Explanations for Query Answers under Existential Rules

İsmail İlkan Ceylan[1*] , **Thomas Lukasiewicz**[1] , **Enrico Malizia**[2] and **Andrius Vaicenavičius**[1]
[1]Department of Computer Science, University of Oxford, UK
[2]Department of Computer Science, University of Exeter, UK
{ismail.ceylan, thomas.lukasiewicz, andrius.vaicenavicius}@cs.ox.ac.uk, e.malizia@exeter.ac.uk

Abstract

Ontology-mediated query answering is an extensively studied paradigm, which aims at improving query answers with the use of a logical theory. As a form of logical entailment, ontology-mediated query answering is fully interpretable, which makes it possible to derive explanations for query answers. Surprisingly, however, explaining answers for ontology-mediated queries has received little attention for ontology languages based on existential rules. In this paper, we close this gap, and study the problem of explaining query answers in terms of minimal subsets of database facts. We provide a thorough complexity analysis for several decision problems associated with minimal explanations under existential rules.

1 Introduction

Ontology-based data access (OBDA) is a popular paradigm in knowledge representation and reasoning [Poggi *et al.*, 2008]. The main goal is to facilitate access to possibly *heterogeneous* and *incomplete* data sources based on a logical theory. This is achieved via an *ontology* that enriches the user query, typically a union of conjunctive queries, with commonsense knowledge. In this framework, the ontology and the user query are viewed as two components of one composite query, called *ontology-mediated query (OMQ)* [Bienvenu *et al.*, 2014]. The task of evaluating such queries is then called *ontology-mediated query answering (OMQA)*.

Ontology languages are mostly fragments of *first-order logic (FOL)*, which result from a simple trade-off between the expressivity of the language and the computational complexity of reasoning in the language. As a form of first-order entailment, ontology-mediated query answering is fully interpretable, which makes it possible to derive explanations for query answers. Explanations are widely considered as an essential component of scientific progress. The fact that many recent artificial intelligence systems operate mostly as a *black box* has led some serious concerns; see, e.g., [Došilović *et al.*, 2018], for a recent survey.

Description logics (DLs) [Baader *et al.*, 2007] and *existential rules* [Calì *et al.*, 2012b; Calì *et al.*, 2013; Calì *et al.*, 2012a], together, encompass the most widely used knowledge representation languages in the context of ontology-mediated query answering. Ontologies have found applications in *data exchange* [Fagin *et al.*, 2005], *medical diagnosis* [Bertaud-Gounot *et al.*, 2012], and *life sciences* [Bard and Rhee, 2004], all of which can potentially benefit from explanations. In fact, there has been a significant amount of interest in tracking down and understanding the causes of various types of entailments in DL ontologies.

A prominent approach is to identify explanations in terms of a subset of the axioms in the ontology [Kalyanpur *et al.*, 2007; Baader and Suntisrivaraporn, 2008]. The benefit of this approach is that it allows us to abstract away from the particular proof technique used to derive an entailment, and hence to pinpoint the sets of axioms that are responsible for an entailment. Such explanation sets are, furthermore, required to be *minimal* with respect to some order, like *subset*, *cardinality*, or *preference* order. These explanations are called *justifications* [Kalyanpur *et al.*, 2007; Horridge *et al.*, 2008; Suntisrivaraporn *et al.*, 2008], and the overall approach is also known as *axiom pinpointing* in the DL literature [Horridge *et al.*, 2009; Baader and Suntisrivaraporn, 2008].

Earlier work on axiom pinpointing, however, is exclusively based on standard reasoning tasks, and hence on deriving explanations based on the axioms of the ontology. Indeed, there is very little work in the direction of explaining query entailments. To date, the only works in explaining query answers is given for the *DL-Lite* family of languages [Borgida *et al.*, 2008; Bienvenu *et al.*, 2019], as we elaborate later, in detail. Surprisingly, explanations are not studied in the context of existential rules.

In the present paper, we close this gap and study the problem of explaining query answers under existential rules. More specifically, given an OMQ, we are interested in explaining this compound query in terms of the minimal satisfying subsets of a given database. Such a minimal subset of the database is called a *minimal explanation, or simply MinEX*. Incorporating ideas from axiom pinpointing [Peñaloza and Sertkaya, 2017], we introduce a class of problems based on the notion of minimal explanation. We conduct a detailed complexity analysis for each of the problems introduced, and provide a host of complexity results that cover a representative set of

[*]Contact Author

existential rules. Our results extend naturally to other existential rule languages. All the proof details can be found in the extended version of this paper available from the authors.

2 Preliminaries

We give a brief overview on existential rules and the paradigm of ontology-mediated query answering [Calì *et al.*, 2012b; Calì *et al.*, 2013; Calì *et al.*, 2012a], and also give some complexity-theoretic background relevant to our study.

2.1 First-Order Logic

We consider a relational vocabulary consisting of mutually disjoint, possibly infinite sets $\mathbf{R}$, $\mathbf{V}$, and $\mathbf{C}$ of *predicates*, *variables*, and *constants*, respectively. A *term* is either a constant or a variable. An *atom* is an expression of the form $p(t_1, \ldots, t_n)$, where p is an n-ary predicate, and $t_1, \ldots, t_n$ are terms. A *ground atom* (or *fact*) has only constants as terms.

A first-order *formula* is built as usual from *atoms* over the given vocabulary, *truth constants* $\top$, $\bot$, *operators* $\neg$, $\vee$, $\wedge$, $\rightarrow$, and *quantifiers* $\exists, \forall$. A *quantifier-free formula* is a formula that does not use quantifiers. A variable in a formula is *quantified* (or *bound*), if it is in the scope of a quantifier; otherwise, it is *free*. A *sentence* is a formula without any free variables.

The semantics of FOL is given by means of *interpretations* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a *possibly infinite* domain, and $\cdot^{\mathcal{I}}$ is an interpretation function that maps every constant a to a domain element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, every predicate p with arity n to a relation $p^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$. A sentence Φ is *satisfied* by an interpretation $\mathcal{I}$, if $\mathcal{I} \models \Phi$, where $\models$ is the standard first-order entailment relation.

A (first-order) theory Σ is a (finite) set of first-order formulas. An interpretation $\mathcal{I}$ is a *model* of a theory Σ, denoted $\mathcal{I} \models \Sigma$, if $\mathcal{I}$ satisfies all $\Phi \in \Sigma$. Σ *entails* a sentence Φ, written $\Sigma \models \Phi$, if all models of Σ are also models of Φ.

2.2 Existential Rules

A *tuple-generating dependency (TGD)* is a first-order formula of the form

$$\forall \mathbf{X}\, \Phi(\mathbf{X}) \rightarrow \exists \mathbf{Y}\, \Psi(\mathbf{X}, \mathbf{Y}),$$

where $\Phi(\mathbf{X})$ is a conjunction of atoms, called the *body* of the TGD, and $\Psi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms, called the *head* of the TGD. Classes of TGDs are also known as *existential rules*, or *Datalog*$^{\pm}$ *languages* in the literature. A *program* (or an *ontology*) is a finite set Σ of TGDs.

TGDs can express the *inclusion* and *join dependencies* of databases. In its general form, however, reasoning with TGDs is undecidable [Beeri and Vardi, 1981], but there are a plethora of decidable fragments of TGDs. We review some known (syntactic) restrictions on TGDs that ensure decidability (and even tractability in most cases).

A TGD is *guarded*, if there exists a body atom that contains (or "guards") all body variables. The class of guarded TGDs, denoted G, is defined as the family of all possible sets of guarded TGDs. A key subclass of guarded TGDs are the *linear* TGDs with just one body atom. The class of linear TGDs is denoted by L. It is easy to verify that $\mathsf{L} \subset \mathsf{G}$.

Stickiness enforces the following property: variables that appear more than once in a body (i.e., join variables) must always be propagated (or "stick") to the inferred atoms [Calì *et al.*, 2012b]. A TGD that enjoys this property is called *sticky*, and the class of sticky TGDs is denoted by S. Weak stickiness generalizes stickiness by considering only "harmful" variables, and defines the class WS of *weakly sticky* TGDs. Observe that $\mathsf{S} \subset \mathsf{WS}$.

A set of TGDs is *acyclic* and belongs to the class A if its predicate graph is acyclic. Equivalently, an acyclic set of TGDs can be seen as a non-recursive set of TGDs. A set of TGDs is *weakly acyclic*, if its predicate graph enjoys a certain acyclicity condition, which guarantees the existence of a finite canonical model; the associated class is denoted WA. It is known that $\mathsf{A} \subset \mathsf{WA} \subset \mathsf{WS}$ [Calì *et al.*, 2012b].

The class of *full* TGDs do not contain any existentially quantified variables. The corresponding class is denoted by F. Restricting full TGDs to satisfy linearity, guardedness, stickiness, or acyclicity yields the classes LF, GF, SF, and AF, respectively. It is known that $\mathsf{F} \subset \mathsf{WA}$ [Fagin *et al.*, 2005].

$\mathcal{L}$	**Data**	fp**-comb.**	ba**-comb.**	**Comb.**
L, LF, AF	$\leq$ AC^0	NP	NP	PSPACE
S, SF	$\leq$ AC^0	NP	NP	EXP
A	$\leq$ AC^0	NP	NEXP	NEXP
G	P	NP	EXP	2EXP
F, GF	P	NP	NP	EXP
WS, WA	P	NP	2EXP	2EXP

Table 1: Complexity of OMQA under existential rules.

2.3 Ontology-Mediated Query Answering

A *database* D is a finite set of facts over a (finite) relational vocabulary. A *conjunctive query (CQ)* is an existentially quantified formula $\exists \mathbf{X}\, \Phi(\mathbf{X}, \mathbf{Y})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms over the set of variables $\mathbf{X}$ and $\mathbf{Y}$; a *union of conjunctive queries (UCQ)* is a disjunction of CQs (over the same free variables). A query is *Boolean* if it is a sentence.

The paradigm of ontology-mediated query answering generalizes query answering over databases by incorporating additional background knowledge in terms of an ontology. Formally, an *ontology-mediated query (OMQ)* is a pair (Q, Σ), where Q is a Boolean query, and Σ is an ontology. Given a database D and an OMQ (Q, Σ), we say that D entails the OMQ (Q, Σ), denoted $D \models (Q, \Sigma)$, if for all models $\mathcal{I} \models \Sigma \cup D$ it holds that $\mathcal{I} \models Q$, where $\models$ is first-order entailment under the *standard name assumption*. *Ontology-mediated query answering (OMQA)* is the task of deciding whether $D \models (Q, \Sigma)$ for a given database D and an OMQ (Q, Σ).

A key paradigm in OMQA is the *FO-rewritability* of queries: an OMQ (Q, Σ) is *FO-rewritable*, if there exists a Boolean UCQ Q_Σ such that, for all databases D that are consistent relative to Σ, we have that $D \models (Q, \Sigma)$ iff $D \models Q_\Sigma$. In this case, Q_Σ is called an *FO-rewriting* of (Q, Σ). A class of programs $\mathcal{L}$ is *FO-rewritable*, if it admits an FO-rewriting for any UCQ and program in $\mathcal{L}$. All languages from Table 1 with AC^0 data complexity are FO-rewritable.

In our complexity analysis, we make the standard assumptions [Vardi, 1982]: the *combined complexity* of query answering is calculated by considering all the components, i.e., the database, the program, and the query, as part of the input. The *bounded-arity combined complexity* (or simply *ba-combined complexity*) assumes that the maximum arity of the predicates in $\mathbf{R}$ is bounded by an integer constant. The *fixed-program combined complexity* (or simply *fp-combined complexity*) is calculated by considering the ontology as fixed. Finally, the *data complexity* is calculated by considering the database as the input, i.e., everything else is fixed. Table 1 summarizes the known complexity results for OMQA in the different classes of TGDs that we consider.

The most relevant complexity classes for our analysis and their relations are given as follows:

$$\mathrm{AC}^0 \subseteq \mathrm{P} \subseteq \mathrm{NP}, \mathrm{coNP} \subseteq \mathrm{D}^{\mathrm{P}} \subseteq \Sigma_2^{\mathrm{P}}, \Pi_2^{\mathrm{P}} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP} \subseteq \mathrm{NEXP}, \mathrm{coNEXP} \subseteq \mathrm{D}^{\mathrm{Exp}} \subseteq \mathrm{P}^{\mathrm{NExp}} \subseteq 2\mathrm{EXP},$$

where $\mathrm{D}^{\mathrm{Exp}}$ denotes the class $\mathrm{NEXP} \wedge \mathrm{coNEXP}$.

3 Explanations for Query Answers

In our framework, an explanation is given in terms of a set of database facts, and we are interested in a minimal set of facts that entail a given OMQ. The following definition is a natural extension of those related to axiom pinpointing [Peñaloza and Sertkaya, 2017] to ontology-mediated query answering.

Definition 1 (MinEX)**.** For a database D and an OMQ (Q, Σ), where Σ is a set of existential rules, and Q is a query, an *explanation* for (Q, Σ) in D is a subset $E \subseteq D$ of facts such that $E \models (Q, \Sigma)$. A *minimal explanation* E, or *MinEX*, for (Q, Σ) in D is an explanation for (Q, Σ) in D that is subset-minimal, i.e., there is no proper subset $E' \subsetneq E$ that is an explanation for (Q, Σ) in D.

When the OMQ (Q, Σ) and the database D are clear from the context, we simply speak about MinEXs without explicitly mentioning (Q, Σ) or D.

We provide a running example that will be used along the paper to illustrate the different problems studied. Briefly stated, the notion of minimal explanations and the associated problems are closely related to minimal hitting set problems [Gainer-Dewar and Vera-Licona, 2017; Gottlob and Malizia, 2018], which appears naturally in several domains. Our running example is from the field of computational biology, motivated by experimental design for protein networks [Klamt *et al.*, 2009; Ramadan *et al.*, 2004].

Example 2. Let us consider the protein containment scenario illustrated in Figure 1. In this example, we are interested in identifying proteins $p_1, \dots, p_6$ in relation to the complexes c_1, c_2, and c_3. We want to find a minimal subset of proteins that covers all complexes, i.e., a minimal subset of proteins that has at least one representative from each complex.

We can express this problem as an OMQ in a way that every answer to this problem is in bijection with a minimal explanation of the OMQ as follows. We define the database:

$$D_p = \{protein(p_i) \mid 1 \leq i \leq 6\},$$

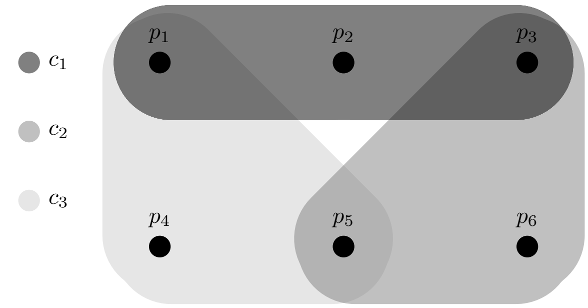

Figure 1: Protein containment in complexes, where proteins are given as $p_1, \dots, p_6$ and complexes c_1, c_2, and c_3 are color-coded (as shown on the left-hand side).

which encodes the set of proteins, and the OMQ $\{Q_p, \Sigma_p\}$:

$$\Sigma_p = \{protein(p_i) \rightarrow \bigwedge_{p_i \text{ in } c_j} covered(c_j) \mid 1 \leq i \leq 6\},$$

$$Q_p = covered(c_1) \wedge covered(c_2) \wedge covered(c_3).$$

The ontology encodes the relation between proteins and complexes, and the query asks whether all complexes c_1, c_2, and c_3 are covered.

Consider now a subset $E \subseteq D_p$. Then, it is easy to verify that $E \models (Q_p, \Sigma_p)$ iff $E \models \{covered(c_i), \Sigma_p\}$, for every complex c_i. Thus, MinEXs for $\{Q_p, \Sigma_p\}$ in D_p are in bijection with minimal protein covers of complexes. ■

We focus on this running example throughout the paper due to its simplicity.

4 Recognizing Minimal Explanations

In this section, we study the fundamental decision problem for MinEXs of deciding whether a given subset of a database is a minimal explanation. This is a natural decision version of the search problem of finding a MinEX.

Problem: IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$
Input: A database D, an OMQ (Q, Σ), where Q is a UCQ and Σ is from the class $\mathcal{L}$ of TGDs, and a set of facts $E \subseteq D$.
Question: Is E a MinEX for (Q, Σ) in D?

IS-MINEX is the most basic problem that is studied in this paper, and serves as a baseline for the other problems. As all the remaining problems studied, IS-MINEX is parametrized with a query language. Let us illustrate this problem in our running example.

Example 3. Recall the database D_p. Observe that the subsets

$$E_1 = \{protein(p_1), protein(p_3)\},$$
$$E_2 = \{protein(p_2), protein(p_5)\},$$
$$E_3 = \{protein(p_2), protein(p_4), protein(p_6)\},$$

of the database D_p are MinEXs for the OMQ (Q_p, Σ_p), and give the minimal protein covers of complexes. However, $\{protein(p_4), protein(p_5), protein(p_6)\}$ is not a MinEX, as it does not cover all complexes and thus does not entail (Q_p, Σ_p). The set $\{protein(p_1), protein(p_2), protein(p_3)\}$ entails (Q_p, Σ_p), but it is not a MinEX, since it is not minimal (i.e., $protein(p_2)$ can be removed without affecting the satisfaction). ■

$\mathcal{L}$	**Data**	fp-**comb.**	ba-**comb.**	**Comb.**
L, LF, AF	$\leq$ P	$\mathrm{D^P}$	$\mathrm{D^P}$	PSPACE
S, SF	$\leq$ P	$\mathrm{D^P}$	$\mathrm{D^P}$	EXP
A	$\leq$ P	$\mathrm{D^P}$	$\mathrm{D^{Exp}}$	$\mathrm{D^{Exp}}$
G	P	$\mathrm{D^P}$	EXP	2EXP
F, GF	P	$\mathrm{D^P}$	$\mathrm{D^P}$	EXP
WS, WA	P	$\mathrm{D^P}$	2EXP	2EXP

Table 2: Complexity results for IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$.

First, we present a general algorithm for deciding IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$. Assume that OMQA in $\mathcal{L}$ is in the class $\mathcal{C}$. Then, IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$ can be decided by one $\mathcal{C}$ check and a polynomial number of co-$\mathcal{C}$ checks as follows: testing whether E is a MinEX involves checking whether E entails (Q, Σ), and checking whether E is minimal. The entailment can be checked with a single call to $\mathcal{C}$. To check minimality of E, it is enough to show that removing any element e of E gives a set that does *not* entail (Q, Σ). Therefore, we need to carry out a polynomial number of non-entailment checks, each in co-$\mathcal{C}$. Hence, we state the following result.

Theorem 4. IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$ *can be decided by a single $\mathcal{C}$ check, followed by a polynomial number of co-$\mathcal{C}$ checks, where $\mathcal{C}$ is the complexity of OMQA in $\mathcal{L}$.*

As a consequence of Theorem 4, we are able to claim all membership results given in Table 2. For example, OMQA in the language G is NP-complete. That is, we need to make an NP test (entailment) followed by polynomially many CONP tests (non-entailment) to decide IS-MINEX$(\mathrm{UCQ}, \mathsf{G})$. Clearly non-entailment tests can also be combined into a single CONP test, which implies that the overall procedure is in $\mathrm{D^P}$. Similar arguments apply to other languages considered.

Therefore, in the rest of this section, we only need to show that these upper bounds are also matching lower bounds for IS-MINEX, as shown in Table 2. First, we show that IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$ is $\mathrm{D^P}$-hard in *fp*-combined complexity even for FO-rewritable languages.

Theorem 5. IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$ *is $\mathrm{D^P}$-hard for languages $\mathcal{L} \in \{\mathsf{LF}, \mathsf{AF}, \mathsf{SF}\}$ in fp- and ba-combined complexity.*

This result implies that IS-MINEX$(\mathrm{UCQ}, \mathcal{L})$ is $\mathrm{D^P}$-hard for all considered languages in *fp*-, *ba*- combined complexity as a consequence of the inclusions between the languages.

This result is obtained by a reduction from the canonical $\mathrm{D^P}$-complete problem SAT-UNSAT, which asks, given two 3CNF formulas, whether the first is satisfiable, and the second is unsatisfiable. In the construction, the database contains facts encoding satisfying assignments of clauses for both formulas and facts enforcing the consistency of the assignments. There is an additional fact, which is a kind of *jolly*, allowing to satisfy the second formula, bypassing the constraints of the consistency in the assignments. In this way, the jolly is required in a MinEX iff the second formula is not satisfiable. The program is empty, therefore, it applies to all languages. The query ensures that there exists a MinEX iff the first formula is satisfiable and the second is unsatisfiable.

Note that many hardness results are a consequence of the hardness of OMQA in the given languages; see, e.g., Table 1. The only case that is not covered by the given results is hence IS-MINEX$(\mathrm{UCQ}, \mathsf{A})$ in *ba*- and combined complexity. The matching lower bound is shown in the following result.

Theorem 6. IS-MINEX$(\mathrm{UCQ}, \mathsf{A})$ *is $\mathrm{D^{Exp}}$-hard in ba-combined complexity.*

This reduction is from a $\mathrm{D^{Exp}}$-complete problem, inspired by the construction given in [Eiter *et al.*, 2016]. The problem is a variant of the tiling problem, which is NEXP-complete. Formally, given a tuple (w_1, w_2, TP_1, TP_2), where w_1 and w_2 are initial tiling conditions, and TP_1 and TP_2 are two tiling problems for the exponential square $2^n \times 2^n$, decide whether TP_1 has no solution with w_1, and TP_2 has a solution with w_2.

The main intuition behind the proof is as follows. We encode the tiling problem in the program Σ. This program is designed in such a way that, together with a database encoding the adjacency rules and the initial condition, Σ entails an atom $tiling^i$ iff TP_i has a solution with w_i. The construction ensures the following. If TP_1 has a solution with w_1, then $tiling^1$ can be derived from the rules in Σ^1, and hence there is no need to include the atom $tiling^1$ in E to entail the query. Hence, E is a MinEX iff TP_1 has *no* solution with w_1 and TP_2 has a solution with w_2.

We observe that IS-MINEX remains tractable in data complexity. In all other cases, IS-MINEX has either the same computational complexity as OMQA (for deterministic classes) or has a higher computational complexity (for non-deterministic classes). This concludes our analysis for IS-MINEX.

5 Set of All Minimal Explanations

In this section, we analyze the problem of deciding whether a given set of subsets of a database is the set of all MinEXs.

Problem: ALL-MINEX$(\mathrm{UCQ}, \mathcal{L})$
Input: A database D, an OMQ (Q, Σ), where Q is a UCQ and Σ is from the class $\mathcal{L}$ of TGDs, and a set $\mathfrak{E} \subseteq \mathcal{P}(D)$.
Question: Is $\mathfrak{E}$ the set of all MinEXs for (Q, Σ) in D?

Example 7. Suppose that we are interested in knowing whether a given set of proteins are all possible minimal covers of complexes. Consider the set $\mathfrak{E}$ given as:

$$\big\{\{protein(p_1), protein(p_3)\}, \{protein(p_1), protein(p_5)\}, \{protein(p_1), protein(p_6)\}, \{protein(p_2), protein(p_5)\}, \{protein(p_3), protein(p_4)\}, \{protein(p_3), protein(p_5)\}, \{protein(p_2), protein(p_4), protein(p_6)\}\big\}.$$

It is easy to verify that $\mathfrak{E}$ is precisely the set of all MinEXs for (Q_p, Σ_p) in D_p. ■

As before, we start with a rather general result for ALL-MINEX$(\mathrm{UCQ}, \mathcal{L})$, where by $\mathcal{C}$, we represent the complexity of OMQA in $\mathcal{L}$. We show that it is sufficient to perform a polynomial number of $\mathcal{C}$ checks and a single co-$(\mathrm{NP}^{\mathcal{C}})$ check. More specifically, given a set $\mathfrak{E}$ of subsets of the database, to decide ALL-MINEX$(\mathrm{UCQ}, \mathcal{L})$, we can proceed as follows. We perform a polynomial number of $\mathcal{C}$ checks to decide whether all sets in $\mathfrak{E}$ entail (Q, Σ). Then, we need to decide whether all sets in $\mathfrak{E}$ are minimal, and there is no

$\mathcal{L}$	**Data**	***fp*-comb.**	***ba*-comb.**	**Comb.**
L, LF, AF	$\leq$ P	D^{P}	D^{P}	PSPACE
S, SF	$\leq$ P	D^{P}	D^{P}	EXP
A	$\leq$ P	D^{P}	$\mathrm{D}^{\mathrm{Exp}}$	$\mathrm{D}^{\mathrm{Exp}}$
G	CONP	D^{P}	EXP	2EXP
F, GF	CONP	D^{P}	D^{P}	EXP
WS, WA	CONP	D^{P}	2EXP	2EXP

Table 3: Complexity results for ALL-MINEX(UCQ, $\mathcal{L}$).

MinEX that is not in $\mathfrak{E}$. This holds if there is no $E' \subseteq D$ entailing (Q, Σ) such that $E' \nsubseteq E$ for all $E \in \mathfrak{E}$. The complement task of guessing a set E' such that $E' \nsubseteq E$ for all $E \in \mathfrak{E}$ and that entails (Q, Σ) is in $\mathrm{NP}^{\mathcal{C}}$, and thus the task of checking whether all sets in $\mathfrak{E}$ are minimal, and there is no MinEX, which is not included in $\mathfrak{E}$, is in co-($\mathrm{NP}^{\mathcal{C}}$).

Theorem 8. ALL-MINEX(UCQ, $\mathcal{L}$) *can be decided by a polynomial number of* $\mathcal{C}$ *checks, followed by a single co-*($\mathrm{NP}^{\mathcal{C}}$) *check, where* $\mathcal{C}$ *is the complexity of OMQA in* $\mathcal{L}$*.*

Importantly, Theorem 8 gives a tight upper bound for all results in Table 3, apart from the data complexity results for FO-rewritable languages. In fact, we show that, ALL-MINEX(UCQ, $\mathcal{L}$) is feasible in polynomial time provided that $\mathcal{L}$ is FO-rewritable, which is summarized in the next result.

Theorem 9. *Let* $\mathcal{L}$ *be a FO-rewritable language over existential rules. Then, computing all MinEXs for a OMQ* (Q, Σ) *in a database* D *over* $\mathcal{L}$ *is feasible in polynomial time in data complexity.*

To prove this result, it suffices to consider the FO-rewriting of the program, and show that determining minimal subsets of a database that entail the rewritten query can be done in polynomial time.

It remains to show the hardness results presented in Table 3. As before, we note that some of the lower bounds immediately follow from the complexity of OMQA in the respective language. We show that ALL-MINEX(UCQ, GF) is CONP-hard in data complexity, by a reduction from UNSAT, by borrowing ideas from [Lukasiewicz *et al.*, 2018].

Theorem 10. ALL-MINEX(UCQ, GF) *is* CONP*-hard in data complexity.*

This implies that ALL-MINEX(UCQ, $\mathcal{L}$) is CONP-hard in data complexity for all languages $\mathcal{L} \in \{\mathsf{G}, \mathsf{F}, \mathsf{WS}, \mathsf{WA}\}$, due to the language inclusions $\mathsf{GF} \subset \mathsf{G}, \mathsf{F}, \mathsf{WS}, \mathsf{WA}$.

The following result settles the hardness results for ALL-MINEX(UCQ, $\mathcal{L}$) in *fp*-, *ba*-, and combined complexity. In particular, we have that the complexity of ALL-MINEX(UCQ, $\mathcal{L}$) and IS-MINEX(UCQ, $\mathcal{L}$) match for all considered languages $\mathcal{L}$ in *fp*-, *ba*-, and combined complexity. The hardness results for ALL-MINEX(UCQ, $\mathcal{L}$) are an adaptation of the proofs for IS-MINEX(UCQ, $\mathcal{L}$) in most cases, and hence we omit the details.

Theorem 11. *The fp-combined, ba-combined, and combined complexity hardness results in Table 3 hold for* ALL-MINEX(UCQ, $\mathcal{L}$).

$\mathcal{L}$	**Data**	***fp*-comb.**	***ba*-comb.**	**Comb.**
L, LF, AF	$\leq$ P	NP	NP	PSPACE
S, SF	$\leq$ P	NP	NP	EXP
A	$\leq$ P	NP	NEXP	NEXP
G	NP	NP	EXP	2EXP
F, GF	NP	NP	NP	EXP
WS, WA	NP	NP	2EXP	2EXP

Table 4: Complexity results for MINEX-IRREL(UCQ, $\mathcal{L}$) and for SMALL-MINEX(UCQ, $\mathcal{L}$).

This result concludes our complexity analysis for ALL-MINEX(UCQ, $\mathcal{L}$).

6 Explanations Excluding Forbidden Sets

The next problem that we consider is the one of finding a minimal explanation that does not include a given sets of facts. Let $\mathfrak{F}$ be a set of subsets of a database D, which intuitively encodes a set of invalid configurations: elements of $\mathfrak{F}$ may be known to be erroneous, or we may want to avoid them for some other reason, depending on the application. Thus, we are interested in finding whether there is an explanation that is not a superset of any of the sets in $\mathfrak{F}$, as formalized next.

Problem: MINEX-IRREL(UCQ, $\mathcal{L}$)
Input: A database D, an OMQ (Q, Σ), where Q is a UCQ and Σ is from the class $\mathcal{L}$ of TGDs, and a set $\mathfrak{F} \subseteq \mathcal{P}(D)$.
Question: Is there a MinEX E for (Q, Σ) in D such that $F \nsubseteq E$, for every $F \in \mathfrak{F}$?

Example 12. Suppose that the set

$$\mathfrak{F} = \{\{protein(p_1)\}, \{protein(p_3), protein(p_5)\}, \{protein(p_2), protein(p_4), protein(p_6)\}\}$$

encodes the configurations of proteins that are not allowed to be in a cover. In this case, $\{protein(p_3), protein(p_4)\}$ is a MinEX, since it is a cover that does not contain any configuration from $\mathfrak{F}$. ■

MINEX-IRREL(UCQ, $\mathcal{L}$) can be decided as follows. Let $\mathcal{C}$ be an oracle for query answering over $\mathcal{L}$. To decide the existence of a MinEX not including the "forbidden" sets, it is sufficient to guess such a subset of a database and then check whether it entails the OMQ using an oracle $\mathcal{C}$. This can be carried out in $\mathrm{NP}^{\mathcal{C}}$. Note that there is no need to check minimality as, if there is a subset E of a database that does not contain any of "forbidden" sets and entails the OMQ, then E has a minimal subset with these properties (due to monotonicity of the entailment relation).

Theorem 13. MINEX-IRREL(UCQ, $\mathcal{L}$) *can be decided in* $\mathrm{NP}^{\mathcal{C}}$*, where* $\mathcal{C}$ *is the complexity of OMQA in* $\mathcal{L}$*. If* $\mathcal{C} =$ NP *(resp.,* $\mathcal{C} =$ NEXP*), then* MINEX-IRREL(UCQ, $\mathcal{L}$) *is also complete for* NP *(resp.,* NEXP*).*

This result above gives a tight upper bound for all results in Table 4, apart from the data complexity results for FO-rewritable languages. For these languages, we know by Theorem 9 that it is possible to compute the set of all MinEXs in polynomial time. But then, we can also find a MinEX that does not contain as a subset any of the "forbidden" sets in polynomial time.

$\mathcal{L}$	Data	fp-comb.	ba-comb.	Comb.
L, LF, AF	$\leq$ P	Σ_2^P	Σ_2^P	PSPACE
S, SF	$\leq$ P	Σ_2^P	Σ_2^P	EXP
A	$\leq$ P	Σ_2^P	$\mathrm{P}^{\mathrm{NEXP}}$	$\mathrm{P}^{\mathrm{NEXP}}$
G	NP	Σ_2^P	EXP	2EXP
F, GF	NP	Σ_2^P	Σ_2^P	EXP
WS, WA	NP	Σ_2^P	2EXP	2EXP

Table 5: Complexity results for MINEX-REL(UCQ, $\mathcal{L}$) and for LARGE-MINEX(UCQ, $\mathcal{L}$).

Theorem 14. *Let $\mathcal{L}$ be a FO-rewritable language over existential rules. Then, finding a MinEX for an OMQ (Q, Σ) in a database D over $\mathcal{L}$ that does not contain any of the sets in $\mathfrak{F}$ is feasible in polynomial time in data complexity.*

This result implies that MINEX-IRREL(UCQ, $\mathcal{L}$) can be decided in polynomial-time in data complexity for FO-rewritable languages $\mathcal{L}$.

The obvious next step is to understand the behavior of the languages that are not FO-rewritable. Our next result states that MINEX-IRREL(UCQ, $\mathcal{L}$) is NP-hard for all such languages. The NP-hardness is obtained via a reduction from the NP-complete problem PATH WITH FORBIDDEN PAIRS [Gabow *et al.*, 1976; Garey and Johnson, 1990]: decide whether there exists a path between two vertices in a graph avoiding a set of given pairs of edges. We encode the reachability in the rules, while in the database we have the facts for the graph edges. The forbidden sets naturally encode the set of forbidden pairs of edges.

Theorem 15. MINEX-IRREL(UCQ, GF) *is* NP-*hard in data complexity.*

The hardness results of MINEX-IRREL(UCQ, $\mathcal{L}$) in the *fp*-combined, *ba*-combined, and combined complexity follow from the hardness of OMQA in the respective languages. All result are summarized in Table 4.

7 Explanations Including Distinguished Facts

We now investigate the problem of deciding whether there is a minimal explanation including a given fact.

Problem: MINEX-REL(UCQ, $\mathcal{L}$)
Input: A database D, an OMQ (Q, Σ), where Q is a UCQ and Σ is from the class $\mathcal{L}$ of TGDs, and a fact $\psi \in D$.
Question: Is there a MinEX E for (Q, Σ) in D such that $\psi \in E$?

Example 16. Suppose that we are interested in covers that contain the protein $\psi = protein(p_6)$ which is a distinguished fact. Observe, for example, that $\{protein(p_1), protein(p_6)\}$ and $\{protein(p_2), protein(p_4), protein(p_6)\}$ are MinEXs for $\{Q_p, \Sigma_p\}$ in D_p, containing the fact ψ. ■

To check the existence of a MinEX that contains a distinguished fact ψ, we can guess a candidate MinEX E, containing ψ and then use an oracle for IS-MINEX(UCQ, $\mathcal{L}$) to check whether E is a MinEX. This gives us a naive method to decide MINEX-REL(UCQ, $\mathcal{L}$).

Theorem 17. MINEX-REL(UCQ, $\mathcal{L}$) *can be decided by a computation in* $\mathrm{NP}^{\text{IS-MINEX}(\text{UCQ},\mathcal{L})}$.

Theorem 17 covers all membership results given in Table 5 for MINEX-REL(UCQ, $\mathcal{L}$) apart from the data complexity results for FO-rewritable languages. For these languages, it is a straightforward consequence of Theorem 9 that finding a MinEX containing a distinguished fact is in polynomial time.

Theorem 18. *Let $\mathcal{L}$ be a FO-rewritable language over existential rules. Then, finding a MinEX for an OMQ (Q, Σ) in a database D over $\mathcal{L}$ that contains a fact ψ is feasible in polynomial time in data complexity.*

As before, we again obtain a hardness result for languages that are not FO-rewritable: MINEX-REL(UCQ, $\mathcal{L}$) is NP-complete in data complexity for these languages $\mathcal{L}$.

Theorem 19. MINEX-REL(UCQ, GF) *is* NP-*hard in data complexity.*

To proof is via a reduction from the NP-complete problem PATH-VIA-NODE [Lapaugh and Papadimitriou, 1984]: given a graph, decide whether there is a path between two vertices passing through a third vertex. The construction is quite similar to the one used to show the NP-hardness of MINEX-IRREL(UCQ, GF) in data complexity.

Theorem 20 shows that the complexity of MINEX-REL(UCQ, $\mathcal{L}$) goes at least one level higher in the polynomial hierarchy, if we focus on *fp*-combined and *ba*-combined complexity. The reduction is from $\mathrm{QBF}^{CNF}_{2,\forall,\neg}$ which is known to be Σ_2^P-complete: given a quantified Boolean formula $\Phi = \exists X \forall Y \neg\phi(X, Y)$, where ϕ is a 3CNF formula, decide whether Φ is valid. The reduction is obtained by using the idea of the 'jolly' introduced in the proof of the D^{P}-hardness in the *fp*-combined complexity of IS-MINEX(UCQ, $\mathcal{L}$).

Theorem 20. MINEX-REL(UCQ, $\mathcal{L}$) *is* Σ_2^P*-hard for languages* $\mathcal{L} \in \{\mathsf{LF}, \mathsf{AF}, \mathsf{SF}\}$ *in fp- and ba-combined complexity.*

MINEX-REL(UCQ, $\mathcal{L}$) is also Σ_2^P-hard in the *fp*-combined and *ba*-combined complexity for all other languages considered as a result of language inclusions.

Our final result concerns the class A: we show that MINEX-REL(UCQ, A)is $\mathrm{P}^{\mathrm{NEXP}}$-hard in these cases, by a reduction from the following $\mathrm{P}^{\mathrm{NEXP}}$-complete problem that is the complement of a problem in [Eiter *et al.*, 2016]: given a triple (m, TP_1, TP_2), where m is an integer in unary notation, and TP_1 and TP_2 are two tiling problems for the exponential square $2^n \times 2^n$, decide whether there exists an initial condition w of length m, such that TP_1 has no solution with w, and TP_2 has a solution with w. The proof extends the ideas used for the $\mathrm{D}^{\mathrm{EXP}}$-hardness proof of IS-MINEX.

Theorem 21. MINEX-REL(UCQ, A) *is* $\mathrm{P}^{\mathrm{NEXP}}$*-hard in ba-combined complexity.*

The other hardness results in Table 5 follow from the hardness of query answering in the respective languages.

8 Cardinality-Based Explanation Problems

In this section, we deal with cardinality-related problems for minimal explanations. Briefly, these problems are helpful when we want to find out whether there is a MinEX smaller or larger than a given size.

Problem: SMALL-MINEX(UCQ, $\mathcal{L}$)
Input: A database D, an OMQ (Q, Σ), where Q is a UCQ and Σ is from the class $\mathcal{L}$ of TGDs, and an integer $n \geq 1$.
Question: Is there a MinEX E for (Q, Σ) in D such that $|E| \leq n$?

Problem: LARGE-MINEX(UCQ, $\mathcal{L}$)
Input: A database D, an OMQ(Q, Σ), where Q is a UCQ and Σ is from the class $\mathcal{L}$ of TGDs, and an integer $n \geq 1$.
Question: Is there a MinEX E for (Q, Σ) in D such that $|E| \geq n$?

Example 22. Let us take $n = 2$. Then, there is a MinEX for (Q_p, Σ_p) in D_p smaller and larger than n. On the other hand, if we take $n = 4$, then there is a MinEX smaller than n, but there is no MinEX larger than n, since all MinEXs for $\{Q_p, \Sigma_p\}$ in D_p are of size at most 3. ∎

Most of the proofs of the results for the problems SMALL-MINEX(UCQ, $\mathcal{L}$) and LARGE-MINEX(UCQ, $\mathcal{L}$) are a result of adaptations of the proofs given for MINEX-IRREL(UCQ, $\mathcal{L}$) and MINEX-REL(UCQ, $\mathcal{L}$), respectively. Hence, we omit the details here.

Theorem 23. *The complexity results in Table 4 and Table 5 hold for* SMALL-MINEX(UCQ, $\mathcal{L}$) *and* LARGE-MINEX(UCQ, $\mathcal{L}$)*, respectively.*

9 Related Work

The study of explanations and diagnosis in logical formalisms dates back to Reiter [1987]. From a broader perspective, our study can be seen as a form a logical abduction, but our results clearly differ from those in propositional abduction [Eiter and Gottlob, 1995].

In this work, we focus on ontology languages, and build on axiom pinpointing [Kalyanpur *et al.*, 2007; Baader and Suntisrivaraporn, 2008; Peñaloza and Sertkaya, 2017]. In axiom pinpointing, an entailment is explained in terms of a minimal set of ontological axioms. Such explanations are called *justifications* in the DL literature [Horridge *et al.*, 2008; Horridge *et al.*, 2009]. Axiom pinpointing is extensively studied in DLs, and some implementations exist [Kalyanpur *et al.*, 2007; Sebastiani and Vescovi, 2009].

Most of the existing approaches to explanations focus on classical reasoning tasks and the associated types of entailments. The problem of explaining query entailments has only been investigated for the *DL-Lite* family of languages [Borgida *et al.*, 2008]. Our work provides a different framework inspired by axiom pinpointing and the associated problems. Another work for explaining query answers for the *DL-Lite* family is given in the context of consistent query answering [Bienvenu *et al.*, 2019]. Our minimal explanations can be seen analogous to the notion of *causes* studied in [Bienvenu *et al.*, 2019]. There are many differences in our approach, though. We are interested in explaining query entailments in the most general fashion (even if there is no inconsistency), and present a unifying perspective for tasks that require explanations. The only work related to explanations in existential rules is given in [Ceylan *et al.*, 2017], where explanations for OMQs under existential rules are studied, but this study is relative to *probabilistic databases* and hence of a very different flavor.

There are interesting *model-theoretic* connections with our framework and more basic formalisms. For instance, for most of the languages that we study, we can define *disjunctive* Datalog programs [Eiter *et al.*, 1997] such that every minimal model of a disjunctive Datalog program will be in bijection with a minimal explanation. These model-theoretic connections are very important, as they reveal the power of the studied problems in terms of well-studied languages.

10 Summary and Outlook

In this paper, we have started a new direction of research by translating several decision problems from axiom pinpointing to provide explanations for OMQs. We have studied the problem of explaining query answers in terms of minimal subsets of database facts, and provided a thorough complexity analysis for several decision problems associated with minimal explanations under existential rules.

The problems investigated in this paper are also closely related to minimal hitting set problems, which have a number of applications in fault diagnosis, computational biology, and data mining [Gainer-Dewar and Vera-Licona, 2017; Gottlob and Malizia, 2018]. Indeed, many important problems in practice (such as protein covers) can be naturally formulated in our framework in terms of ontology-mediated queries, and we hope that our work will be a basis for encoding and solving problems in various application domains of ontologies.

There are many interesting directions for future work, including the study of other ontology languages. We also aim to explore the model-theoretic connections to other formalisms, and make a more fine-grained complexity analysis. There are many other types of problems encountered in the context of explanations, which are also a subject of future study.

Acknowledgments

We are thankful to the anonymous reviewers of this paper. 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

[Baader and Suntisrivaraporn, 2008] Franz Baader and Boontawee Suntisrivaraporn. Debugging SNOMED CT using axiom pinpointing in the description logic EL+. In *Proc. KR-MED*, 2008.

[Baader *et al.*, 2007] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2nd edition, 2007.

[Bard and Rhee, 2004] Jonathan B. L. Bard and Seung Y. Rhee. Ontologies in biology: Design, applications and future challenges. *Nat. Rev. Genet.*, 5:213–222, 2004.

[Beeri and Vardi, 1981] Catriel Beeri and Moshe Y. Vardi. The implication problem for data dependencies. In *Proc. ICALP*, pages 73–85, 1981.

[Bertaud-Gounot *et al.*, 2012] Valérie Bertaud-Gounot, Régis Duvauferrier, and Anita Burgun. Ontology and medical diagnosis. *Inform. Health Soc. Care*, 37(2):51–61, 2012.

[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.

[Bienvenu *et al.*, 2019] Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Computing and explaining query answers over inconsistent DL-Lite knowledge bases. *J. Artif. Intell. Res.*, 64:563–644, 2019.

[Borgida *et al.*, 2008] Alexander Borgida, Diego Calvanese, and Mariano Rodriguez-Muro. Explanation in the DL-Lite family of description logics. In *Proc. OTM*, pages 1440–1457, 2008.

[Calì *et al.*, 2012a] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *J. Web Semant.*, 14:57–83, 2012.

[Calì *et al.*, 2012b] Andrea Calì, Georg Gottlob, and Andreas Pieris. Towards more expressive ontology languages: The query answering problem. *Artif. Intell.*, 193:87–128, 2012.

[Calì *et al.*, 2013] 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.

[Ceylan *et al.*, 2017] İsmail İlkan Ceylan, Stefan Borgwardt, and Thomas Lukasiewicz. Most probable explanations for probabilistic database queries. In *Proc. IJCAI*, pages 950–956, 2017.

[Došilović *et al.*, 2018] Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. Explainable artificial intelligence: A survey. In *Proc. MIPRO*, pages 210–215, 2018.

[Eiter and Gottlob, 1995] Thomas Eiter and Georg Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM J. Comput.*, 24(6):1278–1304, 1995.

[Eiter *et al.*, 1997] Thomas Eiter, Georg Gottlob, and Heikki Mannila. Disjunctive datalog. *ACM Trans. Database Syst.*, 22(3):364–418, 1997.

[Eiter *et al.*, 2016] Thomas Eiter, Thomas Lukasiewicz, and Livia Predoiu. Generalized consistent query answering under existential rules. In *Proc. KR*, pages 359–368, 2016.

[Fagin *et al.*, 2005] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. *TCS*, 336(1):89–124, 2005.

[Gabow *et al.*, 1976] Harold N. Gabow, Shachindra N. Maheshwari, and Leon J. Osterweil. On two problems in the generation of program test paths. *IEEE Trans. Softw. Eng.*, SE-2(3):227–231, 1976.

[Gainer-Dewar and Vera-Licona, 2017] Andrew Gainer-Dewar and Paola Vera-Licona. The minimal hitting set generation problem: Algorithms and computation. *SIAM J. Discrete Math.*, 31(1):63–100, 2017.

[Garey and Johnson, 1990] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1990.

[Gottlob and Malizia, 2018] Georg Gottlob and Enrico Malizia. Achieving new upper bounds for the hypergraph duality problem through logic. *SIAM J. Comput.*, 47(2):456–492, 2018.

[Horridge *et al.*, 2008] Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Laconic and precise justifications in OWL. In *Proc. ISWC*, pages 323–338, 2008.

[Horridge *et al.*, 2009] Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Explaining inconsistencies in OWL ontologies. In *Proc. SUM*, pages 124–137, 2009.

[Kalyanpur *et al.*, 2007] Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding all justifications of OWL DL entailments. In *Proc. ISWC/ASWC*, pages 267–280, 2007.

[Klamt *et al.*, 2009] Steffen Klamt, Utz-Uwe Haus, and Fabian Theis. Hypergraphs and cellular networks. *PLOS Comput. Biol.*, 5(5):e1000385, 2009.

[Lapaugh and Papadimitriou, 1984] Andrea S. Lapaugh and Christos H. Papadimitriou. The even-path problem for graphs and digraphs. *Networks*, 14(4):507–513, 1984.

[Lukasiewicz *et al.*, 2018] Thomas Lukasiewicz, Enrico Malizia, and Cristian Molinaro. Complexity of approximate query answering under inconsistency in Datalog+/–. In *Proc. IJCAI*, pages 1921–1927, 2018.

[Peñaloza and Sertkaya, 2017] Rafael Peñaloza and Barış Sertkaya. Understanding the complexity of axiom pinpointing in lightweight description logics. *Artif. Intell.*, 250:80–104, 2017.

[Poggi *et al.*, 2008] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. In *Journal on Data Semantics X*, pages 133–173. Springer-Verlag, 2008.

[Ramadan *et al.*, 2004] Emad Ramadan, Arijit Tarafdar, and Alex Pothen. A hypergraph model for the yeast protein complex network. In *Proc. IPDPS*, 2004.

[Reiter, 1987] Raymond Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32(1):57–95, 1987.

[Sebastiani and Vescovi, 2009] Roberto Sebastiani and Michele Vescovi. Axiom pinpointing in lightweight description logics via Horn-SAT encoding and conflict analysis. In *Proc. CADE*, pages 84–99, 2009.

[Suntisrivaraporn *et al.*, 2008] Boontawee Suntisrivaraporn, Guilin Qi, Qiu Ji, and Peter Haase. A modularization-based approach to finding all justifications for OWL DL entailments. In *Proc. ASWC*, pages 1–15, 2008.

[Vardi, 1982] Moshe Y. Vardi. The complexity of relational query languages. In *Proc. STOC*, pages 137–146, 1982.