

Explaining Ontology-Mediated Query Answers using Proofs over Universal Models (Technical Report)

Christian Alrabbaa Stefan Borgwardt
Patrick Koopmann Alisa Kovtunova

Abstract

In ontology-mediated query answering, access to incomplete data sources is mediated by a conceptual layer constituted by an ontology, which can be formulated in a description logic (DL) or using existential rules. In the literature, there exists a multitude of complex techniques for incorporating ontological knowledge into queries. However, few of these approaches were designed for explainability of the query answers. We tackle this challenge by adapting an existing proof framework toward conjunctive query answering, based on the notion of universal models. We investigate the data and combined complexity of determining the existence of a proof below a given quality threshold, which can be measured in different ways. By distinguishing various parameters such as the shape of the query, we obtain an overview of the complexity of this problem for several Horn DLs.

1 Introduction

Description logics (DLs) are a family of knowledge representation formalisms that can be seen as decidable fragments of first-order logic using only unary and binary predicates [8]. This family contains very expressive DLs like *SROIQ*, which underlies the standardized Web Ontology Language OWL 2,¹ as well as the light-weight DLs *DL-Lite_R* and *EL*, corresponding to the OWL 2 profiles QL and EL, respectively. We focus here on *Horn DLs* up to *Horn-ALCHOI* [33, 28], whose axioms can be expressed as *existential rules* (with equality) [13]. The complexity of standard reasoning problems such as entailment of axioms or facts (ground atoms) from an ontology (a finite set of axioms) has been studied for decades and is well-understood by now [32, 8]. Another popular reasoning problem for DLs is that of *ontology-mediated query answering (OMQA)*, which generalizes query answering over databases by allowing to query implicit knowledge that is inferred by the ontology [14, 33].

Explaining DL reasoning has a long tradition, starting with the first works on *proofs* for standard DL entailments [30, 11]. A popular and very effective method is to compute *justifications*, which simply point out the axioms from the ontology that are responsible for an entailment [37, 9, 34, 23]. More recently, work

¹<https://www.w3.org/TR/owl2-overview/>

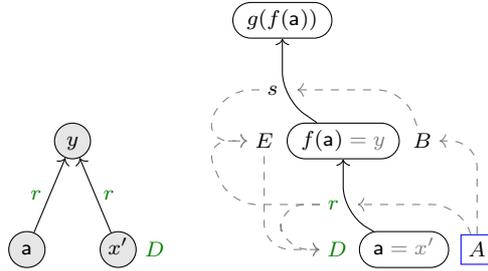


Figure 1: The query (on the left) and the relevant part of the universal model (on the right) from Example 1.

has resumed on techniques to find proofs for explaining more complex logical consequences [24, 25, 3, 4, 5]. On the other hand, if a desired entailment does *not* hold, one needs different explanation techniques such as abduction [27, 19, 17] or counterinterpretations [7]. Explaining answers to conjunctive queries (CQs) has also been investigated before, in the form of abduction for missing answers over *DL-Lite* ontologies [17], provenance for positive answers in *DL-Lite* and \mathcal{EL} [16, 12], as proofs for *DL-Lite* query answering [10, 38, 20], as well as proofs and provenance for rule reasoning [21, 35]. Inspired by the latter, we also investigate proofs for CQ answers, but consider more expressive DLs and want to find *good* proofs according to different quality measures. We focus on Horn DLs, for which every ontology has a *universal model* that captures exactly the query answers over the ontology [13]. While classically models are used for explaining missing entailments [7], this property allows us to use universal models also to explain positive query answers.

Example 1. Consider the fact $A(\mathbf{a})$, the existential rules (which can be expressed in *Horn-ALCHOT*)

$$\begin{aligned} A(x) &\rightarrow \exists y. r(x, y) \wedge B(y), & s(x, y) \wedge r(z, x) &\rightarrow E(x), \\ B(x) &\rightarrow \exists z. s(x, z) \wedge A(z), & E(x) \wedge r(y, x) &\rightarrow D(y), \end{aligned}$$

and the conjunctive query $\mathbf{q}(x) = \exists x', y. r(x, y) \wedge r(x', y) \wedge D(x')$. Individual \mathbf{a} is an answer to \mathbf{q} in this ontology. The query instantiated with this answer is depicted on the left in Fig. 1, using edges for binary predicates and node labels for unary predicates. To explain the answer, we show on the right of the figure the relevant part of the universal model of the ontology, where unary and binary predicates are represented similarly. The nodes represent objects in the model and are identified by Skolem terms, together with the assignments to the variables in the query. For example, $f(\mathbf{a})$ can be described as “the r -successor of \mathbf{a} ”, which has to be present in any model of the ontology due to the first rule. The Skolem functions like f and g are created uniquely for each existentially quantified variable in the rules. In addition to explaining how the query is matched to the universal model, the dashed gray edges indicate a proof of $\mathbf{q}(\mathbf{a})$. For instance, $A(\mathbf{a})$, together with the first rule, implies the existence of the r -successor satisfying B , and $D(\mathbf{a})$ follows from $E(f(\mathbf{a}))$ and $r(\mathbf{a}, f(\mathbf{a}))$ through the last rule. To make this representation more accessible for larger proofs, in real applications we would show proof steps only on demand, whenever a user selects a fact to be explained in the model.

Table 1: Summary of the combined complexity results for $OP_{sk}(\mathcal{L}, \mathbf{m})$.

Measure	<i>DL-Lite</i>		\mathcal{EL}		<i>Horn-ALCHOI</i>
	tree-sh.	CQ	IQ	CQ	CQ
Domain size	NP-c [Th. 6,9]		in EXPTIME [Th. 6]		in NEXPTIME [Th. 13]
Tree size	in P [Th. 8]	NP-c [Th. 6,7]	P-c [Th. 10]	NP-c [Th. 10]	in PSPACE [Th. 13]
Proof size	NP-c [Th. 6,9]		in EXPTIME [Th. 14]		
Proof size bound	polynomial [Lem. 4]		exponential [Lem. 4]		double exponential [Lem. 11]

In previous work [3, 5], we developed a formal framework for proofs in standard DL reasoning. We investigated the complexity of finding *small* proofs according to different proof measures: (*proof*) *size*, i.e. the number of distinct formulas in a proof, and (*proof*) *tree size*, i.e. the size when the proof is presented as tree, as it is done often in practice [25, 2]. In this framework, proofs are generated by a so-called *deriver* that specifies which inferences are possible in a proof.

To be able to reuse results, the present work develops proofs for query answers within the same framework. In particular, in order to explain query answers using universal models, we introduce a special deriver that applies to a large family of Horn-DLs, and in which inferences in the proof directly correspond to the construction of the universal model. For such proofs, if we visualize them as in the example, another proof measure becomes relevant: the *domain size*, which is the number of elements from the universal model that are used in the proof. In the example, the domain size of the proof is 3. After introducing our deriver, we investigate the complexity of finding good proofs w.r.t. the different measures, as well as bounds on the size of the obtained proofs. An overview of our results is shown in Table 1. Because it introduces fresh objects, our deriver is only sound for a Skolemized version of the TBox, and not for the original TBox. At the end of the paper, we have a brief look at another deriver in which all inferences are sound w.r.t. the original TBox, and argue that, while the complexity of the resulting decision problem is often similar, this deriver is less helpful in explaining query answers to users. This paper extends initial results in this direction from a workshop paper [6]. Proof details can be found in the appendix.

2 Preliminaries

Logics. We assume basic knowledge about first-order logic and familiarity with terminology such as variables, terms, atoms, sentences, etc. Throughout the paper, we use \mathcal{L} to refer to fragments of first-order logic. DLs are fragments of the two-variable fragment, for which we assume unary predicates to be taken from a countably infinite set N_C of *concept names*, binary predicates to be taken from a countably infinite set N_R of *role names*, and constants to be taken from a countably infinite set N_I of *individual names* [8]. Moreover, we use \top and \perp as special concept names that are always satisfied or always not satisfied,

Table 2: Sentences of *Horn-ALCCHOI*, where $A, B, C \in \mathbf{N}_C$, $a \in \mathbf{N}_I$, R, R_1, R_2 are *roles* of the form r or r^- (*inverse role*), $r \in \mathbf{N}_R$, and we identify $r^-(x, y)$ with $r(y, x)$.

(i)	$A \sqsubseteq B$	$A(x) \rightarrow B(x)$
(ii)	$A \sqcap B \sqsubseteq C$	$A(x) \wedge B(x) \rightarrow C(x)$
(iii)	$\exists R.A \sqsubseteq B$	$R(x, y) \wedge A(y) \rightarrow B(x)$
(iv)	$A \sqsubseteq \exists R.B$	$A(x) \rightarrow \exists y. R(x, y) \wedge B(y)$
(v)	$A \sqsubseteq \forall R.B$	$A(x) \wedge R(x, y) \rightarrow B(y)$
(vi)	$A \sqsubseteq \{a\}$	$A(x) \rightarrow x = a$
(vii)	$R_1 \sqsubseteq R_2$	$R_1(x, y) \rightarrow R_2(x, y)$

respectively. We focus on *Horn DLs* that can be represented using existential rules with equality [13]. An existential rule is a first-order sentence of the form $\forall \vec{y}, \vec{z}. \psi(\vec{y}, \vec{z}) \rightarrow \exists \vec{u}. \chi(\vec{z}, \vec{u})$, with the *body* $\psi(\vec{y}, \vec{z})$ and the *head* $\chi(\vec{z}, \vec{u})$ being conjunctions of atoms of the form $A(t_1)$, $R(t_1, t_2)$, or $t_1 = t_2$, where t_1 and t_2 are constants or variables from \vec{z} , \vec{u} and \vec{y} . We usually omit the universal quantification.

For DLs, one usually uses a different, dedicated syntax. Table 2 shows the allowed rules in *Horn-ALCCHOI*, together with their representation in DL syntax, where, for simplicity, we assume the rules to be *normalized*. A set \mathcal{T} of such rules is called *TBox* or *ontology*. In *Horn-ALC*, only expressions of the forms (i)–(v) without inverse roles are allowed, \mathcal{EL} further restricts *Horn-ALC* by disallowing (v) and \perp , and *DL-Lite* only allows expressions $R_1 \sqsubseteq R_2$, $A \sqsubseteq C$, $C \sqsubseteq A$, and $A \sqcap B \sqsubseteq \perp$, where R_1, R_2 are (possibly inverse) roles, $A, B \in \mathbf{N}_C$, and C is either a concept name or $\exists R.\top$, for a (possibly inverse) role R .

Query Answering. An *ABox* \mathcal{A} is a set of ground atoms (called *facts*) of the form $A(a)$ or $r(a, b)$, which together with a TBox \mathcal{T} forms a *knowledge base* (KB) $\mathcal{K} = \mathcal{T} \cup \mathcal{A}$. Its *signature* $\text{sig}(\mathcal{K})$ is the set of all concept, role, and individual names $\text{ind}(\mathcal{K})$ occurring in it. A *conjunctive query* (CQ) $\mathbf{q}(\vec{x})$ is an expression of the form $\exists \vec{y}. \phi(\vec{x}, \vec{y})$, where $\phi(\vec{x}, \vec{y})$ is a conjunction of atoms $A(t)$ or $r(s, t)$ and s, t are variables or constants. The variables in \vec{x} are called *answer variables* and \vec{y} are the *existentially quantified variables*. If $\mathbf{q}(\vec{x})$ contains only a single unary atom, it is called *instance query* (IQ). If \vec{x} is empty, then $\mathbf{q}(\vec{x})$ is called *Boolean*. Note that ABox facts are a special case of Boolean CQs with only one atom and no variables. A tuple \vec{a} of constants from \mathcal{A} is a *certain answer* to $\mathbf{q}(\vec{x})$ over a KB \mathcal{K} , written $\mathcal{K} \models \mathbf{q}(\vec{a})$, if every model of \mathcal{K} satisfies the sentence $\mathbf{q}(\vec{a})$. We may write $A(x) \in \mathbf{q}$ to indicate that $A(x)$ is an atom in ϕ . A *union of CQs* (UCQ) is a disjunction of CQs sharing the same answer variables. A CQ $\mathbf{q}(\vec{x})$ is *UCQ-rewritable* over a TBox \mathcal{T} if there exists a UCQ $\mathbf{q}_{\mathcal{T}}(\vec{x})$ such that, for every ABox \mathcal{A} and tuple \vec{a} , $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}(\vec{a})$ iff $\mathcal{A} \models \mathbf{q}_{\mathcal{T}}(\vec{a})$. This is the case, for example, for all CQs over *DL-Lite_R* TBoxes [14]. Since we consider proofs for a given, fixed answer \vec{a} , we consider only the Boolean CQ $\mathbf{q}(\vec{a})$, which we denote in the following simply as \mathbf{q} .

Proofs. Following the formal framework in [3, 4, 5], we view proofs in a logic \mathcal{L} as finite directed hypergraphs (V, E, ℓ) where each vertex $v \in V$ is labeled by

an \mathcal{L} -sentence $\ell(v)$, and every hyperedge is of the form $(S, d) \in E$ the finite set $S \subseteq V$ being the *premises* and $d \in V$ the *conclusion*, which we may depict as

$$\frac{p \quad p \rightarrow q}{q} \quad \text{or} \quad \begin{array}{c} \text{---} p \text{---} \quad \text{---} p \rightarrow q \text{---} \\ \quad \quad \quad \downarrow \\ \quad \quad \quad q \end{array}$$

We call these edges also *inferences*. Proofs can be found by looking at derivation structures. Formally, a *derivation structure* over a KB \mathcal{K} is a possibly infinite hypergraph as above in which each inference (S, d) is *sound*, that is, the labels of S logically entail the label of d , and every leaf (vertex without incoming edges) is labeled by an element of \mathcal{K} . A *proof* for an entailment $\mathcal{K} \models \eta$ is a finite derivation structure that (i) is acyclic, (ii) has exactly one sink (the conclusion), which is labeled by the goal sentence η , and (iii) in which each vertex v is the conclusion of at most one hyperedge (S, v) . The *size* of a proof is the number of its vertices, and the *tree size* is the size of its tree unraveling, starting from the sink.

Proofs are usually generated based on a calculus or some reasoning system. This is formalized by the notion of a *deriver*, which, for a given entailment $\mathcal{K} \models \eta$, generates a derivation structure in which different possible proofs can be found. Formally, a *deriver* \mathfrak{D} for a logic \mathcal{L} is a function that takes as input an \mathcal{L} -theory \mathcal{K} and an \mathcal{L} -sentence η , and returns a derivation structure $\mathfrak{D}(\mathcal{K}, \eta)$ over \mathcal{K} that describes all inference steps that \mathfrak{D} could perform in order to derive η from \mathcal{K} . This structure is not necessarily computed explicitly, but can be accessed through an oracle (in practice, this corresponds, for example, to checking whether an inference conforms to a calculus).

Remark. *We argue that we can make some simplifying assumptions on the shape of Horn-ALCHOI rules.*

- (a) *To keep constructions easier, we assume TBoxes to be normalized as in [33, 18]. Such a normalization can always be performed in polynomial time by introducing fresh names as abbreviations for complex formulas and applying standard transformations. We can transform a proof over a normalized TBox to a proof for the original non-normalized TBox by (i) replacing the new names with the original complex expressions, which may result in intermediate proof steps using atoms like $(\exists r.A)(x)$, and (ii) possibly introducing new inference steps corresponding to normalization steps. This increases the size of the proofs at most polynomially, which is why we believe our results are also relevant to non-normalized TBoxes.*
- (b) *We assume KBs to be consistent. Since for Horn DLs, \perp is only useful to create inconsistencies, we assume in the following that \perp is never used.*

3 A Deriver Using Universal Models

A distinguishing feature of Horn DLs is that every KB has a *universal model* which satisfies exactly the Boolean CQs that are entailed by the KB. In the literature on existential rules, the term *chase* refers to (different variants of) universal models [13]. Intuitively, a chase is constructed by applying rules to facts, where fresh objects are introduced for existential quantified variables. As we illustrate in the introduction, proofs connected to universal models can help

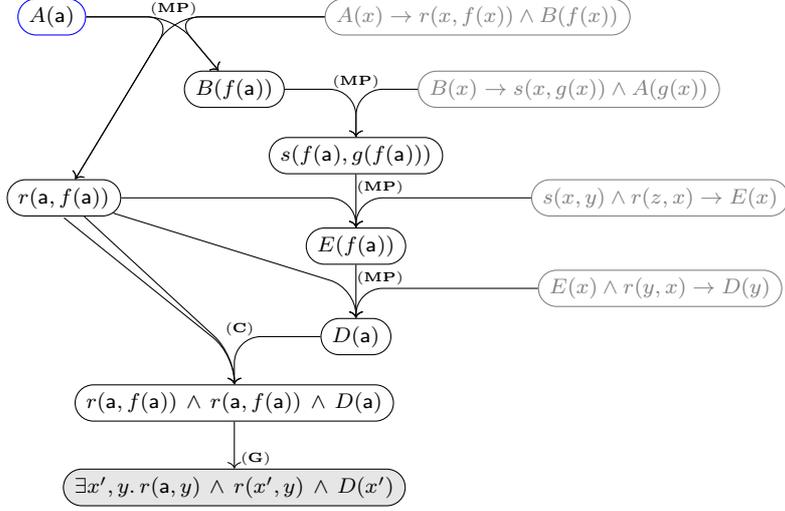


Figure 2: A Skolemized proof for the example (colors are used for the ease of reading)

to explain query answers. However, because we require inferences to be sound, our framework does not allow for an inference mechanism that introduces fresh objects. Our solution is to provide a derivier that is sound w.r.t. the *Skolemized* TBox, rather than the original TBox. By Skolemizing, we eliminate existential quantification using fresh function symbols. The saturation of an ABox using a Skolemized TBox produces the least Herbrand model, which in turn corresponds to the *Skolem chase* (a.k.a. semi-oblivious chase) [29] of the original TBox. In our case, existential quantification only occurs in rules of the form (iv) (see Table 2), which then get transformed into $A(x) \rightarrow r(x, f(x)) \wedge B(f(x))$ where f is unique for each existentially quantified variable. Given a TBox \mathcal{T} , we denote by \mathcal{T}^s the result of Skolemizing all axioms in \mathcal{T} . A universal model of $\mathcal{T} \cup \mathcal{A}$ can then be obtained by “applying” the rules in \mathcal{T}^s to \mathcal{A} until a fixpoint is reached (which may result in an infinite set of atoms).

In the following, let $\mathcal{T} \cup \mathcal{A}$ be a KB in some DL \mathcal{L} and \mathbf{q} a Boolean CQ with $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$, which we want to explain. For this, we define an appropriate derivier over the extended logic \mathcal{L}_{cq} , which contains the results of Skolemizing the rules in Table 2 as well as all Boolean CQs. To provide good explanations, inferences should be simple, i.e. involve only small modifications of the premises. For TBox entailment, in [3, 4, 5], we considered deriviers based on the inference schemas used by consequence-based reasoners. To obtain proofs for CQs, we present the derivier \mathfrak{D}_{sk} , which inspired by the approach from [10] and mainly operates on ground CQs that may use Skolem terms, but no existential quantification. Since ground atoms do not share variables, we mainly need to consider inferences on single atoms, which allows for fine-grained proofs (see Fig. 2). Only at the end we need to compose atoms to obtain the desired CQ \mathbf{q} .

The inference schemas of \mathfrak{D}_{sk} are shown in Fig. 3. In (MP), $\alpha_i(\vec{t}_i)$ and $\beta(\vec{s})$ are ground atoms, $\psi(\vec{y}, \vec{z}) \rightarrow \chi(\vec{z})$ is a Skolemized rule from \mathcal{T}^s , and there must be a substitution π such that $\pi(\psi(\vec{y}, \vec{z})) = \{\alpha_1(\vec{t}_1), \dots, \alpha_n(\vec{t}_n)\}$ and $\beta(\vec{s}) \in \pi(\chi(\vec{z}))$. (E) deals with equalities $t_1 = t_2$ by copying atoms $\alpha(\vec{t})$ that

$$\boxed{
\begin{array}{c}
\frac{\alpha_1(\vec{t}_1) \quad \dots \quad \alpha_n(\vec{t}_n) \quad \psi(\vec{y}, \vec{z}) \rightarrow \chi(\vec{z})}{\beta(\vec{s})} \text{ (MP)} \quad \frac{\alpha(\vec{t}) \quad t_1 = t_2}{\alpha(\vec{t})[t_1 \mapsto t_2]} \text{ (E)} \\
\frac{\alpha_1(\vec{t}_1) \quad \dots \quad \alpha_n(\vec{t}_n)}{\alpha_1(\vec{t}_1) \wedge \dots \wedge \alpha_n(\vec{t}_n)} \text{ (C)} \quad \frac{\phi(\vec{t})}{\exists \vec{x}. \phi(\vec{x})} \text{ (G)}
\end{array}
}$$

Figure 3: Inference schemas in \mathfrak{D}_{sk} (modus ponens, equality, conjunction, generalization).

contain t_1 or t_2 (we consider $=$ -atoms to be symmetric). We only apply **(E)** to replace top-level terms, not nested terms. Replacing also nested terms might be logically sound, but would not improve the readability of the proof, and is also not needed for completeness. To complete the proof, **(C)** combines several ground atoms into a conjunction, and **(G)** generalizes ground terms to variables in order to produce the final CQ (see Fig. 2). Note that the same atom can be used several times as a premise for **(MP)** or **(C)**, which then however results in a *double connection* as in Fig. 2 for $r(\mathbf{a}, f(\mathbf{a}))$. Consequently, the premise (and the subproof above it) would be duplicated in the tree unraveling of the proof.

Definition 1. $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ is an infinite derivation structure over $\mathcal{T}^s \cup \mathcal{A}$ with vertices for the axioms in $\mathcal{T}^s \cup \mathcal{A}$ and all Boolean CQs over $\text{sig}(\mathcal{T}^s \cup \mathcal{A})$, and hyperedges for all possible instances of **(MP)**, **(E)**, **(C)**, and **(G)** over these vertices.² An (admissible) proof in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ is a proof of $\mathcal{T}^s \cup \mathcal{A} \models \mathbf{q}$ that has a label-preserving homomorphism into this derivation structure.

It is easy to check that these inferences are sound. Moreover, they are also complete, i.e. if $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$ holds, then there exists a proof for it (w.r.t. \mathcal{T}^s). To see this, observe that we closely follow the (Skolem) chase construction for existential rules [13, 29], where **(MP)** corresponds to standard chase steps, and **(E)** can be seen as merging domain elements in case of equalities ((**C**) and **(G)** are only relevant to obtain the final CQ). The resulting model M is universal, which means that $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$ implies $M \models \mathbf{q}$, which, in turn, shows that there must be a proof in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$.

4 Finding Good Proofs in \mathfrak{D}_{sk}

We are interested in the worst-case complexity of computing good proofs with our deriver \mathfrak{D}_{sk} . In the following, we denote by $\mathbf{m}_s(\mathcal{P})$ ($\mathbf{m}_t(\mathcal{P})$) the (tree) size of a proof \mathcal{P} . In addition, we consider the *domain size* $\mathbf{m}_d(\mathcal{P})$, which is defined as the number of ground terms appearing in \mathcal{P} . We consider the following decision problem $\text{OP}_{sk}(\mathcal{L}, \mathbf{m}_x)$ for some DL \mathcal{L} and measure $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t, \mathbf{m}_d\}$: given an \mathcal{L} -KB $\mathcal{T} \cup \mathcal{A}$, a (Boolean) CQ \mathbf{q} such that $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$, and a natural number $n > 1$ encoded in binary³, is there a proof \mathcal{P} for \mathbf{q} in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ with $\mathbf{m}_x(\mathcal{P}) \leq n$?

²This derivation structure is uniquely determined except for the names of the vertices, which are irrelevant for our purposes since we use only their labels.

³Unary encoding of n would make the problem much easier due to imposing a small (polynomial) upper bound on the (domain/tree) size of proofs. Hence, binary encoding puts more emphasis on the impact of the KB and the query on the decision problem.

To better isolate the complexity of finding small proofs from that of query answering, we assume $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$ as prerequisite, which fits the intuition that users would request an explanation only after they know that \mathbf{q} is entailed. Similarly to Lemma 7 in [5], instead of looking for arbitrary proofs and homomorphisms into the derivation structure (see Def. 1), one can restrict the search to *subproofs*⁴ of $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$, which we will often do implicitly.

Lemma 1. *For any measure $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t, \mathbf{m}_d\}$, if there is an admissible proof \mathcal{P} w.r.t. $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ with $\mathbf{m}_x(\mathcal{P}) \leq n$, then there exists a subproof \mathcal{P}' of $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ for $\mathcal{T}^s \cup \mathcal{A} \models \mathbf{q}$ with $\mathbf{m}_x(\mathcal{P}') \leq n$.*

Since domain size also satisfies the preconditions of Lemma 7 in [5], the statement of Lemma 1 can be shown similarly.

4.1 The Data Complexity of Finding Good Proofs

It is common in the context of OMQA to distinguish between *data complexity*, where only the data varies, and *combined complexity*, where also the influence of the other inputs is taken into account. This raises the question whether the bound n is seen as part of input for the data complexity or not. It turns out that fixing n trivializes the data complexity, because then n also fixes the set of relevant ABoxes modulo isomorphism, so that the problem can be reduced to UCQ entailment.

Theorem 2. *For a constant n , any \mathcal{L} , and any $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t, \mathbf{m}_d\}$, $\text{OP}_{sk}(\mathcal{L}, \mathbf{m}_x)$ is in AC^0 in data complexity.*

One may argue that, since the size of the proof depends on \mathcal{A} , the bound n on the proof size should be considered part of the input as well. Under this assumption, our decision problem is not necessarily in AC^0 anymore. For example, consider the \mathcal{EL} TBox $\{\exists r.A \sqsubseteq A\}$ and $q(x) \leftarrow A(x)$. For every n , there is an ABox \mathcal{A} such that $A(a)$ is entailed by a sequence of n role atoms, and thus needs a proof of size at least n . Deciding whether this query admits a bounded proof is thus as hard as deciding whether it admits an answer at all in \mathcal{A} , i.e. P-hard [36]. However, the problem stays in AC^0 for DLs over which CQs are UCQ-rewritable, e.g. *DL-Lite_R* [14], because the number of (non-isomorphic) proofs that we need to consider is bounded by the size of the rewriting, which is constant in data complexity.

Theorem 3. *For any $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t, \mathbf{m}_d\}$ and any \mathcal{L} such that all CQs are UCQ-rewritable over \mathcal{L} -TBoxes, $\text{OP}_{sk}(\mathcal{L}, \mathbf{m}_x)$ is in AC^0 in data complexity.*

4.2 Finding Good Proofs with Lightweight Ontologies

We now consider the combined complexity of our problems for *DL-Lite_R* and \mathcal{EL} . In [3, 5], we established general upper bounds for finding proofs of bounded size. These results depend only on the size of the derivation structure obtained for the given input. However, \mathfrak{D}_{sk} does not produce finite derivation structures since there can be Skolem terms of arbitrary nesting depth. Nevertheless, proofs cannot be infinite, and therefore we first study how large proofs in \mathfrak{D}_{sk} can get

⁴see Appendix A for definitions.

in the worst case. In particular, for \mathcal{EL} one can enforce proofs that are binary trees of polynomial depth, and therefore of exponential size.

Lemma 4. *One can construct a TBox $\mathcal{T}_{\mathcal{L},n}$ in time polynomial in n such that $\mathcal{T}_{\mathcal{L},n} \cup \{A(a)\} \models B(a)$, but every proof of the entailment is of (domain/tree) size*

1. *polynomial in n for $\mathcal{L} = DL\text{-}Lite_R$,*
2. *exponential in n for $\mathcal{L} = \mathcal{EL}$.*

Moreover, there exists an \mathcal{EL} -TBox \mathcal{T} for which one can construct an ABox \mathcal{A}_n in time polynomial in n such that $\mathcal{T} \cup \mathcal{A}_n \models A(a)$, but every proof of it is of a tree size exponential in n .

To obtain matching upper bounds, we can bound the number of relevant Skolem terms in \mathfrak{D}_{sk} by investigating which part of the universal model is necessary to satisfy the query \mathbf{q} .

Lemma 5. *For any CQ entailment $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$, there exists a proof of*

1. *(domain/tree) size polynomial in $|\mathcal{T} \cup \mathcal{A}|$ and $|\mathbf{q}|$ if $\mathcal{L} = DL\text{-}Lite_R$,*
2. *(domain) size exponential in $|\mathcal{T}|$ and $|\mathbf{q}|$ and polynomial in $|\mathcal{A}|$ if $\mathcal{L} = \mathcal{EL}$,*
3. *tree size exponential in $|\mathcal{T} \cup \mathcal{A}|$ and $|\mathbf{q}|$ if $\mathcal{L} = \mathcal{EL}$.*

This immediately allows us to show some generic upper bounds by guessing proofs up to the specified sizes.

Theorem 6. *For any $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t, \mathbf{m}_d\}$, $OP_{sk}(\mathcal{EL}, \mathbf{m}_x)$ is in NEXPTIME and $OP_{sk}(DL\text{-}Lite_R, \mathbf{m}_x)$ is in NP.*

In some cases, we can show matching lower bounds via reductions from the Boolean query entailment problem. Using Lemma 5, we can find an upper bound n for any proof showing $\mathcal{K} \models q$ provided that it holds. To satisfy the prerequisites of OP_{sk} , we then extend \mathcal{K} by a second KB \mathcal{K}' in which $\mathcal{K}' \models q$, but only with a proof *larger* than n .

Theorem 7. *For $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t\}$, $OP_{sk}(DL\text{-}Lite_R, \mathbf{m}_x)$ is NP-hard.*

To obtain tractability, we can restrict the shape of the query. The *Gaifman graph* of a query \mathbf{q} is the undirected graph that uses the terms of \mathbf{q} as nodes and has an edge between terms occurring together in an atom. A query is *tree-shaped* if its Gaifman graph is a tree. We can exploit this structure to deterministically explore in polynomial time all relevant proofs of minimal tree size over $DL\text{-}Lite_R$ KBs.

Theorem 8. *Given a $DL\text{-}Lite_R$ KB $\mathcal{T} \cup \mathcal{A}$ and a tree-shaped query \mathbf{q} , one can compute in polynomial time a proof of minimal tree size in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$.*

The central property used in the proof of Theorem 8 is that for tree size the proof of each atom in \mathbf{q} is counted separately, even if two atoms are proven in the same way. Since \mathbf{m}_d and \mathbf{m}_s do not exhibit this redundancy, we can show that the corresponding decision problems are already NP-hard for tree-shaped queries, and even *without a TBox*, via reductions from the propositional satisfiability problem.

Theorem 9. *Let \mathcal{L} be an arbitrary DL and $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_d\}$. For tree-shaped CQs, $\text{OP}_{\text{sk}}(\mathcal{L}, \mathbf{m}_x)$ is NP-hard.*

For \mathcal{EL} , we can similarly show improved complexity bounds for the case of tree size, where the lower bounds are obtained using the same idea as for Theorem 7, however this time using the exponential bound on the tree size from Lemma 5.

Theorem 10. *$\text{OP}_{\text{sk}}(\mathcal{EL}, \mathbf{m}_t)$ is NP-complete in combined, and in P in data complexity. For IQs, the problem is P-complete in combined complexity.*

4.3 Finding Good Proofs with Expressive Ontologies

We continue our journey towards more expressive DLs. First, we establish a more expressive counterpart of Lemma 4. This time, we can even enforce trees of exponential depth, by implementing a binary counter using concept names for the different bit values. To produce the entailment, the proof has to increment the counter all the way to the maximum value, and do so on every branch of a binary tree, which gives us the desired lower bound.

Lemma 11. *One can construct a Horn-ALC-TBox $\mathcal{T}_{\mathcal{L},n}$ in time polynomial in n such that $\mathcal{T}_{\mathcal{L},n} \cup \{A(a)\} \models B(a)$, but every proof of the entailment is of (domain/tree) size doubly exponential in n .*

In the case of (domain) size, we can also find a matching upper bound. The general idea is using a kind of type construction. Intuitively, we identify the terms occurring in the proof based on the predicates they occur in. Because there are at most exponentially many possibilities for this, we can bound the nesting depth of Skolem terms by an exponential, which gives a double exponential bound on domain size and size. For tree size, this is not so straightforward, and we leave the exact bounds for future work.

Lemma 12. *For any CQ entailment $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$ with \mathcal{T} being a Horn-ALCHOI-TBox, there exists a proof of (domain) size double-exponential in \mathcal{T} and polynomial in \mathcal{A} .*

In contrast to Lemma 5 for *DL-Lite* and \mathcal{EL} , we cannot use Lemma 12 to reduce $\text{OP}_{\text{sk}}(\text{Horn-ALCHOI}, \mathbf{m})$ to query entailment in *Horn-ALCHOI* since a double exponential bound cannot be expressed using only polynomially many bits. On the positive side, the fact that the bound n is encoded in binary means that for $\text{OP}_{\text{sk}}(\text{Horn-ALCHOI}, \mathbf{m})$, we do not need to consider proofs of more than exponential size, which gives us a NEXPTIME upper bound for \mathbf{m}_s ; for \mathbf{m}_d it holds as well since there are exponentially many facts over $\text{sig}(\mathcal{T}^s \cup \mathcal{A})$ with a domain bounded by n . Using a technique from [5], we can even improve this to PSPACE in the case of \mathbf{m}_t .

Theorem 13. *$\text{OP}_{\text{sk}}(\text{Horn-ALCHOI}, \mathbf{m}_x)$ is in NEXPTIME for $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_d\}$, and in PSPACE for $\mathbf{m}_x = \mathbf{m}_t$.*

For \mathbf{m}_s , we are able to improve this complexity even further using a more involved technique. The idea is to virtually construct the proof from *proof segments* which are represented using tuples of the form $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$, where t is a term, IN and OUT are sets of atoms of restricted shape that may use a

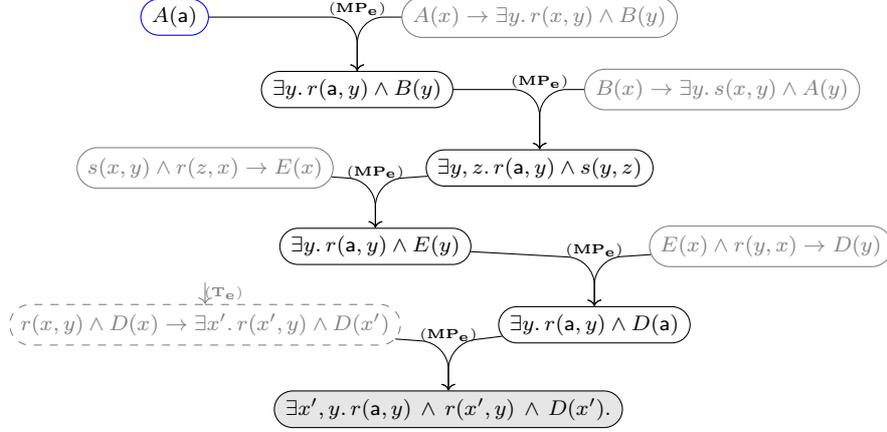


Figure 4: A CQ proof for the example

placeholder $_$ to represent *relative Skolem terms*, and SIZE is an integer. Intuitively, such a tuple tells us that it is possible to derive OUT from IN using a proof of size at most SIZE . t may optionally store what the placeholder $_$ stands for, provided that this is relevant for the query answer. We impose additional syntactic restrictions to ensure that there can be at most exponentially many such tuples. The decision procedure starts from a set of initial proof segments that correspond to proofs of polynomial size, and then step-wise aggregates proof segments to represent larger proofs, with the concise tuple representation making sure that there can be at most exponentially many such operations. We can thus prove the following theorem.

The main observation underlying this algorithm is that *Horn-ALCHOI* rules can only increase or decrease the nesting depth of a term by at most 1, while we can assume that (\mathbf{E}) only replaces terms by constants. This introduces a kind of locality to proofs that allows us to decompose proofs in the way that is required by our method. Since for logics with number restrictions (such as *Horn-ALCHOIQ*), this locality assumption failed, we did not consider such logics yet in our investigations.

Theorem 14. $\text{OP}_{\text{sk}}(\text{Horn-ALCHOI}, m_s)$ is in EXPTIME .

5 Directly Deriving CQs

In addition to connecting proofs to the universal model, \mathcal{D}_{sk} has the advantage that we can work with single atoms, which makes it easy to see how the existential rules are applied. However, the resulting proofs are not sound w.r.t. the ontology \mathcal{T} , but only w.r.t. the Skolemized rules \mathcal{T}^s . In order to be sound w.r.t. \mathcal{T} , inspired by [38, 20], we can work directly with Boolean CQs (see Fig. 4). Because these proofs do not work on universal models, and do not refer to introduced individuals directly, domain size is irrelevant in this setting, which is why we do not consider it here.

The corresponding inference schemas are shown in Fig. 5. Now, the basic inference (MP_e) matches the left-hand side of a rule in \mathcal{T} to part of a CQ

$$\boxed{
\begin{array}{c}
\frac{\exists \vec{x}. \phi(\vec{x}) \quad \psi(\vec{y}, \vec{z}) \rightarrow \exists \vec{u}. \chi(\vec{z}, \vec{u})}{\exists \vec{w}. \rho(\vec{w})} \text{ (MP}_e\text{)} \\
\\
\frac{}{\phi(\vec{x}, \vec{y}) \rightarrow \exists \vec{x}. \phi(\vec{x}, \vec{y})} \text{ (T}_e\text{)} \\
\\
\frac{\exists \vec{x}. \phi(\vec{x}) \wedge t_1 = t_2}{\exists \vec{x}. \phi(\vec{x})[t_1 \mapsto t_2]} \text{ (E}_e\text{)} \quad \frac{\exists \vec{x}. \phi(\vec{x}) \quad \exists \vec{y}. \psi(\vec{y})}{\exists \vec{x}, \vec{u}. \phi(\vec{x}) \wedge \psi(\vec{u})} \text{ (C}_e\text{)} \\
\\
\frac{\exists \vec{x}. \phi(\vec{x}, \vec{a})}{\exists \vec{x}, \vec{y}. \phi(\vec{x}, \vec{y})} \text{ (G}_e\text{)}
\end{array}
}$$

Figure 5: Inference schemas for \mathfrak{D}_{cq} .

and then replaces it by (part of) the right-hand side. Additionally, we allow to keep the replaced atoms from the original CQ. Again, (\mathbf{MP}_e) is admissible only if there exists a substitution π such that $\pi(\psi(\vec{y}, \vec{z})) \subseteq \phi(\vec{x})$, and then $\rho(\vec{w})$ is the result of replacing *any subset of* $\pi(\psi(\vec{y}, \vec{z}))$ in $\phi(\vec{x})$ by *any subset of* $\pi(\chi(\vec{z}, \vec{u}'))$, where the variables \vec{u} are renamed into new existentially quantified variables \vec{u}' to ensure that they are disjoint with \vec{x} . To duplicate variables, we introduce tautological rules such as $P(x, z) \rightarrow \exists z'. P(x, z')$ via (\mathbf{T}_e) , which yields $\exists z, z'. P(\mathbf{b}, z) \wedge P(\mathbf{b}, z')$ when combined with $\exists z. P(\mathbf{b}, z)$ using (\mathbf{MP}_e) . The remaining inference schemas are similar to the ones in \mathfrak{D}_{sk} , but not restricted to ground atoms. For (\mathbf{C}_e) , we rename the variables \vec{y} to \vec{u} to avoid overlap with \vec{x} .

Definition 2 (CQ Deriver). *The derivation structure $\mathfrak{D}_{cq}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ is defined similarly to \mathfrak{D}_{sk} , but using (\mathbf{MP}_e) , (\mathbf{T}_e) , (\mathbf{E}_e) , (\mathbf{C}_e) , and (\mathbf{G}_e) . We also define OP_{cq} analogously to OP_{sk} .*

Proofs obtained through \mathfrak{D}_{cq} are sound w.r.t. the original KB and do not depend on the notion of universal model. However, these proofs are more complex since vertices are not labeled with single atoms anymore, making it harder to understand how a rule is applied in case of an (\mathbf{MP}_e) inference. Indeed, verifying individual (\mathbf{MP}_e) steps is even NP-hard, since it requires to match one set of atoms into another, which is equivalent to database query answering [1]. This could potentially be solved by also showing the substitutions corresponding to these inference steps to the user, but this would lead to even more information being included in a single inference step. In general, we believe that except for the advantage of soundness, proofs based on CQs are less helpful for explaining query answers to users. In case users still prefer an inference system that is sound w.r.t. the original TBox rather than just the Skolemized version, we observe that it is not hard to translate proofs based on \mathfrak{D}_{sk} into proofs in \mathfrak{D}_{cq} and vice versa.

Theorem 15. *Any proof \mathcal{P} in $\mathfrak{D}_{cq}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ can be transformed into a proof in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ in time polynomial in the sizes of \mathcal{P} and \mathcal{T} , and conversely any proof \mathcal{P} in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ can be transformed into a proof in $\mathfrak{D}_{cq}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ in time polynomial in the sizes of \mathcal{P} and \mathcal{T} . The latter also holds for tree proofs.*

This theorem also shows that this deriver is complete for query entailment since we already know that \mathfrak{D}_{sk} is complete. However, it is not the case that

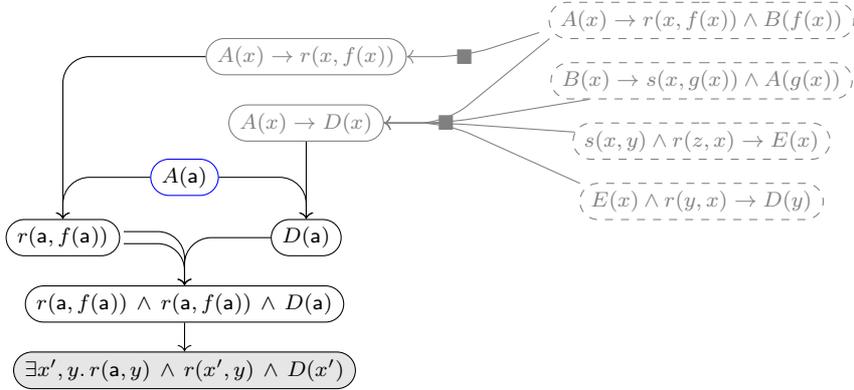


Figure 6: A Skolemized proof for the example with hidden TBox inferences

minimal proofs are equivalent for these two derivers, i.e. a minimal proof may become non-minimal after the transformation. Nevertheless, many of the results we have seen before also apply to \mathfrak{D}_{cq} (see the appendix and [6] for details). However, due to duplication of atoms via (\mathbf{T}_e) , some results can differ (cf. Theorem 9):

Theorem 16. *Let \mathcal{L} be an arbitrary DL. For tree-shaped CQs, $OP_{cq}(\mathcal{L}, \mathbf{m}_s)$ and $OP_{cq}(\mathcal{L}, \mathbf{m}_t)$ are NP-hard.*

6 Conclusion

We have presented a general framework for generating proofs for answers to ontology-mediated queries. The central idea is to explain the reasoning steps that contributed to the answer by referring to a universal model. We have also shown some initial complexity results, and intend to obtain a more precise picture in the future. An interesting future direction is to investigate derivers that combine TBox and query entailment rules, e.g. \mathfrak{D}_{sk} plus the rules of the ELK reasoner [26]. On one extreme end, one could completely hide all TBox reasoning steps, which could result in a proof like in Fig. 6, but it would be interesting to evaluate mixed proofs w.r.t. the comprehensibility of TBox- vs. query-based inferences. For explaining missing answers, we also want to investigate how to find (optimal) counter-interpretations or abduction results [27].

Acknowledgments This work was supported by the DFG grant 389792660 as part of TRR 248 – CPEC (<https://perspicuous-computing.science>), and QuantLA, GRK 1763 (<https://lat.inf.tu-dresden.de/quantla>).

References

- [1] Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases (1995)
- [2] Alrabbaa, C., Baader, F., Borgwardt, S., Dachselt, R., Koopmann, P., Méndez, J.: Evonne: Interactive proof visualization for description logics

- (System description). In: IJCAR (2022). https://doi.org/10.1007/978-3-031-10769-6_16
- [3] Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: Finding small proofs for description logic entailments: Theory and practice. In: LPAR-23 (2020). <https://doi.org/10.29007/nhpp>
- [4] Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: On the complexity of finding good proofs for description logic entailments. In: DL Workshop (2020), <http://ceur-ws.org/Vol-2663/paper-1.pdf>
- [5] Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: Finding good proofs for description logic entailments using recursive quality measures. In: CADE (2021). https://doi.org/10.1007/978-3-030-79876-5_17
- [6] Alrabbaa, C., Borgwardt, S., Koopmann, P., Kovtunova, A.: Finding good proofs for answers to conjunctive queries mediated by lightweight ontologies. In: DL Workshop (2022), to appear.
- [7] Alrabbaa, C., Hieke, W., Turhan, A.: Counter model transformation for explaining non-subsumption in EL. In: Workshop on Formal and Cognitive Reasoning (2021), http://ceur-ws.org/Vol-2961/paper_2.pdf
- [8] Baader, F., Horrocks, I., Lutz, C., Sattler, U.: An Introduction to Description Logic (2017). <https://doi.org/10.1017/9781139025355>
- [9] Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the description logic \mathcal{EL}^+ . In: Annual German Conference on AI (KI) (2007). https://doi.org/10.1007/978-3-540-74565-5_7
- [10] Borgida, A., Calvanese, D., Rodriguez-Muro, M.: Explanation in the *DL-Lite* family of description logics. In: On the Move to Meaningful Internet Systems: OTM (2008). https://doi.org/10.1007/978-3-540-88873-4_35
- [11] Borgida, A., Franconi, E., Horrocks, I.: Explaining \mathcal{ALC} subsumption. In: ECAI (2000), <http://www.frontiersinai.com/ecai/ecai2000/pdf/p0209.pdf>
- [12] Bourgaux, C., Ozaki, A., Peñaloza, R., Predoiu, L.: Provenance for the description logic elhr. In: IJCAI (2020). <https://doi.org/10.24963/ijcai.2020/258>
- [13] Calì, A., Gottlob, G., Lukasiewicz, T.: A general datalog-based framework for tractable query answering over ontologies. *J. Web Semant.* (2012). <https://doi.org/10.1016/j.websem.2012.03.001>
- [14] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. Autom. Reasoning* (2007). <https://doi.org/10.1007/s10817-007-9078-x>
- [15] Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. *Artif. Intell.* (2013). <https://doi.org/10.1016/j.artint.2012.10.003>

- [16] Calvanese, D., Lanti, D., Ozaki, A., Peñaloza, R., Xiao, G.: Enriching ontology-based data access with provenance. In: IJCAI (2019). <https://doi.org/10.24963/ijcai.2019/224>
- [17] Calvanese, D., Ortiz, M., Simkus, M., Stefanoni, G.: The complexity of explaining negative query answers in DL-Lite. In: KR (2012), <http://www.aaai.org/ocs/index.php/KR/KR12/paper/view/4537>
- [18] Carral, D., Dragoste, I., Krötzsch, M.: The combined approach to query answering in Horn-*ALCHOTQ*. In: KR (2018), <https://aaai.org/ocs/index.php/KR/KR18/paper/view/18076>
- [19] Ceylan, İ.İ., Lukasiewicz, T., Malizia, E., Molinaro, C., Vaicenavicius, A.: Explanations for negative query answers under existential rules. In: KR (2020). <https://doi.org/10.24963/kr.2020/23>
- [20] Croce, F., Lenzerini, M.: A framework for explaining query answers in DL-Lite. In: EKAW (2018). https://doi.org/10.1007/978-3-030-03667-6_6
- [21] Elhalawati, A., Krötzsch, M., Mennicke, S.: An existential rule framework for computing why-provenance on-demand for datalog. In: RuleML+RR (2022), to appear.
- [22] Gottlob, G., Kikot, S., Kontchakov, R., Podolskii, V.V., Schwentick, T., Zakharyashev, M.: The price of query rewriting in ontology-based data access. Artif. Intell. (2014). <https://doi.org/10.1016/j.artint.2014.04.004>
- [23] Horridge, M.: Justification Based Explanation in Ontologies. Ph.D. thesis, University of Manchester, UK (2011), https://www.research.manchester.ac.uk/portal/files/54511395/FULL_TEXT.PDF
- [24] Horridge, M., Parsia, B., Sattler, U.: Justification oriented proofs in OWL. In: ISWC (2010). https://doi.org/10.1007/978-3-642-17746-0_23
- [25] Kazakov, Y., Klinov, P., Stupnikov, A.: Towards reusable explanation services in protege. In: DL Workshop (2017), <http://www.ceur-ws.org/Vol-1879/paper31.pdf>
- [26] Kazakov, Y., Krötzsch, M., Simancik, F.: The incredible ELK - From polynomial procedures to efficient reasoning with \mathcal{EL} ontologies. J. Autom. Reason. (2014). <https://doi.org/10.1007/s10817-013-9296-3>
- [27] Koopmann, P.: Signature-based abduction with fresh individuals and complex concepts for description logics. In: IJCAI (2021). <https://doi.org/10.24963/ijcai.2021/266>
- [28] Krötzsch, M., Rudolph, S., Hitzler, P.: Complexities of horn description logics. ACM Trans. Comput. Logic (2013). <https://doi.org/10.1145/2422085.2422087>
- [29] Marnette, B.: Generalized schema-mappings: from termination to tractability. In: PODS (2009). <https://doi.org/10.1145/1559795.1559799>
- [30] McGuinness, D.L.: Explaining Reasoning in Description Logics. Ph.D. thesis, Rutgers University, USA (1996). <https://doi.org/10.7282/t3-q0c6-5305>

- [31] Nielsen, L.R., Andersen, K.A., Pretolani, D.: Finding the K shortest hyperpaths. *Computers & OR* (2005). <https://doi.org/10.1016/j.cor.2003.11.014>
- [32] Ortiz, M., Rudolph, S., Simkus, M.: Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In: *KR (2010)*, <http://aaai.org/ocs/index.php/KR/KR2010/paper/view/1296>
- [33] Ortiz, M., Rudolph, S., Simkus, M.: Query answering in the Horn fragments of the description logics \mathcal{SHOIQ} and \mathcal{SROIQ} . In: *IJCAI (2011)*. <https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-178>
- [34] Peñaloza, R.: Axiom-Pinpointing in Description Logics and Beyond. Ph.D. thesis, Technische Universität Dresden, Germany (2009), <https://nbn-resolving.org/urn:nbn:de:bsz:14-qucosa-24743>
- [35] Ramusat, Y., Maniu, S., Senellart, P.: Efficient provenance-aware querying of graph databases with datalog. In: *GRADES-NDA ACM Workshop (2022)*. <https://doi.org/10.1145/3534540.3534689>
- [36] Rosati, R.: On conjunctive query answering in EL. In: *DL Workshop (2007)*, http://ceur-ws.org/Vol-250/paper_83.pdf
- [37] Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: *IJCAI (2003)*, <http://ijcai.org/Proceedings/03/Papers/053.pdf>
- [38] Stefanoni, G.: Explaining query answers in lightweight ontologies. Diploma thesis, Technische Universität Wien, Austria (2011), <http://www.cs.ox.ac.uk/files/7942/thesis.pdf>

A Additional Definitions: Hypergraphs

Definition 3 (Hypergraph). A (finite, directed, labeled) hypergraph [31] is a triple $H = (V, E, \ell)$, where

- V is a finite set of vertices,
- E is a set of hyperedges (S, d) with a tuple of source vertices S and target vertex $d \in V$, and
- $\ell: V \rightarrow \mathcal{S}_{\mathcal{L}}$ is a labeling function that assigns sentences to vertices.

We extend the function ℓ to hyperedges as follows: $\ell(S, d) := ((\ell(s))_{s \in S}, \ell(d))$.

We assume that the *size* $|\eta|$ of an \mathcal{L} -sentence η is defined in some way, e.g. by the number of symbols in η . The *size* of H , denoted $|H|$, is measured by the size of the labels of its hyperedges:

$$|H| := \sum_{(S,d) \in E} |(S,d)|, \text{ where } |(S,d)| := |\ell(d)| + \sum_{v \in S} |\ell(v)|.$$

A vertex $v \in V$ is called a *leaf* if it has no incoming hyperedges, i.e. there is no $(S, v) \in E$; and v is a *sink* if it has no outgoing hyperedges, i.e. there is no $(S, d) \in E$ such that $v \in S$. We denote the set of all leafs and the set of all sinks in H as $\text{leaf}(H)$ and $\text{sink}(H)$, respectively.

A hypergraph $H' = (V', E', \ell')$ is called a *subgraph* of $H = (V, E, \ell)$ if $V' \subseteq V$, $E' \subseteq E$ and $\ell' = \ell|_{V'}$. In this case, we also say that H *contains* H' and write $H' \subseteq H$. Given two hypergraphs $H_1 = (V_1, E_1, \ell_1)$ and $H_2 = (V_2, E_2, \ell_2)$ s.t. $\ell_1(v) = \ell_2(v)$ for every $v \in V_1 \cap V_2$, the *union* of the two hypergraphs is defined as $H_1 \cup H_2 := (V_1 \cup V_2, E_1 \cup E_2, \ell_1 \cup \ell_2)$.

Definition 4 (Cycle, Tree). Given a hypergraph $H = (V, E, \ell)$ and $s, t \in V$, a path P of length $q \geq 0$ in H from s to t is a sequence of vertices and hyperedges

$$P = (d_0, i_1, (S_1, d_1), d_1, i_2, (S_2, d_2), \dots, d_{q-1}, i_q, (S_q, d_q), d_q),$$

where $d_0 = s$, $d_q = t$, and d_{j-1} occurs in S_j at the i_j -th position, for all j , $1 \leq j \leq q$. If there is such a path of length $q > 0$ in H , we say that t is reachable from s in H . If $t = s$, then P is called a cycle. The hypergraph H is acyclic if it does not contain a cycle. The hypergraph H is connected if every vertex is connected to every other vertex by a series of paths and reverse paths.

A hypergraph $H = (V, E, \ell)$ is called a tree with root $t \in V$ if t is reachable from every vertex $v \in V \setminus \{t\}$ by exactly one path. In particular, the root is the only sink in a tree, and all trees are acyclic and connected.

Definition 5 (Homomorphism). Let $H = (V, E, \ell)$, $H' = (V', E', \ell')$ be two hypergraphs. A homomorphism from H to H' , denoted $h: H \rightarrow H'$, is a mapping $h: V \rightarrow V'$ s.t. for all $(S, d) \in E$, one has $h(S, d) := ((h(v))_{v \in S}, h(d)) \in E'$ and, for all $v \in V$, it holds that $\ell'(h(v)) = \ell(v)$. Such an h is an isomorphism if it is a bijection, and its inverse, $h^{-1}: H' \rightarrow H$, is also a homomorphism.

Definition 6 (Hypergraph Unraveling). The unraveling of an acyclic hypergraph $H = (V, E, \ell)$ at a vertex $v \in V$ is the tree $H_T = (V_T, E_T, \ell_T)$, where V_T consists of all paths in H that end in v , E_T contains all hyperedges $((P_1, \dots, P_n), P)$

where each P_i is of the form $(d_i, i, ((d_1, \dots, d_n), d)) \cdot P$, and $\ell_T(P)$ is the label of the starting vertex of P in H . Moreover, the mapping $h_T: V_T \rightarrow V$ that maps each path to its starting vertex (and in particular $h_T(v) = v$) is a homomorphism from H_T to H .

Definition 7 (Derivation Structure). A derivation structure $\mathcal{D} = (V, E, \ell)$ over a theory \mathcal{U} is a directed, labeled hypergraph that is

- grounded, i.e. every leaf v in \mathcal{D} is labeled by $\ell(v) \in \mathcal{U}$; and
- sound, i.e. for all hyperedges $(S, d) \in E$, the entailment $\{\ell(s) \mid s \in S\} \models \ell(d)$ holds.

Definition 8 (Proof). Given a sentence η and a theory \mathcal{U} , a proof of $\mathcal{U} \models \eta$ is a derivation structure $\mathcal{P} = (V, E, \ell)$ over \mathcal{U} such that

- \mathcal{P} contains exactly one sink $v_\eta \in V$, which is labeled by η ,
- \mathcal{P} is acyclic, and
- every vertex has at most one incoming hyperedge, i.e. there exist no two hyperedges $(S_1, v), (S_2, v) \in E$ with $S_1 \neq S_2$.

A tree proof is a proof that is a tree. A subproof S of a hypergraph H is a subgraph of H that is a proof with $\text{leaf}(S) \subseteq \text{leaf}(H)$.

The tree size $\mathbf{m}_t(\mathcal{P})$ of a proof \mathcal{P} is defined recursively as follows:

$$\begin{aligned} \mathbf{m}_t(v) &:= 1 && \text{for every leaf } v, \\ \mathbf{m}_t(d) &:= 1 + \sum_{v \in S} \mathbf{m}_t(v), && \text{for every } (S, d) \in E, \\ \mathbf{m}_t(\mathcal{P}) &:= \mathbf{m}_t(v), && \text{for the sink vertex } v. \end{aligned}$$

This recursively counts the vertices in subproofs and is equal to the size of a tree unraveling of \mathcal{P} [5].

B Proof Details

B.1 The Data Complexity of Finding Good Proofs

Theorem 2. For a constant n , any \mathcal{L} , and any $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t, \mathbf{m}_d\}$, $\text{OP}_{\text{sk}}(\mathcal{L}, \mathbf{m}_x)$ is in AC^0 in data complexity.

Proof. We fix a set N_C^n of $\leq n$ individual names, and collect in \mathfrak{A} all of the (constantly many) ABoxes \mathcal{A} using only names from $\text{sig}(\mathcal{T}) \cup N_C^n$ for which $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ contains a proof \mathcal{P} with $\mathbf{m}(\mathcal{P}) \leq n$. The latter can be done, e.g. by breadth-first search for proofs up to (tree/domain) size n ; for domain size, observe that, for a given set of ground atoms S , the number of possible inference steps that do not produce any new Skolem terms is bounded by a function that depends only on S and \mathcal{T} . We can identify every such ABox \mathcal{A} with a CQ $q_{\mathcal{A}}$ obtained by replacing all individual names by existentially quantified variables. We now have that, for any ABox \mathcal{A} , there exists an $\mathcal{A}' \in \mathfrak{A}$ with $\mathcal{A} \models q_{\mathcal{A}'}$ iff there exists a proof in $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ with $\mathbf{m}(\mathcal{P}) \leq n$. Consequently, we can

reduce our decision problem to deciding whether \mathcal{A} entails the (fixed) *union of conjunctive queries* (UCQ)

$$\bigvee_{\mathcal{A} \in \mathfrak{A}} q_{\mathcal{A}}.$$

Deciding UCQ entailment without a TBox is possible in AC^0 in data complexity [1]. \square

Theorem 3. *For any $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t, \mathbf{m}_d\}$ and any \mathcal{L} such that all CQs are UCQ-rewritable over \mathcal{L} -TBoxes, $\text{OP}_{\text{sk}}(\mathcal{L}, \mathbf{m}_x)$ is in AC^0 in data complexity.*

Proof. Let Q be a UCQ that is a rewriting of \mathbf{q} over \mathcal{T} , i.e. such that $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$ is equivalent to $\mathcal{A} \models Q$ for all ABoxes \mathcal{A} . For every $q' \in Q$, we determine the minimal (tree/domain) size $n_{q'}$ for $\mathcal{T} \cup \mathcal{A}_{q'} \models \mathbf{q}$, where $\mathcal{A}_{q'}$ is obtained from q' by replacing every variable by a fresh individual name. These ABoxes represent all possibilities of an ABox entailing \mathbf{q} w.r.t. \mathcal{T} (modulo isomorphism), and hence they can be used as (a fixed number of) representatives in the search for small proofs of the entailment. The computation of $n_{q'}$ does not depend on the input data, and hence can be done offline via bounded search in the derivation structure. Let n_{\max} be the maximum of the values $n_{q'}$. To every $n \leq n_{\max}$, we assign the UCQ

$$Q_n = \bigvee \{q' \in Q \mid n_{q'} \leq n\}$$

Given \mathcal{A} and n , we can now decide whether $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ contains a proof \mathcal{P} with $\mathbf{m}(\mathcal{P}) \leq n$ by 1) computing the UCQ $Q_{n'}$ for $n' = \min(n, n_{\max})$, and 2) checking whether $\mathcal{A} \models Q_{n'}$. 2) is the standard query answering problem, which has AC^0 data complexity [1]. To see that the combined task 1)+2) can be done in AC^0 , we note that n_{\max} does not depend on the data, so that we only need to process the least $\log_{n_{\max}}$ bits of n to determine n' , which can be done by a circuit of constant depth. \square

B.2 Finding Good Proofs with Lightweight Ontologies

Lemma 4. *One can construct a TBox $\mathcal{T}_{\mathcal{L}, n}$ in time polynomial in n such that $\mathcal{T}_{\mathcal{L}, n} \cup \{A(a)\} \models B(a)$, but every proof of the entailment is of (domain/tree) size*

1. *polynomial in n for $\mathcal{L} = \text{DL-Lite}_R$,*
2. *exponential in n for $\mathcal{L} = \mathcal{EL}$.*

Moreover, there exists an \mathcal{EL} -TBox \mathcal{T} for which one can construct an ABox \mathcal{A}_n in time polynomial in n such that $\mathcal{T} \cup \mathcal{A}_n \models A(a)$, but every proof of it is of a tree size exponential in n .

Proof. For DL-Lite_R , this is trivial. For \mathcal{EL} , we define

$$\mathcal{T}_{n, \mathcal{EL}} = \left\{ \begin{array}{l} A \sqsubseteq A_1, \quad A_n \sqsubseteq B_n, \quad B_1 \sqsubseteq B \\ A_i \sqsubseteq \exists r.A_{i+1} \sqcap \exists s.A_{i+1}, \\ \exists r.B_{i+1} \sqcap \exists s.B_{i+1} \sqsubseteq B_i \quad \mid \quad 0 < i < n \end{array} \right\}.$$

The fragment of the universal model of $\mathcal{T}_{n, \mathcal{EL}}$ needed to derive $B(a)$ corresponds to a binary tree of depth n , and is thus exponentially large. This also gives a lower bound for the (tree) proof size. Note that $\mathcal{T}_{n, \mathcal{EL}}$ is not in normal form (cf.

Table 2), but it can be transformed into normal form without changing its size or the size of the resulting proofs more than polynomially.

For the second claim, we define

$$\mathcal{A}_n = \{A(a_0), r(a_1, a_0), s(a_1, a_0), \dots, r(a_n, a_{n-1}), s(a_n, a_{n-1})\},$$

where $a_n = a$, and $\mathcal{T} = \{\exists r.A \sqcap \exists s.A \sqsubseteq A\}$. Clearly, $\mathcal{T} \cup \mathcal{A}_n \models A(a)$, for which one has to prove each of $A(a_i)$, $i \in \{0, \dots, n\}$. Moreover, this relies twice on $A(a_{i-1})$, which means that the tree unravelling of the proof will be of size exponential in n . \square

Lemma 5. *For any CQ entailment $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$, there exists a proof of*

1. (domain/tree) size polynomial in $|\mathcal{T} \cup \mathcal{A}|$ and $|\mathbf{q}|$ if $\mathcal{L} = DL-Lite_R$,
2. (domain) size exponential in $|\mathcal{T}|$ and $|\mathbf{q}|$ and polynomial in $|\mathcal{A}|$ if $\mathcal{L} = \mathcal{EL}$,
3. tree size exponential in $|\mathcal{T} \cup \mathcal{A}|$ and $|\mathbf{q}|$ if $\mathcal{L} = \mathcal{EL}$.

Proof. For (domain) size, we can bound the number of relevant Skolem terms in \mathfrak{D}_{sk} by considering only the part of the minimal Herbrand model H that is necessary to satisfy the query \mathbf{q} . For example, in logics with the *polynomial witness property* [22], including $DL-Lite_R$, we know that any query that is entailed is already satisfied after polynomially many chase steps used to construct H .

For tree size, (C) and (G) only need to be applied once, at the very end, to produce the query \mathbf{q} . For $DL-Lite_R$, the remaining rule (MP) is such that it always has one premise that is a CQ. Consequently, we can always construct a proof that is composed of $|\mathbf{q}|$ linear proofs (one for each atom), which are then connected using (C) and produce the conclusion with (G). As argued above, we can assume that the nesting depth of Skolem terms is polynomially bounded by $|\mathcal{T}|$, which means the number of labels on each path is polynomially bounded by $|\mathcal{T}|$ as well. Additionally, we can always simplify any proof in which the same label occurs twice along some path, which means that this polynomial bound transfers also to the depth of our proof. We obtain that we can always find a proof of polynomial tree size.

For \mathcal{EL} , our proof is structured as follows: 1) we define a compressed derivation structure of size polynomial in $\mathcal{T} \cup \mathcal{A}$ and \mathbf{q} , similar as in the proof for Lemma 8, 2) we show that proofs in this structure can be translated into proofs in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ with the same tree size, and vice versa, and 3) we conclude that the nesting depth of Skolem terms in such proofs can be bounded polynomially and therefore the proof size is at most exponential.

We first consider only instance queries $\mathbf{q} = A(a)$. We replace every Skolem term $f(t)$ by a fresh individual name c_f (we ignore what is nested under the Skolem term). The resulting theory \mathcal{T}^c has no nested terms anymore. Our compressed derivation structure \mathcal{D} is now obtained from $\mathfrak{D}_{sk}(\mathcal{T}^c \cup \mathcal{A}, A(a))$ by dropping applications of (G) and (C), which are not needed since we want to derive a CQ with only one atom $A(a)$. \mathcal{D} contains only polynomially many nodes: one for each TBox axiom, and one for each term $A(c)$, $r(c, d)$ with c and d taken from our polynomially bounded set of individual names.

Now we show how proofs \mathcal{P} in the compressed derivation structure \mathcal{D} can be translated into proofs in $\mathfrak{D}_{sk}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ of the same tree size and vice versa. We recursively translate \mathcal{P} into a proof in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, A(a))$ by changing the terms

in the labels of the proof nodes, guided by the role atoms. Specifically, in an atom of the form $r(t, c_f)$, we know that c_f has to be replaced by $f(t)$. Moreover, any inference that had $r(t, c_f)$ as premise is now adapted such that in the other premises, c_f is also replaced by $f(t)$, as well as in all relevant predecessors of those premises. We repeat this step inductively until all fresh individual names are replaced by Skolem terms again. This will happen because 1) along every branch in the proof, any fresh individual name will have to be eliminated eventually, since the conclusion does not contain fresh individual names, 2) on the left-hand side of rules that correspond to \mathcal{EL} axioms, if a variable in an atom $A(x)$ occurs together with another variable, it must do so in an atom of the form $r(x, y)$. It follows that on every branch of the proof, atoms of the form $A(c_f)$ either produce other unary atoms with the same individual name, or eventually occur together with an atom of the form $r(t, c_f)$ in an inference, and in both cases our transformation will replace c_f by $f(t)$. It remains to replace the TBox rules in the proof by the original Skolemized version. The resulting tree-shaped proof is still sound, now a proof in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, A(a))$, and has the same tree size as \mathcal{P} . This transformation can also be easily done in the other direction (simply replacing any term $f(t)$ by c_f).

Now consider the general case where \mathbf{q} does not have to be an instance query. We note that with a little modification, we can also transform proofs in which the final conclusion contains the fresh individual names c_f : the translation procedure will then not succeed in replacing every fresh individual name based on the role atoms in the proof, but we can then check using the inferences introducing c_f which term would be appropriate, obtaining a proper proof in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, A(a))$.

It follows from our construction that one can always find a proof whose depth is polynomially bounded, since the compressed derivation structure is only of polynomial size. Consequently, the nesting depth of Skolem terms must be always polynomially bounded as well. We obtain the desired exponential bounds on (domain/tree) size for CQ proofs w.r.t. \mathcal{EL} TBoxes. Moreover, since the number of function symbols depends only on the TBox, we obtain that the number of Skolem terms is polynomial in the number of individual names occurring in \mathcal{A} , and thus polynomial in \mathcal{A} . Since this also bounds the number of atoms that can occur in a proof, we also obtain bound on the proof size that is polynomial w.r.t. \mathcal{A} . \square

Theorem 7. For $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_t\}$, $\text{OP}_{sk}(\text{DL-Lite}_R, \mathbf{m}_x)$ is NP-hard.

Proof. We reduce the problem of query answering to the given problems. Specifically, let $\mathcal{T} \cup \mathcal{A}$ be a DL-Lite KB, \mathbf{q} a query, and suppose we want to determine whether $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$. We know by Lemma 5 that, if $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$, then there exists a proof for this in $\mathfrak{D}_{sk}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ that is of (tree) size at most $n := p(|\mathcal{T}|, |\mathbf{q}|)$, where p is some polynomial.

We now construct a new KB $\mathcal{T}_0 \cup \mathcal{A}_0$ such that $\mathcal{T}_0 \cup \mathcal{A}_0 \models \mathbf{q}$, but only with a proof of (tree) size $> n$. \mathcal{A}_0 is obtained from the atoms of \mathbf{q} by replacing every quantified variable by a fresh individual name, and each predicate P by P_0 . \mathcal{T}_0 contains for every predicate P occurring in \mathbf{q} and every $i \in \{0, \dots, n\}$ the CI $P_i \sqsubseteq P_{i+1}$, as well as $P_n \sqsubseteq P$. Clearly, $\mathcal{T}_0 \cup \mathcal{A}_0 \models \mathbf{q}$, and every proof for this corresponds to a tree of depth $n + 1$, and is thus larger than n . Moreover, \mathcal{T}_0 , \mathcal{A}_0 , and n are all of polynomial size in the size of the input to the query

answering problem. Now set $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{T}$, $\mathcal{A}_1 = \mathcal{A}_0 \cup \mathcal{A}$. We have $\mathcal{T}_1 \cup \mathcal{A}_1 \models \mathbf{q}$; however, a proof of (tree) size $\leq n$ exists in $\mathfrak{D}_{sk}(\mathcal{T}_1 \cup \mathcal{A}_1, \mathbf{q})$ iff $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$. \square

Theorem 8. *Given a DL-Lite_R KB $\mathcal{T} \cup \mathcal{A}$ and a tree-shaped query \mathbf{q} , one can compute in polynomial time a proof of minimal tree size in $\mathfrak{D}_{sk}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$.*

Proof. We construct a compressed version of $\mathfrak{D}_{sk}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ of polynomial size. We introduce for every role R the individual name $b_{\exists R}$. Our compressed derivation structure is defined inductively as follows, where for a role name P , we identify $P^-(a, b)$ with $P(b, a)$ and $(P^-)^-$ with P .

- every axiom $\alpha \in \mathcal{T} \cup \mathcal{A}$ has a node v_α with $\ell(v_\alpha) = \alpha$,
- for nodes v_1, v_2 with $\ell(v_1) = A(a)$ and $\ell(v_2) = A \sqsubseteq B$ there is an edge $(\{v_1, v_2\}, v_3)$, where $\ell(v_3) = B(a)$,
- for nodes v_1, v_2 with $\ell(v_1) = P(a, b)$ and $\ell(v_2) = P \sqsubseteq Q$, there is an edge $(\{v_1, v_2\}, v_3)$ with $\ell(v_3) = Q(a, b)$,
- for nodes v_1, v_2 with $\ell(v_1) = A(a)$ and $\ell(v_2) = A \sqsubseteq \exists P$, there is an edge $(\{v_1, v_2\}, v_3)$ with $\ell(v_3) = P(a, b_{\exists P^-})$,
- for nodes v_1, v_2 with $\ell(v_1) = P(a, b)$ and $\ell(v_2) = \exists P \sqsubseteq A$, there is an edge $(\{v_1, v_2\}, v_3)$ with $\ell(v_3) = A(a)$,
- for nodes v_1, v_2 with $\ell(v_1) = P(a, b)$ and $\ell(v_2) = \exists P \sqsubseteq \exists Q$, there is an edge $(\{v_1, v_2\}, v_3)$ with $\ell(v_3) = Q(a, b_{\exists Q^-})$.

Due to the conclusions with the fresh individual names, the inferences in this compressed derivation structure are not sound, so that it is not really a derivation structure. But because its size is polynomial, we can use the P procedure from [5, Lemma 11] to compute for every node v a “proof” of minimal size. To use these proofs to construct a tree proof in $\mathfrak{D}_{sk}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$, we still need to match the variables in \mathbf{q} to the constants in the derivation structure.

For every pair $\langle t_1, t_2 \rangle$ of terms occurring in \mathbf{q} together in an atom, and every possible assignment $(t_1 \mapsto a_1, t_2 \mapsto a_2)$ of these terms to individual names from the compressed derivation structure, we assign a cost $\gamma(t_1 \mapsto a_1, t_2 \mapsto a_2)$ that is the sum of the minimal proof sizes for every atom in \mathbf{q} that contains only t_1 and t_2 , with these terms replaced using the assignment. We build a labeled graph, the *cost graph*, with every node a mapping from one term in \mathbf{q} to one constant in our compressed derivation structure, and every edge between two nodes $(t_1 \mapsto a_1)$ and $(t_2 \mapsto a_2)$ labeled with the cost $\gamma(t_1 \mapsto a_1, t_2 \mapsto a_2)$ (no edge if there is no edge between t_1 and t_2 in the Gaifman graph). Every edge in the cost graph corresponds to an edge in the Gaifman graph of \mathbf{q} , but the same edge in the Gaifman graph can be represented by several edges in the cost graph. We can thus transform the cost graph into a directed acyclic graph, making sure that for every edge $(t_1 \mapsto a_1, t_2 \mapsto a_2)$, the edge (t_1, t_2) points from the root towards the leafs of the Gaifman graph, where we choose an arbitrary node as the root.

We now eliminate assignments from the cost graph starting from the leafs:

- Consider a node $(t_1 \mapsto a_1)$ and for some term t_2 , all outgoing edges of the form $(t_1 \mapsto a_1, t_2 \mapsto a_2)$. Once all the nodes $t_2 \mapsto a_2$ have already

been visited by the algorithm, assign to each edge $(t_1 \mapsto a_1, t_2 \mapsto a_2)$ a combined cost obtained by adding to $\gamma(t_1 \mapsto a_1, t_2 \mapsto a_2)$ the costs of every edge reachable from $(t_2 \mapsto a_2)$, and remove all edges for which the resulting value is not minimal. In case several edges have a minimal value assigned, choose an arbitrary edge to remove, so that we obtain a unique edge $(t_1 \mapsto a_1, t_2 \mapsto a_2)$ for $t_1 \mapsto a_1$ and t_2 .

- If an assignment $(t \mapsto a)$ has no incoming edges, remove it from the cost graph.

The algorithm processes each edge in the polynomially sized cost graph at most once, and thus terminates after a polynomial number of steps with an assignment of variables to constants appearing in the compressed derivation structure. We can use this assignment to first construct a minimal proof for \mathbf{q} over the compressed derivation structure, where every atom in \mathbf{q} has its own independent minimal proof. Note that we cannot obtain a smaller proof in \mathbf{q} , since every atom has a minimal proof assigned, and our elimination procedure made sure that there cannot be a different choice of assignments of terms in \mathbf{q} to constants that would lead to smaller proofs anywhere else.

Finally, we substitute every constant with the corresponding Skolem term in $\mathfrak{D}_{\text{sk}}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$, starting from the original individual names and following the structure of the proof, to obtain the desired proof of minimal tree size. For example, we would replace an atom $P(a, b_{\exists P-})$ that is derived in the proof from $A(a)$ and $A \sqsubseteq \exists P$ by $P(a, f(a))$, where f is the Skolem function for the existentially quantified variable in $A \sqsubseteq \exists P$, and replace $b_{\exists P-}$ by $f(a)$ also in subsequent proof steps, provided it refers to a successor of a . More generally, we replace $P(t, b_{\exists P-})$, where t is already a Skolem term, by $P(t, f(t))$, whenever it is derived from $A(t)$ and $A \sqsubseteq \exists P$. \square

Theorem 9. *Let \mathcal{L} be an arbitrary DL and $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_d\}$. For tree-shaped CQs, $\text{OP}_{\text{sk}}(\mathcal{L}, \mathbf{m}_x)$ is NP-hard.*

Proof. We first reduce SAT to $\text{OP}_{\text{sk}}(\mathcal{L}, \mathbf{m}_s)$. Let c_1, \dots, c_m be a set of clauses over propositional variables p_1, \dots, p_k . Each clause c_i is a disjunction of literals of the form p_j or $\overline{p_j}$, where the latter denotes the negation of p_j . In the following, we assume clauses to be represented as sets of literals. W.l.o.g. we assume that for every variable p_i , we also have the clause $p_i \vee \overline{p_i}$. For every variable p_i , we add the facts $T(p_i)$ and $T(\overline{p_i})$ to the ABox \mathcal{A} . For every clause c_i and every literal $l_j \in c_i$, we add a fact $c(c_i, l_j)$. Furthermore, if $i < m$, we add the fact $r(c_i, c_{i+1})$. The conjunctive query \mathbf{q} is Boolean and contains for every clause c_i the atoms $c(x_i^{(c)}, x_i^{(p)})$, $T(x_i^{(p)})$ and moreover, if $i < m$, $r(x_i^{(c)}, x_{i+1}^{(c)})$, so that the query is tree-shaped. We have $\mathcal{A} \models \mathbf{q}$ independently of whether the SAT problem has a solution or not.

We set our bound as $n := 2 + m + (m - 1) + k$, which distributes as follows in a proof if the set of clauses is satisfiable. Assume that $a: \{p_1, \dots, p_k\} \rightarrow \{0, 1\}$ is a satisfying assignment for our set of clauses.

- 2 vertices are needed for the conclusions of **(C)** and **(G)**,
- m vertices contain, for each clause c_i , the atom $c(c_i, l)$, where $l \in c_i$ is made true by the assignment a ,

- $m - 1$ vertices contain the atom $r(c_i, c_{i+1})$ for each $i \in \{1, \dots, m - 1\}$,
- for each variable p_i , depending on whether $a(p_i) = 1$ or $a(p_i) = 0$, we use either $T(p_i)$ or $T(\bar{p}_i)$. This needs another k vertices.

There cannot be a smaller proof since every atom in the query needs to be matched by some ABox fact. Correspondingly, if the SAT problem has a solution, we can construct a proof of the desired size, and if there is a proof of the desired size, we can extract a solution for the SAT problem from it. Because the construction did not use a TBox, it follows that $\text{OP}_{\text{sk}}(\mathcal{L}, \mathbf{m}_s)$ is NP-hard for any logic DL.

For domain size, we can use the bound $n := m + k$ (covering the constants c_i , $1 \leq i \leq m$, and either p_i or \bar{p}_i for $1 \leq i \leq k$) to achieve the same result. \square

Theorem 10. $\text{OP}_{\text{sk}}(\mathcal{EL}, \mathbf{m}_t)$ is NP-complete in combined, and in P in data complexity. For IQs, the problem is P-complete in combined complexity.

Proof. We consider again the compressed derivation structure \mathcal{D} from the proof of Lemma 5. Recall that, for instance queries $A(x)$, a proof of tree size $\leq n$ exists in $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, A(a))$ iff such a proof exists in \mathcal{D} . By [3], one can construct a proof of minimal tree size in \mathcal{D} in deterministic polynomial time.

Now consider the general case where \mathbf{q} does not have to be an instance query. In this case, we additionally may need to replace fresh individual names c_f in the conclusion of a proof in \mathcal{D} by appropriate Skolem terms, by checking the inferences that introduced c_f . We use this in a non-deterministic decision procedure to check whether there exists a proof of tree size at most k . Specifically, we guess an assignment of individual names to each variable in \mathbf{q} (including both names from \mathcal{A} and the fresh ones from the compressed derivation structure). We now use the procedure from [3] to compute a proof of minimal tree size for every atom within the given tree size bound, under the guessed variable assignment, and connect those sub-proofs into the final proof candidate. If the resulting proof has a tree size $\leq n$, we accept. To obtain tractable data complexity, we adapt this procedure so that instead of guessing the assignment, we iterate over all the possibilities, which are polynomially many in the size of data and the TBox if the query is fixed.

For the NP lower bound, we recall from Lemma 5 that proof size is bounded exponentially, which means that the size can always be represented using polynomially many bits. We can use this in a construction similar as for Theorem 7 to reduce the Boolean query entailment problem for \mathcal{EL} to deciding the existence of a proof of bounded size. Let $\mathcal{T} \cup \mathcal{A}$ be an \mathcal{EL} KB, and \mathbf{q} Boolean query. W.l.o.g., we assume that \mathbf{q} contains at least one unary atom. Let furthermore 2^n be a bound on the tree size of a proof for $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$ (cf. Lemma 5). We construct $\mathcal{T}' \cup \mathcal{A}'$ and \mathbf{q}' s.t. $\mathcal{T}' \cup \mathcal{A}' \models \mathbf{q}$, but only with a proof of size less than 2^n if also $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$. For every variable x in \mathbf{q} , we add fresh individual names $a_{x,0}, \dots, a_{x,n}$ and a_x . \mathcal{A}' contains all facts of \mathcal{A} , and in addition for every atom $A(x) \in \mathbf{q}$ a sequence of facts $A_0(a_{x,0}), s(a_{x,1}, a_{x,0}), t(a_{x,1}, a_{x,0}), \dots, s(a_{x,n}, a_{x,n-1}), t(a_{x,n}, a_{x,n-1})$, together with $s_f(a_x, a_{x,n}), t_f(a_x, a_{x,n})$, with s, s_f, t, t_f and A_0 fresh. \mathcal{T}' contains all axioms of \mathcal{T} , plus additionally the axioms $\exists s.A_0 \sqcap \exists t.A_0 \sqsubseteq A_0, \exists s_f.A_0 \sqcap \exists t_f.A_0 \sqsubseteq A$. This ensures that we can infer $A(a_{x,n})$ for every $A(x) \in \mathbf{q}$, but only with a tree proof of size 2^n . For every binary atom $r(x, y) \in \mathbf{q}$, we add $r(a_x, a_y)$. We have $\mathcal{T}' \cup \mathcal{A}' \models \mathbf{q}$, but if

$\mathcal{T} \cup \mathcal{A} \not\models \mathbf{q}$, then every proof will be of tree size larger than 2^n . Since 2^n can be represented using n bits, our decision problem is at least as hard as Boolean query entailment, and thus NP-hard [36].

To show P-hardness for IQs, we observe that the same reduction is possible in LOGSPACE since n can be represented using logarithmically many bits, and computed using a working tape that is logarithmically bounded (for instance by overapproximating it based on the size of $\mathcal{T} \cup \mathcal{A}$). Consequently, 2^n , \mathcal{A}' , and \mathcal{T}' can be computed using a Turing machine with a working tape that is bounded logarithmically. The result then follows from P-hardness of instance query answering in \mathcal{EL} [15]. Regarding data complexity, we observe that \mathcal{T}' is constructed independent of \mathcal{A} and n , so that the same argument applies here. \square

Lemma 11. *One can construct a Horn- \mathcal{ALC} -TBox $\mathcal{T}_{\mathcal{L},n}$ in time polynomial in n such that $\mathcal{T}_{\mathcal{L},n} \cup \{A(a)\} \models B(a)$, but every proof of the entailment is of (domain/tree) size doubly exponential in n .*

Proof. For Horn- \mathcal{ALC} , we use a similar technique as for \mathcal{EL} , but this time using a binary counter enforcing a binary tree of depth 2^n . We use concepts $A_1, \bar{A}_1, \dots, A_n, \bar{A}_n$ as bits for the binary counter.

$$\begin{aligned} \mathcal{T}_{n,\text{Horn-}\mathcal{ALC}} = \{ & A \sqsubseteq \bar{A}_1 \sqcap \dots \sqcap \bar{A}_n, \\ & \exists r.A_i \sqsubseteq \forall r.A_i \sqcap \exists s.A_i \sqcap \forall s.A_i, \\ & \exists r.\bar{A}_i \sqsubseteq \forall r.\bar{A}_i \sqcap \exists s.\bar{A}_i \sqcap \forall s.\bar{A}_i, \\ & \bar{A}_i \sqcap A_{i-1} \sqcap \dots \sqcap A_0 \sqsubseteq \exists r.A_i, \\ & A_i \sqcap A_{i-1} \sqcap \dots \sqcap A_0 \sqsubseteq \exists r.\bar{A}_i, \\ & \bar{A}_i \sqcap (\bar{A}_{i-1} \sqcup \dots \sqcup \bar{A}_0) \sqsubseteq \exists r.\bar{A}_i, \\ & A_i \sqcap (\bar{A}_{i-1} \sqcup \dots \sqcup \bar{A}_0) \sqsubseteq \exists r.A_i, \\ & A_1 \sqcap \dots \sqcap A_n \sqsubseteq B, \\ & \exists r.B \sqcap \exists s.B \sqsubseteq B \quad | \quad 0 < i \leq n \} \end{aligned}$$

Again, this TBox is not in normal form, but it can be normalized with only a polynomial increase in size. \square

For the next results, we make w.l.o.g. an assumption on how **(E)** inferences are applied. In particular, we observe that the only equality atoms that we can derive are due to **(MP)** inferences with a rule of shape (vi). Consequently, all equalities we get are of the form $t = a$. In the following, we assume that in all proofs, when **(E)** is applied with such an equality, we replace t by a and not vice versa. This is without loss of generality as we can always transform the proof by replacing those two terms again. Since no rule has a Skolem function on the left, this does not change the applicability of the rules, so that the resulting graph is still a proof and of the same size. Moreover, we do not miss any smaller proofs in this way.

Using this assumption, we observe that binary atoms that can occur in a proof can only be of a restricted shape: namely, $R(t, f(t))$ or $R(t, a)$, where t is any ground term and R can be an inverse role. This is because binary atoms are only produced by **(MP)** inferences with a rule of shape (iv) or (vii), as well as by **(E)** inferences.

Lemma 12. *For any CQ entailment $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$ with \mathcal{T} being a Horn-ALC \mathcal{HOL} -TBox, there exists a proof of (domain) size double-exponential in \mathcal{T} and polynomial in \mathcal{A} .*

Proof. Let \mathcal{P} be some proof for $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$. It suffices to find a double exponential bound on the domain size, because this also bounds the set of possible labels with unary atoms double exponentially, and thus also the proof size. Let $\text{dom}(\mathcal{P})$ be the Skolem terms used in \mathcal{P} , and $\text{At}(\mathcal{P})$ be the set of atoms used in \mathcal{P} . To every $t \in \text{dom}(\mathcal{P})$, we assign a set $\text{At}(t, \mathcal{P})$ of non ground atoms, where $_$ is a special variable.

$$\begin{aligned} \text{At}(t, \mathcal{P}) = & \{A(_) \mid A(t) \in \text{At}(\mathcal{P})\} \cup \\ & \{R(_, f(_)) \mid R(t, f(t)) \in \text{At}(\mathcal{P})\} \cup \\ & \{A(f(_)) \mid A(f(t)) \in \text{At}(\mathcal{P})\} \\ & \{R(_, a) \mid R(t, a) \in \text{At}(\mathcal{P}), a \in \text{ind}(\mathcal{T})\} \end{aligned}$$

Where R can be a role name or an inverse role. New Skolem terms are only introduced via the Skolemized version of (iv), which produces atoms of the form $R(t, f(t))$ and $B(f(t))$. The only way to create a role atom of a different shape is using **(E)** with an equality atom $(f(t) = t')$. Those atoms are only produced by Rule (vi), which means that t' must then be of the form $a \in \text{ind}(\mathcal{T})$. As a result, we ensure that $\text{At}(t, \mathcal{P})$ refers to all atoms of the form $r(t, t')$ and for each such atom, all unary atoms $A(t')$ corresponding to t' provided t' is not an individual name.

Assume $\text{dom}(\mathcal{P})$ contains two terms t_1, t_2 such that $\text{At}(t_1, \mathcal{P}) = \text{At}(t_2, \mathcal{P})$, and t_1 occurs nested within t_2 . Inspection of the possible shapes (i)–(vii) of TBox rules indicates that any inference step that can be performed using any of the atoms in $\text{At}(t_2, \mathcal{P})$ can also be performed using any of the atoms in $\text{At}(t_1, \mathcal{P})$. Consequently, we can simplify the proof by 1) removing all vertices labeled with atoms that contain the term t_1 , 2) adding all vertices containing the term t_2 , as well as the edges between them, but changing their label by replacing t_2 by t_1 , and 3) adjusting the inference steps that relied on removed vertices, so that now they use one of the newly added vertices with the same labels. Note that since $\text{At}(t_1, \mathcal{P}) = \text{At}(t_2, \mathcal{P})$, this allows us to keep all inference steps relying on the term t_1 , so that the resulting graph is still a proof for $\mathcal{T} \cup \mathcal{A} \models \mathbf{q}$. By applying this operation repeatedly for any pair of terms t_1, t_2 such that t_1 occurs in t_2 and $\text{At}(t_1, \mathcal{P}) = \text{At}(t_2, \mathcal{P})$, we ensure that no such two terms occur in the final proof anymore. There can be at most exponentially many values for $\text{At}(t, \mathcal{P})$, which means that in the final proof, the nesting depth of every Skolem term is exponentially bounded, which bounds the overall number of Skolem terms to double exponential w.r.t. the size of $\mathcal{T} \cup \mathcal{A}$. Since the number of Skolem functions, as well as the elements in $\text{At}(t, \mathcal{P})$ are fully determined by \mathcal{T} , we furthermore obtain that the number of Skolem terms per individual name in \mathcal{A} depends only on \mathcal{T} , so that the size of the domain is polynomial in the size of the ABox. \square \square

Theorem 13. $\text{OP}_{\text{sk}}(\text{Horn-ALC}\mathcal{HOL}, \mathbf{m}_x)$ is in NEXPTIME for $\mathbf{m}_x \in \{\mathbf{m}_s, \mathbf{m}_d\}$, and in PSPACE for $\mathbf{m}_x = \mathbf{m}_t$.

Proof. The claims follow straightforwardly because we only need to guess a proof of size n — which takes non-deterministic exponential time. Since there

are exponentially many facts over $\text{sig}(\mathcal{T}^s \cup \mathcal{A})$ with a domain bounded by n , for \mathfrak{m}_d we can use a similar technique. For tree size, it suffices to observe that the procedure described in [5] to decide the existence of proofs of bounded tree size (Theorem 17) runs in space $p \cdot \log_2 n$, where p is the maximal number of premises in any inference. Correspondingly, this procedure would also run in PSPACE. \square

Before we prove Theorem 14, we show an intermediate result for a generalization of proofs. A *multi-proof* is defined like a proof, however

1. it can have more than one sink,
2. its leafs do not have to come from the KB \mathcal{K} , and
3. it must still be connected.

Intuitively, such a multi-proof can be a part of a larger proof. We have the following result regarding the atoms that can occur in a multi-proof for *Horn-ALC $\mathcal{H}\mathcal{O}\mathcal{I}$* -KBs.

Lemma 17. *Let \mathcal{K} be a Horn-ALC $\mathcal{H}\mathcal{O}\mathcal{I}$ KB, \mathbf{q} a query and \mathcal{P} a multi-proof in $\mathfrak{D}_{\text{sk}}(\mathcal{K}, \mathbf{q})$ such that*

1. \mathcal{P} contains at least one inference,
2. every leaf label that is an atom uses terms of nesting depth at most 1, and
3. apart from the leafs, every node label that is an atom uses a term of nesting depth ≥ 2 .

Then, there exists a term of the form $f(a)$ such that every leaf label that is an atom uses the term $f(a)$, and every node label that is an atom contains $f(a)$ nested within another Skolem term.

Proof. We prove this by induction on the number of inferences in \mathcal{P} .

If \mathcal{P} contains exactly one inference, we observe that the premises use only terms with nesting depth at most 1, while the conclusion must have a nesting depth ≥ 2 . It follows that the inference must be an **(MP)** inference with a rule of type (iv), since this is the only rule that produces a Skolem term of higher nesting depth (note also the observation regarding **(E)** applications before the proof of Lemma 12). Consequently, the premises must be of the form $A(f(a))$ and $A(x) \rightarrow R(x, g(x)) \wedge B(g(x))$, and the conclusion is either $R(f(a), g(f(a)))$ or $B(g(f(a)))$, which means that the claim is satisfied.

Assume that \mathcal{P} satisfies the inductive hypothesis. We consider possible extensions of \mathcal{P} . We can extend \mathcal{P} either by using an **(MP)** inference with a rule of type (i), (iv), (vi) or (vii) and a sink of \mathcal{P} , or we can extend \mathcal{P} using an **(E)** inference or an **(MP)** inference with a rule of type (ii), (iii), or (v), in which case we need to add another premise labeled with an atom, which could be from \mathcal{P} or from another proof that we connect to \mathcal{P} in this way.

In the case of using a rule of type (i), (iv), (vi) or (vii), we notice that the conclusions either use only terms from the premise, or in case of (iv) use a term that contains the a term from the premise nested. Consequently, it follows directly from the inductive hypothesis for \mathcal{P} that the extended proof satisfies the inductive hypothesis as well.

In case of using a rule of type (ii), we notice that the other premise must use the same term, and consequently the extended proof only satisfies the conditions of the lemma if the other premise is derived using a proof that already satisfies it. Specifically, we connect \mathcal{P} with another proof here, for which we can assume that the inductive hypothesis holds as well. Since the conclusion uses the same term as the premises, it follows that the inductive hypothesis applies to the extended proof as well.

Also for rules of type (iii) and (v) we may need to extend \mathcal{P} using another proof that satisfies the conditions of the lemma. Those rules are the only ones that can also decrease the nesting depth of Skolem terms. For (iii), this is the case if the premises are of the form $R(t, g(t))$, $A(g(t))$, $R(x, y) \wedge A(y) \rightarrow B(x)$ and the conclusion is of the form $B(t)$. If $t = f(a)$, then the conclusion has nesting depth 1, and thus the resulting proof will not satisfy the conditions of the lemma. Otherwise, by inductive hypothesis, t contains $f(a)$ nested, and the extended proof satisfies the conditions of the lemma. For (v), the interesting case is if the premises are of the form $A(g(t))$ and $R(g(t), t)$, and the conclusion is $B(t)$. Here, the situation is the same as for the rule of type (iii).

It remains to consider extensions using (E). Here we note that, by our assumption that we always replace complex terms by individual names, we always obtain a term that is of the form $a = b$, $A(a)$, $r(a, t)$ or $r(t, a)$. In the first two cases, we notice that the extended proof does not satisfy the preconditions of the lemma anymore. In the last two cases, if it does, then t must contain $f(a)$ nested by our inductive hypothesis.

We obtain that every multi-proof that satisfies the preconditions of the lemma also satisfies its postconditions. \square

Theorem 14. $OP_{sk}(Horn-ALCHOL, m_s)$ is in EXPTIME.

Proof. Let \mathbf{q} , n , \mathcal{T} and \mathcal{A} be the inputs to our decision problem. We assume that \mathbf{q} is connected since otherwise we could decide the problem by proving the connected components independently. We furthermore assume that the mapping of the quantified variables in the query to the terms from the universal model is already given: for this, it suffices to observe that, since the proof can only access Skolem terms of at most exponential nesting depth (in the bit length of the bound n on the proof size), and the number of variables is linearly bounded by the input, there are at most exponentially many options to try out, so that this can be easily incorporated into an EXPTIME procedure. Since \mathbf{q} is connected and the universal model is tree-shaped when restricted to Skolem terms of depth ≥ 1 , there exists a *root term* $t_{\mathbf{q}}$ such that every variable is matched to a term that contains either $t_{\mathbf{q}}$ or a constant a nested under at most $|\mathbf{q}|$ Skolem functions.

In the following, we fix $_$ to be a fresh constant. Let $T_{\mathbf{q}}$ be a set of *term patterns* that are obtained from the terms matched to the variables in the query by replacing the root term $t_{\mathbf{q}}$ by $_$, and closing $T_{\mathbf{q}}$ under the subterm relation. We note that $T_{\mathbf{q}}$ contains at most polynomially many elements since the number of terms matched by the query is linearly bounded, and each term has at most $|\mathbf{q}|$ subterms after replacing $t_{\mathbf{q}}$ by $_$.

The idea is to construct the minimal proof from *proof segments* which can be deterministically computed using an elimination procedure. A proof segment is a triple $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$ where t is a term, SIZE is an integer and IN and OUT

are sets of atoms using only terms of the form t' , $f(t')$, where $t' \in \text{ind}(\mathcal{T} \cup \mathcal{A}) \cup T_{\mathbf{q}}$ and f is a Skolem term occurring in \mathcal{T}_s . The intuitive meaning of such a proof segment is: “It is possible to derive $\text{OUT}[_ \mapsto t]$ from $\text{IN}[_ \mapsto t] \cup \mathcal{T}^s$ using at most SIZE proof vertices.” To obtain an EXPTIME procedure, we add an additional constraint on t : namely, t must be $_$ or a subterm of some term matched to the query. Note that there are at most exponentially many terms like that.

We call a proof segment *valid* if there is a multi-proof of size SIZE showing $\text{OUT}[_ \mapsto t]$ from $\text{IN}[_ \mapsto t] \cup \mathcal{T}^s$, and *directly valid* if there is such a multi-proof whose labels do not use Skolem terms of nesting depth larger than 1. Note that every directly valid proof segment is also valid. Direct validity of proof segments can be determined in exponential time: for this, we note that, since only Skolem terms of nesting depth 1 are considered, every directly valid proof segment has a witnessing multi-proof that has at most polynomially many nodes (one for each possible label). We can thus enumerate all possible graphs over these nodes in exponential time and check whether one of them corresponds to a multi-proof of size SIZE . Moreover, if we bound SIZE by the bound n on the proof size given as input, there are at most exponentially many possible proof segments. We can thus compute in exponential time the set of all directly valid proof segments for which $\text{SIZE} \leq n$.

We call a proof segment $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$ *initial* if $\text{IN} \subseteq \mathcal{A}$, and *final* if $\text{OUT}[_ \mapsto t]$ contains exactly the atoms of \mathbf{q} after applying our matching of the quantified variables. For our decision problem, we need to determine whether there is a proof segment that is initial, final, valid, and satisfies $\text{SIZE} + 2 \leq n$ (or $\text{SIZE} + 1 \leq n$ if \mathbf{q} contains only one atom), taking into account the additional inference steps with **(C)** and **(G)** at the end of the proof. For this, we use the following incremental procedure to construct new valid proof segments from existing ones. Let $\langle t_1, \text{IN}_1, \text{OUT}_1, \text{SIZE}_1 \rangle$ and $\langle t_2, \text{IN}_2, \text{OUT}_2, \text{SIZE}_2 \rangle$ be two proof segments. If $t_2 = f(t_1)$ or $t_2 = _$, and for the substitution $\sigma : _ \rightarrow f(_)$ we have $(\text{IN}_2)\sigma \subset \text{OUT}_1$, we define as a *down-merging* of $\langle t_1, \text{IN}_1, \text{OUT}_1, \text{SIZE}_1 \rangle$ and $\langle t_2, \text{IN}_2, \text{OUT}_2, \text{SIZE}_2 \rangle$ a proof segment

$$\langle t_1, \text{IN}_1, \text{OUT}_3, \text{SIZE}_1 + \text{SIZE}_2 - |\text{IN}_2| \rangle$$

where

$$\text{OUT}_3 \subseteq \text{OUT}_1 \cup \{ \alpha \sigma \mid \alpha \in \text{OUT}_2 \text{ contains only terms from } T_{\mathbf{q}} \cup N_1 \}.$$

Correspondingly, if $t_1 \in \{t_2, _ \}$ and $\text{IN}_2 \subseteq \text{OUT}_1$, we define as a *simple merging* of $\langle t_1, \text{IN}_1, \text{OUT}_1, \text{SIZE}_1 \rangle$ and $\langle t_2, \text{IN}_2, \text{OUT}_2, \text{SIZE}_2 \rangle$ a proof segment

$$\langle t_1, \text{IN}_1, \text{OUT}_3, \text{SIZE}_1 + \text{SIZE}_2 - |\text{IN}_2| \rangle$$

where $\text{OUT}_3 \subseteq \text{OUT}_1 \cup \text{OUT}_2$. Both the result of simple merging and of down-merging valid proof segments produce proof segments that are also valid: for this, note that we do not need to count the nodes corresponding to IN_2 , as they would correspond to nodes in OUT_1 in the corresponding merged proof. Our procedure now incrementally extends our initial set of directly valid proof segments by adding in each step all possible mergings of proof segments in our set, where we only keep those sets for which $\text{SIZE} \leq n$. This procedure reaches a fixpoint after at most exponentially many steps, as the set of possible proof segments is exponentially bounded. If this fixpoint contains a proof segment that is initial and final, we accept, and otherwise we reject.

Since our procedure only produces valid proof segments, and we only keep those for which SIZE is bounded by n , it is clearly sound, that is, it accepts only if there exists a proof of size at most n . To show that it is also complete, we need to prove the following claim:

Claim. For every proof segment $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$, if there exists a multi-proof \mathcal{P} of size SIZE with premises $\text{IN} \cup \mathcal{T}_s$ and conclusions OUT, then $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$ can be obtained through a sequence of merging operations starting from directly valid proof segments.

Proof of claim. Let \mathcal{P} be as in the claim. We prove the claim by induction over the maximal nesting depth of any term occurring in \mathcal{P} . If the nesting depth is 1, the claim directly holds since $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$ is already directly valid. Otherwise, assume that the claim holds for all proofs for which the maximal nesting depth of any term occurring is k .

We decompose \mathcal{P} into different, possibly overlapping, multi-proofs \mathcal{P}' that satisfy the following conditions:

1. \mathcal{P}' is connected,
2. either (a) all labels use only terms of nesting depth ≤ 1 , or (b) only the premises and the conclusions use terms of nesting depth ≤ 1 , and at least one label uses a higher nesting depth, and
3. there is no proper subproof of \mathcal{P}' that satisfies the above conditions.

If \mathcal{P}' satisfies 2(a), then it corresponds to a directly valid proof segment. If it satisfies 2(b), by Lemma 17, we obtain the following two properties:

- All leaves of \mathcal{P}' with atom labels use a term of the form $f(t)$.
- Every internal vertex in \mathcal{P}' with atom label must use a term that contains $f(t)$.

We obtain that \mathcal{P}' can be represented by a proof segment $\langle _, \text{IN}', \text{OUT}', \text{SIZE}' \rangle$, where SIZE' is the size of \mathcal{P}' , and IN' and OUT' are obtained from the premises and conclusions in \mathcal{P}' by replacing $f(t)$ by $_$. Since by this replacement, we would also reduce the nesting depth of \mathcal{P}' , we obtain that $\langle _, \text{IN}', \text{OUT}', \text{SIZE}' \rangle$ satisfies the claim by our inductive hypothesis.

We obtain that all multi-proofs \mathcal{P}' in our decomposition have a corresponding proof segment that satisfies our claim. Observing that the only possible overlaps between these multi-proofs are by the conclusions (which must be contained in the premises of another multi-proof or contribute to the final conclusion), we observe that the proof segment $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$ can be produced from these proof segments by applying down-merging and simple merging operations, so that finally, $\langle t, \text{IN}, \text{OUT}, \text{SIZE} \rangle$ satisfies the claim as well. ■

We obtain that our method is sound and complete, and runs in exponential time, which completes the proof. □

B.3 Directly Deriving CQs

Theorem 15. *Any proof \mathcal{P} in $\mathfrak{D}_{\text{cq}}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ can be transformed into a proof in $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ in time polynomial in the sizes of \mathcal{P} and \mathcal{T} , and conversely any proof \mathcal{P} in $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ can be transformed into a proof in $\mathfrak{D}_{\text{cq}}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$ in time polynomial in the sizes of \mathcal{P} and \mathcal{T} . The latter also holds for tree proofs.*

Proof. Assume that we have a proof \mathcal{P} in $\mathfrak{D}_{\text{cq}}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$. We first defer all applications of schema (\mathbf{G}_e) to the very end of the proof, which is possible since all other inferences remain applicable to any instance $\exists \vec{x}. \phi(\vec{x}, \vec{a})$ of a CQ $\exists \vec{x}, \vec{y}. \phi(\vec{x}, \vec{y})$. This transformation can only decrease the size of the proof. We then recursively change the labeling function so that it uses Skolem terms rather than quantified variables. Specifically, starting from the leafs of \mathcal{P} , we adapt inferences of schema (\mathbf{MP}_e) so that instead of a rule $\psi(\vec{y}, \vec{z}) \rightarrow \exists \vec{u}. \chi(\vec{z}, \vec{u}) \in \mathcal{T}$, the corresponding Skolemized version $\psi(\vec{y}, \vec{z}) \rightarrow \chi'(\vec{z}) \in \mathcal{T}^s$ is used, with existentially quantified variables in the conclusion replaced by the corresponding Skolem terms. The resulting hypergraph has the same size, since we only changed the labeling, and moreover all CQs are ground. In this process, whenever we apply a rule $\phi(\vec{x}, \vec{y}) \rightarrow \exists \vec{x}. \phi(\vec{x}, \vec{y})$ generated by (\mathbf{T}_e) to a ground CQ using (\mathbf{MP}_e) , we can omit this subproof since $\exists \vec{x}. \phi(\vec{x}, \vec{y})$ is satisfied by the same ground atoms used to match $\phi(\vec{x}, \vec{y})$. Next, we split each modified (\mathbf{MP}_e) inference into a corresponding set of (\mathbf{MP}) inferences – this replaces each vertex v by at most $|\mathcal{T}|$ vertices (at most for each new atom derived from the right-hand side of a rule in \mathcal{T}), and thus increases the size of the hypergraph by a factor polynomial in the size of \mathcal{P} . Even though \mathfrak{D}_{sk} contains no version of the (\mathbf{T}_e) inference schema to generate copies of atoms, we can always use the same ground atom several times in the same inference if necessary. At the same time, we remove all (\mathbf{C}_e) inferences (since (\mathbf{MP}) anyway uses multiple premises instead of a conjunction) and replace them by a single application of (\mathbf{C}) (conjoining all atoms relevant for $\mathbf{q}(\vec{a})$). This can also only decrease the size of the proof. Similarly, the final (\mathbf{G}_e) step becomes an instance of (\mathbf{G}) that produces the final conclusion \mathbf{q} . We obtain a proof in $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$ in time polynomial in the size of \mathcal{P} .

Now assume that we have a proof \mathcal{P} in $\mathfrak{D}_{\text{sk}}(\mathcal{T}^s \cup \mathcal{A}, \mathbf{q})$. We transform \mathcal{P} into a semi-linear proof in $\mathfrak{D}_{\text{sk}}(\mathcal{T} \cup \mathcal{A}, \mathbf{q})$. In the beginning, we keep the Skolemization, which is eliminated in the last step. We first collect any ABox facts from \mathcal{A} that are used in \mathcal{P} into a single CQ using (\mathbf{C}_e) . Then we aggregate inferences of (\mathbf{MP}) into (\mathbf{MP}_e) -inferences. Specifically, provided that for an (\mathbf{MP}) inference (S, d) , all labels of nodes in S occur on a node v that we have already aggregated, we collect all inferences of the form (S, d') (same premises, different conclusion), and transform them into a single inference using (a Skolemized version of) (\mathbf{MP}_e) . When S contains the same atom multiple times, we first generate an appropriate number of copies using (\mathbf{T}_e) and (\mathbf{MP}_e) ; the number of such additional proof steps for each (\mathbf{MP}_e) -inference is bounded by $|\mathcal{T}|$ (more precisely, the maximum number of atoms on the left-hand side of any rule in \mathcal{T}). Depending on which premises are still needed in later inferences, we keep them in the conclusion of each (\mathbf{MP}_e) -inference or not. Since we keep all atoms together in each step, the final application of (\mathbf{C}) is not needed anymore. The resulting proof is de-Skolemized by replacing Skolem terms by existentially quantified variables. This transformation is again polynomial in the size of the original proof: the number of initial applications of (\mathbf{C}_e) is bounded by the size of \mathcal{P} , the aggregated

(\mathbf{MP}_e)-steps can only decrease the size of the proof, and for each of these steps, we need at most $|\mathcal{T}|$ additional (\mathbf{T}_e)- and (\mathbf{MP}_e)-steps. Note also that this process always yields a tree-shaped proof. \square

Due to Theorem 15, many of the results we have seen before also apply to \mathfrak{D}_{cq} (except for domain size, which is not defined in this context). For example, the arguments in the proof of Theorem 2 apply in the same way here, i.e. one only has to consider constantly many ABoxes (modulo isomorphisms) to search for proofs below a given size bound, and similarly for Theorem 3. Lemma 5 also holds for \mathfrak{D}_{cq} because of Theorem 15, which gives us the same upper bounds as in Theorem 6. Moreover, Theorem 15 also ensures that the arguments for NP-hardness in Theorem 7 can be transferred to \mathfrak{D}_{cq} . We can similarly transfer the results of Lemma 12 and Theorem 13; however, the current proof of Theorem 14 works only for \mathfrak{D}_{sk} .

Due to duplication of atoms via (\mathbf{T}_e), Theorem 9 can also be shown for \mathfrak{D}_{cq} and \mathfrak{m}_t (and so Theorem 8 cannot hold for \mathfrak{D}_{cq}):

Theorem 16. *Let \mathcal{L} be an arbitrary DL. For tree-shaped CQs, $\text{OP}_{cq}(\mathcal{L}, \mathfrak{m}_s)$ and $\text{OP}_{cq}(\mathcal{L}, \mathfrak{m}_t)$ are NP-hard.*

Proof. We follow a similar approach as in Theorem 9. The central observation is that we can simulate in \mathfrak{D}_{cq} the behavior of (\mathbf{C}) by copying atoms. Specifically, fix an inference of (\mathbf{C}) with premises $\alpha_1(\vec{t}_1), \dots, \alpha_n(\vec{t}_n)$. In \mathfrak{D}_{sk} these atoms would not occur separately, but together as a non-ground CQ $\exists \vec{x}. \alpha_1(\vec{x}_1) \wedge \dots \wedge \alpha_n(\vec{x}_n)$. Now assume that $\alpha_1(\vec{t}_1) = \alpha_2(\vec{t}_2)$, in which case \mathfrak{D}_{sk} can use the same vertex as both the first and the second premise of the inference. In \mathfrak{D}_{cq} , this corresponds to only $\alpha_1(\vec{x}_1)$ occurring in the current CQ, although both $\alpha_1(\vec{x}_1)$ and $\alpha_2(\vec{x}_2)$ (using the same predicate, but different variables) are needed for a subsequent inference. To obtain a similar effect as in \mathfrak{D}_{sk} , we can first use (\mathbf{T}_e) to derive $\alpha_1(\vec{x}_1) \rightarrow \exists \vec{x}_1. \alpha_1(\vec{x}_1)$, which is then used with (\mathbf{MP}_e) and the current CQ $\phi(\vec{x})$ to obtain a query in which both $\alpha_1(\vec{x}_1)$ and $\alpha_2(\vec{x}_2)$ occur (recall that (\mathbf{MP}_e) can be used in such a way that the new atoms are not replacing others, but are simply added). This way, we can duplicate an arbitrary number of atoms and variables using two steps.

Let \mathcal{A} and \mathbf{q} be as in the proof of Theorem 9, and assume the SAT problem has a satisfying assignment a . We can then construct a tree proof as follows, where we collect the different atoms, one after the other, using (\mathbf{C}_e) into a single query. Since these inferences always have two premises, we only count the leaves in the following, as a binary tree with ℓ leaves and all other vertices binary always has $2\ell - 1$ vertices in total. Specifically, this means that the number of vertices is independent of how we organize the inferences listed in the following.

- We need to collect all clauses of the form $r(c_i, c_{i+1})$, and $c(c_i, l_j)$, where l_i is some literal in c_i evaluated to true under the chosen assignment a . This gives $2m - 1$ leaves.
- Another k leaves are needed to add, for each variable p_i , $T(p_i)$ if $a(p_i) = 1$, and $T(\bar{p}_i)$ if $a(p_i) = 0$.

Since this makes $2m - 1 + k$ leaves, we obtain that the corresponding proof must have $4m + 2k - 3$ vertices in total, independently of how we organize these

inferences. We instantiate (\mathbf{T}_e) with $\mathbf{q}(\vec{x}) \rightarrow \exists \vec{x}. \mathbf{q}(\vec{x})$, where $\mathbf{q}(\vec{x})$ is our query, containing as quantified variables exactly \vec{x} . Since the left-hand side matches our ground query constructed so far, we apply (\mathbf{MP}_e) as final step to produce the conclusion, obtaining a tree proof of size $n := 4m + 2k - 1$.

As in the proof of Theorem 9, we argue that there cannot possibly be a smaller proof for the query, since every leaf of this proof has to be used. Moreover, (\mathbf{G}_e) cannot be used to construct the final CQ since it cannot duplicate atoms and at least one atom of the form $T(p_i)$ or $T(\bar{p}_i)$ has to be used twice due to the additional clauses $p_i \vee \bar{p}_i$. Consequently, if we find a proof of tree size n , we can construct a satisfying assignment from it, and if there is a satisfying assignment, we can construct a proof of size n . It follows that $\text{OP}_{\text{cq}}(\emptyset, \mathbf{m}_t)$ must be NP-hard. The same arguments apply to \mathbf{m}_s since the proofs constructed above are tree-shaped and all vertices have different labels. \square