Semantic, Query, and Deductive Forgetting in One Framework for ALC Ontologies

Mostafa Sakr[0000−0003−4296−5831] and Renate A. Schmidt[0000−0002−6673−3333]

University of Manchester, United Kingdom.
Mostafa.Sakr,Renate.Schmidt@manchester.ac.uk

Abstract. We study the problem of forgetting concept names forgetting from ALC ontologies. We present a unified framework that computes representations of the deductive, query, and semantic forgetting views, and extracts a set axioms indicating the information not preserved in each. This enables users to enhance forgetting views with more information according to their requirement, and obtain a smooth transition between the three variants of forgetting views.

Keywords: Forgetting · Query · Ontology · Description Logic.

1 Introduction

Forgetting creates a restricted view of an ontology by eliminating subsets of concept and role names from the ontology, while preserving the information pertaining to the remainder of the signature [10,28,8]. Different variants of forgetting have been introduced in the literature to support different reasoning tasks. (i) *Deductive forgetting*, or *uniform interpolation* [34,13,28] where all concept descriptions over the non-forgetting symbols in the logic of the input ontology are preserved. This variant supports entailments checking and classification tasks. (ii) *Query forgetting* where the input ontology and the forgetting view allow for retrieving the same answers to arbitrary conjunctive queries against arbitrary ABoxes [19]. This variant supports ontology-based query answering applications. (iii) *Semantic forgetting* which preserves the interpretations of the non-forgetting symbols [25,24,32]. This variant supports all reasoning tasks over the non-forgetting symbols because it preserves the maximum amount of information [5].

In this paper we introduce a unified forgetting framework that eliminates concept names from input ontologies, and computes representations of views for the three forgetting variants. The framework is designed for $\mathcal{ALC}$ ontologies, and can thus operate on a wide range of real-life ontologies. In the first part of the framework, we compute an *intermediate ontology* which does not use any of the forgetting signature, and preserves the interpretations of the non-forgetting symbols. In the second part, the content of the intermediate ontology is customized according to the wanted variant of forgetting. We guarantee the existence of the intermediate ontology by allowing the use of auxiliary concept names, called definers. In the cases when the intermediate ontology does not use definers, it is a semantic forgetting view, and the customization processes are not needed. When definers are present in the intermediate ontology, it is a representation, or approximation, of the semantic forgetting views, and the elimination of the definers becomes the responsibility of the customization processes. We introduce two processes to extract deductive and query forgetting views from the intermediate ontology. To our knowledge the query customization provides the first query forgetting method for $\mathcal{ALC}$ ontologies. It computes two representations, an $\mathcal{ALC}$ representation, and an $\mathcal{ALCI}$ representation (with role inverse). The $\mathcal{ALC}$ representation uses definers which are eliminated in the $\mathcal{ALCI}$ representation. The $\mathcal{ALC}$ representation is also a representation of the deductive forgetting views. It is thus suitable for more reasoning tasks, and is usable with $\mathcal{ALC}$ reasoning tools. The forgetting framework also enables new use cases that are not supported by existing forgetting methods.

(i) Suppose agent A requests an extract of the ontology of agent B which decides to publish a forgetting view in order to hide sensitive portions of its ontology. The suitable variant of forgetting might not be known to agent A when the request is made, for instance if the complete list of reasoning tasks is only settled at a later stage. A solution is publishing the semantic forgetting view of the ontology omitting the sensitive information to agent A, but if it does not exist, an alternative is publishing the intermediate ontology computable by the first part of the framework, and customizing it later by agent A. This use case is not supported by any of the existing forgetting methods.

(ii) In addition to forgetting views, the unified framework creates sets of axioms indicating the content not preserved in each type of forgetting view. This enables users to enhance deductive and query forgetting views with more information based on their requirements, and achieve a smooth transition between the three variants of forgetting views. For instance, in the deductive customization we compute a set Δ^d of clauses indicating the content difference between the intermediate ontology and the deductive forgetting view. An $\mathcal{ALC}$ version of the query forgetting view is then obtained by enhancing a representation of the deductive view with axioms from Δ^d.

Due to space restrictions, all proofs are provided in the long version. A more detailed description of the first part of the framework, and the deductive customization is presented in [33].

2 Related Work

Forgetting was recognized as an important problem in artificial intelligence in [25], where it was found to coincide with the problem of second-order quantifier elimination, a highly undecidable problem concerned with eliminating second-order quantified symbols from second-order theories [1,11,12]. This notion of forgetting coincides with our notion of semantic forgetting.

Undecidability of semantic forgetting for first-order logic has led to the introduction of the weaker notion of deductive forgetting [38], which coincides with the problem of *uniform interpolation* [14,15,34]. Although uniform interpolation is also undecidable for several logics including $\mathcal{ALC}$ [28,39], empirical evaluations have shown that computing deductive forgetting views of real life ontologies is often feasible [20,41,26]. Subsequently, several deductive forgetting methods were developed for the DLs between $\mathcal{ALC}$ and $\mathcal{SHQ}$ [2,9,16,20,21,22,23,26,35,36].

In a modern view of forgetting, the different variants have been investigated based on *inseparability*, a notion originally introduced in [15] and rigorously developed in [6,7,17,18,27]. Forgetting computes a forgetting view that is inseparable from the input ontology. Inseparability here is dependent of the reasoning task of interest. For instance, an ontology and its deductive view are inseparable with respect to entailment checking, but may be separable with respect to instance checking or conjunctive query answering [27]. To address the last two tasks, the notions of *instance forgetting* and *query forgetting* were proposed and studied for extensions of the description logic $\mathcal{EL}$ [19].

For the description logic $\mathcal{ALC}$, semantic and query forgetting are insufficiently studied. Semantic forgetting methods were proposed in [42,43,44,45] for extensions of $\mathcal{ALC}$, but there are cases where they do not compute the forgetting view even if one exists [40]. Approximations to the semantic forgetting view were proposed in [25,32] where the forgetting view is enhanced with auxiliary concept names to avoid nonexistence issues. A similar idea is used in this paper where the intermediate ontology approximates the semantic forgetting view with definers. Query forgetting has not been studied in the context of $\mathcal{ALC}$. One challenge is that query inseparability, the inseparability notion underpinning query forgetting, is undecidable [7]. Although this adds complexity to query forgetting, developing $\mathcal{ALC}$ query forgetting methods remains open. The difference between the two problems is that the former decides whether any two ontologies are query inseparable or not, but the latter asks whether an arbitrary input $\mathcal{ALC}$ ontology and one specific query view are query inseparable or not.

3 Preliminaries and Basic Definitions

Let N_c, N_r be two disjoint sets of concept symbols and role symbols. Concepts in $\mathcal{ALC}$ have the forms: $\bot \mid A \mid \neg C \mid C \sqcap D \mid \exists r.C$ where $A \in N_c, r \in N_r$ and C and D are general concept expressions. We use the following abbreviations: $\top \equiv \neg\bot, \forall r.C \equiv \neg\exists r.\neg C, C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$. An interpretation $\mathcal{I}$ in $\mathcal{ALC}$ is a pair $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where the domain $\Delta^{\mathcal{I}}$ is a nonempty set and $\cdot^{\mathcal{I}}$ is an interpretation function that assigns to each concept symbol $A \in N_c$ a subset of $\Delta^{\mathcal{I}}$ and to each $r \in N_r$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Non-atomic concepts are interpreted according to: $\bot^{\mathcal{I}} := \emptyset$, $(\neg C)^{\mathcal{I}} := \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}, (C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}, (\exists r.C)^{\mathcal{I}} := \{x \in \Delta^{\mathcal{I}} | \exists y : (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$.

The description logic $\mathcal{ALCI}$ extends $\mathcal{ALC}$ allowing concepts of the form $\exists r^-.C$, and the abbreviation $\forall r^-.C \equiv \neg\exists r.^-.\neg C$, where $r \in N_r$, r^- is the *inverse* of r, and C is an $\mathcal{ALCI}$ concept. An interpretation $\mathcal{I}$ in $\mathcal{ALCI}$ extends an $\mathcal{ALC}$ interpretation with the restriction that $(x,y) \in r^{-\mathcal{I}}$ if and only if $(y,x) \in r^{\mathcal{I}}$. The concept $\exists r^-.C$ is then interpreted as $\{x \in \Delta^{\mathcal{I}} | \exists y : (y,x) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$.

An $\mathcal{L}$-TBox, or an ontology, is a set of axioms $C \sqsubseteq D$ and $C \equiv D$, where $\mathcal{L} \in \{\mathcal{ALC}, \mathcal{ALCI}\}$, and C and D are $\mathcal{L}$-concepts. $\mathcal{I}$ is a model of an ontology $\mathcal{O}$ if all axioms in $\mathcal{O}$ are true in $\mathcal{I}$, in symbols $\mathcal{I} \models C \sqsubseteq D$. And, $\mathcal{I} \models C \sqsubseteq D$ if and only if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. We say that $C \sqsubseteq D$ is satisfiable with respect to $\mathcal{O}$ if and only if $\mathcal{I} \models C \sqsubseteq D$ for some model $\mathcal{I}$ of $\mathcal{O}$. We also say that $C \sqsubseteq D$ is a consequence (entailment) of $\mathcal{O}$, in symbols $\mathcal{O} \models C \sqsubseteq D$, if and only if $\mathcal{I} \models C \sqsubseteq D$ for every model $\mathcal{I}$ of $\mathcal{O}$. Let C be a concept and $\mathcal{O}$ an ontology. We denote by $sig(C)$ the set of concept and role names appearing in C, and by $sig(\mathcal{O})$ the set $\bigcup_{C \sqsubseteq D \in \mathcal{O}} sig(C) \cup sig(D)$. We define the function $\mathcal{L}(\mathcal{O})$ to mean the logic used to express $\mathcal{O}$.

An ABox is a set of concept and role assertions over individuals. Let N_I be a set of individuals disjoint with N_c, and N_r. Assertions in an ABox take the forms $A(a)$ and $r(a, b)$ where $A \in N_c$, $r \in N_r$, and $a, b \in N_I$. We denote by $sig(\mathcal{A})$ the concepts and role names in $\mathcal{A}$. An $\mathcal{ALC}$ ($\mathcal{ALCI}$) knowledge-base is a pair $(\mathcal{O}, \mathcal{A})$ of an $\mathcal{ALC}$ ($\mathcal{ALCI}$) ontology $\mathcal{O}$ and an ABox $\mathcal{A}$.

A conjunctive query $q(\vec{x})$ is a first order formula $\exists \vec{y}.\, q(\vec{x}, \vec{y})$ where $\vec{x} = x_1, \ldots, x_k$ are the *answer variables*, $q(\vec{x}, \vec{y})$ is a conjunction of atoms of the forms $A(u)$ and $r(u, v)$, $A \in N_c$, $r \in N_r$, $u, v \in N_I \cup \vec{x} \cup \vec{y}$. If q does not have answer variables then q is called a Boolean conjunctive query. An answer to a query $q(\vec{x})$ is $\vec{a} = a_1, \ldots, a_k$ with the same cardinality as $\vec{x}$ such that $\mathcal{K} \models q(\vec{a})$. That is, for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$ we have $\mathcal{I} \models q(\vec{a})$. The answer to a Boolean conjunctive query is *yes* if $\mathcal{I} \models q$ for every model $\mathcal{I}$ of $\mathcal{K}$, and *no* otherwise.

Definition 1. *Let Σ be a signature. Two models $\mathcal{I}$ and $\mathcal{J}$* Σ-coincide *iff $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$ and $p^{\mathcal{I}} = p^{\mathcal{J}}$ for every concept name or role name $p \in \Sigma$.*

Definition 2. *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two ontologies and $\Sigma \subseteq N_c \cup N_r$. We say that the two ontologies are:*

1. model inseparable wrt. Σ, *in symbols $\mathcal{O}_1 \equiv_{\Sigma}^{\mathcal{M}} \mathcal{O}_2$, iff for every model $\mathcal{I}_1$ of $\mathcal{O}_1$ there is a model $\mathcal{I}_2$ of $\mathcal{O}_2$, and vice versa, such that $\mathcal{I}_1$ and $\mathcal{I}_2$ Σ-coincide.*
2. deductively inseparable wrt. Σ, *in symbols $\mathcal{O}_1 \equiv_{\Sigma}^{C} \mathcal{O}_2$, iff for every axiom α over Σ we have $\mathcal{O}_1 \models \alpha$ iff $\mathcal{O}_2 \models \alpha$, where $\mathcal{L}(\mathcal{O}_1) = \mathcal{L}(\mathcal{O}_2) = \mathcal{L}(\alpha)$.*
3. query inseparable wrt. Σ, *in symbols $\mathcal{O}_1 \equiv_{\Sigma}^{Q} \mathcal{O}_2$, iff for every ABox $\mathcal{A}$, and conjunctive query $q(\vec{x})$ over Σ, we have $(\mathcal{O}_1, \mathcal{A}) \models q(\vec{a})$ iff $(\mathcal{O}_2, \mathcal{A}) \models q(\vec{a})$.*

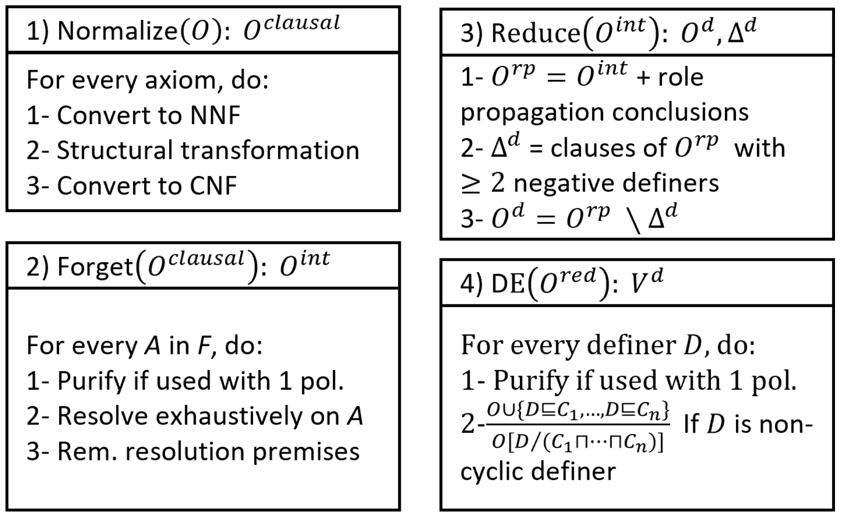
Definition 3. *Let $\mathcal{O}$ be an $\mathcal{ALC}$ ontology, and $\mathcal{F} \subseteq sig(\mathcal{O}) \cap N_c$ a* forgetting signature. *Let $\mathcal{V}$ be an ontology such that $sig(\mathcal{V}) \subseteq sig(\mathcal{O}) \backslash \mathcal{F}$. We say that $\mathcal{V}$ is:*

1. semantic forgetting view *of $\mathcal{O}$ wrt. $\mathcal{F}$ iff $\mathcal{O}$ and $\mathcal{V}$ are model inseparable.*
2. deductive forgetting view *of $\mathcal{O}$ wrt. $\mathcal{F}$ iff $\mathcal{O}$ and $\mathcal{V}$ are deductively inseparable.*
3. query forgetting view *of $\mathcal{O}$ wrt. $\mathcal{F}$ iff $\mathcal{O}$ and $\mathcal{V}$ are query inseparable.*

As observed in [5], semantic forgetting views are also deductive and query forgetting views because they preserve the interpretations of the non-forgetting symbols. In Horn logics, query forgetting views are deductive forgetting views [19], but this is not the case in non-Horn logics. Consider for instance the ontology consisting of the two axioms $A \sqsubseteq B, B \sqsubseteq C \sqcup D$. The empty ontology and the ontology $\{A \sqsubseteq C \sqcup D\}$ are respectively a query and a deductive forgetting views with respect to $\{B\}$, but the empty ontology is not a deductive forgetting view.

4 The Intermediate Ontology

This section describes the first part of the framework. The output of this part is the intermediate ontology $\mathcal{O}^{int}$ The first and second cards in Fig 1 depicts the computation process of $\mathcal{O}^{int}$. They are, the *normalize,* and the *forget,* pro-



1) Normalize(O): $O^{clausal}$

For every axiom, do:
1- Convert to NNF
2- Structural transformation
3- Convert to CNF

2) Forget$(O^{clausal})$: O^{int}

For every A in F, do:
1- Purify if used with 1 pol.
2- Resolve exhaustively on A
3- Rem. resolution premises

3) Reduce(O^{int}): O^{d}, Δ^{d}

1- $O^{rp} = O^{int}$ + role propagation conclusions
2- Δ^{d} = clauses of O^{rp} with $\geq$ 2 negative definers
3- $O^{d} = O^{rp} \setminus \Delta^{d}$

4) DE(O^{red}): V^{d}

For every definer D, do:
1- Purify if used with 1 pol.
2- $\frac{O\cup\{D\sqsubseteq C_1,\ldots,D\sqsubseteq C_n\}}{O[D/(C_1\sqcap\cdots\sqcap C_n)]}$ If D is non-cyclic definer

Fig. 1: Fine-grained forgetting framework.

cesses. The *normalize* process takes the input ontology $\mathcal{O}$, and transforms it to an ontology $\mathcal{O}^{clausal}$ in normal form. Each axiom of $\mathcal{O}$ is transformed separately. First, the axiom is transformed to *negation normal form* where negations are directly applied to concept names and the symbols $\sqsubseteq$ and $\equiv$ are eliminated. Second, clauses below role restrictions that use forgetting symbols are extracted by the *structural transformation* technique in [4,31]. This step introduces auxiliary concept names, called *definers*, which represent subsets of role successors. For instance, if B is a forgetting symbol the clause $A \sqcup \exists r.(B \sqcap C)$ is structurally transformed to the clauses $A \sqcup \exists r.D, \neg D \sqcup (B \sqcap C)$, where D is a definer. Third, the clauses computed from the previous two steps are transformed to *conjunctive normal form.*

The second process in the framework is the *forget* process. Its inputs are the ontology $\mathcal{O}^{clausal}$, and the forgetting signature $\mathcal{F}$. The output is an ontology $\mathcal{O}^{int}$, called *the intermediate ontology.* $\mathcal{O}^{int}$ is constructed from $\mathcal{O}^{clausal}$ by iteratively eliminating each forgetting concept name $A \in \mathcal{F}$. If A occurs with a single polarity across all clauses, it is *purified*, i.e., replaced with $\top$ if it occurs positively, and $\bot$ if it occurs negatively. Otherwise, exhaustive resolution is performed on A-literals in all clauses before discarding these clauses. Standard optimizations such as *tautology deletion* which eliminates tautologous clauses,

Role Propagation

$$\frac{P_0 \sqcup C_0,\ \bigcup_{j=1}^{m}\{P_j \sqcup C_j\}.\ E_0 \sqcup \mathcal{Q}r.D_0,\ \bigcup_{i=1}^{n}\{E_i \sqcup \forall r.D_i\}}{(\bigsqcup_{i=0}^{n} E_i) \sqcup \mathcal{Q}r.(\bigsqcap_{j=0}^{m} C_j)}$$

where $P_0 = \bigsqcup_{i=0}^{n} \neg D_i$, P_j is any concept in P_0, $\mathcal{Q} \in \{\exists, \forall\}$, and C_0 and C_j do not contain a definer.

Simple Definer Elimination

$$\frac{\mathcal{O} \cup \{\neg D \sqcup C_1, ..., \neg D \sqcup C_n\}}{\mathcal{O}[D/\bigsqcap_{i=1}^{n} C_i]}$$

where $D \notin sig(C_i)$, C_i does not contain any negative definers, and $\mathcal{O}$ does not contain D negatively.

Fig. 2: Role Propagation and Simple Definer Elimination rules.

and *subsumption deletion* which deletes a clause if it is subsumed by another, are eagerly performed.

Theorem 1. *Let $\mathcal{O}$ be an input ontology, and $\mathcal{F}$ the given forgetting signature of concept names. Let $\mathcal{O}^{int}$ be the ontology obtained from the normalise and forget processes described above. The two ontologies $\mathcal{O}$ and $\mathcal{O}^{int}$ are model inseparable for the non-forgetting symbols $sig(\mathcal{O})\backslash\mathcal{F}$.*

5 The Deductive Customization

We present the deductive customization process of $\mathcal{O}^{int}$. The process is depicted by the third and fourth cards in Fig 1. The reduce process takes the intermediate ontology $\mathcal{O}^{int}$, and computes two sets $\mathcal{O}^d$ and Δ^d of axioms. $\mathcal{O}^d$ the *deductively reduced ontology* . It is deductively inseparable from the $\mathcal{O}$ and $\mathcal{O}^{int}$ with respect to the non-forgetting signature $sig(\mathcal{O})\backslash\mathcal{F}$. We will use $\mathcal{O}^d$ later to obtain the deductive forgetting view, and it will be the basis of the fine-grained views. Δ^d comprises the $\mathcal{O}^{int}$ clauses that use two or more distinct definers with negative polarity. It indicates the content difference between $\mathcal{O}^{int}$ and $\mathcal{O}^d$.

The ontology $\mathcal{O}^d$ is computed by first making some implicit consequences of $\mathcal{O}^{int}$ explicit. We denote this enhanced snapshot of $\mathcal{O}^{int}$ by $\mathcal{O}^{rp}$. Indeed, $\mathcal{O}^{rp}$ and $\mathcal{O}^{int}$ are logically equivalent, because they have the same models. The ontology $\mathcal{O}^d$ is then obtained as the set of clauses of $\mathcal{O}^{rp}$ that are not in Δ^d. $\mathcal{O}^{rp}$ is obtained by saturating $\mathcal{O}^{int}$ with the conclusions computable by *role propagation* rule in Fig 2. The rule applies on four premises. The first is a Δ^d clause of the form $P_0 \sqcup C_0$, where C_0 is a concept that does not use negative definers, and P_0 is a disjunction of two or more negative definers. The clause $P_0 \sqcup C_0$ is called a *trigger clause*, and the definers occurring in P_0 *trigger definers*, because they influence the other premises of the role propagation rule. When $\mathcal{O}^{int}$ has several clauses $P_0 \sqcup C_0^1, P_0 \sqcup C_0^2, \ldots, P_0 \sqcup C_0^l$ with $l \geq 1$, we *combine* these clauses in a single clause $P_0 \sqcup (\bigsqcap_{i=1}^{l} C_0^i)$.

The second premise is a set of m clauses $P_j \sqcup C_j$, with $1 \leq j \leq m$ and $m \geq 0$. Negative definers cannot occur in C_j, and the concepts P_j are disjunctions of negative definers. The definers in P_j must form a subset of the trigger definers.

The third and the fourth premises are clauses where trigger definers occur positively below role restrictions. The definer of the third premise can occur below existential or universal role restriction. The definers of the fourth premise can only occur below universal role restrictions. There can be $n+1$ clauses in the third and the fourth premises. Each clause uses a different trigger definer. If $\mathcal{O}^{int}$ has many clauses where a trigger definer occurs positively, then we combine them, and use the combined clause in the inference.

The fourth process of the framework is the *DE* (or *Definer Elimination*) process. Its input is the deductively reduced ontology $\mathcal{O}^d$, and its output is an ontology $\mathcal{V}^d$ which is model inseparable to $\mathcal{O}^d$ with respect to $sig(\mathcal{O})\backslash\mathcal{F}$. The ontology $\mathcal{V}^d$ is computed from $\mathcal{O}^d$ by iteratively eliminating definers. Definers occurring with single polarity across all clauses are purified away. Otherwise, the simple definer elimination rule in Fig 2 is applied.

Example 1. Let $\mathcal{O} = \{A \sqsubseteq \forall r.B \sqcap \forall s.\neg B, G \sqsubseteq \exists r.(\neg B \sqcup C), B \sqsubseteq H\}$, and $\mathcal{F} = \{B\}$. The normalization process produces $\mathcal{O}^{clausal} = \{\neg A \sqcup \forall r.D_1, \neg A \sqcup \forall s.D_2, \neg G \sqcup \exists r.D_3, \neg D_1 \sqcup B, \neg D_2 \sqcup \neg B, \neg D_3 \sqcup \neg B \sqcup C, \neg B \sqcup H\}$, where D_1, D_2, and D_3 are definers. The forget process performs resolution on the B-literals which computes the clauses $\{\neg D_1 \sqcup \neg D_2, \neg D_1 \sqcup \neg D_3 \sqcup C, \neg D_1 \sqcup H\}$, and discards the premises $\{\neg D_1 \sqcup B, \neg D_2 \sqcup \neg B, \neg D_3 \sqcup \neg B \sqcup C, \neg B \sqcup H\}$. Now, $\mathcal{O}^{int}$ is $\{\neg A \sqcup \forall r.D_1, \neg A \sqcup \forall s.D_2, \neg G \sqcup \exists r.D_3, \neg D_1 \sqcup \neg D_2, \neg D_1 \sqcup \neg D_3 \sqcup C, \neg D_1 \sqcup H\}$, and Δ^d is the subset $\{\neg D_1 \sqcup \neg D_2, \neg D_1 \sqcup \neg D_3 \sqcup C\}$. We construct $\mathcal{O}^{rp}$ by saturating $\mathcal{O}^{int}$ using role propagation inferences. In this example, only one inference can be performed with the trigger clause $\neg D_1 \sqcup \neg D_3 \sqcup C$. The clause $\neg D_1 \sqcup \neg D_2$ cannot trigger a role propagation inference because D_1 and D_2 occur positively below different roles r and s. The second premise of the inference is the empty set, the third is $\neg G \sqcup \exists r.D_3$, and the fourth is the set $\{\neg A \sqcup \forall r.D_1\}$. The conclusion of the inference is $\neg A \sqcup \neg G \sqcup \exists r.(C \sqcap H)$. The reduced ontology $\mathcal{O}^d$ comprises the clauses of $\mathcal{O}^{rp}$ that are not in Δ^d. That is, $\mathcal{O}^d = \{\neg A \sqcup \forall r.D_1, \neg D_1 \sqcup H, \neg A \sqcup \forall s.D_2, \neg G \sqcup \exists r.D_3, \neg A \sqcup \neg G \sqcup \exists r.(C \sqcap H)\}$. The definer D_1 is eliminated from $\mathcal{O}^d$ using a simple definer elimination inference, which replaces every occurrence of D_1 with H. The definers D_2 and D_3 are purified by replacing them with $\top$ in the third and fourth clauses of $\mathcal{O}^d$. The deductive forgetting view $\mathcal{V}^d$ of $\mathcal{O}$ with respect to $\mathcal{F}$ is therefore the set $\{\neg A \sqcup \forall r.H, \neg G \sqcup \exists r.\top, \neg A \sqcup \neg G \sqcup \exists r.(C \sqcap H)\}$.

The described method does not eliminate all definers. In particular, *cyclic definers*, i.e., definers occurring both positively and negatively in one or more clauses, in $\mathcal{O}^d$ are not attempted to be eliminated. Cyclic definers signal there are syntactic cycles in the original ontology over some forgetting symbols. In some cases, they are eliminable but may result in infinite forgetting views. In the remainder of the cases, finite representations are possible, but may be complex to find and unpractical. For example, for the ontology $\mathcal{O} = \{A \sqsubseteq B, B \sqsubseteq \forall r.B\}$, and forgetting signature $\mathcal{F} = \{B\}$, $\mathcal{O}^{int}$ and $\mathcal{O}^d$ coincide, and are represented by $\{\neg A \sqcup \forall r.D, \neg D \sqcup \forall r.D\}$. The definer D is cyclic because it occurs positively and negatively in $\neg D \sqcup \forall r.D$. Eliminating it gives the clause $\beta = \neg A \sqcup \forall r.\forall r.\forall r....$ with infinite nesting of role restrictions. Suppose $\mathcal{O}$ contains the axiom $\alpha = A \sqsubseteq \forall r.\forall r.\bot$. In this case, the desired ontology will just comprise the axiom α because β is redundant. Although α in this example is explicitly known, in real life it can be obscure, and inferring it can be expensive. In general eliminating cyclic definers can be double exponential in the size of the ontology [28]. The framework skips eliminating them. Empirical evaluations [32,33] shows that real-life ontologies contain relatively fewer cyclic definers.

Theorem 2. *Let $\mathcal{O}$ be an ontology, $\mathcal{F}$ a forgetting signature of concept names, $\mathcal{O}^d$ the deductively reduced ontology of $\mathcal{O}$ with respect to $\mathcal{F}$, and $\mathcal{V}^d$ the ontology obtained by eliminating definers from $\mathcal{O}^d$. The following hold. (i) The two ontologies $\mathcal{O}^d$ and $\mathcal{V}^d$ are model inseparable with respect to $sig(\mathcal{O})\backslash\mathcal{F}$. (ii) The two ontologies $\mathcal{O}$ and $\mathcal{V}^d$ are deductively inseparable with respect to $sig(\mathcal{O})\backslash\mathcal{F}$. (iii) If $\mathcal{V}^d$ does not use definers, then $\mathcal{V}^d$ is the deductive forgetting view of $\mathcal{O}$ with respect to $\mathcal{F}$.*

A distinguishing feature of the fine-grained framework is the computation of the set Δ^d, which contains the clauses of $\mathcal{O}^{rp}$ that are not in $\mathcal{O}^d$. These clauses indicate the content difference between $\mathcal{O}$ and $\mathcal{O}^d$. For example, consider the clause $\neg D_1 \sqcup \neg D_2 \in \Delta^d$ in Example 1. It indicates that the set of r-successors and the set of s-successors of elements in the interpretation of A are disjoint. An information that is not preserved in $\mathcal{O}^d$ and $\mathcal{V}^d$. The set Δ^d can thus give an understanding of the information relative to the non-forgetting symbols that is not preserved in the deductive forgetting views. We can use the Δ^d clauses to enhance $\mathcal{O}^d$ with more content. This reveals a spectrum of fine-grained views between the deductive and the semantic forgetting views.

6 The Query Customization

We present the query customization process, which is an application of the fine-grained process explained above. The first step in this process is the *query reduction* step which takes $\mathcal{O}^{int}$, and computes a snapshot $\mathcal{O}^q$ of $\mathcal{O}^d$ enhanced with clauses from Δ^d such that $\mathcal{O}$ and $\mathcal{O}^q$ are deductively and query inseparable. The second step eliminates the non-cyclic definers from $\mathcal{O}^q$.

6.1 Query Reduction

First, we compute $\mathcal{O}^q$ which is the union of $\mathcal{O}^d$ and a portion of Δ^d to be specified. For this we need to differentiate between existential and universal clauses.

Definition 4. *Let $\mathcal{O}^{int}$ be the intermediate ontology, and $D \in N_d$ a definer occurring in $\mathcal{O}^{int}$. We say D is an* existential (universal) definer *if it occurs with positive polarity only below existential (universal) role restrictions in $\mathcal{O}^{int}$.*

Let Δ^d be the set of clauses from $\mathcal{O}^{int}$ with two or more negative definers, and suppose $\mathcal{C} \in \Delta^d$. We say that $\mathcal{C}$ is existential *if it contains at least one* *existential definer with negative polarity, and* universal *if all negative definers in $\mathcal{C}$ are universal. By Δ^e we denote the set of all existential clauses in Δ^d, and by Δ^u the set of universal definers in Δ^d.*

We now have the following.

$$\mathcal{O}^q = (\mathcal{O}^{rp} \backslash \Delta^d) \cup \Delta^u \tag{1}$$

This means that the query reduced ontology $\mathcal{O}^q$ is just the union of $\mathcal{O}^d$ and Δ^u. The set Δ^e indicates the content difference between $\mathcal{O}^{rp}$ and $\mathcal{O}^q$. Since $\mathcal{O}^{rp}$ and $\mathcal{O}^{int}$ are logically equivalent, it indicates the content difference between $\mathcal{O}^{int}$ and $\mathcal{O}^q$.

Lemma 1. *Let $\mathcal{O}$ be an $\mathcal{ALC}$ ontology, $\mathcal{F}$ a forgetting signature consisting of concept names, and $\mathcal{O}^q$ the query reduced ontology obtained by (1). Then $\mathcal{O}$ and $\mathcal{O}^q$ are query inseparable and deductively inseparable with respect to the signature $sig(\mathcal{O}) \backslash \mathcal{F}$.*

6.2 Definer Elimination

There are three types of definers that may occur in $\mathcal{O}^q$: *cyclic definers, simple definers,* and *complex definers.* As described in Section 4, a definer is cyclic if it occurs in at least one clause of $\mathcal{O}^q$ with positive and negative polarities. We skip the elimination of cyclic definers for the existence and practicality reasons as explained in Section 4.

The remaining definers are either simple, or complex definers, which we define next.

Definition 5. *We say D is a* complex definer*, if D is not a cyclic definer, and one of the following is true about D. (i) D occurs with negative polarity in a clause from the set Δ^u. (ii) There is a clause of the following form in $\mathcal{O}^q$.*

$$C \sqcup \neg D \sqcup \forall r_1.D_1 \sqcup \dots \sqcup \forall r_n.D_n, \tag{2}$$

where $D_1, \dots, D_n$ are complex definers, and $n \geq 1$. We use N_d^+ to denote the set of complex definers.

We say D is a simple definer *if it occurs negatively only in clauses where no other negative definer is used, and (ii) from above does not apply for D.*

Simple definers are eliminated using the method described in the definer elimination process of the deductive customization. That is, they are purified if they occur in $\mathcal{O}^q$ with single polarity, and eliminated using the simple definer elimination rule from Figure 2 when they occur with positive and negative polarities. The ontology obtained by eliminating simple definers from $\mathcal{O}^q$ is denoted by $\mathcal{V}^q_{ALC}$. It is the $\mathcal{ALC}$ representation of the query forgetting view.

Theorem 3. *Let $\mathcal{O}$ be an ontology, $\mathcal{F}$ a signature of concept names, $\mathcal{O}^q$ the query reduced ontology of $\mathcal{O}$ with respect to $\mathcal{F}$, and $\mathcal{V}^q_{ALC}$ the ontology obtained by eliminating simple definers from $\mathcal{O}^q$. The following hold.*

1. *$\mathcal{O}^q$ and $\mathcal{V}^q_{ALC}$ are model inseparable with respect to $sig(\mathcal{O})\backslash\mathcal{F}$.*
2. *$\mathcal{O}$ and $\mathcal{V}^q_{ALC}$ are query inseparable with respect to $sig(\mathcal{O})\backslash\mathcal{F}$.*
3. *$\mathcal{O}$ and $\mathcal{V}^q_{ALC}$ are deductively inseparable with respect to $sig(\mathcal{O})\backslash\mathcal{F}$.*
4. *If no definers remain in $\mathcal{V}^q_{ALC}$, then $\mathcal{V}^q_{ALC}$ is a query and deductive forgetting view of $\mathcal{O}$ with respect to $\mathcal{F}$.*

From a computational point of view, the time to compute $\mathcal{V}^q_{ALC}$ only differs from the time to compute $\mathcal{V}^d$ by the time consumed to compute Δ^u. The results of the empirical evaluation in [33] of the framework has shown that computing $\mathcal{V}^d$ is 2 times faster than using the state-of-the-art forgetting tool Lethe [20], the benchmark for many forgetting tools [2,3,41], on real life ontologies from the Bioportal repository [29,30,37]. Since the clauses of Δ^u are syntactically defined, extracting Δ^u is linear in the size of Δ^d. As such, we may expect similar performance when computing $\mathcal{V}^q_{ALC}$ to that when computing $\mathcal{V}^d$ which is suggested to be practical in real life as noted. We attribute the simplicity of computing $\mathcal{O}^q$ and consequently $\mathcal{V}^q_{ALC}$ to the granularity of the axioms of $\mathcal{O}^{int}$, which in turn can be attributed to the normal form used in the framework.

Complex definers are harder to eliminate and require additional reasoning. We also allow role inverse to be used to find solutions more often. We categorize complex definers in two categories, *type 1* and *type 2* complex definers. The elimination of type 1 complex definers preserves model inseparability up to the eliminated definers, but the elimination of type 2 complex definers only preserves query inseparability.

Definition 6. *Let D_1 be a complex definer. We say D is a* type 2 (complex) definer *if and only if there is a clause of the form $C \sqcup \forall r_1.D_1 \sqcup \forall r_2.D_2 \sqcup \cdots \sqcup \forall r_n.D_n$, where C is a concept, $r_1, \ldots, r_n \in N_r$ are role names, $D_1, D_2, \ldots, D_n$ are complex definers, $n \geq 2$, and one of the following is true. (i) D is a definer from the set $D_1, \ldots, D_n$. (ii) D occurs with negative polarity in C. A complex definer D is a* type 1 (complex) definer *if it occurs positively only in clauses of the form $C \sqcup \forall r.D$ such that C does not contain negative definers.*

The rules for eliminating complex definers are listed in Figure 3. Only the LB rule is needed for eliminating type 1 definers, whereas the three rules are needed for eliminating type 2 definers. We will explain the rules first and their intuition, before explaining the elimination process.

The *upper bound extraction (UB)* rule computes clauses where definers are used as top level negative literals. The form of conclusions of UB inferences is $D \sqsubseteq C$, where C is the concept $\forall s_1^-.C_2 \sqcup \cdots \sqcup \forall s_m^-.C_2 \sqcup E_1 \sqcup \cdots \sqcup E_m$. The concept C can be seen as an upper bound on D, because if a domain element is in the extension of D then it is also in the extension of C. The first and second premises of the rule are clauses where at least two complex definers occur positively below universal role restrictions. The third premise is a set of m clauses. Each uses a pair of negative definers D and D_j, where D is from the first premise, and D_j is from the second premise. To understand the UB rule, think of the premises as

Upper Bound Extraction (UB) $$\frac{C_2 \sqcup \forall s_1.D_1 \sqcup \cdots \sqcup \forall s_m.D_m, \quad \bigcup_{j=1}^{m} \neg D \sqcup \neg D_j \sqcup E_j}{\neg D \sqcup \forall s_1^-.C_2 \sqcup \cdots \sqcup \forall s_m^-.C_2 \sqcup E_1 \sqcup \cdots \sqcup E_m}$$

where $D, D_1, \ldots, D_m$ are type 2 definers, D occurs in a clause $C_1 \sqcup \forall r_1.D \sqcup \forall r_2.D'$ with the type 2 definer D', $m \geq 2$, negative definers do not occur in C_1, and definers do not occur in C_2.

Lower Bound Extraction (LB) $$\frac{C \sqcup \forall r_1.D_1 \sqcup \cdots \sqcup \forall r_n.D_n, \quad \bigcup_{k=1}^{n} \{\neg D_k \sqcup C_k\}}{D_i \sqcup \forall r_i^-.(C \sqcup \bigsqcup_{j=1, j\neq i}^{n} \forall r_j.C_j)}$$

where $1 \leq i \leq n$, $n \geq 1$, $\{D_1, \ldots, D_n\} \subseteq N_d^+$, definers do not occur in C, and negative definers do not occur in C_k.

Relaxing $$\frac{C \sqcup \forall r_1.D_1 \sqcup \cdots \sqcup \forall r_n.D_n, \quad \neg D_1 \sqcup E}{C \sqcup \forall r_1.E \sqcup \forall r_2.D_2 \sqcup \cdots \sqcup \forall r_n.D_n}$$

where $n \geq 2$, and C and E are concepts that do not use definers.

Fig. 3: Complex definer elimination rules

inclusions.

(3) $$\neg C_2 \sqsubseteq \forall s_1.D_1 \sqcup \cdots \sqcup \forall s_m.D_m$$

(4) $$\bigcup_{j=1}^{m} \{D \sqcap D_j \sqsubseteq E_j\}$$

(5) $$D \sqcap \exists s_1^-.\neg C_2 \sqcap \cdots \sqcap \exists s_m^-.\neg C_2 \sqsubseteq E_1 \sqcup \cdots \sqcup E_m$$

The premises present as (3) and (4), and the conclusion as (5). Suppose $\mathcal{I}$ is a model, and $e \in \Delta^{\mathcal{I}}$ is a domain element. Assume e is in the interpretation of the concept on the left hand side of the inclusion in (5). From the conjuncts $\exists s_i^-.\neg C_2$ with $1 \leq i \leq m$, there must exist a domain element $d \in \Delta^{\mathcal{I}}$ such that $d \notin C_2^{\mathcal{I}}$ and $(d, e) \in s_i^{\mathcal{I}}$. From (3) we know that e is in the interpretation of one of $D_1, D_2, \ldots,$ or D_m. Since additionally $e \in D^{\mathcal{I}}$, we get from (4) that $e \in (E_1 \sqcup E_2 \sqcup \cdots \sqcup E_m)^{\mathcal{I}}$, which is expressed by the right hand side of the inclusion in (5).

The *lower bound extraction (LB)* rule computes clauses where a complex definer occurs positively as a top level literal. The conclusion of a LB inference can be written as $E \sqsubseteq D_i$, where D_i is a definer, and E is the negation of the concept $\forall r_i^-.(C \sqcup \bigsqcup_{j=1, j\neq i}^{n} \forall r_j.C_j)$. The concept E is a lower bound of D because a domain element in the extension of E is necessarily in the extension of D. The first premise of a LB inference is a clause where the complex definers $D_1, \ldots, D_n$ occur with positive polarity below universal role restrictions. We require the concept C_1 to be free of definers. This restriction on C_1 is important because $\mathcal{V}_{ALC}^q$ may contain a clause of the form (2). When C_1 contains a definer D', the restriction on C_1 becomes a blocking constraint forcing either the elimination of

D' if it is universal, or performing further inferences such that $D_1, \ldots, D_n$ are no longer complex definers.

The second premise of a LB inference is a set of n clauses of the form $\neg D_k \sqcup C_k$ where $1 \leq k \leq n$. When several clauses $\neg D_k \sqcup C_k^l$ are present for a value of k, where $l \geq 2$, we combine them, and use the combined clause instead. Clauses from the set Δ^u cannot be used in the second premise. This restriction prevents negative definers from occurring below universal role restriction in the conclusion of the rule. If for some D_k, with $1 \leq k \leq n$, no clause in the required form exists, or if D_k occurs negatively only in clauses from Δ^u, we use the clause $\neg D_k \sqcup \top$.

The conclusion of a LB inference is a clause in which the complex definer D_i occurs positively as a top level literal, and $1 \leq i \leq n$. Because of the use of role inverse, the conclusion is an $\mathcal{ALCI}$ concept. Let us explain the rule. Let us assume $i = 1$.

$$\neg C_1 \sqsubseteq \forall r_1.D_1 \sqcup \cdots \sqcup \forall r_n.D_n \tag{6}$$

$$\textstyle\bigcup_{i=2}^{n}\{D_i \sqsubseteq C_i\} \tag{7}$$

$$\exists r_1^-.(\neg C_1 \sqcap \sqcap_{i=2}^{n} \exists r_i.\neg C_i) \sqsubseteq D_1 \tag{8}$$

The axiom in (6) represents the first premise. (7) represents the second premise, and (8) represents the conclusion. Observe that we ignored the clause $D_1 \sqsubseteq C_1$ from (7) because it is not required to compute the conclusion. Suppose $\mathcal{I}$ is an interpretation, and a domain element e is in the interpretation of the left hand side of (8). There is an r_1-predecessor d of e, i.e., $(d, e) \in r_1^{\mathcal{I}}$. Furthermore, the following is true about d. (i) d is not in the interpretation of C_1. (ii) For each i where $2 \leq i \leq n$, d has a successor e_i through r_i, and e_i is not in the interpretation of C_i. From (i) and (6) we know that there is j where $1 \leq j \leq n$ where every r_j successor of d is in the interpretation of D_j. From (ii) and (7) we know that j is not in the range from 2 to n. Therefore, d is in the interpretation of $\forall r_1.D_1$. Since e is a r_1 successor of d, we get that e is in the interpretation of D_1, which is the right hand side of (8).

The *relaxing* rule is a form of resolution. The first premise is a clause of the form $C \sqcup \forall r_1.D_1 \sqcup \cdots \sqcup \forall r_n.D_n$, where $n \geq 2$ and C does not contain definers. The second premise is a clause of the form $D_1 \sqcup E$ where E does not contain definers, if there are m clauses $D_1 \sqcup E^i$ where $1 \leq i \leq m$, we combine them in the clause $\neg D_1 \sqcup (E^1 \sqcap \cdots \sqcap E^m)$ and use the combined clause. The conclusion is a clause that resembles the first premise but replaces D_1 with E.

Type 1 definers are eliminated iteratively. In each iteration, LB inferences are performed exhaustively on all clauses that use the definer positively. The conclusions of the LB inferences will have the definer as a top level literal. We then resolve the obtained conclusions with the Δ^u clauses where the definer occurs. When all possible resolutions have been performed, we remove all clauses that use the definer. The following example explains some further subtleties.

Example 2. Consider the ontology $\mathcal{O} = \{A \sqsubseteq \forall r.B \sqcap \forall s.\neg B\}$, and let $\mathcal{F} = \{B\}$ be a forgetting signature. The query reduced ontology $\mathcal{O}^q$ of $\mathcal{O}$ with respect to $\mathcal{F}$ is the set $\{\neg A \sqcup \forall r.D_1, \neg A \sqcup \forall s.D_2, \neg D_1 \sqcup \neg D_2\}$. Since $\mathcal{O}^q$ does not contain simple definers, we have $\mathcal{V}^q_{ALC} = \mathcal{O}^q$. The two definers D_1 and D_2 are type 1 definers. Suppose we eliminate D_1 first. We perform a LB inference with the clause $\neg A \sqcup \forall r.D_1$ being the first premise, and the clause $\neg D_1 \sqcup \top$ in the second premise. The conclusion is $D_1 \sqcup \forall r^-.\neg A$. We now resolve the obtained conclusion with the clause $\neg D_1 \sqcup \neg D_2$, obtain the clause $\neg D_2 \sqcup \forall r^-.\neg A$, and remove all clauses where D_1 occurs. In the second iteration, D_2 is eliminated. A LB inference is performed. The first premise of the inference is the clause $\neg A \sqcup \forall s.D_2$, and the second premise is $\neg D_2 \sqcup \forall r^-.\neg A$. The conclusion is $D_2 \sqcup \forall s^-.\neg A_2$, which we resolve with $\neg D_2 \sqcup \forall r^-.\neg A$ to obtain $\forall r^-.\neg A \sqcup \forall s^-.\neg A$. Next, the clauses, where D_2 is used, are removed. Since there are no more definers to eliminate, we obtain the query forgetting view $\mathcal{V}^q_{ALCI}$ of $\mathcal{O}$ with respect to $\mathcal{F}$ as the ontology consisting of the clause $\forall r^-.\neg A \sqcup \forall s^-.\neg A$.

Observe that the second premises of the performed LB inferences in the example are not used in computing the conclusion. The LB rule is typically used on the same premises to obtain n conclusions. Each has one definer as top level literal. However, each of the n inferences requires $n-1$ clauses in the second premise to compute the conclusion.

Type 2 definers are eliminated in the same way as type 1 definers but with an additional step which uses the UB rule to obtain clauses where definers occur as top level negative literals. The computed UB conclusions are then used in the second premise of the LB inferences. One restriction on the UB rule is that m clauses are required in the third premise. Each of them uses a different definer from the ones in the second premise. There may be cases where some but not all clauses of the third premise are available. In these cases, we use the Relaxing rule which may compute in versions of the second premise with less definers in them that can be used in UB inferences.

In the following example LB inferences are only possible when UB inferences have been performed.

Example 3. Consider the following ontology $\mathcal{O} = \{A_1 \sqsubseteq \forall r.B_1 \sqcup \forall r.B_2, A_2 \sqsubseteq \forall r.B_3 \sqcup \forall r.B_4, B_1 \sqcap B_3 \sqsubseteq C_1, B_1 \sqcap B_4 \sqsubseteq C_2, B_2 \sqcap B_3 \sqsubseteq C_3, B_2 \sqcap B_4 \sqsubseteq C_4\}$, and forgetting signature $\mathcal{F} = \{B_1, B_2, B_3, B_4\}$. The query reduced ontology $\mathcal{O}^q$ of $\mathcal{O}$ with respect to $\mathcal{F}$ is the set $\{\alpha_1, \alpha_2, \alpha_{1,3}, \alpha_{1,4}, \alpha_{2,3}, \alpha_{2,4}\}$, where $\alpha_1 = \neg A_1 \sqcup \forall r.D_1 \sqcup \forall r.D_2, \alpha_2 = \neg A_2 \sqcup \forall r.D_3 \sqcup \forall r.D_4, \alpha_{1,3} = \neg D_1 \sqcup \neg D_3 \sqcup C_1, \alpha_{1,4} = \neg D_1 \sqcup \neg D_4 \sqcup C_2, \alpha_{2,3} = \neg D_2 \sqcup \neg D_3 \sqcup C_3$, $\alpha_{2,4} = \neg D_2 \sqcup \neg D_4 \sqcup C_4$, and D_1, D_2, D_3, and D_4 are complex type 2 definers.

The conclusions of any LB inference would be tautologous, because the second premise of each such inference would only use tautologous clauses. For instance, suppose we perform one inference with α_1 as first premise and $\neg D_2 \sqcup \top$ in the second premise. The conclusion is $\neg D_1 \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.\top)$ which is a tautology. We can however perform UB inferences and use their conclusions in the second premises of LB inferences.

Four UB inferences can be performed. The first and second inferences use α_2 as a first premise. The second premise of the two inferences is $\{\alpha_{1,3}, \alpha_{1,4}\}$ and $\{\alpha_{2,3}, \alpha_{2,4}\}$ respectively. The conclusions are $ub_1 = \neg D_1 \sqcup \forall r^-.\neg A_2 \sqcup C_1 \sqcup C_2$ and $ub_2 = \neg D_2 \sqcup \forall r^-.\neg A_2 \sqcup C_3 \sqcup C_4$, respectively. The third and fourth inferences use

α_1 as a first premise. The second premise of the two inference is $\{\alpha_{1,3}, \alpha_{2,3}\}$ and $\{\alpha_{1,4}, \alpha_{2,4}\}$ respectively. The conclusions are $ub_3 = \neg D_3 \sqcup \forall r^-.\neg A_1 \sqcup C_1 \sqcup C_3$ and $ub_4 = \neg D_4 \sqcup \forall r^-.\neg A_1 \sqcup C_2 \sqcup C_4$, respectively.

Next, four LB inferences are performed. The first inference has α_1 and ub_2 as the first and second premises respectively, and concludes $D_1 \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_3 \sqcup C_4))$. Similarly, the second inference has α_1 and ub_1, and concludes $D_2 \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_1 \sqcup C_2))$. The third inference has α_2 and ub_4, and concludes $D_3 \sqcup \forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_2 \sqcup C_4))$. The fourth inference has α_2 and ub_3, and concludes $D_4 \sqcup \forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_1 \sqcup C_3))$.

Next, we resolve exhaustively on the definers D_1, D_2, D_3, and D_4. When resolution is complete, the premises of the resolution inferences are removed. The four definers will remain only positively in the clauses α_1 and α_2, and will be purified by replacing them with $\top$. The final query forgetting view $\mathcal{V}^q_{ALCI}$ will consist of the clauses below.

$$\forall r^-.\neg A_2 \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_3 \sqcup C_4)) \sqcup C_1 \sqcup C_2$$
$$\forall r^-.\neg A_2 \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_1 \sqcup C_2)) \sqcup C_3 \sqcup C_4$$
$$\forall r^-.\neg A_1 \sqcup \forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_2 \sqcup C_4)) \sqcup C_1 \sqcup C_3$$
$$\forall r^-.\neg A_1 \sqcup \forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_1 \sqcup C_3)) \sqcup C_2 \sqcup C_4$$
$$\forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_2 \sqcup C_4)) \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_3 \sqcup C_4)) \sqcup C_1$$
$$\forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_2 \sqcup C_4)) \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_1 \sqcup C_2)) \sqcup C_3$$
$$\forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_1 \sqcup C_3)) \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_3 \sqcup C_4)) \sqcup C_2$$
$$\forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_1 \sqcup C_3)) \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_1 \sqcup C_2)) \sqcup C_4$$

Let us see how $\mathcal{V}^q_{ALCI}$ can be used to compute query answers. Consider the two ABoxes $\mathcal{A}_1$ and $\mathcal{A}_2$ modeled by the two graphs in Figure 4. Lower case letters represent individuals, and upper case labels represent concept assertions. For instance, we should understand from the left figure that $A_2(a_1)$ is an assertion in the ABox. For clarity, we abuse the notation by allowing negative assertions, for instance $\neg C_1(b_1)$. The assertion can be otherwise modeled by adding the axiom $E_2 \sqsubseteq \neg C_2$ to $\mathcal{O}$ (and $\mathcal{V}^q_{ALCI}$), and replacing it with the positive assertion $E_2(u)$. A directed arrow from u to v means that the role assertion $r(u, v)$ is in the ABox. Let us pose the query $q_1(x) = C_1(x)$ to the knowledge bases $(\mathcal{V}^q_{ALCI}, \mathcal{A}_1)$ and $(\mathcal{V}^q_{ALCI}, \mathcal{A}_2)$. The answers of the two knowledge bases to the query are b_1 and b_2 respectively, which coincide with those of $(\mathcal{O}, \mathcal{A}_1)$ and $(\mathcal{O}, \mathcal{A}_2)$. The answer b_1 is obtained from $(\mathcal{V}^q_{ALCI}, \mathcal{A}_1)$ directly from the first clause in the listing above, because b_1 is an instance of the negation of the concept $\forall r^-.\neg A_2 \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_3 \sqcup C_4)) \sqcup C_2$. It follows from $(\mathcal{O}, \mathcal{A})$ because a_2 is an instance of A_1. As such, the r-successor b_2 of a_2 is either an instance of B_1 or B_2. Assume it is B_2. Since b_2 is also a r-successor of a_3 which is an instance of A_2, we get that b_2 is either an instance of $B_2 \sqcap B_3$ or $B_2 \sqcap B_4$. From the last two axioms of $\mathcal{O}$, we get that b_2 is an instance of C_3 or C_4. Since neither of them are true, it can only be that all r successors of a_2 are instances of B_1. One such successor is b_1 which is also a successor of a_1. Since a_1 is an instance of A_2, we get that b_1

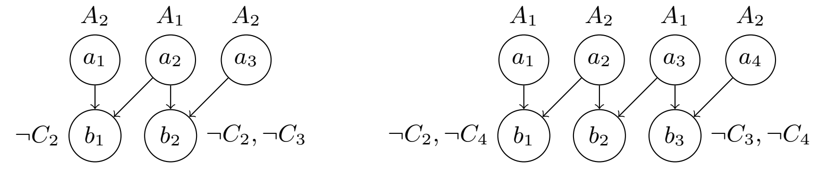


Fig. 4: Graph representation of ABox $\mathcal{A}_1$ (left) and ABox $\mathcal{A}_2$ (right).

is either an instance of C_1 or C_2. But we know from the ABox that it is not an instance of C_2, hence it is an instance of C_1.

The answer b_2 follows from $(\mathcal{V}^q_{ALCI}, \mathcal{A}_2)$ directly from the fifth clause of $\mathcal{V}^q_{ALCI}$ because it is an instance of the negation of the concept $\forall r^-.(\neg A_2 \sqcup \forall r.(\forall r^-.\neg A_1 \sqcup C_2 \sqcup C_4)) \sqcup \forall r^-.(\neg A_1 \sqcup \forall r.(\forall r^-.\neg A_2 \sqcup C_3 \sqcup C_4))$. It follows from $(\mathcal{O}, \mathcal{A}_2)$ because if all r-successors of a_3 are instances of B_2, then b_3 must be either an instance of C_3 or an instance of C_4. Since neither are true, we have that all r-successors of a_3 are instances of B_1, and b_2 is one such successor. In the same way, all r-successors of a_2, including b_2, must be instances of B_3. Hence b_2 is an instance of $B_1 \sqcap B_3$, and consequently an instance of C_1.

Theorem 4. *Let $\mathcal{O}$ be an ontology, $\mathcal{F}$ a forgetting signature of concept names, and $\mathcal{V}^q_{ALCI}$ the ontology computed as explained by the process above. The two ontologies $\mathcal{O}$ and $\mathcal{V}^q_{ALCI}$ are query inseparable with respect to the signature $sig(\mathcal{O})\backslash\mathcal{F}$. Furthermore, if there are no definers present in $\mathcal{V}^q_{ALCI}$, then $\mathcal{V}^q_{ALCI}$ is a query forgetting view of $\mathcal{O}$ with respect to $\mathcal{F}$.*

7 Conclusions

We presented a unified framework for forgetting concept names from $\mathcal{ALC}$ ontologies. The framework computes representations of the semantic, and the deductive forgetting views, and extracts sets of axioms indicating the information not preserved by the each. Not only does this give a new understanding of the content difference between semantic and deductive views, it also allows for computing fine-grained forgetting views in-between. We identified the query forgetting view as one such fine-grained view, and we computed two representations of it. An $\mathcal{ALC}$ representation which is also a representation for the deductive forgetting views, and an $\mathcal{ALCI}$ representations which eliminates further definers from the $\mathcal{ALC}$ representation. We also computed a set Δ^e of axioms indicating the content difference between the $\mathcal{ALC}$ representation of the query forgetting view and the semantic forgetting view. This enables the users who wish to process the forgetting view with $\mathcal{ALC}$ reasoning tools.

In future we plan to extend our framework with role forgetting, and evaluate the use of query forgetting views in real life ontology based data access applications.

References

1. Ackermann, W.: Untersuchungen über das Eliminationsproblem der mathematischen Logik. Mathematische Annalen **110**, 390–413 (1935)
2. Alassaf, R., Schmidt, R.A.: DLS-Forgetter: An Implementation of the DLS Forgetting Calculus for First-Order Logic. In: Calvanese, D., Iocchi, L. (eds.) GCAI 2019. Proceedings of the 5th Global Conference on Artificial Intelligence. EPiC Series in Computing, vol. 65, pp. 127–138. EasyChair (2019). https://doi.org/10.29007/hvz6, `https://easychair.org/publications/paper/TM2K`
3. Alassaf, R., Schmidt, R.A., Sattler, U.: Saturation-based uniform interpolation for multi-agent modal logics. In: Fernández-Duque, D., Palmigiano, A., Pinchinat, S. (eds.) Advances in Modal Logic, Volume 14. pp. 37–58. College Publications, London (2022), `http://www.aiml.net/volumes/volume14`
4. Baaz, M., Uwe, E., Leitsch, A.: Normal form transformations. In: Handbook of Automated Reasoning, pp. 275 – 332. North-Holland (2001)
5. Botoeva, E., Konev, B., Lutz, C., Ryzhikov, V., Wolter, F., Zakharyaschev, M.: Inseparability and conservative extensions of description logic ontologies: A survey. In: Reasoning Web 2016, pp. 27–89. Springer (2017)
6. Botoeva, E., Kontchakov, R., Ryzhikov, V., Wolter, F., Zakharyaschev, M.: Games for query inseparability of description logic knowledge bases. Artificial Intelligence **234**, 78–119 (2016). https://doi.org/https://doi.org/10.1016/j.artint.2016.01.010, `https://www.sciencedirect.com/science/article/pii/S0004370216300017`
7. Botoeva, E., Lutz, C., Ryzhikov, V., Wolter, F., Zakharyaschev, M.: Query inseparability for ALC ontologies. Artificial Intelligence **272**, 1–51 (2019). https://doi.org/10.1016/j.artint.2018.09.003
8. Delgrande, J.: A Knowledge Level Account of Forgetting. Artificial Intelligence Research **60**, 1165–1213 (2017). https://doi.org/10.1613/jair.5530
9. Delgrande, J.P., Wang, K.: A Syntax-Independent Approach to Forgetting in Disjunctive Logic Programs. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. pp. 1482–1488. AAAI'15, AAAI Press (2015)
10. Eiter, T., Kern-Isberner, G.: A Brief Survey on Forgetting from a Knowledge Representation and Reasoning Perspective. KI - Künstliche Intelligenz **33**(1), 9–33 (2019). https://doi.org/10.1007/s13218-018-0564-6, `http://link.springer.com/10.1007/s13218-018-0564-6`
11. Gabbay, D., Ohlbach, H.J.: Quantifier elimination in second-order predicate logic. In: Proc. KR 1992. pp. 425–435. Morgan Kaufmann (1992)
12. Gabbay, D., Schmidt, R.A., Szałas, A.: Second-Order Quantifier Elimination. College Publications (2008)
13. Ghilardi, S.: An algebraic theory of normal forms. Ann. Pure Appl. Logic **71**(3), 189 – 245 (1995)
14. Ghilardi, S., Lutz, C., Wolter, F.: Did I damage my ontology? a case for conservative extensions in description logic. In: Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning. pp. 187–197. KR'06, AAAI Press (2006), `http://dl.acm.org/citation.cfm?id=3029947.3029974`
15. Henkin, L.: An Extension of the Craig-Lyndon Interpolation Theorem. The Journal of Symbolic Logic **28**(3), 201–216 (1963), `http://www.jstor.org/stable/2271066`
16. Herzig, A., Mengin, J.: Uniform interpolation by resolution in modal logic. In: Proceedings of the 11th European Conference on Logics in Artificial Intelligence. vol. 5293 LNAI, pp. 219–231. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)

17. Konev, B., Kontchakov, R., Ludwig, M., Schneider, T., Wolter, F., Zakharyaschev, M.: Conjunctive query inseparability of OWL 2 QL TBoxes. In: Proceedings of the National Conference on Artificial Intelligence. vol. 1, pp. 221–226. AAAI Press, San Francisco (2011)
18. Konev, B., Lutz, C., Walther, D., Wolter, F.: Model-theoretic inseparability and modularity of description logic ontologies. Artificial Intelligence **203**, 66–103 (2013). https://doi.org/10.1016/j.artint.2013.07.004
19. Konev, B., Walther, D., Wolter, F.: Forgetting and uniform interpolation in large-scale description logic terminologies. In: Proc. IJCAI 2009. p. 830–835. Morgan Kaufmann (2009)
20. Koopmann, P., Schmidt, R.A.: Implementation and evaluation of forgetting in ALC-ontologies. In: Proc. WoMo 2013. vol. 1081. CEUR-WS.org (2013)
21. Koopmann, P., Schmidt, R.A.: Uniform interpolation of $\mathcal{ALC}$-ontologies using fixpoints. In: Proc. FroCoS 2013. LNAI, vol. 8152, pp. 87–102. Springer (2013)
22. Koopmann, P., Schmidt, R.A.: Count and forget: Uniform interpolation of $\mathcal{SHQ}$-ontologies. In: Proc. IJCAR 2014. LNAI, vol. 8562, pp. 434–448. Springer (2014)
23. Koopmann, P., Schmidt, R.A.: Saturation-based forgetting in the description logic $\mathcal{SIF}$. In: Proc. DL 2015. CEUR (2015)
24. Lang, J., Liberatore, P., Marquis, P.: Propositional Independence - Formula-Variable Independence and Forgetting. Journal of Artificial Intelligence Research **18**, 391–443 (jun 2003). https://doi.org/10.1613/jair.1113, `https://jair.org/index.php/jair/article/view/10330`
25. Lin, F., Reiter, R.: Forget it! In: AAAI Fall Symp. on Relevance. pp. 154–159 (1994)
26. Ludwig, M., Konev, B.: Towards practical uniform interpolation and forgetting for $\mathcal{ALC}$ TBoxes. In: Proc. DL. CEUR (2013)
27. Lutz, C., Wolter, F.: Deciding inseparability and conservative extensions in the description logic $\mathcal{EL}$. Symbolic Computation **45**, 194–228 (02 2010). https://doi.org/10.1016/j.jsc.2008.10.007
28. Lutz, C., Wolter, F.: Foundations for uniform interpolation and forgetting in expressive description logics. In: Proc. IJCAI 2011. p. 989–995 (2011)
29. Matentzoglu, N., Bail, S., Parsia, B.: A snapshot of the owl web. In: Proc. ISWC. pp. 331–346. Springer (2013)
30. Musen, M., Noy, N., Shah, N., Whetzel, P., Chute, C., Story, M.A., Smith, B.: The national center for biomedical ontology. Journal of the American Medical Informatics Association : JAMIA **19**, 190–5 (11 2011). https://doi.org/10.1136/amiajnl-2011-000523
31. Nonnengart, A., Weidenbach, C.: Computing small clause normal forms. In: Handbook of Automated Reasoning, pp. 335 – 367. North-Holland (2001)
32. Sakr, M., Schmidt, R.A.: Semantic forgetting in expressive description logics. In: Proc. FroCos. pp. 118–136. Springer (2021)
33. Sakr, M., Schmidt, R.A.: Fine-grained forgetting for the description logic alc. In: Proceedings of the 35th International Workshop on Description Logics. CEUR (2022)
34. Visser, A.: Uniform interpolation and layered bisimulation, LNL, vol. 6, p. 139–164. Springer (1996)
35. Wang, K., Wang, Z., Topor, R., Pan, J.Z., Antoniou, G.: Eliminating Concepts and Roles from Ontologies in Expressive Description Logics. Computational Intelligence **30**(2), 205–232 (May 2014). https://doi.org/10.1111/j.1467-8640.2012.00442.x, `http://doi.wiley.com/10.1111/j.1467-8640.2012.00442.x`

36. Wang, Z., Wang, K., Topor, R., Pan, J.Z., Antoniou, G.: Uniform Interpolation for ALC Revisited. In: AI 2009: Advances in Artificial Intelligence. pp. 528–537. Melbourne, Australia (2009), http://www.w3.org/TR/2004/REC-owl-ref-20040210/
37. Whetzel, P.L., Noy, N., Shah, N.H., Alexander, P.R., Nyulas, C., Tudorache, T., Musen, M.A.: Bioportal: enhanced functionality via new web services from the national center for biomedical ontology to access and use ontologies in software applications. Nucleic Acids Research **39**, W541 – W545 (2011)
38. Zhang, Y., Zhou, Y.: Knowledge forgetting: Properties and applications. Artificial Intelligence **173**, 1525–1537 (2009). https://doi.org/10.1016/j.artint.2009.07.005
39. Zhang, Y., Zhou, Y.: Forgetting revisited. In: Proc. KR. AAAI Press (2010)
40. Zhao, Y.: Automated Semantic Forgetting For Expressive Description Logics. Ph.D. thesis, University of Manchester (2018), https://www.research.manchester.ac.uk/portal/files/67402446/FULL{_}TEXT.PDF
41. Zhao, Y., Feng, H., Alassaf, R., Del-Pinto, W., Schmidt, R.A.: The FAME Family-A Family of Reasoning Tools for Forgetting in Expressive Description Logics. In: Proceedings of the 30th International Workshop on Description Logics. Montpellier (2017), http://www.cs.man.ac.uk/
42. Zhao, Y., Schmidt, R.A.: Concept forgetting in alcoi-ontologies using an ackermann approach. In: Arenas, M., Corcho, O., Simperl, E., Strohmaier, M., d'Aquin, M., Srinivas, K., Groth, P.T., Dumontier, M., Heflin, J., Thirunarayan, K., Staab, S. (eds.) The Semantic Web, 14th International Semantic Web Conference, ISWC 2015. Lecture Notes in Computer Science, vol. 9366, pp. 587–602. Springer (2015)
43. Zhao, Y., Schmidt, R.A.: Forgetting concept and role symbols in $\mathcal{ALCOIH}\mu^+(\nabla,\sqcap)$-ontologies. In: Kambhampati, S. (ed.) Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI 2016). pp. 1345–1352. AAAI Press/IJCAI (2016), http://www.ijcai.org/Proceedings/16/Papers/194.pdf
44. Zhao, Y., Schmidt, R.A.: Role forgetting for $\mathcal{ALCOQH}(\nabla)$-ontologies using an ackermann approach. In: Sierra, C. (ed.) Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI'17). pp. 1354–1361. AAAI Press/IJCAI (2017). https://doi.org/10.24963/ijcai.2017/188
45. Zhao, Y., Schmidt, R.A.: On concept forgetting in description logics with qualified number restrictions. In: Lang, J. (ed.) Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI'18). pp. 1984–1990. AAAI Press/IJCAI (2018), https://www.ijcai.org/proceedings/2018/0274.pdf