Pushing the Limits of Reasoning over Ontologies with Hidden Content*

Bernardo Cuenca Grau and **Boris Motik**
Oxford University Computing Laboratory
University of Oxford, UK

Abstract

There is currently a growing interest in techniques for hiding parts of the signature of an ontology $\mathcal{K}_h$ that is being reused by another ontology $\mathcal{K}_v$. Towards this goal, Cuenca Grau, Motik, and Kazakov (2009) recently proposed the *import-by-query framework*, which makes the content of $\mathcal{K}_h$ accessible through a limited query interface. If $\mathcal{K}_v$ reuses the symbols from $\mathcal{K}_h$ in a certain restricted way, one can reason over $\mathcal{K}_v \cup \mathcal{K}_h$ by accessing only $\mathcal{K}_v$ and the query interface. In this paper, we map out the landscape of the import-by-query problem. We show that certain restrictions of our original framework are strictly necessary to make reasoning possible, we propose extensions that overcome some of the expressivity limitations, we present several novel reasoning algorithms, and we outline the limitations of the new framework.

Introduction

The Web Ontology Language (OWL) and its revision OWL 2 are ontology languages standardized by the W3C, and their formal underpinning is provided by description logics (DLs) (Baader et al. 2007)—knowledge representation formalisms with well understood formal properties. OWL ontologies are often used to provide a shared vocabulary for a family of applications, thus making data exchange between the applications easier. Furthermore, constructing ontologies is a labor-intensive task, so reusing (parts of) well-established ontologies when developing new ones is seen as key to reducing ontology development cost. Consequently, the problem of *ontology reuse* has recently received significant attention (Stuckenschmidt, Parent, and Spaccapietra 2009).

An OWL ontology $\mathcal{K}_v$ can reuse an ontology $\mathcal{K}_h$ via *importing*, and the result is logically equivalent to $\mathcal{K}_v \cup \mathcal{K}_h$. OWL reasoners deal with imports by loading both ontologies and merging their contents, which requires physical access to the axioms of $\mathcal{K}_h$. The vendor of $\mathcal{K}_h$, however, might be reluctant to distribute (parts of) the contents of $\mathcal{K}_h$, as doing so might allow competitors to plagiarize $\mathcal{K}_h$. Moreover, $\mathcal{K}_h$ might contain information that is sensitive from a privacy point of view. Finally, one might want to impose a varying cost on the reuse of different parts of $\mathcal{K}_h$. To stipulate that $\mathcal{K}_h$ should not be publicly available, we call the ontology $\mathcal{K}_h$ *hidden* and, by analogy, we call $\mathcal{K}_v$ *visible*.

Motivated by such scenarios, several approaches to *hiding* a subset of the signature of $\mathcal{K}_h$ have been developed. One approach is to publish an Υ-*interpolant* of $\mathcal{K}_h$—an ontology that contains no symbols from Υ and that coincides with $\mathcal{K}_h$ on all logical consequences formed using the symbols not in Υ (Konev, Walter, and Wolter 2009). Once an Υ-interpolant has been published, it can be imported into $\mathcal{K}_v$ without any restrictions, and one can reason over the union of $\mathcal{K}_v$ and the Υ-interpolant using off-the-shelf DL reasoners. Such Υ-interpolants, however, exist only for inexpressive DLs and under certain syntactic restrictions; furthermore, they can be of exponential size, which can be problematic in practice.

In our previous work, we proposed an approach in which $\mathcal{K}_h$ is accessible via a limited query interface that we call an *oracle* (Cuenca Grau, Motik, and Kazakov 2009). The oracle advertises a *public* subset Γ of the signature of $\mathcal{K}_h$, and it can answer queries over $\mathcal{K}_h$ that are expressed in a particular query language and that use only the symbols from Γ. Under certain assumptions, a so-called *import-by-query* algorithm can reason over $\mathcal{K}_v \cup \mathcal{K}_h$ (e.g., determine its satisfiability) without having physical access to the content of $\mathcal{K}_h$, by only posing queries to the oracle for $\mathcal{K}_h$. The idea of accessing an ontology through oracles is similar in spirit to the proposal for query answering in a peer-to-peer setting by Calvanese et al. (2004); however, the latter approach focuses on reusing data rather than schema statements.

Our framework minimizes the information flow between $\mathcal{K}_v$ and $\mathcal{K}_h$. Apart from restricting access to $\mathcal{K}_h$, reasoning can be performed without $\mathcal{K}_h$ having any access to $\mathcal{K}_v$, which is beneficial as $\mathcal{K}_v$ might also contain sensitive information. Furthermore, unlike interpolation, our framework does not require materializing an exponentially large Υ-interpolant, and it can be applicable in cases when Υ-interpolants for $\mathcal{K}_h$ do not exist. In contrast to interpolation, however, our framework imposes restrictions on the way $\mathcal{K}_v$ can reuse the public symbols from $\mathcal{K}_h$.

The formal properties of import-by-query algorithms depend on the oracle query language and the logics used to express $\mathcal{K}_v$ and $\mathcal{K}_h$. In our previous work, we studied the case when oracles support concept satisfiability queries—that is, when queries are concepts over Γ in the DL of $\mathcal{K}_h$, for which the oracle determines their satisfiability w.r.t. $\mathcal{K}_h$.

*Bernardo Cuenca Grau is supported by a Royal Society University Research Fellowship.

We proved that no import-by-query algorithm exists in such a setting even if $\mathcal{K}_v$ and $\mathcal{K}_h$ are expressed in the lightweight description logic $\mathcal{EL}$ (Baader, Brandt, and Lutz 2005). To make reasoning possible, the reuse of the advertised symbols was subjected to the following restrictions.

1. Reuse was required to be modular—that is, $\mathcal{K}_v$ could not change the meaning of the symbols reused from $\mathcal{K}_h$ (Lutz, Walther, and Wolter 2007; Cuenca Grau et al. 2008).
2. Role symbols (i.e., binary predicates) from $\mathcal{K}_h$ could be reused in $\mathcal{K}_v$ only in a particular restricted way.

We presented an import-by-query algorithm that can handle $\mathcal{K}_v$ and $\mathcal{K}_h$ expressed in the DLs $\mathcal{SROIQ}$ (Kutz, Horrocks, and Sattler 2006) and $\mathcal{SRIQ}$ (Horrocks, Kutz, and Sattler 2005), respectively, provided that the mentioned assumptions are satisfied. Our algorithm, however, may issue exponentially many queries to the oracle even if standard reasoning over $\mathcal{K}_v \cup \mathcal{K}_h$ requires only polynomial time.

In this paper, we extend the import-by-query framework in several important ways. To weaken the restrictions on the reuse of the role symbols, we employ a more expressive oracle query language: our queries are ABoxes over the symbols from Γ, for which the oracle decides their satisfiability w.r.t. $\mathcal{K}_h$. ABox satisfiability has been implemented in most state-of-the-art DL reasoners, so such a query language seems like a natural choice. We then study the formal properties of import-by-query algorithms in such a setting and prove the following novel results.

1. We show that, even with ABox satisfiability oracles, the presence of nominals in $\mathcal{K}_h$ can preclude the existence of an import-by-query algorithm.
2. We prove that modular reuse is strictly necessary—that is, that no import-by-query algorithm exists if $\mathcal{K}_v$ does not reuse the symbols from Γ in a modular way.
3. We present an import-by-query algorithm for the case when both $\mathcal{K}_v$ and $\mathcal{K}_h$ are in $\mathcal{EL}$ and reuse is modular.
4. Depending on the expressivity of $\mathcal{K}_v$ and $\mathcal{K}_h$, we show that the presence of cyclic axioms can prevent the existence of an import-by-query algorithm.
5. We present an import-by-query algorithm for the case when both $\mathcal{K}_v$ and $\mathcal{K}_h$ are in $\mathcal{ALCHIQ}$, the reuse is modular, and $\mathcal{K}_v$ satisfies a particular acyclicity restriction.

Our results thus map out the landscape of the import-by-query problem, close several important gaps in our previous work, and provide a starting point for implementation.

Some proofs are given in the full version of this paper (see http://www.comlab.ox.ac.uk/people/boris.motik/pubs/).

Preliminaries

In this section, we recapitulate the DL notation used in this paper and present an overview of the hypertableau reasoning algorthms for DLs (Motik, Shearer, and Horrocks 2009).

Description Logics

We first introduce the description logic $\mathcal{ALCHOIQ}$. A *signature* Σ is a disjoint union of countable sets of *atomic concepts* N_C, *atomic roles* N_R, and *individuals* N_I. A *role* is either atomic or an *inverse role* R^- for $R \in N_R$. For R and R' roles, a *role inclusion axiom* has the form $R \sqsubseteq R'$. The set of *concepts* is the smallest set containing $\top$, $\bot$, A, $\{a\}$, $\neg C$, $C_1 \sqcap C_2$, $C_1 \sqcup C_2$, $\exists R.C$, $\forall R.C$, $\geq n\, R.C$, and $\leq n\, R.C$, for A an atomic concept, a an individual, C, C_1, and C_2 concepts, R a role, and n a nonnegative integer. A *concept inclusion axiom* has the form $C_1 \sqsubseteq C_2$ for C_1 and C_2 concepts, and $C_1 \equiv C_2$ is an abbreviation for $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$. A TBox $\mathcal{T}$ is a finite set of concept and role inclusion axioms. An *assertion* has the form $C(a)$, $R(a,b)$, $\neg R(a,b)$, $a \approx b$, or $a \not\approx b$, for C a concept, R a role, and a, b individuals. An ABox $\mathcal{A}$ is a finite set of assertions. A knowledge base $\mathcal{K} = \mathcal{T} \cup \mathcal{A}$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. We use standard definitions of a Σ-interpretation I, satisfiability of $\mathcal{K}$ in I, satisfiability of a concept C w.r.t. $\mathcal{K}$, and other relevant reasoning problems (Baader et al. 2007). For α a concept, a role, an axiom, or a set of axioms, $\mathsf{sig}(\alpha)$ is the *signature* of α—that is, the set of atomic concepts and atomic roles occurring in α.

The DL $\mathcal{ALCHIQ}$ is obtained from $\mathcal{ALCHOIQ}$ by disallowing *nominal concepts* of the form $\{a\}$. Furthermore, the DL $\mathcal{EL}$ (Baader, Brandt, and Lutz 2005) (resp. $\mathcal{FL}_0$ (Baader et al. 2007)) supports only concepts of the form $\top$, $\bot$, A, $C_1 \sqcap C_2$, and $\exists R.C$ (resp. $\top$, $\bot$, A, $C_1 \sqcap C_2$, and $\forall R.C$) for A and R atomic, disallows role inclusion axioms, and supports only assertions of the form $C(a)$ or $R(a,b)$, with C an $\mathcal{EL}$ (resp. $\mathcal{FL}_0$) concept and R an atomic role.

Hypertableau Algorithms for $\mathcal{ALCHIQ}$ and $\mathcal{EL}$

The hypertableau algorithm for $\mathcal{ALCHIQ}$ starts by preprocessing the input KB into so-called *HT-rules*. Let N_V be a set of variables disjoint with the set of individuals N_I. An *atom* is an expression of the form $C(s)$, $R(s,t)$, or $s \approx t$, where $s, t \in N_V \cup N_I$, C is a concept, and R is a role. A *rule* is an expression of the form

$$U_1 \wedge \ldots \wedge U_m \rightarrow V_1 \vee \ldots \vee V_n \tag{1}$$

where U_i and V_j are atoms, $m \geq 0$, and $n \geq 0$. The conjunction $U_1 \wedge \ldots \wedge U_m$ is called the *body*, and the disjunction $V_1 \vee \ldots \vee V_n$ is called the *head*. The empty body and the empty head are written as $\top$ and $\bot$, respectively. Rules are interpreted as universally quantified FOL implications in the usual way. An *HT-rule* is a rule of the form

$$\begin{gathered}\bigwedge A_i(x) \wedge \bigwedge R_{ij}(x, y_i) \wedge \bigwedge S_{ij}(y_i, x) \bigwedge B_{ij}(y_i) \rightarrow \\ \bigvee C_i(x) \vee \bigvee R'_{ij}(x, y_i) \vee \bigvee S'_{ij}(y_i, x) \vee \\ \bigvee D_{ij}(y_i) \vee \bigvee y_i \approx y_j\end{gathered} \tag{2}$$

where R_{ij}, S_{ij}, R'_{ij}, and S'_{ij} are atomic roles; A_i, B_{ij}, and D_{ij} are atomic concepts; C_i are either atomic concepts or concepts of the form $\geq n\, R.A$, or $\geq n\, R.\neg A$; and each variable y_i occurring in the rule body occurs in the rule body. An HT-rule is *Horn* if it contains at most one atom in the head.

Any $\mathcal{ALCHIQ}$ KB can be transformed into an equisatisfiable set of HT-rules and a *normalized* ABox—that is, an ABox containing only assertions of the form $A(a)$, $\neg A(a)$, $R(a,b)$, or $\neg R(a,b)$ with A and R atomic. The following algorithm checks satisfiability of a set of HT-rules $\mathcal{R}$ and a normalized ABox $\mathcal{A}$. In the rest of this paper, we treat concepts of the form $\exists R.C$ as abbreviations for $\geq 1\, R.C$.

Table 1: Hypertableau Derivation Rules

Derivation Rules for $\mathcal{ALCHIQ}$	
Hyp-rule	If 1. $\varrho \in \mathcal{R}$ with ϱ of the form (1) and 2. a mapping σ from the variables in ϱ to the individuals in $\mathcal{A}$ exists such that 2.1 $\sigma(x)$ is not indirectly blocked for each $x \in N_V$, 2.2 $\sigma(U_i) \in \mathcal{A}$ for each $1 \leq i \leq m$, and 2.3 $\sigma(V_j) \notin \mathcal{A}$ for each $1 \leq j \leq n$, then $\mathcal{A}_1 = \mathcal{A} \cup \{\bot\}$ if $n = 0$; $\mathcal{A}_j := \mathcal{A} \cup \{\sigma(V_j)\}$ for $1 \leq j \leq n$ otherwise.
$\geq$-rule	If 1. $\geq n\, R.C(s) \in \mathcal{A}$ with s not blocked in $\mathcal{A}$ and 2. there are no individuals $u_1, \ldots, u_n$ in $\mathcal{A}$ s.t. $\{\mathsf{ar}(R, s, u_i), C(u_i) \mid 1 \leq i \leq n\} \cup$ $\{u_i \not\approx u_j \mid 1 \leq i < j \leq n\} \subseteq \mathcal{A}$, then $\mathcal{A}_1 := \mathcal{A} \cup \{\mathsf{ar}(R, s, t_i),\ C(t_i) \mid 1 \leq i \leq n\} \cup$ $\{t_i \not\approx t_j \mid 1 \leq i < j \leq n\}$ where $t_1, \ldots, t_n$ are fresh successors of s.
$\approx$-rule	If 1. $s \approx t \in \mathcal{A}$ with $s \neq t$ then $\mathcal{A}_1 := \mathsf{merge}_{\mathcal{A}}(s \to t)$ if t is named or s is a descendant of t, and $\mathcal{A}_1 := \mathsf{merge}_{\mathcal{A}}(t \to s)$ otherwise.
$\bot$-rule	If 1. $s \not\approx s \in \mathcal{A}$ or $\{A(s), \neg A(s)\} \subseteq \mathcal{A}$ or $\{R(s,t), \neg R(s,t)\} \subseteq \mathcal{A}$ with s, t not indirectly blocked and 2. $\bot \notin \mathcal{A}$ then $\mathcal{A}_1 := \mathcal{A} \cup \{\bot\}$.

The $\exists$-rule for $\mathcal{EL}$

$\exists$-rule	If $\exists R.C(s) \in \mathcal{A}$ and $\{R(s, a_C), C(a_C)\} \not\subseteq \mathcal{A}$ then $\mathcal{A}_1 := \mathcal{A} \cup \{R(s, a_C), C(a_C)\}$

Definition 1. ***Individuals.*** *For a set of* named individuals N_I*, the set of all individuals* N_X *is inductively defined as the smallest set such that* $N_I \subseteq N_X$ *and, if* $x \in N_X$*, then* $x.i \in N_X$ *for each integer* i*. The individuals* $N_X \setminus N_I$ *are* unnamed*. An individual* $x.i$ *is a* successor *of* x*, and* x *is a* predecessor *of* $x.i$*;* descendant *and* ancestor *are the transitive closures of successor and predecessor, respectively.*

Pairwise Anywhere Blocking. *The* label $\mathcal{L}_{\mathcal{A}}(s)$ *of an individual* s *and the* label $\mathcal{L}_{\mathcal{A}}(s,t)$ *of an individual pair* $\langle s, t\rangle$ *in an ABox* $\mathcal{A}$ *are defined as follows:*

$$\begin{aligned} \mathcal{L}_{\mathcal{A}}(s) &= \{A \mid A(s) \in \mathcal{A} \text{ and } A \text{ is atomic}\} \\ \mathcal{L}_{\mathcal{A}}(s,t) &= \{R \mid R(s,t) \in \mathcal{A}\} \end{aligned}$$

Let $\prec$ *be a strict ordering on* N_X *containing the ancestor relation. By induction on* $\prec$*, we assign to each individual* s *in* $\mathcal{A}$ *a status as follows:*

- s *is directly blocked by* t *iff the following conditions hold, for* s' *and* t' *the predecessors of* s *and* t*, respectively:*
  - s *and* t *are unnamed,* t *is not blocked, and* $t \prec s$*;*[1]
  - $\mathcal{L}_{\mathcal{A}}(s) = \mathcal{L}_{\mathcal{A}}(t)$ *and* $\mathcal{L}_{\mathcal{A}}(s') = \mathcal{L}_{\mathcal{A}}(t')$*; and*
  - $\mathcal{L}_{\mathcal{A}}(s, s') = \mathcal{L}_{\mathcal{A}}(t, t')$ *and* $\mathcal{L}_{\mathcal{A}}(s', s) = \mathcal{L}_{\mathcal{A}}(t', t)$*.*
- s *is* indirectly blocked *iff its predecessor is blocked.*
- s *is* blocked *iff it is either directly or indirectly blocked.*

[1]When reasoning with $\mathcal{ALCHOIQ}$ knowledge bases, individuals s' and t' are also required to be unnamed; however, this restriction is not needed with $\mathcal{ALCHIQ}$ knowledge bases.

Pruning and Merging. *The ABox* $\mathsf{prune}_{\mathcal{A}}(s)$ *is obtained from* $\mathcal{A}$ *by removing all assertions containing a descendant of* s*. The ABox* $\mathsf{merge}_{\mathcal{A}}(s \to t)$ *is obtained from* $\mathsf{prune}_{\mathcal{A}}(s)$ *by replacing* s *with* t *in all assertions.*

Clash. *An ABox* $\mathcal{A}$ *contains a* clash *if* $\bot \in \mathcal{A}$*; otherwise,* $\mathcal{A}$ *is* clash-free*.*

Derivation Rules. *The algorithm consists of the Hyp-,* $\geq$*-,* $\approx$*-, and* $\bot$*-rule from Table 1, which, given* $\mathcal{R}$ *and a clash-free ABox* $\mathcal{A}$*, derive the ABoxes* $\langle \mathcal{A}_1, \ldots, \mathcal{A}_n\rangle$*. In the Hyp-rule,* $\sigma(U)$ *is obtained from* U *by replacing each variable* x *with* $\sigma(x)$*. For a role* R *and individuals* s *and* t*, the function* $\mathsf{ar}(R, s, t)$ *returns the assertion* $R(s,t)$ *if* R *is atomic, or the assertion* $S(t,s)$ *if* R *is an inverse role and* $R = S^-$*.*

Derivation. *A* derivation *for* $\mathcal{R}$ *and* $\mathcal{A}$ *is a pair* (T, ρ) *where* T *is a finitely branching tree and* ρ *labels the nodes of* T *with ABoxes s.t. (*i*)* $\rho(\epsilon) = \mathcal{A}$ *for* ϵ *the root, and (*ii*) for each node* t*, if a derivation rule is applicable to* $\mathcal{R}$ *and* $\rho(t)$*, then* t *has children* $t_1, \ldots, t_n$ *s.t.* $\langle \rho(t_1), \ldots, \rho(t_n)\rangle$ *are the result of applying one derivation rule to* $\mathcal{R}$ *and* $\rho(t)$*. The algorithm returns* t *if some derivation for* $\mathcal{R}$ *and* $\mathcal{A}$ *has a leaf node labeled with a clash-free ABox, and* f *otherwise.*

The hypertableau algorithm for $\mathcal{ALCHIQ}$ can also be applied to $\mathcal{EL}$ KBs. Motik and Horrocks (2008) showed, however, that a worst-case optimal algorithm can be obtained by modifying the $\geq$-rule. This modified algorithm works on a set $\mathcal{R}$ of $\mathcal{EL}$*-rules*—HT-rules of the form (3), where C is either atomic, or of the form $\exists R.A$ with A atomic.

$$\bigwedge_{i=1}^{k} A_i(x) \wedge \bigwedge_{i=1}^{m} \left[R_i(x, y_i) \wedge \bigwedge_{j=1}^{m_i} B_{ij}(y_i) \right] \to C(x) \quad (3)$$

The following algorithm checks satisfiability of $\mathcal{R} \cup \mathcal{A}$, for $\mathcal{R}$ a set of $\mathcal{EL}$-rules and $\mathcal{A}$ a normalized ABox.

Definition 2. *For each named individual* $a \in N_I$ *and each atomic concept* $A \in N_C$*, let* a_A *be a fresh individual that is uniquely associated with* a *and* A*. The hypertableau algorithm for* $\mathcal{EL}$ *follows Definition 1, but the derivation rules include the Hyp-,* $\bot$*-, and* $\exists$*-rule from Table 1.*

Motivating Example and Definitions

To illustrate our framework, consider a medical research company (MRC) that has developed an ontology about human anatomy. The ontology contains concepts describing organs such as **Heart** and **TV** (tricuspid valve); medical conditions such as **CHD** (congenital heart defect), **VSD** (ventricular septum defect), and **AS** (aortic stenosis); and treatments such as Surgery. The roles **part**, **con**, and treatment relate organs with their parts, medical conditions, and treatments, respectively, and they are used to define concepts such as **VSD_Heart** (a heart with a ventricular septal defect) and **Sur_Heart** (a heart that requires surgical treatment). We focus on reusing schema knowledge, so we assume that the ontology consists only of a TBox $\mathcal{T}_h$, given in Table 2. MRC wants to freely distribute information about organs and conditions, but wants to charge for the information about treatments. To this end, MRC identifies a set Γ of *public* symbols of $\mathcal{T}_h$; we write these symbols in **bold**, and the remaining *private* symbols in sans serif. MRC does

Table 2: Example Knowledge Bases

Hidden Knowledge Base $\mathcal{T}_h$	
γ_1	$\mathbf{Heart} \sqsubseteq \mathbf{Organ} \sqcap \exists \mathbf{part}.\mathbf{TV}$
γ_2	$\mathbf{VSD} \sqsubseteq \mathbf{CHD}$
γ_3	$\mathbf{AS} \sqsubseteq \mathbf{CHD}$
γ_4	$\mathbf{VSD\_Heart} \equiv \mathbf{Heart} \sqcap \exists \mathbf{con}.\mathbf{VSD}$
γ_5	$\mathbf{VSD\_Heart} \sqsubseteq \exists \mathsf{treatment}.\mathsf{Surgery}$
γ_6	$\mathbf{Sur\_Heart} \equiv \mathbf{Heart} \sqcap \exists \mathsf{treatment}.\mathsf{Surgery}$
Visible Knowledge Base $\mathcal{K}_v$	
δ_1	$VSD\_Patient \equiv Patient \sqcap \exists hasOrg.\mathbf{VSD\_Heart}$
δ_2	$HS\_Patient \equiv Patient \sqcap \exists hasOrg.\mathbf{Sur\_Heart}$
δ_3	$AS\_Patient \equiv Patient \sqcap \exists hasOrg.(\mathbf{Heart} \sqcap \exists \mathbf{con}.\mathbf{AS})$
δ_4	$Ab\_TV \sqsubseteq \mathbf{TV}$
δ_5	$Dis\_TV \sqsubseteq Ab\_TV$
δ_6	$EA\_Heart \equiv \mathbf{VSD\_Heart} \sqcap \exists \mathbf{part}.Dis\_TV$
δ_7	$EA\_Patient \equiv Patient \sqcap \exists hasOrg.EA\_Heart$
δ_8	$Ab\_TV\_Heart \equiv \mathbf{Heart} \sqcap \exists \mathbf{part}.Ab\_TV$
δ_9	$TVD\_Patient \equiv Patient \sqcap \exists hasOrg.Ab\_TV\_Heart$

not want to distribute the axioms of $\mathcal{T}_h$, as this might allow competitors to copy parts of the ontology.

Consider also a health-care provider (HCP) that reuses $\mathcal{T}_h$ to describe types of patients such as $VSD\_Patient$ (patients with a ventricular septum defect), $HS\_Patient$ (patients requiring heart surgery), $AS\_Patient$ (patients with aortic stenosis), $EA\_Patient$ (patients with Ebstein's anomaly), and $TVD\_Patient$ (patients with a tricuspid valve defect). Since the TBox $\mathcal{T}_h$ does not describe Ebstein's anomaly, HCP defines EA_Heart as a heart with a ventricular septum defect and with a displaced tricuspid valve Dis_TV; furthermore, it defines a displaced tricuspid valve as abnormal, and Ab_TV_Heart as a heart with an abnormal tricuspid valve. The ontology is shown in Table 2, and its private symbols are written in *italic*. Although our example does not use ABox assertions, we allow the visible ontology to contain such assertions in general, so we denote it with $\mathcal{K}_v$. HCP can use $\mathcal{T}_h \cup \mathcal{K}_v$ to conclude $VSD\_Patient \sqsubseteq HS\_Patient$ (patients with ventricular septum defect require heart surgery) and $EA\_Patient \sqsubseteq TVD\_Patient$ (patients with Ebstein's anomaly are a kind of patients with a tricuspid valve defect).

To support such scenarios, in our previous work we proposed the *import-by-query* framework (Cuenca Grau, Motik, and Kazakov 2009). Instead of publishing (a subset of) the axioms of $\mathcal{T}_h$, MRC can publish an *oracle* for $\mathcal{T}_h$—a service that can answer queries over $\mathcal{T}_h$ provided that the queries use only the public symbols of $\mathcal{T}_h$. We presented an *import-by-query algorithm* that allows HCP to reason over $\mathcal{K}_v \cup \mathcal{T}_h$ by using the axioms of $\mathcal{K}_v$ and the oracle. Our framework was based on oracles that decide the satisfiability of a concept C in the DL of $\mathcal{T}_h$, provided that $\mathsf{sig}(C) \subseteq \Gamma$. For reasoning to be possible, we imposed the following restrictions:

R1. $\mathcal{T}_h$ was not allowed to contain nominals.

R2. The TBox of $\mathcal{K}_v$ had to be modular w.r.t. Γ; that is, its axioms could not affect the meaning of the symbols in Γ.

R3. Let a concept C be Γ-*modal* if, for $R \in \Gamma$, it is of the form $\exists R.C$, $\forall R.C$, $\geq n\, R.C$, and $\leq n\, R.C$, and let C be Γ-*restricted* if $\mathsf{sig}(C) \subseteq \Gamma$; then, we required each Γ-modal concept in $\mathcal{K}_v$ to be Γ-restricted.

Restriction R3 is particularly severe, as it prevents mixing roles from $\mathcal{T}_h$ with concepts from $\mathcal{K}_v$ in modal restrictions. Axioms δ_6 and δ_8 from Table 2 violate this restriction.

To overcome these limitations, in this paper we introduce two new (but closely related) types of oracles, which are more powerful than the oracles based on concept satisfiability. An *ABox satisfiability oracle* is given an ABox $\mathcal{A}$ with $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$, and it checks the satisfiability of $\mathcal{A} \cup \mathcal{T}_h$. An *ABox entailment oracle* is given an ABox $\mathcal{A}$ and an assertion α with $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$ and $\mathsf{sig}(\alpha) \subseteq \Gamma$, and it checks whether $\mathcal{A} \cup \mathcal{T}_h \models \alpha$. An ABox entailment oracle can always simulate an ABox satisfiability oracle, and the converse holds provided that $\mathcal{A}$ allows for assertions of the form $\neg C(s)$ and $\neg R(s,t)$. Assertions of the former type are available in many DLs; furthermore, assertions of the latter type can be added to most DLs without any problems, which is why we included such assertions in the definition of the DLs that can do so in the previous section. Therefore, we consider in this paper mainly ABox satisfiability oracles; we use ABox entailment oracles only when the DL of $\mathcal{T}_h$ is Horn and thus does not support assertions of the form $\neg C(s)$.

In practice, it is natural to express a query ABox $\mathcal{A}$ in the same DL as $\mathcal{T}_h$. To obtain general results about infeasibility of reasoning, however, it is useful to allow the DL of $\mathcal{A}$ to be more expressive than the DL of $\mathcal{T}_h$, so that $\mathcal{K}_v$ can "learn more about the models of $\mathcal{T}_h$." We therefore parameterize our oracles with a DL $\mathcal{L}$ that determines the types of assertions allowed in $\mathcal{A}$.

Definition 3. *Let $\mathcal{T}_h$ be a TBox, $\mathcal{L}$ a description logic, and Γ a signature. An* ABox satisfiability *oracle $\Omega^{\mathsf{a}}_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ is a function that, for each $\mathcal{L}$-ABox $\mathcal{A}$ such that $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$, returns t iff $\mathcal{T}_h \cup \mathcal{A}$ is satisfiable. An* ABox entailment *oracle $\Omega^{\mathsf{e}}_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ is a function that, for each $\mathcal{L}$-ABox $\mathcal{A}$ such that $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$ and each $\mathcal{L}$-assertion α that mentions only the individuals in $\mathcal{A}$ such that $\mathsf{sig}(\alpha) \subseteq \Gamma$, returns t iff $\mathcal{T}_h \cup \mathcal{A} \models \alpha$.*

An import-by-query algorithm *takes as input a knowledge base $\mathcal{K}_v$ and an oracle $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ with $\mathsf{sig}(\mathcal{K}_v) \cap \mathsf{sig}(\mathcal{T}_h) \subseteq \Gamma$, and it terminates after a finite number of computation steps returning t iff $\mathcal{K}_v \cup \mathcal{T}_h$ is satisfiable.*

We use the generic term *ABox query oracle* (or simply *oracle*) for either an ABox satisfiability or an ABox entailment oracle. Furthermore, if $\mathcal{L}$ is the same as the logic of $\mathcal{T}_h$, we abbreviate $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ to $\Omega_{\mathcal{T}_h,\Gamma}$.

We finally show that we can without loss of generality assume $\mathcal{K}_v$ to contain no concept that is both Γ-modal and Γ-restricted (such as $\exists \mathbf{con}.\mathbf{AS}$ in axiom δ_3). Intuitively, this is because we can always treat such concepts as "atomic" from the point of view of $\mathcal{K}_v$ and rely on the oracle to compute all relevant consequences of such concepts.

Theorem 1. *Each import-by-query algorithm applicable to an oracle $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ and an input knowledge base $\mathcal{K}_v \in \mathcal{DL}$ that does not contain concepts that are both Γ-modal and Γ-restricted can be converted into an import-by-query algo-*

rithm that handles input knowledge bases containing such concepts, provided that $\mathcal{L}$ allows for $\mathcal{DL}$-concepts.

Proof. Let IbQ' be an import-by-query algorithm satisfying the assumptions of the theorem. For C a concept and α a concept, axiom, or knowledge base, let us say that C is Γ-*outermost* in α if C is Γ-modal and it does not occur in α as a proper subconcept of another Γ-modal subconcept D.

Let $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ be an oracle and $\mathcal{K}_v \in \mathcal{DL}$ a knowledge base that could be handled by IbQ' if each Γ-outermost concept in it were replaced with an atomic concept. Let S be the set of Γ-outermost concepts in $\mathcal{K}_v$, and let X_C be a fresh atomic concept for each $C \in S$. We define Γ', $\mathcal{T}_h'$, and $\mathcal{K}_v'$ as follows: $\Gamma' = \Gamma \cup \{X_C \mid C \in S\}$; $\mathcal{K}_v'$ is obtained from $\mathcal{K}_v$ by replacing each concept $C \in S$ with X_C; and $\mathcal{T}_h' = \mathcal{T}_h \cup \{X_C \equiv C \mid C \in S\}$.

Clearly, $\mathcal{K}_v \cup \mathcal{T}_h$ is equisatisfiable with $\mathcal{K}_v' \cup \mathcal{T}_h'$, and IbQ' is applicable to $\Omega_{\mathcal{T}_h',\Gamma',\mathcal{L}}$ and $\mathcal{K}_v'$. Now let IbQ be the algorithm that on $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ and $\mathcal{K}_v$ behaves as follows.

- IbQ simulates the steps of IbQ' on $\Omega_{\mathcal{T}_h',\Gamma',\mathcal{L}}$ and $\mathcal{K}_v'$ while treating all concepts in S as if they were atomic.
- Whenever IbQ' queries $\Omega_{\mathcal{T}_h',\Gamma',\mathcal{L}}$ with an ABox $\mathcal{A}'$, IbQ queries $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ with an ABox $\mathcal{A}$ obtained from $\mathcal{A}'$ by replacing each concept X_C with C.

Algorithm IbQ clearly returns on $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ and $\mathcal{K}_v$ the same value as IbQ' on $\Omega_{\mathcal{T}_h',\Gamma',\mathcal{L}}$ and $\mathcal{K}_v'$; hence, if $\mathcal{L}$ allows for $\mathcal{DL}$-concepts, IbQ is an import-by-query algorithm for $\Omega_{\mathcal{T}_h,\Gamma,\mathcal{L}}$ and $\mathcal{K}_v$. □

Limits of ABox Query Oracles

In this section we revisit restrictions R1–R3 from our previous work and explore the limitations of the framework based on ABox query oracles. We first justify R1, which was adopted in our previous work for technical reasons, without a formal justification.

Theorem 2. *No import-by-query algorithm based on ABox query oracles exists if $\mathcal{K}_v$ is in a DL without the finite model property and the DL of $\mathcal{T}_h$ provides for nominals and disjunction, even if $\mathcal{L} = \mathcal{ALCHOIQ}$ and $\Gamma = \emptyset$.*

Proof. Assume that an import-by-query algorithm exists, and let $\mathcal{K}_v$ be any knowledge base satisfiable only in infinite models. Since the algorithm terminates, the maximum size of the query ABoxes is bounded; furthermore, since these ABoxes are in $\mathcal{ALCHOIQ}$, an integer n depending only on $\mathcal{K}_v$ and Γ exists such that each satisfiable ABox passed to the oracle has a model containing at most n objects. Let $\mathcal{T}_h^1 = \emptyset$ and $\mathcal{T}_h^2 = \{\top \sqsubseteq \{a_1\} \sqcup \ldots \sqcup \{a_n\}\}$. Clearly, $\mathcal{K}_v \cup \mathcal{T}_h^1$ is satisfiable, but $\mathcal{K}_v \cup \mathcal{T}_h^2$ is not. Consider now an arbitrary $\mathcal{L}$-ABox $\mathcal{A}$ such that $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$. Clearly, $\Omega^{\mathsf{a}}_{\mathcal{T}_h^1,\Gamma}(\mathcal{A}) = \mathsf{t}$ implies $\Omega^{\mathsf{a}}_{\mathcal{T}_h^2,\Gamma}(\mathcal{A}) = \mathsf{t}$, and the converse holds by the monotonicity of first-order logic. Thus, $\Omega^{\mathsf{a}}_{\mathcal{T}_h^1,\Gamma}(\mathcal{A}) = \Omega^{\mathsf{a}}_{\mathcal{T}_h^2,\Gamma}(\mathcal{A})$ for any $\mathcal{A}$, so our algorithm returns the same result when applied to the oracles for $\mathcal{T}_h^1$ and $\mathcal{T}_h^2$; however, this contradicts the fact that $\mathcal{K}_v \cup \mathcal{T}_h^1$ is satisfiable but $\mathcal{K}_v \cup \mathcal{T}_h^2$ is not. □

We next revisit R2. In this paper, we use the *deductive* notion of modularity (Lutz, Walther, and Wolter 2007) and say that $\mathcal{K}_v$ is *modular* w.r.t. Γ if, for all concepts C and D in the DL of $\mathcal{K}_v$ such that $\mathsf{sig}(C) \subseteq \Gamma$ and $\mathsf{sig}(D) \subseteq \Gamma$, $\mathcal{K}_v \models C \sqsubseteq D$ implies $\emptyset \models C \sqsubseteq D$. Previously, we adopted modularity as a "reasonable" assumption, without formal justification. We next present a very strong result: modularity is necessary for the existence of an import-by-query algorithm. Intuitively, without modularity $\mathcal{K}_v$ can arbitrarily influence the models of $\mathcal{T}_h$, and the oracle cannot take this into account since it has no access to the axioms of $\mathcal{K}_v$.

Theorem 3. *For any logic $\mathcal{DL}$ containing at least the constructors of $\mathcal{EL}$ and at most the constructors of $\mathcal{ALCHIQ}$, no import-by-query algorithm based on ABox query oracles exists if $\mathcal{K}_v$ and $\mathcal{T}_h$ are in $\mathcal{DL}$ and $\mathcal{L} = \mathcal{ALCHIQ}$, unless $\mathcal{K}_v$ is modular in the public signature Γ of $\mathcal{T}_h$.*

Proof. Consider any signature Γ and any $\mathcal{ALCHIQ}$ knowledge base $\mathcal{K}_v$ such that $\mathcal{K}_v$ is not modular w.r.t. Γ. Then, possibly complex $\mathcal{DL}$ concepts C and D exist such that $\mathsf{sig}(C) \subseteq \Gamma$, $\mathsf{sig}(D) \subseteq \Gamma$, $\mathcal{K}_v \models C \sqsubseteq D$, and $\emptyset \not\models C \sqsubseteq D$. Assume now that an import-by-query algorithm exists that terminates on $\mathcal{K}_v$, Γ, and any $\mathcal{T}_h$ in a DL as specified in the theorem. Without loss of generality, we can assume $\mathcal{K}_v$ to be satisfiable; otherwise, the import-by-query problem is trivial. Let $\mathcal{T}_h^1$ and $\mathcal{T}_h^2$ be as follows, where $R \notin \Gamma$ and $A \notin \Gamma$.

$$\begin{aligned}\mathcal{T}_h^1 &= \emptyset\\ \mathcal{T}_h^2 &= \{\top \sqsubseteq \exists R.(A \sqcap C),\ A \sqcap D \sqsubseteq \bot\}\end{aligned}$$

Clearly, $\mathcal{K}_v \cup \mathcal{T}_h^1$ is satisfiable, but $\mathcal{K}_v \cup \mathcal{T}_h^2$ is not. Consider now an arbitrary $\mathcal{L}$-ABox $\mathcal{A}$ such that $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$. If $\mathcal{A} \cup \mathcal{T}_h^1$ is unsatisfiable, so is $\mathcal{A} \cup \mathcal{T}_h^2$. Conversely, assume that $\mathcal{A} \cup \mathcal{T}_h^1$ is satisfiable in a model I. Since $\emptyset \not\models C \sqsubseteq D$, an interpretation I_C and a domain element $x \in \triangle^{I_C}$ exist such that $x \in C^{I_C}$ but $x \notin D^{I_C}$. Let I' be the following interpretation:

$$\begin{aligned}\triangle^{I'} &= \triangle^{I} \cup \triangle^{I_C}\\ R^{I'} &= \{\langle o, x\rangle \mid o \in \triangle^{I'}\}\\ X^{I'} &= X^{I} \cup X^{I_C} \text{ for each atomic concept } X \notin \Gamma\\ A^{I'} &= \{x\}\end{aligned}$$

Clearly, $I' \models \mathcal{T}_h^2$; furthermore, since the concepts in $\mathcal{A}$ are in $\mathcal{ALCHIQ}$ and do not contain R, we have $I' \models \mathcal{A}$ as well. Hence, $\Omega^{\mathsf{a}}_{\mathcal{T}_h^1,\Gamma,\mathcal{L}}(\mathcal{A}) = \Omega^{\mathsf{a}}_{\mathcal{T}_h^2,\Gamma,\mathcal{L}}(\mathcal{A})$, which proves our claim as in the proof of Theorem 2. □

We finally focus on R3. The proof of nonexistence of an import-by-query algorithm from our previous work uses an $\mathcal{EL}$ ontology $\mathcal{K}_v$ that entails a cyclic axiom of the form $A \sqsubseteq \exists R.A$ with $R \in \Gamma$ but $A \notin \Gamma$. We introduced R3 as a possible way of invalidating this proof.

Later in this paper, we provide an import-by-query algorithm based on ABox query oracles for $\mathcal{K}_v$ and $\mathcal{T}_h$ in $\mathcal{EL}$, and where R3 is not required to hold. This shows the advantage of ABox-based over concept-based oracles. However, we next show that ABox query oracles are not sufficiently expressive if is allowed to $\mathcal{T}_h$ contain universal quantifiers.

Theorem 4. *No import-by-query algorithm based on ABox query oracles exists for $\mathcal{K}_v$ in $\mathcal{EL}$ and $\mathcal{T}_h$ in $\mathcal{FL}_0$, even if the TBox of $\mathcal{K}_v$ is modular, $\mathcal{T}_h$ is Horn, and $\mathcal{L} = \mathcal{ALCHIQ}$.*

Proof. Assume that an import-by-query algorithm exists, and let $\mathcal{K}_v = \{A(a), Z(a), A \sqsubseteq \exists R.A\}$ and $\Gamma = \{R, Z\}$. The TBox of $\mathcal{K}_v$ is modular w.r.t. Γ: for each interpretation I for Γ, the interpretation J such that $X^J = X^I$ for each $X \in \Gamma$ and $X^J = \emptyset$ for each $X \notin \Gamma$ is a model of the TBox of $\mathcal{K}_v$, which implies deductive modularity.

Since the algorithm terminates on Γ and $\mathcal{K}_v$, there is a bound on the number of questions posed to an oracle that depends only on Γ and $\mathcal{K}_v$. Thus, the number of individuals (resp. the number of existentially quantified concepts) in each ABox passed to the oracle is bounded by some integer n (resp. m). Let $k = n + m + 1$ and let $C_1, \ldots, C_k$ be distinct and fresh atomic concepts. Consider the following Horn-$\mathcal{FL}_0$ TBoxes:

$$\begin{aligned}\mathcal{T}_h^1 &= \{C_i \sqcap C_j \sqsubseteq \bot \mid 1 \leq i < j \leq k\} \cup \{Z \sqsubseteq C_1\} \cup \\ &\quad \{C_{i-1} \sqsubseteq \forall R.C_i \mid 1 < i \leq k\} \cup \{C_k \sqsubseteq \forall R.C_1\} \\ \mathcal{T}_h^2 &= \mathcal{T}_h^1 \cup \{C_k \sqsubseteq \bot\}\end{aligned}$$

Clearly, $\mathcal{K}_v \cup \mathcal{T}_h^1$ is satisfiable, whereas $\mathcal{K}_v \cup \mathcal{T}_h^2$ is not. We next show that, for each $\mathcal{L}$-ABox $\mathcal{A}$ with $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$ with at most n individuals and concepts of quantifier depth at most m, we have $\Omega^{\mathsf{a}}_{\mathcal{T}_h^1,\Gamma}(\mathcal{A}) = \Omega^{\mathsf{a}}_{\mathcal{T}_h^2,\Gamma}(\mathcal{A})$, which proves our claim as in the proof of Theorem 2. Due to the monotonicity of first-order logic, satisfiability of $\mathcal{A} \cup \mathcal{T}_h^2$ implies satisfiability of $\mathcal{A} \cup \mathcal{T}_h^1$, and we next show the converse. We say that an individual c in $\mathcal{A}$ is j *steps away* from b_0 if $\{Z(b_0), R(b_0, b_1), \ldots, R(b_{j-1}, b_j)\} \subseteq \mathcal{A}$ for some individuals $b_1, \ldots, b_j$ with $b_j = c$; in such a case, we have $\mathcal{A} \cup \mathcal{T}_h^1 \models C_j(c)$. Let $\mathcal{R}_h^1$ and $\mathcal{A}'$ be the result of transforming $\mathcal{T}_h^1$ and $\mathcal{A}$ into HT-rules. Since the algorithm from Definition 1 is sound and complete, there is a clash-free ABox $\mathcal{A}''$ labeling a leaf of a derivation for $\mathcal{R}_h^1$ and $\mathcal{A}'$. Since $\mathcal{A}$ contains at most m existentially quantified concepts, $\mathcal{A}''$ contains at most m unnamed individuals. But then, $\mathcal{A}'' \cup \mathcal{T}_h^2$ can only be unsatisfiable if individuals b and c exist such that c is k steps away from b. Since $\mathcal{A}$ contains at most $n + m$ individuals and $n + m < k$, an individual d exists that is both j_1 and j_2 steps away from b, where $j_1 \neq j_2$. But then, $\mathcal{A}'' \cup \mathcal{T}_h^1$ is unsatisfiable, and $\mathcal{A} \cup \mathcal{T}_h^1$ is unsatisfiable as well, which is a contradiction. □

The proof of Theorem 4 again assumes that $\mathcal{K}_v$ entails an axiom $A \sqsubseteq \exists R.A$ with $R \in \Gamma$ and $A \notin \Gamma$, which implies $A \sqsubseteq \exists R^n.A$ for arbitrary n. Through universal quantification, $\mathcal{T}_h$ can now "propagate" information along an R-chain to an unknown level m. An import-by-query algorithm cannot determine up to which depth the model of $\mathcal{K}_v$ needs to be examined, which prevents termination. Later in this paper, we present a sufficient acyclicity restriction on the axioms of $\mathcal{K}_v$ that bounds n and thus ensures termination.

Finally, the proof of the following theorem shows that acyclicity and modularity are not sufficient if $\mathcal{K}_v$ can propagate statements about the symbols private to $\mathcal{K}_v$ into a model of $\mathcal{T}_h$; this can be achieved, for example, using universal quantifiers. In a subsequent section, we introduce a safety condition that prevents such propagation and, ultimately, allows us to devise an import-by-query algorithm.

Theorem 5. *No import-by-query algorithm based on ABox query oracles exists for $\mathcal{K}_v$ in $\mathcal{FL}_0$ and $\mathcal{T}_h$ in $\mathcal{EL}$, even if the TBox of $\mathcal{K}_v$ is modular and $\mathcal{L} = \mathcal{ALCHIQ}$.*

Proof. Assume that an algorithm exists. Let $\Gamma = \{R, B, Z\}$ and let $\mathcal{K}_v = \{A(a), Z(a), A \sqsubseteq \forall R.A, A \sqsubseteq B\}$. That the TBox of $\mathcal{K}_v$ is modular w.r.t. Γ can be shown as in the proof of Theorem 4. Since the algorithm terminates, there is a bound on the number of oracle queries that depends only on Γ and $\mathcal{K}_v$. Let n be maximum quantifier depth of an $\mathcal{L}$-concept in a query ABox, and let $\mathcal{T}_h^1$ and $\mathcal{T}_h^2$ be as follows:

$$\mathcal{T}_h^1 = \{Z \sqsubseteq \underbrace{\exists R \ldots \exists R}_{n+1 \text{ times}}.D\} \quad \mathcal{T}_h^2 = \mathcal{T}_h^1 \cup \{B \sqcap D \sqsubseteq \bot\}$$

Clearly, $\mathcal{K}_v \cup \mathcal{T}_h^1$ is satisfiable, whereas $\mathcal{K}_v \cup \mathcal{T}_h^2$ is not. We show that $\Omega^{\mathsf{a}}_{\mathcal{T}_h^1,\Gamma,\mathcal{L}}(\mathcal{A}) = \Omega^{\mathsf{a}}_{\mathcal{T}_h^2,\Gamma,\mathcal{L}}(\mathcal{A})$ for each $\mathcal{L}$-ABox $\mathcal{A}$ with $\mathsf{sig}(\mathcal{A}) \subseteq \Gamma$ and with concepts of quantifier depth at most n. This clearly holds if $\mathcal{T}_h^1 \cup \mathcal{A}$ is unsatisfiable, so assume that $\mathcal{T}_h^1 \cup \mathcal{A}$ is satisfiable. Let $\mathcal{R}_h^1$ and $\mathcal{A}'$ be the result of transforming $\mathcal{T}_h^1$ and $\mathcal{A}$ into a set of HT-rules and a normalized ABox. Since the algorithm from Definition 1 is sound and complete, there is a clash-free ABox $\mathcal{A}''$ labeling a leaf of a derivation for $\mathcal{R}_h^1$ and $\mathcal{A}'$. Since D does not occur in $\mathcal{A}'$, if $D(s) \in \mathcal{A}''$, then s is at least $n + 1$ steps away from any individual a such that $Z(a) \in \mathcal{A}''$. Since $\mathcal{A}$ contains concepts with quantifier depth at most n, we have that $D(s) \in \mathcal{A}''$ implies $B(s) \notin \mathcal{A}''$; but then, no derivation rule is applicable to $\mathcal{R}_h^2 \cup \mathcal{A}''$ for $\mathcal{R}_h^2$ the result of transforming $\mathcal{T}_h^2$ into HT-rules, so $\mathcal{T}_h^2 \cup \mathcal{A}$ is satisfiable. Thus, $\Omega^{\mathsf{a}}_{\mathcal{T}_h^1,\Gamma,\mathcal{L}}(\mathcal{A}) = \Omega^{\mathsf{a}}_{\mathcal{T}_h^2,\Gamma,\mathcal{L}}(\mathcal{A}) = \mathsf{t}$, which proves our claim as in the proof of Theorem 2. □

Import-by-Query Algorithms

We next identify positive cases for which an import-by-query algorithm exists. For simplicity, in all algorithms we assume that $\mathcal{K}_v$ does not contain concepts that are both Γ-modal and Γ-restricted; by Theorem 1 this is without loss of generality. Our algorithms extend the hypertableau algorithms for $\mathcal{ALCHIQ}$ and $\mathcal{EL}$ given in the preliminaries.

Import-by-Query in $\mathcal{EL}$

In this section we present an import-by-query algorithm based on ABox entailment oracles that is applicable when $\mathcal{K}_v$ and $\mathcal{T}_h$ are in $\mathcal{EL}$. The only relevant negative result is given in Theorem 3, so $\mathcal{K}_v$ must be modular w.r.t. Γ. We use a stronger condition and require $\mathcal{K}_v$ to be *local* w.r.t. Γ; for $\mathcal{K}_v$ in $\mathcal{EL}$, this is the case if $\mathsf{sig}(C) \not\subseteq \Gamma$ for each concept inclusion $C \sqsubseteq D \in \mathcal{K}_v$ (Cuenca Grau et al. 2008). While the design of an import-by-query algorithm that requires only modularity is an open problem, we do not believe locality to be a severe limitation in practice: determining modularity of $\mathcal{K}_v$ w.r.t. Γ is EXPTIME-complete (Lutz and Wolter 2009), and no practical algorithm is presently known. Note that our running example satisfies the locality requirement.

Our algorithm is based on the hypertableau framework, so $\mathcal{K}_v$ must first be converted into a set $\mathcal{R}_v$ of $\mathcal{EL}$-rules and a

normalized ABox $\mathcal{A}_v$. It is straightforward to see that, if $\mathcal{K}_v$ is local and does not contain concepts that are Γ-modal and Γ-restricted, then $\mathcal{R}_v$ is $\mathcal{EL}$-safe, as specified next.

Definition 4. *A set $\mathcal{R}_v$ of $\mathcal{EL}$-rules is $\mathcal{EL}$-*safe *w.r.t. a signature Γ if, for each rule $\varrho \in \mathcal{R}_v$,*

- *ϱ contains a body atom α such that $\mathsf{sig}(\alpha) \notin \Gamma$, and*
- *for each body atom in ϱ of the form $R(x, y_i)$ with $R \in \Gamma$, there is a body atom in ϱ of the form $B(y_i)$ with $B \notin \Gamma$.*

Our algorithm takes a set $\mathcal{R}_v$ of $\mathcal{EL}$-safe rules and a normalized ABox $\mathcal{A}_v$. It applies the standard $\mathcal{EL}$ hypertableau derivation rules, as well as an additional rule that, given an ABox $\mathcal{A}_i$ in a derivation, asks the oracle to "complete" $\mathcal{A}_i$ with the relevant assertions entailed by $\mathcal{T}_h \cup \mathcal{A}_i$.

Definition 5. *The $\mathcal{EL}$ Ω^{e}-algorithm takes a set $\mathcal{R}_v$ of $\mathcal{EL}$-rules, a normalized ABox $\mathcal{A}_v$, and an ABox entailment oracle $\Omega^{\mathsf{e}}_{\mathcal{T}_h,\Gamma}$ such that $\mathcal{R}_v$ is $\mathcal{EL}$-safe w.r.t. Γ. The algorithm is obtained by extending the algorithm in Definition 2 with the following derivation rule, where $\mathcal{A}|_\Gamma$ is constructed from $\mathcal{A}$ by removing each assertion $\alpha \in \mathcal{A}$ such that $\mathsf{sig}(\alpha) \not\subseteq \Gamma$:*

> *(Ω^{e}-rule): If for some $C \in \Gamma \cup \{\bot\}$ and individual s in $\mathcal{A}$ we have $\Omega^{\mathsf{e}}_{\mathcal{T}_h,\Gamma}(\mathcal{A}|_\Gamma, C(s)) = \mathsf{t}$ and $C(s) \notin \mathcal{A}$, then $\mathcal{A}_1 := \mathcal{A} \cup \{C(s)\}$.*

Our algorithm is indeed an import-by-query algorithm, and it can be implemented to run in polynomial time, as shown by the following theorem.

Theorem 6. *The $\mathcal{EL}$ Ω^{e}-algorithm is an import-by-query algorithm and it can be implemented such that it runs in time polynomial in the size of $\mathcal{R}_v \cup \mathcal{A}_v$ with a polynomial number of calls to $\Omega^{\mathsf{e}}_{\mathcal{T}_h,\Gamma}$.*

We next explain the intuition behind this result. The $\mathcal{EL}$ Ω^{e}-algorithm is deterministic, so each derivation of the algorithm has a single leaf node labeled with a uniquely defined ABox $\mathcal{A}^{\mathsf{e}}$. We prove that $\mathcal{R}_v \cup \mathcal{A}_v \cup \mathcal{T}_h$ is satisfiable iff $\bot \notin \mathcal{A}^{\mathsf{e}}$. To this end, let $\mathcal{R}_h$ be the result of transforming $\mathcal{T}_h$ into $\mathcal{EL}$-rules, and let $\mathcal{A}^{\mathcal{EL}}$ be the ABox obtained by applying the standard $\mathcal{EL}$ hypertableau algorithm to $\mathcal{R}_v$, $\mathcal{A}_v$, and $\mathcal{R}_h$. Since the latter algorithm is sound and complete, it suffices to show that $\bot \in \mathcal{A}^{\mathsf{e}}$ iff $\bot \in \mathcal{A}^{\mathcal{EL}}$. It is straightforward to see that $\mathcal{A}^{\mathsf{e}} \subseteq \mathcal{A}^{\mathcal{EL}}$; thus, $\bot \in \mathcal{A}^{\mathsf{e}}$ implies $\bot \in \mathcal{A}^{\mathcal{EL}}$. For the converse, we prove that $\mathcal{A}^{\mathcal{EL}} \subseteq \mathcal{A}^{\mathsf{e}} \cup \mathsf{sat}(\mathcal{R}_h, \mathcal{A}^{\mathsf{e}}|_\Gamma)$, where $\mathsf{sat}(\mathcal{R}_h, \mathcal{A}^{\mathsf{e}}|_\Gamma)$ is the result of applying the standard $\mathcal{EL}$ algorithm to $\mathcal{A}^{\mathsf{e}}|_\Gamma$ and $\mathcal{R}_h$; in other words, we show that the assertions in $\mathcal{A}^{\mathcal{EL}} \setminus \mathcal{A}^{\mathsf{e}}$ can be obtained by applying the standard $\mathcal{EL}$ algorithm to $\mathcal{A}^{\mathsf{e}}|_\Gamma$ and $\mathcal{R}_h$. Intuitively, this can be done for two reasons: first, the Ω^{e}-rule "transfers" all relevant consequences of $\mathcal{R}_h$ from $\mathcal{A}^{\mathcal{EL}}$ into $\mathcal{A}^{\mathsf{e}}$; and second, $\mathcal{EL}$-safety ensures that the $\mathcal{EL}$-rules in $\mathcal{R}_v$ do not propagate information from the visible into the hidden part.

We illustrate these ideas by means of an example. Let $\Gamma = \{C, R, S\}$ and let $\mathcal{R}_v$ and $\mathcal{T}_h$ be defined as follows:

$$\begin{aligned}\mathcal{R}_v &= \{\ A(x) \to \exists R.B(x), \quad B(x) \to \exists S.A(x),\\ &\qquad A(x) \wedge C(x) \to \exists T.C(x)\ \}\\ \mathcal{T}_h &= \{\ \exists R.\top \sqsubseteq C, \quad C \sqsubseteq \exists S.D\ \}\end{aligned}$$

Figure 1(a) shows the ABox $\mathcal{A}^{\mathsf{e}}$ obtained by applying the $\mathcal{EL}$ Ω^{e}-algorithm to $\mathcal{R}_v$ and $\mathcal{T}_h$. Note that the Ω^{e}-rule introduces assertion $C(a_A)$ into $\mathcal{A}^{\mathsf{e}}$. Figure 1(b) shows the ABox obtained by applying the standard $\mathcal{EL}$ algorithm to $\mathcal{R}_h$ and $\mathcal{A}^{\mathsf{e}}|_\Gamma$; this ABox contains the assertions necessary to satisfy $\mathcal{T}_h$. Finally, Figure 1(c) shows the final ABox $\mathcal{A}^{\mathcal{EL}}$. Due to $\mathcal{EL}$-safety, assertions $S(a_A, a_D)$ and $S(a_C, a_D)$ cannot trigger an application of an $\mathcal{EL}$-rule in $\mathcal{R}_v$; hence, the $\mathcal{EL}$-rules in $\mathcal{R}_v$ are "confined" to individuals a_A, a_B, and a_C, for which the Ω^{e}-rule adds all relevant assertions to $\mathcal{A}^{\mathsf{e}}$.

Import-by-Query in $\mathcal{ALCHIQ}$

In this section we present an import-by-query algorithm based on ABox satisfiability oracles that is applicable to $\mathcal{K}_v$ and $\mathcal{T}_h$ in $\mathcal{ALCHIQ}$. Our algorithm is based on the hypertableau framework, so $\mathcal{K}_v$ must first be converted into a set $\mathcal{R}_v$ of HT-rules and a normalized ABox $\mathcal{A}_v$. To ensure modularity as required by Theorem 3, we require $\mathcal{R}_v$ to satisfy the safety condition from Definition 4. We next devise further restrictions on $\mathcal{R}_v$ that allow us to overcome the negative results of Theorems 4 and 5.

Safety and Acyclicity To overcome the negative result of Theorem 5, we extend the notion of safety to prevent the transfer of information private to $\mathcal{R}_v$ into $\mathcal{T}_h$. This prevents, for example, $\mathcal{K}_v$ from containing axioms of the form $A \sqsubseteq \forall R.B$ where $R \in \Gamma$ and $\{A, B\} \cap \Gamma = \emptyset$.

Definition 6. *A set of HT-rules $\mathcal{R}_v$ is HT-*safe *w.r.t. a signature Γ if each rule $\varrho \in \mathcal{R}_v$ is $\mathcal{EL}$-safe w.r.t. Γ and, in addition, for each atom in the body of ϱ of the form $R(x, y_i)$ or $R(y_i, x)$ with $R \in \Gamma$, the body of ϱ contains atoms of the form $A(x)$ and $B(y_i)$ such that $A \notin \Gamma$ and $B \notin \Gamma$.*

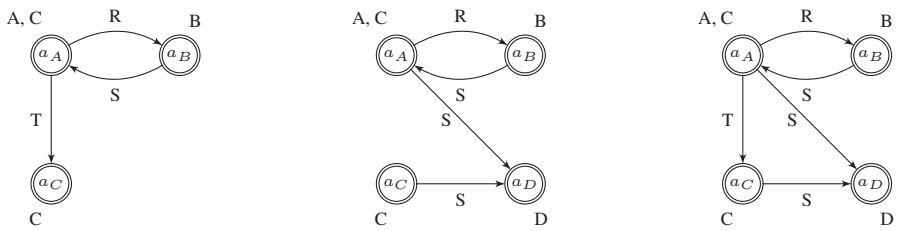
As to the negative result of Theorem 4, note that this result relies on the fact that the visible knowledge base can entail a cyclic axiom $A \sqsubseteq \exists R.A$ with $R \in \Gamma$ and $A \notin \Gamma$. We next present a sufficient test for the detection of such cycles. The test first constructs a graph-like structure G that "summarizes" the models of $\mathcal{R}_v \cup \mathcal{T}_h \cup \mathcal{A}_v$; more precisely, the projection of each model of $\mathcal{R}_v \cup \mathcal{T}_h \cup \mathcal{A}_v$ to the symbols in $\mathsf{sig}(\mathcal{R}_v)$ can always be homomorphically embedded into G. The structure G satisfies the conditions from Table 3. Intuitively, since the axioms of $\mathcal{T}_h$ are not physically available, Conditions 4–7 reflect in G any possible consequence of $\mathcal{T}_h$. Conditions 1–3 reflect in G the information that could be derived using $\mathcal{R}_v \cup \mathcal{A}_v$ and the possible consequences of $\mathcal{T}_h$. The proof of Proposition 1 shows that G can be obtained from $\mathcal{R}_v$ and $\mathcal{A}_v$ as the least fixpoint of a monotonic operator that mimics the conditions from Table 3.

Definition 7. *Let Γ be a signature, $\mathcal{R}$ a set of HT-rules, $\mathcal{A}$ an ABox, and let $V = V_1 \cup V_2$ be defined as follows:*

$$\begin{aligned}V_1 &= \{\ v_a \ \mid\ a \text{ is an individual occurring in } \mathcal{A}\ \}\\ V_2 &= \{\ v_A, v_{\neg A} \ \mid\ A \text{ is a concept in } \mathsf{sig}(\mathcal{R}) \cup \mathsf{sig}(\mathcal{A})\ \}\end{aligned}$$

A structure $G = (\sim, E, \lambda)$ for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ is a triple with the following elements:

- *$\sim$ is an equivalence relation on V. Let W be the set of equivalence classes of $\sim$ and $[\cdot]_\sim : V \to W$ the function that assigns to each $v \in V$ its equivalence class $[v]_\sim$.*



(a) $\mathcal{A}^{\mathsf{e}}$ (b) $\mathsf{sat}(\mathcal{R}_h, \mathcal{A}^{\mathsf{e}}|_\Gamma)$ (c) $\mathcal{A}^{\mathcal{EL}}$

Figure 1: An Illustration of the Completeness Argument for the $\mathcal{EL}$ Ω^{e}-Algorithm

Table 3: Conditions for Structure Stability

1. $C \in \lambda([v_a]_\sim)$ for each $C(a) \in \mathcal{A}$; $R \in \lambda([v_a]_\sim, [v_b]_\sim)$ for each $R(a,b) \in \mathcal{A}$; and $A \in \lambda([v_A]_\sim)$ and $\neg A \in \lambda([v_{\neg A}]_\sim)$ for each $A \in \mathsf{sig}(\mathcal{R}) \cup \mathsf{sig}(\mathcal{A})$.
2. If $\geq n\, R.C \in \lambda(w)$, then $\langle w, [v_C]_\sim \rangle \in E$ and
   - $R \in \lambda(w, [v_C]_\sim)$ if R is an atomic role, or
   - $S \in \lambda([v_C]_\sim, w)$ if R is an inverse role with $R = S^-$.
3. For each $\varrho \in \mathcal{R}$ of the form (2) and each $w, w_1, \ldots, w_n \in W$, if for all body atoms of ϱ we have $A_i \in \lambda(w)$, $R_{ij} \in \lambda(w, w_i)$, $S_{ij} \in \lambda(w_i, w)$, and $B_{ij} \in \lambda(w_i)$, then for each head atom of ρ we have $C_i \in \lambda(w)$, $R'_{ij} \in \lambda(w, w_i)$, $S'_{ij} \in \lambda(w_i, w)$, $D_{ij} \in \lambda(w_i)$, and $w_i = w_j$.
4. $A \in \lambda(w)$ and $\neg A \in \lambda(w)$ for each $w \in W$ and $A \in \Gamma$.
5. If $R' \in \lambda(w, w')$ for some $R' \in \Gamma$, then $R \in \lambda(w, w')$ for each $R \in \Gamma$.
6. If $R \in \lambda(w, w')$ for some $R \in \Gamma$, then $R \in \lambda(w', w)$.
7. If, for some $R \in \Gamma$ and $R' \in \Gamma$, we have that $R \in \lambda(w, w_1)$ and $R' \in \lambda(w, w_2)$, or $R \in \lambda(w, w_1)$ and $R' \in \lambda(w_2, w)$, or $R \in \lambda(w_1, w)$ and $R' \in \lambda(w_2, w)$, then $w_1 = w_2$.

- *$E \subseteq W \times W$ is a relation on W.*
- *λ is a function that assigns to each $w \in W$ a possibly empty set of concepts $\lambda(w)$ and each $\langle w, w' \rangle \in W \times W$ a possibly empty set of atomic roles $\lambda(w, w')$.*

A structure *$G = (\sim, E, \lambda)$ for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ is* stable *if it satisfies the conditions in Table 3.*

We define the partial order $\leq$ on structures such that, for $G_1 = (\sim_1, E_1, \lambda_1)$ and $G_2 = (\sim_2, E_2, \lambda_2)$, we have $G_1 \leq G_2$ iff $\sim_1 \subseteq \sim_2$ and, for each $v, v' \in V$,

- *$\langle [v]_{\sim_1}, [v']_{\sim_1} \rangle \in E_1$ implies $\langle [v]_{\sim_2}, [v']_{\sim_2} \rangle \in E_2$,*
- *$\lambda_1([v]_{\sim_1}) \subseteq \lambda_2([v]_{\sim_2})$, and*
- *$\lambda_1([v]_{\sim_1}, [v']_{\sim_1}) \subseteq \lambda_2([v]_{\sim_2}, [v']_{\sim_2})$.*

A dependency structure *is each smallest structure (w.r.t. $\leq$) that is stable for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ.*

Proposition 1. *The dependency structure for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ is unique.*

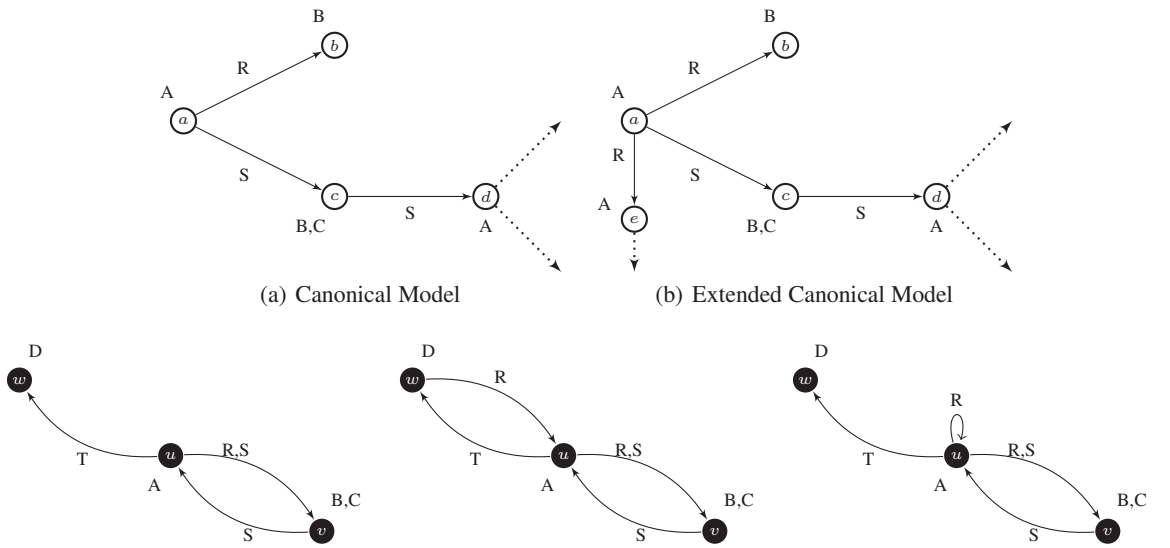
Proof (Sketch). We define an operator T that maps a structure $G_1 = (\sim_1, E_1, \lambda_1)$ into a structure $G_2 = (\sim_2, E_2, \lambda_2)$. Let W_1 be the set of equivalence classes of $\sim_1$. The structure G_2 is obtained by initially setting $G_2 := G_1$ and then modifying G_2 as follows:

- λ_2 is extended such that it satisfies Conditions 1 and 4 from Table 3.
- For each $w \in W_1$ and each concept $\geq n\, R.C \in \lambda_1(w)$, the pair $\langle w, [v_C]_{\sim_1} \rangle$ is added to E_2; if R is atomic, then R is added to $\lambda_2(w, [v_C]_\sim)$; and if R is of the form S^- with S atomic, then S is added to $\lambda_2([v_C]_\sim, w)$.
- For each HT-rule $\varrho \in \mathcal{R}$ of the form (2) and each $w, w_1, \ldots, w_n \in W_1$ such that, for each body atom of ϱ, we have $A_i \in \lambda_1(w)$, $R_{ij} \in \lambda_1(w, w_i)$, $S_{ij} \in \lambda_1(w_i, w)$, and $B_{ij} \in \lambda_1(w_i)$, the following modifications are performed for each head atom of ϱ: C_i is added to $\lambda_2(w)$, D_{ij} is added to $\lambda_2(w_i)$, R'_{ij} is added to $\lambda_2(w, w_i)$, S'_{ij} is added to $\lambda_2(w_i, w)$, and relation $\sim_2$ is extended such that w_i becomes equal to w_j.
- For each $w, w' \in W_1$ such that $R' \in \lambda_1(w, w')$ for some $R' \in \Gamma$, R is added to $\lambda_2(w, w')$ for each $R \in \Gamma$.
- For each $w, w' \in W_1$ such that $R \in \lambda_1(w, w')$ for some $R \in \Gamma$, R is added to $\lambda_2(w', w)$.
- For each $w, w_1, w_2 \in W_1$ such that $R \in \lambda_1(w, w_1)$ and $R' \in \lambda_1(w, w_2)$, or $R \in \lambda_1(w, w_1)$ and $R' \in \lambda_1(w_2, w)$, or $R \in \lambda_1(w_1, w)$ and $R' \in \lambda_1(w_2, w)$ for some $R \in \Gamma$ and $R' \in \Gamma$, relation $\sim_2$ is extended such that w_1 becomes equal to w_2.

It is straightforward to check that T is monotone on the lattice of all structures for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ, and that if $T(G) = G$, then G is stable. Thus, by the well-known Knaster-Tarski theorem, T has a unique least fixpoint, which corresponds to the dependency structure for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ. □

Our test then checks whether the dependency structure G for $\mathcal{R}_v$ and $\mathcal{A}_v$ contains a "harmful cycle"—that is, a cycle that is not confined to $\mathcal{A}_v$ and that involves only roles from Γ. Proposition 2 shows that the overall check can be performed in polynomial time.

Definition 8. *Let G be the dependency structure for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ. A pair $\langle w, w' \rangle \in W \times W$ is* harmful *in G if $\langle w, w' \rangle \in E$, $w \cap V_2 \neq \emptyset$, $w' \cap V_2 \neq \emptyset$, and an atomic role $R \in \Gamma$ exists such that $R \in \lambda(w, w')$ or $R \in \lambda(w', w)$. A structure G contains a* harmful cycle *if $w_1, \ldots, w_n \in W$ exist such that $\langle w_i, w_j \rangle$ is harmful for each $1 \leq i \leq n$ and $j = i + 1 \mod n$; furthermore, G is* acyclic *if it does not contain a harmful cycle. Finally, $\mathcal{R} \cup \mathcal{A}$ is* acyclic *w.r.t. Γ if the dependency structure for $\mathcal{R} \cup \mathcal{A}$ and Γ is acyclic.*



(a) Canonical Model (b) Extended Canonical Model

(c) Dependency Structure (d) Extended Dependency Structure (I) (e) Extended Dependency Structure (II)

Figure 2: Dependency Structures and Acyclicity

Proposition 2. *Acyclicity of $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ can be checked in polynomial time.*

Proof (Sketch). Let V be as specified in Definition 7, let $G = (\sim, E, \lambda)$ be a structure for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ, let W be the set of equivalence classes of $\sim$, and let $|G|$ be the size of G (which we assume to be defined in a straightforward way). Since $|V|$ is linear in the size of $\mathcal{R}$, $\mathcal{A}$, and Γ, we have that $|W|$ is linear, and $|G|$ is polynomial. The structure $T(G)$ can be computed in polynomial time. The only nontrivial case is the third item in the definition of T, which requires matching an HT-rule $\varrho \in \mathcal{R}$ to G. Let n be the number of variables in ϱ. Each consequent atom has at most two variables and the body atoms of ϱ are connected in a tree-like manner, so the relevant consequent atoms of ϱ can be computed in at most $|V|^2 \times n$ steps. Thus, the dependency structure for $\mathcal{R} \cup \mathcal{A}$ w.r.t. Γ can be computed by polynomially many applications of T, each which can be computed in polynomial time. □

The acyclicity condition significantly relaxes condition R3 from our previous work; for example, it allows us to express axioms δ_6 and δ_8 from Table 2. Intuitively, it ensures that "canonical" models of $\mathcal{R}_v \cup \mathcal{A}_v$ (i.e., models containing the least possible information derivable from $R_v \cup \mathcal{A}_v$) do not contain infinite chains of roles from Γ. We use this fact in our algorithm to define a suitable blocking condition. We explain this intuition on an example where $\Gamma = \{C, R, T\}$, $\mathcal{A}_v = \{A(a)\}$, and $\mathcal{R}_v$ contains the HT-rules (4)–(9).

$$A(x) \rightarrow \exists R.B(x) \tag{4}$$
$$A(x) \rightarrow \exists S.B(x) \tag{5}$$
$$A(x) \rightarrow \exists S.C(x) \tag{6}$$
$$S(x,y) \wedge S(x,z) \rightarrow y \approx z \tag{7}$$
$$B(x) \wedge C(x) \rightarrow \exists S.A(x) \tag{8}$$
$$R(x,y) \wedge C(y) \rightarrow \exists T.D(x) \tag{9}$$

Figure 2(a) shows a canonical model I of $\mathcal{R}_v \cup \mathcal{A}_v$, and Figure 2(c) shows the relevant part of the corresponding dependency structure G (for simplicity, we do not show the part of the structure corresponding to $\mathcal{A}_v$). The repetitive structure of I is represented in G as a cycle over nodes u and v. Since $S \notin \Gamma$, this cycle is not harmful, and $R_v \cup \mathcal{A}_v$ is acyclic w.r.t. Γ. Note, however, that a dependency structure overestimates the canonical models; for example, G contains a link between u and w labeled with T, which is not reflected in I. This becomes important if, for example, we extend R_v with the HT-rule (10). This extension clearly does not change the canonical models of $R_v \cup \mathcal{A}_v$; however, the new dependency structure, shown in Figure 2(d), contains a harmful cycle. This is the price we pay for a polynomial acyclicity test: a more detailed acyclicity check could enumerate all canonical models, but this would often require (at least) exponential time. Nevertheless, dependency structures provide us with a sufficient check. For example, assume that we extended $\mathcal{R}_v$ with the HT-rule (11). The corresponding dependency structure, shown in 2(e), contains a self-loop in u, which is harmful; this reflects the infinite R-chain in the canonical model shown in Figure 2(b).

$$D(x) \rightarrow \exists R.A(x) \tag{10}$$
$$S(x,y) \wedge C(y) \rightarrow \exists R.A(x) \tag{11}$$

An Import-by-Query Algorithm We next present our import-by-query algorithm that assumes $\mathcal{R}_v \cup \mathcal{A}_v$ to be HT-safe and acyclic w.r.t. Γ. We modify the standard hypertableau algorithm in three ways. First, we introduce several cut rules that nondeterministically guess all "relevant" assertions involving the symbols in Γ. Second, we use the Ω^{a}-rule to check whether the guesses related to Γ are indeed consistent with $\mathcal{T}_h$. Third, we use a relaxed blocking condition.

Definition 9. *The $\mathcal{ALCHIQ}$ Ω^{a}-algorithm takes an oracle $\Omega^{\mathsf{a}}_{\mathcal{T}_h,\Gamma}$ for $\mathcal{T}_h$ an $\mathcal{ALCHIQ}$ TBox, a normalized ABox $\mathcal{A}_v$,*

Table 4: Additional Rules

A-cut	If 1.	s is not indirectly blocked in $\mathcal{A}$ and
	2.	$\{A(s), \neg A(s)\} \cap \mathcal{A} = \emptyset$ with $A \in \Gamma$
	then	$\mathcal{A}_1 := \mathcal{A} \cup \{A(s)\}$ and $\mathcal{A}_2 := \mathcal{A} \cup \{\neg A(s)\}$.
R-cut	If 1.	s and t are not indirectly blocked in $\mathcal{A}$,
	2.	$R'(s,t) \in \mathcal{A}$ with $R' \in \Gamma$, and
	3.	$\{R(s,t), \neg R(s,t)\} \cap \mathcal{A} = \emptyset$ with $R \in \Gamma$
	then	$\mathcal{A}_1 := \mathcal{A} \cup \{R(s,t)\}$ and $\mathcal{A}_2 := \mathcal{A} \cup \{\neg R(s,t)\}$.
R^--cut	If 1.	s and t are not indirectly blocked in $\mathcal{A}$,
	2.	$R(s,t) \in \mathcal{A}$ with $R \in \Gamma$, and
	3.	$\{R(t,s), \neg R(t,s)\} \cap \mathcal{A} = \emptyset$,
	then	$\mathcal{A}_1 := \mathcal{A} \cup \{R(t,s)\}$ and $\mathcal{A}_2 := \mathcal{A} \cup \{\neg R(t,s)\}$.
$\approx$-cut	If 1.	s, s_1, s_2 are not indirectly blocked in $\mathcal{A}$ and
		atomic roles $R, R' \in \Gamma$ exist such that
	1.1	$\{R(s,s_1), R'(s,s_2)\} \subseteq \mathcal{A}$ or
	1.2	$\{R(s,s_1), R'(s_2,s)\} \subseteq \mathcal{A}$ or
	1.3	$\{R(s_1,s), R'(s_2,s)\} \subseteq \mathcal{A}$
	then	$\mathcal{A}_1 := \mathcal{A} \cup \{s_1 \approx s_2\}$ and $\mathcal{A}_2 := \mathcal{A} \cup \{s_1 \not\approx s_2\}$.
Ω^{a}-rule	If	$\Omega^{\mathsf{a}}_{\mathcal{T}_h,\Gamma}(\mathcal{A}\vert_\Gamma) = \mathsf{f}$ and $\bot \notin \mathcal{A}$
	then	$\mathcal{A}_1 := \mathcal{A} \cup \{\bot\}$.

and a set of HT-rules $\mathcal{R}_v$ such that $\mathcal{R}_v \cup \mathcal{A}_v$ is acyclic w.r.t. Γ and $\mathcal{R}_v$ is HT-safe w.r.t. Γ. The algorithm is obtained by modifying Definition 1 as given next.

Blocking. *An unnamed individual s is* blocking-relevant *in $\mathcal{A}$ if, for s' the predecessor of s, we have*

$$\mathcal{L}_{\mathcal{A}}(s,s') \cap \Gamma = \mathcal{L}_{\mathcal{A}}(s',s) \cap \Gamma = \emptyset.$$

Then, each individual s in an ABox $\mathcal{A}$ is assigned a blocking status in the same way as in Definition 1, with the difference that s is directly blocked by t if, in addition to the conditions in Definition 1, both s and t are blocking-relevant.

Derivation Rules. *The derivation rules are given in Tables 1 and 4. By $\mathcal{A}|_\Gamma$ we denote the ABox obtained from $\mathcal{A}$ by removing each assertion containing an indirectly blocked individual and each assertion α such that* $\mathsf{sig}(\alpha) \not\subseteq \Gamma$.

Our algorithm is indeed an import-by-query algorithm.

Theorem 7. *The $\mathcal{ALCHIQ}$ Ω^{a}-algorithm is an import-by-query algorithm and it can be implemented such that it runs in* N2ExpTime *in the size of $\mathcal{R}_v \cup \mathcal{A}_v$ with an exponential number of calls to $\Omega^{\mathsf{a}}_{\mathcal{T}_h,\Gamma}$.*

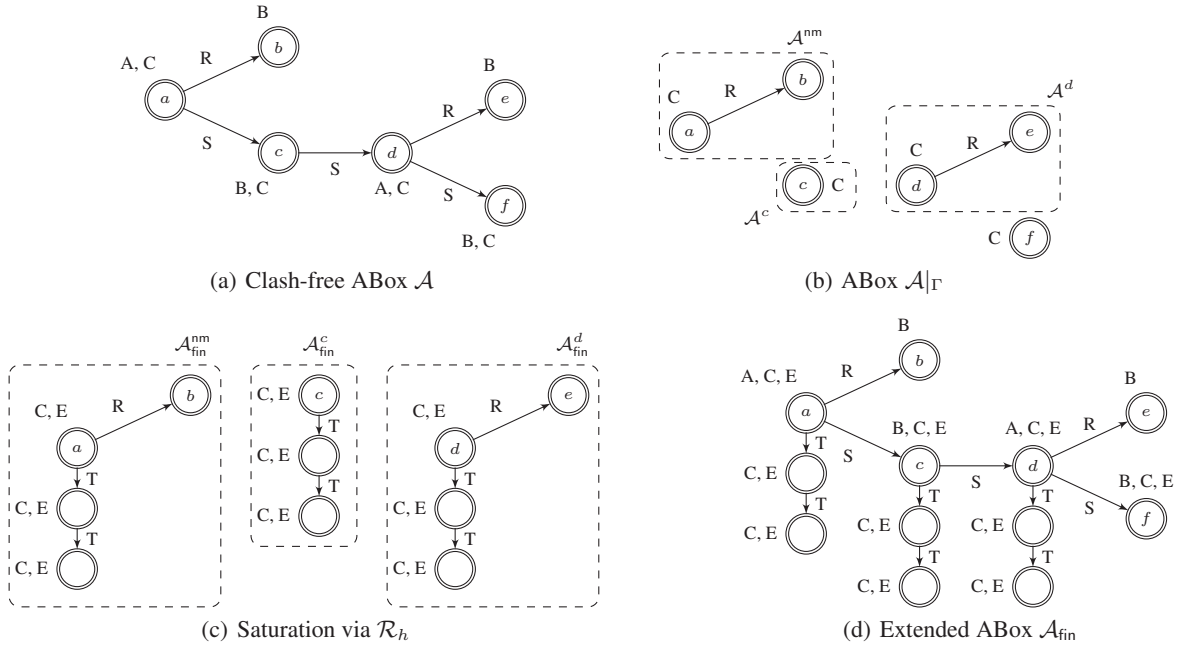
We explain next the intuitions behind the proofs. All derivation rules are clearly sound. Furthermore, due to acyclicity, the chains of assertions involving roles from Γ are bounded in length, which enables blocking and ensures termination. We next sketch the completeness argument. Let $\mathcal{A}$ be a clash-free ABox labeling the leaf of a derivation for $\mathcal{R}_v$, $\mathcal{A}_v$, and $\Omega^{\mathsf{a}}_{\mathcal{T}_h,\Gamma}$, and let $\mathcal{R}_h$ be the set of HT-rules corresponding to $\mathcal{T}_h$. To prove that $\mathcal{R}_v \cup \mathcal{A}_v \cup \mathcal{T}_h$ is satisfiable, we extend $\mathcal{A}$ to a clash-free ABox $\mathcal{A}_{\mathsf{fin}}$ such that no derivation rule of the standard hypertableau algorithm is applicable to $\mathcal{R}_v \cup \mathcal{R}_h$ and $\mathcal{A}_{\mathsf{fin}}$; thus, $\mathcal{R}_v \cup \mathcal{R}_h \cup \mathcal{A}_{\mathsf{fin}}$ is satisfiable, and so is $\mathcal{R}_v \cup \mathcal{T}_h \cup \mathcal{A}_{\mathsf{fin}}$. The construction of $\mathcal{A}_{\mathsf{fin}}$ proceeds as follows:

1. We take the projection $\mathcal{A}|_\Gamma$ of $\mathcal{A}$ to Γ and split it up. In particular, we define $\mathcal{A}^{\mathsf{nm}}$ to contain all assertions of $\mathcal{A}|_\Gamma$ involving individuals reachable from a named individual; furthermore, for each nonblocked blocking-relevant individual t, we define $\mathcal{A}^t$ to contain all assertions of $\mathcal{A}|_\Gamma$ involving individuals reachable from t.
2. We apply the standard hypertableau algorithm to $\mathcal{R}_h$ and $\mathcal{A}^{\mathsf{nm}}$, and $\mathcal{R}_h$ and each $\mathcal{A}^t$; let $\mathcal{A}^{\mathsf{nm}}_{\mathsf{fin}}$ and $\mathcal{A}^t_{\mathsf{fin}}$ be clash-free ABoxes labeling leaves of the respective derivations. The Ω^{a}-rule is not applicable to $\mathcal{A}$ so such ABoxes exist.
3. We define $\mathcal{A}_{\mathsf{fin}}$ as the union of $\mathcal{A}$, $\mathcal{A}^{\mathsf{nm}}_{\mathsf{fin}}$, and all $\mathcal{A}^t_{\mathsf{fin}}$, plus all assertions $C(s)$ such that s is blocked in $\mathcal{A}$ by the blocker s', $C(s') \in \mathcal{A}^{s'}_{\mathsf{fin}}$, and $\mathsf{sig}(C) \subseteq \mathsf{sig}(\mathcal{R}_h)$.

Call the individuals from $\mathcal{A}$ *old*, and the individuals introduced in the second step *new*; we then observe the following. (1) Due to the cut rules, any assertion derivable by the hypertableau calculus is present in $\mathcal{A}$ positively or negatively, so the second step above cannot derive new assertions involving only old individuals. (2) ABoxes $\mathcal{A}^{\mathsf{nm}}$ and $\mathcal{A}^t$ are disjoint, so the HT-rules from $\mathcal{R}_h$ can be applied in $\mathcal{A}_{\mathsf{fin}}$ only to subsets that correspond to $\mathcal{A}^{\mathsf{nm}}$ and $\mathcal{A}^t$. (3) Due to (1), no HT-rule from $\mathcal{R}_v$ can become applicable to assertions involving only old individuals. (4) Due to HT-safety, no HT-rule from $\mathcal{R}_v$ can become applicable to an assertion of $\mathcal{A}_{\mathsf{fin}}$ that involves a new individual. (5) Due to (1) and the third step from the construction above, if an individual s is blocked in $\mathcal{A}$, $\mathcal{A}^{\mathsf{nm}}_{\mathsf{fin}}$, or $\mathcal{A}^t_{\mathsf{fin}}$, then s is blocked in $\mathcal{A}_{\mathsf{fin}}$ as well. Observations (1)–(6) then imply that no derivation rule of the standard hypertableau algorithm is applicable to $\mathcal{R}_v \cup \mathcal{R}_h$ and $\mathcal{A}_{\mathsf{fin}}$, which proves completeness.

Consider our running example in which $\mathcal{R}_v$ contains the HT-rules (4)–(9), $\mathcal{A}_v = \{A(a)\}$, $\Gamma = \{C, R, T\}$, and $\mathcal{T}_h = \{\exists R.\top \sqsubseteq C,\ \ C \sqsubseteq \exists T.C,\ \ C \sqsubseteq E\}$. The $\mathcal{ALCHIQ}$ Ω^{a}-algorithm produces a derivation in which a leaf is labeled with the ABox $\mathcal{A}$ shown in Figure 3(a); for simplicity, we do not show the negative assertions. Individual f is directly blocked by c in $\mathcal{A}$, and assertions $C(a)$ and $C(d)$ are due to the application of the A-cut rule. To construct $\mathcal{A}_{\mathsf{fin}}$, the assertions containing a symbol not in Γ are removed, resulting in the ABox $\mathcal{A}|_\Gamma$ shown in Figure 3(b). This ABox is then split into connected components $\mathcal{A}^{\mathsf{nm}}$, $\mathcal{A}^c$, and $\mathcal{A}^d$; note that c and d are the only nonblocked blocking-relevant individuals. Next, $\mathcal{A}^{\mathsf{nm}}$, $\mathcal{A}^c$, and $\mathcal{A}^d$ are completed w.r.t. $\mathcal{R}_h$ using the standard hypertableau algorithm; figure 3(c) shows the resulting ABoxes $\mathcal{A}^{\mathsf{nm}}_{\mathsf{fin}}$, $\mathcal{A}^c_{\mathsf{fin}}$, and $\mathcal{A}^d_{\mathsf{fin}}$. Note that the guesses of $C(a)$ and $C(d)$ in $\mathcal{A}$ are consistent with the axiom $\exists R.\top \sqsubseteq C$ from $\mathcal{T}_h$. Finally, $\mathcal{A}_{\mathsf{fin}}$ is obtained by taking the union of $\mathcal{A}$, $\mathcal{A}^{\mathsf{nm}}_{\mathsf{fin}}$, $\mathcal{A}^c_{\mathsf{fin}}$, and $\mathcal{A}^d_{\mathsf{fin}}$, and adding $E(f)$; the latter is because f is blocked by c and $E(c) \in \mathcal{A}^c_{\mathsf{fin}}$. The result is shown in Figure 3(d), and clearly no derivation rule of the standard hypertableau algorithm is applicable to $\mathcal{A}_{\mathsf{fin}}$.

We finish this section with a note that, similarly as in our previous work (Cuenca Grau, Motik, and Kazakov 2009, Section 5.2), if $\mathcal{T}_h$ is Horn, we can use an entailment oracle instead of a satisfiability oracle, dispense with the nondeterministic cut rules, and use an oracle query rule that deterministically completes an ABox with the missing assertions. Such an algorithm issues oracle queries "on demand," so it is "goal oriented" and thus better suited to implementation.



(a) Clash-free ABox $\mathcal{A}$

(b) ABox $\mathcal{A}|_{\Gamma}$

(c) Saturation via $\mathcal{R}_h$

(d) Extended ABox $\mathcal{A}_{\mathsf{fin}}$

Figure 3: Completeness of the $\mathcal{ALCHIQ}$ Ω^{a}-algorithm

Conclusion

In this paper, we have extended the import-by-query framework from our previous work and have lifted many of its original restrictions. Our results provide a flexible way for ontology designers to ensure selective access to their ontologies. Our framework thus provides key theoretical insights into the issues surrounding ontology privacy. Furthermore, we believe our algorithms to be practicable when applied to Horn ontologies; thus, our results provide a starting point for the development of practical import-by-query systems.

The problem of import-by-query is novel, and we see many open questions. From a theoretical point of view, it would be interesting to explore the formal connection between import-by-query and interpolation. Furthermore, a problem that is relevant to both theory and practice is to allow the hidden ontology to selectively export data and not just schema statements. Finally, the framework should be implemented and tested in practice.

References

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. F., eds. 2007. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2nd edition.

Baader, F.; Brandt, S.; and Lutz, C. 2005. Pushing the $\mathcal{EL}$ Envelope. In *Proc. IJCAI*, 364–369.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2004. What to Ask to a Peer: Ontology-based Query Reformulation. In *Proc. KR*, 469–478.

Cuenca Grau, B.; Horrocks, I.; Kazakov, Y.; and Sattler, U. 2008. Modular Reuse of Ontologies: Theory and Practice. *JAIR* 31:273–318.

Cuenca Grau, B.; Motik, B.; and Kazakov, Y. 2009. Import-by-Query: Ontology Reasoning under Access Limitations. In *Proc. IJCAI*, 727–733. AAAI Press.

Horrocks, I.; Kutz, O.; and Sattler, U. 2005. The irresistible $\mathcal{SRIQ}$. In *Proc. of OWLED*.

Konev, B.; Walter, D.; and Wolter, F. 2009. Forgetting and uniform interpolation in large-scale description logic terminologies. In *Proc. IJCAI*. AAAI Press.

Kutz, O.; Horrocks, I.; and Sattler, U. 2006. The Even More Irresistible $\mathcal{SROIQ}$. In *Proc. KR*, 68–78.

Lutz, C., and Wolter, F. 2009. Deciding inseparability and conservative extensions in the description logic $\mathcal{EL}$. *J. of Symbolic Computation* 45:151–286.

Lutz, C.; Walther, D.; and Wolter, F. 2007. Conservative Extensions in Expressive Description Logics. In *Proc. IJCAI*, 453–458.

Motik, B., and Horrocks, I. 2008. Individual Reuse in Description Logic Reasoning. In *Proc. IJCAR*, 242–258.

Motik, B.; Shearer, R.; and Horrocks, I. 2009. Hypertableau Reasoning for Description Logics. *JAIR*.

Stuckenschmidt, H.; Parent, C.; and Spaccapietra, S., eds. 2009. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *LNCS*. Springer.