined with the help of auxiliary terms which help to give structure to TBox inclusions but such a structure might be considered proprietary knowledge that should not be exposed, or it could simply be of little interest for ontology users. By forgetting

This research was supported by EPSRC grant EP/E065279/1.

these intermediate auxiliary terms, we obtain an ontology that is functionally equivalent, yet harder to read, understand, and modify by humans. *Logical difference*: when modifying an ontology, an ontology engineer usually wants to ensure that her changes do not interfere with the meaning of the terms outside the fragment under consideration. Such a correctness guarantee can be achieved by checking the equivalence of the ontologies resulting from forgetting the terms under consideration before and after the changes occurred. Further applications of forgetting can be found in [11, 12].

Ignoring parts of an ontology can be formalised with the help of *predicate forgetting* and its dual *uniform interpolation*, which have both been extensively studied in the AI and DL literature [3,6,8,11,13,16–18]. However, to the best of our knowledge, previous research in this area in the setting of DL mainly concentrates on theoretical foundations of forgetting and, except for practical work on lightweight description logics [11], we are not aware of any attempts to compute uniform interpolants for real-life ontologies in practice. This could be partly explained by the high computational complexity of this task and by the fact that uniform interpolants do not always exist.

In this paper we develop an algorithm based on clausal resolution for computing uniform interpolants of TBoxes formulated in the description logic $\mathcal{ALC}$ which can preserve all the consequences that do not make use of some given *concept names*. We also present an experimental evaluation of our approach which demonstrates that in many practical cases uniform interpolants exist and that they can be computed with our algorithm. All missing proofs can be found in the full version of this paper, which is available from `http://lat.inf.tu-dresden.de/~michel/publications/`.

2 Preliminaries

We start with introducing the description logic $\mathcal{ALC}$. Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be countably infinite and mutually disjoint sets of *concept names* and *role names*. *$\mathcal{ALC}$-concepts* are built according to the following syntax rule

$$C ::= \quad A \quad | \quad \top \quad | \quad \neg C \quad | \quad \exists r.C \quad | \quad C \sqcap D,$$

where $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$. As usual, other $\mathcal{ALC}$ concept constructors are introduced as abbreviations: $\bot$ stands for $\neg\top$, $C \sqcup D$ stands for $\neg(\neg C \sqcap \neg D)$ and $\forall r.C$ stands for $\neg\exists r.\neg C$. An $\mathcal{ALC}$-TBox $\mathcal{T}$ is a finite set of $\mathcal{ALC}$-inclusions of the form $C \sqsubseteq D$, where C and D are $\mathcal{ALC}$-concepts. A concept equation $C \equiv D$ is an abbreviation for the two inclusions $C \sqsubseteq D$ and $D \sqsubseteq C$. An $\mathcal{ALC}$-TBox is *acyclic* if all its inclusions are of the form $A \sqsubseteq C$ and $A \equiv C$, where $A \in \mathsf{N_C}$ and C is an $\mathcal{ALC}$-concept, such that no concept name occurs more than once on the left-hand side and $\mathcal{T}$ contains no cycle in its definitions, that is, it does not contain inclusions $A_1 \bowtie C_1, \ldots, A_k \bowtie C_k$, where $\bowtie \in \{\sqsubseteq, \equiv\}$, such that A_{i+1} occurs in C_i, for $i = 1, \ldots, k-1$ and $A_k = A_1$.

A signature Σ is a finite subset of $\mathsf{N_C} \cup \mathsf{N_R}$. The signature of a concept C, denoted by $\mathsf{sig}(C)$, is the set of concept and role names that occur in C. If $\mathsf{sig}(C) \subseteq \Sigma$, we call C a Σ-concept. We assume that the two previous definitions also apply to concept inclusions/equations $C \bowtie D$ with $\bowtie \in \{\sqsubseteq, \equiv\}$ and to TBoxes $\mathcal{T}$. The size of a concept C is the length of the string that represents it, where concept names and role names are considered to be of length one. The size of an inclusion/equation $C \bowtie D$ with $\bowtie \in \{\sqsubseteq, \equiv\}$

is the sum of the sizes of C and D plus one. The size of a TBox $\mathcal{T}$ is the sum of the sizes of its inclusions.

The semantics of $\mathcal{ALC}$ is given by *interpretations* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where the *domain* $\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is a function mapping each concept name A to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role name r to a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The *extension* $C^{\mathcal{I}}$ of a concept C is defined by induction as follows:

$$\begin{array}{llll} \top^{\mathcal{I}} & := \Delta^{\mathcal{I}} & (\exists r.C)^{\mathcal{I}} & := \{\, d \in \Delta^{\mathcal{I}} \mid \exists e \in C^{\mathcal{I}} : (d,e) \in r^{\mathcal{I}} \,\} \\ (\neg C)^{\mathcal{I}} & := \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} & (C \sqcap D)^{\mathcal{I}} & := C^{\mathcal{I}} \cap D^{\mathcal{I}}. \end{array}$$

Then $\mathcal{I}$ *satisfies* a concept inclusion $C \sqsubseteq D$, in symbols $\mathcal{I} \models C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.

We say that an interpretation $\mathcal{I}$ is a *model of a TBox* $\mathcal{T}$ if $\mathcal{I} \models C \sqsubseteq D$ for all $C \sqsubseteq D \in \mathcal{T}$. An $\mathcal{ALC}$-inclusion $C \sqsubseteq D$ *follows from* (or is *entailed by*) a TBox $\mathcal{T}$ if every model of $\mathcal{T}$ is a model of $C \sqsubseteq D$, in symbols $\mathcal{T} \models C \sqsubseteq D$. We use $\models C \sqsubseteq D$ to denote that $C \sqsubseteq D$ follows from the empty TBox. Finally, a TBox $\mathcal{T}'$ *follows from* (or is *entailed by*) a TBox $\mathcal{T}$ if every model of $\mathcal{T}$ is a model of $\mathcal{T}'$, in symbols $\mathcal{T} \models \mathcal{T}'$.

We now introduce the main notion that we study in this paper.

Definition 1. *Let $\mathcal{T}$ be an $\mathcal{ALC}$-TBox and let $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ be a signature. We say that an $\mathcal{ALC}$-TBox $\mathcal{T}_\Sigma$ is a Σ-*uniform interpolant *of the TBox $\mathcal{T}$ iff $\mathsf{sig}(\mathcal{T}_\Sigma) \subseteq \Sigma$, $\mathcal{T} \models \mathcal{T}_\Sigma$, and for every $\mathcal{ALC}$ Σ-concept inclusion $C \sqsubseteq D$ with $\mathcal{T} \models C \sqsubseteq D$ it holds that $\mathcal{T}_\Sigma \models C \sqsubseteq D$.*

Uniform interpolation can be seen as the dual notion of *forgetting*: a TBox $\mathcal{T}_\Upsilon$ is the result of forgetting about a signature Υ in a TBox $\mathcal{T}$ iff $\mathcal{T}_\Upsilon$ is a uniform interpolant of $\mathcal{T}$ w.r.t. $\Sigma = \mathsf{sig}(\mathcal{T}) \setminus \Upsilon$. As the following example shows, uniform interpolants of $\mathcal{ALC}$-TBoxes do not always exist.

Example 2. Let $\mathcal{T} = \{A \sqsubseteq B,\ B \sqsubseteq C \sqcap \exists r.B\}$ and $\Sigma = \{A, C, r\}$. Then there does not exist a Σ-uniform interpolant of $\mathcal{T}$ as (in particular) the infinite number of consequences of the form $A \sqsubseteq \exists r.C$, $A \sqsubseteq \exists r.\exists r.C, \ldots$ cannot be captured by an $\mathcal{ALC}$-TBox $\mathcal{T}'$ with $\mathsf{sig}(\mathcal{T}') \subseteq \Sigma$. On the other hand, for $\mathcal{T}' = \{A \sqsubseteq B,\ B \sqsubseteq C \sqcap \exists r.B,\ D \equiv B\}$ and $\Sigma' = \{A, C, D, r\}$, a Σ'-uniform interpolant of $\mathcal{T}'$ is $\{A \sqsubseteq D,\ D \sqsubseteq C \sqcap \exists r.D\}$.

A 2-ExpTime-complete bound on the complexity for deciding the existence of a Σ-uniform interpolant in $\mathcal{ALC}$ and a worst-case triple-exponential procedure for computing a Σ-uniform interpolant if it exists, have been given in [13]. The proof of the upper bound proceeds by 'internalisation' of the TBox: if a Σ-uniform interpolant exists, then there exists a concept $C_{\mathcal{T}}$ of size double exponential in the size of $\mathcal{T}$ having the property that for any Σ-inclusion $C \sqsubseteq D$ it holds that $\mathcal{T} \models C \sqsubseteq D$ iff $\models C \sqcap C_{\mathcal{T}} \sqsubseteq D$. Then the authors compute C', a *Σ-concept uniform interpolant* [3] of $C_{\mathcal{T}}$, and show that $\top \sqsubseteq C'$ is a Σ-uniform interpolant of $\mathcal{T}$.

3 Computing Uniform Interpolants by $\mathcal{ALC}$-Resolution

The aim of our work is to investigate a practical approach for computing uniform interpolants when they exist. Note that the procedure given in [13] is inherently inefficient as it requires one to explicitly construct the double-exponential size internalisation $C_{\mathcal{T}}$ of a given TBox $\mathcal{T}$.

Our approach is to introduce a resolution-like calculus for $\mathcal{ALC}$ that derives consequences of a TBox $\mathcal{T}$ such that a concept inclusion $C \sqsubseteq D$ is entailed by $\mathcal{T}$ iff a contradiction can be derived from $\mathcal{T}$ and $C \sqcap \neg D$. Similarly to [8], we then show that any derivation can be restructured in such a way that inferences on selected concept names always precede inferences on other concept names. Then, if the signature Σ is such that $\mathsf{sig}(\mathcal{T}) \setminus \Sigma$ only contains concept names, we generate a set of Σ-consequences $\mathcal{T}'$ of $\mathcal{T}$ by applying the inference rules in a forward chaining manner such that for an arbitrary Σ-inclusion $C \sqsubseteq D$ a contradiction can be derived from $\mathcal{T}$ and $C \sqcap \neg D$ iff a contradiction can be derived from $\mathcal{T}'$ and $C \sqcap \neg D$. Thus, if the forward-chaining process terminates, $\mathcal{T}'$ is a Σ-uniform-interpolant for $\mathcal{T}$.

$\mathcal{ALC}$-Resolution. $\mathcal{ALC}$-resolution operates on $\mathcal{ALC}$ formulae in conjunctive normal form defined according to the following grammar (this is similar to [8]):

$$\begin{aligned} \text{Literal} &::= A \mid \neg A \mid \forall r.\text{Clause} \mid \exists r.\text{CNF} \\ \text{Clause} &::= \text{Literal} \mid \text{Clause} \sqcup \text{Clause} \mid \bot \\ \text{CNF} &::= \top \mid \text{Clause} \mid \text{Clause} \sqcap \text{CNF} \end{aligned}$$

To simplify the presentation, we assume that clauses are sets of literals and that CNF expressions are sets of clauses. Then $\bot$ corresponds to the empty clause and $\top$ to the empty set of clauses. In the following, the calligraphic letters $\mathcal{C}, \mathcal{D}, \mathcal{E}$ symbolise clauses and $\mathcal{F}, \mathcal{G}$ represent sets of clauses. Similarly to first-order formulae, every $\mathcal{ALC}$ concept can be transformed into an equivalent set of $\mathcal{ALC}$ clauses. The depth of a clause $\mathcal{C}$, $\mathrm{Depth}(\mathcal{C})$, is defined to be the maximal nesting depth of the quantifiers contained in $\mathcal{C}$.

We additionally assume that every clause is assigned a *type*. Clauses obtained from the clausification of TBox inclusions are of the type *universal*, and clauses resulting from the clausification of inclusions to be tested for entailment by the TBox are of the type *initial*. The type of a derived clause is determined by the types of the clauses from which it is derived and by the derivation rule that is used.

Example 3. The clausification of $\mathcal{T}$ from Example 2 produces three universal clauses: $\neg A \sqcup B,\ \neg B \sqcup C,\ \neg B \sqcup \exists r.B$.

We now introduce the two resolution calculi $\mathfrak{T}$ and $\mathfrak{T}^u$. The former calculus assumes the TBox to be empty, whereas the latter takes TBox inclusions into account. Thus, $\mathfrak{T}$ derives the empty clause from the set of initial clauses stemming from the clausification of an inclusion $C \sqsubseteq D$ iff $\models C \sqsubseteq D$; and $\mathfrak{T}^u$ derives the empty clause from the universal clauses stemming from the clausification of a TBox $\mathcal{T}$ and the initial clauses stemming from the clausification of an inclusion $C \sqsubseteq D$ iff $\mathcal{T} \models C \sqsubseteq D$.

The calculus $\mathfrak{T}$ is defined with the help of the relation $\Rightarrow_\alpha$ given in Fig. 1. For every $\alpha \in \mathsf{N_C} \cup \{\bot\}$, the relation $\Rightarrow_\alpha$ associates with a set of clauses $\mathcal{N}$ a new clause $\mathcal{C}$ which can be 'derived' from the set $\mathcal{N}$ by 'resolving' on α. $\mathfrak{T}$ now consists of the following two inference rules.

$$\frac{\mathcal{C}}{\mathcal{E}}\ (\text{if } \mathcal{C} \Rightarrow_\alpha \mathcal{E}) \qquad\qquad \frac{\mathcal{C} \quad \mathcal{D}}{\mathcal{E}}\ (\text{if } \mathcal{C}, \mathcal{D} \Rightarrow_\alpha \mathcal{E}),$$

where $\mathcal{C}, \mathcal{D}$, and $\mathcal{E}$ are initial clauses.

$$\begin{array}{rl}
(\text{rule } \bot) & \mathcal{C}'_1 \sqcup \forall r.\bot,\ \ \mathcal{C}'_2 \sqcup \exists r.\mathcal{F} \Longrightarrow_\bot \mathcal{C}'_1 \sqcup \mathcal{C}'_2 \\
(\text{rule } A) & \mathcal{C}'_1 \sqcup A,\ \ \mathcal{C}'_2 \sqcup \neg A \Longrightarrow_A \mathcal{C}'_1 \sqcup \mathcal{C}'_2 \\
(\text{rule } \forall\exists) & \mathcal{C}'_1 \sqcup \forall r.\mathcal{C}_1,\ \ \mathcal{C}'_2 \sqcup \exists r.(\mathcal{C}_2, \mathcal{F}) \Longrightarrow_\alpha \mathcal{C}'_1 \sqcup \mathcal{C}'_2 \sqcup \exists r.(\mathcal{C}_2, \mathcal{F}, \mathcal{C}_3) \\
 & \hfill \text{if } \mathcal{C}_1, \mathcal{C}_2 \Longrightarrow_\alpha \mathcal{C}_3 \\
(\text{rule } \forall\forall) & \mathcal{C}'_1 \sqcup \forall r.\mathcal{C}_1,\ \ \mathcal{C}'_2 \sqcup \forall r.\mathcal{C}_2 \Longrightarrow_\alpha \mathcal{C}'_1 \sqcup \mathcal{C}'_2 \sqcup \forall r.\mathcal{C}_3 \\
 & \hfill \text{if } \mathcal{C}_1, \mathcal{C}_2 \Longrightarrow_\alpha \mathcal{C}_3 \\
(\text{rule } \exists_1) & \mathcal{C}' \sqcup \exists r.(\mathcal{C}_1, \mathcal{F}) \Longrightarrow_\alpha \mathcal{C}' \sqcup \exists r.(\mathcal{C}_1, \mathcal{F}, \mathcal{C}_2) \\
 & \hfill \text{if } \mathcal{C}_1 \Longrightarrow_\alpha \mathcal{C}_2 \\
(\text{rule } \exists_2) & \mathcal{C}' \sqcup \exists r.(\mathcal{C}_1, \mathcal{C}_2, \mathcal{F}) \Longrightarrow_\alpha \mathcal{C}' \sqcup \exists r.(\mathcal{C}_1, \mathcal{C}_2, \mathcal{F}, \mathcal{C}_3) \\
 & \hfill \text{if } \mathcal{C}_1, \mathcal{C}_2 \Longrightarrow_\alpha \mathcal{C}_3 \\
(\text{rule } \forall) & \mathcal{C}' \sqcup \forall r.\mathcal{C}_1 \Longrightarrow_\alpha \mathcal{C}' \sqcup \forall r.\mathcal{C}_2 \\
 & \hfill \text{if } \mathcal{C}_1 \Longrightarrow_\alpha \mathcal{C}_2
\end{array}$$

Figure 1. Rules of $\Longrightarrow_\alpha$.

The calculus $\mathfrak{T}^u$ operates initial and universal clauses and also consists of two rules:

$$\frac{\mathcal{C}}{\mathcal{E}}\ (\text{if } \mathcal{C} \Rightarrow_\alpha \mathcal{E}) \qquad\qquad \frac{\mathcal{C}' \quad \mathcal{D}}{\mathcal{E}'}\ (\text{if } \mathcal{C}', \mathcal{D} \Rightarrow^u_\alpha \mathcal{E}'),$$

where $\mathcal{C}, \mathcal{C}', \mathcal{D}$ are initial or universal clauses, and $\mathcal{C}', \mathcal{D} \Rightarrow^u_\alpha \mathcal{E}'$ holds iff either $\mathcal{C}', \mathcal{D} \Rightarrow_\alpha \mathcal{E}'$, or $\mathcal{D}$ is a universal clause and there exist role names $r_1, \ldots, r_n \in \mathsf{N_R}$ $(n \geq 1)$ such that $\mathcal{C}', \forall r_1. \ldots \forall r_n.\mathcal{D} \Rightarrow_\alpha \mathcal{E}'$. (Intuitively, the calculus $\mathfrak{T}^u$ allows for inferences with universal clauses at arbitrary nesting levels of quantifiers, which the calculus $\mathfrak{T}$ does not.) Then $\mathcal{E}$ is a universal clause if $\mathcal{C}$ is a universal clause, and an initial clause otherwise. Similarly, $\mathcal{E}'$ is a universal clause if both $\mathcal{C}'$ and $\mathcal{D}$ are universal clauses, and an initial clause otherwise.

Example 4. For the universal clauses from Example 3, we have for instance,

$$\neg A \sqcup B, \neg B \sqcup \exists r.B \ \Rightarrow_B \neg A \sqcup \exists r.B \text{ by (rule } A).$$

So, the universal clause $\neg A \sqcup \exists r.B$ is derivable by $\mathfrak{T}^u$ from $\neg A \sqcup B$ and $\neg B \sqcup \exists r.B$. As $\neg B \sqcup C$ is a universal clause and

$$\neg B \sqcup \exists r.B, \forall r.\neg B \sqcup C \ \Rightarrow_B \neg B \sqcup \exists r.(B, C) \text{ by (rule } \forall\exists),$$

the universal clause $\neg B \sqcup \exists r.(B, C)$ is derivable by $\mathfrak{T}^u$ from $\neg B \sqcup \exists r.B$ and $\neg B \sqcup C$. By applying the inference rules to old and newly generated clauses, one can conclude that the universal clauses $\neg A \sqcup \exists r.(B, C)$ and $\neg A \sqcup \exists r.(B, \exists r.B)$ are also derivable by $\mathfrak{T}^u$ from $\mathcal{N} = \{\neg A \sqcup B,\ \neg B \sqcup C,\ \neg B \sqcup \exists r.B\}$.

For $x \in \{\mathfrak{T}, \mathfrak{T}^u\}$, a x-derivation (tree) Δ built from a set of clauses $\mathcal{N}$ is a finite binary tree where each leaf is labelled with a clause from $\mathcal{N}$ and each non-leaf node n is labelled with a clause $\mathcal{C}$ such that $\mathcal{C}$ results from an x-inference on the parent(s) of n in Δ. We say that Δ is a derivation of a clause $\mathcal{C}$ if the root of Δ is labelled with $\mathcal{C}$. A derivation of the empty clause is called a *refutation*. Every path $n_1, \ldots, n_m$ of nodes in Δ where n_1 is a leaf node and n_m is the root node induces an *inference path* $\alpha_2, \ldots, \alpha_m$ where $\alpha_i \in \mathsf{N_C} \cup \{\bot\}$ $(2 \leq i \leq m)$ denotes the concept name, or $\bot$, which has been resolved upon to obtain the clause that is the label of the node n_i. For a signature $\Upsilon \subseteq \mathsf{N_C}$ and a strict total order $\succ \subseteq \Upsilon \times \Upsilon$, a derivation Δ is a $(x, \Upsilon, \succ)$-derivation if for every inference path $\alpha_1, \ldots, \alpha_n$ of Δ (with $\alpha_i \in \mathsf{N_C} \cup \{\bot\}$ for every $1 \le i \le n$) there exists $0 \le k \le n$ such that $\{\alpha_1, \ldots, \alpha_k\} \subseteq \Upsilon$, $\alpha_j \succ \alpha_{j+1}$ or $\alpha_j = \alpha_{j+1}$ for every $1 \le j < k$, and $\alpha_j \notin \Upsilon$ for every $k < j \le n$.

The calculus $\mathfrak{T}$ is a direct adaption of the calculus for the modal logic K introduced in [7] to $\mathcal{ALC}$, and so $\models \top \sqsubseteq C$ iff the empty clause can be derived from the clausification of $\neg C$. However, as we show in the full version of the paper, the proof given in [7] of the fact that any derivation in $\mathfrak{T}$ can be reordered so that inferences on concept names from Υ always precede inferences on other concept names, or $\bot$, appears to have some gaps. We prove the following theorem which extends the results stated in [8].

Theorem 5 **($\mathfrak{T}$-Completeness).** *Let $\Upsilon \subseteq \mathsf{N_C}$, let $\succ \subseteq \Upsilon \times \Upsilon$ be a strict total order on Υ and let C be an $\mathcal{ALC}$ concept. Then it holds that $\models C \sqsubseteq D$ iff there exists a $(\mathfrak{T}, \Upsilon, \succ)$-derivation of the empty clause from the initial clauses* $\mathrm{Cls}(C \sqcap \neg D)$.

To prove completeness for $\mathfrak{T}^u$, we observe the following link between derivations in $\mathfrak{T}$ and $\mathfrak{T}^u$. Let $\mathcal{N}$ be a set of clauses and let

$$\mathrm{Univ}_0(\mathcal{N}) = \mathcal{N}; \quad \mathrm{Univ}_{i+1}(\mathcal{N}) = \mathrm{Univ}_i(\mathcal{N}) \cup \bigcup_{r \in \mathsf{N_R} \cap \mathsf{sig}(\mathcal{N})} \{\, \forall r.\mathcal{D} \mid \mathcal{D} \in \mathrm{Univ}_i(\mathcal{N}) \,\}$$

and $\mathrm{Univ}(\mathcal{N}) = \bigcup_{i \ge 0} \mathrm{Univ}_i(\mathcal{N})$.

Theorem 6. *Let $\mathcal{M}$ be a set of initial clauses and let $\mathcal{N}$ be a set of universal clauses. Additionally, let Δ be a $(\mathfrak{T}, \Upsilon, \succ)$-refutation from $\mathcal{M} \cup \mathrm{Univ}(\mathcal{N})$ such that there exists $n \in \mathbb{N}$ with $\mathrm{Depth}(\mathcal{C}) \le n$ for every $\mathcal{C} \in$ Clauses(Δ). Then there exists a $(\mathfrak{T}^u, \Upsilon, \succ)$-derivation Δ^u of the empty clause from $\mathcal{M} \cup \mathcal{N}$ such that $\mathrm{Depth}(\mathcal{C}) \le n$ for every $\mathcal{C} \in$ Clauses(Δ^u).*

We then use Theorem 5 and the fact that every $\mathcal{ALC}$-TBox can be internalised. Notice that the actual TBox internalisation $C_{\mathcal{T}}$ does not have to be computed as it is only used for the proof of completeness.

Corollary 7 **($\mathfrak{T}^u$-Completeness).** *Let $\mathcal{T}$ be an $\mathcal{ALC}$-TBox, let $\Upsilon \subseteq \mathsf{N_C}$, let $\succ \subseteq \Upsilon \times \Upsilon$ be a strict total order on Υ and let C be an $\mathcal{ALC}$ concept. Then it holds that $\mathcal{T} \models C \sqsubseteq D$ iff there exists a $(\mathfrak{T}^u, \Upsilon, \succ)$-derivation of the empty clause from the universal clauses* $\mathrm{Cls}(\mathcal{T})$ *and the initial clauses* $\mathrm{Cls}(C \sqcap \neg D)$.

Computing Uniform Interpolants. The procedure UNIFORMINTERPOLANT depicted in Algorithm 1 takes as input an $\mathcal{ALC}$-TBox $\mathcal{T}$, a signature $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ such that $\Sigma \cap \mathsf{N_R} = \mathsf{sig}(\mathcal{T}) \cap \mathsf{N_R}$ and a strict total order $\succ \subseteq \Upsilon \times \Upsilon$ over $\Upsilon = \mathsf{sig}(\mathcal{T}) \setminus \Sigma$. Following the outline of [8], after the clausification of $\mathcal{T}$, the procedure iterates over the concept names contained in Υ in descending order according to the relation $\succ$. In each iteration all possible $\mathfrak{T}^u$-inferences on the current concept name $A \in \Upsilon$ from the clause set $\mathcal{N}$ obtained in the previous iteration are computed. Finally, after iterating over all the concept names from $\Upsilon = \mathsf{sig}(\mathcal{T}) \setminus \Sigma$, the operator 'Supp' is applied on the resulting clauses, which replaces all occurrences of Υ concept names in clauses with $\top$ and then simplifies the resulting CNF.

Algorithm 1

```
1:  procedure UNIFORMINTERPOLANT($\mathcal{T}$, $\Sigma$, $\succ$)
2:      $\Upsilon := \mathsf{sig}(\mathcal{T}) \setminus \Sigma$
3:      $\mathcal{N} := \mathrm{Cls}(\mathcal{T})$
4:      while $\Upsilon \neq \emptyset$ do
5:          $A := \max_{\succ}(\Upsilon)$
6:          $\mathcal{N} := \mathrm{Res}^{\infty}_{\mathfrak{I}^u,\{A\}}(\mathcal{N})$
7:          $\Upsilon := \Upsilon \setminus \{A\}$
8:      end while
9:      return $\mathcal{F}_{\Sigma}(\mathcal{T}) = \mathrm{Supp}(\mathsf{sig}(\mathcal{T}) \setminus \Sigma, \mathcal{N})$
10: end procedure
```

Example 8. For the clauses derived in Example 4, $\mathrm{Supp}(\{B\}, \neg A \sqcup C) = A \sqcup C$, $\mathrm{Supp}(\{B\}, \neg A \sqcup \exists r.B) = \neg A \sqcup \exists r.\top$, $\mathrm{Supp}(\{B\}, \neg A \sqcup \exists r.(B, C)) = \neg A \sqcup \exists r.C$.

One can show that if Algorithm 1 terminates, for all $\mathcal{ALC}$ Σ-concepts C, D such that there exists a $(\mathfrak{I}^u, \Upsilon, \succ)$-refutation Δ^u from the universal clauses $\mathrm{Cls}(\mathcal{T})$ and the initial clauses $\mathrm{Cls}(C \sqcap \neg D)$ it holds that $\mathcal{F}_{\Sigma}(\mathcal{T}) \models C \sqsubseteq D$. Thus, it follows from Corollary 7 that if Algorithm 1 terminates, it computes a Σ-uniform interpolant of $\mathcal{T}$. However, Algorithm 1 does not terminate if a uniform interpolant does not exist. For example, when applied to $\mathcal{T}$ from Example 2, Algorithm 1 can generate, among others, the infinite sequence of universal clauses $\neg A \sqcup \exists r.C$, $\neg A \sqcup \exists r.(C, \exists r.C), \ldots$ and so it does not terminate. Moreover, as $\mathcal{T}$ from Example 2 is a subset of $\mathcal{T}'$, and so $\mathrm{Cls}(\mathcal{T}) \subseteq \mathrm{Cls}(\mathcal{T}')$, Algorithm 1 will derive, among others, the same clauses when it is applied on $\mathcal{T}'$. Thus, in some cases Algorithm 1 does not terminate even though a uniform interpolant exists.

To guarantee termination on all inputs, we focus on the notion of depth-bounded uniform interpolation (related to the notion of 'bounded forgetting' [19]). Let $\mathcal{T}$ be an $\mathcal{ALC}$-TBox and let $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ be a signature. We say that an $\mathcal{ALC}$-TBox $\mathcal{T}_{\Sigma}$ is a *depth n-bounded uniform interpolant* of the TBox $\mathcal{T}$ w.r.t. Σ iff $\mathsf{sig}(\mathcal{T}_{\Sigma}) \subseteq \Sigma$, $\mathcal{T} \models \mathcal{T}_{\Sigma}$, and for every $\mathcal{ALC}$ Σ-concept inclusion $C \sqsubseteq D$ with $\mathcal{T} \models C \sqsubseteq D$ and $\max\{\mathrm{Depth}(C), \mathrm{Depth}(D)\} \leq n$ it holds that $\mathcal{T}_{\Sigma} \models C \sqsubseteq D$. Let $\mathcal{F}_{\Sigma,m}(\mathcal{T})$ be the outcome of Algorithm 1 where in Step 6 only clauses up to depth m are generated. The following example shows that it might be necessary to consider intermediate clauses of a depth $m > n$ in order to preserve all the Σ-consequences of depth n entailed by $\mathcal{T}$.

Example 9. Let $\mathcal{T} = \{\neg A \sqcup \exists r.C, \neg C \sqcup \exists s.\top, B \sqcup \forall s.\bot\}$, $\Sigma = \{A, B, r, s\}$, $\Upsilon = \{C\}$ and $\succ = \emptyset$. Then every $(\mathfrak{I}^u, \Upsilon, \succ)$-refutation from the universal clauses $\mathrm{Cls}(\mathcal{T})$ and the initial clauses $\{A, \forall r.\neg B\}$ derives the clause $\neg A \sqcup \exists r.(C, \exists s.\top)$.

We establish, however, that by choosing the maximal depth of derived clauses appropriately, the procedure depicted in Algorithm 1 computes uniform interpolants that preserve consequences up to a specified depth n.

Theorem 10. *Let $\mathcal{T}$ be an $\mathcal{ALC}$-TBox, $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ a signature such that $\Sigma \cap \mathsf{N_R} = \mathsf{sig}(\mathcal{T}) \cap \mathsf{N_R}$, and let $n \geq 0$. Set $m = n + 2^{|\mathsf{sub}(\mathrm{Cls}(\mathcal{T}))|+1} + \max\{\, \mathrm{Depth}(\mathcal{C}) \mid \mathcal{C} \in \mathrm{Cls}(\mathcal{T}) \,\}$. Then it holds that $\mathcal{F}_{\Sigma,m}(\mathcal{T})$ is a depth n-bounded uniform interpolant of the TBox $\mathcal{T}$ w.r.t. Σ.*

We can combine this result with the bound on the size of uniform interpolants in [13]: for any $\mathcal{ALC}$-TBox $\mathcal{T}$ and signature Σ (with $\Sigma \cap \mathsf{N_R} = \mathsf{sig}(\mathcal{T}) \cap \mathsf{N_R}$), if a Σ-uniform interpolant of $\mathcal{T}$ exists, there exists m, which can be computed based on the bound in Theorem 10 and the results of [13], such that $\mathcal{F}_{\Sigma,m}(\mathcal{T})$ is a Σ-uniform interpolant of $\mathcal{T}$.

The bound in Theorem 10 can be significantly improved if the TBox is acyclic. For an acyclic $\mathcal{ALC}$-TBox $\mathcal{T}$ we define $\mathrm{ExpansionDepth}(\mathcal{T}) = \max\{\, \mathrm{Depth}(A[\mathcal{T}]) \mid A \in \mathsf{sig}(\mathcal{T})\,\}$, where $A[\mathcal{T}]$ denotes the concept obtained by exhaustively replacing every concept B with C_B if $B \sqsubseteq C_B \in \mathcal{T}$ or $B \equiv C_B \in \mathcal{T}$.

Theorem 11. *Let $\mathcal{T}$ be an acyclic $\mathcal{ALC}$ TBox, $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ a signature such that $\Sigma \cap \mathsf{N_R} = \mathsf{sig}(\mathcal{T}) \cap \mathsf{N_R}$, and let $n \geq 0$. Set $m = \mathrm{ExpansionDepth}(\mathcal{T}) + n$. Then it holds that $\mathcal{F}_{\Sigma,m}(\mathcal{T})$ is a uniform interpolant limited to consequence depth n of the TBox $\mathcal{T}$ w.r.t. Σ.*

Note that in the description logic $\mathcal{EL}$ (i.e. the fragment of $\mathcal{ALC}$ that does not allow $\bot$, negation, disjunction, or universal quantification) the acyclicity of a TBox guarantees the existence of uniform interpolants [11] for any signature Σ. Interestingly, this is not so in the case of $\mathcal{ALC}$. Moreover, as the following example shows, there exists an acyclic $\mathcal{EL}$-TBox $\mathcal{T}$ and a signature Σ for which no $\mathcal{ALC}$ Σ-uniform interpolant exists.

Example 12. Consider $\Sigma = \{A, A_0, A_1, A_2, E, r\}$ and $\mathcal{T} = \{A \sqsubseteq \exists r.B,\ A_0 \sqsubseteq \exists r.(A_1 \sqcap B),\ E \equiv A_1 \sqcap B \sqcap \exists r.(A_2 \sqcap B)\}$. Then for every $n \geq 0$

$$\mathcal{T} \models A_0 \sqcap \bigsqcap_{i=1}^{n} \underbrace{\forall r. \ldots \forall r.}_{i} (A \sqcap \neg E \sqcap (A_1 \sqcup A_2)) \sqsubseteq \underbrace{\exists r. \ldots \exists r.}_{n} A_1.$$

This infinite sequence of $\mathcal{ALC}$ consequences of $\mathcal{T}$ cannot be captured by any $\mathcal{ALC}$ Σ-TBox $\mathcal{T}'$, which can be proved formally using the criterion $(*_m)$ of Theorem 9 in [13].

4 Case Study

We have implemented a prototype of an inference computation architecture using the calculus $\mathfrak{T}^u$ and the inference relation $\Rightarrow_\alpha$ in Java. It has turned out that our initial implementation of Algorithm 1 did not perform well in practice. This is in particular due to the fact that clauses can contain sets $\mathcal{F}$ of other clauses in existential literals $\exists r.\mathcal{F}$, which renders all the possible inferences on clauses from $\mathcal{F}$ 'explicit'. For example, if we resolve the universal clause which just consists of the existential literal $\exists r.(A)$ with the universal clauses $\neg A \sqcup B_1, \ldots, \neg A \sqcup B_n$ on the concept name A, then not only the clauses $\exists r.(A, B_1)$, $\exists r.(A, B_2)$, $\exists r.(A, B_3)$,... could be derived but all clauses of the form $\exists r.(A, \mathcal{G})$, where $\mathcal{G}$ is a subset of $\{B_1, \ldots, B_n\}$.

A common technique to reduce the number of inferences that have to be made is to use forward- and backward deletion of subsumed clauses [2]. However, it is known [1] that the subsumption lemma (stating that if a clause $\mathcal{E}$ results from an inference involving two clauses $\mathcal{C}$ and $\mathcal{D}$, and if there exist clauses $\mathcal{C}'$, $\mathcal{D}'$ such that $\mathcal{C}'$ subsumes $\mathcal{C}$ and $\mathcal{D}'$ subsumes $\mathcal{D}$, then either $\mathcal{E}$ is subsumed by one of $\mathcal{C}'$, $\mathcal{D}'$, or a clause $\mathcal{E}'$ can be derived from $\mathcal{C}'$ and $\mathcal{D}'$ such that $\mathcal{E}'$ subsumes $\mathcal{E}$) does not hold even in the modal logic K for the standard *minimal subsumption relation* $\leq_s$ and $\Rightarrow_\alpha$. To be able to prove that one can safely discard subsumed clauses, we have modified the inference relation $\Rightarrow_\alpha$ by introducing the following additional rule

$$(\text{rule } \exists_f) \qquad \mathcal{C}_1 \sqcup \forall r.\mathcal{D},\ \ \mathcal{C}_2 \sqcup \exists r.\mathcal{F} \Longrightarrow_{\exists_f} \mathcal{C}_1 \sqcup \mathcal{C}_2 \sqcup \exists r.(\mathcal{F}, \mathcal{D}).$$

We will denote the resulting inference relation by $\Rightarrow^f_\alpha$ with $\alpha \in \mathsf{N_C} \cup \{\bot, \exists_f\}$. One can then prove that a variant of the subsumption lemma holds for the relations $\leq_s$ and $\Rightarrow^f_\alpha$, which allows us to employ forward- and backward deletion of subsumed clauses in our implementation.

In order to further speed up computations, we first extract the locality-based $\top\bot^*$ $\varSigma$-module [5, 14] for a given TBox $\mathcal{T}$. The locality-based module entails the same $\varSigma$-inclusions as the TBox $\mathcal{T}$ but it is often considerably smaller in size. We also rely on ontologies to have structure: if a concept name occurs in several inclusions, it is likely that it occurs in the same syntactic pattern. Thus,

1. If the clause set contains some clauses $\mathcal{C}_1 \sqcup \mathcal{D}_\varUpsilon, \ldots, \mathcal{C}_m \sqcup \mathcal{D}_\varUpsilon$ such that for every $1 \leq i \leq m$ we have $\mathsf{sig}(\mathcal{C}_i) \cap \varUpsilon = \emptyset$, we rewrite them into $X \sqcup \mathcal{D}_\varUpsilon$, where $X \equiv \mathcal{C}_1 \sqcap \ldots \sqcap \mathcal{C}_m$, perform forgetting on $\varUpsilon$ symbols and then replace X with its definition.
2. If the clause set contains a clause $\mathcal{C} \sqcup \exists r.(\mathcal{F}_\varUpsilon, \mathcal{G}_1) \sqcup \ldots \sqcup \exists r.(\mathcal{F}_\varUpsilon, \mathcal{G}_m)$ where $\mathsf{sig}(\mathcal{G}_i) \cap \varUpsilon = \emptyset$ for every $1 \leq i \leq m$, we rewrite it into $\mathcal{C} \sqcup \exists r.(\mathcal{F}_\varUpsilon, Y)$, where $Y \equiv \mathcal{G}_1 \sqcup \ldots \sqcup \mathcal{G}_m$, perform forgetting on $\varUpsilon$ and then replace Y with its definition.

Experimental setting. All experiments were conducted on PCs equipped with an Intel Core i5-2500K CPU running at 3.30GHz. 15 GiB of RAM were allocated to the Java VM and an execution timeout of 60 CPU minutes was imposed on each problem. We only considered $\mathcal{ALC}$-fragments of ontologies, i.e. for a given $\mathcal{ALC}$-ontology $\mathcal{T}$ we removed all the axioms which contain non-$\mathcal{ALC}$ concept (or role) constructors to obtain its $\mathcal{ALC}$-fragment. We used Algorithm 1 to forget concept names one by one (i.e. for $\mathsf{sig}(\mathcal{T}) \setminus \varSigma = \{A_1, \ldots, A_n\}$, Algorithm 1 was applied iteratively on $A_1, \ldots, A_n$), and we did *not* impose a bound on the depth of clauses. After each run of Algorithm 1 we further simplified the resulting clause set $\mathcal{N}$ by removing clauses $\mathcal{D}$ for which there exists $\mathcal{C} \in \mathcal{N}$ with $\mathcal{C} \leq_s \mathcal{D}$. Thus, in all the experiments reported on in this section we computed *true $\varSigma$-uniform interpolants* (i.e. not a depth-bounded variant). The correctness of our extensions to Algorithm 1 can be shown by model-theoretic arguments.

Exhibiting Hidden Relations. We applied our uniform interpolation tool to compute uniform interpolants w.r.t. small signatures $\varSigma \subseteq \mathsf{sig}(\mathcal{T})$ with $\mathsf{sig}(\mathcal{T}) \cap \mathsf{N_R} = \varSigma \cap \mathsf{N_R}$ for a fragment of version 1.4 of the *Drug Interaction Knowledge Base*[3] (DIKB) and for a fragment of version 08.10e of the *National Cancer Institute Thesaurus*[4] (NCI) that are of expansion depth 3 (that is, we removed all the axioms from both ontologies that led to an expansion depth greater than 3). The resulting DIKB fragment is a small acyclic terminology that contains 120 concept names, 27 roles names, and 127 axioms. The NCI fragment is also an acyclic terminology with 53571 concept names, 78 role names and 62494 axioms (of which 2362 are of the form $A \equiv C$). For each considered sample size x and terminology $\mathcal{T}$ we generated 100 signatures $\varSigma$ by randomly choosing x concept names from $\mathsf{sig}(\mathcal{T})$ and by adding all the role names from $\mathsf{sig}(\mathcal{T})$ to $\varSigma$.

[3] Available from `http://bioportal.bioontology.org/ontologies/1672`
[4] Available from `http://evs.nci.nih.gov/ftp1/NCI_Thesaurus`

Ontology	$\lvert\Sigma \cap \mathsf{N_C}\rvert$	Successful Computations	Average Nr of Axioms	Average Maximal Axiom Size
DIKB v1.4	5	85	7.482	19.176
	10	60	14.033	24.933
	15	44	25.114	26.568
NCI v08.10e	50	65	21.369	30.538
	100	56	41.089	62.839
	150	41	63.146	104.756

Table 1. Computing Uniform Interpolants of DIKB and of NCI Limited to Expansion Depth 3

The results that we obtained for the two ontologies are shown in Table 1. The third column gives the number of signatures for which a uniform interpolant could be computed within the given time limit, and the average number of axioms and the average of the maximal size of the axioms contained in each uniform interpolant are shown in the subsequent columns. Most uniform interpolants could be computed within 60 seconds. In a few cases, however, the computations took up to 3487 seconds.

One can see that the number of successful computations decreased with increasing size of $\Sigma \cap \mathsf{N_C}$, which seems to be due to the fact that the $\top\bot^*$ Σ-modules then contain more symbols that lead to a large number of inferences. Most uniform interpolants that we have obtained are relatively small and contain a lot of expressions of the form $\exists r_1 \ldots \exists r_n.\top$. In some cases the process of forgetting certain intermediate concept names generated a few hundred clauses that were simplified or deleted in the remaining computation steps.

Ontology Obfuscation. As a proof of concept, we applied our uniform interpolation tool on (a fragment of) the *Lipid Ontology*[5] (LiPrO) to forget 45 concept names which are intermediate concept names in the ontology's induced concept hierarchy, i.e. those concept names group certain subconcepts together to give structure to the ontology. LiPrO is an acyclic terminology with 593 axioms, 574 concept names and one role name. The maximal size of an axiom is 50.

It then took 192 CPU seconds to compute the uniform interpolant, which contains 3415 axioms with a maximal size of 283. The uniform interpolant that we computed thus approximately contains 6 times more axioms than the original ontology and the maximal axiom size has increased by a factor of 6 as well. Notice that most of the original structure of the ontology has been destroyed while preserving all the consequences entailed by the retained concept names.

Logical Difference. We also used our implementation for the computation of the logical difference [9] between two versions of the NCI thesaurus on various signatures. The Σ-logical difference between $\mathcal{ALC}$-TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ is the set $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of all $\mathcal{ALC}$-concept inclusions $C \sqsubseteq D$ such that $\mathsf{sig}(C \sqsubseteq D) \subseteq \Sigma$, $\mathcal{T}_1 \models C \sqsubseteq D$, and $\mathcal{T}_2 \not\models C \sqsubseteq D$. Notice that if $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ is nonempty, it is infinite already.

It is easy to see that $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$ if, and only if, $\mathcal{T}_2 \models \mathcal{T}_1^{(\Sigma)}$ where $\mathcal{T}_1^{(\Sigma)}$ is a Σ-uniform interpolant of $\mathcal{T}_1$. Moreover, if $\mathcal{T}_2 \not\models \mathcal{T}_1^{(\Sigma)}$, every inclusion $C \sqsubseteq D \in \mathcal{T}_1^{(\Sigma)}$

[5] Available from `http://bioportal.bioontology.org/ontologies/1183`

$\lvert(\mathsf{sig}(\mathcal{T}) \setminus \Sigma)$ $\cap \mathsf{N_C}\rvert$	Successful Computations	Average Nr of Witnesses	Average Maximal Witness Size
50	100	1976.600	69.380
100	99	1975.939	74.969
150	95	1974.874	72.284

Table 2. Computing Uniform Interpolants of NCI-08.10e Limited to Expansion Depth 3

with $\mathcal{T}_2 \not\models C \sqsubseteq D$ is said to be a *witness* of $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$. In our experiments we used the reasoner FaCT++ v1.6.2 [15] to determine whether any axiom $C \sqsubseteq D \in \mathcal{T}_1^{(\Sigma)}$ is a witness of $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$. To make the experiments more challenging for the reasoner, we focused on comparing (a fragment of) NCI v08.10e (as $\mathcal{T}_1$) with (a fragment of) NCI v.08.09d (as $\mathcal{T}_2$), both limited to an expansion depth of 3, on large signatures Σ with $\Sigma \cap \mathsf{N_R} = \mathsf{sig}(\mathcal{T}) \cap \mathsf{N_R}$. In that way the computed uniform interpolants remain rather large as well. For each sample size $x \in \{50, 100, 150\}$ we again generated 100 signatures by randomly choosing $|\mathsf{sig}(\mathcal{T}) \cap \mathsf{N_C}| - x$ concept names from $\mathsf{sig}(\mathcal{T})$ and by including all the role names from $\mathsf{sig}(\mathcal{T})$. The results that we obtained are now shown in Table 2. Note that these results are not directly comparable with the logical difference for the description logics of the $\mathcal{EL}$ family [9, 10] as illustrated by Example 12.

One can observe that as size of $\mathsf{sig}(\mathcal{T}) \setminus \Sigma$ increases, i.e. more symbols have to be forgotten from the $\top\bot^*$ Σ-modules, the success rate dropped slightly. Overall, the average number of witnesses and the average maximal size of the witnesses remains comparable throughout the different sample sizes. Also, the axioms generated by the computation of the uniform interpolant did not pose a problem for FaCT++ as computing the logical difference for a given signature never took more than 20 seconds in our experiments.

5 Conclusion

In this paper we presented an approach based on clausal resolution for computing uniform interpolants of $\mathcal{ALC}$-TBoxes $\mathcal{T}$ w.r.t. signatures $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ that contain all the role names present in $\mathcal{T}$. We proved that whenever the saturation process under $\mathcal{ALC}$-resolution terminates, the algorithm computes a uniform interpolant. To guarantee termination on all inputs, we introduced a depth-bounded version of our algorithm. We showed that by choosing an appropriate bound on the depth of clauses, one can axiomatise all Σ-inclusions implied by the given TBox up to a specified depth. Combined with a known bound on the size of uniform interpolants, our depth-bounded procedure always computes a uniform interpolant if it exists.

In the second part of this paper we investigated how often our unrestricted resolution-based algorithm terminates with a uniform interpolant by applying our prototype implementation on a number of case studies. Our findings suggest that despite a high computational complexity uniform interpolants can be computed in many practical cases. The computation procedure could further benefit from better redundancy elimination techniques, which, together with extending our approach to forgetting role names, constitutes future work. It would also be interesting to explore proof strategies for our resolution calculi that guarantee termination when uniform interpolants exist.

References

1. Auffray, Y., Enjalbert, P., Hébrard, J.J.: Strategies for modal resolution: Results and problems. Journal of Automated Reasoning 6(1), 1–38 (1990)
2. Bachmair, L., Ganzinger, H.: Resolution theorem proving. In: Handbook of automated reasoning, vol. 1, chap. 2, pp. 19–99. Elsevier (2001)
3. ten Cate, B., Conradie, W., Marx, M., Venema, Y.: Definitorially complete description logics. In: Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2006). pp. 79–89. AAAI Press (2006)
4. Collberg, C.S., Thomborson, C.D., Low, D.: Manufacturing cheap, resilient, and stealthy opaque constructs. In: Proceedings of the 25th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '98). pp. 184–196. ACM (1998)
5. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: theory and practice. Journal of Artificial Intelligence Research (JAIR) 31, 273–318 (2008)
6. Eiter, T., Ianni, G., Schindlauer, R., Tompits, H., Wang, K.: Forgetting in managing rules and ontologies. In: Proceedings of the 2006 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2006). pp. 411–419. IEEE Computer Society (2006)
7. Enjalbert, P., del Cerro, L.F.: Modal resolution in clausal form. Theoretical Computer Science 65(1), 1–33 (1989)
8. Herzig, A., Mengin, J.: Uniform interpolation by resolution in modal logic. In: Proceedings of the 11th European Conference on Logics in Artificial Intelligence (JELIA 2008). Lecture Notes in Computer Science, vol. 5293, pp. 219–231. Springer (2008)
9. Konev, B., Walther, D., Wolter, F.: The logical difference problem for description logic terminologies. In: Proceedings of the 4th International Joint Conference on Automated Reasoning (IJCAR 2008). Lecture Notes in Computer Science, vol. 5195, pp. 259–274. Springer (2008)
10. Konev, B., Ludwig, M., Walther, D., Wolter, F.: The logical difference for the lightweight description logic $\mathcal{EL}$. Journal of Artificial Intelligence Research (JAIR) 44, 633–708 (2012)
11. Konev, B., Walther, D., Wolter, F.: Forgetting and uniform interpolation in large-scale description logic terminologies. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009). pp. 830–835 (2009)
12. Lutz, C., Seylan, I., Wolter, F.: An automata-theoretic approach to uniform interpolation and approximation in the description logic $\mathcal{EL}$. In: Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2012). AAAI Press (2012)
13. Lutz, C., Wolter, F.: Foundations for uniform interpolation and forgetting in expressive description logics. In: Walsh, T. (ed.) Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011). pp. 989–995 (2011)
14. Sattler, U., Schneider, T., Zakharyaschev, M.: Which kind of module should I extract? In: Proceedings of the 22nd International Workshop on Description Logics (DL 2009). CEUR Workshop Proceedings, vol. 477. CEUR-WS.org (2009)
15. Tsarkov, D., Horrocks, I.: FaCT++ Description Logic reasoner: System description. In: Proceedings of the Third International Joint Conference on Automated Reasoning (IJCAR 2006). Lecture Notes in Computer Science, vol. 4130, pp. 292–297. Springer (2006)
16. Wang, K., Wang, Z., Topor, R., Pan, J.Z., Antoniou, G.: Eliminating concepts and roles from ontologies in expressive descriptive logics. Computational Intelligence (Accepted for publication), DOI: 10.1111/j.1467-8640.2012.00442.x
17. Wang, Z., Wang, K., Topor, R., Pan, J.Z.: Forgetting Concepts in DL-Lite. In: Proceedings of the 5th European Semantic Web Conference (ESWC2008). Lecture Notes in Computer Science, vol. 5021, pp. 245–257. Springer (2008)

18. Wang, Z., Wang, K., Topor, R.W., Zhang, X.: Tableau-based forgetting in $\mathcal{ALC}$ ontologies. In: Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010). Frontiers in Artificial Intelligence and Applications, vol. 215, pp. 47–52 (2010)
19. Zhou, Y., Zhang, Y.: Bounded forgetting. In: Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011). AAAI Press (2011)