Forgetting and Uniform Interpolation for $\mathcal{ALC}$-Ontologies with ABoxes

Patrick Koopmann,* Renate A. Schmidt

The University of Manchester, UK
{koopmanp, schmidt}@cs.man.ac.uk

Abstract. We present a method to compute uniform interpolants of $\mathcal{ALC}$-ontologies with ABoxes. Uniform interpolants are restricted views of ontologies that only use a specified set of symbols, but share all entailments in that signature with the original ontology. This way, it allows to select or remove information from an ontology based on a signature, which has applications in privacy, ontology analysis and ontology reuse. We show that in general, uniform interpolants of $\mathcal{ALC}$-ontologies with ABoxes may require disjunctive statements or nominals in the ABox. An evaluation of the method suggests however, that in most practical cases uniform interpolants can be represented as a classical $\mathcal{ALC}$-ontology.

1 Introduction

We present a method to compute uniform interpolants of $\mathcal{ALC}$-ontologies with ABoxes. Modern applications, especially in bio-informatics or medical domains, often demand ontologies that cover a large vocabulary. With rising complexity, these ontologies become harder to maintain and modify. Uniform interpolation and forgetting deal with the problem of reducing the vocabulary used in an ontology in such a way that all entailments over the reduced vocabulary are preserved. In contrast to modules, uniform interpolants are completely in the desired signature and may contain different axioms than the input ontology.

There are a range of applications for uniform interpolation. One can use it to extract a part of an ontology for reuse, if only a subset of the terms defined in the ontology is interesting for an application [21]. It also provides a way to remove confidential information from an ontology to be shared among users with different privileges [7]. By forgetting concepts that give structure to an ontology, one can obfuscate it for other users [14]. Furthermore, uniform interpolants for small signatures help exhibiting hidden relations in the ontology [8]. Further applications are mentioned in [8, 16].

Uniform interpolation of TBoxes has been extensively studied in the description logic literature [22, 17, 16, 14, 11, 10, 9, 12], whereas forgetting symbols from ABoxes has only gained little attention in the past. We think however that the applications mentioned, especially in information hiding, clearly motivate the study of uniform interpolation for ontologies with ABoxes.

* Patrick Koopmann is supported by an EPSRC doctoral training award.

$\mathcal{ALC}$-uniform interpolants preserve all entailments of the form $C \sqsubseteq D$, $C(a)$ and $r(a, b)$, where C, D and r are $\mathcal{ALC}$-concepts and roles that only use concept and role symbols from a specified signature. As it turns out, traditional $\mathcal{ALC}$-ontologies are not always sufficient to represent uniform interpolants that preserve all these entailments. It is already known that cyclic definitions in the TBox may require the use of fixpoint operators or additional symbols in the uniform interpolant [16, 11]. ABoxes bring a problem of a different nature, which can be remedied by allowing disjunctions in the ABox of the uniform interpolant. ABoxes with disjunctions can be represented as classical ABoxes using additional role symbols [2]. In order to represent the uniform interpolant fully in the desired signature, we present an alternative approach using nominals.

To illustrate this, take as an example the ontology $\mathcal{O}$ containing the TBox axiom $A \sqsubseteq \forall r.(\neg B \sqcup A)$ and the ABox assertions $r(a, b)$, $r(b, a)$ and $B(b)$. The $\mathcal{ALC}$-uniform interpolant of this ontology for $\Sigma = \{A, r\}$ has to preserve all $\mathcal{ALC}$-entailments that only use A and r. This is the case for the ABox $\mathcal{A} = \{r(a, b), r(b, a), \neg A(a) \vee A(b)\}$. $\mathcal{A}$ is therefore an $\mathcal{ALC}$-uniform interpolant of $\mathcal{O}$ for Σ. We can replace the disjunction in this ABox by the $\mathcal{ALCO}$-concept assertion $(\neg A \sqcup \exists r.(\{b\} \sqcap A))(a)$. But in this case, it is impossible to preserve all entailments of $\mathcal{O}$ in an $\mathcal{ALC}$-ontology without disjunctions in the ABox. Our evaluation suggests however, that in most practical cases uniform interpolants can be represented without nominals or disjunctions.

In the next section we present the description logics used in this paper, and give a formal definition of uniform interpolation. In Section 3, we describe a method based on a new resolution calculus that is used to compute a clausal representation of the uniform interpolant. The method introduces new symbols to the signature, as well as disjunctions of concept assertions. In Section 4 we describe how these introduced symbols are eliminated and how a uniform interpolant without disjunctions can be computed. The result may contain fixpoints and nominals. Since fixpoints are not commonly supported by current description logic reasoners, we describe how uniform interpolants can be represented in $\mathcal{ALC}$, respectively $\mathcal{ALCO}$, by either using additional symbols or approximating the uniform interpolant. We evaluate our method in Section 5 and give an overview of related work in Section 6. We conclude with a short conclusion in Section 7. Proofs of all lemmas and theorems can be found in the long version of this paper [13].

2 Preliminaries

We present the description logics $\mathcal{ALC}$, $\mathcal{ALC}\mu$, $\mathcal{ALCO}$ and $\mathcal{ALCO}\mu$, and give a formal definition of disjunctive ABoxes and uniform interpolants.

Let N_c, N_r, N_i and N_v be four disjoint sets of respectively *concept symbols*, *role symbols*, *individuals* and *concept variables*. *$\mathcal{ALCO}\mu$-concepts* are of the form $\top$, A, $\neg C$, $C \sqcup D$, $\exists r.C$, $\{a\}$ and $\mu X.C[X]$, where $A \in N_c$, $r \in N_r$, $a \in N_i$, $X \in N_v$, C and D are $\mathcal{ALCO}\mu$-concepts and $C[X]$ is an $\mathcal{ALCO}\mu$-concept in which X occurs positively (under an even number of negations) as a replacement for a concept symbol. We define further concepts as abbrevations: $\bot = \neg\top$, $C \sqcap D = \neg(\neg C \sqcup \neg D)$, $\forall r.C = \neg\exists r.\neg C$, $\nu X.C[X] = \neg\mu X.\neg C[\neg X]$ and $\{a_1, ..., a_n\} = \{a_1\} \sqcup \ldots \sqcup \{a_n\}$. $\mu X.C[X]$ denotes the *least fixpoint* of $C[X]$ and $\nu X.C[X]$ the *greatest fixpoint*.

A *TBox* $\mathcal{T}$ is a set of *axioms*, which are either *concept inclusions* $C \sqsubseteq D$ or *concept equivalence axioms* $C \equiv D$, where C and D are $\mathcal{ALCO}\mu$-concepts. An *ABox* $\mathcal{A}$ is a set of *concept assertions* $C(a)$ and *role assertions* $r(a, b)$, where $r \in N_r$, $a, b \in N_i$ and C is any $\mathcal{ALCO}\mu$-concept. An ontology $\mathcal{O}$ is a tuple $\langle\mathcal{T}, \mathcal{A}\rangle$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox. If an ontology does not contain fixpoint expressions $\mu X.C[X]$ or $\nu X.C[X]$, it is an $\mathcal{ALCO}$-ontology. If it does not contain nominal expressions $\{a\}$ or $\{a_1, ..., a_n\}$, it is an $\mathcal{ALC}\mu$-ontology. If it contains neither nominals nor fixpoint expressions, it is in an $\mathcal{ALC}$-ontology. In the same way, we define $\mathcal{ALC}$ ($\mathcal{ALCO}$, $\mathcal{ALC}\mu$) -axioms, -concepts and -assertions.

The semantics is defined as usual (see, e.g., [3] for $\mathcal{ALCO}$ and [4] for least and greatest fixpoints). In particular, we write $\mathcal{O} \models \alpha$, where α is a concept inclusion, concept assertion or role assertion, if α is true in all models of $\mathcal{O}$.

A *disjunctive concept assertion* is of the form $C_1(a_1) \vee \ldots \vee C_n(a_n)$, where the C_i are $\mathcal{ALCO}\mu$-concepts and the a_i individuals, $1 \leq i \leq n$. Given an ontology $\mathcal{O}$, let $\mathcal{O} \models C_1(a_1) \vee \ldots \vee C_n(a_n)$ iff $\mathcal{O} \models C_i(a_i)$ for at least one $1 \leq i \leq n$. A *disjunctive ABox* is a set of role assertions, concept assertions and disjunctive concept assertions. To make the distinction clear, we refer to ABoxes which do not contain disjunctive concept assertions as *classical ABoxes*.

A *signature* $\Sigma \subseteq (N_c \cup N_r)$ is a set of concept and role symbols. Given a concept, axiom, assertion, TBox, ABox or ontology E, the *signature of* E, denoted by $sig(E)$, is the set of concept and role symbols occurring in E. Given an ontology $\mathcal{O}$ and a signature Σ, $\mathcal{O}^\Sigma$ is an $\mathcal{ALC}$-*uniform interpolant* of $\mathcal{O}$ for Σ, iff (i) $sig(\mathcal{O}^\Sigma) \subseteq \Sigma$ and (ii) $\mathcal{O}^\Sigma \models \alpha$ iff $\mathcal{O} \models \alpha$ for every $\mathcal{ALC}$-axiom and non-disjunctive $\mathcal{ALC}$-assertion α with $sig(\alpha) \subseteq \Sigma$. Since we do not define any other kinds of uniform interpolants, we refer to $\mathcal{ALC}$-uniform interpolants simply as *uniform interpolants* in the remainder of this paper. Note that we do not require $\mathcal{ALC}$-uniform interpolants to be expressed in $\mathcal{ALC}$ itself. We just require them to preserve all entailments that can be expressed in $\mathcal{ALC}$.

If $\Sigma = sig(\mathcal{O}) \setminus \{x\}$, where x is a single concept or role symbol, we call the uniform interpolant $\mathcal{O}^\Sigma$ the result of *forgetting* x from $\mathcal{O}$, and denote it by $\mathcal{O}^{-x}$.

3 Uniform Interpolation on ABoxes

The method for computing uniform interpolants works on a clausal form of the input ontology, which uses an additional set of special concept symbols.

Definition 1. *Let $N_d \subseteq N_c$ be a set of designated concept symbols called* definers. *A* TBox literal *is a concept of the form A, $\neg A$, $\exists r.D$ or $\forall r.D$, where $A \in N_c$ and $r \in N_r$ and $D \in N_d$. A* TBox clause *is a concept inclusion of the form $\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n$, where each L_i is a TBox literal. An* ABox literal *is a concept assertion of the form $L(a)$, where L is a TBox literal and $a \in N_i$. An* ABox clause *is a disjunctive concept assertion of the form $L_1 \vee \ldots \vee L_n$,*

TBox Resolution: $$\frac{C_1 \sqcup A \qquad C_2 \sqcup \neg A}{C_1 \sqcup C_2}$$

provided $C_1 \cup C_2$ does not contain more than one negative definer-literal.

TBox Role Propagation: $$\frac{C_1 \sqcup \forall r.D_1 \qquad C_2 \sqcup \mathsf{Q}r.D_2}{C_1 \sqcup C_2 \sqcup \mathsf{Q}r.D_{12}}$$

where $\mathsf{Q} \in \{\exists, \forall\}$ and D_{12} is a (possibly new) definer representing $D_1 \sqcap D_2$, provided $C_1 \cup C_2$ does not contain more than one negative definer-literal.

TBox Existential Role Restriction Elimination: $$\frac{C \sqcup \exists R.D \qquad \neg D}{C}$$

Fig. 1. TBox rules

where every L_i is an ABox literal. An ontology is in clausal form *if its TBox consists only of TBox clauses and its ABox consists only of ABox clauses and role assertions.*

We omit the leading '$\top \sqsubseteq$' in TBox clauses and assume that every clause is represented as a set. This means, no literal appears twice in a clause and the order of the literals is not important. Every ontology can be transformed into clausal form using standard structural transformation techniques, where each concept occurring under a role restriction $\exists r.C$ or $\forall r.C$ is replaced by a new definer D, for which we add a new axiom $D \sqsubseteq C$. The resulting ontology can be brought into clausal form by applying standard conjunctive normal form transformations. As a result, the TBox of the input ontology is represented by a set of TBox clauses, and the ABox of the input ontology is represented by a set of role assertions, ABox clauses as well as TBox clauses, which give meaning to the introduced definer symbols. The transformation can be done in polynomial time. Moreover, most ontologies are of a form that allows for a transformation in nearly linear time.

Example 1 (Normal form). The ontology $\mathcal{O} = \{A \sqsubseteq \forall r.B,\ (C \sqcup \forall r.(A \sqcup \neg B))(a),\ r(a,b)\}$ is not in clausal form. The ontology $\mathbf{N} = \{\neg A \sqcup \forall r.D_1,\ \neg D_1 \sqcup B,\ C(a) \vee (\forall r.D_2)(a),\ \neg D_2 \sqcup A \sqcup \neg B,\ r(a,b)\}$, where $D_1, D_2 \in N_d$, is in clausal form. $\mathbf{N}$ is equi-satisfiable with $\mathcal{O}$, and is therefore the clausal representation of $\mathcal{O}$.

Literals of the form $\neg D$ or $\neg D(a)$, $D \in N_d$ are called *negative definer literals*. An important invariant of the described transformation, which our calculus preserves, is that every TBox clause contains at most one negative definer literal and that ABox clauses do not contain any negative definer literals. This invariant ensures that we can always undo the structural transformation to eliminate introduced definer symbols, to get back to the original signature of the ontology.

As shown in [11], the rules in Figure 1 can be used to both decide satisfiability and compute uniform interpolants of $\mathcal{ALC}$-TBoxes in clausal form. A difference

ABox Resolution:

$$\frac{C_1 \vee A(a) \qquad C_2 \vee \neg A(a)}{C_1 \vee C_2} \qquad\qquad \frac{C_1 \sqcup \overline{L} \qquad C_2 \vee L(a)}{C_1(a) \vee C_2}$$

provided $C_1 \cup C_2$ does not contain a negative definer literal.

ABox Role Propagation:

$$\frac{C_1 \vee (\forall r.D_1)(a) \qquad C_2 \vee (\mathsf{Q}r.D_2)(a)}{C_1 \vee C_2 \vee (\mathsf{Q}r.D_{12})(a)} \qquad\qquad \frac{C_1 \vee (\forall r.D_1)(a) \qquad C_2 \sqcup \mathsf{Q}r.D_2}{C_1 \vee C_2(a) \vee (\mathsf{Q}r.D_{12})(a)}$$

$$\frac{C_1 \sqcup \forall r.D_1 \qquad C_2 \vee (\mathsf{Q}r.D_2)(a)}{C_1(a) \vee C_2 \vee (\mathsf{Q}r.D_{12})(a)}$$

where $\mathsf{Q} \in \{\exists, \forall\}$ and D_{12} is a (possibly new) definer representing D_{12}, provided $C_1 \cup C_2$ does not contain a negative definer literal.

Role Assertion Instantiation:

$$\frac{C_1 \vee (\forall r.D)(a) \qquad r(a,b)}{C_1 \vee D(b)} \qquad\qquad \frac{C_1 \sqcup \forall r.D \qquad r(a,b)}{C_1(a) \vee D(b)}$$

provided $C_1 \cup C_2$ does not contain a negative definer literal.

ABox Existential Role Restriction Elimination:

$$\frac{C \vee (\exists r.D)(a) \qquad \neg D}{C}$$

Fig. 2. ABox Rules

to standard resolution calculi is that new definer symbols are introduced dynamically during the process. If the role propagation rule is applied, and there is a definer D_{12} for which $\mathbf{N} \models D_{12} \sqsubseteq D_1 \sqcap D_2$ holds, where $\mathbf{N}$ is our current set of clauses, D_{12} is used in the conclusion. If such a definer does not exist, D_{12} is introduced as a new definer and the clauses $\neg D_{12} \sqcup D_1$ and $\neg D_{12} \sqcup D_2$ are added to the current clause set. This technique is key for termination of the calculus, since only a finite number of definer symbols is introduced [11].

If the ontology is consistent, ABox assertions have no influence on entailed concept inclusions. This means the TBox of the uniform interpolant can be computed using the TBox rules. In order to compute the ABox of the uniform interpolant, we need the ABox rules shown in Figure 2. Define $\overline{A} = \neg A$ and $\overline{\neg A} = A$, and for any TBox clause $C = L_1 \sqcup \ldots \sqcup L_n$, let $C(a)$ denote $L_1(a) \vee \ldots \vee L_n(a)$. The ABox resolution, role propagation and existential role restriction elimination rules are adaptions of the corresponding TBox rules. The role assertion instantiation rules make use of role assertions to propagate universal role restrictions. While the conclusion of every ABox rule is an ABox clause, the ABox role propagation rules may introduce new definers, and thus lead to additional TBox clauses in the current clause set.

Example 2 (ABox rules). Continuing on the clause set from the last example, we can apply ABox role propagation on $\neg A \sqcup \forall r.D_1$ and $C(a) \vee (\forall r.D_2)(a)$, which results in the clause $\neg A(a) \vee C(a) \vee (\forall r.D_{12})(a)$, where D_{12} is a new definer that is introduced with the two clauses $\neg D_{12} \sqcup D_1$ and $\neg D_{12} \sqcup D_2$. Applying role assertion instantiation on the new derived clause and $r(a, b)$ results in $\neg A(a) \vee C(a) \vee D_{12}(b)$.

Since the number of introduced definers is bounded and clauses are represented as sets, it can be verified that the saturation of any finite set of clauses by the rules in Figure 1 and 2 is always finite. The calculus is also sound and refutationally complete, as the following theorem shows.

Theorem 1. *The TBox and ABox rules form a sound and refutationally complete calculus and provide a decision procedure for satisfiability of $\mathcal{ALC}$-ontologies with non-empty TBoxes and ABoxes.*

The calculus is used in the same way for forgetting concept and role symbols as in [11, 9]. When forgetting a concept symbol A, the resolution rules only resolve A and definers, and the role propagation and role assertion instantiation rules are only applied if this makes new resolution steps on A possible.

When forgetting a role symbol r, the role assertion instantiation rule is applied for all role assertions with this r. Then the role propagation rules on r are used to derive all clauses of the form $C \sqcup \exists r.D$ and $C \vee (\exists r.D)(a)$, where D is an unsatisfiable definer. These existential restrictions are eliminated using the existential role restriction elimination rules. In order to decide whether a definer is unsatisfiable, we can either use our own calculus or an external reasoner. (For a rule-based representation of this procedure, see [9].) Afterwards we remove all clauses that containing the symbol A, respectively r, that we want to forget.

If our initial set of clauses is $\mathbf{N}$, we denote the result by $\mathbf{N}^{-x}$, where x is the symbol we want to forget. $\mathbf{N}^{-x}$ is the clausal representation of the result of forgetting x. In order to compute a uniform interpolant for a given signature Σ, we iteratively forget every symbol $x \notin \Sigma$ using this method.

Theorem 2. *Let $\boldsymbol{N}$ be any set of TBox and ABox clauses and x any concept or role symbol. Then, $\boldsymbol{N}^{-x} \models \alpha$ iff $\boldsymbol{N} \models \alpha$, for any $\mathcal{ALC}$-axiom or $\mathcal{ALC}$-assertion α with $x \notin sig(\alpha)$.*

4 Eliminating Definer Symbols and Disjunctive Assertions

In this section we describe how the introduced definer symbols are eliminated, and how the result can be represented in an ontology without disjunctive concept assertions.

For eliminating definers, the same technique as in [11, 9] is used. First, all TBox clauses that contain a positive definer literal of the form D and all ABox

Non-Cyclic Definer Elimination:

$$\frac{\mathcal{O} \cup \{D \sqsubseteq C\}}{\mathcal{O}^{[D \mapsto C]}} \qquad \text{provided } D \notin \mathit{sig}(C)$$

Definer Purification:

$$\frac{\mathcal{O}}{\mathcal{O}^{[D \mapsto \top]}} \qquad \text{provided } D \text{ occurs only positively in } \mathcal{O}$$

Cyclic Definer Elimination:

$$\frac{\mathcal{O} \cup \{D \sqsubseteq C[D]\}}{\mathcal{O}^{[D \mapsto \nu X.C[X]]}} \qquad \text{provided } D \in \mathit{sig}(C[D])$$

Fig. 3. Rules for eliminating definer symbols

clauses that contain literals of the form $D(a)$ are removed. As shown in the long version of this paper [13], these clauses are not needed to preserve entailments in the desired signature, since all necessary inferences on them have already been computed. In the resulting clause set, for every definer D, the set of clauses of the form $\neg D \sqcup C_i$, $1 \leq i \leq n$, is replaced by a single concept inclusion $D \sqsubseteq C_1 \sqcap \ldots \sqcap C_n$. The resulting ontology is saturated using the rules in Figure 3, where $\mathcal{O}^{[D \mapsto C]}$ denotes the result of replacing every definer D in $\mathcal{O}$ by the concept C. These rules implement Ackermann's Lemma [1] and its generalisation [18], which have been used in the context of second-order quantifier elimination (see also [6]).

Theorem 3. *Let $\mathcal{O}$ be an $\mathcal{ALC}$-ontology and $\Sigma \subseteq \mathit{sig}(\mathcal{O})$. Let $\mathcal{O}^{\Sigma}$ be the result of forgetting all symbols in $\mathit{sig}(\mathcal{O}) \setminus \Sigma$ from the clausal representation of $\mathcal{O}$, and eliminating all introduced definers. Then, $\mathcal{O}^{\Sigma}$ is an $\mathcal{ALC}$ or $\mathcal{ALC}\mu$-ontology with possibly disjunctive ABox, and $\mathcal{O}^{\Sigma}$ is a uniform interpolant of $\mathcal{O}$ for Σ.*

The cyclic definer elimination rule introduces fixpoint operators to the ontology. It is in general not always possible to find a finite uniform interpolant without fixpoint operators. An alternative approach is to allow additional symbols in the result, and simply not apply the cyclic definer elimination rule. By keeping the cyclic definer symbols as "helper concepts" in the ontology, we obtain an ontology that is not completely in the desired signature, but does not use fixpoint operators and preserves all entailments we are interested in. A third alternative is to approximate the fixpoint expressions in $\mathcal{ALC}$ as described in [11].

Using the role assertion instantiation rules, clauses with more than one individual can be derived, which will be represented as disjunctive concept assertions in the uniform interpolant. In general, there is no way to represent disjunctive concept assertions as classical concept assertions in $\mathcal{ALC}$ if the respective individuals are connected by role assertions, as the following lemma shows.

Lemma 1. *There are $\mathcal{ALC}$-ontologies with disjunctive ABoxes that cannot be approximated by a finite $\mathcal{ALC}\mu$-ontology with a classical ABox in such a way that* *all entailments of the form $C(a)$ or $C \sqsubseteq D$ are preserved, where C and D are $\mathcal{ALC}$-concepts. This even holds if the ontology contains no cyclic role assertions.*

A consequence is that uniform interpolants of $\mathcal{ALC}$-ontologies in general cannot be represented by $\mathcal{ALC}$ or $\mathcal{ALC}\mu$-ontologies with classical ABoxes.

Theorem 4. *There are $\mathcal{ALC}$-ontologies $\mathcal{O}$ and signatures Σ, such that there is no uniform interpolant of $\mathcal{O}$ for Σ that is a finite $\mathcal{ALC}\mu$-ontology without disjunctive concept assertions.*

A solution is to use nominals. If the individuals occurring in a disjunctive concept assertions are connected via role assertions, the assertion can be transformed using the following rule.

$$\frac{C \vee C_1(a) \vee C_2(b) \qquad r_1(a,a_1) \quad r_2(a_1,a_2) \quad \dots \quad r_{n+1}(a_n,b)}{C \vee (C_1 \sqcup \exists r_1.\dots\exists r_{n+1}.(\{b\} \sqcap C_2))(a)} \tag{1}$$

If $r(a,b) \in \mathcal{O}$, the concept $\exists r.\{b\}$ is already satisfied by a. The concept assertion $(\exists r.(\{b\} \sqcap C_2)(a)$ states additionally that the individual b also satisfies C_2. Therefore, concept assertions for b can be encoded inside concept assertions for a, if there is some path along the role assertions from a to b.

If there is no chain of role assertions between two individuals a and b, we cannot represent any information about b in a concept assertion of the form $C(a)$. This happens when the connecting role symbols have been removed from the signature. However, it is possible to saturate clauses with unconnected individuals until these clauses are not needed anymore to preserve entailments of the form $C(a)$. For example, if the uniform interpolant contains $A(a)$ and $\neg A(a) \vee B(b)$, we can also represent it by the classical ABox $\mathcal{A} = \{A(a), B(b)\}$. To make this approach practical, we use ordered resolution.

Let $\prec$ be any ordering between concept symbols and roles, and extend it to a total ordering $\prec_l$ between TBox literals such that $\forall r.D \prec \exists r.D \prec A \prec \neg A$ for all $r \in N_r$, $D \in N_d$ and $A \in N_c$. Given an ABox clause C, a literal $L(a) \in C$ is maximal if for all literals of the form $L'(a) \in C$ we have $L' \prec_l L$. Observe that if an ABox clause contains more than one individual name, it has more than one maximal literal.

We further keep a set M_d of marked definer symbols, which tells us which additional clauses we have to saturate. At the beginning, this set is empty. A clause is *selected* if it contains a pair of individuals that are not connected by a chain of role assertions, or a literal of the form $\neg D$ with $D \in M_d$.

Given a set of clauses $\mathbf{N}$, the set $\mathbf{N}^c$ is computed by saturating $\mathbf{N}$ using the TBox and ABox rules of Figures 1 and 2, with the following side conditions.

- At least one clause in the premise of the rule must be selected.
- Each rule is only applied on the maximal literals of the clauses.
- If a maximal literal of selected clause is of the form $\exists r.D$ or $(\exists r.D)(a)$, add D to M_d.

The set $\mathbf{N}^c$ is saturated using Rule (1), and all concept assertions containing more than one individual are removed. We denote the result by $\mathbf{N}^{\mathcal{O}}$. Since

Ontology	Input		Experiment		Output		
	ABox Assertions	Assertion Size	Timeouts	Duration	ABox Assertions	Assertion Size	Disjunctive Assertions
Family	1,340	3.00	35.65%	69.3 sec.	2,093	7.63	73.37%
Semintec	65,189	2.72	6.57%	129.5 sec.	108,417	3.47	36.09%
UOBM	198,308	2.77	11.54%	200.1 sec.	217,265	3.58	80.62%

Fig. 4. Results for computing uniform interpolants with disjunctive ABox assertions.

Rule (1) produces concept assertions that are not clauses, but $\mathcal{ALCO}$-assertions, $\mathbf{N}^{\mathcal{O}}$ is not a clause set, but an $\mathcal{ALCO}$-ontology.

Theorem 5. *Let $\boldsymbol{N}$ be any set of clauses. Then, $\boldsymbol{N}^{\mathcal{O}}$ is an $\mathcal{ALC}$ or $\mathcal{ALCO}$-ontology with classical ABox, and we have $\boldsymbol{N}^{\mathcal{O}} \models \alpha$ iff $\boldsymbol{N} \models \alpha$, where α is of the form $C \sqsubseteq D$, $C(a)$ or $r(a, b)$, and C and D are $\mathcal{ALC}$-concepts.*

Using this technique, uniform interpolants with disjunctions in the ABox can be transformed to $\mathcal{ALCO}$ or $\mathcal{ALCO}\mu$-ontologies with classical ABox.

5 Evaluation

In order to investigate how our method performs in practice, we implemented it in Scala[1] using the OWL-API,[2] with some of the optimisations described in [10]. Since we already evaluated uniform interpolation on TBoxes in [10] and [9], we focused on ontologies which have an ABox that is large compared to the TBox.

The evaluation was based on three ontologies. Family[3] is the family ontology of Robert Stevens. It has a relatively complex TBox and its ABox consists only of role assertions. Semintec[4] is an ontology about bank transactions, and has a simpler TBox but also a much larger ABox. The University Ontology Benchmark Generator (UOBM)[5] comes with a tool that allows to generate ABoxes, and has a more complex TBox than Semintec. For our evaluation we generated an ontology for one university. All ontologies have been reduced to $\mathcal{ALC}$ by removing all axioms which are not in $\mathcal{ALC}$, where we replaced domain and range restrictions, as well as number restrictions of the form $\geq 1r.C$ and $\leq 0r.C$ by equivalent expressions in $\mathcal{ALC}$. We then generated 350 random signatures for each ontology, where each symbol was assigned a probability of 25% to be forgotten, and we computed uniform interpolants with and without disjunctive assertions. The uniform interpolants were represented in $\mathcal{ALC}$ and $\mathcal{ALCO}$ respectively using helper concepts, so that they could in theory be exported to OWL. The timeout for each experimental run was set to 60 minutes.

[1] `http://www.scala-lang.org`
[2] `http://owlapi.sourceforge.net`
[3] `http://www.cs.man.ac.uk/~stevensr`
[4] `http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm`
[5] `http://www.cs.ox.ac.uk/isg/tools/UOBMGenerator`

Ontology	Input		Experiment		Output		
	ABox Assertions	Assertion Size	Timeouts	Duration	ABox Assertions	Assertion Size	Nominals
Family	1,340	3.00	43.14%	376.0 sec.	1,400	5.35	6.03%
Semintec	65,189	2.72	6.57%	223.1 sec.	105,993	3.40	0.00%
UOBM	198,308	2.77	26.29%	442.9 sec.	191,965	3.18	0.00%

Fig. 5. Results for computing uniform interpolants with classical ABoxes.

The results for computing uniform interpolants with disjunctive ABoxes are shown in Figure 4 and for computing uniform interpolants with classical ABoxes in Figure 5. The table shows the number of assertions in the ABox, the average size of an assertion of the respective ontology, the percentage of timeouts, the average duration, the average sizes of the output and the percentage of uniform interpolants that contained disjunctive assertions and nominals respectively.

Eliminating disjunctive assertions mostly took longer than forgetting the symbols. But as a result, only in a few cases nominals were necessary in the uniform interpolant, so that the uniform interpolants could in most cases be represented as $\mathcal{ALC}$-ontologies. Also, the uniform interpolants with classical ABox had similar sizes to the respective input ontologies, except for the Semintec ontology. If nominals were used, the respective assertions were usually very complex. The timeouts were mainly caused by a small number of symbols. For example, forgetting the concept Person from the Family ontology always lead to a timeout, which caused 25% of the timeouts for this ontology.

We did not evaluate our method on ontologies with larger ABoxes, and there are probably some optimisations possible for large ABoxes that will improve the performance. However, in most realistic cases it is likely that, if the TBox is fixed, the duration and the size of uniform interpolants are linear in the size of the ABox, since individuals are usually not overly connected and the reasoner can therefore process them in small independent groups for each symbol.

6 Discussion and Related Work

A lot of recent work covers uniform interpolation of TBoxes in description logics of varying expressivity, including DL-Lite [22], $\mathcal{EL}$ [8, 17, 15], $\mathcal{ALC}$ [16, 20, 14, 11, 10], $\mathcal{ALCH}$ [9] and $\mathcal{SHQ}$ [12].

The first approach to compute uniform uniform interpolants of $\mathcal{ALC}$-TBoxes is based on a representation of the input ontology in disjunctive normal form, from which incrementally a uniform interpolant is approximated [19], inspired by earlier work on uniform interpolation of $\mathcal{ALC}$-concepts [5]. This idea was further developed in [21, 20] to make use of existing tableaux calculi.

Uniform interpolation of $\mathcal{ALC}$-TBoxes is theoretically analysed in [16], where it is shown that deciding whether a uniform interpolant can be represented finitely without fixpoints is 2ExpTime-complete, and that the size of these uniform interpolants is in the worst case triple-exponentially with respect to the size of the input ontology.

The early approaches for uniform interpolation in $\mathcal{ALC}$ required the input-ontology to be either directly or effectively transformed into disjunctive normal form, which is an unusual representation for ontologies. They also derive consequences in an unrestricted way, which makes them unpractical for larger ontologies. Practical uniform interpolation requires a more goal-oriented approach, which is followed in [14] and [11], where concept symbols are eliminated using resolution. Whereas the method presented in [14] can only approximate uniform interpolants in the general case, the algorithm in [11] always terminates with a finite representation, which is achieved by using fixpoint operators. This method was evaluated on a larger corpus in [10], showing that the worst case complexity is hardly reached in practice and that for certain signatures uniform interpolants can be computed in short amounts of time. In [9], the method was extended to forgetting role symbols. The method also inspired a new calculus for uniform interpolation in $\mathcal{SHQ}$, which allows not only to interpolate more expressive ontologies, but also to preserve further entailments of $\mathcal{ALC}$-ontologies [12].

Uniform interpolation involving ABoxes was first investigated for the lightweight description logic DL-Lite [22]. Ontologies with ABoxes were also considered by the first approach for uniform interpolation in $\mathcal{ALC}$ [21]. But this approach only works if the ABox is of a specific form, since the result is always represented in $\mathcal{ALC}$.

7 Conclusion and Future Work

We presented a method for uniform interpolation of $\mathcal{ALC}$-ontologies with ABoxes. In general, it is not always possible to represent these uniform interpolants in $\mathcal{ALC}$ or using fixpoints. A solution is to either represent uniform interpolants in $\mathcal{ALCO}\mu$ or to allow disjunctions in the ABox. Our evaluation suggests however, that these cases do not occur often in practice, and that uniform interpolants of $\mathcal{ALC}$-ontologies with large ABoxes can be practically computed in most cases. We are currently working on extending the presented approach to more expressive description logics, for example, with number restrictions or inverse roles.

In this work we focused on preserving entailments that are expressible in $\mathcal{ALC}$. Since the output of our method is represented in $\mathcal{ALCO}\mu$, it might be interesting to preserve $\mathcal{ALCO}$-entailments as well. There are some entailments of this sort that our algorithm does not respect. For example, from the two ABox clauses $A(a) \vee \neg B(a)$ and $B(b)$ one can deduce $(A \sqcup \neg\{b\})(a)$. Another restriction is our uniform interpolants only preserve entailments of the form $C \sqsubseteq D$, $C(a)$ and $r(a,b)$. It would be interesting to investigate whether preserving further entailments, such as conjunctive queries, is possible.

Our method can only be used for forgetting concept and role symbols. An open question is how forgetting individuals can be performed. This would extend the possibilities of the approach for applications such as information hiding.

References

1. Ackermann, W.: Untersuchungen über das Eliminationsproblem der mathematischen Logik. Mathematische Annalen 110(1), 390–413 (1935)
2. Areces, C., Blackburn, P., Hernandez, B.M., Marx, M.: Handling Boolean ABoxes. In: Proc. DL'03. CEUR-WS.org (2003)
3. Baader, F., Nutt, W.: Basic description logics. In: Baader, F., Calvanese, D., McGuiness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.) The Description Logic Handbook, chap. 2. Cambridge University Press, second edn. (2007)
4. Calvanese, D., Giacomo, G.D., Lenzerini, M.: Reasoning in expressive description logics with fixpoints based on automata on infinite trees. In: Proc. IJCAI '99. pp. 84–89. Morgan Kaufmann (1999)
5. ten Cate, B., Conradie, W., Marx, M., Venema, Y.: Definitorially complete description logics. In: Proc. KR'06. pp. 79–89 (2006)
6. Gabbay, D.M., Schmidt, R., Szalas, A.: Second Order Quantifier Elimination: Foundations, Computational Aspects and Applications. College Publ. (2008)
7. Grau, B.C.: Privacy in ontology-based information systems: A pending matter. Semantic Web 1(1-2), 137–141 (2010)
8. Konev, B., Walther, D., Wolter, F.: Forgetting and uniform interpolation in large-scale description logic terminologies. In: Proc. IJCAI '09. pp. 830–835. AAAI Press (2009)
9. Koopmann, P., Schmidt, R.A.: Forgetting concept and role symbols in $\mathcal{ALCH}$-ontologies. In: Proc. LPAR'13. LNCS, vol. 8312, pp. 552–567. Springer (2013)
10. Koopmann, P., Schmidt, R.A.: Implementation and evaluation of forgetting in $\mathcal{ALC}$-ontologies. In: Proc. WoMO'13. CEUR-WS.org (2013)
11. Koopmann, P., Schmidt, R.A.: Uniform interpolation of $\mathcal{ALC}$-ontologies using fixpoints. In: Proc. FroCoS'13. LNCS, vol. 8152, pp. 87–102. Springer (2013)
12. Koopmann, P., Schmidt, R.A.: Count and forget: Uniform interpolation of $\mathcal{SHQ}$-ontologies. In: Proc. IJCAR'14. Springer (2014), to appear.
13. Koopmann, P., Schmidt, R.A.: Forgetting and uniform interpolation of $\mathcal{ALC}$-ontologies with ABoxes—long version. Tech. rep., University of Manchester (2014), `http://www.cs.man.ac.uk/~koopmanp/DL_KoopmannSchmidt2014_long.pdf`
14. Ludwig, M., Konev, B.: Towards practical uniform interpolation and forgetting for $\mathcal{ALC}$ TBoxes. In: Proc. DL'13. pp. 377–389. CEUR-WS.org (2013)
15. Lutz, C., Seylan, I., Wolter, F.: An automata-theoretic approach to uniform interpolation and approximation in the description logic $\mathcal{EL}$. In: Proc. KR'12. pp. 286–296. AAAI Press (2012)
16. Lutz, C., Wolter, F.: Foundations for uniform interpolation and forgetting in expressive description logics. In: Proc. IJCAI '11. pp. 989–995. AAAI Press (2011)
17. Nikitina, N.: Forgetting in general $\mathcal{EL}$ terminologies. In: Proc. DL'11. pp. 345–355. CEUR-WS.org (2011)
18. Nonnengart, A., Szałas, A.: A fixpoint approach to second-order quantifier elimination with applications to correspondence theory. In: Logic at Work, pp. 307–328. Springer (1999)
19. Wang, K., Wang, Z., Topor, R., Pan, J., Antoniou, G.: Concept and role forgetting in $\mathcal{ALC}$ ontologies pp. 666–681 (2009)
20. Wang, K., Wang, Z., Topor, R., Pan, J.Z., Antoniou, G.: Eliminating concepts and roles from ontologies in expressive descriptive logics. Computational Intelligence (2012)

21. Wang, Z., Wang, K., Topor, R., Zhang, X.: Tableau-based forgetting in $\mathcal{ALC}$ ontologies. In: Proc. ECAI '10. pp. 47–52. IOS Press (2010)
22. Wang, Z., Wang, K., Topor, R.W., Pan, J.Z.: Forgetting for knowledge bases in DL-Lite. Ann. Math. Artif. Intell. 58(1–2), 117–151 (2010)