First Order Rewritability in Ontology-Mediated Querying in Horn Description Logics

David Toman and Grant Weddell

Cheriton School of CS, University of Waterloo, Canada
{david,gweddell}@uwaterloo.ca

Abstract

We consider first-order (FO) rewritability for query answering in ontology-mediated querying (OMQ) in which ontologies are formulated in Horn fragments of description logics (DLs). In general, OMQ approaches for such logics rely on non-FO rewriting of the query and/or on non-FO completion of the data, called an ABox. Specifically, we consider the problem of FO rewritability in terms of Beth definability, and show how Craig interpolation can then be used to effectively construct the rewritings, when they exist, from the Clark's completion of Datalog-like programs encoding a given DL TBox and optionally a query. We show how this approach to FO rewritability can also be used to (a) capture integrity constraints commonly available in backend relational data sources, (b) capture constraints inherent in mapping such sources to an ABox, and (c) can be used as an alternative to deriving so-called perfect rewritings of queries in the case of DL-Lite ontologies.

1 Introduction

We consider first-order (FO) rewritability for query answering in the setting of ontology-mediated querying (OMQ) over a knowledge base (KB). The KB is assumed to be formulated in terms of underlying Horn description logics (DLs) in the FunDL family (McIntyre, Toman, and Weddell 2019) as well as in the $\mathcal{ALC}$ family. Dialects in the FunDL family are unusual in that they forgo roles and instead adopt features that are interpreted as partial functions.

Typical OMQ approaches generally rely on either reformulating a given query by incorporating the KB's terminological knowledge (Calvanese et al. 2005, 2007) and then executing the reformulated query over the explicit data in the KB as a relational query, or, for more expressive logics, on a Datalog completion of the explicit data with respect to the KB's terminological knowledge over which the OMQ is answered (Kontchakov et al. 2010, 2011; Lutz et al. 2013; Lutz, Toman, and Wolter 2009a). In the latter case, data completion can always be expressed as a Datalog program. This then raises the *FO rewritability* problem: determining if *a particular OMQ instance* can be equivalently expressed as an FO query over the explicit data in the knowledge base.

Earlier work on OMQ for the FunDL family of DLs has presented what was called a *combined combined approach* to OMQ, and has shown that it is essential to preserve tractability of OMQ in the presence of (limited) value restrictions (Toman and Weddell 2013; St. Jacques, Toman, and Weddell 2016; McIntyre et al. 2019). Based on this approach, we study FO rewritability of OMQ for the particular cases in which the underlying DL is Horn-$\mathcal{SHIQ}$ in the $\mathcal{ALC}$ family, and when it is Horn-$\mathcal{DLFD}$ in the FunDL family. More specifically, for these dialects, we show how the combined combined approach, with the help of *Beth definability* (Beth 1953) applied on the *Clark's completion* (Clark 1977) of the Datalog program used for the completion of the explicit data in the knowledge base, can be used to characterize FO rewritability of OMQs. We also show how *Craig interpolation* (Craig 1957) can then be used to construct such an FO rewriting, when it exists.

The existence of such a rewriting enables an OMQ front-end to a relational data source that underlies an ABox to operate entirely by a more refined query reformulation of a given user query. This yields an SQL query directly executable over a backend relational data source, with no requirement to update tables beforehand.

Our contributions are as follows.

1. We show how to decide uniform FO rewritability of OMQ in Horn-$\mathcal{SHIQ}$ and in Horn-$\mathcal{DLFD}$ via Clark's completion of Datalog programs and Beth definability;
2. We show how our framework extends to *query specific OMQ* by extending existing results for Horn-$\mathcal{DLFD}$; and
3. We show how a variant of the perfect rewriting approach to OMQ can be synthesized by appeal again to Beth definability and Craig interpolation.

This paper integrates earlier preliminary work that was the first to consider FO rewritability of OMQ via Beth definability and Clark's completion for $\mathcal{ALC}$ and FunDL dialects of DLs (Toman and Weddell 2020, 2021). FO rewritability for Horn logics in the $\mathcal{ALC}$ family has also been studied by others, e.g., see (Bienvenu et al. 2016, 2014). This other work has also developed algorithms for generating such rewritings efficiently for logics in the $\mathcal{EL}$ family (Hansen et al. 2015). Our approach seems to provide an alternative path to detecting rewritability and to generating rewritings. A feature of our approach is its link to interpolation-based query optimization (Hudek, Toman, and Weddell 2015; Toman and Weddell 2011). Establishing limits on the size of rewritings (Bienvenu et al. 2017) does provide a guide on what rewritings are reasonable to consider during query optimization.

The use of database constraints that must already hold in explicit data given by a data source, possibly combined with constraints implied by data mapping rules, has been explored in several systems that implement variants of perfect rewriting (Calvanese et al. 2007), such as Ontop and MASTRO (Bagosi et al. 2014; Calvanese et al. 2011) and others. Here, we also show how Beth definability and Clark's completion seamlessly accommodate such constraints into the rewriting via interpolation.

Beth definability and Craig interpolation have been used for other purposes, such as query reformulation under FO constraints (Borgida et al. 2010; Hudek, Toman, and Weddell 2015; Toman and Weddell 2011; Benedikt et al. 2016; Toman and Weddell 2017). That use, however, is orthogonal to the topic of this paper.

The remainder of the paper is organized as follows. Section 2 provides the necessary background and definitions. Here, we review Horn-$\mathcal{SHIQ}$, Horn-$\mathcal{DLFD}$ and the combined combined approach to OMQ. Our main results then follow in Section 3 in which we show how the above-mentioned artifacts, Clark's completion of Datalog programs for example, can be employed to both decide FO rewritability and to synthesize FO rewritings of ABox completion in the combined combined approach to OMQ. Section 4 shows how a Datalog program computing an ABox completion can usefully incorporate database constraints enforced by backend relational database systems. In Section 5, we show how our framework is an alternative approach to perfect rewriting of queries for knowledge bases formulated in DL-Lite. In Section 6, we discuss several limitations and possible extensions of this framework.

2 Background and Definitions

The following define the DL dialects Horn-$\mathcal{SHIQ}$ and Horn-$\mathcal{DLFD}$, respective members of the $\mathcal{ALC}$ and FunDL families that will concern us.

Definition 1 (Concepts in $\mathcal{ALC}$ and FunDL) *Let* R, F, PC, *and* IN *be disjoint sets of primitive role names, primitive feature names, primitive concept names and individual names respectively. General concepts, roles and path functions are defined as follows:*

(for both dialects) *Concepts that are primitive concept names* $A \in \mathsf{PC}$*, or of the form* $C_1 \sqcap C_2$ *for conjunction,* $\bot$ *for bottom, and* $\top$ *for top;*

(for Horn-$\mathcal{SHIQ}$) *Roles* R *of the form* P *and* P^- *(an inverse role) for* $P \in \mathsf{R}$*, and additional concepts of the form* $\forall R.C$ *for value restriction,* $\exists R.C$ *for existential restriction, or either* $(\geq\ n\ R.C)$ *or* $(\leq\ n\ R.C)$*, where* $n \geq 0$*, for a number restriction;*

(for Horn-$\mathcal{DLFD}$) *Path functions* Pf *of the form* id *for identity or the form* $f_1.f_2\ldots f_k$*, where* $f_i \in \mathsf{F}$ *and* $k > 0$*, for function composition, and additional concepts of the form* $\forall f.C$ *for value restriction,* $\exists f^{-1}$ *for unqualified inverse, and* $C : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$ *for a path functional dependency (PFD).*

The semantics is with respect to a structure $\mathcal{I} = (\triangle^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *in which* $\triangle^{\mathcal{I}}$ *is a domain of objects and* $\cdot^{\mathcal{I}}$ *an interpretation function seeded by fixing the interpretations of primitive concept names* A *to be subsets of* $\triangle^{\mathcal{I}}$*, primitive role names* R *to be subsets of* $\triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$*, primitive feature names to be partial functions on* $\triangle^{\mathcal{I}}$*, and individual names* a *to be elements of* $\triangle^{\mathcal{I}}$*, and is extended to complex concepts* C *and roles* R *in the standard way (Baader et al. 2003; McIntyre et al. 2019). Subsumption between concepts and roles, assertions, knowledge bases and their consistency, logical implication, and other reasoning problems are also defined in the standard way.*

In the following, we restrict our attention to KBs in normal forms that are expressively equivalent to the larger logics. For Horn-$\mathcal{SHIQ}$ KBs, we follow the definition in (Eiter et al. 2012), and for Horn-$\mathcal{DLFD}$ the definition in (McIntyre et al. 2019). For more general but expressively equivalent syntax, e.g., that allows other forms of qualified number restrictions, see (Hustadt, Motik, and Sattler 2005; St. Jacques, Toman, and Weddell 2016; McIntyre et al. 2019).

Definition 2 (Horn-$\mathcal{SHIQ}$ KBs (Eiter et al. 2012)) *A Horn-$\mathcal{SHIQ}$* knowledge base $\mathcal{K}$ *consists of a TBox* $\mathcal{T}$ *and an ABox* $\mathcal{A}$. $\mathcal{T}$ *in normal form consists of role subsumptions of the form* $R_1 \sqsubseteq R_2$ *that define a role hierarchy, transitivity assertions* $\mathsf{trans}(R)$*, and concept subsumptions that adhere to one of the following forms:*[1]

$$\begin{array}{ll} A \sqcap B \sqsubseteq C, & A \sqsubseteq \forall R.B, \\ A \sqsubseteq \exists R.B, & \exists R.A \sqsubseteq B, \text{ or} \\ A \sqsubseteq (\leq 1\ R.B), & \end{array}$$

where $A, B, C \in \mathsf{PC} \cup \{\top, \bot\}$. *Roles* R *are called* simple *when neither they nor any of their subroles are transitive. To avoid a well known source of undecidability, we require that any number restriction occurring in* $\mathcal{T}$ *will mention only simple roles. An ABox* $\mathcal{A}$ *consists of concept assertions, role assertions, equality axioms and inequality axioms with the respective forms* $A(a)$*,* $R(a, b)$*,* $a = b$ *and* $a \neq b$*. A Horn-$\mathcal{ALCHQI}$ KB is a Horn-$\mathcal{SHIQ}$ KB without transitivity assertions.*

Definition 3 (Horn-$\mathcal{DLFD}$ KBs (McIntyre et al. 2019)) *A Horn-$\mathcal{DLFD}$ KB also consists of a TBox* $\mathcal{T}$ *and an ABox* $\mathcal{A}$*. Here,* $\mathcal{T}$ *in normal form contains subsumptions of the form* $C \sqsubseteq D$*, where concepts* C *and* D *are defined by the following grammar:*

$$\begin{array}{lcl} C & ::= & A \mid \forall f.A \mid A \sqcap B \\ D & ::= & A \mid \bot \mid \forall f.A \mid \exists f^{-1} \mid A : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf} \end{array}$$

The PFD concept constructor is, in addition, restricted to either the form (a) $A : \mathsf{Pf}.\mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$*, called a*

[1]Note that subsumptions of the form "$A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B$" are also allowed in (Eiter et al. 2012). Here, we are appealing to an obvious conservative extension to replace such subsumptions with strictly binary use of conjunction to further simplify our presentation.

key PFD, or the form (b) $A : f_1, \ldots, f_k \rightarrow f$, which is a functional dependency.[2] *We also assume w.l.o.g. that at least one of the concept descriptions C and D is primitive. An ABox $\mathcal{A}$ consists of concept assertions, equality axioms and inequality axioms with the respective forms* $\mathrm{A}(a)$, $a.f = b$, $a = b$, *and* $a \neq b$.

Note that in Horn-$\mathcal{DLFD}$ the $\forall f.\cdot$ concept constructor serves both as a value restriction and an existential restriction, thereby ensuring the logic is Horn. To simplify further development, we assume in both cases that ABoxes also contain atoms of the form $\top(a)$ for every constant symbol a present in a given ABox. This is w.l.o.g. since computing the so-called *active domain* of a relational instance is FO definable (Abiteboul, Hull, and Vianu 1995).

Example 4 Subsumption constraints comprising a simple Horn-$\mathcal{DLFD}$ TBox are given as follows. The ontology concerns EMPloyees, DEParTments, managers, and supervisors, in particular that: (a) employees have employees as supervisors, (b) departments have managers who are always employees, and (c) departments have special cases of MATHematics and SCIence departments.

$$\begin{aligned} \mathtt{EMP} &\sqsubseteq \forall \mathtt{supervisedBy}.\mathtt{EMP} \\ \mathtt{DEPT} &\sqsubseteq \forall \mathtt{managedBy}.\mathtt{EMP} \\ \mathtt{MATH} &\sqsubseteq \mathtt{DEPT} \\ \mathtt{SCI} &\sqsubseteq \mathtt{DEPT} \end{aligned}$$

Although not given for space reasons, there could also be constraints expressing keys, functional dependences, disjointness, and so on, that can be expressed in Horn-$\mathcal{DLFD}$.

Conjunctive queries and OMQ. Conjunctive queries are, as usual, formed from atomic queries (or *atoms*) of the form $A(x)$ and either $R(x, y)$ in Horn-$\mathcal{SHIQ}$ or $f(x) = y$ in Horn-$\mathcal{DLFD}$, where x and y are variables, using conjunction and existential quantification in prenex normal form. As usual, we conflate a conjunctive query with the set of its constituent atoms and a list of *answer variables* to simplify notation.

Definition 5 (Conjunctive Query) *Let ψ be a set of atoms $A(x_i)$ and either $R(x_{i_1}, x_{i_2})$ or $f(x_{i_1}) = x_{i_2}$, where A is a primitive concept name or $\top$, R a role name, f a feature name, and $\vec{x}$ a tuple of variables. We call the expression $\varphi = \{\vec{x} \mid \psi\}$ a* conjunctive query (CQ)*, and define* $\mathsf{Atoms}(\varphi)$ *to be ψ.*

A CQ φ is also a notational variant of the formula $\exists \vec{y}. \bigwedge_{\phi \in \psi} \phi$ in which $\vec{y}$ contains all variables appearing in ψ but not in $\vec{x}$.[3] We also omit set braces when explicitly listing atoms in ψ to improve readability. With this understanding, the usual definition of certain answers is assumed and given as follows:

Definition 6 (Certain Answer) *Let $\mathcal{K}$ be a knowledge base and $\varphi = \{\vec{x} \mid \psi\}$ a CQ. A* certain answer *to φ over $\mathcal{K}$ is a tuple of constant symbols $\vec{a}$ such that $\mathcal{K} \models \varphi(\vec{a})$ (where $\varphi(\vec{a})$ is short for $\varphi[\vec{x} \mapsto \vec{a}]$).*

Our primary concern is then given by the following problem:

Definition 7 ((Uniform) FO Query Rewritability) *Given a TBox $\mathcal{T}$, the problem of* query rewritability *is to determine if there is a query reformulation $\varphi_{\mathcal{T}}$ for* every *CQ φ such that, for every ABox $\mathcal{A}$ and tuple of constant symbols $\vec{a}$, $(\mathcal{T}, \mathcal{A}) \models \varphi(\vec{a})$ iff $\mathcal{A} \models \varphi_{\mathcal{T}}(\vec{a})$.*

Later on, we consider a query-specific variant of this problem: whether such a rewriting exists for a given CQ. The following observations will also be useful in regard to this problem.

Observation 8 (Transitivity) *Consider a Horn-$\mathcal{SHIQ}$ knowledge base with a TBox $\{\mathsf{trans}(R)\}$. Then the CQ $\{(x, y) \mid R(x, y)\}$ cannot be FO rewritable since this would allow one to answer the* connectivity *question with respect to any ABox considered as a graph of R-edges.*

Analogously to transitive roles, allowing equality and inequality between ABox objects, and therefore not adopting the *unique name assumption* (UNA), leads immediately to non-rewritability:

Observation 9 (Equality) *Consider a KB in which $\mathcal{T} = \emptyset$ and a CQ $\{(x, y) \mid x = y\}$. Again, this query solves the (undirected) connectivity problem in an ABox with explicit equalities between individuals and thus cannot have an FO rewriting.*

Hence, hereon, it suffices to consider the Horn-$\mathcal{ALCHQI}$ sub-dialect of Horn-$\mathcal{SHIQ}$ without transitive roles, and to also adopt UNA for both Horn-$\mathcal{SHIQ}$ and Horn-$\mathcal{DLFD}$. Thus, an ABox in either case will consist of only concept assertions, role assertions and equality axioms of the form $a.f = b$.

To study FO rewritability of conjunctive queries over Horn-$\mathcal{ALCHQI}$ or Horn-$\mathcal{DLFD}$ knowledge bases, we begin with the following manifestation of a combined combined approach to OMQ originally developed for Horn-$\mathcal{DLFD}$ (McIntyre et al. 2019; St. Jacques, Toman, and Weddell 2016; Toman and Weddell 2013).[4] Here, we define the approach in a way that also accommodates Horn-$\mathcal{ALCHQI}$, in preparation to showing in the next section how to decide query rewritability with respect to knowledge bases expressed in either of our logics. Indeed, the following proposition holds for both the dialects under consideration:

[2]Reasons for these restrictions can be found in (St. Jacques, Toman, and Weddell 2016). The latter can be generalized to $A : \mathsf{Pf}.f\ \mathsf{Pf}_2, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}.g$ (McIntyre et al. 2019), a development that is beyond the scope of this paper. Allowing the general form leads to undecidability (Toman and Weddell 2008).

[3]Note that it is not necessary to place any restrictions on the variables $\vec{x}$. Indeed, one can add additional atoms $\top(x_i)$ to ensure variables in $\vec{x}$ also appear in ψ, if desired, without any impact on the remaining results.

[4]Note that the original *combined approach* (Kontchakov et al. 2010; Lutz, Toman, and Wolter 2009b) used the TBox subsumptions to complete the ABox but not to rewrite the CQ. The approach presented here *combines* this combined approach with a variation on *perfect rewriting* (Calvanese et al. 2007) and was called the *combined combined approach* as a consequence.

Proposition 10 (The Combined Combined Approach) *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a consistent knowledge base and φ a conjunctive query. Then there is a UCQ query $\varphi_{\mathcal{T}}$ and a Datalog program $\Pi_{\mathcal{T}}$, both of which can be effectively constructed from $\mathcal{T}$, such that*

$$\mathcal{K} \models \varphi(\vec{a}) \iff \Pi_{\mathcal{T}}(\mathcal{A}) \models \varphi_{\mathcal{T}}(\vec{a})$$

for any tuple of constant symbols $\vec{a}$, and where $\Pi_{\mathcal{T}}(\mathcal{A})$ is the minimal model of $\Pi_{\mathcal{T}}$ when evaluated over $\mathcal{A}$.

For details please see (Eiter et al. 2012) and (Toman and Weddell 2013; St. Jacques, Toman, and Weddell 2016; McIntyre et al. 2019), respectively. The proposition uses a Datalog program $\Pi_{\mathcal{T}}$ to define an ABox completion over which the query $\varphi_{\mathcal{T}}$, the rewriting of the original user query, is evaluated to compute the certain answers. The Proposition also shows that the *data complexity* of OMQ is complete for PTIME.

3 Uniform Rewriteability

Note that the existence of $\varphi_{\mathcal{T}}$ in Proposition 10 indicates that the non-rewritability of CQs is confined to the interaction of the TBox with explicit data given by an ABox which is captured by the Datalog program $\Pi_{\mathcal{T}}$.

Example 11 Consider the TBox from Example 4 and an ABox of the form

$$\mathcal{A} = \{\texttt{EMP}(a_k)\} \cup \{\texttt{supervisedBy}(a_i, a_{i+1}) \mid 0 \le i < 2k\}.$$

In a completion of $\mathcal{A}$ all a_i have to belong to the concept EMP for all $k \le i \le 2k$. This requires the closure process to solve the *reachability* (from a_k) problem and thus cannot be expressed by a FO query.

Datalog programs and clauses that follow use the standard syntax and semantics, and, in particular, predicates used in such programs are classified as either EDB (extensional predicates), those for which we have explicit data, and IDB (intensional predicates), predicates whose interpretation is defined by the *minimal model semantics* of Datalog (Ullman 1982, 1988, 1989).

Definition 12 (Datalog Program $\Pi_{\mathcal{T}}$) The Datalog program $\Pi_{\mathcal{T}}$ used in Proposition 10 consists of completion rules obtained by translating subsumptions that are logical consequences of $\mathcal{T}$. The form of these subsumptions for Horn-$\mathcal{ALCHQI}$ and their translation are given as follows:

(consequences of $\mathcal{T}$)	(completion rule in $\Pi_{\mathcal{T}}$)
$\mathrm{A}_1 \sqcap \mathrm{A}_2 \sqsubseteq \mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_{A_1}(x), \mathsf{C}_{A_2}(x)$
$\mathrm{A} \sqsubseteq \forall R.\mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_A(y), \mathsf{R}_R(y, x)$
$\exists R.\mathrm{A} \sqsubseteq \mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_A(y), \mathsf{R}_R(x, y)$
$R \sqsubseteq S$	$\mathsf{R}_S(x, y) \leftarrow \mathsf{R}_R(x, y)$

For every primitive concept B and role R, we introduce unary EDB predicates $\mathsf{P}_B(x)$ and $\mathsf{P}_R(x, y)$ together with additional clauses $\mathsf{C}_B(x) \leftarrow \mathsf{P}_B(x)$, $\mathsf{R}_R(x, y) \leftarrow \mathsf{P}_R(x, y)$, and $\mathsf{R}_{R^-}(x, y) \leftarrow \mathsf{P}_R(y, x)$ to account for *explicit* data of the form $A(a)$ and $R(a, b)$ in an ABox, and IDB predicates $\mathsf{C}_B(x)$ and $\mathsf{R}_R(x, y)$ corresponding to the *completion* of the ABox w.r.t. $\mathcal{T}$.

For Horn-$\mathcal{DLFD}$ the Datalog program $\Pi_{\mathcal{T}}$ is obtained in an analogous fashion, where $\mathsf{P}_f(x, y)$ serves as a synonym for an assertion $f(x) = y$ to conform with standard Datalog syntax:

(consequences of $\mathcal{T}$)	(completion rule in $\Pi_{\mathcal{T}}$)
$\mathrm{A}_1 \sqcap \mathrm{A}_2 \sqsubseteq \mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_{A_1}(x), \mathsf{C}_{A_2}(x)$
$\mathrm{A} \sqsubseteq \forall f.\mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_A(y), \mathsf{P}_f(y, x)$
$\forall f.\mathrm{A} \sqsubseteq \mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_A(y), \mathsf{P}_f(x, y)$

In the programs $\Pi_{\mathcal{T}}$, we call the $\mathsf{C}_B(x)$ and $\mathsf{R}_R(x, y)$ predicates *intensional* (or IDB predicates) and denote the set of all these predicates in $\Pi_{\mathcal{T}}$ by $\mathsf{IDB}(\mathcal{T})$. Similarly, we call all the $\mathsf{P}_*(\ldots)$ predicates *extensional* (or EDB) and denote the set of all these by $\mathsf{EDB}(\mathcal{T})$.

Note that it is unnecessary for the program $\Pi_{\mathcal{T}}$ to consider at-most restrictions, unqualified inverses and PFDs since we are assuming that the KB is consistent. Moreover, $\Pi_{\mathcal{T}}$ can also ignore the creation of anonymous individuals, e.g., implied by existential restrictions, since that task is delegated to query reformulation $\varphi_{\mathcal{T}}$ in Proposition 10.

Example 13 The Datalog program for the TBox in Example 4 is as follows:

$$\begin{aligned}
\mathsf{C}_{\texttt{EMP}}(x) &\leftarrow \mathsf{C}_{\texttt{EMP}}(y), \mathsf{P}_{\texttt{supervisedBy}}(y, x)\\
\mathsf{C}_{\texttt{EMP}}(x) &\leftarrow \mathsf{C}_{\texttt{DEPT}}(y), \mathsf{P}_{\texttt{managedBy}}(y, x)\\
\mathsf{C}_{\texttt{DEPT}}(x) &\leftarrow \mathsf{C}_{\texttt{MATH}}(x)\\
\mathsf{C}_{\texttt{DEPT}}(x) &\leftarrow \mathsf{C}_{\texttt{SCI}}(x)
\end{aligned}$$

To test for FO definability of the completion (i.e., all the C_A (R_R) predicates that stand for the completed ABox instance), we use the following construction:

Definition 14 (Clark's Completion $\Sigma_{\mathcal{T}}$) *The* Clark's Completion (Clark 1977) $\Sigma_{\mathcal{T}}$ *of* $\Pi_{\mathcal{T}}$ *is given by a set of formulas*

$$\begin{aligned}
\mathsf{C}_B(x) &\leftrightarrow \mathsf{P}_B(x) \vee (\exists y.\alpha_1) \vee \ldots \vee (\exists y.\alpha_n)\\
\mathsf{R}_R(x, y) &\leftrightarrow \mathsf{P}_R(x, y) \vee \beta_1 \vee \ldots \vee \beta_m
\end{aligned}$$

corresponding to clauses $\mathsf{C}_B(x) \leftarrow \alpha_i$ and $\mathsf{R}_R(x, y) \leftarrow \beta_j$ in $\Pi_{\mathcal{T}}$, grouped by the heads of the $\mathsf{IDB}(\mathcal{T})$ predicates.

The *bodies* α_i (β_i) are introduced in Definition 12. Note also that the *Clark's Completion* is no longer a Datalog program. This completion, however, closes the original Datalog program in the following sense:

Proposition 15 ((Clark 1977), simplified for this paper) *Let $(\mathcal{T}, \mathcal{A})$ be a knowledge base. Then for every predicate $P \in \mathsf{IDB}(\mathcal{T}) \cup \mathsf{EDB}(\mathcal{T})$, ABox $\mathcal{A}$, and a vector of constants $\vec{a}$ (of appropriate arity) we have*

- $\Pi_{\mathcal{T}} \cup \mathcal{A}_{db} \models P(\vec{a})$ *implies* $\Sigma_{\mathcal{T}} \cup \mathcal{A}_{db} \models P(\vec{a})$*, and*
- *The goal $P(\vec{a})$* finitely failing $\Pi_{\mathcal{T}} \cup \mathcal{A}_{db}$ *implies* $\Sigma_{\mathcal{T}} \cup \mathcal{A}_{db} \models \neg P(\vec{a})$,

where $\mathcal{A}_{db}$ is the closed world variant of $\mathcal{A}$, a set of ground facts such that all facts not in $\mathcal{A}_{db}$ *are false.*[5]

[5] This too can be accomplished by Clark's completion of $\mathcal{A}$, but we will not rely on this fact in the rest of the paper.

Example 16 The Clarke's completion of the Datalog program for the TBox in Example 4 is as follows:

$$\begin{array}{rcl}
\mathsf{C}_{\texttt{EMP}}(x) & \leftrightarrow & \mathsf{P}_{\texttt{EMP}}(x) \\
& \vee & (\exists y.\mathsf{C}_{\texttt{EMP}}(y) \wedge \mathsf{P}_{\texttt{supervisedBy}}(y,x)) \\
& \vee & (\exists y.\mathsf{C}_{\texttt{DEPT}}(y) \wedge \mathsf{P}_{\texttt{managedBy}}(y,x)) \\
\mathsf{C}_{\texttt{DEPT}}(x) & \leftrightarrow & \mathsf{P}_{\texttt{DEPT}}(x) \\
& \vee & \mathsf{C}_{\texttt{MATH}}(x) \\
& \vee & \mathsf{C}_{\texttt{SCI}}(x) \\
\mathsf{C}_{\texttt{MATH}}(x) & \leftrightarrow & \mathsf{P}_{\texttt{MATH}}(x) \\
\mathsf{C}_{\texttt{SCI}}(x) & \leftrightarrow & \mathsf{P}_{\texttt{SCI}}(x)
\end{array}$$

Note that Clark's result works in the much more general setting of logic programs with function symbols and possibly infinite resolution proofs and under the *Negation As Failure* semantics. Note also, Clark's completion differs from, e.g., the closed world assumption (CWA) (Reiter 1977), and its variants, in a crucial way. For example, for a clause of the form $p \leftarrow p$ (and for cycles in programs to be completed in general), the completion simply generates a formula $p \leftrightarrow p$ that in turn allows models in which p can be true and models in which p is false. This is indeed the heart of our approach since this behaviour of the Clark's completion feeds into the Beth definability test below. Moreover, had we used $\Pi_{\mathcal{T}}$ instead of $\Sigma_{\mathcal{T}}$, none of the definability results could possibly hold, even in the absence of role/feature subsumptions (such as role hierarchies).

Proposition 17 (Projective Beth Definability (Beth 1953)) *Let Σ be an FO theory over symbols in L and $L' \subseteq L$. Then the following are equivalent:*

1. *For M_1 and M_2 models of Σ such that $M_{1|L'} = M_{2|L'}$, it holds that $M_1 \models \varphi[\vec{a}]$ iff $M_2 \models \varphi[\vec{a}]$ for all M_1, M_2, and $\vec{a}$ tuples of constants, and*
2. *φ is equivalent under Σ to a formula ψ in L' (we say φ* is Beth definable over Σ and L'*).*

This gives us a complete characterization of FO rewritability of the ABox closure of individual primitive concept names with respect to Horn-$\mathcal{ALCHQI}$ and Horn-$\mathcal{DLFD}$ TBoxes as follows:

Theorem 18 *Let $\mathcal{T}$ be a TBox in one of DL dialects considered. Then the completion of an ABox $\mathcal{A}$ w.r.t. $\mathcal{T}$ is* FO definable *if and only if every predicate in $\mathsf{IDB}(\mathcal{T})$ is Beth definable over $\Sigma_{\mathcal{T}}$ and $L' = \mathsf{EDB}(\mathcal{T})$.*

Proof (sketch): Follows immediately from the properties of Beth definability (Proposition 17) and the definition and properties of the Clark's completion (Proposition 15).

Observe that one can restrict the alphabet of the ABox (L') to target only ABoxes over restricted signature(s).

Example 19 Applying the above test on the Clarke's completion in Example 16 of the Datalog program for the TBox $\mathcal{T}_{\text{Example}}$ in Example 4 reveals that $\mathsf{C}_{\texttt{DEPT}}$ is Beth definable w.r.t. $\mathsf{EDB}(\mathcal{T})$ (as in every model the interpretation of $\mathsf{C}_{\texttt{DEPT}}$ is the same) while $\mathsf{C}_{\texttt{EMP}}$ is not definable (as in different models $\mathsf{C}_{\texttt{EMP}}$ can be interpreted differently due to the cycle in its definition).

Given $\Sigma_{\mathcal{T}}$, one can now reformulate (1) in Proposition 17 as a logical implication problem by making a copy of all formulas of $\Sigma_{\mathcal{T}}$ in which all non-logical symbols *not in* $\mathsf{EDB}(\mathcal{T})$ are starred. Hence, the definability questions for $\mathsf{C}_A(x)$ and $\mathsf{R}_R(x,y)$ can be expressed as respective logical implication questions of the form:

$$\begin{array}{l}
\Sigma_{\mathcal{T}} \cup \Sigma^*_{\mathcal{T}} \models \forall x.\mathsf{C}_A(x) \to \mathsf{C}^*_A(x) \\
\Sigma_{\mathcal{T}} \cup \Sigma^*_{\mathcal{T}} \models \forall x,y.\mathsf{R}_R(x,y) \to \mathsf{R}^*_R(x,y)
\end{array}$$

Note that, on closer inspection, all formulas in $\Sigma_{\mathcal{T}}$ can be written as $\mathcal{ALCI}$ subsumptions. Note also that, without role constructors, there is no need to check for the definability of $\mathsf{R}_R(x,y)$ atoms since they are always definable (we elaborate on the role of role constructors in Section 6). Hence:

Theorem 20 *Let $\mathcal{T}$ be a TBox in one of the dialects considered. Then the existence of*

1. *the FO rewritability of the completion of $\mathcal{A}$ with respect to $\mathcal{T}$, and*
2. *the uniform query rewritability over $\mathcal{T}$*

are decidable and in EXPTIME.

Proof (sketch): The first claim follows immediately from Theorem 18 applied to all predicates C_B and R_R and the decidability and complexity of reasoning in $\mathcal{ALCI}$. The second claim follows by observing that (i) definability of atomic queries implies definability of arbitrary UCQs using the combined combined approach, and that (ii) non-definability of a single atomic query exhibits the need for a non FO ABox completion for queries containing/consisting of this atom.

Note that the above holds due to the *specific* structure of $\Pi_{\mathcal{T}}$ and is not applicable to general Datalog programs. Indeed, (Chaudhuri and Vardi 1997) show much higher bounds for general programs. A matching lower bound can be obtained for expressive fragments of Horn-$\mathcal{ALC}$ (for which the complexity of reasoning is EXPTIME-complete).[6] However, since the size (and the construction) of rewritings will commonly dominate this cost, even for the simplest ontology languages (Kikot et al. 2012), exact complexity bounds are mostly of academic interest.

Construction of rewritings. To obtain an algorithm that constructs rewritings from our characterization of FO rewritability, we utilize Craig interpolation:

Proposition 21 (Craig Interpolation (Craig 1957)) *Let φ and ϕ be FO formulas such that $\models \varphi \to \phi$. Then there is an FO formula ψ, called the* Craig interpolant*, containing only symbols common to φ and ϕ such that $\models \varphi \to \psi$ and $\models \psi \to \phi$.*

Moreover, the interpolant can be extracted, typically in linear time, from a proof of $\models \varphi \to \phi$, as long as a reasonably structural proof system, such as resolution, (cut-free) sequent calculus, and/or analytic tableau is used. Combining the above construction with the rewriting $\varphi_{\mathcal{T}}$ we get:

[6] It was noted in (Toman and Weddell 2020) that the exact complexity is open for PTIME fragments of Horn-$\mathcal{DLFD}$, such as $\mathcal{CFD}_{nc}$ and $\mathcal{CFDI}^{\forall-}_{kc}$.

Theorem 22 *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a consistent knowledge base in either Horn-$\mathcal{ALCHQI}$ or Horn-$\mathcal{DLFD}$. Then the data complexity (i.e., in $|\mathcal{A}|$) of uniform conjunctive query answering is in AC^0 whenever the ABox completion with respect to $\mathcal{T}$ is FO definable with respect to $\Sigma_\mathcal{T}$.*

Proof (sketch): Let ψ_P be FO definitions of $P \in \mathsf{EDB}(\mathcal{T})$ w.r.t. $\Sigma_\mathcal{T}$. Then

$$\mathcal{K} \models \varphi(\vec{a}) \iff \mathcal{A}_{db} \models \varphi_\mathcal{T}[\psi_P[\vec{y}/\vec{x}]/P(\vec{y}) \mid P(\vec{y}) \in \mathsf{Atoms}(\varphi)](\vec{a}).$$

The claim follows since

$$\varphi_\mathcal{T}[\psi_P[\vec{y}/\vec{x}]/P(\vec{y}) \mid P(\vec{y}) \in \mathsf{Atoms}(\varphi)]$$

is an FO formula, in particular, a UCQ.

Example 23 Applying the interpolant construction on the Clarke's completion in Example 16 will obtain:

$$\mathsf{C}_{\mathtt{DEPT}}(x) \leftrightarrow (\mathsf{P}_{\mathtt{DEPT}}(x) \vee \mathsf{P}_{\mathtt{MATH}}(x) \vee \mathsf{P}_{\mathtt{SCI}}(x)).^7$$

Note that there is no interpolant for $\mathsf{C}_{\mathtt{EMP}}$.

Query-specific Rewritability. Our approach also provides tools for deciding the non-uniform query-specific problems. Indeed, although one can explicitly construct $\varphi_\mathcal{T}$, in deciding FO rewritability of a CQ it is only necessary to determine the atomic formulas for which interpolants are needed in the reformulation. It is therefore sufficient to focus on determining the smallest set of such atoms, up to equivalence under $\mathcal{T}$, since the rewriting $\varphi_\mathcal{T}$ is a UCQ. This yields our desired result:

Theorem 24 *Let $\mathcal{T}$ be a TBox and φ a CQ. Then the following are equivalent:*

1. *φ is FO rewritable with respect to $\mathcal{T}$ and*
2. *$\Sigma_\mathcal{T} \cup \Sigma^*_\mathcal{T} \models \forall\vec{x}.P(\vec{x}) \to P^*(\vec{x})$ for all $P(\vec{x}) \in \mathsf{Atoms}(\varphi_\mathcal{T}) \cap \mathsf{IDB}(\mathcal{T})$.*

The exact complexity again depends on the complexity of (2) in Proposition 17. In the general case, an EXPTIME bound follows from (Horrocks et al. 2000), but again, a more refined analysis is in order for fragments of our DLs. In our example, the CQ $\{x \mid \mathtt{DEPT}(x)\}$ is FO rewritable (follows from Example 23), while $\{x \mid \mathtt{EMP}(x)\}$ is not.

4 Integration of Database Constraints

In OMQ, it is usually assumed that the ABox is virtual, and therefore defined by mapping rules over the TBox signature and the schema of an underlying relational data source. Thus, a backend (relational) system will enforce so-called *integrity constraints* such as *view definitions*, *primary keys* and *foreign keys*. This is important since integrity constraints ensure that parts of the corresponding ABox will not only be *consistent* with their definition but will also be a *model*, and therefore may not require a completion. Unary *foreign keys* implied by database schemata and/or the mapping rules that hold over the ABox signature seem to be of main utility due to the structure of ABox closure defined by $\Pi_\mathcal{T}$. We illustrate this in the case of Horn-$\mathcal{DLFD}$.[8] Indeed, for a foreign key to hold in a relational database, either its *target already exists* in the appropriate table or the source is `NULL`, which can then be interpreted as *value unknown* and is taken care of by the query rewriting $\varphi_\mathcal{T}$.

Definition 25 (Adding Foreign Keys) *Let $A \sqsubseteq B$, $A \sqsubseteq \forall f.B$, and $\forall f.A \sqsubseteq B$ be Horn-$\mathcal{DLFD}$ subsumptions that correspond to unary foreign keys implied by the backend data source. For each such constraint, add $\mathsf{P}_B(x) \leftarrow \mathsf{C}_A(x)$, $\mathsf{P}_B(x) \leftarrow \exists y.\mathsf{C}_A(y), \mathsf{P}_f(y,x)$, and $\mathsf{P}_B(x) \leftarrow \exists y.\mathsf{C}_A(y), \mathsf{P}_f(x,y)$, respectively, to the Clark's completion.*

Augmenting $\Sigma_\mathcal{T}$ in this way enables sidestepping parts of the ABox completion that are mandated by the foreign keys and thus *already exist* in the original instance of the ABox.

Example 26 Applying the interpolant construction on the Clarke's completion in Example 16 will now obtain:

$$\mathsf{C}_{\mathtt{DEPT}}(x) \leftrightarrow (\mathsf{P}_{\mathtt{DEPT}}(x) \vee \mathsf{P}_{\mathtt{MATH}}(x) \vee \mathsf{P}_{\mathtt{SCI}}(x))$$
$$\mathsf{C}_{\mathtt{EMP}}(x) \leftrightarrow \mathsf{P}_{\mathtt{EMP}}(x)$$

The reason that $\mathsf{C}_{\mathtt{EMP}}$ is now definable reduces to observing that all explicit `EMP` objects are already recorded in the EMPloyee table as dictated by the DB constraints.

This also shows that our Horn-$\mathcal{DLFD}$ TBox from Example 4, with the help of DB constraints, satisfies the conditions of uniform query rewritability. The approach also provides decidability for the non-uniform (query-specific) problems (as before): to decide FO rewritability of a CQ, one only needs to determine the atomic formulas for which interpolants are needed in the reformulation $\varphi_\mathcal{T}$.

5 A One-step Construction of Rewritings

We have been considering the test for existence of rewritings and the construction of such rewritings as a two part process: (i) the construction of an ABox completion, and (ii) subsequent query reformulation. We have already noted that non-rewritability can always be traced to part (i) of this process. An interesting question that emerges, however, is whether such a two-part process is needed. In this section we outline a *one-step* approach to the problem that continues to be based on Clark's completion and on Beth definability, optionally followed by Craig interpolation. Now, however, we apply both techniques to the full TBox, i.e., including *existential restrictions* that may generate anonymous objects, and to the user query (to save space, in Horn-$\mathcal{ALC}$ only).

Definition 27 (Logic Program for Horn-$\mathcal{ALC}$ TBox)

(entailed by $\mathcal{T}$)	*(completion rule in $\Pi_\mathcal{T}$)*
$\mathrm{A}_1 \sqcap \mathrm{A}_2 \sqsubseteq \mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_{A_1}(x), \mathsf{C}_{A_2}(x)$
$\mathrm{A} \sqsubseteq \forall R.\mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_A(y), \mathsf{R}_R(y,x)$
$\exists R.\mathrm{A} \sqsubseteq \mathrm{B}$	$\mathsf{C}_B(x) \leftarrow \mathsf{C}_A(y), \mathsf{R}_R(x,y)$
$\mathrm{A} \sqsubseteq \exists R.\mathrm{B}$	$\mathsf{R}_R(x, f_R(x)) \leftarrow \mathsf{C}_A(x)$, *and* $\mathsf{C}_B(f_R(x)) \leftarrow \mathsf{C}_A(x)$

[7]This interpolant has been automatically generated by an experimental query reformulation system (Hudek, Toman, and Weddell 2015; Toman and Weddell 2017).

[8]The FunDL family was developed primarily to capturing relational schemata. Examples illustrating this can be found in (St. Jacques, Toman, and Weddell 2016; McIntyre et al. 2019). However, a similar approach will work for Horn-$\mathcal{ALCHQI}$ as well.

Note that the construction of a Datalog program in Section 3 omitted the last rule (for $\mathrm{A} \sqsubseteq \exists R.\mathrm{B}$) since the effect of that subsumption has been accommodated by query reformulation. The clauses stemming from the existential restrictions contain Skolem functions f_R and hence the resulting set of clauses is no longer a Datalog program. To define the Clark's completion, observe that the clauses for $\mathrm{A} \sqsubseteq \exists R.\mathrm{B}$ can be equivalently written as

$$\begin{aligned} \mathsf{R}_R(x,y) &\leftarrow y = f_R(x), \mathsf{C}_A(x) \\ \mathsf{C}_B(x) &\leftarrow x = f_R(y), \mathsf{C}_A(y) \end{aligned}$$

as shown in (Clark 1977). Now, since the heads of the clauses in $\Pi_{\mathcal{T}}$ have a common form (i.e., $\mathsf{C}_B(x)$ or $\mathsf{R}_R(x,y)$), we can create the completion $\Sigma_{\mathcal{T}}$ as in Definition 14. (This requires adding the standard equality axioms or assuming equality is an *interpreted* predicate.) For a given CQ φ of the form $\{\vec{x} \mid \psi\}$, extend the completion with

$$\Sigma_{\mathcal{T},\varphi} = \Sigma_{\mathcal{T}} \cup \{Q(\vec{x}) \leftrightarrow \psi\}$$

(with Q a new symbol). Then one can apply the definability test as in the previous case to obtain the rewritability test:

Theorem 28 *Let $\mathcal{T}$ be a Horn-$\mathcal{ALC}$ TBox and φ a CQ. Then the following are equivalent:*

1. $\Sigma_{\mathcal{T},\varphi} \cup \Sigma^*_{\mathcal{T},\varphi} \models \forall\vec{x}.Q(\vec{x}) \to Q^*(\vec{x})$, and
2. φ is FO rewritable with respect to $\mathcal{T}$.

The theorem provides a direct, sound and complete test for FO rewritability of CQs with respect to Horn-$\mathcal{ALC}$ TBoxes. However, unlike in the case of Datalog, we need to ensure that the test is still decidable and has a reasonable computational complexity. Note that the actual complexity in this case is tied to the proof system used to prove (1) in Theorem 28: as in DL reasoners, not every proof system achieves the optimal complexity bound. For Horn-$\mathcal{ALC}$, a reduction to the Ackermann prefix with equality (Ackermann 1928) seems feasible, with the aim of obtaining the complexity bound using Fürer's result (Fürer 1981). However, if one is interested in generating the rewriting in the form of an interpolant, a suitable proof system that supports interpolant generation, such as Analytic Tableau (Fitting 1996), (cut-free) Sequent Calculus, or Resolution, is needed. Alternative blocking-style techniques used in DL tableau reasoners are very likely to apply here as well. An intriguing possibility is to also just limit the depth of terms in a general high-performance theorem prover along the lines described by Chomicki for Datalog$_{nS}$ (Chomicki 1995). Both of these options are subjects of future research.

An interesting application of Theorem 28 emerges when the given TBox is formulated in DL-Lite variants, for which interpolants will then correspond to the result obtained by *perfect query reformulation* developed in (Calvanese et al. 2005, 2007). It is relatively easy to observe that the *one-step* interpolation-based approach always succeeds and produces essentially the *perfect rewritings* of conjunctive queries.

6 Summary and Extensions

In this section, we briefly discuss several common extensions of Horn-$\mathcal{ALC}$ that we have omitted so far in our development to keep the presentation of the main ideas cleaner. In the light of Theorem 28, it is relatively immediate that any extension that leads to Horn $\Pi_{\mathcal{T}}$ can be accommodated. Note that, to extend the two-step combined combined approach, we would need to modify, often in non-trivial manner, the query reformulation algorithm. For an example that accommodates inverse features and a variety of equality generating dependencies called *path-functional dependencies*, see (McIntyre et al. 2019).

Additional concept and role constructors, and the induced subsumptions, can be classified in three groups:

1. Constructors that lead to *full* Horn rules, i.e., without existential quantifiers in their heads, that preserve the tree model property. Rules corresponding to these constructors can simply be added in Definition 12 and Definition 27 without any major impact on the query reformulation $\varphi_{\mathcal{T}}$ in Proposition 10;
2. Constructors that lead to *embedded* Horn rules with existential quantifiers in their heads that continue to preserve the tree model property. Here, both Definition 12 and Proposition 10 need to be extended to account for the possibility of additional anonymous individuals. Alternatively, one can capture all of the effects by naturally extending Definition 27 and proceeding with a one-step definability test; and
3. Constructors that break the tree-model property. Examples relate to transitivity assertions and nominals; here, it is not always clear how to make Proposition 10 hold. However, extending Definition 27 and subsequently using Theorem 28 will still work.

Using Theorem 28, while sound and complete for determining rewritability, does not come for free. With each extension, one needs to revisit the decidability and complexity of the definability test which, ultimately, becomes undecidable. This happens even in cases when only unary function symbols are needed but where unrestricted use of binary predicates, such as roles, are allowed.

Extensions that are unlikely to be possible. Of course, there are limits to the definability-based approach:

(*beyond Horn logics*) The approach for Horn logics relies crucially on the existence of a unique minimal model that can be characterized using the Clark's completion. This insight then makes Beth definability and Craig interpolation work. It remains unclear how this idea could generalize to logics without the minimal model property (i.e., non-Horn). For these reasons PTIME-coNP boundaries (Lutz and Wolter 2012) are unlikely to be resolved using these techniques.

(*beyond FO logics*) The synthesis of the rewritings is tied to Craig interpolation. Hence synthesizing, e.g., linear Datalog or dealing with dichotomies on the NL-PTIME (Lutz and Sabellek 2017) boundary seems also to be beyond the capabilities of the techniques used in this paper. Applying results on interpolation in non-first order logics, such as the μ-calculus (D'Agostino and Hollenberg 2000), will be the focus of future research. However, the combined combined approach already gives one an explicit Datalog rewriting, so the space to be explored seems to be rather limited.

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley.

Ackermann, W. 1928. Uber die Erfullbarkeit gewisser Zahlausdrucke. *Mathematische Annalen*, 100: 638–649.

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F. 2003. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press.

Bagosi, T.; Calvanese, D.; Hardi, J.; Komla-Ebri, S.; Lanti, D.; Rezk, M.; Rodriguez-Muro, M.; Slusnys, M.; and Xiao, G. 2014. The Ontop Framework for Ontology Based Data Access. In *The Semantic Web and Web Science - 8th Chinese Conference, CSWS 2014, Revised Selected Papers*, 67–77.

Benedikt, M.; Leblay, J.; ten Cate, B.; and Tsamoura, E. 2016. *Generating Plans from Proofs: The Interpolation-based Approach to Query Reformulation*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

Beth, E. W. 1953. On Padoa's method in the theory of definition. *Indagationes Mathematicae*, 15: 330–339.

Bienvenu, M.; Hansen, P.; Lutz, C.; and Wolter, F. 2016. First Order-Rewritability and Containment of Conjunctive Queries in Horn Description Logics. In *Proceedings of the 29th International Workshop on Description Logics, Cape Town, South Africa*.

Bienvenu, M.; Kikot, S.; Kontchakov, R.; Podolskii, V. V.; Ryzhikov, V.; and Zakharyaschev, M. 2017. The Complexity of Ontology-Based Data Access with OWL 2 QL and Bounded Treewidth Queries. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017*, 201–216.

Bienvenu, M.; ten Cate, B.; Lutz, C.; and Wolter, F. 2014. Ontology-Based Data Access: A Study through Disjunctive Datalog, CSP, and MMSNP. *ACM Trans. Database Syst.*, 39(4): 33:1–33:44.

Borgida, A.; de Bruijn, J.; Franconi, E.; Seylan, I.; Straccia, U.; Toman, D.; and Weddell, G. E. 2010. On Finding Query Rewritings under Expressive Constraints. In *SEBD*, 426–437.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; Poggi, A.; Rodriguez-Muro, M.; Rosati, R.; Ruzzi, M.; and Savo, D. F. 2011. The MASTRO system for ontology-based data access. *Semantic Web*, 2(1): 43–53.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2005. *DL-Lite*: Tractable Description Logics for Ontologies. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, 602–607.

Calvanese, D.; de Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family. *J. of Automated Reasoning*, 39(3): 385–429.

Chaudhuri, S.; and Vardi, M. Y. 1997. On the Equivalence of Recursive and Nonrecursive Datalog Programs. *J. Comput. Syst. Sci.*, 54(1): 61–78.

Chomicki, J. 1995. Depth-Bounded Bottom-Up Evaluation of Logic Program. *J. Log. Program.*, 25(1): 1–31.

Clark, K. L. 1977. Negation as Failure. In *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, France, 1977*, 293–322.

Craig, W. 1957. Three uses of the Herbrand-Genzen theorem in relating model theory and proof theory. *Journal of Symbolic Logic*, 22: 269–285.

D'Agostino, G.; and Hollenberg, M. 2000. Logical Questions Concerning The mu-Calculus: Interpolation, Lyndon and Los-Tarski. *J. Symb. Log.*, 65(1): 310–332.

Eiter, T.; Ortiz, M.; Simkus, M.; Tran, T.; and Xiao, G. 2012. Query Rewriting for Horn-SHIQ Plus Rules. In *Proc. 26th AAAI Conference on Artificial Intelligence*.

Fitting, M. 1996. *First-Order Logic and Automated Theorem Proving, Second Edition*. Graduate Texts in Computer Science. Springer Publishers.

Fürer, M. 1981. Alternation and the Ackermann Case of the Decision Problem. *L'Enseignement Math.*, 27: 137–162.

Hansen, P.; Lutz, C.; Seylan, I.; and Wolter, F. 2015. Efficient Query Rewriting in the Description Logic EL and Beyond. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, 3034–3040.

Horrocks, I.; Sattler, U.; Tessaris, S.; and Tobies, S. 2000. How to Decide Query Containment Under Constraints Using a Description Logic. In *Logic for Programming and Automated Reasoning, 7th International Conference, LPAR 2000, Reunion Island, France, November 11-12, 2000, Proceedings*, 326–343.

Hudek, A. K.; Toman, D.; and Weddell, G. E. 2015. On Enumerating Query Plans Using Analytic Tableau. In *Automated Reasoning with Analytic Tableaux and Related Methods - 24th International Conference, TABLEAUX 2015, Wrocław, Poland, September 21-24, 2015. Proceedings*, 339–354.

Hustadt, U.; Motik, B.; and Sattler, U. 2005. Data Complexity of Reasoning in Very Expressive Description Logics. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 466–471. Professional Book Center.

Kikot, S.; Kontchakov, R.; Podolskii, V. V.; and Zakharyaschev, M. 2012. Long Rewritings, Short Rewritings. In *Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7-10, 2012*.

Kontchakov, R.; Lutz, C.; Toman, D.; Wolter, F.; and Zakharyaschev, M. 2010. The Combined Approach to Query Answering in DL-Lite. In *Proc. KR*, 247–257.

Kontchakov, R.; Lutz, C.; Toman, D.; Wolter, F.; and Zakharyaschev, M. 2011. The Combined Approach to Ontology-Based Data Access. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2656–2661.

Lutz, C.; and Sabellek, L. 2017. Ontology-Mediated Querying with the Description Logic EL: Trichotomy and Linear Datalog Rewritability. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, 1181–1187.

Lutz, C.; Seylan, I.; Toman, D.; and Wolter, F. 2013. The Combined Approach to OBDA: Taming Role Hierarchies Using Filters. In *ISWC (1)*, 314–330.

Lutz, C.; Toman, D.; and Wolter, F. 2009a. Conjunctive Query Answering in the Description Logic $\mathcal{EL}$ Using a Relational Database System. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2070–2075.

Lutz, C.; Toman, D.; and Wolter, F. 2009b. Conjunctive Query Answering in the Description Logic EL Using a Relational Database System. In *Proc. IJCAI*, 2070–2075.

Lutz, C.; and Wolter, F. 2012. Non-Uniform Data Complexity of Query Answering in Description Logics. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012*.

McIntyre, S.; Borgida, A.; Toman, D.; and Weddell, G. E. 2019. On Limited Conjunctions and Partial Features in Parameter-Tractable Feature Logics. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, 2995–3002.

McIntyre, S.; Toman, D.; and Weddell, G. E. 2019. FunDL - A Family of Feature-Based Description Logics, with Applications in Querying Structured Data Sources. In *Description Logic, Theory Combination, and All That - Essays Dedicated to Franz Baader on the Occasion of His 60th Birthday*, 404–430.

Reiter, R. 1977. On Closed World Data Bases. In Gallaire, H.; and Minker, J., eds., *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, France, 1977*, Advances in Data Base Theory, 55–76. New York: Plemum Press.

St. Jacques, J.; Toman, D.; and Weddell, G. E. 2016. Object-Relational Queries over $\mathcal{CFDI}_{nc}$ Knowledge Bases: OBDA for the SQL-Literate. In *Proc. IJCAI*, 1258–1264.

Toman, D.; and Weddell, G. E. 2008. On Keys and Functional Dependencies as First-Class Citizens in Description Logics. *J. Aut. Reasoning*, 40(2-3): 117–132.

Toman, D.; and Weddell, G. E. 2011. *Fundamentals of Physical Design and Query Compilation*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

Toman, D.; and Weddell, G. E. 2013. Conjunctive Query Answering in $\mathcal{CFD}_{nc}$: A PTIME Description Logic with Functional Constraints and Disjointness. In *Australasian Conference on Artificial Intelligence*, 350–361.

Toman, D.; and Weddell, G. E. 2017. An Interpolation-based Compiler and Optimizer for Relational Queries (System design Report). In *IWIL@LPAR 2017 Workshop and LPAR-21 Short Presentations, Maun, Botswana, May 7-12, 2017*.

Toman, D.; and Weddell, G. E. 2020. First Order Rewritability for Ontology Mediated Querying in Horn-DLFD. In *Proceedings of the 33rd International Workshop on Description Logics (DL 2020)*, volume 2663 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Toman, D.; and Weddell, G. E. 2021. FO Rewritability for OMQ using Beth Definability and Interpolation. In *Proceedings of the 34th International Workshop on Description Logics (DL 2021)*, CEUR Workshop Proceedings.

Ullman, J. 1982. *Principles of Database Systems*. Computer Science Press.

Ullman, J. 1988. *Principles of Database and Knowledge-Base Systems*, volume 1. Computer Science Press.

Ullman, J. 1989. *Principles of Database and Knowledge-Base Systems*, volume 2. Computer Science Press.