Preferred Explanations for Ontology-Mediated Queries under Existential Rules

İsmail İlkan Ceylan[1], **Thomas Lukasiewicz**[1], **Enrico Malizia**[2], **Cristian Molinaro**[3], **Andrius Vaicenavičius**[1]
[1] Department of Computer Science, University of Oxford, UK
[2] DISI, University of Bologna, Italy
[3] DIMES, University of Calabria, Italy
{ismail.ceylan, thomas.lukasiewicz, andrius.vaicenavicius}@cs.ox.ac.uk, enrico.malizia@unibo.it, cristian.molinaro@unical.it

Abstract

Recently, explanations for query answers under existential rules have been investigated, where an explanation is an inclusion-minimal subset of a given database that, together with the ontology, entails the query. In this paper, we take a step further and study explanations under different minimality criteria. In particular, we first study cardinality-minimal explanations and hence focus on deriving explanations of minimum size. We then study a more general preference order induced by a weight distribution. We assume that every database fact is annotated with a (penalization) weight, and we are interested in explanations with minimum overall weight. For both preference orders, we study a variety of explanation problems, such as recognizing a preferred explanation, all preferred explanations, a relevant or necessary fact, and the existence of a preferred explanation not containing forbidden sets of facts. We provide a detailed complexity analysis for all the aforementioned problems, thereby providing a more complete picture for explaining query answers under existential rules.

Introduction

Ontology-based data access (Poggi et al. 2008) emerged as a paradigm for better means of querying data sources, and has become one of the focal points of research in knowledge representation and reasoning. The core idea is to enrich user queries with an *ontology*, which encodes the background knowledge over the application domain. Intuitively, ontological knowledge provides a conceptual abstraction over the domain, and it helps to deduce more facts from (possibly incomplete) data sources, resulting in more complete sets of answers to user queries. It is common practice to view the ontology and the user query as a composite query, called *ontology-mediated query (OMQ)*. The task of evaluating such queries is then called *ontology-mediated query answering (OMQA)* (Bienvenu et al. 2014).

Description logics (DLs) (Baader et al. 2007) and existential rules (a.k.a. Datalog$^{\pm}$) (Calì, Gottlob, and Kifer 2013; Calì, Gottlob, and Lukasiewicz 2012) are two families of logic-based knowledge representation languages, which are commonly used to formulate ontologies. OMQA relative to languages in these families is extensively studied and well-understood. Numerous systems have been developed to support OMQA and related tasks (Calvanese et al. 2017; Nenov et al. 2015; Bellomarini, Sallinger, and Gottlob 2018).

Broadly speaking, a major challenge in artificial intelligence systems is in explaining the various conclusions drawn by such systems. One advantage of logic-based systems is that they are well-suited to explain various logical inferences in a principled manner. In fact, this can be achieved in various ways, depending on the desired form of an *explanation*. Given a conclusion, derived from a knowledge base, an explanation could be a *proof* relative to the underlying deduction calculi, or alternatively, it could be a *set of axioms and/or facts*, responsible for the derived conclusion.

There is a large body of work for deriving explanations in DLs, which can be dated back to earlier works in the literature (McGuinness and Borgida 1995; Borgida, Franconi, and Horrocks 2000). Explanations for classical reasoning tasks (Kalyanpur et al. 2007; Baader and Suntisrivaraporn 2008; Peñaloza and Sertkaya 2017) as well as for OMQA (Borgida, Calvanese, and Rodriguez-Muro 2008; Ceylan et al. 2020b) have been widely studied ever since. Contrastingly, literature for existential rules is very sparse, which motivated recent work (Ceylan et al. 2019), aiming at explaining OMQA under existential rules. The idea is to view every explanation as an *inclusion-minimal* subset of a given database that, together with the ontology, entails the query. Defining explanations as subsets of the database that entail an OMQ is particularly suitable for monotone queries.

In this work, we take a step further for explaining OMQA under existential rules, and consider *cardinality-* and *weight-minimal* explanations. A cardinality-minimal (resp., weight-minimal) explanation is a database subset that is smallest in size (resp., has the smallest weight w.r.t. to a weight function), and together with the ontology entails the query. In the weight case, we assume the weight function to be given by annotating every database fact with a weight, representing a *penalization degree* for the corresponding fact, i.e., the higher the weight of a fact the less it is to be preferred in an explanation. Observe that cardinality-minimal explanations are a special case of weight-minimal explanations, when all facts have the same weight. Clearly, every cardinality-minimal explanation is also an inclusion-minimal explanation. Finally, we note that the study of different minimality criteria, based on e.g. weights and cardinality, is common in other contexts, including consistent query answering (Bienvenu, Bourgaux, and Goasdoué 2014; Lukasiewicz, Malizia, and Molinaro 2018; Lukasiewicz, Malizia, and Vaicenavičius 2019).

We argue in detail how weight- and cardinality-minimal explanations can be useful in certain application domains. As a concrete example, we consider a road map as part of our running example, where the task is to identify the most *preferred route* from the current station to a target station, specified by a query. Suppose that the preference is for the *fastest route*, then every fact corresponding to connections between stations can be annotated with a weight indicating the time between the stations, and every weight-minimal explanation then corresponds to the most *time-efficient route*.

For the aforementioned preference orders, we study five problems, namely, deciding whether a given set of facts is a preferred explanation (IS-MINEX), deciding whether a given collection of sets of facts contains exactly *all* preferred explanations (ALL-MINEX), deciding whether there exists some preferred explanation containing a *distinguished fact* (MINEX-REL), deciding whether there exists some preferred explanation *not* containing any of the *forbidden sets* of facts (MINEX-IRREL), and deciding whether *all* preferred explanations contain a *distinguished fact* (MINEX-NEC).

We conduct a detailed complexity analysis for each of the problems introduced, and provide a host of complexity results for a large class of existential rules, which can be naturally extended to other existential rule languages. Our findings show that the complexity of these problems are in most cases different from the inclusion-minimal case, given by Ceylan et al. (2019), and require different techniques and reductions. We show that the complexity results for the cardinality- and weight-minimal cases coincide for IS-MINEX and ALL-MINEX, but differ in all the remaining problems, where the weight-minimal case is computationally harder.

Preliminaries

In this section, we recall some basics on existential rules (Calì, Gottlob, and Pieris 2012; Calì, Gottlob, and Kifer 2013; Calì, Gottlob, and Lukasiewicz 2012) and the paradigm of ontology-mediated query answering, and give some complexity-theoretic background relevant to our study.

General

We assume a relational vocabulary of mutually disjoint (possibly infinite) sets $\mathbf{R}$, $\mathbf{C}$, $\mathbf{N}$, and $\mathbf{V}$ of *predicates*, *constants*, *nulls*, and *variables*, respectively. Each predicate is associated with an *arity* (a non-negative integer). A *term* is a constant, a null, or a variable. An *atom* is an expression $p(t_1, \ldots, t_n)$, 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. Conjunctions of atoms are often identified with the sets of their atoms.

An *instance* I is a (possibly infinite) set of atoms containing constants and nulls only. A *database* D is a finite instance that contains only constants. A *homomorphism* is a mapping $h\colon \mathbf{C} \cup \mathbf{N} \cup \mathbf{V} \to \mathbf{C} \cup \mathbf{N} \cup \mathbf{V}$ that is the identity on $\mathbf{C}$ and maps $\mathbf{N}$ to $\mathbf{C} \cup \mathbf{N}$. With a slight abuse of notation, homomorphisms are applied also to (sets of) atoms.

A *conjunctive query (CQ)* $Q(\mathbf{X})$ is a first-order formula of the form $\exists \mathbf{Y} \phi(\mathbf{X}, \mathbf{Y})$, where $\phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of null-free atoms. The *answer* to $Q(\mathbf{X})$ over an instance I, denoted $Q(I)$, is the set of all tuples $\mathbf{t} \in \mathbf{C}^{|\mathbf{X}|}$ for which there is a homomorphism h such that $h(\phi(\mathbf{X}, \mathbf{Y})) \subseteq I$ and $h(\mathbf{X}) = \mathbf{t}$. A *union of conjunctive queries (UCQ)* $Q(\mathbf{X})$ has the form $Q_1(\mathbf{X}) \vee \cdots \vee Q_n(\mathbf{X})$, where each $Q_i(\mathbf{X})$ is a CQ. The answer to $Q(\mathbf{X})$ over an instance I, denoted $Q(I)$, is defined as the set of tuples $\bigcup_{1 \leq i \leq n} Q_i(I)$. A *Boolean* UCQ Q is a UCQ where all variables are existentially quantified; Q is *true* over I, denoted $I \models Q$, if $Q(I) \neq \emptyset$. Here, we focus on Boolean UCQs and refer to them simply as UCQs.

Existential Rules

A *tuple-generating dependency (TGD)* σ is a first-order formula of the form $\forall \mathbf{X} \forall \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y}) \to \exists \mathbf{Z}\, \Psi(\mathbf{X}, \mathbf{Z})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ and $\Psi(\mathbf{X}, \mathbf{Z})$ are conjunctions of atoms without nulls, called the *body* and the *head* of the TGD, respectively, and denoted $body(\sigma)$ and $head(\sigma)$, respectively. Classes of TGDs are also known in the literature as *existential rules*, or *Datalog*$^{\pm}$ *languages*. An instance I satisfies σ, written $I \models \sigma$, if whenever there exists a homomorphism h such that $h(\Phi(\mathbf{X}, \mathbf{Y})) \subseteq I$, then there exists $h' \supseteq h|_{\mathbf{X}}$, where $h|_{\mathbf{X}}$ is the restriction of h on $\mathbf{X}$, such that $h'(\Psi(\mathbf{X}, \mathbf{Z})) \subseteq I$. For brevity, we omit the quantifiers in front of TGDs. A *program* (or *ontology*) is a finite set Σ of TGDs. An instance I satisfies Σ, written $I \models \Sigma$, if I satisfies every TGD of Σ.

We briefly recall the Datalog$^{\pm}$ languages that are relevant to our study, namely, linear (L) (Calì, Gottlob, and Lukasiewicz 2012), guarded (G) (Calì, Gottlob, and Kifer 2013), sticky (S) (Calì, Gottlob, and Pieris 2012), and acyclic TGDs (A), along with the "weak" (proper) generalizations weakly sticky (WS) (Calì, Gottlob, and Pieris 2012) and weakly acyclic TGDs (WA) (Fagin et al. 2005), and their "full" (i.e., existential-free) proper restrictions linear full (LF), guarded full (GF), sticky full (SF), and acyclic full TGDs (AF), and full TGDs (F) in general. We also recall the following further inclusions: $\mathsf{L} \subset \mathsf{G}$ and $\mathsf{F} \subset \mathsf{WA} \subset \mathsf{WS}$. If a program Σ belongs to a language $\mathcal{L}$, we also call Σ an *$\mathcal{L}$-program*; $\mathcal{L}_{\emptyset}$ denotes the language including only the empty program.

Ontology-Mediated Query Answering

An *ontology-mediated query (OMQ)* is a pair (Q, Σ), where Q is a query, and Σ is an ontology. Consider a database D and an OMQ (Q, Σ). The set of *models* of $\langle D, \Sigma \rangle$, denoted $mods(D, \Sigma)$, is the set of instances $\{I \mid I \supseteq D \wedge I \models \Sigma\}$. We say that D *entails* (Q, Σ), denoted $D \models (Q, \Sigma)$, if $I \models Q$ for every $I \in mods(D, \Sigma)$. *Ontology-mediated query answering* is the task of deciding whether $D \models (Q, \Sigma)$. We use $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ to refer to the ontology-mediated query answering problem when the query is a UCQ and the program is from $\mathcal{L}$. Table 1 summarizes the complexity for $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ in the different languages $\mathcal{L}$ considered.

An OMQ (Q, Σ) is *FO-rewritable*, if there is a first-order query Q_{Σ} such that, for all databases D, $D \models (Q, \Sigma)$ iff $D \models Q_{\Sigma}$; in which case, Q_{Σ} is an *FO-rewriting* of (Q, Σ). A class of programs $\mathcal{L}$ is *FO-rewritable*, if it admits an FO-rewriting for every UCQ and program in $\mathcal{L}$. Languages in Table 1 with AC^0 data complexity are FO-rewritable.

$\mathcal{L}$	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 $\text{OMQA}(\text{UCQ}, \mathcal{L})$ for existential rules.

Computational Complexity

In our complexity analysis, we make the standard assumptions (Vardi 1982): the *combined complexity* is evaluated by considering the database, the program, and the query, as part of the input. The *bounded-arity combined complexity* (or *ba-combined*) assumes that the maximum arity of the predicates in $\mathbf{R}$ is bounded by an integer constant. The *fixed-program combined complexity* (or *fp-combined*) is evaluated by considering the ontology as fixed. Finally, the *data complexity* is calculated by considering everything fixed but the database.

Besides the more standard complexity classes, we will also refer to the following ones. The complexity class AC^0 is the class of all decision problems that can be solved by uniform families of Boolean circuits of polynomial size and constant depth. The class $\text{D}^{\text{P}} = \text{NP} \wedge \text{coNP}$ (resp., $\text{D}^{\text{Exp}} = \text{NExp} \wedge \text{coNExp}$) is the class of all problems that are the conjunction of a problem in NP (resp., NEXP) and a problem in coNP (resp., coNEXP). The class Θ_2^{P} (resp., $\text{P}^{\text{NExp}[O(\log n)]}$) is the class of all problems that can be decided in polynomial time by a deterministic Turing machine with a logarithmic number of calls to an NP (resp., a NEXP) oracle; while Δ_2^{P} (resp., P^{NExp}) is the class of all problems that can be decided in polynomial time by a deterministic Turing machine with a polynomial number of calls to an NP (resp., a NEXP) oracle.

Preferred Explanations

In this section, we define the concept of preferred explanation, define the problems studied in the paper, and motivate them with the help of a concrete example.

Given a database D, a preorder $\preccurlyeq$ on $\mathcal{P}(D)$, and subsets E, E' of D, we write $E \prec E'$ if $E \preccurlyeq E'$ and $E' \not\preccurlyeq E$.

Definition 1 ($\preccurlyeq$-MinEX)**.** For a database D and an OMQ (Q, Σ) such that $D \models (Q, \Sigma)$, we say that $E \subseteq D$ is an *explanation* for $D \models (Q, \Sigma)$ if $E \models (Q, \Sigma)$.

Furthermore, given a preorder $\preccurlyeq$ on $\mathcal{P}(D)$, $E \subseteq D$ is a $\preccurlyeq$-*minimal explanation*, or $\preccurlyeq$*-MinEX*, for $D \models (Q, \Sigma)$ if it is an explanation for $D \models (Q, \Sigma)$ and it is $\preccurlyeq$*-minimal*, i.e., there is no explanation $E' \subseteq D$ for $D \models (Q, \Sigma)$ with $E' \prec E$.

Ceylan et al. (2019) considered $\subseteq$-MinEXs. Here we concentrate on two other fundamental preference orders, common in the literature (Eiter and Gottlob 1995; Eiter, Gottlob, and Leone 1997; Bienvenu, Bourgaux, and Goasdoué 2014), namely, orders induced by cardinality and a weight function.

$\leq$-MinEX: For preferences induced by *cardinality* ($\leq$), the preference is given to subsets of the database that are smaller in size. Given two sets D_1 and D_2, $D_1 \leq D_2$ iff $|D_1| \leq |D_2|$. Such preferences are particularly relevant for domains where explanations correspond to optimal answers that are composed of equally costly components.

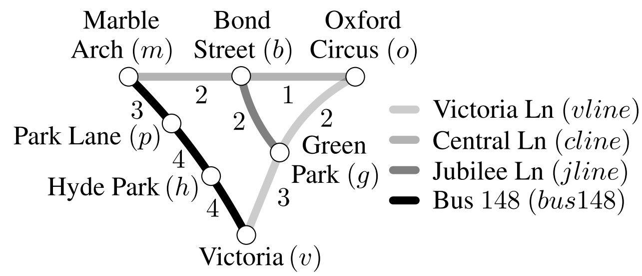

Figure 1: Excerpt from the London tube and bus map, where each color denotes a different line, as shown in the key.

$\leq_w$-MinEX: For preferences induced by *weights* ($\leq_w$), we assume that the facts in the database D are assigned (rational) weights by a function $w\colon D \to \mathbb{Q}$, which defines an order over the subsets of the database D such that $D_1 \leq_w D_2$ iff $\sum_{\alpha \in D_1} w(\alpha) \leq \sum_{\alpha \in D_2} w(\alpha)$, for any $D_1, D_2 \subseteq D$. The meaning of the weights is simple: facts with higher weights are less preferred in the explanations.

In the following, cardinality- and weight-minimal explanations are also more generally called preferred explanations.

Example 2. Consider the road map in Figure 1. The database $D = \{\mathit{stop}(v), \mathit{stop}(h), \mathit{stop}(p), \mathit{stop}(m), \mathit{stop}(g), \mathit{stop}(b), \mathit{stop}(o), \mathit{link}(\mathit{bus148}, v, h), \mathit{link}(\mathit{bus148}, h, p), \mathit{link}(\mathit{bus148}, p, m), \mathit{link}(\mathit{vline}, v, g), \mathit{link}(\mathit{vline}, g, o), \mathit{link}(\mathit{jline}, g, b), \mathit{link}(\mathit{cline}, o, b), \mathit{link}(\mathit{cline}, b, m), \mathit{reach}(v)\}$ encodes the stops, the links between them, and that we start at the Victoria (v) station.

The duration between stops and the changing time between lines can be represented by weights. We set $w(\mathit{stop}(i)) = 3$ for all stops i, i.e., three minutes to change line at any stop. Each *link* fact is assigned the weight capturing the duration of the travel along that link, as indicated in Figure 1, e.g., $w(\mathit{link}(\mathit{jline}, g, b)) = 2$. Also, $w(\mathit{reach}(v)) = 0$.

The program Σ encodes reachability, and ensures that changing a line requires getting off at a stop:

$$\mathit{link}(X, Y_1, Y_2) \to \mathit{link}(X, Y_2, Y_1),$$
$$\mathit{link}(X, Y_1, Y_2) \wedge \mathit{link}(X, Y_2, Y_3) \to \mathit{link}(X, Y_1, Y_3),$$
$$\mathit{reach}(Y_1) \wedge \mathit{link}(X, Y_1, Y_2) \wedge \mathit{stop}(Y_2) \to \mathit{reach}(Y_2).$$

We aim to travel to Marble Arch, so the query is $Q = \mathit{reach}(m)$. The fastest route to Marble Arch from Victoria corresponds to a $\leq_w$-MinEX for $D \models (Q, \Sigma)$. Then:

$$\begin{aligned} E_1 = \{&\mathit{reach}(v), \mathit{stop}(m), \mathit{link}(\mathit{bus148}, v, h), \\ &\mathit{link}(\mathit{bus148}, h, p), \mathit{link}(\mathit{bus148}, p, m)\}, \\ E_2 = \{&\mathit{reach}(v), \mathit{stop}(o), \mathit{stop}(m), \mathit{link}(\mathit{vline}, v, g), \\ &\mathit{link}(\mathit{vline}, g, o), \mathit{link}(\mathit{cline}, o, b), \\ &\mathit{link}(\mathit{cline}, b, m)\}, \end{aligned}$$

are both explanations with the minimal weight 14. ∎

Explanation Problems

In this section, we introduce a number of problems, adapted from Ceylan et al. (2019), associated with preferred explanations. We define these problems in a general way, which can be parameterized with any preference order $\preccurlyeq$. Our formulation is tightly connected to the abduction literature, and our problems can be seen as counterparts of those in abduction.

The first and most basic problem, IS-MINEX, is a decision version of the problem of finding a preferred explanation.

Problem: IS-MINEX($\preccurlyeq$, UCQ, $\mathcal{L}$)
Input: A database D, a set of facts $E \subseteq D$, and an OMQ (Q, Σ), where Q is a UCQ and Σ is an $\mathcal{L}$-program.
Question: Is E a $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$?

Example 3. E_1 and E_2 from Example 2 are weight-minimal explanations for $D \models (Q, \Sigma)$, and correspond to the fastest route from Victoria to Marble Arch station. The set

$$E_3 = \{reach(v), stop(g), stop(b), stop(m), \\ link(vline, v, g), link(jline, g, b), \\ link(cline, b, m)\},$$

where $E_3 \models (Q, \Sigma)$ and is inclusion-minimal, it is not a $\leq_w$-MinEX, as $w(E_3) = 16 > w(E_1) = 14$. ■

The following problem is a decision version of the problem of finding all preferred explanations.

Problem: ALL-MINEX($\preccurlyeq$, UCQ, $\mathcal{L}$)
Input: A database D, a set $\mathcal{E}$ of subsets of D, and an OMQ (Q, Σ), where Q is a UCQ and Σ is an $\mathcal{L}$-program.
Question: Is $\mathcal{E}$ the set of all $\preccurlyeq$-MinEXs for $D \models (Q, \Sigma)$?

The next problem asks whether a particular fact f is *relevant*, that is, f is contained in some preferred explanation. This problem is important to separate facts that contribute to a preferred explanation from those that do not.

Problem: MINEX-REL($\preccurlyeq$, UCQ, $\mathcal{L}$)
Input: A database D, a fact $f \in D$, and an OMQ (Q, Σ), where Q is a UCQ and Σ is an $\mathcal{L}$-program.
Question: Is there a $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$ including f?

We illustrate ALL-MINEX and MINEX-REL below.

Example 4. Observe that $\mathcal{E} = \{E_1, E_2\}$ are all the $\leq_w$-MinEXs for $D \models (Q, \Sigma)$. While there is no $\leq_w$-MinEX for $D \models (Q, \Sigma)$ that contains $stop(g)$, there is a $\leq_w$-MinEX for $D \models (Q, \Sigma)$ that contains $stop(o)$. ■

The next problem asks whether there is a preferred explanation that does not contain any *forbidden* set. This problem applies when we know that some combination of facts correspond to invalid configurations (e.g., invalid routes), and so we want to find explanations that do not contain these facts.

Problem: MINEX-IRREL($\preccurlyeq$, UCQ, $\mathcal{L}$)
Input: A database D, a set $\mathcal{F}$ of subsets of D, and an OMQ (Q, Σ), where Q is a UCQ and Σ is an $\mathcal{L}$-program.
Question: Is there a $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$ not containing any of the sets in $\mathcal{F}$?

Example 5. Suppose that the link between h and p cannot be used, i.e., $link(bus148, h, p)$ is forbidden. Then, E_2 gives a quickest route that does not contain this forbidden link.

The next problem asks whether a particular fact f is *necessary*, that is, f is contained in every preferred explanation. This problem is important for computing the core of a preferred explanation, which consists of the intersection of all preferred explanations—intuitively, a necessary fact is intrinsically related to the entailment.

Problem: MINEX-NEC($\preccurlyeq$, UCQ, $\mathcal{L}$)
Input: A database D, a fact $f \in D$, and an OMQ (Q, Σ), where Q is a UCQ and Σ is an $\mathcal{L}$-program.
Question: Does every $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$ contain f?

Example 6. Facts $reach(v)$ and $stop(m)$ are the only two necessary facts for any $\leq_w$-minimal explanation for $D \models (Q, \Sigma)$. Any other fact is not necessary. This can be seen by taking the intersection of the two $\leq_w$-minimal explanations, which is $E_1 \cap E_2 = \{reach(v), stop(m)\}$. ■

Complexity Analysis

In this section, we analyze the complexity of the explanation problems presented above, for cardinality- and weight-minimal explanations. Our results are reported in Tables 2 to 4. We first discuss membership and then hardness results.

Membership Results

To show the membership results we proceed as follows. First, we provide general results (Theorems 7 and 8), relying on general algorithms, which imply all the membership results in Table 2 (resp., Tables 3 and 4) apart from the $\mathrm{D^P}$ (resp., Θ_2^{P}, Δ_2^{P}) ones in the *fp*-combined and *ba*-combined complexity and the P ones, for which we will need tighter statements. Then, we provide tighter upper bounds (Theorems 9 and 10) for the case where OMQA(UCQ, $\mathcal{L}$) is in NP, thereby establishing the $\mathrm{D^P}$ (resp., Θ_2^{P}, Δ_2^{P}) membership results in in Table 2 (resp., Tables 3 and 4) in the *fp*-combined and *ba*-combined complexity. Finally, we show membership in P for FO-rewritable languages in the data complexity for all the problems that we consider (Theorem 11).

We start by general results applying to all cases (even if the resulting upper bounds are not tight in some cases).

The first problems considered are IS-MINEX and ALL-MINEX, which turn out to have the same complexity, also for the two minimality criteria that we consider, as shown in Table 2. Intuitively, for IS-MINEX, deciding whether E is a $\preccurlyeq$-MinEX requires to check that E entails the OMQ and that there is no subset of the database $E' \prec E$ that entails the OMQ. If the complexity of OMQA is in $\mathcal{C}$, then the first check is in $\mathcal{C}$, and the second check is the complement of checking whether there is a subset of the database $E' \prec E$ that entails the OMQ, and so is in co-($\mathrm{NP}^{\mathcal{C}}$). This idea can be generalized to ALL-MINEX (Ceylan et al. 2019).

Theorem 7. *If* OMQA(UCQ, $\mathcal{L}$) *is in the complexity class $\mathcal{C}$ in the combined (resp., ba-combined, fp-combined, and data) complexity, then* IS-MINEX($\preccurlyeq$, UCQ, $\mathcal{L}$) *(resp.,* ALL-MINEX($\preccurlyeq$, UCQ, $\mathcal{L}$)*) can be decided by a test (resp., a linear number of tests) in $\mathcal{C}$ followed by a test in co-($\mathrm{NP}^{\mathcal{C}}$), in the combined (resp., ba-combined, fp-combined, and data) complexity, where $\preccurlyeq\ \in \{\leq, \leq_w\}$.*

$\mathcal{L}$	**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	coNP	$\mathrm{D^P}$	EXP	2EXP
F, GF	coNP	$\mathrm{D^P}$	$\mathrm{D^P}$	EXP
WS, WA	coNP	$\mathrm{D^P}$	2EXP	2EXP

Table 2: Complexity of IS-MINEX$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$ and ALL-MINEX$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$, where $\preccurlyeq \in \{\leq, \leq_w\}$.

$\mathcal{L}$	**Data**	*fp*-**comb.**	*ba*-**comb.**	**Comb.**
L, LF, AF	$\leq$ P	Θ_2^{P}	Θ_2^{P}	PSPACE
S, SF	$\leq$ P	Θ_2^{P}	Θ_2^{P}	EXP
A	$\leq$ P	Θ_2^{P}	$\leq \mathrm{P}^{\mathrm{NExp}_{\log}}$ †	$\leq \mathrm{P}^{\mathrm{NExp}_{\log}}$ †
G	Θ_2^{P}	Θ_2^{P}	EXP	2EXP
F, GF	Θ_2^{P}	Θ_2^{P}	Θ_2^{P}	EXP
WS, WA	Θ_2^{P}	Θ_2^{P}	2EXP	2EXP

Table 3: Complexity of MINEX-REL$(\leq, \mathrm{UCQ}, \mathcal{L})$, MINEX-IRREL$(\leq, \mathrm{UCQ}, \mathcal{L})$, and MINEX-NEC$(\leq, \mathrm{UCQ}, \mathcal{L})$. †Shorthand for $\mathrm{P}^{\mathrm{NExp}[O(\log n)]}$.

We now proceed with MINEX-REL, MINEX-IRREL, and MINEX-NEC, which turn out to have the same complexity for a given minimality criterion; see Tables 3 and 4. The solutions of these three problems share the necessity of finding a $\preccurlyeq$-MinEX: for MINEX-REL, a $\preccurlyeq$-MinEX containing the distinguished fact; for MINEX-IRREL, a $\preccurlyeq$-MinEX excluding all the forbidden fact-sets; and for MINEX-NEC, a $\preccurlyeq$-MinEX *excluding* the given fact f (i.e., a counterexample of f's necessity, to subsequently flip the answer). Computing a $\preccurlyeq$-MinEX requires the evaluation of the size and the weight of the $\leq$-minimal and $\leq_w$-minimal MinEXs, respectively: for this reason, oracle calls are needed to perform a binary search. The complexity of $\leq_w$ is higher in many cases.

Theorem 8. *If* $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ *is in the complexity class* $\mathcal{C}$ *in the combined (resp., ba-combined, fp-combined, and data) complexity, then* MINEX-REL$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$, MINEX-IRREL$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$, *and* MINEX-NEC$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$ *are in* $\mathrm{P}^{\mathrm{NP}^{\mathcal{C}}[O(\log n)]}$ *and in* $\mathrm{P}^{\mathrm{NP}^{\mathcal{C}}}$ *for* $\leq$ *and* $\leq_w$*, respectively, in the combined (resp., ba-combined, fp-combined, and data) complexity.*

We now focus on the cases where $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ is in NP, and establish tighter upper bounds than those provided by the theorems introduced so far. Intuitively, the tighter upper bounds can be obtained, because the guess of the $\preccurlyeq$-MinEX and the guess of the entailment witness (recall that $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ is in NP) can be combined and performed by a single machine. Therefore, in Theorem 9, which is the correspondent of Theorem 7, we obtain a membership in $\mathrm{D^P}$; and in Theorem 10, which is the correspondent of Theorem 8, we obtain memberships in Θ_2^{P} and Δ_2^{P}.

Theorem 9. *If* $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ *is in* NP *in the*

$\mathcal{L}$	**Data**	*fp*-**comb.**	*ba*-**comb.**	**Comb.**
L, LF, AF	$\leq$ P	Δ_2^{P}	Δ_2^{P}	PSPACE
S, SF	$\leq$ P	Δ_2^{P}	Δ_2^{P}	EXP
A	$\leq$ P	Δ_2^{P}	$\leq \mathrm{P}^{\mathrm{NExp}}$	$\leq \mathrm{P}^{\mathrm{NExp}}$
G	Δ_2^{P}	Δ_2^{P}	EXP	2EXP
F, GF	Δ_2^{P}	Δ_2^{P}	Δ_2^{P}	EXP
WS, WA	Δ_2^{P}	Δ_2^{P}	2EXP	2EXP

Table 4: Complexity of MINEX-REL$(\leq_w, \mathrm{UCQ}, \mathcal{L})$, MINEX-IRREL$(\leq_w, \mathrm{UCQ}, \mathcal{L})$, and MINEX-NEC$(\leq_w, \mathrm{UCQ}, \mathcal{L})$.

ba-combined (resp., fp-combined) complexity, then IS-MINEX$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$ *and* ALL-MINEX$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$ *are in* $\mathrm{D^P}$ *in the ba-combined (resp., fp-combined) complexity, where* $\preccurlyeq\ \in \{\leq, \leq_w\}$.

Theorem 10. *If* $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ *is in* NP *in the ba-combined (resp., fp-combined) complexity, then* MINEX-REL$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$, MINEX-IRREL$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$, *and* MINEX-NEC$(\preccurlyeq, \mathrm{UCQ}, \mathcal{L})$ *are in* Θ_2^{P} *and in* Δ_2^{P} *for* $\leq$ *and* $\leq_w$*, respectively, in the ba-combined (resp., fp-combined) complexity.*

In the data complexity, the tractability of all problems for the FO-rewritable languages follows from the fact that all explanations can be computed in polynomial time.

Theorem 11. *For the FO-rewritable language over existential rules, all problems considered for both cardinality- and weight-minimal explanations are in* P *in the data complexity.*

Proof sketch.. Let D be a database, and (Q, Σ) be an OMQ. For an FO-rewritable language, there exists a query Q_Σ, which is a union of conjunctive queries, such that for every database D', $D' \models (Q, \Sigma)$ iff $D' \models Q_\Sigma$. Note that Q_Σ depends only on Q and Σ, which are fixed in the data complexity, so Q_Σ can be constructed in constant time. Let $Q_\Sigma = Q_\Sigma^1 \vee \cdots \vee Q_\Sigma^m$, where $Q_\Sigma^i = \exists \mathbf{Y}^i \Phi^i(\mathbf{Y}^i)$.

Let $\mathcal{D}_{Q_\Sigma}^i = \{\Phi^i(\mathbf{t}) \mid \mathbf{t} \in \mathit{adom}^{|\mathbf{Y}^i|}\}$, where adom is the set of all constants appearing in D, and $\Phi^i(\mathbf{t})$ is viewed as a set of facts. Thus, $\mathcal{D}_{Q_\Sigma}^i$ is a set of sets of facts and can be computed in polynomial time, as adom has polynomial size, and $|\mathbf{Y}^i|$ is a constant. Furthermore, $\mathcal{D}_{Q_\Sigma} = \cup_{i=1}^m \mathcal{D}_{Q_\Sigma}^i$ can be computed in polynomial time, as m is a constant. It is not difficult to see that all elements in $\mathcal{D}_{Q_\Sigma}$ that are subsets of D are all the explanations for $D \models (Q, \Sigma)$. This set can be computed in polynomial time in the size of $\mathcal{D}_{Q_\Sigma}$, and so in the size of D. Then, preferred explanations can be computed in polynomial time as well, and, as a consequence, all problems that we consider can be decided in polynomial time. ☐

Hardness Results

We start with the problem of recognizing cardinality-minimal explanations and prove that it is intractable already for GF existential rules in the data complexity, even if query answering is known to be in polynomial time for this fragment. This is in strong contrast with the complexity of recognizing inclusion-minimal explanations, which remains in polynomial time.

Theorem 12. $\text{IS-MINEX}(\leq, \text{UCQ}, \text{GF})$ *is* coNP*-hard in the data complexity.*

$\text{IS-MINEX}(\leq, \text{UCQ}, \mathcal{L}_\emptyset)$ is D^P-hard in the *fp*-combined complexity. This is implicit in the D^P-hardness proof for the *fp*-combined complexity of $\text{IS-MINEX}(\subseteq, \text{UCQ}, \mathcal{L}_\emptyset)$ (Ceylan et al. 2019, Thm 5), since the constructed set in that proof is inclusion- and also cardinality-minimal.

$\text{IS-MINEX}(\subseteq, \text{UCQ}, \text{A})$ was shown D^Exp-hard in the *ba*-combined complexity (Ceylan et al. 2019). The reduction used in that work immediately applies to our case, as the database and the OMQ constructed in that proof admit a unique $\subseteq$-MinEX, which is hence also a $\leq$-MinEX.

The other hardness results for $\text{IS-MINEX}(\leq, \text{UCQ}, \mathcal{L})$ in Table 2 are proven by showing that it is as hard as OMQA in the *ba*-combined and combined complexity.

Theorem 13. $\text{IS-MINEX}(\leq, \text{UCQ}, \mathcal{L})$ *is at least as hard as* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *in the ba-combinded and combined complexity.*

Proof sketch. We reduce $\text{OMQA}(\text{UCQ}, \mathcal{L})$ to $\text{IS-MINEX}(\leq, \text{UCQ}, \mathcal{L})$. Let D be a database and (Q, Σ) be an OMQ, which is an instance of $\text{OMQA}(\text{UCQ}, \mathcal{L})$. Let D_Q be a set of facts obtained by grounding variables in Q with fresh constant symbols. Then certainly $D_Q \models (Q, \Sigma)$. Let $D' = \{p(), r()\}$, $\Sigma' = \Sigma \cup \{p() \to f \mid f \in D\} \cup \{r() \to g \mid g \in D_Q\}$, $Q' = Q \wedge p()$, and $E = \{p()\}$, where p and r are fresh predicates. Note that $D' \models (Q', \Sigma')$ and thus it forms an instance of IS-MINEX. Furthermore, it is not difficult to verify that $D \models (Q, \Sigma)$ iff E is a $\leq$-MinEX for $D' \models (Q', \Sigma')$. □

As noted earlier, the cardinality order is a special case of the weight one. Hence, all the hardness results for the former immediately extends to the latter. This implies all the hardness results for $\text{IS-MINEX}(\leq_w, \text{UCQ}, \mathcal{L})$ in Table 2.

We now focus on ALL-MINEX under the cardinality criterion. We start with a coNP-hardness in the data complexity.

Theorem 14. $\text{ALL-MINEX}(\leq, \text{UCQ}, \text{GF})$ *is* coNP*-hard in the data complexity.*

The D^P-hardness of $\text{ALL-MINEX}(\leq, \text{UCQ}, \mathcal{L}_\emptyset)$ and the D^Exp-hardness of $\text{ALL-MINEX}(\leq, \text{UCQ}, \text{A})$ are implicit in the proofs of the D^P-hardness and D^Exp-hardness of $\text{ALL-MINEX}(\subseteq, \text{UCQ}, \mathcal{L}_\emptyset)$ and $\text{ALL-MINEX}(\subseteq, \text{UCQ}, \text{A})$, respectively (Ceylan et al. 2019, Thm 11). This holds because the database and the OMQ constructed in those proofs admit a unique $\subseteq$-MinEX, which is hence also a $\leq$-MinEX.

The other hardness results for $\text{ALL-MINEX}(\leq, \text{UCQ}, \mathcal{L})$ in Table 2 follow from the construction used to prove Theorem 13, where the set of all MinEXs is $\{\{p()\}\}$.

All the hardness results for the cardinality order immediately extend to the weight order, thereby establishing all the hardness results for $\text{ALL-MINEX}(\leq_w, \text{UCQ}, \mathcal{L})$ in Table 2.

We move to MINEX-REL. Under cardinality-minimality, MINEX-REL is hard already in the data complexity for GF existential rules, and it is at the second level of the polynomial hierarchy. The reduction is from the problem COMP-SAT: for two sets of 3CNF formulas, decide whether one set contains a greater number of satisfiable formulas compared to the other (Lukasiewicz and Malizia 2016, 2017).

Theorem 15. $\text{MINEX-REL}(\leq, \text{UCQ}, \text{GF})$ *is* Θ_2^P*-hard in the data complexity.*

MINEX-REL under cardinality-minimality is Θ_2^P-hard in the *fp*-combined complexity even with an empty program.

Theorem 16. $\text{MINEX-REL}(\leq, \text{UCQ}, \mathcal{L}_\emptyset)$ *is* Θ_2^P*-hard in the fp-combined complexity.*

Proof sketch.. The reduction is from the Θ_2^P-complete problem INMINCOVER: for a graph $G = (V, E)$ and a node w, is w in a cardinality-minimal vertex cover of G? We first show the Θ_2^P-completeness of INMINCOVER. Lopatenko and Bertossi (2006) showed that the following problem is Θ_2^P-complete: given a graph $G = (V, E)$ and a vertex w, decide whether w belongs to all the cardinality-maximal independent sets of G. It is known that a subset I of V is a cardinality-maximal independent set of G iff $V \setminus I$ is a cardinality-minimal vertex cover of G. Then, deciding whether w belongs to all the cardinality-maximal independent sets of G is equivalent to deciding whether w does not belong to any cardinality-minimal vertex cover of G. Since INMINCOVER is the complement of the latter problem and Θ_2^P is a deterministic class, INMINCOVER is Θ_2^P-complete.

The database built in the reduction is $D = \{edge(0,1), edge(1,0), edge(1,1)\} \cup \{vertex(v,0), vertex(v,1) \mid v \in V\}$, where the fact $vertex(v,1)$ is intuitively associated with the vertex v being in a vertex cover.

The program is empty, and the query comprises three parts:

The first imposes the presence of all $edge$ facts in any explanation: $edge\_test = edge(0,1) \wedge edge(1,0) \wedge edge(1,1)$.

The second part requires all facts $vertex(v,0)$ to be present in any explanation: $all\_vertices = \bigwedge_{v \in V} vertex(v,0)$.

The third part captures how an edge is covered: $cover = \bigwedge_{(u,v) \in E} (edge(X_u, X_v) \wedge vertex(u, X_u) \wedge vertex(v, X_v))$.

The query is $Q = edge\_test \wedge all\_vertices \wedge cover$. To conclude, the distinguished fact is $vertex(w,1)$.

It can be shown that w is in a minimum-size vertex cover of G iff $vertex(w,1)$ is in a $\leq$-MinEX for $D \models (Q, \Sigma)$. □

We obtain the other hardness results for $\text{MINEX-REL}(\leq, \text{UCQ}, \mathcal{L})$ by inspection of Theorem 13's proof, where $\{p()\}$ is the only $\leq$-MinEX, and we check whether $p()$ is relevant.

We now analyze $\text{MINEX-REL}(\leq_w, \text{UCQ}, \mathcal{L})$'s hardness. First, we can prove the Δ_2^P-hardness in the data complexity. The reduction is from the lexicographically minimum satisfying assignment problem (MSA) (Krentel 1988).

Theorem 17. $\text{MINEX-REL}(\leq_w, \text{UCQ}, \text{GF})$ *is* Δ_2^P*-hard in the data complexity.*

Also, $\text{MINEX-REL}(\leq_w, \text{UCQ}, \mathcal{L}_\emptyset)$ can be shown to be Δ_2^P-hard in the *fp*-combined complexity. The reduction is from the MSA problem (Krentel 1988).

Theorem 18. $\text{MINEX-REL}(\leq_w, \text{UCQ}, \mathcal{L}_\emptyset)$ *is* Δ_2^P*-hard in the fp-combined complexity.*

The other hardness results of MINEX-REL for $\leq_w$ follow from the hardness of MINEX-REL for $\leq$.

We conclude with the hardness of MINEX-IRREL and MINEX-NEC. Observe that MINEX-NEC can be reduced to the complement of MINEX-IRREL: indeed, f is in every $\preccurlyeq$-MinEX iff there is no $\preccurlyeq$-MinEX not containing $\{f\}$. Hence, it suffices to show the hardness results only for MINEX-NEC.

MINEX-NEC$(\leq, \mathrm{UCQ}, \mathsf{GF})$'s Θ_2^{P}-hardness in the data complexity is shown by Theorem 15's proof, in which either *every* $\leq$-MinEX contains a distinguished fact f or none.

As for the *fp*-combined complexity, we can show that MINEX-NEC$(\leq, \mathrm{UCQ}, \mathcal{L}_\emptyset)$ is Θ_2^{P}-hard. The reduction is from a variant of the problem of deciding whether a given vertex is *outside* all cardinality-maximum independent sets of a graph (Lopatenko and Bertossi 2016, Lemma 3.(2) and Lemma 6); the construction is as in the proof of Theorem 16.

Theorem 19. MINEX-NEC$(\leq, \mathrm{UCQ}, \mathcal{L}_\emptyset)$ *is* Θ_2^{P}*-hard in the fp-combined complexity.*

We derive the other hardness results for MINEX-NEC$(\leq, \mathrm{UCQ}, \mathcal{L})$ in Table 3 via Theorem 13's proof, where $\{p()\}$ is the only $\leq$-MinEX, and we check whether $p()$ is necessary.

For the hardness of MINEX-NEC$(\leq_w, \mathrm{UCQ}, \mathcal{L})$, in the reductions proving Theorems 17 and 18, the $\leq_w$-MinEX is unique. So, these reductions also apply to MINEX-NEC$(\leq_w, \mathrm{UCQ}, \mathcal{L})$, showing its Δ_2^{P}-hardness in Table 4.

The other hardness results of MINEX-NEC for $\leq_w$ follow from the hardness of MINEX-NEC for $\leq$. Moreover, notice that all hardness results for MINEX-NEC are for deterministic classes, hence MINEX-IRREL's hardness results follow.

Related Work

The literature on explanations under existential rules was sparse until recently. We have introduced and studied a wide range of problems, based on earlier work by Ceylan et al. (2019), where an explanation is an inclusion-minimal subset of the database, which, together with the program, entails the query. Ceylan et al. (2019) focus on inclusion-minimal explanations and do not consider other minimality criteria. This work investigates these problems under cardinality- and weight-minimal explanations. Explanations under existential rules have been studied for probabilistic databases (Ceylan, Borgwardt, and Lukasiewicz 2017), which is relative to a different (probabilistic) data model. Explanations under existential rules have been also investigated under inconsistency-tolerant semantics (Lukasiewicz, Malizia, and Molinaro 2020) and in the case of negative query answers (i.e., non-entailments) (Ceylan et al. 2020a). All these studies are technically different from the current study, as the underlying frameworks and/or the tasks are different.

The DL literature on explanations is very rich. Explanations are first considered in DLs by McGuinness and Borgida (1995), which is then followed by Borgida, Franconi, and Horrocks (2000). The main goal of these early works is to explain the classical problem of concept subsumption, and explanations are given in the form of proofs, based on the underlying deductive calculi. Subsequent work in DLs mostly focused on explanations, which are minimal subsets of axioms in a logical theory instead of the specific proofs (Schlobach and Cornet 2003). This approach is known as *axiom pinpointing* (Kalyanpur et al. 2007; Baader and Suntisrivaraporn 2008; Peñaloza and Sertkaya 2017), and such explanations are called *justifications* in the DL literature (Horridge, Parsia, and Sattler 2008). This line of research has resulted in a number of systems (Kalyanpur et al. 2007; Sebastiani and Vescovi 2009). These works focus on explaining classical reasoning tasks and not on query answering.

The problem of explaining OMQA has been investigated for the *DL-Lite* family of languages (Borgida, Calvanese, and Rodriguez-Muro 2008). Recently, explanations for OMQA relative to a large class of DLs have been investigated (Ceylan et al. 2020b), which also builds on the study given for existential rules (Ceylan et al. 2019). These works are closely related, but they focus only on inclusion-minimal explanations. Note that explanations for OMQA are also considered under different minimality criteria in DLs by Bienvenu, Bourgaux, and Goasdoué (2019), where the goal is to derive explanations under inconsistency-tolerant semantics. Differently, we provide explanations under the standard semantics. Explanations under inconsistency-tolerant semantics have also been investigated for existential rule languages in (Arioua, Tamani, and Croitoru 2015; Arioua, Buche, and Croitoru 2017; Hecham et al. 2017; Lukasiewicz, Malizia, and Molinaro 2020).

More generally, deriving explanations for query answers can be seen as a form of logical abduction (Reiter 1987; Eiter and Gottlob 1995; Eiter, Gottlob, and Leone 1997), which is also widely investigated in DLs—see, e.g., (Klarman, Endriss, and Schlobach 2011; Calvanese et al. 2013).

In that setting, the goal is to explain non-entailments (or negative entailments), which makes these works very different, as then we are interested in adding a minimal set of facts to the database (instead of considering subsets of the database) to satisfy the entailment. Explanations have been investigated also for logic programs under different semantics, e.g., see (Damásio, Analyti, and Antoniou 2013; Pontelli, Son, and El-Khatib 2009; Cabalar, Fandiño, and Fink 2014).

Conclusions

We studied explanations for OMQA under existential rules under two different minimality criteria, namely, *cardinality-minimal* and *weight-minimal* explanations, and provided a thorough complexity analysis for several decision problems. Our study provides a more complete picture for explanations under existential rules, and also implies upper bounds for many DLs, which can be embedded into existential rules.

An intriguing direction for future work is to investigate the problem of explaining query entailment for non-monotone queries. This poses different challenges, such as the definition of an explanation itself, which has to take into account both what in the database made the entailment hold and what not in the database made the entailment hold.

Acknowledgments

This work was supported by the Alan Turing Institute under the UK EPSRC grant EP/N510129/1, by the AXA Research Fund, by the EPSRC grants EP/R013667/1, EP/L012138/1, and EP/M025268/1, and by the EU TAILOR grant.

References

Arioua, A.; Buche, P.; and Croitoru, M. 2017. Explanatory dialogues with argumentative faculties over inconsistent knowledge bases. *Expert Syst. Appl.* 80: 244–262.

Arioua, A.; Tamani, N.; and Croitoru, M. 2015. Query answering explanation in inconsistent Datalog+/- knowledge bases. In *Proc. DEXA*, 203–219.

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2007. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2nd edition.

Baader, F.; and Suntisrivaraporn, B. 2008. Debugging SNOMED CT using axiom pinpointing in the description logic $\mathcal{EL}^+$. In *Proc. KR-MED*.

Bellomarini, L.; Sallinger, E.; and Gottlob, G. 2018. The Vadalog System: Datalog-based reasoning for knowledge graphs. *Proc. VLDB Endow.* 11(9): 975–987.

Bienvenu, M.; Bourgaux, C.; and Goasdoué, F. 2014. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *Proc. AAAI*, 996–1002.

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.; Cate, B. T.; Lutz, C.; and Wolter, F. 2014. Ontology-based data access: A study through disjunctive Datalog, CSP, and MMSNP. *ACM Trans. Database Syst.* 39(4): 33:1–33:44.

Borgida, A.; Calvanese, D.; and Rodriguez-Muro, M. 2008. Explanation in the *DL-Lite* family of description logics. In *Proc. OTM*, 1440–1457.

Borgida, A.; Franconi, E.; and Horrocks, I. 2000. Explaining $\mathcal{ALC}$ subsumption. In *Proc. ECAI*, 209–213.

Cabalar, P.; Fandiño, J.; and Fink, M. 2014. A complexity assessment for queries involving sufficient and necessary causes. In *Proc. JELIA*, 297–310.

Calì, A.; Gottlob, G.; and Kifer, M. 2013. Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res.* 48: 115–174.

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.

Calì, A.; Gottlob, G.; and Pieris, A. 2012. Towards more expressive ontology languages: The query answering problem. *Artif. Intell.* 193: 87–128.

Calvanese, D.; Cogrel, B.; Komla-Ebri, S.; Kontchakov, R.; Lanti, D.; Rezk, M.; Rodriguez-Muro, M.; and Xiao, G. 2017. Ontop: Answering SPARQL queries over relational databases. *Semantic Web J.* 8(3): 471–487.

Calvanese, D.; Ortiz, M.; Šimkus, M.; and Stefanoni, G. 2013. Reasoning about explanations for negative query answers in *DL-Lite*. *J. Artif. Intell. Res.* 48(1): 635–669.

Ceylan, İ. İ.; Borgwardt, S.; and Lukasiewicz, T. 2017. Most probable explanations for probabilistic database queries. In *Proc. IJCAI*, 950–956.

Ceylan, İ. İ.; Lukasiewicz, T.; Malizia, E.; Molinaro, C.; and Vaicenavičius, A. 2020a. Explanations for negative query answers under existential rules. In *Proc. KR*, 223–232.

Ceylan, İ. İ.; Lukasiewicz, T.; Malizia, E.; and Vaicenavičius, A. 2020b. Explanations for ontology-mediated query answering in description logics. In *Proc. ECAI*.

Ceylan, İ. İ.; Lukasiewicz, T.; Malizia, E.; and Vaicenavičius, A. 2019. Explanations for query answers under existential rules. In *Proc. IJCAI*, 1639–1646.

Damásio, C. V.; Analyti, A.; and Antoniou, G. 2013. Justifications for logic programming. In *Proc. LPNMR*, 530–542.

Eiter, T.; and Gottlob, G. 1995. The complexity of logic-based abduction. *J. ACM* 42(1): 3–42.

Eiter, T.; Gottlob, G.; and Leone, N. 1997. Abduction from logic programs: Semantics and complexity. *Theor. Comput. Sci.* 189(1-2): 129–177.

Fagin, R.; Kolaitis, P. G.; Miller, R. J.; and Popa, L. 2005. Data exchange: Semantics and query answering. *Theor. Comput. Sci.* 336(1): 89–124.

Hecham, A.; Arioua, A.; Stapleton, G.; and Croitoru, M. 2017. An empirical evaluation of argumentation in explaining inconsistency-tolerant query answering. In *Proc. DL*.

Horridge, M.; Parsia, B.; and Sattler, U. 2008. Laconic and precise justifications in OWL. In *Proc. ISWC*, 323–338.

Kalyanpur, A.; Parsia, B.; Horridge, M.; and Sirin, E. 2007. Finding all justifications of OWL DL entailments. In *Proc. ISWC/ASWC*, 267–280.

Klarman, S.; Endriss, U.; and Schlobach, S. 2011. ABox abduction in the description logic $\mathcal{ALC}$. *J. Autom. Reasoning* 46(1): 43–80.

Krentel, M. W. 1988. The complexity of optimization problems. *J. Comput. Syst. Sci.* 36(3): 490 – 509.

Lopatenko, A.; and Bertossi, L. 2006. Complexity of consistent query answering in databases under cardinality-based and incremental repair semantics. In *Proc. ICDT*, 179–193.

Lopatenko, A.; and Bertossi, L. 2016. Complexity of consistent query answering in databases under cardinality-based and incremental repair semantics (extended version). Technical Report arXiv:1605.07159.

Lukasiewicz, T.; and Malizia, E. 2016. On the complexity of mCP-nets. In *Proc. IJCAI*, 558–564.

Lukasiewicz, T.; and Malizia, E. 2017. A novel characterization of the complexity class Θ_k^{P} based on counting and comparison. *Theor. Comput. Sci.* 694: 21–33.

Lukasiewicz, T.; Malizia, E.; and Molinaro, C. 2018. Complexity of approximate query answering under inconsistency in Datalog+/–. In *Proc. IJCAI*, 1921–1927.

Lukasiewicz, T.; Malizia, E.; and Molinaro, C. 2020. Explanations for inconsistency-tolerant query answering under existential rules. In *Proc. AAAI*, 2909–2916.

Lukasiewicz, T.; Malizia, E.; and Vaicenavičius, A. 2019. Complexity of inconsistency-tolerant query answering in

Datalog+/– under cardinality-based repairs. In *Proc. AAAI*, 2962–2969.

McGuinness, D. L.; and Borgida, A. 1995. Explaining subsumption in description logics. In *Proc. IJCAI*, 816–821.

Nenov, Y.; Piro, R.; Motik, B.; Horrocks, I.; Wu, Z.; and Banerjee, J. 2015. RDFox: A highly-scalable RDF store. In *Proc. ISWC*, 3–20.

Peñaloza, R.; and Sertkaya, B. 2017. Understanding the complexity of axiom pinpointing in lightweight description logics. *Artif. Intell.* 250: 80–104.

Poggi, A.; Lembo, D.; Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Rosati, R. 2008. Linking data to ontologies. In *Journal on Data Semantics X*, 133–173. Springer, Berlin.

Pontelli, E.; Son, T. C.; and El-Khatib, O. 2009. Justifications for logic programs under answer set semantics. *Theory Pract. Log. Program.* 9(1): 1–56.

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

Schlobach, S.; and Cornet, R. 2003. Non-standard reasoning services for the debugging of description logic terminologies. In *Proc. IJCAI*, 355–360.

Sebastiani, R.; and Vescovi, M. 2009. Axiom pinpointing in lightweight description logics via Horn-SAT encoding and conflict analysis. In *Proc. CADE*, 84–99.

Vardi, M. Y. 1982. The complexity of relational query languages. In *Proc. STOC*, 137–146.