Explanations for Ontology-Mediated Query Answers

Andrius Vaicenavičius

St Anne's College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2020

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisors Prof. Thomas Lukasiewicz and Dr. İsmail İlkan Ceylan for their continuous support, encouragement, and patience throughout my DPhil studies. At each step of the project, I have always felt that they looked out for my best interests. I am very grateful to Prof. Thomas Lukasiewicz for being incredibly supportive since the very beginning of my DPhil project when I was figuring out which research direction to take. I would like to thank Dr. İsmail İlkan Ceylan for helping me to develop as an independent researcher. I appreciate him spending countless hours working together and discussing research ideas. Besides my supervisors, I would like to thank Prof. Enrico Malizia who has always been available to answer my technical questions and discuss the proofs. I have found all the interactions with him very encouraging and cheerful.

I am also very grateful to the fellow members of the Intelligent Systems Lab for a friendly atmosphere and their enthusiasm.

I would like to acknowledge the UK Engineering and Physical Sciences Research Council for funding my DPhil work.

Finally, I would like to thank my friends at Oxford for countless memorable moments and my family for their care.

Contents

1 Introduction **1**

2 Preliminaries **10**

2.1 Complexity Theory 10

2.2 Existential Rules 13

2.3 Description Logics 16

2.4 Ontology-Mediated Query Answering 18

2.4.1 Complexity of Query Answering under Existential Rules . . . 20

2.4.2 Complexity of Query Answering in Description Logics 22

3 Explanations for Query Answers under Existential Rules **24**

3.1 Minimal Explanations 24

3.2 Explanation Problems 26

3.3 Recognising Minimal Explanations 29

3.4 All Minimal Explanations 37

3.5 Irrelevant Facts for Explanations 43

3.6 Relevant Facts for Explanations 47

3.7 Small Explanations 54

3.8 Large Explanations 57

3.9 Overview of Complexity Results 60

4 Explanations for Query Answers in Description Logics **63**

4.1 Minimal Explanations 63

4.2 Explanation Problems 65

4.3 Recognising Minimal Explanations 67

4.4 All Minimal Explanations 75

4.5 Irrelevant Facts for Explanations 80

4.6 Relevant Facts for Explanations 83

4.7 Overview of Complexity Results 88

5 Explanations for Negative Query Answers under Existential Rules 91
5.1 Explanations for Negative Query Answers 92
5.2 Explanation Problems for Negative Query Answers 94
5.3 Recognising and Existence of Minimal Explanations 96
5.4 Relevant Facts for Explanations 106
5.5 Necessary Facts for Explanations 116
5.6 Overview of the Complexity Results 128

6 Preferred Explanations under Existential Rules 130
6.1 Preferred Explanations 131
6.2 Explanation Problems 133
6.3 Explanations with Minimal Cardinality 135
6.3.1 Smallest Explanation 135
6.3.2 All Smallest Explanations 140
6.3.3 Relevant Facts for Preferred Explanations 142
6.3.4 Necessary and Irrelevant Facts for Preferred Explanations . . 150
6.4 Explanations with Minimal Weight 152
6.4.1 Recognising Minimal Explanations 152
6.4.2 Results for Other Explanation Problems 152
6.5 Overview of Complexity Results 158

7 Related Work 160
7.1 Historical Perspective on Logic-based Abduction 160
7.2 Provenance 161
7.3 Explaining Ontological Reasoning Tasks 162
7.4 Explaining Positive Query Answers 164
7.5 Explaining Negative Query Answers 164
7.6 Explaining Query Answers under Inconsistency-tolerant Semantics . . 165

8 Conclusions 166
8.1 Summary 166
8.2 Future Work 171

Bibliography 174

List of Figures

1.1 Excerpt from the London public transport map. 2

2.1 Containment of complexity classes considered in this work. 11
2.2 Relationships between the Datalog$^{\pm}$ languages considered. 15
2.3 Relationships between the DLs considered. 19

3.1 Protein network, where each complex c_i contains some of the proteins $p_1, \ldots, p_6$. 25

4.1 Illustration of possible faults in different parts of a system. 64
4.2 Graph of the construction of $clause_i(v_{\ell_1^i}, v_{\ell_2^i}, v_{\ell_3^i})$. 86

5.1 Mechanical system and its network of issues 92

6.1 Excerpt from the London public transport map. 132

Chapter 1

Introduction

Equipping information systems with reasoning facilities is a very fundamental and long-lasting endeavour in artificial intelligence. Modern information management systems realise this by encoding the explicit domain knowledge in the form of ontologies. Ontologies are formalised using mathematical *logic*, which enables them to be utilised by various systems in an unambiguous way. Such knowledge-based systems are widely adopted in the industry. For example, 75% of the Fortune 500 companies have some kind of smart data programme underway [34], and it is predicted that the application of databases enriched with reasoning facilities will grow at 100% annually over the next few years [122].

Description logics (DLs) [11] and *existential rules* [38, 39, 37] are the two most widely used families of knowledge representation languages to express ontologies. The languages in both families are typically fragments of first-order logic, and thus have clear and well-defined semantics, allowing for automated reasoning. These languages use the *open-world assumption* suitable for reasoning with incomplete data, which assumes that if we do not know whether a statement is true, it can be either true or false. This is in contrast with closed-world assumption, which assumes that any statement that is true is also known to be true.

DLs capture sets of individuals via *concepts* and binary relationships between individuals via *roles*, which are unary and binary predicates, respectively. For example, we may have concepts *Man* and *Woman*, which capture sets of men and women, respectively, and a role *sibling_of*, which captures sibling pairs. All DLs allow to construct iteratively complex concepts and roles. For example, we can construct a concept for people whose siblings are all female, that is, $\forall \mathit{sibling\_of}.\mathit{Woman}$, or we can iteratively construct a concept for all men whose all siblings are female, that is, $\mathit{Man} \sqcap \forall \mathit{sibling\_of}.\mathit{Woman}$. The allowed constructs for the new concepts and roles are a characteristic of each DL. An ontology, also known as a TBox, encodes domain

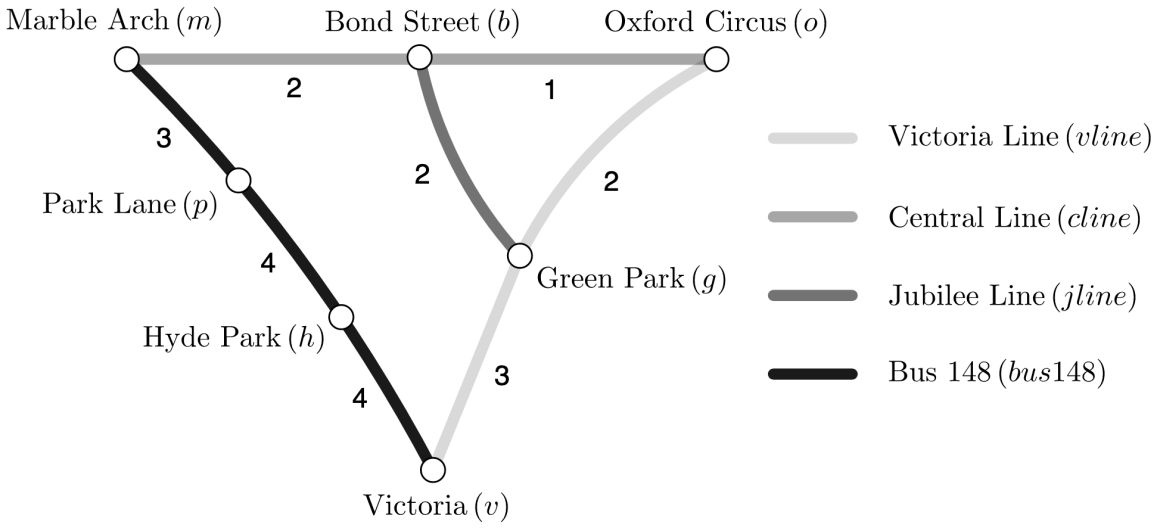

Figure 1.1: Excerpt from the London public transport map.

knowledge about concepts and roles in the form of of first-order axioms. For example, take *child_of* to be a role capturing child and parent pairs. We may capture that every person is a child of some woman, i.e., has a mother, with a TBox axiom $Person \sqsubseteq \exists child\_of\,.\,Woman$. Also, we may capture that every man and every woman is a person with an axiom $Man \sqcup Woman \sqsubseteq Person$. Thus the TBox here is the sets containing the two TBox axioms above. An ABox consists of assertions on individuals. For example, we can capture that Peter is a man with a concept assertion $Man(Peter)$, that Anna is a woman with a concept assertion $Woman(Anna)$, and that *Peter* is a child of *Anna* with a role assertion $child\_of(Peter, Anna)$. Thus our ABox is $\{Man(Peter), Woman(Anna), child\_of(Peter, Anna)\}$. Thus, the following assertions follow from the ABox and the TBox above: $Person(Peter)$, $Person(Anna)$, and $\exists child\_of\,.\,Woman(Anna)$.

Systems based on DLs have seen a wide adoption in the industry. In particular, the DL-based language OWL 2 [84], which is part of the W3C's Semantic Web technology stack, has been widely used in many applications. It has been applied in areas such as biology and medicine [92] with many prominent ontologies developed, including a medical ontology SNOMED CT[1] and a biological ontology GENE ONTOLOGY.[2] Many tools such as reasoners and ontology editors have been developed for languages based on DLs. One of the most popular ontology editors is Protégé,[3] which can be used with a variety of OWL reasoners.

[1]See http://www.snomed.org/
[2]See http://geneontology.org/
[3]See https://protege.stanford.edu/

Another prominent family of knowledge representation languages are existential rules. Informally, they can be seen as first-order implications between conjunctions of function-free atoms, which may contain existentially quantified variables in the head of such an implication. They are well-suited for data-intensive applications, as they allow higher-arity relations, which naturally occur in standard relational databases. This is in contrast to description logics, where usually no predicates with arity greater than two are allowed. We illustrate existential rules on the example from the public transport domain, shown in Figure 1.1.

Example 1.1. We can capture the stops in the network, shown in Figure 1.1, with unary facts, for example, we add a fact $\mathit{stop}(v)$ to capture that 'Victoria' is a stop in the network. Also, we can capture the links between the stops with binary facts, for example, we add a fact $\mathit{link}(\mathit{bus148}, v, h)$ to capture that 'Bus 148' travels between 'Victoria' and 'Hyde Park' stops. We can capture all of this network with the following set of the facts:

$$\begin{aligned} D_{st} = \{&\mathit{stop}(v), \mathit{stop}(h), \mathit{stop}(p), \mathit{stop}(m), \mathit{stop}(g), \mathit{stop}(b), \mathit{stop}(o), \\ &\mathit{link}(\mathit{bus148}, v, h), \mathit{link}(\mathit{bus148}, h, p), \mathit{link}(\mathit{bus148}, p, m), \mathit{link}(\mathit{vline}, v, g), \\ &\mathit{link}(\mathit{vline}, g, o), \mathit{link}(\mathit{jline}, g, b), \mathit{link}(\mathit{cline}, o, b), \mathit{link}(\mathit{cline}, b, m)\}. \end{aligned}$$

Also, we can capture that we are currently in the stop 'Victoria' with a fact $\mathit{start}(v)$, and thus we take the database to be $D = D_{st} \cup \{\mathit{start}(v), \mathit{destination}(m)\}$.

Then, we capture the knowledge about travelling along this network in an ontology Σ, also called a *program*. The program Σ consists of four rules. The first rule captures that the *start* stop is already reached, and the second one that if the destination can be reached then we can arrive to it:

$$\begin{aligned} &\mathit{start}(X) \to \mathit{reach}(X), \\ &\mathit{destination}(X) \wedge \mathit{reach}(X) \to \mathit{arrive}(X). \end{aligned}$$

The third rule captures that if a public transport line X has a link between the stops Y_1 and Y_2, and a link between the stops Y_2 and Y_3, then there is a link between Y_1 and Y_3:

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

The fourth rule in the program Σ captures how we can reach a stop or change a public transport line:

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

Notice that the program Σ generates new data from the database D. For example, we see that the facts $link(vline, v, o)$ or $reach(g)$ follow from the database D and the program Σ. ∎

Existential rules are an extension of *Datalog* rules, and so they are also known as *Datalog*$^{\pm}$ [13, 37]. Compared to Datalog, Datalog$^{\pm}$ rules allow existential quantifiers in rule heads. Also, existential rules are the logical translation of conceptual graph rules, used in graph-based knowledge representation [142]. Furthermore, existential rules have been studied in databases under the name of *tuple-generating dependencies (TGDs)* [1]. In that setting, they have been used to capture a wide range of schema constraints. Practical applications of existential rules range from bioinformatics [123] to modelling complex structures of chemical compounds [119, 87].

One of the most important reasoning tasks for ontology-based systems is *ontology-mediated query answering (OMQA)* [137, 22]. OMQA is a sophisticated paradigm that allows the user to query *heterogeneous* and potentially incomplete data sources using a familiar vocabulary and obtain more complete answers. The use of an ontology allows us to combine different data sources by matching entries that have the same meaning and infer additional information from data by capturing domain knowledge. In this framework, the user query and the ontology are viewed as two components of one composite query, called *ontology-mediated query (OMQ)* [22]. Ontology-mediated query answering has become one of the focal points of research in the field of knowledge representation and reasoning with numerous systems developed to support OMQA and related tasks [44, 124, 17].

Let us consider two potential applications of OMQA, one from an autonomous driving industry and another one from a medical domain. One common issue when accessing different data sources are differences in vocabularies. For example, there are are many labelled datasets for autonomous driving systems, including ApolloScape, Berkley DeepDrive, and Mapillary Vistas. These datasets often label the same objects differently, for example, a traffic sign object in ApolloScape is labelled as "traffic_sign", in Mapillary as "traffic-sign", and in Berkley DeepDrive as "Traffic sign". We can access these multiple data sources in a unified manner by connecting these vocabularies via an ontological layer. Another common issue is incompleteness of data. For instance, in the medical domain, a specialist might want to estimate the number of patients in a particular hospital that are treated for cancer. If the database contains only specific entries for diseases, querying such data might be impractical. For example, if a patient is diagnosed with chronic lymphocytic leukemia (CLL) and

the system does not contain the knowledge that CLL is a form of cancer, the search for all the patients who are treated for cancer would not return this person.

With the growth of intelligent information systems, there is a need to explain their behaviour. For example, the European Commission report [93] states that the manufacturers of autonomous vehicles "should ensure that individuals can obtain factual, intelligible explanations of the decision-making processes and justifications made by these systems, particularly in the event of individually or group-related adverse or unwanted consequences". There has been an increasing interest in explainable artificial intelligence, which is in contrast with black box models, where internal workings of such systems are not understood even by the developers [15]. Also, in knowledge representation, since devising an ontology is an error-prone task, there has been a significant amount of research on explaining unexpected consequences of an ontology, which we cover in this thesis in more detail in Chapter 7. Surprisingly, there has been little work on explaining OMQA under the classical semantics.

In this work, we consider explanations for ontology-mediated query answers under the classical semantics [135, 41]. In particular, given a database D, and an ontology-mediated query (Q, Σ), where Q is a query, and Σ is an ontology, if the database entails the OMQ, that is, $D \models (Q, \Sigma)$, we take a minimal explanation, or MinEX, for $D \models (Q, \Sigma)$ to be a minimal subset E of D that entails this ontology-mediated query. Note that the program is not a part of such an explanation, it is a part of the OMQ. We illustrate this notion on the setting, presented in Example 1.1.

Example 1.2. In the database D and the program Σ of Example 1.1, we have captured the network in Figure 1.1 and the fact that we are currently in the stop 'Victoria'. Suppose that we aim to travel to the stop 'Marble Arch', so we take the query to be $Q = \mathit{arrive}(m)$. It is easy to see that the routes to 'Marble Arch' correspond to subset-minimal explanations for $D \models (Q, \Sigma)$. One such route is by 'Bus 148':

$$E_1 = \{\mathit{start}(v), \mathit{stop}(m), \mathit{link}(\mathit{bus148}, v, h), \mathit{link}(\mathit{bus148}, h, p), \mathit{link}(\mathit{bus148}, p, m)\},$$

another one follows 'Victoria Line' to 'Oxford Circus' and then 'Central Line' to 'Marble Arch':

$$\begin{aligned} E_2 = \{&\mathit{start}(v), \mathit{stop}(o), \mathit{stop}(m), \mathit{link}(\mathit{vline}, v, g), \\ &\mathit{link}(\mathit{vline}, g, o), \mathit{link}(\mathit{cline}, o, b), \mathit{link}(\mathit{cline}, b, m)\}, \end{aligned}$$

and the last one travels along one link on 'Victoria Line', then 'Jubilee Line' and then 'Central Line':

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

Also, we capture travelling times along different routes by assigning weights to the facts in the database. In particular, we may assign the weights, as given in Figure 1.1, to *link* facts to capture travelling time between adjacent stops, and assign the weight 3 to all *stop* facts to capture that changing the line takes time. We can see that the explanations E_1 and E_2 are both weight-minimal with the weight 14, while E_3 has the weight 16. Therefore, we deduce that E_1 and E_2 correspond to the shortest routes in the network. ∎

Contributions

In this thesis, I provide a comprehensive complexity analysis of a wide range of computational problems associated with explaining ontology-mediated query answers. I study explanations both for positive and negative ontology-mediated query answers, both for existential rules and description logics, and under different minimality criterion. This allows me to indicate similarities and point out differences of complexity of explaining ontology-mediated query answers for these different settings.

I discuss two variants of MinEX, one for positive query answers and another for negative query answers by incorporating ideas from propositional abduction [71] and axiom pinpointing [135]. For positive query answers, I define a MinEX to be a minimal subset of the data that entails such a query. For negative query answers, I define a MinEX to be a minimal subset of hypotheses (a set of facts outside of the database), which, when added to the database, entails the query and is consistent with the ontology.

I consider a wide range of computational problems associated with explaining ontology-mediated query answers. In particular, I consider the problems of recognising a MinEX, Is-MinEX, recognising the set of all MinEXs, All-MinEX, deciding if a fact is contained in some (resp., every) explanation, MinEX-Rel (resp., MinEX-Nec), deciding if a set contains all relevant (resp., all necessary) facts for explaining a query answer, MinEX-AllRel (resp., MinEX-AllNec), deciding if there is an explanation that avoids forbidden sets of facts, MinEX-Irrel, and the problem of deciding if there is a MinEX smaller (resp., larger) than some given threshold

number, Small-MinEX (resp., Large-MinEX). Further, I consider the problem MinEX-Exists$^{\not\models}$ of deciding whether there is any MinEX for a negative query answer. Note that this problem is only relevant for negative query answers, as for positive query answers, the existence of a MinEX is guaranteed. I exhibit the universality of these notions by considering different ontology languages (both description logics and existential rules), different query languages (instance queries and unions of conjunctive queries) and different minimality criterion (subset, cardinality, and weight minimality).

I provide algorithms for solving these problems, which establish tight upper bounds in our complexity analysis. Many of these algorithms use only OMQA as a subtask and, since there are existing OMQA engines, these algorithms can be readily implemented. These general results allow us to compare the resources needed to solve different problems. For example, I observe that, in many cases recognising a single minimal explanation, Is-MinEX is as hard as recognising the set of all minimal explanations, All-MinEX. I show that for some problems near-identical algorithms apply, e.g., MinEX-Rel and Large-MinEX. Also, I show that problems associated with computing preferred explanations often have more involved algorithms. Furthermore, when OMQA is complete for some expressive deterministic complexity class, we observe that the complexity of solving these explanations problems is dominated by the task of query answering, and so the complexity does not increase when compared with OMQA. I also provide optimised algorithms when ontology-mediated query answering is FO-rewritable. For those cases, when the query and the ontology are fixed, we show how the set of all subset-minimal explanations can be constructed in polynomial time.

In this thesis, I derive the hardness of these problems by encoding well-known computational problems using the methods from complexity theory. These results pinpoint where the hardness of the discussed problems stem from. Since the choice of an ontology language has a major impact on the complexity of explanation problems, these reductions illustrate the expressive capabilities of these ontology languages. In these reductions, we commonly encode well-known variants of satisfiability, graph, and tiling problems. This further contributes to understanding the corresponding ontology languages and what they are capable of expressing.

Finally, this work gives a unifying view on the complexity of explaining ontology-mediated query answers, which can be useful when devising systems with explanation facilities. In this work, I summarise our results in tables, so that informed decisions can be made.

List of Published and Submitted Papers

I have produced the following papers during my DPhil project. The list is in chronological order. This thesis is based on the four latest papers as they all are concerned with explaining ontology-mediated query answers.

1. "Complexity of Inconsistency-Tolerant Query Answering in Datalog+/- under Cardinality-Based Repairs" with *Thomas Lukasiewicz, Enrico Malizia*, published in AAAI 2019;

2. "Explanations for Query Answers under Existential Rules" with *İsmail İlkan Ceylan, Thomas Lukasiewicz, Enrico Malizia*, published in IJCAI 2019;

3. "Explanations for Ontology-Mediated Query Answering in Description Logics" with *İsmail İlkan Ceylan, Thomas Lukasiewicz, Enrico Malizia*, published in ECAI 2020;

4. "Explanations for Negative Query Answers under Existential Rules" with *İsmail İlkan Ceylan, Thomas Lukasiewicz, Enrico Malizia, Cristian Molinaro*, published in KR 2020;

5. "Preferred Explanations for Ontology-Mediated Queries under Existential Rules" with *İsmail İlkan Ceylan, Thomas Lukasiewicz, Enrico Malizia, Cristian Molinaro*, published in AAAI 2021.

An extended abstract of the paper "Explanations for Query Answers under Existential Rules" was published in the workshop on *Explainable Logic-Based Knowledge Representation (XLoKR 2020) at KR 2020* [52], and an extended abstract of the paper "Explanations for Ontology-Mediated Query Answering in Description Logics" was published in the *33rd International Workshop on Description Logics (DL 2020)* [51].

Organisation of Thesis

The thesis is organised as follows:

Chapter 2: This chapter introduces the theoretical background of this work. I cover the relevant theory of computational complexity. I then introduce the two most widely used languages for ontology-mediated query answering, namely, existential

rules and description logics. I proceed to formally introduce ontology-mediated query answering and provide the known complexity results for the problem of ontology-mediated query answering for different fragments of existential rules and description logics.

Chapter 3: In this chapter, I introduce the notion of a minimal explanation and the problems associated with explaining positive ontology-mediated query answers, i.e., explaining entailment. I provide a detailed complexity analysis of these problems under different fragments of existential rules with respect to different complexity measures.

Chapter 4: I study the complexity of the problems, associated with computing minimal explanations, for different description logics, query languages (instance queries and unions of conjunctive queries), and different complexity measures.

Chapter 5: This chapter concerns explaining negative ontology-mediated query answers, i.e., query non-entailment, under existential rules. I introduce the notion of a minimal explanation for a negative query answer and the problems associated with explaining such non-entailments. This is followed by a complexity analysis of these problems for different fragments of existential rules.

Chapter 6: The last theoretical chapter is concerned with computing preferred explanations for ontology-mediated query answers under existential rules. First, I reformulate the explanation problems for general minimality criterion. Then, I proceed with a complexity analysis for two minimality criterion, namely, cardinality-minimality and weight-minimality, under different fragments of existential rules.

Chapter 7: In this chapter, I review related work. In particular, I discuss how my work adapts the ideas of some of the existing works, how it differs and how it generalises some of the existing approaches.

Chapter 8: I discuss the contributions, the limitations of this work, and possible future research directions.

Chapter 2

Preliminaries

In this chapter, we present the relevant background for our study. First, we briefly recall the relevant background on complexity theory. We then give the relevant background on knowledge representation languages. In particular, we introduce the two most widely studied languages used in the context of ontology-mediated query answering, namely, *existential rules* (also known as Datalog$^{\pm}$) and *description logics*. Finally, we introduce the problem of ontology-mediated query answering and its complexity with respect to different knowledge representation languages.

2.1 Complexity Theory

In this section, we recall the complexity classes used in this work. These complexity classes differentiate computational problems by the resources needed to solve them. In this work, we study *decision problems*, which are 'yes' or 'no' questions. We note that decision problems correspond to languages, which are taken to contain the encodings of the inputs of the decision problem for which the answer is 'yes'. In the rest of this work, we will talk about decision problems and languages interchangeably. The computational models that we use to define these complexity classes are deterministic and nondeterministic Turing machines (TMs), and Boolean circuits. For a thorough introduction to computational complexity, we refer the reader to the standard texts by Johnson [99] and Papadimitriou [127].

We introduce the classes, shown in Figure 2.1. The least expressive class that we consider is AC^0. To define this class, we introduce *Boolean circuits*, and refer the reader to the standard texts for a detailed introduction [153]. A *Boolean circuit* C_n with n inputs and a single output, intuitively, is a Boolean function with n inputs that can be constructed using logic gates (AND, OR and NOT) and wires. Given a string w over $\{0,1\}$ of length n, we write $C_n(w)$ for the output of the circuit C_n with w as

$$\mathrm{AC}^0 \subseteq \mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{P} \subsetneqq \begin{matrix}\mathrm{NP}\\ \mathrm{coNP}\end{matrix} \subseteq \mathrm{D}^{\mathrm{P}} \subseteq \Theta_2^{\mathrm{P}} \subseteq \Delta_2^{\mathrm{P}} \subsetneqq \begin{matrix}\Sigma_2^{\mathrm{P}}\\ \Pi_2^{\mathrm{P}}\end{matrix} \subseteq \mathrm{D}_2^{\mathrm{P}} \subseteq \mathrm{PSPACE} \subseteq$$

$$\mathrm{EXP} \subseteq \mathrm{NEXP} \subseteq \mathrm{D}^{\mathrm{EXP}} \subseteq \mathrm{P}^{\mathrm{NEXP}} \subseteq 2\mathrm{EXP}.$$

Figure 2.1: Containment of complexity classes considered in this work.

an input. The *depth* of a Boolean circuit is the maximum number of gates between an input and an output of a circuit, and the *size* of a circuit is the total number of gates in that circuit. The *fan-in* of a gate is the number of inputs that the gate takes. Note that a particular Boolean circuit can only capture an input of a fixed length, so to be able to define classes of problems, we consider collections of Boolean circuits. A *family of Boolean circuits* $\mathfrak{C}$ is an infinite list of circuits $\{C_n \mid n \in \mathbb{N}\}$, where C_n is a Boolean circuit with n input gates. We say that $\mathfrak{C}$ *decides* a language L over $\{0,1\}$ if, for every string w over $\{0,1\}$, $w \in L$ if and only if $C_{|w|}(w) = 1$. The class AC^0 is defined to be the set of decision problems that can be decided by a family of Boolean circuits with constant depth and polynomial number of unbounded fan-in AND and OR gates (without NO gates) [153].

A number of fundamental complexity classes are defined in terms of Turing machines whose *time* resources are bounded. One of the most fundamental complexity classes is P, the class of decision problems that can be solved by deterministic TMs in polynomial time in the size of the input. Another fundamental class that appears throughout this work is NP, the class of decision problems that can be solved by nondeterministic TMs in polynomial time in the size of the input. The class NP has a useful property – instances of the problems in this class with the answer 'yes' have concise proofs verifiable in polynomial time by a deterministic TM. The class EXP is the set of decision problems that can be solved by TMs in exponential time. An even more expressive class is 2EXP, which is the set of problems solvable by TMs in double exponential time, i.e., in $O(2^{2^{p(n)}})$, where n is the size of the input, and p is a polynomial. Another class is the set of decision problems, which can be solved by a NTM in exponential time in the size of the input. This class is denoted by NEXP.

We impose bounds on the space resources, rather than on time, in a TM to define complexity classes. To define such classes, we consider slightly modified TMs. They consist of two tapes: an input tape and a work tape, where an input tape is a read-only tape, which contains only an input, and the work tape is a read-write tape on

which the computation happens. In particular, given an input, we limit how many cells on a work tape, as a function on the size of the input, a TM can use in a computation. The classes L and NL are the sets of decision problems solvable by deterministic and nondeterministic TMs in logarithmic space in the size of the input, respectively. Another class is PSPACE, which is the set of decision problems solvable by deterministic TMs in polynomial space in the size of the input.

Other complexity classes can be derived from the introduced classes by taking their complements. A *complement* of a decision problem is obtained by reversing the 'yes' and 'no' answers. For a complexity class $\mathcal{C}$, the *complement of* $\mathcal{C}$, co-$\mathcal{C}$, denotes the class of problems whose complements are in $\mathcal{C}$. The complement classes that appear throughout this work are coNP and coNEXP. For any deterministic complexity class, such as AC^0, L, or P, we have that its complement is contained in the original class, and so they closed under taking complements. Furthermore, we have that NL is closed under complementation, that is, $\mathrm{coNL} = \mathrm{NL}$ [98, 150].

Another way of obtaining new complexity classes is through computation with *oracles*. Intuitively, an oracle is a black box, which can solve a problem at unit cost. Given a complexity class $\mathcal{C}$, we define $\mathrm{P}^{\mathcal{C}}$ to be a class of problems, which are solvable by polynomial-time TMs, which have access to a $\mathcal{C}$ oracle, and we define $\mathrm{NP}^{\mathcal{C}}$ to be a class of problems solvable by polynomial-time NTMs, which have access to a $\mathcal{C}$ oracle. Some of the most prominent oracle classes are Σ_k^{P}, Π_k^{P}, and Δ_k^{P} for $k \in \mathbb{N}$, forming the *polynomial hierarchy (PH)* [148]. Let $\Sigma_0^{\mathrm{P}} = \Pi_0^{\mathrm{P}} = \Delta_0^{\mathrm{P}} = \mathrm{P}$. Then the *polynomial hierarchy (PH)* is given by $\Sigma_k^{\mathrm{P}} = \mathrm{NP}^{\Sigma_{k-1}^{\mathrm{P}}}$, $\Delta_k^{\mathrm{P}} = \mathrm{P}^{\Sigma_{k-1}^{\mathrm{P}}}$, and $\Pi_k^{\mathrm{P}} = \text{co-}\Sigma_k^{\mathrm{P}}$ for all $k \geq 1$. Note that, in the first level of the polynomial hierarchy, we have that $\Sigma_1^{\mathrm{P}} = \mathrm{NP}$, $\Pi_1^{\mathrm{P}} = \mathrm{coNP}$, and $\Delta_1^{\mathrm{P}} = \mathrm{P}$. In the second level, $\Sigma_2^{\mathrm{P}} = \mathrm{NP}^{\mathrm{NP}}$, $\Delta_2^{\mathrm{P}} = \mathrm{P}^{\mathrm{NP}}$, and $\Pi_2^{\mathrm{P}} = \text{co-}(\mathrm{NP}^{\mathrm{NP}})$. We further observe that, for any $k \geq 1$, $\mathrm{P}^{\Sigma_k^{\mathrm{P}}} = \mathrm{P}^{\Pi_k^{\mathrm{P}}}$, and $\mathrm{NP}^{\Sigma_k^{\mathrm{P}}} = \mathrm{NP}^{\Pi_k^{\mathrm{P}}}$. In particular, $\mathrm{NP}^{\mathrm{NP}} = \mathrm{NP}^{\mathrm{coNP}}$. We can also impose a bound on the number of oracle calls. We highlight such a bound in the brackets after the oracle class. In this work, we consider, for $k \geq 1$, the classes $\Theta_k^{\mathrm{P}} = \mathrm{P}^{\Sigma_{k-1}^{\mathrm{P}}[O(\log n)]}$ of problems solvable by a polynomial-time TM, which is allowed to query a $\Sigma_{k-1}^{\mathrm{P}}$ oracle at most logarithmic number of times in the size of the input.

We may also obtain new complexity classes by taking conjunctions of the already defined ones. We define the class $\mathrm{D}^{\mathrm{P}} = \{L_1 \cap L_2 \mid L_1 \in \mathrm{NP}, L_2 \in \mathrm{coNP}\}$ that is the set of all the languages, which are an intersection of a language in NP and a language in coNP [129]. Note that any problem in D^{P} is solvable by a single NP test, followed by a single coNP test. It is common to write $\mathrm{D}^{\mathrm{P}} = \mathrm{NP} \wedge \mathrm{coNP}$. The class D^{P} can be generalised to the classes $\mathrm{D}_k^{\mathrm{P}} = \{L_1 \cap L_2 \mid L_1 \in \Sigma_k^{\mathrm{P}}, L_2 \in \Pi_k^{\mathrm{P}}\}$ for $k \geq 1$ [66, 18].

Similarly, any problem in $\mathrm{D}^{\mathrm{P}}_k$ is solvable by a Σ^{P}_k test, followed by a Π^{P}_k test, and thus it is common to write $\mathrm{D}^{\mathrm{P}}_k = \Sigma^{\mathrm{P}}_k \wedge \Pi^{\mathrm{P}}_k$. In particular, $\mathrm{D}^{\mathrm{P}}_1 = \mathrm{D}^{\mathrm{P}}$. Furthermore, we study the problems in the class $\mathrm{D}^{\mathrm{Exp}} = \{L_1 \cap L_2 \mid L_1 \in \mathrm{NExp}, L_2 \in \mathrm{coNExp}\}$. Analogously, any problem in $\mathrm{D}^{\mathrm{Exp}}$ is solvable by a single NExp test, followed by a single coNExp test, and we write $\mathrm{D}^{\mathrm{Exp}} = \mathrm{NExp} \wedge \mathrm{coNExp}$.

For a complexity class $\mathcal{C}$ above P, a problem L is *$\mathcal{C}$-hard* if, for every problem $L' \in \mathcal{C}$, there is a polynomial reduction from L' to L. For the complexity class P, a problem L is P-*hard* if, for every problem $L' \in \mathcal{C}$, there is a logspace reduction from L' to L. If a problem L belongs to a class $\mathcal{C}$ and is also $\mathcal{C}$-hard, then L is said to be *$\mathcal{C}$-complete*. For example, we have seen that Sat is in NP. It is also known to be NP-hard, and thus NP-complete [81].

The relationships between different complexity classes introduced in this section are presented in Figure 2.1.

2.2 Existential Rules

In this section, we consider *existential rules*, also known as *Datalog*$^{\pm}$ [37, 38, 39]. As common for data management, extensional data are expressed using the relational database model [57, 58] for which existential rules are well-suited, as they allow high-arity relations (unlike description logics, introduced later). They can be seen to have a number of origins. First, they are an extension of *Datalog* rules, hence the name *Datalog*$^{\pm}$ [37]. An important difference between Datalog and Datalog$^{\pm}$ is that, in Datalog, the rules form a query, while, in Datalog$^{\pm}$, they are a part of an ontology. Existential rules have been studied in databases under the name of *tuple-generating dependencies (TGDs)* [1]. Finally, the logical translation of conceptual graph rules, used in graph-based knowledge representation, are exactly existential [142].

We define the syntax and semantics of existential rules. We consider a relational vocabulary σ consisting of mutually disjoint, possibly infinite sets $\mathbf{R}$ of predicate, $\mathbf{C}$ of constant, $\mathbf{V}$ of variable names, and $\mathbf{N}$ of labelled nulls. Each predicate is associated with an arity, i.e., a non-negative integer. Note that we can see this as the relational vocabulary of first-order logic, extended with nulls. Now we define the fundamental syntactic blocks for existential rules.

Definition 2.1. A *term* is a constant, a null, or a variable. An *atom* is an expression of the form $p(t_1, \ldots, t_n)$, where p is an n-ary predicate from $\mathbf{R}$, and $t_1, \ldots, t_n$ are terms. A *ground atom* (or *fact*) has only constants as terms. ∎

Definition 2.2 (Syntax)**.** A *tuple-generating dependency (TGD)* σ is a first-order formula of the form $\forall \mathbf{X} \forall \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \Psi(\mathbf{X}, \mathbf{Z})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms, called the *body* of the TGD and denoted $body(\sigma)$, and $\Psi(\mathbf{X}, \mathbf{Z})$ is a conjunction of atoms, called the *head* of the TGD and denoted $head(\sigma)$, all without nulls. Classes of TGDs are also known as *existential rules*, or *Datalog*$^{\pm}$ *rules*.

A *negative constraint (NC)* ν is a first-order formula of the form $\forall \mathbf{X}\, \Phi(\mathbf{X}) \rightarrow \bot$, where $\Phi(\mathbf{X})$ is a conjunction of atoms without nulls, called the *body* of ν and denoted $body(\nu)$, and $\bot$ denotes the truth constant *false*. ∎

Definition 2.3 (Knowledge base)**.** A *database* D is a finite set of facts. A *program* (or *ontology*) is a finite set Σ of TGDs and NCs. We denote by Σ_T and Σ_{NC} the subsets of Σ containing the TGDs and NCs, respectively.

A *knowledge base* is a pair (D, Σ), where D is a database and Σ is a program. ∎

To define the semantics of TGDs and NCs, we introduce the following notions. An *instance* I is a (possibly infinite) set of atoms containing constants and nulls only. A *database* D is a finite instance in which atoms contain only constants. A *homomorphism* is a mapping $h\colon \mathbf{C} \cup \mathbf{N} \cup \mathbf{V} \rightarrow \mathbf{C} \cup \mathbf{N} \cup \mathbf{V}$ 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. Their semantics is as follows.

Definition 2.4 (Semantics)**.** An instance I satisfies a TGD 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 a homomorphism $h' \supseteq h|_{\mathbf{X}}$, where $h|_{\mathbf{X}}$ is the restriction of h on $\mathbf{X}$, such that $h'(\Psi(\mathbf{X}, \mathbf{Z})) \subseteq I$. An instance I satisfies an NC ν, written $I \models \nu$, if there is no homomorphism h such that $h(\Phi(\mathbf{X})) \subseteq I$. Given a set Σ of TGDs and NCs, an instance I satisfies Σ, written $I \models \Sigma$, if I satisfies each TGD and NC of Σ.

A *model* of a knowledge base (D, Σ) is an instance I such that $D \subseteq I$ and $I \models \Sigma$. We say that a knowledge base is *consistent* if it has a model. Otherwise, it is *inconsistent*.

∎

For brevity, we omit the universal quantifiers in front of TGDs and NCs, and use the comma (instead of $\wedge$) for conjoining atoms. Given a class of TGDs $\mathcal{L}$, we denote by $\mathcal{L}_\bot$ the formalism obtained by combining $\mathcal{L}$ with arbitrary NCs. For a program Σ, we denote by $\Sigma \in \mathcal{L}$ (resp., $\Sigma \in \mathcal{L}_\bot$) that all TGDs of Σ belong to $\mathcal{L}$ (resp., $\mathcal{L}_\bot$).

In its general form, reasoning with TGDs is undecidable [16], that is, it is undecidable whether any given set of TGDs has a model and thus many tasks involving

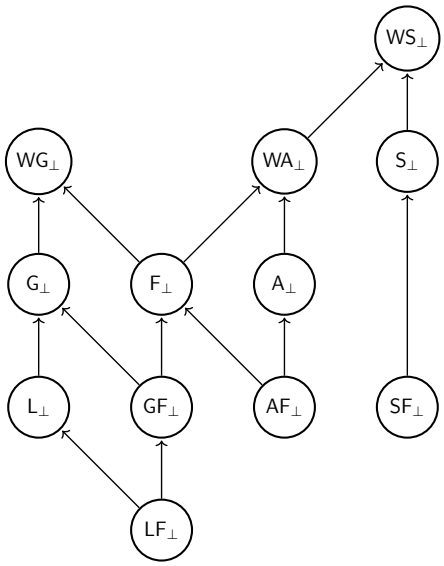

Figure 2.2: Relationships between the Datalog$^{\pm}$ languages considered.

TGDs become undecidable, including ontology-mediated query answering, which we will discuss later. A number of decidable fragments of TGDs have been identified in the literature. We review some known (syntactic) restrictions on TGDs that ensure decidability (and even tractability in many cases). Furthermore, adding negative constraints does not increase the complexity [37], and thus we focus on sets of TGDs in the rest of this section.

We focus on the following types of restrictionson TGDs: *guardedness*, *stickiness*, *acyclicity*, and *fullness* [39, 38]. For the first three of these, we consider the "*weak*" versions too, namely, *weak guardedness* [39], *weak stickiness* [38], and *weak acyclicity* [77]. A TGD is *guarded*, if there exists a body atom (also known as "guard") that contains all body variables. The class of guarded TGDs is denoted by G. A key subclass of guarded TGDs is the set of *linear* TGDs, whose bodies consist of a single atom, and thus immediately are guarded. The set of linear TGDs is denoted by L. The class that extends guarded TGDs is the family of *weakly guarded* TGDs, denoted by WG; such TGDs require only the body variables that are considered "harmful" to appear in one of the body atoms (i.e., "the guard") [39]. In particular, a body variable is considered "harmful" if it appears in an *affected* position of a predicate with respect to the set of TGDs Σ. Intuitively, a position of a predicate p is *affected* with respect to Σ if there is some database D such that a null appears in that position in some fact with a predicate p in the chase of the knowledge base (D, Σ), where the chase is the database obtained by iteratively repairing the database so that it satisfies all the dependencies in Σ [39]. Observe that $\mathsf{L} \subset \mathsf{G} \subset \mathsf{WG}$. Another widely studied

restriction on TGDs is *stickiness*, which is inherently different from guardedness. A TGD is *sticky* if every variable that appears more than once in the body must appear (or "stick") in the head atoms. The class of sticky TGDs is denoted by S. *Weak stickiness* generalises stickiness by considering only "harmful" variables, and defines the class WS of weakly sticky TGDs. Observe that $\mathsf{S} \subset \mathsf{WS}$. The third type of syntactic restriction on TGDs is *acyclicity*. A set of TGDs is *acyclic* and belongs to the class A if its predicate graph is acyclic, where a predicate graph of a set of TGDs Σ is a directed graph whose nodes are predicate symbols that appear in Σ and there is an edge between predicate symbols r and p if there is some TGD σ, where r appears in the body and p appears in the head of σ. This class of TGDs can also be seen to contain only non-recursive sets of TGDs. A set of TGDs is *weakly acyclic*, if its dependency graph enjoys a certain acyclicity condition, which guarantees the existence of a finite canonical model [77]. In particular, some of the edges are labelled as *special* in this dependency graph and we say that a set of TGDs is *weakly acyclic* if there is no cycle in this dependency graph that contains this special edge. We have that $\mathsf{A} \subset \mathsf{WA}$. It is also known that $\mathsf{WA} \subset \mathsf{WS}$ [38]. Another key fragment is *full* TGDs. Such TGDs do not have any existentially quantified variables, and the class of such TGDs is denoted by F. Full TGDs are equivalent to Datalog rules [1]. We can require TGDs to be full in addition to satisfying the other restrictions such as linearity, guardedness, stickiness, or acyclicity. These classes are denoted by $\mathsf{LF}, \mathsf{GF}, \mathsf{SF}$, and AF, respectively. It is known that $\mathsf{F} \subset \mathsf{WA}$ [77, 65] and $\mathsf{F} \subset \mathsf{WG}$ [39]. We illustrate the connection between the introduced classes in Figure 2.2.

2.3 Description Logics

Another prominent family of knowledge representation languages are *description logics, DLs* [59]. First, we define the syntax of description logics. A *vocabulary* of a DL consists of three countably infinite, pairwise disjoint sets of symbols, namely, the set of individual names, $\mathsf{N_I}$, the set of concept names, $\mathsf{N_C}$, and the set of role names, $\mathsf{N_R}$. DL concepts are built inductively from concept and role names, using the constructors given in Table 2.1. We sometimes refer to any inductively built concept as a *general concept*, and to a concept from $\mathsf{N_C}$ as an *atomic concept*.

Then, concepts and roles are used to construct axioms and assertions, which, in turn, form knowledge bases.

Definition 2.5 (Knowledge Base)**.** A *general concept inclusion (GCI)* is an expression of the form $C \sqsubseteq D$, where C and D are general concepts. A *role inclusion* is an

Table 2.1: Syntax and semantics of concept descriptions (top), TBox axioms (middle), and ABox assertions (bottom) for DLs.

Name	Syntax	Semantics
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
negation ($\mathcal{C}$)	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
union ($\mathcal{U}$)	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\left\{a \mid \exists b \in C^{\mathcal{I}}.(a,b) \in r^{\mathcal{I}}\right\}$
universal restriction	$\forall r.C$	$\left\{a \mid \forall b \in C^{\mathcal{I}}.(a,b) \in r^{\mathcal{I}}\right\}$
nominal ($\mathcal{O}$)	$\{a\}$	$\left\{a^{\mathcal{I}}\right\}$
at least restrictions ($\mathcal{Q}$)	$\leq n\, r.C$	$\{d \mid \#\{e \mid (d,e) \in r^{\mathcal{I}}\} \leq n\}$
at most restrictions ($\mathcal{Q}$)	$\geq n\, r.C$	$\{d \mid \#\{e \mid (d,e) \in r^{\mathcal{I}}\} \geq n\}$
role inverse ($\mathcal{I}$)	r^-	$\{(b,a) \mid (a,b) \in r^{\mathcal{I}}\}$
TBox Axioms		
GCI	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
role inclusion ($\mathcal{H}$)	$r \sqsubseteq s$	$r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$
transitivity axiom	$\mathit{trans}(r)$	$r^{\mathcal{I}} = (r^{\mathcal{I}})^+$
ABox Assertions		
concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$r(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

expression of the form $r \sqsubseteq s$ and a *transitivity axiom* is an expression $\mathit{trans}(r)$, where r and s are roles. A *TBox axiom* is a GCI, a role inclusion, or a transitivity axiom, and a *TBox*, or an *ontology*, is a finite set of TBox axioms.

A *concept assertion* is an expression of the form $C(a)$, and a *role assertion* is an expression of the form $r(a,b)$, where $C \in \mathsf{N_C}, r \in \mathsf{N_R}$, and $a, b \in \mathsf{N_I}$. An *ABox assertion* is either a concept assertion or a role assertion. An *ABox* is a finite set of ABox assertions. A *knowledge base* is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is a TBox, and $\mathcal{A}$ is an ABox. ∎

The semantics of DLs is given in terms of *interpretations*, where concepts are unary predicates and roles are binary predicates.

Definition 2.6. An *interpretation* $\mathcal{I}$ is a pair $\left(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\right)$ of a non-empty *interpretation domain* $\Delta^{\mathcal{I}}$ and an *interpretation function* $\cdot^{\mathcal{I}}$. This function assigns to each individual

name $a \in \mathsf{N_I}$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, to each concept name $A \in \mathsf{N_C}$ a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and to each role name $r \in \mathsf{N_R}$ a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Interpretations are then extended to arbitrary concept descriptions, axioms, and assertions as shown on the right-most column of Table 2.1.

An interpretation $\mathcal{I}$ is a *model of a TBox* $\mathcal{T}$, denoted $\mathcal{I} \models \mathcal{T}$, if it satisfies all the axioms in $\mathcal{T}$. Similarly, $\mathcal{I}$ is a *model of an ABox* $\mathcal{A}$, denoted $\mathcal{I} \models \mathcal{A}$, if it satisfies all the assertions in $\mathcal{T}$. Finally, $\mathcal{I}$ is a *model of a knowledge base* $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, denoted $\mathcal{I} \models \mathcal{K}$, if $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$.

We say that a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is *consistent* if it has a model, otherwise, it is *inconsistent*. ∎

In this work, we consider a wide range of different DLs that are all well-known in the literature. The DL $\mathcal{ALC}$ allows for *GCIs* as TBox axioms, and the concept language of $\mathcal{ALC}$ allows for the constructors *top*, *bottom*, *negation*, *conjunction*, and *existential restrictions*. The DL $\mathcal{S}$ extends $\mathcal{ALC}$ with *transitivity axioms*. The letters $\mathcal{H}$, $\mathcal{I}$, $\mathcal{O}$ and $\mathcal{Q}$ denote *role inclusions*, *inverse roles*, *nominals*, and *qualified number restrictions*, respectively. Depending on what is allowed in the language, we obtain a variety of DLs, such as the very expressive DL $\mathcal{SHIQ}$, which allows all the constructors given in Table 2.1.

The DL $\mathcal{EL}$ is a sub-logic of $\mathcal{ALC}$ which only allows the constructors *top*, *conjunction*, and *existential restrictions*. Note that it does *not* allow the negation constructor. The DL *DL-Lite*$_{core}$ allows axioms of the form $B \sqsubseteq C$, where B and C are concepts that are defined as follows:

$$B := A \mid \exists r.\top \mid \exists r^{-}.\top, \qquad C := B \mid \neg B.$$

The DL *DL-Lite*$_{\mathcal{R}}$ also allows a restricted type of role inclusions. For further details, we refer the reader to the relevant literature [11, 6]. Figure 2.3 summarizes the language inclusions between these DLs.

2.4 Ontology-Mediated Query Answering

Ontology-mediated query answering is a popular paradigm in knowledge representation used to facilitate querying of *heterogeneous* and *incomplete* data sources [22]. This is achieved via an *ontology* that enriches the user query with common-sense knowledge. In this framework, the ontology and the user query are viewed as two components of one composite query, called *ontology-mediated query (OMQ)* [22]. The task

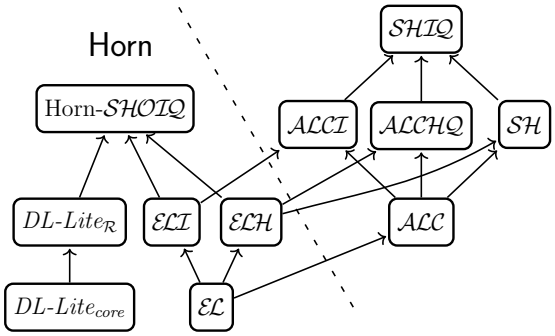

Figure 2.3: Relationships between the DLs considered.

of evaluating such queries is then called *ontology-mediated query answering (OMQA)*. In this section, the notions are introduced using the existential rules terminology but all the definitions apply to description logics with a database replaced by an ABox and a program replaced by a TBox.

Definition 2.7 (Queries)**.** A *conjunctive query* (CQ) Q is a formula of the form $\exists \mathbf{Y} \phi(\mathbf{X}, \mathbf{Y})$, where $\mathbf{X}$ and $\mathbf{Y}$ are sets of variables, and $\phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms (without nulls). A *Boolean conjunctive query* (BCQ) Q is a special case of a CQ of the form $\exists \mathbf{X} \phi(\mathbf{X})$ without free variables, that is, without variables which are not bound by a quantifier. A *union of conjunctive queries (UCQ)* is a disjunction of conjunctive queries, which share free variables $\mathbf{X}$. A *first-order query (FOQ)* is any first-order formula. ∎

In the context of description logics, we take an *instance query (IQ)* to be a concept or a role assertion. Note that the relationship between these query languages is as follows:

$$\text{IQ} \subseteq \text{BCQ} \subseteq \text{CQ} \subseteq \text{UCQ} \subseteq \text{FOQ}$$

In this thesis, we focus on explaining *Boolean query answers*, that is, the queries which do not have any free variables.

Definition 2.8. An *ontology-mediated query (OMQ)* is a pair (Q, Σ), where Q is a query, and Σ is an ontology. Let $\mathcal{L}$ be an ontology language. If Σ is a logical theory from the class $\mathcal{L}$ of ontology languages, then we say that (Q, Σ) is an $\mathcal{L}$-OMQ. Consider a database D and an OMQ (Q, Σ). We say that D *entails* (Q, Σ), denoted $D \models (Q, \Sigma)$, if $\mathcal{I} \models Q$ for every model $\mathcal{I}$ of $D \cup \Sigma$, where $\mathcal{I} \models Q$ denotes that Q is satisfied by an interpretation $\mathcal{I}$ as usual in first-order logic [70].

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). We denote by $\text{OMQA}(\mathcal{Q}, \mathcal{L})$

the problem of OMQA when the queries belong to the query language $\mathcal{Q}$ and ontologies belonging to $\mathcal{L}$. ∎

We will sometimes write $(D, \Sigma) \models Q$, which has the same meaning as $D \models (Q, \Sigma)$. In particular, we will test whether $(D, \Sigma) \not\models \bot$ in order to check whether a knowledge base (D, Σ) is consistent.

A key paradigm in ontology-mediated query answering is the first-order rewritability of queries. Intuitively, if an OMQ is FO-rewritable, we can rewrite it into a FO-query.

Definition 2.9. An OMQ (Q, Σ) is *FO-rewritable*, if there exists a FO-query Q_Σ such that, for any database D 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 ontologies $\mathcal{L}$ is *FO-rewritable*, if it admits an FO-rewriting for every query and ontology in $\mathcal{L}$. ∎

In our complexity analysis, we make the standard assumptions [151]. The *combined complexity* is calculated 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 is bounded by an integer constant. The *fixed-program combined complexity* (or *fp-combined*) is calculated by considering the program as fixed. Finally, the *data complexity* is calculated by considering only the database as the input, i.e., everything else is fixed. Since, in description logics, the arity of predicates is bounded, we consider only *data* and *combined* complexities.

2.4.1 Complexity of Query Answering under Existential Rules

We review the complexity of OMQA under different fragments of existential rules, which is summarised in Table 2.2. Query answering is a core task for all the problems associated with explaining OMQ answers. Therefore, these results are fundamental to our work.

First, we cover the results in data complexity. We recall that answering first-order queries on relational databases is in AC^0 in data complexity [152], which implies that UCQ answering for FO-rewritable TGDs is in AC^0 in data complexity. It was observed that if a set of TGDs enjoys the *bounded derivation-depth property (BDDP)*, then it is FO-rewritable [37]. In particular, a set of TGDs is said to enjoy the BDDP if, whenever an OMQ is entailed, there is a constant k (independent of the data) such that the query holds in the level k of the chase [37]. It was shown that S enjoys the BDDP by Calì et al. [35], and later that L enjoys this property as well [38]. It is

Table 2.2: Complexity of OMQA($\mathcal{Q}, \mathcal{L}$) for $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$ relative to different fragments of existential rules.

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

also straight-forward to see that AF satisfies this property. This implies all the AC^0 membership results in Table 2.2. For WA, it was shown by Fagin et al. [77] that in data complexity OMQA can be solved in polynomial time by an algorithm that runs on universal solutions. For WS, Calì et al. [36] provided an algorithm that solves OMQA in alternating logarithmic space, which coincides with P. These two results imply all the P membership results in data complexity in Table 2.2. The P-hardness of OMQA(BCQ, GF) in data complexity was show by providing a reduction from propositional logic programming [37]. This result implies all the P-hardness results in data complexity in Table 2.2. We further note that some of the results can be inferred in alternative ways, for example, since the full fragment of Datalog$^{\pm}$ corresponds to Datalog, we have that OMQA for the full fragment in data complexity is P-complete.

Next, we cover the complexity results in *fp*-combined, *ba*-combined, and combined complexities. In all those measures, compared to the data complexity, we see an increase in the complexity. OMQA is NP-hard for all classes in *fp*-complexity, which immediately follows from NP-hardness of conjunctive query evaluation in databases [54]. This implies also all the NP-hardness results in *ba*-combined complexity. Furthermore, OMQA is in NP both for WS and WA in *fp*-combined complexity [114], which shows the membership of OMQA in NP in *fp*-complexity for all the languages studied. In *ba*-combined complexity, for the acyclic language of existential rules, we have that OMQA is NEXP-hard, as it can encode an NEXP-complete problem TILING [78]. Furthermore, OMQA(BCQ, A) was shown to be in NEXP in combined complexity by exploiting results on nonrecursive logic programs [61], and thus we have that this problem is NEXP-complete in both *ba*-combined and combined complexity. It was shown that OMQA(BCQ, G) is EXP-complete in *ba*-combined and 2EXP-complete in combined complexity by Calì et al. [39]. For WS and WA, OMQA was shown to be 2EXP-complete both in *ba*-combined and combined complexities by Fagin et al. [77] and Calì et al. [36], respectively. The remaining

Table 2.3: Complexity of OMQA($\mathcal{Q}, \mathcal{L}$) relative to different DLs.

Description Logic	IQ		UCQ	
	data	comb.	data	comb.
DL-Lite$_{core}$, *DL-Lite*$_{\mathcal{R}}$	$\leq$ AC^0	NL	$\leq$ AC^0	NP
$\mathcal{EL}, \mathcal{ELH}$	P	P	P	NP
$\mathcal{ELI}$, Horn-$\mathcal{SHOIQ}$	P	Exp	P	Exp
$\mathcal{ALC}$, $\mathcal{ALCHQ}$	coNP	Exp	coNP	Exp
$\mathcal{ALCI}$, $\mathcal{SH}$, $\mathcal{SHIQ}$	coNP	Exp	coNP	2Exp

results in combined complexity were obtained as follows. By simulating the behavior of a deterministic polynomial space machine, Lukasiewicz et al. [114] showed that OMQA PSpace-hard for L, LF, and AF in combined complexity. Calì et al. [35] showed that lossless Datalog fact inference problem is Exp-hard. Since each lossless Datalog program is a set of sticky TGDs, we have that OMQA is Exp-hard for S and SF. Finally, it was shown by Calì et al. [36] that OMQA is 2Exp-hard both for WA and WA in combined complexity.

2.4.2 Complexity of Query Answering in Description Logics

Now we present the complexity of ontology-mediated query answering in description logics, which is summarised in Table 2.3.

We start with the least expressive description logics. The lightweight DLs *DL-Lite*$_{core}$ and *DL-Lite*$_{\mathcal{R}}$ are FO-rewritable, and thus, as we have seen in Section 2.4.1, query answering for these languages is in AC^0 in data complexity. It is well-known that standard reasoning tasks are NL-complete in these languages [11]. Since instance query answering can be reduced to these standard reasoning tasks, same results apply. On the other hand, for these languages, query answering is NP-complete in combined complexity for UCQs [6, 40]. For the lightweight description logics $\mathcal{EL}$, $\mathcal{ELH}$, and $\mathcal{ELI}$, the tractability results follow from the works of Krisnadhi and Lutz [109], Krötzsch and Rudolph [110], Rosati [141]. in combined complexity, these DLs behave rather differently. In particular, instance query answering for $\mathcal{EL}$ and $\mathcal{ELH}$ is P-complete, while it is Exp-complete for $\mathcal{ELI}$ [10]. Moreover, UCQ answering for $\mathcal{EL}$ and $\mathcal{ELH}$ is NP-complete, and it is Exp-complete for $\mathcal{ELI}$ in combined complexity. For a closely related logic Horn-$\mathcal{SHOIQ}$ it also holds that query answering (both for IQs and UCQs) is P-complete in the data and Exp-complete in combined complexity [75, 126].

The description logics discussed thus far are all Horn. Beyond the DLs covered thus far, the least expressive one that we study is $\mathcal{ALC}$, for which (instance) query answering is coNP-complete in data complexity [42] and the upper bound holds for DLs up to $\mathcal{SHIQ}$ [82]. Standard reasoning tasks in $\mathcal{ALC}$ are already Exp-complete [143], which gives the hardness results for query answering in combined complexity. The membership results in Table 2.3 for DLs above $\mathcal{ALC}$ were provided by Lutz [117]. For $\mathcal{ALCI}$, UCQ answering is 2Exp-hard [117] and, for $\mathcal{SHIQ}$, it is in 2Exp [82]. This implies 2Exp-completeness results in Table 2.3.

Chapter 3

Explanations for Query Answers under Existential Rules

As a form of logical entailment, ontology-mediated query answering is fully interpretable, which makes it possible to derive explanations for query answers. There has been a significant amount of interest in tracking down and understanding the causes of various types of entailments in ontologies. However, explaining ontology-mediated answers had received little attention, prior to our work [48]. In this chapter, we adapt ideas from axiom pinpointing and propositional abduction to explaining ontology-mediated query answers. We provide a thorough complexity analysis for several decision problems associated with minimal explanations under existential rules.

This chapter is based on the paper published in the proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) in 2019 [48].

3.1 Minimal Explanations

In our approach, a minimal explanation for an ontology-mediated query answer is an inclusion-minimal subset of a database, which supports such query answer. The main advantage of such a definition is that it allows to abstract away from a particular proof technique. Our notion of a minimal explanation draws inspiration from existing research in knowledge representation. The notion of a *cause* in the works on explaining ontology-mediated query answers under inconsistency-tolerant semantics coincides with our notion of a minimal explanation [24]. In propositional abduction, a minimal explanation is a minimal set of hypotheses that supports the entailment of the observed manifestations [71], and, in axiom pinpointing, a minimal explanation is a minimal set of axioms that supports the entailment of a chosen axiom [135].

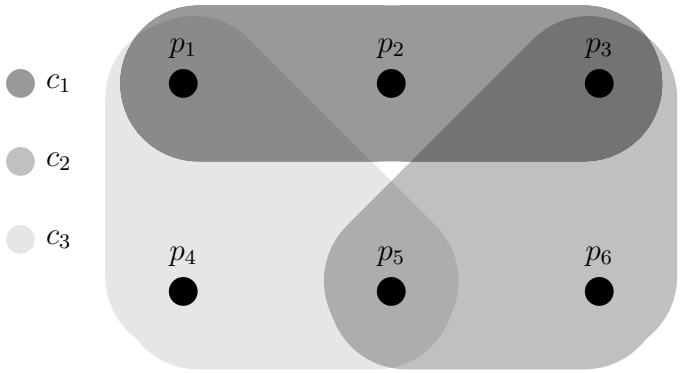

Figure 3.1: Protein network, where each complex c_i contains some of the proteins $p_1, \ldots, p_6$.

In this work, we assume that the ontology is fixed, and so we take explanations to be subsets of the database.

Definition 3.1 (MinEX)**.** For a database D and an OMQ (Q, Σ), where Σ is a set of existential rules, and Q is a query, such that $D \models (Q, \Sigma)$, an *explanation for* $D \models (Q, \Sigma)$ is a subset $E \subseteq D$ of facts entailing the OMQ (Q, Σ), i.e., $E \models (Q, \Sigma)$.

A *minimal explanation* E*, or* MinEX*, for* $D \models (Q, \Sigma)$ is an explanation for $D \models (Q, \Sigma)$ that is *subset-minimal*, i.e., there is no proper subset $E' \subsetneq E$ that is an explanation for $D \models (Q, \Sigma)$.

Often, we say that E is a MinEX without explicitly mentioning the database or the OMQ when they are clear from the context.

We provide a running example that used throughout this chapter to illustrate the different problems studied. This example is taken from the experimental design for protein networks [106, 139].

Example 3.2. We consider the protein containment scenario illustrated in Figure 3.1. In this example, we have the proteins $p_1, \ldots, p_6$ and the complexes c_1, c_2, c_3, where each complex is a set of some of the proteins. We want to find a minimal subset of the proteins that covers all the complexes, i.e., a minimal subset of the proteins that has at least one representative from each complex.[1]

We can express this problem as the search for MinEXs in a way that protein covers are in bijection with MinEXs for an OMQ entailment. Consider a database that encodes all the proteins:

$$D_{prot} = \{protein(p_i) \mid 1 \leq i \leq 6\}.$$

[1]This example could be adapted to any other task that is equivalent to finding minimal transversals of hypergraphs [80, 83].

The following program states which proteins are contained in which complexes:

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

and the following query asks for every complex to be covered:

$$Q_{prot} = covered(c_1) \wedge covered(c_2) \wedge covered(c_3).$$

Consider now a subset $E \subseteq D_{prot}$. Clearly, $E \models (Q_{prot}, \Sigma_{prot})$ if and only if $E \models (covered(c_i), \Sigma_{prot})$ for every complex c_i. Thus, MinEXs for $D_{prot} \models (Q_{prot}, \Sigma_{prot})$ are in bijection with the minimal protein covers of complexes. ∎

The objective of this work is to understand the complexity of the tasks associated with MinEXs under existential rules.

3.2 Explanation Problems

In this section, we give an overview of the six problems, considered in this chapter. These problems are inspired by the decision problems, studied in axiom pinpointing [135] and logic-based abduction [71], where explanations are inclusion-minimal subsets supporting entailment. These problems concern recognising MinEXs, determining roles of facts in explaining and cardinality of minimal explanations. We introduce them for any query language $\mathcal{Q}$ and the class of TGDs $\mathcal{L}$.

The most fundamental problem that we study is recognising a MinEX for a given OMQ entailment. This is a natural decision version of the functional problem of constructing a minimal explanation.

Problem: Is-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: A database D, a query Q from the class $\mathcal{Q}$, a program Σ from the class $\mathcal{L}$ of TGDs, such that $D \models (Q, \Sigma)$, and a set of facts $E \subseteq D$.
Question: Is E a MinEX for $D \models (Q, \Sigma)$?

Note that the first argument in Is-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$ indicates that we are considering the problem of recognising *subset-minimal* minimal explanations. We will consider explanations under different minimality criterion in Chapter 6. In the complexity analysis, Is-MinEX serves as a baseline for the other problems studied. We illustrate this problem on the setting, presented in Example 3.2.

Example 3.3. The following sets are MinEXs for $D_{prot} \models (Q_{prot}, \Sigma_{prot})$:

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

They give the minimal protein covers of the complexes. On the contrary, the subset $\{protein(p_4), protein(p_5), protein(p_6)\}$ is not a MinEX, as these proteins do not cover all the complexes and thus does *not entail* $(Q_{prot}, \Sigma_{prot})$. Also, a subset might entail the OMQ but not be a MinEX. For example, $\{protein(p_1), protein(p_2), protein(p_3)\}$ entails $(Q_{prot}, \Sigma_{prot})$, but it is not a MinEX, as it is *not* minimal; the fact $protein(p_2)$ can be removed without breaking the entailment. ∎

Another fundamental problem is recognising the set of all the MinEXs for a given OMQ entailment. This problem is the decision version of the problem of constructing the set of all MinEXs, which is particularly important when we want to gain complete understanding of how a particular OMQ can be entailed.

Problem: All-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: A database D, a query Q from the class $\mathcal{Q}$, a program Σ from the class $\mathcal{L}$ of TGDs, and a set $\mathcal{E} \subseteq \mathcal{P}(D)$ of subsets of the database.
Question: Is $\mathcal{E}$ the set of all MinEXs for $D \models (Q, \Sigma)$?

In our running example, the set of all MinEXs corresponds to the set of all minimal protein covers.

Example 3.4. The following set of subsets of the database D_{prot} is exactly the set of all MinEXs for $D_{prot} \models (Q_{prot}, \Sigma_{prot})$:

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

This set corresponds to the set of all minimal protein covers of the complexes. ∎

In some applications, the user might specify some subsets of the database as "forbidden". For example, the user might know that some facts are incompatible or might be interested in explanations not containing some specific sets of facts. We

study this problem, MinEX-Irrel, of deciding whether there is a MinEX for an OMQ entailment that avoids forbidden sets of facts.

Problem: MinEX-Irrel$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: A database D, a query Q from the class $\mathcal{Q}$, a program Σ from the class $\mathcal{L}$ of TGDs, and a set $\mathcal{F} \subseteq \mathcal{P}(D)$.
Question: Is there a MinEX E for $D \models (Q, \Sigma)$ such that $F \not\subseteq E$ for every $F \in \mathcal{F}$?

We illustrate this problem on our running example.

Example 3.5. Suppose that the set captures the configurations of proteins that are not allowed in a cover:

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

In this case, $\{protein(p_3), protein(p_4)\}$ is a desirable MinEX, which does not contain any set from $\mathcal{F}$ and covers all the complexes. On the other hand, note that neither $\{protein(p_1), protein(p_3)\}$ nor $\{protein(p_3), protein(p_5)\}$ are desirable MinEXs as they both contain some forbidden set. ∎

Another fundamental problem, MinEX-Rel, is deciding the relevance of a fact in explaining the entailment of a particular OMQ. We say that a fact ψ is *relevant* for $D \models (Q, \Sigma)$ iff there is a MinEX E for $D \models (Q, \Sigma)$ with $\psi \in E$. This problem is of interest when we want to decide whether a particular fact *can* contribute to the OMQ entailment.

Problem: MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: A database D, a query Q from the class $\mathcal{Q}$, a program Σ from the class $\mathcal{L}$ of TGDs, and a fact $\psi \in D$.
Question: Is ψ relevant for $D \models (Q, \Sigma)$, i.e., is there a MinEX E for $D \models (Q, \Sigma)$ such that $\psi \in E$?

We illustrate this problem on our running example.

Example 3.6. Suppose that we are interested whether a fact $\psi = protein(p_6)$ is contained in some MinEX. Observe that the sets $\{protein(p_1), protein(p_6)\}$ and $\{protein(p_2), protein(p_4), protein(p_6)\}$ are MinEXs for $D_{prot} \models (Q_{prot}, \Sigma_{prot})$ that contain this fact. Therefore $\psi = protein(p_6)$ is relevant for $D_{prot} \models (Q_{prot}, \Sigma_{prot})$, and, in particular, there is a protein cover that contains the protein p_6. ∎

We also study two cardinality-based problems for MinEXs. These problems can be seen as decision versions of the optimisation problems of finding the size of a smallest and a largest MinEX. The first problem, Small-MinEX, asks whether there is a MinEX whose size is smaller than some given threshold number, and the second, Large-MinEX, whether there is a MinEX whose size is larger than some given threshold number.

Problem: Small-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: A database D, a query Q from the class $\mathcal{Q}$, a program Σ from the class $\mathcal{L}$ of TGDs, and an integer $n \geq 1$.
Question: Is there a MinEX E for $D \models (Q, \Sigma)$ such that $|E| \leq n$?

Problem: Large-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: A database D, a query Q from the class $\mathcal{Q}$, a program Σ from the class $\mathcal{L}$ of TGDs, and an integer $n \geq 1$.
Question: Is there a MinEX E for $D \models (Q, \Sigma)$ such that $|E| \geq n$?

We illustrate these problems on our running example.

Example 3.7. Observe that every MinEX for the OMQ entailment, defined in Example 3.2, is either of size 2 or 3. For example, $\{protein(p_1), protein(p_3)\}$ is a MinEX of size 2 and $\{protein(p_2), protein(p_4), protein(p_6)\}$ is a MinEX of size 3, which are a smallest and a largest MinEX for $D_{prot} \models (Q_{prot}, \Sigma_{prot})$, respectively. ∎

3.3 Recognising Minimal Explanations

In this section, we start the complexity analysis by considering the fundamental problem Is-MinEX: Given a database D, an OMQ (Q, Σ) (with $D \models (Q, \Sigma)$) and a subset $E \subseteq D$, is E a MinEX for $D \models (Q, \Sigma)$?

We start by showing the membership results in Table 3.1. We first present a general approach to solve Is-MinEX. This general algorithm uses an oracle for the ontology-mediated query answering problem and thus it is applicable to all classes of TGDs and complexity measures.

We start by showing the membership results in Table 3.1. We first present a general approach to solve Is-MinEX. This general algorithm uses an oracle for the ontology-mediated query answering problem and thus it is applicable to all classes of TGDs and complexity measures.

A fundamental insight in this algorithm is that non-entailment is preserved by removing elements from the databases. Namely, if a set S does not entail an OMQ,

Table 3.1: Complexity results for Is-MinEX($\subseteq, \mathcal{Q}, \mathcal{L}$) with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**Data**	*fp*-**comb.**	*ba*-**comb.**	**Comb.**
L, LF, AF	$\leq \text{AC}^0$	D^P	D^P	PSpace
S, SF	$\leq \text{AC}^0$	D^P	D^P	Exp
A	$\leq \text{AC}^0$	D^P	D^{Exp}	D^{Exp}
G	P	D^P	Exp	2Exp
F, GF	P	D^P	D^P	Exp
WS, WA	P	D^P	2Exp	2Exp

then every subset S' of S does not entail the OMQ either. This implies that to ensure a subset-minimality of E, it is enough to check that removing any *single* element from E yields a set that does *not* entail the OMQ, that is, we do not need to check this for all the subsets of E.

Theorem 3.8. *For a class $\mathcal{L}$ of TGDs, if* OMQA(UCQ, $\mathcal{L}$) *is in the complexity class $\mathcal{C}$, then* Is-MinEX($\subseteq$, UCQ, $\mathcal{L}$) *can be decided by a check $\mathcal{C}$ and a linear number of checks in co-$\mathcal{C}$.*

Proof. Let D be a database, (Q, Σ) be an $\mathcal{L}$-OMQ, and E be a subset of D. Testing whether E is a MinEX for $D \models (Q, \Sigma)$ involves checking whether (i) E entails (Q, Σ) and (ii) E is subset-minimal. The entailment can be checked with a single $\mathcal{C}$ test. To check subset-minimality of E, as observed above, it is enough to check if removing any element from E results in a set that does *not* entail (Q, Σ) (rather than checking this for all subsets of E). Therefore, we need to perform $|E|$ *non*-entailment checks, i.e. a linear number of checks, each of them in co-$\mathcal{C}$. □

We observe that the above result implies all the membership results in Table 3.1 apart from the AC^0, D^P and D^{Exp} membership results. For example, if OMQA is in P, by Theorem 3.8, we know that we can solve Is-MinEX with a check in P test and a linear number of checks in co-P. Note that co-P = P, and so Is-MinEX is solvable with a linear number checks in P, which can be combined as a single polynomial time test. Therefore, Is-MinEX is in P in this case. The Exp and 2Exp membership results follow similarly.

The following corollary shows that D^P and D^{Exp} membership results in Table 3.1 follow from arguments based on the proof of Theorem 3.8.

Corollary 3.9. *Let $\mathcal{L}$ be a class of TGDs.*

- *If* OMQA(UCQ, $\mathcal{L}$) *is in* NP, *then* Is-MinEX($\subseteq$, UCQ, $\mathcal{L}$) *is in* $\text{D}^\text{P}_;$

- *If* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in* NExp*, then* Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* D^{Exp}.

Proof. In the proof of Theorem 3.8, observe that, when $\mathcal{C} \in \{\text{NP}, \text{NExp}\}$, a linear number of co-$\mathcal{C}$ tests can be combined as a single co-$\mathcal{C}$ test by combining linearly many certificates. This implies that minimality can be checked in co-$\mathcal{C}$. Thus we obtain that Is-MinEX is in $\mathcal{C} \wedge$ co-$\mathcal{C}$ and the result follows.

□

Next, we show that Is-MinEX is in AC^0 in data complexity for FO-rewritable languages, that is, the complexity does not increase from OMQA.

Theorem 3.10. *If a class of TGDs* $\mathcal{L}$ *is FO-rewritable, then* Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* AC^0 *in data complexity.*

Proof. Let (Q, Σ) be a fixed $\mathcal{L}$-OMQ over a fixed relational vocabulary σ, where $\mathcal{L}$ is FO-rewritable. Since (Q, Σ) is FO-rewritable, there exists a FO-formula Q_Σ such that, for any set of facts E over σ, we have that

$$E \models Q_\Sigma \text{ iff } E \models (Q, \Sigma). \tag{3.1}$$

It is well-known that a problem is in AC^0 if it can be expressed by a fixed FO-formula [153]. Hence, in the rest of the proof, we show that in this case, we can express Is-MinEX by a fixed FO-formula. First, we can encode subset-minimality, that is, that removing any fact yields a set that does not entail the OMQ (Q, Σ), in a FO-formula ϕ_{min}. Let $\mathbf{R}'$ be the set of predicate symbols in the relational vocabulary σ, and so they are the only predicates that might appear in the OMQ (Q, Σ) and an input explanation E. We assume that $\mathbf{R}$ is finite as we are in the data complexity. Take ϕ_{min} to be

$$\phi_{min} = \bigwedge_{p \in \mathbf{R}'} \left(\forall \mathbf{X}_p \left(p\left(\mathbf{X}_p\right) \rightarrow \neg Q_\Sigma^{\neq p}(\mathbf{X}_p) \right) \right),$$

where $Q_\Sigma^{\neq p}(\mathbf{X}_p)$ is obtained from Q_Σ by replacing each p-atom $p(\mathbf{t})$ in Q_Σ by $\big(p(\mathbf{t}) \wedge (p(\mathbf{t}) \neq p(\mathbf{X}_p)\big)$. Now we show that for any set E of facts over σ, we have that

$$E \models \phi_{min} \text{ iff no proper subset of } E \text{ entails } (Q, \Sigma). \tag{3.2}$$

Observe that $E \models Q_\Sigma^{\neq p}(\mathbf{c})$ iff $E \setminus \{p(\mathbf{c})\}$ does not entail the OMQ (Q, Σ). Therefore we have that $E \models \forall \mathbf{X}_p \left(p\left(\mathbf{X}_p\right) \rightarrow \neg Q_\Sigma^{\neq p}(\mathbf{X}_p) \right)$ iff, for any p-fact $p(\mathbf{c}) \in E$, we have that $E \setminus \{p(\mathbf{c})\}$ does not entail the OMQ (Q, Σ). Since ϕ_{min} takes a conjunction over all the predicate symbols in $\mathbf{R}$', we have that Equation (3.2) holds.

We take $\phi = Q_\Sigma \wedge \phi_{min}$. Let D be a set of facts over σ and $E \subseteq D$. We claim that

$$E \models \phi \text{ iff } E \text{ is a MinEX for } D \models (Q, \Sigma). \tag{3.3}$$

Note that Equation (3.3) is a straightforward consequence of Equation (3.1) and Equation (3.2). This shows indeed that we can encode the property of being a MinEX as a FO-formula when $\mathcal{L}$ is FO-rewritable. □

Summarising the membership results, we see that the upper bounds for the complexity of recognising a minimal explanation, compared to the complexity of query answering, stay the same when OMQA is complete for a *deterministic class*, and increase when OMQA is complete for a *nondeterministic class*.

In the rest of this section, we show that these upper bounds for Is-MinEX have matching lower bounds. First, we show that Is-MinEX$(\subseteq, \text{UCQ}, \mathsf{GF})$ is P-hard in data complexity. We reduce OMQA$(\text{UCQ}, \mathsf{GF})$ to the corresponding Is-MinEX problem.

Theorem 3.11. Is-MinEX$(\subseteq, \text{BCQ}, \mathsf{GF})$ *is* P*-hard in data complexity.*

Proof. We show hardness of Is-MinEX$(\subseteq, \text{BCQ}, \mathsf{GF})$ in data complexity by reduction from OMQA$(\text{BCQ}, \mathsf{GF})$ in data complexity. Let D be a database and (Q, Σ) be an OMQ, where (Q, Σ) is fixed. We reduce this problem to Is-MinEX$(\subseteq, \text{BCQ}, \mathsf{GF})$ in data complexity. We construct a database D', a subset E' and an OMQ (Q', Σ').

We obtain D' from D and Q as follows. For each fact $p(\mathbf{a}) \in D$, add to D' two facts $p'(\mathbf{a}, i)$ and $r(i-1, i)$, where p' is a fresh predicate symbol for p, and i is an incremental integer starting from 1 that is used to "enumerate" all the facts in D' coming from D (i.e. different $p'(\cdot)$ facts have different integers i in the last position). Further, we add two additional facts $u(0)$ and $num\_facts(|D|)$ to D'. Finally, add a grounding of Q to D', where each variable is grounded with a fresh constant, and call this set D'_{ground}, which ensures that Q is entailed by this new database. We take the set E' to be equal to $D' \setminus D'_{ground}$. Observe that D' can be computed in logarithmic space.

The OMQ (Q', Σ') in Is-MinEX$(\subseteq, \text{BCQ}, \mathsf{GF})$ is obtained as follows. The program Σ' contains Σ. Moreover, for each predicate p of the relational schema (which is fixed, because we are in data complexity setting), add to Σ' the rules $p'(\mathbf{X}, J) \to p(\mathbf{X})$ and $p'(\mathbf{X}, J) \to t(J)$. Further, the program Σ' contains two other rules, namely, $u(I) \wedge r(I, J) \wedge t(J) \to u(J)$ and $u(J) \wedge num\_facts(J) \to all()$, which ensure that the fact $all()$ holds only if all $p'(\mathbf{X}, J)$ facts are present. The query is $Q' = Q \wedge all()$.

Observe that (Q', Σ') is independent from D (and hence fixed). We further note that $D' \models (Q', \Sigma')$.

We show that $D \models (Q, \Sigma)$ if and only if E' is a MinEX for $D' \models (Q', \Sigma')$.

($\Rightarrow$) Suppose that $D \models (Q, \Sigma)$. Observe that via the rules $p'(\mathbf{X}, J) \rightarrow p(\mathbf{X})$ in Σ' all the facts in D follow from E'. This implies that $E' \models (Q, \Sigma')$. By construction, we have that $E' \models (all(), \Sigma')$, and thus $E' \models (Q', \Sigma')$. On the other hand, E' is subset-minimal for (Q', Σ') as, by construction, removing any fact from E' would break entailment of $(all(), \Sigma')$.

($\Leftarrow$) For the converse, suppose that E' is a MinEX for $D' \models (Q', \Sigma')$. Note that all the facts from D follow from E' and Σ'. Since there are no other inferred facts that might contribute to the entailment of (Q, Σ) (and $E' \models (Q, \Sigma')$), we have that $D \models (Q, \Sigma)$. □

As a result, we have that Is-MinEX$(\subseteq, \text{BCQ}, \mathcal{L})$ is P-hard for any language $\mathcal{L} \in \{\mathsf{GF}, \mathsf{G}, \mathsf{F}, \mathsf{WA}, \mathsf{WS}\}$ in data complexity due to the language containment. Next, we show that the problem Is-MinEX$(\subseteq, \text{BCQ}, \mathcal{L})$ is D^{P}-hard in *fp*-combined complexity even when the program is empty.

Theorem 3.12. Is-MinEX$(\subseteq, \text{BCQ}, \mathcal{L}_\emptyset)$ *is* D^{P}*-hard in fp-combined complexity, where* $\mathcal{L}_\emptyset$ *is the class of TGDs that contains only the empty set.*

Proof. We reduce to Is-MinEX$(\subseteq, \text{BCQ}, \mathcal{L}_\emptyset)$ the following D^{P}-complete problem:

Problem: Sat-Unsat [129]
Input: Two 3CNF formulas ϕ and ψ.
Question: Is ϕ satisfiable and ψ unsatisfiable?

Let $(\phi(x_1, \ldots, x_n), \psi(y_1, \ldots, y_m))$ be an instance of Sat-Unsat. We build an instance of Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$, formed by a database D, a subset E of D and a query Q. We take Σ to be the empty set, which is not part of the input as we are in *fp*-combined complexity.

We encode the satisfiability and unsatisfiability of the formulas ϕ and ψ via the database D and the query Q. First, we construct the database D. We add to D the facts that will be used to impose the consistency of the truth assignments to the literals:

$$\begin{array}{llll} simlit(0,0), & simlit(0,2), & opplit(0,1), & simlit(2,2), \\ simlit(1,1), & simlit(1,2), & opplit(1,0), & opplit(2,2), \end{array}$$

where 0 and 1 are constants with the meaning of the Boolean values *true* and *false*, respectively, and 2 is an additional constants used as a "jolly." Intuitively, it will be used to bypass the check of satisfaction of ψ.

Further, we add to D the facts that capture the satisfying assignments to the *literals* for a *clause*:

$$clsat_\phi(\alpha_1, \alpha_2, \alpha_3), \qquad\qquad clsat_\psi(\alpha_1, \alpha_2, \alpha_3),$$

where each $\alpha_i \in \{0, 1\}$ and at least one of α_i for $i \in \{1, 2, 3\}$ is 1. Note that there are 7 such facts for each of ϕ and ψ. Further, we add to D the facts:

$$clsat_\phi(2, 2, 2), \qquad\qquad clsat_\psi(2, 2, 2),$$

which intuitively will be used to bypass the satisfiability checks of ϕ and ψ, respectively. We define the set of *structural* facts D_{st} to be $D \backslash \{clsat_\phi(2, 2, 2), clsat_\psi(2, 2, 2)\}$. We take the candidate explanation E to be $D \setminus \{clsat_\phi(2, 2, 2)\}$.

Next, we construct the query Q, which is rather complicated, and so we present it gradually by giving intuition for each piece introduced. The first piece of the query checks whether all the structural facts are in the candidate MinEX:

$$config \equiv \bigwedge_{p(\mathbf{c}) \in D_{st}} p(\mathbf{c}).$$

The second piece of the query "assigns" truth values to the variables x_i's and y_i's (captured by the query variables T_i^ϕ and T_i^ψ) and "copies" this assignment onto an occurrence of x_i and y_i as a positive literal in the formulas ϕ and ψ, respectively. Below we use the following notation: $\ell_{j,k}^\alpha$ is the k^{th} literal in the j^{th} clause in the formula $\alpha \in \{\phi, \psi\}$, and $v_{j,k}^\alpha$ is the variable of $\ell_{j,k}^\alpha$. This piece of the query is

$$copy \equiv \bigwedge_{i=1}^{n} simlit(T_i^\phi, T_{j,k}^\phi) \wedge \bigwedge_{i=1}^{m} simlit(T_i^\psi, T_{j,k}^\psi).$$

where, in each atom $simlit(T_i^\alpha, T_{j,k}^\alpha)$, the query variables T_i^α and $T_{j,k}^\alpha$ represent the same variable. Observe that, in order for *copy* to work properly, each variable x_i and y_i must appear as a positive literal at least once. This can be assumed without loss of generality, because if x_i (resp., y_i) always appears as a negative literal in all the clauses of ϕ (resp., ψ), then we can replace all the occurrences of the negative literal $\neg x_i$ (resp., $\neg y_i$) with the positive literal x_i (resp., y_i) without altering the satisfiability of ϕ (resp., ψ).

The third piece of the query forces that the ground values assigned to the various variables $T^{\phi}_{j,k}$ and $T^{\psi}_{j,k}$ simulating the assignments to the literals are consistent. In the notation below, for $\alpha \in \{\phi, \psi\}$, $\ell^{\alpha}_{j,k} \sim \ell^{\alpha}_{j',k'}$ means that $v^{\alpha}_{j,k} = v^{\alpha}_{j',k'}$ and the literals $\ell^{\alpha}_{j,k}$ and $\ell^{\alpha}_{j',k'}$ are both positive or negative, while $\ell^{\alpha}_{j,k} \nsim \ell^{\alpha}_{j',k'}$ means that $v^{\alpha}_{j,k} = v^{\alpha}_{j',k'}$ and one literal is positive and the other is negative. We take this part of the query to be

$$\mathit{consist} \equiv \bigwedge_{\alpha \in \{\phi,\psi\}} \left(\bigwedge_{\ell^{\alpha}_{j,k} \sim \ell^{\alpha}_{j',k'}} \mathit{simlit}(T^{\alpha}_{j,k}, T^{\alpha}_{j',k'}) \wedge \bigwedge_{\ell^{\alpha}_{j,k} \nsim \ell^{\alpha}_{j',k'}} \mathit{opplit}(T^{\alpha}_{j,k}, T^{\alpha}_{j',k'}) \right),$$

where $T^{\alpha}_{j,k}$ is a variable with the same meaning as above.

The last piece of the query captures satisfiability of ϕ and ψ:

$$\mathit{satisfied} \equiv \bigwedge_{c_j \in \phi} \mathit{clsat}_{\phi}(T^{\phi}_{j,1}, T^{\phi}_{j,2}, T^{\phi}_{j,3}) \wedge \bigwedge_{c_j \in \psi} \mathit{clsat}_{\psi}(T^{\psi}_{j,1}, T^{\psi}_{j,2}, T^{\psi}_{j,3}).$$

The resulting query is

$$Q = \mathit{config} \wedge \mathit{copy} \wedge \mathit{consist} \wedge \mathit{satisfied}.$$

Observe that $D \models (Q, \Sigma)$ since $\mathit{clsat}_{\phi}(2,2,2)$ and $\mathit{clsat}_{\psi}(2,2,2)$ are in D. Assigning the value 2 to all the query variables T^{α}_{i} and $T^{\alpha}_{j,k}$ satisfies the query. It remains to show that ϕ is satisfiable and ψ is unsatisfiable if and only if E is a MinEX for $D \models (Q, \Sigma)$.

($\Rightarrow$) For the forward direction, suppose that (ϕ, ψ) is a 'yes'-instance of Sat-Unsat. Assume for a contradiction that E is not a MinEX for $D \models (Q, \Sigma)$. Then either $E \not\models (Q, \Sigma)$ or E is not subset-minimal. In the former case, this implies that ϕ is unsatisfiable, since the containment of $\mathit{clsat}_{\psi}(2,2,2)$ in E guarantees that all clsat_{ψ} conjuncts in the query are satisfied, which is a contradiction. In the latter case, we have that $E \setminus \{\mathit{clsat}_{\psi}(2,2,2)\} \models (Q, \Sigma)$ as the *config* part of the query ensures that all facts, apart from $\mathit{clsat}_{\psi}(2,2,2)$ must be in every explanation. This implies that ψ is satisfiable: a contradiction. Hence, E is a MinEX for $D \models (Q, \Sigma)$.

($\Leftarrow$) For the other direction, suppose now that (ϕ, ψ) is a 'no'-instance of Sat-Unsat. This means that ϕ is unsatisfiable or ψ is satisfiable. In the former case, $E \not\models (Q, \Sigma)$ as the query is constructed so that it is entailed only if ϕ is satisfiable, and hence E is not an explanation at all, and in the latter case we have that E is an explanation but not minimal, since $E \setminus \{\mathit{clsat}_{\psi}(2,2,2)\} \models (Q, \Sigma)$. □

This result implies that Is-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ is D^{P}-hard for all considered languages $\mathcal{L}$ in *fp*- and *ba*-combined complexity. We are only left with the complexity results in the *ba*-combined and combined complexity.

In the following, we show the $\mathrm{D}^{\mathrm{EXP}}$-hardness of Is-MinEX$(\subseteq, \mathrm{UCQ}, \mathsf{A})$ in *ba*-combined and the combined complexity.

Theorem 3.13. Is-MinEX$(\subseteq, \mathrm{BCQ}, \mathsf{A})$ *is* $\mathrm{D}^{\mathrm{EXP}}$*-hard in ba-combined complexity.*

Proof. We give a reduction from the following $\mathrm{D}^{\mathrm{EXP}}$-complete problem:

Problem: ExpTiling-NonExpTiling [76]
Input: Two tiling problems TP_1 and TP_2 for the exponential square $2^n \times 2^n$, and two initial tiling conditions w_1 and w_2.
Question: Does TP_1 have no solution with w_1 *and* TP_2 have a solution with w_2?

For the tiling problems, we use the encoding, presented in [76]. We create two copies for $\Sigma_{TP_i,|w_i|}$, D_{TP_i}, and D_{w_i} for $i = 1, 2$, using disjoint predicates (we index them with superscript i). That is, we take $\Sigma^1 = {\Sigma_{TP_1,|w_1|}}^1$, $D^1 = {D_{TP_1}}^1 \cup {D_{w_1}}^1$, and $\Sigma^2 = {\Sigma_{TP_2,|w_2|}}^2$, $D^2 = {D_{TP_2}}^2 \cup {D_{w_2}}^2$. We take the program to be $\Sigma = \Sigma^1 \cup \Sigma^2$, and the database to be $D = D^1 \cup D^2 \cup \{\mathit{tiling}^1(), \mathit{tiling}^2()\}$.

The query is $Q = D^1, D^2, \mathit{tiling}^1(), \mathit{tiling}^2()$, where with symbols D^1 and D^2 in the query we mean the conjunction of all database facts from D^1 and D^2, respectively. Observe that $D \models (Q, \Sigma)$. We consider $E = D \setminus \{\mathit{tiling}^2()\} = D^1 \cup D^2 \cup \{\mathit{tiling}^1()\}$ as the set that we have to decide whether it is a MinEX.

We show that TP_1 does not have solution with w_1 *and* TP_2 has a solution with w_2 if and only if E is a MinEX for $D \models (Q, \Sigma)$. Notice that the set of facts D^i entails the atom $\mathit{tiling}^i()$ via the mediation of the ontology Σ^i if and only if TP_i has a solution with initial condition w_i.

($\Rightarrow$) Suppose that $\langle TP_1, TP_2, w_1, w_2\rangle$ is a 'yes'-instance and assume by contradiction that E is not a MinEX for $D \models (Q, \Sigma)$. There are two cases, or E is not minimal or $E \not\models (Q, \Sigma)$.

For the former case, since the presence of the facts of D^1 and D^2 is required in E in order to satisfy Q, if E is not minimal, then it must be the case that the fact $\mathit{tiling}^1()$ can be removed from E and still have $E \models (Q, \Sigma)$. However, this implies that the fact $\mathit{tiling}^1()$ can be derived via the mediation Σ^1, which means that TP_1 has a solution with w_1: a contradiction. For the latter case, since the facts of D_1, of D_2, and $\mathit{tiling}^1()$, are in E, if $E \not\models (Q, \Sigma)$, then it must be the case that the fact $\mathit{tiling}^2()$ cannot be derived via the mediation of Σ^2, which is a contradiction. Thus, E is a MinEX for $D \models (Q, \Sigma)$.

($\Leftarrow$) Suppose that E is a MinEX for $D \models (Q, \Sigma)$. Since $E \models (Q, \Sigma)$, the fact $\mathit{tiling}^2()$ can be derived via the mediation of Σ^2, which implies that TP_2 has a solution with w_2. Moreover, since E is minimal and no fact of D_1 or D_2 can be removed from

E without losing the property of entailing the OMQ, the fact $\mathit{tiling}^1()$ is required to be in E to entail the OMQ, which implies that $\mathit{tiling}^1()$ cannot be derived via the mediation of Σ^1 and hence TP_1 has no solution with w_1. Thus, $\langle TP_1, TP_2, w_1, w_2\rangle$ is a 'yes'-instance. □

The remaining hardness results in Table 3.1 follow from the observation that Is-MinEX is no easier than OMQA in *ba*-combined and combined complexity. In particular, in *ba*-combined and combined complexity we can encode the database in the program in polynomial time. This implies that OMQA can be reduced to Is-MinEX in polynomial time.

Theorem 3.14. *For any class $\mathcal{L}$ of TGDs studied here,* Is-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *in ba-combined (resp. combined) complexity is as hard as* OMQA$(\mathrm{BCQ}, \mathcal{L})$ *in ba-combined (resp. combined) complexity.*

Proof. Let D be a database and let (Q, Σ) be an OMQ, constituting an instance of OMQA$(\mathrm{BCQ}, \mathcal{L})$. We construct an instance of Is-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L})$. We take D' to be $f()$ together with the query Q grounded with fresh constants. Further, we take $\Sigma' = \Sigma \cup \{f() \to p(\mathbf{a}) \mid p(\mathbf{a}) \in D\}$ and $E = \{f()\}$. Notice that $D' \models (Q, \Sigma')$ since D' contains a grounding of Q. It is easy to see that $D \models (Q, \Sigma)$ if and only if E is a MinEX for $D' \models (Q, \Sigma')$. This follows because if $f()$ is present, every fact in D is inferred via Σ'. Notice that the result holds for all classes of TGDs considered here, as the rules added to the program are linear, acyclic and full. □

3.4 All Minimal Explanations

In this section, we analyse the problem of deciding whether a given set of subsets of a database is the set of all MinEXs: Given a database D, an OMQ (Q, Σ) with $D \models (Q, \Sigma)$, and a set $\mathcal{E}$ of subsets of D, is $\mathcal{E}$ *exactly* the set of all MinEXs for $D \models (Q, \Sigma)$? This is a natural decision version of the problem of constructing the set of all MinEXs.

As in the section on Is-MinEX, we show the membership results for All-MinEX for UCQs and then proceed to give the hardness results for All-MinEX for BCQs. Thus we obtain the completeness results for All-MinEX both for BCQs and UCQs, presented in Table 3.2. For the cases indicated with "$\leq$", we give just membership results. Otherwise, we provide both membership and hardness results. We note that there are significant differences between All-MinEX and Is-MinEX. First of all, in All-MinEX we need to determine whether *all* elements in a given set of subsets

Table 3.2: Complexity results for All-MinEX($\subseteq, \mathcal{Q}, \mathcal{L}$) with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

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

of D are MinEXs, while in Is-MinEX, we need to determine whether a *single* given subset is a MinEX. Furthermore, in All-MinEX, we need to ensure that there is *no* MinEX outside of the given set of subsets; we do not need to check this kind of condition in Is-MinEX.

As before, we start with a general membership result for All-MinEX($\subseteq, \text{UCQ}, \mathcal{L}$) by providing a general approach to solve this problem.

Theorem 3.15. *For a class $\mathcal{L}$ of TGDs, if* OMQA($\text{UCQ}, \mathcal{L}$) *is in the complexity class $\mathcal{C}$, then* All-MinEX($\subseteq, \text{UCQ}, \mathcal{L}$) *can be decided by a linear number of checks in $\mathcal{C}$ and a single check in co-*($\text{NP}^{\mathcal{C}}$).

Proof. Let D be a database and (Q, Σ) be an $\mathcal{L}$-OMQ. Let $\mathcal{E} \subseteq \mathcal{P}(D)$ be a candidate set of all MinEXs. The set $\mathcal{E}$ is exactly the set of *all* MinEXs for $D \models (Q, \Sigma)$ if and only if (1) all sets $E \in \mathcal{E}$ are MinEXs for $D \models (Q, \Sigma)$ and (2) there is no MinEX missing from $\mathcal{E}$. We can further observe that the condition (1) holds if and only if (1a) all sets $E \in \mathcal{E}$ entail (Q, Σ); and (1b) all sets $E \in \mathcal{E}$ are subset-minimal w.r.t. the entailment of (Q, Σ).

Consider the condition (1a). We can check (1a) using $|E|$ (i.e., a linear number of) entailment checks, each of them in $\mathcal{C}$.

Consider now conditions (1b) and (2). We argue that these conditions hold if and only if there is *no* subset $E' \subseteq D$, which *entails* the OMQ (Q, Σ) and does not contain as a subset any $E \in \mathcal{E}$. Indeed, clearly, if one of the conditions (1b) or (2) does not hold, then either some $E \in \mathcal{E}$ is not minimal, which implies that there is $E' \subsetneq E$ that entails the OMQ, or there is a MinEX E' not in $\mathcal{E}$. In either of these cases, we have that such E' exists. For the converse, assume that there exists such a set E'. Since E' does not contain any set $E \in \mathcal{E}$ as a subset, there are two cases: either E' is a strict subset of a set $E \in \mathcal{E}$; or E' does not contain any set $E \in \mathcal{E}$. In the first case, since $E' \subsetneq E$, E is not minimal w.r.t. the entailment of (Q, Σ), and

hence condition (1b) does not hold. In the second case, E' is a superset of a MinEX that is not included in $\mathcal{E}$, and hence condition (2) does not hold.

Note that we can verify the existence of such E' by guessing a subset $E' \subseteq D$ (in NP) and then checking (in the complexity class $\mathcal{C}$) that $E' \models (Q, \Sigma)$, hence this is feasible in $\mathrm{NP}^{\mathcal{C}}$. Therefore, checking whether the conditions (1b) and (2) hold is feasible in co-($\mathrm{NP}^{\mathcal{C}}$). □

Notice that Theorem 3.15 gives tight upper bounds for all results in Table 3.2, apart from the membership results of All-MinEX in P, D^{P} and $\mathrm{D}^{\mathrm{Exp}}$. For example, if OMQA is in P, then All-MinEX can be decided in coNP as $\mathrm{P}^{\mathrm{P}} = \mathrm{P}$, co-($\mathrm{NP}^{\mathrm{P}}$) = coNP, and an coNP machine can carry out an extra P test. Other results follow similarly. If $\mathcal{C} \in \{\mathrm{PSpace}, \mathrm{Exp}, 2\mathrm{Exp}\}$, then $\mathrm{P}^{\mathcal{C}} = \mathcal{C}$ and co-($\mathrm{NP}^{\mathcal{C}}$) = co-$\mathcal{C} = \mathcal{C}$. So in these cases, All-MinEX is in $\mathcal{C}$.

We show that whenever OMQA is in NP, the problem All-MinEX is in D^{P}.

Theorem 3.16. *For any class $\mathcal{L}$ of TGDs, if* OMQA(UCQ, $\mathcal{L}$) *is in* NP*, then* All-MinEX($\subseteq$, UCQ, $\mathcal{L}$) *is in* D^{P}.

Proof. Consider the proof of Theorem 3.15. The condition (1a) can be checked with a linear number of NP tests, that can be combined in a single NP test. We claim that checking that the conditions (1b) and (2) hold is feasible in coNP. Indeed, checking that condition (1b) or (2) do *not* hold require the guess of set $E' \subseteq D$ that is not a superset of any explanation in $\mathcal{E}$ and entails (Q, Σ) (see the proof of Theorem 3.15). Note that we can combine the certificate of a guess of E' and the certificate for checking entailment, and thus checking the existence of such E' is in NP. Therefore, checking the conditions (1b) and (2) is in coNP. □

Similarly, we show that whenever OMQA is in NExp, we have that All-MinEX is in $\mathrm{D}^{\mathrm{Exp}}$. Intuitively, checking whether each set in the candidate set $\mathcal{E}$ of all MinEXs entails the OMQ is in NExp, and checking whether every set in the candidate set of all MinEXs is minimal and that there is no MinEX outside of the candidate set of all MinEXs is in coNExp.

Theorem 3.17. *For any class $\mathcal{L}$ of TGDs, if* OMQA(UCQ, $\mathcal{L}$) *is in* NExp*, then* All-MinEX($\subseteq$, UCQ, $\mathcal{L}$) *is in* $\mathrm{D}^{\mathrm{Exp}}$.

Proof. We build on the proof of Theorem 3.15. The condition (1a) can be checked with a linear number of NExp tests, that can be combined in a single NExp test. We note when checking that the conditions (1b) and (2) do *not* hold can be done

in NExp, as the initial NP guess can be incorporated in the NExp machine. Thus the conditions (1b) and (2) can be ensured in coNExp. Therefore, in this case, All-MinEX is in $\mathrm{D}^{\mathrm{Exp}}$. □

When the program is from a class of TGDs for which OMQA is FO-rewritable, the set of all MinEXs can be computed in polynomial time in data complexity, as shown next.

Theorem 3.18. *Let $\mathcal{L}$ be a class of TGDs for which* OMQA *is FO-rewritable and (Q, Σ) be an $\mathcal{L}$-OMQ. Then computing all MinEXs for $D \models (Q, \Sigma)$ is feasible in polynomial time in data complexity.*

Proof. Let D be a database and let (Q, Σ) be an $\mathcal{L}$-OMQ with a UCQ Q. Since $\mathcal{L}$ is class of TGDs for which OMQA is FO-rewritable, there is a UCQ Q_Σ such that, for every $D' \subseteq D$, $D' \models (Q, \Sigma)$ if and only if $D' \models Q_\Sigma$. We assume w.l.o.g. that

$$Q_\Sigma = Q_1^\Sigma \vee \cdots \vee Q_m^\Sigma,$$

where $Q_i^\Sigma = \exists X_1^i \ldots \exists X_{k_i}^i \, \phi_i^\Sigma(X_1^i, \ldots, X_{k_i}^i)$ and ϕ_i^Σ is a conjunction of atoms. We recall that the active domain of D, $dom(D)$, is the set of constants that appear in D. In the rest of the proof, given $\mathbf{a} \in dom(D)^{k_i}$, we denote by $\{\phi_i^\Sigma(\mathbf{a})\}$ the set of the facts that are the conjuncts in the formula $\phi_i^\Sigma(\mathbf{a})$.

Let $\mathcal{D}_i^\Sigma$ be the set containing all the sets of facts that can be obtained by grounding $\phi_i^\Sigma(X_1^i, \ldots, X_{k_i}^i)$ with the constants in the active domain, i.e., $\mathcal{D}_i^\Sigma = \big\{\{\phi_i^\Sigma(\mathbf{a})\} \mid \mathbf{a} \in dom(D)^{k_i}\big\}$. Note that $\mathcal{D}_i^\Sigma$ is a set of sets of facts. Notice that k_i is a constant and the active domain, $dom(D)$, is of polynomial size in the size of the database D. Therefore, it takes polynomial time to compute $\mathcal{D}_i^\Sigma$. Furthermore, since there is a constant number m of disjuncts in Q_Σ, it takes polynomial time to compute $\mathcal{D}^\Sigma = \cup_{i=1}^m \mathcal{D}_i^\Sigma$.

Let $\mathcal{E}'$ be the set of subset-minimal elements of $\mathcal{D}^\Sigma$, that is, $\mathcal{E}' = \{E \in \mathcal{D}^\Sigma \mid \nexists G \in \mathcal{D}^\Sigma \text{ s.t. } G \subsetneq E\}$. This set can be computed in polynomial time in the size of $\mathcal{D}^\Sigma$, and so in the size of D (computable as well in polynomial time). Let $\mathcal{E} = \{E \in \mathcal{E}' \mid E \subseteq D\}$ be the set of the sets in $\mathcal{E}'$ that are subsets of D.

We show that $\mathcal{E}$ is exactly the set of *all* MinEXs for $D \models (Q, \Sigma)$. Let E be a MinEX for $D \models (Q, \Sigma)$. We show that $E \in \mathcal{E}$. We have that $E \models Q_\Sigma$, and so $E \models Q_i^\Sigma$ for some i. Notice that $E_i^\Sigma \subseteq E$ for some $E_i^\Sigma \in \mathcal{D}_i^\Sigma$ as $\mathcal{D}_i^\Sigma$ contains all the subset-minimal sets of facts that support the entailment of (Q, Σ). Since E is also subset-minimal, we have that $E \in \mathcal{E}$. Therefore, we have that all MinEXs for

$D \models (Q, \Sigma)$ are in $\mathcal{E}$. Also, it is clear from the construction that every set in E is a MinEX for $D \models (Q, \Sigma)$. □

In the rest of this section, we provide the matching lower bounds for the membership results provided above. As before, we show hardness for BCQs. First, we show that the problem All-MinEX$(\subseteq, \text{BCQ}, \mathsf{GF})$ is coNP-hard in data complexity by a reduction from Unsat. We note that this proof is inspired by the constructions provided by Lukasiewicz et al. [115].

Theorem 3.19. All-MinEX$(\subseteq, \text{BCQ}, \mathsf{GF})$ *is* coNP*-hard in data complexity.*

Proof. We exhibit a reduction from the coNP-complete problem Unsat. Let $\phi(X)$ be a 3CNF formula over variables $X = \{x_1, \ldots, x_n\}$ and clauses $C = \{c_1, \ldots, c_m\}$, which is an instance of Unsat. We further assume w.l.o.g. that $\phi(X)$ has at least two clauses that do not share any literals. From $\phi(X)$, we build the database D. Also, we construct the OMQ (Q, Σ), which is independent of $\phi(X)$. For each variable $x_i \in X$, we add to D the facts $val(x_i, 0)$ and $val(x_i, 1)$, where x_i is a constant representing the respective variable in ϕ, and 0 and 1 are constants representing Boolean values *false* and *true*, respectively. For each clause c_j, we add to the database D:

- a fact $succ\_cl(j-1, j)$, which captures that the j^{th} clause is the successor of the $(j-1)^{\text{th}}$ clause; and
- a fact that encodes c_j. For example, for the j^{th} clause $c_j = (x_p \vee x_q \vee \neg x_r)$, there is the fact $cl(j, x_p, p, x_q, p, x_r, n)$ in D, where p and n are constants representing whether a variable appears as a positive or a negative literal, respectively.

In addition, the database contains the fact $satchain(0)$ that acts as an initial condition in the satisfiability check, and the fact $maxcl(m)$ that captures the number of clauses in the formula.

The program Σ consists of the following rules. The first rule in the program states that if some variable is assigned both *false* and *true* values, then $sat()$ holds:

$$val(X, 0) \wedge val(X, 1) \rightarrow sat().$$

The program also contains the rules that capture different ways of satisfying a clause:

$$cl(J, X, p, \_, \_, \_, \_) \wedge val(X, 1) \rightarrow satcl(J),$$
$$cl(J, X, n, \_, \_, \_, \_) \wedge val(X, 0) \rightarrow satcl(J),$$

$$cl(J, \_, \_, Y, p, \_, \_) \wedge val(Y, 1) \rightarrow satcl(J),$$
$$cl(J, \_, \_, Y, n, \_, \_) \wedge val(Y, 0) \rightarrow satcl(J),$$
$$cl(J, \_, \_, \_, \_, Z, p) \wedge val(Z, 1) \rightarrow satcl(J),$$
$$cl(J, \_, \_, \_, \_, Z, n) \wedge val(Z, 0) \rightarrow satcl(J).$$

The last two rules in the program capture that $sat()$ is entailed if the val facts contained give a satisfying assignment to the formula ϕ:

$$satchain(I) \wedge succ\_cl(I, J) \wedge satcl(J) \rightarrow satchain(J),$$
$$maxcl(M) \wedge satchain(M) \rightarrow sat().$$

Note that there are two ways how $sat()$ might be entailed. In the first case, both $val(X, 0)$ and $val(X, 1)$ facts are present for some variable X, and, in the second case, val facts give a satisfying assignment for ϕ. We take the query to be

$$Q = sat().$$

Observe that the program and the query do not depend on $\phi(X)$, and that $D \models (Q, \Sigma)$. Also, the program Σ is guarded and full. To conclude, we take a candidate set of all MinEXs to be:

$$\mathcal{E} = \{\{val(x_i, 0), val(x_i, 1)\} \,|x_i \in X\}\,.$$

Given the first rule of Σ, each set in $\mathcal{E}$ is a MinEX.[2] Therefore, in order to show that the reduction is correct, we simply need to prove that ϕ is unsatisfiable if and only if there is no MinEX outside $\mathcal{E}$.

($\Rightarrow$) Suppose that there is a MinEX outside of $\mathcal{E}$. Let E be such a MinEX. First, note that $\{val(x_i, 0), val(x_i, 1)\} \not\subseteq E$, as otherwise E would not be minimal. Therefore, E encodes a proper (not necessarily complete) truth assignment for X. This means that the first rule in the program cannot be triggered. The remaining rules encode the satisfiability of ϕ. Since $E \models (Q, \Sigma)$, we have that E encodes a satisfying assignment for ϕ, and thus ϕ is satisfiable.

($\Leftarrow$) Suppose that ϕ is satisfiable and let v be a satisfying assignment to X. Then take $E = \{val(x_i, v(x_i))\}$. By construction, $E \models (Q, \Sigma)$ and does not contain any set in $\mathcal{E}$. Therefore, E must contain a MinEX for $D \models (Q, \Sigma)$ outside of $\mathcal{E}$.

First, we claim that any set of facts E such that $\{val(x_i, 0), val(x_i, 1)\} \subseteq E$ is not a MinEX outside $\mathcal{E}$. Indeed, if $\{val(x_i, 0), val(x_i, 1)\} \subsetneq E$, then E is not a MinEX because it is not minimal, as $\{val(x_i, 0), val(x_i, 1)\}$ entails the OMQ. □

[2]Since we assumed that $\phi(X)$ has at least two clauses that do not share any literal, neither $\{val(x_i, 0)\}$ nor $\{val(x_i, 1)\}$, for any x_i, can entail the OMQ.

Table 3.3: Complexity results for the problems MinEX-Irrel$(\subseteq, \mathcal{Q}, \mathcal{L})$ and Small-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**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

This implies that All-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$ is coNP-hard in data complexity for the languages $\mathcal{L} \in \{\mathsf{G}, \mathsf{F}, \mathsf{WS}, \mathsf{WA}\}$, since GF is contained in each of $\mathsf{G}, \mathsf{F}, \mathsf{WS}$ and WA.

The remaining hardness results for All-MinEX in Table 3.2 are obtained by minimal variations of the proofs for Is-MinEX. In the constructions in the proofs of Theorems 3.12, 3.13 and 3.14, we have constructed a subset E of the database that we have shown to be a MinEX. It can be easily seen that this constructed subset is the *only* MinEX in all these constructions. Therefore, we have that all these arguments apply to All-MinEX when we take $\mathcal{E} = \{E\}$ as it is the set of *all* MinEXs. So these theorems restated with Is-MinEX replaced by All-MinEX hold.

3.5 Irrelevant Facts for Explanations

In this section, we analyse the complexity of MinEX-Irrel, the problem of deciding whether there is a minimal explanation that avoids a collection of forbidden sets of facts: Given a database D, an OMQ (Q, Σ) with $D \models (Q, \Sigma)$, and a set $\mathcal{F} \subseteq \mathcal{P}(D)$ of subsets of D, is there a MinEX for $D \models (Q, \Sigma)$ that does not contain as a subset any $F \in \mathcal{F}$?

The motivation for studying this problem is similar to that of Peñaloza and Sertkaya [135]. For example, if $\mathcal{F}$ is the set of all explanations that we are aware of, then this problem checks whether there is any MinEX outside of this set $\mathcal{F}$.

Following the structure of the previous two sections, we first provide membership results for UCQs and then proceed to give hardness results when the query language is taken to be BCQ. We first provide a general algorithm that covers most of the membership results presented in Table 3.3.

Theorem 3.20. *For a class $\mathcal{L}$ of TGDs, if* OMQA$(\text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* MinEX-Irrel$(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* $\text{NP}^{\mathcal{C}}$.

Proof. Let D be a database, (Q, Σ) be an ontology-mediated query, and $\mathcal{F} \subseteq \mathcal{P}(D)$ be a set of forbidden sets. To check whether there is a MinEX E for $D \models (Q, \Sigma)$ that does not contain any set in $\mathcal{F}$, it is enough to check whether there is *any* subset $E' \subseteq D$ that entails the OMQ (Q, Σ) and does not contain any of the forbidden sets in $\mathcal{F}$. Notice that we do not need to check the minimality of E'. Indeed, if such a set E' exists, then it contains a MinEX E for $D \models (Q, \Sigma)$, which does not contain any of the forbidden sets in $\mathcal{F}$. We can check the existence of such E' by guessing it in NP, and then checking whether it entails the OMQ via an oracle call in $\mathcal{C}$. □

The above result gives all the membership results in Table 3.3, apart from the membership results in P, NP and NExp. We illustrate how the above theorem implies the membership results in Table 3.3. If $\mathcal{C} = \text{P}$, then MinEX-Irrel can be decided in NP as the NP machine can express a P test. If $\mathcal{C} = \text{Exp}$, then, since $\text{NP}^{\text{Exp}} = \text{Exp}$, we have that MinEX-Irrel in Exp. The PSpace and 2Exp membership results follow similarly. Among the remaining membership results, we show the NP membership results in Table 3.3.

Theorem 3.21. *For any class $\mathcal{L}$ of TGDs, if* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in* NP*, then the problem* $\text{MinEX-Irrel}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* NP*.*

Proof. If OMQA is in NP, by inspecting the proof of Theorem 3.20, MinEX-Irrel can be solved by guessing in NP a subset that does not contain any of the forbidden sets, and then by checking with another NP test whether it entails the OMQ. Note that the two certificates for the NP tests can be combined, and thus the whole task in NP. □

Now we show the membership results in NExp.

Theorem 3.22. *For any class $\mathcal{L}$ of TGDs, if* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in* NExp*, then the problem* $\text{MinEX-Irrel}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* NExp*.*

Proof. If OMQA is in NExp, by inspecting the proof of Theorem 3.20, we see that MinEX-Irrel can be solved by guessing in NP a subset that does not contain any of the forbidden sets and checking entailment in NExp. Note that the guess of the subset (which is in NP) can be expressed in an NExp machine, and thus the whole task in NExp. □

The only remaining membership results in Table 3.3 are the membership of MinEX-Irrel in P in data complexity for the classes of TGDs for which OMQA is FO-rewritable. This is a straightforward consequence of Theorem 3.18.

Corollary 3.23. *Let $\mathcal{L}$ be a class of TGDs for which* OMQA *is FO-rewritable and (Q, Σ) be an $\mathcal{L}$-OMQ. Then* MinEX-Irrel$(\subseteq, \mathrm{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

Proof. By Theorem 3.18, it is possible to compute the set $\mathcal{E}$ of all MinEXs in polynomial time. Then we can find whether there is a MinEX that does not contain any of the forbidden sets in polynomial time by going through all the MinEXs in $\mathcal{E}$. Identifying whether there is a required MinEX takes polynomial time. □

It only remains to show the matching lower bounds for MinEX-Irrel. Our next result shows that MinEX-Irrel$(\subseteq, \mathrm{BCQ}, \mathsf{GF})$ is NP-hard in data complexity. This proof adapts the ideas, presented in the proof of [135, Theorem 5]. In particular, the guarded full language allows us to express reachability, and thus we can express an NP-complete graph problem.

Theorem 3.24. MinEX-Irrel$(\subseteq, \mathrm{BCQ}, \mathsf{GF})$ *is* NP*-hard in data complexity.*

Proof. We reduce to MinEX-Irrel$(\subseteq, \mathrm{BCQ}, \mathsf{GF})$ the following NP-complete problem:

Problem: Path with Forbidden Edge Pairs [79] [81, Problem GT54]
Input: A directed acyclic graph $G = (V_G, E_G)$, two vertices s, $t \in V_G$, and a set $\mathcal{F}_{E_G} \subseteq E_G \times E_G$ of pairs of edges.
Question: Is there a path P from s to t in G such that, for every pair of edges $(e, e') \in \mathcal{F}_{E_G}$, $\{e, e'\} \not\subseteq P$ (i.e., the path does not pass through both of the edges in the pair)?

We express this problem as MinEX-Irrel$(\subseteq, \mathrm{BCQ}, \mathsf{GF})$ as follows. We construct the database to encode the graph, the starting node s, an additional edge from t to τ (in order to have a fixed query), and an additional (dummy) fact (in order for the entire D to always entail the OMQ):

$$D = \{r(u, v) \mid (u, v) \in E_G\} \cup \{p(s)\} \cup \{r(t, \tau)\} \cup \{p(\tau)\}.$$

The program captures what is reachable by a path from a certain node. It encodes that, if there is a path from s to some node X and there is an edge $(X, Y) \in E_G$, then there is a path from s to Y:

$$\Sigma = \{p(X) \wedge r(X, Y) \rightarrow p(Y)\}.$$

The query asks whether there is a path from s to τ (notice that the query is fixed, and that $D \models (Q, \Sigma)$ because $p(\tau) \in D$):

$$Q = p(\tau).$$

The forbidden sets are the pairs of edges in $\mathcal{F}$ plus the dummy fact:

$$\mathcal{F} = \{\{r(u_1, u_2), r(v_1, v_2)\} \mid ((u_1, u_2), (v_1, v_2)) \in \mathcal{F}_{E_G}\} \cup \{p(\tau)\}.$$

We prove that there there is a simple path in G from s to t not using any pair of edges in $\mathcal{F}_{E_G}$ if and only if there is a MinEX, not containing any set in $\mathcal{F}$.

($\Rightarrow$) Suppose that there is a path from s to t that does not contain any of the forbidden pairs in $\mathcal{F}_{E_G}$. Let P be such a path, and let $E = \{r(u, v) \mid (u, v) \in P\} \cup \{p(s)\} \cup \{r(t, \tau)\}$. Notice that E entails the OMQ, because P connects s to t and $r(t, \tau)$ is in E. Furthermore, E is minimal, as otherwise some edge could be removed from P, but P is a path and hence removal of any edge would break the connectivity. Finally, since P does not contain any of the forbidden pairs in $\mathcal{F}_{E_G}$, by construction, E does not contain any of the sets in $\mathcal{F}$.

($\Leftarrow$) Suppose that there is a MinEX E for $D \models (Q, \Sigma)$ not containing any set in $\mathcal{F}$. Let $P = \{(u, v) \in E_G \mid r(u, v) \in E\}$. Since $E \models (Q, \Sigma)$ and $p(\tau) \notin E$, we have by construction that P connects s to t. Furthermore, as E is minimal for $D \models (Q, \Sigma)$, we have that P must be a path, otherwise, E would not be minimal. □

The remaining hardness results of MinEX-Irrel$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ in *fp*-combined, *ba*-combined, and combined complexity are obtained by reducing OMQA to MinEX-Irrel.

Theorem 3.25. *For any class $\mathcal{L}$ of TGDs,* MinEX-Irrel$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *in fp-combined (resp., ba- and combined) complexity is as hard as* $\mathrm{OMQA}(\mathrm{BCQ}, \mathcal{L})$ *in fp-combined (resp., ba- and combined) complexity.*

Proof. We can reduce OMQA to MinEX-Irrel as follows. Let a database D and an OMQ (Q, Σ) be an instance of $\mathrm{OMQA}(\mathrm{BCQ}, \mathcal{L})$. Consider the database $D' = D \cup D_Q$, where D_Q is obtained by grounding the query Q with fresh constants. We take the program and the query as in the input. First, observe that $D' \models (Q, \Sigma)$ since $D_Q \models (Q, \Sigma)$ by construction. Then, we take $\mathcal{F} = \{\{f\} \mid f \in D_Q\}$. It is not hard to see that $D \models (Q, \Sigma)$ if and only if there is a MinEX for $D' \models (Q, \Sigma)$ not containing any sets in $\mathcal{F}$. Notice that the result holds for all classes of TGDs, as we do not change the program in the reduction. □

This completes our analysis of MinEX-Irrel and covers all the complexity results in Table 3.3.

Table 3.4: Complexity results for the problems MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$ and Large-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**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}	P^{NExp}	P^{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

3.6 Relevant Facts for Explanations

We provide a complexity analysis for the problem MinEX-Rel of deciding whether a distinguished fact is relevant for an OMQ entailment, which is summarised in Table 3.4: Given a database D, an OMQ (Q, Σ) with $D \models (Q, \Sigma)$, and a fact $\psi \in D$, asks whether there is a MinEX for $D \models (Q, \Sigma)$ that contains ψ.E As before, we start by providing the membership results for UCQs, and then show the hardness results for BCQs.

We give a general approach to solve MinEX-Rel. 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$(\subseteq, \text{UCQ}, \mathcal{L})$ to check whether E is a MinEX. This yields the following result.

Theorem 3.26. *For any class $\mathcal{L}$ of TGDs, if* Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$ *is in $\mathcal{C}$, then the problem* MinEX-Rel$(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* $\text{NP}^{\mathcal{C}}$.

Proof. Let D be a database, let (Q, Σ) be an $\mathcal{L}$-OMQ, and let $\psi \in D$ be a fact in the database. We can check whether there is a MinEX containing ψ by guessing a subset of the database $E \subseteq D$ that contains the distinguished fact ψ (in NP), and then checking whether E is a MinEX for $D \models (Q, \Sigma)$ by calling an oracle for Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$. □

This theorem covers all the membership results for MinEX-Rel, given in Table 3.4, apart from the membership results in P in data complexity when OMQA is FO-rewritable. For example, when Is-MinEX is in P, we have that MinEX-Rel is in $\text{NP}^{\text{P}} = \text{NP}$. When Is-MinEX is in D^{P}, we have that MinEX-Rel is in Σ_2^{P} as $\text{NP}^{\text{D}^{\text{P}}} = \Sigma_2^{\text{P}}$. If $\mathcal{C} \in \{\text{PSpace}, \text{Exp}, \text{2Exp}\}$, we have that $\text{NP}^{\mathcal{C}} = \mathcal{C}$. The other results follow similarly.

The membership of MinEX-Rel in P in data complexity for the classes of TGDs for which OMQA is FO-rewritable follows easily from Theorem 3.18. That result states that we can compute the set of all MinEXs in polynomial time in the data complexity for the classes of TGDs for which OMQA is FO-rewritable.

Corollary 3.27. *Let $\mathcal{L}$ be a class of TGDs for which* OMQA *is FO-rewritable and (Q, Σ) be an $\mathcal{L}$-OMQ. Then* MinEX-Rel$(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

Proof. By Theorem 3.18, we can compute the set $\mathcal{E}$ of all MinEXs in polynomial time in data complexity. Once we have the set $\mathcal{E}$, we can check in polynomial time whether there is a MinEX that contains the distinguished fact ψ. □

As in the previous sections, we provide matching lower bounds. First, we show that MinEX-Rel$(\subseteq, \text{BCQ}, \mathsf{GF})$ is NP-hard in data complexity. This proof adapts the ideas, presented in the proof of [135, Theorem 7]. The fundamental idea for this reduction is similar to the NP-hardness proof of MinEX-Irrel$(\subseteq, \text{BCQ}, \mathsf{GF})$ in data complexity, where we encode graph reachability.

Theorem 3.28. MinEX-Rel$(\subseteq, \text{BCQ}, \mathsf{GF})$ *is* NP*-hard in data complexity.*

Proof. We reduce to MinEX-Rel$(\subseteq, \text{BCQ}, \mathsf{GF})$ the following NP-complete problem:

Problem: Path via Node [111]
Input: A directed graph $G = (V_G, E_G)$ and three vertices $s, t, m \in V_G$.
Question: Is there a simple path from s to t in G that passes through m?

Let $G = (V_G, E_G)$ be a graph with three vertices $s, t, m \in V_G$, which is an instance of Path via Node. Let $G^m = (V_G^m, E_G^m)$ be a graph obtained from G by replacing m with two fresh nodes m_1 and m_2, introducing an edge (m_1, m_2), replacing all incoming edges to m with incoming edges to m_1, and replacing all outgoing edges from m with outgoing edges from m_2. In particular, we have that

$$\begin{aligned} V_G^m &= (V_G \setminus \{m\}) \cup \{m_1, m_2\}, \text{ and} \\ E_G^m &= \{(u, v) \mid (u, v) \in E_G \wedge u, v \neq m\} \cup \{(u, m_1) \mid (u, m) \in E_G \wedge u \neq m\} \cup \\ &\quad \{(m_2, v) \mid (m, v) \in E_G \wedge m \neq v\} \cup \{(m_1, m_2)\}. \end{aligned}$$

By construction, there is a simple path from s to t in G through m if and only if there is a simple path from s to t in G^m through the edge (m_1, m_2). Thus, in the rest of this proof we focus on G^m.

We construct an instance of MinEX-Rel as follows. The database encodes the graph G^m and introduces a new node τ, which is connected by an edge to t. We

introduce τ to ensure that the query is independent of the database. So the database is

$$D = \{r(u, v) \mid (u, v) \in E_G^m\} \cup \{p(s), p(\tau), r(t, \tau)\}.$$

The program expresses reachability: if a vertex X is reachable, and there is an edge (X, Y), then a vertex Y is reachable too:

$$\Sigma = \{p(X) \wedge r(X, Y) \rightarrow p(Y)\}.$$

The query asks whether there is a path from s to τ. Notice that the query is fixed, and that $D \models (Q, \Sigma)$ because $p(\tau) \in D$. So we take it to be

$$Q = p(\tau).$$

Finally, we pick $\psi = r(m_1, m_2)$ as the distinguished fact, and show that there is a simple path in G^m from s to t through the edge (m_1, m_2) if and only if there is a MinEX for $D \models (Q, \Sigma)$ containing the distinguished fact $\psi = r(m_1, m_2)$.

($\Rightarrow$) Suppose that there is a simple path P in G^m from s to t through (m_1, m_2). Let $E = \{r(u, v) \mid (u, v) \in P\} \cup \{p(s)\} \cup \{r(t, \tau)\}$. Notice that E entails the OMQ because P connects s to t and $r(t, \tau)$ is in E. Moreover, E is minimal, otherwise, P would not be a simple path (that is, some edge could be removed from P). Finally, observe that $\psi = r(m_1, m_2)$ is in E.

($\Leftarrow$) Suppose that there is a MinEX E for $D \models (Q, \Sigma)$ that contains the distinguished fact $\psi = r(m_1, m_2)$. Observe that such a MinEX E does not contain $p(\tau)$, otherwise, E would not be minimal (ψ could be removed). Take $P = \{(u, v) \mid r(u, v) \in E\}$, which is a subset of E_G^m. Then, P is a path from s to t in G^m that goes through (m_1, m_2). First, notice that P must connect s to t in G^m, as otherwise E would not entail the OMQ (remember that $p(\tau)$ does not belong to E). Second, P must be a path, as otherwise, E would *not* be minimal. Finally, P goes through (m_1, m_2) as $\psi = r(m_1, m_2)$ is in E. □

Next, we show that MinEX-Rel$(\subseteq, \text{BCQ}, \mathcal{L})$ is Σ_2^{p}-hard in *fp*-combined complexity. In this reduction, we reduce a Σ_2^{p}-complete satisfiability problem to MinEX-Rel by encoding a formula in the database and the satisfiability in the query.

Theorem 3.29. MinEX-Rel$(\subseteq, \text{BCQ}, \mathcal{L}_\emptyset)$ *is* Σ_2^{p}*-hard in fp-combined complexity.*

Proof. We reduce to MinEX-Rel$(\subseteq, \text{BCQ}, \mathcal{L})$ the following Σ_2^{p}-complete problem:
Problem: $\text{QBF}_{2,\forall,\neg}^{CNF}$
Input: A quantified Boolean formula $\Phi = \exists X \forall Y \neg\phi(X, Y)$, with X and Y disjoint

sets of Boolean variables, and ϕ a (non-quantified) 3CNF formula.
Question: Is Φ valid?

Let $\Phi = \exists X \forall Y \, \neg\phi(X, Y)$ be an instance of $\mathrm{QBF}^{CNF}_{2,\forall,\neg}$. We construct the database D, the OMQ (Q, Σ), and the fact ψ as follows. For each variable $x_i \in X$, add the following facts to the database D:

$$val(x_i, 0), \qquad\qquad val(x_i, 1),$$

where x_i is a constant representing the respective variable in Φ, and 0 and 1 are constants representing Boolean values *false* and *true*, respectively. We add to D a number of facts to ensure the consistency of the truth assignments to the literals:

$$\begin{array}{llll} simlit(0,0), & simlit(0,2), & opplit(0,1), & simlit(2,2), \\ simlit(1,1), & simlit(1,2), & opplit(1,0), & opplit(2,2), \end{array}$$

where 0 and 1 are constants with the same meaning as in Theorem 3.12, 2 is an additional constants used as a "jolly" (intuitively, it will be used to bypass, in some circumstances, the check of the satisfaction of $\phi(X, Y)$).

Also, we add the facts that encode the satisfying assignments to the literals for any clause:

$$\begin{array}{llll} clsat(1,1,1), & clsat(1,1,0), & clsat(1,0,1), & clsat(1,0,0), \\ clsat(0,1,1), & clsat(0,1,0), & clsat(0,0,1). & \end{array}$$

In D there is also the fact $clsat(2,2,2)$, which, intuitively, will be used in some circumstances, to bypass the check of the satisfaction of $\phi(X, Y)$. We define the set of *structural* facts D_{st} to be $D \setminus \{clsat(2,2,2)\}$, that is, all the facts in D apart from $clsat(2,2,2)$.

We take the program Σ to be empty. The query Q is constituted by the following pieces. The first piece of the query checks whether all the structural facts are present:

$$config \equiv \bigwedge_{p(\mathbf{c}) \in D^{St}} p(\mathbf{c}).$$

The second piece of the query "reads" the assignment for the variables in X encoded in the MinEX and imposes that at least one fact between $val(x_i, 0)$ and $val(x_i, 1)$ must be present for each variable $x_i \in X$:

$$assignX \equiv \bigwedge_{i=1}^{n} val(x_i, T_i).$$

The third piece of the query "copies" the assignment to each variable x_i onto one occurrence of x_i as a positive literal in the formula $\phi(X, Y)$. We write $\ell_{j,k}$ for the k^{th} literal in the j^{th} clause in the formula ϕ. We take this part of the query to be

$$copy \equiv \bigwedge_{i=1}^{n} simlit(T_i, T_{j,k}),$$

where, in each atom $simlit(T_i, T_{j,k})$, $T_{j,k}$ is the variable (of the query) representing the value of the literal $\ell_{j,k}$ for which $\ell_{j,k} = x_i$ in the formula ϕ. Observe that, for *copy* to work properly, each variable x_i must appear as a positive literal in the clauses of $\phi(X, Y)$ at least once. This can be assumed without loss of generality, because if x_i always appears as a negative literal in $\phi(X, Y)$, then we can replace all the occurrences of the negative literal $\neg x_i$ with the positive literal x_i without altering the satisfiability properties of $\phi(X, Y)$.

The fourth piece of the query forces the ground values assigned to the various variables $T_{j,k}$ simulating the assignments to the literals to be consistent. We denote by $v_{j,k}$ the variable of the literal $\ell_{j,k}$. Furthermore, $\ell_{j,k} \sim \ell_{j',k'}$ means that $v_{j,k} = v_{j',k'}$ and the literals $\ell_{j,k}$ and $\ell_{j',k'}$ are both positive or negative, while $\ell_{j,k} \nsim \ell_{j',k'}$ means that $v_{j,k} = v_{j',k'}$ and one literal is positive and the other is negative. We take this part of the query to be

$$consist \equiv \bigwedge_{\ell_{j,k} \sim \ell_{j',k'}} simlit(T_{j,k}, T_{j',k'}) \wedge \bigwedge_{\ell_{j,k} \nsim \ell_{j',k'}} opplit(T_{j,k}, T_{j',k'}),$$

where $T_{j,k}$ is a variable (of the query) representing the value of the literal $\ell_{j,k}$. The last piece of the query checks the satisfiability of $\phi(X, Y)$:

$$satisfied \equiv \bigwedge_{j=1}^{m} clsat(T_{j,1}, T_{j,2}, T_{j,3}).$$

Hence the the query is

$$Q = config \wedge assignX \wedge copy \wedge consist \wedge satisfied,$$

and we choose the distinguished fact ψ to be $clsat(2, 2, 2)$.

Observe that $D \models (Q, \Sigma)$ as $clsat(2, 2, 2) \in D$. We need to show that Φ is valid if and only if there is a MinEX E for $D \models (Q, \Sigma)$ that contains ψ. Note that that checking the validity of $\Phi = \exists X \forall Y \neg \phi(X, Y)$ is tantamount to checking whether there exists a truth assignment σ_X for X such that $\phi(X/\sigma_X, Y)$ is not satisfiable. Therefore, we instead show that there exists a truth assignment σ_X for X such that $\phi(X/\sigma_X, Y)$ is not satisfiable if and only if there exists a MinEX for $D \models (Q, \Sigma)$ including ψ.

($\Rightarrow$) Assume that Φ is valid, hence there exists an assignment $\tilde{\sigma}_X$ for the variables X such that $\phi(X/\tilde{\sigma}_X, Y)$ is not satisfiable. Consider the set $E_{\tilde{\sigma}_X}$ to be

$$D_{st} \cup \{val(x_i, 0) \mid \tilde{\sigma}_X(x_i) = \mathit{false}\} \cup \{val(x_i, 1) \mid \tilde{\sigma}_X(x_i) = \mathit{true}\} \cup \{clsat(2, 2, 2)\}.$$

By construction, $E_{\tilde{\sigma}_X}$ entails the OMQ and contains the fact $\psi = \{clsat(2, 2, 2)\}$. Moreover, we argue that $E_{\tilde{\sigma}_X}$ is also minimal. Indeed, since $\phi(X/\tilde{\sigma}_X, Y)$ is not satisfiable, removing $clsat(2, 2, 2)$ from $E_{\tilde{\sigma}_X}$ breaks the entailment of (Q, Σ). And also removing any other fact from $E_{\tilde{\sigma}_X}$ would cause the set not to entail the OMQ. In particular, removing any fact from D^{St} results in the *config* part of the query not being entailed and removing any *val* fact results in the *assignX* part of the query not being entailed. Therefore, $E_{\tilde{\sigma}_X}$ is a MinEX for $D \models (Q, \Sigma)$.

($\Leftarrow$) Assume that Φ is not valid. This means that, for every assignment σ_X for the variables X, $\phi(X/\sigma_X, Y)$ is satisfiable. Suppose, for a contradiction, that there is a MinEX E that contains $cl(2, 2, 2)$. First, note that since E is minimal, it contains exactly one of the facts $val(x_i, 0)$ and $val(x_i, 1)$ for each variable x_i. Therefore, E gives a proper assignment σ_E to X. As for every assignment σ to X the formula $\phi(X/\sigma, Y)$ is satisfiable, $\phi(X/\sigma_E, Y)$ is satisfiable. Therefore, by construction, $cl(2, 2, 2)$ can be removed from E without breaking the entailment of the OMQ (Q, Σ), which is a contradiction to the minimality of E. □

The above result implies that MinEX-Rel($\subseteq$, BCQ, $\mathcal{L}$) is Σ_2^{p}-hard in *fp*-combined and *ba*-combined complexity for all the considered languages $\mathcal{L}$ as a result of language inclusions. Thus, the Σ_2^{p}-hardness results in Table 3.4 follow.

We now analyse acyclic existential rules, and show that MinEX-Rel($\subseteq$, BCQ, A) is $\mathrm{P}^{\mathrm{NExp}}$-hard in *ba*-combined complexity.

Theorem 3.30. MinEX-Rel($\subseteq$, BCQ, A) *is* $\mathrm{P}^{\mathrm{NExp}}$*-hard in ba-combined complexity.*

Proof. We reduce to MinEX-Rel($\subseteq$, BCQ, A) the following $\mathrm{P}^{\mathrm{NExp}}$-complete problem:

Problem: Extended-Tiling-Problem [76, (complement problem)]
Input: Two tiling problems TP_1 and TP_2 for the exponential square $2^n \times 2^n$, and an integer m in unary notation.
Question: Does there exist an initial condition w of length m such that TP_1 has no solution with w and TP_2 has a solution with w?

In the reduction, for the tiling problems, we use the encoding, presented in [76]. We create two copies for $\Sigma_{TP_i,|w|}$ and D_{TP_i} for $i = 1, 2$. In this reduction we do not

use the facts from D_w, because we need to build a reduction simulating a test for all initial conditions w of length m. We take $\Sigma^1 = \Sigma_{TP_1,|w|}$, $\Sigma^2 = \Sigma_{TP_2,|w|}$,

$$\begin{aligned}\Sigma_{init} = \bigcup_{\substack{0\leq j<m \\ t,t' \text{distinct tiles}}} & \{init_j(t), init_j(t') \rightarrow tiling^1(), \\ & init_j(t) \wedge init_j(t') \rightarrow tiling^2(), \\ & init_j(t) \wedge init_j(t') \rightarrow init\_all()\} \\ & \cup \{init_0(W_0) \wedge \cdots \wedge init_{m-1}(W_{m-1}) \rightarrow init\_all()\}.\end{aligned}$$

and the program to be $\Sigma = \Sigma^1 \cup \Sigma^2 \cup \Sigma_{init}$. Also, we take $D^1 = D_{TP_1}$, $D^2 = D_{TP_2}$ and the database to be $D = D^1 \cup D^2 \cup \{init_j(t) \mid \text{ for all } 0 \leq j < m \text{ and tiles } t\} \cup \{tiling^1()\}$.

The query is $Q = D^1 \wedge D^2 \wedge init\_all() \wedge tiling^1() \wedge tiling^2()$, where symbols D^1 and D^2 in the query mean the conjunction of all database facts from D^1 and D^2, respectively. Notice that $D \models (Q, \Sigma)$. We take the distinguished fact to be $\psi = tiling^1()$.

Let us consider the possible explanations for the OMQ entailment. For a set of facts $E \subseteq D$, if, for some $0 \leq j < m$ and two distinct tiles t and t', E is such that $\{init_j(t), init_j(t')\} \subsetneq E$, then E is an explanation, as it entails the OMQ, but it is *not* minimal. Hence, a MinEX E including $\psi = tiling^1()$, if it exists, must not contain both facts $init_j(t)$ and $init_j(t')$ for any $0 \leq j < m$ and two distinct tiles t and t'. Moreover, a MinEX including $\psi = tiling^1()$ must also satisfy the part $init\_all()$ of the query, and hence it has to include at least one fact $init_j(t)$, for each $0 \leq j < m$. Thus, all sets of facts candidate to be MinEXs including $\psi = tiling^1()$ have to include exactly one fact $init_j(t)$, for each $0 \leq j < m$. This means that these MinEXs encode a proper initial condition w for the tiling problems. Observe that a set of facts encoding a proper initial condition w entails the atom $tiling^i()$ via the mediation of the ontology Σ^i if and only if TP_i has a solution with initial condition w.

It suffices to show that 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 if and only if $D \models (Q, \Sigma)$.

($\Rightarrow$) Assume that there exists an initial condition $\tilde{w}$ of length m such that TP_1 has no solution with $\tilde{w}$ and TP_2 has a solution with $\tilde{w}$. Consider the following set of facts $E_{\tilde{w}} = \{init_j(t) \mid \text{ the } j^{\text{th}} \text{ tile of } \tilde{w} \text{ is } t\} \cup \{tiling^1()\} \cup D^1 \cup D^2$.

First, we show that $E_{\tilde{w}}$ entails (Q, Σ). Clearly, $E_{\tilde{w}}$ satisfies the parts D^1, D^2, and $tiling^1()$, of Q, as they are contained in $E_{\tilde{w}}$. Moreover, $E_{\tilde{w}}$ has a fact $init_j(t)$ for each for each $0 \leq j < m$, hence also the part $init\_all()$ of the query is satisfied by $E_{\tilde{w}}$. To conclude, since we are assuming that TP_2 has a solution with $\tilde{w}$, also the part $tiling^2()$ of Q is satisfied via the mediation of the ontology Σ^2.

Observe that $E_{\tilde{w}}$ is minimal, since none of the facts in $D^1 \cup D^2$ or $\mathit{init}_j(\cdot)$ can be removed from $E_{\tilde{w}}$, otherwise, the query would not be entailed.

Let us focus on $\mathit{tiling}^1()$. Since we are assuming that TP_1 has no solution with w, the fact $\mathit{tiling}^1()$ cannot be entailed from $E_{\tilde{w}}$ via the mediation of the ontology Σ^1, and hence the presence of $\mathit{tiling}^1()$ in $E_{\tilde{w}}$ is essential for the entailment of the query.

($\Leftarrow$) Assume that there exists a MinEX E including $\psi = \mathit{tiling}^1()$. We need to show that there exists an initial condition w_E of length m such that TP_1 has no solution with w_E and TP_2 has a solution with w_E.

Since E is a MinEX containing $\mathit{tiling}^1()$, it must be the case that E encodes a proper initial tiling condition w_E (see the discussion above). Observe that the set E does not contain the fact $\mathit{tiling}^2()$. However E entails the query, which means that $\mathit{tiling}^2()$ is entailed from E via the mediation of the ontology Σ^2, implying that TP_2 has a solution with w_E. Moreover, the minimality of E implies that $\mathit{tiling}^1()$ cannot be removed fro E without losing the entailment property. Hence, TP_1 has no solution with w_E. □

The other hardness results in Table 3.4 follow from the hardness of ontology-mediated query answering in the respective classes of TGDs.

Theorem 3.31. *For any class $\mathcal{L}$ of TGDs,* MinEX-Rel$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *in ba-combined (resp. combined) complexity is as hard as* $\mathrm{OMQA}(\mathrm{BCQ}, \mathcal{L})$ *in ba-combined (resp. combined) complexity.*

Proof. Let D be a database and (Q, Σ) be an $\mathcal{L}$-OMQ, which form an instance of OMQA. We reduce OMQA to MinEX-Rel as follows. Consider the database $D' = D_Q \cup \{f()\}$, where D_Q is the set obtained by grounding a query Q with fresh constants and $f()$ is a fact with a fresh predicate symbol. We take the program $\Sigma' = \Sigma \cup \{f() \rightarrow p(\mathbf{c}) \mid p(\mathbf{c}) \in D\}$. We take the query to be the same. Observe that $D' \models (Q, \Sigma')$ since $D_Q \models Q$. Note that if $f()$ is present then all the facts in D are inferred via Σ'. Therefore, it is not hard to see that $D \models (Q, \Sigma)$ if and only if there is a MinEX for $D' \models (Q', \Sigma)$ including $f()$. Notice that Σ' belongs to the same class of TGDs as Σ since the extra facts of the form $f() \rightarrow p(\mathbf{c})$ belong to all the considered classes of TGDs. □

3.7 Small Explanations

In this section, we analyse the problem, Small-MinEX, of deciding whether there is a MinEX whose size is smaller than some given threshold: Given a database D, an

OMQ (Q, Σ) with $D \models (Q, \Sigma)$, and a number n, asks whether there is a MinEX for $D \models (Q, \Sigma)$ of size smaller than or equal to n. This problem is a decision version of the optimisation problem, which asks for the size of a smallest minimal explanation.

We first provide the membership results for UCQs and proceed with the hardness results for BCQs. First, notice that the membership results for MinEX-Irrel hold for Small-MinEX. The proof of Theorem 3.20 can be adapted to show that Small-MinEX can be decided in $\mathrm{NP}^{\mathcal{C}}$, where $\mathcal{C}$ is the complexity of OMQA.

Theorem 3.32. *For a class $\mathcal{L}$ of TGDs, if* OMQA$(\mathrm{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* Small-MinEX$(\subseteq, \mathrm{UCQ}, \mathcal{L})$ *is in* $\mathrm{NP}^{\mathcal{C}}$.

Proof. Let D be a database, let (Q, Σ) be an OMQ, and let n be a number. To find whether there is a MinEX of size at most n, we can guess in NP a subset E of the database of size at most n and then check whether it entails the OMQ via an oracle call in $\mathcal{C}$. Note that if E entails the OMQ, then it must be the case that it contains as a subset a MinEX that is of size smaller or equal to n. □

From this theorem all the membership results in Table 3.3, apart from the P, NP and NExp membership results, follow. We further note that the NP and NExp membership results follow by the same arguments as in the proofs of Theorems 3.21 and 3.22, respectively.

We show that Small-MinEX is tractable in data complexity for the classes of TGDs for which OMQA is FO-rewritable. Indeed, by Theorem 3.18, we know that, for these classes of TGDs, we can compute all the MinEXs in polynomial time in data complexity. Having this set of all MinEXs, we can check whether there is a MinEX of the size smaller than n in polynomial time. Thus, in this case, finding a MinEX of size at most n is feasible in polynomial time.

Corollary 3.33. *Let $\mathcal{L}$ be a class of TGDs for which* OMQA *is FO-rewritable. Then computing a MinEX for $D \models (Q, \Sigma)$, where Σ is from $\mathcal{L}$, and whose size is at most n is feasible in polynomial time in data complexity.*

This leaves only the matching lower bounds to be shown. First, we note that the proof for the NP-hardness of MinEX-Irrel$(\subseteq, \mathrm{BCQ}, \mathsf{GF})$ in data complexity does not apply to Small-MinEX. Thus we provide a new reduction from Vertex Cover to show NP-hardness of Small-MinEX$(\subseteq, \mathrm{UCQ}, \mathsf{GF})$ in data complexity.

Theorem 3.34. Small-MinEX$(\subseteq, \mathrm{BCQ}, \mathsf{GF})$ *is* NP*-hard in data complexity.*

Proof. Recall that a *vertex cover* of a graph is a set of vertices that includes at least one endpoint of every edge of the graph. We reduce to SMALL-MINEX$(\subseteq, \mathrm{BCQ}, \mathsf{GF})$ the following NP-complete problem.

Problem: VERTEX COVER [104]
Input: A graph $G = (V_G, E_G)$, and an integer k.
Question: Is there a vertex cover in G of size at most k?

Let (G, k) be an instance of VERTEX COVER with $G = (V_G, E_G)$. We build the database D and the OMQ (Q, Σ) from an instance of VERTEX COVER as follows. Suppose that $E_G = \{e_1, \ldots, e_m\}$. We encode the graph G in the database D via an incidence graph, i.e., we have a fact $vc(v)$ for each vertex v of G and a fact $inc(v, e)$ if a vertex v is incident on an edge e:

$$D_G = \{vc(v_i) \mid v_i \in V\} \cup \{inc(v_i, e_j) \mid \text{ edge } e_j \text{ is incident on vertex } v_i\},$$

There are further the *structural* facts in the database, which will be used to test whether the MinEX encodes a vertex cover for G:

$$D_{st} = \{chain(e_i, e_{i+1}) \mid 1 \leq i \leq |E_G| - 1\} \cup \{chain(s, e_1), chain(e_m, t)\} \cup \{all(s), cov(t)\}.$$

Hence, the database is the union of these two sets: $D = D_G \cup D_{st}$.

The program consists of two rules. The first one encodes that if a vertex is in a cover, then all the edges, that are adjacent to it, are covered. The second one captures that the fact $all(t)$ is entailed only if all the edges are covered. The program is

$$\Sigma = \{vc(V) \wedge inc(V, E) \rightarrow cov(E), all(E) \wedge chain(E, F) \wedge cov(F) \rightarrow all(F)\}.$$

The query is $Q = all(t)$. Observe that the query does not depend on the VERTEX COVER instance, as the symbol 't' is constant. We take the threshold of the size of the MinEX to be $n = k + 2|E_G| + 3$. The intuition for this number is that exactly $|E_G|$ of $inc(v, e)$ facts, all $(|E_G| + 1)$ of *chain* facts, $all(s)$ and $cov(t)$ have to be in a MinEX for $D \models (Q, \Sigma)$. Also, we expect at most k of the $vc(v)$ facts to be in a MinEX. This adds up to the number n above.

Observe that $D \models (Q, \Sigma)$ as, intuitively, the set of all vertices of the graph is a vertex cover of the graph. Now we show that there is a vertex cover of G of size at most k if and only if there is a MinEX for $D \models (Q, \Sigma)$ of size at most n.

($\Rightarrow$) Suppose that there is a vertex cover V_c of G of size at most k. Consider the set $E = D_{st} \cup \{vc(v_i) \mid v_i \in V_c\} \cup E'$, where E' contains, for each edge $e_j \in E_G$,

exactly one fact $inc(v_i, e_j)$ such that $v_i \in V_c$. Notice that E is of size at most n (as defined above), and that E entails the OMQ by construction. Therefore, E is a (not necessarily minimal) explanation of size at most n for the OMQ entailment. However, this is enough to conclude that there exists a MinEX (contained in E) for the OMQ entailment of size at most n.

($\Leftarrow$) Suppose that there is a MinEX E of size at most n. Since E entails the OMQ, by the construction of the program Σ, it must be the case that the set of vertices $V_c = \{v_i \in V \mid vc(v_i) \in E\}$ is a vertex cover of G of size at most k. □

The other hardness results in Table 3.3 for Small-MinEX follow from the hardness of ontology-mediated query answering in the respective classes of TGDs.

Theorem 3.35. *For any class $\mathcal{L}$ of TGDs, the problem* Small-MinEX$(\subseteq, \text{BCQ}, \mathcal{L})$ *in fp-combined (resp., ba-, and combined) complexity is as hard as* OMQA$(\text{BCQ}, \mathcal{L})$ *in fp-combined (resp., ba-, and combined) complexity.*

Proof. Let a database D and an OMQ (Q, Σ) be an instance of OMQA. We can reduce OMQA to Small-MinEX as follows. Take $D' = D \cup D_Q$, where D_Q is a set obtained by grounding the query with fresh constants. We take the query to be $Q' = Q \wedge (\bigwedge_{f \in D} f)$. Further, we take the program to be the same and the number n to be $|D|$. First, observe that $D' \models (Q', \Sigma)$, and so D', together with (Q', Σ) is indeed an instance of Small-MinEX.

It is not hard to see that there is a MinEX for $D' \models (Q', \Sigma)$ of size smaller or equal to $|D|$ iff $D \models (Q, \Sigma)$. This is so because every MinEX for $D' \models (Q', \Sigma)$ has to contain D by the construction of the query Q'. Notice that the result holds for all classes of TGDs as we do not change the program in the reduction, just the database and the query. □

3.8 Large Explanations

In this section, we present a complexity analysis for Large-MinEX, the problem asking whether there is a MinEX whose size is larger than some given threshold number: Given a database D, an OMQ (Q, Σ) with $D \models (Q, \Sigma)$, and a number n, asks whether there is a MinEX for $D \models (Q, \Sigma)$ of size larger than or equal to n. This problem is a decision version of the problem asking to find the size of the largest minimal explanation. We first cover membership the results for UCQs and then the hardness results for BCQs, all presented in Table 3.4.

We first show that Large-MinEX($\subseteq$, UCQ, $\mathcal{L}$) is solvable by an NP machine calling an oracle for Is-MinEX($\subseteq$, UCQ, $\mathcal{L}$) by a similar argument to the one for MinEX-Rel in the proof of Theorem 3.26.

Theorem 3.36. *For any class $\mathcal{L}$ of TGDs, if* Is-MinEX($\subseteq$, UCQ, $\mathcal{L}$) *is in $\mathcal{C}$, then* Large-MinEX($\subseteq$, UCQ, $\mathcal{L}$) *is in* $\mathrm{NP}^{\mathcal{C}}$.

Proof. Let D be a database, (Q, Σ) be an OMQ and n be a number with $D \models (Q, \Sigma)$. We can decide whether there is a MinEX E for $D \models (Q, \Sigma)$ of size at least n by guessing a subset E of the database of size at least n in NP and check whether it is a MinEX by calling an oracle for Is-MinEX($\subseteq$, UCQ, $\mathcal{L}$). □

As for MinEX-Rel, all the membership results in Table 3.4 follow from the above theorem, apart from the membership in P in data complexity, which we cover next.

When $\mathcal{L}$ is a class of TGDs for which OMQA is FO-rewritable, by Theorem 3.18, we can compute all MinEXs in polynomial time and then we can decide in polynomial time whether there is a MinEX of size at least n. Therefore, we have the following result, which completes the picture of the membership results for Large-MinEX.

Corollary 3.37. *Let $\mathcal{L}$ be a class of TGDs for which* OMQA *is FO-rewritable. Then computing a MinEX for $D \models (Q, \Sigma)$, where Σ is from $\mathcal{L}$, and whose size is at least n is feasible in polynomial time in data complexity.*

We proceed with the hardness results. First, we show that in data complexity Large-MinEX is NP-hard for the guarded full TGDs. We note that the construction, provided for MinEX-Rel is not applicable, and thus we provide a novel reduction.

Theorem 3.38. Large-MinEX($\subseteq$, BCQ, GF) *is* NP*-hard in data complexity.*

Proof. We reduce to Large-MinEX($\subseteq$, BCQ, GF) the following NP-complete problem:

Problem: Hamiltonian Path [81]
Input: A directed graph $G = (V_G, E_G)$.
Question: Is there a Hamiltonian path in G?

Let $G = (V_G, E_G)$ be a directed graph, which is an instance of Hamiltonian Path. Let $H = (V_H, E_H)$ be a directed graph obtained from G by adding two new vertices s and t, and adding edges so that s has an outgoing edge to every vertex of G and t has an incoming edge from every vertex of G, that is, $V_H = V_G \cup \{s, t\}$ and

$E_H = E_G \cup \{(s, v), (v, t) \mid v \in G\}$. We can view H as G extended with a "source" node s and a "sink" node t. Observe that G has a Hamiltonian path if there is a Hamiltonian path from s to t in H. Thus, in the rest of the proof, we consider H.

We encode the problem of deciding the existence of a Hamiltonian path from s to t in H as follows. First, we take the database to encode the graph H and the "source" node s:

$$D = \{r(u, v) \mid (u, v) \in E_H\} \cup \{p(s)\}.$$

The program consists of a single axiom, which encodes the reachability:

$$\Sigma = \{p(X) \wedge r(X, Y) \rightarrow p(Y)\}$$

The query asks whether $p(t)$ holds, that is, whether it is reachable from s. It is $Q = p(t)$. and we take the threshold for the size of the MinEX to be $n = |V_H|$.

We show that there is a Hamiltonian path from s to t in H if and only if there is a MinEX for $D \models (Q, \Sigma)$ of size at least n.

($\Rightarrow$) Suppose that there is a Hamiltonian path P from s to t in H. Take a subset $E = \{r(u, v) \mid (u, v) \in P\} \cup \{p(s)\}$. Observe that $E \models (Q, \Sigma)$ as t is reachable from s via the path P. Moreover, we claim that E is minimal. Indeed, we cannot remove $p(s)$ from E, otherwise no $p(v)$ fact could be entailed and so $p(t)$. Also, no $r(u, v)$ fact can be removed, since P is a path, this would result in $p(t)$ not being entailed.

($\Leftarrow$) Suppose that there is a MinEX E for $D \models (Q, \Sigma)$ of size at least n. Certainly, in order to entail the query, E contains $p(s)$ and thus E contains $|V_H| - 1$ facts that correspond to edges of H. Since E is minimal, the $r(u, v)$ facts in E must give a path P (that is, no vertex is visited twice), otherwise E would not be minimal. This also implies that P must be a Hamiltonian path from s to t as it contains $|V_H| - 1$ edges. □

The above result implies all the NP-hardness results in Table 3.4 since all those classes contain GF.

The remaining hardness proofs can be obtained by adapting the proofs for MinEX-Rel. The Σ_2^{P}-hardness of Large-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ in *fp*-combined complexity can be shown by slightly modifying the reduction, given in the proof of Theorem 3.29. In particular, simply take the threshold for the size of the MinEX to be $n = |D_{St}| + |X| + 1$.

Theorem 3.39. *For any class $\mathcal{L}$ of TGDs,* Large-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *is Σ_2^{P}-hard in fp-combined complexity.*

The $\mathrm{P}^{\mathrm{NExp}}$-hardness of Large-MinEX$(\subseteq, \mathrm{BCQ}, \mathsf{A})$ in *ba*-combined complexity can be shown by extending the reduction in the proof of Theorem 3.30 by simply taking the threshold for the size of the MinEX to be $n = m + 1$.

Theorem 3.40. Large-MinEX$(\subseteq, \mathrm{BCQ}, \mathsf{A})$ *is* $\mathrm{P}^{\mathrm{NExp}}$*-hard in ba-combined complexity.*

The other hardness results in Table 3.4 for Large-MinEX follow from the hardness of ontology-mediated query answering in the respective classes of TGDs.

Theorem 3.41. *For any class $\mathcal{L}$ of TGDs, the problem* Large-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *in ba-combined (resp. combined) complexity is as hard as* OMQA$(\mathrm{UCQ}, \mathcal{L})$ *in ba-combined (resp. combined) complexity.*

Proof. Let D be a database and (Q, Σ) be an OMQ, which form an instance of OMQA. We can reduce OMQA to Large-MinEX as follows. The database is $D' = D \cup \{f(), g_1(), g_2()\}$, where $f()$, $g_1()$, and $g_2()$ are nulary facts with fresh predicate symbols. The program is $\Sigma' = \Sigma \cup \{f() \rightarrow p(\mathbf{c}) \mid p(\mathbf{c}) \in D_Q \cup \{g_1(), g_2()\}\}$, where D_Q is the set obtained by grounding the query Q with fresh constants. The query is $Q' = Q \wedge g_1() \wedge g_2()$. Finally, we take the number n to be 2. Observe that $D' \models (Q, \Sigma')$ since $f()$ via Σ' infers all the facts in D_Q, and thus $D' \models (Q', \Sigma')$ as $g_1(), g_2() \in D'$. Therefore, D', (Q', Σ') and n form an instance of Large-MinEX. Now we show that $D \models (Q, \Sigma)$ iff there is a MinEX E for $D' \models (Q', \Sigma')$ of size at least $n = 2$.

($\Rightarrow$) Suppose that $D \models (Q, \Sigma)$. We know that there is a MinEX E for $D \models (Q, \Sigma)$. We further observe that $E \cup \{g_1(), g_2()\}$ must be a MinEX for $D' \models (Q', \Sigma')$, which has size at least $n = 2$.

($\Leftarrow$) Suppose that there is a MinEX E' for $D' \models (Q', \Sigma')$ of size at least $n = 2$. Note that $\{f()\}$ is a MinEX for $D' \models (Q', \Sigma')$, therefore, no other MinEX contains $f()$. Therefore, any other MinEX has to contain $\{g_1(), g_2()\}$ and thus must be of size at least $n = 2$ and the entailment of Q can only follow from the facts in D. □

3.9 Overview of Complexity Results

In this chapter, we have provided an analysis of the complexity of the problems associated with explaining ontology-mediated query answers as inclusion-minimal subsets of the database that support the entailment. Such subsets are called MinEXs and, in this chapter, we study them under different fragments of existential rules.

We have studied a wide range of problems, all introduced formally in Section 3.2. These problems include recognising a MinEX, Is-MinEX, recognising the set of all MinEXs, All-MinEX, deciding if a distinguished fact is included in some MinEX, MinEX-Rel, deciding if there is a MinEX that avoids a given set of subsets of the database, MinEX-Irrel, and deciding whether there is a MinEX smaller (resp., larger) than some given threshold number, Small-MinEX (resp., Large-MinEX). These problems have been chosen to represent different types of properties that we might want to consider. In particular, the problems MinEX-Rel and Large-MinEX require to check the property that is preserved when adding facts to a subset of a database, and the problems MinEX-Irrel and Small-MinEX require to check the property that is preserved when removing facts from a subset of a database. We give some of the intuitions for the complexity results, given in Tables 3.1–3.4. They range from membership in AC^0 to 2Exp-completeness.

We discuss some of the complexity results, shown in this chapter. First of all, we observe that rather surprisingly the complexity of Is-MinEX in data complexity does not increase, compared to OMQA. The problem remains tractable primarily because in order to ensure minimality of an explanation, we need to carry out an extra polynomial number of non-entailment checks (compared to OMQA), each of which is in polynomial time in data complexity. Even more surprisingly, Is-MinEX is in AC^0 for FO-rewritable languages in data complexity since we can capture the property of being a MinEX with a FO-formula. For All-MinEX, we already witness intractability in data complexity for guarded TGDs since ensuring that there is no MinEX outside of our candidate set of all MinEXs requires an coNP check. Surprisingly, we see that, in *fp*-, *ba*- and combined complexity measures, Is-MinEX and All-MinEX are complete for the same complexity classes. It is primarily because, when solving All-MinEX, for any set $\mathcal{E}$ of subsets of the database D, the tasks of checking minimality of all the sets in $\mathcal{E}$ and ensuring that there is no MinEX outside of $\mathcal{E}$ can be combined. This results in matching complexities of the two problems for these complexity measures.

For the problems MinEX-Irrel and Small-MinEX, the complexity results coincide. For both of these problems, observe intractability in data complexity as we need to guess a subset that satisfies the conditions (i.e., avoids forbidden sets or is smaller than a threshold), which requires an NP machine. On the other hand, surprisingly, in *fp*-, *ba*- and combined complexity measures, these problems have the same complexity as OMQA. Observe that if there is an explanation that either avoids forbidden sets or is smaller than some threshold number, then this property is

preserved by removing elements. Therefore, we do *not* need to ensure minimality of a guessed subset.

Finally, for the problems MinEX-Rel and Large-MinEX, the complexity results match too. For both of the problems, we observe the results in the second level of the polynomial hierarchy already in *fp*-combined complexity. This is primarily because we need to guess a subset that satisfies the property (either contains a distinguished fact or is larger than some threshold number) and check whether that set is a MinEX. Note that ensuring whether a subset is a MinEX is intractable in *fp*-combined complexity for all fragments of existential rules, and thus MinEX-Rel and Large-MinEX are in Σ_2^{P} for all of these fragments in *fp*-combined complexity. Furthermore, we show Σ_2^{P}-completeness in *fp*-combined complexity by a reduction from a variant of a satisfiability problem.

Chapter 4

Explanations for Query Answers in Description Logics

In this chapter, we present a study of the problems associated with explanations for ontology-mediated query answers in description logics. This chapter provides the complexity analysis analogous to the one in Chapter 3 with the ontology language being description logics, rather than existential rules. We observe that none of the description logics that we study are equivalent to the classes of existential rules. Among many differences, description logics allows predicates of arity at most 2, while in existential rules predicates can have any arity. Therefore, many of the results and, in particular, the proofs for hardness results are different in this chapter, compared to Chapter 3. In many of the proofs in Chapter 3, we have made use of predicates of arity 3 or higher, which are not available in description logics. As description logics and existential rules are the two most widely used ontology languages for ontology-mediated query answering, this chapter completes the picture of the complexity of explaining positive ontology-mediated query answers via inclusion-minimal subset of the data.

This chapter is based on the paper published in the proceedings of the European Conference on Artificial Intelligence (ECAI) in 2020 [50].

4.1 Minimal Explanations

In this chapter, we restate the definition of a minimal explanation for a given OMQ entailment, introduced in Chapter 3, using the standard description logics terminology. We view explanations as minimal subsets of the ABox that entail the OMQ.

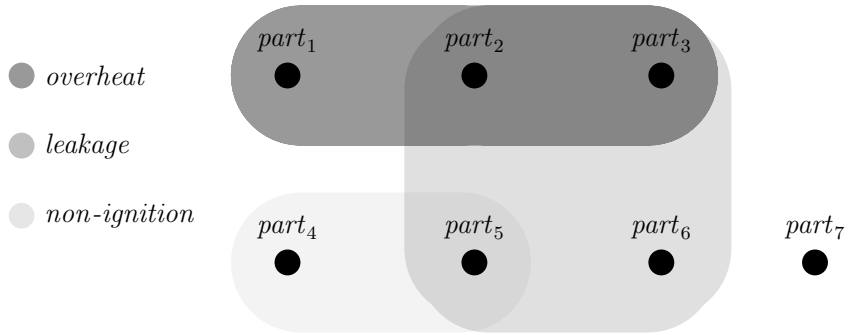

Figure 4.1: Illustration of possible faults in different parts of a system.

Definition 4.1 (MinEX)**.** Let $\mathcal{A}$ be an ABox and $(Q, \mathcal{T})$ be an OMQ such that the knowledge base $(\mathcal{T}, \mathcal{A})$ is consistent and $\mathcal{A} \models (Q, \mathcal{T})$. An *explanation* for $\mathcal{A} \models (Q, \mathcal{T})$ is a subset $\mathcal{E} \subseteq \mathcal{A}$ such that $\mathcal{E} \models (Q, \mathcal{T})$.

A *minimal explanation* $\mathcal{E}$, or a *MinEX*, for $\mathcal{A} \models (Q, \mathcal{T})$ is an explanation for $\mathcal{A} \models (Q, \mathcal{T})$ that is *subset-minimal*, i.e., there is no proper subset $\mathcal{E}' \subsetneq \mathcal{E}$ that is an explanation for $\mathcal{A} \models (Q, \mathcal{T})$.

We simply say that $\mathcal{E}$ is a MinEX if the ABox and the OMQ are clear from the context. Importantly, this definition assumes the initial knowledge base to be consistent, which is not explicit in the original definition in Chapter 3, but it is implied there as that chapter focuses on existential rules without any negative constraints.

The choice of consistency of the initial knowledge base enables us to focus on the complexity of recognising explanations as we do not need to carry out extra checks to ensure consistency. In particular, note that we do not need to deal with inconsistency as, if $(\mathcal{T}, \mathcal{A})$ is consistent, then for any subset $\mathcal{B} \subseteq \mathcal{A}$, the knowledge base $(\mathcal{T}, \mathcal{B})$ is also consistent.

We provide a running example that will be used in this section to illustrate the computational problems. The following example shows that MinEXs can capture *minimal hitting sets*, and therefore can be applied in a variety of fields [80], including, computational biology [48, 105], data mining [29], and model-based fault diagnosis [140]. Our running example shows how a *fault diagnosis* problem can be expressed.

Example 4.2. Suppose we have an engine that might experience a number of possible failures, each caused by a fault of one of the constituent parts, presented in Figure 4.1. We can encode *fault diagnosis* as follows. We define an ABox $\mathcal{A} = \mathcal{A}_{faults} \cup \mathcal{A}_{st}$, where the first component of the ABox encodes the parts that may fail:

$$\mathcal{A}_{faults} = \{Fault(part_i) \mid 1 \leq i \leq 7\},$$

and the second component of the ABox encodes the causes of the failures:

$$\begin{aligned}\mathcal{A}_{st} = &\{\mathit{caused}(\mathit{overheat}, \mathit{part}_i) \mid i \in \{1,2,3\}\} \cup \\ &\{\mathit{caused}(\mathit{leakage}, \mathit{part}_i) \mid i \in \{2,3,5,6\}\} \cup \\ &\{\mathit{caused}(\mathit{non\text{-}ignition}, \mathit{part}_i) \mid i \in \{4,5\}\}.\end{aligned}$$

The TBox consists of a single axiom that expresses that, if a part failed, it causes the corresponding failures:

$$\mathcal{T} = \{\exists \mathit{caused}.\mathit{Fault} \sqsubseteq \mathit{Failure}\}.$$

The query asks for a minimal set of faults that explains the observed failures, together with the facts that capture how failures can be caused:

$$Q = \mathit{Failure}(\mathit{overheat}) \wedge \mathit{Failure}(\mathit{leakage}) \wedge \mathit{Failure}(\mathit{non\text{-}ignition}) \wedge \bigwedge_{\psi \in \mathcal{A}_{st}} \psi.$$

By construction, MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$ correspond to minimal covers of the graph in Figure 4.1. For example, we can see that $\mathcal{E} = \{\mathit{Fault}(\mathit{part}_3), \mathit{Fault}(\mathit{part}_5)\} \cup \mathcal{A}_{st}$ is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$. ∎

4.2 Explanation Problems

In this section, we state the problems, introduced in Section 3.2 for existential rules, using the standard description logics terminology. We consider the four problems, namely, recognising whether a set is a MinEX, Is-MinEX, recognising if a set contains all MinEXs, All-MinEX, deciding if a fact is contained in some MinEX, MinEX-Rel, and deciding if there is a MinEX that does not contain any forbidden set, MinEX-Irrel.

The most fundamental problem is recognising a minimal explanation.

Problem: Is-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: An ABox $\mathcal{A}$, an $\mathcal{L}$-OMQ $(Q, \mathcal{T})$, and a subset $\mathcal{E} \subseteq \mathcal{A}$, where Q is a query from the class $\mathcal{Q}$, $(\mathcal{T}, \mathcal{A})$ is consistent and $\mathcal{A} \models (Q, \mathcal{T})$.
Question: Is $\mathcal{E}$ a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$?

Let us illustrate Is-MinEX on the setting, given in Example 4.2.

Example 4.3. Consider the following subset of the ABox $\mathcal{A}$:

$$\begin{aligned}
\mathcal{E}_1 &= \{\mathit{Fault}(\mathit{part}_2), \mathit{Fault}(\mathit{part}_5\} \cup \mathcal{A}_{st},\\
\mathcal{E}_2 &= \{\mathit{Fault}(\mathit{part}_1), \mathit{Fault}(\mathit{part}_4), \mathit{Fault}(\mathit{part}_6)\} \cup \mathcal{A}_{st},\\
\mathcal{E}_3 &= \{\mathit{Fault}(\mathit{part}_1), \mathit{Fault}(\mathit{part}_4), \mathit{Fault}(\mathit{part}_5)\} \cup \mathcal{A}_{st}, \text{ and}\\
\mathcal{E}_4 &= \{\mathit{Fault}(\mathit{part}_4), \mathit{Fault}(\mathit{part}_6)\} \cup \mathcal{A}_{st}.
\end{aligned}$$

Note that $\mathcal{E}_1$ and $\mathcal{E}_2$ are MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$, while $\mathcal{E}_3$ and $\mathcal{E}_4$ are not. The set $\mathcal{E}_3$ is not a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ since its subset $\{\mathit{Fault}(\mathit{part}_1), \mathit{Fault}(\mathit{part}_5)\} \cup \mathcal{A}_{st}$ entails the OMQ $(Q, \mathcal{T})$, and so $\mathcal{E}_3$ is not minimal. The set $\mathcal{E}_4$ is not a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ as $\mathcal{E}_4 \not\models (Q, \mathcal{T})$. ∎

Another fundamental problem is recognising the set of all MinEXs.

Problem: All-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: An ABox $\mathcal{A}$, an $\mathcal{L}$-OMQ $(Q, \mathcal{T})$, and a set $\mathfrak{E} \subseteq \mathcal{P}(\mathcal{A})$ of subsets of $\mathcal{A}$, where the query Q is from the class $\mathcal{Q}$, $(\mathcal{T}, \mathcal{A})$ is consistent and $\mathcal{A} \models (Q, \mathcal{T})$.
Question: Is $\mathfrak{E}$ precisely the set of all MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$?

We illustrate All-MinEX on our running example.

Example 4.4. In our example, the set $\mathfrak{E}$ of all MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$ is

$$\begin{aligned}
\mathfrak{E} = \big\{ &\{\mathit{Fault}(\mathit{part}_1), \mathit{Fault}(\mathit{part}_4), \mathit{Fault}(\mathit{part}_6)\} \cup \mathcal{A}_{st},\\
&\{\mathit{Fault}(\mathit{part}_1), \mathit{Fault}(\mathit{part}_5)\} \cup \mathcal{A}_{st}, \{\mathit{Fault}(\mathit{part}_2), \mathit{Fault}(\mathit{part}_4)\} \cup \mathcal{A}_{st},\\
&\{\mathit{Fault}(\mathit{part}_2), \mathit{Fault}(\mathit{part}_5)\} \cup \mathcal{A}_{st}, \{\mathit{Fault}(\mathit{part}_3), \mathit{Fault}(\mathit{part}_4)\} \cup \mathcal{A}_{st},\\
&\{\mathit{Fault}(\mathit{part}_3), \mathit{Fault}(\mathit{part}_5)\} \cup \mathcal{A}_{st}\big\}.
\end{aligned}$$

∎

In some applications, we are interested in the existence of a MinEX avoiding certain configurations, that is, the problem of deciding if there is a MinEX that does not contain any of the forbidden sets of assertions.

Problem: MinEX-Irrel$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: An ABox $\mathcal{A}$, an $\mathcal{L}$-OMQ $(Q, \mathcal{T})$, and a set $\mathfrak{F} \subseteq \mathcal{P}(\mathcal{A})$ of subsets of $\mathcal{A}$, where the query Q is from the class $\mathcal{Q}$, $(\mathcal{T}, \mathcal{A})$ is consistent and $\mathcal{A} \models (Q, \mathcal{T})$.
Question: Is there a MinEX $\mathcal{E}$ for $\mathcal{A} \models (Q, \mathcal{T})$ such that $F \not\subseteq \mathcal{E}$ for every $F \in \mathfrak{F}$?

We illustrate MinEX-Irrel on our running example, by assuming that in our engine, some sets of parts cannot *all* be faulty. This constraint can be formulated as an instance of MinEX-Irrel.

Example 4.5. Suppose that none of the sets in the following set are allowed to appear in an explanation:

$$\mathfrak{F}_1 = \{\{\mathit{Fault}(\mathit{part}_5)\}, \{\mathit{Fault}(\mathit{part}_1), \mathit{Fault}(\mathit{part}_6)\}\}.$$

Observe that there is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ not containing any of the sets in $\mathfrak{F}_1$, for example, $\{\mathit{Fault}(\mathit{part}_2), \mathit{Fault}(\mathit{part}_4)\} \cup \mathcal{A}_{st}$. On the other hand, if the set of the forbidden sets is $\mathfrak{F}_2 = \mathfrak{F}_1 \cup \{\{\mathit{Fault}(\mathit{part}_4)\}\}$, then there is no MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ such that no set in $\mathfrak{F}_2$ is contained in $\mathcal{E}$. ∎

The last problem that we investigate is related to the role of an individual assertion in explaining OMQ answers. This problem asks whether there is a MinEX containing a distinguished assertion. We say that an assertion ψ is *relevant* for $\mathcal{A} \models (Q, \mathcal{T})$ iff ψ appears in some MinEX for $\mathcal{A} \models (Q, \mathcal{T})$.

Problem: MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$
Input: An ABox $\mathcal{A}$, an OMQ $(Q, \mathcal{T})$, and an assertion $\psi \in \mathcal{A}$, where the query Q is from the class $\mathcal{Q}$, $(\mathcal{T}, \mathcal{A})$ is consistent and $\mathcal{A} \models (Q, \mathcal{T})$.
Question: Is $\psi \in \mathcal{E}$ relevant for $\mathcal{A} \models (Q, \mathcal{T})$, i.e., is there a MinEX $\mathcal{E}$ for $\mathcal{A} \models (Q, \mathcal{T})$ such that $\psi \in \mathcal{E}$?

We illustrate the problem MinEX-Rel on our running example.

Example 4.6. Suppose that we want to know whether a particular part might be responsible for the observed failures. For example, suppose that we want to find out whether $\psi_1 = \mathit{Fault}(\mathit{part}_1)$ or $\psi_2 = \mathit{Fault}(\mathit{part}_7)$ is relevant for $\mathcal{A} \models (Q, \mathcal{T})$. Note that there is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ that contains ψ_1, namely, the set $\{\mathit{Fault}(\mathit{part}_1), \mathit{Fault}(\mathit{part}_5)\} \cup \mathcal{A}_{st}$. On the other hand, there is no MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ that contains ψ_2. ∎

4.3 Recognising Minimal Explanations

In this section, we provide a complexity analysis for Is-MinEX. We first provide the membership results, and then proceed to prove the matching lower bounds. We observe that the generic approach for solving Is-MinEX for existential rules, presented in the proof of Theorem 3.8, also applies here. Suppose that OMQA is in the complexity class $\mathcal{C}$. To ensure that a set $\mathcal{E}$ is a MinEX for the OMQ entailment, we need to check that it entails the query and that it is minimal. The entailment can be checked with a single $\mathcal{C}$ test. The latter can be done by checking whether removing any single assertion from $\mathcal{E}$ would invalidate the OMQ, and hence it can be done with

a polynomially-many co-$\mathcal{C}$ checks. Thus the whole procedure is in $\mathrm{P}^{\mathcal{C}}$. We state the following corollary of Theorem 3.8.

Corollary 4.7. *For any DL $\mathcal{L}$, if* $\mathrm{OMQA}(\mathcal{Q}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* $\textsc{Is-MinEX}(\subseteq, \mathcal{Q}, \mathcal{L})$ *can be decided by a check in $\mathcal{C}$ and a linear number of checks in co-$\mathcal{C}$.*

We observe that Corollary 4.7 covers all the membership results shown in Table 4.1, apart from the membership results in AC^0 and in D^{P}. We illustrate how the results in Table 4.1 follow from Corollary 4.7. If OMQA is in P, then Is-MinEX is in P by the same argument as in Chapter 3; a single test in P and a linear number of tests in co-$\mathcal{C}$ can be carried out by a single test in P. Similarly, if $\mathcal{C} \in \{\textsc{Exp}, \textsc{2Exp}\}$, then co-$\mathcal{C} = \mathcal{C}$, and so Is-MinEX can be decided with a polynomial number of $\mathcal{C}$ tests, which can be carried out by a single $\mathcal{C}$ test.

If OMQA is in NP or coNP, we have that Is-MinEX is in D^{P} by a similar argument as in the proof of Corollary 3.9.

Corollary 4.8. *If* $\mathrm{OMQA}(\mathcal{Q}, \mathcal{L})$ *is in* NP *or* coNP, $\textsc{Is-MinEX}(\subseteq, \mathcal{Q}, \mathcal{L})$ *is in* D^{P}.

Proof. If OMQA is in NP, the same argument as in the proof of Corollary 3.9 applies.

If OMQA is in coNP, then note that Is-MinEX can be decided by a single coNP test and a linear number of NP tests. Note that a linear number of NP tests can be carried out by a single NP machine when we combine the certificates. Therefore Is-MinEX is solvable by a single coNP test and a single NP test, and therefore it is in D^{P}. □

We further note that the proof of Theorem 3.10 applies to description logics, and thus we have the following corollary. Since IQ is a special case of UCQ, the following result applies to instance queries too.

Corollary 4.9. *If $\mathcal{L}$ is a FO-rewritable DL, then* $\textsc{Is-MinEX}(\subseteq, \mathrm{UCQ}, \mathcal{L})$ *is in* AC^0 *in data complexity.*

We proceed with showing the hardness results, presented in Table 4.1. First, we show that $\textsc{Is-MinEX}(\subseteq, \mathrm{IQ}, \mathcal{ELI})$ is P-hard in data complexity. We achieve this by reducing $\mathrm{OMQA}(\mathrm{IQ}, \mathcal{ELI})$ to $\textsc{Is-MinEX}(\subseteq, \mathrm{IQ}, \mathcal{ELI})$, and thus showing that recognising a MinEX is not easier than OMQA in this case. In this reduction, we replace each role assertion with two role assertions and introduce reachability type axioms to the TBox to ensure that each MinEX must contain all the bounded assertions in the ABox.

Table 4.1: Complexity results for Is-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$.

Description Logic	IQ		UCQ	
	data	comb.	data	comb.
DL-Lite$_{core}$, *DL-Lite*$_{\mathcal{R}}$	$\leq$ AC^0	$\leq$ P	$\leq$ AC^0	D$^{\text{P}}$
$\mathcal{EL}, \mathcal{ELH}$	$\leq$ P	P	$\leq$ P	D$^{\text{P}}$
$\mathcal{ELI}$, Horn-$\mathcal{SHOIQ}$	P	Exp	P	Exp
$\mathcal{ALC}, \mathcal{ALCHQ}$	D$^{\text{P}}$	Exp	D$^{\text{P}}$	Exp
$\mathcal{ALCI}, \mathcal{SH}, \mathcal{SHIQ}$	D$^{\text{P}}$	Exp	D$^{\text{P}}$	2Exp

Theorem 4.10. Is-MinEX$(\subseteq, \text{IQ}, \mathcal{ELI})$ *is* P*-hard in data complexity, and* Exp*-hard in combined complexity.*

Proof. We provide a reduction from OMQA(IQ, $\mathcal{ELI}$) to Is-MinEX$(\subseteq, \text{IQ}, \mathcal{ELI})$. Let $\mathcal{A}$ be an ABox and $(Q, \mathcal{T})$ be an OMQ with $Q = C(a)$. We construct an ABox $\mathcal{A}'$, a subset $\mathcal{E}' \subseteq \mathcal{A}'$ and an OMQ $(Q', \mathcal{T}')$ such that $\mathcal{A} \models (Q, \mathcal{T})$ iff $\mathcal{E}$ is a MinEX for $\mathcal{A}' \models (Q', \mathcal{T}')$. Note that, in this construction, $(Q', \mathcal{T}')$ depends only on $(Q, \mathcal{T})$ and is independent of $\mathcal{A}$. On the other hand, $\mathcal{A}'$ and $\mathcal{E}'$ depends on $\mathcal{A}$ and $(Q, \mathcal{T})$.

We construct $\mathcal{A}'$ as follows. Take any order of the elements in $\mathcal{A}$. For the i^{th} concept assertion $B(b)$ in $\mathcal{A}$, we add $B(b)$, $s_B(i, b)$ and $n(i, i-1)$ to $\mathcal{A}'$, and, for the j^{th} role assertion $r(b, c)$ in $\mathcal{A}$, we add $r_1(j, b)$, $r_2(j, c)$ and $n(j, j-1)$ to $\mathcal{A}'$. Further, we add the assertions $U(0), n(\tau, k)$, $n(\tau, a)$ and $T(\tau)$ to $\mathcal{A}'$, where a is the individual in the query $Q = C(a)$ and k is the number of elements in $\mathcal{A}$. We also add $F(\tau)$ to $\mathcal{A}'$. Then we take $\mathcal{E}'$ to be $\mathcal{A}' \setminus \{F(\tau)\}$. Note that the construction of $\mathcal{A}'$ and $\mathcal{E}'$ is in logarithmic space in the size of $\mathcal{A}$ as we only need to store the number of the assertion considered.

Now we construct $\mathcal{T}'$ as follows. Add to $\mathcal{T}'$ every axiom in $\mathcal{T}$ with each occurrence of $\exists r.B$ replaced by $\exists r_1^-.(\exists r_2.B)$. Note that, by construction, $\exists r.B(a)$ holds in $\mathcal{A}$ iff $\exists r_1^-.(\exists r_2.B)(a)$ holds in $\mathcal{E}'$. Further, add to $\mathcal{T}'$ the GCI that ensures if $n(i+1, i)$, $U(i)$ and $T(i)$ hold, then so does $U(i+1)$:

$$\exists n.(U \sqcap T) \sqsubseteq U.$$

This axiom acts as a *reachability* axiom. Let $\mathsf{N_C}$ be a finite set of concepts, which is fixed as we are in data complexity. For each concept symbol $B \in \mathsf{N_C}$, add to $\mathcal{T}'$ the GCI that ensures that $T(i)$ holds if $s_B(i, b)$ and $B(b)$ are present:

$$\exists s_B.B \sqsubseteq T.$$

Also, let $\mathsf{N_R}$ be a finite set of role symbols, which is also fixed as we are in data complexity. For each role symbol $r \in \mathsf{N_R}$, add the role inclusion that ensure that $T(i)$ holds if $r_1(i, a)$ and $r_2(i, b)$ are present:

$$\exists r_1.\top \sqcap \exists r_2.\top \sqsubseteq T.$$

We add to $\mathcal{T}'$ two more axioms: $\exists n.C \sqsubseteq C$, and $C \sqcap U \sqsubseteq F$. The first one ensures that $C(\tau)$ holds if $C(a)$ and $n(\tau, a)$ do, and the second one ensures that $F(\tau)$ holds if $C(\tau)$ and $U(\tau)$ do.

We take the instance query to be $Q' = F(\tau)$.

We note that $\mathcal{A}' \models (Q', \mathcal{T}')$ since $F(\tau)$ is in $\mathcal{A}'$, and thus $\mathcal{A}', \mathcal{E}'$ and $(Q', \mathcal{T}')$ is an instance of Is-MinEX. Now we show that $\mathcal{A} \models (Q, \mathcal{T})$ iff $\mathcal{E}'$ is a MinEX for $\mathcal{A}' \models (Q', \mathcal{T}')$.

($\Rightarrow$) Suppose that $\mathcal{A} \models (Q, \mathcal{T})$. Recall that $Q = C(a)$. First, we show that $\mathcal{E}' \models (Q', \mathcal{T}')$. Recall that $\mathcal{E}'$ contains all concept assertions from $\mathcal{A}$ and role assertions $r_1(j, a)$ and $r_2(j, b)$ for each $r(a, b)$ in $\mathcal{A}'$. As $\mathcal{T}'$ contains $\mathcal{T}$ with each occurrence of $\exists r.B$ replaced by $\exists r_1^-.(\exists r_2.B)$, we have that r_1 and r_2 simulate r. As a result, we have that $\mathcal{E} \models (C(a), \mathcal{T}')$. As $n(\tau, a) \in \mathcal{E}$ and $(\exists n.C \sqsubseteq C) \in \mathcal{T}'$, we have that $\mathcal{E} \models (C(\tau), \mathcal{T}')$. As $(C \sqcap U \sqsubseteq F) \in \mathcal{T}$, to show that $\mathcal{E} \models (F(\tau), \mathcal{T}')$, it is enough to show that $\mathcal{E} \models (U(\tau), \mathcal{T}')$. Note that, by construction of $\mathcal{E}'$ and $\mathcal{T}'$, $\mathcal{E}' \models (T(i), \mathcal{T}')$ for all $i \leq k$, where k is the number of elements in $\mathcal{A}'$. Since $U(0)$ is in $\mathcal{E}'$ and $(\exists n.(U \sqcap T) \sqsubseteq U) \in \mathcal{T}'$, we have that $\mathcal{E}' \models (U(i), \mathcal{T}')$ for all $i \leq k$. Further, there is an assertion $n(\tau, k)$, and therefore $U(\tau)$ holds. This implies that $\mathcal{E} \models (Q', \mathcal{T}')$. Further, $\mathcal{E}$ is minimal for $(Q', \mathcal{T}')$ since removing any element from $\mathcal{E}'$ would result in some $(T(i), \mathcal{T}')$ not being entailed. As a result, $(U(i+1), \mathcal{T}')$ would not be entailed, which would result in $(U(\tau), \mathcal{T}')$ and $(Q', \mathcal{T}')$ not being entailed by $\mathcal{E}'$.

($\Leftarrow$) For the other direction, suppose that $\mathcal{E}'$ is a MinEX for $\mathcal{A}' \models (Q', \mathcal{T}')$. We have seen above that $\mathcal{E}$ and $\mathcal{T}'$ simulates $\mathcal{A}$ and $\mathcal{T}$, and so we have $\mathcal{A} \models (Q, \mathcal{T})$. This completes the proof.

□

The above theorem implies that Is-MinEX($\subseteq$, IQ, Horn-$\mathcal{SHOIQ}$) is P-hard in data complexity as $\mathcal{ELI}$ is contained in Horn-$\mathcal{SHOIQ}$. By analogous language inclusions, we have the Exp-hardness of Is-MinEX for (IQ, Horn-$\mathcal{SHOIQ}$), (IQ, $\mathcal{ALCI}$), and (IQ, $\mathcal{SHIQ}$) in combined complexity.

Somewhat surprisingly, it is unclear whether Is-MinEX($\subseteq$, IQ, $\mathcal{EL}$) is also P-hard in data complexity, and so it is open whether Theorem 4.10 can be strengthened in

this direction. The reason behind this is that our proof heavily relies on inverse roles to enforce all ABox assertions to be included in the minimal explanation, which is necessary for Is-MinEX to simulate OMQA faithfully.

All the other P-hardness (resp., Exp-hardness) results given in Table 4.1 are covered by the following result.

Theorem 4.11. Is-MinEX$(\subseteq, \text{IQ}, \mathcal{EL})$ *is* P*-hard in combined complexity, and* Is-MinEX$(\subseteq, \text{IQ}, \mathcal{ALC})$ *is* Exp*-hard in combined complexity.*

Proof. Let $\mathcal{L}$ be a DL. We reduce concept subsumption to Is-MinEX in combined complexity. As shown by Artale et al. [6], we have that $\mathcal{T} \models (C \sqsubseteq D)$ iff $\{C(a)\} \models (\{D(a)\}, \mathcal{T})$, where a is a fresh individual name. Therefore, we have that $\mathcal{T} \models (C \sqsubseteq D)$ iff $\{C(a)\}$ is a MinEX for $\{C(a), D(a)\} \models (\{D(a)\}, \mathcal{T})$. Since concept subsumption in $\mathcal{EL}$ is P-hard and in $\mathcal{ALC}$ is Exp-hard, the results follow. □

The above result implies that Is-MinEX$(\subseteq, \text{IQ}, \mathcal{ELH})$ is P-hard in combined complexity and Is-MinEX$(\subseteq, \text{IQ}, \mathcal{L})$ is Exp-hard in combined complexity for $\mathcal{L} \in \{\mathcal{ALCHQ}, \mathcal{ALCI}, \mathcal{SH}, \mathcal{SHIQ}\}$ due to the description logics containment, presented in Figure 2.3. Also, since IQ is a special case of a UCQ, we have that these results hold for UCQs too.

Our next result covers all D^{P}-hardness results for Is-MinEX in data complexity in Table 4.1. This next result is arguably the most interesting result of this section.

Theorem 4.12. Is-MinEX$(\subseteq, \text{IQ}, \mathcal{ALC})$ *is* D^{P}*-hard in data complexity.*

Proof. We reduce the D^{P}-complete problem Is-MUS [128] to Is-MinEX$(\subseteq, \text{IQ}, \mathcal{ALC})$ in data complexity. The problem Is-MUS is defined as follows:

Problem: Is-MUS
Input: A CNF formula ϕ with at most three literals per clause and at most three occurrences of each variable.
Question: Is ϕ unsatisfiable, yet removing any clause renders ϕ satisfiable?

Let ϕ be an instance of Is-MUS. Suppose that ϕ has m clauses $c_1, \ldots, c_m$, and ℓ variables $x_1, \ldots, x_\ell$. Without loss of generality, we assume that in each clause positive literals are followed by negative. Therefore there are four types of clauses, depending on the number of negative literals. We say that a clause c_j is of type C_{t_j} if there are $t_j \in \{0, 1, 2, 3\}$ negative literals in c_j. For example, the clause $(x_1 \vee \neg x_3 \vee \neg x_4)$ is of type C_2. We take the following ABox:

$$\mathcal{A} = \{C_{t_1}(c_1), \ldots, C_{t_m}(c_m), \mathit{Tau}(\tau), \mathit{Unsat}(\tau)\} \cup \{r_0(\tau, x_k) \mid 1 \leq k \leq \ell\} \cup$$
$$\{r_{p_i}(c_j, x_k) \mid x_k \text{ is the } i^{\text{th}} \text{ literal of } c_j\} \cup \{r_{n_i}(c_j, x_k) \mid \neg x_k \text{ is the } i^{\text{th}} \text{ literal of } c_j\}.$$

We take the candidate MinEX to be $\mathcal{E} = \mathcal{A} \setminus \{Unsat(\tau)\}$. For the construction of the TBox $\mathcal{T}$, we use the following abbreviations:

$$\Psi_0 = C_0 \sqcap \exists r_{p_1}.\top \sqcap \exists r_{p_2}.\top \sqcap \exists r_{p_3}.\top,$$
$$\Psi_1 = C_1 \sqcap \exists r_{p_1}.\top \sqcap \exists r_{p_2}.\top \sqcap \exists r_{n_3}.\top,$$
$$\Psi_2 = C_2 \sqcap \exists r_{p_1}.\top \sqcap \exists r_{n_2}.\top \sqcap \exists r_{n_3}.\top,$$
$$\Psi_3 = C_3 \sqcap \exists r_{n_1}.\top \sqcap \exists r_{n_2}.\top \sqcap \exists r_{n_3}.\top.$$

The formula Ψ_t encodes the fact that a clause is of type C_t and there is a *correctly* negated literal for each of its three positions.

We construct TBox $\mathcal{T}$ in three steps, giving an intuition for each of those components. The first component $\mathcal{T}_{lit}$ captures that at least one literal of a clause has to be satisfied. That is, if the assertions that encode a clause c_j are present, then the assertion $L_i(c_j)$ holds for some $i \in \{1, 2, 3\}$, where L_i encodes that the i^{th} literal takes value T:

$$\mathcal{T}_{lit} = \bigcup_{0 \leq t \leq 3} \{\Psi_t \sqsubseteq L_1 \sqcup L_2 \sqcup L_3\}.$$

The second component $\mathcal{T}_{var}$ ensures that if the literal takes the value *true* then the underlying variable takes the appropriate truth value T or F:

$$\mathcal{T}_{var} = \{L_1 \sqsubseteq \forall r_{p_1}.T \sqcap \forall r_{n_1}.F,\ L_2 \sqsubseteq \forall r_{p_2}.T \sqcap \forall r_{n_2}.F,\ L_3 \sqsubseteq \forall r_{p_3}.T \sqcap \forall r_{n_3}.F\},$$

The last component $\mathcal{T}_{inc}$ ensures that if some variable takes both values T and F, then the assignment is inconsistent:

$$\mathcal{T}_{inc} = \{Tau \sqcap \exists r_0.(T \sqcap F) \sqsubseteq Unsat\}.$$

Thus the TBox is $\mathcal{T} = \mathcal{T}_{lit} \cup \mathcal{T}_{var} \cup \mathcal{T}_{inc}$. We take the query to be $Q = Unsat(\tau)$. Since $Unsat(\tau)$ is in $\mathcal{A}$, we have that $\mathcal{A} \models (Q, \mathcal{T})$, and so $\mathcal{A}$, $\mathcal{E}$ and (Q, Σ), as constructed above, give an instance of Is-MinEX.

Now we show that $\mathcal{E}$ is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ iff ϕ is a MUS.

($\Rightarrow$) For the forward direction, suppose that $\mathcal{E}$ is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$. We show that ϕ must be a MUS. As in the proof of [125, Theorem A.1], we have that ϕ is unsatisfiable as, by construction, if $\mathcal{E} \models (Unsat(\tau), \mathcal{T})$, then any assignment to variables of ϕ yields *false*. It remains to show ϕ is minimal, that is, removing any clause c_j from ϕ gives a satisfiable formula. Note that removing $C_{t_j}(c_j), r_{p_i}(c_j, x_k)$ or $r_{n_i}(c_j, x_k)$ from $\mathcal{E}$ corresponds to removing the clause c_j from ϕ, as $(\Psi_{t_j}(c_j), \mathcal{T})$ is not entailed by $\mathcal{E}$ without any of these assertions. Removing $r_0(\tau, x_k)$ corresponds to

removing all clauses where x_k appears as a variable. Finally, if we remove $Tau(\tau)$ from $\mathcal{A}$, then $(Unsat(\tau), \mathcal{T})$ cannot be entailed. As, for all $\psi \in \mathcal{E}$, $\mathcal{E} \setminus \{\psi\} \not\models (Unsat(\tau), \mathcal{T})$, we have that removing any clause from ϕ yields a satisfiable formula.

($\Leftarrow$) Suppose that ϕ is a MUS. We show that $\mathcal{E}$ is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$. By construction, as in the proof of [125, Theorem A.1], we have that $\mathcal{E} \models (Q, \mathcal{T})$. It remains to show that $\mathcal{E}$ is minimal. Suppose, for a contradiction, that there is $\psi \in \mathcal{E}$ such that $\mathcal{E} \setminus \{\psi\} \models (Unsat(\tau), \mathcal{T})$. We have seen above that if removing a single assertion does not break the entailment, then there is a clause that can be removed from ϕ and the resulting formula is still unsatisfiable. But this is a contradiction to ϕ being a minimal unsatisfiable formula. □

Next, we show the D^{P}-hardness of Is-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L}_\emptyset)$ in combined complexity. Note that since BCQ is a special case of UCQ, and $\mathcal{L}_\emptyset$ is contained in every DL $\mathcal{L}$, the above implies that Is-MinEX$(\subseteq, \mathrm{UCQ}, \mathcal{L})$ is D^{P}-hard for every DL $\mathcal{L}$. We provide a reduction from the D^{P}-hard problem, 3Col-NonCol, of determining whether one graph is 3-colourable and another is not 3-colourable. The problem 3Col-NonCol can be shown to be D^{P}-hard by reduction from Sat-Unsat, using the same encoding as in the reduction of 3Sat to 3Col [147]. We show the D^{P}-hardness of Is-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L}_\emptyset)$ by encoding the two graphs in the query and the allowed colourings of the edges in the ABox. The following result generalises Theorem 3.12. On the other hand, the proof of Theorem 3.12 does not apply here as it uses the predicates of arity three. We leave both proofs in this thesis as they provide reductions from different D^{P}-complete problems and gives us a deeper insight into the complexity of these problems.

Theorem 4.13. Is-MinEX$(\subseteq, \mathrm{BCQ}, \mathcal{L}_\emptyset)$ *is* D^{P}*-hard in combined complexity.*

Proof. We provide a reduction from the D^{P}-complete problem 3Col-NonCol, defined below. To see that the following problem is D^{P}-hard note that the D^{P}-complete problem Sat-Unsat [129] can be reduced to 3Col-NonCol by using the same encodings as in the polynomial time reduction of 3Sat to 3Col [147].

Problem: 3Col-NonCol
Input: Graphs G and H.
Question: Is G 3-colourable and H not 3-colourable?

Let G and H be graphs with edge sets E_G and E_H, respectively. We construct an

instance of Is-MinEX($\subseteq$, BCQ, $\mathcal{L}_\emptyset$). First, we construct the ABox as follows:

$$\begin{aligned}\mathcal{A}_{st} = \{&c_G(r,g), c_G(g,r), c_G(r,b), c_G(b,r), c_G(g,b), c_G(b,g)\\ &c_H(r,g), c_H(g,r), c_H(r,b), c_H(b,r), c_H(g,b), c_H(b,g)\}\end{aligned}$$

and

$$\mathcal{A} = \mathcal{A}_{st} \cup \{c_G(w,w), c_H(w,w)\}\,.$$

We take the candidate MinEX to be $\mathcal{E} = \mathcal{A} \setminus \{c_G(w,w)\}$.

We take the TBox $\mathcal{T}$ to be an empty set. The query consists of three parts. The first part checks the containment of structural facts:

$$\mathit{Config} \equiv \bigwedge_{\psi \in \mathcal{A}_{st}} \psi,$$

and the second and third parts correspond to checking whether G and H are colourable with the given colours, respectively:

$$\begin{aligned}\mathit{Coloured}_G &\equiv \bigwedge_{(X_i,X_j)\in E_G} c_G(X_i,X_j), \text{ and}\\ \mathit{Coloured}_H &\equiv \bigwedge_{(Y_i,Y_j)\in E_H} c_H(Y_i,Y_j).\end{aligned}$$

The query is $Q = \mathit{Config} \wedge \mathit{Coloured}_G \wedge \mathit{Coloured}_H$. We observe that $\mathcal{A} \models (Q,\mathcal{T})$ as both $c_G(w,w)$ and $c_H(w,w)$ are in $\mathcal{A}$, and thus assigning to all the query variables the value w satisfies the OMQ $(Q,\mathcal{T})$. Therefore, the above gives an instance of Is-MinEX. Now we show that $\mathcal{E}$ is a MinEX for $\mathcal{A} \models (Q,\mathcal{T})$ iff G is 3-colourable and H is not 3-colourable.

($\Rightarrow$) Suppose that $\mathcal{E}$ is a MinEX for $\mathcal{A} \models (Q,\mathcal{T})$. We show that this implies that G is 3-colourable and H is not 3-colourable. Since $\mathcal{E} \models (Q,\mathcal{T})$, we have that $E \models (Q_G,\mathcal{T})$. This, by construction, implies that G is 3-colourable. Since $\mathcal{E}$ is minimal w.r.t the OMQ entailment, we have that $\mathcal{E} \setminus \{c_H(w,w)\} \not\models (Q,\mathcal{T})$. This implies that H is not 3-colourable.

($\Leftarrow$) For the converse, suppose that G is 3-colourable and H is not 3-colourable. First, we show that $\mathcal{E} \models (Q,\mathcal{T})$. Certainly $\mathcal{E} \models \mathit{Config}$ as $\mathcal{A}_{st} \subseteq \mathcal{E}$. As G is 3-colourable, assigning to the query variables X_i, corresponding to the vertices of G, the colouring of G, we have that $\mathcal{E} \models (Q_G,\mathcal{T})$. Also, $\mathcal{E}$ entails $(Q_H,\mathcal{T})$ as $c_H(w,w) \in \mathcal{A}$. It remains to show that $\mathcal{E}$ is minimal. Note that removing any assertion in $\mathcal{A}_{st}$ from $\mathcal{E}$ gives a set that does not entail $(\mathit{Config},\mathcal{T})$, and so $(Q,\mathcal{T})$. Therefore it remains to show that $\mathcal{A} \setminus \{c_H(w,w)\} \not\models (Q,\mathcal{T})$. In particular, as H is not 3-colourable, by construction, $\mathcal{A} \setminus \{c_H(w,w)\} \not\models (Q_H,\mathcal{T})$, and so $\mathcal{A} \setminus \{c_H(w,w)\} \not\models (Q,\mathcal{T})$. □

Table 4.2: Complexity results for All-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$.

Description Logic	IQ		UCQ	
	data	comb.	data	comb.
DL-Lite$_{core}$, *DL-Lite*$_{\mathcal{R}}$	$\leq$ P	$\leq$ P	$\leq$ P	D$^{\text{P}}$
$\mathcal{EL}, \mathcal{ELH}$	coNP	coNP	coNP	D$^{\text{P}}$
$\mathcal{ELI}$, Horn-$\mathcal{SHOIQ}$	coNP	Exp	coNP	Exp
$\mathcal{ALC}, \mathcal{ALCHQ}$	Π_2^{P}	Exp	Π_2^{P}	Exp
$\mathcal{ALCI}, \mathcal{SH}, \mathcal{SHIQ}$	Π_2^{P}	Exp	Π_2^{P}	2Exp

It remains to show the 2Exp-hardness results for Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$ in combined complexity. We show that, in combined complexity, OMQA$(\text{UCQ}, \mathcal{L})$ can be reduced to Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$ for any description logic $\mathcal{L}$. Note that the general reduction of OMQA to Is-MinEX for existential rules, given in Theorem 3.14, does not apply here as we cannot encode the ABox in the TBox in description logics.

Theorem 4.14. *For any DL $\mathcal{L}$ studied here, we have that* Is-MinEX$(\subseteq, \text{UCQ}, \mathcal{L})$ *in combined complexity is as hard as* OMQA$(\text{UCQ}, \mathcal{L})$ *in combined complexity.*

Proof. Let $\mathcal{A}$ be an ABox, $\mathcal{E} \subseteq \mathcal{A}$ be a subset and $(Q, \mathcal{T})$ be an $\mathcal{L}$-OMQ. We construct an instance of Is-MinEX as follows. We take the ABox $\mathcal{A}' = \mathcal{A} \cup \{T(a)\}$, where a is a fresh individual, and the candidate MinEX to be $\mathcal{E} = \mathcal{A}$. We take the query to be $Q' = (Q \wedge \bigwedge_{f \in \mathcal{A}} f) \vee T(a)$. Note that $\mathcal{A}' \models (Q', \mathcal{T})$, and thus it is an instance of Is-MinEX. It is easy to see that $\mathcal{A}$ is a MinEX for $\mathcal{A}' \models (Q', \mathcal{T})$ iff $\mathcal{A} \models (Q, \mathcal{T})$. □

4.4 All Minimal Explanations

Next, we analyse the complexity of the problem of recognising the set of all MinEXs and show the complexity results, presented in Table 4.2. We first provide the membership results and then proceed to give the matching hardness results. As earlier, we first provide a general approach to solve a problem. We observe that the general algorithm for All-MinEX under existential rules, provided in the proof of Theorem 3.15, applies here. Thus we have the following result.

Corollary 4.15. *For any DL $\mathcal{L}$, if* OMQA$(\text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* All-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$ *can be decided by a linear number of $\mathcal{C}$ checks and a single co-*$(\text{NP}^{\mathcal{C}})$ *check.*

The above result implies all the membership results in Table 4.2, apart from the membership results in P, in D^{P}, and Σ_2^{P}. We illustrate how some of the membership results in Table 4.2 follow. For example, if OMQA is in P, then All-MinEX is decidable by a linear number of P tests and a single co-$(\mathrm{NP}^{\mathrm{P}})$ = coNP. Note that coNP machine can express these P tests, and so All-MinEX is in coNP in this case. If OMQA is in coNP, then All-MinEX is decidable by a linear number of coNP tests and a single co-$(\mathrm{NP}^{\mathrm{coNP}}) = \Pi_2^{\mathrm{P}}$. Similarly as above these tests can be combined as a single Π_2^{P} test and so All-MinEX in this case is in Π_2^{P}. If OMQA is in $\mathcal{C} \in \{\mathrm{Exp}, \mathrm{2Exp}\}$, then All-MinEX is decidable by a linear number of tests in $\mathcal{C}$, followed by a single test in co-$(\mathrm{NP}^{\mathcal{C}})$. Note that, in this case, co-$(\mathrm{NP}^{\mathcal{C}})$ = co-$\mathcal{C} = \mathcal{C}$. It can be further easily seen that a linear number of tests in $\mathcal{C}$ can be captured by a single test in $\mathcal{C}$, and therefore All-MinEX, in this case, is in $\mathcal{C}$.

When OMQA is in NP, the membership of All-MinEX in D^{P} follows by the same argument as in the proof of Theorem 3.16.

Corollary 4.16. *For any DL $\mathcal{L}$ and $\mathcal{Q} \in \{\mathrm{IQ}, \mathrm{UCQ}\}$, if* OMQA$(\mathcal{Q}, \mathcal{L})$ *is in* NP*, then* All-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$ *is in* D^{P}*.*

Next, we deal with the case when OMQA is FO-rewritable. Note that whenever a DL $\mathcal{L}$ is FO-rewritable, we have that the proof of Theorem 3.18 applies, and we can construct all MinEXs in polynomial time in data complexity. Therefore, the following result is a corollary of Theorem 3.18 and [24, Proposition 4.8].

Corollary 4.17. *Let $\mathcal{L}$ be a DL for which* OMQA *is FO-rewritable and let $(Q, \mathcal{T})$ be an $\mathcal{L}$-OMQ. Then computing all MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$ is feasible in polynomial time in data complexity.*

The above implies that All-MinEX is in P in data complexity when the query is IQ or UCQ and the TBox is from *DL-Lite*$_{core}$ or *DL-Lite*$_{\mathcal{R}}$.

We proceed with the hardness results, presented in Table 4.2. We first show that All-MinEX$(\subseteq, \mathrm{IQ}, \mathcal{EL})$ is coNP-hard in data complexity. We show it by reducing a coNP-complete problem that has a very close nature to All-MinEX, namely, All-MV [12, Lemma 1]. This problem asks to decide if a given set $\mathfrak{V}$ contains *all* minimal valuations satisfying a monotone Boolean formula ϕ. In our reduction, the ABox $\mathcal{A}_\phi$ encodes ϕ, and $\mathcal{T}$ is empty. Each MinEX for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ corresponds to a minimal valuation for ϕ. This result implies all the coNP-hardness results for All-MinEX, presented in Table 4.2.

Theorem 4.18. All-MinEX$(\subseteq, \mathrm{IQ}, \mathcal{EL})$ *is* coNP*-hard in data complexity.*

Proof. We show the coNP-hardness of ALL-MINEX($\subseteq$, IQ, $\mathcal{EL}$) in data complexity by reduction from the coNP-hard problem ALL-MV [12, Lemma 1].

Problem: ALL-MV
Input: A monotone Boolean formula ϕ, a set $\mathfrak{V}$ of minimal valuations satisfying ϕ.
Question: Is $\mathfrak{V}$ precisely the set of all minimal valuations satisfying ϕ?

Let ϕ be a formula, $\mathfrak{V}$ be a set of minimal valuations for ϕ, and $sub(\phi)$ be the set of all subformulas of ϕ. The ABox consists of two parts. The first part of the ABox contains the assertions that capture whether a variable of ϕ is assigned the value *true*:

$$\mathcal{A}_x = \{T(x) \mid x \text{ is a variable in } \phi\},$$

and the second part encodes the formula ϕ:

$$\mathcal{A}_\phi = \bigcup_{\psi_1 \wedge \psi_2 \in sub(\phi)} \{c_1(\psi_1 \wedge \psi_2, \psi_1), c_2(\psi_1 \wedge \psi_2, \psi_2)\} \bigcup_{\psi_1 \vee \psi_2 \in sub(\phi)} \{d(\psi_1 \vee \psi_2, \psi_1), d(\psi_1 \vee \psi_2, \psi_2)\} \cup \{e(t, \phi)\}.$$

Thus the ABox is $\mathcal{A} = \mathcal{A}_x \cup \mathcal{A}_\phi$. We take the TBox to be

$$\mathcal{T} = \{\exists c_1.T \sqcap \exists c_2.T \sqsubseteq T, \exists d.T \sqsubseteq T, \exists e.T \sqsubseteq T\},$$

and the query to be $Q = T(t)$. Note that $\mathcal{A} \models (Q, \mathcal{T})$ as ϕ is a monotone formula.

The candidate set $\mathfrak{E}_\mathfrak{V}$ of all MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$ is constructed as follows. For each valuation $\mathcal{V}$, we inductively construct the sets $satsub(\phi)$ (of satisfied subformulas of ϕ by $\mathcal{V}$) and a subset $\mathcal{E}_\mathcal{V}$ of $\mathcal{A}$. Let $\mathcal{V}$ be a minimal valuation of ϕ. Initially, we take $satsub(\phi) = \{x \mid \mathcal{V}(x) = T\}$ to be a subset of $sub(\phi)$ and $\mathcal{E}_\mathcal{V} = \{T(x) \mid \mathcal{V}(x) = T\}$. The proceed as follows:

(i) If $\psi \in satsub(\phi)$ and $d(\psi \vee \psi', \psi) \in \mathcal{A}_\phi$, add $d(\psi \vee \psi', \psi)$ to $\mathcal{E}_\mathcal{V}$ and $\psi \vee \psi'$ to $satsub(\phi)$;

(ii) If $\psi \in satsub(\phi)$ and $d(\psi' \vee \psi, \psi) \in \mathcal{A}_\phi$, add $d(\psi' \vee \psi, \psi)$ to $\mathcal{E}_\mathcal{V}$ and $\psi' \vee \psi$ to $satsub(\phi)$;

(iii) If $\psi, \psi' \in satsub(\phi)$ and $c_1(\psi \wedge \psi', \psi), c_2(\psi \wedge \psi', \psi') \in \mathcal{A}_\phi$, add $c_1(\psi \wedge \psi', \psi), c_2(\psi \wedge \psi', \psi')$ to $\mathcal{E}_\mathcal{V}$, and $\psi \wedge \psi'$ to $satsub(\phi)$.

We finish if no new facts can be added to $\mathcal{E}_\mathcal{V}$. Note that the construction of $\mathcal{E}_\mathcal{V}$ is polynomial in the size of $\mathcal{A}$ as $\mathcal{E}_\mathcal{V} \subseteq \mathcal{A}$ and deciding if any fact can be added to $\mathcal{E}_\mathcal{V}$ takes polynomial time in the size of $\mathcal{A}$. We take $\mathfrak{E}_\mathfrak{V} = \{\mathcal{E}_\mathcal{V} \mid \mathcal{V} \in \mathfrak{V}\}$. We show

that $\mathfrak{V}$ is the set of all minimal valuations for ϕ iff $\mathfrak{E}_{\mathfrak{V}}$ is the set of all MinEXs for $\mathcal{A}_\phi \models (Q, \mathcal{T})$.

($\Rightarrow$) Suppose that $\mathfrak{V}$ is the set of all minimal valuations. We need to show that $\mathfrak{E}_{\mathfrak{V}}$ is the set of all MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$. For this we need to show that (i) $\mathcal{E}_\mathcal{V} \models (Q, \mathcal{T})$ for all $\mathcal{V} \in \mathfrak{V}$, and (ii) there is no $\mathcal{E} \subseteq \mathcal{A}$ such that $\mathcal{E}_\mathcal{V} \not\subseteq \mathcal{E}$ for all $\mathcal{V} \in \mathfrak{V}$, and $\mathcal{E} \models (Q, \mathcal{T})$. By construction, we have (i). It remains to show (ii). Suppose, for a contradiction, that there is $\mathcal{E} \subseteq \mathcal{A}$ that satisfies the conditions. But then, by construction, $\mathcal{E}$ gives a valuation, which (or any of its sub-valuations) is not contained in $\mathfrak{V}$ (or any of its sub-valuations). This is a contradiction.

($\Leftarrow$) For the converse, suppose that $\mathfrak{E}_{\mathfrak{V}}$ is the set of all MinEXs for $\mathcal{A} \models (Q, \mathcal{T})$. Note that by construction it implies that each $\mathcal{V} \in \mathfrak{V}$ is a minimal valuation. It remains to show that $\mathfrak{V}$ contains *all* minimal valuations of ϕ. Suppose, for a contradiction, that there is a minimal valuation $\mathcal{V}' \notin \mathfrak{V}$. But then, by construction, $\mathcal{E}_{\mathcal{V}'}$ is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ that is not in $\mathfrak{E}_{\mathfrak{V}}$, which is a contradiction. □

Note that this result strengthens Theorem 3.19 given in Chapter 3 for existential rules, since $\mathcal{EL}$ can be embedded in the guarded full language of existential rules.

We now show Π_2^{p}-hardness of All-MinEX($\subseteq$, IQ, $\mathcal{ALC}$) in data complexity, which is the most intricate result of this section and builds on the ideas presented in the proof of Theorem 4.12.

Theorem 4.19. All-MinEX($\subseteq$, IQ, $\mathcal{ALC}$) *is* Π_2^{p}*-hard in data complexity.*

Proof. In this reduction, we use the Π_2^{p}-complete problem $\forall\exists$3SAT [148, 156].

Problem: $\forall\exists$3SAT
Input: Boolean formula $\phi(X, Y)$ in 3CNF.
Question: Is it true that $\Phi = \forall X\, \exists Y\, \phi(X, Y)$ holds?

Let $\phi(X, Y)$ be an instance of $\forall\exists$3SAT. Say $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_m\}$ and $\phi = c_1 \wedge \cdots \wedge c_\ell$.

This reduction is based on the proof of Theorem 4.12. Let $\phi(X, Y)$ be a 3CNF formula. We construct the ABox in two steps. The first components consists of the

structural facts that capture the structure of ϕ:

$$\begin{aligned}\mathcal{A}_{st} = &\{C(c_1), \ldots, C(c_\ell), Tau(\tau)\} \\ &\cup \{r_0(\tau, x_k) \mid 1 \le k \le n\} \cup \{r_0(\tau, y_k) \mid 1 \le k \le m\} \\ &\cup \left\{r_{p_i}(c_j, x_k) \mid i \le 3, j \le \ell, k \le n, x_k \text{ is } i^{\text{th}} \text{ literal of } c_j\right\} \\ &\cup \left\{r_{n_i}(c_j, x_k) \mid i \le 3, j \le \ell, k \le n, \neg x_k \text{ is } i^{\text{th}} \text{ literal of } c_j\right\} \\ &\cup \left\{r_{p_i}(c_j, y_k) \mid i \le 3, j \le \ell, k \le m, y_k \text{ is } i^{\text{th}} \text{ literal of } c_j\right\} \\ &\cup \left\{r_{n_i}(c_j, y_k) \mid i \le 3, j \le \ell, k \le m, \neg y_k \text{ is } i^{\text{th}} \text{ literal of } c_j\right\},\end{aligned}$$

and the second component encodes the truth value assignment to x_i variables:

$$\mathcal{A}_x = \{T(x_i), F(x_i) \mid 1 \le i \le n\}.$$

Take $\mathcal{A}_\phi = \mathcal{A}_{st} \cup \mathcal{A}_x$. The TBox is constructed in three steps. The first component ensures that at least one of the literals in each clause is satisfied:

$$\mathcal{T}_{lit} = \{C \sqsubseteq L_1 \sqcup L_2 \sqcup L_3\}.$$

The second component ensures that, if $L_i(c_j)$ holds, then the underlying variable takes the appropriate truth value T or F:

$$\mathcal{T}_{var} = \{L_1 \sqsubseteq \forall r_{p_1}.T \sqcap \forall r_{n_1}.F,\ L_2 \sqsubseteq \forall r_{p_2}.T \sqcap \forall r_{n_2}.F,\ L_3 \sqsubseteq \forall r_{p_3}.T \sqcap \forall r_{n_3}.F\}.$$

The last component ensures that if, in a satisfying assignment, some variable takes both values T and F, then the assignment is inconsistent:

$$\mathcal{T}_{inc} = \{Tau \sqcap \exists r_0.(T \sqcap F) \sqsubseteq Unsat\}.$$

We take the query to be $Q = Unsat(\tau)$, and the candidate set of all MinEXs for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ to be $\mathfrak{E} = \{\{Tau(\tau), r_0(\tau, x_i), T(x_i), F(x_i)\} \mid 1 \le i \le n\}$. Note that any $\mathcal{E} \in \mathfrak{E}$ entails $(Q, \mathcal{T})$.

Now we show that $\mathfrak{E}$ is the set of all MinEXs for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ iff $\forall X\, \exists Y\, \phi(X, Y)$ is true.

($\Rightarrow$) Suppose that $\forall X\, \exists Y\, \phi(X, Y)$ does not hold, that is, $\exists X\, \forall Y\, \neg\phi(X, Y)$ holds. Let v_X be an assignment to X such that, for any assignment v_Y to Y, $\phi(v_X(X), v_Y(Y))$ is false. Then $\mathcal{E} = \mathcal{A}_{st} \cup \{T(x) \mid v_X(x) = T\} \cup \{F(x) \mid v_X(x) = F\}$ contains a MinEX for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ that is not contained in $\mathfrak{E}$. Thus $\mathfrak{E}$ is not the set of *all* MinEXs for $\mathcal{A}_\phi \models (Q, \mathcal{T})$.

($\Leftarrow$) Suppose that $\mathfrak{E}$ is not the set of all MinEXs for $\mathcal{A}_\phi \models (Q, \mathcal{T})$. We show that $\forall X\, \exists Y\, \phi(X, Y)$ does not hold, that is, $\exists X\, \forall Y\, \neg\phi(X, Y)$ holds. Let $\mathcal{E}$ be a MinEX

Table 4.3: Complexity results for MinEX-Irrel$(\subseteq, \mathcal{Q}, \mathcal{L})$.

Description Logic	IQ		UCQ	
	data	comb.	data	comb.
DL-Lite$_{core}$, *DL-Lite*$_{\mathcal{R}}$	$\leq$ P	$\leq$ P	$\leq$ P	NP
$\mathcal{EL}, \mathcal{ELH}$	NP	NP	NP	NP
$\mathcal{ELI}$, Horn-$\mathcal{SHOIQ}$	NP	Exp	NP	Exp
$\mathcal{ALC}, \mathcal{ALCHQ}$	Σ_2^{P}	Exp	Σ_2^{P}	Exp
$\mathcal{ALCI}, \mathcal{SH}, \mathcal{SHIQ}$	Σ_2^{P}	Exp	Σ_2^{P}	2Exp

for $\mathcal{A}_\phi \models (Q, \mathcal{T})$, not in $\mathfrak{E}$. First, note that $Tau(\tau) \in \mathcal{E}$ as otherwise $(Unsat(\tau), \mathcal{T})$ cannot be entailed. Also, as $\mathcal{E} \notin \mathfrak{E}$, $\{r_0(\tau, x), T(x), F(x)\} \not\subseteq \mathcal{E}$ for any $x \in X$. Thus $\mathcal{E}$ gives a proper assignment v_X to X (or a subset of X) such that $v_X(x) = T$ if $T(x) \in \mathcal{E}$ and $v_X(x) = F$ if $F(x) \in \mathcal{E}$. As $\mathcal{E} \models (Unsat(\tau), \mathcal{T})$, by construction, we have that whatever truth values we assign v_Y to variables in Y, the formula $\phi(v_X(X), v_Y(Y))$ is false.

□

The remaining hardness results, such as Exp- or 2Exp-hardness, follow immediately from the proofs of the Is-MinEX results by taking the set of all MinEXs to contain the only one constructed MinEX. This concludes our analysis of All-MinEX.

4.5 Irrelevant Facts for Explanations

In this section, we analyse the problem MinEX-Irrel of deciding if there is a MinEX that does not contain any of the forbidden sets of assertions. As before, we show a generic membership result that follows from our work on existential rules, in particular, from Theorem 3.20. MinEX-Irrel can be solved by first guessing in NP a set $\mathcal{E}$ not containing any of the forbidden sets, and then checking in $\mathcal{C}$ whether $\mathcal{E}$ entails the OMQ. If $\mathcal{E}$ entails the OMQ, then it must contain a MinEX as a subset.

Corollary 4.20. *For any DL $\mathcal{L}$ and query language $\mathcal{Q}$, if* OMQA$(\mathcal{Q}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* MinEX-Irrel$(\subseteq, \mathcal{Q}, \mathcal{L})$ *is in* NP$^{\mathcal{C}}$.

The above theorem covers all the membership results, presented in Table 4.3, except for the P-membership results and the NP-membership results for UCQs in combined complexity. For example, if OMQA is in coNP, then MinEX-Irrel is in NP$^{\text{coNP}} = \Sigma_2^{\text{P}}$. If OMQA is in P, then MinEX-Irrel is in NP$^{\text{P}}$ = NP, and, if OMQA is in $\mathcal{C} \in \{\text{Exp}, \text{2Exp}\}$, then MinEX-Irrel is in NP$^{\mathcal{C}} = \mathcal{C}$. Now we cover

the remaining membership results. First, note that if OMQA is in NP, then Theorem 3.21 applies, and thus the NP-membership results for UCQs in combined complexity follow. In particular, since the two NP tests are *independent*, they can be carried out by a single NP machine whose certificate is obtained by concatenating the two NP certificates. Therefore, MinEX-Irrel is in NP in that case.

Corollary 4.21. *For any DL $\mathcal{L}$ and query language $\mathcal{Q}$, if* OMQA$(\mathcal{Q}, \mathcal{L})$ *is in* NP*, then* MinEX-Irrel$(\subseteq, \mathcal{Q}, \mathcal{L})$ *is in* NP*.*

The polynomial time upper bounds for FO-rewritable languages *DL-Lite*$_{core}$ and *DL-Lite*$_{\mathcal{R}}$ in data complexity follow from Corollary 4.17. We can construct the set of *all* MinEXs in polynomial time by Corollary 4.17, and then check in polynomial time if this set contains a MinEX that does not contain any forbidden set.

Corollary 4.22. *Let $\mathcal{L}$ be a DL for which* OMQA *is FO-rewritable and let $(Q, \mathcal{T})$ be an $\mathcal{L}$-OMQ. Then computing a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ that does not contain a forbidden set of assertions $\mathfrak{F}$ is feasible in polynomial time in data complexity.*

We proceed with the hardness results for MinEX-Irrel. First, we show the NP-hardness for MinEX-Irrel$(\subseteq, \text{IQ}, \mathcal{EL})$ in data complexity. We note that the exact same reduction works as in the proof of Theorem 3.24 when restated in the DL terminology.

Corollary 4.23. MinEX-Irrel$(\subseteq, \text{IQ}, \mathcal{EL})$ *is* NP*-hard in data complexity.*

Proof. We reduce an NP-complete problem Path with Forbidden Pairs [79, 81]. to MinEX-Rel as in the proof of Theorem 3.24.

Let $G = (V, E)$ be a graph with two vertices s, $t \in V$ and let $\mathcal{F} \subseteq E \times E$ be a set. We restate the construction in the DL terminology and refer the reader to the proof of Theorem 3.24 to see why this reduction works. We take $\mathcal{A} = \{r(v, u) \mid (u, v) \in E\} \cup \{r(\tau, t)\} \cup \{P(s)\}$, $\mathcal{T} = \{\exists r.P \sqsubseteq P\}$, $Q = P(\tau)$, and $\mathfrak{F} = \{\{r(v_1, u_1), r(v_2, u_2)\} \mid \{(u_1, v_1), (u_2, v_2)\} \in \mathcal{F}\}$. We have that there is a MinEX $\mathcal{E}$ for $\mathcal{A} \models (Q, \mathcal{T})$ not containing any set in $\mathfrak{F}$ iff there is a simple path in G from s to t not using any pair of edges in $\mathcal{F}$. □

It remains to show that MinEX-Irrel$(\subseteq, \text{IQ}, \mathcal{ALC})$ is Σ_2^{p}-hard in data complexity, which implies all Σ_2^{p}-hardness results in Table 4.3.

Theorem 4.24. MinEX-Irrel$(\subseteq, \text{IQ}, \mathcal{ALC})$ *is* Σ_2^{p}*-hard in data complexity.*

Proof. We use the reduction from the following Σ_2^{p}-complete problem.

Problem: $\text{QBF}_{2,\forall,\neg}^{CNF}$

Input: A quantified Boolean formula $\Phi = \exists X \forall Y \, \neg\phi(X, Y)$, with X and Y disjoint sets of Boolean variables, and ϕ a 3CNF formula.

Question: Is Φ valid?

Let $\Phi = \exists X \forall Y \neg\phi(X, Y)$ be an instance of $\text{QBF}_{2,\forall,\neg}^{CNF}$. Note that Φ is valid iff there is an assignment to X such that, for all assignments to Y, the formula $\phi(X, Y)$ does *not* hold. In this reduction, for ϕ, we construct the ABox $\mathcal{A}_\phi$ and the OMQ $(Q, \mathcal{T})$ as in the proof of Theorem 4.19. We just take $\mathfrak{F} = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$, where $\mathcal{F}_i = \{T(x_i), F(x_i)\}$.

Now we show that $\exists X \forall Y \neg\phi(X, Y)$ iff there is a MinEX for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ not containing any set in $\mathfrak{F}$.

($\Rightarrow$) Suppose that $\exists X \forall Y \neg\phi(X, Y)$ holds. Therefore, there is an assignment to X such that, for all assignments to Y, the formula $\phi(X, Y)$ does *not* hold. Let v_X be such an assignment to X. Add to $\mathcal{E}$ all facts in $\mathcal{A}_{st}$. Further, add the facts from $\mathcal{A}_x$ to $\mathcal{E}$ to simulate the assignment v_X, that is, take $T(x_i)$ if $v_X(x_i) = T$ and $F(x_i)$ if $v_X(x_i) = F$. We know that, for any assignment v_Y to Y, $\phi(v_X(X), v_Y(Y))$ does *not* hold. By construction, we have that for any model $\mathcal{I}$ of $(\mathcal{E}, \mathcal{T})$, as it encodes v_X, we have that $\mathcal{I} \models \mathit{Unsat}(\tau)$, and so $\mathcal{E}$ contains a MinEX for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ not containing any set in $\mathfrak{F}$. Thus such a MinEX exists.

($\Leftarrow$) For the other direction, suppose that there is a MinEX $\mathcal{E}$ for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ that does not contain any set in $\mathfrak{F}$. Let $\mathcal{E}$ be such a MinEX. Let $\mathcal{E}'$ be possibly an extension of $\mathcal{E}$ such that exactly one element ($T(x_i)$ or $F(x_i)$) from each $\mathcal{F} \in \mathfrak{F}$ is in $\mathcal{E}'$ and $\mathcal{A}_{st} \subset \mathcal{E}'$. Note that $\mathcal{E}'$ does not contain any $\mathcal{F} \in \mathfrak{F}$ and $\mathcal{E}' \models (Q, \mathcal{T})$ as $\mathcal{E} \subseteq \mathcal{E}'$. Let v_X be an assignement to X given by $\mathcal{E}'$, that is, $v_X(x_i) = T$ if $T(x_i) \in \mathcal{E}'$ and $v_X(x_i) = F$ if $F(x_i) \in \mathcal{E}'$. In particular, what it means that, for any assignment v_Y to Y, we have that $\phi(v_X(X), v_Y(Y))$ does not hold, and therefore $\exists X \forall Y \neg\phi(X, Y)$ holds. □

Finally, we show that MinEX-Irrel is as hard as OMQA. This follows by the same argument as the reduction in the proof of Theorem 3.25 of OMQA to MinEX-Irrel for existential rules. We observe that the reduction in the proof of Theorem 3.25 only changes the data, and thus it works both for data and combined complexities, while in this case, we require the hardness results in combined complexity to complete the complexity picture in Table 3.3.

Table 4.4: Complexity results for MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$.

Description Logic	IQ		UCQ	
	data	comb.	data	comb.
DL-Lite$_{core}$, *DL-Lite*$_{\mathcal{R}}$	$\leq$ P	$\leq$ P	$\leq$ P	Σ_2^{P}
$\mathcal{EL}, \mathcal{ELH}$	NP	NP	NP	Σ_2^{P}
$\mathcal{ELI}$, Horn-$\mathcal{SHOIQ}$	NP	Exp	NP	Exp
$\mathcal{ALC}, \mathcal{ALCHQ}$	Σ_2^{P}	Exp	Σ_2^{P}	Exp
$\mathcal{ALCI}, \mathcal{SH}, \mathcal{SHIQ}$	Σ_2^{P}	Exp	Σ_2^{P}	2Exp

Corollary 4.25. *For any DL $\mathcal{L}$ and query language $\mathcal{Q}$,* MinEX-Irrel$(\subseteq, \mathcal{Q}, \mathcal{L})$ *in combined complexity is as hard as* OMQA$(\mathcal{Q}, \mathcal{L})$ *in combined complexity.*

4.6 Relevant Facts for Explanations

The last problem that we investigate is related to the role of an individual assertion in explaining OMQA. This problem, MinEX-Rel, asks whether there is a MinEX containing a distinguished assertion.

As usual, we first provide the membership results and then proceed to give the matching hardness results. First, for showing the membership results, we note that the same algorithm as proposed in the proof of Theorem 3.26 applies here. Therefore, we can solve MinEX-Rel by guessing a subset of the ABox that contains a distinguished fact (in NP), and then checking if it is a MinEX using an oracle for Is-MinEX. Thus we have the following corollary.

Corollary 4.26. *If* Is-MinEX$(\subseteq, \mathcal{Q}, \mathcal{L})$ *is in $\mathcal{C}$, then* MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$ *is in* $\text{NP}^{\mathcal{C}}$.

This result covers all membership results, presented in Table 4.4, apart from the membership in P. For example, if Is-MinEX is in D^{P}, then MinEX-Rel can be decided in $\text{NP}^{\text{D}^{\text{P}}} = \Sigma_2^{\text{P}}$. Other results follow similarly.

We observe that we have that MinEX-Rel for FO-rewritable languages in polynomial time in data complexity as a consequence of Corollary 4.17. We can first compute the set of all MinEXs in polynomial time by Corollary 4.17 and then check if any of these MinEXs contains a distinguished assertion, which yields the following result.

Corollary 4.27. *Let $\mathcal{L}$ be a DL for which* OMQA *is FO-rewritable and let $(Q, \mathcal{T})$ be an $\mathcal{L}$-OMQ. Then computing a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ that contains a distinguished assertion ψ is feasible in polynomial time in data complexity.*

Next, we provide the matching lower bounds. Our next result concerns the problem MinEX-Rel$(\subseteq, \text{IQ}, \mathcal{EL})$, and we show that this is NP-hard in data complexity using the same reduction as in the proof of Theorem 3.28 restated using the standard DL terminology. We refer the reader to that proof for more details.

Corollary 4.28. MinEX-Rel$(\subseteq, \text{IQ}, \mathcal{EL})$ *is* NP*-hard in data complexity.*

Proof. We provide a reduction from an NP-complete problem Path via node [111] as in the proof of Theorem 3.28, stated in the standard DL terminology.

Let G be a graph and $s, t, m \in V$, which together form an instance of Path via node. We construct an instance of MinEX-Rel as follows. We take $\mathcal{A} = \{r(v,u) \mid (u,v) \in E, u, v \neq m\} \cup \{r(m_1, u) \mid (u,m) \in E, u \neq m\} \cup \{r(v, m_2) \mid (m,v) \in e, m \neq v\} \cup \{r(m_2, m_1)\} \cup \{r(\tau, t)\} \cup \{P(s)\}$, $\mathcal{T} = \{\exists r.P \sqsubseteq P\}$, $Q = P(\tau)$, and $\psi = r(m_2, m_1)$.

There is a MinEX $\mathcal{E}$ for $\mathcal{A} \models (Q, \mathcal{T})$ containing $\psi = r(m_2, m_1)$ iff there is a simple path in G from s to t that crosses m. □

Next, we show that MinEX-Rel is hard for a class in the second level of the polynomial hierarchy already in data complexity. We show that MinEX-Rel$(\subseteq, \text{IQ}, \mathcal{ALC})$ is Σ_2^{P}-hard in data complexity. This implies all the Σ_2^{P}-hardness results in data complexity in Table 4.4. The proof of the following theorem borrows ideas from the proof of Theorem 4.24

Theorem 4.29. MinEX-Rel$(\subseteq, \text{IQ}, \mathcal{ALC})$ *is* Σ_2^{P}*-hard in data complexity.*

Proof. We reduce a Σ_2^{P}-complete problem $\text{QBF}_{2,\forall,\neg}^{CNF}$ to MinEX-Rel$(\subseteq, \text{IQ}, \mathcal{ALC})$ in data complexity. Let $\Phi = \exists X \forall Y\, \neg\phi(X, Y)$ be an instance of $\text{QBF}_{2,\forall,\neg}^{CNF}$, where ϕ is 3CNF, $X = (x_1, \ldots, x_n)$ and $Y = (y_1, \ldots, y_\ell)$.

Note that checking the validity of $\Phi = \exists X \forall Y\, \neg\phi(X, Y)$ is tantamount to checking whether there exists a truth assignment v_X to X such that $\phi(v_X(x), Y)$ is not satisfiable.

In this reduction, for ϕ, we take the ABox $\mathcal{A}_\phi$ and the OMQ $(Q, \mathcal{T})$ as in the proof of Theorem 4.19 with two variations. Instead of the set $\mathcal{A}_x = \{T(x_i), F(x_i) \mid x_i \text{ is a variable in } X\}$, we have $\mathcal{A}_x = \{T(x_i) \mid x_i \text{ is a variable in } X\}$, and we have an extra TBox axiom $(\neg T \sqsubseteq F)$ in $\mathcal{T}'$. We take the distinguished fact ψ to be $Tau(\tau)$.

Now we show that Φ is valid iff there is a MinEX for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ that contains a distinguished assertion $\mathit{Tau}(\tau)$.

($\Rightarrow$) Suppose that Φ is valid. Let v_X be an assignment to X such that $\phi(v_X(X), Y)$ is not satisfiable. Then $\mathcal{E}_X = \mathcal{A}_{st} \cup \{T(x_i) \mid v_X(x_i) = T\} \models (Q, \mathcal{T})$, and so $\mathcal{E}_X$ contains as a subset a MinEX for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ with a distinguished fact $\mathit{Tau}(\tau)$.

($\Leftarrow$) Suppose that there is a MinEX $\mathcal{E}$ for $\mathcal{A}_\phi \models (Q, \mathcal{T})$ containing $\mathit{Tau}(\tau)$. Let v_X be an assignment to X constructed as follows, if $T(x_i) \in \mathcal{E}$, set $v_X(x_i) = T$, and, if $T(x_i) \notin \mathcal{E}$, set $v_X(x_i) = F$. For other $x_i \in X$, assign any truth value. By construction, we have that, for any assignment $v_Y(Y)$ to Y, $\phi(v_X(X), v_Y(Y))$ is false. Thus the result holds. □

Finally, we show the Σ_2^{P}-hardness results in combined complexity, where the complexity is one level higher in the polynomial hierarchy, as compared to the complexity of OMQA in the respective languages. We reduce the Σ_2^{P}-hard problem $\mathrm{QBF}_{2,\forall,\neg}^{CNF}$ to MinEX-Rel$(\subseteq, \mathrm{UCQ}, \mathcal{L})$ in combined complexity, where $\mathcal{L}$ may be any DL, as we take the TBox to be empty in the reduction. We note that the following generalises Theorem 3.29 as it only uses predicates of arity at most two.

Theorem 4.30. MinEX-Rel$(\subseteq, \mathrm{UCQ}, \mathcal{L}_\emptyset)$ *is* Σ_2^{P}*-hard in combined complexity.*

Proof. We provide a reduction from the Σ_2^{P}-complete problem $\mathrm{QBF}_{2,\forall,\neg}^{CNF}$. Let $\Phi = \exists X \forall Y\, \neg\phi(X, Y)$ be an instance of $\mathrm{QBF}_{2,\forall,\neg}^{CNF}$, where ϕ is a conjunction of clauses $c_1, \ldots, c_k$. We assume that a clause c_i is $\ell_1^i \vee \ell_2^i \vee \ell_3^i$, where ℓ_j^i is a literal with a variable from X or Y.

We construct the ABox as follows. For each variable $x \in X$, we add to the ABox the facts $\mathit{Val}_x(t)$ and $\mathit{Val}_x(f)$:

$$\mathcal{A}_{val} = \{\, \mathit{Val}_x(t),\, \mathit{Val}_x(f) \mid x \in X \text{ is a variable in } \phi \},$$

where a constant t stands for *true*, and f for *false*. Also, we add structural facts:

$$\mathcal{A}_{st} = \{c(t, f), c(f, t), c(f, b), c(b, f), c(t, b), c(b, t)\}.$$

Further, we add $c(b, b)$ to obtain the ABox:

$$\mathcal{A} = \mathcal{A}_{val} \cup \mathcal{A}_{st} \cup \{c(b, b)\},$$

where we fix $\{c(b, b)\}$ as the distinguished fact. We take the TBox to be empty, that is, $\mathcal{T} = \emptyset$.

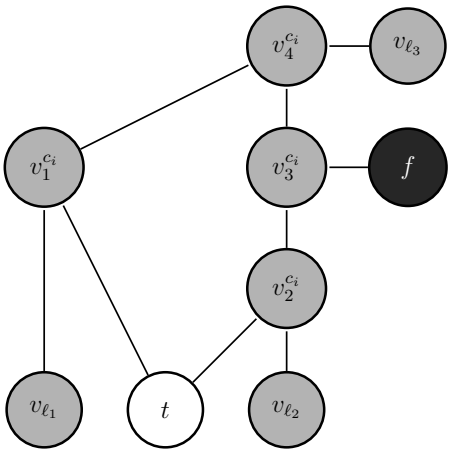

Figure 4.2: Graph of the construction of $clause_i(v_{\ell_1^i}, v_{\ell_2^i}, v_{\ell_3^i})$.

The query is rather complicated; therefore, we describe it in parts. The variables, appearing in the query, are $V_X = \{v_x, v_{\neg x} \mid x \in X\}, V_Y = \{v_y, v_{\neg y} \mid y \in Y\}$, and $V_C = \{v_1^{c_i}, v_2^{c_i}, v_3^{c_i}, v_4^{c_i} \mid c_i \text{ is a clause in } \phi\}$.

The first part of the query ensures that at least one of the facts $Val_x(t)$ and $Val_x(f)$ is present:

$$Val = \bigwedge_{x \in X} Val_x(v_x).$$

The second part of the query ensures that all the assertions in $\mathcal{A}_{st}$ are present, that is,

$$Config = \bigwedge_{\theta \in \mathcal{A}_{st}} \theta.$$

We next have to encode ϕ. If $c(b,b)$ is not in an explanation, the following structural encoding $clause_i$ simulates the ternary clause c_i of ϕ, with only binary relations. In particular, for each clause $c_i = \ell_1^i \vee \ell_2^i \vee \ell_3^i$ of ϕ with literals ℓ_1^i, ℓ_2^i, and ℓ_3^i, we define $clause_i(v_{\ell_1^i}, v_{\ell_2^i}, v_{\ell_3^i})$ to be

$$\begin{aligned} clause_i(v_{\ell_1^i}, v_{\ell_2^i}, v_{\ell_3^i}) = c(v_{\ell_1^i}, v_1^{c_i}) \wedge c(t, v_1^{c_i}) \wedge c(t, v_2^{c_i}) \wedge\ & c(v_{\ell_2^i}, v_2^{c_i}) \wedge c(f, v_3^{c_i}) \wedge \\ & c(v_{\ell_3^i}, v_4^{c_i}) \wedge\ c(v_1^{c_i}, v_4^{c_i}) \wedge c(v_2^{c_i}, v_3^{c_i}) \wedge c(v_3^{c_i}, v_4^{c_i}). \end{aligned}$$

This construction ensures that assigning f to *all* variables in $clause_i$ is not allowed. That is, $\exists v_1^{c_i}, v_2^{c_i}, v_3^{c_i}, v_4^{c_i}\ clause_i(f,f,f)$ does not hold in $\mathcal{A}_{val} \cup \mathcal{A}_{st}$ (if $c(b,b)$ is not present in an explanation). Note that, on the other hand, if $c(b,b)$ is present in an explanation, then any assignment to the variables $v_{\ell_1^i}, v_{\ell_2^i}, v_{\ell_3^i}$ satisfies $clause_i$.

The third and fourth parts of the query capture the structure of the formula ϕ and, if $c(b,b)$ is not present in an explanation, that the variables corresponding to

literals in the query are consistent and can take only values t or f. More specifically, the third component captures every clause of the formula:

$$\mathit{Satisfied} = \bigwedge_{i=1}^{k} \mathit{clause}_i(v_{\ell_1^i}, v_{\ell_2^i}, v_{\ell_3^i}).$$

Furthermore, whenever $c(b,b)$ is not in an explanation, the fourth component ensures that the variables in $V_X \cup V_Y$ corresponding to the opposite literals in ϕ take the opposite values (t and f) and cannot be assigned b:

$$\mathit{Consistent} = \bigwedge_{z \in X \cup Y} \left(c(v_z, v_{\neg z}) \wedge c(v_z, b) \wedge c(v_{\neg z}, b)\right).$$

Finally, we note that if $c(b,b)$ *is* present in a MinEX, then any clause_i is trivially satisfied by any assignment (even if we were to map them all to f).

The query is

$$Q = \exists V_X, V_Y, V_C \;\; \mathit{Val} \wedge \mathit{Config} \wedge \mathit{Satisfied} \wedge \mathit{Consistent}.$$

We now prove that $\exists X \forall Y \, \neg\phi(X,Y)$ holds iff there is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$ containing $c(b,b)$.

($\Rightarrow$) Suppose that $\exists X \forall Y \, \neg\phi(X,Y)$ holds. This implies that there is an assignment σ_X to X such that $\exists Y \, \phi(\sigma_X(X), Y)$ does *not* hold. Based on this assignment, we define the set $\mathcal{E} = \{\mathit{Val}_x(f) \mid \sigma_X(x) = 0\} \cup \{\mathit{Val}_x(t) \mid \sigma_X(x) = 1\} \cup \mathcal{A}_{st} \cup \{c(b,b)\}$ and show that it is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$. Observe first that $\mathcal{E} \models (Q, \mathcal{T})$, as assigning b to all the variables in $V_Y \cup V_C$ satisfies all parts of the query. It remains to show that $\mathcal{E}$ is minimal for $\mathcal{A} \models (Q, \mathcal{T})$. Note that $\{\mathit{Val}_x(f) \mid \sigma_X(x) = 0\} \cup \{\mathit{Val}_x(t) \mid \sigma_X(x) = 1\}$ has to be in $\mathcal{E}$ for the *Val* part of the query to be satisfied, and $\mathcal{A}_{st}$ has to be in $\mathcal{E}$ for the *Config* part of the query to be satisfied. The only fact that may be removed is $c(b,b)$. Thus, $\mathcal{E}$ is minimal if $\mathcal{E} \setminus \{c(b,b)\} \not\models (Q, \mathcal{T})$. Assume, by contradiction, that $\mathcal{E} \setminus \{c(b,b)\} \models (Q, \mathcal{T})$. Then, there is an assignment ρ to $V_X \cup V_Y \cup V_C$ that satisfies the query. Then, $\rho(v_x) = f$ (resp., t) whenever $\sigma_X(x) = 0$ (resp., 1). As $c(b,b)$ is not in a MinEX, for each $z \in X \cup Y$, one of $\rho(v_z)$ and $\rho(v_{\neg z})$ is t, and the other one is f. Also, in this case, clause_i simulates c_i for each i. This implies that ρ satisfies the clauses of the formula. Define σ_Y to be an assignment to Y such that $\sigma_Y(y) = 0$ if $\rho(v_y) = f$, and $\sigma_Y(y) = 1$ if $\rho(v_y) = t$. We have that $\phi(\sigma_X(X), \sigma_Y(Y))$ holds, which is a contradiction to the fact that $\exists X \forall Y \, \neg\phi(X,Y)$ holds.

($\Leftarrow$) Suppose that there is a MinEX $\mathcal{E}$ for $\mathcal{A} \models (Q, \mathcal{T})$ containing the distinguished fact $c(b,b)$. As $\mathcal{E} \models \mathit{Config}$, we have that $\mathcal{A}_{st} \subseteq \mathcal{E}$. Also, since $\mathcal{E} \models \exists V_X \; \mathit{Val}$, one of

$Val_x(t)$ and $Val_x(f)$ is in $\mathcal{E}$ for each $x \in X$. Observe that if both $Val_x(t)$ and $Val_x(f)$ were present in $\mathcal{E}$, then $\mathcal{E}$ would not be minimal. This is so, as $c(b,b)$ is in $\mathcal{E}$, and thus any assignment to $V_X \cup V_Y \cup V_C$ entails the *Satisfied* and *Consistent* parts of the query. Let σ_X be an assignment to X given by the Val_x facts in $\mathcal{E}$. We want to show that there is no assignment σ_Y to Y such that $\phi(\sigma_X(X), \sigma_Y(Y))$ holds. Assume, by contradiction, that there is such an assignment σ_Y to Y. Then, by construction, $\mathcal{E} \setminus \{c(b,b)\}$ would entail the OMQ $(Q, \mathcal{T})$. But, we know that $\mathcal{E}$ is minimal, and so there is no assignment σ_Y to Y such that $\phi(\sigma_X(X), \sigma_Y(Y))$ holds. Thus, $\exists X \forall Y \neg\phi(X,Y)$ holds. □

The remaining hardness results follow from the fact that $\mathrm{OMQA}(\mathcal{Q}, \mathcal{L})$ in combined complexity can be reduced to MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$ in combined complexity for DLs $\mathcal{L}$ containing $\mathcal{EL}$.

Theorem 4.31. *For any DL $\mathcal{L}$ that can express $\mathcal{EL}$ and any query language $\mathcal{Q} \in \{\mathrm{IQ}, \mathrm{UCQ}\}$, the problem* MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$ *in combined complexity is as hard as* $\mathrm{OMQA}(\mathcal{Q}, \mathcal{L})$ *in combined complexity.*

Proof. We show a reduction from $\mathrm{OMQA}(\mathcal{Q}, \mathcal{L})$ to MinEX-Rel$(\subseteq, \mathcal{Q}, \mathcal{L})$. We have two query to consider, namely, IQ and UCQ, which we consider in turn. Suppose we have an ABox $\mathcal{A}$ and an $\mathcal{L}$-OMQ $(Q, \mathcal{T})$ with an instance query $Q = C(a)$. Then we take two new concept symbols D and F, and take a new TBox to be $\mathcal{T}' = \mathcal{T} \cap \{C \sqcap D \sqsubseteq F\}$. Then note that $\mathcal{A} \models (Q, \mathcal{T})$ iff $D(a)$ is relevant for $\mathcal{A} \cup \{D(a)\} \models (F(a), \mathcal{T})$.

Now suppose that Q is a UCQ. Then we take the ABox to be $\mathcal{A}' = \mathcal{A} \cup \{C(a), D(a)\}$, where C, D are fresh concept symbols and a is a fresh individual. We take the query to be $Q' = (Q \wedge C(a)) \vee D(a)$. We take a distinguished fact to be $\psi = C(a)$. Note that $\mathcal{A}' \models (Q', \mathcal{T})$ and thus we have constructed an instance of MinEX-Rel. Now it is easy to see that $\mathcal{A} \models (Q, \mathcal{T})$ iff ψ is relevant for $\mathcal{A}' \models (Q', \mathcal{T})$. □

4.7 Overview of Complexity Results

In this chapter, we have studied the complexity of computational problems, associated with explaining ontology-mediated query answers in the context of DLs. This chapter extends the work of Chapter 4 by considering DLs, instead of existential rules, and thus contributes to providing a more complete picture of explanations for OMQA.

We have studied the problems, introduced in Chapter 4, namely, Is-MinEX, All-MinEX, MinEX-Rel, and MinEX-Irrel. For each of these explanation problems, we have conducted a detailed complexity analysis. We allowed queries in the form of *instance queries*, or *unions of conjunctive queries*, which were coupled with ontologies formed in a DL that ranges from the light-weight *DL-Lite*$_{core}$ and $\mathcal{EL}$ to expressive fragments such as $\mathcal{SHIQ}$.

First of all, we observe that, for *DL-Lite*$_{core}$ and *DL-Lite*$_{\mathcal{R}}$, all the computational problems studied are tractable in data complexity. This is primarily because for these DLs, by Corollary 4.17, we can construct the set of all MinEXs in polynomial time in data complexity, and thus checking all the properties, associated with MinEXs, remains in polynomial time.

For other DLs than *DL-Lite*$_{core}$ and *DL-Lite*$_{\mathcal{R}}$, we have a considerably different picture in data complexity, compared to the one for existential rules, presented in Chapter 3. We have shown that Is-MinEX remains tractable in data complexity for the DLs between $\mathcal{EL}$ and Horn-$\mathcal{SHOIQ}$. We have managed to show that Is-MinEX($\subseteq$, IQ, $\mathcal{ELI}$) is P-hard in data complexity by a reduction from OMQA(IQ, $\mathcal{ELI}$). Note that, for such a reduction, we had to show that, given an ABox $\mathcal{A}$ and an OMQ $(Q, \mathcal{T})$, which form an instance of OMQA, we can construct an ABox $\mathcal{A}'$, a subset $\mathcal{E} \subseteq \mathcal{A}'$, and an OMQ $(Q', \mathcal{T}')$ such that $\mathcal{A} \models (Q, \mathcal{T})$ iff $\mathcal{E}$ is a MinEX for $\mathcal{A}' \models (Q', \mathcal{T}')$. Therefore, the complexity of OMQA does not automatically transfer to Is-MinEX. For $\mathcal{ELI}$, we have managed to ensure the minimality of $\mathcal{E}$ using inverse roles. For $\mathcal{EL}$ and $\mathcal{ELH}$, we lack such a feature and have not managed to reduce OMQA to Is-MinEX in data complexity. Therefore, it is open whether Is-MinEX($\subseteq$, IQ, $\mathcal{EL}$) and Is-MinEX($\subseteq$, IQ, $\mathcal{ELH}$) are also P-hard in data complexity, and whether Theorem 4.10 can be strengthened in this direction.

Recall that, for existential rules in Chapter 3 the complexity of Is-MinEX in data complexity did not increase, compared to the complexity of OMQA, and remained tractable. For the DL $\mathcal{ALC}$, we have seen that Is-MinEX($\subseteq$, IQ, $\mathcal{ALC}$) is D^{P}-complete in data complexity, while OMQA(IQ, $\mathcal{ALC}$) is coNP-complete in data complexity. So the complexity of Is-MinEX increases compared to OMQA already in data complexity in this case. This is because, to check whether $\mathcal{E}$ is a MinEX for $\mathcal{A} \models (Q, \mathcal{T})$, we need to check whether $\mathcal{E} \models (Q, \mathcal{T})$, which can be done in coNP, and whether $\mathcal{E}$ is subset-minimal with such a property, which can be done in NP. For all the other problems, we have observed that for $\mathcal{ALC}$ already in data complexity they are complete for a class in the second level of the polynomial hierarchy, which is in stark contrast with the complexity results in data complex-

ity for existential rules. In particular, All-MinEX$(\subseteq, \text{IQ}, \mathcal{ALC})$ is Π_2^{P}-complete, and MinEX-Irrel$(\subseteq, \text{IQ}, \mathcal{ALC})$ together with MinEX-Rel$(\subseteq, \text{IQ}, \mathcal{ALC})$ are Σ_2^{P}-complete in data complexity. This is primarily because OMQA$(\text{IQ}, \mathcal{ALC})$ is coNP-complete in data complexity. Note that for existential rules OMQA was always tractable in data complexity.

Our combined complexity analysis is less surprising in that the complexity is typically dominated by the complexity of OMQA. In particular, whenever OMQA is Exp-complete, we have that all the studied problems are also Exp-complete. On the other hand, for *DL-Lite*$_{core}$, *DL-Lite*$_{\mathcal{R}}$, $\mathcal{EL}$ and $\mathcal{ELH}$, when the query language is UCQ, we have that OMQA is NP-complete in combined complexity. Therefore, we have MinEX-Rel is Σ_2^{P}-complete in this case since once we guess a subset of the database that contains a distinguished assertion (in NP), we need to check its minimality (in coNP). Similarly, we have that All-MinEX is D^{P}-compete in this case. On the other hand, MinEX-Irrel remains NP-complete as we do not need to check minimality to ensure that there is a MinEX that does not contain any of the forbidden sets.

We have observed significantly different landscape of complexity results in this chapter, compared to Chapter 3. This is primarily because there is a number of differences in the complexity of OMQA. Despite that, we have managed to transfer a number of membership results from Chapter 3. On the other hand, the hardness results in this chapter are mostly novel since the constructors used in the DLs studied are different compared to existential rules. Also, DLs also have only unary and binary predicates, while existential rules can have predicated of any arity.

Chapter 5

Explanations for Negative Query Answers under Existential Rules

In this chapter, we continue our line of research and address another important problem of explaining why a query is *not* entailed, also known as a negative query answer. To answer this question, we adopt the approach developed by Calvanese et al. [43] to explain negative query answers in *DL-Lite* description logics. In particular, we take an explanation to be a set of facts, whose addition to the database forces the entailment of an OMQ. This is a natural choice in our case as a non-entailment can be attributed to missing facts. We continue this line of research, started by Calvanese et al., and address the computational problems, associated with explaining negative OMQ answers under existential rules. We consider various problems related to explaining non-entailments from the abduction literature and also introduce new problems. For all considered problems, we give a detailed complexity analysis for a wide range of existential rule languages and complexity measures.

There have been also works on explaining negative query answers under *inconsistency-tolerant semantics* [24]. Though, in that case, the objective is different, given that a query answer holds under the least restrictive inconsistency-tolerant semantics, known as *brave*, the goal is to explain why this answer does not hold under more restrictive semantics, such as AR and IAR. Therefore, our approach is different in nature as we deal with classical semantics.

This chapter is based on the paper published in the proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR-20) [49].

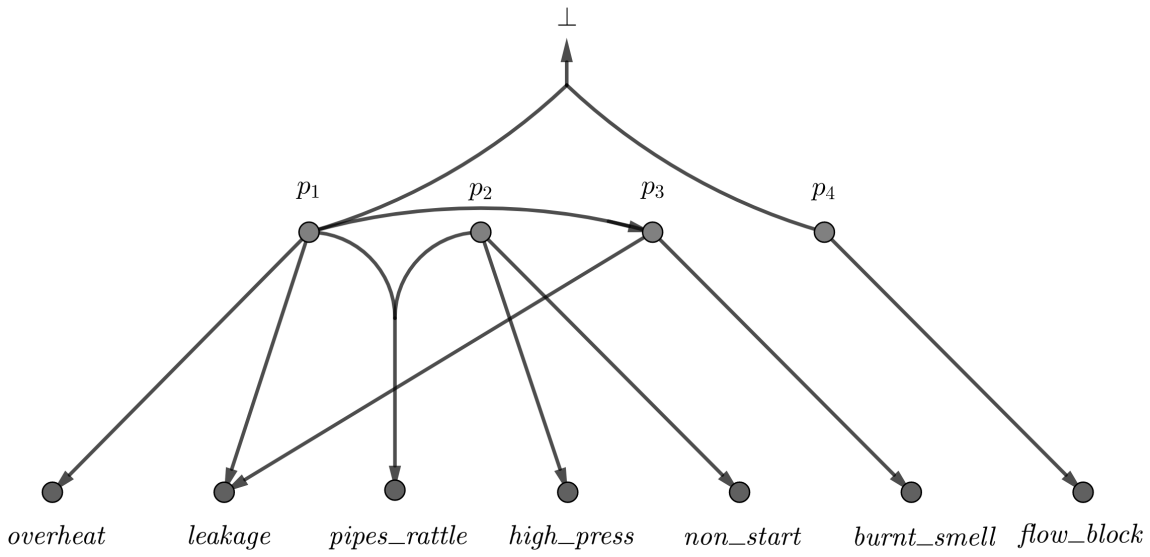

Figure 5.1: Mechanical system and its network of issues

5.1 Explanations for Negative Query Answers

In this section, we formally define explanations for negative ontology-mediated query answers. Given that a database does *not* entail an OMQ, we take an explanation to be a subset of a specified set of hypotheses, which, together with the database, entails the OMQ and is consistent with the ontology.

In this case, unlike in Chapter 3, we allow the program to contain both TGDs and *negative constraints*. This is meaningful as negative constraints allow to express the conditions to be met by the added facts from the set of hypothesis.

Definition 5.1. Let D be a database, let (Q, Σ) be an OMQ, with $D \not\models (Q, \Sigma)$, and let H be a finite set of facts. An *explanation for* $D \not\models (Q, \Sigma)$ *w.r.t.* H is a subset $E \subseteq H$ such that $(D \cup E, \Sigma)$ is consistent and $D \cup E \models (Q, \Sigma)$.

A *minimal explanation* (or *MinEX*) *for* $D \not\models (Q, \Sigma)$ *w.r.t.* H is an explanation E for $D \not\models (Q, \Sigma)$ w.r.t. H that is subset-minimal, i.e., no set $E' \subsetneq E$ is an explanation for $D \not\models (Q, \Sigma)$ w.r.t. H.

If H is clear from the context, we say that E is a MinEX for $D \not\models (Q, \Sigma)$, and, if D, H and (Q, Σ) are clear from the context, we simply say that E is a MinEX. Notice that, since $D \not\models (Q, \Sigma)$, the knowledge base (D, Σ) must be consistent. If (D, Σ) were inconsistent, then, for any Q, we would have that $D \models (Q, \Sigma)$. Also, as introduced in Chapter 2, given a program Σ, we write Σ_{NC} for the set of negative constraints in Σ, and Σ_T for the set of TGDs in Σ.

We illustrate how minimal explanations for negative answers can be applied with an example. Suppose we have a mechanical system where a number of issues may occur. We would like to understand, when such issues occur, which parts of the system may be causing them. This problem is known as *fault diagnosis*, which can be modelled via explanations of negative query answers as follows.

Example 5.2. We encode the following mechanical system and the possible issues, illustrated in Figure 5.1. The system consists of four parts p_1, p_2, p_3, and p_4, where faults in these parts may cause some of the issues, namely, *overheat*, *leakage*, *high_press*, *non_start*, *burnt_smell*, *flow_block*, and *pipes_rattle*, as illustrated in Figure 5.1. For example, if p_1 is faulty, then the issues *overheat* and *leakage* occur; if p_3 is faulty, then the issues *leakage* and *burnt_smell* occur; or if p_1 and p_2 are faulty, then *pipes_rattle* occurs. We encode this system as follows. First of all, the set of hypotheses captures the parts of the system, which might fail:

$$H = \{\mathit{fault}(p_1), \mathit{fault}(p_2), \mathit{fault}(p_3), \mathit{fault}(p_4)\}.$$

We capture the functionality of the system and how the issues may be caused via facts in the database D and ontological axioms in the program Σ. Below the binary predicate $\mathit{fault\_issue}(X, Y)$ describes that when the part X is faulty, we may observe the issue Y. Thus, the database is

$$\begin{aligned} D = \{&\mathit{fault\_issue}(p_1, \mathit{overheat}), \mathit{fault\_issue}(p_1, \mathit{leakage}),\\ &\mathit{fault\_issue}(p_2, \mathit{high\_press}), \mathit{fault\_issue}(p_2, \mathit{non\_start}),\\ &\mathit{fault\_issue}(p_3, \mathit{leakage}), \mathit{fault\_issue}(p_3, \mathit{burnt\_smell}),\\ &\mathit{fault\_issue}(p_4, \mathit{flow\_block})\}. \end{aligned}$$

The program Σ consists of three rules. The first rule in Σ (an NC) says that the parts p_1 and p_4 cannot be faulty at the same time. The second one encodes that if p_1 is faulty, then so is p_3. The third one says that when p_1 and p_2 are both faulty, then we observe rattling pipes. The last one encodes that if X is a part causing the issue Y, and X is faulty, then we observe the issue Y. Thus the program is

$$\begin{aligned} \Sigma = \{&\mathit{fault}(p_1) \wedge \mathit{fault}(p_4) \rightarrow \bot, \mathit{fault}(p_1) \rightarrow \mathit{fault}(p_3),\\ &\mathit{fault}(p_1) \wedge \mathit{fault}(p_2) \rightarrow \mathit{issue}(\mathit{pipes\_rattle}),\\ &\mathit{fault\_issue}(X, Y) \wedge \mathit{fault}(X) \rightarrow \mathit{issue}(Y)\}. \end{aligned}$$

Finally, in the query, we capture the observed failures. In the following query, we capture that the issues of *leakage* and *non_start* have been observed:

$$Q = \mathit{issue}(\mathit{leakage}) \wedge \mathit{issue}(\mathit{non\_start}).$$

Note that certainly $D \not\models (Q, \Sigma)$, and the minimal explanations in this case are

$$E_1 = \{\mathit{fault}(p_1), \mathit{fault}(p_2)\} \text{ and } E_2 = \{\mathit{fault}(p_2), \mathit{fault}(p_3)\}.$$

■

5.2 Explanation Problems for Negative Query Answers

In this section, we introduce the computational problems, studied in this chapter. The first and most fundamental computational problem is deciding whether a set of facts is a minimal explanation for a negative query answer. This problem is a decision version of the problem of constructing a minimal negative explanation.

Problem: Is-MinEX$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$.
Input: A database D, an $\mathcal{L}$-OMQ (Q, Σ), where Q is a query from the class $\mathcal{Q}$ and $D \not\models (Q, \Sigma)$, a finite set of facts H, and $E \subseteq H$.
Question: Is E a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H?

Example 5.3. Consider again the database D, the program Σ, and the set H of hypotheses as in Example 5.2. Take the query to be

$$Q = \mathit{issue}(\mathit{leakage}) \wedge \mathit{issue}(\mathit{burnt\_smell}).$$

Then the sets $E_1 = \{\mathit{fault}(p_1)\}$ and $E_2 = \{\mathit{fault}(p_3)\}$ are MinEXs for $D \not\models (Q, \Sigma)$ w.r.t. H, while $E' = \{\mathit{fault}(p_1), \mathit{fault}(p_3)\}$ is an explanation, but not a minimal one. ■

Another computational problem is deciding whether there exists a minimal explanation at all for a negative OMQ answer. Observe that, in Chapter 3, where we studied explanations for positive query answers, we were guaranteed the existence of a MinEX. This was because the knowledge base (D, Σ) was always consistent and $D \models (Q, \Sigma)$. On the other hand, in this case, we might have negative constraints in the program, and thus adding new facts to the database might introduce inconsistency. This means that the existence of an explanation for a negative query answer

is not guaranteed, as we illustrate below. For these reasons, we consider the problem, MinEX-Exists$^{\not\models}$, of deciding if any negative explanation exists.

Problem: MinEX-Exists$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$.
Input: A database D, an $\mathcal{L}$-OMQ (Q, Σ), where Q is a query from the class $\mathcal{Q}$ and $D \not\models (Q, \Sigma)$, and a finite set of facts H.
Question: Is there a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H?

Example 5.4. Consider again the database D, the program Σ, and the set H of hypotheses as in Example 5.2. If we take the query to be

$$Q = \mathit{issue}(\mathit{flow\_block}) \wedge \mathit{issue}(\mathit{burnt\_smell}),$$

then there is a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H. For example, $E = \{\mathit{fault}(p_3), \mathit{fault}(p_4)\}$ is such a MinEX. On the other hand, if the query is

$$Q = \mathit{issue}(\mathit{flow\_block}) \wedge \mathit{issue}(\mathit{overheat}),$$

there is no MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H. To see this, note that the only fact that implies $\mathit{issue}(\mathit{overheat})$ is $\mathit{fault}(p_1)$ and the only fact that causes $\mathit{issue}(\mathit{flow\_block})$ is $\mathit{fault}(p_4)$ but the NC $\mathit{fault}(p_1) \wedge \mathit{fault}(p_4) \rightarrow \bot$ encodes that they cannot happen together. ∎

The next two computational problems are recognising relevant facts and recognising necessary facts. Recall that a fact ψ is *relevant* for $D \not\models (Q, \Sigma)$ w.r.t. H iff ψ appears in *some* MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H. We say that a fact ψ is *necessary* for $D \not\models (Q, \Sigma)$ w.r.t. H iff ψ appears in *every* MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H.

Problem: MinEX-Rel$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$.
Input: A database D, an $\mathcal{L}$-OMQ (Q, Σ), where Q is a query from the class $\mathcal{Q}$ and $D \not\models (Q, \Sigma)$, a finite set of facts H, and a fact ψ.
Question: Is ψ relevant for $D \not\models (Q, \Sigma)$ w.r.t. H?

Problem: MinEX-Nec$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$.
Input: A database D, an $\mathcal{L}$-OMQ (Q, Σ), where Q is a query from the class $\mathcal{Q}$ and $D \not\models (Q, \Sigma)$, a finite set of facts H, and a fact ψ.
Question: Is ψ necessary for $D \not\models (Q, \Sigma)$ w.r.t. H?

Example 5.5. Consider again the database D, the program Σ, and the set H of hypotheses as in Example 5.2. If the query is

$$Q = \mathit{issue}(\mathit{non\_start}) \wedge \mathit{issue}(\mathit{leakage}),$$

the sets $E_1 = \{fault(p_1), fault(p_2)\}$ and $E_2 = \{fault(p_2), fault(p_3)\}$ are all the minimal explanations for $D \not\models (Q, \Sigma)$. Hence, in this case, the facts $fault(p_1)$, $fault(p_2)$, and $fault(p_3)$ are relevant, while the only necessary fact is $fault(p_3)$. ∎

The computational problems introduced so far are those commonly addressed in the context of abductive reasoning and explaining negative query answers [71, 135, 48]. In addition, we introduce two novel important problems. The first one asks whether a set H' of facts contains exactly all the relevant facts, i.e., H' is the union of all MinEXs. The problem is particularly interesting, as H' can be seen as a minimal over-approximation of all MinEXs, i.e., for every MinEX E, it holds that $E \subseteq H'$, and H' is the smallest set enjoying this property.

Problem: MinEX-AllRel$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$.
Input: A database D, an $\mathcal{L}$-OMQ (Q, Σ), where Q is a query from the class $\mathcal{Q}$ and $D \not\models (Q, \Sigma)$, a finite set of facts H, and a set $H' \subseteq H$.
Question: Does H' contain exactly all the relevant facts for $D \not\models (Q, \Sigma)$ w.r.t. H?

The second novel computational problem that we consider asks whether a set H' contains exactly all the necessary facts, i.e., H' is the intersection of all MinEXs. Interestingly, H' can be seen as a maximal under-approximation of all MinEXs, i.e., for every MinEX E, it holds that $H' \subseteq E$, and H' is the biggest set enjoying this property.

Problem: MinEX-AllNec$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$.
Input: A database D, an $\mathcal{L}$-OMQ (Q, Σ), where Q is a query from the class $\mathcal{Q}$ and $D \not\models (Q, \Sigma)$, a finite set of facts H, and a set $H' \subseteq H$.
Question: Does H' contain exactly all the necessary facts for $D \not\models (Q, \Sigma)$ w.r.t. H?

We can illustrate the problems on our running example as follows.

Example 5.6. For the setting in Example 5.5, with the query

$$Q = issue(non\_start) \wedge issue(leakage),$$

the sets of all relevant and necessary facts are $\{fault(p_1), fault(p_2), fault(p_3)\}$ and $\{fault(p_3)\}$, respectively. ∎

5.3 Recognising and Existence of Minimal Explanations

We now analyse the complexity of the problem of recognising a MinEX, Is-MinEX$^{\not\models}$, and the complexity of the problem of deciding whether a MinEX for a negative query

Table 5.1: Complexity results for Is-MinEX$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**Data**	*fp*-**comb.**	*ba*-**comb.**	**Comb.**
$\mathsf{L}_\bot$, $\mathsf{LF}_\bot$, $\mathsf{AF}_\bot$	$\leq$ P	D$^{\text{P}}$	D$^{\text{P}}$	PSpace
$\mathsf{S}_\bot$, $\mathsf{SF}_\bot$	$\leq$ P	D$^{\text{P}}$	D$^{\text{P}}$	Exp
$\mathsf{A}_\bot$	$\leq$ P	D$^{\text{P}}$	D$^{\text{Exp}}$	D$^{\text{Exp}}$
$\mathsf{G}_\bot$	P	D$^{\text{P}}$	Exp	2Exp
$\mathsf{F}_\bot$, $\mathsf{GF}_\bot$	P	D$^{\text{P}}$	D$^{\text{P}}$	Exp
$\mathsf{WS}_\bot$, $\mathsf{WA}_\bot$	P	D$^{\text{P}}$	2Exp	2Exp

answer exists, MinEX-Exists$^{\not\models}$. We start with the problem Is-MinEX$^{\not\models}$, which, given a database D, an OMQ (Q, Σ) (with $D \not\models (Q, \Sigma)$), a finite set of facts H, and a subset $E \subseteq H$, asks whether E is a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H. The following theorem proves all the upper bounds in Table 5.1, apart from D$^{\text{P}}$ and D$^{\text{Exp}}$ ones, which follow from the proof of the following result.

Theorem 5.7. *For any language $\mathcal{L}$ of existential rules, if* OMQA(UCQ, $\mathcal{L}$) *is in the complexity class $\mathcal{C}$, then* Is-MinEX$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *can be decided by a check in $\mathcal{C}$ and a linear number of checks in co-$\mathcal{C}$.*

Proof. Let D, (Q, Σ), H and $E \subseteq H$ be an instance of Is-MinEX$^{\not\models}$. To decide whether E is a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H, we need to check whether (i) $(D \cup E, \Sigma)$ is *consistent*, (ii) $D \cup E \models (Q, \Sigma)$, and (iii) for every $\psi \in E$, whether $D \cup E \setminus \{\psi\} \not\models (Q, \Sigma)$. Notice that $(D \cup E \setminus \{\psi\}, \Sigma)$ is guaranteed to be consistent if $(D \cup E, \Sigma)$ is consistent, which is checked by the condition (i) above.

First of all, the condition (i) can be ensured by checking whether for every $\nu \in \Sigma_{NC}$, $D \cup E \not\models (Q_\nu, \Sigma_T)$, where Q_ν is the body of the NC ν. This takes a linear number of non-entailment checks, each in co-$\mathcal{C}$. The condition (ii) can be checked in $\mathcal{C}$ as it is simply OMQA. Finally, the condition (iii) can be checked with a linear number (in the size of the database) of non-entailment checks, each in co-$\mathcal{C}$. Therefore the results follow; Is-MinEX$^{\not\models}$ can be decided with a $\mathcal{C}$ check and a linear number of co-$\mathcal{C}$ checks. □

We indicate how the above theorem implies the membership results in Table 5.1. If OMQA is in $\mathcal{C} \in \{\text{P}, \text{PSpace}, \text{Exp}, \text{2Exp}\}$, since $\mathcal{C}$ is a deterministic class, it is closed under complementation, and thus co-$\mathcal{C} = \mathcal{C}$. Therefore Is-MinEX$^{\not\models}$ is solvable with a linear number (in the size of the data) of $\mathcal{C}$ checks, which can be combined as a single $\mathcal{C}$ test, hence the overall procedure is in $\mathcal{C}$.

Table 5.2: Complexity results for MinEX-Exists$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**Data**	*fp*-**comb.**	*ba*-**comb.**	**Comb.**
$\mathsf{L}_\bot$, $\mathsf{LF}_\bot$, $\mathsf{AF}_\bot$	$\leq$ P	NP	Σ_2^{P}	PSpace
$\mathsf{S}_\bot$, $\mathsf{SF}_\bot$	$\leq$ P	NP	Σ_2^{P}	Exp
$\mathsf{A}_\bot$	$\leq$ P	NP	P^{NExp}	P^{NExp}
$\mathsf{G}_\bot$	NP	NP	Exp	2Exp
$\mathsf{F}_\bot$, $\mathsf{GF}_\bot$	NP	NP	Σ_2^{P}	Exp
$\mathsf{WS}_\bot$, $\mathsf{WA}_\bot$	NP	NP	2Exp	2Exp

The D^{P} and D^{Exp} membership results in Table 5.1 follow from the proof of Theorem 3.8. The following argument is similar to the one in Corollary 3.9.

Corollary 5.8. *Let $\mathcal{L}$ be a class of TGDs.*

- *If* OMQA(UCQ, $\mathcal{L}$) *is in* NP*, then* Is-MinEX$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* $\text{D}^{\text{P}}_{\parallel}$
- *If* OMQA(UCQ, $\mathcal{L}$) *is in* NExp*, then* Is-MinEX$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* D^{Exp}.

Proof. In the proof of Theorem 5.7, observe that, when $\mathcal{C} \in \{\text{NP}, \text{NExp}\}$, all co-$\mathcal{C}$ are all independent, and thus they can be combined as a single co-$\mathcal{C}$ test by combining linearly many certificates. This implies that minimality can be checked in co-$\mathcal{C}$. Thus we obtain that Is-MinEX$^{\not\models}$ is in $\mathcal{C} \wedge$ co-$\mathcal{C}$ and the result follows.

□

We now focus on the hardness results for Is-MinEX$^{\not\models}$. In Chapter 3, we studied the problem of deciding whether a set E is a minimal explanation for $D \models (Q, \Sigma)$, called Is-MinEX. All the lower bounds in Table 5.1 are obtained from the following result that establishes a connection between Is-MinEX$^{\not\models}$ and Is-MinEX.

Theorem 5.9. *For any language $\mathcal{L}$ of existential rules,* Is-MinEX$(\subseteq, \text{BCQ}, \mathcal{L})$ *is reducible in polynomial time to* Is-MinEX$^{\not\models}(\subseteq, \text{BCQ}, \mathcal{L})$.

Proof. Let D be a database, (Q, Σ) be an OMQ, where Σ is a set of TGDs, and $E \subseteq D$, which forms an instance of Is-MinEX$(\subseteq, \text{BCQ}, \mathcal{L})$. Note that Σ does not contain NCs here. It can be easily verified that E is a minimal explanation for $D \models (Q, \Sigma)$ iff E is a minimal explanation for $\{\emptyset\} \not\models (Q, \Sigma)$ w.r.t. $H = D$.

□

We now focus on the problem MinEX-Exists$^{\not\models}$ of deciding the existence of a MinEX for a given negative query answer. We start by providing the membership results with UCQ as the query language. The following theorem proves all the upper bounds in Table 5.2, apart from the P and NP ones, for which we need tighter results stated next. Intuitively, to decide whether there exists a minimal explanation for a negative query answer, it suffices to check whether there is any explanation, i.e., there is no need to check the minimality. This is so because all the query that we study are monotonic.

Theorem 5.10. *For any language $\mathcal{L}$ of existential rules, if* OMQA(UCQ, $\mathcal{L}$) *is in the complexity class $\mathcal{C}$, then* MinEX-Exists$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *can be decided in* $\text{NP}^{\mathcal{C}}$.

Proof. Let D, (Q, Σ) and H be an instance of MinEX-Exists$^{\not\models}$. To decide whether there exists a MinEX E for $D \not\models (Q, \Sigma)$ w.r.t. H, we need to guess a subset E of H and check whether (i) $(D \cup E, \Sigma)$ is consistent, and (ii) $D \cup E \models (Q, \Sigma)$. We can guess E in NP, and then, as in the proof of Theorem 5.7, check the condition (i) with a linear number of non-entailment checks, each in co-$\mathcal{C}$, and check the condition (ii) with a single $\mathcal{C}$ check. This implies that MinEX-Exists$^{\not\models}$, in this case, is in $\text{NP}^{\mathcal{C}}$ as the NP machine can flip the answer of an oracle. □

Note that Theorem 5.10 implies all the membership results in Table 5.2, apart from the membership in P in data complexity for FO-rewritable languages and the membership in NP in *fp*-combined complexity. We illustrate how the membership results follow from the above result. If OMQA is in P, we have that MinEX-Exists$^{\not\models}$ is in $\text{NP}^{\text{P}} = \text{NP}$. In *ba*-combined complexity, when OMQA is in NP, we have that MinEX-Exists$^{\not\models}$ is in $\text{NP}^{\text{NP}} = \Sigma_2^{\text{P}}$. When OMQA is in NExp, we have that MinEX-Exists$^{\not\models}$ is in NP^{NExp}. Note that $\text{NP}^{\text{NExp}} = \text{P}^{\text{NExp}}$ [89], and thus MinEX-Exists$^{\not\models}$ is in P^{NExp} in this case. Finally, in *ba*-combined and combined complexity, if OMQA is in $\mathcal{C} \in \{\text{PSpace}, \text{Exp}, \text{2Exp}\}$, we have that MinEX-Exists$^{\not\models}$ is in $\text{NP}^{\mathcal{C}} = \mathcal{C}$.

The following theorem provides the NP upper bounds in *fp*-combined complexity in Table 5.2. The result is obtained from the proof of Theorem 5.7 with the additional observation that in *fp*-combined setting, for the Datalog$^{\pm}$ languages considered, checking whether a set of facts is consistent is feasible in P, because the negative constraints are fixed.

Theorem 5.11. *For all languages $\mathcal{L}$ of existential rules considered in this work,* MinEX-Exists$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* NP *in fp-combined complexity.*

Proof. In *fp*-combined complexity, since Σ_{NC} is fixed, the condition (i) in the proof of Theorem 5.10 can indeed be checked in polynomial time. To see this, observe that since the program is fixed, for each $\nu \in \Sigma_{NC}$, we need to check whether $D \not\models (Q_\nu, \Sigma)$, where the OMQ (Q_ν, Σ) is fixed. Thus it is equivalent to checking non-entailment in data complexity, which is in P (cf. Table 2.2). Checking the condition (ii) in the proof of Theorem 5.10 is in NP. As we can guess E along with the certificate witnessing that $D \cup E \models (Q, \Sigma)$, the overall procedure is in NP.

□

We conclude the discussion of the membership results for MinEX-Exists$^{\not\models}$ by showing that the problem is in P in data complexity for all the FO-rewritable languages considered, namely, $\mathsf{L}_\bot$, $\mathsf{S}_\bot$, and $\mathsf{A}_\bot$ as well as their full restrictions. We show this by slightly modifying the proof of Theorem 3.18.

Theorem 5.12. *For any FO-rewritable language $\mathcal{L}$ of existential rules, the problem* MinEX-Exists$^{\not\models}(\subseteq, \mathrm{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

Proof. Consider an instance of MinEX-Exists$^{\not\models}(\subseteq, \mathrm{UCQ}, \mathcal{L})$ consisting of a database D, an $\mathcal{L}$-OMQ (Q, Σ), and a finite set of facts H. Let Σ_T be the set of TGDs in Σ. As $\mathcal{L}$ is a FO-rewritable language, there exists a UCQ Q_{Σ_T} such that for every database D', we have that $D' \models (Q, \Sigma_T)$ iff $D' \models Q_{\Sigma_T}$. Note that Q_{Σ_T} depends only on Q and Σ_T, which are fixed in data complexity, so Q_{Σ_T} can be constructed in constant time. As Q_{Σ_T} is a UCQ, suppose that

$$Q_{\Sigma_T} = Q^1_{\Sigma_T} \vee \cdots \vee Q^m_{\Sigma_T}$$

where $Q^i_{\Sigma_T} = \exists \mathbf{Y}^i \Phi^i(\mathbf{Y}^i)$. Let $\mathcal{D}^i_{Q_{\Sigma_T}} = \{\Phi^i(\mathbf{t}) \mid \mathbf{t} \in \mathit{adom}(D \cup H)^{|\mathbf{Y}^i|}\}$, where $\mathit{adom}(D \cup H)$ is the set of all constants appearing in $D \cup H$ and $\Phi^i(\mathbf{t})$ is viewed as a set of facts. Thus, $\mathcal{D}^i_{Q_{\Sigma_T}}$ is a set of sets of facts and can be computed in polynomial time, as $\mathit{adom}(D \cup H)$ has polynomial size and $|\mathbf{Y}^i|$ is a constant. Furthermore, $\mathcal{D}_{Q_{\Sigma_T}} = \cup_{i=1}^m \mathcal{D}^i_{Q_{\Sigma_T}}$ can be computed in polynomial time as m is a constant. We can now decide whether there exists a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H by checking whether there exists an element X in $\mathcal{D}_{Q_{\Sigma_T}}$ s.t. $X \subseteq D \cup H$ and $(D \cup X, \Sigma)$ is consistent. This can be done in further polynomial time as consistency can be checked in polynomial time in data complexity. □

We proceed with hardness results. As for the hardness results for MinEX-Exists$^{\not\models}$, the following theorem gives the NP lower bounds in data complexity in Table 5.2 for the non-FO-rewritable languages, namely, $\mathsf{GF}_\bot$ and its generalizations. Its proof is

a reduction from the satisfiability of 3CNF Boolean formulas $\phi(X)$. Intuitively, in a fixed program, there are rules encoding the satisfiability of $\phi(X)$. The set H contains facts associated with truth assignments for the variables X. A candidate explanation E encodes a truth assignment σ_E for the variables X, and E is actually an explanation iff σ_E satisfies $\phi(X)$.

We note that this reduction is very close in nature to the reduction in the proof of Theorem 3.19, where we show that All-MinEX$(\subseteq, \text{BCQ}, \mathsf{GF})$ is coNP-hard in data complexity.

Theorem 5.13. MinEX-Exists$^{\not\models}$ $(\subseteq, \text{BCQ}, \mathsf{GF}_\bot)$ *is* NP-hard *in data complexity.*

Proof. We provide a reduction from the NP-complete problem Sat. We take $\varphi = c_1 \wedge \cdots \wedge c_m$ to be a 3CNF formula over variables $X = \{x_1, \ldots, x_n\}$. We construct an instance of MinEX-Exists$^{\not\models}(\subseteq, \text{BCQ}, \mathsf{GF}_\bot)$, consisting of a database D_φ, an OMQ (Q, Σ), and a finite set of facts H_φ. As we are in data complexity, only D_φ and H_φ depend on φ.

We construct the database D_φ as follows. For each clause c_j $(1 \leq j \leq m)$, we add a fact $succ\_cl(j-1, j)$ and a fact encoding c_j with the predicate symbol cl. For example, for a clause $(x_p \vee x_q \vee \neg x_r)$, there is the fact $cl(j, x_p, p, x_q, p, x_r, n)$ in D_φ, where p and n are constants representing whether a variable appears in a positive or negative literal, respectively. In addition, D_φ contains the two facts $satchain(0)$ and $maxcl(m)$.

We construct the program Σ. It contains a single NC, which ensure that no variable is assigned both 0 and 1 by a *val* fact, which is

$$val(X, 1), val(X, 0) \rightarrow \bot.$$

Furthermore, the program contains a number of TGDs. A number of TGDs capture the satisfiability of a clause, which are

$$\begin{aligned}
&cl(J, X, p, _, _, _, _), val(X, 1) \rightarrow satcl(J),\\
&cl(J, X, n, _, _, _, _), val(X, 0) \rightarrow satcl(J),\\
&cl(J, _, _, Y, p, _, _), val(Y, 1) \rightarrow satcl(J),\\
&cl(J, _, _, Y, n, _, _), val(Y, 0) \rightarrow satcl(J),\\
&cl(J, _, _, _, _, Z, p), val(Z, 1) \rightarrow satcl(J),\\
&cl(J, _, _, _, _, Z, n), val(Z, 0) \rightarrow satcl(J).
\end{aligned}$$

The following two TGDs ensure that the nulary fact is entailed only if all the clauses of ϕ (and thus ϕ) are satisfied. These TGDs are

$$satchain(I), succ\_cl(I, J), satcl(J) \rightarrow satchain(J),$$
$$maxcl(M), satchain(M) \rightarrow sat().$$

We take the query to be $Q = sat()$. Finally, H_φ is defined as follows. For each variable $x_i \in X$, there are facts $val(x_i, 1)$ and $val(x_i, 0)$ in H_φ, where x_i, 1, and 0 are constants, with 1 and 0 representing the Boolean values *true* and *false*, respectively. Observe that Σ and Q do not depend on φ. Also, notice that Σ is guarded and full.

We now show that φ is satisfiable iff there exists a (minimal) explanation for $(D_\varphi, \Sigma) \not\models q$ w.r.t. H_φ. We note that the correctness proof is

($\Rightarrow$) Suppose that φ is satisfiable. Then there is a truth assignment $v : X \rightarrow \{T, F\}$ that satisfies φ. It is easy then to verify that $E = \{val(x_i, t) \mid \phi(x_i) = T\} \cup \{val(x_i, f) \mid \phi(x_i) = F\}$ is an explanation, and thus a minimal one exists.

($\Leftarrow$) Suppose that φ is not satisfiable. Then there is no truth assignment satisfying φ, by construction, there is no way to entail $sat()$, and thus there is no explanation. □

The following theorem proves the Σ_2^{p}-hardness results of MinEX-Exists$^{\not\models}$ in *ba*-combined complexity in Table 5.2.

Theorem 5.14. MinEX-Exists$^{\not\models}(\subseteq, \mathrm{UCQ}, \mathcal{L})$ *is* Σ_2^{p}-hard *in ba-combined complexity for all languages* $\mathcal{L}$ *of considered here.*

Proof. We provide a reduction from a Σ_2^{p}-complete problem $\mathrm{QBF}_{2,\forall}^{DNF}$ of deciding validity of a quantified Boolean formula $\exists X \forall Y \varphi(X, Y)$, where $\varphi(X, Y)$ is a 3DNF formula. Let n be the number of variables in X and m the number of implicants of $\varphi(X, Y)$.

We build an instance of MinEX-Exists$^{\not\models}$ consisting of a database D, an OMQ (Q, Σ), and a finite set of facts H as follows. We start by defining H. For each variable $x_i \in X$, H contains the following two facts:

$$val(x_i, 0), \qquad\qquad val(x_i, 1).$$

The database D contains the following four facts to impose the consistency of the truth assignments to the literals:

$$simlit(0, 0), opplit(0, 1), simlit(1, 1), \text{ and } opplit(1, 0).$$

Also, D contains the fact $impl\_unsatisfied(0,0,0)$, which is used to check whether an implicant in φ is not satisfied. Here the constants 0 and 1 represent the Boolean values *false* and *true*, respectively. The predicate $simlit(\cdot,\cdot)$ will be used to impose that when a variable appears twice as a positive or a negative literal in two different places in the formula, the two literals must have the same truth value. On the other hand, the predicate $opplit(\cdot,\cdot)$ will be used to impose that when a variable appears as a positive literal in one place in the formula and as a negative literal in another place of the formula, the two literals must have opposite truth values. The predicate $impl\_unsatisfied(\cdot,\cdot,\cdot)$ states the truth assignment to the *literals* (and not to the variables) that do *not* satisfy an implicant.

The program Σ consists of two NCs. The first NC ensures that no variable is assigned both *false* and *true* values:

$$val(X,0), val(X,1) \rightarrow \bot.$$

The second NC is made of several pieces. The first piece "reads" onto the variables T_i the assignment on the variables in X encoded in the explanation:

$$assignX \equiv \bigwedge_{i=1}^{n} val(x_i, T_i).$$

The second piece "copies" the assignment of each variable x_i onto a positive occurrence of x_i in $\varphi(X,Y)$. Here we denote by $\ell_{j,k}$ the k^{th} literal in the j^{th} implicant of formula $\varphi(X,Y)$, and by $v_{j,k}$ the variable of $\ell_{j,k}$. So the second piece of this NC is

$$copy \equiv \bigwedge_{i=1}^{n} simlit(T_i, T_{j,k}),$$

where in each predicate $simlit(T_i, T_{j,k})$, $T_{j,k}$ is a variable for the Boolean value of a literal $\ell_{j,k} = x_i$ in $\varphi(X,Y)$. Observe that, in order for *copy* to work properly, each variable x_i must appear as a positive literal in $\varphi(X,Y)$ at least once. This can be assumed w.l.o.g., because if x_i always appears as a negative literal in all the implicants of $\varphi(X,Y)$, then we can replace all the occurrences of the negative literal $\neg x_i$ with the positive literal x_i without altering the satisfiability properties of $\varphi(X,Y)$.

The third piece forces the ground values 0 and 1 assigned to the variables $T_{j,k}$, simulating the assignments to the literals to be consistent. Below, $\ell_{j,k} \sim \ell_{j',k'}$ means that $v_{j,k} = v_{j',k'}$ and the literals $\ell_{j,k}$ and $\ell_{j',k'}$ are both positive or negative, while $\ell_{j,k} \nsim \ell_{j',k'}$ means that $v_{j,k} = v_{j',k'}$ and one literal is positive and the other is negative.

Thus the third piece is

$$\mathit{consist} \equiv \bigwedge_{\ell_{j,k} \sim \ell_{j',k'}} \mathit{simlit}(T_{j,k}, T_{j',k'}) \wedge \bigwedge_{\ell_{j,k} \nsim \ell_{j',k'}} \mathit{opplit}(T_{j,k}, T_{j',k'}),$$

where $T_{j,k}$ is a variable with the same meaning as above.

The last piece of this NC checks if $\varphi(X, Y)$ is unsatisfied. We take it to be

$$\mathit{unsatisfied} \equiv \bigwedge_{j=1}^{m} \mathit{impl\_unsatisfied}(T_{j,1}, T_{j,2}, T_{j,3}),$$

We can now state the second NC of Σ:

$$\mathit{assignX}, \mathit{copy}, \mathit{consist}, \mathit{unsatisfied} \rightarrow \bot.$$

Notice that Σ has no TGDs, and thus it belongs to all classes of languages we consider. Furthermore, all predicates have bounded arity.

We construct the query to ensure that a fact $\mathit{val}(x_i, \cdot)$ has been taken from H for every $x_i \in X$. Thus the query is

$$Q = \exists T_1 \ldots \exists T_n \ \mathit{val}(x_1, T_1), \ldots, \mathit{val}(x_n, T_n),$$

where n is the number of variables in X. We now show that $\exists X \forall Y \varphi(X, Y)$ is valid iff there exists an explanation for $D \not\models (Q, \Sigma)$ w.r.t. H.

($\Rightarrow$) Let $\sigma_X : X \rightarrow \{T, F\}$ be a truth assignment s.t. $\forall Y \varphi(X/\sigma_X, Y)$ is valid. We show that $E = \{\mathit{val}(x_i, t) \mid \sigma_X(x_i) = T\} \cup \{\mathit{val}(x_i, f) \mid \sigma_X(x_i) = F\}$ is an explanation (and thus a minimal one exists). For a contradiction, suppose that $(D \cup E, \Sigma)$ is not consistent. This means that the last NC is violated, that is, there is a truth assignment $\sigma_Y : Y \rightarrow \{T, F\}$ s.t. $\varphi(X/\sigma_X, Y/\sigma_Y)$ is not satisfied, which is a contradiction. Thus, $(D \cup E, \Sigma)$ is consistent. Certainly, $D \cup E \models (Q, \Sigma)$.

($\Leftarrow$) Let E be a minimal explanation for $D \not\models (Q, \Sigma)$ w.r.t. H. In order for $(D \cup E, \Sigma)$ to be consistent and entail the query, E must contain exactly one one fact $\mathit{val}(x_i, \cdot)$ for each $1 \leq i \leq n$. Let $\sigma_E : X \rightarrow \{T, F\}$ be the truth assignment s.t. $\sigma_E(x_i) = T$ iff $\mathit{val}(x_i, t) \in E$. Since $(D \cup E, \Sigma)$ does not violate the last NC, there is no way to assign truth values to variables in Y so that the NC is violated, which means that $\forall Y \varphi(X/\sigma_E, Y)$ is valid. □

Now we show the two $\mathrm{P}^{\mathrm{NExp}}$-hardness results in Table 5.2 for $\mathsf{A}_\bot$ in *ba*-combined and combined complexity. The result can be shown via a reduction from a $\mathrm{P}^{\mathrm{NExp}}$-complete problem Extended-Tiling-Problem, defined by Eiter et al. [76]. The

idea of the reduction is to have the candidate explanations taken from H to encode the possible initial tiling conditions. Then, there are rules that allow us to derive the query if TP_1 has a solution with the initial condition w encoded in the explanation, and TP_2 has no solution with w.

Theorem 5.15. MinEX-Exists$^{\not\models}$ $(\subseteq, \mathrm{BCQ}, \mathsf{A}_\perp)$ *is* $\mathrm{P}^{\mathrm{NExp}}$-hard *in ba-combined and combined complexity.*

Proof. We give a reduction from the following $\mathrm{P}^{\mathrm{NEXP}}$-complete problem [76]:

Problem: Extended-Tiling-Problem
Input: Two tiling problems TP_1 and TP_2 for the exponential square $2^n \times 2^n$, and an integer m in unary notation.
Question: Does there exist an initial condition w of length m such that TP_1 has a solution with w and TP_2 has no solution with w?

We build an instance of MinEX-Exists$^{\not\models}$ $(\subseteq, \mathrm{BCQ}, \mathsf{A}_\perp)$ consisting of a database D, an OMQ (Q, Σ), and a finite set of facts H as follows. For the tiling problems, we use the encoding as in [76] to create two copies for $\Sigma_{TP_i,|w|}$ and D_{TP_i} for $i = 1, 2$, using disjoint predicates. Notice that we do not construct D_w (as we will be looking for an initial tiling). Also, notice that $\Sigma_{TP_i,|w|}$ depends only on the length of w, that is, it does not depend on any actual initial tiling.

We take the database to be $D = {D_{TP_1}}^1 \cup {D_{TP_2}}^2$. We construct the program Σ as follows. For all $0 \leq j < m$ and all pairs $(t_{\ell'}, t_{\ell''})$ of different tile *constants*, the following NCs are added to Σ:

$$\mathit{Init}_j^1(t_{\ell'}) \wedge \mathit{Init}_j^1(t_{\ell''}) \to \perp,\ \mathit{Init}_j^2(t_{\ell'}) \wedge \mathit{Init}_j^2(t_{\ell''}) \to \perp,\ \mathit{Init}_j^1(t_{\ell'}) \wedge \mathit{Init}_j^2(t_{\ell''}) \to \perp.$$

We also add to Σ the NC: $\mathit{Tiling}^2() \to \perp$. Also, the program Σ contains ${\Sigma_{TP_1,|w|}}^1$ and ${\Sigma_{TP_2,|w|}}^2$ as well as the following TGD:

$$\bigwedge_{j=0}^{m-1} \mathit{Init}_j^1(\_) \wedge \mathit{Init}_j^2(\_) \to \mathit{AllInitTaken}().$$

Notice that the program remains acyclic after the addition of the TGD above. We take the set of hypothesis to be $H = \{\mathit{Init}_j^i(t) \mid \text{ for all } 1 \leq i \leq 2,\ 0 \leq j < m,\ \text{tiles } t\}$. The query is $Q = \mathit{Tiling}^1() \wedge \mathit{AllInitTaken}()$.

We show that there exists an initial tiling condition w of length m s.t. TP_1 has a solution with w *and* TP_2 has no solution with w iff there exists a (minimal) explanation for $D \not\models (Q, \Sigma)$ w.r.t. H.

($\Rightarrow$) Let $w = w_0 \dots w_{m-1}$ be an initial tiling condition of length m s.t. TP_1 has a solution with w *and* TP_2 has no solution with w. Let $E = \{Init^i_j(w_j) \mid 1 \leq i \leq 2, 0 \leq j < m\}$. We show that E is an explanation for $D \not\models (Q, \Sigma)$ w.r.t. H (and thus a minimal one exists). Clearly, $(D \cup E, \Sigma)$ does not violate the first group of NCs introduced above. By the results in [76], $D \cup E \not\models (Tiling^2(), \Sigma)$ and thus the last NC is not violated. Hence, $(D \cup E, \Sigma)$ is consistent. By the results in [76], $D \cup E \models (Tiling^1(), \Sigma)$. Furthermore, by construction of E, it is easy to see that $D \cup E \models (AllInitTaken(), \Sigma)$. Hence, $D \cup E \models (Q, \Sigma)$.

($\Leftarrow$) Let E be a minimal explanation for $D \not\models (Q, \Sigma)$ w.r.t. H. Since $(D \cup E, \Sigma)$ is consistent and $D \cup E \models (Q, \Sigma)$, then $D \cup E \models (AllInitTaken(), \Sigma)$ and the first group of NCs introduced above is not violated. This means that E is exactly of the form $E = \{Init^i_j(w_j) \mid 1 \leq i \leq 2, 0 \leq j < m\}$ for some $w = w_0 \dots w_{m-1}$. Since $D \cup E \models (Q, \Sigma)$, then $D \cup E \models (Tiling^1(), \Sigma)$. By the results in [76], TP_1 has a solution with w. Since $(D \cup E, \Sigma)$ is consistent, then $D \cup E \not\models (Tiling^2(), \Sigma)$. By the results in [76], TP_2 has no solution with w. □

The remaining hardness results in Table 5.2 in *fp*-combined, *ba*-combined, and combined complexity can be obtained from the following lemma showing a link between the complexity of OMQA and of MinEX-Exists$^{\not\models}$.

Lemma 5.16. *For any class $\mathcal{L}$ of existential rules,* MinEX-Exists$^{\not\models}(\subseteq, \text{BCQ}, \mathcal{L})$ *in ba-combined (resp., combined) complexity is as hard as* OMQA$(\text{BCQ}, \mathcal{L})$ *in ba-combined (resp., combined) complexity.*

Proof. Let D be a database and (Q, Σ) an OMQ. Without loss of generality, we assume that (D, Σ) is consistent. It can be easily verified that $D \models (Q, \Sigma)$ iff there exists a minimal explanation for $D \not\models (Q \wedge p(), \Sigma)$ w.r.t. $H = \{p()\}$, where p is a fresh 0-ary predicate. Note that the nulary fact $p()$ is needed to ensure that $D \not\models (Q \wedge p(), \Sigma)$. □

5.4 Relevant Facts for Explanations

In this section, we deal with the problems regarding relevant facts, i.e., facts appearing in at least one minimal explanation. In particular, we analyse the problem of deciding whether a fact is relevant and the problem of deciding whether a set of facts is the set of all the relevant facts.

We start by looking at the problem MinEX-Rel$^{\not\models}$ of deciding whether a fact is relevant or not. We first provide the membership results. The following theorem proves all the upper bounds in Table 5.3, except for those in data complexity for

Table 5.3: Complexity results for MinEX-Rel$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**Data**	***fp*-comb.**	***ba*-comb.**	**Comb.**
$\mathsf{L}_\bot$, $\mathsf{LF}_\bot$, $\mathsf{AF}_\bot$	$\leq$ P	Σ_2^{P}	Σ_2^{P}	PSpace
$\mathsf{S}_\bot$, $\mathsf{SF}_\bot$	$\leq$ P	Σ_2^{P}	Σ_2^{P}	Exp
$\mathsf{A}_\bot$	$\leq$ P	Σ_2^{P}	P^{NExp}	P^{NExp}
$\mathsf{G}_\bot$	NP	Σ_2^{P}	Exp	2Exp
$\mathsf{F}_\bot$, $\mathsf{GF}_\bot$	NP	Σ_2^{P}	Σ_2^{P}	Exp
$\mathsf{WS}_\bot$, $\mathsf{WA}_\bot$	NP	Σ_2^{P}	2Exp	2Exp

FO-languages, for which we need a tighter statement. Intuitively, to decide whether ψ is relevant, it suffices to guess a set E of facts containing ψ (feasible in NP), and then, via an oracle call, check that E is a minimal explanation.

Theorem 5.17. *For any language $\mathcal{L}$ of existential rules, if* Is-MinEX$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* MinEX-Rel$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* $\text{NP}^{\mathcal{C}}$.

Proof. Consider an instance of MinEX-Rel$^{\not\models}$ consisting of a database D, an OMQ (Q, Σ), a finite set of facts H, and a fact ψ. We can guess a set of facts $E \subseteq H$ with $\psi \in E$ in NP, and then using an oracle for Is-MinEX$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ check whether E is a minimal explanation for $D \not\models (Q, \Sigma)$ w.r.t. H. □

This theorem implies all the membership results in Table 5.3, apart from the membership results in P in data complexity. If Is-MinEX$^{\not\models}$ is in D^{P}, then, by Theorem 5.17, MinEX-Rel$^{\not\models}$ is in $\text{NP}^{\text{D}^{\text{P}}}$. Note that $\text{NP}^{\text{D}^{\text{P}}}$ is in Σ_2^{P} since a procedure in NP calling an oracle in D^{P} can be substituted by a procedure in NP calling twice an oracle in NP. Therefore, MinEX-Rel$^{\not\models}$ is in Σ_2^{P} in this case. Similarly, if Is-MinEX$^{\not\models}$ is in D^{Exp}, then MinEX-Rel$^{\not\models}$ is in $\text{NP}^{\text{D}^{\text{Exp}}}$, and a procedure in NP calling an oracle for a problem in D^{Exp} can be substituted by a procedure in NP calling twice an oracle in NExp. Therefore $\text{NP}^{\text{D}^{\text{Exp}}}$ is in NP^{NExp}. As we have seen above $\text{NP}^{\text{NExp}} = \text{P}^{\text{NExp}}$, and thus MinEX-Rel$^{\not\models}$ is in P^{NExp} in this case. Other results follow similarly.

For the FO-rewritable languages, the following theorem proves the P-membership results in data complexity in Table 5.3. To prove this, we again exploit the rewritability of the OMQ (Q, Σ) into an FO query Q_Σ and the possibility to compute in polynomial time the set $\mathcal{D}_{Q_\Sigma}$ containing all the polynomially many sets of facts that can possibly satisfy Q_Σ. With $\mathcal{D}$ at hand, its elements can be considered in turn to verify whether there exists a minimal explanation containing the fact to be tested for relevance.

Theorem 5.18. *For any language $\mathcal{L}$ of existential rules for with* OMQA *is FO-rewritable,* MinEX-Rel$^{\not\models}(\subseteq, \mathrm{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

Proof. Consider an instance of MinEX-Rel$^{\not\models}(\subseteq, \mathrm{UCQ}, \mathcal{L})$ consisting of a database D, an OMQ (Q, Σ), a finite set of facts H, and a fact ψ. For a FO-rewritable language, we can compute $\mathcal{D}_{Q_{\Sigma_T}}$ defined in the proof of Theorem 5.12 in polynomial time. We can now decide whether there exists a minimal explanation for $D \not\models (Q, \Sigma)$ w.r.t. H containing ψ by checking whether there exists a set X of facts in $\mathcal{D}_{Q_{\Sigma_T}}$ with $X \subseteq D \cap H$ and $\psi \in X \cap H$ such that

(i) $X \cap H$ is an explanation for $D \not\models (Q, \Sigma)$ w.r.t. H, and

(ii) for any proper subset X' of $X \cap H$, $D \cup X' \not\models (Q, \Sigma)$.

Since $\mathcal{D}_{Q_{\Sigma_T}}$ is of polynomial size, both entailment and consistency can be checked in polynomial time in data complexity. The procedure above is in P. □

We now consider the hardness results. To study the hardness of MinEX-Rel$^{\not\models}$, we can resort to its "positive" variant. In Chapter 3, we studied the problem of deciding whether a fact ψ belongs to some minimal explanation for $D \models (Q, \Sigma)$, called MinEX-Rel. It is possible to show that MinEX-Rel reduces to MinEX-Rel$^{\not\models}$, thereby obtaining all the hardness results in Table 5.3.

Theorem 5.19. *For any language $\mathcal{L}$ of existential rules,* MinEX-Rel$(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *is reducible in polynomial time to* MinEX-Rel$^{\not\models}(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *in data, fp-combined, ba-combined, and combined complexity.*

Proof. Let D be a database, (Q, Σ) an OMQ, where Σ is a set of TGDs, and $\psi \in D$. As already mentioned in the proof of Theorem 5.9, for any subset $E \subseteq D$, we have that E is a minimal explanation for $D \models (Q, \Sigma)$ iff $E \cup \{p()\}$ is a MinEX for $\emptyset \not\models (Q \wedge p(), \Sigma)$ w.r.t. $H = D \cup \{p()\}$, where p is a fresh 0-ary predicate. Thus, a fact ψ belongs to some minimal explanation for $D \models (Q, \Sigma)$ iff ψ belongs to some MinEX for $\emptyset \not\models (Q \wedge p(), \Sigma)$ w.r.t. $D \cup \{p()\}$. □

We now analyse the problem MinEX-AllRel$^{\not\models}$ of deciding whether a set contains all and only the relevant facts. We focus on the membership results first. The following theorem proves all the upper bounds in Table 5.4. Intuitively, to check that H' is the set of all and only the relevant facts, it suffices to check that all facts in H' are relevant, and all facts outside H' are not relevant.

Table 5.4: Complexity results for MinEX-AllRel$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**Data**	*fp*-**comb.**	*ba*-**comb.**	**Comb.**
$\mathsf{L}_\bot$, $\mathsf{LF}_\bot$, $\mathsf{AF}_\bot$	$\leq$ P	D^P_2	D^P_2	PSpace
$\mathsf{S}_\bot$, $\mathsf{SF}_\bot$	$\leq$ P	D^P_2	D^P_2	Exp
$\mathsf{A}_\bot$	$\leq$ P	D^P_2	P^{NExp}	P^{NExp}
$\mathsf{G}_\bot$	D^P	D^P_2	Exp	2Exp
$\mathsf{F}_\bot$, $\mathsf{GF}_\bot$	D^P	D^P_2	D^P_2	Exp
$\mathsf{WS}_\bot$, $\mathsf{WA}_\bot$	D^P	D^P_2	2Exp	2Exp

Theorem 5.20. *For any language $\mathcal{L}$ of existential rules, if* MinEX-Rel$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$ then* MinEX-AllRel$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *can be decided by a linear number of checks in $\mathcal{C}$ and a linear number of checks in co-$\mathcal{C}$.*

Proof. To decide whether H' contains exactly all relevant facts for $D \not\models (Q, \Sigma)$ w.r.t. H, we need to check that (i) every fact in H' is relevant and (ii) every fact in $H \setminus H'$ is not relevant. The condition (i) can be checked with a linear number of $\mathcal{C}$ checks and the condition (ii) can be checked with a linear number of co-$\mathcal{C}$ checks. This implies the result. □

We can see that the above theorem implies all the membership results in Table 5.4 as follows. If MinEX-Rel$^{\not\models}$ is in NP, then MinEX-AllRel$^{\not\models}$ can be solved with a linear number of NP tests and a linear number coNP tests. Since all these tests are independent, all NP tests can be combined as a single NP tests (by concatenating the certificates) and, similarly, all coNP tests as a single coNP test. Therefore, MinEX-AllRel$^{\not\models}$ is in $\text{D}^\text{P} = \text{NP} \wedge \text{coNP}$ in this case. Similarly, if MinEX-Rel$^{\not\models}$ is in Σ^P_2, we have that MinEX-AllRel$^{\not\models}$ can be solved with a linear number of Σ^P_2 and a linear number of Π^P_2 checks. Similarly, as above, a linear number of Σ^P_2 checks can be combined as a single Σ^P_2 check and a linear number of Π^P_2 checks can be combined as a single Π^P_2 check. Therefore MinEX-AllRel$^{\not\models}$ is in $\text{D}^\text{P}_2 = \Sigma^\text{P}_2 \wedge \Pi^\text{P}_2$. If MinEX-Rel$^{\not\models}$ is in a deterministic class $\mathcal{C} \in \{\text{P}, \text{Exp}, \text{2Exp}, \text{P}^{\text{NExp}}\}$, then co-$\mathcal{C} = \mathcal{C}$ and so MinEX-AllRel$^{\not\models}$ can be solved with two $\mathcal{C}$ tests. Since $\text{P} \subseteq \mathcal{C}$, we have that the two $\mathcal{C}$ tests can be combined as a single $\mathcal{C}$ test. Therefore MinEX-AllRel$^{\not\models}$ is in $\mathcal{C}$.

Now we show the hardness results for MinEX-AllRel$^{\not\models}$, presented in Table 5.4. the following theorem proves all the D^P-hardness results in data complexity in Table 5.4 via a reduction from the classical D^P-complete problem Sat-Unsat. We note

that some of the constructions of this proof are similar as in the proofs of Theorem 3.19 and Theorem 5.13. We refer the reader to these proofs for more intuition on the construction. In this construction, in a fixed program, there are rules to check the satisfiability of the formulas. In H, there are facts to encode possible truth assignments for the Boolean variables X, plus some additional facts to recognise the satisfaction of ϕ and ψ, for the specific truth assignments in the candidate explanations. The idea is to have all facts in H to be relevant, apart from the one associated with the satisfaction of ψ.

Theorem 5.21. MINEX-ALLREL$^{\not\models}$ $(\subseteq, \mathrm{BCQ}, \mathsf{GF}_\bot)$ *is* D^P-hard *in data complexity.*

Proof. We provide a reduction from the D^P-complete problem SAT-UNSAT, introduced in the proof of Theorem 3.12. Let $\Phi = (\varphi, \psi)$ be an instance of SAT-UNSAT, where $\varphi = c_1^\varphi \wedge \cdots \wedge c_{m_\varphi}^\varphi$ is a 3CNF formula over variables $X^\varphi = \{x_1^\varphi, \ldots, x_{n_\varphi}^\varphi\}$ and $\psi = c_1^\psi \wedge \cdots \wedge c_{m_\psi}^\psi$ is a 3CNF formula over variables $X^\psi = \{x_1^\psi, \ldots, x_{n_\psi}^\psi\}$. We derive an instance of MINEX-ALLREL$^{\not\models}(\subseteq, \mathrm{BCQ}, \mathsf{GF}_\bot)$ consisting of a database D_Φ, an OMQ (Q, Σ), a finite set of facts H_Φ, and a set $H'_\Phi \subseteq H_\Phi$, where only D_Φ, H_Φ, and H'_Φ depend on Φ, as follows.

We start with the database D_Φ, which is built as follows. For each clause c_j^α, where $\alpha \in \{\varphi, \psi\}$ and $1 \leq j \leq m_\alpha$, we add to D_Φ a fact $succ\_cl^\alpha(j-1, j)$, where $j-1$ and j are numeric constants, intuitively stating that the j^{th} clause is the successor of the $(j-1)^{\text{th}}$ clause. Also, for each clause c_j^α, we add a fact encoding it. For example, for a clause $(x_p \vee x_q \vee \neg x_r)$, there is a fact $cl^\alpha(j, x_p, p, x_q, p, x_r, n)$ in the database D_Φ, where p and n are constants representing whether a variable appears as a positive or negative literal, respectively. For each $\alpha \in \{\varphi, \psi\}$, we add the facts $satchain^\alpha(0)$ and $maxcl^\alpha(m_\alpha)$ to the database D_Φ.

We construct the set H_Φ of hypotheses as follows. For each variable x_i^α, $\alpha \in \{\varphi, \psi\}$ and $1 \leq i \leq n_\alpha$, add to H_Φ the facts $val^\alpha(x_i^\alpha, 0)$ and $val^\alpha(x_i^\alpha, 1)$, which capture the truth assignments to a variable x_i^α. We also add the fact $jolly()$, which will be used to bypass a satisfiability check. We add to H_Φ two more facts, which will need to be present when the satisfiability check is not bypassed. These facts are $p^\varphi(), p^\psi()$. Then, $H'_\Phi = H_\Phi \setminus \{p^\psi()\}$.

The program Σ is defined as follows. For each $\alpha \in \{\varphi, \psi\}$, we add the rules, whose role is similar as in the proofs of Theorem 3.19 and Theorem 5.13:

$$val^\alpha(X, 1), val^\alpha(X, 0) \to \bot,$$
$$cl^\alpha(J, X, p, \_, \_, \_, \_), val^\alpha(X, 1) \to satcl^\alpha(J),$$
$$cl^\alpha(J, X, n, \_, \_, \_, \_), val^\alpha(X, 0) \to satcl^\alpha(J),$$

$$cl^\alpha(J,\_,\_,Y,p,\_,\_), val^\alpha(Y,1) \to satcl^\alpha(J),$$
$$cl^\alpha(J,\_,\_,Y,n,\_,\_), val^\alpha(Y,0) \to satcl^\alpha(J),$$
$$cl^\alpha(J,\_,\_,\_,\_,Z,p), val^\alpha(Z,1) \to satcl^\alpha(J),$$
$$cl^\alpha(J,\_,\_,\_,\_,Z,n), val^\alpha(Z,0) \to satcl^\alpha(J),$$
$$satchain^\alpha(I), succ\_cl^\alpha(I,J), satcl^\alpha(J) \to satchain^\alpha(J),$$
$$maxcl^\alpha(M), satchain^\alpha(M) \to sat^\alpha().$$

We further add the new rules how the nulary fact $query()$ might be entails. For each $\alpha \in \{\psi, \phi\}$, we add

$$sat^\alpha(), p^\alpha() \to query(),$$
$$val^\alpha(X,Y), jolly() \to query().$$

We take the query to be $Q = query()$. Observe that the OMQ (Q, Σ) does not depend on φ, and Σ is in $\mathsf{GF}_\bot$, that is, guarded and full. Before proving the correctness of the reduction, we point out that $jolly()$ and all facts $val^\alpha(x_i^\alpha, \star)$, where $\star \in \{0,1\}$, are relevant. In fact, it is easy to see that, for $\star \in \{0,1\}$, the set $E = \{val^\alpha(x_i^\alpha, \star), jolly()\}$ is a MinEX for $D_\Phi \not\models (Q,\Sigma)$ w.r.t. H_Φ.

We now show that φ is satisfiable and ψ is unsatisfiable iff H'_Φ is an inclusion-minimal set containing all relevant facts for $D_\Phi \not\models (Q,\Sigma)$ w.r.t. H_Φ, that is, all facts in H'_Φ are relevant and $p^\psi()$ is not relevant.

($\Rightarrow$) We first show that every fact in H'_Φ is relevant. As shown above, $jolly()$ and all facts $val^\alpha(x_i^\alpha, \star)$, where $\star \in \{0,1\}$, are relevant. We now show that also $p^\varphi()$ is relevant. Since φ is satisfiable, let $\sigma : X^\varphi \to \{T, F\}$ be a truth assignment satisfying it. We define $D_\sigma = \{val^\varphi(x_i^\varphi, 1) \mid \sigma(x_i^\varphi) = T\} \cup \{val^\varphi(x_i^\varphi, 0) \mid \sigma(x_i) = F\}$. Let $E = D_\sigma \cup \{p^\varphi()\}$. It is easy to see that $(D_\Phi \cup E, \Sigma)$ is consistent and $D_\Phi \cup E \models (Q,\Sigma)$, as σ satisfies φ.

Furthermore, there cannot be a subset E' of E s.t. $D_\Phi \cup E' \models (Q,\Sigma)$ and E' does not contain $p^\varphi()$, because otherwise Q would not be entailed anymore (as $jolly() \notin E$ and $p^\psi() \notin E$). Thus, there is a set $E' \subseteq E$ that is a minimal explanation and contains $p^\varphi()$.

It remains to show that $p^\psi()$ is not relevant. For a contradiction, suppose that $p^\psi()$ is relevant and let E be a minimal explanation containing it. If $D_\Phi \cup E \models (Q,\Sigma)$ via $sat^\psi(), p^\psi() \to query()$, then ψ is satisfiable: a contradiction. If $D_\Phi \cup E \models (Q,\Sigma)$

via $sat^{\varphi}(), p^{\varphi}() \rightarrow query()$ or $val^{\alpha}(X,Y), jolly() \rightarrow query()$, then $p^{\psi}()$ is not needed, that is, E is not minimal: a contradiction.

($\Leftarrow$) We prove the contrapositive. Suppose φ is unsatisfiable or ψ is satisfiable. Reasoning as above, it is easy to see that if φ is unsatisfiable, then $p^{\varphi}()$ is not relevant. Likewise, if ψ is satisfiable, then $p^{\psi}()$ is relevant. □

The following theorem proves all the $\mathrm{D}_2^{\mathrm{p}}$ hardness results in Table 5.4 in *fp*-combined complexity. It also gives hardness results matching the upper bounds for $\mathsf{L}_{\perp}$, $\mathsf{F}_{\perp}$, $\mathsf{S}_{\perp}$, and their specializations in *ba*-combined complexity. We show a reduction from the prototypical $\mathrm{D}_2^{\mathrm{p}}$-complete problem $\mathrm{QBF}_{2,\forall,\neg}^{CNF} \wedge \mathrm{QBF}_{2,\exists}^{CNF}$ of deciding the validity of two quantified Boolean formulas $\Phi = \exists X \forall Y \neg \phi(X,Y)$ and $\Psi = \forall X \exists Y \psi(X,Y)$.

Theorem 5.22. *For any language $\mathcal{L}$ of existential rules,* MinEX-AllRel$^{\models}(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *is* $\mathrm{D}_2^{\mathrm{p}}$-hard *in fp-combined complexity.*

Proof. We reduce to MinEX-AllRel$^{\models}$ the following prototypical $\mathrm{D}_2^{\mathrm{p}}$-hard problem:
Problem: $\mathrm{QBF}_{2,\forall,\neg}^{CNF} \wedge \mathrm{QBF}_{2,\exists}^{CNF}$
Input: Two independent quantified Boolean formulas $\Phi = \exists X \forall Y \neg \phi(X,Y)$ and $\Psi = \forall X \exists Y \psi(X,Y)$, where ϕ and ψ are a (non-quantified) 3CNF formulas.
Question: Are Φ and Ψ are both valid?

Notice that by an argument similar to the one in the proof of Theorem 3.9 in [113], X and Y can be assumed w.l.o.g. to be the same in $\phi(X,Y)$ and $\psi(X,Y)$.

Let $I = (\Phi, \Psi)$ be an instance of $\mathrm{QBF}_{2,\forall,\neg}^{CNF} \wedge \mathrm{QBF}_{2,\exists}^{CNF}$. We build an instance of MinEX-AllRel$^{\models}$ consisting of a database D_I, an OMQ (Q_I, Σ), a finite set of facts H_I, and a set $H_I' \subseteq H_I$, as follows—only D_I, Q_I, H_I, and H_I' depend on I.

By an argument similar to the one in the proof of Theorem 3.9 in [113], we can assume w.l.o.g. that variables X and Y of the formulas Φ and Ψ are the same, and that ϕ and ψ have the same number of clauses.

We build the database D_I as follows. There are facts in D_I that are used to impose the consistency of the truth assignments to the literals:

$$\begin{array}{llll} simlit(0,0), & simlit(0,2), & opplit(0,1), & simlit(2,2), \\ simlit(1,1), & simlit(1,2), & opplit(1,0), & opplit(2,2), \end{array}$$

where 0 and 1 are constants representing Boolean values false and true, respectively, and 2 is an additional constants used as a "jolly" (intuitively, it will be used to bypass, in some circumstances, the check of the satisfaction of $\phi(X,Y)$ and $\psi(X,Y)$). The predicate $simlit(\cdot,\cdot)$ will be used to impose that when a variable appears twice as

a positive or a negative literal in two different places in the formulas, then the two literals must have the same truth value. On the other hand, the predicate $\mathit{opplit}(\cdot,\cdot)$ will be used to impose that when a variable appears as a positive literal in one place in the formula and as a negative literal in another place of the formula, then the two literals must have different truth values.

There are facts in D_I that are used to select possible ways of satisfying the clauses in the formula:

$$\begin{array}{llll} \mathit{clsat}^{\alpha}(1,1,1), & \mathit{clsat}^{\alpha}(1,1,0), & \mathit{clsat}^{\alpha}(1,0,1), & \mathit{clsat}^{\alpha}(1,0,0), \\ \mathit{clsat}^{\alpha}(0,1,1), & \mathit{clsat}^{\alpha}(0,1,0), & \mathit{clsat}^{\alpha}(0,0,1). & \end{array}$$

where 0 and 1 are constants with the same meaning as above, and the superscript α is one among $\{\phi,\psi\}$. The predicate $\mathit{clsat}^{\alpha}(\cdot,\cdot,\cdot)$ states what truth assignments to the *literals* (and not to the variables) satisfy a clause of ϕ and ψ.

The program Σ contains only two NCs. For each $\alpha \in \{\psi,\phi\}$, we add to Σ the NC: $\mathit{val}^{\alpha}(X,0), \mathit{val}^{\alpha}(X,1) \rightarrow \bot$. Since Σ has no TGDs, Σ is trivially $\mathsf{LF}_{\bot}$, $\mathsf{AF}_{\bot}$, and $\mathsf{SF}_{\bot}$. The set H_I is as follows. For each variable $x_i \in X$, in H_I there are facts:

$$\mathit{val}(x_i,0) \qquad\qquad \mathit{val}(x_i,1),$$

where x_i is a constant representing the respective variable in X, and 0 and 1 are constants with the meaning as above. We also add to H_I the facts

$$\mathit{clsat}^{\psi}(2,2,2), \qquad\qquad \mathit{clsat}^{\phi}(2,2,2).$$

Intuitively, the facts $\mathit{clsat}^{\alpha}(2,2,2)$ allow to bypass, in some circumstances, the check of the satisfaction of $\phi(X,Y)$ and $\psi(X,Y)$.

We take the candidate set of all the relevant facts to be $H'_I = H_I \setminus \{\mathit{clsat}^{\psi}(2,2,2)\}$. To conclude, let us define the query Q_I. The query checks the satisfiability of $\phi(X,Y)$ and $\psi(X,Y)$ once a truth assignment for X has been fixed by an explanation. The first piece of the query "reads" onto the variables T_i the truth assignments for the variables in X encoded in the explanation:

$$\mathit{assignX} \equiv \bigwedge_{i=1}^{n} \mathit{val}(x_i,T_i).$$

Below we use the following notation: $\ell^{\phi}_{j,k}$ (resp., $\ell^{\psi}_{j,k}$) is the k^{th} literal in the j^{th} clause, c_j, of formula $\phi(X,Y)$ (resp., $\psi(X,Y)$), and $v^{\phi}_{j,k}$ (resp., $v^{\psi}_{j,k}$) is the variable of

$\ell^{\phi}_{j,k}$ (resp., $\ell^{\psi}_{j,k}$). The second piece "copies" the assignment of each variable x_i onto an occurrence of x_i as a positive literal in the formulas $\phi(X,Y)$ and $\psi(X,Y)$:

$$copy^{\alpha} \equiv \bigwedge_{i=1}^{n} simlit(T_i, T^{\alpha}_{j,k}),$$

where the superscript α is one among $\{\phi, \psi\}$, and in each predicate $simlit(T_i, T^{\alpha}_{j,k})$, $T^{\phi}_{j,k}$ (resp., $T^{\psi}_{j,k}$) is a variable for the Boolean value of the literal $\ell^{\phi}_{j,k} = x_i$ (resp., $\ell^{\psi}_{j,k} = x_i$) in $\phi(X,Y)$ (resp., $\psi(X,Y)$). Observe that, in order for $copy^{\alpha}$ to work properly, each variable x_i must appear as a positive literal in the clauses of $\phi(X,Y)$ and $\psi(X,Y)$ at least once. This can be assumed without loss of generality, because if x_i always appears as a negative literal in all the clauses of $\phi(X,Y)$ or $\psi(X,Y)$, then we can replace all the occurrences of the negative literal $\neg x_i$ with the positive literal x_i without altering the satisfiability properties of $\phi(X,Y)$ or $\psi(X,Y)$.

The third piece forces the assignment to $T^{\alpha}_{j,k}$ (simulating the assignments to the literals) to be consistent. In the notation below, $\ell^{\alpha}_{j,k} \sim \ell^{\alpha}_{j',k'}$ means that $v^{\alpha}_{j,k} = v^{\alpha}_{j',k'}$ and the literals $\ell^{\alpha}_{j,k}$ and $\ell^{\alpha}_{j',k'}$ are both positive or negative, while $\ell^{\alpha}_{j,k} \nsim \ell^{\alpha}_{j',k'}$ means that $v^{\alpha}_{j,k} = v^{\alpha}_{j',k'}$ and one literal is positive and the other is negative. Thus the third piece is

$$consist^{\alpha} \equiv \bigwedge_{\ell^{\alpha}_{j,k} \sim \ell^{\alpha}_{j',k'}} simlit(T^{\alpha}_{j,k}, T^{\alpha}_{j',k'}) \wedge \bigwedge_{\ell^{\alpha}_{j,k} \nsim \ell^{\alpha}_{j',k'}} opplit(T^{\alpha}_{j,k}, T^{\alpha}_{j',k'}),$$

where the superscript α is a one among $\{\phi, \psi\}$, and $T^{\alpha}_{j,k}$ is a variable with the same meaning as above.

The last piece checks the satisfiability of $\phi(X,Y)$ and $\psi(X,Y)$. For each $\alpha \in \{\psi, \phi\}$, we add to the program Σ:

$$satisfied^{\alpha} \equiv \bigwedge_{j=1}^{m} clsat^{\alpha}(T^{\alpha}_{j,1}, T^{\alpha}_{j,2}, T^{\alpha}_{j,3}),$$

To conclude, the query Q_I is a follows:

$$Q_I = assignX, copy^{\phi}, consist^{\phi}, satisfied^{\phi}, copy^{\psi}, consist^{\psi}, satisfied^{\psi}.$$

We now show that $I = (\Phi, \Psi)$ is a 'yes'-instance of $\mathrm{QBF}^{CNF}_{2,\forall,\neg} \wedge \mathrm{QBF}^{CNF}_{2,\exists}$ iff D_I, (Q_I, Σ), H_I, and H'_I form a 'yes'-instance of MinEX-AllRel$^{\not\models}$. The intuition for the proof is the following. Since $assignX$ is in the query, and the facts of the predicate $val(\cdot,\cdot)$ are only in H, every MinEX must contain $val(x_i, 0)$ or $val(x_i, 1)$ for each Boolean variable $x_i \in X$. Furthermore, due to the NC '$val^{\alpha}(X,0), val^{\alpha}(X,1) \rightarrow \bot$',

there is at most one fact among $val(x_i, 0)$ or $val(x_i, 1)$, for each Boolean variable $x_i \in X$, in any explanation. This means that any explanation encodes a complete truth assignment for the Boolean variables in X. Moreover, observe that adding the fact $clsat^{\phi}(2,2,2)$ (resp., $clsat^{\psi}(2,2,2)$) to *any* set E of facts $val(\cdot,\cdot)$ encoding a complete truth assignment for X is such that $D_I \cup E \cup \{clsat^{\phi}(2,2,2)\} \models (satisfied^{\phi}, \Sigma)$ (resp., $D_I \cup E \cup \{clsat^{\psi}(2,2,2)\} \models (satisfied^{\psi}, \Sigma)$). However, although $E \cup \{clsat^{\phi}(2,2,2), clsat^{\psi}(2,2,2)\}$ is an explanation for $D_I \not\models (Q_I, \Sigma)$, it is not guaranteed to be a *minimal* one. Indeed, if the truth assignment σ_E encoded in E is such that $\phi(X/\sigma_E, Y)$ (resp., $\psi(X/\sigma_E, Y)$) is satisfiable, then $clsat^{\phi}(2,2,2)$ (resp., $clsat^{\psi}(2,2,2)$) is not necessary to be present in the explanation to satisfy the part $satisfied^{\phi}$ (resp., $satisfied^{\psi}$) of Q_I. The properties above show that all the facts $val(\cdot,\cdot)$ are relevant, while the relevance of $clsat^{\phi}(2,2,2)$ and $clsat^{\psi}(2,2,2)$ depends on the validity of Φ and Ψ.

Note that checking the validity of $\Phi = \exists X \forall Y \neg\phi(X,Y)$ is tantamount to checking whether there exists a truth assignment $\widetilde{\sigma}_X$ to X such that $\phi(X/\widetilde{\sigma}_X, Y)$ is not satisfiable. On the other hand, checking the validity of $\Psi = \forall X \exists Y \psi(X,Y)$ is tantamount to checking whether, for all truth assignments $\widetilde{\sigma}_X$ to X, $\phi(X/\widetilde{\sigma}_X, Y)$ is satisfiable.

($\Rightarrow$) If (Φ, Ψ) is a 'yes'-instance of $\mathrm{QBF}^{CNF}_{2,\forall,\neg} \wedge \mathrm{QBF}^{CNF}_{2,\exists}$, then Φ and Ψ are both valid. Since Φ is valid, there exists an assignment $\widetilde{\sigma}_X$ such that $\phi(X/\widetilde{\sigma}_X, Y)$ is not satisfiable, and hence the minimal explanation encoding $\widetilde{\sigma}_X$ requires the presence of $clsat^{\phi}(2,2,2)$ in order to satisfy the part $satisfied^{\phi}$ of the query Q_I. Thus, $clsat^{\phi}(2,2,2)$ is a relevant fact. Remember that $clsat^{\phi}(2,2,2) \in H'_I$ by construction. To conclude, since H'_I excludes only $clsat^{\psi}(2,2,2)$, H'_I is the set of all and only the relevant facts iff $clsat^{\psi}(2,2,2)$ is not relevant. Since we are assuming that Ψ is valid, for any truth assignment $\widetilde{\sigma}_X$, $\psi(X/\widetilde{\sigma}_X, Y)$ is satisfiable. This implies that no minimal explanations contain $clsat^{\psi}(2,2,2)$, which hence is not relevant.

($\Leftarrow$) If (Φ, Ψ) is a 'no'-instance of $\mathrm{QBF}^{CNF}_{2,\forall,\neg} \wedge \mathrm{QBF}^{CNF}_{2,\exists}$, then Φ or Ψ is not valid. If Φ is not valid, then, for all truth assignments $\widetilde{\sigma}_X$, $\phi(X/\widetilde{\sigma}_X, Y)$ is satisfiable, and hence no minimal explanations contains $clsat^{\phi}(2,2,2)$. However, $clsat^{\phi}(2,2,2) \in H'_I$ by construction, and hence H'_I contains a fact that is not relevant. If Ψ is not valid, then there exists a truth assignment $\widetilde{\sigma}_X$ such that $\phi(X/\widetilde{\sigma}_X, Y)$ is not satisfiable. The minimal explanation encoding $\widetilde{\sigma}_X$ must include $clsat^{\psi}(2,2,2)$ in order to satisfy the part $satisfied^{\psi}$ of Q_I. Hence, $clsat^{\psi}(2,2,2)$ is a relevant fact that, by construction, does not belong to H'_I. □

The remaining hardness results in *ba*-combined complexity as well as all the hardness results in combined complexity are established via the following theorem, which shows that MinEX-Exists$^{\not\models}$ can be reduced to the complement of MinEX-AllRel$^{\not\models}$.

Theorem 5.23. *For any language $\mathcal{L}$ of existential rules, we have that the problem* MinEX-Exists$^{\not\models}(\subseteq, \text{BCQ}, \mathcal{L})$ *is reducible in polynomial time to the complement of* MinEX-AllRel$^{\not\models}(\subseteq, \text{BCQ}, \mathcal{L})$.

Proof. Consider an instance of MinEX-Exists$^{\not\models}$ consisting of a database D, an OMQ (Q, Σ), and a finite set of facts H. We derive an instance of the complement of MinEX-AllRel$^{\not\models}$. We consider a database $D^* := D$, an OMQ (Q^*, Σ^*), where $Q^* := Q \wedge p()$ and $\Sigma^* := \Sigma$, a finite set of hypotheses $H^* := H \cup \{p()\}$, and the candidate set of all relevant hypotheses $H' := \emptyset$, where p is a fresh 0-ary predicate.

We show that there exists a MinEX E for $D \not\models (Q, \Sigma)$ w.r.t. H iff H' is *not* the set of all relevant fact for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^*.

($\Rightarrow$) If there exists a minimal explanation E for $D \not\models (Q, \Sigma)$ w.r.t. H, then it is easy to see that $E \cup \{p()\}$ is a minimal explanation for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^*, and thus H' is not a minimal set containing all relevant facts for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^*, since $p() \notin H'$.

($\Leftarrow$) If there does not exist a a minimal explanation E for $D \not\models (Q, \Sigma)$ w.r.t. H, then there does not exist a minimal explanation for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^*, which implies that there are no relevant facts. Therefore, $H' = \emptyset$ is a minimal set containing all the relevant facts for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^*. □

5.5 Necessary Facts for Explanations

In this section, we study the problems regarding necessary facts, i.e., facts appearing in all the minimal explanations for negative query answers. In particular, we analyse the problem of deciding whether a fact is necessary and the problem of deciding whether a set of facts is exactly the set of all the necessary facts.

We first focus on the problem MinEX-Nec$^{\not\models}$ of deciding whether a fact is necessary. As for the membership results, the following theorem proves all the upper bounds in Table 5.5 but the P and coNP ones, for which we provide tighter results later in this section. Intuitively, we can *dis*prove that a fact ψ is necessary by checking that there is a (non-necessarily minimal) explanation excluding ψ. Hence, we can answer that a fact ψ is *not* necessary by guessing a set $E \subseteq H \setminus \{\psi\}$ of facts (feasible in NP), and then checking that E is consistent and entails the OMQ.

Table 5.5: Complexity results for MinEX-Nec$^{\not\models}(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**Data**	***fp*-comb.**	***ba*-comb.**	**Comb.**
$\mathsf{L}_\bot$, $\mathsf{LF}_\bot$, $\mathsf{AF}_\bot$	$\leq$ P	coNP	Π_2^{P}	PSpace
$\mathsf{S}_\bot$, $\mathsf{SF}_\bot$	$\leq$ P	coNP	Π_2^{P}	Exp
$\mathsf{A}_\bot$	$\leq$ P	coNP	P^{NExp}	P^{NExp}
$\mathsf{G}_\bot$	coNP	coNP	Exp	2Exp
$\mathsf{F}_\bot$, $\mathsf{GF}_\bot$	coNP	coNP	Π_2^{P}	Exp
$\mathsf{WS}_\bot$, $\mathsf{WA}_\bot$	coNP	coNP	2Exp	2Exp

Theorem 5.24. *For any language $\mathcal{L}$ of existential rules, if* OMQA(UCQ, $\mathcal{L}$) *is in the complexity class $\mathcal{C}$, then* MinEX-Nec$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in co-*$(\text{NP}^{\mathcal{C}})$.

Proof. Let us consider the complementary problem of MinEX-Nec$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$. Let D, (Q, Σ), H and $\psi \in H$ be an instance of this problem. To decide whether there exists a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H that does *not* contain ψ, we need to guess a subset E of $H \setminus \{\psi\}$ that does not contain ψ and check whether

(i) $(D \cup E, \Sigma)$ is consistent, and

(ii) $D \cup E \models (Q, \Sigma)$.

Notice that the task (i) can be done by checking whether, for every $\nu \in \Sigma_{NC}$, $D \cup E \not\models (Q_\nu, \Sigma_T)$, where Q_ν is the body of ν. This check takes a linear number of co-$\mathcal{C}$ tests. The task (ii) takes a single $\mathcal{C}$ check. Notice that it is not necessary to check whether E is minimal. Therefore, the complement of MinEX-Nec$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ can be solved in $\text{NP}^{\mathcal{C}}$, and thus MinEX-Nec$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ is in co-$(\text{NP}^{\mathcal{C}})$. □

The above theorem implies all the membership results in Table 5.5, apart from the membership in coNP in *fp*-combined complexity and the membership in P in data complexity. For example, if OMQA is P, we have that MinEX-Nec$^{\not\models}$ is in co-(NP^{P}) = coNP. If OMQA is in MinEX-Nec$^{\not\models}$ is in co-$(\text{NP}^{\text{NP}}) = \Sigma_2^{\text{P}}$. Other results follow similarly.

The following theorem shows that MinEX-Nec$^{\not\models}$ can be decided in coNP in *fp*-combined complexity for any language $\mathcal{L}$ of existential rules. The result is obtained from the proof of Theorem 5.24 with the additional observation that in *fp*-combined complexity, for the languages of existential rules considered, checking whether a set of facts is consistent is feasible in P. This holds as the negative constraints are fixed.

Theorem 5.25. *For any language $\mathcal{L}$ of existential rules,* MinEX-Nec$^{\not\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in* coNP *in fp-combined complexity.*

Proof. Let us consider the complementary problem. in *fp*-combined complexity, since Σ_{NC} is fixed, the condition (i) in the proof of Theorem 5.24 can indeed be checked in polynomial time as Q_ν's are fixed and OMQA answering is tractable in data complexity (cf. Table 2.2). Checking the condition (ii) in the proof of Theorem 5.24 is in NP for all the languages we consider. As we can guess E along with the certificate witnessing that $D \cup E \models (Q, \Sigma)$, the overall procedure is in NP. □

We conclude the discussion of the membership results for MinEX-Nec$^{\not\models}$ by showing that the problem is in P in data complexity for all the FO-rewritable languages considered, namely, $\mathsf{L}_\bot$, $\mathsf{S}_\bot$, and $\mathsf{A}_\bot$ (as well as their full restrictions). We show this by extending the proof of Theorem 5.12 with the ideas from Theorem 3.18. Note that, by Theorem 3.18, we can compute the set of all MinEXs in polynomial time and thus we can decide in polynomial time whether every MinEX contains the distinguished fact.

Theorem 5.26. *For any FO-rewritable language $\mathcal{L}$,* MinEX-Nec$^{\not\models}(\subseteq, \mathrm{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

Proof. Consider an instance of MinEX-Rel$^{\not\models}$ consisting of a database D, an OMQ (Q, Σ), a finite set of facts H, and a fact ψ. In the proof of Theorem 5.12, we have constructed the set $\mathcal{D}_{Q_T}$ of all the groundings of the query rewriting Q_{Σ_T}, which contains all MinEXs. Note that, for each set $X \in \mathcal{D}_{Q_T}$, we can decide in polynomial time whether it is a (not-necessarily minimal) explanation by checking whether $X \subseteq D \cup H$ and $(D \cup X, \Sigma)$ is consistent. Therefore, we can construct the set $\mathcal{D}^{\mathrm{EX}}_{Q_T}$ of all explanations in $\mathcal{D}_{Q_T}$ in polynomial time. In further polynomial time, we can decide which sets in $\mathcal{D}^{\mathrm{EX}}_{Q_T}$ are minimal, and thus compute the set of all MinEXs $\mathcal{E}$ for $D \not\models (Q, \Sigma)$ w.r.t. H.

Now we can decide if there is a MinEX that contains ψ by checking, in polynomial time, whether any $E \in \mathcal{E}$ contains ψ. As the above process is a combination of polynomial time tasks, the whole task is in polynomial time. □

Now we proceed with the hardness results for MinEX-Nec$^{\not\models}$. We show that the complement of MinEX-Exists$^{\not\models}$ can be reduced to MinEX-Nec$^{\not\models}$. This observation covers all the hardness results in Table 5.5

Theorem 5.27. *The complement of* MinEX-Exists$^{\not\models}(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *is polynomial-time reducible to* MinEX-Nec$^{\not\models}(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *for any language $\mathcal{L}$ of existential rules.*

Table 5.6: Complexity results for MinEX-AllNec$^{\models}(\subseteq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**Data**	*fp*-**comb.**	*ba*-**comb.**	**Comb.**
$\mathsf{L}_\bot$, $\mathsf{LF}_\bot$, $\mathsf{AF}_\bot$	$\leq$ P	D^P	D_2^P	PSpace
$\mathsf{S}_\bot$, $\mathsf{SF}_\bot$	$\leq$ P	D^P	D_2^P	Exp
$\mathsf{A}_\bot$	$\leq$ P	D^P	P^{NExp}	P^{NExp}
$\mathsf{G}_\bot$	D^P	D^P	Exp	2Exp
$\mathsf{F}_\bot$, $\mathsf{GF}_\bot$	D^P	D^P	D_2^P	Exp
$\mathsf{WS}_\bot$, $\mathsf{WA}_\bot$	D^P	D^P	2Exp	2Exp

Proof. Let D be a database, (Q, Σ) be an OMQ, and H be a finite set of facts, which together form an instance of the complement of MinEX-Exists$^{\models}(\subseteq, \text{BCQ}, \mathcal{L})$. Let p be a 0-ary fresh predicate. It is easy to see that there does *not* exist a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H iff $p()$ belongs to every MinEX for $D \not\models (Q, \Sigma)$ w.r.t. $H \cup \{p()\}$. □

We now study the problem MinEX-AllNec$^{\models}$ of deciding whether a given set is exactly the set of all the necessary facts for explaining an OMQ non-entailment. We start by looking at the membership results for the query language UCQ. The following theorem proves all upper bounds in Table 5.6. Intuitively, to check that H' is the set of all and only the necessary facts, it suffices to check that all facts in H' are necessary and all facts outside of H' are not necessary.

Theorem 5.28. *For any language $\mathcal{L}$ of existential rules, if we have that the problem* MinEX-Nec$^{\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* MinEX-AllNec$^{\models}(\subseteq, \text{UCQ}, \mathcal{L})$ *can be decided by a linear number of checks in $\mathcal{C}$ and a linear number of checks in co-$\mathcal{C}$.*

Proof. Let D be a database, (Q, Σ) be an $\mathcal{L}$-OMQ, H be a set of hypotheses, and $H' \subseteq H$ a candidate set of all the necessary facts, all of which form an instance of MinEX-AllNec$^{\models}(\subseteq, \text{UCQ}, \mathcal{L})$. To decide whether H' contains exactly all the necessary facts for $D \not\models (Q, \Sigma)$ w.r.t. H, we need to check that (i) every fact in H' is necessary and (ii) every fact in $H \setminus H'$ is not necessary. Observe that the condition (i) can be checked with a linear number of $\mathcal{C}$ calls and the condition (ii) can be checked with a linear number of co-$\mathcal{C}$ calls. □

We can see that the membership results n Table 5.6 follow easily from the above theorem. For example, if MinEX-Nec$^{\models}$ is in Π_2^P, we have that MinEX-AllNec$^{\models}$

can be decided by a linear number of Π_2^{p} tests and a linear number of Σ_2^{p} tests, which can be combined as a single Π_2^{p} and a single Σ_2^{p} test. Therefore MinEX-AllNec$^{\not\models}$ is in $\mathrm{D}_2^{\mathrm{p}}$ in this case. Other membership results follow similarly.

As for the hardness results, the following theorem proves all the D^{p}-hardness results in Table 5.6 in data complexity for the languages considered here that are not FO-rewritable. The proof is via a reduction from Sat-Unsat, which is based on the ideas, presented in the proofs of Theorem 3.19 and Theorem 5.13. In this construction, we add to H the facts associated with the possible truth assignments for the variables X of the formulas, and the additional facts allowing us to recognise if the formulas ϕ and ψ are unsatisfiable. In this case, the idea is to have all facts in H to be non-necessary, apart from the one associated with the unsatisfiability of ψ.

Theorem 5.29. MinEX-AllNec$^{\not\models}(\subseteq, \mathrm{BCQ}, \mathsf{GF}_\bot)$ *is* D^{p}-hard *in data complexity.*

Proof. We provide a reduction from the D^{p}-complete problem Sat-Unsat. Let $\Phi = (\varphi, \psi)$ be an instance of Sat-Unsat, where $\varphi = c_1^\varphi \wedge \cdots \wedge c_{m_\varphi}^\varphi$ is a 3CNF formula over variables $X^\varphi = \{x_1^\varphi, \ldots, x_{n_\varphi}^\varphi\}$ and $\psi = c_1^\psi \wedge \cdots \wedge c_{m_\psi}^\psi$ is a 3CNF formula over variables $X^\psi = \{x_1^\psi, \ldots, x_{n_\psi}^\psi\}$. This proof uses many of the constructions, introduced in the proofs of Theorems 5.13 and 5.21. We refer the reader to these proofs for more intuition. We construct an instance of MinEX-AllNec$^{\not\models}$ $(\subseteq, \mathrm{BCQ}, \mathsf{GF}_\bot)$ consisting of a database D_Φ, an OMQ (Q, Σ), a finite set of facts H_Φ, and a set $H'_\Phi \subseteq H_\Phi$, where only D_Φ, H_Φ, and H'_Φ depend on Φ, while (Q, Σ) is independent of the instance of Sat-Unsat.

First, we construct the database D_Φ. For each clause c_j^α, where $\alpha \in \{\varphi, \psi\}$ and $1 \leq j \leq m_\alpha$, there is a fact $succ\_cl^\alpha(j-1, j)$ in D_Φ, where $j-1$ and j are numeric constants, intuitively stating that the j^{th} clause is the successor of the $(j-1)^{\text{th}}$ clause. Also, for each clause c_j^α, there is a fact in D_Φ that encodes it. For example, for a clause $c_j^\alpha = (x_p^\alpha \vee x_q^\alpha \vee \neg x_r^\alpha)$, there is a fact $cl^\alpha(j, x_p^\alpha, p, x_q^\alpha, p, x_r^\alpha, n)$ in D_Φ, where p and n are constants representing whether a variable appears as a positive or negative literal, respectively. Furthermore, for each $\alpha \in \{\varphi, \psi\}$, we add the facts $satchain^\alpha(0)$ and $maxcl^\alpha(m_\alpha)$ to D_Φ.

Now we construct the set of hypotheses H_Φ. For each variable x_i^α, where $\alpha \in \{\varphi, \psi\}$ and $1 \leq i \leq n_\alpha$, we add to H_Φ the facts $val^\alpha(x_i^\alpha, 0)$ and $val^\alpha(x_i^\alpha, 1)$. Also, we add the facts $p^\varphi()$ and $p^\psi()$ to H_Φ. We take the candidate set of all the necessary facts to be $H'_\Phi = \{p^\psi()\}$.

The construction of the program Σ is based on the construction in the proof of Theorem 5.21. For each $\alpha \in \{\varphi, \psi\}$, add to Σ the following rules:

$$
\begin{aligned}
&val^{\alpha}(X,1) \wedge val^{\alpha}(X,0) \rightarrow \bot,\\
&cl^{\alpha}(J,X,p,\_,\_,\_,\_) \wedge val^{\alpha}(X,1) \rightarrow satcl^{\alpha}(J),\\
&cl^{\alpha}(J,X,n,\_,\_,\_,\_) \wedge val^{\alpha}(X,0) \rightarrow satcl^{\alpha}(J),\\
&cl^{\alpha}(J,\_,\_,Y,p,\_,\_) \wedge val^{\alpha}(Y,1) \rightarrow satcl^{\alpha}(J),\\
&cl^{\alpha}(J,\_,\_,Y,n,\_,\_) \wedge val^{\alpha}(Y,0) \rightarrow satcl^{\alpha}(J),\\
&cl^{\alpha}(J,\_,\_,\_,\_,Z,p) \wedge val^{\alpha}(Z,1) \rightarrow satcl^{\alpha}(J),\\
&cl^{\alpha}(J,\_,\_,\_,\_,Z,n) \wedge val^{\alpha}(Z,0) \rightarrow satcl^{\alpha}(J),\\
&satchain^{\alpha}(I) \wedge succ\_cl^{\alpha}(I,J) \wedge satcl^{\alpha}(J) \rightarrow satchain^{\alpha}(J),\\
&maxcl^{\alpha}(M) \wedge satchain^{\alpha}(M) \rightarrow sat^{\alpha}(),\\
&sat^{\alpha}() \rightarrow c^{\alpha}(), \quad p^{\alpha}() \rightarrow c^{\alpha}(),\\
&c^{\varphi}() \wedge c^{\psi}() \rightarrow query(), \quad p^{\varphi}() \wedge p^{\psi}() \rightarrow query().
\end{aligned}
$$

The query is $Q = query()$. Observe that Σ and Q do not depend on φ. Also, notice that Σ is guarded and full.

Before proving the correctness of the reduction, we point out that every fact $val^{\alpha}(x_i^{\alpha}, \star)$, where $\star \in \{0,1\}$, is not necessary. In fact, it is easy to see that $E = \{p^{\psi}(), p^{\varphi}()\}$ is a minimal explanation for $D_{\Phi} \not\models (Q,\Sigma)$ w.r.t. H_{Φ}.

We now show that φ is satisfiable and ψ is unsatisfiable iff H'_{Φ} is an inclusion-maximal set containing only necessary facts for $D_{\Phi} \not\models (Q,\Sigma)$ w.r.t. H_{Φ}, that is, $p^{\psi}()$ is the necessary fact in H_{ϕ}.

($\Rightarrow$) Suppose φ is satisfiable and ψ is unsatisfiable. We first show that every fact in $H_{\Phi} \setminus \{p^{\psi}()\}$ is not necessary. As shown above, all the facts $val^{\alpha}(x_i^{\alpha}, \star)$, where $\star \in \{0,1\}$, are not necessary. We now show that also $p^{\varphi}()$ is not necessary. Since φ is satisfiable, let $\sigma : X^{\varphi} \rightarrow \{T,F\}$ be a truth assignment satisfying it. We define $D_{\sigma} = \{val^{\varphi}(x_i^{\varphi},1) \mid \sigma(x_i^{\varphi}) = T\} \cup \{val^{\varphi}(x_i^{\varphi},0) \mid \sigma(x_i) = F\}$. Let $E = D_{\sigma} \cup \{p^{\psi}()\}$. It is easy to see that $(D_{\Phi} \cup E, \Sigma)$ is consistent and $D_{\Phi} \cup E \models (Q,\Sigma)$, as σ satisfies φ. Thus, there is a set $E' \subseteq E$ that is a minimal explanation and does not contain $p^{\varphi}()$. It remains to show that $p^{\psi}()$ is necessary. For a contradiction, suppose that $p^{\psi}()$ is not necessary and let E be a minimal explanation that does not contain it. The only way for $D_{\Phi} \cup E$ to entail (Q,Σ) is via $c^{\varphi}(), c^{\psi}() \rightarrow query()$, which requires $c^{\psi}()$ to be entailed, which in turn requires $sat^{\psi}()$ to be entailed (as $p^{\psi}() \notin E$), which is entailed only if ψ is satisfiable. This is a contradiction.

($\Leftarrow$) We prove the contrapositive. Suppose φ is unsatisfiable or ψ is satisfiable. Reasoning as above, it is can be easily verified that if φ is unsatisfiable, then $p^{\varphi}()$ is necessary. If ψ is satisfiable, then $p^{\psi}()$ is not necessary. □

Next, we prove the D^{P}-hardness results in *fp*-combined complexity in Table 5.6. We provide a reduction from Sat-Unsat. The set H includes a couple of facts allowing to recognise the unsatisfiability of φ and ψ. We check that the fact associated with the unsatisfiability of ψ is necessary.

Theorem 5.30. MinEX-AllNec$^{\models}(\subseteq, \mathrm{BCQ}, \mathcal{L}_{\emptyset})$ *is* D^{P}-hard *in fp-combined complexity.*

Proof. We reduce Sat-Unsat to MinEX-AllNec$^{\models}(\subseteq, \mathrm{BCQ}, \mathcal{L}_{\emptyset})$ in *fp*-combined complexity. Let $I = (\varphi, \psi)$ be an instance of Sat-Unsat, where $\varphi = c_1^{\varphi} \vee \cdots \vee c_{m_\varphi}^{\varphi}$ is a 3CNF formula over variables $X^{\varphi} = \{x_1^{\varphi}, \ldots, x_{n_\varphi}^{\varphi}\}$ and $\psi = c_1^{\psi} \vee \cdots \vee c_{m_\psi}^{\psi}$ is a 3CNF formula over variables $X^{\psi} = \{x_1^{\psi}, \ldots, x_{n_\psi}^{\psi}\}$. We build an instance of MinEX-AllNec$^{\models}$ consisting of a database D_I, an OMQ (Q_I, Σ), a finite set of facts H_I, and a set $H_I' \subseteq H_I$ as follows. Note that only D_I, Q_I, H_I, and H_I' depend on I.

We start with the database D_I, which is built as follows. There are facts in D_I that are used to impose the consistency of the truth assignments to the literals:

$$\begin{array}{lll} \mathit{simlit}(0,0) & \mathit{simlit}(1,1) & \mathit{simlit}(2,2) \\ \mathit{opplit}(0,1) & \mathit{opplit}(1,0) & \mathit{opplit}(2,2), \end{array}$$

where 0 and 1 are constants representing Boolean values *false* and *true*, respectively, and 2 is an additional constants used as a "jolly" (intuitively, it will be used to bypass, in some circumstances, the check of the satisfaction of φ and ψ). The predicate $\mathit{simlit}(\cdot,\cdot)$ will be used to impose that when a variable appears twice as a positive or a negative literal in two different places in the formulas, then the two literals must have the same truth value. On the other hand, the predicate $\mathit{opplit}(\cdot,\cdot)$ will be used to impose that when a variable appears as a positive literal in one place in the formula and as a negative literal in another place of the formula, then the two literals must have different truth values. There are the facts in D_I that are used to select possible ways of satisfying the clauses in the formula $\alpha \in \{\varphi, \psi\}$:

$$\begin{array}{llll} \mathit{clsat}^{\alpha}(1,1,1), & \mathit{clsat}^{\alpha}(1,1,0), & \mathit{clsat}^{\alpha}(1,0,1), & \mathit{clsat}^{\alpha}(1,0,0), \\ \mathit{clsat}^{\alpha}(0,1,1), & \mathit{clsat}^{\alpha}(0,1,0), & \mathit{clsat}^{\alpha}(0,0,1), & \end{array}$$

where 0 and 1 are constants with the same meaning as above. The facts $clsat^{\alpha}(\cdot,\cdot,\cdot)$ capture what truth assignments to the *literals* (and not to the variables) satisfy a clause of $\alpha \in \{\varphi, \psi\}$.

The set of hypothesis is $H_I = \{clsat^{\phi}(2,2,2), clsat^{\psi}(2,2,2)\}$. Intuitively, facts $clsat^{\alpha}(2,2,2)$ allow to bypass, in some circumstances, the check of the satisfaction of $\alpha \in \{\varphi, \psi\}$. We take the candidate set of all the necessary facts to be $H'_I = \{clsat^{\psi}(2,2,2)\}$. We take the program Σ to be empty.

The query Q_I is defined as follows. The aim of the query is to check the satisfiability of φ and ψ. To characterise the query, we describe its various components. Below we use the following notation: $\ell^{\alpha}_{j,k}$ is the k^{th} literal in the j^{th} clause, c_j, of formula $\alpha \in \{\varphi, \psi\}$, and $v^{\alpha}_{j,k}$ is the variable of $\ell^{\alpha}_{j,k}$. Furthermore, below $T^{\alpha}_{j,k}$ is the variable used in the query to represent the Boolean value of the literal $\ell^{\alpha}_{j,k}$.

There is a piece forcing that the ground values 0 and 1 assigned to the various variables $T^{\alpha}_{j,k}$ simulating the assignments to the literals are consistent. In the notation below, $\ell^{\alpha}_{j,k} \sim \ell^{\alpha}_{j',k'}$ means that $v^{\alpha}_{j,k} = v^{\alpha}_{j',k'}$ and the literals $\ell^{\alpha}_{j,k}$ and $\ell^{\alpha}_{j',k'}$ are both positive or negative, while $\ell^{\alpha}_{j,k} \nsim \ell^{\alpha}_{j',k'}$ means that $v^{\alpha}_{j,k} = v^{\alpha}_{j',k'}$ and one literal is positive and the other is negative. For each $\alpha \in \{\varphi, \psi\}$, we take

$$consist^{\alpha} \equiv \bigwedge_{\ell^{\alpha}_{j,k} \sim \ell^{\alpha}_{j',k'}} simlit(T^{\alpha}_{j,k}, T^{\alpha}_{j',k'}) \wedge \bigwedge_{\ell^{\alpha}_{j,k} \nsim \ell^{\alpha}_{j',k'}} opplit(T^{\alpha}_{j,k}, T^{\alpha}_{j',k'}).$$

Another piece of the query checks the satisfiability of $\alpha \in \{\varphi, \psi\}$:

$$satisfied^{\alpha} \equiv \bigwedge_{j=1}^{m} clsat^{\alpha}(T^{\alpha}_{j,1}, T^{\alpha}_{j,2}, T^{\alpha}_{j,3}).$$

To conclude, the query is

$$Q_I = consist^{\varphi}, satisfied^{\varphi}, consist^{\psi}, satisfied^{\psi},$$

where all the variables are existentially quantified. Observe that $D_I \cup H \models (Q_I, \Sigma)$, and thus the set of MinEXs is non-empty.

We now show that $I = (\varphi, \psi)$ is a 'yes'-instance of Sat-Unsat iff H'_I is exactly the set of all the necessary facts for $D_I \not\models (Q_I, \Sigma)$ w.r.t. H_I.

($\Rightarrow$) Let $I = (\varphi, \psi)$ be a 'yes'-instance of Sat-Unsat, which means that φ is satisfiable and ψ is not. Since φ is satisfiable, there is an assignment of the constants 0 and 1 to all the variables $T^{\varphi}_{j,k}$ such that the parts $consist^{\varphi}$, $satisfied^{\varphi}$ of the query are satisfied. Therefore, a set $\{clsat^{\psi}(2,2,2)\}$ is a (non-necessarily minimal) explanation. This implies that there exists an explanation excluding $clsat^{\varphi}(2,2,2)$ is not a necessary

fact, and observe that H'_I does not contain $clsat^{\varphi}(2,2,2)$. On the other hand, since ψ is unsatisfiable, there is no assignment of the constants 0 and 1 to all the variables $T^{\psi}_{j,k}$ such that the parts $consist^{\psi}$, $satisfied^{\psi}$ of the query are satisfied. Therefore, the only way to satisfy these parts of the query is by adding the fact $clsat^{\psi}(2,2,2)$, which hence belongs to all minimal explanations, implying that $clsat^{\psi}(2,2,2)$ is necessary. Observe that $clsat^{\psi}(2,2,2)$ belongs to H'_I, which conclude this direction of the proof.

($\Leftarrow$) Assume that $I = (\varphi, \psi)$ is a 'no'-instance of Sat-Unsat, which implies that φ is unsatisfiable or ψ is satisfiable. An argument similar to the one in the direction '($\Rightarrow$)' above proves that if φ is unsatisfiable, then $clsat^{\varphi}(2,2,2)$ is necessary, contradicting hence that H'_I is the set of all and only the necessary facts; and if ψ is satisfiable, then $clsat^{\psi}(2,2,2)$ is not necessary, contradicting hence that H'_I is the set of all and only the necessary facts. □

The following theorem proves the $\mathrm{D}^{\mathrm{p}}_2$-hardness of the problem MinEx-AllNec$^{\not\models}$ in *ba*-combined complexity for all the languages we consider.

Theorem 5.31. MinEx-AllNec$^{\not\models}(\subseteq, \mathrm{BCQ}, \mathcal{L})$ *is* $\mathrm{D}^{\mathrm{p}}_2$-hard *in ba-combined complexity for any language* $\mathcal{L}$ *of existential rules considered in this chapter.*

Proof. We reduce to MinEx-AllNec$^{\not\models}$ the $\mathrm{D}^{\mathrm{p}}_2$-complete problem $\mathrm{QBF}^{CNF}_{2,\forall,\neg} \wedge \mathrm{QBF}^{CNF}_{2,\exists}$, introduced in the proof of Theorem 5.22. Let $I = (\Phi, \Psi)$ be an instance of $\mathrm{QBF}^{CNF}_{2,\forall,\neg} \wedge \mathrm{QBF}^{CNF}_{2,\exists}$. Suppose that $\Phi = \exists X \forall Y \neg \phi(X,Y)$ and $\Psi = \forall X \exists Y \psi(X,Y)$, where $\phi(X,Y)$ and $\psi(X,Y)$ are 3CNF formulas. Recall the the problem asks whether Φ and Ψ are both valid. We build a MinEx-AllNec$^{\not\models}$ instance consisting of a database D_I and an OMQ (Q_I, Σ_I), a finite set of facts H_I, and a set $H'_I \subseteq H_I$ as follows.

By an argument similar to the one in the proof of Theorem 3.9 in [113], we can assume w.l.o.g. that variables X and Y of formulas ϕ and ψ are the same, and that the number of clauses in $\phi(X,Y)$ and $\psi(X,Y)$ is the same. Let n be the number of variables in X and let m be the number of clauses of $\phi(X,Y)$ and $\psi(X,Y)$.

We start by defining H_I. For each $\alpha \in \{\phi, \psi\}$ and for each variable $x_i \in X$, H_I contains the following facts:

$$val^{\alpha}(x^{\alpha}_i, 0), \qquad val^{\alpha}(x^{\alpha}_i, 1), \qquad val^{\alpha}(x^{\alpha}_i, 2),$$

where 0 and 1 are constants representing the Boolean values *false* and *true*, respectively, while 2 is an additional constant that will allow to bypass, in some circumstances, the satisfiability tests for $\phi(X,Y)$ and $\psi(X,Y)$. We take the candidate set of all necessary facts to be

$$H'_I = \{val^{\psi}(x^{\psi}_i, 2) \mid x_i \in X\}.$$

We construct the database D_I as follows. We add the facts $simlit(0,0)$, $simlit(1,1)$, $opplit(0,1)$, and $opplit(1,0)$ to D_I, which are used to impose the consistency of the truth assignments to the literals. Also, we add the facts to D_I that encode the satisfying assignments to literals for the clause:

$$\begin{array}{llll} clsat(1,1,1), & clsat(1,1,0), & clsat(1,0,1), & clsat(1,0,0), \\ clsat(0,1,1), & clsat(0,1,0), & clsat(0,0,1). & \end{array}$$

We define the program Σ_I as follows. For each $\alpha \in \{\phi, \psi\}$, we add the following NCs to Σ_I:

$$\begin{aligned} val^\alpha(X,0), val^\alpha(X,1) &\rightarrow \bot, \\ val^\alpha(X,0), val^\alpha(X,2) &\rightarrow \bot, \\ val^\alpha(X,1), val^\alpha(X,2) &\rightarrow \bot. \end{aligned}$$

There are two additional NCs that are made of several pieces, which we now define. The following pieces are defined for each $\alpha \in \{\phi, \psi\}$. The first piece "reads" onto the variables T_i^α the assignment on the variables in X of α encoded in the explanation:

$$assignX^\alpha \equiv \bigwedge_{i=1}^{n} val^\alpha(x_i^\alpha, T_i^\alpha).$$

Below, we use the following notation: $\ell_{j,k}^\alpha$ is the k^{th} literal in the j^{th} clause of formula α, and $v_{j,k}^\alpha$ is the variable of $\ell_{j,k}^\alpha$. The second piece "copies" the assignment of each variable x_i onto a positive occurrence of x_i in $\alpha \in \{\phi, \psi\}$:

$$copy^\alpha \equiv \bigwedge_{i=1}^{n} simlit(T_i^\alpha, T_{j,k}^\alpha),$$

where in each predicate $simlit(T_i^\alpha, T_{j,k}^\alpha)$, $T_{j,k}^\alpha$ is a variable for the Boolean value of a literal $\ell_{j,k}^\alpha = x_i$ in $\alpha \in \{\phi, \psi\}$. Observe that, in order for $copy^\alpha$ to work properly, each variable x_i must appear as a positive literal in α at least once. This can be assumed w.l.o.g., because if x_i always appears as a negative literal in all the clauses of α, then we can replace all the occurrences of the negative literal $\neg x_i$ with the positive literal x_i without altering the satisfiability properties of α.

The third piece forces the ground values 0 and 1 assigned to the variables $T_{j,k}^\alpha$, simulating the assignments, to the literals to be consistent. Below, $\ell_{j,k}^\alpha \sim \ell_{j',k'}^\alpha$ means that $v_{j,k}^\alpha = v_{j',k'}^\alpha$, and the literals $\ell_{j,k}^\alpha$ and $\ell_{j',k'}^\alpha$ are both positive or negative, while

$\ell^\alpha_{j,k} \nsim \ell^\alpha_{j',k'}$ means that $v^\alpha_{j,k} = v^\alpha_{j',k'}$, and one literal is positive and the other is negative. For $\alpha \in \{\phi, \psi\}$, we take

$$consist^\alpha \equiv \bigwedge_{\ell^\alpha_{j,k} \sim \ell^\alpha_{j',k'}} simlit(T^\alpha_{j,k}, T^\alpha_{j',k'}) \wedge \bigwedge_{\ell^\alpha_{j,k} \nsim \ell^\alpha_{j',k'}} miopplit(T^\alpha_{j,k}, T^\alpha_{j',k'}).$$

The following piece checks if $\alpha \in \{\phi, \psi\}$ is satisfied:

$$satisfied^\alpha \equiv \bigwedge_{j=1}^{m_\alpha} clsat(T^\alpha_{j,1}, T^\alpha_{j,2}, T^\alpha_{j,3}).$$

Therefore, for each $\alpha \in \{\phi, \psi\}$, we add to the program Σ the NC:

$$assignX^\alpha, copy^\alpha, consist^\alpha, satisfied^\alpha \rightarrow \bot.$$

We take the query to be

$$Q_I = \bigwedge_{i=1}^{n} val^\phi(x^\phi_i, T_i) \wedge \bigwedge_{i=1}^{n} val^\psi(x^\psi_i, Z_i),$$

where each variable is existentially quantified. The aim of this definition of the query is to ensure that a fact $val^\phi(x^\phi_i, \cdot)$ and a fact $val^\psi(x^\psi_i, \cdot)$ have been taken from H for every $x_i \in X$.

Notice that Σ_I has no TGDs, and thus it is linear full, acyclic full, and sticky full. Furthermore, all predicates have bounded arity.

Below we show that Φ and Ψ are valid iff every fact in H'_I is necessary and every fact in $H_I \setminus H'_I$ is not necessary. In the rest of the proof, we will use the following notation. For each $\alpha \in \{\psi, \phi\}$, we define $D^\alpha_2 = \{val^\alpha(x^\alpha_i, 2) \mid x_i \in X\}$. For each $\alpha \in \{\phi, \psi\}$, given a truth assignment $\sigma : X \rightarrow \{true, false\}$, we define $D^\alpha_\sigma = \{val^\alpha(x^\alpha_i, 0) \mid \sigma(x_i) = false\} \cup \{val^\alpha(x^\alpha_i, 1) \mid \sigma(x_i) = true\}$.

We start by pointing out that, for each $\alpha \in \{\phi, \psi\}$ and for every $x_i \in X$, the facts $val^\alpha(x^\alpha_i, 0)$ and $val^\alpha(x^\alpha_i, 1)$ are not necessary (regardless of the initial instance I). In fact, it can be easily verified that $D^\phi_2 \cup D^\psi_2$ is a minimal explanation for $D_I \not\models (Q_I, \Sigma_I)$ w.r.t. H_I, as $(D_I \cup D^\phi_2 \cup D^\psi_2, \Sigma_I)$ is consistent and $D_I \cup D^\phi_2 \cup D^\psi_2$ entails (Q_I, Σ_I).

Thus, we need to show that Φ and Ψ are valid iff every fact in D^ψ_2 is necessary and every fact in D^ϕ_2 is not necessary.

($\Rightarrow$) Suppose Φ and Ψ are valid. Since Φ is valid, there exists a truth assignment $\sigma : X \rightarrow \{true, false\}$ such that $\forall Y \neg\phi(X/\sigma, Y)$ is valid, i.e. $\phi(X/\sigma, Y)$ is not satisfiable. For this reason, there is no homomorphism from the body of the NC '$assignX^\phi, copy^\phi, consist^\phi, satisfied^\phi \rightarrow \bot$' to $D_I \cup D^\phi_\sigma$, which means that $(D_I \cup D^\phi_\sigma, \Sigma_I)$ is consistent.

Let $E = D^{\phi}_{\sigma} \cup D^{\psi}_{2}$. Observe that E is consistent with Σ_I and entails (Q_I, Σ_I), hence E is an explanation. Moreover, no fact can be removed from E without losing the entailment of the OMQ, hence E is a minimal explanation. As E is a minimal explanation for $D_I \not\models (Q_I, \Sigma_I)$ w.r.t. H_I, every fact in D^{ϕ}_{2} is not necessary.

We now show that every fact D^{ψ}_{2} is necessary. For a contradiction, suppose that there is a minimal explanation E that does not contain a fact $val^{\psi}(x^{\psi}_{k}, 2)$. Since $D_I \cup E \models (Q_I, \Sigma_I)$, E contains a fact $val^{\psi}(x^{\psi}_{i}, \cdot)$ for every $x_i \in X$. Since $val^{\psi}(x^{\psi}_{k}, 2) \notin E$, either $val^{\psi}(x^{\psi}_{k}, 0)$ or $val^{\psi}(x^{\psi}_{k}, 1)$ is in E. Since $(D_I \cup E, \Sigma_I)$ is consistent, because we assume E to be an explanation, and contains $val^{\psi}(x^{\psi}_{k}, 0)$ or $val^{\psi}(x^{\psi}_{k}, 1)$, E does not contain any fact $val^{\psi}(x^{\psi}_{i}, 2)$. Thus, there is a truth assignment $\sigma : X \to \{true, false\}$ such that $D^{\psi}_{\sigma} \subseteq E$. Since Ψ is valid, $\exists Y \psi(X/\sigma, Y)$ is valid, i.e. $\psi(X/\sigma, Y)$ is satisfiable. Therefore, there exists a homomorphism from the body of the NC '$assignX^{\psi}, copy^{\psi}, consist^{\psi}, satisfied^{\psi} \to \bot$' to $D_I \cup E$. This implies that $(D_I \cup E, \Sigma_I)$ is not consistent, which is a contradiction.

($\Leftarrow$) Suppose that Φ is not valid or that Ψ is not valid. We show that if Φ is not valid, then every fact in D^{ϕ}_{2} is necessary; if Ψ is not valid, then every fact in D^{ψ}_{2} is not necessary.

Suppose that Φ is not valid. For a contradiction, suppose that there is a fact in D^{ϕ}_{2} that is not necessary. This implies that there is a minimal explanation E that does not contain a fact $val^{\phi}(x^{\phi}_{k}, 2)$. Reasoning as above, since $(D_I \cup E, \Sigma_I)$ is consistent, because E is assumed to be an explanation, and $D_I \cup E \models (Q_I, \Sigma_I)$, there must be a truth assignment $\sigma : X \to \{true, false\}$ such that $D^{\phi}_{\sigma} \subseteq E$. As Φ is not valid, $\exists Y \phi(X/\sigma, Y)$ is valid, i.e. $\phi(X/\sigma, Y)$ is satisfiable. This means that there exists a homomorphism from the body of the NC '$assignX^{\phi}, copy^{\phi}, consist^{\phi}, satisfied^{\phi} \to \bot$' to $D_I \cup E$. This implies that $(D_I \cup E, \Sigma_I)$ is inconsistent, which is a contradiction. Thus, every fact in D^{ϕ}_{2} is necessary.

Suppose that Ψ is not valid. Then, there exists a truth assignment $\sigma : X \to \{true, false\}$ such that $\forall Y \neg \psi(X/\sigma, Y)$ is valid, i.e. $\psi(X/\sigma, Y)$ is not satisfiable. Let $E = D^{\psi}_{\sigma} \cup D^{\phi}_{2}$. By following a reasoning similar to the one above, E is a minimal explanation for $D_I \not\models (Q_I, \Sigma_I)$ w.r.t. H_I. Since E does not contain any fact of D^{ψ}_{2}, every fact in D^{ψ}_{2} is not necessary. □

The remaining hardness results in *ba*-combined complexity as well as all the hardness results in combined complexity are established via the following theorem, which shows that MinEX-Exists$^{\not\models}$ can be reduced to the complement of MinEX-AllNec$^{\not\models}$.

Theorem 5.32. *For any language $\mathcal{L}$ of existential rules, we have that the problem* MinEX-Exists$^{\not\models}(\subseteq, \text{BCQ}, \mathcal{L})$ *is reducible in polynomial time to the complement of* MinEX-AllNec$^{\not\models}(\subseteq, \text{BCQ}, \mathcal{L})$.

Proof. Consider an instance of MinEX-Exists$^{\not\models}$ consisting of a database D, an OMQ (Q, Σ), and a finite set of facts H. We derive an instance of the complement of MinEX-AllNec$^{\not\models}$ consisting of a database $D^* = D$, an OMQ $(Q^*, \Sigma^*) = (Q, \Sigma)$, a finite set of facts $H^* = H \cup \{p()\}$, and $H' = H^*$, where p is a fresh 0-ary predicate.

($\Rightarrow$) If there exists a MinEX E for $D \not\models (Q, \Sigma)$ w.r.t. H, then E is also a MinEX for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^*. Notice that $p() \notin E$ and thus $p()$ is not necessary, which means that H' is not a maximal set containing only necessary facts.

($\Leftarrow$) If there does not exist a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H, then there does not exist a minimal explanation for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^* (the presence of $p()$ in H^* is useless), which implies that all facts are necessary for $D^* \not\models (Q^*, \Sigma^*)$ w.r.t. H^*, and thus H' is a maximal set containing only necessary facts. □

5.6 Overview of the Complexity Results

In this section, we discuss the complexity results obtained in this chapter, outlining the most interesting ones, pointing out differences and similarities between the problems in this chapter and the problems studied in other chapters. In this chapter, we have given a precise picture of the complexity for all the negative explanation problems, introduced in Section 5.1. An overview of our complexity results is given in Tables 5.1–5.6. They range from membership in P to 2Exp-completeness; all entries without '$\leq$' are completeness results.

First, one may wonder whether there are reductions from some already considered problems for explanations for *positive* query answers, studied in Chapter 3, into some problem for negative query answers, introduced in Section 5.1. This would establish a connection between these problems and transfer the hardness results. As it turns out it is possible to reduce Is-MinEX to Is-MinEX$^{\not\models}$ and MinEX-Rel to MinEX-Rel$^{\not\models}$ as was shown in Theorem 5.9 and Theorem 5.19, respectively. Furthermore, we can show that the same upper bounds hold for these corresponding problems. Therefore, we have that the complexity results match for these corresponding problems. For the remaining problems, however, there is no straightforward reduction between the two settings, and hence, they required novel proofs. To see the difference between these two scenarios consider, for example, the database $D = \{R(a)\}$, the OMQ (Q, Σ) with $Q = \exists X, Y\ R(X) \wedge S(Y)$ and $\Sigma = \emptyset$, and the set of abducible facts

$H = \{R(b), S(b)\}$. Then $E = \{R(b), S(b)\}$ is a MinEX for $D \cup H \models (Q, \Sigma)$, but it is not a MinEX for $D \not\models (Q, \Sigma)$ w.r.t. H because it is not minimal.

We discuss some of the complexity results for the other considered problems. First, we can see that MinEX-Nec$^{\not\models}$ and MinEX-Rel$^{\not\models}$ are always in complementary complexity classes, except in *fp*-combined complexity, where MinEX-Nec$^{\not\models}$ is coNP-complete, while MinEX-Rel$^{\not\models}$ is Σ_2^{p}-complete. Intuitively, the similarities and differences could be explained that these tasks can be seen as complementary, but MinEX-Rel$^{\not\models}$ needs an extra minimality check. To see this, note that to decide whether a fact ψ is *not* necessary, we can guess a subset E of H, not containing ψ and check whether E entails the OMQ, and verifying minimality of E can be avoided. On the other hand, to decide relevance of a fact, we can guess a subset E of H for which we need to check both that it entails the OMQ and it is minimal, and the minimality check cannot be avoided. For the cases when checking minimality does not increase the complexity of MinEX-Rel$^{\not\models}$, the two problems are in complementary complexity classes. In *fp*-combined complexity, such a minimality check is in coNP and yields a membership of MinEX-Rel$^{\not\models}$ in $\mathrm{NP}^{\mathrm{coNP}} = \Sigma_2^{\mathrm{p}}$. For similar reasons, we have that, for MinEX-AllRel$^{\not\models}$ and MinEX-AllNec$^{\not\models}$, the results coincide in all the complexity measures apart from *fp*-combined complexity.

Another interesting observation is that the complexity of the problems studied in this work does not only depend on the complexity of OMQA. For the linear, full, and sticky languages, as well as for their sublanguages, OMQA is NP-complete both in *fp*-combined and in *ba*-combined complexity. One may expect that, for a fixed problem, the complexity does not change in such cases, as the complexity of OMQA remains the same across them. However, this is not the case for MinEX-Exists$^{\not\models}$, MinEX-Nec$^{\not\models}$, and MinEX-AllNec$^{\not\models}$, where the complexity goes from NP-, coNP-, and D^{p}-completeness in *fp*-combined complexity to Σ_2^{p}-, Π_2^{p}-, and $\mathrm{D}_2^{\mathrm{p}}$-completeness in *ba*-combined complexity, respectively. The reason is consistency checking in *fp*-combined complexity can be reduced to OMQA in data complexity since, for such a check, the query is fixed too, which is $(\bot, \Sigma)$. In contrast, in *ba*-combined complexity, the program is not fixed anymore, and thus the OMQ $(\bot, \Sigma)$ is not fixed either. Therefore, checking inconsistency of the knowledge base increases the overall complexity by one level.

Chapter 6

Preferred Explanations under Existential Rules

In the thesis thus far, we have taken explanations for a query answer to be inclusion-minimal subsets that support an entailment of such a query. In this chapter, we take a step further and study preferred explanations, which are subsets of the database under different minimality criterion that support a query entailment. In existing literature on knowledge representation, preferences have been commonly expressed using *cardinality* and *weight* orders [71, 21]. We study cardinality and weight-minimal explanations for a query answer, which are subsets of the database of the smallest size and the smallest weight, respectively, that entail such a query. In the weight case, we assume that each fact is assigned a weight representing a *penalisation degree* for that fact. Observe that cardinality-minimal explanations are a special case of weight-minimal explanations when each fact is assigned the same weight.

For the two aforementioned preference orders, we study five already introduced 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 sets of *forbidden 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 mentioned, 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 results for these problems are in most cases different from the ones

in the inclusion-minimal case, presented in Chapter 3. Therefore the analysis in this chapter requires many new techniques and reductions.

This chapter is based on a paper published at the 35th AAAI Conference on Artificial Intelligence (AAAI-21) [53].

6.1 Preferred Explanations

In this section, we extend the notion of MinEX, introduced in Chapter 3, to incorporate different minimality criterion. We also motivate the choice of the two minimality criteria studied in this chapter with an example. Given a preorder $\preccurlyeq$ on D, and subsets E, E' of a database D, we write $E \prec E'$ if $E \preccurlyeq E'$ and $E' \not\preccurlyeq E$. The following definition extends inclusion-minimal explanations to arbitrary preorders $\preccurlyeq$. As in Chapter 3, we assume that the program consists only of TGDs, that is, it does not contain any negative constraints. This is because we are interested in explaining only query answers that follow from consistent knowledge bases.

Definition 6.1 ($\preccurlyeq$-MinEX)**.** For a database D, a preorder $\preccurlyeq$ on $\mathcal{P}(D)$, and an OMQ (Q, Σ), where Σ is a set of existential rules and Q is a query, such that $D \models (Q, \Sigma)$, an *explanation for* $D \models (Q, \Sigma)$ is a subset $E \subseteq D$ of facts entailing the OMQ (Q, Σ), i.e. $E \models (Q, \Sigma)$.

A *$\preccurlyeq$-minimal explanation E, or* $\preccurlyeq$-MinEX*, for* $D \models (Q, \Sigma)$ is an explanation for $D \models (Q, \Sigma)$ that is *$\preccurlyeq$-minimal*, i.e., there is no explanation E' for $D \models (Q, \Sigma)$ with $E' \prec E$.

We concentrate on two fundamental preference orders, common in the literature [74, 21], namely, orders induced by cardinality and a weight function. 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, we write

$$D_1 \leq D_2 \text{ iff } |D_1| \leq |D_2|.$$

Such preferences are particularly relevant for domains, where explanations correspond to optimal answers that are composed of equally costly components.

For preferences induced by *weights* ($\leq_w$), we assume that the facts in the database D are assigned (rational) weights by a function $w : D \to \mathbb{Q}$, which defines an order over the subsets of the database D. Given two subsets $D_1, D_2 \subseteq D$, we write

$$D_1 \leq_w D_2 \text{ iff } \sum_{\alpha \in D_1} w(\alpha) \leq \sum_{\alpha \in D_2} w(\alpha).$$

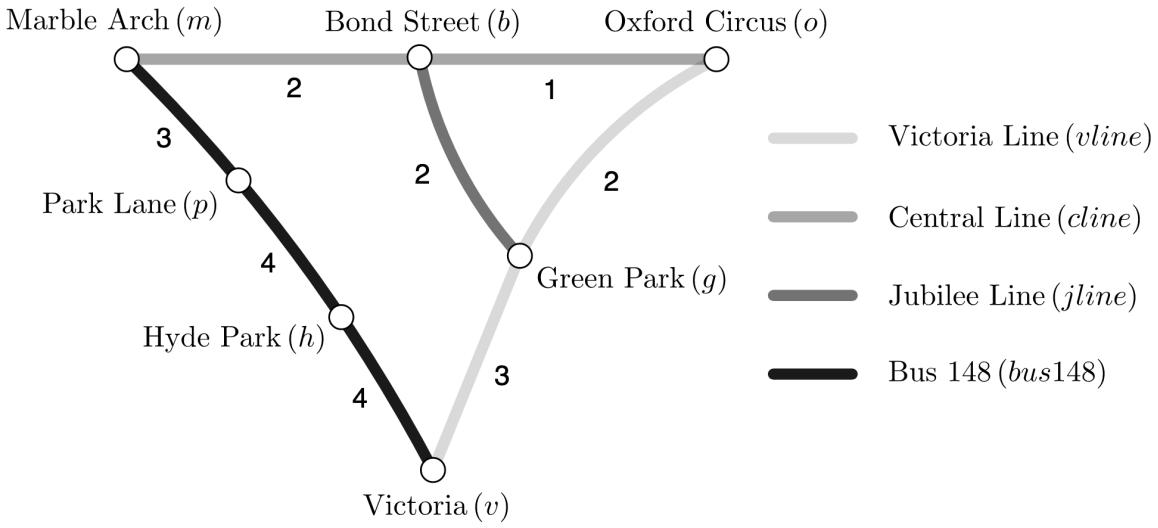

Figure 6.1: Excerpt from the London public transport map.

Intuitively, the facts with higher weights are less preferred in the explanations, and thus weights can be seen as *penalisation*. We illustrate our notions on an example, which we have already seen in Chapter 1.

$D = D_{st} \cup \{start(v), destination(m)\}$.

Then, we capture the knowledge about travelling along this network in an ontology Σ, also called a *program*. The program Σ consists of four rules. The first rule captures that the *start* stop is already reached, and the second one that if the destination can be reached then we can arrive at it:

$$start(X) \rightarrow reach(X),$$
$$destination(X) \wedge reach(X) \rightarrow arrive(X).$$

Example 6.2. Let us consider the road map, given in Figure 6.1, and the database that encodes the stops and the links between them:

$$\begin{aligned} D_{st} = \{&stop(v), stop(h), stop(p), stop(m), stop(g), stop(b), stop(o),\\ &link(bus148, v, h), link(bus148, h, p), link(bus148, p, m), link(vline, v, g),\\ &link(vline, g, o), link(jline, g, b), link(cline, o, b), link(cline, b, m)\}. \end{aligned}$$

Suppose that we are seeking the routes from the Victoria stop (v), which we capture with the $start(v)$ fact. So the database is $D = D_{st} \cup \{start(v), destination(m)\}$.

The duration between stops and the time required for changing between lines can be represented with weights. We set $w(stop(i)) = 3$ for all stops i, i.e., three minutes for changing a line in any stop. Each *link* fact is assigned the weight capturing the duration of the travel along that link, as indicated in Figure 6.1, e.g.,

$w(\mathit{link}(\mathit{jline}, g, b)) = 2$. Also, as $\mathit{start}(v)$ captures that we start at the stop v, we assign the weight 0 to this fact.

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

$$\begin{aligned}
&\mathit{start}(X) \to \mathit{reach}(X),\\
&\mathit{destination}(X) \wedge \mathit{reach}(X) \to \mathit{arrive}(X),\\
&\mathit{link}(X, Y_1, Y_2) \to \mathit{link}(X, Y_2, Y_1),\\
&\mathit{link}(X, Y_1, Y_2) \wedge \mathit{link}(X, Y_2, Y_3) \to \mathit{link}(X, Y_1, Y_3),\\
&\mathit{reach}(Y_1) \wedge \mathit{link}(X, Y_1, Y_2) \wedge \mathit{stop}(Y_2) \to \mathit{reach}(Y_2).
\end{aligned}$$

Suppose we aim to travel to Marble Arch, so the query is $Q = \mathit{arrive}(m)$. The fastest routes to Marble Arch correspond to $\leq_w$-MinEXs for $D \models (Q, \Sigma)$, which are

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

■

6.2 Explanation Problems

In this section, we introduce a number of problems associated with preferred explanations, adapted from Chapter 3. We define these problems in a rather general way, which can be parametrized with any preference order $\preccurlyeq$.

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, Σ is an $\mathcal{L}$-program, and $D \models (Q, \Sigma)$.
Question: Is E a $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$?

Example 6.3. The sets E_1 and E_2 from Example 6.2 are $\leq_w$-minimal explanations for $D \models (Q, \Sigma)$, which correspond to the fastest route from Victoria to Marble Arch station. Consider now the set

$$\begin{aligned}
E_3 = \{&\mathit{start}(v), \mathit{stop}(g), \mathit{stop}(b), \mathit{stop}(m),\\
&\mathit{link}(\mathit{vline}, v, g), \mathit{link}(\mathit{jline}, g, b), \mathit{link}(\mathit{cline}, b, m)\}.
\end{aligned}$$

Although $E_3 \models (Q, \Sigma)$ and it is inclusion-minimal, it is not a $\leq_w$-MinEX since it is not weight-minimal. Note that $w(E_3) = 16 > w(E_1) = 14$. ∎

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

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

The next problem asks whether a particular fact is contained in some preferred explanation. We say that a fact ψ is $\preccurlyeq$-*relevant* for $D \models (Q, \Sigma)$ if ψ appears in some $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$.

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

We illustrate both All-MinEX and MinEX-Rel on our running example.

Example 6.4. 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 combinations 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, Σ is an $\mathcal{L}$-program, and $D \models (Q, \Sigma)$.
Question: Is there a $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$ not containing any of the sets in $\mathcal{F}$?

Example 6.5. Suppose that the link between the stops h and p cannot be used, i.e., $\mathcal{F} = \{\{link(bus148, h, p)\}\}$ is the set of forbidden subsets of the database. Then E_2 gives a quickest route that does not contain this forbidden link, and thus E_2 is a $\leq_w$-MinEX for $D \models (Q, \Sigma)$.

On the other hand, if we also forbid to use the link between g and o, that is, $\mathcal{F} = \{\{link(bus148, h, p)\}, \{link(vline, g, o)\}\}$, then every $\leq_w$-MinEX contains some forbidden set.

Table 6.1: Complexity of Is-MinEX and All-MinEX for $(\preccurlyeq, \mathcal{Q}, \mathcal{L})$ with $\preccurlyeq \in \{\leq, \leq_w\}$ and $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

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

The next problem asks whether a particular fact is contained in every preferred explanation. We say that a fact ψ is $\preccurlyeq$-*necessary* for $D \models (Q, \Sigma)$ if ψ appears in every $\preccurlyeq$-MinEX for $D \models (Q, \Sigma)$.

Problem: MinEX-Nec$(\preccurlyeq, \text{UCQ}, \mathcal{L})$
Input: A database D, a fact $\psi \in D$, and an OMQ (Q, Σ), where Q is a UCQ, Σ is an $\mathcal{L}$-program, and $D \models (Q, \Sigma)$.
Question: Is ψ $\preccurlyeq$-necessary for $D \models (Q, \Sigma)$?

Example 6.6. The facts $start(v)$ and $stop(m)$ are the only two $\leq_w$-necessary facts for any for $D \models (Q, \Sigma)$. Any other fact is not $\leq_w$-necessary. This can be seen by taking the intersection of the only two $\leq_w$-minimal explanations, which is $E_1 \cap E_2 = \{start(v), stop(m)\}$. ∎

6.3 Explanations with Minimal Cardinality

After introducing all five problems studied in this chapter, and illustrating them on our running example, we now move to the technical results for each of the discussed preference orders. In this section, we analyse the computational problems associated with computing cardinality-minimal explanations.

6.3.1 Smallest Explanation

We consider the problem of recognising a cardinality-minimal explanation. We start with showing membership results in Table 6.1 and then proceed to give the matching hardness results. First, we present a general algorithm that covers most of the membership results.

Theorem 6.7. *For any class $\mathcal{L}$ of TGDs, if* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* $\text{Is-MinEX}(\leq, \text{UCQ}, \mathcal{L})$ *can be decided by a test in $\mathcal{C}$ and a test in co-*$(\text{NP}^{\mathcal{C}})$.

Proof. Let D be a database, (Q, Σ) be an OMQ, and $E \subseteq D$ be a candidate $\leq$-MinEX. Note that we can check whether $E \models (Q, \Sigma)$ with a single $\mathcal{C}$ check. We can check whether E is *not* $\leq$-minimal by guessing a subset $E' \subseteq D$ with $|E'| < |E|$ in NP, and then checking whether $E' \models (Q, \Sigma)$ in $\mathcal{C}$. Therefore, checking whether E is $\leq$-minimal is in co-$(\text{NP}^{\mathcal{C}})$. □

This already implies all the membership results for Is-MinEX in Table 6.1, apart from the tractability results in data complexity and the D^{P} membership results. For example, if OMQA is in P, then Is-MinEX can be decided with a single P and a single coNP test. These two can be combined as a single coNP, which solves Is-MinEX. The remaining cases are dominated by the complexity of query answering. To see that, note that, if OMQA is in $\mathcal{C} \in \{\text{Exp}, \text{2Exp}, \text{PSpace}\}$, then co-$(\text{NP}^{\mathcal{C}}) = \mathcal{C}$ and so, by Theorem 6.7, Is-MinEX can be decided with two $\mathcal{C}$ tests. For the class $\mathcal{C}$ as indicated above, it can be shown that these two tests can be captured by a single $\mathcal{C}$ test. Therefore, Is-MinEX is in $\mathcal{C}$ in this case.

Next, we show that whenever OMQA is in NP, we have that Is-MinEX is in D^{P} by analysing the proof of Theorem 6.7.

Theorem 6.8. *For any language $\mathcal{L}$ of existential rules, if* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in* NP*, then* $\text{Is-MinEX}(\leq, \text{UCQ}, \mathcal{L})$ *is in* D^{P}.

Proof. If the complexity of OMQA is in NP, then, by Theorem 6.7, Is-MinEX can be solved with a single NP and a single test that is a complement to an NP test, calling a single NP test. Note that the two NP tests are independent, and therefore they can be captured as a single NP tests, where a certificate is obtained by concatenating the certificates of these two NP tests. Therefore, in this case, Is-MinEX is solvable with a single NP and a single coNP test, that is, it is in D^{P}. □

The tractability of Is-MinEX in data complexity for FO-rewritable languages follows from Theorem 3.18, which states that computing the set of all subset-minimal explanations for the OMQ is feasible in polynomial time in data complexity. It is easy to extend this result to compute the set of all cardinality-minimal explanations: we first compute all the subset-minimal explanations in polynomial time, and then select the cardinality-minimal ones, also in polynomial time.

Corollary 6.9. *Let $\mathcal{L}$ be a FO-rewritable language over existential rules. Then* Is-MinEX$(\leq, \mathrm{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

We proceed with the hardness results for Is-MinEX. First, we show that recognising cardinality-minimal explanations is intractable already for guarded full existential rules in data complexity, even though query answering is known to be in polynomial time in this fragment. This is in strong contrast with the complexity of recognising subset-minimal explanations, which remains in polynomial time in data complexity.

Theorem 6.10. Is-MinEX$(\leq, \mathrm{UCQ}, \mathsf{GF})$ *is* coNP*-hard in data complexity.*

Proof. We reduce a coNP-complete problem UNSAT to Is-MinEX$(\leq, \mathrm{UCQ}, \mathsf{GF})$ in data complexity. Let ϕ be a 3CNF formula of the form $c_1 \wedge \cdots \wedge c_m$, where c_i is a clause. Assume that $x_1, \ldots, x_k$ are variables in ϕ. Without loss of generality, assume that assigning the value 0 all the variables x_i does not satisfy ϕ. We construct a database D, an OMQ (Q, Σ), and a subset $E \subseteq D$ as follows.

First, we construct the database D in three steps. The first part of the database D_{var} captures the possible assignments to variables. For each variable x_i in ϕ, we add to D_{var} the facts $val(i, 0)$ and $val(i, 1)$.

The second part of the database D_{cl} captures the clauses of ϕ. For each clause $c_j = x_{j_1} \vee \neg x_{j_2} \vee \neg x_{j_3}$ in ϕ, add to D_{cl} the fact

$$cl(j, j_1, p, j_2, n, j_3, n).$$

The third part of the database D_{reach} is added to enable the reachability type constructions to ensure containment of cl, var facts and satisfiability of ϕ. For each $i \in \{0, \ldots, k-1\}$, we add to D_{reach} the fact $succ\_var(i, i+1)$, which captures that the $i+1^{\text{th}}$ variable is the successor of the i^{th} variable. Similarly, for each $j \in \{0, m-1\}$, we add to D_{reach} the fact $succ\_cl(j, j+1)$, which captures that that the $j+1^{\text{th}}$ clause is the successor of the j^{th} clause. We further add to D_{reach} facts that capture the numbers of variables and clauses, and act as initial conditions in the arguments:

$$no\_of\_var(k), no\_of\_cl(m), \text{ and } var\_ch(0), cl\_ch(0), sat\_ch(0).$$

Finally, we add $sat()$ to D. This fact will be contained in a MinEX only if ϕ is not satisfiable. Therefore the database is $D = D_{var} \cup D_{cl} \cup D_{reach} \cup \{sat()\}$.

We construct the program Σ as follows. It consists of three parts. The first part Σ_{val} ensures that $all\_val()$ holds if, for each variable x_i, at least one of $val(j, 0)$ and

$val(j, 1)$ is present. , and Note that capital letters and the underscores (_) stand for fresh universally quantified variables, small letters and numbers stand for constants:

$$\begin{aligned}\Sigma_{val} = \{&val(J, \_) \to in\_val(J),\\ &var\_ch(I) \wedge succ\_var(I, J) \wedge in\_val(J) \to var\_ch(J),\\ &no\_of\_var(M) \wedge var\_ch(M) \to all\_val()\}.\end{aligned}$$

The second part Σ_{cl} ensures that that $all\_cl()$ holds if all cl facts are present:

$$\begin{aligned}\Sigma_{cl} = \{&cl(J, \_, \_, \_, \_, \_, \_) \to in\_cl(J),\\ &cl\_ch(I) \wedge succ\_cl(I, J) \wedge in\_cl(J) \to cl\_ch(J),\\ &no\_of\_cl(N) \wedge cl\_ch(N) \to all\_cl()\}.\end{aligned}$$

The third part ensures that $sat()$ holds if the assignment, given by val facts satisfies the formula ϕ:

$$\begin{aligned}\Sigma_{sat} = \{&cl(J, I_1, p, \_, \_, \_, \_) \wedge val(I_1, 1) \to sat(J),\\ &cl(J, I_1, n, \_, \_, \_, \_) \wedge val(I_1, 0) \to sat(J),\\ &cl(J, \_, \_, I_2, p, \_, \_) \wedge val(I_2, 1) \to sat(J),\\ &cl(J, \_, \_, I_2, n, \_, \_) \wedge val(I_2, 0) \to sat(J),\\ &cl(J, \_, \_, \_, \_, I_3, p) \wedge val(I_3, 1) \to sat(J),\\ &cl(J, \_, \_, \_, \_, I_3, n) \wedge val(I_3, 0) \to sat(J),\\ &sat\_ch(I) \wedge succ\_cl(I, J) \wedge sat(J) \to sat\_ch(J),\\ &no\_of\_cl(N) \wedge sat\_ch(N) \to sat()\}.\end{aligned}$$

Thus the program is $\Sigma = \Sigma_{val} \cup \Sigma_{cl} \cup \Sigma_{sat}$. We take the query to be

$$Q = sat\_ch(0) \wedge all\_val() \wedge all\_cl() \wedge sat().$$

Finally, we take the candidate $\leq$-MinEX for $D \models (Q, \Sigma)$ to be

$$E = D_{cl} \cup \{var(i, 0) \mid x_i \text{ is a variable in } \phi\} \cup D_{reach} \cup \{sat()\}$$

It is easy to see that both D and E entail (Q, Σ). Now we show that E is a $\leq$-MinEX for $D \models (Q, \Sigma)$ iff ϕ is unsatisfiable.

($\Rightarrow$) For the forward direction, suppose that E is a $\leq$-MinEX for $D \models (Q, \Sigma)$. Assume, for a contradiction, that if ϕ was satisfiable, and let σ be a satisfying assignment. Then we can construct the following $\leq$-MinEX E' for (Q, Σ). Take

$E' = D_{cl} \cup \{var(i, v(x_i)) \mid x_i \text{ is a variable in } \phi\} \cup D_{reach}$. Note that, by construction, $E' \models (Q, \Sigma)$. Note that $|E'| = |E| - 1 < |E|$. This is a contradiction to the minimality of E.

($\Leftarrow$) For the backwards direction, suppose that ϕ is unsatisfiable. Assume, for a contradiction, that E is not a $\leq$-MinEX for (Q, Σ). We have observed that $E \models (Q, \Sigma)$, and so the only way E might fail to be a MinEX if it is not cardinality-minimal. Let $E' \subseteq D$ be a subset such that $E' \models (Q, \Sigma)$ and $|E'| < |E|$. Then, by construction, E' must contain D_{cl}, exactly one of $val(i, 0)$ and $val(i, 1)$ for each i, and D_{reach}. Therefore, $sat() \notin E'$, and $E' \models (sat(), \Sigma)$. Therefore, by construction, *val* facts give a satisfying assignment to ϕ, and so ϕ is satisfiable, which is a contradiction. □

Next, we show that the complexity of Is-MinEX increases in *fp*-combined complexity and is D^{P}-hard for any language of existential rules (even if the program is empty). This result follows from the proof of Theorem 3.12, showing the D^{P}-hardness of recognising a subset-minimal explanation in *fp*-combined complexity. Note that in that proof the constructed set is not only subset-minimal but also cardinality-minimal.

Corollary 6.11. Is-MinEX$(\leq, \mathrm{UCQ}, \mathcal{L}_\emptyset)$ *is* D^{P}*-hard in fp-combined complexity.*

In Theorem 3.13, we have shown that Is-MinEX$(\subseteq, \mathrm{UCQ}, \mathsf{A})$ is $\mathrm{D}^{\mathrm{Exp}}$-hard in *ba*-combined complexity. The reduction used in that work immediately applies to the cardinality-minimal explanations, as the database and the OMQ constructed in that proof admit always a unique MinEX, which is clearly cardinality-minimal. Therefore, we have the following result, which implies all the $\mathrm{D}^{\mathrm{Exp}}$-hardness results in Table 6.1.

Corollary 6.12. Is-MinEX$(\leq, \mathrm{UCQ}, \mathsf{A})$ *is* $\mathrm{D}^{\mathrm{Exp}}$*-hard in ba-combined complexity.*

We complete the analysis of Is-MinEX with the result that Is-MinEX is as hard as OMQA in *ba*-combined and combined complexity. The following result covers all the remaining hardness results in Table 6.1.

Theorem 6.13. *For any language* $\mathcal{L}$ *of existential rules,* OMQA$(\mathrm{UCQ}, \mathcal{L})$ *is polynomial-time reducible to* Is-MinEX$(\leq, \mathrm{UCQ}, \mathcal{L})$ *in ba-combined and combined complexity.*

Proof. We reduce OMQA$(\mathrm{UCQ}, \mathcal{L})$ to Is-MinEX$(\leq, \mathrm{UCQ}, \mathcal{L})$. Let D be a database and (Q, Σ) be an OMQ, which is an instance of OMQA$(\mathrm{UCQ}, \mathcal{L})$. 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 $D' = \{p(), r()\}$, $\Sigma' = \Sigma \cup \{p() \to f \mid f \in D\} \cup \{r() \to$

$g \mid g \in D_Q\}$, and $E = \{p()\}$. 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 $D' \models (Q, \Sigma')$. □

6.3.2 All Smallest Explanations

We continue the analysis with the problem of recognising all cardinality-minimal explanations. Rather surprisingly, we show that All-MinEX has the same complexity as Is-MinEX for the languages under consideration in this case, which is in contrast to the subset-minimal case, where we observed the differences in complexity for these two problems in data complexity. All membership results for All-MinEX are covered by the following theorem, except for the P and D^{P} ones, for which we need tighter statements.

Theorem 6.14. *For any language $\mathcal{L}$ of existential rules, if* $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ *is in $\mathcal{C}$, then* $\textsc{All-MinEX}(\leq, \mathrm{UCQ}, \mathcal{L})$ *can be decided by a linear number of $\mathcal{C}$ tests and a single test in co-*$(\mathrm{NP}^{\mathcal{C}})$.

Proof. Let $\mathcal{E}$ be a set of subsets of the database D, and (Q, Σ) be an OMQ. First, we check in polynomial time whether all sets in $\mathcal{E}$ are of the same size. If not, we return that $\mathcal{E}$ is not the set of all $\leq$-MinEXs for $D \models (Q, \Sigma)$. Otherwise, we check whether each $E \in \mathcal{E}$ entails the OMQ (Q, Σ) by calling a linear number of $\mathcal{C}$ tests. Suppose that all sets in $\mathcal{E}$ are of size k. Note that to check whether every set in $\mathcal{E}$ is cardinality-minimal and that there is no $\leq$-MinEX outside of $\mathcal{E}$, it is enough to check that there is no set outside of $\mathcal{E}$ of size k or smaller that entails the OMQ. We can check the existence of such a set with a single NP test calling a single $\mathcal{C}$ test. □

Note that all the membership results for All-MinEX, apart from the tractability results in data complexity and the membership D^{P} in *fp*-combined and *ba*-combined complexity. The following covers the D^{P}-membership results in Table 6.1. It follows from the proof .

Theorem 6.15. *For any language $\mathcal{L}$ of existential rules, if* $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ *is in* NP*, then* $\textsc{Is-MinEX}(\leq, \mathrm{UCQ}, \mathcal{L})$ *is in* D^{P}.

Proof. If the complexity of OMQA is in NP, then, by the proof of Theorem 6.14, All-MinEX can be solved with (i) a single NP and (ii) a linear number of tests that are complement to an NP test, calling a single NP test. Note that in (ii) the two NP tests (one calling another) are independent and thus the two NP tests can be

expressed as a single NP tests, where a certificate is obtained by concatenating the certificates of these two tests. Therefore (ii) can be checked with a linear number of coNP tests. Note that, from the proof of Theorem 6.14, we can see that all of these coNP tests are independent too. Thus, they can be captured by a single coNP tests for which a certificate is obtained by concatenating the certificates of these linearly many coNP tests. Therefore, All-MinEX is solvable with a single NP and a single coNP test, that is, it is in D^{P}. □

For FO-rewritable languages, the membership results in P for All-MinEX follow by a similar argument to the one for Corollary 6.9 for Is-MinEX. By Theorem 3.18, we can compute all subset-minimal explanations in polynomial time, and then compute the set of all $\leq$-MinEXs in polynomial time.

Corollary 6.16. *Let $\mathcal{L}$ be a FO-rewritable language over existential rules. Then* All-MinEX$(\leq, \mathrm{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

We proceed with the hardness results for All-MinEX. First, we show that the All-MinEX is coNP-hard in data complexity when a program is guarded full. The proof of this result borrows some ideas from the proof of Theorem 6.10.

Theorem 6.17. All-MinEX$(\leq, \mathrm{UCQ}, \mathsf{GF})$ *is* coNP*-hard in data complexity.*

Proof. We reduce a coNP-complete problem UNSAT to All-MinEX$(\leq, \mathrm{UCQ}, \mathsf{GF})$ in data complexity. Let ϕ be a 3CNF formula of the form $c_1 \wedge \cdots \wedge c_m$, where c_i is a clause. Assume that $x_1, \ldots, x_n$ are variables in ϕ. We construct a database D, an OMQ (Q, Σ) and a set $\mathcal{E}$ of subsets of D as follows.

The database consists of two parts D_{sat} and D_{unsat}. The first part D_{sat} consists of the three parts D_{val}, D_{cl} and D_{reach} as defined in the proof of Theorem 6.10. we take $D_{sat} = D_{val} \cup D_{cl} \cup D_{reach}$. Note that D_{sat} here is equal to $D \setminus \{sat()\}$ in the proof of Theorem 6.10.

The other part of the database D_{unsat} has $M = |D_{sat}| - n$ facts. The idea behind this construction is that D_{unsat} will be the only cardinality-minimal explanation only if ϕ is unsatisfiable. If ϕ is satisfiable, then some subsets of D_{sat} of size M will also be MinEXs. We take D_{unsat} to contain, for each $i \in \{0, \ldots, M-3\}$, a fact $succ\_unsat(i, i+1)$, and further two facts, namely, $unsat(0)$ and $no\_for\_unsat(M-2)$.

We construct the program Σ as follows. We take $\Sigma_{val}, \Sigma_{cl}$ and Σ_{sat} as in the proof of Theorem 6.10. Another part Σ_{unsat} of the program captures that $unsat()$ holds iff

all the facts in D_{unsat} are present:

$$\Sigma_{unsat} = \{unsat(I) \wedge r_{unsat}(I, J) \rightarrow unsat(J), \\ no\_of\_unsat(N) \wedge unsat(N) \rightarrow unsat()\}.$$

We further add two rules, which capture that a fact $processed()$ can be inferred via some subset of D_{sat} that gives a satisfying assignment of ϕ or the facts in D_{unsat}:

$$\Sigma_{proc} = \{in\_val() \wedge in\_cl() \wedge sat() \rightarrow processed(), unsat() \rightarrow processed()\}.$$

Thus the program is $\Sigma = \Sigma_{val} \cup \Sigma_{cl} \cup \Sigma_{sat} \cup \Sigma_{unsat} \cup \Sigma_{proc}$. We take the query to be

$$Q = processed().$$

We take the candidate set of all $\leq$-MinEX to be $\mathcal{E} = \{D_{unsat}\}$. We show that $\mathcal{E}$ is the set of all $\leq$-MinEXs for $D \models (Q, \Sigma)$ iff ϕ is unsatisfiable.

($\Rightarrow$) Suppose that ϕ is satisfiable. Then from a satisfying assignment σ, we can construct a $\leq$-MinEX E' for $D \models (Q, \Sigma)$ as follows. Take $E' = D_{cl} \cup \{val(i, \sigma(x_i)) \mid x_i \text{ is a variable in } \phi\} \cup D_{reach}$. Note that, by construction, $E' \models (Q, \Sigma)$. Note that $|E'| = M - n$. This shows that $\mathcal{E}$ is not the set of all $\leq$-MinEXs as it is missing some $\leq$-MinEXs.

($\Leftarrow$) Suppose that $\mathcal{E}$ is not a $\leq$-MinEX for $D \models (Q, \Sigma)$. Let $E' \subseteq D$ be a $\leq$-MinEX not in subset $\mathcal{E}$. As $E' \models (Q, \Sigma)$, by construction, either $E' \models unsat()$ or $E' \models in\_val() \wedge in\_cl() \wedge sat()$. In the former case, by construction, E' must be D_{unsat}, which we know is not the case. Therefore, E' entails $in\_val() \wedge in\_cl() \wedge sat()$ and is of size at most M. The set E' must contain all the facts in D_{cl}, D_{reach} and at least n facts from D_{val}, that is, it must be at least of size M. That is, E' is exactly of size M and contains n *val* facts, which give a satisfying assignment to ϕ, and so ϕ is satisfiable. □

The remaining hardness results for All-MinEX in Table 6.1 follow from the proofs of Corollaries 6.11 and Corollary 6.12, and Theorem 6.13. In those proofs, the constructed $\leq$-MinEX E is unique, and thus we can take the candidate set of all $\leq$-MinEXs to be $\mathcal{E} = \{E\}$ for those proofs to apply to All-MinEX.

6.3.3 Relevant Facts for Preferred Explanations

We proceed with the problem MinEX-Rel, which asks whether a distinguished fact is contained in some cardinality-minimal explanation for an OMQ answer. As before, we first provide a general algorithm that covers most of the membership results in Table 6.2.

Table 6.2: Complexity of MinEX-Rel, MinEX-Irrel, and MinEX-Nec for $(\leq, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**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}	$\leq \text{P}^{\text{NExp}}$	$\leq \text{P}^{\text{NExp}}$
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

Theorem 6.18. *For any language $\mathcal{L}$ of existential rules, if* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* $\text{MinEX-Rel}(\leq, \text{UCQ}, \mathcal{L})$ *is in* $\text{P}^{\text{NP}^{\mathcal{C}}[O(\log n)]}$*, i.e., can be decided with a logarithmic number of* $\text{NP}^{\mathcal{C}}$ *tests.*

Proof. Let D be a database, (Q, Σ) be an OMQ, and $\psi \in D$ be a distinguished fact. First, we carry out a binary search on $\{1, \ldots, |D|\}$ to find the size of a smallest subset of D that entails (Q, Σ), i.e., the size of a cardinality-minimal explanation. At each step, for a number k, we guess a subset of D of size k and check whether it entails the OMQ in $\mathcal{C}$, and continue our binary search accordingly. We can clearly find the size k' of a smallest MinEX with a logarithmic number of $\mathcal{C}$ calls in the size of the database.

Having found the size k', we only need to verify whether there exists a $\leq$-MinEX containing ψ. To do so, simply guess a subset of D of size k' that contains ψ, and check in $\mathcal{C}$ whether it entails the OMQ. As this is just a single check, the number of times that we need to call an NP test followed by a $\mathcal{C}$ test remains logarithmic. □

Let us briefly discuss the implications of this result. If OMQA is in P, then we have that MinEX-Rel is in $\text{P}^{\text{NP}^{\text{P}}[O(\log n)]} = \text{P}^{\text{NP}[O(\log n)]} = \Theta_2^{\text{P}}$. If OMQA is in NExp, then MinEX-Rel is in $\text{P}^{\text{NP}^{\text{NExp}}[O(\log n)]} = \text{P}^{\text{P}^{\text{NExp}}[O(\log n)]} = \text{P}^{\text{NExp}}$. If OMQA is in $\mathcal{C} \in \{\text{PSpace}, \text{Exp}, \text{2Exp}\}$, then MinEX-Rel is in $\text{P}^{\text{NP}^{\mathcal{C}}[O(\log n)]} = \text{P}^{\mathcal{C}[O(\log n)]} = \mathcal{C}$. Notice that the above result does not cover the P membership results in data complexity, and the Θ_2^{P} membership results in *fp*-combined and *ba*-combined complexity. We first show that the Θ_2^{P} membership results for MinEX-Rel in *fp*- and *ba*-combined complexity in Table 6.2 follow from the proof of Theorem 6.18.

Theorem 6.19. *For any language $\mathcal{L}$ of existential rules, if* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in* NP*, then* $\text{MinEX-Rel}(\leq, \text{UCQ}, \mathcal{L})$ *is in* Θ_2^{P}.

Proof. If the complexity of OMQA is in NP, then, by Theorem 6.18, MinEX-Rel can be solved with a logarithmic number of NP tests, calling a single NP test. Note that the two NP tests can be combined by concatenating the certificates, and so, in this case, MinEX-Rel is solvable with a logarithmic number of NP tests, that is, it is in Θ_2^{p}. □

As before, the data complexity tractability results for FO-rewritable languages are not covered by the generic algorithm too, but follow by similar arguments as Corollary 6.9. In this case, we can compute all MinEXs and check whether any cardinality-minimal explanation contains a distinguished fact in polynomial time.

Corollary 6.20. *Let $\mathcal{L}$ be a FO-rewritable language over existential rules. Then* MinEX-Rel$(\leq, \mathrm{UCQ}, \mathcal{L})$ *is in* P *in data complexity.*

Interestingly, MinEX-Rel is hard already in data complexity for the guarded, full existential rules as it lies in the second level of the polynomial hierarchy.

Theorem 6.21. MinEX-Rel$(\leq, \mathrm{UCQ}, \mathsf{GF})$ *is Θ_2^{p}-hard in data complexity.*

Proof. We reduce from the following Θ_2^{p}-complete problem:

Problem: Comp-SAT
Input: A pair (A, B) of sets of 3CNF formulas
Question: Is the number of satisfiable formulas in A is greater than the number of satisfiable formulas in B?

This problem remains Θ_2^{p}-complete even if we impose the following further conditions [113]:

- $|A| = |B|$;
- all formulas in A and B have the same number of clauses and the same variables;
- If $A = \{\phi_1^a, \ldots, \phi_n^a\}$ and $B = \{\phi_1^b, \ldots, \phi_n^b\}$, then satisfiable formulas come first in the ordering, i.e., for any $u \in \{a, b\}$ and k, if ϕ_{k+1}^u is satisfiable, then so is ϕ_k^u.

Given these restrictions, the problem asks whether there is a number k such that ϕ_k^a is satisfiable and ϕ_k^b is unsatisfiable.

Let $A = \{\phi_1^a, \ldots, \phi_n^a\}$ and $B = \{\phi_1^b, \ldots, \phi_n^b\}$ be an instance of Comp-SAT with the restrictions, given above. Suppose that each formula has m clauses and the variables $x_1, \ldots, x_v$. Now we construct an instance of MinEX-Rel$(\leq, \mathrm{UCQ}, \mathsf{GF})$, where only the database depends on the instance of Comp-SAT.

The database D is constructed as follows. The first part of the database D_{val} captures the possible assignments to variables. This consists of the facts

$$val(u, k, i, 0, \ell), \qquad val(u, k, i, 1, \ell),$$

where $u \in \{a, b\}$ identifies one of the sets A and B, $k \in \{1, \ldots, n\}$ is the index of a formula in that set, $i \in \{1, \ldots, v\}$ is the index of a variable, and $\ell \in \{1, 2, 3, 4\}$ gives four copies of each assignment fact. In particular, *val* facts encode assignments to the variables.

The second part of the database D_{cl} captures the clauses of the formulas, both in A and in B. For example, for a clause $c_j = x_{j_1} \vee \neg x_{j_2} \vee \neg x_{j_3}$ in ϕ_k^a in A, add to D_{cl} the fact $cl(a, k, j, j_1, p, j_2, n, j_3, n)$. In the *cl* facts, the first argument is either a or b indicating the set of formulas, the second argument gives the index of the formula in that set, the third gives the index of a clause of that formula, the fourth argument gives the index of a variable of the first literal of the clause and the fifth argument indicates whether that literal is positive of negative, similarly, sixth and seventh arguments specify the second literal of the clause, and the eighth and ninth arguments specify the third literal.

We add facts to the database that refer to the unsatisfiability of a formula. We add three copies of such fact for each formula both in A and B:

$$unsat(u, k, \ell),$$

where $u \in \{a, b\}, 1 \leq k \leq n$, and $\ell \in \{1, 2, 3\}$.

We add a number of structural facts to the database. These can be divided into four parts. The first part D_{reach} is a set of binary facts that are used as connecting facts in the reachability constructions. For each $u \in \{a, b\}$, $1 \leq k \leq n$ and for $i \in \{1, \ldots, v\}$, we introduce facts

$$succ\_var(u, k, i - 1, i)$$

that are used for reachability argument on variables,

$$succ\_cl(u, k, j - 1, j)$$

for $1 \leq j \leq m$ that are used for reachability argument on clauses,

$$succ\_form(k - 1, k)$$

for $1 \leq k \leq n$ that are used for reachability argument on formulas.

We introduce the facts D_{ch} that act as starting points for reachability arguments, namely, $var\_ch(u, k, 0)$, $cl\_ch(u, k, 0)$, $sat\_ch(u, k, 0)$, and $proc\_ch(0)$, where $u \in \{a, b\}$ and $1 \leq k \leq n$.

Also, we also add the facts D_{no} that capture how many variables, clauses and formulas there are $no\_of\_var(v)$, $no\_of\_cl(m)$, and $no\_of\_form(n)$, respectively. The database also contains the fact $sat\_unsat()$.

We construct the OMQ in such a way that $sat\_unsat()$ must be in a $\leq$-MinEX if there is k such that ϕ_k^a is satisfiable and ϕ_k^b is unsatisfiable. Now we construct the program. The first part of the program Σ_{val} ensures that four copies of a valuation fact is present for each variable. In particular, $in\_val(u, k)$ holds if there are all four copies of a val fact for $u \in \{a, b\}$ and $k \in \{1, \ldots, n\}$:

$$\bigwedge_{1 \leq \ell \leq 4} val(U, K, J, V, \ell) \rightarrow in\_val(U, K, J),$$
$$var\_ch(U, K, I) \wedge succ\_var(U, K, I, J) \wedge in\_val(U, K, J) \rightarrow var\_ch(U, K, J),$$
$$no\_of\_var(M) \wedge var\_ch(U, K, M) \rightarrow in\_val(U, K),$$

The second part of the program Σ_{cl} ensures that all cl facts are present. In particular, $in\_cl(u, k)$ holds if all facts $cl(u, k, \_, \_, \_, \_, \_, \_)$ from D_{cl} are present:

$$cl(U, K, J, \_, \_, \_, \_, \_) \rightarrow in\_cl(U, K, J),$$
$$cl\_ch(U, K, I) \wedge succ\_cl(U, K, I, J) \wedge in\_cl(U, K, J) \rightarrow cl\_ch(U, K, J),$$
$$no\_of\_cl(N) \wedge cl\_ch(U, K, N) \rightarrow in\_cl(U, K).$$

The next part of the program Σ_{sat} captures the satisfiability of a formula. We note that $sat(u, k)$ holds if the k^{th} formula from the set $u \in \{a, b\}$ is satisfiable by the assignment given by val facts. Note that val assignment is allowed to be improper, that is assigning more than one value to some variables. As we are looking into cardinality-minimal explanations, we shall see that any assignment that is cardinality-minimal must be a proper assignment.

$$cl(U,K,J,I_1,p,\_,\_,\_,\_) \wedge \bigwedge_{1\leq\ell\leq 4} val(U,K,I_1,1,\ell) \rightarrow sat(U,K,J),$$
$$cl(U,K,J,I_1,n,\_,\_,\_,\_) \wedge \bigwedge_{1\leq\ell\leq 4} val(U,K,I_1,0,\ell) \rightarrow sat(U,K,J),$$
$$cl(U,K,J,\_,\_,I_2,p,\_,\_) \wedge \bigwedge_{1\leq\ell\leq 4} val(U,K,I_2,1,\ell) \rightarrow sat(U,K,J),$$
$$cl(U,K,J,\_,\_,I_2,n,\_,\_) \wedge \bigwedge_{1\leq\ell\leq 4} val(U,K,I_2,0,\ell) \rightarrow sat(U,K,J),$$
$$cl(U,K,J,\_,\_,\_,\_,I_3,p) \wedge \bigwedge_{1\leq\ell\leq 4} val(U,K,I_3,1,\ell) \rightarrow sat(U,K,J),$$
$$cl(U,K,J,\_,\_,\_,\_,I_3,n) \wedge \bigwedge_{1\leq\ell\leq 4} val(U,K,I_3,0,\ell) \rightarrow sat(U,K,J),$$

$$sat\_ch(U,K,I) \wedge succ\_cl(U,K,I,J) \wedge sat(U,K,J) \rightarrow sat\_ch(U,K,J),$$
$$no\_of\_cl(N) \wedge sat\_ch(U,K,N) \rightarrow sat(U,K).$$

Also, add to Σ_{sat}

$$unsat(U,K,1) \wedge unsat(U,K,2) \wedge unsat(U,K,3) \rightarrow unsat(U,K)$$

Next, we introduce a number of formulas that ensure that it is recorded in the database whether there is k such that ϕ_k^a is satisfiable, and ϕ_k^b is unsatisfiable. We say that $record(k)$ holds if (un)satisfiability is captured of both ϕ_k^a and ϕ_k^b.

$$sat(a,K) \wedge sat(b,K) \rightarrow record(K),$$
$$sat(a,K) \wedge unsat(b,K) \wedge sat\_unsat() \rightarrow record(K),$$
$$unsat(a,K) \wedge sat(b,K) \rightarrow record(K),$$
$$unsat(a,K) \wedge unsat(b,K) \rightarrow record(K).$$

The final part of the program Σ_{proc} ensures, that the query $processed()$ is entailed if all the structural facts and (un)satisfiability is captured of all formulas both in A and B.

$$in\_val(a,K) \wedge in\_val(b,K) \rightarrow in\_val(K),$$
$$in\_cl(a,K) \wedge in\_cl(b,K) \rightarrow in\_cl(K),$$
$$in\_val(K) \wedge in\_cl(K) \wedge record(K) \rightarrow proc(K),$$
$$proc\_ch(I) \wedge succ\_form(I,J) \wedge proc(J) \rightarrow proc\_ch(J),$$
$$no\_of\_form(M) \wedge proc\_ch(M) \rightarrow processed().$$

Finally, the query is $Q = \mathit{processed}()$. The distinguished fact is $\mathit{sat\_unsat}()$.

We show that there is k such that ϕ^a_k is satisfiable and ϕ^b_k is unsatisfiable iff there is a $\leq$-MinEX for $D \models (Q, \Sigma)$ that contains $\mathit{sat\_unsat}()$. First of all, we note that if a subset $E \subseteq D$ entails the OMQ (Q, Σ), then a number of facts have to be present, for example, at least three copies of $\mathit{val}(u, k, i, \star, \ell)$, where $\star \in \{0, 1\}$. Also, a number of structural facts.

Note that if ϕ^u_k is satisfiable, for $\mathit{sat}(u, k)$ to hold, we need a single extra fact $\mathit{sat\_ch}(u, k, 0)$ to be present. If ϕ^u_k is satisfiable or unsatisfiable, for $\mathit{unsat}(u, k)$ to hold, we need three extra $\mathit{unsat}(u, k, \ell)$ facts. If ϕ^u_k is unsatisfiable, for $\mathit{sat}(u, k)$ to hold we need at least five extra facts, namely, $\mathit{sat\_ch}(u, k, 0)$ and four extra val fact copies for some variables (so that the assignment is improper). As a result, for cardinality-minimality, we prefer to capture the satisfiability of each formula correctly. Note that recording the satisfiability of ϕ^u_k correctly requires fewer facts than recording incorrectly with the difference of two facts. Intuitively, this differences ensures that every $\leq_w$-MinEX will contain $\mathit{sat\_unsat}()$ iff there is k such that ϕ^a_k is satisfiable and ϕ^b_k is unsatisfiable.

($\Rightarrow$) Suppose that there is k such that ϕ^a_k is satisfiable and ϕ^b_k is unsatisfiable. Let D be a database and (Q, Σ) be an OMQ constructed as above. We construct a $\leq$-MinEX E for $D \models (Q, \Sigma)$ as follows. Add D_{cl}, D_{reach}, D_{no} to E. Also, for each $u \in \{a, b\}$ and $k \in \{1, \ldots, n\}$, add to E the facts $\mathit{var\_ch}(u, k, 0)$, $\mathit{cl\_ch}(u, k, 0)$, and $\mathit{proc\_ch}(0)$. For each $u \in \{a, b\}$ and $k \in \{1, \ldots, n\}$, if ϕ^u_k is satisfiable with valuation σ, add to E the facts $\mathit{sat\_ch}(u, k, 0)$, and $\mathit{val}(u, k, j, \sigma(x_j), \ell)$ for each $j \in \{1, \ldots, v\}$ and $\ell \in \{1, 2, 3, 4\}$. If ϕ^u_k is unsatisfiable, then add $\mathit{unsat}(u, k, \ell)$ for $\ell \in \{1, 2, 3\}$, and $\mathit{val}(u, k, j, 0, \ell)$ for each $j \in \{1, \ldots, v\}$ and $\ell \in \{1, 2, 3, 4\}$. Further, add $\mathit{sat\_unsat}()$ to E. It is easy to see that $E \models (Q, \Sigma)$. By the reasoning above, it can be seen that E is cardinality-minimal.

($\Leftarrow$) Suppose that there is a $\leq$-MinEX E for $D \models (Q, \Sigma)$ that contains $\mathit{sat\_unsat}()$. As we have seen above, E captures satisfiability and unsatisfiability of formulas ϕ^u_k. And, further, $\mathit{sat\_unsat}()$ is present implies that there is k such that ϕ^a_k is satisfiable and ϕ^b_k is unsatisfiable. □

Next, we show that MinEX-Rel is Θ^{p}_2-hard even with the empty program in *fp*-combined complexity. Interestingly, the complexity for MinEX-Rel does not increase for the languages containing GF as we move up from the data complexity to *fp*-combined complexity.

Theorem 6.22. MinEX-Rel$(\leq, \mathrm{UCQ}, \mathcal{L}_\emptyset)$ *is* Θ^{p}_2*-hard in fp-combined complexity.*

Proof. We reduce from the following Θ_2^{p}-complete problem:

Problem: InCover
Input: A graph G and a distinguished node w in G.
Question: Is w contained in a vertex cover of G of minimum size?

To see that this problem is Θ_2^{p}-complete, note that Θ_2^{p} is closed under complement as it is a deterministic class. It is known that InAllIS, which asks whether a particular vertex is in all independent sets of maximum size, is Θ_2^{p}-complete. Note that InCover is the complement of InAllIS, and so is Θ_2^{p}-complete. Let $G = (V, E)$ be a graph, and w be a vertex of G. Let $V = \{v_1, \ldots, v_n\}$. We construct a database D, an ontology-mediated query (Q, Σ), and a distinguished fact as follows. We take the database to be

$$D = \{\mathit{edge}(0,1), \mathit{edge}(1,0), \mathit{edge}(1,1)\} \cup \{\mathit{vertex}(v,0), \mathit{vertex}(v,1) \mid v \in V\}.$$

In particular, the fact $\mathit{vertex}(v,1)$ corresponds to the vertex v being in a vertex cover. Also, at least one of the two constants in each *edge* fact is 1 as we require at least one of the endpoints of every edge to be in a vertex cover.

We take the program to be empty and the query consists of three parts. The first part requires the three *edge* facts to be present

$$\mathit{edge} = \mathit{edge}(0,1) \wedge \mathit{edge}(1,0) \wedge \mathit{edge}(1,1).$$

The second part requires the facts $\mathit{vertex}(v,0)$ to be present for all $v \in V$

$$\mathit{in} = \bigwedge_{v \in V} \mathit{vertex}(v,0).$$

The last part of the query captures how an edge is covered

$$\mathit{cover} = \bigwedge_{(u,v) \in E} (\mathit{edge}(X_u, X_v) \wedge \mathit{vertex}(u, X_u) \wedge \mathit{vertex}(v, X_v)).$$

Thus, the query is

$$Q = \mathit{edge} \wedge \mathit{in} \wedge \mathit{cover}.$$

We take the distinguished fact to be $\mathit{vertex}(w,1)$.

We claim that the vertex w belongs to a vertex cover of G of minimum size iff there is a $\leq$-MinEX for $D \models (Q, \Sigma)$ that contains $\mathit{vertex}(w,1)$.

($\Rightarrow$) Suppose that w belongs to a vertex cover of G of minimum size. Let V' be a vertex cover of minimum size that contains w. Take the following subset of the database $E = \{\mathit{edge}(0,1), \mathit{edge}(1,0), \mathit{edge}(1,1)\} \cup \{\mathit{vertex}(v,0) \mid v \in$

$V\} \cup \{\mathit{vertex}(v,1) \mid v \in V'\}$. Note that E contains $\mathit{vertex}(w,1)$. First, observe that $E \models (Q,\Sigma)$. To see this, note that the *edge* and *in* parts of the query are certainly entailed, and the *cover* part of the query is entailed as V' contains at least one endpoint of each edge. Next, we show that E is a smallest MinEX for (Q,Σ). Note that every $\leq$-MinEX for (Q,Σ) has to contain all *edge* facts and $\mathit{vertex}(v,0)$ facts for all $v \in V$. Notice that if there is an explanation that has fewer facts than E, then it has fewer $\mathit{vertex}(v,1)$ facts, each of them corresponding to a vertex being in a cover. This would imply that there is a cover of size smaller than V', which is a contradiction.

($\Leftarrow$) Suppose that there is a $\leq$-MinEX for $D \models (Q,\Sigma)$ that contains $\mathit{vertex}(w,1)$. Let E be such $\leq$-MinEX. Let $V' = \{v \mid \mathit{vertex}(v,1) \in E\}$. Notice that $w \in V'$. We claim that V' is a vertex cover of minimum size. Suppose, for a contradiction, that V' is not a vertex cover. Hence, there is an edge $(u,v) \in E$ such that $\{u,v\} \cap V' = \emptyset$. This means that $\mathit{vertex}(u,1)$ and $\mathit{vertex}(v,1)$ do not belong to E. However, $E \models (Q,\Sigma)$, which means that $E \models \mathit{edge}(X_u,X_v) \wedge \mathit{vertex}(u,X_u) \wedge \mathit{vertex}(v,X_v)$ too. Observe that, in this case, any homomorphism mapping $(\mathit{edge}(X_u,X_v) \wedge \mathit{vertex}(u,X_u) \wedge \mathit{vertex}(v,X_v))$ to E has to map X_u and X_v to the constant 0, because $\mathit{vertex}(u,1)$ and $\mathit{vertex}(v,1)$ do not belong to E. Therefore, it must be the case that $\mathit{edge}(0,0)$ belongs to E: a contradiction; hence V' is a vertex cover. Suppose, for a contradiction, that there is a cover V'' of smaller size than V'. Then take $E' = \{\mathit{edge}(0,1), \mathit{edge}(1,0), \mathit{edge}(1,1)\} \cup \{\mathit{vertex}(v,0) \mid v \in V\} \cup \{\mathit{vertex}(v,1) \mid v \in V''\}$. By above, we have that it entails the OMQ but it is of size smaller than E, which is a contradiction. □

The remaining hardness results follow from the proof of Theorem 6.13 by taking the distinguished fact to be $\psi = p()$. Note that the fact ψ is contained in every explanation, and thus it is $\leq$-relevant.

6.3.4 Necessary and Irrelevant Facts for Preferred Explanations

Now we analyse the problems MinEX-Irrel and MinEX-Nec, asking whether there is a $\leq$-MinEX not containing any forbidden set of facts and whether a fact is necessary, respectively. We first provide general approaches to solve MinEX-Irrel and MinEX-Nec for cardinality-minimal explanations. Surprisingly, in this case, the algorithms for MinEX-Irrel and MinEX-Nec are similar to that for MinEX-Rel. This is in contrast to the subset-minimality setting, where we observed significant

differences in complexity for these problems. We show all membership results in Table 6.2, but the Θ_2^{p} and P ones, with the following theorem.

Theorem 6.23. *For any language $\mathcal{L}$ of existential rules, if* $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ *is in $\mathcal{L}$, then* MinEX-Irrel$(\leq, \mathrm{UCQ}, \mathcal{L})$ *and* MinEX-Nec$(\leq, \mathrm{UCQ}, \mathcal{L})$ *are in* $\mathrm{P}^{\mathrm{NP}^{\mathcal{C}}[O(\log n)]}$.

Proof. As in the proof of Theorem 6.18, we can carry out a logarithmic number of NP tests, each calling a $\mathcal{C}$ test, to determine the size k' of a $\leq$-MinEX.

For MinEX-Irrel, we need to check if there is a subset that does not contain any of the forbidden sets of facts. We can guess such a subset in NP and then we check whether it entails the OMQ in $\mathcal{C}$. For MinEX-Nec, we need to ensure that there is no $\leq$-MinEX that does not contain the distinguished fact. The existence of such MinEX can be checked with an NP test, calling a single $\mathcal{C}$ test.

As for both problems we require a single extra check in $\mathrm{NP}^{\mathcal{C}}$, the number of times that we need to call an NP test followed by a $\mathcal{C}$ test remains logarithmic. □

This implies all membership results in Table 6.2 apart from the tractability ones in data complexity and Θ_2^{p} ones in *fp*- and *ba*-combined complexity. Note that, for both MinEX-Irrel and MinEX-Nec, we have that the Θ_2^{p} membership results follow by the same argument as Theorem 6.19 and the tractability results follow by the same argument as Corollary 6.9.

We proceed with the hardness results for MinEX-Irrel and MinEX-Nec. First, we note that MinEX-Nec can be reduced to the complement of MinEX-Irrel. To see this, note that f is contained in every $\leq$-MinEX iff there is no $\leq$-MinEX that does not contain $\{f\}$. Therefore, it suffices to show the hardness results only for MinEX-Nec.

To show the Θ_2^{p}-hardness of MinEX-Nec$(\leq, \mathrm{UCQ}, \mathsf{GF})$ in data complexity, we note that the proof of Theorem 6.21 immediately applies to MinEX-Nec, since in this construction there is a $\leq$-MinEX that contains a distinguished fact ψ, then every *every* $\leq$-MinEX contains this distinguished fact ψ, and so it is necessary.

To show the Θ_2^{p}-hardness of MinEX-Nec in *fp*-combined complexity, we observe that the construction in Theorem 6.22 gives a reduction of a Θ_2^{p}-complete problem InAllMinCover [112] to MinEX-Nec by taking a distinguished fact the same as in that proof. This problem asks whether, given a graph G and a distinguished node w, the node w belongs to *every* vertex cover of G of minimum size. Thus, we have shown that the problem MinEX-Nec$(\leq, \mathrm{UCQ}, \mathcal{L}_\emptyset)$ is Θ_2^{p}-hard in *fp*-combined complexity.

Finally, Theorem 6.13 immediately applies to MinEX-Nec if we take a distinguished fact to be $\psi = p()$. Thefeore, the remaining hardness results for MinEX-Nec in Table 6.2 in *ba*-combined and *combined* complexity hold.

6.4 Explanations with Minimal Weight

In this section, we consider problems associated with weight-minimal explanations. First, notice that the cardinality order is a special case of the weight order when all facts are assigned the same weight. We show that the complexity remains the same for the problems Is-MinEX and All-MinEX, but increases for the other problems in most cases.

6.4.1 Recognising Minimal Explanations

In this section, we study the problems of recognising a $\leq_w$-MinEX and the set of all $\leq_w$-MinEXs. We first analyse the problem of recognising weight-minimal explanations, Is-MinEX. We note that for this problem all the membership results follow from the analysis for Is-MinEX with respect to the cardinality order. In particular, note that the proofs of Theorem 6.7, Theorem 6.8, and Corollary 6.9 apply to the weight-minimal explanations. The only change that has to be done in those proofs is, when checking minimality, we need to guess a subset E' that has smaller *weight* than E, that is, $E' \leq_w E$, rather than smaller size. Therefore, the membership results for Is-MinEX$(\leq_w, \mathrm{UCQ}, \mathcal{L})$, presented in Table 6.1, hold.

As noted earlier, the cardinality order is a special case of the weight order. Therefore, all the hardness results for the cardinality order immediately apply to the weight order. This observation implies all the hardness results for Is-MinEX$(\leq_w, \mathrm{UCQ}, \mathcal{L})$ in Table 6.1.

By analogous arguments, we conclude that the results for All-MinEX with respect to the weight order follow from the analysis of All-MinEX with respect to the cardinality order.

6.4.2 Results for Other Explanation Problems

We proceed with the remaining explanation problems with respect to the weight order, which turn out to have different computational complexity from the problems considered so far.

Table 6.3: Complexity of MinEX-Rel, MinEX-Irrel, and MinEX-Nec for $(\leq_w, \mathcal{Q}, \mathcal{L})$ with $\mathcal{Q} \in \{\text{BCQ}, \text{UCQ}\}$.

$\mathcal{L}$	**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}	P^{NExp}	P^{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

We first give a general membership result for MinEX-Rel. Observe that the algorithm for MinEX-Rel $(\leq, \text{UCQ}, \mathcal{L})$ in the proof of Theorem 6.18 applies to MinEX-Rel $(\leq_w, \text{UCQ}, \mathcal{L})$ when a binary search is carried out on the sum of weights of all the facts in the database, rather than the number of facts. Note that the weights might be exponential, i.e., $O(2^n)$, in the size of the input as we are using binary encodings for weights, and thus the sum of the weights can be exponential in the size of the input. As binary search takes logarithmic time in the size of the search space, the algorithm takes a polynomial number of steps. Hence, we state the following result.

Theorem 6.24. *For any language $\mathcal{L}$ of existential rules, if* $\text{OMQA}(\text{UCQ}, \mathcal{L})$ *is in the complexity class $\mathcal{C}$, then* $\text{MinEX-Rel}(\leq_w, \text{UCQ}, \mathcal{L})$ *is in* $\text{P}^{\text{NP}^{\mathcal{C}}}$.

We observe that the above result implies all the membership results in Table 6.3, apart from the Δ_2^{P} membership results in *fp*- and *ba*-combined complexity and the tractability results in data complexity. We note that, by adapting the proof of Theorem 6.19, we immediately have that for any language $\mathcal{L}$ of existential rules, we have that whenever $\text{OMQA}(\text{UCQ}, \mathcal{L})$ is in NP, then $\text{MinEX-Rel}(\leq_w, \text{UCQ}, \mathcal{L})$ is in Δ_2^{P}. Also, we note that $\text{MinEX-Rel}(\leq_w, \text{UCQ}, \mathcal{L})$ is tractable in data complexity for any FO-rewritable language $\mathcal{L}$ of existential rules. It follows by the same arguments as Corollary 6.9.

We proceed with the hardness results for $\text{MinEX-Rel}(\leq_w, \text{UCQ}, \mathcal{L})$. First, we show the Δ_2^{P}-hardness in data complexity. Note that it is essential to use exponential weights to express a Δ_2^{P}-hard problem.

Theorem 6.25. $\text{MinEX-Rel}(\leq_w, \text{UCQ}, \text{GF})$ *is Δ_2^{P}-hard in data complexity.*

Proof. We reduce from the following Δ_2^{p}-complete problem MSA [108]: Given a *satisfiable* 3CNF formula ϕ on $X = \{x_1, \ldots, x_n\}$, decide whether lexicographically maximum assignment $\sigma(X)$ with respect to $\langle x_1, \ldots, x_n \rangle$ satisfying $\phi(X)$, which we denote by $\sigma_m(X)$, fulfills $\sigma_m(x_n) = 1$.

Let ϕ be a satisfiable 3CNF formula on $X = \langle x_1, \ldots, x_n \rangle$, which is an instance of the above problem.

First, we construct the database D in three steps. The first part of the database D_{val} captures the possible assignments to variables. That is, for each variable x_i in ϕ, add to D_{val} the facts $val(i, 0)$ and $val(i, 1)$. Furthermore, assign to each $val(i, 0)$ the weight $2^n + 2^{n-i}$, and to each $val(i, 1)$ the weight 2^n. In particular, note that, for any i, $val(i, 0)$ has weight higher than all $val(j, 0)$ for $i < j \leq n$ together. Furthermore, note that the sum of the weights of any n facts from D_{val} is smaller than the sum of any $n + 1$ facts from D_{val}. This will ensure that all weight-minimal explanations give proper assignments to the variables. All the remaining facts are structural and will be forced by the OMQ to be present in every explanation. Therefore we assign the weight 0 to all the remaining facts in the database D.

The second part of the database D_{cl} captures the clauses of ϕ. For each clause $c_j = x_{j_1} \vee \neg x_{j_2} \vee \neg x_{j_3}$ in ϕ, add to D_{cl} the fact

$$cl(j, j_1, p, j_2, n, j_3, n).$$

The third part of the database D_{reach} is added to enable the reachability type constructions to ensure containment of cl, var facts and satisfiablity of ϕ. We add to D_{reach} facts $r_{val}(i, i+1)$ for each $i \in [0, n-1]$, and $succ\_cl(i, i+1)$ for each $i \in [0, m-1]$. We further add to D_{reach} facts

$$no\_of\_var(k), no\_of\_cl(m), \text{ and } var\_ch(0), cl\_ch(0), sat\_ch(0).$$

Therefore the database is $D = D_{val} \cup D_{cl} \cup D_{reach}$. We take the distinguished fact to be $val(i, n)$.

We construct the program Σ as follows. It consists of three parts. The first part Σ_{val} ensures that $in\_val()$ holds if, for each variable x_i, at least one of $val(j, 0)$ and $val(j, 1)$ is present is in every MinEX. The second part Σ_{cl} ensures that that $in\_val()$ holds if all cl facts are in every MinEX, and the third part ensures that $sat()$ holds if the assignment, given by val facts satisfies the formula ϕ. Note that capital letters and the underscores (_) stand for fresh universally quantified variables, small letters

and numbers stand for constants.

$$\begin{aligned}\Sigma_{val} = \{&val(J, \_) \to in\_val(J),\\ &var\_ch(I) \wedge r_{val}(I, J) \wedge in\_val(J) \to var\_ch(J),\\ &no\_of\_var(M) \wedge var\_ch(M) \to in\_val()\},\end{aligned}$$

$$\begin{aligned}\Sigma_{cl} = \{&cl(J, \_, \_, \_, \_, \_, \_) \to in\_cl(J),\\ &cl\_ch(I) \wedge succ\_cl(I, J) \wedge in\_cl(J) \to cl\_ch(J),\\ &no\_of\_cl(N) \wedge cl\_ch(N) \to in\_cl()\},\end{aligned}$$

and

$$\begin{aligned}\Sigma_{sat} = \{&cl(J, I_1, p, \_, \_, \_, \_) \wedge val(I_1, 1) \to sat(J),\\ &cl(J, I_1, n, \_, \_, \_, \_) \wedge val(I_1, 0) \to sat(J),\\ &cl(J, \_, \_, I_2, p, \_, \_) \wedge val(I_2, 1) \to sat(J),\\ &cl(J, \_, \_, I_2, n, \_, \_) \wedge val(I_2, 0) \to sat(J),\\ &cl(J, \_, \_, \_, \_, I_3, p) \wedge val(I_3, 1) \to sat(J),\\ &cl(J, \_, \_, \_, \_, I_3, n) \wedge val(I_3, 0) \to sat(J),\\ &sat\_ch(I) \wedge succ\_cl(I, J) \wedge sat(J) \to sat\_ch(J),\\ &no\_of\_cl(N) \wedge sat\_ch(N) \to sat()\}.\end{aligned}$$

Thus the program is $\Sigma = \Sigma_{val} \cup \Sigma_{cl} \cup \Sigma_{sat}$. The query is $Q = in\_val() \wedge in\_cl() \wedge sat()$.

We show that lexicographically maximum satisfying assignment σ_m assigns 1 to x_n iff $\leq_w$-MinEX for $D \models (Q, \Sigma)$ contains $val(n, 1)$.

Suppose that lexicographically maximum satisfying assignment σ_m assigns 1 to x_n. Take $E = \{val(i, \sigma_m(x_i)) \mid 1 \leq i \leq n\} \cup D_{cl} \cup D_{reach}$. We have seen above that E is $\leq_w$-minimal, otherwise σ_m is not a lexicographically minimum assignment.

Similarly, if $\leq_w$-MinEX for $D \models (Q, \Sigma)$ contains $val(n, 1)$ then we have that the lexicographically maximum satisfying assignment σ_m assigns 1 to x_n. □

We show Δ_2^{p}-hardness of MinEX-Rel($\leq_w$, UCQ, $\mathcal{L}_\emptyset$) in *fp*-combined complexity.

Theorem 6.26. MinEX-Rel($\leq_w$, UCQ, $\mathcal{L}_\emptyset$) *is* Δ_2^{p}*-hard in fp-combined complexity.*

Proof. As in the proof of Theorem 6.25, we reduce from the Δ_2^{p}-complete problem MSA [108]. Let ϕ be a satisfiable 3CNF formula on $X = \langle x_1, \ldots, x_n \rangle$, which is an instance of MSA. We construct the database as follows. The first part of the database

captures the possible assignments to variables. That is, for each variable x_i in ϕ, add to D_{val} the facts

$$val(i, 0), \qquad val(i, 1).$$

Further, assign to each $val(i, 0)$ the weight $n \cdot 2^{n-i}$, and to each $val(i, 1)$ the weight 1. In particular, note that, for any i, $val(i, 0)$ has weight higher than all $val(j, 0)$ for $1 \leq j < i$ and all $val(k, 1)$ for $1 \leq k \leq n$ facts together. All the remaining facts are structural and will be forced by the OMQ to be present in every explanation. Therefore we assign the weight 0 to all the remaining facts in the database D.

There are facts in D that will be used to impose the consistency of the truth assignments to the literals:

$$simlit(0, 0), \qquad simlit(1, 1) \qquad opplit(0, 1), \qquad opplit(1, 0),$$

where 0 and 1 are constants with the same meaning as above.

There are facts in D that capture satisfying assignments to literals for a clause $clsat(\alpha_1, \alpha_2, \alpha_3)$, where each $\alpha_i \in \{0, 1\}$ and at least one of α_i for $i \in \{1, 2, 3\}$ is 1. Note that there are 7 such facts for each of the formulas. The predicate $clsat(\cdot, \cdot, \cdot)$ states what assignments to the *literals* (and not to the variables) satisfy a clause.

We take the program Σ to be empty.

The query Q is constituted by the following pieces. The first piece of the query checks that there is at least one *val* fact for each variable of ϕ:

$$val \equiv \bigwedge_{i=1}^{n} val(i, T_i).$$

We require all the structural *clsat*, *simlit*, and *opplit* facts to be in an explanation:

$$config \equiv \bigwedge_{p(\mathbf{c}) \in D_{st}} p(\mathbf{c}).$$

A third piece of the query "copies" the assignment on each variable x_i onto the occurrences of x_i as a positive literal in the formula ϕ. We use the following notation: $l_{j,k}$ is the k^{th} literal in the j^{th} clause, c_j, in ϕ, and $v_{j,k}$ is the variable of $l_{j,k}$.

$$copy \equiv \bigwedge_{i=1}^{n} simlit(T_i, T_{j,k}),$$

where, in each predicate $simlit(T_i, T_{j,k})$, $T_{j,k}$ is a variable for the Boolean value of the literal $l_{j,k} = x_i$ in ϕ. Observe that, in order for *copy* to work properly, each variable x_i must appear as a positive literal in the clauses of ϕ at least once. This can be

assumed without loss of generality, because if x_i always appears as a negative literal in all the clauses of ϕ, then we can replace all the occurrences of the negative literal $\neg x_i$ with the positive literal x_i without altering the satisfiability properties of ϕ.

A fourth piece of the query forces that the ground values assigned to the various variables $T_{j,k}$ simulating the assignments to the literals are consistent. In the notation below, $\ell_{j,k} \sim \ell_{j',k'}$ means that $v_{j,k} = v_{j',k'}$ and the literals $\ell_{j,k}$ and $\ell_{j',k'}$ are both positive or negative, while $\ell_{j,k} \nsim \ell_{j',k'}$ means that $v_{j,k} = v_{j',k'}$ and one literal is positive and the other is negative. We take this part of the query to be

$$\mathit{consist} \equiv \bigwedge_{\ell_{j,k} \sim \ell_{j',k'}} \mathit{simlit}(T_{j,k}, T_{j',k'}) \wedge \bigwedge_{\ell_{j,k} \nsim \ell_{j',k'}} \mathit{opplit}(T_{j,k}, T_{j',k'}),$$

where $T_{j,k}$ is a variable (of the query) representing the value of the literal $\ell_{j,k}$.

The last piece of the query checks $\phi(X, Y)$'s satisfiability:

$$\mathit{satisfied} \equiv \bigwedge_{c_j \in \phi} \mathit{clsat}(T_{j,1}, T_{j,2}, T_{j,3}).$$

To conclude, the query is $Q = \mathit{val}, \mathit{config}, \mathit{copy}, \mathit{consist}, \mathit{satisfied}$.

We show that lexicographically maximum assignment σ_m for ϕ assigns $\sigma_m(x_n) = 1$ iff $\mathit{val}(n, 1)$ is contained in a $\leq_w$-MinEX for $D \models (Q, \Sigma)$.

Note that the *structural* facts have to be in every explanation. Only *val* facts may be removed from D without breaking the entailment. In particular, every MinEX contains exactly one *val* fact as the query for each variable x_i in ϕ contains $\mathit{val}(i, T_i)$. Also, it is easy to see that $\leq_w$-minimal explanations correspond to lexicographically maximum satisfying assignments by the way the weights are assigned to *val* facts. □

The remaining hardness results follow from the proofs of the hardness results for MinEX-Rel$(\leq, \mathrm{UCQ}, \mathcal{L})$ as the cardinality order is a special case of the weight order.

Finally, we cover the complexity results for MinEX-Irrel and MinEX-Nec. First, the membership results follow by similar arguments, as in MinEX-Rel.

Theorem 6.27. *For any language $\mathcal{L}$ of existential rules, if* $\mathrm{OMQA}(\mathrm{UCQ}, \mathcal{L})$ *is in $\mathcal{C}$, then* MinEX-Irrel$(\leq_w, \mathrm{UCQ}, \mathcal{L})$ *and* MinEX-Nec$(\leq_w, \mathrm{UCQ}, \mathcal{L})$ *are in* $\mathrm{P}^{\mathrm{NP}^{\mathcal{C}}}$.

This result implies all the upper bounds in Table 6.2 except for the Δ_2^{P}-membership results in *fp*- and *ba*-combined complexity and the tractability results in data complexity for FO-rewritable languages. These results follow by slight variations of the argument as for MinEX-Rel.

Note that in the proofs of Theorems 6.25 and 6.26, a $\leq_w$-MinEX is unique. Therefore, these results also apply to MinEX-Nec. As in the case of cardinality-minimal explanations, MinEX-Nec($\leq_w$, UCQ, $\mathcal{L}$) can be reduced to the complement of MinEX-Irrel($\leq_w$, UCQ, $\mathcal{L}$). Since all the hardness results for MinEX-Nec are for deterministic classes, they immediately apply to MinEX-Irrel.

6.5 Overview of Complexity Results

In this chapter, we studied *cardinality-* and *weight-minimal* explanations for OMQA under existential rules. For these two preference orders, we studied five problems, already introduced in earlier chapters, namely, Is-MinEX, All-MinEX, MinEX-Rel, MinEX-Irrel and MinEX-Nec. We have conducted a detailed complexity analysis for each of these problems, 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, presented in Chapter 3, and require different techniques and reductions.

The complexity landscape is very different for preferred explanations, compared to inclusion-minimal explanations. As one might expect, no computational problems become easier when we seek preferred explanations, rather than inclusion-minimal ones. This can be intuitively seen by observing that every cardinality- and weight-minimal explanation is also an inclusion-minimal explanation. So checking cardinality- and weight-minimality can be seen as an extra condition to be checked. In our complexity analysis of the problems associated with preferred explanations, we have shown that the results for Is-MinEX and All-MinEX coincide for both cardinality- and weight-minimal explanations. The results for the three other problems, namely, MinEX-Rel, MinEX-Irrel and MinEX-Nec, coincide for cardinality-minimal explanations. Furthermore, the results for these three problems coincide with the ones for weight-minimal explanations. This is surprising, given that in Chapter 3, we had different complexity results for all of these three problems MinEX-Rel, MinEX-Irrel and MinEX-Nec.

First, we note why the results for Is-MinEX and All-MinEX coincide. The primary reason is that checking the minimality of a single set is not easier than ensuring the minimality of multiple subsets w.r.t. the preference orders studied. For Is-MinEX, to ensure that a given subset of the database is a preferred explanation we need to ensure that there is no explanation that is *strictly* more preferred w.r.t.

that order. Similarly, given a set of subsets of the database, ensuring minimality of every subset w.r.t. some preference order and checking that there is no MinEX outside of this set is equivalent to ensuring that there is no explanation outside of the candidate set that is *no less* preferred than all the subsets in the candidate set. Both of these require guessing a subset of the database and then checking the entailment. Therefore the complexity of both problems coincide both for cardinality and weight orders.

For other problems, the preference order has an impact on the complexity. For cardinality-minimality, we see that MinEX-Rel, MinEX-Irrel and MinEX-Nec all have the same complexity. Similarly, for weight-minimality, the complexity results for these three problems coincide. This is rather surprising, compared to inclusion-minimal explanations, where MinEX-Irrel was significantly easier than MinEX-Rel in many cases. One of the reasons why this is so is that now for MinEX-Irrel, it is not enough to ensure that some not necessarily minimal explanation exists, we need to ensure that such subset is not only subset-minimal but also cardinality/weight-minimal. For cardinality-minimal explanations, I prove that these problems are Θ_2^{P}-complete already in data complexity for guarded full existential rules. This result holds as the size of a smallest MinEX can be computed by a binary search on the size of the database. On the other hand, for the weight-minimality case, these problems are Δ_2^{P}-complete already in data complexity for guarded full existential rules. This increase in complexity occurs because we need to carry out a binary search on the total sum of weights, which might be exponential in the size of the input.

Chapter 7

Related Work

In this section, we review the existing literature on explanations for logical formalisms and present the larger context to our work. From a broader perspective, our work studies *abductive reasoning*, since we seek to find the causes for the observations. Abductive reasoning has been studied for several formalisms, such as propositional logic [71], logic programs [72], default theories [73], probabilistic temporal logic [121], and DLs [144, 135]. Abstracting away from subtle differences, the study of explanations can be classified in accordance to the underlying *logical formalism* (e.g., propositional logic, description logics and existential rules) and the *reasoning task* to be explained (e.g., query answering, and consistency checking). We show how our work relates to, generalises, and differs from the existing approaches in this area.

7.1 Historical Perspective on Logic-based Abduction

Abductive reasoning has been studied in the AI community since the 1970s [138, 130, 55] with applications in fault diagnosis, belief revision, automated planning, and others. A number of models for abduction have been studied. As common in such works, H denotes the set of hypotheses, M the observed manifestations and T denotes the domain knowledge. In a *set-cover-based* approach [33], each subset of the hypotheses H covers some subset of the manifestations M. The objective of this approach is to find a subset of H that covers all of the manifestations M. A *probabilistic* approach [136] models hypotheses H and manifestations M as events, and T contains probabilistic knowledge about H and M. A solution to a probabilistic abduction problem is a subset $X \subseteq H$ such that $P(X \mid M)$ is maximised. Other two

approaches are *consistency-based diagnosis* [140], and *propositional abduction* [62, 63, 64, 71], which we discuss in more detail in turn.

The study of explanations and diagnosis in logical formalisms dates back to the work of Reiter in 1987 [140]. He studied the consistency-based diagnosis of explaining the discrepancy between the observed and the correct system behaviour by employing first-order logic. According to this approach, the hypotheses in H represent the components of the system to be diagnosed, and the observed manifestations are captured in M. The functioning of the system is described via a set of first-order sentences T, known as the *system description*. The system description T involves literals of the form $\neg AB(c)$ expressing that a component c functions properly, that is, not abnormally. A *diagnosis* is a minimal set A of components such that $T \cup M \cup \{\neg AB(x) \mid x \in H \setminus A\} \cup \{AB(x) \mid x \in A\}$ is consistent. This approach to diagnosis is practical when we have a good description of the proper functioning of the system, but little knowledge or experience about the system behaviour in case of failures.

Another early relevant approach is *propositional abduction* and the work by Eiter and Gottlob [71], which analyses the complexity of problems associated with *propositional abduction problem (PAP)*. In this approach, a propositional theory T formalises a particular application domain, a set M of atomic formulas describes the observed manifestations, and a set H of formulas contains possible hypotheses. Then, an *explanation* for M is a subset of hypotheses $S \subseteq H$ such that $T \cup S$ is consistent and logically entails M. This work contains the complexity analysis of a number of decision problems, including deciding if an explanation exists, if a particular hypothesis is relevant, and if a particular hypothesis is necessary. These problems have inspired some of the problems studied in this thesis, like MinEX-Rel and MinEX-Nec. Further, this work studied a number of different minimality criterion, including, subset-minimality, minimum cardinality and weight-minimality (penalisation). This framework is subsumed by our approach, as we can capture hypotheses in the database (or the set of hypotheses in the negative case), manifestations in the query, and the theory in the program. Furthermore, the complexity of the logic-based abduction problems for the $\mathcal{EL}$ family of description logics was studied by Bienvenu [20].

7.2 Provenance

Provenance for database queries has been widely studied in the database community [31] and later was adopted for ontology-based data access [45, 30]. The notions

of provenance can be divided into the three main categories, namely, why-, how- and where-provenance. The *why*-provenance provides a witness, a subset of the database, for a query [60, 32], the *how*-provenance is more general than the why-provenance as it also provides *how* the answer is derived from a witness [86], and the *where*-provenance captures where the output data came from in the input database [154, 32]. Provenance gained more attention when the connection to semirings, known as *provenence semirings* [86, 85], was discovered, which provided an abstract algebraic tool to record and track provenance information.

The classical database provenance has recently been extended for ontology-based data access to capture why-provenance [45, 30]. Calvanese et al., taking inspiration from database theory, enriched ontology-based data access with provenance semirings. To ensure that provenance polynomials are finite they considered idempotent semirings with which they studied *DL-Lite*$_{\mathcal{R}}$ ontologies [45]. This was followed by a work by Bourgaux et al. [30], where they adapted the provenance setting of Calvanese et al. for the $\mathcal{ELH}^r$ variant of $\mathcal{EL}$. The main difference between our approach and the work on provenance is that, in our work, we seek *minimal* subsets of the database that support the query entailment, while, in the context of provenance, all possible explanations are relevant, irrespective of whether they are minimal or not. This makes our approach and techniques significantly different when compared with the work on provenance.

7.3 Explaining Ontological Reasoning Tasks

In the context of ontologies, explanations have been first studied to understand the consequences of an ontology. This line of research has been motivated by the fact that devising ontologies is an error-prone task, where small changes in the ontology might have unexpected consequences. Such research has primarily used DLs to express ontologies. The earliest works considered formal proofs as explanations and the later ones considered subsets of the TBox as explanations. Providing formal proofs for concept subsumption was first considered by McGuinness and Borgida [120] and Borgida et al. [25], who provided algorithms to obtain such proofs based on different deductive calculi. This approach greatly differs from ours, as we abstract away from a particular proof by considering subsets that support the entailment.

A subsequent line of research focused on *axiom pinpointing* [144, 101, 97]. In axiom pinpointing, explanations are taken to be minimal subsets of the ontology (TBox) that yield a given consequence. Such explanations are called *justifications* [94, 95],

or *MinAs*, which stands for "minimal axiom set" [9, 135]. The earliest works in axiom pinpointing focused on explaining why a concept is *unsatisfiable*. The first such work, to our knowledge, was by Schlobach and Cornet [144], which was inspired by medical applications. The authors provided an algorithm to pinpoint the axioms in an unfoldable $\mathcal{ALC}$ TBox that cause unsatisfiability of a concept by extending the standard tableau algorithm [145] with labels. Later, this problem was addressed in the Semantic Web setting for repairing unsatisfiable concepts in OWL ontologies [102], which are based on the expressive DL $\mathcal{SHOIN}(\mathcal{D})$. Also, justifications have been considered to explain inconsistency [95]. Various types of justifications have been considered depending on different notions of semantic minimality [94]. Later works in axiom pinpointing mainly focused on a more general task of explaining any concept subsumption (GCI) [9, 95, 135]. Axiom pinpointing has been extensively studied for the Semantic Web ontologies, OWL [103, 94, 149], and for the SNOMED CT medical ontology [9]. Also, there are some systems developed for computing justifications [103, 146].

The work by Peñaloza and Sertkaya [135] is closely related to ours. It analyses the complexity of decision, counting, and enumeration problems, associated with axiom pinpointing by incorporating and extending many of their previous works [132, 133, 134]. The authors give a nearly complete picture of the complexity of axiom pinpointing for lightweight DLs. Our work [48, 50] was inspired by this study. In particular, they studied the problems of recognising a MinA, IS-MINA, recognising the set of all MinAs, ALL-MINA, deciding if a particular axiom is relevant for explaining, MINA-RELEVANCE, deciding if there is an explanation that avoids all of the forbidden sets of facts, MINA-IRRELEVANCE. By adapting these problems, we have introduced the problems IS-MINEX, ALL-MINEX, MINEX-REL and MINEX-IRREL, when studying explanations for OMQ answers. Further, some of the proofs in this thesis, such as Theorems 3.24 and 3.28, have employed the ideas from this work. We further note that for more expressive description logics, the complexity of the problems associated with axiom pinpointing were considered as well [131].

Some other works on axiom pinpointing focused on analysing the problems associated with computing a pinpointing formula [7, 8] that provides an encoding of all MinAs. However, it is known that extracting the MinAs from such representations is already a hard problem. Beside computing justifications efficiently, several works addressed the problem of making them understandable for the user, either by studying their cognitive complexity [96], or by grouping justifications that have a similar structure to help to handle a large number of justifications [14].

7.4 Explaining Positive Query Answers

We review the research that focuses on explanations for positive query answers. The literature on explaining positive query answers is rather sparse. This motivated our analysis of explanations for positive query answers. There have been some works on explanations for ontology-mediated query answering under standard and inconsistency-tolerant semantics, which we review in the rest of this section. We also note that the problem of explaining query results has been studied in the database community as well [56].

To our knowledge, prior to our work, explanations for ontology-mediated query answering under standard semantics have been only studied for the *DL-Lite* family of languages [26]. This work provides formal proofs as explanations for query answers and thus differs from our approach. The first work that adapts the ideas from logic-based abduction and axiom pinpointing to explaining ontology-mediated query entailment was provided by us [48], which provided the complexity analysis of explanations problems under existential rules. We followed up with another work focusing on explaining OMQ answers in description logics [50]. To our knowledge, the only other work on explaining positive query entailment under existential rules preceding our work is relative to *probabilistic databases* and hence of a very different flavor [47].

7.5 Explaining Negative Query Answers

In this section, we review the existing work on explaining non-entailment of a query. It has been studied for DLs under the name of *ABox aduction* [43, 107, 67, 68] and in the context of databases [91]. We observe that, to our knowledge, our work [49] has been the only work thus far on negative explanations under existential rules.

The closest work to ours is by Calvanese et al. [41, 43]. In this work, as is common in the ABox abduction research, they start with a set of abducible *predicates*, called *signature*. Then, given a query that is *not* entailed from the knowledge base, the idea is to find a set of assertions such that, when they are added to the ABox, the entailment holds (avoiding inconsistencies). This formulation is suitable when abducible predicates are known, but relevant constants may not be known. They analysed the computational complexity of the problems associated with negative explanations in the lightweight DL *DL-Lite*$_{core}$. They analyse the problems of recognising an explanation, deciding if an explanation exists, if an assertion is relevant and if an assertion

is necessary. They consider arbitrary, inclusion-minimal, and cardinality-minimal explanations. We analyse these problems under existential rules in addition to the novel explanation problems such as MinEX-AllRel$^{\not\models}$ and MinEX-AllNec$^{\not\models}$. One important difference is in the way abducibles are defined. In our case, explanations are subsets of a given finite set H of abducible *facts*, unlike in the work by Calvanese et al. [43] as noted above. We also note that Du et al. [68] introduced a special kind of minimal explanations, called *representative* explanations, from which all minimal explanations can be retrieved.

Explanations for negative query answers have been considered in the database context as minimal sets to be added to support the entailment [91]. Also, in the database community, different approaches have been taken of finding subqueries, which are responsible in pruning a missing answer [19].

7.6 Explaining Query Answers under Inconsistency-tolerant Semantics

Explaining query answers under inconsistency-tolerant semantics has been studied both for description logics [23, 69, 24] and existential rules [88, 116]. We review such works in this section.The work that is most similar to ours is by Bienvenu et al. [24], which studies explanations for both positive and negative query answers over inconsistent *DL-Lite*$_{core}$ knowledge bases. This work defines explanations as subsets of the ABox for positive and negative query answers under brave, AR, and IAR semantics. They study the problems of recognising a minimal explanation, deciding if a particular assertion is relevant, and if a particular assertion is necessary. Despite the fact that the work considers similar problems, the fact that a different semantics is considered makes this work and the analysis in this work substantially different. We further note that explanations for query answers under the inconsistency-tolerant semantics ICR was also considered [4], where argumentation framework was used for explaining. Another argumentation framework was provided for BCQ explanations under IAR and brave semantics [3], which was implemented in the DALEK prototype [5]. Further, empirical evaluation was provided for assessing different methods for computing explanations under ICR semantics [88]. Also, preferred explanations for negative query answers were considered under the inconsistency-tolerant semantics IAR [69]. In this work, the preference order, given by cardinality-preserving substitutions, were considered. Different works have focused on the efficient computation of explanations, e.g., see [67, 68, 155].

Chapter 8

Conclusions

In this chapter, I summarise this work and indicate future research directions. I give a brief overview of what has been achieved in this thesis, outlining the main contributions and the results of interest. I also indicate what future work I foresee in this line of research.

8.1 Summary

In this thesis, I have studied explanations for ontology-mediated query answers and the associated computational problems. I have provided a comprehensive complexity analysis of these computational problems for a wealth of different description logics and fragments of existential rules. I have considered these problems for different query languages, such as instance queries, Boolean conjunctive queries and unions of conjunctive queries, and different complexity measures, such as data, *fp*-combined, *ba*-combined, and combined complexity.

Before this work, there was a substantial amount of research on explaining ontological reasoning, however, explaining ontology-mediated query answers under the classical semantics has received very little attention. In this work, I have contributed to closing this gap. I have adapted the ideas from a common approach for explaining ontological reasoning, known as axiom pinpointing, to introduce the notion of a minimal explanation, MinEX, for an ontology-mediated query answer. For MinEXs, I studied the associated computational problems.

Below I discuss the results and the contributions of each chapter.

Chapter 3: In this chapter, I have analysed explanations for positive ontology-mediated query answers under existential rules. The problems studied include recognising a MinEX (Is-MinEX), recognising the set of all MinEXs (All-MinEX), deciding if a distinguished fact is contained in some MinEX (MinEX-Rel), deciding if

there is a MinEX that does not contain any of the forbidden sets (MinEX-Irrel), and deciding if there is a MinEX smaller (resp., larger) than some given threshold number (Small-MinEX (resp., Large-MinEX)). These problems are adaptations of the problems, studied in axiom pinpointing [135].

I point out some of the complexity results for these problems, which range from membership in AC^0 to 2Exp-completeness. First, observe that for the FO-rewritable description logics studied, namely, L, A, S and their full restrictions, all the problems are tractable in data complexity since the set of all MinEXs can be computed in polynomial time. Interestingly, the complexity of Is-MinEX in data complexity does not increase, compared to OMQA, in all the cases studied. The reason for this is that, to ensure minimality, it is enough to check that removing any single element breaks the entailment, and so a linear number of non-entailment checks suffices. I show that the problem All-MinEX is intractable already in data complexity for the guarded full fragment of existential rules. The primary reason for this is that we need to ensure that there is *no* MinEX outside our candidate set of all MinEXs. Surprisingly, in *fp*-combined, *ba*-combined, and combined complexity measures, the complexity results for Is-MinEX and All-MinEX coincide. This holds primarily because, for All-MinEX, checking minimality of each set and checking that there is no MinEX outside of the input set of subsets can be combined as a single check. In particular, both problems are harder compared to OMQA in some cases. That is, they are D^{P}-complete in *fp*-combined complexity, compared to the NP-completeness of OMQA, and are $\mathrm{D}^{\mathrm{Exp}}$-complete in *ba*-combined complexity for the acyclic existential rules, A, compared to the NExp-completeness of OMQA.

For MinEX-Irrel and Small-MinEX, I have proven that these problems are intractable in data complexity for guarded full existential rules, namely, they are NP-complete in this case, compared to the P-completeness of OMQA. On the other hand, surprisingly, in *fp*-combined, *ba*-combined, and combined complexity measures, these problems have the same complexity as OMQA, and thus are generally easier than Is-MinEX and All-MinEX. This is primarily because we do not need to check minimality of a guessed subset – if there is an explanation that does not contain any of the forbidden sets or is smaller than some given threshold number, then there is a minimal one as well with that property.

Finally, for MinEX-Rel and Large-MinEX, I have shown intractability in data complexity for the guarded full existential rules and the Σ_2^{P}-completeness in *fp*-combined complexity for any language considered. This hardness stems from the fact that we need to guess a subset that satisfies the conditions and then ensure that

it is minimal. For example, for Is-MinEX, we did not need to guess a subset, just to ensure that it is a MinEX, and for MinEX-Irrel, we had to guess a subset satisfying a specified property and check the entailment, but we did not need to check minimality. For MinEX-Rel and Large-MinEX, we need to guess a subset and check that it is a MinEX, which makes these two problems the hardest.

Chapter 4: In this chapter, I have studied the complexity of explaining ontology-mediated query answers in description logics. This chapter can be seen as an analogous study to Chapter 3 but for description logics, rather than existential rules. I have considered a wide range of different description logics, two query languages (IQs and UCQs) and two complexity measures (data and combined complexity). Due to the different nature of the languages, the complexity landscape for description logics is rather different compared to the results for existential rules.

For the problems studied, the results range from the membership in AC^0 to the 2Exp-completeness. First, I observe that for the FO-rewritable description logics studied, namely, *DL-Lite*$_{core}$ and *DL-Lite*$_{\mathcal{R}}$, all of the problems are tractable in data complexity. The reason for this is that we can compute the set of all MinEXs in polynomial time in this case. For Is-MinEX, I prove the D^{P}-completeness in data complexity for instances queries and the description logic $\mathcal{ALC}$, which is an increase in the complexity, compared to the coNP-hardness of OMQA. This is in contrast to existential rules, where Is-MinEX is no harder than OMQA in data complexity. Interestingly, All-MinEX is already Π_2^{P}-complete in data complexity for instance queries and the DL $\mathcal{ALC}$, which contrasts with existential rules too, where All-MinEX remains in the first level of polynomial hierarchy in data complexity. Also, in combined complexity the results for Is-MinEX and All-MinEX coincide for the same reasons as in the case of existential rules – for All-MinEX, checking minimality and checking that there is no MinEX outside our candidate set can be combined.

I also show that MinEX-Irrel is already in the second level of the polynomial hierarchy, namely, Σ_2^{P}-complete, in data complexity for the DL $\mathcal{ALC}$, while it stays in the same complexity classes as OMQA in combined complexity for all the description logics considered. For MinEX-Rel, I prove that the complexity results in data complexity coincide with the results for MinEX-Irrel. On the other hand, in combined complexity for UCQ and the description logics *DL-Lite*$_{core}$ and $\mathcal{EL}$, I show that MinEX-Rel is Σ_2^{P}-complete, while OMQA and MinEX-Irrel are NP-complete in those cases. Therefore, MinEX-Rel is the hardness problem that was studied in this chapter.

Chapter 5: In this chapter, I have studied explanations for negative ontology-mediated query answers under existential rules. Since the database does *not* entail an OMQ, in this case, MinEXs are minimal subsets of a finite set of *hypotheses*, which, when added to the database, entail the OMQ. Some problems that I study are adaptations of the problems considered in the positive case, such as Is-MinEX$^{\not\models}$ and MinEX-Rel$^{\not\models}$. I analyse other problems as well. One such problem is deciding whether a MinEX for a negative query answer exists, MinEX-Exists$^{\not\models}$, and another one is deciding if a distinguished fact is contained in every MinEX, MinEX-Nec$^{\not\models}$. Observe that MinEX-Exists$^{\not\models}$ is not applicable in the positive case as the existence of a MinEX is always guaranteed. I introduce two novel computational problems MinEX-AllRel$^{\not\models}$ and MinEX-AllNec$^{\not\models}$, which asks whether a given set of facts contains all the relevant and all the necessary facts for explaining a negative query answer, respectively.

I give a precise picture of the complexity for all the negative explanation problems, which range from membership in P to 2Exp-completeness. First of all, notice that the results for Is-MinEX$^{\not\models}$ coincide with the results for Is-MinEX, presented in Chapter 3. Observe that for Is-MinEX$^{\not\models}$, in addition to checking that a candidate MinEX entails the OMQ and is minimal, we need to check that E together with the database is consistent with the ontology. But this consistency check turns out not to increase the complexity of Is-MinEX$^{\not\models}$, compared to Is-MinEX, as it takes at most linear number of non-entailment checks. Also, I reduce Is-MinEX to Is-MinEX$^{\not\models}$, and so the hardness results follow from the analysis in Chapter 3.

I show that MinEX-Rel$^{\not\models}$ and MinEX-Nec$^{\not\models}$ are complete for the complementary complexity classes, except in *fp*-combined complexity, when the problem MinEX-Rel$^{\not\models}$ is Σ_2^{p}-complete, while MinEX-Nec$^{\not\models}$ is coNP-complete. Intuitively, the reason is that to decide whether a fact ψ is *not* necessary, we can guess a subset E of H and check whether E is an explanation that does not include ψ, and verifying minimality of E can be avoided. Checking that a fact ψ is relevant also requires guessing a subset E of H and checking whether E is an explanation, but the minimality check can be avoided. This also explains the difference between MinEX-AllRel$^{\not\models}$ and MinEX-AllNec$^{\not\models}$ in *fp*-combined complexity.

I show that the complexity of some of the problems depends not only on the complexity of OMQA. For the linear, full, and sticky languages, as well as for their sublanguages, OMQA is NP-complete both in *fp*-combined and in *ba*-combined complexity measures. One may expect that, for a fixed problem, the complexity does not change in such cases, as the complexity of OMQA remains the same

across them. However, this is not the case for MinEX-Exists$^{\not\models}$, MinEX-Nec$^{\not\models}$, and MinEX-AllNec$^{\not\models}$, where the complexity goes from NP-, coNP-, and D^{p}-completeness in *fp*-combined complexity to Σ_2^{p}-, Π_2^{p}-, and $\mathrm{D}_2^{\mathrm{p}}$-completeness in *ba*-combined complexity, respectively. The reason is that in *fp*-combined complexity, consistency checking can be reduced to OMQA in data complexity since for such a check the query is fixed too. In contrast, in *ba*-combined complexity, the program is not fixed anymore, and checking consistency requires a linear number of checks for *non*-entailment, whose cost increases the overall complexity by one level.

Chapter 6: In this chapter, I have analysed the complexity of problems associated with computing preferred minimal explanations for ontology-mediated query answers under existential rules. This chapter can be seen as an analogous study to the one presented in Chapter 3; instead of considering inclusion-minimal explanations, I consider cardinality-minimal and weight-minimal explanations.

For Is-MinEX and All-MinEX, both for cardinality and weight orders, I prove that the results coincide. Interestingly, in this case, All-MinEX has the same complexity as in the inclusion-minimal case. On the other hand, I show the increase in complexity for Is-MinEX as checking inclusion-minimality of a specific explanation is not sufficient, we need to ensure that there is no other MinEX with smaller cardinality/weight. For other problems, there are more stark differences between different preference orders. For cardinality-minimality, I show that MinEX-Rel, MinEX-Irrel and MinEX-Nec all have the same complexity. Similarly, for weight-minimality, the complexity results for these three problems coincide. This is rather surprising, compared to inclusion-minimal explanations, where MinEX-Irrel was significantly easier than MinEX-Rel in many cases. One of the reasons why this is so is that now for MinEX-Irrel, it is not enough to ensure that some not necessarily minimal explanation exists, we need to ensure that such subset is not only subset-minimal but also cardinality/weight-minimal. For cardinality-minimal explanations, I prove that these problems are Θ_2^{p}-complete already in data complexity for guarded full existential rules. This result holds as the size of a smallest MinEX can be computed by a binary search on the size of the database. On the other hand, for the weight-minimality case, these problems are Δ_2^{p}-complete already in data complexity for guarded full existential rules. This increase in complexity occurs because we need to carry out a binary search on the total sum of weights, which might be exponential in the size of the input.

8.2 Future Work

In this section, I cover some of the ways that this work can be extended. The first direction to extend this work is to provide the corresponding analysis for description logics to the analysis obtained for existential rules. The second way is to study all the introduced problems for different preference orders. The third direction is to study different types of computational problems, such as counting and enumeration problems. Finally, the open problems left by this work should be addressed.

One certain way to extend this work is to complete the analysis of explanation problems for description logics. In this work, I have analysed explaining OMQA for existential rules more extensively than for description logics. I have analysed only explaining positive ontology-mediated query answers both for existential rules and description logics in Chapters 3 and 4, respectively. The studies for description logics, corresponding to Chapters 5 and 6, are still to be carried out. This thesis shows that such works would be an important contribution to the study of explaining ontology-mediated query answering. In particular, one can see that there is a significant difference between the results and proof techniques for the same problems for existential rules in Chapter 3, and description logics in Chapter 4. These differences stem from the different expressivity capabilities of these two ontological languages. Existential rules allow predicates of unbounded arity, while description logics allows only unary and binary predicates. On the other hand, some description logics allow disjunction in the rule heads and negation. For example, due to these differences, query answering for all the existential rules studied in this work remains tractable in data complexity, while it is coNP-complete for the description logic $\mathcal{ALC}$ in data complexity.

Another direction of extending this work is considering different computational problems, including counting and enumeration problems. In axiom pinpointing, an approach that inspired ours, both counting and enumeration problems have been considered [135]. Some of the counting problems that would be of interest in this setting include counting the number of all minimal explanations and counting the number of explanations that contain a distinguished (relevant) fact. The enumeration problems of particular interest would be the problem of enumerating all MinEXs as well as enumerating all the minimal explanations according to some specified order. Enumerating explanations in a specific order is particularly relevant when we would like to terminate the procedure after enough explanations have been provided and we know that the remaining explanations will be less preferred.

Notice that counting minimal explanations is closely related to probabilistic query evaluation with ontologies. OMQA over probabilistic data has been studied both in the context of description logics [100, 46] and existential rules [28, 27]. Recently, a dichotomy result has been obtained for probabilistic query evaluation for the whole class of homomorphism-closed queries over probabilistic graphs [2]. Notice that homomorphism-closed queries contain a large class of ontology-mediated queries. It is an interesting research direction to see if these results have implications on the problems like counting the number of minimal explanations.

Another important future research direction is analysing explanation problems under different preference orders. In this thesis, I have dedicated one chapter to studying the problems associated with computing preferred explanations for positive query answers under existential rules. I have considered cardinality-minimality and weight-minimality as minimality criterion. As we have seen, preferred explanations are often desirable when we have extra knowledge about the data and we can rank the reliability of the facts. The cases for which the analysis still should be carried out include the problems associated with explaining positive query answers for DLs and the problems associated with explaining negative query answers both for existential rules and description logics. In this work, I have seen an increase in the computational complexity of the problems studied when considered under different minimality criterion. It is important to gain a complete understanding of the complexity of computing preferred explanations for different settings as this would help users and developers to make informed decisions about using and developing such systems. Further, it would be interesting to study more preference orders such as *priority levels*, which was studied in the context of propositional abduction [71].

A further direction for extending this work is to classify our problems in a more fine-grained way. There has been much progress in understanding complexity of OMQA. In particular, a trichotomy result was obtained for OMQs based on the description logic $\mathcal{EL}$ [118] and a dichotomy result for OMQs based on a number of variants of guarded fragment of first-order logic [90]. It would be interesting to investigate this direction further. One potential direction could be investigating whether a dichotomy result for All-MinEX under existential rules in data complexity can be obtained. In this thesis, we have seen that this problem is either in P or in coNP in data complexity for the languages of existential rules studied. It would be interesting to see if such a dichotomy holds. Also, I would like to find if dichotomy or trichotomy results hold for other problems too.

The last direction in which my work should be extended is closing the open problems specified in this thesis. In particular, I have shown that the problem of recognising a minimal explanation is in AC^0 in data complexity when query answering is FO-rewritable. For all the other problems, in this case, they are in P as the set of all MinEXs can be computed in polynomial time but it is not known whether they are in AC^0 or P-hard. My intuition suggests that the latter is more likely to be the case but it is still to be shown. Although in Theorem 4.10, we have shown that Is-MinEX$(\subseteq, \text{IQ}, \mathcal{ELI})$ is P-hard, somewhat surprisingly, it is unclear whether Is-MinEX$(\subseteq, \text{IQ}, \mathcal{EL})$ is also P-hard in data complexity, and so it is open whether Theorem 4.10 can be strengthened in this direction. Another open problem is showing the P^{NExp}-hardness of MinEX-Rel, MinEX-Irrel, and MinEX-Nec for the cardinality order in the acyclic fragment of existential rules in *ba*-combined complexity.

Bibliography

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases: The Logical Level*. Addison-Wesley Longman Publishing Co., 1st edition, 1995.

[2] A. Amarilli and İ. İ. Ceylan. A dichotomy for homomorphism-closed queries on probabilistic graphs. In *Proceedings of the 23rd International Conference on Database Theory, (ICDT-20)*, volume 155 of *LIPIcs*, pages 5:1–5:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[3] A. Arioua and M. Croitoru. Dialectical characterization of consistent query explanation with existential rules. In *Proceedings of the 29th International Florida Artificial Intelligence Research Society Conference (FLAIRS-16)*, pages 621–625, 2016.

[4] A. Arioua, N. Tamani, and M. Croitoru. Query answering explanation in inconsistent Datalog+/– knowledge bases. In *Proceedings of the 26th International Conference on Database and Expert Systems Applications (DEXA-15)*, pages 203–219, 2015.

[5] A. Arioua, M. Croitoru, and P. Buche. DALEK: A tool for dialectical explanations in inconsistent knowledge bases. In *Proceedings of the 6th International Conference on Computational Models of Argument. (COMMA-16)*, volume 287, pages 461–462, 2016.

[6] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *Journal of Artificial Intelligence Research*, 36:1–69, 2009.

[7] F. Baader and R. Peñaloza. Automata-based axiom pinpointing. *Journal of Automated Reasoning*, 45(2):91–129, 2010.

[8] F. Baader and R. Peñaloza. Axiom pinpointing in general tableaux. *Journal of Logic and Computation*, 20(1):5–34, 2010.

[9] F. Baader and B. Suntisrivaraporn. Debugging SNOMED CT using axiom pinpointing in the description logic $\mathcal{EL}^+$. In *Proceedings of the 3rd International Conference on Knowledge Representation in Medicine (KR-MED-08)*, volume 410 of *CEUR Workshop Proceedings*, 2008.

[10] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 364–369, 2005.

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

[12] F. Baader, R. Peñaloza, and B. Suntisrivaraporn. Pinpointing in the description logic $\mathcal{EL}^+$. In *Proceedings of the 30th Annual German Conference on Artificial Intelligence (KI-07)*, pages 52–67, 2007.

[13] J. Baget, M. Leclère, M. Mugnier, and E. Salvat. On rules with existential variables: Walking the decidability line. *Artificial Intelligence*, 175(9-10):1620–1654, 2011.

[14] S. Bail, B. Parsia, and U. Sattler. The logical diversity of explanations in OWL ontologies. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM-13)*, pages 559–568, 2013.

[15] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82 – 115, 2020. ISSN 1566-2535.

[16] C. Beeri and M. Y. Vardi. The implication problem for data dependencies. In *Proceedings of the 8th International Colloquium on Automata, Languages, and Programming (ICALP-81)*, volume 115 of *LNCS*, pages 73–85. Springer-Verlag, 1981.

[17] L. Bellomarini, E. Sallinger, and G. Gottlob. The Vadalog System: Datalog-based reasoning for knowledge graphs. *Proceedings of the VLDB Endowment*, 11(9):975–987, 2018. ISSN 2150-8097.

[18] P. Besnard, T. Schaub, H. Tompits, and S. Woltran. Representing paraconsistent reasoning via quantified propositional logic. In L. E. Bertossi, A. Hunter, and T. Schaub, editors, *Inconsistency Tolerance [result from a Dagstuhl seminar]*, volume 3300 of *Lecture Notes in Computer Science*, pages 84–118.

[19] N. Bidoit, M. Herschel, and K. Tzompanaki. Query-based why-not provenance with nedexplain. In *Proceedings of the 17th International Conference on Extending Database Technology (EDBT-14)*, pages 145–156, 2014.

[20] M. Bienvenu. Complexity of abduction in the $\mathcal{EL}$ family of lightweight description logics. In *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR-08)*, page 220–230. AAAI Press, 2008.

[21] M. Bienvenu, C. Bourgaux, and F. Goasdoué. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI-14)*, pages 996–1002, 2014.

[22] M. Bienvenu, B. T. Cate, C. Lutz, and F. Wolter. Ontology-based data access: A study through disjunctive Datalog, CSP, and MMSNP. *ACM Transactions on Database Systems (TODS)*, 39(4):33:1–33:44, 2014.

[23] M. Bienvenu, C. Bourgaux, and F. Goasdoué. Explaining inconsistency-tolerant query answering over description logic knowledge bases. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 900–906, 2016.

[24] M. Bienvenu, C. Bourgaux, and F. Goasdoué. Computing and explaining query answers over inconsistent *DL-Lite* knowledge bases. *Journal of Artificial Intelligence Research*, 64:563–644, 2019.

[25] A. Borgida, E. Franconi, and I. Horrocks. Explaining $\mathcal{ALC}$ subsumption. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-00)*, pages 209–213, 2000.

[26] A. Borgida, D. Calvanese, and M. Rodriguez-Muro. Explanation in the *DL-Lite* family of description logics. In *Proceedings of the 2008 OTM Confederated International Conference "On the Move to Meaningful Internet Systems (OTM-08)*, pages 1440–1457, 2008.

[27] S. Borgwardt, İ. İ. Ceylan, and T. Lukasiewicz. Ontology-mediated queries for probabilistic databases. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI-17)*, pages 1063–1069. AAAI Press, 2017.

[28] S. Borgwardt, İ. İ. Ceylan, and T. Lukasiewicz. Ontology-mediated query answering over log-linear probabilistic data. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, pages 2711–2718. AAAI Press, 2019.

[29] E. Boros, V. Gurvich, L. Khachiyan, and K. Makino. On maximal frequent and minimal infrequent sets in binary matrices. *Annals of Mathematics and Artificial Intelligence*, 39(3):211–221, 2003.

[30] C. Bourgaux, A. Ozaki, R. Peñaloza, and L. Predoiu. Provenance for the description logic $\mathcal{ELH}^r$. In C. Bessiere, editor, *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI-20)*, pages 1862–1869, 2020.

[31] P. Buneman. The providence of provenance. In G. Gottlob, G. Grasso, D. Olteanu, and C. Schallhart, editors, *Big Data*, pages 7–12, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[32] P. Buneman, S. Khanna, and W. C. Tan. Why and where: A characterization of data provenance. In *Proceedings of the 8th International Conference on Database Theory (ICDT-01)*, volume 1973 of *Lecture Notes in Computer Science*, pages 316–330. Springer, 2001.

[33] T. Bylander, D. Allemang, M. C. Tanner, and J. R. Josephson. The computational complexity of abduction. *Artificial Intelligence*, 49(1-3):25–60, 1991.

[34] K. Cagle. Why 'ontology' will be a big word in your company's future. https://www.forbes.com/sites/cognitiveworld/2018/07/20/why-ontology-will-be-a-big-word-in-your-companys-future/, 2018. Accessed: 2020-11-02.

[35] A. Calì, G. Gottlob, and A. Pieris. Advanced processing for ontological queries. *VLDB Endowment*, 3(1):554–565, 2010.

[36] A. Calì, G. Gottlob, and A. Pieris. Query answering under non-guarded rules in Datalog+/-. In *Proceedings of the 4th International Conference on Web Reasoning and Rule Systems (RR-10)*, volume 6333 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2010.

[37] A. Calì, G. Gottlob, and T. Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *Journal of Web Semantics*, 14: 57–83, 2012.

[38] A. Calì, G. Gottlob, and A. Pieris. Towards more expressive ontology languages: The query answering problem. *Artificial Intelligence*, 193:87–128, 2012.

[39] A. Calì, G. Gottlob, and M. Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *Journal of Automated Reasoning*, 48: 115–174, 2013.

[40] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.

[41] D. Calvanese, M. Ortiz, M. Simkus, and G. Stefanoni. The complexity of explaining negative query answers in *DL-Lite*. In *Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR-12)*, 2012.

[42] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. *Artificial Intelligence*, 195: 335–360, 2013.

[43] D. Calvanese, M. Ortiz, M. Simkus, and G. Stefanoni. Reasoning about explanations for negative query answers in *DL-Lite*. *Journal of Artificial Intelligence Research*, 48:635–669, 2013.

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

[45] D. Calvanese, D. Lanti, A. Ozaki, R. Peñaloza, and G. Xiao. Enriching ontology-based data access with provenance. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, pages 1616–1623, 2019.

[46] İ. İ. Ceylan and R. Peñaloza. Probabilistic query answering in the Bayesian description logic $\mathcal{BEL}$. In *Proceedings of the 9th International Conference on Scalable Uncertainty Management (SUM-15)*, volume 9310 of *Lecture Notes in Computer Science*, pages 21–35. Springer, 2015.

[47] İ. İ. Ceylan, S. Borgwardt, and T. Lukasiewicz. Most probable explanations for probabilistic database queries. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI-17)*, pages 950–956, 2017.

[48] İ. İ. Ceylan, T. Lukasiewicz, E. Malizia, and A. Vaicenavičius. Explanations for query answers under existential rules. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, pages 1639–1646, 2019.

[49] İ. İ. Ceylan, T. Lukasiewicz, E. Malizia, C. Molinaro, and A. Vaicenavičius. Explanations for negative query answers under existential rules. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR-20)*. AAAI Press, September 2020.

[50] İ. İ. Ceylan, T. Lukasiewicz, E. Malizia, and A. Vaicenavičius. Explanations for ontology-mediated query answering in description logics. In *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI-20)*, volume 325, pages 672–679, 2020.

[51] İ. İ. Ceylan, T. Lukasiewicz, E. Malizia, and A. Vaicenavicius. Explanations for ontology-mediated query answering in description logics (extended abstract). In S. Borgwardt and T. Meyer, editors, *Proceedings of the 33rd International Workshop on Description Logics (DL-20)*, volume 2663, 2020.

[52] İ. İ. Ceylan, T. Lukasiewicz, E. Malizia, and A. Vaicenavicius. Explanations for query answers under existential rules (extended abstract). In *Proceedings of the International Workshop on Explainable Logic-Based Knowledge Representation (XLoKR-20)*, 2020.

[53] İ. İ. Ceylan, T. Lukasiewicz, E. Malizia, C. Molinaro, and A. Vaicenavičius. Preferred explanations for ontology-mediated queries under existential rules. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-21)*, 2021.

[54] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proceedings of the 9th Annual ACM Symposium on Theory of Computing (STOC-77)*, page 77–90, 1977.

[55] E. Charniak and D. McDermott. *Introduction to artificial intelligence*. Addison-Wesley series in computer science. Addison-Wesley, 1985.

[56] J. Cheney, L. Chiticariu, and W. C. Tan. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4):379–474, 2009.

[57] E. F. Codd. Derivability, redundancy and consistency of relations stored in large data banks. *Research Report / RJ / IBM*, RJ599, 1969.

[58] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[59] R. Cornet and N. de Keizer. Forty years of SNOMED: a literature review. *BMC Medical Informatics and Decision Making*, 8(S-1):S2, 2008.

[60] Y. Cui, J. Widom, and J. L. Wiener. Tracing the lineage of view data in a warehousing environment. *ACM Transactions on Database Systems (TODS)*, 25(2):179–227, 2000.

[61] E. Dantsin and A. Voronkov. Complexity of query answering in logic databases with complex values. In *Proceedings of the 4th International Symposium on Logical Foundations of Computer Science (LFCS-97)*, volume 1234 of *Lecture Notes in Computer Science*, pages 56–66. Springer, 1997.

[62] J. de Kleer. An assumption-based TMS. *Artificial Intelligence*, 28(2):127–162, 1986.

[63] J. de Kleer. Extending the ATMS. *Artificial Intelligence*, 28(2):163–196, 1986.

[64] J. de Kleer. Problem solving with the ATMS. *Artificial Intelligence*, 28(2): 197–224, 1986.

[65] A. Deutsch, A. Nash, and J. B. Remmel. The chase revisited. In *Proceedings of the 27th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS-08)*, pages 149–158, 2008.

[66] S. Doutre, M. Lafages, and M. Lagasquie-Schiex. Argumentation frameworks with higher-order attacks: Labellings and complexity. In *Proceedings of the 32nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI-20)*, pages 1210–1217.

[67] J. Du, G. Qi, Y. Shen, and J. Z. Pan. Towards practical ABox abduction in large OWL DL ontologies. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-11)*, pages 1160–1165, 2011.

[68] J. Du, K. Wang, and Y. Shen. A tractable approach to abox abduction over description logic ontologies. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI-14)*, pages 1034–1040, 2014.

[69] J. Du, K. Wang, and Y. Shen. Towards tractable and practical ABox abduction over inconsistent description logic ontologies. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI-15)*, pages 1489–1495, 2015.

[70] H. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical logic (2nd ed.)*. Undergraduate texts in mathematics. Springer, 1994. ISBN 978-3-540-94258-0.

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

[72] T. Eiter, G. Gottlob, and N. Leone. Semantics and complexity of abduction from default theories. *Artificial Intelligence*, 90(1-2):177–223, 1997.

[73] T. Eiter, G. Gottlob, and N. Leone. Abduction from logic programs: Semantics and complexity. *Theoretical Computer Science*, 189(1-2):129–177, 1997.

[74] T. Eiter, G. Gottlob, and H. Mannila. Disjunctive Datalog. *ACM Transactions on Database Systems (TODS)*, 22(3):364–418, 1997.

[75] T. Eiter, G. Gottlob, M. Ortiz, and M. Simkus. Query answering in the description logic Horn-$\mathcal{SHIQ}$. In *Proceedings of the 11th European Conference on Logics in Artificial Intelligence (JELIA-11)*, volume 5293 of *Lecture Notes in Computer Science*, pages 166–179. Springer, 2008.

[76] T. Eiter, T. Lukasiewicz, and L. Predoiu. Generalized consistent query answering under existential rules. In *Proceedings of the 15th International Conference on Principles of Knowledge Representation and Reasoning (KR-16)*, pages 359–368, 2016.

[77] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: semantics and query answering. *Theoretical Computer Science*, 336(1):89–124, 2005.

[78] M. Fürer. The computational complexity of the unconstrained limited domino problem (with implications for logical decision problems). In *Proceedings of the Symposium "Rekursive Kombinatorik"*, volume 171 of *Lecture Notes in Computer Science*, pages 312–319. Springer, 1983.

[79] H. N. Gabow, S. N. Maheswari, and L. J. Osterweil. On two problems in the generation of program test paths. *IEEE Transactions on Software Engineering*, 2(3):227–231, 1976.

[80] A. Gainer-Dewar and P. Vera-Licona. The minimal hitting set generation problem: Algorithms and computation. *SIAM Journal on Discrete Mathematics (SIDMA)*, 31(1):63–100, 2017.

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

[82] B. Glimm, C. Lutz, I. Horrocks, and U. Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *Journal of Artificial Intelligence Research*, 31: 157–204, 2008.

[83] G. Gottlob and E. Malizia. Achieving new upper bounds for the hypergraph duality problem through logic. *SIAM Journal on Computing*, 47(2):456–492, 2018.

[84] B. C. Grau, I. Horrocks, B. Motik, B. Parsia, P. F. Patel-Schneider, and U. Sattler. OWL 2: The next step for OWL. *Journal of Web Semantics*, 6(4):309–322, 2008.

[85] T. J. Green and V. Tannen. The semiring framework for database provenance. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS-17)*, pages 93–99. ACM, 2017.

[86] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *Proceedings of the 26th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS-07)*, pages 31–40. ACM, 2007.

[87] J. Hastings, D. Magka, C. R. Batchelor, L. Duan, R. Stevens, M. Ennis, and C. Steinbeck. Structure-based classification and ontology in chemistry. *Journal of Cheminformatics*, 4:8, 2012.

[88] A. Hecham, A. Arioua, G. Stapleton, and M. Croitoru. An empirical evaluation of argumentation in explaining inconsistency-tolerant query answering. In *Proceedings of the 30th International Workshop on Description Logics (DL-17)*, 2017.

[89] L. A. Hemachandra. The strong exponential hierarchy collapses. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC-87)*, page 110–122. Association for Computing Machinery, 1987.

[90] A. Hernich, C. Lutz, F. Papacchini, and F. Wolter. Dichotomies in ontology-mediated querying with the guarded fragment. *ACM Transactions on Computational Logic (TOCL)*, 21(3):20:1–20:47, 2020.

[91] M. Herschel and M. A. Hernández. Explaining missing answers to SPJUA queries. *VLDB Endowment*, 3(1):185–196, 2010.

[92] R. Hoehndorf, P. N. Schofield, and G. V. Gkoutos. The role of ontologies in biological and biomedical research: a functional perspective. *Briefings in Bioinformatics*, 16(6):1069–1080, 2015.

[93] Horizon 2020 Commission Expert Group to advise on specific ethical issues raised by driverless mobility (E03659). Ethics of connected and automated vehicles: recommendations on road safety, privacy, fairness, explainability and responsibility. 2020.

[94] M. Horridge, B. Parsia, and U. Sattler. Laconic and precise justifications in OWL. In *Proceedings of the 7th International Semantic Web Conference (ISWC-08)*, pages 323–338, 2008.

[95] M. Horridge, B. Parsia, and U. Sattler. Explaining inconsistencies in OWL ontologies. In *Proceedings of the 3rd International Conference on Scalable Uncertainty Management (SUM-09)*, pages 124–137, 2009.

[96] M. Horridge, S. Bail, B. Parsia, and U. Sattler. The cognitive complexity of OWL justifications. In *Proceedings of the 10th International Semantic Web Conference (ISWC-11)*, pages 241–256, 2011.

[97] M. Horridge, B. Parsia, and U. Sattler. Extracting justifications from bioportal ontologies. In *Proceedings of the 11th International Semantic Web Conference (ISWC-12)*, pages 287–299, 2012.

[98] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17(5):935–938, 1988.

[99] D. S. Johnson. A catalog of complexity classes. In *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, pages 67–161. Elsevier and MIT Press, 1990.

[100] J. C. Jung and C. Lutz. Ontology-based access to probabilistic data with OWL QL. In *Proceedings of the 11th International Semantic Web Conference (ISWC-12)*, volume 7649 of *Lecture Notes in Computer Science*, pages 182–197. Springer, 2012.

[101] A. Kalyanpur, B. Parsia, E. Sirin, and J. A. Hendler. Debugging unsatisfiable classes in OWL ontologies. *Journal of Web Semantics*, 3(4):268–293, 2005.

[102] A. Kalyanpur, B. Parsia, E. Sirin, and B. Cuenca-Grau. Repairing unsatisfiable concepts in OWL ontologies. In *Proceedings of the 3rd European Semantic Web Conference (ESWC-06)*, pages 170–184, 2006.

[103] A. Kalyanpur, B. Parsia, M. Horridge, and E. Sirin. Finding all justifications of OWL DL entailments. In *Proceedings of the 6th International Semantic Web Conference and the 2nd Asian Semantic Web Conference (ISWC/ASWC-07)*, pages 267–280, 2007.

[104] R. M. Karp. Reducibility among combinatorial problems. In *Proceedings of a Symposium on the Complexity of Computer Computations*, pages 85–103. Springer, 1972.

[105] S. Klamt and E. D. Gilles. Minimal cut sets in biochemical reaction networks. *Bioinformatics*, 20(2):226–234, 2004.

[106] S. Klamt, U. Haus, and F. J. Theis. Hypergraphs and cellular networks. *PLoS Computational Biology*, 5(5), 2009.

[107] S. Klarman, U. Endriss, and S. Schlobach. ABox abduction in the description logic $\mathcal{ALC}$. *Journal of Automated Reasoning*, 46(1):43–80, 2011.

[108] M. W. Krentel. The complexity of optimization problems. *Journal of Computer and System Sciences*, 36(3):490 – 509, 1988.

[109] A. Krisnadhi and C. Lutz. Data complexity in the $\mathcal{EL}$ family of description logics. In *Proceedings of the 14th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR-14)*, volume 4790 of *Lecture Notes in Computer Science*, pages 333–347. Springer, 2007.

[110] M. Krötzsch and S. Rudolph. Conjunctive queries for $\mathcal{EL}$ with composition of roles. In *Proceedings of the 20th International Workshop on Description Logics (DL-07)*, volume 250 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

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

[112] A. Lopatenko and L. E. Bertossi. Complexity of consistent query answering in databases under cardinality-based and incremental repair semantics. In *Proceedings of the 11th International Conference on Database Theory (ICDT-07)*, pages 179–193, 2007.

[113] T. Lukasiewicz and E. Malizia. A novel characterization of the complexity class Θ_k^P based on counting and comparison. *Theoretical Computer Science*, 694: 21–33, 2017.

[114] T. Lukasiewicz, M. V. Martinez, A. Pieris, and G. I. Simari. From classical to consistent query answering under existential rules. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI-15)*, pages 1546–1552, 2015.

[115] T. Lukasiewicz, E. Malizia, and C. Molinaro. Complexity of approximate query answering under inconsistency in Datalog+/–. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-18)*, pages 1921–1927, 2018.

[116] T. Lukasiewicz, E. Malizia, and C. Molinaro. Explanations for inconsistency-tolerant query answering under existential rules. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI-20)*, pages 2909–2916, 2020.

[117] C. Lutz. The complexity of conjunctive query answering in expressive description logics. In *Proceedings of the 4th International Joint Conference on Automated Reasoning (IJCAR-08)*, volume 5195 of *Lecture Notes in Computer Science*, pages 179–193. Springer, 2008.

[118] C. Lutz and L. Sabellek. Ontology-mediated querying with the description logic $\mathcal{EL}$: trichotomy and linear datalog rewritability. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI-17)*, pages 1181–1187, 2017.

[119] D. Magka, B. Motik, and I. Horrocks. Modelling structured domains using description graphs and logic programming. In *Proceedings of the 9th Extended Semantic Web Conference (ESWC-12)*, volume 7295 of *Lecture Notes in Computer Science*, pages 330–344. Springer, 2012.

[120] D. L. McGuinness and A. Borgida. Explaining subsumption in description logics. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 816–821, 1995.

[121] C. Molinaro, A. Sliva, and V. S. Subrahmanian. Super-solutions: Succinctly representing solutions in abductive annotated probabilistic temporal logic. *ACM Transactions on Computational Logic (TOCL)*, 15(3):18:1–18:35, 2014.

[122] S. Moore. Gartner top 10 data and analytics trends for 2019. https://www.gartner.com/smarterwithgartner/gartner-top-10-data-analytics-trends/, 2019. Accessed: 2020-11-02.

[123] C. Mungall. Experiences using logic programming in bioinformatics. In *Proceedings of the 25th International Conference on Logic Programming (ICLP-09)*, volume 5649 of *Lecture Notes in Computer Science*, pages 1–21. Springer, 2009.

[124] Y. Nenov, R. Piro, B. Motik, I. Horrocks, Z. Wu, and J. Banerjee. RDFox: A highly-scalable RDF store. In *Proceedings of the 14th International Semantic Web Conference (ISWC-14)*, volume 9367, pages 3–20, 2015.

[125] L. A. Nguyen. Horn knowledge bases in regular description logics with PTime data complexity. *Fundamenta Informaticae*, 104(4):349–384, 2010.

[126] M. Ortiz, S. Rudolph, and M. Simkus. Query answering in the horn fragments of the description logics $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 1039–1044. IJCAI/AAAI, 2011.

[127] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[128] C. H. Papadimitriou and D. Wolfe. The complexity of facets resolved. *Journal of Computer and System Sciences*, 37(1):2–13, 1988.

[129] C. H. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). *Journal of Computer and System Sciences*, 28(2):244–259, 1984.

[130] G. Paul. Approaches to abductive reasoning: an overview. *Artificial lntelligence Review*, 7(2):109–152, 1993.

[131] R. Peñaloza. Axiom pinpointing. *CoRR*, abs/2003.08298, 2020. URL https://arxiv.org/abs/2003.08298.

[132] R. Peñaloza and B. Sertkaya. Axiom pinpointing is hard. In *Proceedings of the 22nd International Workshop on Description Logics (DL-09)*, volume 477, 2009.

[133] R. Peñaloza and B. Sertkaya. On the complexity of axiom pinpointing in the $\mathcal{EL}$ family of description logics. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-10)*, 2010.

[134] R. Peñaloza and B. Sertkaya. Complexity of axiom pinpointing in the DL-Lite family of description logics. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI-10)*, volume 215, pages 29–34, 2010.

[135] R. Peñaloza and B. Sertkaya. Understanding the complexity of axiom pinpointing in lightweight description logics. *Artificial Intelligence*, 250:80–104, 2017.

[136] Y. Peng and J. A. Reggia. Abductive inference models for diagnostic problem-solving. *Knowledge Engineering Review*, 6(2):127–129, 1991.

[137] A. Poggi, D. Lembo, D. Calvanese, G. D. Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *Journal on Data Semantics*, 10:133–173, 2008.

[138] H. E. Pople. On the mechanization of abductive logic. In *Proceedings of the 3rd International Joint Conference on Artificial Intelligence (IJCAI-73)*, page 147–152, 1973.

[139] E. Ramadan, A. Tarafdar, and A. Pothen. A hypergraph model for the yeast protein complex network. In *Proceedings of the 18th International Parallel & Distributed Processing Symposium (IPDPS-04)*, 2004.

[140] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32 (1):57–95, 1987.

[141] R. Rosati. On conjunctive query answering in $\mathcal{EL}$. In *Proceedings of the 20th International Workshop on Description Logics (DL-07)*, volume 250 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[142] E. Salvat and M.-L. Mugnier. Sound and complete forward and backward chainings of graph rules. In *Proceedings of the 4th International Conference on Conceptual Structures (ICCS-96)*, pages 248–262, 1996.

[143] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 466–471. Morgan Kaufmann, 1991.

[144] S. Schlobach and R. Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 355–360, 2003.

[145] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1 – 26, 1991.

[146] R. Sebastiani and M. Vescovi. Axiom pinpointing in lightweight description logics via Horn-SAT encoding and conflict analysis. In *Proceedings of the 22nd International Conference on Automated Deduction (CADE-09)*, pages 84–99, 2009.

[147] L. Stockmeyer. Planar 3-colorability is polynomial complete. *SIGACT News*, 5 (3):19–25, 1973. doi: 10.1145/1008293.1008294.

[148] L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1 – 22, 1976.

[149] B. Suntisrivaraporn, G. Qi, Q. Ji, and P. Haase. A modularization-based approach to finding all justifications for OWL DL entailments. In *Proceedings of the 3rd Asian Semantic Web Conference (ASWC-08)*, pages 1–15, 2008.

[150] R. Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26(3):279–284, 1988.

[151] M. Y. Vardi. The complexity of relational query languages. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing (STOC-82)*, pages 137–146, 1982.

[152] M. Y. Vardi. On the complexity of bounded-variable queries. In *Proceedings of the 14th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS-95)*, pages 266–276. ACM Press, 1995.

[153] H. Vollmer. *Introduction to Circuit Complexity - A Uniform Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 1999.

[154] Y. R. Wang and S. E. Madnick. A polygen model for heterogeneous database systems: The source tagging perspective. In *Proceedings of the 16th International Conference on Very Large Data Bases (VLDB-90)*, pages 519–538. Morgan Kaufmann, 1990.

[155] Z. Wang, M. Chitsaz, K. Wang, and J. Du. Towards scalable and complete query explanation with OWL 2 EL ontologies. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM-15)*, pages 743–752, 2015.

[156] C. Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):23 – 33, 1976.