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) 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 ALC family. Dialects in the FunDL family are unusual in that they forgo roles and instead adopt 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-SHIQ in the ALC family, and when it is Horn-ALC 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 unifies 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-SHIQ and in Horn-ALC 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-ALC;
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.

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-SHIQ in the ALC family, and when it is Horn-ALC 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-SHIQ and in Horn-ALC 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-ALC;
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 ALC and FunDL dialects of DLs (Toman and Weddell 2020, 2021). FO rewritability for Horn logics in the 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 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-SHIQ, Horn-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-SHIQ and Horn-DLFD, respective members of the ALC and FunDL families that will concern us.

Definition 1 (Concepts in 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 ∈ PC, or of the form C1 ⊓ C2 for conjunction, ⊓ for bottom, and ⊤ for top;

(for Horn-SHIQ) Roles R of the form P and P− (an inverse role) for P ∈ R, and additional concepts of the form ∀ R.C for value restriction, ∃ R.C for existential restriction, or either (≥ n R.C) or (≤ n R.C), where n ≥ 0, for a number restriction;

(for Horn-DLFD) Path functions Pf of the form id for identity or the form f1.f2...fk, where f1 ∈ F and k > 0, for function composition, and additional concepts of the form ∀ f.C for value restriction, ∃ f−1 for unqualified inverse, and C : Pf1,...,Pfk → Pf for a path functional dependency (PFD).

The semantics is with respect to a structure I = (ΔI,⊤I) in which ΔI is a domain of objects and ⊤I an interpretation function seeded by fixing the interpretations of primitive concept names A to be subsets of ΔI, primitive role names R to be subsets of ΔI × ΔI, primitive feature names f to be partial functions on ΔI, and individual names a to be elements of ΔI, 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-SHIQ KBs, we follow the definition in (Eiter et al. 2012), and for Horn-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-SHIQ KBs (Eiter et al. 2012)) A Horn-SHIQ knowledge base K consists of a TBox T and an ABox A. T in normal form consists of role subsumptions of the form R1 ⊆ R2 that define a role hierarchy, transitivity assertions trans(R), and concept subsumptions that adhere to one of the following forms:1

\[
\begin{align*}
A \sqcap B & \subseteq C, \\
A & \subseteq \forall R.B, \\
A & \subseteq \exists R.B, \\
A & \subseteq (\leq 1 R.B),
\end{align*}
\]

where A, B, C ∈ PC ⊔ {⊤, ⊥}. 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 T will mention only simple roles. An ABox 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 ≠ b.

A Horn-ALCHQI KB is a Horn-SHIQ KB without transitivity assertions.

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

\[
\begin{align*}
C & ::= A | \forall f.A | A \sqcap B \\
D & ::= A | \sqcap | \forall f.A | \exists f^{-1} | A : Pf1,...,Pfk \rightarrow Pf
\end{align*}
\]

The PFD concept constructor is, in addition, restricted to either the form (a) A : Pf : Pf1,...,Pfk → Pf, called a

1Note that subsumptions of the form “A1 ⊓ ... ⊓ An ⊆ 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 \( A \) consists of concept assertions, equality axioms and inequality axioms with the respective forms \( A(a), a.f = b, a = b, \) 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 SClence departments.

\[
\begin{align*}
\text{EMP} &\sqsubseteq \forall \text{supervisedBy.EMP} \\
\text{DEPT} &\sqsubseteq \forall \text{managedBy.EMP} \\
\text{MATH} &\sqsubseteq \text{DEPT} \\
\text{SCI} &\sqsubseteq \text{DEPT}
\end{align*}
\]

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 prefix normal form. As usual, we confute a conjunctive query with the set of its constituent atoms and a list of answer variables to simplify notation.

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

A CQ \( \varphi \) is also a notational variant of the formula \( \exists \bar{y} \land_{\varphi \in \psi} \varphi \) in which \( \bar{y} \) contains all variables appearing in \( \psi \) but not in \( \bar{x} \).\(^3\) We also omit set braces when explicitly listing atoms in \( \psi \) 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 \( \varphi = \{ \bar{x} \mid \psi \} \) a CQ. A certain answer to \( \varphi \) over \( \mathcal{K} \) is a tuple of constant symbols \( \bar{a} \) such that \( \mathcal{K} \models \varphi(\bar{a}) \) (where \( \varphi(\bar{a}) \) is short for \( \varphi[\bar{x} \mapsto \bar{a}] \)).

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 \( \varphi_T \) for every CQ \( \varphi \) such that, for every ABox \( A \) and tuple of constant symbols \( \bar{a} \), \( (\mathcal{T}, A) \models \varphi(\bar{a}) \iff (\mathcal{T}, A) \models \varphi_T(\bar{a}) \).

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 \{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 \( A : Pf_1 f Pf_2, \ldots, Pf_k \rightarrow Pf \ g \) (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 \( \bar{x} \). Indeed, one can add additional atoms \( \top(x_i) \) to ensure variables in \( \bar{x} \) also appear in \( \psi \), if desired, without any impact on the remaining results.
Proposition 10 (The Combined Combined Approach)
Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a consistent knowledge base and $\varphi$ a conjunctive query. Then there is a UCQ query $\varphi^T$ and a Datalog program $\Pi_T$, both of which can be effectively constructed from $\mathcal{T}$, such that

$$\mathcal{K} \models \varphi^T(\bar{a}) \iff \Pi_T(\mathcal{A}) \models \varphi^T(\bar{a})$$

for any tuple of constant symbols $\bar{a}$, and where $\Pi_T(\mathcal{A})$ is the minimal model of $\Pi_T$ 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 $\Pi_T$ to define an ABox completion over which the query $\varphi^T$, the rewriting of the original user query, is evaluated to compute the correct answers. The Proposition also shows that the data complexity of OMQ is complete for PTIME.

3 Uniform Rewritability

Note that the existence of $\varphi^T$ 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 $\Pi_T$.

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

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

In a completion of $\mathcal{A}$ all $a_i$ have to belong to the concept EMP for all $k \leq i \leq 2k$. 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 $\Pi_T$) The Datalog program $\Pi_T$ 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-\textsc{AcllQle} and their translation are given as follows:

$$\text{(consequences of } \mathcal{T} \text{) \hspace{1cm} (completion rule in } \Pi_T \text{)}$$

$$\begin{align*}
A_1 \sqcap A_2 &\subseteq B & C_B(x) &\leftarrow C_{A_1}(x), C_{A_2}(x) \\
A &\subseteq \forall R.B & C_B(x) &\leftarrow C_{A_1}(y), R_B(x, y) \\
\exists R.A &\subseteq B & C_B(x) &\leftarrow C_{A_1}(y), R_B(x, y) \\
R &\subseteq S & R_S(x, y) &\leftarrow R_B(x, y)
\end{align*}$$

For every primitive concept $B$ and role $R$, we introduce unary EDB predicates $P_B(x)$ and $P_B(x, y)$ together with additional clauses $C_B(x) \leftarrow P_B(x)$, $R_B(x, y) \leftarrow P_B(x, y)$, and $R_B(x, y) \leftarrow P_B(y, x)$ to account for explicit data of the form $(a)(a, b)$ in an ABox, and IDB predicates $C_B(x)$ and $R_B(x, y)$ corresponding to the completion of the ABox w.r.t. $\mathcal{T}$.

For Horn-\textsc{DlfD} the Datalog program $\Pi_T$ is obtained in an analogous fashion, where $P_f(x, y)$ serves as a synonym for an assertion $f(x) = y$ to conform with standard Datalog syntax:

$$\begin{align*}
\text{(consequences of } \mathcal{T} \text{) \hspace{1cm} (completion rule in } \Pi_T \text{)}
A_1 \sqcap A_2 &\subseteq B & C_B(x) &\leftarrow C_{A_1}(x), C_{A_2}(x) \\
A &\subseteq \forall f.B & C_B(x) &\leftarrow C_{A_1}(y), P_f(y, x) \\
\forall f.A &\subseteq B & C_B(x) &\leftarrow C_{A_1}(y), P_f(x, y)
\end{align*}$$

In the programs $\Pi_T$, we call the $C_B(x)$ and $R_B(x, y)$ predicates intensional (or IDB predicates) and denote the set of all these predicates in $\Pi_T$ by $\text{EDB}(\mathcal{T})$. Similarly, we call all the $P_f(\ldots)$ predicates extensional (or EDB) and denote the set of all these by $\text{EDB}(\mathcal{T})$.

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

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

$$\begin{align*}
C_{\text{EMP}}(x) &\leftarrow C_{\text{EMP}}(y), P_{\text{supervisedBy}}(y, x) \\
C_{\text{EMP}}(x) &\leftarrow C_{\text{EMP}}(y), P_{\text{managedBy}}(y, x) \\
C_{\text{DEPT}}(x) &\leftarrow C_{\text{MATH}}(x) \\
C_{\text{DEPT}}(x) &\leftarrow C_{\text{SCI}}(x)
\end{align*}$$

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

Definition 14 (Clark’s Completion $\Sigma_T$) The Clark’s Completion (Clark 1977) $\Sigma_T$ of $\Pi_T$ is given by a set of formulas

$$\begin{align*}
C_B(x) &\leftrightarrow P_B(x) \vee (\exists y. \alpha_1) \vee \ldots \vee (\exists y. \alpha_n) \\
R_B(x, y) &\leftrightarrow P_B(x, y) \vee \beta_1 \vee \ldots \vee \beta_n
\end{align*}$$


\text{corresponding to clauses} $C_B(x) \leftrightarrow \alpha_i$ and $R_B(x, y) \leftrightarrow \beta_j$ in $\Pi_T$, grouped by the heads of the $\text{IDB}(\mathcal{T})$ predicates.

The bodies $\alpha_i$ ($\beta_j$) 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 $P \in \text{IDB}(\mathcal{T}) \cup \text{EDB}(\mathcal{T})$, ABox $\mathcal{A}$, and a vector of constants $\bar{a}$ (of appropriate arity) we have

- $\Pi_T \cup A_{db} \models P(\bar{a}) \text{ implies } \Sigma_T \cup A_{db} \models P(\bar{a})$, and
- The goal $P(\bar{a})$ finitely failing $\Pi_T \cup A_{db}$ implies $\Sigma_T \cup A_{db} \models \neg P(\bar{a})$,

where $A_{db}$ is the closed world variant of $A$, a set of ground facts such that all facts not in $A_{db}$ are false.\footnote{This too can be accomplished by Clark’s completion of $A$, 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:

\[
\begin{align*}
C_{\text{EMP}}(x) & \iff P_{\text{EMP}}(x) \\
& \lor (\exists y. C_{\text{EMP}}(y) \land P_{\text{supervisedBy}}(y, x)) \\
& \lor (\exists y. C_{\text{DEPT}}(y) \land P_{\text{managedBy}}(y, x)) \\
C_{\text{DEPT}}(x) & \iff P_{\text{DEPT}}(x) \\
& \lor C_{\text{MATH}}(x) \\
& \lor C_{\text{SCI}}(x) \\
C_{\text{MATH}}(x) & \iff P_{\text{MATH}}(x) \\
C_{\text{SCI}}(x) & \iff P_{\text{SCI}}(x)
\end{align*}
\]

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 \( \Pi_T \) instead of \( \Sigma_T \), 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 \( \Sigma \) 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 \( \Sigma \) such that \( M_1|_{L'} = M_2|_{L'} \), it holds that \( M_1 \models \varphi[a] \) iff \( M_2 \models \varphi[a] \) for all \( M_1, M_2 \), and it tuples of constants, and
2. \( \varphi \) is equivalent under \( \Sigma \) to a formula \( \psi \) in \( L' \) (we say \( \varphi \) is Beth definable over \( \Sigma \) and \( L' \)).

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

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

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

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

Given \( \Sigma_T \), one can now reformulate (1) in Proposition 17 as a logical implication problem by making a copy of all formulas of \( \Sigma_T \) in which all non-logical symbols not in \( \text{EDB}(T) \) are starred. Hence, the definability questions for \( C_A(x) \) and \( R_R(x, y) \) can be expressed as respective logical implication questions of the form:

\[
\Sigma_T \lor \Sigma_T' \models \forall x. C_A(x) \rightarrow C_A^*(x) \\
\Sigma_T \lor \Sigma_T' \models \forall x, y. R_R(x, y) \rightarrow R_R^*(x, y)
\]

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

Theorem 20: Let \( T \) be a TBox in one of the dialects considered. Then the existence of

1. the FO rewritability of the completion of \( A \) with respect to \( T \), and
2. the uniform query rewritability over \( T \) are decidable and in \( \text{EXPTIME} \).

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 \( \text{ALCQI} \). The second claim follows by observing that (i) definability of atomic queries implies definability of arbitrary UCQs using the 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 \( \Pi_T \) 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-\( \text{ALC} \) (for which the complexity of reasoning is \( \text{EXPTIME} \)-complete). 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 \( \varphi \) and \( \phi \) be FO formulas such that \( \models \varphi \rightarrow \phi \). Then there is an FO formula \( \psi \), called the Craig interpolant, containing only symbols common to \( \varphi \) and \( \phi \) such that \( \models \varphi \rightarrow \psi \) and \( \models \psi \rightarrow \phi \).

Moreover, the interpolant can be extracted, typically in linear time, from a proof of \( \models \varphi \rightarrow \phi \), 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 \( \varphi_T \) we get:

\[\text{CFD}nc \quad \text{and} \quad \text{CFD}I_{kc}^{-} \]
Theorem 22 Let \( K = (\mathcal{T}, \mathcal{A}) \) be a consistent knowledge base in either Horn-\(\mathcal{ALC}\mathcal{H}\mathcal{Q}\mathcal{I}\) or Horn-\(\mathcal{DLFD}\). Then the data complexity (i.e., in \(|\mathcal{A}|\)) of uniform conjunctive query answering is in \(\mathcal{AC}^0\) whenever the ABox completion with respect to \( \mathcal{T} \) is FO definable with respect to \( \Sigma_\mathcal{T} \).

Proof (sketch): Let \( \psi_P \) be FO definitions of \( P \in \text{EDB}(\mathcal{T}) \) w.r.t. \( \Sigma_\mathcal{T} \). Then
\[
\mathcal{K} \models \varphi(\overline{\alpha}) \iff \mathcal{A}_{db} \models \varphi_T[\psi_P[\overline{g}/\overline{x}]/P(\overline{y}) \mid P(\overline{y}) \in \text{Atoms}(\varphi)](\overline{a}).
\]
The claim follows since
\[
\varphi_T[\psi_P[\overline{g}/\overline{x}]/P(\overline{y}) \mid P(\overline{y}) \in \text{Atoms}(\varphi)]
\]
is an FO formula, in particular, a UCQ.

Example 23 Applying the interpolant construction on the Clarke’s completion in Example 16 will obtain:
\[
C_{\text{DEPT}}(x) \leftrightarrow (P_{\text{DEPT}}(x) \lor P_{\text{MATH}}(x) \lor P_{\text{SCI}}(x)).
\]
Note that there is no interpolant for \( C_{\text{EMP}} \).

Query-specific Rewritability. Our approach also provides tools for deciding the non-uniform query-specific problems. Indeed, although one can explicitly construct \( \varphi_T \), 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 \( \varphi_T \) is a UCQ. This yields our desired result:

Theorem 24 Let \( \mathcal{T} \) be a TBox and \( \varphi \) a CQ. Then the following are equivalent:

1. \( \varphi \) is FO rewritable with respect to \( \mathcal{T} \) and \( \Sigma_\mathcal{T} \cup \Sigma^*_\mathcal{T} \models \forall \overline{x}.P(\overline{x}) \rightarrow P^*(\overline{x}) \)
   for all \( P(\overline{x}) \in \text{Atoms}(\varphi_T) \cap \text{IDB}(\mathcal{T}) \).

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 \( \{ x \mid \text{DEPT}(x) \} \) is FO rewritable (follows from Example 23), while \( \{ x \mid \text{EMP}(x) \} \) 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-\(\mathcal{DLFD}\), 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 \( \varphi_T \).

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 \( P_B(x) \leftarrow C_A(x), P_B(x) \leftarrow \exists y.C_A(y), P_B(x,y), \) and \( P_B(x) \leftarrow \exists y.C_A(y), P_B(x,y), \) respectively, to the Clark’s completion.

Augmenting \( \Sigma_T \) 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:
\[
C_{\text{DEPT}}(x) \leftrightarrow (P_{\text{DEPT}}(x) \lor P_{\text{MATH}}(x) \lor P_{\text{SCI}}(x))
\]
It is 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-\(\mathcal{DLFD}\), 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 \( \varphi_T \).

Definition 27 (Logic Program for Horn-\(\mathcal{ALC}\) TBox)
(1) \( \text{entailed by } \mathcal{T} \)
(2) \( \text{completion rule in } \Pi_\mathcal{T} \)
\[
A_1 \sqcap A_2 \sqsubseteq B \quad C_B(x) \leftarrow C_{A_1}(x), C_{A_2}(x)
\]
\[
A \sqsubseteq \forall R.B \quad C_B(x) \leftarrow C_A(y), R_B(y,x)
\]
\[
\exists R.A \sqsubseteq B \quad C_B(x) \leftarrow C_A(y), R_B(x,y)
\]
\[
A \sqsubseteq \exists R.B \quad R_B(x,f_R(x)) \leftarrow C_A(x), \text{ and } C_B(f_R(x)) \leftarrow C_A(x)
\]

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-\(\mathcal{ALC}\mathcal{H}\mathcal{Q}\mathcal{E}\) as well.
Note that the construction of a Datalog program in Section 3 omitted the last rule (for \( \Lambda \sqsubseteq \exists R.B \)) 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 \( \Lambda \sqsubseteq \exists R.B \) can be equivalently written as

\[
\begin{align*}
R_H(x, y) &\leftrightarrow y = f_R(x), C_A(x) \\
C_B(x) &\leftrightarrow x = f_R(y), C_A(y)
\end{align*}
\]

as shown in (Clark 1977). Now, since the heads of the clauses is no longer a Datalog program, to define the Clark's completion, we need Skolem functions.

The clauses stemming from the existential restrictions subsumption has been accommodated by query reformulation. Theorem 28 without any major impact on the query reformulation \( \varphi_T \) in Proposition 10.

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 \( \varphi_T \).
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 Sabelleke 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 \( \mu \)-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.

### 6 Summary and Extensions

In this section, we briefly discuss several common extensions of Horn-\( \Lambda \sqsubseteq \) 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 \( \Pi_T \) 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 \( \varphi_T \) 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 Sabelleke 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 \( \mu \)-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


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.


