Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

From Conjunctive Queries to Instance Queries in Ontology-Mediated Querying

Cristina Feier[1], **Carsten Lutz**[1], **Frank Wolter**[2]
[1] University of Bremen, Germany
[2] University of Liverpool, UK
feier@uni-bremen.de, clu@uni-bremen.de, wolter@liverpool.ac.uk

Abstract

We consider ontology-mediated queries (OMQs) based on expressive description logics of the $\mathcal{ALC}$ family and (unions) of conjunctive queries, studying the rewritability into OMQs based on instance queries (IQs). Our results include exact characterizations of when such a rewriting is possible and tight complexity bounds for deciding rewritability. We also give a tight complexity bound for the related problem of deciding whether a given MMSNP sentence is equivalent to a CSP.

1 Introduction

An ontology-mediated query (OMQ) is a database-style query enriched with an ontology that contains domain knowledge, aiming to deliver more complete answers [Calvanese *et al.*, 2009; Bienvenu *et al.*, 2014; Bienvenu and Ortiz, 2015]. In OMQs, ontologies are often formulated in a description logic (DL) and query languages of interest include conjunctive queries (CQs), unions of conjunctive queries (UCQs), and instance queries (IQs). While CQs and UCQs are widely known query languages that play a fundamental role also in database systems and theory, IQs are more closely linked to DLs. In fact, an IQ takes the form $C(x)$ with C a concept formulated in the DL that is also used for the ontology, and thus the expressive power of IQs depends on the ontology language. OMQs based on (U)CQs are more powerful than OMQs based on IQs as the latter only serve to return all objects from the data that are instances of a given class.

It is easy to see that IQs can express tree-shaped CQs with a single answer variable as well as unions thereof. In fact, this observation has been used in many technical constructions in the area, see for example [Calvanese *et al.*, 1998; Glimm *et al.*, 2008; Lutz, 2008; Eiter *et al.*, 2012a]. Intriguingly, though, it was observed by Zolin [2007] that tree-shaped CQs are not the limit of IQ-rewritability when we have an expressive DL such as $\mathcal{ALC}$ or $\mathcal{ALCI}$ at our disposal. For example, the CQ $r(x, x)$, which asks to return all objects from the data that are involved in a reflexive r-loop, can be rewritten into the equivalent $\mathcal{ALC}$-IQ $P \rightarrow \exists r.P(x)$. Here, P behaves like a monadic second-order variable due to the open-world assumption made for OMQs: we are free to interpret P in any possible way and when making P true at an object we are forced to make also $\exists r.P$ true if and only if the object is involved in a reflexive r-loop. It is an interesting question, raised in [Zolin, 2007; Kikot and Zolin, 2013; Kikot *et al.*, 2013], to precisely characterize the class of CQs that are rewritable into IQs. An important step into this direction has been made by Kikot and Zolin [2013] who identify a large class of CQs that are rewritable into IQs: a CQ is rewritable into an $\mathcal{ALCI}$-IQ if it is connected and every cycle passes through the (only) answer variable; for rewritability into an $\mathcal{ALC}$-IQ, one additionally requires that all variables are reachable from the answer variable in a directed sense. It remained open whether these classes are depleting, that is, whether they capture all CQs that are IQ-rewritable.

There are two additional motivations to study the stated question. The first one comes from concerns about the practical implementation of OMQs. When the ontology is formulated in a more inexpressive 'Horn DL', OMQ evaluation is possible in PTIME data complexity and a host of techniques for practically efficient OMQ evaluation is available, see for example [Pérez-Urbina *et al.*, 2010; Eiter *et al.*, 2012b; Trivela *et al.*, 2015; Lutz *et al.*, 2009]. In the case of expressive DLs such as $\mathcal{ALC}$ and $\mathcal{ALCI}$, OMQ evaluation is CONP-complete in data complexity and efficient implementation is much more challenging. In particular, there are hardly any systems that fully support such OMQs when the actual queries are (U)CQs. In contrast, the evaluation of OMQs based on (expressive DLs and) IQs is supported by several systems such as Pellet, Hermit, and PAGOdA [Sirin *et al.*, 2007; Glimm *et al.*, 2014; Zhou *et al.*, 2015]. For this reason, rewriting (U)CQs into IQs has been advocated in [Zolin, 2007; Kikot and Zolin, 2013; Kikot *et al.*, 2013] as an approach towards efficient OMQ evaluation with expressive DLs and (U)CQs. The experiments and optimizations reported in [Kikot *et al.*, 2013] show the potential (and challenges) of this approach.

The second motivation stems from the connection between OMQs and constraint satisfaction problems (CSPs) [Bienvenu *et al.*, 2014; Lutz and Wolter, 2017]. Let $(\mathcal{L}, \mathcal{Q})$ denote the class of OMQs based on ontologies formulated in the DL $\mathcal{L}$ and the query language $\mathcal{Q}$. It was observed in [Bienvenu *et al.*, 2014] that $(\mathcal{ALCI}, \mathrm{IQ})$ is closely related to the complement of CSPs while $(\mathcal{ALCI}, \mathrm{UCQ})$ is closely related to the complement of the logical generalization MMSNP of CSP; we further remark that MMSNP is a notational variant

of the complement of (Boolean) monadic disjunctive Datalog. Thus, characterizing OMQs from $(\mathcal{ALCI}, \text{UCQ})$ that are rewritable into $(\mathcal{ALCI}, \text{IQ})$ is related to characterizing MMSNP sentences that are equivalent to a CSP, and we also study the latter problem. In fact, the main differences to the OMQ case are that unary queries are replaced with Boolean ones and that predicates can have unrestricted arity.

The main aim of this paper is to study the rewritability of OMQs from $(\mathcal{L}, \text{(U)CQ})$ into OMQs from $(\mathcal{L}, \text{IQ})$, considering as $\mathcal{L}$ the basic expressive DL $\mathcal{ALC}$ as well as extensions of $\mathcal{ALC}$ with inverse roles, role hierarchies, the universal role, and functional roles. We provide precise characterizations, tight complexity bounds for deciding whether a given OMQ is rewritable, and show how to construct the rewritten query when it exists. In fact, we prove that the classes of CQs from [Kikot and Zolin, 2013] are depleting, but we go significantly beyond that: while [Zolin, 2007; Kikot and Zolin, 2013; Kikot *et al.*, 2013] aim to find IQ-rewritings that work for *any* ontology, we consider the more fine-grained question of rewriting into an IQ an OMQ $(\mathcal{T}, \Sigma, q(x))$ where $\mathcal{T}$ is a DL TBox formalizing the ontology, Σ is an ABox signature, and $q(x)$ is the actual query. The 'any ontology' setup then corresponds to the special case where $\mathcal{T}$ is empty and Σ is full. However, gaving a non-empty TBox or a non-full ABox signature results in additional (U)CQs to become rewritable. While we admit modification of the TBox during rewriting, it turns out that this is mostly unnecessary: only in some rather special cases, a moderate *extension* of the TBox pays off. All this requires non-trivial generalizations of the query classes and IQ-constructions from [Kikot and Zolin, 2013]. Our completeness proofs involve techniques that stem from the connection between OMQs and CSP such as a lemma about ABoxes of high girth due to Feder and Vardi [1998]. The rewritings we construct are of polynomial size when we work with the empty TBox, but can otherwise become exponential in size.

Regarding IQ-rewritability as a decision problem, we show NP-completeness for the case of the empty TBox. This can be viewed as an underapproximation for the case with non-empty TBox and ABox signature. With non-empty TBoxes, complexities are higher. When the ABox signature is full, we obtain 2ExpTime-completeness for DLs with inverse roles and an ExpTime lower bound and a coNExpTime upper bound for DLs without inverse roles. With unrestricted ABox signature, the problem is 2NExpTime-complete for DLs with inverse roles and NExpTime-hard (and in 2NExpTime) for DLs without inverse roles. All lower bounds hold for CQs and all upper bounds capture UCQs. We also prove that it is 2NExpTime-complete to decide whether a given MMSNP sentence is equivalent to a CSP. This problem was known to be decidable [Madelaine and Stewart, 2007], but the complexity was open.

We also consider $\mathcal{ALCIF}$, the extension of $\mathcal{ALCI}$ with functional roles, for which IQ-rewritability turns out to be undecidable and much harder to characterize. We give a rather subtle characterization for the case of the empty TBox and full ABox signature and show that the decision problem is then decidable and NP-complete. Since it is not clear how to apply CSP techniques, we use an approach based on ultrafilters, starting from what was done for $\mathcal{ALC}$ without functional roles in [Kikot and Zolin, 2013].

Full proofs are in the appendix, available at http://www.informatik.uni-bremen.de/tdki/research/papers.html.

2 Preliminaries

We use standard description logic notation and refer to [Baader *et al.*, 2017] for full details. In contrast to the standard DL literature, we carefully distinguish between the *concept language* and the *TBox language*. We consider four concept languages. Recall that $\mathcal{ALC}$*-concepts* are formed according to the syntax rule

$$C, D ::= A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists r.C \mid \forall r.C$$

where A ranges over *concept names* and r over *role names*. As usual, we use $C \to D$ as an abbreviation for $\neg C \sqcup D$. $\mathcal{ALCI}$*-concepts* additionally admit the use of *inverse roles* r^- in concept constructors $\exists r^-.C$ and $\forall r^-.C$. With a *role*, we mean a role name or an inverse role. $\mathcal{ALC}^u$*-concepts* additionally admit the use of the *universal role* u in concept constructors $\exists u.C$ and $\forall u.C$. In $\mathcal{ALCI}^u$*-concepts*, both inverse roles and the universal role are admitted.

We now introduce several TBox languages. For $\mathcal{L}$ one of the four concept languages introduced above, an $\mathcal{L}$*-TBox* is a finite set of *concept inclusions* $C \sqsubseteq D$ where C and D are $\mathcal{L}$ concepts. So each concept language also serves as a TBox language, but there are additional TBox languages of interest. We include the letter $\mathcal{H}$ in the name of a TBox language to indicate that *role inclusions* $r \sqsubseteq s$ are also admitted in the TBox and likewise for the letter $\mathcal{F}$ and *functionality assertions* $\mathsf{func}(r)$ where in both cases r, s are role names or inverse roles in case that the concept language used admits inverse roles. So it should be understood, for example, what we mean with an $\mathcal{ALCHI}^u$-TBox and an $\mathcal{ALCFI}$-TBox. As usual, the semantics is defined in terms of interpretations, which take the form $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I}$ a non-empty *domain* and $\cdot^\mathcal{I}$ an *interpretation function*. An interpretation is a *model* of a TBox $\mathcal{T}$ if it satisfies all inclusions and assertions in $\mathcal{T}$, defined in the usual way. We write $\mathcal{T} \models r \sqsubseteq s$ if every model of $\mathcal{T}$ also satisfies the role inclusion $r \sqsubseteq s$.

An *ABox* is a set of *concept assertions* $A(a)$ and *role assertions* $r(a, b)$ where A is a concept name, r a role name, and a, b are *individual names*. We use $\mathsf{ind}(\mathcal{A})$ to denote the set of all individual names that occur in $\mathcal{A}$. An interpretation is a *model* of an ABox $\mathcal{A}$ if it *satisfies* all concept and role assertions in $\mathcal{A}$, that is, $a \in A^\mathcal{I}$ when $A(a)$ is in $\mathcal{A}$ and $(a, b) \in r^\mathcal{I}$ when $r(a, b)$ is in $\mathcal{A}$. An ABox is *consistent with a TBox* $\mathcal{T}$ if $\mathcal{A}$ and $\mathcal{T}$ have a common model. A *signature* Σ is a set of concept and role names. We use $\mathsf{sig}(\mathcal{T})$ to denote the set of concept and role names that occur in the TBox $\mathcal{T}$, and likewise for other syntactic objects such as ABoxes. A Σ*-ABox* is an ABox $\mathcal{A}$ such that $\mathsf{sig}(\mathcal{A}) \subseteq \Sigma$.

A *conjunctive query (CQ)* is of the form $q(\mathbf{x}) = \exists \mathbf{y}\, \varphi(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ and $\mathbf{y}$ are tuples of variables and $\varphi(\mathbf{x}, \mathbf{y})$ is a conjunction of *atoms* of the form $A(x)$ or $r(x, y)$ with A a concept name, r a role name, and $x, y \in \mathbf{x} \cup \mathbf{y}$. We call $\mathbf{x}$ the *answer variables* of $q(\mathbf{x})$ and $\mathbf{y}$ *quantified variables*. For purposes of uniformity, we use $r^-(x, y)$ as an alternative notation to denote an atom $r(y, x)$ in a CQ. In fact, when

speaking about *an atom* $r(x, y)$ *in a CQ* $q(\mathbf{x})$, we generally also include the case that $r = s^-$ and $s(y, x)$ is the actual atom in $q(\mathbf{x})$, unless explicitly noted otherwise. Every CQ $q(\mathbf{x}) = \exists \mathbf{y}\, \varphi(\mathbf{x}, \mathbf{y})$ gives raise to a directed graph G_q whose nodes are the elements of $\mathbf{x} \cup \mathbf{y}$ and that contains an edge from x to y if $\varphi(\mathbf{x}, \mathbf{y})$ contains an atom $r(x, y)$. The corresponding undirected graph is denoted G_q^u (it might contain self loops). We can thus use standard terminology from graph theory to CQs, saying for example that a CQ is *connected*. A *homomorphism* from $q(\mathbf{x})$ to an interpretation $\mathcal{I}$ is a function $h : \mathbf{x} \cup \mathbf{y} \to \Delta^{\mathcal{I}}$ such that $h(x) \in A^{\mathcal{I}}$ for every atom $A(x)$ of $q(\mathbf{x})$ and $(h(x), h(y)) \in r^{\mathcal{I}}$ for every atom $r(x, y)$ of $q(\mathbf{x})$. We write $\mathcal{I} \models q(\mathbf{a})$ and call $\mathbf{a}$ an *answer to* $q(\mathbf{x})$ *on* $\mathcal{I}$ if there is a homomorphism from $q(\mathbf{x})$ to $\mathcal{I}$ with $h(\mathbf{x}) = \mathbf{a}$.

A *union of conjunctive queries (UCQ)* $q(\mathbf{x})$ is a disjunction of one or more CQs that all have the same answer variables $\mathbf{x}$. We say that a UCQ is *connected* if every CQ in it is. The *arity* of a (U)CQ is the number of answer variables in it. For $\mathcal{L} \in \{\mathcal{ALC}, \mathcal{ALCI}, \mathcal{ALC}^u, \mathcal{ALCI}^u\}$, an *$\mathcal{L}$-instance query ($\mathcal{L}$-IQ)* takes the form $C(x)$ where C is an $\mathcal{L}$ concept and x a variable. We write $\mathcal{I} \models C(a)$ if $a \in C^{\mathcal{I}}$. All instance queries have arity 1.

An *ontology-mediated query (OMQ)* takes the form $Q = (\mathcal{T}, \Sigma, q(\mathbf{x}))$ with $\mathcal{T}$ a TBox, $\Sigma \subseteq \mathsf{sig}(\mathcal{T}) \cup \mathsf{sig}(q)$ an ABox signature, and $q(\mathbf{x})$ a query.[1] The *arity* of Q is the arity of $q(\mathbf{x})$ and Q is *Boolean* if it has arity zero. When Σ is $\mathsf{sig}(\mathcal{T}) \cup \mathsf{sig}(q)$, then for brevity we denote it with Σ_{full} and speak of the *full ABox signature*. Let $\mathcal{A}$ be a Σ-ABox. A tuple $\mathbf{a} \in \mathsf{ind}(\mathcal{A})$ is an *answer* to Q on $\mathcal{A}$ if $\mathcal{I} \models q(\mathbf{a})$ for all models $\mathcal{I}$ of $\mathcal{A}$ and $\mathcal{T}$. We say that Q is *empty* if for all Σ-ABoxes $\mathcal{A}$, there is no answer to Q on $\mathcal{A}$. Let Q_1, Q_2 be OMQs, $Q_i = (\mathcal{T}_i, \Sigma, q_i(\mathbf{x}))$ for $i \in \{1, 2\}$. Then Q_1 is *contained* in Q_2, written $Q_1 \subseteq Q_2$, if for all Σ-ABoxes $\mathcal{A}$, every answer to Q_1 on $\mathcal{A}$ is also an answer to Q_2 on $\mathcal{A}$. Further, Q_1 and Q_2 are *equivalent*, written $Q_1 \equiv Q_2$, if $Q_1 \subseteq Q_2$ and $Q_2 \subseteq Q_1$.

We use $(\mathcal{L}, \mathcal{Q})$ to refer to the *OMQ language* in which the TBox is formulated in the language $\mathcal{L}$ and where the actual queries are from the language $\mathcal{Q}$, such as in $(\mathcal{ALCF}, \text{UCQ})$. For brevity, we generally write $(\mathcal{L}, \text{IQ})$ instead of $(\mathcal{L}, \mathcal{L}'\text{-IQ})$ when $\mathcal{L}'$ is the concept language underlying the TBox language $\mathcal{L}$, so for example $(\mathcal{ALCHI}, \text{IQ})$ is short for $(\mathcal{ALCHI}, \mathcal{ALCI}\text{-IQ})$.

Definition 1. Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language. An OMQ $Q = (\mathcal{T}, \Sigma, q(\mathbf{x}))$ is *$(\mathcal{L}, \mathcal{Q})$-rewritable* if there is an OMQ Q' from $(\mathcal{L}, \mathcal{Q})$ such that the answers to Q and to Q' are identical on any Σ-ABox that is consistent with $\mathcal{T}$. In this case, we say that Q *is rewritable* into Q' and call Q' a *rewriting* of Q.

Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ-language. *IQ-rewritability in* $(\mathcal{L}, \mathcal{Q})$ is the problem to decide whether a given (unary) OMQ $Q = (\mathcal{T}, \Sigma, q(x))$ from $(\mathcal{L}, \mathcal{Q})$ is $(\mathcal{L}, \text{IQ})$-rewritable; for brevity, we simply speak of *IQ-rewritability* of Q when this is the case. The following examples show that IQ-rewritability of Q depends on several factors. All claims made are sanctioned by results established in this paper.

Example 2. (1) IQ-rewritability depends on the topology of the actual query. Let $q_1(x) = r(x, x)$. The OMQ $(\emptyset, \Sigma_{\mathsf{full}}, q_1(x))$ is rewritable into the OMQ $(\emptyset, \Sigma_{\mathsf{full}}, C(x))$ from $(\mathcal{ALC}, \text{IQ})$ where C is $P \to \exists r.P$. In contrast, let $q_2(x) = \exists y\, s(x, y) \wedge r(y, y)$. The OMQ $(\emptyset, \Sigma_{\mathsf{full}}, q_2(x))$ is not rewritable into an OMQ from $(\mathcal{ALCI}, \text{IQ})$.

(2) If we are not allowed to extend the TBox, IQ-rewritability depends on whether or not inverse roles are available. Let $\Sigma = \{r, s\}$ and $q(x) = \exists y\, r(y, x) \wedge s(y, x)$. The OMQ $Q = (\emptyset, \Sigma, q(x))$ is rewritable into the OMQ $(\emptyset, \Sigma, C(x))$ from $(\mathcal{ALCI}, \text{IQ})$ where C is $P \to \exists r^-.\exists s.P$. Q is also rewritable into the OMQ $(\mathcal{T}, \Sigma, C'(x))$ from $(\mathcal{ALC}, \text{IQ})$ where $\mathcal{T} = \{\exists s.P \sqsubseteq \forall r.P'\}$, and C is $P \to P'$, but it is not rewritable into any OMQ $(\mathcal{T}, \Sigma, C''(x))$ from $(\mathcal{ALC}, \text{IQ})$ with $\mathcal{T} = \emptyset$.

(3) IQ-rewritability depends on the TBox. Let $q(x) = \exists x_1 \exists y_1 \exists y_2 \exists z\, A(x) \wedge r(x, x_1) \wedge r(x_1, y_1) \wedge r(x_1, y_2) \wedge r(y_1, z) \wedge r(y_2, z) \wedge B_1(y_1) \wedge B_2(y_2)$. The OMQ $(\emptyset, \Sigma_{\mathsf{full}}, q(x))$ is not rewritable into an OMQ from $(\mathcal{ALCI}, \text{IQ})$. Let $\mathcal{T} = \{A \sqsubseteq \exists r.\exists r.(B_1 \sqcap B_2 \sqcap \exists r.\top)\}$. The OMQ $(\mathcal{T}, \Sigma_{\mathsf{full}}, q(x))$ is rewritable into the OMQ $(\mathcal{T}, \Sigma_{\mathsf{full}}, A(x))$ from $(\mathcal{ALC}, \text{IQ})$.

(4) IQ-rewritability depends on the ABox signature. Let $q(x)$ be the CQ from (3) without the atom $A(x)$ and let $\mathcal{T}$ be as in (3). The OMQ $(\mathcal{T}, \Sigma_{\mathsf{full}}, q(x))$ is not rewritable into an OMQ from $(\mathcal{ALCI}, \text{IQ})$. Let $\Sigma = \{A\}$. The OMQ $(\mathcal{T}, \Sigma, q(x))$ is rewritable into the OMQ $(\mathcal{T}, \Sigma, A(x))$ from $(\mathcal{ALC}, \text{IQ})$.

Note that we are allowed to completely rewrite the TBox when constructing IQ-rewritings, which might seem questionable from a practical perspective. Fortunately, though, it turns out the TBox can always be left untouched or, in some rare cases, only needs to be slightly extended. Also note that an alternative definition of IQ-rewritability obtained by dropping the restriction to ABoxes consistent with $\mathcal{T}$ in Definition 1. All results obtained in this paper hold under both definitions. We comment on this throughout the paper and refer to the alternative version as *unrestricted IQ-rewritability*.

3 Characterizations

We aim to provide characterizations of OMQs that are IQ-rewritable. On the one hand, these characterizations clarify which OMQs are IQ-rewritable and which are not. On the other hand, they form the basis for deciding IQ-rewritability. We first concentrate on the case of DLs (and IQs) with inverse roles and then move on to DLs without inverse roles. In the final part of this section, we consider the case where the TBox is empty, both with and without inverse roles.

3.1 The Case With Inverse Roles

To state the characterization, we need some preliminaries. Let $q(x)$ be a CQ. A *cycle* in $q(x)$ is a sequence of non-identical atoms $r_0(x_0, x_1), \ldots, r_{n-1}(x_{n-1}, x_n)$ in $q(x)$, $n \geq 1$, where[2]

1. $r_0, \ldots, r_{n-1}$ are (potentially inverse) roles,
2. $x_i \neq x_j$ for $0 \leq i < j < n$, and $x_0 = x_n$.

[1] The requirement $\Sigma \subseteq \mathsf{sig}(\mathcal{T}) \cup \mathsf{sig}(q)$ is harmless since symbols in the ABox that are not from $\mathsf{sig}(\mathcal{T}) \cup \mathsf{sig}(q)$ do not affect answers.

[2] We require the atoms be non-identical to prevent $r(x_0, x_1)$, $r^-(x_1, x_0)$ from being a cycle (both atoms are identical).

The *length* of this cycle is n. We say that $q(x)$ is *x-acyclic* if every cycle in it passes through x and use $q^{\mathsf{con}}(x)$ to denote the result of restricting $q(x)$ to those atoms that only use variables reachable in G_q^u from x. Both notions are lifted to UCQs by applying them to every CQ in the UCQ. A *contraction* of $q(x)$ is a CQ obtained from $q(x)$ by zero or more variable identifications, where the identification of x with any other variable yields x.

Let $\mathcal{T}$ be an $\mathcal{ALCHI}^u$-TBox and $q(x)$ a UCQ. We use $q_{\mathsf{acyc}}(x)$ to denote the UCQ that consists of all x-acyclic CQs obtained by starting with a contraction of a CQ from $q(x)$ and then replacing zero or more atoms $r(y,z)$ with $s(y,z)$ when $\mathcal{T} \models s \sqsubseteq r$. We write $q^{\mathsf{con}}_{\mathsf{acyc}}(x)$ to denote $(q_{\mathsf{acyc}})^{\mathsf{con}}(x)$.

Theorem 3. *Let $\mathcal{L} \in \{\mathcal{ALCI}, \mathcal{ALCHI}\}$ and let $Q = (\mathcal{T}, \Sigma, q(x))$ be a unary OMQ from $(\mathcal{L}, \text{UCQ})$ that is non-empty. Then the following are equivalent:*

1. *Q is IQ-rewritable, that is, it is rewritable into an OMQ $Q' = (\mathcal{T}', \Sigma, C(x))$ from $(\mathcal{L}, \text{IQ})$;*
2. *Q is rewritable into an OMQ $Q' = (\mathcal{T}, \Sigma, C(x))$ from $(\mathcal{L}, \text{IQ})$;*
3. *$Q \equiv (\mathcal{T}, \Sigma, q^{\mathsf{con}}_{\mathsf{acyc}}(x))$.*

When $\mathcal{L}$ is replaced with $\mathcal{L}^u$, then the same equivalences hold except that $q^{\mathsf{con}}_{\mathsf{acyc}}$ is replaced with q_{acyc}.

Note that Theorem 3 excludes empty OMQs, but these are trivially IQ-rewritable. It implies that, in the considered cases, it is never necessary to modify the TBox when constructing an IQ-rewriting. Further, it emerges from the proof that it is never necessary to introduce fresh role names in the rewriting (while fresh concept names are crucial). Theorem 3 also applies to unrestricted IQ-rewritability (where also ABoxes are admitted that are inconsistent with the TBox from the OMQ): unrestricted IQ-rewritability trivially implies IQ-rewritability and the converse is an easy consequence of the fact that every OMQ that is IQ-rewritable has an IQ-rewriting based on the same TBox.

We now give some ideas about the proof of Theorem 3. The most interesting implication is "1 $\Rightarrow$ 3". A central step is to show that if $Q = (\mathcal{T}, \Sigma, q(x))$ is IQ-rewritable into an OMQ Q', then $Q \subseteq Q_{\mathsf{acyc}} := (\mathcal{T}, \Sigma, q_{\mathsf{acyc}}(x))$, that is, when $\mathcal{A} \models Q(a)$ for some Σ-ABox $\mathcal{A}$, then $\mathcal{A} \models Q_{\mathsf{acyc}}(a)$. To this end, we first construct from $\mathcal{A}$ a Σ-ABox $\mathcal{A}^g$ of high girth (that is, without small cycles) in a way such that (a) $\mathcal{A}^g$ homomorphically maps to $\mathcal{A}$ and (b) from $\mathcal{A} \models Q'(a)$ it follows that $\mathcal{A}^g \models Q'(a)$, thus $\mathcal{A}^g \models Q(a)$. Due to the high girth of $\mathcal{A}^g$ and exploiting (a variation of the) tree model property for $\mathcal{ALCHI}$, we can then show that $\mathcal{A}^g \models Q(a)$ implies $\mathcal{A}^g \models Q_{\mathsf{acyc}}(a)$. Because of (a), it follows that $\mathcal{A} \models Q_{\mathsf{acyc}}(a)$. In the direction "3 $\Rightarrow$ 2", we construct actual rewritings, based on the following lemma, an extension of a result of Kikot and Zolin [Kikot and Zolin, 2013] with TBoxes and ABox signatures (and UCQs instead of CQs).

Lemma 4. *Let $Q = (\mathcal{T}, \Sigma, q(x))$ be an OMQ from $(\mathcal{ALCHI}^u, \text{UCQ})$. Then*

1. *if $q(x)$ is x-acyclic and connected, then Q is rewritable into an OMQ $(\mathcal{T}, \Sigma, C(x))$ with $C(x)$ an $\mathcal{ALCI}$-IQ and*
2. *if $q(x)$ is x-acyclic, then Q is rewritable into an OMQ $(\mathcal{T}, \Sigma, C(x))$ with $C(x)$ an $\mathcal{ALCI}^u$-IQ.*

The size of the IQs $C(x)$ is polynomial in the size of $q(x)$.

We give the construction of the $\mathcal{ALCI}$-IQ $q'(x)$ in Point 1 of Lemma 4. Let $Q = (\mathcal{T}, \Sigma, q(x))$ be an OMQ from $(\mathcal{ALCHI}^u, \text{UCQ})$ with $q(x)$ x-acyclic and connected. To construct $q'(x)$, we first construct for each CQ $p(x)$ in $q(x)$ an *$\mathcal{ELI}$-concept* C_p, that is, an $\mathcal{ALCI}$-concept that uses only the constructors $\sqcap$, $\exists r.C$, and $\exists r^-.C$. In fact, since $p(x)$ is x-acyclic and connected, we can repeatedly choose and remove atoms of the form $r(x,y)$ that occur in a cycle in $p(x)$ and will eventually end up with a tree-shaped CQ $p'(x)$.[3] Here, *tree-shaped* means that the undirected graph $G^u_{p'}$ is a tree and that there are no multi-edges, that is, if $r(y,z)$ is an atom, then there is no atom $s(y,z)$ with $s \neq r$. Next, extend $p'(x)$ to obtain another tree-shaped CQ $p''(x)$ by taking a fresh concept name $P \notin \Sigma$, and adding $r(x',y)$ and $P(x')$ for each removed atom $r(x,y)$, x' a fresh variable. We can now view $p''(x)$ as an $\mathcal{ELI}$-concept C_p in the obvious way. The desired $\mathcal{ALCI}$-IQ $q'(x)$ is $(P \rightarrow \bigsqcup_{p(x) \text{ a CQ in } q(x)} C_p)(x)$.

3.2 The Case Without Inverse Roles

We consider OMQs whose TBoxes are formulated in a DL $\mathcal{L}$ that does not admit inverse roles. Note that inverse roles are then also not admitted in the IQ used in the rewriting. We first observe that this has less impact than one might expect: inverse roles in the IQ-rewriting can be eliminated and in fact Points 1 and 3 from Theorem 3 are still equivalent. However, there is also a crucial difference: unless the universal role is present, the elimination of inverse roles requires an extension of the TBox and thus the equivalence of Points 1 and 2 of Theorem 3 fails. In fact, this is illustrated by Point (2) of Example 2. We thus additionally characterize IQ-rewritability without modifying the TBox. We also show that, with the universal role, it is not necessary to extend the TBox.

We start with some preliminaries. An *extended conjunctive query (eCQ)* is a CQ that also admits atoms of the form $C(x)$, C a (potentially compound) concept, and *UeCQs* and *extended ABoxes (eABoxes)* are defined analogously. The semantics is defined in the expected way. Every eCQ $q(x)$ gives rise to an eABox $\mathcal{A}_q$ by viewing the variables in $q(x)$ as individual names and the atoms as assertions.

Let $q(x)$ be an eCQ. We use $\mathsf{dreach}(q)$ to denote the set of all variables reachable from x in the directed graph G_q and say that $q(x)$ is *x-accessible* if $\mathsf{dreach}(q)$ contains all variables. For V a set of variables from $q(x)$ that includes x, $q(x)|_V$ denotes the restriction of $q(x)$ to the atoms that use only variables from V.

Let $\mathcal{T}$ be an $\mathcal{ALC}$-TBox. An eCQ $p(x)$ is a *$\mathcal{T}$-decoration* of a CQ $q(x)$ if

1. $p(x)$ is obtained from $q(x)$ by adding, for each $y \in \mathsf{dreach}(q)$ and each subconcept C of $\mathcal{T}$, the atom $C(y)$ or the atom $\neg C(y)$;
2. the eABox $\mathcal{A}_p$ is consistent with $\mathcal{T}$.

[3] Note that x is the answer variable and recall that we might have $r = s^-$ and thus also choose atoms $s(y,x)$.

For a UCQ $q(x)$, we use $q^{\mathsf{deco}}(x)$ to denote the UeCQ that consists of all eCQs $p(x)|_{\mathsf{dreach}(p)}(x)$, where $p(x)$ is a $\mathcal{T}$-decoration of a CQ from $q(x)$. We write $q^{\mathsf{deco}}_{\mathsf{acyc}}(x)$ to denote $(q_{\mathsf{acyc}})^{\mathsf{deco}}(x)$. We now give the results announced above.

Theorem 5. *Let $\mathcal{L} \in \{\mathcal{ALC}, \mathcal{ALCH}\}$ and let $Q = (\mathcal{T}, \Sigma, q(x))$ be a unary OMQ from $(\mathcal{L}, UCQ)$ that is non-empty. Then the following are equivalent:*

1. *Q is rewritable into an OMQ from $(\mathcal{L}, IQ)$;*
2. *Q is rewritable into an OMQ $(\mathcal{T} \cup \mathcal{T}', \Sigma, C(x))$ from $(\mathcal{L}, IQ)$;*
3. *Q is rewritable into an OMQ from $(\mathcal{LI}, IQ)$;*

If $\Sigma = \Sigma_{\mathsf{full}}$, then the following are equivalent:

4. *Q is rewritable into an OMQ $Q' = (\mathcal{T}, \Sigma_{\mathsf{full}}, C(x))$ from $(\mathcal{L}, IQ)$;*
5. *$Q \equiv (\mathcal{T}, \Sigma_{\mathsf{full}}, q^{\mathsf{deco}}_{\mathsf{acyc}}(x))$.*

If, furthermore, $\mathcal{L}$ is replaced with $\mathcal{L}^u$ and $\mathcal{LI}$ with $\mathcal{LI}^u$, then Conditions 1 to 3 are further equivalent to:

6. *Q is rewritable into an OMQ $Q' = (\mathcal{T}, \Sigma, C(x))$ from $(\mathcal{L}^u, IQ)$.*

Characterizing IQ-rewritability in the case where $\mathcal{L} \in \{\mathcal{ALC}, \mathcal{ALCH}\}$, the TBox (is non-empty and) cannot be extended, and $\Sigma \neq \Sigma_{\mathsf{full}}$ remains an open problem.

In the directions "$3 \Rightarrow 2$", "$5 \Rightarrow 4$", and "$3 \Rightarrow 6$", we have to construct IQ-rewritings. This is done by starting with the rewriting from the proof of Lemma 4 and then modifying it appropriately. As in the case of Theorem 3, it is straightforward to see that all results stated in Theorem 5 also apply to unrestricted IQ-rewritability.

3.3 The Case of Empty TBoxes

We consider OMQs in which the TBox is empty as an important special case. Since it is then not interesting to have an ABox signature, this corresponds to the rewritability of (U)CQs into $\mathcal{L}$-instance queries, for some concept language $\mathcal{L}$ (and thus no OMQs are involved). The importance of this case is due to the fact that it provides an 'underapproximation' of the IQ-rewritability of OMQs, while also being easier to characterize and computationally simpler.

We say that an UCQ $q(x)$ is *$\mathcal{L}$-IQ-rewritable* if there is an $\mathcal{L}$-IQ $q'(x)$ that is equivalent to $q(x)$ in the sense that the OMQs $(\emptyset, \Sigma_{\mathsf{full}}, q(x))$ and $(\emptyset, \Sigma_{\mathsf{full}}, q'(x))$ are equivalent (and in passing, we define the equivalence between two UCQs in exactly the same way). The following proposition makes precise what we mean by underapproximation.

Proposition 6. *Let $\mathcal{L} \in \{\mathcal{ALC}, \mathcal{ALCI}, \mathcal{ALC}^u, \mathcal{ALCI}^u\}$. If a UCQ $q(x)$ is $\mathcal{L}$-IQ-rewritable, then so is any OMQ $(\mathcal{T}, \Sigma, q(x))$ from $(\mathcal{LH}, UCQ)$.*

Proposition 6 is essentially a corollary of Theorem 7 below. As illustrated by Case (3) of Example 2, its converse fails.

We now characterize IQ-rewritability in the case of the empty TBox. A *subquery* of a CQ $q(x)$ is a CQ $q'(x)$ obtained from $q(x)$ by dropping atoms. A *subquery* of a UCQ $q(x)$ is a UCQ obtained by including as a CQ at most one subquery of each CQ in $q(x)$.

Theorem 7. *Let $q(x)$ be a UCQ. Then*

1. *$q(x)$ is rewritable into an $\mathcal{ALCI}$-IQ iff there is a subquery $q'(x)$ of $q(x)$ that is x-acyclic, connected, and equivalent to $q(x)$;*
2. *$q(x)$ is rewritable into an $\mathcal{ALC}$-IQ iff there is a subquery $q'(x)$ of $q(x)$ that is x-acyclic, x-accessible, and equivalent to $q(x)$.*

When $\mathcal{L}$-IQs are replaced with $\mathcal{L}^u$-IQs, then the same equivalences hold except that connectedness/x-accessibility is dropped.

Note that Theorem 7 also characterizes rewritability of CQs; the query $q'(x)$ is then also a CQ rather than a UCQ. This is in contrast to Theorems 3 and 5 where the queries $q^{\mathsf{con}}_{\mathsf{acyc}}(x)$ and $q^{\mathsf{deco}}_{\mathsf{acyc}}(x)$ are UCQs even when the query $q(x)$ from the OMQ that we start with is a CQ. Another crucial difference is that $q^{\mathsf{con}}_{\mathsf{acyc}}(x)$ and $q^{\mathsf{deco}}_{\mathsf{acyc}}(x)$ can be of size exponential in the size of the original OMQ while the query $q'(x)$ in Theorem 7 is of size polynomial in the size of $q(x)$.

4 Complexity

We determine the complexity of deciding IQ-rewritability in various OMQ languages, based on the established characterizations and starting with the case of empty TBoxes.

Theorem 8. *For every $\mathcal{Q} \in \{CQ, UCQ\}$ and $\mathcal{L} \in \{\mathcal{ALC}, \mathcal{ALCI}, \mathcal{ALC}^u, \mathcal{ALCI}^u\}$, it is* NP*-complete to decide whether a given query from $\mathcal{Q}$ is $\mathcal{L}$-IQ-rewritable.*

The upper bound in Theorem 8 is by guessing the query $q'(x)$ from Theorem 7 and verifying that it satisfies the properties stated there. The lower bound is by a reduction from 3-colorability.

We next consider the case where TBoxes can be non-empty, starting with the assumption that the ABox signature is full since this results in (slightly) lower complexity.

Theorem 9. *Let $\mathcal{Q} \in \{CQ, UCQ\}$. For OMQs based on the full ABox signature, IQ-rewritability is*

1. EXPTIME*-hard in $(\mathcal{ALC}, \mathcal{Q})$ and in* coNEXPTIME *in $(\mathcal{ALCH}, \mathcal{Q})$ and*
2. 2EXPTIME*-complete in $(\mathcal{ALCI}, \mathcal{Q})$ and $(\mathcal{ALCHI}, \mathcal{Q})$.*

The lower bounds are by reduction from OMQ evaluation on ABoxes of the form $\{A(a)\}$, A a concept name, which is EXPTIME-complete in $(\mathcal{ALCH}, \text{CQ})$ and 2EXPTIME-complete in $(\mathcal{ALCHI}, \text{CQ})$ [Lutz, 2008]. The upper bounds are derived from the OMQ containment checks suggested by Condition 3 of Theorem 3 and Condition 4 of Theorem 5. Since we work with the full ABox signature, the non-emptiness condition from these theorems is void (there are no empty OMQs) and OMQ containment is closely related to OMQ evaluation, which allows us to derive upper bounds for the former from the latter; in fact, these bounds are exactly the ones stated in Theorem 9. We have to exercise some care, for two reasons: first, we admit UCQs as the actual query and thus the trivial reduction of OMQ containment to OMQ evaluation that is possible for CQs (which can be viewed as an ABox) does not apply. And second, we aim for upper bounds that exactly match the complexity of OMQ containment while

the UCQs $q^{\mathsf{con}}_{\mathsf{acyc}}(x)$ and $q^{\mathsf{deco}}_{\mathsf{acyc}}(x)$ involved in the containment checks are of exponential size. What rescues us is that each of the CQs in these UCQs is only of polynomial size.

We finally consider the case where the ABox signature is unrestricted.

Theorem 10. *IQ-rewritability is*

1. NEXPTIME*-hard in* $(\mathcal{ALC}, CQ)$ *and*
2. 2NEXPTIME*-complete in all of* $(\mathcal{ALCI}, CQ)$, $(\mathcal{ALCI}, UCQ)$, $(\mathcal{ALCHI}, CQ)$, $(\mathcal{ALCHI}, UCQ)$.

The lower bound in Point 1 is by reduction from OMQ emptiness in $(\mathcal{ALC}, \text{CQ})$, which is NEXPTIME-complete [Baader *et al.*, 2016]. For the one in Point 2, we use a reduction from OMQ containment, which is 2NEXPTIME-complete in $(\mathcal{ALCI}, \text{CQ})$ [Bourhis and Lutz, 2016]. The upper bounds are obtained by appropriate containment checks as suggested by our characterizations, and we again have to deal with UCQs with exponentially many CQs. Note that Theorem 10 leaves open the complexity of IQ-rewritability in $(\mathcal{ALC}, \text{CQ})$, between NEXPTIME and 2NEXPTIME. The same gap exists for OMQ containment [Bourhis and Lutz, 2016] as well as in the related problems of FO-rewritability and Datalog-rewritability [Feier *et al.*, 2017].

5 Functional Roles

We consider DLs with functional roles. A fundamental observation is that for the basic such DL $\mathcal{ALCF}$, IQ-rewritability is undecidable. This can be proved by a reduction from OMQ emptiness in $(\mathcal{ALCF}, \text{IQ})$ [Baader *et al.*, 2016].

Theorem 11. *In* $(\mathcal{ALCF}, CQ)$*, IQ-rewritability is undecidable.*

In the following, we show that decidability is regained in the case where the TBox is empty (apart from functionality assertions). This is challenging because functionality assertions have a strong and subtle impact on rewritability. As before, the only interesting ABox signature to be combined with 'empty' TBoxes is the full ABox signature. We use $\mathcal{F}$ to denote the TBox language in which TBoxes are sets of functionality assertions and concentrate on rewriting into IQs that may use inverse roles.

Example 12. Consider the CQ $p(x) = \exists y(s(x, y) \land r(y, y))$ from Point 1 of Example 2. Then $Q_s = (\mathcal{T}_s, \Sigma_{\mathsf{full}}, p(x))$ and $Q_r = (\mathcal{T}_r, \Sigma_{\mathsf{full}}, p(x))$ with $\mathcal{T}_w = \{\mathsf{func}(w)\}$ for $w \in \{r, s\}$ are both rewritable into an OMQ $(\mathcal{T}_w, \Sigma_{\mathsf{full}}, q_w(x))$ with $q_w(x)$ an $\mathcal{ALCI}$-IQ. The rewritings are neither trivial to find nor entirely easy to understand. In fact, for $q_s(x)$ we can use $\forall s.P \to \exists s.(P \to \exists r.P)$. For $q_r(x)$, we introduce three fresh concept names rather than a single one and use them in a way inspired by graph colorings:

$$q_r(x) = (\forall s. \bigsqcup_{1 \le i \le 3} P_i) \to (\exists s.(\bigsqcap_{1 \le i \le 3} (P_i \to \exists r.P_i)).$$

Before giving a characterization of rewritable queries, we introduce some preliminaries. Let $q(x)$ be a CQ and $\mathcal{T}$ an $\mathcal{ALCIF}$-TBox. A sequence $x_0, \dots, x_n$ of variables in $q(x)$ is a *functional path in* $q(x)$ *from* x_0 *to* x_n *w.r.t.* $\mathcal{T}$ if for all $i < n$ there is a role r such that $\mathsf{func}(r) \in \mathcal{T}$ and $r(x_i, x_{i+1})$ is in $q(x)$. We say that $q(x)$ is *f-acyclic w.r.t.* $\mathcal{T}$ if for every cycle $r_0(x_0, x_1), \dots, r_{n-1}(x_{n-1}, x_n)$ in $q(x)$, one of the following holds:

- there is a functional path in $q(x)$ from x to some x_i;
- $\mathsf{func}(r_i) \in \mathcal{T}$ or $\mathsf{func}(r_i^-) \in \mathcal{T}$ for all $i < n$ and there is a functional path $y_0, \dots, y_m$ in $q(x)$ with $x_0 = y_0 = y_m$ such that $\{x_0, \dots, x_{n-1}\} \subseteq \{y_0, \dots, y_m\}$.

We are now ready to state the characterization.

Theorem 13. *An OMQ* $Q = (\mathcal{T}, \Sigma_{\mathit{full}}, q(x))$ *from* $(\mathcal{F}, UCQ)$ *is rewritable into an OMQ from* $(\mathcal{F}, \mathcal{ALCI}\text{-}IQ)$ *iff there is a subquery* $q'(x)$ *of* $q(x)$ *that is f-acyclic, connected, and equivalent to* $q(x)$.

When $\mathcal{ALCI}$*-IQ is replaced with* $\mathcal{ALCI}^u$*-IQ, the same equivalence holds except that connectedness is dropped.*

The proof of Theorem 13 extends the ultrafilter construction from [Kikot and Zolin, 2013]. We remark that the "if" direction in Theorem 13 even holds for OMQs $Q = (\mathcal{T}, \Sigma, q(x))$ from $(\mathcal{ALCIF}, \text{UCQ})$. Thus, the case of the 'empty' TBox can again be seen as an underapproximation of the general case. We further remark that $\mathcal{T}$ remains unchanged in the construction of the IQ-rewritings and that the constructed rewritings are of polynomial size.

Theorem 14. *For OMQs from* $(\mathcal{F}, UCQ)$*, rewritability into* $(\mathcal{F}, \mathcal{ALCI}\text{-}IQ)$ *is* NP*-complete.*

6 MMSNP and CSP

Recall from the introduction that the OMQ languages studied in this paper are closely related to CSPs and their logical generalization MMSNP. In fact, the techniques used to establish the results in Sections 3 and 4 can be adapted to determine the complexity of deciding whether a given MMSNP sentence is equivalent to a CSP. In a nutshell, we prove that an MMSNP-sentence is equivalent to a CSP iff it is preserved under disjoint union and equivalent to a generalized CSP (a CSP with multiple templates), and that both properties can be reduced to containment between MMSNP sentences which is 2NEXPTIME-complete [Bourhis and Lutz, 2016]. The latter reduction involves constructing an MMSNP sentence φ_{acyc} that is reminiscent of the query q_{acyc} in Theorem 3. Full details are given in the appendix.

Theorem 15. *It is* 2NEXPTIME*-complete to decide whether a given MMSNP-sentence is equivalent to a CSP.*

7 Conclusion

We have made a leap forward in understanding the relation between (U)CQs and IQs in ontology-mediated querying. Interesting open problems include a characterization of IQ-rewritability for DLs with functional roles when the TBox is non-empty and characterizations for DLs with transitive roles. The remarks after Theorem 4 and 10 mention further problems left open. In addition, it would be worthwhile to continue the effort from [Kikot *et al.*, 2013] to understand the value of IQ-rewritings for the purposes of efficient practical implementation.

Acknowledgements

Cristina Feier and Carsten Lutz were supported by ERC Consolidator Grant 647289 CODA. Frank Wolter was supported by EPSRC UK grant EP/M012646/1.

References

[Baader *et al.*, 2016] Franz Baader, Meghyn Bienvenu, Carsten Lutz, and Frank Wolter. Query and predicate emptiness in ontology-based data access. *J. Artif. Intell. Res.*, 56:1–59, 2016.

[Baader *et al.*, 2017] Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[Bienvenu and Ortiz, 2015] Meghyn Bienvenu and Magdalena Ortiz. Ontology-mediated query answering with data-tractable description logics. In *Proc. of Reasoning Web*, volume 9203 of *LNCS*, pages 218–307. Springer, 2015.

[Bienvenu *et al.*, 2014] Meghyn Bienvenu, Balder ten Cate, Carsten Lutz, and Frank Wolter. Ontology-based data access: A study through disjunctive Datalog, CSP, and MMSNP. *ACM Trans. Database Syst.*, 39(4):33:1–33:44, 2014.

[Bourhis and Lutz, 2016] Pierre Bourhis and Carsten Lutz. Containment in monadic disjunctive datalog, MMSNP, and expressive description logics. In *Proc. of KR*, pages 207–216. AAAI Press, 2016.

[Calvanese *et al.*, 1998] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. On the decidability of query containment under constraints. In *Proc. of PODS1998*, pages 149–158. ACM Press, 1998.

[Calvanese *et al.*, 2009] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, and Riccardo Rosati. Ontologies and databases: The DL-Lite approach. In *Proc. of Reasoning Web 2009*, volume 5689 of *LNCS*, pages 255–356. Springer, 2009.

[Eiter *et al.*, 2012a] Thomas Eiter, Magdalena Ortiz, and Mantas Simkus. Conjunctive query answering in the description logic SH using knots. *J. Comput. Syst. Sci.*, 78(1):47–85, 2012.

[Eiter *et al.*, 2012b] Thomas Eiter, Magdalena Ortiz, Mantas Simkus, Trung-Kien Tran, and Guohui Xiao. Query rewriting for Horn-SHIQ plus rules. In *Proc. of AAAI*. AAAI Press, 2012.

[Feder and Vardi, 1998] Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1998.

[Feier *et al.*, 2017] Cristina Feier, Antti Kuusisto, and Carsten Lutz. Rewritability in monadic disjunctive datalog, MMSNP, and expressive description logics (invited talk). In *Proc. ICDT*, volume 68 of *LIPIcs*, pages 1:1–1:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[Glimm *et al.*, 2008] Birte Glimm, Carsten Lutz, Ian Horrocks, and Ulrike Sattler. Conjunctive query answering for the description logic SHIQ. *J. Artif. Intell. Res.*, 31:157–204, 2008.

[Glimm *et al.*, 2014] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: An OWL 2 reasoner. *J. of Autom. Reasoning*, 53(3):245–269, 2014.

[Kikot and Zolin, 2013] Stanislav Kikot and Evgeny Zolin. Modal definability of first-order formulas with free variables and query answering. *J. Applied Logic*, 11(2):190–216, 2013.

[Kikot *et al.*, 2013] Stanislav Kikot, Dmitry Tsarkov, Michael Zakharyaschev, and Evgeny Zolin. Query answering via modal definability with FaCT++: First blood. In *Proc. DL*, volume 1014 of *CEUR Workshop Proceedings*, pages 328–340. CEUR-WS.org, 2013.

[Lutz and Wolter, 2017] Carsten Lutz and Frank Wolter. The data complexity of description logic ontologies. *Logical Methods in Computer Science*, 13(4), 2017.

[Lutz *et al.*, 2009] Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In *Proc. IJCAI*, pages 2070–2075, 2009.

[Lutz, 2008] Carsten Lutz. The complexity of conjunctive query answering in expressive description logics. In *Proc. of IJCAR*, volume 5195 of *LNCS*, pages 179–193. Springer, 2008.

[Madelaine and Stewart, 2007] Florent R. Madelaine and Iain A. Stewart. Constraint satisfaction, logic and forbidden patterns. *SIAM J. Comput.*, 37(1):132–163, 2007.

[Pérez-Urbina *et al.*, 2010] Héctor Pérez-Urbina, Boris Motik, and Ian Horrocks. Tractable query answering and rewriting under description logic constraints. *JAL*, 8(2):186–209, 2010.

[Sirin *et al.*, 2007] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *J. Web Sem.*, 5(2):51 – 53, 2007.

[Trivela *et al.*, 2015] Despoina Trivela, Giorgos Stoilos, Alexandros Chortaras, and Giorgos B. Stamou. Optimising resolution-based rewriting algorithms for OWL ontologies. *J. Web Sem.*, 33:30–49, 2015.

[Zhou *et al.*, 2015] Yujiao Zhou, Bernardo Cuenca Grau, Yavor Nenov, Mark Kaminski, and Ian Horrocks. Pagoda: Pay-as-you-go ontology query answering using a datalog reasoner. *J. Artif. Intell. Res.*, 54:309–367, 2015.

[Zolin, 2007] Evgeny Zolin. Modal logic applied to query answering and the case for variable modalities. In *Proc. of DL*, volume 250 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.