Import-by-Query: Ontology Reasoning under Access Limitations

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

Abstract
To enable ontology reuse, the Web Ontology Language (OWL) allows an ontology $K_v$ to import an ontology $K_h$. To reason with such a $K_v$, a reasoner needs physical access to the axioms of $K_h$. For copyright and/or privacy reasons, however, the authors of $K_h$ might not want to publish the axioms of $K_h$; instead, they might prefer to provide an oracle that can answer a (limited) set of queries over $K_h$, thus allowing $K_v$ to import $K_h$ “by query.” In this paper, we study import-by-query algorithms, which can answer questions about $K_v \cup K_h$ by accessing only $K_v$ and the oracle. We show that no such algorithm exists in general, and present restrictions under which importing by query becomes feasible.

1 Introduction

The Web Ontology Language (OWL) and its revision OWL 2 are widely used ontology languages whose formal underpinnings are provided by description logics (DLs) [Baader et al., 2007]—a family of knowledge representation formalisms with well-understood formal properties. Ontologies are used, for example, in several countries to describe electronic patient records (EPR). In such a system, patients’ data typically involves ontological descriptions of human anatomy, medical conditions, drugs and treatments, and so on. The latter domains have already been described in well-established reference ontologies such SNOMED-CT and GALEN. In order to save resources, increase interoperability between applications, and rely on experts’ knowledge, an EPR application should preferably reuse these reference ontologies.

For example, assume that some reference ontology $K_h$ describes concepts such as the “ventricular septum defect.” An EPR application might reuse the concepts and roles from $K_h$ to define its own ontology $K_v$ of concepts such as “patients having a ventricular septum defect.” It is generally accepted that ontology reuse should be modular—that is, the axioms of $K_v$ should not affect the meaning of the symbols reused from $K_h$ [Lutz et al., 2007; Cuenca Grau et al., 2008].

To enable reuse, OWL allows $K_v$ to import $K_h$. OWL reasoners deal with imports by internally merging the axioms of the two ontologies; thus, to process $K_v \cup K_h$, an EPR application would require physical access to the axioms of $K_h$.

The vendor of $K_h$, however, might be reluctant to distribute the axioms of $K_h$, as doing this might allow the competitors to plagiarize $K_h$. Moreover, $K_h$ might contain information that is sensitive from a privacy point of view and should not be shared. Finally, the vendor of $K_h$ might impose different costs for reusing parts of $K_h$. To reflect this situation, we say that $K_h$ is hidden and, by analogy, $K_v$ is visible.

This problem could be addressed if $K_h$ were made accessible via an oracle (i.e., a limited query interface), thus allowing $K_v$ to import $K_h$ “by query.” In this paper, we study import-by-query algorithms, which can answer questions about $K_v \cup K_h$ by accessing only $K_v$ and the oracle. We focus on schema reasoning problems, such as concept subsumption and satisfiability, which are useful during ontology development; this is in contrast to the information integration [Lenzerini, 2002] and peer-to-peer [Calvanese et al., 2004] scenarios, which focus on the reuse of data.

We proceed as follows. In Section 3 we formalize the import-by-query problem and fix the appropriate query language. Then, in Section 4 we show that no import-by-query algorithm exists in general even if $K_v$ and $K_h$ are expressed in the light-weight description logic $\mathcal{EL}$ [Baader et al., 2005]. In Section 5, we present such an algorithm for the case when $K_v$ reuses only atomic concepts from $K_h$, and this is done in a modular way. Under certain assumptions, our algorithm is worst-case optimal; however, it is unlikely to be suitable for practice. Therefore, for the case when $K_h$ is expressed in a Horn DL [Hustadt et al., 2005], we present a practical algorithm that extends the state-of-the-art tableau algorithms [Kutz et al., 2006]. Finally, in Section 6 we extend our results to the case when $K_v$ also reuses roles from $K_h$, but this is done in a syntactically restricted way. Our results may also increase the performance of reasoning: if $K_v$ is non-Horn but $K_h$ is, then $K_v \cup K_h$ can be reasoned with by applying a general-purpose tableau algorithm only to $K_v$ and using a more efficient algorithm for $K_h$.

2 Preliminaries
The formal underpinnings of OWL 2 are provided by the DL $\mathcal{SROIQ}$ [Kutz et al., 2006]. The syntax of $\mathcal{SROIQ}$ is defined w.r.t. a signature $\Sigma$, which is the union of disjoint countable sets of atomic concepts, atomic roles, and individuals. A role is either an atomic role or an inverse role $R^-$ for $R$ an atomic role. For $R$ and $R_i$ roles, a role inclusion axiom has
the form \( R_1 \ldots R_n \subseteq R \), and a role disjointness axiom has the form \( \text{Dis}(R_1, R_2) \). The set of concepts is the smallest set containing \( \top, \bot, \neg C, C_1 \sqcap C_2, \exists R.C, \forall R.Self \), and \( \geq n R.C \), for \( A \) an atomic concept, \( a \) an individual, \( C, C_1 \), and \( C_2 \) concepts, \( R \) a role, and \( n \) a nonnegative integer. Concepts of the form \{\} are called nominals. Furthermore, \( \bot \) is an abbreviation for \( \neg \top, C_1 \sqcap C_2, \forall R.C, \forall R.Self \), and \( \geq n \) \( R.C \) for \( A \) an atomic concept, \( a \) an individual, \( C_1 \), \( C_2 \), and \( C_3 \) concepts, \( R \) a role, and \( n \) a nonnegative integer. A concept inclusion axiom has the form \( C_1 \subseteq C_2 \) for \( C_1 \) and \( C_2 \) concepts, and a concept equivalence \( C_1 \equiv C_2 \) is an abbreviation for \( C_1 \subseteq C_2 \) and \( C_2 \subseteq C_1 \). A TBox \( T \) is a finite set of concept inclusion, role inclusion, and role disjointness axioms. An assertion has the form \( C(a), R(a,b), \) or \( b \neq b \), for \( C \) a concept, \( R \) a role, and \( a \) and \( b \) individuals. An ABox \( A \) is a finite set of assertions. A SROIQ knowledge base is a pair \( K = (T, A) \) where \( T \) is a TBox and \( A \) is an ABox. By a suitable syntactic test, certain roles in \( K \) can be identified as being simple. To ensure decidability of reasoning, the role axioms in \( T \) must satisfy a syntactic restriction which we omit for brevity, and simple roles must not occur in \( \geq n R.C \), \( \exists R.Self \), and role disjointness axioms. The definition of SROIQ by [Kutz et al., 2006] provides other constructs, all of which are expressible by the ones presented above.

A interpretation \( I = (\Delta^I, \cdot^I) \) consists of a nonempty domain set \( \Delta^I \) and a function \( \cdot^I \) that assigns an object \( a^I \in \Delta^I \) to each individual \( a \) and a set \( A^I \subseteq \Delta^I \) to each atomic concept \( A \), and a relation \( R^I \subseteq \Delta^I \times \Delta^I \) to each atomic role \( R \). Table 1 defines the extension of \( \cdot^I \) to roles and concepts, and the satisfaction of axioms in \( I \). An interpretation \( I \) is a model of \( K \), written \( I \models K \), if \( I \) satisfies all axioms in \( K \); if such \( I \) exists, then \( K \) is satisfiable. A concept \( C \) is satisfiable w.r.t. \( K \) if a model \( I \) of \( K \) exists such that \( C^I \neq \emptyset \). A nonempty set of interpretations \( S \) is compatible if for each \( I_1, I_2 \subseteq S \) we have \( \Delta^I_1 \subseteq \Delta^I_2 \) and \( a^I_1 = a^I_2 \) for each individual \( a \); the intersection of such \( S \) is defined in the obvious way.

SROIQ is obtained from SROIQ by disallowing nominals. ELC [Baader et al., 2005] supports only concepts of the form \( \top, \bot, A, C_1 \sqcap C_2 \), and \( \exists R.C \) for \( A \) an atomic concept and \( R \) an atomic role, and it supports no axioms about roles. Significant effort has been devoted to the development of DL languages with good computational properties, such as EL, DL-Lite [Calvanese et al., 2007], and Horn-SROIQ [Hustadt et al., 2005]. Each knowledge base \( K \) expressed in one of these languages is Horn in the sense that the intersection of every compatible set of models of \( K \) is also a model of \( K \).

For \( \alpha \) a concept, a role, an axiom, or a knowledge base, \( \text{sig}(\alpha) \) is the signature of \( \alpha \)—that is, the set of atomic concepts, atomic roles, and individuals occurring in \( \alpha \). A position \( p \) is a finite sequence of integers. The empty position is denoted with \( \epsilon \). If a position \( p_1 \) is a proper prefix of a position \( p_2 \), then \( p_1 \) is above \( p_2 \), and \( p_2 \) is below \( p_1 \). The subterm \( \alpha\|_p \) of a concept or axiom \( \alpha \) at a position \( p \) is defined as follows: \( \alpha\|_\epsilon = \alpha \); \( (C_1 \sqcap C_2)\|_p = C_1\|_p \sqcap C_2\|_p \) for \( \sqcap \in \{\sqcap, \sqcup\} \) and \( i \in \{1,2\} \); and \( \alpha\|_p = C\|_p \) for the form \( \neg C, \exists R.C, \forall R.C, \forall R.Self \), and \( C(a) \) for the form \( C \). The concept closure \( \text{cls}(\alpha) \) of \( K = (T, A) \) is the smallest set that contains all subterms of \( \neg C \sqcup D \) for each \( C \subseteq D \in T \) and of \( C \) for each \( C(a) \in A \).

### 3 Importing Ontologies by Query

To illustrate the notion of import-by-query, Table 2 shows a reference knowledge base \( K_h \) whose axioms are to be kept hidden, but that is reused in a visible knowledge base \( K_v \). The hidden knowledge base \( K_h \) provides concepts describing organs such as Heart, and medical conditions such as CHD (congenital heart defect), VSD (ventricular septum defect), and AS (aortic stenosis). Furthermore, the role cond relates organs to medical conditions and is used to define concepts such as CHD\_Heart (a heart with a congenital heart disorder) and VSD\_Heart (a heart with a ventricular septal defect). The shared symbols of \( K_h \) are written in bold font. In addition to these, \( K_h \) might contain nonshared symbols; however, for the sake of brevity, we do not show any axioms involving such symbols. The visible knowledge base \( K_v \) provides the concept Pat representing patients, and it defines various types of patients by relating the organs from \( K_h \) with the patients using the hasOrgan role. In addition, \( K_v \) extends the list of defects in \( K_h \) by EA (Ebstein’s anomaly). The symbols private to \( K_v \) are written in italic font.

When reusing ontologies, it is commonly accepted that \( K_v \) should not affect the meaning of the symbols reused from \( K_h \)—that is, \( K_v \sqcup K_h \models \alpha \) should imply \( K_h \models \alpha \) for each
axiom $\alpha$ containing only the reused symbols [Lutz et al., 2007; Cuenca Grau et al., 2008]. This is guaranteed if the TBox $T_v$ of $K_v$ is local w.r.t. the set $\Gamma$ of concepts and roles imported from $K_h$—that is, if $I \models T_v$ for each interpretation $I$ in which, for each concept or role $X \not\in \Gamma$, we have $X^I = \emptyset$. For example, $\delta_1$ is local w.r.t. $\{\text{CHD-Heart}\}$ because $\delta_1$ is satisfied in any interpretation that interprets the nonshared symbols as $\emptyset$. [Cuenca Grau et al., 2008] have shown how to check this condition using a DL reasoner.

To formalize the notion of import-by-query, we introduce the notion of a $\Gamma$-oracle, which is responsible for advertising the shared signature $\Gamma$ of $K_h$ and answering satisfiability of (not necessarily atomic) concepts w.r.t. $K_h$. Concept satisfiability is available in all DL reasoners known to us, so it provides us with a natural query language for $\Gamma$-oracles; we leave the investigation of richer query languages to future work.

**Definition 1.** Let $K$ be a KB and $\Gamma \subseteq \text{sig}(K)$ a signature. The $\Gamma$-oracle for $K$ is the function $\Omega_K$ defined for each concept $C$ (in the same DL as $K$) with $\text{sig}(C) \subseteq \Gamma$ such that $\Omega_K(C) = t$ if $C$ is satisfiable w.r.t. $K$, and $\Omega_K(C) = \text{t otherwise}.$ An import-by-query algorithm checks whether $K_v \cup K_h$ is satisfiable; other relevant reasoning problems, such as concept subsumption, can be solved using the well-known transformations.

The notion of an algorithm in the following definition can be made precise using a formal computation model such as Turing machines in the obvious way.

**Definition 2.** An import-by-query algorithm takes a $\Gamma$-oracle $\Omega_K$, and a KB $K_v$ with $\text{sig}(K_v) \cap \text{sig}(K_h) \subseteq \Gamma$ as input, and it terminates after a finite number of computation steps returning $t$ if $K_v \cup K_h$ is satisfiable.

**4 The Limits of Import-by-Query Reasoning**

We next show that no import-by-query algorithm exists even for a light-weight DL such as $\mathcal{EL}$.

**Theorem 1.** No import-by-query algorithm exists if $K_v$ and $K_h$ are in $\mathcal{EL}$, $\Gamma$ is allowed to contain at least one atomic role, and the TBox of $K_v$ is local in $\Gamma$.

**Proof.** Consider an application of an import-by-query algorithm to $K_v$ given in (1) and $\Gamma = \{R\}$. Clearly, the TBox of $K_h$ is local in $\Gamma$. Since the algorithm terminates on all inputs, the number of questions posed to any $\Gamma$-oracle is bounded by some integer $m$ and, consequently, the quantifier depth of each concept $C$ passed to the $\Gamma$-oracle is bounded by an integer $n$, where both $m$ and $n$ depend only on $\Gamma$ and $K_v$. Let $K_1$ and $K_2$ be as in (2) and (3), respectively.

\[
K_v = \{ A(a), A \sqsubseteq \exists R.A \} \\
K_1 = \emptyset \\
K_2 = \{ \exists R.. \exists R. T \sqsubseteq \bot \} \quad n + 1 \text{ times}
\]

For each $\mathcal{EL}$ concept $C$ of quantifier depth at most $n$ with $\text{sig}(C) \subseteq \Gamma$, we have $K_v \models C \sqsubseteq \bot$ iff $K_2 \models C \sqsubseteq \bot$, so $\Omega_{K_1}(C) = \Omega_{K_2}(C)$. Thus, when applied to $K_v$ and $\Omega_{K_1}$, the algorithm returns the same value as when it is applied to $K_v$ and $\Omega_{K_2}$. Since $K_v \cup K_1$ is satisfiable but $K_v \cup K_2$ is not, the algorithm does not satisfy Definition 2.

**5 Importing Atomic Concepts**

The proof of Theorem 1 relies on the fact that $K_v$ reuses a role from $K_h$. We now present an import-by-query algorithm for the case when no role is reused. In our example, this allows one to express axioms $\delta_1$, $\delta_2$, and $\delta_3$, which, together with $K_h$, allow us to conclude $\text{VSD-Pat} \subseteq \text{CHD-Pat}$.

**5.1 Interfacing Models Point-Wise**

The following definition identifies valid inputs for our algorithm. In particular, we allow $K_v$ to be any OWL 2 ontology that reuses the symbols of $K_h$ in a local way; however, we disallow the usage of nominals in $K_h$ for technical reasons.

**Definition 3.** Let $K_v = \langle T_v, A_v \rangle$ and $K_h = \langle T_h, A_h \rangle$ be KBs such that $\Gamma = \text{sig}(K_v) \cap \text{sig}(K_h)$ contains only atomic concepts. Then, $K_v$ is safe for import-by-query into $K_v$ if $K_v$ is in $\mathcal{SROIQ}$, $K_h$ is in $\mathcal{SRIQ}$, and $T_v$ is local w.r.t. $\Gamma$.

Our core observation is that a model of $K_v \cup K_h$ can be obtained by taking a model $I$ of $K_v$, and extending it at each point $x \in \Delta^I$ with a fresh model $J_x$ of $K_h$ that contains a point $y \in \Delta^{J_x}$ such that $x$ and $y$ coincide on the interpretation of the concepts in $\Gamma$. This is a consequence of the fact that (i) $K_v$ uses the concepts from $\Gamma$ in a local way, and (ii) $K_h$ does not contain nominals, so the union of all models $J_x$ is also a model of $K_h$. To formalize this idea, we use the following notion: for $S = \{D_1, \ldots, D_n\}$ a nonempty finite set of concepts, a selection w.r.t. $S$ is a concept of the form $L_1 \sqcap \cdots \sqcap L_n$ where each $L_i$ is either $D_i$ or $\neg D_i$; furthermore, $T$ is the only selection w.r.t. $S = \emptyset$.

**Lemma 1.** Let $K_h$ be safe for import-by-query into $K_v$, and let $\Gamma = \text{sig}(K_v) \cap \text{sig}(K_h)$. Then, $K_v$ is safe for import-by-query if a model of $K_v$ exists such that $\Omega_{K_v}(C) = t$ for each selection $C$ w.r.t. $\Gamma$ such that $C^I \neq \emptyset$.

**Proof.** ($\Rightarrow$) If $I$ is a model of $K_v \cup K_h$, then clearly $I \models K_v$, and $\Omega_{K_v}(C) = t$ for each selection $C$ w.r.t. $\Gamma$ with $C^I \neq \emptyset$.

($\Leftarrow$) Let $I = \langle \Delta^I, \cdot, \cdot \rangle$ be a model of $K_v$, and consider each $x \in \Delta^I$ and the selection $C$ w.r.t. $\Gamma$ such that $x \in C^I$. Since $\Omega_{K_v}(C) = t$, an interpretation $J_x = \langle \Delta^{J_x}, \cdot, \cdot \rangle$ exists such that $J_x \models K_h$ and $y \in \Delta^{J_x}$ for some $y \in \Delta^{J_x}$. W.l.o.g. we assume that $y = x$: $\Delta^{J_x} \sqcap \Delta^I = \{x\}$; $\Delta^{J_x} \sqcap \Delta^{J_x} = \emptyset$ for each $x_1, x_2 \in \Delta^I$ with $x_1 \neq x_2$; and $J_x = \emptyset$ for each $x \in \text{sig}(K_v) \setminus \Gamma$. Let $M = \langle \Delta^M, \cdot \rangle$ be such that

\[
\Delta^M = \bigcup_{x \in \Delta^I} \Delta^{J_x}, \\
X^M = \bigcup_{x \in \Delta^I} X^{J_x} \\
a^M = a^{J_x} \text{ for each individual } a \text{ and some (arbitrarily chosen) interpretation } J_x.
\]

$\mathcal{SRIQ}$ does not allow for nominals, so it is invariant under disjoint unions—that is, the union of any number of disjoint models of $K_h$ is also a model of $K_h$ [Baader et al., 2002]. Thus, $M \models K_h$. Furthermore, since $T_v$ is local in $\Gamma$, we have $M \models T_v \cup K_h$. Finally, let $N = \langle \Delta^N, \cdot \rangle$ be an interpretation defined by $\Delta^N = \Delta^M$ and

\[
X^N = \begin{cases}
\{X^I\} \text{ for each } X \in \text{sig}(K_v) \setminus \Gamma \\
X^M \text{ for each } X \in \text{sig}(K_h)
\end{cases}
\]
Algorithm 1 Import-by-Query Algorithm

Algorithm: ibq($K_v$, $\Omega_k_v, S$)
Inputs: a knowledge base $K_v$, a $\Gamma$-oracle $\Omega_k_v$, and a set of concepts $S$ over the signature $\Gamma$

1. Compute the set $N$ of all axioms of the form $C \subseteq \perp$ such that $C$ is a selection w.r.t. $S$ with $\Omega_k_v(C) = f$.
2. Return $t$ iff the $\mathcal{SROIQ}$ knowledge base $K_v \cup N$ is satisfiable.

$N$ and $M$ have the same domains and they coincide on the interpretation of the symbols in $\text{sig}(K_v)$, so $N \models K_v$. To show that $N \models K_v$, we first prove the following claim ($\ast$): for each $C \in \text{cls}(K_v)$, we have $C^N = C^I \cup (C^M \setminus \Delta^I)$. The proof of ($\ast$) is by induction on the structure of concepts, so consider each $C \in \text{cls}(K_v)$.

If $C$ is an atomic concept with $C \in \Gamma$, then by the definition of $M$ we have $C^I = C^M \setminus \Delta^I$, so $C^I \cup (C^M \setminus \Delta^I) = (C^M \setminus \Delta^I) \cup (C^M \setminus \Delta^I) = C^M$, by the definition of $N$, we have $C^M = C^N$, which implies ($\ast$).

If $C$ is a nomal or an atomic concept with $C \notin \Gamma$, then $C^M = \emptyset$ and $C^M = C^I$, trivially implying ($\ast$).

If $C = \neg D$, then $C^N = (\Delta^I \cup (\Delta^M \setminus \Delta^I)) \cup \Delta^N = (\Delta^I \setminus D^N) \cup ((\Delta^M \setminus \Delta^I) \setminus D^N)$. By applying the induction hypothesis, the first disjunct reduces to $\Delta^I \setminus D^I$, and, since, $\Delta^N \setminus D^N = \Delta^I \setminus (D^M \setminus (\Delta^M \setminus \Delta^I)) = (\Delta^I \setminus (D^I \setminus (D^M \setminus (\Delta^M \setminus (\Delta^I))))$, the second one reduces to $\Delta^I \setminus (D^I \setminus (D^M \setminus \Delta^I))$. But then, ($\ast$) holds.

If $C = D_1 \sqcap D_2$, then $C^N = D_1^N \sqcap D_2^N$, which is equal to $(D_1^I \sqcap (D_2^M \setminus \Delta^I)) \sqcup (D_1^I \sqcap (D_2^M \setminus \Delta^I))$ by the induction hypothesis; but $(D_1^I \setminus \Delta^I) \sqcap (D_2^I \setminus \Delta^I) = (D_1^I \setminus \Delta^I) \sqcup (D_2^I \setminus \Delta^I) = (D_1^I \setminus \Delta^I) \sqcup (D_2^I \setminus \Delta^I) \sqcup (D_1^I \setminus \Delta^I) \sqcup (D_2^I \setminus \Delta^I)$; finally, $(D_1^I \setminus \Delta^I) \sqcap (D_2^I \setminus \Delta^I) = (D_1^I \setminus \Delta^I) \sqcup (D_2^I \setminus \Delta^I) \sqcup (D_1^I \setminus \Delta^I)$.

If $C = \exists R.D$ or $C = \forall R.S$, since $R \notin \text{sig}(K_v)$, we have $R^M = \emptyset$ and $C^M = \emptyset$; furthermore, $R^N = R^I$ and $D^I \subseteq D^N$ by the induction hypothesis, so $C^N = C^I$. This completes the proof of ($\ast$).

Consider now each axiom $\alpha$ in $K_v$. For $\alpha$ a concept inclusion axiom, we assume w.l.o.g. that it is of the form $T \subseteq C$. By ($\ast$), $C^N = C^I \cup (C^M \setminus \Delta^I)$. Since $I \models \alpha$, we have $C^I = \Delta^I$; furthermore, since $T$ is local w.r.t. $\Gamma$, we have $M \models \alpha$, so $C^M = \Delta^M$; thus, $C^N = \Delta^N$, so $N \models \alpha$. For a role assertion, a role inclusion, or a role disjointness axiom, we have $N \models \alpha$ because $N$ coincides with $I$ on the interpretation of all roles from sig$(K_v)$. For $\alpha = C(a)$, we have $a^N \in C^N$ by ($\ast$) and $a \notin \Gamma$. Finally, for $\alpha = a \neq b$, we have $a^N \neq b^N$ because $\{a, b\} \cap \Gamma = \emptyset$. Thus, $N \models K_v$.

Lemma 1 motivates Algorithm 1.

Theorem 2. Let $K_h$ be safe for import-by-query into $K_v$, $\Gamma = \text{sig}(K_v) \sqcup \text{sig}(K_h)$, and $\Omega_k_h$, the $\Gamma$-oracle for $K_h$. Then, ibq($K_v$, $\Omega_k_h$, $\Gamma$) is an import-by-query algorithm, and it can be implemented such that it runs in $\mathcal{NEXP}$ with an exponential number of calls to $\Omega_k_h$.

Proof. That ibq($K_v$, $\Omega_k_h$, $\Gamma$) is an import-by-query algorithm is a direct consequence of Lemma 1. Furthermore, the number of selections w.r.t. $\Gamma$ is exponential in the size of $\Gamma$, so $N$ can be computed by an exponential number of calls to $\Omega_k_h$. Let $\text{ri}(-)$ be the transformation by [Kazakov, 2008] for eliminating role inclusion axioms from $\mathcal{SROIQ}$ KBs. Then, $\text{ri}(K_v)$ is equisatisfiable with and exponentially larger than $K_v$. [Kazakov, 2008]. Furthermore, $N$ contains the same concepts as $\text{ri}(K_h)$ and no role inclusions axioms, so $\text{ri}(K_v \cup N) = \text{ri}(K_v) \cup N = K'$. Thus, $K'$ is equisatisfiable with and exponentially larger than $K_v$. We can check satisfiability of $K'$ by transforming $K'$ polynomially into an equisatisfiable formula $\varphi$ of the two-variable fragment with counting, and deciding the satisfiability of $\varphi$ in $\mathcal{NEXP}$ [Pratt-Hartmanis, 2005]. Clearly, the overall algorithm runs in $\mathcal{NEXP}$ with exponentially many calls to $\Omega_k_h$. \hfill\Box

5.2 Importing Horn Ontologies

Algorithm 1 is unlikely to be suitable for practice because Step 1 is exponential in the size of $\Gamma$. In this section, we present a practical algorithm for the case when $K_v$ is Horn.\footnote{From the infrastructure point of view, the $\Gamma$-oracle for $K_h$ should indicate to clients if $K_h$ is (known to be) Horn.}

This algorithm calls the $\Gamma$-oracle “on demand,” which makes it “more goal-oriented.” The correctness of the algorithm is based on the following observation about Horn KBs.

Proposition 1. Let $K$ be a Horn knowledge base, $C$ a conjunction of atomic concepts, and $A_1, \ldots, A_n$ atomic concepts such that $C \sqcap \neg A_i$ is satisfiable w.r.t. $K$ for each $1 \leq i \leq n$. Then, $C \sqcap \neg A_1 \cap \cdots \cap \neg A_n$ is satisfiable w.r.t. $K$ as well.

Proof. Let $K_i = K \cup \{C(a), \neg A_i(a)\}$ for $1 \leq i \leq n$ and $a$ an individual not occurring in $K$. Let $I_i$ be a model of each $K_i$; w.l.o.g. we assume that the set $S = \{I_i \mid 1 \leq i \leq n\}$ is compatible (e.g., we can select $I_1$ to be Herbrand models of $K_i$). Let $J$ be the intersection of $S$. Since $K$ is Horn, we have $J \models K$. Furthermore, $a^J \in C^I$ and $a^J \notin A_i^I$ for each $1 \leq i \leq n$; therefore, $a^J \in (C \sqcap \neg A_1 \cap \cdots \cap \neg A_n)^I$.

We extend the tableau algorithms used in many state-of-the-art DL reasoners. Our extension, however, is largely independent from the intricacies of these algorithms, so we introduce an abstraction of a tableau algorithm as a tuple $T = (C, R)$ with the following structure.

- $C$ assigns to each ABox $A$ a value from $\{t, f\}$ such that $C(A) = t$ only if $A$ is unsatisfiable. $A$ contains a clash if $C(A) = t$; otherwise, $A$ is clash-free.
- $R$ is a set of derivation rules, where each $\rho \in R$ assigns to each pair $(T, A)$ a set of $n$-tuples of ABoxes (tuples in this set can vary in arity). A rule $\rho$ is applicable to $T$ and $A$ if $\rho(T, A) \neq \emptyset$.

A derivation for $K = (T, A)$ by $T = (C, R)$ is a pair $(\Theta, \sigma)$ where $\Theta$ is a finitely branching tree and $\sigma$ labels each node $v$ of $\Theta$ with an ABox $\sigma(v)$ such that (i) $\sigma(v) = A$ for $v$ the root of $\Theta$; (ii) if $C(\sigma(v)) = t$ or no derivation rule in $R$ is applicable to $(T, \sigma(v))$, then $v$ is a leaf of $\Theta$; (iii) if $C(\sigma(v)) = f$ and a derivation rule in $R$ is applicable to $(T, \sigma(v))$, then $v$ has children $v_1, \ldots, v_n$ such that $(\sigma(v_1), \ldots, \sigma(v_n)) \in \rho(T, \sigma(v))$ for some (arbitrarily chosen) derivation rule $\rho \in R$.

$T$ is terminating if, for each $K$, each derivation for $K$ by $T$ can be constructed using finitely many steps. $T$ is sound.
if, for each model $I$ of each $⟨T, A⟩$, each derivation rule $ρ ∈ R$, and each $⟨A_1, \ldots, A_n⟩ ∈ ρ(T, A)$, an interpretation $I’$ exists such that $X^I = X^{I’}$ for each $X$ such that $A’ = σ(v)$ is clash-free, the value of $M(A’)$ is defined and $M(A’) = X^{I’}$. Furthermore, we assume that $A’$ is clash-free.

We now show how to extend $T$ to an import-by-query algorithm for the case when $K_h$ is Horn.

**Definition 4.** Let $T = (C, R)$ be a sound, complete, and terminating tableau algorithm, $Γ$ a set of atomic concepts, and $Ω_{K_h}$ a $Γ$-oracle. The tableau algorithm $T_{Γ, Ω_{K_h}}$ is obtained by extending $T$ with the ask-rule as follows: $\text{ask}(T, A)$ is defined for each $(T, A)$ as the smallest set such that, for each individual $s$ in $A$, the concept $C$ obtained as the conjunction of all $A_i ∈ Γ$ with $A_i(s) ∈ A$, and each $B ∈ Γ ∪ \{\top\}$ with $Ω_{K_h}(C ∩ ∼B) = f$, we have $(A ∪ \{B(s)\}) ∈ \text{ask}(T, A)$.

Intuitively, the ask-rule deterministically adds $B(s)$ to each ABox that contains assertions $A_1(s), \ldots, A_n(s)$ such that $K_h = A_1 \cap \ldots \cap A_n = B$.

**Theorem 3.** Let $K_h = ⟨T_h, A_h⟩$ be a Horn knowledge base that is safe for import-by-query into $K_v = ⟨T_v, A_v⟩$, let $Γ = \text{sig}(A_v) \cap \text{sig}(K_h)$, let $Ω_{K_h}$ be the $Γ$-oracle for $K_h$, and let $T$ be a sound, complete, and terminating tableau algorithm. Then, $T_{Γ, Ω_{K_h}}$ satisfies the following two claims:

1. if $K_v ∪ K_h$ is satisfiable, then each derivation for $K_v$ by $T_{Γ, Ω_{K_h}}$ contains a branch on which all nodes are labeled with clash-free ABoxes; and
2. if a derivation for $K_v$ by $T_{Γ, Ω_{K_h}}$ contains a leaf labeled with a clash-free ABox, then $K_v ∪ K_h$ is satisfiable.

**Proof.** (Claim 1) Assume that $I$ is a model of $K_v ∪ K_h$, and consider each derivation for $K_v$ by $T_{Γ, Ω_{K_h}}$. We assume w.l.o.g. that the derivation rules of $T_{Γ, Ω_{K_h}}$ do not introduce assertions involving symbols from $\text{sig}(K_h) \setminus Γ$. Consider now the tuple $⟨A_1, \ldots, A_n⟩$ obtained from $A_v$ by an application of a derivation rule of $T_{Γ, Ω_{K_h}}$. If the derivation rule is from $T$, since $T$ is sound, $I$ can be extended to a model $I’$ of some $⟨T_v, A_i⟩$; since this extension does not involve the symbols in $\text{sig}(K_h)$, we have $I’ = K_h$ as well. For the ask-rule, $n = 1$ and $s^I = C^I$, so $Ω_{K_h}(C ∩ ∼B) = f$ implies $s^I ∈ B^I$ and $I = ⟨T_v, A_i⟩ ∪ K_h$. By repeating this claim inductively, we conclude that the derivation contains a branch on which each node is labeled with an ABox $A’$ such that $⟨T_v, A’⟩ ∪ K_h$ is satisfiable; thus, each $A’$ is clash-free.

(Proof of Claim 2) Let $A’$ be a clash-free ABox labeling a leaf of a derivation for $K_v$ by $T_{Γ, Ω_{K_h}}$, and let $M(A’) = (Δ^A, C^A)$. Furthermore, let $C$ be a selection w.r.t. $Γ$ such that $C’ = Ω_{K_h}(C ∩ ∼B) = f$ and let $D$ be the conjunction of all atomic concepts that occur positively in $C$. By $⟨Ω⟩$ and the fact that $C$ is maximal, an individual $s$ in $A$ exists such that $A(s) ∈ A’$ for each $A_i ∈ D$, and $B_i(s) \notin A'$ for each atomic concept $B_j$, $1 ≤ j ≤ n$ that occurs negatively in $C$. Since the ask-rule is not applicable to $A’$, then $D ∩ ∼B_i$ is satisfiable w.r.t. $K_h$ for each $1 ≤ j ≤ n$. Since $K_h$ is Horn, by Proposition 1 we have that $D ∩ ∼B_i$ is satisfiable w.r.t. $K_h$ as well. But then, $K_v ∪ K_h$ is satisfiable by Lemma 1.

Each derivation for $K_v$ by $T_{Γ, Ω_{K_h}}$ is clearly finite. Furthermore, the value of $\text{ask}(T, A)$ can be determined by asking $Ω_{K_h}(C)$ for each selection $C$ w.r.t. $Γ$ occurring in $A$. Therefore, a derivation for $K_v$ by $T_{Γ, Ω_{K_h}}$ can be constructed by a finite number of steps, which provides us with an import-by-query algorithm. Such an algorithm may, in the worst case, make an exponential number of calls to the oracle; however, such calls are made as needed, which makes this algorithm more amenable to implementation than Algorithm 1.

### 6 Importing Atomic Roles

We now extend the results from Section 5 and allow the reuse of roles under the following syntactic restriction.

**Definition 5.** For $Γ$ a set of atomic concepts and roles, we say that a concept is $Γ$-modal if it is of the form $∃R.σ$, $∀R.C$, or $σ ∨ R.C$, for $R ∈ Γ$.

Let $K_v$ and $K_h$ be KBs such that $Γ = \text{sig}(K_v) \cap \text{sig}(K_h)$ contains both concepts and roles. Then, $K_v$ is safe for import-by-query into $K_h$ if, in addition to the conditions from Definition 3, roles from $Γ$ do not occur in role inclusion and disjointness axioms in $K_v$; for each $∃R.σ$ or $∀R.C$, if $R ∈ Γ$ then $R$ is simple in $K_h$; and $\text{sig}(C) ⊆ Γ$ for each $Γ$-modal concept $C$ in $\text{cls}(K_v)$.

For satisfiability of $K_v ∪ K_h$ to be decidable, only simple roles from $K_h$ can occur in certain concepts in $K_v$ [Horrocks et al., 2000]. Thus, the $Γ$-oracle for $K_v$ should also advertise to clients which roles in $Γ$ are simple. This is a syntactic check that is provided by most DL reasoners.

In our example, Definition 5 allows us to express $δ_3$: the role $\text{cond}$ from $K_h$ occurs in a $Γ$-modal concept $∃\text{cond}.AS$, but $AS$ is from $K_h$ as well. This is in contrast to $δ_4$, in which $∃\text{cond}.EA$ contains $EA$ that is not from $K_h$. Note that $δ_1$, $δ_3$, and $K_h$ allow $AS$ to use $\text{cond}$ in $\text{Pat}$, while $\text{cond}$ is disallowed.

By using the appropriate set $S$, Algorithm 1 is an import-by-query algorithm for the case of shared roles as well. In the following theorem, we say that position $p$ in a concept or axiom $α$ is $Γ$-outermost if $α|p$ is a $Γ$-modal concept, and $α|q$ is not a $Γ$-modal concept for each position $q$ above $p$.

**Theorem 4.** Let $K_v$, $Γ$, and $Ω_{K_h}$ be as in Theorem 2 with the difference that $Γ$ can also contain atomic roles, and let

$$S = \{A ∈ Γ | A is an atomic concept\} ∪ \{α|p | α ∈ K_v and p is Γ-outermost in α\}.$$
Then, \( ibq(K_v, \Omega_{h}, S) \) is an import-by-query algorithm, and it can be implemented such that it runs in \( \text{N2ExpTime} \) with an exponential number of calls to \( \Omega_{h} \).

**Proof.** Let \( Q_D \) be a fresh atomic concept uniquely associated with each \( D \in S \). Furthermore, let \( K'_v \) be the knowledge base obtained from \( K_v \) by replacing in each axiom \( \alpha \in K_v \) the concept \( \alpha_{|p} \) with \( Q_{\alpha_{|p}} \) for each \( \Gamma \)-outermost position \( p \) in \( \alpha \). Also, let \( K'_h \) be obtained from \( K_h \) by adding the axiom \( Q_C \equiv C \) for each \( C \in S \). Finally, let \( \Gamma' = \text{sig}(K'_v) \cap \text{sig}(K'_h) \), and let \( \Omega_{h} \) be the \( \Gamma' \)-oracle such that \( \Omega_{h}(C_1) = \Omega_{h}(C_2) \) for each \( C_1 \) and \( C_2 \) where \( C_2 \) is obtained from \( C_1 \) by replacing all \( Q_D \) with \( D \). Since \( K_v \) satisfies the condition from Definition 5 and \( \Gamma' \) contains only atomic concepts, \( K'_v, K'_h, \) and \( \Omega_{h} \) satisfy the preconditions of Theorem 2. Furthermore, it is obvious that \( ibq(K'_v, \Omega_{h}, \Gamma') = ibq(K_v, \Omega_{h}, S) \), so the latter is an import-by-query algorithm. The proof for the algorithm’s running time is the same as in Theorem 2. \( \square \)

When each \( \Gamma \)-modal concept \( C \) occurring in \( K_v \) is Horn [Hustadt et al., 2005], the tableau algorithm from Definition 4 can be extended to the case when \( \Gamma \) contains roles by using the set \( S \) from Theorem 4 instead of \( \Gamma \) in the ask-rule.

Finally, the results from this section can be extended to the case when \( K_v \) contains concepts of the form \( q_R \in \mathbb{R} \) with \( R \in \Gamma \) and \( \text{sig}(D) \not\subseteq \Gamma \), provided that the unfolding of \( D \) in \( K_v \) and results in a concept containing only symbols from \( \Gamma \). In our example, the nonshared symbol \( EA \) in \( \delta_1 \) can be unfolded with its definition in \( \delta_1 \), resulting in \( EA_{Pat} \equiv \text{Pat} \cap \exists \text{hasOrgan},(\text{Heart} \cap \exists \text{cond.CHD}) \); after this preprocessing step, we can use the import-by-query algorithm to conclude \( EA_{Pat} \subseteq \text{CHD}_{Pat} \).

### 7 Related Work

In a peer-to-peer setting, [Calvanese et al., 2004] consider the problem of answering a query \( q \) over two KBs \( K_m \) and \( K_h \) with disjoint signatures and a set \( M \) of mappings of the form \( q_h \rightarrow q, \) by reformulating \( q \) as queries that can be evaluated over \( K_v \) and \( K_h \) in isolation. The query reformulation algorithm accesses only \( K_v \) and \( M \); thus, \( q \) can be answered by means of an oracle for \( K_h \). In such a setting, however, a satisfiable \( K_h \) cannot affect the subsumption of concepts in \( K_v \). Consider the following example:

\[
\begin{align*}
K_h &= \{ B_h \sqsubseteq A_h \} \\
M &= \{ A_h(x) \sim A_v(x) \} \\
K_v &= \{ C_v \sqsubseteq B_v \} \\
B_v(x) \rightarrow B_v(x)
\end{align*}
\]

Now \( K_v \cap K_h \cup M \not\subseteq C_v \sqsubseteq A_v \), since the mappings in \( M \) are unidirectional. Thus, whereas [Calvanese et al., 2004] consider simple schemas (i.e., both \( K_h \) and \( K_v \) must be in DL-Lite) and conjunctive query answering, we focus on rich TBoxes and schema reasoning.

[Baader et al., 2002] study the transfer of decidability results when combining decidable logics. In particular, they show how to integrate algorithms that decide satisfiability of \( K_v \) and \( K_h \) independently into an algorithm that decides satisfiability of \( K_v \cap K_h \), provided that the two KBs do not share roles and do not contain nominals. This situation is similar to the one in Section 5, with the difference that we allow \( K_v \) to contain nominals but require it to be local in \( \Gamma \).

### 8 Conclusion

In this paper, we have studied the problem of importing an ontology without knowing its axioms. We have shown that this problem does not have a general solution. Furthermore, we have identified solvable cases, for which we have presented two algorithms. In future work, one might consider relaxing the syntactic restrictions on the usage of roles, particularly if one were to extend the query language of the oracle.

**References**


