Consistent Query Answering over Existential Rules with Open and Closed Predicates

Lorenzo Marconi, Riccardo Rosati

DIAG, Sapienza University of Rome
{marconi,rosati}@diag.uniroma1.it

Abstract

We study Consistent Query Answering (CQA) over knowledge bases with existential rules. Specifically, we propose a novel framework for CQA that combines previous approaches, allowing for the simultaneous presence of both *open* and *closed* predicates, i.e. predicates interpreted under open- and closed-world assumption, respectively. We establish the data complexity of answering unions of conjunctive queries in such a new framework under the so-called AR semantics and for different classes of existential rules. We also provide new complexity results for the standard (i.e. non-inconsistency tolerant) query answering in the presence of both open and closed predicates. Our results show that, for certain classes of rules, the complexity of CQA matches that of non-inconsistency-tolerant query answering.

1 Introduction

In modern data systems and knowledge bases (KBs), it is common to encounter situations where the information to be represented is complete, and others handling incomplete or extensible knowledge. In logical KR frameworks, this distinction is often modeled by adopting the so-called *closed-world assumption* (CWA) or *open-world assumption* (OWA), respectively. Under CWA, the absence of a certain data implies that it is false. Conversely, under OWA one treats missing information as "unknown", thus accommodating incomplete knowledge.

Real-life applications, however, often require managing partially complete knowledge. This may occur both when some data is actually not available (or must remain undisclosed, as in (Benedikt et al. 2016)) or when, by design choice, additional knowledge can be inferred from an underlying, complete dataset. This is the scenario that we consider in the current paper, and it is implemented by specifying which predicates used for modeling a given domain of interest are interpreted in a "closed" (complete) or "open" (incomplete) way. In the literature on Description Logics (DLs), a proposal for integrating OWA and CWA was realized by using the so-called DBoxes (Seylan, Franconi, and de Bruijn 2009; Franconi, Ibáñez-García, and Seylan 2011), i.e. sets of assertions involving closed predicates that, when coupled with a DL TBox, allow one to perform such a kind of reasoning. More recently, the distinction between open and closed predicates has also been considered for ontology-mediated query answering, both when the intensional knowledge is expressed in some DL (Lutz, Seylan, and Wolter 2013, 2019; Ahmetaj, Ortiz, and Simkus 2020; Cauli, Ortiz, and Piterman 2021) or in some class of existential rules (Bienvenu and Bourhis 2019). In (Benedikt et al. 2016), closed and open predicates take the form of *visible* and *invisible* relations, respectively, and the end user can pose both positive and negative queries over such a data source. In the subsequent article (Benedikt et al. 2021), the authors established important complexity results for query answering in this scenario.

In a framework with open and closed predicates, *inconsistencies* may arise when a KB consisting of a database coupled with logical rules allows one to infer new knowledge regarding closed predicates (which, by definition, should not occur). In the current literature on this topic, however, it remains unclear how to treat such inconsistencies, in particular for the task of query answering. Querying databases and knowledge bases in an inconsistency-tolerant way is known as *Consistent Query Answering* (CQA) (Arenas, Bertossi, and Chomicki 1999). The central notion in the CQA framework is the one of *repair*. Given a database $\mathcal{D}$ possibly inconsistent with a given set of logical rules Σ, a repair is a new, consistent database that is "as close as possible" to $\mathcal{D}$ and consistent with Σ. Generally, a database can have multiple repairs. CQA in its original form addresses the following question: given a query q, what are the answers that hold true across all repairs of the database?

Various types of repairs and semantics of query answering have been proposed,[1] depending on how the above concepts of "closeness" and "consistency" are defined. A notable notion of repair–which is the one we refer to in this paper–defines it as an inclusion-maximal subset of the original database that conforms to the given rules. CQA for this kind of repair was deeply investigated in the closed scenario, see e.g. (Chomicki and Marcinkowski 2005; Afrati and Kolaitis 2009; ten Cate, Fontaine, and Kolaitis 2012). The open case has also been extensively studied, in particular in the contexts of DLs, e.g. (Rosati 2011; Bienvenu 2012; Bienvenu, Bourgaux, and Goasdoué 2019), and existential rules,

[1] We refer the reader to (Bertossi 2019) for a recent review.

e.g. (Lukasiewicz et al. 2022; Calautti et al. 2022). None of these works, however, considered the case where both open and closed predicates coexist in the same KB.

The logical rules that we consider in this paper are *existential rules* (Calì, Gottlob, and Lukasiewicz 2012; Baget et al. 2011) that include *tuple-generating dependencies* (*TGDs*) and *denial* constraints.

Example 1. *A university* $\mathsf{u_x}$ *has complete information about its scholars, and it distinguishes between visiting and hired scholars (modeled through unary predicates* V *and* H*). Visiting scholars can not also be hired, and hired scholars must have a registered contract (binary predicate* C*) with the university. Predicates* V*,* H *and* C *are closed.*

Both visiting and hired scholars must be affiliated with some universities. However, the university $\mathsf{u_x}$ *does not know all the (possibly multiple) affiliations of each scholar, so this relation is modeled through an open binary predicate* A*. Moreover, each hired scholar is affiliated with* $\mathsf{u_x}$*.*

We then define the following set Σ *of existential rules:*

$$\tau_1\colon \forall s(V(s) \wedge H(s) \rightarrow \bot) \quad \tau_2\colon \forall s(H(s) \rightarrow \exists c\, C(s,c))$$
$$\tau_3\colon \forall s(V(s) \rightarrow \exists u\, A(s,u)) \quad \tau_4\colon \forall s(H(s) \rightarrow A(s,\mathsf{u_x}))$$

Intuitively, both τ_1 *and* τ_2 *serve as integrity constraints, since they can not be used for inferring new knowledge (as* C *is a closed predicate). Indeed, in order to be consistent with these dependencies, a KB must satisfy the first and the second rule already in its ground data. For instance, the database* $\{V(\mathsf{meg}), H(\mathsf{meg})\}$ *is not consistent with* Σ *both because a fact of the form* $C(\mathsf{meg}, \_)$ *is missing and because* meg *is at the same time a visiting and hired scholar. On the other hand,* τ_3 *and* τ_4 *can be used for inferring new knowledge. For instance, the database* $\{H(\mathsf{meg}), C(\mathsf{meg}, \mathsf{c_0})\}$ *(where* $\mathsf{c_0}$ *is a contract ID) is consistent with* Σ*, and* τ_4 *can be used for inferring the fact* $A(\mathsf{meg}, \mathsf{u_x})$*.*

We investigate the *AR-entailment*[2] semantics of inconsistency-tolerant query answering, a well-established notion in the CQA literature (Lembo et al. 2015; Bienvenu and Bourgaux 2017; Lukasiewicz et al. 2022), which requires a query to hold in all repairs, and we focus on unions of conjunctive queries (UCQs). In our analysis, we also consider a non-inconsistency-tolerant semantics, called *OC-entailment* (from the words "open" and "closed") in the paper, which corresponds to standard skeptical reasoning in all the models of the KB in the presence of both open and closed predicates, in order to compare its complexity to that of AR-entailment. For both such semantics, we analyze the data complexity (i.e., the complexity with respect to the database size) of the associated decision problems.

Our analysis focuses on different classes of existential rules. First, besides the general class of TGDs, we consider four well-known subclasses thereof, namely *acyclic*, *full*, *guarded* and *linear* dependencies. We also consider an orthogonal classification, based on the position of open and closed predicates inside the dependencies, resulting in the classes of *open-head* and *closed-head* dependencies. This last distinction was suggested by the intuition that the position of closed and open predicates in the dependencies affects the complexity of reasoning.

Some of the results that we have obtained exploit the ones provided in previous works, in particular (Lukasiewicz et al. 2022) and (Marconi and Rosati 2024), which provide a comprehensive study of the complexity of CQA for existential rules under OWA and CWA, respectively. Also, some results for the classes of linear and guarded dependencies follow from the ones of (Benedikt et al. 2021).

The main implications of our findings (which are summarized in Table 1 and Table 2) are the following:

- not surprisingly, in most of the cases considered, the presence of both open and closed predicates (and in particular of both open-head and closed-head rules) significantly increases the complexity of the two query entailment problems compared to cases where predicates are either all open or all closed. Nevertheless, we identify some tractable cases, and a new class of dependencies for which CQA is *first-order rewritable* (Artale et al. 2009);
- open- and closed-head dependencies have a different impact on the complexity of the two entailment problems;
- for some classes of rules, the complexity of CQA is the same as the complexity of standard (non-inconsistency-tolerant) query answering.

The paper is structured as follows. We first introduce preliminary notions and definitions (Section 2), as well as the notion of repair and the decision problems studied in the paper (Section 3). We then provide complexity results for such problems, namely OC-entailment (Section 4) and AR-entailment (Section 5). Finally, Section 6 concludes the paper with a brief discussion of our results.

2 Preliminaries

In this section, we introduce the syntax and semantics of databases, existential rules (hereafter called *dependencies*) and queries, encompassing both open and closed predicates.

A *mixed predicate signature* (or simply *signature*) $\mathcal{S}$ is a set of predicate symbols with an associated arity, partitioned in $\mathcal{S}_o$ and $\mathcal{S}_c$, i.e. the sets of *open predicates* and *closed predicates* respectively. The arity of a predicate p is indicated as *arity*(p). A *predicate atom* (or simply *atom*) is an expression of the form $p(\mathbf{t})$, where p is a predicate of arity n and $\mathbf{t}$ is a n-tuple of terms (i.e. variables or constants). We say that an atom α is *ground* (or that α is a *fact*) if no variable occurs in it. Given a predicate signature $\mathcal{S}$, a *database* $\mathcal{D}$ for $\mathcal{S}$ is a set of facts over $\mathcal{S}$. We assume that a signature always contains a closed, 0-ary predicate $\bot$, and that no database contains the fact $\bot$. We indicate as *const*$(\mathcal{D})$ the set of all the constants occurring in $\mathcal{D}$.

Given a first-order (FO) formula ϕ, sometimes we use the notation $\phi(\mathbf{x})$ to emphasize its free variables $\mathbf{x}$. A *conjunctive query* (*CQ*) is a non-empty conjunction of predicate atoms possibly occurring in the scope of an existential quantifier. Given a CQ q, we denote by *Atoms*(q) and *Vars*(q) the sets of predicate atoms and variables occurring in it, respectively. A *union of conjunctive queries* (*UCQ*) is a disjunction

[2]The acronym AR stands for "ABox repair" and it comes from the realm of DL ontologies.

$q_1(\mathbf{x}) \vee \ldots \vee q_n(\mathbf{x})$ where every $q_i(\mathbf{x})$ is a CQ. Given a CQ (resp., UCQ) q, if no variable of q is free, then we say that q is *Boolean*, and we call it *BUCQ* (resp., *BCQ*) for short.

We are now ready to define the notion of dependency that we use throughout the paper.

Definition 1 (Dependency)**.** *Given a predicate signature $\mathcal{S}$, a* dependency *for $\mathcal{S}$ is a FO sentence over $\mathcal{S}$ of the form:*

$$\forall \mathbf{x} \left(\phi(\mathbf{x}) \rightarrow \exists \mathbf{y}\, \psi(\mathbf{y}, \mathbf{z})\right) \tag{1}$$

where ϕ and ψ are conjunctions of atoms, and $\mathbf{z} \subseteq \mathbf{x}$. We call $\mathbf{z}$ the frontier variables *of the dependency.*

In the rest of the paper, we omit that databases and dependencies are given for a certain signature. Given a dependency τ of the form (1), we call *body* and *head* of τ the CQs $\mathit{body}(\tau) = \phi(\mathbf{x})$ and $\mathit{head}(\tau) = \exists \mathbf{y}\, \psi(\mathbf{y}, \mathbf{z})$, respectively, and we say that τ is *single-head* if $|\mathit{Atoms}(\mathit{head}(\tau))| = 1$.

The above notion comprises the well-known *tuple-generating dependencies (TGD)* (corresponding to the dependencies τ in which the predicate $\bot$ does not occur in $\mathit{head}(\tau)$) as well as the *denial constraints*, also known as *negative constraints* (Lukasiewicz et al. 2022) (corresponding to the dependencies τ in which $\bot$ occurs in $\mathit{head}(\tau)$).

Semantics. Dependencies and BUCQs are subclasses of the class of domain-independent relational calculus sentences (Abiteboul, Hull, and Vianu 1995). Given a domain-independent sentence ϕ of the relational calculus and a set of facts $\mathcal{D}$ over the same signature, we define $\mathit{Eval}(\phi, \mathcal{D})$ as the function returning true iff the evaluation of ϕ in the Herbrand model of $\mathcal{D}$ is true. Moreover, given any FO theory Φ, we assume that the predicate $\bot$ is interpreted as empty in every model of Φ.

Definition 2 (Mixed model and OC-entailment)**.** *Given a set of dependencies Σ and a database $\mathcal{D}$, a* mixed model *of $\langle \Sigma, \mathcal{D} \rangle$ is any FO interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ s.t. (i) $\mathcal{I}$ is a FO model of the theory $\Sigma \cup \mathcal{D}$ and (ii) for every closed predicate p_c, $p_c^{\mathcal{I}} = \{\mathbf{t}^{\mathcal{I}} \mid p_c(\mathbf{t}) \in \mathcal{D}\}$.*

Given a domain-independent FO sentence ϕ, we say that ϕ is OC-entailed *by $\langle \Sigma, \mathcal{D} \rangle$ (in symbols $\langle \Sigma, \mathcal{D} \rangle \models \phi$) if ϕ evaluates to true in every mixed model of $\langle \Sigma, \mathcal{D} \rangle$. We say that $\mathcal{D}$ is* consistent with Σ *if a mixed model exists for $\langle \Sigma, \mathcal{D} \rangle$.*

Subclasses of dependencies. We categorize dependencies into five subclasses, namely acyclic, denials, full, guarded and linear, respectively denoted as A, D, F, G and L.

We say that a dependency of the form (1) is *full* if $\mathit{Vars}(\psi) \subseteq \mathit{Vars}(\phi)$, *linear* if $|\mathit{Atoms}(\phi)| = 1$, and *guarded* (Baget et al. 2011; Calì, Gottlob, and Lukasiewicz 2012) if there exists an atom α of ϕ such that $\mathit{Vars}(\alpha) = \mathit{Vars}(\phi)$. We also call it a *denial* in case $\psi = \bot$.

Given a set of dependencies Σ, we call *dependency graph* of Σ the directed graph $\mathcal{G}(\Sigma)$ whose vertices are the dependencies of Σ and such that there is one edge from the vertex τ_1 to τ_2 iff the head of τ_1 contains an atom whose predicate appears in a predicate atom of the body of τ_2. We say that Σ is *acyclic* if there is no cyclic path in $\mathcal{G}(\Sigma)$.

We also refer to three additional classes, namely AFL, $\mathsf{C}_\emptyset$ and C^*, which correspond to the intersection $\mathsf{A} \cap \mathsf{F} \cap \mathsf{L}$, the class containing only the empty set and the class containing every possible set of dependencies, respectively. We remark that every linear dependency is also guarded (i.e. $\mathsf{L} \subset \mathsf{G}$), and that sets consisting of denial dependencies are both acyclic and full (i.e. $\mathsf{D} \subset \mathsf{A} \cap \mathsf{F}$).

The chase. We now recall the notion of *chase* (specifically, we refer to the *restricted chase* (Carral, Dragoste, and Krötzsch 2017)), which intuitively involves enhancing a database with additional information derived from the specified dependencies. Formally, given a database $\mathcal{D}$ and a set of dependencies Σ, we write $\mathit{Chase}(\mathcal{D}, \Sigma)$ to indicate the output of the following procedure. First, initialize a set of atoms $\mathcal{C}$ with $\mathcal{D}$. Then, for each dependency τ in Σ, if there exists a mapping $\sigma : \mathit{Vars}(\mathit{body}(\tau)) \rightarrow \mathit{const}(\mathcal{C})$ such that $\mathit{Eval}(\sigma(\mathit{body}(\tau)), \mathcal{C})$ is true and $\mathit{Eval}(\sigma(\mathit{head}(\tau)), \mathcal{C})$ is false, then add the atoms of $\sigma(\mathit{head}(\tau))$ to $\mathcal{C}$, where distinct variables are replaced with distinct fresh *labeled nulls* (i.e. special constants that do not already occur in $\Sigma \cup \mathcal{C}$). It is well-known that such a procedure is guaranteed to terminate for acyclic and for full sets of dependencies.

We assume the reader to be familiar with the basic notions of computational complexity. In this paper we focus on the *data complexity* of the problems studied (i.e. the complexity with respect to the size of the database); so, for the sake of brevity, we will omit this in the formal statements.

3 CQA Framework for Mixed Predicates

We now give a notion of repair for databases with existential rules and mixed signatures, which naturally extends the ones provided for open signatures (Lukasiewicz et al. 2022) and for closed signatures (Chomicki and Marcinkowski 2005).

Definition 3 (Repair)**.** *Given a database $\mathcal{D}$ and a set of dependencies Σ, a* repair *of $\langle \Sigma, \mathcal{D} \rangle$ is a maximal subset of $\mathcal{D}$ that is consistent with Σ.*

It is immediate to see that a repair always exists for every $\langle \Sigma, \mathcal{D} \rangle$ (given any Σ, the empty set is consistent with Σ).

We define the semantics for inconsistency-tolerant query answering as follows.

Definition 4 (AR-entailment)**.** *Given a database $\mathcal{D}$, a set of dependencies Σ, and a BUCQ q, we say that $\langle \Sigma, \mathcal{D} \rangle$* AR-entails *$q$ (in symbols $\langle \Sigma, \mathcal{D} \rangle \models_{\mathrm{AR}} q$) if $\langle \Sigma, \mathcal{D}' \rangle \models q$ for every repair $\mathcal{D}'$ of $\langle \Sigma, \mathcal{D} \rangle$.*

Obviously, if the database $\mathcal{D}$ is consistent with the given set of dependencies, then $\mathcal{D}$ itself would be the only repair of $\langle \Sigma, \mathcal{D} \rangle$, hence implying that:

Proposition 1. *Let Σ be a set of dependencies, $\mathcal{D}$ be a database, and q be a BUCQ. Then, $\langle \Sigma, \mathcal{D} \rangle \models_{\mathrm{AR}} q$ implies $\langle \Sigma, \mathcal{D} \rangle \models q$. Moreover, if $\mathcal{D}$ is consistent with Σ, $\langle \Sigma, \mathcal{D} \rangle \models_{\mathrm{AR}} q$ holds iff $\langle \Sigma, \mathcal{D} \rangle \models q$.*

Example 2. *Let us consider the set Σ of dependencies of Example 1 and the database $\mathcal{D} = \{V(\mathsf{meg}), H(\mathsf{meg}), C(\mathsf{meg}, \mathsf{c_0})\}$. For solving the inconsistency caused by the denial τ_1, we have to remove either $V(\mathsf{meg})$ or $H(\mathsf{meg})$ from $\mathcal{D}$, hence giving rise to two repairs, i.e. $\{V(\mathsf{meg}), C(\mathsf{meg}, \mathsf{c_0})\}$ and $\{H(\mathsf{meg}), C(\mathsf{meg}, \mathsf{c_0})\}$. Then, it is easy to see that the two BUCQs $q_1 = \exists s, u\, A(s, u) \wedge$*

$C(s, \mathsf{c_0})$ *and* $q_2 = H(\mathsf{meg}) \vee V(\mathsf{meg})$ *are both AR-entailed by* $\langle \Sigma, \mathcal{D} \rangle$*, while the query* $q_3 = \exists s\, V(s)$ *is not.*

For solving tasks involving reasoning with both open and closed predicates, it may be very helpful to partition the set of dependencies into two subsets, each one containing dependencies having, respectively, only open or closed predicates in their head. Intuitively, dependencies contained in the first subset are used for inferring new knowledge, whilst the ones contained in the second subset are used as constraints.

Definition 5 (Open-and closed-head dependencies)**.** *We say that a dependency* τ *is* closed-head *if no open predicate occurs in head*(τ)*, and* open-head *if all the atoms occurring in head*(τ) *are atoms with an open predicate. Given a set* Σ *of dependencies, we denote by* Σ_c *and* Σ_o *the sets of closed- and open-head dependencies occurring in* Σ*, respectively.*

For example, considering the set Σ of dependencies of Example 1, we have that $\Sigma_o = \{\tau_3, \tau_4\}$ and $\Sigma_c = \{\tau_1, \tau_2\}$.

Note that every set of dependencies Σ can be equivalently (w.r.t. the entailment tasks studied in this paper) transformed into a set of dependencies Σ' such that $\Sigma' = \Sigma'_o \cup \Sigma'_c$. In other words, dependencies having a "mixed" head can be split into dependencies without mixed heads. For instance, the sentence $\forall \mathbf{x}\,(\phi(\mathbf{x}) \rightarrow \exists y, z, w\,(P_o(y, z) \wedge P_c(z, w)))$, where $\phi(\mathbf{x})$ is any conjunction of atoms, $P_o \in \mathcal{S}_o$ and $P_c \in \mathcal{S}_c$, is equivalent to the following set of dependencies:

$$\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \exists y, z, w\, O^{aux}(y, z, w))$$
$$\forall y, z, w(O^{aux}(y, z, w)) \rightarrow P_o(y, z)$$
$$\forall y, z, w(O^{aux}(y, z, w)) \rightarrow P_c(z, w)$$

where O^{aux} is a fresh open predicate. Therefore, hereinafter we assume w.l.o.g. that Σ is such that $\Sigma = \Sigma_o \cup \Sigma_c$ and that all its dependencies are single-head.

We can now formally define the decision problems that we study, which are parameterized w.r.t. C_o and C_c, both generic subclasses of C^*.

Input: a database $\mathcal{D}$, a set Σ of dependencies s.t. $\Sigma_o \in \mathsf{C}_o$ and $\Sigma_c \in \mathsf{C}_c$, and a BUCQ q.

Questions: $\textsc{QAns}(\mathsf{C}_o, \mathsf{C}_c)$: does $\langle \Sigma, \mathcal{D} \rangle \models q$?
$\textsc{QAns}_{\text{AR}}(\mathsf{C}_o, \mathsf{C}_c)$: does $\langle \Sigma, \mathcal{D} \rangle \models_{\text{AR}} q$?

In all the decision problems studied, we assume w.l.o.g. that predicates and constants occurring in Σ also occur in $\mathcal{D}$.

4 OC-entailment

In this section, we recall some relevant previous findings on OC-entailment (Proposition 3 and Proposition 4) and provide new complexity results for the QANS problem (Theorem 1 and Theorem 2). To prove the upper bounds, we define new techniques. In particular, we exhibit a novel query rewriting technique (Definition 6) which allows us to prove the AC^0 upper bound of OC-entailment in the case when the set of open-head dependencies belongs to AFL. Besides being interesting on their own, these results are helpful for proving the complexity of CQA (that is, the $\textsc{QAns}_{\text{AR}}$ problem) in the next section.

First, analogously to the standard FO semantics, consistency can be checked through query entailment also in the framework where open and closed predicates are mixed, as stated by the next property, whose proof is straightforward.

Proposition 2. *Let* Σ *be a set of dependencies, and let* $\mathcal{D}$ *be a database. Then,* $\mathcal{D}$ *is consistent with* Σ *iff* $\langle \Sigma, \mathcal{D} \rangle \not\models \bot$.

Moreover, in the case when the database is consistent with the given dependencies, OC-entailment of a BUCQ coincides with checking whether the query is entailed by the database and the open-head dependencies.

Lemma 1. *Let* Σ *be a set of dependencies such that* $\Sigma_o \in \mathsf{F}$*, let* $\mathcal{D}$ *be a database consistent with* Σ*, and let* q *be a BUCQ. Then,* $\langle \Sigma, \mathcal{D} \rangle \models q$ *iff* $\langle \Sigma_o, \mathcal{D} \rangle \models q$.

In general, Lemma 1 does not hold when $\Sigma_o \notin \mathsf{F}$. As counter-example, one can take $\Sigma = \{\forall x\,(C_1(x) \rightarrow \exists y\, O(x, y)), \forall x, y\,(O(x, y) \rightarrow C_2(y))\}$, $\mathcal{D} = \{C_1(a), C_2(a)\}$ and $q = \exists x\, O(x, a)$, where $C_1, C_2 \in \mathcal{S}_c$ and $O \in \mathcal{S}_o$. It is easy to see that $\langle \Sigma, \mathcal{D} \rangle \models q$ and $\langle \Sigma_o, \mathcal{D} \rangle \not\models q$.

The next two propositions summarize the known results about the data complexity of OC-entailment.

First, we refer to some of the results presented in (Lukasiewicz et al. 2022). We remark that such findings, as well as the ones mentioned in Proposition 5, have been established for entailment of BCQs: however, all of them easily extend to the entailment of BUCQs.

Proposition 3 (Lukasiewicz et al. 2022)**.**

- $\textsc{QAns}(\mathsf{A}, \mathsf{D})$ *is in* AC^0.
- $\textsc{QAns}(\mathsf{L}, \mathsf{D})$ *is in* AC^0.
- $\textsc{QAns}(\mathsf{F}, \mathsf{C}_\emptyset)$ *and* $\textsc{QAns}(\mathsf{F}, \mathsf{D})$ *are P-complete.*
- $\textsc{QAns}(\mathsf{G}, \mathsf{C}_\emptyset)$ *and* $\textsc{QAns}(\mathsf{G}, \mathsf{D})$ *are P-complete.*

Next, we recall some results established in (Benedikt et al. 2021) for the *positive query implication* problem (PQI). We remark that in the framework of (Benedikt et al. 2021) the database is restricted to the closed (i.e. visible) predicates: however, such a limitation can be easily overcome by adding auxiliary closed predicates and AFL open-head dependencies. E.g., let $O \in \mathcal{S}_c$ and let $\langle \Sigma, \mathcal{D} \rangle$ be such that $\mathcal{D}$ contains the facts $\mathcal{D}'' = \{O(t_1), \ldots, O(t_n)\}$. We define $\Sigma' = \Sigma \cup \{\forall x\,(O'(x) \rightarrow O(x))\}$ and $\mathcal{D}' = \mathcal{D} \setminus \mathcal{D}'' \cup \{O'(t_1), \ldots, O'(t_n)\}$. It is immediate to verify that both OC-entailment and AR-entailment of BUCQs over the predicates occurring in $\Sigma \cup \mathcal{D}$ coincide for $\langle \Sigma, \mathcal{D} \rangle$ and $\langle \Sigma', \mathcal{D}' \rangle$. Consequently, the results of (Benedikt et al. 2021) for PQI also hold for OC-entailment in our framework.

Proposition 4 (Benedikt et al. 2021)**.**

- $\textsc{QAns}(\mathsf{L}, \mathsf{AFL})$ *is EXP-hard.*
- $\textsc{QAns}(\mathsf{L}, \mathsf{L})$ *and* $\textsc{QAns}(\mathsf{G}, \mathsf{G})$ *are EXP-complete.*

The first property of the above proposition follows in a straightforward way from the reduction shown in the proof of (Benedikt et al. 2021, Theorem 4.6), which actually uses a set of dependencies Σ such that $\Sigma_o \in \mathsf{L}$ and $\Sigma_c \in \mathsf{AFL}$.

We now provide new complexity results for the QANS problem, starting with the following lower bound, which can be proven through a reduction from 3CNF.

Theorem 1. $\textsc{QAns}(\mathsf{A}, \mathsf{AFL})$ *is coNP-hard.*

As for the new upper bounds for OC-entailment, we start by defining a query rewriting technique for the case when $\Sigma_o \in \mathsf{AFL}$. In what follows, given any dependency τ, we

Algorithm 1: expand

Require: A database $\mathcal{D}$, a set of dependencies $\Sigma_o \in \mathsf{A}$, a mapping μ of the set of labeled nulls $\mathcal{N}$ to constants;
Ensure: A database;
1: $\mathcal{C} \leftarrow \mathcal{D}$;
2: **repeat**
3: $\quad \mathcal{C}' \leftarrow \mathcal{C}$;
4: $\quad \mathcal{C} \leftarrow \mathit{Chase}(\mathcal{C}, \Sigma_o)$;
5: $\quad \mathcal{C} \leftarrow \mu(\mathcal{C})$;
6: **until** $\mathcal{C} = \mathcal{C}'$;
7: **return** $\mathcal{C}$;

define $q_b(\tau)$ as the FO formula $\exists \mathbf{x}\, \mathit{body}(\tau)$, where $\mathbf{x}$ are all the variables of $\mathit{body}(\tau)$ that are not frontier variables of τ.

Let q be a BUCQ and let $\mathit{UCQRewrite}(q, \Sigma_o)$ be the BUCQ returned by any known query rewriting algorithm for AFL dependencies under the open-world assumption (e.g. the one presented in (König et al. 2015)), i.e. a new BUCQ q_r whose evaluation in any database $\mathcal{D}$ is true iff $\langle \Sigma_o, \mathcal{D} \rangle \models q$ holds.

Definition 6. *Let Σ be a set of dependencies such that $\Sigma_o \in \mathsf{AFL}$ and $\Sigma_c \in \mathsf{C}^*$, and let q be a BUCQ. We define the following FO sentence $\phi_{oc}(q, \Sigma)$:*

$$\phi_{oc}(q, \Sigma) = \phi_{inc}(\Sigma_c) \vee \mathit{UCQRewrite}(q, \Sigma_o)$$

where $\phi_{inc}(\Sigma_c)$ is the sentence:

$$\phi_{inc}(\Sigma_c) = \bigvee_{\tau \in \Sigma_c} \exists \mathbf{x}\, (\mathit{UCQRewrite}(q_b(\tau), \Sigma_o) \wedge \neg \mathit{head}(\tau))$$

in which $\mathbf{x}$ are the frontier variables of the respective τ.

The function defined above allows us to prove FO-rewritability of OC-entailment in case $\Sigma_o \in \mathsf{AFL}$.

Lemma 2. *Let Σ be a set of dependencies such that $\Sigma_o \in \mathsf{AFL}$ and $\Sigma_c \in \mathsf{C}^*$. For every database $\mathcal{D}$, $\mathit{Eval}(\phi_{oc}(q, \Sigma), \mathcal{D})$ is true iff $\langle \Sigma_c, \mathcal{D} \rangle \models q$.*

Example 3. *Consider the unary predicates P and C, and the binary predicate B, used respectively for modelling people, cities and the birthplace relationship, and let our knowledge on cities be complete, i.e. $C \in \mathcal{S}_c$. In this setting, it is natural to define a set Σ of dependencies such that:*

$$\Sigma_o = \{\forall x, y\, (B(x, y) \rightarrow P(x))\},$$
$$\Sigma_c = \{\forall x, y\, (B(x, y) \rightarrow C(y))\}.$$

Consider also the database $\mathcal{D} = \{C(\mathsf{nyc}), B(\mathsf{kim}, \mathsf{nyc}), B(\mathsf{kim}, \mathsf{usa})\}$, which is evidently inconsistent with Σ. Since no mixed model for $\langle \Sigma, \mathcal{D} \rangle$ exists, then even a query like $q = P(\mathsf{tom})$ is OC-entailed by $\langle \Sigma, \mathcal{D} \rangle$. This is confirmed by the fact that $\mathit{Eval}(\phi_{oc}(q, \Sigma), \mathcal{D})$ is true, where $\phi_{oc}(q, \Sigma)$ is:

$$\exists x, y\, (B(x, y) \wedge \neg C(y)) \vee P(\mathsf{tom}) \vee \exists z\, B(\mathsf{tom}, z).$$

We now introduce Algorithms 1 and 2, designed for checking OC-entailment when the set Σ of dependencies is such that $\Sigma_o \in \mathsf{A}$ and $\Sigma_c \in \mathsf{C}^*$. Algorithms 1, at each iteration, computes the chase of the current database w.r.t. the open-head dependencies Σ_o and then it uses the mapping h for replacing the labeled nulls with constants. The insertion

Algorithm 2: OC-Ent-A+C*

Require: A set of dependencies Σ s.t. $\Sigma_o \in \mathsf{A}$ and $\Sigma_c \in \mathsf{C}^*$, a database $\mathcal{D}$, a BUCQ q;
Ensure: A Boolean value;
1: $m \leftarrow r \cdot h^{(k+1)(b^k)+1}$, where
$\quad k = |\Sigma_o| \qquad h = |\mathit{Pred}(\mathcal{D})| \cdot |\mathit{const}(\mathcal{D})|^r$
$\quad r = \max_{p \in \mathit{Pred}(\mathcal{D})} \mathit{arity}(p) \qquad b = \max_{\tau \in \Sigma_o} |\mathit{Atoms}(\mathit{body}(\tau))|$
2: $\mathcal{N} \leftarrow \{n_i \mid n_i \text{ is a labeled null and } 1 \leq i \leq m\}$;
3: **if** there exists a partial mapping $\mu : \mathcal{N} \rightarrow \mathit{const}(\mathcal{D})$ s.t.
$\quad (i)$ $\mathit{expand}(\mathcal{D}, \Sigma_o, \mu)$ is consistent with Σ_c
$\quad (ii)$ $\mathit{Eval}(q, \mathit{expand}(\mathcal{D}, \Sigma_o, \mu))$ is false **then**
4: $\quad$ **return** false;
5: **end if**
6: **return** true;

of such constants in the atoms of $\mathcal{C}$ may activate other chase steps. This process continues until no new atom is added to $\mathcal{C}$. Then, Algorithm 2 uses the above described *expand* function to identify a possible FO model of $\Sigma_o \cup \mathcal{D}$ in which q evaluates to false.

Lemma 3. *Let $\mathcal{D}$ be a database, and let Σ be a set of dependencies such that $\Sigma_o \in \mathsf{A}$. Then, for every BUCQ q, $\langle \Sigma, \mathcal{D} \rangle \models q$ iff Algorithm 2 returns $false$.*

The same idea above applies to the case when $\Sigma_o \in \mathsf{F}$ as well, except that we do not need to map labeled nulls (which are never generated by the chase in case of full dependencies) to constants. We can thus prove the following property.

Lemma 4. *Let Σ be a set of dependencies such that Σ_o is full. Then, for every database $\mathcal{D}$ and for every BUCQ q, $\langle \Sigma, \mathcal{D} \rangle \models q$ iff $\langle \Sigma_c, \mathit{Chase}(\mathcal{D}, \Sigma_o) \rangle \models q$.*

We are now ready to formally demonstrate the new upper bounds for OC-entailment.

Theorem 2.

1. $\text{QANS}(\mathsf{AFL}, \mathsf{C}^*)$ *is in* AC^0.
2. $\text{QANS}(\mathsf{F}, \mathsf{C}^*)$ *is in P.*
3. $\text{QANS}(\mathsf{A}, \mathsf{C}^*)$ *is in coNP.*

5 AR-entailment

In this section, we focus on the data complexity of QANS_{AR}. We first focus on the simplest class of open-head dependencies, i.e. AFL, and show that, when such a class is combined with acyclic closed-head dependencies, AR-entailment is already Π_2^p-hard (Theorem 3), while in combination with linear open-head dependencies the above problem is tractable (Theorem 4), and even FO rewritable when also the closed-head dependencies belong to AFL (Lemma 6). For the latter case, we exhibit a novel query rewriting technique (Definition 7). Then, we analyze the cases involving the larger classes of linear (Corollary 1), acyclic (Theorem 5) and full (Theorem 6 and Theorem 7) open-head dependencies, showing the intractability of AR-entailment in all these cases.

First we note that, by fixing $\Sigma_c \in \mathsf{D}$, the QANS_{AR} problem coincides with the problem studied in (Lukasiewicz et al. 2022). Indeed, that work studies the purely open case

in which inconsistencies can only be due to the violation of some denial constraint.

Proposition 5 (Lukasiewicz et al. 2022)**.** *For every class* $\mathsf{X} \in \{\mathsf{A}, \mathsf{F}, \mathsf{L}, \mathsf{G}\}$, $\text{QANS}_{\text{AR}}(\mathsf{X}, \mathsf{D})$ *is coNP-complete.*

On the other hand, the case in which $\Sigma_o \in \mathsf{C}_\emptyset$ collapses to the purely closed case, for which all the results we are interested in have already been shown. We recall them below.

Proposition 6.

1. $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{AFL})$ *is in* AC^0.
2. $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{D})$ *is coNP-complete.*
3. $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{A})$ *is coNP-complete.*
4. $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{F})$ *is coNP-complete.*
5. $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{L})$ *is P-complete.*
6. $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{G})$ *is* Π_2^p*-complete.*
7. $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{C}^*)$ *is* Π_2^p*-complete.*

In the above proposition, Property 2 and the lower bounds for Properties 3 and 4 come from (Chomicki and Marcinkowski 2005). The lower bound for Property 5 has been established in (ten Cate, Fontaine, and Kolaitis 2012).[3] The other results are from (Marconi and Rosati 2024).[4]

Next, we examine the case when $\Sigma_o \in \mathsf{AFL}$, for which we are able to show that the following holds.

Theorem 3. $\text{QANS}_{\text{AR}}(\mathsf{AFL}, \mathsf{A})$ *is* Π_2^p*-hard.*

As we further investigate the case in which Σ_o belongs to AFL, we show the following crucial property that holds when closed-head dependencies are linear.

Lemma 5. *Let* Σ *be a set of dependencies such that* $\Sigma_o \in \mathsf{AFL}$ *and* $\Sigma_c \in \mathsf{L}$*, and let* $\mathcal{D}$ *be a database. Then, only one repair of* $\langle \Sigma, \mathcal{D} \rangle$ *exists, and it can be computed in polynomial time in data complexity.*

Moreover, for the case where both Σ_o and Σ_c belong to AFL, we can define a suitable query rewriting technique.

Definition 7. *Let* Σ *be a set of dependencies such that both* Σ_o *and* Σ_c *belong to* AFL*, and let* α *be an atom with an open predicate. We define:*

$$\mathcal{D}(\alpha, \Sigma) = \mathit{Chase}(\{\alpha\}, \Sigma)$$
$$\mathcal{D}^c(\alpha, \Sigma) = \{\beta \mid \beta \in \mathcal{D}(\alpha, \Sigma) \wedge \mathit{Pred}(\beta) \in \mathcal{S}_c\}$$

where $\mathit{Pred}(\beta)$ *is the predicate of* β*.*[5] *Moreover, for a BUCQ* q*, we define* $\phi_{ar}(q, \Sigma)$ *as the following FO sentence:*[6]

$$\bigvee_{q_i \in \mathit{UCQRewrite}(q, \Sigma_o)} \exists \mathbf{x}_i \Big(\bigwedge_{\alpha \in \mathit{Atoms}(q_i)} \Big(\alpha \wedge \bigwedge_{\beta \in \mathcal{D}^c(\alpha, \Sigma)} \beta \Big) \Big)$$

where $\mathbf{x}_i$ *are all the variables occurring in* q_i*.*

[3] Actually, that work also proved the upper bound for $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{F})$ and $\text{QANS}_{\text{AR}}(\mathsf{C}_\emptyset, \mathsf{L})$, but using CQs as queries.

[4] Although (Marconi and Rosati 2024) focuses on more expressive queries and dependencies, all the hardness proofs shown there employ queries and dependencies that are suitable for our needs.

[5] We assume that possible variables occurring in α are treated like constants by the Chase function.

[6] With a slight abuse of notation, in this formula we treat the UCQ $\mathit{UCQRewrite}(q, \Sigma_o)$ as a set of CQs.

The above rewriting function can be used to prove that $\text{QANS}_{\text{AR}}(\mathsf{AFL}, \mathsf{AFL})$ is FO-rewritable, as stated below.

Lemma 6. *Let* Σ *be a set of dependencies such that both* Σ_o *and* Σ_c *belong to* AFL*, and let* q *be a BUCQ. For every database* $\mathcal{D}$*,* $\mathit{Eval}(\phi_{ar}(q, \Sigma), \mathcal{D})$ *is true iff* $\langle \Sigma, \mathcal{D} \rangle \models_{\text{AR}} q$*.*

Example 4. *Let us consider the set of dependencies* Σ*, database* $\mathcal{D}$ *and BUCQ* q *of Example 3. We note that the only repair of* $\langle \Sigma, \mathcal{D} \rangle$ *is* $\mathcal{D}' = \{B(\mathsf{kim}, \mathsf{nyc}), C(\mathsf{nyc})\}$*.*

Differently from the case of OC-entailment, the query q *is not AR-entailed by* $\langle \Sigma, \mathcal{D} \rangle$*, since* $\langle \Sigma, \mathcal{D}' \rangle \not\models q$*. Indeed, we have that* $\mathit{Eval}(\phi_{ar}(q, \Sigma), \mathcal{D})$ *is false, where* $\phi_{ar}(q, \Sigma)$ *is:*

$$P(\mathsf{tom}) \vee \exists x \, (B(\mathsf{tom}, x) \wedge C(x)).$$

Conversely, considering the query $q' = P(\mathsf{kim})$*, we have that* $\langle \Sigma, \mathcal{D} \rangle \models_{\text{AR}} q'$ *and that* $\mathit{Eval}(\phi_{ar}(q', \Sigma), \mathcal{D})$ *is true.*

Thanks to Lemma 5 and Lemma 6 we can identify two cases in which AR-entailment is tractable.

Theorem 4.

1. $\text{QANS}_{\text{AR}}(\mathsf{AFL}, \mathsf{AFL})$ *is in* AC^0.
2. $\text{QANS}_{\text{AR}}(\mathsf{AFL}, \mathsf{L})$ *is in P.*

Theorem 4 provides a polynomial upper bound for the case when $\Sigma_o \in \mathsf{AFL}$ and $\Sigma_c \in \mathsf{L}$. We point out that, in the dual case when $\Sigma_c \in \mathsf{AFL}$ and $\Sigma_o \in \mathsf{L}$, it is impossible to decide AR-entailment in polynomial time, due to the intrinsic complexity of OC-entailment. Indeed, as the database used in the reduction of (Benedikt et al. 2021, Theorem 4.6) is consistent with the given dependencies, the next result follows as a corollary of Proposition 1 and Proposition 4.

Corollary 1. $\text{QANS}_{\text{AR}}(\mathsf{L}, \mathsf{AFL})$ *is EXP-hard.*

We now analyze the case when Σ_o is either acyclic or full. The next lower bound can be shown through a suitable reduction from the case in which $\Sigma_c \in \mathsf{C}^*$ and $\Sigma_o \in \mathsf{C}_\emptyset$.

Theorem 5. $\text{QANS}_{\text{AR}}(\mathsf{A}, \mathsf{AFL})$ *is* Π_2^p*-hard.*

We are also able to provide other reductions of the above kind, which allow us to prove the following hardness results.

Theorem 6.

1. $\text{QANS}_{\text{AR}}(\mathsf{F}, \mathsf{AFL})$ *is coNP-hard.*
2. $\text{QANS}_{\text{AR}}(\mathsf{F}, \mathsf{A})$ *is* Π_2^p*-hard.*
3. $\text{QANS}_{\text{AR}}(\mathsf{F}, \mathsf{L})$ *is* Π_2^p*-hard.*

As for the upper bounds, we consider the procedure that applies the definition of AR-entailment, i.e. it checks whether there exists a repair for which the given query is not OC-entailed. We then need to establish the complexity of checking if a database $\mathcal{D}'$ is a repair of $\langle \Sigma, \mathcal{D} \rangle$. The complexity of this task is addressed by the following lemma.

Lemma 7. *Given a set* Σ *of dependencies and two databases* $\mathcal{D}$ *and* $\mathcal{D}'$*, the problem of checking if* $\mathcal{D}'$ *is a repair of* $\langle \Sigma, \mathcal{D} \rangle$ *is:*

1. *in P if* $\Sigma_o \in \mathsf{F}$ *and* $\Sigma_c \in \mathsf{F}$*;*
2. *in coNP if* $\Sigma_o \in \mathsf{F}$ *and* $\Sigma_c \in \mathsf{C}^*$*;*
3. *in coNP if* $\Sigma_o \in \mathsf{A}$ *and* $\Sigma_c \in \mathsf{C}^*$*.*

It becomes now easy to see that the following holds.

Theorem 7.

Σ_o \ Σ_c	$\mathsf{C}_\emptyset$	D	AFL	A	F	L	G	C^*
$\mathsf{C}_\emptyset$	in AC^0 [P3]	in AC^0 [P3]	in AC^0 [P6]	in AC^0 [P6]	in AC^0 [P6]	in AC^0 [P6]	in AC^0 [P6]	in AC^0 [P6]
AFL	in AC^0 [P3]	in AC^0 [P3]	in AC^0 [T2]	in AC^0 [T2]	in AC^0 [T2]	in AC^0 [T2]	in AC^0 [T2]	in AC^0 [T2]
A	in AC^0 [P3]	in AC^0 [P3]	coNP [T1/T2]	coNP [T1/T2]	coNP [T1/T2]	coNP [T1/T2]	coNP [T1/T2]	coNP [T1/T2]
F	P [P3]	P [P3]	P [P3/T2]	P [P3/T2]	P [P3/T2]	P [P3/T2]	P [P3/T2]	P [P3/T2]
L	in AC^0 [P3]	in AC^0 [P3]	EXP [P4]	EXP-hard [P4]	EXP-hard [P4]	EXP [P4]	EXP [P4]	EXP-hard [P4]
G	P [P3]	P [P3]	EXP [P4]	EXP-hard [P4]	EXP-hard [P4]	EXP [P4]	EXP [P4]	EXP-hard [P4]

Table 1: Data complexity of QANS. [PN] and [TN] are abbreviations for Proposition N, and Theorem N, respectively.

Σ_o \ Σ_c	$\mathsf{C}_\emptyset$	D	AFL	A	F	L	G	C^*
$\mathsf{C}_\emptyset$	in AC^0 [P3]	coNP [P6]	in AC^0 [P6]	coNP [P6]	coNP [P6]	P [P6]	Π_2^p [P6]	Π_2^p [P6]
AFL	in AC^0 [P3]	coNP [P3/P5]	in AC^0 [T4]	Π_2^p [T3/T7]	coNP [P6/T7]	P [P6/T4]	Π_2^p [P6/T7]	Π_2^p [T3/T7]
A	in AC^0 [P3]	coNP [P5]	Π_2^p [T5/T7]	Π_2^p [T3/T7]	Π_2^p [T5/T7]	Π_2^p [T5/T7]	Π_2^p [T5/T7]	Π_2^p [T5/T7]
F	P [P3]	coNP [P5]	coNP [T6/T7]	Π_2^p [T6/T7]	coNP [T6/T7]	Π_2^p [T6/T7]	Π_2^p [T6/T7]	Π_2^p [T6/T7]
L	in AC^0 [P3]	coNP [P5]	EXP [C1/T8]	EXP-hard [C1]	EXP-hard [C1]	EXP [C1/T8]	EXP [C1/T8]	EXP-hard [C1]
G	P [P3]	coNP [P5]	EXP [C1/T8]	EXP-hard [C1]	EXP-hard [C1]	EXP [C1/T8]	EXP [C1/T8]	EXP-hard [C1]

Table 2: Data complexity of QANS_{AR}. [PN], [TN] and [CN] stand for Proposition N, Theorem N and Corollary N, respectively.

1. $\text{QANS}_{\text{AR}}(\mathsf{F}, \mathsf{F})$ *is in coNP.*
2. $\text{QANS}_{\text{AR}}(\mathsf{F}, \mathsf{C}^*)$ *is in* Π_2^p.
3. $\text{QANS}_{\text{AR}}(\mathsf{A}, \mathsf{C}^*)$ *is in* Π_2^p.

We conclude this section by providing a membership result for the case where all the dependencies are guarded, which is established in a similar way to the previous case.

Theorem 8. $\text{QANS}_{\text{AR}}(\mathsf{G}, \mathsf{G})$ *is in EXP.*

6 Discussion and Conclusions

Based on the properties shown in Section 4 and Section 5, we are able to provide a series of data complexity results for OC-entailment and AR-entailment for all the combinations of the classes of open-head dependencies and closed-head dependencies defined in the paper. All such results are summarized in Table 1 and Table 2, respectively. In the tables, all entries refer to completeness results (unless otherwise indicated) and are annotated with references to the paper's statements supporting them (previous findings are reported through propositions). The case $\Sigma_o \in \mathsf{C}^*$ is missing in both tables as it is well known that even $\text{QANS}(\mathsf{C}^*, \mathsf{C}_\emptyset)$ is undecidable (Beeri and Vardi 1981). These outcomes highlight the following interesting aspects of the problems studied.

First, the distinction between closed-head and open-head dependencies plays a crucial role in the complexity analysis conducted in this paper. Consider, for instance, the results for $\text{QANS}_{\text{AR}}(\mathsf{AFL}, \mathsf{L})$, and $\text{QANS}_{\text{AR}}(\mathsf{L}, \mathsf{L})$: if we classify the set of dependencies as a whole, then these two cases collapse in the same class of linear dependencies, for which the data complexity is EXP-complete. Conversely, our classification shows that, if the open-headed dependencies are also acyclic and full, then the data complexity decreases to P.

Furthermore, closed-head and open-head dependencies have a different impact on the data complexity of AR-entailment: for example, as already observed, while $\text{QANS}_{\text{AR}}(\mathsf{AFL}, \mathsf{L})$ is P-complete, the "complementary" problem $\text{QANS}_{\text{AR}}(\mathsf{L}, \mathsf{AFL})$ is EXP-complete. The same holds for OC-entailment.

As for future work, it would be very interesting to establish exact complexity bounds for the cases left open by the present analysis. Also, it would be interesting to extend the present approach to CQA and our complexity analysis to the case of Description Logic knowledge bases, in which the intensional information is expressed by DL TBox assertions rather than existential rules. Moreover, our analysis could be further refined, considering e.g. other, orthogonal, classifications of dependencies (and/or databases). Finally, towards the implementation of CQA prototypes and tools, it would be nice to further investigate and possibly extend the tractable cases identified in our analysis.

Acknowledgements

This work was partially supported by: projects FAIR (PE0000013) and SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the EU - NextGenerationEU; the MUR PRIN 2022LA8XBH project Polar (POLicy specificAtion and enfoRcement for privacy-enhanced data management).

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley. ISBN 0-201-53771-0.

Afrati, F. N.; and Kolaitis, P. G. 2009. Repair checking in inconsistent databases: algorithms and complexity. In Fagin, R., ed., *Database Theory - ICDT 2009, 12th International Conference, St. Petersburg, Russia, March 23-25, 2009, Proceedings*, volume 361 of *ACM International Conference Proceeding Series*, 31–41. ACM.

Ahmetaj, S.; Ortiz, M.; and Simkus, M. 2020. Polynomial rewritings from expressive Description Logics with closed predicates to variants of Datalog. *Artif. Intell.*, 280: 103220.

Arenas, M.; Bertossi, L.; and Chomicki, J. 1999. Consistent Query Answers in Inconsistent Databases. In *Proceedings of the Eighteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS 1999, 68–79. New York, NY, USA: Association for Computing Machinery. ISBN 1581130627.

Artale, A.; Calvanese, D.; Kontchakov, R.; and Zakharyaschev, M. 2009. The *DL-Lite* Family and Relations. *J. of Artificial Intelligence Research*, 36: 1–69.

Baget, J.; Leclère, M.; Mugnier, M.; and Salvat, E. 2011. On rules with existential variables: Walking the decidability line. *Artif. Intell.*, 175(9-10): 1620–1654.

Beeri, C.; and Vardi, M. Y. 1981. The Implication Problem for Data Dependencies. In Even, S.; and Kariv, O., eds., *Automata, Languages and Programming, 8th Colloquium, Acre (Akko), Israel, July 13-17, 1981, Proceedings*, volume 115 of *Lecture Notes in Computer Science*, 73–85. Springer.

Benedikt, M.; Bourhis, P.; Cate, B. t.; and Puppis, G. 2016. Querying Visible and Invisible Information. In *Proc. of the 31st IEEE Symp. on Logic in Computer Science (LICS)*, 1–10.

Benedikt, M.; Bourhis, P.; ten Cate, B.; Puppis, G.; and Vanden Boom, M. 2021. Inference from Visible Information and Background Knowledge. *ACM Trans. on Computational Logic*, 22(2): 13:1–13:69.

Bertossi, L. E. 2019. Database Repairs and Consistent Query Answering: Origins and Further Developments. In Suciu, D.; Skritek, S.; and Koch, C., eds., *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, 48–58. ACM.

Bienvenu, M. 2012. On the Complexity of Consistent Query Answering in the Presence of Simple Ontologies. In Hoffmann, J.; and Selman, B., eds., *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. AAAI Press.

Bienvenu, M.; and Bourgaux, C. 2017. Inconsistency-Tolerant Querying of Description Logic Knowledge Bases. In *Reasoning Web. Semantic Technologies for Intelligent Data Access – 12th Int. Summer School Tutorial Lectures (RW)*, 156–202. Cham: Springer.

Bienvenu, M.; Bourgaux, C.; and Goasdoué, F. 2019. Computing and Explaining Query Answers over Inconsistent DL-Lite Knowledge Bases. *J. Artif. Intell. Res.*, 64: 563–644.

Bienvenu, M.; and Bourhis, P. 2019. Mixed-World Reasoning with Existential Rules under Active-Domain Semantics. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI 2019, 1558–1565. AAAI Press. ISBN 9780999241141.

Calautti, M.; Greco, S.; Molinaro, C.; and Trubitsyna, I. 2022. Preference-based inconsistency-tolerant query answering under existential rules. *Artif. Intell.*, 312: 103772.

Calì, A.; Gottlob, G.; and Lukasiewicz, T. 2012. A general Datalog-based framework for tractable query answering over ontologies. *J. Web Semant.*, 14: 57–83.

Carral, D.; Dragoste, I.; and Krötzsch, M. 2017. Restricted Chase (Non)Termination for Existential Rules with Disjunctions. In Sierra, C., ed., *Proc. of the 26th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 922–928. ijcai.org.

Cauli, C.; Ortiz, M.; and Piterman, N. 2021. Closed- and Open-world Reasoning in DL-Lite for Cloud Infrastructure Security. In *Proc. of the 18th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR)*, 174–183.

Chomicki, J.; and Marcinkowski, J. 2005. Minimal-change integrity maintenance using tuple deletions. *Inf. Comput.*, 197(1-2): 90–121.

Franconi, E.; Ibáñez-García, Y. A.; and Seylan, I. 2011. Query Answering with DBoxes is Hard. In van Ditmarsch, H.; Fernández-Duque, D.; Goranko, V.; Jamroga, W.; and Ojeda-Aciego, M., eds., *Proceedings of the 7th Workshop on Methods for Modalities, M4M 2011, and the 4th Workshop on Logical Aspects of Multi-Agent Systems, LAMAS 2011, Osuna, Spain, November 10-12, 2011*, volume 278 of *Electronic Notes in Theoretical Computer Science*, 71–84. Elsevier.

König, M.; Leclère, M.; Mugnier, M.; and Thomazo, M. 2015. Sound, complete and minimal UCQ-rewriting for existential rules. *Semantic Web*, 6(5): 451–475.

Lembo, D.; Lenzerini, M.; Rosati, R.; Ruzzi, M.; and Savo, D. F. 2015. Inconsistency-tolerant query answering in ontology-based data access. *J. Web Semant.*, 33: 3–29.

Lukasiewicz, T.; Malizia, E.; Martinez, M. V.; Molinaro, C.; Pieris, A.; and Simari, G. I. 2022. Inconsistency-Tolerant Query Answering for Existential Rules. *Artif. Intell.*, 307(C).

Lutz, C.; Seylan, I.; and Wolter, F. 2013. Ontology-Based Data Access with Closed Predicates is Inherently Intractable (Sometimes). In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI 2013, 1024–1030. AAAI Press. ISBN 9781577356332.

Lutz, C.; Seylan, I.; and Wolter, F. 2019. The Data Complexity of Ontology-Mediated Queries with Closed Predicates. *Log. Methods Comput. Sci.*, 15(3).

Marconi, L.; and Rosati, R. 2024. Consistent Query Answering for Existential Rules under Tuple-Deletion Semantics. *CoRR*, abs/2401.05743.

Rosati, R. 2011. On the Complexity of Dealing with Inconsistency in Description Logic Ontologies. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI 2011, 1057–1062. AAAI Press. ISBN 9781577355144.

Seylan, I.; Franconi, E.; and de Bruijn, J. 2009. Effective Query Rewriting with Ontologies over DBoxes (Extended Abstract). In Grau, B. C.; Horrocks, I.; Motik, B.; and Sattler, U., eds., *Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK, July 27-30, 2009*, volume 477 of *CEUR Workshop Proceedings*. CEUR-WS.org.

ten Cate, B.; Fontaine, G.; and Kolaitis, P. G. 2012. On the data complexity of consistent query answering. In Deutsch, A., ed., *15th International Conference on Database Theory, ICDT 2012, Berlin, Germany, March 26-29, 2012*, 22–33. ACM.