A Horn extension of DL-Lite with NL data complexity

Janos Arpasi[1], Bartosz Jan Bednarczyk[1,2] and Magdalena Ortiz[1]

[1]Institute of Logic and Computation, TU Wien
[2]Computer Science Department, University of Wrocław

Abstract

The literature on ontology-mediated query answering (OMQA) has been shaped by two key results: first-order rewritability for DL-Lite, and PTime-hardness of data complexity for essentially every description logic beyond it. This has effectively positioned DL-Lite as the only practical choice for query rewriting, restricting OMQA solutions to first-order queries and ontologies that can be rewritten into them. This AC_0 vs. PTime dichotomy is especially limiting if we consider that OMQA targets graph-structured data, and that standard graph query languages (including the recent ISO standards GQL and SQL/PGQ) are typically NL-complete. Towards identifying a rich Horn DL that can be rewritten into graph query languages and that can still express many $\mathcal{ELI}$ and DL-Lite ontologies, we introduce a stratification mechanism for $\mathcal{ELI}$ that controls the interaction between conjunction and recursion. In this way, we obtain $\mathcal{ELHI}^{\bot}_{\preceq}$, a description logic that strictly extends the core DL-Lite, supports reachability axioms and restricted conjunction, and allows for reasoning in NL. We establish the NL upper bound via a rewriting into nested two-way regular path queries, a fragment of GQL, providing initial evidence that our ontology language is a promising candidate for extending OMQA to graph query languages.

Keywords

complexity, query rewriting, data complexity, lightweight description logics

1. Introduction

One of the seminal results in the field of Ontology-Mediated Query Answering (OMQA) is the AC^0 data complexity of DL-Lite, which facilitates pure query rewritings into First-Order Logic queries [1]. Specifically, given an ontology-mediated query $(q, \mathcal{T})$, where q is a conjunctive query and $\mathcal{T}$ is a DL-Lite-TBox, one can derive an FO-query $q_{\mathcal{T}}$ (in particular, a union of conjunctive queries) such that the answers to $q_{\mathcal{T}}$ and $(q, \mathcal{T})$ coincide over any ABox $\mathcal{A}$. This result effectively reduces ontology-mediated query evaluation to the evaluation of standard database queries, bypassing the need for specialised reasoning engines at runtime. Equally influential is the observation that DL-Lite is essentially the only major family of description logics (DLs) admitting such FO-rewritings [2]. Even relatively inexpressive logics, such as $\mathcal{EL}$, are PTime-complete in data complexity [2]. Consequently, they cannot be rewritten into FO, nor into any query language with subpolynomial evaluation complexity. This delineates a long-standing barrier that has shaped much of the research landscape in OMQA for the past two decades.

However, this barrier may have been accepted too readily. There is a range of graph query languages that can express fundamental graph features such as reachability and path navigation, surpassing the expressive power of FO while maintaining data complexity in NL [3]. These languages have matured significantly over the past decade, culminating in the standardisation of GQL and its SQL-inspired counterpart, SQL/PGQ, in 2023 [4]. These languages are obvious candidates for OMQA, making it a true graph data access paradigm and opening up the possibility of finally moving beyond DL-Lite and FO-rewritability. The foundations of ontology-mediated queries based on graph query language—specifically, conjunctive regular path queries (CRPQs) and their variants— were laid over a decade ago, and the landscape of their computational complexity was almost fully charted [5, 6, 7, 8]. Yet these results never led to the attempted adoption of ontology-mediated graph queries in practical systems. Furthermore, none of the mentioned works addressed the fundamental tension between the limited

DL 2026: 39th International Workshop on Description Logics, July 17–19, 2026, Lisbon, Portugal

janos.arpasi@tuwien.ac.at (J. Arpasi); bartek@cs.uni.wroc.pl (B. J. Bednarczyk); magdalena.ortiz@tuwien.ac.at (M. Ortiz)
https://bartoszjanbednarczyk.github.io (B. J. Bednarczyk)
0000-0002-8267-7554 (B. J. Bednarczyk); 0000-0002-2344-9658 (M. Ortiz)

expressivity of DL-Lite and the loss of rewritability in more expressive settings.

Recent advances in graph query languages offer new momentum to overcome this impasse. Finally, state-of-the-art graph query engines (like Neo4j and its query language Cypher [9]) are moving towards full support of the navigational core of GQL and SQL/PGQ,[1] which supports all the extensions of CRPQs considered so far in the OMQA literature, including *nested 2-way conjunctive regular path queries (N2RPQs)*, their most expressive variant [10]. We now see a promising path towards the ultimate goal of *designing useful, expressive yet practicable ontology-mediated graph query languages* that allow us to leverage existing ontologies in the OWL profiles and to pair them with the flexible navigational features of graph query languages.

In this paper, we address a specific aspect of this goal that is surprisingly underexplored in the literature: the identification of Horn DLs that strictly extend DL-Lite and admit rewritings into graph query languages. Largely discouraged by the early negative results—like the mentioned PTime-hardness of plain $\mathcal{EL}$, and the same lower bound for any DL supporting qualified existential restrictions in the left-hand side of axioms and inverse roles [2]—hardly any research effort has been invested into cutting out meaningful description logics that admit rewritings into CRPQs and their extensions. To our knowledge, there are only two partial exceptions. *Harmless linear* $\mathcal{ELHI}$ [11] disallows the use of concept conjunction in the left-hand side of axioms, essentially restricting $\mathcal{ELHI}$ to axioms of the forms $\mathrm{A} \sqsubseteq \mathrm{B}$, $\mathrm{A} \sqsubseteq \exists s.\mathrm{B}$ and $\exists s.\mathrm{A} \sqsubseteq \mathrm{B}$, with s a possibly inverse role, and imposing an additional *harmlessness* constraint that restricts the interaction of inverses and qualified existential restrictions to keep the complexity in NL. The resulting description logic admits rewriting of atomic queries into a simple class of CRPQs, but the lack of conjunction makes it very limited as a Horn DL. Motivated by the central role of conjunction in many real-world ontologies, quasi-linear $\mathcal{ELHI}$ takes a slightly different approach, enabling restricted use of conjunction at the expense of additional restrictions on the use of inverse roles[12]. The resulting ontology language appears to be useful in practice, but it is not a proper extension of DL-Lite. We aim to push the boundary of description logics that admit reasoning in NL and rewritings into graph query languages, extending DL-Lite without compromising too much of the additional expressiveness of $\mathcal{ELHI}$. In particular, instead of disallowing the use of the conjunction, we leverage the expressiveness of N2RPQs to support it in a controlled way.

In this paper, we propose $\mathcal{ELHI}^{\bot}_{\sqsubseteq}$ as the first NL-data-complete ontology language that fully subsumes DL-Lite and that supports conjunction and reachability axioms of the form $\exists r.\mathrm{A} \sqsubseteq \mathrm{A}$. To maintain NL data complexity, $\mathcal{ELHI}^{\bot}_{\sqsubseteq}$ utilises a stratification of concept names that guarantees a bound on the alternation induced by the interaction of reachability, inverses, and conjunction, preventing the complexity from escalating to PTime. We demonstrate NL-membership for instance checking via a rewriting into N2RPQs. Finally, we show that while the data complexity is low, the combined complexity of $\mathcal{ELHI}^{\bot}_{\sqsubseteq}$ is PSpace-complete; this is consistent in spirit with the restriction of ExpTime logics to linear recursion [13], and witnesses the additional expressive power of our fragment in comparison to DL-Lite, $\mathcal{EL}$, and harmless linear $\mathcal{ELHI}$, all of which allow for reasoning in polynomial time.

All missing proofs are given in the appendix of the extended version of this paper [14].

2. Preliminaries

We start by recalling the basics on description logics (DLs) [15] and query answering [16].

DLs. We fix countably infinite pairwise disjoint sets of *individual names* $\mathbf{N_I}$, *concept names* $\mathbf{N_C}$, and *role names* $\mathbf{N_R}$ and introduce the description logic $\mathcal{ELHI}^{\bot}$. Starting from $\mathbf{N_C}$ and $\mathbf{N_R}$, the set $\mathbf{C}_{\mathcal{ELHI}^{\bot}}$ of $\mathcal{ELHI}^{\bot}$*-concepts* is built using the following concept constructors: *conjunction* $(\mathrm{C} \sqcap \mathrm{D})$, *existential restriction* $(\exists r.\mathrm{C})$ and *bottom* and *top concepts* $(\bot, \top)$, with the following grammar:

$$\mathrm{C}, \mathrm{D} ::= \bot \mid \top \mid \mathrm{A} \mid \mathrm{C} \sqcap \mathrm{D} \mid \exists r.\mathrm{C} \mid \exists \bar{r}.\mathrm{C},$$

[1] https://neo4j.com/docs/cypher-manual/current/appendix/gql-conformance/

where C, D are $\mathbf{C}_{\mathcal{ELHI}^\bot}$-concepts, $\mathrm{A} \in \mathbf{N_C}$ is a concept name, $r \in \mathbf{N_R}$ is a role name, and $\bar{r}$ denotes its inverse. We will always assume that the inverse of $\bar{r}$ is just r, and use (possibly decorated) letters s to denote possibly inverted roles.

KBs. *Assertions* take the form $\mathrm{C}(\mathrm{a})$, $s(\mathrm{a},\mathrm{b})$ for $\mathrm{a},\mathrm{b} \in \mathbf{N_I}$, $\mathrm{C} \in \mathbf{N_C} \cup \{\top, \bot\}$ and (possibly inverted) role s. A *general concept inclusion* (GCI) has the form $\mathrm{C} \sqsubseteq \mathrm{D}$ for concepts C, D. We employ $\mathrm{C} \equiv \mathrm{D}$ as a shorthand for the two GCIs $\mathrm{C} \sqsubseteq \mathrm{D}$ and $\mathrm{D} \sqsubseteq \mathrm{C}$. A *knowledge base* (KB) $\mathcal{K} \coloneqq (\mathcal{A}, \mathcal{T})$ is composed of a finite non-empty set $\mathcal{A}$ (*ABox*) of assertions and a finite non-empty set $\mathcal{T}$ (*TBox*) of GCIs. The elements of $\mathcal{A} \cup \mathcal{T}$ are called *axioms*. Let $\mathsf{ind}(\mathcal{K})$, $\mathsf{con}(\mathcal{K})$, $\mathsf{rol}(\mathcal{K})$ denote the sets of all individual, concept, and role names from $\mathcal{K}$, respectively (including $\top, \bot$ in $\mathsf{con}(\mathcal{K})$ and role inverses in $\mathsf{rol}(\mathcal{K})$). We use the analogous notation for TBoxes and ABoxes as well. A KB $\mathcal{K}$ is in *normal form* if all its GCIs conform to the following pattern:

$$\mathrm{A} \sqsubseteq \mathrm{B}, \quad \mathrm{A} \sqcap \mathrm{B} \sqsubseteq \mathrm{C}, \quad \mathrm{A} \sqsubseteq \exists s.\mathrm{B}, \quad \exists s.\mathrm{A} \sqsubseteq \mathrm{B}, \quad s \subseteq s'$$

where $\mathrm{A}, \mathrm{B}, \mathrm{C}$ are either concept names, $\top$, or $\bot$, while s, s' are possibly inverted roles. Without loss of generality, we also require that $\top$ and $\bot$ do not appear in GCIs of the form $\mathrm{A} \sqcap \mathrm{B} \sqsubseteq \mathrm{C}$, and that trivially unsatisfiable formulae like $\exists s.\bot$ are always replaced with $\bot$. Given a TBox $\mathcal{T}$ we write $s \subseteq^+_{\mathcal{T}} s'$ if there exists a sequence of (possibly inverted) roles $s_0, \ldots, s_n$ such that $s_0 = s$, $s_n = s'$, and for all $i < n$ we have $s_i \subseteq s_{i+1} \in \mathcal{T}$.

Table 1
Concepts and roles in $\mathcal{ELHI}^\bot$.

Name	Syntax	Semantics
bottom concept	$\bot$	$\emptyset$
top concept	$\top$	$\Delta^{\mathcal{I}}$
concept name	A	$\mathrm{A}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
role name	r	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
inverse role	$\bar{r}$	$\{(\mathrm{e},\mathrm{d}) \mid (\mathrm{d},\mathrm{e}) \in r^{\mathcal{I}}\}$
conc. negation	$\neg\mathrm{C}$	$\Delta^{\mathcal{I}} \setminus \mathrm{C}^{\mathcal{I}}$
conc. intersection	$\mathrm{C} \sqcap \mathrm{D}$	$\mathrm{C}^{\mathcal{I}} \cap \mathrm{D}^{\mathcal{I}}$
exist. restriction	$\exists s.\mathrm{C}$	$\{\mathrm{d} \mid \exists \mathrm{e}.(\mathrm{d},\mathrm{e}) \in s^{\mathcal{I}} \wedge \mathrm{e} \in \mathrm{C}^{\mathcal{I}}\}$

Table 2
Axioms in $\mathcal{ELHI}^\bot$.

Axiom α	$\mathcal{I} \models \alpha$, if	
$\mathrm{C} \sqsubseteq \mathrm{D}$	$\mathrm{C}^{\mathcal{I}} \subseteq \mathrm{D}^{\mathcal{I}}$	TBox $\mathcal{T}$
$s \subseteq s'$	$s^{\mathcal{I}} \subseteq s'^{\mathcal{I}}$	
$\mathrm{C}(\mathrm{a})$	$\mathrm{a}^{\mathcal{I}} \in \mathrm{C}^{\mathcal{I}}$	ABox $\mathcal{A}$
$r(\mathrm{a},\mathrm{b})$	$(\mathrm{a}^{\mathcal{I}}, \mathrm{b}^{\mathcal{I}}) \in r^{\mathcal{I}}$	

The semantics of $\mathcal{ELI}^\bot$ is defined via *interpretations* $\mathcal{I} \coloneqq (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ composed of a non-empty set $\Delta^{\mathcal{I}}$ called the *domain of* $\mathcal{I}$ and an *interpretation function* $\cdot^{\mathcal{I}}$ mapping individual names to elements of $\Delta^{\mathcal{I}}$, concept names to subsets of $\Delta^{\mathcal{I}}$, and role names to subsets of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. This mapping is extended to complex concepts (see Table 1) and finally used to define *satisfaction* of assertions and GCIs (see Table 2). *Structures* are interpretations with a partial assignment of individual names. We say that an interpretation $\mathcal{I}$ *satisfies* a KB $\mathcal{K} \coloneqq (\mathcal{A}, \mathcal{T})$ (or $\mathcal{I}$ is a *model* of $\mathcal{K}$, written: $\mathcal{I} \models \mathcal{K}$) if it satisfies all axioms of $\mathcal{A} \cup \mathcal{T}$. A KB is *consistent* (or *satisfiable*) if it has a model and *inconsistent* (or *unsatisfiable*) otherwise.

Automata and paths. By a *nondeterministic automaton* (NFA) $\mathcal{A}$ we understand, as usual, a tuple $(Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, Σ is a finite alphabet, $\delta \subseteq Q \times \Sigma \times Q$ is the transition relation, $q_0 \in Q$ is the initial state and $F \subseteq Q$ is the set of accepting states. All NFA considered in this paper are over finite subsets of $\{r, \bar{r}, \mathrm{C}? \mid r \in \mathbf{N_R}, \mathrm{C} \in \mathbf{N_C}\}$, where the expressions of the form C? are called *tests*. For a natural number n we also define *n-nested NFA* (nNFA) by induction: a 0NFA is just an NFA, and an $(n{+}1)$NFA is an NFA whose alphabet may contain letters $\mathcal{A}?$ for $\mathcal{A}$ being an nNFA.

By a *run* of an NFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ over an interpretation $\mathcal{I}$, we understand a finite sequence of tuples $(\mathrm{d}_0, q_0, w_1, q_1, \mathrm{d}_1), \ldots, (\mathrm{d}_{\ell-1}, q_{\ell-1}, w_\ell, q_\ell, \mathrm{d}_\ell)$ in $\Delta^{\mathcal{I}} \times \delta \times \Delta^{\mathcal{I}}$ such that for all $i < \ell$ we have:

(i) if $w_{i+1} = r$, then $(\mathrm{d}_i, \mathrm{d}_{i+1}) \in r^{\mathcal{I}}$,

(ii) if $w_{i+1} = \bar{r}$, then $(\mathrm{d}_{i+1}, \mathrm{d}_i) \in r^{\mathcal{I}}$, and

(iii) if $w_{i+1} = \mathrm{C}?$ for a concept name C, then $\mathrm{d}_i = \mathrm{d}_{i+1}$ and $\mathrm{d}_i \in \mathrm{C}^{\mathcal{I}}$.

A run is *accepting* if $q_\ell \in F$. We say that a run *starts from* d if $\mathrm{d}_0 = \mathrm{d}$. We define *runs* of an nNFA for all $n > 0$ analogously, adding the following extra condition:

(iv) if $w_{i+1} = \mathcal{B}?$ for an $(n-1)$NFA $\mathcal{B}$, then $\mathrm{d}_i = \mathrm{d}_{i+1}$ and there is an accepting run of $\mathcal{B}$ starting from d_i.

Queries. An *instance query* (IQ) q is an expression of the form $\mathrm{C}(\mathrm{a})$, where a is an individual name and C is a concept name. An interpretation $\mathcal{I}$ satisfies such a query (written $\mathcal{I} \models q$) if $\mathrm{a}^{\mathcal{I}} \in \mathrm{C}^{\mathcal{I}}$. A *nested two-way regular path query* (N2RPQ) q is an expression of the form $\exists x \mathcal{A}(\mathrm{a}, x)$ for an individual name a and an nNFA $\mathcal{A}$. An interpretation $\mathcal{I}$ satisfies such a query (written $\mathcal{I} \models q$) if there is an accepting run of $\mathcal{A}$ in $\mathcal{I}$ starting from $\mathrm{a}^{\mathcal{I}}$. In the *query entailment problem*, we are given a KB $\mathcal{K} \coloneqq (\mathcal{A}, \mathcal{T})$ and a query q and ask if $\mathcal{K} \models q$, i.e. whether every model of $\mathcal{K}$ satisfies q. Here we are interested in computational complexity of this problem. If all of $\mathcal{A}$, $\mathcal{T}$ and q are part of the input, we talk about *combined complexity*. If only $\mathcal{A}$ is part of the input, while $\mathcal{T}$ and q are fixed beforehand, we talk about *data complexity*. Given a query language $\mathcal{L}$ (say, first-order logic or N2RPQ) say that a TBox $\mathcal{T}$ is *$\mathcal{L}$-rewritable* if for every instance query q there is a query q' in $\mathcal{L}$ such that for all ABoxes $\mathcal{A}$ and all $\mathrm{a} \in \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{A}, \mathcal{T}) \models q(\mathrm{a})$ if and only if $\mathcal{A} \models q'(\mathrm{a})$. Briefly speaking, the query entailment problem over KBs with *$\mathcal{L}$-rewritable* TBoxes boils down to just evaluating $\mathcal{L}$-queries over ABoxes.

3. Our Logic and the Rewriting

In this section, we introduce $\mathcal{ELHI}^{\perp}_{\preceq}$ and present the rewriting of instance queries into N2RPQs.

Definition 3.1. *We say that a KB $\mathcal{K} = (\mathcal{A}, \mathcal{T})$ in normal form is* stratified *whenever there exists a preorder $\preceq$ on $\mathsf{con}(\mathcal{K}) \cup \mathsf{rol}(\mathcal{K})$ satisfying all the following conditions (where $\prec$ denotes the strict part of $\preceq$, and $\mathrm{A}, \mathrm{B}, \mathrm{C}$ are concept names, and D stands for a concept name or $\top$):*

- *Whenever $(\mathrm{A} \sqsubseteq \mathrm{B}) \in \mathcal{T}$, then $\mathrm{A} \preceq \mathrm{B}$.*
- *Whenever $(\mathrm{A} \sqcap \mathrm{B} \sqsubseteq \mathrm{C}) \in \mathcal{T}$, then $\mathrm{A} \preceq \mathrm{C}$, $\mathrm{B} \preceq \mathrm{C}$, and at least one of $\mathrm{A} \prec \mathrm{C}$ or $\mathrm{B} \prec \mathrm{C}$ holds.*
- *Whenever $(\mathrm{A} \sqsubseteq \exists s.\mathrm{D}) \in \mathcal{T}$, then $\mathrm{A} \preceq \mathrm{D}$ or $\mathrm{D} = \top$; and $\mathrm{A} \preceq s$.*
- *Whenever $(\exists s.\mathrm{D} \sqsubseteq \mathrm{B}) \in \mathcal{T}$, then either $\mathrm{D} \prec \mathrm{B}$, $\mathrm{D} = \mathrm{B}$, or $\mathrm{D} = \top$; and $s \preceq \mathrm{D}$.*
- *Whenever $(r \subseteq s) \in \mathcal{T}$, then $r \preceq s$.*
- *We assume that $r \preceq \overline{r}$ and $\overline{r} \preceq r$ for all role names r.*

By $\mathcal{ELHI}^{\perp}_{\preceq}$ we denote the fragment of $\mathcal{ELHI}^{\perp}$ whose KBs are in normal form and stratified. ◀

Note that the above conditions on $\preceq$ do not apply to GCIs with $\perp$ on the right-hand side, and hence disjointness of concepts is not subject to any restriction in $\mathcal{ELHI}^{\perp}_{\preceq}$. It is easy to see that $\mathcal{ELHI}^{\perp}_{\preceq}$ strictly extends DL-Lite [15, Def. 7.9] while being a strict fragment of $\mathcal{ELHI}^{\perp}$. A separating example is the non-first-order-rewritable TBox $\mathcal{T} \coloneqq \{\exists r.\mathrm{A} \sqsubseteq \mathrm{A}\}$, which belongs to our logic but not to DL-Lite [15, Consequence of Thm. 7.8]. On the other hand, $\mathcal{T}' \coloneqq \{\exists r.\mathrm{A} \sqcap \exists s.\mathrm{A} \sqsubseteq \mathrm{A}\}$ is a TBox that belongs to $\mathcal{ELI}^{\perp}$ but not to $\mathcal{ELHI}^{\perp}_{\preceq}$. The rewritings of the instance query $\mathrm{A}(\mathrm{a})$ over both $\mathcal{T}$ and $\mathcal{T}'$ are inherently recursive and thus fall outside first-order logic. A reader familiar with Datalog [15, Sec. 7.3] will recognize that the rewriting for $\mathcal{T}$ fits into the fragment of Datalog with linear recursion, whereas the one for $\mathcal{T}'$ does not. Consequently, the rewriting of $\mathcal{T}'$ into nested automata or well-known graph query languages such as GQL is not possible, while the rewriting of $\mathcal{T}$ can be expressed as a regular path query, *e.g.* as $(r^*; \mathrm{A}?)(\mathrm{a}, x)$. Requiring recursion to be linear is a crucial condition for guaranteeing the existence of rewritings of instance queries. In our approach, this is enforced by the preorder $\preceq$ in Definition 3.1 (in particular by the second condition).

Example 3.2. *It is a well-known fact that in expressive DLs such as $\mathcal{ELI}$, one can simulate concept conjunction by means of role inverses. Indeed, the GCI $\mathrm{A} \sqcap \mathrm{B} \sqsubseteq \mathrm{C}$ can be replaced by the following set of GCIs $\mathcal{T} \coloneqq \{\mathrm{A} \sqsubseteq \exists r.\top,\ \exists \bar{r}.\mathrm{B} \sqsubseteq \mathrm{D},\ \exists r.\mathrm{D} \sqsubseteq \mathrm{C}\}$, where the role r and the concept name D are fresh. We stress that reusing this "conjunction emulation" approach for arbitrary concepts fails in our logic due to the presence of the built-in order $\preceq$. Indeed, it is routine to check that, in the presence of $\preceq$, the following facts hold for $\mathcal{T}$: $\mathrm{A} \preceq r$, $r \preceq \mathrm{B}$, $\mathrm{B} \prec \mathrm{D}$, $r \preceq \mathrm{C}$, $\mathrm{D} \prec \mathrm{C}$. In particular, this implies $\mathrm{A} \prec \mathrm{C}$ and $\mathrm{B} \prec \mathrm{C}$, and thus A and B cannot be arbitrary concepts.* ◀

The main technical contribution of this paper is the rewriting of instance queries under $\mathcal{ELHI}^{\bot}_{\preceq}$-ontologies into nested two-way regular path queries. We state the result formally below.

Theorem 3.3. *Take a $\mathcal{ELHI}^{\bot}_{\preceq}$-TBox $\mathcal{T}$ and an instance query q. We can compute an $\mathrm{N2RPQ}$ q' such that for all ABoxes $\mathcal{A}$ and all $a \in \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{A}, \mathcal{T}) \models q(a)$ if and only if $\mathcal{A} \models q'(a)$.* ◀

In the remainder of the section, we fix q and $\mathcal{T}$ as in the statement of Theorem 3.3, and present how to construct the desired query q'.[2] Soundness and completeness of the rewriting are established in the next two sections, respectively. Our construction of q' is inductive and based on the *height* of the concept/role names α in $\mathcal{T}$, namely the length of the longest chain of concept or role names $\alpha_1 \prec \alpha_2 \prec \ldots \prec \alpha_n$ for which $\alpha_n = \alpha$. This notion is lifted to concepts from $\mathcal{T}$ as follows: the height of $\top$ and $\bot$ is 0, and the height of concepts of the form $\exists s.\mathrm{A}$ is the maximum of heights of s and A. For convenience, let $\mathcal{T}\restriction_n$ and $\mathcal{A}\restriction_n$ respectively denote the set of GCIs and assertions from $\mathcal{T}$ and $\mathcal{A}$ that solely contain concepts of height at most n. Our inductive claim is stated below ($\mathcal{T}\restriction_{-1}$ is just the empty set).

Inductive assumption for $n \in \mathbb{N}$: For each concept name A in $\mathcal{T}$ of height n, we can construct an n-nested NFA $\mathscr{A}_{\mathrm{A}}$ over $\{\mathrm{B}? \mid \mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_n)\} \cup \mathsf{rol}(\mathcal{T}\restriction_n) \cup \{\mathscr{A}_{\mathrm{B}}? \mid \mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_{n-1})\}$ such that for all ABoxes $\mathcal{A}$ and all $\mathrm{a} \in \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{A}, \mathcal{T}) \models \mathrm{A}(\mathrm{a})$ if and only if $\mathcal{A} \models \exists x \mathscr{A}_{\mathrm{A}}(\mathrm{a}, x)$.

Less formally, our goal is to construct an automaton that witnesses the satisfaction of A at a given node in an ABox, using the GCIs in $\mathcal{T}$ as inference rules. Its alphabet consists of (possibly inverted) role names, tests over concept names of height at most n (enabling the automaton to read labels from the ABox), and inductively constructed automaton-tests detecting membership in concepts of height strictly less than n (enabling it to delegate the verification of certain concepts to "simpler" automata). The states are pairs $(\mathit{premise}, \mathit{goal})$, where the first component is a set of concept names from $\mathcal{T}$ and the second one is a single one of those. Along a run, the automaton tracks the concepts from $\mathcal{T}$ already known to hold at the current node (the *premise*), as well as the one that remains to be verified there (the *goal*). Transitions are either derived from the GCIs in $\mathcal{T}$, mimicking a (reversed) entailment proof via successive rule applications, or read off from the ABox, recording which concept names hold at the current node. For instance, whenever a GCI $\mathrm{A} \sqsubseteq \mathrm{B}$ is present in $\mathcal{T}$ and B is the goal, the automaton may replace it with A. Finally, a state is accepting if and only if its goal is contained in the premise, *i.e.* the goal has already been proven.

Base case. We handle the base case first, as it already captures the key challenges of our technique. For the reader's convenience, we interweave formal definitions with informal descriptions.

Definition 3.4. *Fix a concept name $\mathrm{A} \in \mathsf{con}(\mathcal{T}\restriction_0)$. We define an NFA $\mathscr{A}_{\mathrm{A}} \coloneqq (Q, \Sigma, \delta, q_0, F)$ as follows.*

- *We set $\Sigma \coloneqq \{\mathrm{B}? \mid \mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_0)\} \cup \mathsf{rol}(\mathcal{T}\restriction_0)$, as already stated in the inductive claim. Note that in the base case no automaton-tests are used.*
- *The state set Q consists of all pairs $(\mathit{premise}, \mathit{goal})$ for $\mathit{premise} \subseteq \mathsf{con}(\mathcal{T}\restriction_0)$ and $\mathit{goal} \in \mathsf{con}(\mathcal{T}\restriction_0)$.*
- *The initial state is $q_0 \coloneqq (\{\top\}, \mathrm{A})$. The intuition is straightforward: we wish to verify A at the current node, but we currently only know that $\top$ holds there.*
- *The set of accepting states F consists of all pairs in Q where $\mathit{goal} \in \mathit{premise}$ or $\bot \in \mathit{premise}$.*

[2] W.l.o.g. we assume that the concept names from q occur in $\mathcal{T}$; otherwise the entailment is trivially false.

- *The transition relation δ is defined as the minimal subset of $Q \times \Sigma \times Q$ such that, for every state $q \coloneqq (\mathit{premise}, \mathit{goal})$, all the following conditions hold.*

  (weak) *For any subset $\mathit{premise}'$ of $\mathit{premise}$, there is a transition from q to $(\mathit{premise}' \cup \{\top\}, \mathit{goal})$ labelled "$\top?$". This allows the automaton to forget some of the information in the premise, which will be crucial for handling GCIs with conjunctions in the premise.*

  (data) *For any concept name $\mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_0)$, there is a transition from q to $(\mathit{premise} \cup \{\mathrm{B}\}, \mathit{goal})$ labelled "$\mathrm{B}?$". This allows the automaton to read from the ABox which concept names hold at the current node and add them to the premise.*

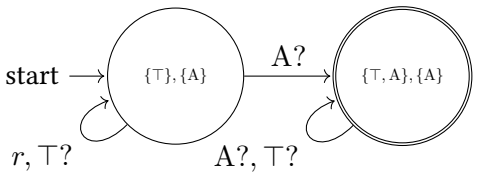
  (sbus) *For any GCI $\mathrm{C} \sqsubseteq \mathrm{D}$ from $\mathcal{T}\restriction_0$, if $\mathit{goal} = \mathrm{D}$, there exists a transition from q to $(\mathit{premise}, \mathrm{C})$ labelled "$\top?$". This formalizes the following backward reasoning: to prove D given the GCI $\mathrm{C} \sqsubseteq \mathrm{D}$, it suffices to prove C.*

  (succ) *If $\mathit{goal} = \mathrm{D}$, then for any GCI $\exists s.\mathrm{C} \sqsubseteq \mathrm{D}$ in $\mathcal{T}\restriction_0$, there are transitions from q to $(\{\top\}, \mathrm{C})$ labelled "s'" for each s' satisfying $s' \subseteq^+_{\mathcal{T}} s$. This formalizes the following reasoning: given the GCI $\exists s.\mathrm{C} \sqsubseteq \mathrm{D}$, the automaton may move to an s-neighbouring node (or s'-neighbouring node if s' is constrained to be a subrole of s) in the ABox and verify that C holds there. The premise is then reset to $\{\top\}$ since the automaton does not have any non-trivial information on that neighbour. The fact that it suffices to track a single successor at a time—which is essential for the feasibility of our construction—follows from the linear recursion enforced by Definition 3.1.*

  (anon) *For any concepts B and D from $\mathsf{con}(\mathcal{T}\restriction_0)$, if the KB $(\{\mathrm{C}(a) \mid \mathrm{C} \in \mathit{premise} \cup \{\mathrm{B}\}\}, \mathcal{T}\restriction_0)$ entails $\mathrm{D}(a)$ and $\mathit{goal} = \mathrm{D}$, there is a transition from q to $(\mathit{premise}, \mathrm{B})$ labelled "$\top?$". Though this transition may appear redundant, it is crucial for correctness: it enables the automaton to infer the existence of anonymous individuals satisfying certain concepts, which is essential for handling GCIs with existential restrictions in the premise.*

  Note that GCIs of the form $\mathrm{A} \sqcap \mathrm{B} \sqsubseteq \mathrm{C}$ play no role in the definition of δ here. By the second condition of Definition 3.1 and by the definition of a normal form, such GCIs cannot solely involve concepts from $\mathsf{con}(\mathcal{T}\restriction_0)$. ◀

We illustrate our construction of $\mathcal{A}_\mathrm{A}$ for a (non-first-order-rewritable) TBox $\mathcal{T} \coloneqq \{\exists r.\mathrm{A} \sqsubseteq \mathrm{A}\}$ and an instance query $q \coloneqq \mathrm{A}(\mathrm{a})$. The relevant *fragment* of $\mathcal{A}_\mathrm{A}$ is depicted below[3]. It is equivalent to the regular path query $(r^*; \mathrm{A}?)(\mathrm{a}, x)$, the expected rewriting for this query and TBox.



Inductive step. We now proceed with the inductive step. The main difference with the base case is that we can now also use automaton-tests in the transitions to deal with GCIs involving conjunction.

Definition 3.5. *Fix $n > 0$ and a concept name $\mathrm{A} \in \mathsf{con}(\mathcal{T}\restriction_n)$, and assume that $\mathcal{A}_\mathrm{B}$ has already been defined for each $\mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_{n-1})$. We define a nested NFA $\mathcal{A}_\mathrm{A} \coloneqq (Q, \Sigma, \delta, q_0, F)$ as follows.*

- *The components Σ, Q, q_0, and F are defined as in the base case, with the sole difference that Σ now also contains automaton-tests of the form $\mathcal{A}_\mathrm{B}?$ for each $\mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_{n-1})$, and all occurrences of $\mathcal{T}\restriction_0$ are replaced with $\mathcal{T}\restriction_n$.*
- *The transition relation δ is the minimal subset of $Q \times \Sigma \times Q$ such that, for every state $q \coloneqq (\mathit{premise}, \mathit{goal})$, all conditions from the base case (after replacing $\mathcal{T}\restriction_0$ with $\mathcal{T}\restriction_n$) hold, as well as the following three conditions are satisfied.*

[3] The resulting automaton contains also other states, *e.g.* the ones with $\bot$ in the premise.

(noc) *For any GCI* $\mathrm{B} \sqcap \mathrm{C} \sqsubseteq \mathrm{D}$ *in* $\mathcal{T}\restriction_n$ *with* $\mathrm{B} \in \mathit{premise}$ *and* $\mathit{goal} = \mathrm{D}$*, there exists a transition from* q *to* $(\mathit{premise}, \mathrm{C})$ *labelled "*$\top?$*". The case of the GCI* $\mathrm{B} \sqcap \mathrm{C} \sqsubseteq \mathrm{D}$ *with* $\mathrm{C} \in \mathit{premise}$ *and* $\mathrm{D} \in \mathit{goal}$ *is handled analogously. This formalises the standard backward reasoning step for conjunction.*

(aut) *For any concept name* $\mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_{n-1})$ *there is a transition from* q *to* $(\mathit{premise} \cup \{\mathrm{B}\}, \mathit{goal})$ *labelled "*$\mathcal{A}_\mathrm{B}?$*". As the height of* B *is strictly less than that of* A*, the automaton can "delegate" the proof of* B *to the nested automaton* $\mathcal{A}_\mathrm{B}$ *(well-defined by the inductive hypothesis).* ◀

4. Soundness

In this section we establish the "only if" direction of Theorem 3.3, restated for convenience below.

Lemma 4.1. *Let* $\mathcal{T}$ *be a stratified* $\mathcal{ELHI}^\bot_\precsim$*-TBox,* $\mathrm{A}(a)$ *be an instance query, and* $\mathcal{A}_\mathrm{A}$ *be an nNFA from Definition 3.5. Then for all for all ABoxes* $\mathcal{A}$ *we have that* $\mathcal{A} \models \exists x \mathcal{A}_\mathrm{A}(a, x)$ *implies* $(\mathcal{A}, \mathcal{T}) \models \mathrm{A}(a)$*.* ◀

Proof. Fix a TBox $\mathcal{T}$. The proof is by induction on the height n of the concept name A. We assume that for all $n' < n$ and concept names $\mathrm{B} \in \mathsf{con}(\mathcal{T}\restriction_{n'})$, the statement of Lemma 4.1 holds for the instance query $\mathrm{B}(\mathrm{a})$.

Then we consider an instance query $\mathrm{A}(\mathrm{a})$ with $\mathrm{A} \in \mathsf{con}(\mathcal{T}\restriction_n)$. Fix an ABox $\mathcal{A}$, a model $\mathcal{I}$ of $(\mathcal{A}, \mathcal{T})$, and let $\rho = \rho_1 \cdots \rho_\ell$ be an accepting run of $\mathcal{A}_\mathrm{A}$ in $\mathcal{I}$ starting from $\mathrm{a}^\mathcal{I}$, which exists by entailment $\mathcal{A} \models \exists x \mathcal{A}_\mathrm{A}(\mathrm{a}, x)$). For convenience, let ρ_i unfold to $(\mathrm{d}_{i-1}, \delta_i, \mathrm{d}_i)$ with $q_i = (\mathit{premise}_i, \mathit{goal}_i)$ and $\delta_i = (q_{i-1}, w_i, q_i)$. In the appendix, we prove the following claim. The proof is by induction on i, backwards from $i = \ell$ to $i = 0$.

If d_i satisfies all concepts from $\mathit{premise}_i$ in $\mathcal{I}$, then it also satisfies all concepts from goal_i.

From this claim, the statement of the lemma follows, as the initial state of $\mathcal{A}_\mathrm{A}$ is $(\{\top\}, \mathrm{A})$, while the accepting states have their goal already proven. □

5. Completeness

The goal of this section is to establish the remaining direction of Theorem 3.3. But before we can proceed with the proof, we need some additional definitions. First, we recall the standard notion of a derivation.

Definition 5.1. *Let* $\mathcal{T}$ *be an* $\mathcal{ELHI}^\bot_\precsim$*-TBox. A* $\mathcal{T}$-derivation *of* $\mathrm{A}(a)$ *from an ABox* $\mathcal{A}$ *is a finite sequence* $\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_n$ *of ABoxes such that* $\mathcal{A}_0$ *is a subset of* $\mathcal{A}$*,* $\mathcal{A}_n$ *contains* $\mathrm{A}(a)$*, and for every* $i < n$ *the ABox* $\mathcal{A}_{i+1}$ *can be derived from* $\mathcal{A}_i$ *(denoted* $\mathcal{A}_i \rightsquigarrow_\mathcal{T} \mathcal{A}_{i+1}$*),* i.e. *there exists a GCI* $\mathrm{B} \sqsubseteq \mathrm{C}$ *in* $\mathcal{T}$*,* $s' \subseteq^+_\mathcal{T} s$*, and an individual name* b *such that:*[4]

- *Both* B *and* C *are concept names or* $\top$*,* $\mathrm{B}(b) \in \mathcal{A}_i$*, and* $\mathcal{A}_{i+1} = \mathcal{A}_i \cup \{\mathrm{C}(b)\}$*;*
- B *is of the form* $\mathrm{B}_1 \sqcap \mathrm{B}_2$*, both* $\mathrm{B}_1(b)$ *and* $\mathrm{B}_2(b)$ *belong to* $\mathcal{A}_i$*, and* $\mathcal{A}_{i+1} = \mathcal{A}_i \cup \{\mathrm{C}(b)\}$*;*
- B *is a concept name or* $\top$*,* C *is of the form* $\exists s.\mathrm{C}_1$*,* $\mathrm{B}(b) \in \mathcal{A}_i$*, and* $\mathcal{A}_{i+1} = \mathcal{A}_i \cup \{s(b, c), \mathrm{C}_1(c)\}$ *for some fresh individual name* c*;*
- B *is of the form* $\exists s.\mathrm{B}_1$*, both* $s'(b, c)$ *and* $\mathrm{B}_1(c)$ *belong to* $\mathcal{A}_i$*, and* $\mathcal{A}_{i+1} = \mathcal{A}_i \cup \{\mathrm{C}(b)\}$*; or*
- $\bot(b) \in \mathcal{A}_i$*, and* $\mathcal{A}_{i+1} = \{\mathrm{A}(a)\}$*.*

In all of these cases, if $\mathrm{B}(b)$ *is (one of) the mentioned concept assertions in* $\mathcal{A}_i$ *and* $\mathrm{A}(a)$ *is the newly added concept assertion to* $\mathcal{A}_{i+1}$*, we say that* $\mathrm{B}(b)$ was used to infer $\mathrm{A}(a)$*. We call the number* n *the* length *of the derivation. We say that* $\mathrm{A}(a)$ *is* $\mathcal{T}$-derivable *from* $\mathcal{A}$ *whenever there exists a* $\mathcal{T}$*-derivation of* $\mathrm{A}(a)$ *from* $\mathcal{A}$*.* ◀

[4] For brevity, we abuse the notation and write $\top(\mathrm{b}) \in \mathcal{A}_i$ to indicate that b appears in $\mathcal{A}_i$.

The following example illustrates the notion of derivations and will be continued later.

Example 5.2. *Consider an ABox* $\mathcal{A}_{\text{ex}} \coloneqq \{\mathrm{A}(a)\}$*, an* $\mathcal{ELHI}^{\bot}_{\vec{\leq}}$*-TBox* $\mathcal{T}_{\text{ex}} \coloneqq \{\mathrm{A} \sqsubseteq \mathrm{B},\ \mathrm{A} \sqcap \mathrm{B} \sqsubseteq \mathrm{C},\ \mathrm{C} \sqsubseteq \exists r.\top,\ \exists r.\top \sqsubseteq \mathrm{D}\}$*, and an atomic query* $q_{\text{ex}} \coloneqq \mathrm{D}(a)$*. We enumerate the GCIs of* $\mathcal{T}_{\text{ex}}$ *in order of appearance as* $\alpha_0, \ldots, \alpha_3$*. The reader may easily verify that the sequence* $\mathcal{A}_0, \ldots, \mathcal{A}_4$ *is a* $\mathcal{T}_{\text{ex}}$*-derivation of* q_{ex} *from* $\mathcal{A}_{\text{ex}}$*, where* $\mathcal{A}_0 = \mathcal{A}$*,* $\mathcal{A}_1 = \mathcal{A}_0 \cup \{\mathrm{B}(a)\}$*,* $\mathcal{A}_2 = \mathcal{A}_1 \cup \{\mathrm{C}(a)\}$*,* $\mathcal{A}_3 = \mathcal{A}_2 \cup \{r(a,b), \top(b)\}$*, and* $\mathcal{A}_4 = \mathcal{A}_3 \cup \{\mathrm{D}(a)\}$*. In particular, for all* $i < 4$ *we have* $\mathcal{A}_i \rightsquigarrow_{\{\alpha_i\}} \mathcal{A}_{i+1}$*.* ◀

The following lemma links derivations with entailment. Its proof is standard, and hence we omit it.

Lemma 5.3 (folklore)**.** *Let* $\mathcal{A}$ *be an ABox,* $\mathcal{T}$ *be a stratified* $\mathcal{ELHI}^{\bot}_{\vec{\leq}}$*-TBox, and* $\mathrm{A}(a)$ *be an instance query. Then* $(\mathcal{A}, \mathcal{T}) \models \mathrm{A}(a)$ *if and only if* $\mathrm{A}(a)$ *can be* $\mathcal{T}$*-derived from* $\mathcal{A}$*.* ◀

Our built-in stratification can also be reflected in the derivations, leading to the following notion.

Definition 5.4. *Let* $\mathcal{T}$*,* $\mathcal{A}$*, and* $\mathrm{A}(a)$ *be as in Definition 5.1, and suppose that* A *is of height* ℓ*. A* $(\mathcal{T}, \ell)$-derivation *of* $\mathrm{A}(a)$ *from* $\mathcal{A}$ *is a finite sequence* $(\mathcal{A}_0, \mathrm{A}_0(a_0)), (\mathcal{A}_1, \mathrm{A}_1(a_1)), \ldots, (\mathcal{A}_n, \mathrm{A}_n(a_n))$ *of ABoxes and instance queries (called* active queries*) with concepts from* $\mathsf{con}(\mathcal{T}\!\downharpoonright_{\ell})$ *such that:*

- $\mathrm{A}_n(a_n) = \mathrm{A}(a)$*, for all* $i \leq n$*, the sequence* $\mathcal{A}_0, \ldots, \mathcal{A}_i$ *is a* $\mathcal{T}\!\downharpoonright_{\ell}$*-derivation of* $\mathrm{A}_i(a_i)$ *from* $\mathcal{A}$*, and*
- *for all* $i < n$*: if* $\mathcal{A}_i \rightsquigarrow_{\{\mathrm{C} \sqsubseteq \mathrm{D}\}} \mathcal{A}_{i+1}$ *for some GCI* $\mathrm{C} \sqsubseteq \mathrm{D}$ *from* $\mathcal{T}\!\downharpoonright_{\ell}$ *with* D *of height* ℓ*, then*
  1. *if* B' *is a concept name of height* ℓ *and occurs in* C*, then* $\mathrm{A}_i(a_i) = \mathrm{B}'(a_i)$*;*
  2. $\mathrm{A}_{i+1}(a_{i+1}) = \mathrm{B}''(a_{i+1})$ *for the concept name* B'' *that occurs in* D*.* ◀

The following example illustrates the notion of stratified derivation.

Example 5.5. *Let* $\mathcal{A}_{\text{ex}}$*,* $\mathcal{T}_{\text{ex}}$*, and* q_{ex} *be as in Example 5.2. Suppose that the heights of* A*,* B*,* C*,* r*,* D *are* 0*,* 1*,* 2*,* 2*, and* 3*, respectively. The reader may verify that the sequence* $(\mathcal{A}_0, \mathrm{A}_0(a_0)), \ldots, (\mathcal{A}_4, \mathrm{A}_4(a_0))$ *is a* $(\mathcal{T}_{\text{ex}}, 3)$*-derivation of* q_{ex} *from* $\mathcal{A}_{\text{ex}}$*, where* $\mathrm{A}_0(a_0) = \mathrm{A}(a)$*,* $\mathrm{A}_1(a_1) = \mathrm{B}(a)$*,* $\mathrm{A}_2(a_2) = \mathrm{C}(a)$*,* $\mathrm{A}_3(a_3) = \top(b)$*, and* $\mathrm{A}_4(a_4) = \mathrm{D}(a)$*.* ◀

With the next lemma, we connect stratified derivations with the entailment of instance queries.

Lemma 5.6. *Let* $\mathcal{T}$ *be an* $\mathcal{ELHI}^{\bot}_{\vec{\leq}}$*-TBox,* $\mathcal{A}$ *be an ABox, and* $\mathrm{A}(a)$ *be an instance query such that* $(\mathcal{A}, \mathcal{T}) \models \mathrm{A}(a)$*. Then there exists a* $(\mathcal{T}, \ell)$*-derivation of* $\mathrm{A}(a)$ *from* $\mathcal{A}$ *with* ℓ *being the height of* A*.* ◀

Proof Sketch. Our proof proceeds by induction and our inductive statement is:

For every d, for every ℓ, and every instance query $\mathrm{A}(a)$ with concept A of height ℓ for which there exists a derivation of $(\mathcal{A}, \mathcal{T}) \models \mathrm{A}(a)$ of length d, there also exists a $(\mathcal{T}, \ell)$-derivation of $\mathrm{A}(a)$ from $\mathcal{A}$.

For the case $d = 0$ it suffices to take the pair $(\mathcal{A}, \mathrm{A}(a))$ as the desired $(\mathcal{T}, \ell)$-derivation of $\mathrm{A}(a)$ from $\mathcal{A}$. For the induction step we can perform a case distinction based on the GCI that was used in the last step of the derivation. We provide a detailed execution of this argument in the appendix. □

We next establish an important property of "stratified" derivations.

Lemma 5.7. *Let* $\mathcal{T}$ *be an* $\mathcal{ELHI}^{\bot}_{\vec{\leq}}$*-TBox,* $\mathcal{A}$ *be an ABox, and let* $(\mathcal{A}_0, \mathrm{A}_0(a_0)), (\mathcal{A}_1, \mathrm{A}_1(a_1)), \ldots, (\mathcal{A}_n, \mathrm{A}_n(a_n))$ *be a* $(\mathcal{T}, \ell)$*-derivation of some concept* $\mathrm{A}(a)$ *from* $\mathcal{A}$*. For all* $i \geq 0$*, if* $a_i \in \mathsf{ind}(\mathcal{A})$ *and* $a_{i-1} \notin \mathsf{ind}(\mathcal{A})$*, such that* $\mathrm{A}_{i-1}(a_{i-1})$ *was used to infer* $\mathrm{A}_i(a_i)$*, then:*

- *if there is no* $j < i$ *with* $a_j = a_i$*, then* $(\mathcal{T}\!\downharpoonright_{\ell}, \mathcal{A}_{i-1}\!\downharpoonright^{a_i}_{\ell-1}) \models \mathrm{A}_i(a_i)$*; and*
- *otherwise for some* $j < i$ *with* $a_j = a_i$ *we have* $(\mathcal{T}\!\downharpoonright_{\ell}, \mathcal{A}_{i-1}\!\downharpoonright^{a_i}_{\ell-1} \cup \{\mathrm{A}_j(a_i)\}) \models \mathrm{A}_i(a_i)$*,*

where $\mathcal{A}_i\!\downharpoonright^{a}_{\ell-1}$ *is the restriction of* $\mathcal{A}_i$ *to assertions about a concerning concepts of height at most* $\ell - 1$*.* ◀

Informally, in this lemma we claim, that in the described setting, $\mathrm{A}_i(\mathrm{a}_i)$ can be inferred solely by considering the already known facts about a_i of some lower height, together with the TBox restricted to lower levels, and, if it exists as an active query, a single concept assertion on a_i of height ℓ.

Proof. Without loss of generality, assume that $\mathcal{A}$ contains $\top(\mathrm{a})$ for all $\mathrm{a} \in \mathsf{ind}(\mathcal{A})$. Consider the directed implication graph of $\mathcal{A}_i$ (denoted $\mathcal{G}_{\mathcal{A}_i}$), whose vertices are the concept assertions in $\mathcal{A}_i$. There is a directed edge from $\mathrm{B}(\mathrm{b})$ to $\mathrm{C}(\mathrm{c})$ if there exists $j < i$ such that $\mathrm{B}(\mathrm{b}) \notin \mathcal{A}_j$, $\mathrm{C}(\mathrm{c}) \in \mathcal{A}_j$, and $\mathrm{C}(\mathrm{c})$ was used to derive $\mathrm{B}(\mathrm{b})$ and add it to $\mathcal{A}_{j+1}$. By Definition 5.1, if such an edge exists, then either $\mathrm{b} = \mathrm{c}$ or $s(\mathrm{b}, \mathrm{c}) \in \mathcal{A}_{j+1}$. Inspecting Definition 5.1, we observe that no derivation step creates edges between existing individuals. Hence, along any path in $\mathcal{G}_{\mathcal{A}_i}$ from an "anonymous" individual $\mathrm{c} \notin \mathsf{ind}(\mathcal{A})$ connected to a_i (i.e., $s(\mathrm{a}_i, \mathrm{c}) \in \mathcal{A}_i$) to a "named" individual $\mathrm{b} \in \mathsf{ind}(\mathcal{A})$, the first named individual encountered is a_i. Thus every maximal path ρ in $\mathcal{G}_{\mathcal{A}_i}$ starting at $\mathrm{A}_i(\mathrm{a}_i)$ and proceeding to $\mathrm{A}_{i-1}(\mathrm{a}_{i-1})$, must have some $k > 2$ such that $\rho_k = \mathrm{C}(\mathrm{a}_i)$ for some C, since the last vertex of ρ lies in $\mathcal{A}$. Furthermore, at most one such maximal path starting at $\mathrm{A}_i(\mathrm{a}_i)$ can contain a vertex $\mathrm{A}_j(\mathrm{a}_i)$ of height ℓ. Otherwise, suppose two paths ρ and ρ' exist. Then they coincide up to some $\rho_k = \rho'_k = \mathrm{B}(\mathrm{b})$, but diverge at the next step. By the definition of the implication graph, this implies that $\mathrm{B}(\mathrm{b})$ was derived using a rule $\mathrm{C} \sqcap \mathrm{D} \sqsubseteq \mathrm{B}$, with $\mathrm{b} = \mathrm{c} = \mathrm{d}$. Then one of C, D must have strictly lower height, contradicting stratification if a later vertex of height ℓ reappears on the diverging path. Since only active queries can have height ℓ, the assertion $\mathrm{A}_j(\mathrm{a}_i)$ must have been active at some stage prior to i. All other assertions about a_i on paths from $\mathrm{A}_i(\mathrm{a}_i)$ have lower height and thus already occur in $\mathcal{A}_{i-1}$. It remains to derive $\mathrm{A}_i(\mathrm{a}_i)$ from $(\mathcal{T}\!\downarrow_\ell, \mathcal{A}_{i-1}\!\downarrow_{\ell-1}^{\mathrm{a}_i} \cup \{\mathrm{A}_j(\mathrm{a}_i)\})$. If no such $\mathrm{A}_j(\mathrm{a}_i)$ occurs, we take $\mathrm{A}_j(\mathrm{a}_i) = \top(\mathrm{a}_i)$. Let S be the set of assertions in $\mathcal{G}_{\mathcal{A}_i}$ reachable from $\mathrm{A}_i(\mathrm{a}_i)$ without passing through assertions about a_i. Let $\overline{s} = (s_1, \ldots, s_{|S|})$ enumerate S in the order of first occurrence in the derivation. Define:

$$\mathcal{A}'_0 = \mathcal{A}_{i-1}\!\downarrow_{\ell-1}^{\mathrm{a}_i} \cup \{\mathrm{A}_j(\mathrm{a}_i)\},$$

$$\mathcal{A}'_{k+1} = \mathcal{A}'_k \cup \{s_{k+1}\} \cup \{r(\mathrm{b}, \mathrm{c}) \mid \mathrm{b}, \mathrm{c} \in \mathsf{ind}(\mathcal{A}'_k \cup \{s_{k+1}\}),\ r(\mathrm{b}, \mathrm{c}) \in \mathcal{A}_i\}.$$

We claim this yields a valid derivation of $\mathrm{A}_i(\mathrm{a}_i)$. First, $\overline{s}$ is a reverse topological ordering of S: if $(\mathrm{B}(\mathrm{b}), \mathrm{C}(\mathrm{c}))$ is an edge, then $\mathrm{C}(\mathrm{c})$ was derived before $\mathrm{B}(\mathrm{b})$, hence appears earlier in $\overline{s}$. Now consider $s_{k+1} = \mathrm{B}(\mathrm{b})$ derived via $\mathrm{D} \sqsubseteq \mathrm{E}$. For any predecessor $\mathrm{C}(\mathrm{c})$ of $\mathrm{B}(\mathrm{b})$, we have $\mathrm{C}(\mathrm{c}) \in S$ or $\mathrm{c} = \mathrm{a}_i$. In the latter case, $\mathrm{C}(\mathrm{a}_i)$ is either $\mathrm{A}_j(\mathrm{a}_i)$ or already in $\mathcal{A}'_0$. By the ordering of $\overline{s}$, all such $\mathrm{C}(\mathrm{c})$ belong to $\mathcal{A}'_k$. If $\mathrm{D} \sqsubseteq \mathrm{E}$ does not involve existential restrictions, it applies directly. If it is of the form $\exists r.\mathrm{C} \sqsubseteq \mathrm{B}$, then all required role assertions are present in $\mathcal{A}'_k$ by construction. If it is of the form $\mathrm{C} \sqsubseteq \exists r.\mathrm{B}$, then b is introduced as a fresh individual, and $\mathcal{A}'_{k+1}$ correctly reflects this extension. Thus, each step is valid, completing the derivation. □

We can now demonstrate how to retrieve accepting runs of the nested automata from the stratified derivations guaranteed by the entailment of instance queries. In this way, we can finally establish the remaining direction of Theorem 3.3, restated below for convenience.

Lemma 5.8. *Let $\mathcal{T}$ be a stratified $\mathcal{ELHI}_{\geq}^{\perp}$-TBox, $\mathrm{A}(a)$ be an instance query with A of height n, and $\mathcal{A}_\mathrm{A}$ be the n-nested NFA for A constructed in Definition 3.5. Then for all ABoxes $\mathcal{A}$ we have that $(\mathcal{A}, \mathcal{T}) \models \mathrm{A}(a)$ implies $\mathcal{A} \models \exists x\, \mathcal{A}_\mathrm{A}(a, x)$,* i.e. *there exists a accepting run of $\mathcal{A}_\mathrm{A}$ starting from $a^{\mathcal{I}}$.* ◀

Proof Sketch. Fix $\mathcal{T}$ and $\mathcal{A}$ as in the statement of the lemma, and let $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$. We proceed by induction over the length d of the run. More specifically, we prove the following inductive claim:

For all $d \in \mathbb{N}$, all $\ell \in \mathbb{N}$, all $\mathrm{a} \in \mathsf{ind}(\mathcal{A})$, and all concept names $\mathrm{A} \in \mathsf{con}(\mathcal{T}\!\downarrow_\ell)$, whenever there is a $(\mathcal{T}, \ell)$-derivation of $\mathrm{A}(\mathrm{a})$ from $\mathcal{A}$ of length d, then there is an accepting run of $\mathcal{A}_\mathrm{A}$ starting from $\mathrm{a}^{\mathcal{I}}$.

The base case is immediate. Indeed, take any ℓ and $\mathrm{A} \in \mathsf{con}(\mathcal{T}\!\downarrow_\ell)$, and suppose that there is a $(\mathcal{T}, \ell)$-derivation of $\mathrm{A}(\mathrm{a})$ from $\mathcal{A}$ of length d. Then such a derivation has the form $(\mathcal{A}_0, \mathrm{A}(\mathrm{a}))$ for some $\mathcal{A}_0 \subseteq \mathcal{A}$ with $\mathrm{A}(\mathrm{a}) \in \mathcal{A}_0$. Thus, to construct an accepting run of $\mathcal{A}_\mathrm{A}$ we can step from $q_0 :=$

$(\{\top\}, \mathrm{A})$ to $q_1 \coloneqq (\{\mathrm{A}, \top\}, \mathrm{A})$, via a (data)-transition, which is already a final state. More formally, $(\mathrm{a}^{\mathcal{I}}, q_0, \mathrm{A}?, q_1, \mathrm{a}^{\mathcal{I}})$ is the desired run of $\mathcal{A}_{\mathrm{A}}$ starting from $\mathrm{a}^{\mathcal{I}}$.

For the inductive step, we again do a case distinction on the last step of the $(\mathcal{T}, \ell)$-derivation to simulate that derivation step in the automaton. In particular "reasoning in the anonymous part" is possible thanks to Lemma 5.7. In the appendix we provide a full version of this argument. □

The following example illustrates the proof above.

Example 5.9. *Let $\mathcal{A}_{\mathrm{ex}}$, $\mathcal{T}_{\mathrm{ex}}$, and q_{ex} be as in Example 5.2, and let $(\mathcal{A}_0, \mathrm{A}_0(a_0)), \dots, (\mathcal{A}_4, \mathrm{A}_4(a_0))$ be the $(\mathcal{T}_{\mathrm{ex}}, 3)$-derivation of q_{ex} from $\mathcal{A}_{\mathrm{ex}}$ from Example 5.5. Fix a model $\mathcal{I} \models (\mathcal{T}_{\mathrm{ex}}, \mathcal{A}_{\mathrm{ex}})$. By inspecting the proof of Lemma 5.8, we see that the following run $(a^{\mathcal{I}}, \delta_0, a^{\mathcal{I}}), \dots, (a^{\mathcal{I}}, \delta_4, a^{\mathcal{I}})$ of $\mathcal{A}_{\mathrm{D}}$ starting from $a^{\mathcal{I}}$ is produced, where:*

- $\delta_0 = ((\{\top\}, \mathrm{D}), \top?, (\{\top\}, \mathrm{C}))$ *corresponds to the use of a (anon)-transition parametrised by* C*, invoking the anonymous part of $\mathcal{I}$.*
- $\delta_1 = ((\{\top\}, \mathrm{C}), \mathrm{A}?, (\{\top, \mathrm{A}\}, \mathrm{C}))$ *corresponds to the use of a (data)-transition reading $\mathrm{A}(a)$ from the ABox.*
- $\delta_2 = ((\{\top, \mathrm{A}\}, \mathrm{C}), \top?, (\{\top, \mathrm{A}\}, \mathrm{B}))$ *corresponds to the use of a (noc)-transition invoking the GCI* $\mathrm{A} \sqcap \mathrm{B} \sqsubseteq \mathrm{C}$.
- $\delta_3 = ((\{\top, \mathrm{A}\}, \mathrm{B}), \top?, (\{\top, \mathrm{A}\}, \mathrm{A}))$ *corresponds to the use of a (sbus)-transition invoking the GCI* $\mathrm{A} \sqsubseteq \mathrm{B}$. *This puts the automaton in an accepting state.* ◀

We have proved that our rewriting of instance queries under $\mathcal{ELHI}^{\bot}_{\preceq}$-ontologies into N2RPQs is sound and complete. As evaluating N2RPQs is feasible in NL [17], we obtain the desired complexity result.

Corollary 5.10. *Take a fixed $\mathcal{ELHI}^{\bot}_{\preceq}$-TBox $\mathcal{T}$ and an instance query q. For every ABox $\mathcal{A}$, deciding $(\mathcal{A}, \mathcal{T}) \models q(a)$ is feasible in NL.*

6. Combined Complexity

We conclude the paper by establishing PSpace-completeness of instance query entailment in $\mathcal{ELHI}^{\bot}_{\preceq}$.

Lower bound. We reduce from QBF, which is PSpace-hard even for prenex 3-DNF formulas [18, Thm. 6.1]. The reduction already works for linear orders $\preceq$. Consider $\varphi = Q_1 x_1 \cdots Q_n x_n \psi$, where $Q_i \in \{\exists, \forall\}$ and $\psi = \bigvee_{j=1}^{m} \lambda_{j1} \wedge \lambda_{j2} \wedge \lambda_{j3}$, where λs are propositional literals. We define a singleton ABox $\mathcal{A}$, a TBox $\mathcal{T}$ as will follow, and a concept $\mathrm{C}_0^{\mathrm{True}}$ such that C_0 holds for the only individual in $\mathcal{A}$ if and only if φ is valid. We employ the following concept and role names

$$\{\mathrm{L}_i, \mathrm{X}_i^0, \mathrm{X}_i^1, \mathrm{C}_i^{\mathrm{True}}, \mathrm{C}_i^{\mathrm{True},0}, \mathrm{C}_i^{\mathrm{True},1} \mid 0 \le i \le n\} \cup \{\mathrm{A}_j^1, \mathrm{A}_j^2 \mid 1 \le j \le m\} \qquad \{r_{0,i}, r_{1,i} \mid 0 \le i \le n\}.$$

Intended models are binary trees of depth n rooted at a, where level i (marked by L_i) encodes the value of x_i via X_i^0 or X_i^1. Parent-to-child connections on level i are realised by roles $r_{d,i}$ (where $d \in \{0, 1\}$ denotes a direction). The concepts $\mathrm{C}_i^{\mathrm{True}}$ propagate truth of the suffix formula, and $\mathrm{A}_j^1, \mathrm{A}_j^2$ simulate conjunctions of literals. Let $\mathcal{A} = \{\mathrm{L}_0(\mathrm{a})\}$ and query $\mathrm{C}_0^{\mathrm{True}}(\mathrm{a})$. The TBox $\mathcal{T}$ is defined as:

$$\mathcal{T} \coloneqq \{\mathrm{L}_i \sqsubseteq \exists r_{0,i+1}.\mathrm{L}_{i+1}, \mathrm{L}_i \sqsubseteq \exists r_{1,i+1}.\mathrm{L}_{i+1} \mid i \in \{0, \dots, n-1\}\} \tag{1}$$

$$\cup \{\exists \overline{r}_{0,i+1}.\mathrm{L}_i \sqsubseteq \mathrm{X}_{i+1}^0, \exists \overline{r}_{1,i+1}.\mathrm{L}_i \sqsubseteq \mathrm{X}_{i+1}^1 \mid i \in \{0, \dots, n-1\}\} \tag{2}$$

$$\cup \{\exists \overline{r}_{d,i}.\mathrm{X}_i^0 \sqsubseteq \mathrm{X}_i^0, \exists \overline{r}_{d,i}.\mathrm{X}_i^1 \sqsubseteq \mathrm{X}_i^1 \mid i \in \{1, \dots, n\}, d \in \{0, 1\}\} \tag{3}$$

$$\cup \{\mathrm{L}_n \sqcap \mathrm{X}_i^1 \sqsubseteq \mathrm{A}_j^1 \mid \lambda_{j1} = x_i, j \in \{1, \dots, m\}, i \in \{1, \dots, n\}\} \tag{4}$$

$$\cup \{\mathrm{L}_n \sqcap \mathrm{X}_i^0 \sqsubseteq \mathrm{A}_j^1 \mid \lambda_{j1} = \neg x_i, j \in \{1,\dots,m\}, i \in \{1,\dots,n\}\} \tag{5}$$
$$\cup \{\mathrm{A}_j^1 \sqcap \mathrm{X}_i^1 \sqsubseteq \mathrm{A}_j^2 \mid \lambda_{j2} = x_i, j \in \{1,\dots,m\}, i \in \{1,\dots,n\}\} \tag{6}$$
$$\cup \{\mathrm{A}_j^1 \sqcap \mathrm{X}_i^0 \sqsubseteq \mathrm{A}_j^2 \mid \lambda_{j2} = \neg x_i, j \in \{1,\dots,m\}, i \in \{1,\dots,n\}\} \tag{7}$$
$$\cup \{\mathrm{A}_j^2 \sqcap \mathrm{X}_i^1 \sqsubseteq \mathrm{C}_n^{\mathrm{True}} \mid \lambda_{j3} = x_i, j \in \{1,\dots,m\}, i \in \{1,\dots,n\}\} \tag{8}$$
$$\cup \{\mathrm{A}_j^2 \sqcap \mathrm{X}_i^0 \sqsubseteq \mathrm{C}_n^{\mathrm{True}} \mid \lambda_{j3} = \neg x_i, j \in \{1,\dots,m\}, i \in \{1,\dots,n\}\} \tag{9}$$
$$\cup \{\exists r_{0,i+1}.\mathrm{C}_{i+1}^{\mathrm{True}} \sqsubseteq \mathrm{C}_i^{\mathrm{True},0}, \exists r_{1,i+1}.\mathrm{C}_{i+1}^{\mathrm{True}} \sqsubseteq \mathrm{C}_i^{\mathrm{True},1} \mid i \in \{0,\dots,n-1\}\} \tag{10}$$
$$\cup \{\mathrm{C}_i^{\mathrm{True},0} \sqsubseteq \mathrm{C}_i^{\mathrm{True}}, \mathrm{C}_i^{\mathrm{True},1} \sqsubseteq \mathrm{C}_i^{\mathrm{True}} \mid i \in \{0,\dots,n-1\}, Q_{i+1} = \exists\} \tag{11}$$
$$\cup \{\mathrm{C}_i^{\mathrm{True},0} \sqcap \mathrm{C}_i^{\mathrm{True},1} \sqsubseteq \mathrm{C}_i^{\mathrm{True}} \mid i \in \{0,\dots,n-1\}, Q_{i+1} = \forall\} \tag{12}$$

Finally, we can order our subset of $\mathbf{N_C} \cup \mathbf{N_R}$ with $\preceq$ as follows:

$$\mathrm{L}_0 \prec r_{0,1} \prec r_{1,1} \prec \mathrm{X}_1^0 \prec \mathrm{X}_1^1 \prec \mathrm{L}_1 \prec r_{0,2} \prec r_{1,2} \prec \mathrm{X}_2^0 \prec \mathrm{X}_2^1 \prec \dots \mathrm{L}_n \prec$$
$$\mathrm{A}_1^1 \prec \mathrm{A}_1^2 \prec \mathrm{A}_2^1 \prec \mathrm{A}_2^2 \prec \dots \mathrm{A}_m^1 \prec \mathrm{A}_m^2 \prec$$
$$\mathrm{C}_n^{\mathrm{True}} \prec \mathrm{C}_n^{\mathrm{True},0} \prec \mathrm{C}_n^{\mathrm{True},1} \prec \mathrm{C}_{n-1}^{\mathrm{True}} \prec \mathrm{C}_{n-1}^{\mathrm{True},0} \prec \mathrm{C}_{n-1}^{\mathrm{True},1} \prec \dots \prec \mathrm{C}_0^{\mathrm{True}}$$

The proof of the following lemma is standard. Check the appendix for details.

Lemma 6.1. *$(\mathcal{A},\mathcal{T}) \models \mathrm{C}_0^{True}(a)$ if and only if φ is valid. Hence, instance query entailment in $\mathcal{ELHI}_{\preceq}^{\perp}$ is PSpace-hard (in combined complexity).* ◀

Upper bound. Let $\mathcal{A}$ be an input ABox, $\mathcal{T}$ an $\mathcal{ELHI}_{\preceq}^{\perp}$-TBox, and $q \coloneqq \mathrm{A}(\mathrm{a})$ an instance query. By Theorem 3.3, to decide whether $(\mathcal{T},\mathcal{A}) \models q$, it suffices to construct the nNFA $\mathscr{A}$ given by the rewriting and check for an accepting run of $\mathscr{A}$ on $\mathcal{A}$ starting at a. The size of $\mathscr{A}$ is exponential in $|\mathcal{T}|$ (due to subsets of concepts in the premise component of states), so it cannot be materialised explicitly. Instead, we construct it on the fly during evaluation. Any accepting run has length bounded by the number of states of $\mathscr{A}$, hence exponential in $|\mathcal{T}|$. Such a bound can be tracked using an exponentially large counter encoded in polynomial space. The procedure guesses, step-by-step, a run of $\mathscr{A}$ on $\mathcal{A}$ starting at a, incrementing the counter and verifying the correctness of each transition. If an accepting state is reached within the bound, we accept; otherwise, we reject. Since only the current state and the counter need to be stored, the procedure runs in polynomial space. Thus, instance query entailment in $\mathcal{ELHI}_{\preceq}^{\perp}$ is in PSpace (in combined complexity). The only subtle point is the verification of transitions. While most transitions of $\mathscr{A}$ can be checked directly using the ABox and the current state, Steps (anon) and (aut) require additional care. For (aut) transitions, observe that the automaton test $\mathscr{A}_{\mathrm{B}}?$ involves only concepts of strictly lower height than A. Thus, by an inductive construction over heights and by the use of PSpace oracles, this step can be fully implemented in polynomial space. For (anon) transitions, which correspond to classical reasoning in $\mathcal{ELHI}_{\preceq}^{\perp}$, we adapt the standard alternating polynomial-space algorithm for $\mathcal{ELHI}^{\perp}$. The key observation is that the number of alternations is bounded by the height of the input concept. By the classical result that alternating PSpace with polynomially many alternations coincides with PSpace [19, Thm. 4.2], this step is also feasible in polynomial space. We thus conclude:

Theorem 6.2. *Instance query entailment in $\mathcal{ELHI}_{\preceq}^{\perp}$ is PSpace-complete (in combined complexity).* ◀

7. Conclusions and Future Work

In this work, we introduced $\mathcal{ELHI}_{\preceq}^{\perp}$, a novel fragment of $\mathcal{ELHI}^{\perp}$ that extends DL-Lite and is based on the idea of "stratified conjunction." By means of a suitable rewriting into nested two-way regular path queries (N2RPQs), we established that the instance query entailment problem for our logic is NL-complete with respect to data complexity.

Our future and ongoing work will pursue several directions. First, we aim to move beyond instance queries and investigate the combined complexity of query answering for $\mathcal{ELHI}_{\preceq}^{\perp}$ with respect to the

full class of nested C2RPQs. We are optimistic that this can be achieved through a suitable adaptation of the techniques presented by Bienvenu et al. [6]. Second, we intend to provide support for concrete domains. To do so, we plan to combine the query rewriting approach of Baader et al. [20] with the recent model of nested C2RPQs with data values introduced by Figueira et al. [21]. Finally, since the target of our rewriting is (a fragment of) GQL, we plan to investigate the practical applicability of our approach, by making the algorithm more amenable to implementation and evaluating a prototype on real-world datasets.

Acknowledgments

This research was funded in whole by the Austrian Science Fund (FWF) 10.55776/PIN8884924.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT and Claude in order to: grammar and spelling check, polish the text. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family, Journal of Automated Reasoning (2007).

[2] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Data Complexity of Query Answering in Description Logics, Artificial Intelligence (2013).

[3] L. Libkin, W. Martens, F. Murlak, L. Peterfreund, D. Vrgoc, Querying Graph Data: Where We Are and Where to Go, in: PODS 2025, ACM, 2025.

[4] N. Francis, A. Gheerbrant, P. Guagliardo, L. Libkin, V. Marsault, W. Martens, F. Murlak, L. Peterfreund, A. Rogova, D. Vrgoc, A Researcher's Digest of GQL (Invited Talk), in: ICDT 2023, LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[5] M. Bienvenu, M. Ortiz, M. Šimkus, Regular Path Queries in Lightweight Description Logics: Complexity and Algorithms, Journal of Artificial Intelligence Research (2015).

[6] M. Bienvenu, D. Calvanese, M. Ortiz, M. Šimkus, Nested Regular Path Queries in Description Logics, in: KR, 2014.

[7] D. Calvanese, T. Eiter, M. Ortiz, Regular Path Queries in Expressive Description Logics with Nominals, in: IJCAI 2009, 2009.

[8] M. Ortiz, S. Rudolph, M. Simkus, Query Answering in the Horn Fragments of the Description Logics SHOIQ and SROIQ, in: IJCAI 2011, IJCAI/AAAI, 2011.

[9] A. Gheerbrant, L. Libkin, L. Peterfreund, A. Rogova, Database Theory in Action: Cypher, GQL, and Regular Path Queries, in: ICDT 2025, LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025.

[10] N. Francis, A. Gheerbrant, P. Guagliardo, L. Libkin, V. Marsault, W. Martens, F. Murlak, L. Peterfreund, A. Rogova, D. Vrgoc, GPC: A Pattern Calculus for Property Graphs, in: PODS 2023, ACM, 2023.

[11] M. M. Dimartino, P. T. Wood, A. Calì, A. Poulovassilis, Efficient Ontology-Mediated Query Answering: Extending DL-LiteR and Linear ELH, Journal of Artificial Intelligence Research (2025).

[12] B. Löhnert, N. Augsten, C. Okulmus, M. Ortiz, Towards Practicable Algorithms for Rewriting Graph Queries Beyond DL-Lite, in: The Semantic Web, Springer Nature Switzerland, 2025.

[13] E. Dantsin, T. Eiter, G. Gottlob, A. Voronkov, Complexity and Expressive Power of Logic Programming, ACM Computing Surveys (2001).

[14] J. Arpasi, B. J. Bednarczyk, M. Ortiz, A Horn Extension of DL-Lite with NL Data Complexity, 2026. arXiv:2605.13367.
[15] F. Baader, I. Horrocks, C. Lutz, U. Sattler, An Introduction to Description Logic, Cambridge University Press, 2017.
[16] M. Ortiz, M. Šimkus, Reasoning and Query Answering in Description Logics, in: Reasoning Web, 2012.
[17] J. L. Reutter, M. Romero, M. Y. Vardi, Regular Queries on Graph Databases, Theory Comput. Syst. 61 (2017).
[18] T. J. Schaefer, The Complexity of Satisfiability Problems, in: STOC, 1978.
[19] A. K. Chandra, D. Kozen, L. J. Stockmeyer, Alternation, Journal of the ACM (1981).
[20] F. Baader, S. Borgwardt, M. Lippmann, Query Rewriting for DL-Lite with N-Ary Concrete Domains, in: IJCAI, 2017.
[21] D. Figueira, A. Jeż, A. W. Lin, Data Path Queries Over Embedded Graph Databases, in: PODS 2022, 2022.