Complexity of Approximate Query Answering under Inconsistency in Datalog$^{\pm}$

Thomas Lukasiewicz[1], Enrico Malizia[2], and Cristian Molinaro[3]

[1] Department of Computer Science, University of Oxford, UK
thomas.lukasiewicz@cs.ox.ac.uk
[2] Department of Computer Science, University of Exeter, UK
e.malizia@exeter.ac.uk
[3] DIMES, University of Calabria, Italy
cmolinaro@dimes.unical.it

Abstract. Several semantics have been proposed to query inconsistent ontological knowledge bases, including the intersection of repairs and the intersection of closed repairs as two approximate inconsistency-tolerant semantics. In this paper, we analyze the complexity of conjunctive query answering under these two semantics for a wide range of Datalog$^{\pm}$ languages. We consider both the standard setting, where errors may only be in the database, and the generalized setting, where also the rules of a Datalog$^{\pm}$ knowledge base may be erroneous.

1 Introduction

Description logics (DLs) and existential rules from the context of Datalog$^{\pm}$ are popular ontology languages. In real-world ontology-based applications involving large amounts of data (such as ontology-based data extraction and/or integration), it is very likely that the data are inconsistent with the ontology, and thus inconsistency-tolerant semantics for ontology-based query answering are urgently needed.

Consistent query answering, first developed for relational databases [1] and then generalized as the AR semantics for several DLs [12], is the most widely accepted semantics for querying inconsistent ontologies. Query answering under the AR semantics is known to be a hard problem, even for very simple languages [12]. For this reason, several other semantics have been recently developed with the aim of approximating consistent query answering [12,2,17,4].

In particular, in [12], besides the AR semantics, three other inconsistency-tolerant query answering semantics are proposed, including the approximate *intersection of repairs (IAR)* semantics, in which an answer is considered to be valid, if it can be inferred from the intersection of the repairs (and the ontology). The *intersection of closed repairs (ICR)* [2] is another approximate semantics, in which an answer is valid, if it can be inferred from the intersection of the closure of the repairs (and the ontology).

The complexity of query answering under the AR semantics when the ontology is described using one of the central DLs is well-understood. The data and combined complexity were studied by [20] for a wide spectrum of DLs, while [2] identified cases for simple ontologies (within the *DL-Lite* family) for which tractable data complexity results can be obtained. In [17,19,16], the data and different types of combined complexity, respectively, of the AR semantics have been studied for ontologies described via existential rules and negative constraints. [3] analyzed the data and combined complexity of query answering under the AR and IAR semantics for different notions of maximal repairs over the language *DL-Lite*$_{\mathcal{R}}$. Recently, the AR semantics was extended to the *generalized repair* semantics (GAR) and its computational complexity analyzed [8]. In the GAR semantics, also ontological rules may be removed, and some database atoms and rules are assumed to be non-removable.

This paper continues this line of research and integrates the generalized repair semantics of [8] with the two intersection-based approximate repair semantics. We analyze the complexity of approximate inconsistency-tolerant query answering for a wide range of Datalog$^{\pm}$ languages and for several different complexity measures:

- ▷ We consider different popular inconsistency-tolerant semantics, namely, the IAR and the ICR semantics, in both their standard and their generalized repair variants.
- ▷ We consider the most popular Datalog$^{\pm}$ languages: linear, guarded, sticky, and acyclic existential rules, along with "weak" generalizations, as well as full restrictions, and full (i.e., non-existential) rules in general.
- ▷ Our analysis concerns the data, fixed-program combined, bounded-arity combined, and combined complexity.

2 Datalog$^{\pm}$

We briefly recall some basics on existential rules from the context of Datalog$^{\pm}$ [6].

General. We assume a set $\mathbf{C}$ of *constants*, a set $\mathbf{N}$ of *labeled nulls*, and a set $\mathbf{V}$ of regular *variables*. A *term* t is a constant, null, or variable. We also assume a set of *predicates*, each associated with an arity, i.e., a non-negative integer. An *atom* has the form $p(t_1, \ldots, t_n)$, where p is an n-ary predicate, and $t_1, \ldots, t_n$ are terms. Conjunctions of atoms are often identified with the sets of their atoms. An *instance* I is a (possibly infinite) set of atoms $p(\mathbf{t})$, where $\mathbf{t}$ is a tuple of constants and nulls. A *database* D is a finite instance that contains only constants. A *homomorphism* is a substitution $h\colon \mathbf{C}\cup\mathbf{N}\cup\mathbf{V} \to \mathbf{C}\cup\mathbf{N}\cup\mathbf{V}$ that is the identity on $\mathbf{C}$ and that maps $\mathbf{N}$ to $\mathbf{C} \cup \mathbf{N}$. A *conjunctive query* (CQ) q has the form $\exists \mathbf{Y} \phi(\mathbf{X}, \mathbf{Y})$, where $\phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms without nulls. The *answer* to q over an instance I, denoted $q(I)$, is the set of all tuples $\mathbf{t}$ over $\mathbf{C}$ for which there is a homomorphism h such that $h(\phi(\mathbf{X}, \mathbf{Y})) \subseteq I$ and $h(\mathbf{X}) = \mathbf{t}$. A *Boolean CQ* (BCQ) q is a CQ $\exists \mathbf{Y} \phi(\mathbf{Y})$, i.e., all variables are existentially quantified; q is *true* over I, denoted $I \models q$, if $q(I) \neq \emptyset$, i.e., there is a homomorphism h with $h(\phi(\mathbf{Y})) \subseteq I$.

Dependencies. A *tuple-generating dependency* (TGD) σ is a first-order formula $\forall \mathbf{X} \forall \mathbf{Y}\, \varphi(\mathbf{X}, \mathbf{Y}) \to \exists \mathbf{Z}\, p(\mathbf{X}, \mathbf{Z})$, where $\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z} \subseteq \mathbf{V}$, $\varphi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms, and $p(\mathbf{X}, \mathbf{Z})$ is an atom, all without nulls. For clarity, we consider single-atom-head TGDs; however, our results can be extended to TGDs with a conjunction of atoms in the head. An instance I satisfies σ, written $I \models \sigma$, if the following holds: whenever there exists a homomorphism h such that $h(\varphi(\mathbf{X}, \mathbf{Y})) \subseteq I$, then there exists $h' \supseteq h|_{\mathbf{X}}$, where $h|_{\mathbf{X}}$ is the restriction of h on $\mathbf{X}$, such that $h'(p(\mathbf{X}, \mathbf{Z})) \in I$. A *negative constraint (NC)* ν is a first-order formula $\forall \mathbf{X}\, \varphi(\mathbf{X}) \to \bot$, where $\mathbf{X} \subseteq \mathbf{V}$, $\varphi(\mathbf{X})$ is a conjunction of atoms without nulls and $\bot$ denotes the truth constant *false*. An instance I satisfies ν, written $I \models \nu$, if there is no homomorphism h such that $h(\varphi(\mathbf{X})) \subseteq I$. Given a set Σ

of TGDs and NCs, I satisfies Σ, written $I \models \Sigma$, if I satisfies each TGD and NC of Σ. Given a class of TGDs $\mathbb{C}$, we denote by $\mathbb{C}_\perp$ the formalism obtained by combining $\mathbb{C}$ with arbitrary NCs. Finite sets of TGDs and NCs are also called *programs*, and TGDs are also called *existential rules*.

Knowledge Bases. A *knowledge base* is a pair (D, Σ), where D is a database, and Σ is a program. For programs Σ, Σ_T and Σ_{NC} are the subsets of Σ containing the TGDs and NCs of Σ, respectively. The set of *models* of $KB = (D, \Sigma)$, denoted $mods(KB)$, is the set of instances $\{I \mid I \supseteq D \wedge I \models \Sigma\}$. We say that KB is *consistent*, if $mods(KB) \neq \emptyset$, otherwise KB is *inconsistent*. The *answer* to a CQ q relative to KB is the set of tuples $ans(q, KB) = \bigcap\{q(I) \mid I \in mods(KB)\}$. The answer to a BCQ q is *true*, denoted $KB \models q$, if $ans(q, KB) \neq \emptyset$. The decision version of the *CQ answering* problem is as follows: given a knowledge base KB, a CQ q, and a tuple of constants $\mathbf{t}$, decide whether $\mathbf{t} \in ans(q, KB)$. Since CQ answering can be reduced in LOGSPACE to BCQ answering, we focus on BCQs. The *combined complexity* of BCQ answering considers the database, the set of dependencies, and the query as part of the input. The *bounded-arity combined* (or *ba-combined*) complexity assumes that the arity of the underlying schema is bounded by an integer constant. The *fixed-program combined* (or *fp-combined*) *complexity* considers the sets of TGDs and NCs as fixed; the *data complexity* also assumes the query fixed.

The Datalog$^\pm$ languages that we consider to guarantee decidability are among the most frequently analyzed in the literature, namely, linear (L) [6], guarded (G) [5], sticky (S) [7], and acyclic TGDs (A), along with their "weak" (proper) generalizations weakly guarded (WG) [5], weakly sticky (WS) [7], and weakly acyclic TGDs (WA) [9], as well as their "full" (proper) restrictions linear full (LF), guarded full (GF), sticky full (SF), and acyclic full TGDs (AF), respectively, and full (i.e., existential-free) TGDs (F) in general. We also recall the following further inclusions: $\mathsf{L} \subset \mathsf{G}$, $\mathsf{F} \subset \mathsf{WA} \subset \mathsf{WS}$, and $\mathsf{F} \subset \mathsf{WG}$. We refer to, e.g., [8] for a more detailed overview and complexity results.

3 Approximate Inconsistency Semantics

We now recall three prominent inconsistency-tolerant semantics for ontology-based query answering, namely, the *ABox repair* (AR) semantics and its approximation by the *intersection of repairs* (IAR) and the *intersection of closed repairs* (ICR) semantics [12,2]; all three are based on the notion of *repair*, which is a maximal consistent subset of the given database. Furthermore, we newly define generalized repair variants [8] of the two intersection-based approximate repair semantics.

Classically, errors leading to inconsistencies are assumed to be only in the database, and not in the ontology. [8] have introduced the *generalized inconsistency semantics* allowing for errors also in the ontology, and for parts of the database and the ontology to be without errors. More specifically, for a knowledge base (D, Σ), the generalized semantics allows also (i) to minimally remove TGDs from Σ, and (ii) to partition both D and Σ into a hard and a soft part of non-removable and removable elements, respectively. The so partitioned database (resp., program) is called *flexible database* (resp., *program*).

A *flexible database* is a pair (D_h, D_s) of databases, called the *hard* and *soft database*, respectively. A *flexible program* is a pair (Σ_h, Σ_s) consisting of a finite set Σ_h of TGDs and NCs and a finite set Σ_s of TGDs, called the *hard* and *soft program*, respectively. A *flexible knowledge base* is a pair $((D_h, D_s), (\Sigma_h, \Sigma_s))$, where (D_h, D_s) is a flexible database, and (Σ_h, Σ_s) is a flexible program. Note that a (standard) knowledge base (D, Σ) is a special case of a flexible one $((D_h, D_s), (\Sigma_h, \Sigma_s))$, where $D_h = \emptyset$, $D_s = D$, $\Sigma_h = \Sigma$, and $\Sigma_s = \emptyset$. Below, we provide definitions for flexible knowledge bases that generalize the ones for (standard) knowledge bases.

For knowledge bases $KB' = (D', \Sigma')$ and $KB'' = (D'', \Sigma'')$, we write $KB' \subseteq KB''$, if $D' \subseteq D''$ and $\Sigma' \subseteq \Sigma''$. A *selection* of a flexible knowledge base $((D_h, D_s), (\Sigma_h, \Sigma_s))$ is a knowledge base (D', Σ') such that $D_h \subseteq D' \subseteq (D_h \cup D_s)$ and $\Sigma_h \subseteq \Sigma' \subseteq (\Sigma_h \cup \Sigma_s)$. A *repair* of a flexible knowledge base FKB is an inclusion-maximal consistent selection of FKB. We denote by $Rep(FKB)$ the set of all repairs of FKB. Notice that for (standard) knowledge bases, a repair is usually defined as a maximal consistent subset of the database. However, when a flexible knowledge base models a standard one (i.e., $D_h = \emptyset$ and $\Sigma_s = \emptyset$), the definition above coincides with the classical one.

Example 1. Consider the flexible database (D_h, D_s) given by

$$D_h = \{Postdoc(p), Researcher(p), leaderOf(p', g')\} \text{ and } D_s = \{Prof(p), leaderOf(p, g)\},$$

asserting that p is a postdoc, a researcher, a professor, and the leader of the research group g, and that p' is the leader of g'. Consider also the flexible program (Σ_h, Σ_s) defined as

$$\begin{aligned}
\Sigma_h = \{&Prof(X) &&\to Researcher(X),\\
&Postdoc(X) &&\to Researcher(X),\\
&Prof(X), Postdoc(X) &&\to \bot,\\
&leaderOf(X, Y) &&\to Group(Y)\},\\
\Sigma_s = \{&leaderOf(X, Y) &&\to Prof(X)\},
\end{aligned}$$

expressing that professors and postdocs are researchers, professors and postdocs form disjoint sets, and $leaderOf$ has $Prof$ as domain and $Group$ as range. It is easy to see that $mods(D, \Sigma) = \emptyset$, since p violates the disjointness constraint.

The flexible knowledge base $((D_h, D_s), (\Sigma_h, \Sigma_s))$ has two repairs (D', Σ') and (D'', Σ''):

$$\begin{aligned}
D' &= D_h \cup \{leaderOf(p, g)\}, & \Sigma' &= \Sigma_h,\\
D'' &= D_h, & \Sigma'' &= \Sigma_h \cup \Sigma_s.
\end{aligned}$$

In both, the atom $Prof(p)$ is removed; in the first one, also the rule $leaderOf(X, Y) \to Prof(X)$ is removed, while in the second one, the atom $leaderOf(p, g)$ is removed.

We now define the inconsistency-tolerant semantics considered. For a knowledge base $KB = (D, \Sigma)$, the *closure* $Cn(KB)$ of KB is the set of all ground atoms, built from constants in D and Σ, entailed by D and the TGDs of Σ. Let FKB be a flexible knowledge base, and let q be a BCQ.

- FKB entails q under the *generalized ABox repair* (GAR) *semantics*, if, for all $KB' \in Rep(FKB)$, $KB' \models q$.
- FKB entails q under the *generalized intersection of repairs* (GIAR) *semantics*, if $(D^*, \Sigma^*) \models q$, where $D^* = \bigcap\{D' \mid (D', \Sigma') \in Rep(FKB)\}$ and $\Sigma^* = \bigcap\{\Sigma' \mid (D', \Sigma') \in Rep(FKB)\}$.

	Data	**fp-comb.**	**ba-comb.**	**Comb.**
$\mathsf{L}_\perp, \mathsf{LF}_\perp, \mathsf{AF}_\perp$	in AC^{0+}	NP	Π_2^p	PSPACE
$\mathsf{S}_\perp, \mathsf{SF}_\perp$	in AC^{0+}	NP	Π_2^p	EXP
$\mathsf{F}_\perp, \mathsf{GF}_\perp$	co-NP	Θ_2^{P}	Π_2^p	EXP
$\mathsf{G}_\perp$	co-NP$^+$	Θ_2^{P}	EXP	2EXP
$\mathsf{A}_\perp$	in $\text{AC}^{0\star}$	NP	P^{NEXP}	P^{NEXP}
$\mathsf{WS}_\perp, \mathsf{WA}_\perp$	co-NP$^+$	Θ_2^{P}	2EXP	2EXP
$\mathsf{WG}_\perp$	EXP	EXP	EXP	2EXP

	Data	**fp-comb.**	**ba-comb.**	**Comb.**
$\mathsf{L}_\perp, \mathsf{LF}_\perp, \mathsf{AF}_\perp$	co-NP$^+$	Θ_2^{P}	Π_2^p	PSPACE
$\mathsf{S}_\perp, \mathsf{SF}_\perp$	co-NP$^+$	Θ_2^{P}	Π_2^p	EXP
$\mathsf{F}_\perp, \mathsf{GF}_\perp$	co-NP	Θ_2^{P}	Π_2^p	EXP
$\mathsf{G}_\perp$	co-NP$^+$	Θ_2^{P}	EXP	2EXP
$\mathsf{A}_\perp$	co-NP	Θ_2^{P}	P^{NEXP}	P^{NEXP}
$\mathsf{WS}_\perp, \mathsf{WA}_\perp$	co-NP$^+$	Θ_2^{P}	2EXP	2EXP
$\mathsf{WG}_\perp$	EXP	EXP	EXP	2EXP

Fig. 1. Complexity of IAR and $GIAR$ (left) and of ICR and $GICR$ (right) BCQ answering; all entries without "in" are completeness results. $^+$[19] for $\mathsf{L}_\perp$, $\mathsf{S}_\perp$, $\mathsf{G}_\perp$, $\mathsf{WS}_\perp$, and $\mathsf{WA}_\perp$. *[18].

– FKB entails q under the *generalized intersection of closed repairs* (GICR) *semantics*, if $(D_I, \Sigma^*) \models q$, where $D_I = \bigcap\{Cn(KB') \mid KB' \in Rep(FKB)\}$ and $\Sigma^* = \bigcap\{\Sigma' \mid (D', \Sigma') \in Rep(FKB)\}$.

In the definition above, observe that if FKB is a standard knowledge base, then $\Sigma^* = \Sigma$, and thus the definition above generalizes the AR, IAR, and ICR semantics for standard knowledge bases to the case of flexible knowledge bases. We talk of BCQ answering under the GAR, GIAR, and GICR semantics when flexible knowledge bases can be arbitrary, and we talk of BCQ answering under the AR, IAR, and ICR semantics when flexible knowledge bases model standard knowledge bases (i.e., $D_h = \emptyset$ and $\Sigma_s = \emptyset$).

4 Complexity Results

We give a precise picture of the complexity of BCQ answering from existential rules under the IAR, ICR, $GIAR$, and $GICR$ semantics, which is summarized in Fig. 1.

4.1 Membership Results

IAR semantics. The following theorem proves all upper bounds equal to and above co-NP in Fig. 1, left side, excluding the Θ_2^{P} memberships.

Theorem 1. *If BCQ answering from databases under programs over some Datalog$^\pm$ language L is in* $\mathbf{C}$ *in the data (resp., fp-combined, ba-combined, and combined) complexity, then IAR-BCQ answering from databases under programs over L is in* co-NP$^{\mathbf{C}}$ *in the data (resp., fp-combined, ba-combined, and combined) complexity.*

Consider now the Datalog$^\pm$ fragments whose BCQ answering in the data complexity is in AC^0, i.e., $\mathsf{L}_\perp$, $\mathsf{S}_\perp$, $\mathsf{A}_\perp$, $\mathsf{LF}_\perp$, $\mathsf{AF}_\perp$, and $\mathsf{SF}_\perp$. In such cases, the following theorem states the upper bound in the *fp*-combined complexity.

Theorem 2. *IAR-BCQ answering for* $\mathsf{L}_\perp$, $\mathsf{S}_\perp$, *and* $\mathsf{A}_\perp$ *(and* $\mathsf{LF}_\perp$, $\mathsf{AF}_\perp$, *and* $\mathsf{SF}_\perp$*) is in* NP *in the fp-combined complexity.*

The following theorem proves all Θ_2^{P} upper bounds in Fig. 1, left side.

Theorem 3. *IAR-BCQ answering for* $\mathsf{WS}_\perp$ *and* $\mathsf{G}_\perp$ *(and* $\mathsf{WA}_\perp$, $\mathsf{F}_\perp$, *and* $\mathsf{GF}_\perp$*) is in* Θ_2^{P} *in the fp-combined complexity.*

ICR semantics. The following theorem proves all upper bounds in Fig. 1, right side, including the $\mathrm{P}^{\mathrm{NEXP}} = \text{co-NP}^{\mathrm{NEXP}}$ membership for $\mathsf{A}_\perp$, excluding memberships in Θ_2^{P} and the combined complexity.

Theorem 4. *If BCQ answering from databases under programs over some Datalog$^\pm$ language L is in $\mathbf{C}$ in the data (resp., fp- and ba-combined) complexity, then ICR-BCQ answering from databases under programs over L is in* co-NP$^{\mathbf{C}}$ *in the data (resp., fp- and ba-combined) complexity.*

BCQ answering under the ICR semantics for all the considered Datalog$^\pm$ fragments but $\mathsf{WG}_\perp$ is in Θ_2^{P} in the *fp*-combined complexity.

Theorem 5. *ICR-BCQ answering for all the considered Datalog$^\pm$ fragments but* $\mathsf{WG}_\perp$ *is in Θ_2^{P} in the fp-combined complexity.*

As for the combined complexity, we get the following theorem.

Theorem 6. *If BCQ answering from databases under programs over some Datalog$^\pm$ language L is in the deterministic complexity class $\mathbf{C}$ in the combined complexity, then ICR-BCQ answering from databases under programs over L is in* PSPACE $\cdot\, \mathbf{C}$ *in the combined complexity.*

The membership results above can be extended to the generalized semantics case [15].

4.2 Hardness Results

As BCQ answering under the IAR and ICR semantics for Datalog$^\pm$ fragments L coincides with BCQ answering for L when there are no inconsistencies, we immediately obtain hardness for all NP, PSPACE, EXP, and 2EXP entries in Fig. 1.

IAR semantics. Hardness for co-NP of BCQ answering under the IAR semantics in the data complexity is shown by a reduction from deciding unsatisfiability of 3CNF formulas (UNSAT). It produces a knowledge base with a fixed $\mathsf{GF}_\perp$ program and fixed query. This proves all open co-NP-hardness results in Fig. 1, left side.

Theorem 7. *IAR-BCQ answering for* $\mathsf{GF}_\perp$ *(and* $\mathsf{F}_\perp$*) is* co-NP*-hard in the data complexity.*

We can show that IAR-BCQ answering for $\mathsf{A}_\perp$ is $\mathrm{P}^{\mathrm{NEXP}}$-hard in the ba-combined complexity, proving all $\mathrm{P}^{\mathrm{NEXP}}$-hardness results in Fig. 1, left side. Intuitively, the reduction for the $\mathrm{P}^{\mathrm{NEXP}}$-hardness proof in [8] for AR-BCQ answering for $\mathsf{A}_\perp$ in the ba-combined complexity is turned into a $\mathrm{P}^{\mathrm{NEXP}}$-hardness proof for IAR-BCQ answering in this case.

Theorem 8. *IAR-BCQ answering for* $\mathsf{A}_\perp$ *is* $\mathrm{P}^{\mathrm{NEXP}}$*-hard in the ba-combined complexity.*

BCQ answering under IAR semantics for $\mathsf{GF}_\perp$ (and thus also for $\mathsf{F}_\perp$, $\mathsf{G}_\perp$, $\mathsf{WA}_\perp$, and $\mathsf{WS}_\perp$) in the *fp*-combined complexity can be shown to be Θ_2^{P}-hard via a reduction from the Θ_2^{P}-complete problem COMP-SAT: Given two sets A and B of Boolean formulas, decide whether A contains more satisfiable formulas than B [13,14].

Theorem 9. *IAR-BCQ answering for* $\mathsf{GF}_\perp$ *(and* $\mathsf{F}_\perp$*,* $\mathsf{G}_\perp$*,* $\mathsf{WA}_\perp$*, and* $\mathsf{WS}_\perp$*) is* Θ_2^{P}-hard *in the fp-combined complexity.*

ICR semantics. ICR-BCQ answering in the data complexity is co-NP-hard by a reduction from UNSAT. This proves all open CO-NP-hardness results in Fig. 1, right side.

Theorem 10. *ICR-BCQ answering for* $\mathsf{LF}_\perp$*,* $\mathsf{AF}_\perp$*, and* $\mathsf{SF}_\perp$ *(and* $\mathsf{GF}_\perp$*,* $\mathsf{F}_\perp$*, and* $\mathsf{A}_\perp$*) is* CO-NP*-hard in the data complexity.*

BCQ answering under ICR semantics is Θ_2^{P}-hard in the *fp*-combined complexity, by a reduction from COMP-SAT, for all remaining entries in Fig. 1, right side, but for $\mathsf{WG}_\perp$.

Theorem 11. *ICR-BCQ answering in the fp-combined complexity is* Θ_2^{P}-hard *for all the considered Datalog*$^\pm$ *fragments.*

BCQ answering under the ICR semantics for $\mathsf{AF}_\perp$ (and thus also for $\mathsf{F}_\perp$) is Π_2^{P}-hard in the ba-combined complexity, by a reduction from $NQBF_{2,\forall}$, which is a variant of quantified Boolean formulas with two quantifiers starting with a universal one [10,21].

Theorem 12. *ICR-BCQ answering is* Π_2^{P}-hard *in the ba-combined complexity for* $\mathsf{AF}_\perp$ *(and* $\mathsf{F}_\perp$*).*

The following result shows that ICR-BCQ answering for $\mathsf{A}_\perp$ is $\mathrm{P}^{\mathrm{NEXP}}$-hard in the ba-combined complexity, proving all $\mathrm{P}^{\mathrm{NEXP}}$-hardness results in Fig. 1, right side.

Theorem 13. *ICR-BCQ answering for* $\mathsf{A}_\perp$ *is* $\mathrm{P}^{\mathrm{NEXP}}$*-hard in the ba-combined complexity.*

The following shows all Π_2^{P}-hardness results in Fig. 1, right side. The Π_2^{P}-hardness results in Fig. 1, left side, are proved similarly.

Theorem 14. *ICR-BCQ answering for* $\mathsf{L}_\perp$*,* $\mathsf{LF}_\perp$*,* $\mathsf{AF}_\perp$*,* $\mathsf{S}_\perp$*,* $\mathsf{SF}_\perp$*,* $\mathsf{F}_\perp$*, and* $\mathsf{GF}_\perp$ *is* Π_2^{P}*-hard in the ba-combined complexity.*

Also for the hardness results, it is possible to show that they extend to the generalized semantics case [15].

5 Summary and Outlook

We have given a precise picture of the complexity of BCQ answering under different approximate inconsistency-tolerant semantics for the most popular Datalog$^\pm$ languages and complexity measures. In addition to the standard setting, we have also considered the more general setting where also ontological rules may be removed.

Future research lines include considering other classes of existential rules and to define other semantics for inconsistency-tolerant ontological query answering. Another interesting direction for future work is to carry out a complexity analysis of the local generalized semantics [8]. Also, a more fine-grained way to analyze the complexity of query answering would be a non-uniform approach, looking at the complexity of a single ontology or a single ontology-mediated query (see, e.g., [11]).

Acknowledgements. This work was supported by The Alan Turing Institute under the UK EPSRC grant EP/N510129/1, and by the EPSRC grants EP/R013667/1, EP/L012138/1, and EP/M025268/1.

References

1. Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proc. PODS*, pages 68–79, 1999.
2. Meghyn Bienvenu. On the complexity of consistent query answering in the presence of simple ontologies. In *Proc. AAAI*, pages 705–711, 2012.
3. Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *Proc. AAAI*, pages 996–1002, 2014.
4. Meghyn Bienvenu and Riccardo Rosati. Tractable approximations of consistent query answering for robust ontology-based data access. In *Proc. IJCAI*, pages 775–781, 2013.
5. Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res.*, 48:115–174, 2013.
6. Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *J. Web Sem.*, 14:57–83, 2012.
7. Andrea Calì, Georg Gottlob, and Andreas Pieris. Towards more expressive ontology languages: The query answering problem. *Artif. Intell.*, 193:87–128, 2012.
8. Thomas Eiter, Thomas Lukasiewicz, and Livia Predoiu. Generalized consistent query answering under existential rules. In *Proc. KR*, pages 359–368, 2016.
9. Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.
10. Gianluigi Greco, Enrico Malizia, Luigi Palopoli, and Francesco Scarcello. On the complexity of core, kernel, and bargaining set. *Artificial Intelligence*, 175(12–13):1877–1910, 2011.
11. André Hernich, Carsten Lutz, Fabio Papacchini, and Frank Wolter. Dichotomies in ontology-mediated querying with the guarded fragment. In *Proc. PODS*, pages 185–199, 2017.
12. Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant semantics for description logics. In *Proc. RR*, pages 103–117, 2010.
13. Thomas Lukasiewicz and Enrico Malizia. On the complexity of mCP-nets. In *Proc. AAAI*, pages 558–564, 2016.
14. Thomas Lukasiewicz and Enrico Malizia. A novel characterization of the complexity class Θ_k^P based on counting and comparison. *Theor. Comput. Sci.*, 694:21–33, 2017.
15. Thomas Lukasiewicz, Enrico Malizia, and Cristian Molinaro. Complexity of approximate query answering under inconsistency in datalog$^\pm$. In *Proc. IJCAI*, 2018. To appear.
16. Thomas Lukasiewicz, Maria Vanina Martinez, Andreas Pieris, and Gerardo I. Simari. From classical to consistent query answering under existential rules. In *Proc. AAAI*, pages 1546–1552, 2015.
17. Thomas Lukasiewicz, Maria Vanina Martinez, and Gerardo I. Simari. Inconsistency handling in Datalog+/– ontologies. In *Proc. ECAI*, pages 558–563, 2012.
18. Thomas Lukasiewicz, Maria Vanina Martinez, and Gerardo I. Simari. Inconsistency-tolerant query rewriting for linear Datalog+/–. In *Proc. Datalog 2.0*, pages 123–134, 2012.
19. Thomas Lukasiewicz, Maria Vanina Martinez, and Gerardo I. Simari. Complexity of inconsistency-tolerant query answering in Datalog+/–. In *Proc. OTM*, pages 488–500, 2013.
20. Riccardo Rosati. On the complexity of dealing with inconsistency in description logic ontologies. In *Proc. IJCAI*, pages 1057–1062, 2011.
21. Marcus Schaefer. Graph Ramsey theory and the polynomial hierarchy. *Journal of Computer and System Sciences*, 62(2):290–322, 2001.