Embracing Change by Abstraction
Materialization Maintenance for Large ABoxes

Markus Brenner and **Birte Glimm**
University of Ulm, Germany
markus.brenner@uni-ulm.de, birte.glimm@uni-ulm.de

Abstract

Abstraction Refinement is a technique which allows for reducing materialization of an ontology with a large ABox to materialization of a smaller (compressed) 'abstraction' of this ontology. In this paper, we show how Abstraction Refinement can be adopted for incremental ABox materialization by combining it with the well-known DRed algorithm for materialization maintenance. The combination is non-trivial and to preserve correctness, already Horn $\mathcal{ALCHI}$ requires more complex abstractions. Nevertheless, we show that significant benefits can be obtained for synthetic and real-world ontologies.

1 Introduction

Most ontology reasoners support the task of materialization (i.e., they compute and explicitly store all entailed atomic concept and role assertions for the individuals in the ontology), which allows for directly evaluating conjunctive instance queries. Computing the materialization is computationally expensive and approaches such as summarization [Dolby *et al.*, 2009], ABox modularization [Wandelt and Möller, 2012], or Abstraction Refinement [Glimm *et al.*, 2014; 2017] attempt to "compress" the size of the dataset over which the materialization is computed to be able to materialize large ABoxes. Even using efficient materialization techniques, recomputing all consequences whenever input data changes can cause a significant delay before user queries can be answered again, which might be prohibitive for some application scenarios. Incremental maintenance algorithms originating from the database and Datalog communities (see, e.g., [Volz *et al.*, 2003; Motik *et al.*, 2015]) have been applied to description logics and the semantic web for incremental classification [Kazakov and Klinov, 2013], incremental materialization via Datalog [Volz *et al.*, 2005] and RDF stream reasoning [Barbieri *et al.*, 2010]. Apart from the technique used in the RDFox system [Motik *et al.*, 2014], the mentioned materialization algorithms focus on and are optimized for ontologies that can be expressed in the form of (Datalog) rules, i.e., proper existentials are not supported. Furthermore, how incremental maintenance algorithms can be combined with data compression techniques is an open problem, only addressed in a sketchy way by Steigmiller et al. [Steigmiller *et al.*, 2015] in the form of a representative cache maintaining individuals in an incremental fashion.

We address this problem by extending the Abstraction Refinement method s.t. abstractions capture information beyond the simple presence of assertions. This ultimately allows us to deal with changes of the underlying ABox. To show the correctness of the obtained theoretical framework, we devise novel proof techniques. Our empirical evaluation with synthetic and real-world ontologies shows up to four times improved materialization times compared to the approach without Abstraction Refinement in different change scenarios.

2 Preliminaries

The syntax of $\mathcal{ALCHI}$ is defined using a vocabulary consisting of countably infinite disjoint sets $\mathsf{N_C}$ of *atomic concepts*, $\mathsf{N_R}$ of *atomic roles*, and $\mathsf{N_I}$ of *individuals*. A *role* is either an atomic role or an *inverse role* r^- with $r \in \mathsf{N_R}$. We denote atomic roles with lower case and (possibly non-atomic) roles with upper case letters. We define $R^- := r^-$ if $R = r$ and $R^- := r$ if $R = r^-$. An $\mathcal{ALCHI}$ concept is defined as

$$C ::= \top \mid \bot \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \forall R.C \mid \exists R.C,$$

where $A \in \mathsf{N_C}$ and R is a role. Let C, D be concepts, R, S roles and a, b individuals. A *TBox* is a set of concept and role inclusion axioms of the form $C \sqsubseteq D$ and $R \sqsubseteq S$, respectively. An *ABox* is a set of (concept and role) assertions of the form $C(a)$ and $R(a, b)$, respectively. An *ontology* $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ consists of an ABox and a TBox. To simplify the presentation, we do not distinguish between $R(a, b)$ and $R^-(b, a)$ as well as $R \sqsubseteq S$ and $R^- \sqsubseteq S^-$. We use $\mathsf{con}(\mathcal{O})$, $\mathsf{rol}(\mathcal{O})$, and $\mathsf{ind}(\mathcal{O})$ for the sets of atomic concepts, atomic roles, and individuals occurring in $\mathcal{O}$, respectively.

An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consists of a non-empty set $\Delta^\mathcal{I}$, the *domain* of $\mathcal{I}$, and an *interpretation function* $\cdot^\mathcal{I}$, that assigns to each $A \in \mathsf{N_C}$ a subset $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, to each $R \in \mathsf{N_R}$ a binary relation $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, and to each $a \in \mathsf{N_I}$ an element $a^\mathcal{I} \in \Delta^\mathcal{I}$. This assignment is extended to roles by $(r^-)^\mathcal{I} = \{\langle e, d\rangle \mid \langle d, e\rangle \in r^\mathcal{I}\}$ and, inductively, to complex concepts as $\top^\mathcal{I} = \Delta^\mathcal{I}$, $\bot^\mathcal{I} = \emptyset$, $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$, $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$, $(C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$, and

$$(\forall r.C)^\mathcal{I} = \{d \in \Delta^\mathcal{I} \mid \langle d, e\rangle \in R^\mathcal{I} \rightarrow e \in C^\mathcal{I}\},$$
$$(\exists r.C)^\mathcal{I} = \{d \in \Delta^\mathcal{I} \mid e \in C^\mathcal{I} : \langle d, e\rangle \in R^\mathcal{I}\}.$$

An interpretation $\mathcal{I}$ *satisfies* an axiom α, written $\mathcal{I} \models \alpha$, if

$\alpha = C \sqsubseteq D$ and $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $\alpha = R \sqsubseteq S$ and $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$, $\alpha = C(a)$ and $a^{\mathcal{I}} \in C^{\mathcal{I}}$, or $\alpha = R(a, b)$ and $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$; $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$, written $\mathcal{I} \models \mathcal{O}$, if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$; $\mathcal{O}$ *entails* an axiom α, written $\mathcal{O} \models \alpha$, if every model of $\mathcal{O}$ satisfies α.

An $\mathcal{ALCHI}$ ontology $\mathcal{O}$ is *Horn* [Krötzsch *et al.*, 2013] and in *normalized* form if, for every $C(a) \in \mathcal{O}$, $C \in \mathsf{N_C}$ and, every $C \sqsubseteq D \in \mathcal{O}$, is in one of the forms $\top \sqsubseteq A$, $A \sqsubseteq B$, $A \sqsubseteq \exists R.B$, $A \sqsubseteq \bot$, $A \sqcap B \sqsubseteq C$, $A \sqsubseteq \forall R.B$ where $A, B, C \in \mathsf{N_C}$ and R is a role. W.l.o.g., we assume in the remainder every ontology as normalized by applying a structural transformation; see e.g. [Kazakov, 2009].

For $\mathcal{ALCHI}$ ontologies, determining entailed role assertions can easily be done using a precomputed role hierarchy. Hence, we focus on the task of computing entailed concept assertions: For an ontology $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, we say that $\mathcal{A}$ is *materialized* w.r.t. $\mathcal{T}$, if $\mathcal{O} \models A(a)$ implies $A(a) \in \mathcal{A}$ for each $A \in \mathsf{con}(\mathcal{O})$ and $a \in \mathsf{ind}(\mathcal{O})$. The materialization of $\mathcal{A}$ w.r.t. $\mathcal{T}$ is the smallest super-set of $\mathcal{A}$ that is materialized w.r.t. $\mathcal{T}$.

2.1 Abstraction Refinement

The main idea of the Abstraction Refinement method [Glimm *et al.*, 2014; 2017] is to materialize an ontology $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with a potentially large ABox $\mathcal{A}$ by constructing a smaller ABox $\mathcal{B}$ s.t. the materialization of $\mathcal{B} \cup \mathcal{T}$ can be computed by a general-purpose reasoner, and transferring the new entailments back to $\mathcal{O}$. The ABox $\mathcal{B}$ is usually called the *abstraction* of the *original* ABox $\mathcal{A}$. Homomorphisms are used for transferring back entailments:

Definition 1. *For $\mathcal{A}$ and $\mathcal{B}$ ABoxes, a mapping $h\colon \mathsf{ind}(\mathcal{B}) \to \mathsf{ind}(\mathcal{A})$ is called a* homomorphism (from $\mathcal{B}$ to $\mathcal{A}$) *if $h(\mathcal{B}) = \cup_{\alpha \in \mathcal{B}} h(\alpha) \subseteq \mathcal{A}$, where $h(C(a)) := C(h(a))$ and $h(R(a, b)) := R(h(a), h(b))$. An individual $b \in \mathsf{ind}(\mathcal{B})$ is a* representative *of an individual $a \in \mathsf{ind}(\mathcal{A})$ if there exists a homomorphism $h\colon \mathsf{ind}(\mathcal{B}) \to \mathsf{ind}(\mathcal{A})$ s.t. $h(b) = a$.*

The following property enables the transfer of entailments:

Lemma 1. *Let $h\colon \mathsf{ind}(\mathcal{B}) \to \mathsf{ind}(\mathcal{A})$ be a homomorphism between the ABoxes $\mathcal{B}$ and $\mathcal{A}$. Then, for every TBox $\mathcal{T}$ and every axiom β, $\mathcal{B} \cup \mathcal{T} \models \beta$ implies $\mathcal{A} \cup \mathcal{T} \models h(\beta)$.*

Abstractions are based on asserted roles and concepts for single individuals using *types*:

Definition 2 (Type)**.** *Let $\mathcal{A}$ be an ABox and a an individual. The* concept type *of a w.r.t. $\mathcal{A}$ is a set of concepts $\tau_C(a) = \{C \mid C(a) \in \mathcal{A}\}$. The* role type *of a w.r.t. $\mathcal{A}$ is a set of roles $\tau_R(a) = \{R \mid \exists b : R(a, b) \in \mathcal{A}\}$. The* (combined) type *of a w.r.t. $\mathcal{A}$ is a pair $(\tau_C(a), \tau_R(a))$, where $\tau_C(a)$ and $\tau_R(a)$ are the concept and role type of a w.r.t. $\mathcal{A}$, respectively.*

Example 1. *Let $\mathcal{A} = \{A(a), A(b), r(a, b)\}$. Then $\tau_C(a) = \tau_C(b) = \{A\}$, $\tau_R(a) = \{r\}$, $\tau_R(b) = \{r^-\}$, $\tau(a) = \tau_1 = \langle \{A\}, \{r\} \rangle$, and $\tau(b) = \tau_2 = \langle \{A\}, \{r^-\} \rangle$.*

The abstract ABox is then constructed by introducing one representative for each type with the respective assertions.

Definition 3 (Abstraction)**.** *Let $\tau = \langle \tau_C, \tau_R \rangle$ be a type. The* abstraction for *τ is an ABox*

$$\mathcal{B}_\tau = \{C(v_\tau) \mid C \in \tau_C\} \cup \{R(v_\tau, w_\tau^R) \mid R \in \tau_R\},$$

where v_τ and w_τ^R are distinguished abstract individuals *for the type τ. The* abstraction *of an ABox $\mathcal{A}$ is $\bigcup_{a \in \mathsf{ind}(\mathcal{A})} \mathcal{B}_{\tau(a)}$, where $\tau(a)$ is the type for a w.r.t. $\mathcal{A}$.*

Function `update`($\Delta\mathcal{B}$, $\mathcal{A}$)
  In: $\Delta\mathcal{B}$: abstract assertions, $\mathcal{A}$: ABox
  Out: $\Delta\mathcal{A}$: update using $\Delta\mathcal{B}$
  $\Delta\mathcal{A} \leftarrow \emptyset$
  for $C(v_{\tau(a)}) \in \Delta\mathcal{B}$ **do**
    $\Delta\mathcal{A} \leftarrow \Delta\mathcal{A} \cup C(a)$
  for $R(a, b) \in \mathcal{A}$ *and* $C(w^R_{\tau(a)}) \in \Delta\mathcal{B}$ **do**
    $\Delta\mathcal{A} \leftarrow \Delta\mathcal{A} \cup C(b)$
  return $\Delta\mathcal{A}$

Algorithm 1: `materializeAR`($\mathcal{O}$)
  In: $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$: ontology
  Out: $\mathcal{A}^\infty$: materialization of $\mathcal{A}$ w.r.t. $\mathcal{T}$
1 $\mathcal{A}^\infty \leftarrow \mathcal{A}$, $\Delta\mathcal{A} \leftarrow \emptyset$
2 **repeat**
3 $\mathcal{A}^\infty \leftarrow \mathcal{A}^\infty \cup \Delta\mathcal{A}$ *// not needed in loop round 1*
4   Let $\mathcal{B}$ be the abstraction of $\mathcal{A}$ (Definition 3)
5   Let $\mathcal{B}^\infty$ be the materialization of $\mathcal{B}$ w.r.t. $\mathcal{T}$
6 $\Delta\mathcal{B} \leftarrow \mathcal{B}^\infty \setminus \mathcal{B}$, $\Delta\mathcal{A} \leftarrow$ `update`($\Delta\mathcal{B}$, $\mathcal{A}$)
7 **until** $\Delta\mathcal{A} \subseteq \mathcal{A}^\infty$
8 **return** $\mathcal{A}^\infty$

Example 2. *The abstraction for $\mathcal{A}$ in Example 1 is $\mathcal{B} = \mathcal{B}_{\tau(a)} \cup \mathcal{B}_{\tau(b)}$, where $\mathcal{B}_{\tau(a)} = \mathcal{B}_{\tau_1} = \{A(v_{\tau_1}), r(v_{\tau_1}, w^r_{\tau_1})\}$, $\mathcal{B}_{\tau(b)} = \mathcal{B}_{\tau_2} = \{A(v_{\tau_2}), r^-(v_{\tau_2}, w^{r^-}_{\tau_2})\}$.*

Intuitively, the abstraction is a disjoint union of ABoxes simulating combined types. Note that each mapping $h\colon \mathsf{ind}(\mathcal{B}) \to \mathsf{ind}(\mathcal{A})$ s.t. $h(v_\tau) \in \{a \in \mathsf{ind}(\mathcal{A}) \mid \tau(a) = \tau\}$, and $h(w_\tau^R) \in \{b \in \mathsf{ind}(\mathcal{A}) \mid R(h(v_\tau), b) \in \mathcal{A}\}$ is a homomorphism from $\mathcal{B}$ to $\mathcal{A}$, which allows for transferring entailments from the abstraction back to the original ABox (Lemma 1). For $\mathcal{B}^\infty$ the materialization of $\mathcal{B} \cup \mathcal{T}$ and $\Delta\mathcal{B} = \mathcal{B}^\infty \setminus \mathcal{B}$, the function `update`($\Delta\mathcal{B}$, $\mathcal{A}$) uses the above defined homomorphism to generate an *update using* $\Delta\mathcal{B}$.

Algorithm 1 then defines the *Abstraction Refinement method* for materializing a Horn $\mathcal{ALCHI}$ ontology. The loop is repeated until a fix-point is reached and the update contains no assertions not already present in the materialization. The method is sound, complete, and terminating for Horn $\mathcal{ALCHI}$ [Glimm *et al.*, 2014] and, with some extensions, even for Horn $\mathcal{SHOIF}$ [Glimm *et al.*, 2017].

3 Incremental Materialization

For computing the materialization, we compute the closure of the ABox assertions using a modified version of the materialization rules given by Glimm *et al.* (2017) for Horn $\mathcal{SHOIF}$ restricted to Horn $\mathcal{ALCHI}$ as shown in Figure 1. Premises are given above the horizontal line and the conclusions below. Side conditions are given after the colon and restrict the expressions to which the rules are applicable. For example, rule

$$\mathbf{R}_{\sqsubseteq}\ \frac{A_1(a)\ \cdots\ A_n(a)}{B(a)} : \mathcal{T} \models \bigsqcap_i A_i \sqsubseteq B$$

$$\mathbf{R}_{\forall}\ \frac{A_1(a)\ \cdots\ A_n(a)\ R(a,b)}{B(b)} : \mathcal{T} \models \bigsqcap_i A_i \sqsubseteq \forall S.B \text{ and } \mathcal{T} \models R \sqsubseteq S$$

Figure 1: Materialization rules ($A, B \in \mathsf{N_C}$, R, S roles, $a, b \in \mathsf{N_I}$)

$\mathbf{R}_{\sqsubseteq}$ produces one inference for each individual a and concepts $A_1, \ldots, A_n, B$ s.t. $\mathcal{T} \models A_0 \sqcap \ldots \sqcap A_n \sqsubseteq B$ with the premise $\{A_1(a), \ldots, A_n(a)\}$ and the conclusion $B(a)$. Note that TBox axioms are only used in side conditions and never as premises of the rules, which allows us to focus on ABox reasoning and leave TBox reasoning to a suitable reasoner. Theorem 1 can be shown in a very similar (but simpler) way to the proof used by Glimm *et al.* (2017).

Theorem 1. *The rules in Figure 1 are sound and complete for the materialization of a normalized Horn $\mathcal{ALCHI}$ ontology.*

3.1 The Delete/Rederive Algorithm

For the changes, we adopt the database view, in which only known ABox facts can be added or deleted. This already provides enough expressivity for many use cases, e.g. stream reasoning [Barbieri *et al.*, 2010]. Due to the monotonicity of DL reasoning, additions can be handled by simply 'continuing' materialization. Hence, we only focus on deletions and assume: Given a Horn $\mathcal{ALCHI}$ ontology $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, the materialization $\mathcal{A}^\infty$ of $\mathcal{A}$ w.r.t. $\mathcal{T}$, and a set $\mathcal{A}^- \subseteq \mathcal{A}$ of deletions, we want to determine the materialization of $\mathcal{A} \setminus \mathcal{A}^-$ w.r.t. $\mathcal{T}$ using $\mathcal{A}^\infty$. This is non-trivial and requires the identification and removal of assertions in $\mathcal{A}^\infty$ no longer derivable from $\mathcal{A} \setminus \mathcal{A}^-$. The Delete/Rederive (DRed) algorithm [Gupta *et al.*, 1993; Staudt and Jarke, 1996] initially overestimates the necessary deletions and then determines facts still derivable from $\mathcal{A} \setminus \mathcal{A}^-$. The overestimation is obtained by continuously (over-)deleting facts, which can be derived *using already deleted facts*. We formalize this using restricted rule applications.

Definition 4 (Restricted Derivation)**.** *Given a set of deduction rules $\mathcal{R}$, an ontology $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, and an ABox $\mathcal{A}'$, an axiom α can directly be derived from $\mathcal{O}$ under restriction $\mathcal{A}'$, written $\mathcal{O} \vdash^{\mathcal{R}}_{\mathcal{A}'} \alpha$, if α can be derived from $\mathcal{O}$ by a rule in $\mathcal{R}$ s.t. at least one of the premises is in $\mathcal{A}'$.*

In the remainder, we simply write $\mathcal{A} \cup \mathcal{T} \vdash_{\mathcal{A}'} \alpha$ and assume $\mathcal{R}$ to consist of the rules from Figure 1.

For the presentation of the DRed algorithm, we follow the presentation style of Motik *et al.* (2015), but we adapt it to a consequence-based calculus, to avoid complications with ensuring termination of a Datalog program in the presence of function symbols, which are required to handle existential quantifiers. DRed, formalized in Algorithm 2, consists of three consecutive phases: (1) compute the *overdeletion*, (2) *rederive* deletions with an alternative derivation in one step, (3) *reinsert* facts with alternative derivations. Phase (1) and (3) can directly be used to extend the Abstraction Refinement method for incremental materialization maintenance.

Function `overdelete` ($\mathcal{T}$, $\mathcal{A}^\infty$, $\mathcal{A}^-$)
- **In**: $\mathcal{T}$: TBox, $\mathcal{A}^\infty$: materialization, $\mathcal{A}^-$: deletions
- **Out**: $\mathcal{D}_{\text{all}}$: overdeletion w.r.t. $\mathcal{T}$
- $\mathcal{D}_{\text{new}} \leftarrow \mathcal{A}^-$, $\mathcal{D}_{\text{all}} \leftarrow \emptyset$, $\mathcal{D}_{\text{prev}} \leftarrow \mathcal{D}_{\text{new}} \setminus \mathcal{D}_{\text{all}}$
- **while** $\mathcal{D}_{prev} \neq \emptyset$ **do**
  - $\mathcal{D}_{\text{all}} \leftarrow \mathcal{D}_{\text{all}} \cup \mathcal{D}_{\text{prev}}$,
  - $\mathcal{D}_{\text{new}} \leftarrow \{\alpha \mid \mathcal{A}^\infty \cup \mathcal{T} \vdash_{\mathcal{D}_{\text{prev}}} \alpha\}$,
  - $\mathcal{D}_{\text{prev}} \leftarrow \mathcal{D}_{\text{new}} \setminus \mathcal{D}_{\text{all}}$
- **return** $\mathcal{D}_{all}$

Function `reinsert` ($\mathcal{T}$, $\mathcal{A}^\infty_{new}$, $\mathcal{D}_{new}$)
- **In**: $\mathcal{T}$: TBox, $\mathcal{A}^\infty_{\text{new}}$: overdeletion, $\mathcal{D}_{\text{new}}$: rederivation
- **Out**: $\mathcal{A}^\infty_{\text{new}}$: materialization of $\mathcal{A} \setminus \mathcal{A}^-$ w.r.t. $\mathcal{T}$
- $\mathcal{D}_{\text{prev}} \leftarrow \mathcal{D}_{\text{new}} \setminus \mathcal{A}^\infty_{\text{new}}$
- **while** $\mathcal{D}_{prev} \neq \emptyset$ **do**
  - $\mathcal{A}^\infty_{\text{new}} \leftarrow \mathcal{A}^\infty_{\text{new}} \cup \mathcal{D}_{\text{prev}}$,
  - $\mathcal{D}_{\text{new}} \leftarrow \{\alpha \mid \mathcal{A}^\infty_{\text{new}} \cup \mathcal{T} \vdash_{\mathcal{D}_{\text{prev}}} \alpha\}$,
  - $\mathcal{D}_{\text{prev}} \leftarrow \mathcal{D}_{\text{new}} \setminus \mathcal{A}^\infty_{\text{new}}$
- **return** $\mathcal{A}^\infty_{new}$

Hence, we define the according functions `overdelete` and `reinsert`, which are then used in Algorithm 2. The computed set of deletions $\mathcal{D}_{\text{all}}$ in the function `overdelete` can be seen as a sort of closure for $\mathcal{A}^\infty \cup \mathcal{T} \vdash_{\mathcal{D}_{\text{all}}} \alpha$. The search for facts with an alternative derivation in the reinsertion phase is restricted to assertions in the overdeletion set $\mathcal{D}_{\text{all}}$ using an additional premise for the derivation rules. The computed set $\mathcal{D}_{\text{new}}$ is then used in Phase (3) by the function `reinsert` to restore the correct materialization. Rule applicability is restricted to avoid unnecessary rule applications. Although $\mathcal{D}_{\text{all}}$ does not appear in the reinsertion phase, due to the way the overdeletion is generated, Phase (3) determines $\mathcal{A}^\infty_{\text{new}}$ as a closure for $\mathcal{A}^\infty_{\text{new}} \cup \mathcal{T} \vdash_{\mathcal{A}^\infty_{\text{new}} \cap \mathcal{D}_{\text{all}}} \alpha$.

3.2 DRed and Abstraction Refinement

We adopt DRed in the general Abstraction Refinement way: We construct, for each of the different phases, suitable abstractions of the ABox on which we perform the respective operations. Interleaved refinement steps (for overdeletion and reinsertion repeatedly until the fix-point) transfer results back to the original ABox and yield an adapted abstraction. Since DRed operations on abstractions additionally require knowledge about the set of deletions, we extend the original definitions to bi-types and bi-abstractions.

Definition 5 (Bi-Type)**.** *Given ABoxes $\mathcal{A}_1, \mathcal{A}_2$, the* bi-type *of an individual $a \in \mathsf{ind}(\mathcal{A}_1 \cup \mathcal{A}_2)$ w.r.t. $(\mathcal{A}_1, \mathcal{A}_2)$ is a quadruple $(\tau^1_C(a), \tau^1_R(a), \tau^2_C(a), \tau^2_R(a))$, where $(\tau^1_C(a), \tau^1_R(a))$ is the combined type of a w.r.t. $\mathcal{A}_1$ and $(\tau^2_C(a), \tau^2_R(a))$ is the combined type of a w.r.t. $\mathcal{A}_2$.*

Definition 6 (Bi-Abstraction)**.** *Given two ABoxes $\mathcal{A}_1$, $\mathcal{A}_2$ and a bi-type $\tau = (\tau^1_C, \tau^1_R, \tau^2_C, \tau^2_R)$ w.r.t. $(\mathcal{A}_1, \mathcal{A}_2)$, the* bi-abstraction for τ *is an ABox $\mathcal{B}^1_\tau \cup \mathcal{B}^2_\tau$, where $\mathcal{B}^1_\tau = \{C(v_\tau) \mid C \in \tau^1_C\} \cup \{R(v_\tau, w^R_\tau) \mid R \in \tau^1_R\}$, $\mathcal{B}^2_\tau = \{C(v_\tau) \mid C \in \tau^2_C\} \cup \{R(v_\tau, w^R_\tau) \mid R \in \tau^2_R\}$, and v_τ and w^R_τ are distinguished* abstract individuals *for the bi-type τ. The*

Algorithm 2: `DRed(`$\mathcal{O}$, $\mathcal{A}^\infty$, $\mathcal{A}^-$`)`

In: $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$: ontology, $\mathcal{A}^\infty$: materialization, $\mathcal{A}^-$: deletions
Out: $\mathcal{A}^\infty_{\text{new}}$: materialization of $\mathcal{A} \setminus \mathcal{A}^-$ w.r.t. $\mathcal{T}$
▷ *overdeletion phase* ◁
$\mathcal{D}_{\text{all}} \leftarrow$ `overdelete(`$\mathcal{O}$, $\mathcal{A}^\infty$, $\mathcal{A}^-$`)`
▷ *rederivation phase* ◁
$\mathcal{A}^\infty_{\text{new}} \leftarrow \mathcal{A}^\infty \setminus \mathcal{D}_{\text{all}}$, $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{all}} \cap (\mathcal{A} \setminus \mathcal{A}^-)$
Apply the rules (Fig. 1) to $\mathcal{A}^\infty_{\text{new}}$ w.r.t. $\mathcal{T}$ s.t. the conclusion is added to $\mathcal{D}_{\text{new}}$ and with the additional side condition that the conclusion occurs in $\mathcal{D}_{\text{all}}$
▷ *reinsertion phase* ◁
$\mathcal{A}^\infty_{\text{new}} \leftarrow$ `reinsert(`$\mathcal{O}$, $\mathcal{A}^\infty_{new}$, $\mathcal{D}_{new}$`)`
return $\mathcal{A}^\infty_{new}$

bi-abstraction of $(\mathcal{A}_1, \mathcal{A}_2)$ *is* $\mathcal{B} = \mathcal{B}^1 \cup \mathcal{B}^2$ *with*

$$\mathcal{B}^1 = \bigcup_{a \in \mathsf{ind}(\mathcal{A})} \mathcal{B}^1_{\tau(a)} \quad \textit{and} \quad \mathcal{B}^2 = \bigcup_{a \in \mathsf{ind}(\mathcal{A})} \mathcal{B}^2_{\tau(a)},$$

where $\tau(a)$ is the bi-type for a w.r.t. $(\mathcal{A}_1, \mathcal{A}_2)$ and $\mathcal{B}^1_{\tau(a)} \cup \mathcal{B}^2_{\tau(a)}$ is the bi-abstraction for $\tau(a)$.

The following example highlights how bi-abstractions also differentiate types based on their (over-)deleted assertions, while still aggregating 'similar' cases.

Example 3. *Let $\mathcal{A} = \{A(a_1), A(a_2), A(a_3), r(a_1,b), r(a_2,b), r(a_3,b)\}$ and $\mathcal{A}^- = \{A(a_1), A(a_3)\}$, the combined type of a_1, a_2, and a_3 w.r.t. $\mathcal{A}$ is $(\{A\}, \{r\})$. To distinguish a_1 and a_3 from a_2, we consider the bi-types w.r.t. $(\mathcal{A} \setminus \mathcal{A}^-, \mathcal{A}^-)$: $\tau(a_1) = \tau(a_3) = (\emptyset, \{r\}, \{A\}, \emptyset)$, $\tau(a_2) = (\{A\}, \{r\}, \emptyset, \emptyset)$, $\tau(b) = (\emptyset, \{r^-\}, \emptyset, \emptyset)$.*

Similar to the DRed algorithm, the Abstraction Refinement Delete and Rederive algorithm (`ARDRed`) consists of three consecutive phases: (1) compute the *overdeletion* using the function `overdelete` on abstractions followed by the refinement step until a fix-point is reached, (2) *rederive* deletions in the abstraction with an alternative derivation in one step, (3) *reinsert* facts with alternative derivations using Abstraction Refinement. For the reinsertion phase, we again restrict the derivation rules using the second part of bi-types, which represents the deleted assertions, as opposed to using the overdeletion set $\mathcal{D}_{\text{all}}$ in DRed.

We note two things about the overdeletion phase of Algorithm 3: First, the input ABox $\mathcal{B}^1 \cup \mathcal{B}^2$ for `overdelete` is not necessarily fully materialized, which can delay some derivations until the next loop round. Second, the function `update` uses the initial role assertions in $\mathcal{A}^\infty$, i.e. deletions are also propagated over deleted roles. The rederivation phase does not share parts with DRed (Algorithm 2), as we cannot restrict rederivations in a similar way, due to missing information about the concept deletions for role successors.

3.3 Correctness

It can easily be verified that the Abstraction Refinement aspects of the rederivation and reinsertion phases operate like the original Abstraction Refinement procedure and that the used rules are sound. Hence, soundness follows from the completeness of the overdeletion phase, i.e. we need to show that assertions no longer derivable from $\mathcal{A} \setminus \mathcal{A}^-$ are not present when starting the rederivation phase. We make the notion of overdeletion precise using ABox justifications:

Algorithm 3: `ARDRed(`$\mathcal{O}$, $\mathcal{A}^\infty$, $\mathcal{A}^-$`)`

In: $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$: ontology, $\mathcal{A}^\infty$: materialization, $\mathcal{A}^-$: deletions
Out: $\mathcal{A}^\infty_{\text{new}}$: materialization of $\mathcal{A} \setminus \mathcal{A}^-$ w.r.t. $\mathcal{T}$
▷ *overdeletion phase* ◁
$\mathcal{D}_{\text{new}} \leftarrow \mathcal{A}^-$, $\mathcal{D}_{\text{all}} \leftarrow \emptyset$
while $\mathcal{D}_{new} \setminus \mathcal{D}_{all} \neq \emptyset$ **do**
  $\mathcal{D}_{\text{all}} \leftarrow \mathcal{D}_{\text{all}} \cup \mathcal{D}_{\text{new}}$, $\mathcal{D}_{\text{new}} \leftarrow \emptyset$
  Let $\mathcal{B}^1 \cup \mathcal{B}^2$ be the *bi-abstraction* w.r.t. $(\mathcal{A}^\infty \setminus \mathcal{D}_{\text{all}}, \mathcal{D}_{\text{all}})$
  $\mathcal{E}_{\text{all}} \leftarrow$ `overdelete(`$\mathcal{T}$, $\mathcal{B}^1 \cup \mathcal{B}^2$, $\mathcal{B}^2$`)`
  $\mathcal{D}_{\text{new}} \leftarrow$ `update(`$\mathcal{E}_{all} \setminus \mathcal{B}^2$, $\mathcal{A}^\infty$`)`
▷ *rederivation phase* ◁
$\mathcal{A}^\infty_{\text{new}} \leftarrow \mathcal{A}^\infty \setminus \mathcal{D}_{\text{all}}$, $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{all}} \cap (\mathcal{A} \setminus \mathcal{A}^-)$, $\mathcal{E}_{\text{new}} \leftarrow \emptyset$
Let $\mathcal{B}^1 \cup \mathcal{B}^2$ be the *bi-abstraction* w.r.t. $(\mathcal{A}^\infty_{\text{new}}, \mathcal{D}_{\text{all}})$
Apply the rules (Fig. 1) to $\mathcal{B}^1$ w.r.t. $\mathcal{T}$ s.t. the conclusion is added to $\mathcal{E}_{\text{new}}$ and with the additional side condition for $\mathbf{R}_\sqsubseteq$ that the conclusion occurs in $\mathcal{B}^2$
$\mathcal{D}_{\text{new}} \leftarrow (\mathcal{D}_{\text{new}} \cup$ `update(`$\mathcal{E}_{new}$, $\mathcal{A}^\infty_{new}$`)`$) \cap \mathcal{D}_{\text{all}}$
▷ *reinsertion phase* ◁
while $\mathcal{D}_{new} \setminus \mathcal{A}^\infty_{new} \neq \emptyset$ **do**
  $\mathcal{A}^\infty_{\text{new}} \leftarrow \mathcal{A}^\infty_{\text{new}} \cup \mathcal{D}_{\text{new}}$
  Let $\mathcal{B}^1 \cup \mathcal{B}^2$ be the *bi-abstraction* w.r.t. $(\mathcal{A}^\infty_{\text{new}}, \mathcal{D}_{\text{all}})$
  $\mathcal{E}_{\text{new}} \leftarrow$`reinsert(`$\mathcal{T}$, $\mathcal{B}^1$, $\mathcal{B}^1 \cap \mathcal{B}^2$`)`
  $\mathcal{D}_{\text{new}} \leftarrow$ `update(`$\mathcal{E}_{new} \setminus \mathcal{B}^1$, $\mathcal{A}^\infty_{new}$`)`
return $\mathcal{A}^\infty_{new}$

Definition 7 (ABox Justification, Overdeletion)**.** *Let $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ be an ontology. For an axiom α s.t. $\mathcal{O} \models \alpha$, an* ABox justification *w.r.t. $\mathcal{T}$ is any set $J \subseteq \mathcal{A}$ s.t. $J \cup \mathcal{T} \models \alpha$; J is* minimal, *if $\forall J' \subset J : J' \cup \mathcal{T} \not\models \alpha$.*

Let $\mathcal{A}^\infty$ be the materialization of $\mathcal{A}$ w.r.t. $\mathcal{T}$ and $\mathcal{A}^- \subseteq \mathcal{A}$ deletions. Any superset of the set $\{\alpha \in \mathcal{A}^\infty \mid J \cap \mathcal{A}^- \neq \emptyset$ for $J \subseteq \mathcal{A}$ a minimal ABox justification for α w.r.t. $\mathcal{T}\}$ is an overdeletion *of $\mathcal{A}^\infty$ w.r.t. $\mathcal{A}^-$.*

For the overdeletion and reinsertion phases, we execute functions `overdelete` and `reinsert` over bi-abstractions and use homomorphisms to transfer the obtained results. Both functions construct closures via restricted derivations. For our argumentation, we consider two important properties induced by restricted derivations:

Lemma 2 (Restricted Derivation Properties)**.** *Let $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ be an ontology and $\mathcal{A}_1, \mathcal{A}_2$ ABoxes with $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\mathsf{ind}(\mathcal{A}_1) \subseteq \mathsf{ind}(\mathcal{A})$ and $\mathcal{B}$ an ABox, s.t. there is a homomorphism $h : \mathsf{ind}(\mathcal{A}) \to \mathsf{ind}(\mathcal{B})$ from $\mathcal{A}$ to $\mathcal{B}$. Then the following properties hold for any assertion α:*

$$\mathcal{A} \cup \mathcal{T} \vdash_{\mathcal{A}_1} \alpha \textit{ implies } \mathcal{A} \cup \mathcal{T} \vdash_{\mathcal{A}_2} \alpha. \tag{1}$$

$$\mathcal{A} \cup \mathcal{T} \vdash_{\mathcal{A}_1} \alpha \textit{ implies } \mathcal{B} \cup \mathcal{T} \vdash_{h(\mathcal{A}_1)} h(\alpha). \tag{2}$$

The overdeletion/reinsertion phases terminate after reaching the fix-point, where no further information can be transferred using homomorphisms. We show that, in the fix-point, the closures constructed over bi-abstractions can be lifted up to closures over the original ABoxes. For the inner overdeletion (line 5), we have $\mathcal{B}^1 \cup \mathcal{B}^2$ as the bi-abstraction w.r.t. $(\mathcal{A}^\infty \setminus \mathcal{D}_{\text{all}}, \mathcal{D}_{\text{all}})$ (line 4). The inner overdeletion (line 5) extends $\mathcal{B}^2$ with assertions α to determine a closure over $\mathcal{B}^1 \cup \mathcal{B}^2 \cup \mathcal{T} \vdash_{\mathcal{B}^2} \alpha$. For the inner reinsertion, we have $\mathcal{B}^1 \cup \mathcal{B}^2$ as the bi-abstraction w.r.t. $(\mathcal{A}^\infty_{\text{new}}, \mathcal{D}_{\text{all}})$ (line 13). The inner reinsertion (line 14) constructs a closure for $\mathcal{B}^1 \cup \mathcal{T} \vdash_{\mathcal{B}^1 \cap \mathcal{B}^2} \alpha$. Note that, compared to the closure constructed by `reinsert` in the DRed algorithm, this is actually a restriction since (bi-)abstractions only contain concept assertions for individuals of the form v_τ. Still, this restriction does not affect the final result. We can lift these closures up to $\mathcal{A}^\infty \cup \mathcal{T} \vdash_{\mathcal{D}_{\text{all}}} \alpha$ for the overdeletion and $\mathcal{A}^\infty_{\text{new}} \cup \mathcal{T} \vdash_{\mathcal{A}^\infty_{\text{new}} \cap \mathcal{D}_{\text{all}}} \alpha$ for the reinsertion. Lemma 3 provides the proof for the overdeletion phase. The proof for the reinsertion phase is analogous.

Lemma 3 (Overdeletion Fix-Point)**.** *Let $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ be a normalized Horn $\mathcal{ALCHI}$ ontology with materialization $\mathcal{A}^\infty$ and deletions $\mathcal{A}^- \subseteq \mathcal{A}$. Further, let $\mathcal{D}_{all}$ be s.t. $\mathcal{A}^- \subseteq \mathcal{D}_{all} \subseteq \mathcal{A}^\infty$ and $\mathcal{B} = \mathcal{B}^1 \cup \mathcal{B}^2$ the bi-abstraction w.r.t. $(\mathcal{A}^\infty \setminus \mathcal{D}_{all}, \mathcal{D}_{all})$.*

For each individual $a \in \mathsf{ind}(\mathcal{A})$ with bi-type $\tau = (\tau^1_C, \tau^1_R, \tau^2_C, \tau^2_R)$, each atomic concept $A \in \mathsf{con}(\mathcal{O})$, and each role $R \in \mathsf{rol}(\mathcal{O})$, if

1. *$\mathcal{B}^1 \cup \mathcal{B}^2 \cup \mathcal{T} \vdash_{\mathcal{B}^2} A(v_\tau)$ implies $A(a) \in \mathcal{D}_{all}$ and*
2. *$\mathcal{B}^1 \cup \mathcal{B}^2 \cup \mathcal{T} \vdash_{\mathcal{B}^2} A(w^R_\tau)$ and $R(a, b) \in \mathcal{A}$ implies $A(b) \in \mathcal{D}_{all}$,*

then, for each assertion α s.t. $\mathcal{A}^\infty \cup \mathcal{T} \vdash_{\mathcal{D}_{all}} \alpha$, $\alpha \in \mathcal{D}_{all}$.

Sketch. Extend $\mathcal{B} = \mathcal{B}^1 \cup \mathcal{B}^2$ to $\mathcal{B}' = \mathcal{B}^{1'} \cup \mathcal{B}^{2'}$ as follows: If $r(a, b) \in \mathcal{A}^\infty \setminus \mathcal{D}_{\text{all}}$, extend $\mathcal{B}^1$ with $r(v_{\tau(a)}, v_{\tau(b)})$; if $r(a, b) \in \mathcal{A}^-$, extend $\mathcal{B}^2$ with $r(v_{\tau(a)}, v_{\tau(b)})$. Note that there is a homomorphism h from $\mathcal{A}^\infty$ to $\mathcal{B}'$, thus, by Lemma 2 (2), if $\mathcal{A}^\infty \cup \mathcal{T} \vdash_{\mathcal{D}_{\text{all}}} \alpha$, then $\mathcal{B}' \cup \mathcal{T} \vdash_{h(\mathcal{D}_{\text{all}})} h(\alpha)$. We can then show that $h(\mathcal{D}_{\text{all}}) \subseteq \mathcal{B}^{2'}$, hence, by $\mathcal{B}' \cup \mathcal{T} \vdash_{\mathcal{B}^{2'}} h(\alpha)$.

We next sketch why $\mathcal{B}'$ has the same atomic concept assertions as direct derivations under restriction $\mathcal{B}^{2'}$ as $\mathcal{B}$ under restriction $\mathcal{B}^2$. Let $r(a, b) \in \mathcal{A}^\infty$. By considering the construction of $\mathcal{B}'$ from $\mathcal{B}$, the rules in Figure 1 and the conditions of restricted derivations, we can show that introducing the new role assertion $r(v_{\tau(a)}, v_{\tau(b)})$ does not result in new restricted entailments. Hence, according to condition 1, $A(a) \in \mathcal{D}_{\text{all}}$. □

Lemma 4 (Soundness)**.** *Let $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ be a Horn $\mathcal{ALCHI}$ ontology, $\mathcal{A}^\infty$ the materialization of $\mathcal{A}$ w.r.t. $\mathcal{T}$, $\mathcal{A}^- \subseteq \mathcal{A}$ a set of deletions, and $\mathcal{A}^\infty_{new}$ the result of* `ARDRed(`$\mathcal{O}$, $\mathcal{A}^\infty$, $\mathcal{A}^-$`)`. *If $\alpha \in \mathcal{A}^\infty_{new}$, then $(\mathcal{A} \setminus \mathcal{A}^-) \cup \mathcal{T} \models \alpha$.*

Sketch. We mainly need to show that the set $\mathcal{D}_{\text{all}}$ obtained as a closure according to Lemma 3 is complete and contains all assertions that are to be removed in the sense of Definition 7. To the contrary of what is to be shown, assume there is some $\alpha \notin \mathcal{D}_{\text{all}}$ s.t. a minimal ABox justification J for α contains an assertion from $\mathcal{A}^-$. By Theorem 1, α can be derived via a number of rule applications from J and, thus, there is a number of premises, of which at least one also has a minimal ABox justification $J' \subseteq J$ with $J' \cap \mathcal{A}^- \neq \emptyset$. By doing so repeatedly, we eventually determine some assertion β s.t. $\mathcal{A}^\infty \cup \mathcal{T} \vdash_{\mathcal{D}_{\text{all}}} \beta$, which is a contradiction. The ARDred algorithm removes $\mathcal{D}_{\text{all}}$ from $\mathcal{A}^\infty$ and, as shown earlier, only sound assertions are reinserted. □

We now show completeness using the previous results.

Lemma 5 (Completeness)**.** *Let $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ be a Horn $\mathcal{ALCHI}$ ontology, $\mathcal{A}^\infty$ the materialization of $\mathcal{A}$ w.r.t. $\mathcal{T}$, $\mathcal{A}^- \subseteq \mathcal{A}$ a set of deletions, and $\mathcal{A}^\infty_{new}$ the result of* `ARDRed(`$\mathcal{O}$, $\mathcal{A}^\infty$, $\mathcal{A}^-$`)`. *If $(\mathcal{A} \setminus \mathcal{A}^-) \cup \mathcal{T} \models \alpha$, then $\alpha \in \mathcal{A}^\infty_{new}$.*

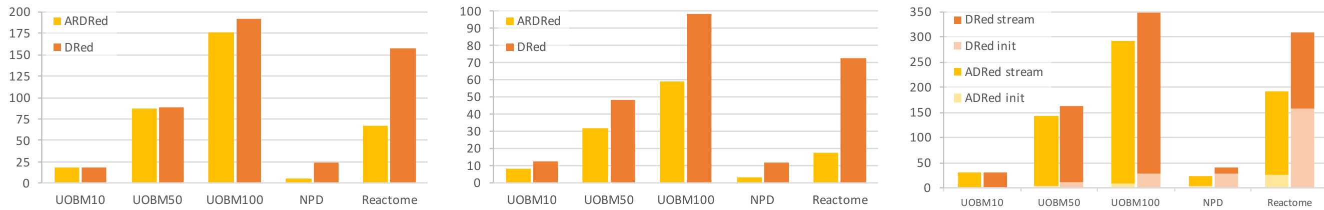
Sketch. First, we consider the inputs $\mathcal{A}^\infty_{\text{new}}$ and $\mathcal{D}_{\text{new}}$ as they are used in the closure construction of the reinsertion phase. By Lemma 4, $\mathcal{D}_{\text{all}}$ contains the overdeletion of $\mathcal{A}^\infty$ w.r.t. $\mathcal{A}^-$. For $\mathcal{A}^\infty_{\text{new}}$, we can then confirm that $\mathcal{A} \setminus \mathcal{A}^- \subseteq \mathcal{A}^\infty_{\text{new}}$ using the definition of the algorithm. For $\mathcal{D}_{\text{new}}$, we need to consider the effects of the rederivation phase. Using the rules in Figure 1 and the construction of the bi-abstraction $\mathcal{B}_1 \cup \mathcal{B}_2$, we can show that $\mathcal{D}_{\text{new}}$ is determined s.t. it contains at least all assertions, which can directly be rederived from $\mathcal{A}^\infty \setminus \mathcal{D}_{\text{all}}$. It remains to show that under these conditions, constructing $\mathcal{A}^\infty_{\text{new}}$ as a closure over $\mathcal{A}^\infty_{\text{new}} \cup \mathcal{T} \vdash_{\mathcal{A}^\infty_{\text{new}} \cap \mathcal{D}_{\text{all}}} \alpha$ results in $\mathcal{A}^\infty_{\text{new}}$ s.t. $(\mathcal{A} \setminus \mathcal{A}^-) \cup \mathcal{T} \models \alpha$ implies $\alpha \in \mathcal{A}^\infty_{\text{new}}$. Assuming the contrary, let α be some concept assertion, s.t. $(\mathcal{A} \setminus \mathcal{A}^-) \cup \mathcal{T} \models \alpha$ and $\alpha \notin \mathcal{A}^\infty_{\text{new}}$. Due to the initial construction of $\mathcal{A}^\infty_{\text{new}}$, there must be some 'initial culprit' $\beta \notin \mathcal{A}^\infty_{\text{new}}$, which can directly be derived from $\mathcal{A}^\infty_{\text{new}}$ using a rule from Figure 1. We can then show that the conditions of the construction of $\mathcal{A}^\infty_{\text{new}}$ are violated. □

Theorem 2 (Correctness)**.** *Let $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ be a Horn $\mathcal{ALCHI}$ ontology, $\mathcal{A}^\infty$ the materialization of $\mathcal{A}$ w.r.t. $\mathcal{T}$, $\mathcal{A}^- \subseteq \mathcal{A}$ a set of deletions.* `ARDRed(`$\mathcal{O}$, $\mathcal{A}^\infty$, $\mathcal{A}^-$`)` *terminates and the returned $\mathcal{A}^\infty_{new}$ is s.t. $\alpha \in \mathcal{A}^\infty_{new}$ iff $(\mathcal{A} \setminus \mathcal{A}^-) \cup \mathcal{T} \models \alpha$.*

Sketch. The overdeletion and reinsertion loops of the algorithm terminate and, once an iteration cannot determine new assertions, we obtain the termination of the algorithm as a direct consequence of the already shown soundness (Lemma 4) and completeness (Lemma 5) results. In particular, termination occurs in the worst case, when $\mathcal{D}_{\text{all}} = \mathcal{A}^\infty$ (overdeletion) and $\mathcal{A}^\infty_{\text{new}} = \mathcal{A}^\infty$ (reinsertion). □

4 Implementation and Evaluation

For the evaluation, we focus on directly comparing the algorithms, i.e. classical DRed and ARDRed. To do so, we have implemented both from scratch in Java including support for additions. The code and the experimental data are publicly available [Glimm and Brenner, 2018]. As basis for the test cases, we used the UOBM benchmark [Ma *et al.*, 2006], for which we generated instances of 10, 50 and 100 universities (denoted UOBM_{10}, UOBM_{50}, and UOBM_{100}, respectively) and the well-known NPD and Reactome ontologies. All ontologies were preprocessed by dropping non-Horn $\mathcal{ALCHI}$ axioms and by normalizing the remaining axioms. Table 1 shows the number of TBox and ABox axioms of the preprocessed ontologies.



(a) add-only case (b) add/remove case (c) remove-only case

Figure 2: Materialization time in s. For the remove-only case, the initial reasoning times over $\mathcal{A}_0$ (init) are noticeable (and, therefore, included) and the results for UOBM are scaled by 0.1 for visual clarity.

			add-only		add/remove		remove-only					
			ARDRed	DRed	ARDRed	DRed	ARDRed			DRed		
	$\lvert\mathcal{T}\rvert$	$\lvert\mathcal{A}\rvert$	stream		stream		init	stream	sum	init	stream	sum
UOBM_{10}	498	1926897	19.03	18.40	8.44	12.45	7.22	311.28	318.50	18.26	287.41	305.67
UOBM_{50}	498	9751681	87.88	89.29	32.22	48.17	42.22	1 401.40	1 443.62	114.84	1 503.09	1 617.94
UOBM_{100}	498	19571755	176.26	191.79	59.49	98.24	96.56	2 815.68	2 912.24	281.48	3 191.51	3 472.99
NPD	1241	911517	6.15	24.83	3.06	11.93	3.39	20.03	23.42	27.78	12.95	40.73
Reactome	597	7087410	67.43	157.04	17.93	72.98	25.83	165.29	191.12	158.94	150.44	309.38

Table 1: Test ontologies with the number of axioms ($|\mathcal{T}|$) and assertions ($|\mathcal{A}|$) for their Horn $\mathcal{ALCHI}$ subsets with the stream materialization times in seconds. Materialization time for $\mathcal{A}_0$ (init) is only given for the remove-only case (the times for the other cases are < 200 ms).

An incremental test consists of a TBox $\mathcal{T}$ and a base ABox $\mathcal{A}$ from which we obtain an initial ABox $\mathcal{A}_0$ and 100 change ABoxes containing assertions to be either added or deleted, depending on the test scenario. Each change ABox contains all assertions for 1% of the individuals from $\mathcal{A}$.[1] We evaluate three scenarios: (1) *add-only* starts with an almost empty $\mathcal{A}_0$ and step by step adds the change ABoxes; (2) *remove-only* starts with $\mathcal{A}_0 = \mathcal{A}$ and then removes the change ABoxes; (3) *add-remove* alternates between additions and removals, starting with two initial addition steps to avoid adding/removing the same individuals over and over again. All tests were executed on a server with two Intel hexa core processors at 2.60GHz (without using parallelization) and each test execution was assigned 200 GB of the overall 500 GB RAM to allow for keeping all test data in memory to avoid I/O impact. The number of 100 change ABoxes is large enough to even out JVM behavior (e.g. code optimization at runtime).

Results are shown in Table 1 and visualized in Figure 2. It can be seen that our approach outperforms the classical DRed implementation in almost all cases. We can also observe the typical Abstraction Refinement behavior, in that the advantage of ARDRed over DRed increases with the ABox size: For UOBM_{10}, DRed still outperforms ARDRed in the add-only and remove-only cases. Figure 2c shows initial reasoning times stacked under the streaming times to display accumulated results. When running an incremental reasoning system for a longer time period, initial reasoning times eventually become irrelevant. Considering this argument, ARDRed behaves slightly worse than DRed for NPD and Reactome in the *remove-only* case. Our analysis shows that this is due to the unoptimized bi-types, which currently require individuals to start with the same materialization and to receive the same deletions, to be classified into the same type and we presume that this restriction is too strong for practical ontologies. This does not seem to be a problem for the UOBM ontologies, which we attribute to the synthetic data generation. The improvement achieved by ARDRed over DRed is less noticeable than that for non-incremental materialization by Abstraction Refinement. We attribute this to a generally increased overhead of the abstraction construction and refinement aspects in the incremental setting, which is confirmed by the fact that the initial materialization times for $\mathcal{A}_0$ in the *remove-only* setting are comparable to those of the original Abstraction Refinement approach. The runtimes of the *add-remove* case are generally lower than the runtimes of the *add-only* case, which in return are lower than those of the *remove-only* case. The first can be explained by the general number of assertions, which is lower in the *add-remove* case (as we keep adding and removing the same number of individuals). The second result is also to be expected, as deletion of assertions is far more expensive than adding assertions.

[1] We exclude individuals which are used like nominals (implicitly belonging to the terminological part), e.g. the individual realizing the SwimmingClass concept in UOBM. Such individuals make up far less than 1% of the total individuals.

5 Conclusion and Future Work

The introduced novel materialization maintenance algorithm for Horn $\mathcal{ALCHI}$ ontologies combines DRed and Abstraction Refinement. Benefits of this approach lie in the summarization of similar deletion and reasoning tasks, paving the road for efficient maintenance of materializations of large ABoxes.

For the implementation, further optimizations for the Abstraction Refinement part (e.g. caching of types, further summarization of types in the overdeletion) and the DRed part (e.g. integrating the Backward/Forward optimization [Motik *et al.*, 2015]) are paths to be explored.

Acknowledgements

This work was done within the project "Reasoning over Large Amounts of Data in Ontologies via Abstraction and Refinement" (GL 779/3-2) and the Transregional Collaborative Research Centre SFB/TRR 62 "Companion-Technology for Cognitive Technical Systems" funded by the German Research Foundation (DFG).

References

[Barbieri *et al.*, 2010] Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, Emanuele Della Valle, and Michael Grossniklaus. Incremental reasoning on streams and rich background knowledge. In *Extended Semantic Web Conference*, pages 1–15. Springer, 2010.

[Dolby *et al.*, 2009] Julian Dolby, Achille Fokoue, Aditya Kalyanpur, Edith Schonberg, and Kavitha Srinivas. Scalable Highly Expressive Reasoner (SHER). *J. of Web Semantics*, 7(4):357–361, 2009.

[Glimm and Brenner, 2018] Birte Glimm and Markus Brenner. Dataset and Software for Abstraction Materialization Maintenance, April 2018. Zenodo. https://doi.org/10.5281/zenodo.1229328.

[Glimm *et al.*, 2014] Birte Glimm, Yevgeny Kazakov, Thorsten Liebig, Trung-Kien Tran, and Vincent Vialard. Abstraction refinement for ontology materialization. In *Proc. of the 13th Int. Semantic Web Conference, ISWC 2014*, pages 180–195, 2014.

[Glimm *et al.*, 2017] Birte Glimm, Yevgeny Kazakov, and Trung-Kien Tran. Ontology materialization by abstraction refinement in horn SHOIF. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1114–1120. AAAI Press, 2017.

[Gupta *et al.*, 1993] Ashish Gupta, Inderpal Singh Mumick, and V. S. Subrahmanian. Maintaining views incrementally. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, SIGMOD '93, pages 157–166. ACM, 1993.

[Kazakov and Klinov, 2013] Yevgeny Kazakov and Pavel Klinov. Incremental reasoning in OWL EL without bookkeeping. In *Proceedings of the 12th International Semantic Web Conference (ISWC 2013)*, volume 8218 of *Lecture Notes in Computer Science*, pages 232–247. Springer, 2013.

[Kazakov, 2009] Yevgeny Kazakov. Consequence-Driven Reasoning for Horn $\mathcal{SHIQ}$ Ontologies. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence, IJCAI 2009*, pages 2040–2045, 2009.

[Krötzsch *et al.*, 2013] Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler. Complexities of Horn Description Logics. *ACM Trans. Comput. Log.*, 14(1), 2013.

[Ma *et al.*, 2006] Li Ma, Yang Yang, Zhaoming Qiu, Guotong Xie, Yue Pan, and Shengping Liu. Towards a complete owl ontology benchmark. *The Semantic Web: Research and Applications*, pages 125–139, 2006.

[Motik *et al.*, 2014] Boris Motik, Yavor Nenov, Robert Piro, Ian Horrocks, and Dan Olteanu. Parallel Materialisation of Datalog Programs in Centralised, Main-Memory RDF Systems. In *Proc. of the Twenty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2014*, pages 129–137, 2014.

[Motik *et al.*, 2015] Boris Motik, Yavor Nenov, Robert Edgar Felix Piro, and Ian Horrocks. Incremental update of datalog materialisation: the backward/forward algorithm. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 1560–1568. AAAI Press, 2015.

[Staudt and Jarke, 1996] Martin Staudt and Matthias Jarke. Incremental maintenance of externally materialized views. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 75–86. Morgan Kaufmann Publishers Inc., 1996.

[Steigmiller *et al.*, 2015] Andreas Steigmiller, Birte Glimm, and Thorsten Liebig. Completion graph caching for expressive description logics. In *Proceedings of the 28th International Workshop on Description Logics*, volume 1350 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[Volz *et al.*, 2003] Raphael Volz, Steffen Staab, and Boris Motik. Incremental maintenance of materialized ontologies. In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE - OTM Confederated International Conferences, CoopIS, DOA, and ODBASE*, volume 2888 of *Lecture Notes in Computer Science*, pages 707–724. Springer, 2003.

[Volz *et al.*, 2005] Raphael Volz, Steffen Staab, and Boris Motik. Incrementally maintaining materializations of ontologies stored in logic databases. *Journal on Data Semantics II*, pages 1–34, 2005.

[Wandelt and Möller, 2012] Sebastian Wandelt and Ralf Möller. Towards ABox Modularization of Semi-Expressive Description Logics. *J. of Applied Ontology*, 7(2):133–167, 2012.