Abstraction Refinement for Ontology Materialization

Birte Glimm[1], Yevgeny Kazakov[1], Thorsten Liebig[2], Trung-Kien Tran[1], and Vincent Vialard[2]

[1] University of Ulm, Ulm, Germany, <first name>.<last name>@uni-ulm.de
[2] derivo GmbH, Ulm, Germany, <lastname>@derivo.de

Abstract. We present a new procedure for ontology materialization (computing all entailed instances of every atomic concept) in which reasoning over a large ABox is reduced to reasoning over a smaller "abstract" ABox. The abstract ABox is obtained as the result of a fixed-point computation involving two stages: 1) abstraction: partition the individuals into equivalence classes based on told information and use one representative individual per equivalence class, and 2) refinement: iteratively split (refine) the equivalence classes, when new assertions are derived that distinguish individuals within the same class. We prove that the method is complete for Horn $\mathcal{ALCHOI}$ ontologies, that is, all entailed instances will be derived once the fixed-point is reached. We implement the procedure in a new database-backed reasoning system and evaluate it empirically on existing ontologies with large ABoxes. We demonstrate that the obtained abstract ABoxes are significantly smaller than the original ones and can be computed with few refinement steps.

1 Introduction

Ontology based data access (OBDA) is an increasingly popular paradigm in the area of knowledge representation and information systems. In ODBA, a TBox with background knowledge is used to enrich and integrate large, incomplete, and possibly semi-structured data, which users can then access via queries. To efficiently handle large data sets (called ABoxes), OBDA approaches assume that the data is stored in a database or triple store. Nevertheless, the assumption of complete data that is typically made in databases does not hold and reasoning is required to compute the (entailed) types of individuals or answers to queries in general.

Different reasoning approaches have been developed in the OBDA context: (i) Query rewriting or backward-chaining approaches answer a query, by "compiling" the background knowledge of the TBox into the query [2,12]. The analysis of which languages are FO rewritable (i.e., which queries can be answered by query rewriting) inspired the development of DL-Lite [2] and resulted in the OWL QL profile [11] of the Web Ontology Language OWL 2. (ii) Materialization or forward-chaining techniques take the opposite approach by pre-computing all entailed information upfront, independent of the queries [1,14,8]. After extending the ABox with all pre-computed facts, the unmodified queries can be evaluated over the enriched data only (i.e., without considering the schema). The idea of query answering via materialization is directly present in the OWL RL profile [11], which specifies a suitable set of materialization rules. (iii) Finally, also combined approaches have been proposed, which allow for smaller rewritten

queries by materializing some (but not all) entailments [10,9] or for computing the materialization dynamically as required for a given query.

In this paper, we focus on the materialization of entailed facts for large ABoxes that are stored in a (graph)database or triple store and where the schema is expressed in terms of a Horn $\mathcal{ALCHOI}$ ontology. For full OWL RL support, functionality and property chains have to be encoded, but Horn $\mathcal{ALCHOI}$ also goes beyond OWL RL (and OWL QL). For example, existential quantification (*owl:someValuesFrom*) is fully supported, which is a feature that is difficult for standard materialization and rewriting approaches. While the principle of materialization is conceptually simple, it requires considerable computational resources in particular for large ABoxes or expressive TBox languages. Furthermore, reasoners for the language we tackle, are typically main memory and refutation-based, i.e., to show that an individual a is an instance of the class C, the reasoner tries to derive a contradiction for the ontology (temporarily) extended with $\neg C(a)$ (asserting that a is *not* an instance of C). Consequently, handling large ABoxes directly is infeasible.

Our approach for handling large ABoxes is based on the assumption that individuals with similar asserted types are likely to have the same inferred types. We group such individuals into equivalence classes and compute the types just for one representative individual. For building the initial equivalence classes, we also consider the role (property) assertions in the ABox, but we do not simply merge individuals. Instead, we iteratively compute a so-called *abstraction* that contains one representative individual for each equivalence class plus representative individuals for its direct role successors and predecessors in the ABox. We show how derivations for the latter individuals can be used in the refinement process to split equivalence classes for individuals that no longer have the same assertions. The number of individuals in the abstraction is always bounded exponentially in the number of different concepts and roles and linearly in the size of the original ABox; hence the abstraction is relatively small when the number of individuals is much larger than the number of concepts and roles used in the ontology.

We implement the technique in a database-backed system that interacts with a highly optimized in-memory reasoner that materializes the abstract ABox. The database engine needs to support only simple operations and does not require any knowledge of the TBox. We show that the procedure is sound and it is complete for computing the entailed types of individuals in Horn $\mathcal{ALCHOI}$ ontologies.

The paper is structured as follows: We next introduce directly related approaches. In Section 3, we present some preliminaries and continue with the presentation of the theoretical foundation of our approach in Section 4. In Section 5, we prove completeness of our procedure. In Section 6, we evaluate the procedure on a range of real-world ontologies with large ABoxes, and conclude in Section 7. Full proofs and further details are available in a technical report [5].

2 Related Work

In this section, we focus on work that is closely related to our aim of abstracting the ABox. The SHER approach [4,3] merges similar individuals to obtain a compressed, so-called *summary* ABox, which is then used for (refutation-based) consistency check-

Table 1. The syntax and semantics of $\mathcal{ALCHOI}$

	Syntax	Semantics
Roles:		
atomic role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ (given by $\mathcal{I}$)
inverse role	R^{-}	$\{\langle d, e\rangle \mid \langle d, e\rangle \in R^{\mathcal{I}}\}$
Concepts:		
atomic concept	A	$A \subseteq \Delta^{\mathcal{I}}$ (given by $\mathcal{I}$)
nominal	o	$o^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, $\|o^{\mathcal{I}}\| = 1$ (given by $\mathcal{I}$)
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
existential restriction	$\exists R.C$	$\{d \mid \exists e \in C^{\mathcal{I}} : \langle d, e\rangle \in R^{\mathcal{I}}\}$
universal restriction	$\forall R.C$	$\{d \mid \forall e \in \Delta^{\mathcal{I}} : \langle d, e\rangle \in R^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$
Axioms:		
concept inclusion	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
role inclusion	$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$R(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in R^{\mathcal{I}}$

ing. The technique (as well as ours) is based on the observation that individuals with the same asserted types are likely to have the same entailed types. Since merging in SHER is only based on asserted concepts, the resulting summary ABox might be inconsistent even if the original ABox is consistent w.r.t. the TBox. To remedy this, justifications [6] are used to decide which merges caused the inconsistency and to refine the summary accordingly. Justification-based refinements are also needed for query answering since SHER is not a materialization approach and performs reasoning at query time. We avoid justification computation by partitioning individuals of the same type into equivalence classes. Such partitioning guarantees the soundness of derived atomic concept assertions. We also have to perform refinement steps, but the refinement is to incrementally derive more consequences. What is computed before remains sound.

Wandelt and Möller propose a technique for (refutation-based) instance retrieval over $\mathcal{SHI}$ ontologies based on modularization [15,16]. As an optimization and similar to our approach, they group individuals into equivalence classes based on the asserted types of an individual, its successors, predecessors and the asserted types of the successors and predecessors.[3] The assertions that define the equivalence class of an individual are used for finding sound entailments. For checking entailments that cannot be read-off from these assertions, it might be necessary to fall-back to (refutation-based) reasoning over the (possibly large) ABox module for the individual. Instead of falling back to modules of the original ABox, we propose an iterative refinement process for the equivalence classes. The refinement is based on semantic criteria, i.e., only when individuals are semantically distinguishable, we refine the equivalence class, whereas the modules defined by Wandelt and Möller are syntactically defined.

[3] We ignore the types of successors and predecessors to achieve larger equivalence classes.

3 Preliminaries

We first define the syntax and semantics of the Description Logic (DL) $\mathcal{ALCHOI}$, which is the main ontology language we consider in this paper.

The syntax of $\mathcal{ALCHOI}$ is defined using a vocabulary (signature) consisting of countably infinite disjoint sets N_C of *concept names*, N_O of *nominals*, N_R of *role names*, and N_I of *individual names*. Note that concepts are called classes and roles are called properties in OWL. Complex *concepts* and *axioms* are defined recursively in Table 1. An *ontology* $\mathcal{O}$ is a finite set of axioms and we often write $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, where $\mathcal{A}$ is an *ABox* consisting of the concept and role assertions in $\mathcal{O}$ and $\mathcal{T}$ a *TBox* consisting of the concept and role inclusions in $\mathcal{O}$. We use $\mathsf{con}(\mathcal{O})$, $\mathsf{rol}(\mathcal{O})$, $\mathsf{ind}(\mathcal{O})$ for the sets of concept names, role names, and individual names 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 $A \in N_C$ a subset $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, to each $o \in N_O$ a singleton subset $o^\mathcal{I} \subseteq \Delta^\mathcal{I}$, $\|o^\mathcal{I}\| = 1$, to each $R \in N_R$ a binary relation $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, and to each $a \in N_I$ an element $a^\mathcal{I} \in \Delta^\mathcal{I}$. This assignment is extended to complex concepts as shown in Table 1. $\mathcal{I}$ *satisfies* an axiom α (written $\mathcal{I} \models \alpha$) if the corresponding condition in Table 1 holds. $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$ (written $\mathcal{I} \models \mathcal{O}$) if $\mathcal{I}$ satisfies all axioms in $\mathcal{O}$. We say that $\mathcal{O}$ *is consistent* if $\mathcal{O}$ has a model. We say that $\mathcal{O}$ *entails* an axiom α (written $\mathcal{O} \models \alpha$), if every model of $\mathcal{O}$ satisfies α. We say that $\mathcal{O}$ is *concept materialized* if $A(a) \in \mathcal{O}$ whenever $\mathcal{O} \models A(a)$, $A \in \mathsf{con}(\mathcal{O})$ and $a \in \mathsf{ind}(\mathcal{O})$; $\mathcal{O}$ is *role materialized* if $R(a, b) \in \mathcal{O}$ whenever $\mathcal{O} \models R(a, b)$, $R \in \mathsf{rol}(\mathcal{O})$, $a, b \in \mathsf{ind}(\mathcal{O})$; $\mathcal{O}$ is *materialized* if it is both concept and role materialized.

Remark 1. Some definitions do not present nominals as primitive symbols, but use a special *nominal constructor* $\{a\}$ with individual a (in this case, $\{a\}^\mathcal{I} = \{a^\mathcal{I}\}$). We can easily convert such ontologies to our representation by renaming every nominal $\{a\}$ with the corresponding nominal symbol o_a and adding a concept assertion $o_a(a)$. This transformation is a conservative extension, i.e., it preserves all original entailments.

4 Computing ABox Materialization by Abstraction

The typical OBDA scenario is such that the ABox contains a large number of individuals and its size is significantly larger than the size of the TBox. Hence, the number of different concept names is typically much smaller than the number of different individuals, which also means that many individuals are instances of the same concepts. If we can identify individuals that yield the same consequences, we can compute the materialization by computing entailed consequences only for representative individuals.

4.1 Isomorphic Individuals and Individual Types

In order to (syntactically) characterize individuals that yield the same consequences, we study structure-preserving transformations of ABoxes.

Definition 1. *Let $\mathcal{A}$ and $\mathcal{B}$ be two ABoxes and $h\colon \mathsf{ind}(\mathcal{A}) \to \mathsf{ind}(\mathcal{B})$ a mapping from the individuals in $\mathcal{A}$ to individuals in $\mathcal{B}$. We extend h to axioms in a straightforward way:*

$h(C(a)) = C(h(a))$, $h(R(a,b)) = R(h(a), h(b))$, *and* $h(\alpha) = \alpha$ *for other axioms* α. *We say that* h *is a* homomorphism *(from* $\mathcal{A}$ *to* $\mathcal{B}$*) if* $h(\mathcal{A}) \subseteq \mathcal{B}$. *An individual* a *in* $\mathcal{A}$ *is* homomorphic *to an individual* b *in* $\mathcal{B}$ *if there exists a homomorphism* h *from* $\mathcal{A}$ *to* $\mathcal{B}$ *such that* $h(a) = b$*; in addition, if* b *is homomorphic to* a*, then* a *and* b *are* isomorphic.

Example 1. Consider the ABox $\mathcal{A} = \{R(a,a), R(a,b), R(b,b)\}$. Then the mappings $h_1 = \{a \mapsto b,\ b \mapsto b\}$ and $h_2 = \{a \mapsto a,\ b \mapsto a\}$ are homomorphisms from $\mathcal{A}$ to $\mathcal{A}$. Since $h_1(a) = b$ and $h_2(b) = a$, the individuals a and b are isomorphic. Note that there is no *isomorphism* h from $\mathcal{A}$ to $\mathcal{A}$ (a bijective homomorphism such that its inverse is also a homomorphism) such that $h(a) = b$ or $h(b) = a$.

It is easy to show that entailed axioms are preserved under homomorphisms between ABoxes. In particular, isomorphic individuals are instances of the same concepts.

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

Proof. Suppose that $\mathcal{A} \cup \mathcal{T} \models \alpha$. Then $h(\mathcal{A} \cup \mathcal{T}) \models h(\alpha)$. Since $h(\mathcal{A} \cup \mathcal{T}) = h(\mathcal{A}) \cup h(\mathcal{T}) = h(\mathcal{A}) \cup \mathcal{T} \subseteq \mathcal{B} \cup \mathcal{T}$, by monotonicity we obtain $\mathcal{B} \cup \mathcal{T} \models h(\alpha)$. □

Corollary 1. *If individuals* a *and* b *in an ABox* $\mathcal{A}$ *are isomorphic, then for every TBox* $\mathcal{T}$ *and every concept* C*, we have* $\mathcal{A} \cup \mathcal{T} \models C(a)$ *if and only if* $\mathcal{A} \cup \mathcal{T} \models C(b)$.

If an ABox does not have role assertions, the isomorphic individuals in an ABox are exactly those that have the same concepts in the assertions. Hence, we can identify isomorphic individuals by just looking at their *types*— the set of concepts of which the individual is an (asserted) instance. Clearly, the number of different types, and hence the maximal number of individuals that are not isomorphic to each other is at most exponential in the number of different concepts used in the ABox. With role assertions, however, we cannot decide whether individuals are isomorphic by just looking at their assertions. In fact, the number of non-isomorphic individuals can be arbitrary large even if just one role is used in role assertions and there are no concept assertions.

Example 2. Consider an ABox $\mathcal{A} = \{R(a_{i-1}, a_i) \mid 1 < i \leq n\}$. It can be easily shown that the only homomorphism $h : \mathsf{ind}(\mathcal{A}) \to \mathsf{ind}(\mathcal{A})$ from $\mathcal{A}$ to $\mathcal{A}$ is the identity $h = \{a_i \mapsto a_i \mid 1 \leq i \leq n\}$, i.e., no different individuals in $\mathcal{A}$ are isomorphic to each other. In fact, it is easy to find a TBox $\mathcal{T}$ with which all individuals in $\mathcal{A}$ entail different sets of assertions. Indeed, take $\mathcal{T} = \{\top \sqsubseteq A_1,\ A_{i-1} \sqsubseteq \forall R.A_i, 1 < i \leq n\}$. Then we have $\mathcal{A} \cup \mathcal{T} \models A_j(a_i)$ if and only if $1 \leq j \leq i \leq n$.

From Example 2 one can see that with many role assertions, an ABox is less likely to have many isomorphic individuals. Note from Corollary 1 that if two individuals are isomorphic, then they entail the same assertions w.r.t. *every* TBox. Clearly, this property is too strong for our purpose, as we need to deal with just one given TBox. It can be that many (non-isomorphic) individuals are still materialized in the same way, because the number of different concept names used in the TBox is bounded. To take this into account, we propose a slightly different approach. Instead of partitioning the individuals in the ABox in equivalence classes according to the isomorphism relation

(which might be already too fine-grained for our TBox), we start with an approximation to this relation, which makes more individuals equivalent. As soon as entailed assertions are obtained using a reasoner that distinguish elements within the same equivalence class, we refine our approximation and repeat the process until the fixpoint.

Definition 2. *Let $\mathcal{A}$ be an ABox. The* type *of an individual a (w.r.t. $\mathcal{A}$) is a triple $tp(a) = (tp_{\downarrow}(a), tp_{\rightarrow}(a), tp_{\leftarrow}(a))$ where $tp_{\downarrow}(a) = \{C \mid C(a) \in \mathcal{A}\}$, $tp_{\rightarrow}(a) = \{R \mid \exists b : R(a,b) \in \mathcal{A}\}$, and $tp_{\leftarrow}(a) = \{S \mid \exists c : S(c,a) \in \mathcal{A}\}$.*

Intuitively, the type of an individual is obtained by considering all assertions in which this individual occurs in the ABox, and ignoring all other individuals in these assertions. Note that isomorphic individuals have the same types, so the relation between individuals of the same types is an approximation to the isomorphism relation.

4.2 Abstraction of an ABox

If we compress the ABox by simply merging all individuals with the same type into one, we might obtain unexpected entailments, even if all individuals are isomorphic.

Example 3. Consider the following ABox $\mathcal{A} = \{R(a,b), R(b,a)\}$. Clearly, a and b are isomorphic in $\mathcal{A}$. Let $\mathcal{B} = \{R(a,a)\}$ be obtained from $\mathcal{A}$ by replacing individual b with a, and let $\mathcal{T} = \{\top \sqsubseteq B \sqcup C,\ \exists R.B \sqsubseteq C\}$. It is easy to check that $\mathcal{B} \cup \mathcal{T} \models C(a)$, but $\mathcal{A} \cup \mathcal{T} \not\models C(a)$ (and hence $\mathcal{A} \cup \mathcal{T} \not\models C(b)$).

We could follow the approach in the SHER system and compute justifications for entailed assertions to determine which individuals should not be merged, but our goal is to avoid such computationally expensive operations. Instead of merging all individuals with the same type into one, we realize every individual type in our *abstract ABox*. With abstract ABoxes defined as follows, we can guarantee that assertions that are entailed for the representative individuals also hold for the original individuals.

Definition 3 (ABox Abstraction). *The* abstraction *of an ABox $\mathcal{A}$ is an ABox $\mathcal{B} = \bigcup_{a \in \mathsf{ind}(\mathcal{A})} \mathcal{B}_{tp(a)}$, where for each type $tp = (tp_{\downarrow}, tp_{\rightarrow}, tp_{\leftarrow})$, we define $\mathcal{B}_{tp} = \{C(x_{tp}) \mid C \in tp_{\downarrow}\} \cup \{R(x_{tp}, y^R_{tp}) \mid R \in tp_{\rightarrow}\} \cup \{S(z^S_{tp}, x_{tp}) \mid S \in tp_{\leftarrow}\}$, where x_{tp}, y^R_{tp}, and z^S_{tp} are fresh distinguished* abstract individuals*.*

Intuitively, the abstraction of an ABox is a disjoint union of small ABoxes witnessing each individual type realized in the ABox.

Example 4. Consider the ABox $\mathcal{A} = \{A(a), A(d), R(a,b), R(a,e), R(b,c), R(b,e), R(c,a), R(d,c), R(e,d)\}$. We have $\mathrm{tp}(b) = \mathrm{tp}(c) = \mathrm{tp}(e) = \mathrm{tp}_1 = (\emptyset, \{R\}, \{R\})$ and $\mathrm{tp}(a) = \mathrm{tp}(d) = \mathrm{tp}_2 = (\{A\}, \{R\}, \{R\})$. The abstraction of $\mathcal{A}$ is $\mathcal{B} = \mathcal{B}_{\mathrm{tp}_1} \cup \mathcal{B}_{\mathrm{tp}_2}$ with $\mathcal{B}_{\mathrm{tp}_1} = \{R(x_{\mathrm{tp}_1}, y^R_{\mathrm{tp}_1}), R(z^R_{\mathrm{tp}_1}, x_{\mathrm{tp}_1})\}$, $\mathcal{B}_{\mathrm{tp}_2} = \{A(x_{\mathrm{tp}_2}), R(x_{\mathrm{tp}_2}, y^R_{\mathrm{tp}_2}), R(z^R_{\mathrm{tp}_2}, x_{\mathrm{tp}_2})\}$.

The following lemma shows the soundness of concept assertions derived from the abstraction.

Lemma 2. *Let $\mathcal{A}$ be an ABox, $\mathcal{B}$ its abstraction, and $\mathcal{T}$ a TBox. Then for every type $tp = (tp_{\downarrow}, tp_{\rightarrow}, tp_{\leftarrow})$, every $a \in \mathsf{ind}(\mathcal{A})$ with $tp(a) = tp$ w.r.t. $\mathcal{A}$, and every concept C:*

(1) $\mathcal{B} \cup \mathcal{T} \models C(x_{tp})$ *implies* $\mathcal{A} \cup \mathcal{T} \models C(a)$*,*
(2) $\mathcal{B} \cup \mathcal{T} \models C(y^R_{tp})$ *and* $R(a, b) \in \mathcal{A}$ *implies* $\mathcal{A} \cup \mathcal{T} \models C(b)$*, and*
(3) $\mathcal{B} \cup \mathcal{T} \models C(z^S_{tp})$ *and* $S(c, a) \in \mathcal{A}$ *implies* $\mathcal{A} \cup \mathcal{T} \models C(c)$*.*

Proof. Consider all mappings $h : \mathsf{ind}(\mathcal{B}) \to \mathsf{ind}(\mathcal{A})$ such that:

$$h(x_{\text{tp}}) \in \{a \in \mathsf{ind}(\mathcal{A}) \mid \text{tp}(a) = \text{tp}\},$$
$$h(y^R_{\text{tp}}) \in \{b \mid R(h(x_{\text{tp}}), b) \in \mathcal{A}\}, \text{ and}$$
$$h(z^S_{\text{tp}}) \in \{c \mid S(c, h(x_{\text{tp}})) \in \mathcal{A}\}.$$

Clearly, $h(\mathcal{B}) \subseteq \mathcal{A}$ for every such mapping h. Furthermore, for every $a \in \mathsf{ind}(\mathcal{A})$, every $R(a, b) \in \mathcal{A}$ and every $S(c, a) \in \mathcal{A}$, there exists h with $h(x_{\text{tp}}) = a$, $h(y^R_{\text{tp}}) = b$, and $h(z^S_{\text{tp}}) = c$ for $\text{tp} = \text{tp}(a)$. Hence, claims (1)–(3) follow by Lemma 1. □

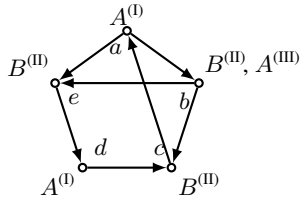
4.3 Abstraction Refinement

Note that the individuals from an ABox $\mathcal{A}$ may correspond to several abstract individuals in the ABox abstraction $\mathcal{B}$: Each individual a corresponds to the abstract individual x_{tp} for $\text{tp} = \text{tp}(a)$. In addition, if $R(b, a) \in \mathcal{A}$ or $S(a, b) \in \mathcal{A}$ for some individual b, then a also corresponds to y^R_{tp} and z^S_{tp} respectively for $\text{tp} = \text{tp}(b)$. The additional individuals y^R_{tp} and z^S_{tp} were introduced intentionally to refine the initial abstraction when new assertions of abstract individuals are derived, which in turn, can be used to derive new assertions of individuals in $\mathcal{A}$. Specifically, assume that we have materialized all entailed atomic assertions for the abstract ABox $\mathcal{B}$ w.r.t. the TBox using a reasoner. By Lemma 2, the corresponding assertions must also be entailed in the original ABox $\mathcal{A}$. In particular, by case (1), the new assertions computed for the individual x_{tp}, also hold for every individual a in $\mathcal{A}$ with $\text{tp}(a) = \text{tp}$. If we add all such assertions to the original ABox $\mathcal{A}$, these individuals would still have the same types, so even by building a new abstraction for the extended ABox, we would not derive new assertions for the abstraction. On the other hand, if we add the new assertion according to cases (2) and (3) of Lemma 2, we may obtain different assertions for individuals that previously had the same types. Indeed, if $R(a, b) \in \mathcal{A}$, and we have derived a new assertion $A(b)$ using case (2) of the lemma, then it is not necessary that a similar assertion $A(b')$ will be derived for every b' with $\text{tp}(b') = \text{tp}(b)$, because it is not necessarily the case that there exists $R(a', b') \in \mathcal{A}$ with $\text{tp}(a') = \text{tp}(a)$, for which this case also applies. Hence, adding the newly derived assertions using Lemma 2 may refine the types of the original individuals and, in turn, result in a new abstraction, for which new assertions can be derived once again.

The above suggests the following materialization procedure based on abstraction refinement. Given an ontology $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ we proceed as follows:

1. Build an initial abstraction $\mathcal{B}$ of $\mathcal{A}$ according to Definition 3.
2. Materialize $\mathcal{B} \cup \mathcal{T}$ using a reasoner.
3. Extend $\mathcal{A}$ with the newly derived assertions according to Lemma 2.
4. Update the types of the individuals in $\mathcal{A}$ and re-compute its abstraction $\mathcal{B}$.
5. Repeat from Step 2 until no new assertions can be added to $\mathcal{A}$.

ABox: $\mathcal{A} =$



TBox: $\mathcal{T} =$

$A \sqsubseteq \forall R.B$

$B \sqsubseteq \forall R^-.A$

Materialized abstract ABoxes	Abstractions I	II	III
$y^R_{\mathrm{tp}_1}$	$\{c, e, a, d\}$		
x_{tp_1}	$\{b, c, e\}$		
$z^R_{\mathrm{tp}_1}$	$\{a, b, d\}$		
$+B$ $y^R_{\mathrm{tp}_2}$	$\{b, e, c\}$	$\{b, e, c\}$	$\{b, e, c\}$
A x_{tp_2}	$\{a, d\}$	$\{a, d\}$	$\{a, d\}$
$z^R_{\mathrm{tp}_2}$	$\{c, e\}$	$\{c, e\}$	$\{c, e\}$
$y^R_{\mathrm{tp}_3}$		$\{c, e, a, d\}$	$\{a, d\}$
B x_{tp_3}		$\{b, e, c\}$	$\{c, e\}$
$+A$ $z^R_{\mathrm{tp}_3}$		$\{a, b, d\}$	$\{b, d, a\}$
$+B$ $y^R_{\mathrm{tp}_4}$			$\{c, e\}$
A, B x_{tp_4}			$\{b\}$
$+A$ $z^R_{\mathrm{tp}_4}$			$\{a\}$

Fig. 1. The abstractions I-III produced in Example 5. Each abstraction consists of the ABoxes corresponding to the four individual types. The inferred assertions are indicated with the "+" sign and are added to the corresponding original individuals shown in each column. The materialized assertions in the original ABox are labeled with the first iteration in which they appear.

Example 5 (Example 4 continued). Let $\mathcal{A}_{\mathrm{I}}$ be the ABox $\mathcal{A}$ from Example 4 and $\mathcal{T} = \{A \sqsubseteq \forall R.B,\ B \sqsubseteq \forall R^-.A\}$ a TBox. Let $\mathcal{B}_{\mathrm{I}}$ be the abstraction $\mathcal{B}$ of $\mathcal{A}_{\mathrm{I}} = \mathcal{A}$ computed in Example 4 (see Figure 1). By materializing $\mathcal{B}_{\mathrm{I}}$ w.r.t. $\mathcal{T}$ we get $B(y^R_{\mathrm{tp}_2})$, from which we obtain $\mathcal{A}_{\mathrm{II}} = \mathcal{A}_{\mathrm{I}} \cup \{B(b), B(e), B(c)\}$ using Lemma 2. Recomputing the types of individuals in $\mathcal{A}_{\mathrm{II}}$ yields $\mathrm{tp}(b) = \mathrm{tp}(c) = \mathrm{tp}(e) = \mathrm{tp}_3$= ($\{B\}$, $\{R\}$, $\{R\}$), while the types of a and d remain unchanged. The abstraction of $\mathcal{A}_{\mathrm{II}}$ is thus $\mathcal{B}_{\mathrm{II}} = \mathcal{B}_{\mathrm{tp}_2} \cup \mathcal{B}_{\mathrm{tp}_3}$, where $\mathcal{B}_{\mathrm{tp}_3} = \{B(x_{\mathrm{tp}_3}),\ R(x_{\mathrm{tp}_3}, y^R_{\mathrm{tp}_3}),\ R(z^R_{\mathrm{tp}_3}, x_{\mathrm{tp}_3})\}$. By materializing $\mathcal{B}_{\mathrm{II}}$, we get $A(z^R_{\mathrm{tp}_3})$, from which we obtain $\mathcal{A}_{\mathrm{III}} = \mathcal{A}_{\mathrm{II}} \cup \{A(b)\}$. We again recompute the types of individuals in $\mathcal{A}_{\mathrm{III}}$, which gives $\mathrm{tp}(b) = \mathrm{tp}_4 = (\{A, B\}, \{R\}, \{R\})$, while the types of the other individuals do not change. The abstraction of $\mathcal{A}_{\mathrm{III}}$ is thus $\mathcal{B}_{\mathrm{III}} = \mathcal{B}_{\mathrm{tp}_2} \cup \mathcal{B}_{\mathrm{tp}_3} \cup \mathcal{B}_{\mathrm{tp}_4}$, where $\mathcal{B}_{\mathrm{tp}_4} = \{A(x_{\mathrm{tp}_4}),\ B(x_{\mathrm{tp}_4}),\ R(x_{\mathrm{tp}_4}, y^R_{\mathrm{tp}_4}),\ R(z^R_{\mathrm{tp}_4}, x_{\mathrm{tp}_4})\}$. Materializing $\mathcal{B}_{\mathrm{III}}$ yields $B(y^R_{\mathrm{tp}_4})$ and $A(z^R_{\mathrm{tp}_4})$, which correspond to $B(c)$, $B(e)$, and $A(a)$. However, those assertions already exist in $\mathcal{A}_{\mathrm{III}}$, so the procedure terminates.

The abstraction refinement procedure terminates since after every iteration except the last one, new atomic assertions must be added to $\mathcal{A}$, and there is a bounded number of such assertions. Specifically, the number of iterations is at most $\|\mathsf{ind}(\mathcal{O})\| \times \|\mathsf{con}(\mathcal{O})\|$. The number of realized individual types in every ABox $\mathcal{A}$, and hence the size of every abstract ABox $\mathcal{B}$, is at most exponential in the number of different concepts and roles in $\mathcal{O}$. In practice, however, it is likely to be much smaller since not every possible type is realized in real ontologies. Note also that in practice, it is not necessary to add the newly derived assertions explicitly to the original ABox—one can recompute the new

types using some simple operations on the sets of individuals (intersection and unions), and keep the derived assertions only once for every new equivalence class. Note also that without nominals, we can exploit that $\mathcal{B}$ is a disjoint union of very simple ABoxes corresponding to the types of individuals, so they can be materialized independently of each other. This is particularly useful for updating the abstraction since only those ABoxes that correspond to newly created types should be materialized at every iteration.

5 Completeness

Lemma 2 guarantees that at every point of the iteration, our abstraction refinement procedure adds only entailed assertions to the ABox $\mathcal{A}$. In other words, our procedure is sound. The main question of this section is, whether our procedure is complete, i.e., whether we always compute all entailed atomic assertions in this way. Unfortunately, this is in general not the case, as demonstrated by the following example.

Example 6. Consider the ABox $\mathcal{A} = \{A(a),\ R(a,b),\ B(b)\}$ and the TBox $\mathcal{T} = \{B \sqsubseteq C \sqcup D, \exists R.C \sqsubseteq C, A \sqcap C \sqsubseteq \forall R.D\}$. Note that $\mathcal{A} \cup \mathcal{T} \models D(b)$. Let us compute the materialization using abstraction. We have $\mathrm{tp}(a) = (\{A\}, \{R\}, \emptyset)$ and $\mathrm{tp}(b) = (\{B\}, \emptyset, \{R\})$. Therefore $\mathcal{B} = \mathcal{B}_{\mathrm{tp}(a)} \cup \mathcal{B}_{\mathrm{tp}(b)}$, where $\mathcal{B}_{\mathrm{tp}(a)} = \{A(x_{\mathrm{tp}(a)}), R(x_{\mathrm{tp}(a)}, y^R_{\mathrm{tp}(a)})\}$ and $\mathcal{B}_{\mathrm{tp}(b)} = \{B(x_{\mathrm{tp}(b)}), R(z^R_{\mathrm{tp}(b)}, x_{\mathrm{tp}(b)})\}$. It is easy to see that $\mathcal{B} \cup \mathcal{T}$ does not entail any new atomic concept assertions. Hence, our procedure terminates after the first iteration without producing the entailment $\mathcal{A} \cup \mathcal{T} \models D(b)$.

The primary reason why our method does not work in this example is that our abstraction breaks the ABox into disconnected parts, which cannot communicate the non-deterministic choices, e.g., for the disjunction $C \sqcup D$. The only communication between ABoxes happens through the entailment of new assertions. If the ontology language does not allow such non-deterministic constructors, it is possible to obtain a complete procedure.

5.1 Horn $\mathcal{ALCHOI}$

While the results on the previous sections hold for $\mathcal{ALCHOI}$ in general (end even extensions thereof), we restrict ontologies in this section to a Horn fragment of $\mathcal{ALCHOI}$:

Definition 4 (Horn $\mathcal{ALCHOI}$). *An $\mathcal{ALCHOI}$ ontology $\mathcal{O}$ is* Horn *if, for every concept assertion $D(a)$ and every axiom $C \sqsubseteq D$, the concepts C and D satisfy, respectively, the following grammar definitions:*

$$C_{(i)} ::= \top \mid \bot \mid A \mid o \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \exists R.C, \tag{1}$$

$$D_{(i)} ::= \top \mid \bot \mid A \mid o \mid D_1 \sqcap D_2 \mid \exists R.D \mid \forall R.D \mid \neg C. \tag{2}$$

Intuitively, negations and universal restrictions should not occur negatively, and disjunctions should not occur positively. We can also allow TBox axioms that are equivalent to Horn axioms. For example, $A \sqcap \neg\forall R.\bot \sqsubseteq \neg\forall S.(B \sqcap C)$ is not Horn according to Definition 4, but is equivalent to the Horn axiom $A \sqcap \exists R.\top \sqsubseteq \exists S.\neg(B \sqcap C)$.

It is a well-known property of Horn languages that every consistent Horn ontology has a so-called *canonical model* that entails exactly the consequences entailed by the ontology. For the purpose of the paper, we require a weaker version of this property that speaks only about entailment of atomic concept assertions.

Theorem 1 (Weak Canonical Model Property for Horn $\mathcal{ALCHOI}$). *Every consistent Horn $\mathcal{ALCHOI}$ ontology $\mathcal{O}$ has a model $\mathcal{I}$ such that $\mathcal{I} \models A(a)$ implies $\mathcal{O} \models A(a)$ for every atomic concept assertion $A(a)$ with $a \in \mathsf{ind}(\mathcal{O})$ and $A \in \mathsf{con}(\mathcal{O})$.*

Theorem 1 can be proved using the property that Horn $\mathcal{ALCHOI}$ models are closed under cross-products. Then a canonical model is obtained from the cross-product of models refuting (finitely many) atomic non-types.

Before formulating our completeness result, we need to solve one small technical problem illustrated in the following example.

Example 7. Consider $\mathcal{A} = \{A(a),\ B(b),\ R(a,b)\}$ and $\mathcal{T} = \{A \sqcap \exists R.B \sqsubseteq C\}$, which consist of Horn axioms. Clearly, $\mathcal{A} \cup \mathcal{T} \models C(a)$. $\mathcal{A}$ realizes two different individual types: $\mathrm{tp}(a) = \mathrm{tp}_1 = (\{A\}, \{R\}, \emptyset)$ and $\mathrm{tp}(b) = \mathrm{tp}_2 = (\{B\}, \emptyset, \{R\})$, so our abstraction $\mathcal{B} = \mathcal{B}_{\mathrm{tp}_1} \cup \mathcal{B}_{\mathrm{tp}_2}$ consist of two ABoxes $\mathcal{B}_{\mathrm{tp}_1} = \{A(x_{\mathrm{tp}_1}),\ R(x_{\mathrm{tp}_1}, y^R_{\mathrm{tp}_1})\}$, and $\mathcal{B}_{\mathrm{tp}_2} = \{B(x_{\mathrm{tp}_2}),\ R(z^R_{\mathrm{tp}_2}, x_{\mathrm{tp}_2})\}$. In neither of these ABoxes we obtain a new assertion after materialization, so our procedure terminates without deriving $C(a)$.

In order to see how to fix this problem, note that $\mathcal{B}_{\mathrm{tp}_2} \cup \mathcal{T} \models (\exists R.B)(z^R_{\mathrm{tp}_2})$, so there is an entailed assertion, just not an atomic one. To capture this inference, we introduce a new concept that "defines" $\exists R.B$. Specifically, let $\mathcal{T}' = \{\exists R.B \sqsubseteq X,\ A \sqcap X \sqsubseteq C\}$ where X is a fresh concept name. Clearly, $\mathcal{T}'$ is a conservative extension of $\mathcal{T}$ (one can extend every model of $\mathcal{T}$ to a model of $\mathcal{T}'$ by interpreting X as $\exists R.B$), so the assertions for A, B, and C entailed by $\mathcal{T}'$ are the same as for $\mathcal{T}$. However, with $\mathcal{T}'$ we can derive a new assertion $\mathcal{B}_{\mathrm{tp}_2} \cup \mathcal{T}' \models X(z^R_{\mathrm{tp}_2})$. If we now add the corresponding assertion $X(a)$ to $\mathcal{A}$ and recompute the abstraction for the updated type $\mathrm{tp}(a) = \mathrm{tp}_3 = (\{A, X\}, \{R\}, \emptyset)$ ($\mathrm{tp}(b)$ does not change), we have $\mathcal{B}_{\mathrm{tp}_3} = \{A(x_{\mathrm{tp}_3}),\ X(x_{\mathrm{tp}_3}),\ R(x_{\mathrm{tp}_3}, y^R_{\mathrm{tp}_3})\}$, and obtain $\mathcal{B}_{\mathrm{tp}_3} \cup \mathcal{T}' \models C(x_{\mathrm{tp}_3})$, which gives us the intended result.

Example 7 suggests that to achieve completeness, we need to represent existential restrictions on the left hand side of the axioms using new atomic concepts. Note that $\exists R.B \sqsubseteq X$ is equivalent to $B \sqsubseteq \forall R^-.X$. Thus we can just require that there are no existential restrictions on the left hand side of concept inclusions, and all universal restrictions on the right have only atomic concepts as fillers.

Definition 5 (Normal Form for Horn $\mathcal{ALCHOI}$). *Horn $\mathcal{ALCHOI}$ axioms $D(a)$ and $C \sqsubseteq D$ are in* normal form *if they satisfy the following grammar definitions:*

$$C_{(i)} ::= \top \mid \bot \mid A \mid o \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \tag{3}$$

$$D_{(i)} ::= \top \mid \bot \mid A \mid o \mid D_1 \sqcap D_2 \mid \exists R.D \mid \forall R.A \mid \neg C \tag{4}$$

Intuitively, in addition to the constraints for Horn $\mathcal{ALCHOI}$ ontologies given by (1) and (2) of Definition 4, negative occurrences of existential restrictions are not allowed, and (positive) occurrences of universal restrictions can only have concept names as fillers. It is easy to convert axioms to such a normal form using the well-known structural transformation. Specifically, we can repeatedly replace every existential restriction

$\exists R.D$ in (2) with a fresh concept name X and add a new axiom $D \sqsubseteq \forall R^-.X$. Likewise, we can replace every universal restriction $\forall R.C$ in (1) with $\forall R.Y$ for a fresh concept name Y and add an axiom $Y \sqsubseteq C$. As with Horn axioms, we do not actually need the axioms in the TBox to be syntactically in the normal form. It is sufficient that they are equivalent to axioms in the normal form – the reasoner will still produce the same result. For example, an axiom $\exists R.(A_1 \sqcap A_2) \sqsubseteq B_1 \sqcap B_2$ can be left untouched because it is equivalent to an axiom $A_1 \sqcap A_2 \sqsubseteq \forall R^-.B_1 \sqcap \forall R^-.B_2$ in normal form. Note that the axiom $A \sqcap \exists R.B \sqsubseteq C$ in $\mathcal{T}$ from Example 7 is not equivalent to the pair of axioms $\exists R.B \sqsubseteq X$, $A \sqcap X \sqsubseteq C$ in $\mathcal{T}'$ because the latter axioms contain a new symbol X. In fact, $A \sqcap \exists R.B \sqsubseteq C$ is not equivalent to any axiom(s) in normal form.

5.2 Completeness Proof

We are now ready to show the following completeness result:

Theorem 2. *Let $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ be a normalized Horn $\mathcal{ALCHOI}$ ontology and $\mathcal{B}$ the abstraction of $\mathcal{A}$. $\mathcal{O}$ is concept materialized if, for every type $tp = (tp_\downarrow, tp_\rightarrow, tp_\leftarrow)$, every individual $a \in \mathsf{ind}(\mathcal{A})$ with $tp(a) = tp$, and every atomic concept A, we have:*

(1) $\mathcal{B} \cup \mathcal{T} \models A(x_{tp})$ implies $A(a) \in \mathcal{A}$,
(2) $\mathcal{B} \cup \mathcal{T} \models A(y^R_{tp})$ and $R(a,b) \in \mathcal{A}$ implies $A(b) \in \mathcal{A}$, and
(3) $\mathcal{B} \cup \mathcal{T} \models A(z^S_{tp})$ and $S(c,a) \in \mathcal{A}$ implies $A(c) \in \mathcal{A}$.

Proof. To prove Theorem 2, we extend the abstraction $\mathcal{B}$ of $\mathcal{A}$ with new role assertions $R(x_{\mathsf{tp}(a)}, x_{\mathsf{tp}(b)})$ for every $R(a,b) \in \mathcal{A}$. Let us denote this extended abstract ABox by $\mathcal{B}'$. Since, for every $C(a) \in \mathcal{A}$, we also have $C \in \mathsf{tp}_\downarrow(a)$ and, thus, $C(x_{\mathsf{tp}(a)}) \in \mathcal{B} \subseteq \mathcal{B}'$, the mapping $h : \mathsf{ind}(\mathcal{A}) \to \mathsf{ind}(\mathcal{B}')$ defined by $h(a) = x_{\mathsf{tp}(a)}$ is a homomorphism from $\mathcal{A}$ to $\mathcal{B}'$. Therefore, by Lemma 1, if $\mathcal{A} \cup \mathcal{T} \models A(a)$, then $\mathcal{B}' \cup \mathcal{T} \models A(x_{\mathsf{tp}(a)})$. The key part of the proof is to demonstrate that in this case we also have $\mathcal{B} \cup \mathcal{T} \models A(x_{\mathsf{tp}(a)})$. That is, the extended abstract ABox $\mathcal{B}'$ does not entail new atomic concept assertions compared to $\mathcal{B}$. It follows then that $A(a) \in \mathcal{A}$ by condition (1) of the theorem. This implies that $\mathcal{O}$ is concept materialized.

To prove that $\mathcal{B}'$ entails the same atomic concept assertions as $\mathcal{B}$, we use the remaining conditions (2) and (3) of Theorem 2 and the canonical model property formulated in Theorem 1. Note that since new role assertions of the form $R(x_{\mathsf{tp}(a)}, x_{\mathsf{tp}(b)})$ are added to $\mathcal{B}'$ only if $R(a,b) \in \mathcal{A}$, we have $R \in \mathsf{tp}_\rightarrow(a)$ and $R \in \mathsf{tp}_\leftarrow(b)$ by Definition 2. Therefore, we already had role assertions $R(x_{\mathsf{tp}(a)}, y^R_{\mathsf{tp}(a)}) \in \mathcal{B}$ and likewise $R(z^R_{\mathsf{tp}(b)}, x_{\mathsf{tp}(b)}) \in \mathcal{B}$ for the same role R. Furthermore, by condition (2) of Theorem 2, if $\mathcal{B} \cup \mathcal{T} \models A(y^R_{\mathsf{tp}(a)})$, then since $R(a,b) \in \mathcal{A}$, we also have $A(b) \in \mathcal{A}$, and thus $A(x_{\mathsf{tp}(b)}) \in \mathcal{B}$. Likewise, by condition (3), if $\mathcal{B} \cup \mathcal{T} \models A(z^R_{\mathsf{tp}(b)})$, then $A(x_{\mathsf{tp}(a)}) \in \mathcal{B}$. The following lemma shows that with these properties for $\mathcal{B}$, after adding the new role assertion $R(x_{\mathsf{tp}(a)}, x_{\mathsf{tp}(b)})$ to $\mathcal{B}$, no new atomic concept assertions can be entailed.

Lemma 3 (Four-Individual Lemma). *Let $\mathcal{O}$ be a normalized Horn $\mathcal{ALCHOI}$ ontology such that $\{R(x_1, y_1), R(z_2, x_2)\} \subseteq \mathcal{O}$ for some x_1, y_1, z_2, x_2, and R. Further, assume that for every concept name A we have:*

(1) $\mathcal{O} \models A(y_1)$ *implies* $\mathcal{O} \models A(x_2)$*, and*
(2) $\mathcal{O} \models A(z_2)$ *implies* $\mathcal{O} \models A(x_1)$*.*

Finally, let $\mathcal{O}' = \mathcal{O} \cup \{R(x_1, x_2)\}$*. Then for every concept name* A *and every individual* a*, we have* $\mathcal{O}' \models A(a)$ *implies* $\mathcal{O} \models A(a)$*.*

Proof (Sketch). Suppose that $\mathcal{O}' \models A(a)$. We will show that $\mathcal{O} \models A(a)$. If $\mathcal{O}$ is inconsistent then this holds trivially. Otherwise, there exists a model $\mathcal{I}$ of $\mathcal{O}$ satisfying Theorem 1. From $\mathcal{I}$ we construct an interpretation $\mathcal{I}'$ that coincides with $\mathcal{I}$ apart from the interpretation of role names. With the given individuals x_1 and x_2, for every role name S we define

$$S^{\mathcal{I}'} = S^{\mathcal{I}} \cup \begin{cases} \{(x_1^{\mathcal{I}}, x_2^{\mathcal{I}})\} & \text{if } \mathcal{O} \models R \sqsubseteq S \text{ and } \mathcal{O} \not\models R \sqsubseteq S^- \\ \{(x_2^{\mathcal{I}}, x_1^{\mathcal{I}})\} & \text{if } \mathcal{O} \models R \sqsubseteq S^- \text{ and } \mathcal{O} \not\models R \sqsubseteq S \\ \{(x_1^{\mathcal{I}}, x_2^{\mathcal{I}}), (x_2^{\mathcal{I}}, x_1^{\mathcal{I}})\} & \text{if } \mathcal{O} \models R \sqsubseteq S \text{ and } \mathcal{O} \models R \sqsubseteq S^- \\ \emptyset & \text{otherwise} \end{cases}$$

We will prove that $\mathcal{I}' \models \mathcal{O}'$, which implies $\mathcal{I}' \models A(a)$ since $\mathcal{O}' \models A(a)$, and thus $\mathcal{I} \models A(a)$ by definition of $\mathcal{I}'$, from which $\mathcal{O} \models A(a)$ follows since $\mathcal{I}$ satisfies Theorem 1.

First, we prove by induction that for each C and D defined respectively by (3) and (4), we have $C^{\mathcal{I}} = C^{\mathcal{I}'}$ and $D^{\mathcal{I}} \subseteq D^{\mathcal{I}'}$. The only non-trivial case is $D = \forall S.A$ with $\mathcal{O} \models R \sqsubseteq S$ and $S \in \mathsf{rol}(\mathcal{O})$ (the case where S is an inverse role can be proved analogously). Take any $d \in D^{\mathcal{I}}$. To show that $d \in D^{\mathcal{I}'}$, we need to prove that $d' \in A^{\mathcal{I}'}$ for every d' with $\langle d, d' \rangle \in S^{\mathcal{I}'}$. If $\langle d, d' \rangle \in S^{\mathcal{I}}$, this is obvious. Otherwise, $\langle d, d' \rangle = \langle x_1^{\mathcal{I}}, x_2^{\mathcal{I}} \rangle$. By assumption, $\mathcal{I} \models R(x_1, y_1)$ and $\mathcal{O} \models R \sqsubseteq S$, hence, $\mathcal{I} \models S(x_1, y_1)$, which, together with $d = x_1^{\mathcal{I}} \in D^{\mathcal{I}}$, implies $y_1^{\mathcal{I}} \in A^{\mathcal{I}}$. Thus $\mathcal{I} \models A(y_1)$. Since $\mathcal{I}$ satisfies Theorem 1, we have $\mathcal{O} \models A(y_1)$. By Condition (1), $\mathcal{O} \models A(x_2)$. Thus $d' = x_2^{\mathcal{I}} \in A^{\mathcal{I}} = A^{\mathcal{I}'}$, and hence, $d = x_1^{\mathcal{I}} \in D^{\mathcal{I}'}$, what was required to show.

It remains to show that $\mathcal{I}' \models \mathcal{O}'$ with $\mathcal{O}' = \mathcal{O} \cup \{R(x_1, x_2)\}$. Since $\mathcal{I} \models \mathcal{O}$, for every $C \sqsubseteq D \in \mathcal{O}$ we have $C^{\mathcal{I}'} = C^{\mathcal{I}} \subseteq D^{\mathcal{I}} \subseteq D^{\mathcal{I}'}$, for every $D(a) \in \mathcal{O}$ we have $a^{\mathcal{I}'} = a^{\mathcal{I}} \in D^{\mathcal{I}} \subseteq D^{\mathcal{I}'}$, and for every $R(a, b) \in \mathcal{O}$ we have $\langle a^{\mathcal{I}'}, b^{\mathcal{I}'} \rangle = \langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}} \subseteq R^{\mathcal{I}'}$. Finally, the definition of $\mathcal{I}'$ ensures that for every role inclusion $S \sqsubseteq P \in \mathcal{O}$ we have $S^{\mathcal{I}'} \subseteq P^{\mathcal{I}'}$, and that $\mathcal{I}' \models R(x_1, x_2)$. Thus $\mathcal{I}' \models \mathcal{O}'$. □

By repeatedly applying Lemma 3 for each $x_1 = x_{\mathsf{tp}(a)}$, $y_1 = y^R_{\mathsf{tp}(a)}$, $x_2 = x_{\mathsf{tp}(b)}$, $z_2 = z^R_{\mathsf{tp}(b)}$ and R such that $R(a, b) \in \mathcal{A}$, we obtain that $\mathcal{B}'$ entails only those atomic assertions that are entailed by $\mathcal{B}$, which completes the proof of Theorem 2. □

6 Implementation and Evaluation

To evaluate the feasibility of our approach, we implemented the procedure sketched in Section 4.3 in Java. The system relies on Neo4j 1.9.4[4] to store the ABoxes and uses an external OWL reasoner for materializing the abstractions.

[4] `http://www.neo4j.org`

Table 2. Test suite statistics with the number of atomic concepts in the ontology (#A and #A_N for the normalized ontology), roles (#R), individuals (#I), role (#$R(a,b)$) and concept (#$A(a)$) assertions, and the number of concept assertions inferred by the system

Ontology	#A	#A_N	#R	#I	#$R(a,b)$	#$A(a)$	#$A(a)$ inferred
Gazetteer	710	711	15	516 150	604 164	11 112	538 799
Coburn	719	1 161	109	123 515	237 620	297 002	535 124
LUBM 1	43	49	25	17 174	49 336	18 128	48 326
LUBM 10	43	49	25	207 426	630 753	219 680	593 485
LUBM 50	43	49	25	1 082 818	3 298 813	1 147 136	3 097 722
LUBM 100	43	49	25	2 153 645	6 645 928	2 281 035	6 164 538
LUBM 500	43	49	25	10 189 173	31 073 066	10 793 509	29 169 321
UOBM 1	69	90	35	24 925	153 571	44 680	142 747
UOBM 10	69	90	35	242 549	1 500 628	434 115	1 381 722
UOBM 50	69	90	35	1 227 445	7 594 996	2 197 035	6 991 583
UOBM 100	69	90	35	2 462 012	15 241 909	4 409 891	14 027 772
UOBM 500	69	90	35	12 379 113	76 612 736	22 168 148	72 215 007

The goal of our evaluation is to estimate whether our assumption that in relatively simple ontologies with large ABoxes the number of realized types and, consequently, the size of the abstract ABoxes is small. Furthermore, we analyze whether it is indeed the case that real-world ontologies have relatively simple axioms that do not require many refinement steps, where a refinement step is the process of refining the individual types.

We selected ontologies with large ABoxes that are also used to evaluate other approaches.[5] Table 2 provides relevant metrics for the test ontologies. Gazetteer is from the NCBO BioPortal, Coburn is a large bio ontology from the Phenoscape project, and LUBM (UOBM) refers to the Lehigh University Benchmark[6] (the University Ontology Benchmark).[7] LUBM n (UOBM n) denotes the data set for n universities. Gazetteer is genuinely within Horn $\mathcal{ALEO}$ and the remaining ontologies have been converted to Horn $\mathcal{ALCHOI}$. Note that the increase of normalized concepts (A_N) in comparison to the original concepts (A) in Table 2 is a rough indicator of TBox complexity, which adds extra workload to reasoners.

Tables 3 and 4 show the results of computing and iteratively refining the abstract ABoxes until the fixpoint. The second column in Table 3 shows the number of refinement steps. The third and fourth (fifth and sixth) columns show the number of individuals (assertions) in the first and last abstraction, respectively, while the last four columns show the according relative reduction in percent compared to the original ABoxes. Table 4 shows the type statistics, i.e. the number of types and the average number of individuals, concept names, and property names per type.

In general, the abstract ABoxes are significantly smaller than the original ones and the ontologies can be materialized with few refinement steps. For ontologies with sim-

[5] Download and references at `http://www.derivo.de/dl14-ontologies/`

[6] `http://swat.cse.lehigh.edu/projects/lubm`

[7] `http://www.cs.ox.ac.uk/isg/tools/UOBMGenerator`

Table 3. Number of refinement steps, size of the abstract ABoxes, and size of the abstract ABoxes in comparison with the original ABoxes for the first and the last abstraction

Ontology	# of steps	Abstract ABox size				Abstract ABox size (%)			
		# indiv.		# assertions		% indiv.		% assertions	
Gazetteer	1	5 640	5 640	5 142	8 512	1.09	1.09	0.79	1.31
Coburn	2	3 992	4 059	5 569	8 633	3.23	3.29	1.04	2.16
LUBM 1	1	139	139	143	254	0.81	0.81	0.21	0.38
LUBM 10	1	154	154	158	281	0.07	0.07	0.02	0.03
LUBM 50	1	148	148	152	271	0.01	0.01	0.003	0.006
LUBM 100	1	148	148	152	271	0.007	0.007	0.002	0.003
LUBM 500	1	148	148	152	271	0.001	0.001	0.001	0.001
UOBM 1	2	25 378	39 322	31 659	101 324	101.82	157.76	15.97	51.11
UOBM 10	2	98 266	169 579	125 056	448 400	40.51	69.92	6.46	23.18
UOBM 50	2	226 176	395 956	290 652	1 057 854	18.43	32.26	2.97	10.80
UOBM 100	2	311 574	547 361	402 188	1 472 058	12.66	22.23	2.05	7.49
UOBM 500	2	596 135	1 033 685	772 920	2 806 786	4.82	8.35	0.78	2.84

ple TBoxes, which contain mostly atomic concept inclusions, domains and ranges for roles, and conjunctions, only one refinement step is required. This is the case since any concept assertion derived for a successor or predecessor of an abstract individual is also derived for the individual itself. LUBM and UOBM additionally contain universal quantifications, e.g. $\mathsf{Department} \sqsubseteq \forall \mathsf{headOf}^{-}.\mathsf{Chair}$ (rewritten from $\exists \mathsf{headOf}.\mathsf{Department} \sqsubseteq \mathsf{Chair}$), but these axioms do not create long propagations of concept Assertions Over roles. For LUBM, many individuals have similar types and can be grouped into equivalence classes. This results in an extremely good compression with abstractions of nearly constant size for arbitrarily many LUBM universities. For instance, the final abstractions are just 0.38 % (for LUBM 1) and 0.001 % (for LUBM 500) of the size of the original ABox. This and the fact that no refinement is needed (i.e. concepts are not propagated over chains of successors or predecessors) also explains that other related approaches like SHER and Wandelt's and Möller's approach show a very good performance for LUBM. UOBM also contains nominals and the individuals are more connected than in LUBM. Thus, UOBM requires one more refinement step compared to LUBM.

Our qualitative performance evaluation confirms the correlation between the size of abstract ABoxes and the total time for the materialization. We compared the respective materialization times of the original ABox with the sum of materialization times for all abstract ABoxes using the reasoners HermiT and Konclude.[8] For ontologies with small abstract ABoxes such as LUBM, Gazetteer and Coburn, the sum of the reasoning times for all abstract ABoxes is less than a tenth of the reasoning time for the original ABox. While the runtimes for the abstractions of UOBM 1 are still 2 to 4 times that of the original ABox, the runtimes for UOBM 50 are already down by 50%. The original UOBM 100 ontology could neither be processed by HermiT nor by Konclude within a 32 GB RAM limit run on Intel Xeon E5-2440 6 cores, but its abstraction can easily be materialized, e.g., within 84 seconds and 8 GB RAM by Konclude. Currently, we re-compute

[8] See `http://www.hermit-reasoner.com` and `http://www.konclude.com`

Table 4. Statistics about individual types for the first and the last abstraction: number of individual types, average number of individuals, concept names, and property names per individual type

Ontology	# Individual types		Individual type statistics					
			# indiv. / type		$\#A_N$ / type		$\#R$ / type	
Gazetteer	1 845	1 845	280	280	0.73	2.56	2.05	2.05
Coburn	1 056	1 072	117	115	2.49	5.27	2.79	2.79
LUBM 1	30	30	572	572	1.13	4.83	3.63	3.63
LUBM 10	29	29	7 153	7 153	1.14	5.38	4.31	4.31
LUBM 50	27	27	40 104	40 104	1.15	5.56	4.48	4.48
LUBM 100	27	27	79 765	79 765	1.15	5.56	4.48	4.48
LUBM 500	27	27	377 377	377 377	1.15	5.56	4.48	4.48
UOBM 1	3 104	4 705	8	5	3.02	14.18	7.18	7.35
UOBM 10	11 453	17 347	21	14	3.34	15.48	7.58	7.86
UOBM 50	25 636	43 283	48	28	3.52	16.29	7.82	8.14
UOBM 100	34 992	59 184	70	42	3.59	16.62	7.91	8.24
UOBM 500	65 148	108 691	190	114	3.71	17.31	8.15	8.51

the abstraction after each refinement step. There is certainly room for optimizations, e.g. by updating the types and computing the abstractions incrementally.

7 Conclusions and Future Work

We have presented an approach for ontology materialization based on abstraction refinement. The main idea is to represent ABox individuals using several (overlapping) equivalent classes and to use information derived for their abstract representatives to refine the abstraction. Although the approach does not necessarily guarantee that the abstraction is always smaller than the original ABox, the method particularly pays off for ontologies with large ABoxes and relatively small and simple TBoxes.

Currently, our approach is complete for Horn $\mathcal{ALCHOI}$ ontologies due to the property that only (deterministically) derived assertions are used for abstraction refinement. We could potentially extend our approach to non-Horn ontology languages by exploiting additional information about non-deterministically derived instances as provided, for example, by the HermiT reasoner [7]. Some Horn features, on the other hand, could be supported easily, e.g., it is easy to support transitive roles and role chains by using the well-known encoding of these axioms via concept inclusions [13].

In this paper we mainly focus on concept materialization since role materialization for Horn $\mathcal{ALCHOI}$ can essentially be computed by expanding role hierarchies (special care needs to be taken of nominals though). When ontologies contain role chains and functional roles, however, materialization of role assertions becomes less trivial, e.g. the encoding of role chains is not enough and a naive encoding of functionality is inefficient. We currently investigate how these features can efficiently be supported.

Since the abstraction consists of disjoint parts, these parts can be processed independently of each other (if nominals are taken care of). This can be used in the refinement steps to process only the parts that have really changed or for an efficient support of updates to the ABox. In addition, the abstract ABoxes could serve not only as a generic

interface for communication with the reasoner, but also as a compact representation of the materialization. This can be particularly useful when answering instance and conjunctive queries over the materialized ABoxes, where the abstraction can be used to prune the search space.

References

1. B. Bishop, A. Kiryakov, D. Ognyanoff, I. Peikov, Z. Tashev, and R. Velkov. OWLIM: A family of scalable semantic repositories. *Semantic Web J.*, 2(1):33–42, 2011.
2. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. of Automated Reasoning*, 39(3):385–429, 2007.
3. J. Dolby, A. Fokoue, A. Kalyanpur, E. Schonberg, and K. Srinivas. Scalable highly expressive reasoner (SHER). *J. of Web Semantics*, 7(4):357–361, 2009.
4. A. Fokoue, A. Kershenbaum, L. Ma, E. Schonberg, and K. Srinivas. The summary ABox: Cutting ontologies down to size. In *Proc. of the 5th Int. Semantic Web Conf. (ISWC 2006)*, volume 4273 of *LNCS*, pages 343–356. Springer, 2006.
5. B. Glimm, Y. Kazakov, T. Liebig, T.-K. Tran, and V. Vialard. Abstraction Refinement for Ontology Materialization. Technical report, University of Ulm and derivo GmbH, 2014. `https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.090/Publikationen/2014/ISWC2014S38TR.pdf`.
6. A. Kalyanpur, B. Parsia, M. Horridge, and E. Sirin. Finding all justifications of OWL DL entailments. In *Proc. of the 6th Int. Semantic Web Conf. (ISWC 2007)*, volume 4825 of *LNCS*, pages 267–280. Springer, 2007.
7. I. Kollia and B. Glimm. Optimizing SPARQL query answering over OWL ontologies. *J. of Artificial Intelligence Research (JAIR)*, 48:253–303, 2013.
8. V. Kolovski, Z. Wu, and G. Eadon. Optimizing enterprise-scale OWL 2 RL reasoning in a relational database system. In *Proc. of the 9th Int. Semantic Web Conf. (ISWC 2010)*, volume 6496 of *Lecture Notes in Computer Science*, pages 436–452. Springer, 2010.
9. R. Kontchakov, C. Lutz, D. Toman, F. Wolter, and M. Zakharyaschev. The combined approach to query answering in DL-Lite. In *Proc. of the 12th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2010)*. AAAI Press, 2010.
10. C. Lutz, D. Toman, and F. Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 2070–2075, 2009.
11. B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz, editors. *OWL 2 Web Ontology Language: Profiles (Second Edition)*. W3C Recommendation, 11 December 2012. Available at `http://www.w3.org/TR/owl2-profiles/`.
12. H. Pérez-Urbina, B. Motik, and I. Horrocks. Tractable query answering and rewriting under description logic constraints. *J. of Applied Logic*, 8(2):186–209, 2010.
13. F. Simančík. Elimination of complex RIAs without automata. In *Proc. of the 25th Int. Workshop on Description Logics*, 2012.
14. J. Urbani, S. Kotoulas, J. Maassen, F. van Harmelen, and H. Bal. WebPIE: A web-scale parallel inference engine using MapReduce. *J. of Web Semantics*, 10:59–75, 2012.
15. S. Wandelt. *Efficient instance retrieval over semi-expressive ontologies*. PhD thesis, Hamburg University of Technology, 2011.
16. S. Wandelt and R. Möller. Towards ABox modularization of semi-expressive description logics. *J. of Applied Ontology*, 7(2):133–167, 2012.