THE DATA COMPLEXITY OF DESCRIPTION LOGIC ONTOLOGIES

CARSTEN LUTZ AND FRANK WOLTER

University of Bremen, Germany
e-mail address: clu@uni-bremen.de

University of Liverpool
e-mail address: wolter@liverpool.ac.uk

ABSTRACT. We analyze the data complexity of ontology-mediated querying where the ontologies are formulated in a description logic (DL) of the $\mathcal{ALC}$ family and queries are conjunctive queries, positive existential queries, or acyclic conjunctive queries. Our approach is non-uniform in the sense that we aim to understand the complexity of each single ontology instead of for all ontologies formulated in a certain language. While doing so, we quantify over the queries and are interested, for example, in the question whether all queries can be evaluated in polynomial time w.r.t. a given ontology. Our results include a PTIME/CONP-dichotomy for ontologies of depth one in the description logic $\mathcal{ALCFI}$, the same dichotomy for $\mathcal{ALC}$- and $\mathcal{ALCI}$-ontologies of unrestricted depth, and the non-existence of such a dichotomy for $\mathcal{ALCF}$-ontologies. For the latter DL, we additionally show that it is undecidable whether a given ontology admits PTIME query evaluation. We also consider the connection between PTIME query evaluation and rewritability into (monadic) Datalog.

1. INTRODUCTION

In recent years, the use of ontologies to access instance data has become increasingly popular [PLC+08, KZ14, BO15]. The general idea is that an ontology provides domain knowledge and an enriched vocabulary for querying, thus serving as an interface between the query and the data, and enabling the derivation of additional facts. In this emerging area, called ontology-mediated querying, it is a central research goal to identify ontology languages for which query evaluation scales to large amounts of instance data. Since the size of the data typically dominates the size of the ontology and the size of the query by orders of magnitude, the central measure for such scalability is *data complexity*—the complexity of query evaluation where only the data is considered to be an input, but both the query and the ontology are fixed.

In description logic (DL), ontologies take the form of a TBox, data is stored in an ABox, and the most important classes of queries are conjunctive queries (CQs) and variations thereof, such as positive existential queries (PEQs). A fundamental observation regarding this setup is that, for expressive DLs such as $\mathcal{ALC}$ and $\mathcal{SHIQ}$, the complexity of query evaluation is CONP-complete and thus intractable [Sch93, HMS07, GLHS08].[1] The classical approach to avoiding this problem is to replace $\mathcal{ALC}$ and $\mathcal{SHIQ}$ with less expressive DLs that are 'Horn' in the sense that they

---

[1]When speaking of complexity, we *always* mean data complexity

can be embedded into the Horn fragment of first-order (FO) logic. Horn DLs typicall admit query evaluation in PTIME, examples include a variety of logics from the $\mathcal{EL}$ [BBL05] and DL-Lite families [CDGL$^+$07] as well as Horn-$\mathcal{SHIQ}$, a large fragment of $\mathcal{SHIQ}$ with PTIME query evaluation [HMS07].

It may thus seem that the data complexity of query evaluation in the presence of DL ontologies is understood rather well. However, all results discussed above are at the *level of logics*, i.e., traditional results about data complexity concern a class of TBoxes that is defined in a syntactic way in terms of expressibility in a certain DL language, but no attempt is made to identify more structure *inside* these classes. Such a more fine-grained study, however, seems very natural both from a theoretical and from a practical perspective; in particular, it is well-known that ontologies which emerge in practice tend to use 'expensive' language constructs that can result in CONP-hardness of data complexity, but they typically do so in an extremely restricted and intuitively 'harmless' way. This distinction between hard and harmless cases cannot be analyzed on the level of logics. The aim of this paper is to initiate a more fine-grained study of data complexity that is *non-uniform* in the sense that it does not treat all TBoxes formulated in the same DL in a uniform way.

When taking a non-uniform perspective, there is an important choice regarding the level of granularity. First, one can analyze the complexity on the *level of TBoxes*, quantifying over the actual query. Then, query evaluation for a TBox $\mathcal{T}$ is in PTIME if every query (from the class under consideration) can be evaluated in PTIME w.r.t. $\mathcal{T}$ and it is CONP-hard if there is at least one query that is CONP-hard to evaluate w.r.t. $\mathcal{T}$. And second, one might take an even more fine-grained approach where the query is not quantified away and the aim is to classify the complexity *on the level of ontology-mediated queries (OMQs)*, that is, combinations of a TBox and an actual query. From a practical perspective, both setups make sense; when the actual queries are fixed at the design time of the application, one would probably prefer to work on the level of OMQs whereas the level of TBoxes seems more appropriate when the queries can be freely formulated at application running time. A non-uniform analysis on the level of OMQs has been carried out in [BtCLW14]. In this paper, we concentrate on the level of TBoxes. The ultimate goal of our approach is as follows:

For a fixed DL $\mathcal{L}$ and query language $\mathcal{Q}$, classify all TBoxes $\mathcal{T}$ in $\mathcal{L}$ according to the complexity of evaluating queries from $\mathcal{Q}$ w.r.t. $\mathcal{T}$.

We consider the basic expressive DL $\mathcal{ALC}$, its extensions $\mathcal{ALCI}$ with inverse roles and $\mathcal{ALCF}$ with functional roles, and their union $\mathcal{ALCFI}$. As query languages, we cover CQs, acyclic CQs, and PEQs (which have the same expressive power as unions of conjunctive queries, UCQs, which are thus implicitly also covered). In the current paper, we mainly concentrate on understanding the boundary between PTIME and CONP-hardness of query evaluation w.r.t. DL TBoxes, mostly neglecting other relevant classes such as AC^0, LOGSPACE, and NLOGSPACE.

Our main results are as follows (they apply to all query languages mentioned above).

1. There is a PTIME/CONP-dichotomy for query evaluation w.r.t. $\mathcal{ALCFI}$-TBoxes of depth one, i.e., TBoxes in which no existential or universal restriction is in the scope of another existential or universal restriction.

The proof rests on interesting model-theoretic characterizations of polynomial time CQ-evaluation which are discussed below. Note that this is a relevant case since most TBoxes from practical applications have depth one. In particular, all TBoxes formulated in DL-Lite and its extensions proposed in [CDGL$^+$07, ACKZ09] have depth one, and the same is true for more than 80 percent of the 429 TBoxes in the BioPortal ontology repository. In connection with Point 1 above, we also show that PTIME query evaluation coincides with rewritability into monadic Datalog (with

inequalities, to capture functional roles). As in the case of data complexity, what we mean here is that *all* queries are rewritable into monadic Datalog w.r.t. the TBox $\mathcal{T}$ under consideration.

2. There is a PTIME/CONP-dichotomy for query evaluation w.r.t. $\mathcal{ALCI}$-TBoxes.

This is proved by showing that there is a PTIME/CONP-dichotomy for query evaluation w.r.t. $\mathcal{ALCI}$-TBoxes if and only if there is a PTIME/NP-dichotomy for non-uniform constraint satisfaction problems with finite templates (CSPs). The latter is known as the Feder-Vardi conjecture that was recently proved in [Bul17, Zhu17], as the culmination of a major research programme that combined complexity theory, graph theory, logic, and algebra [BJK05, KS09, Bul11, Bar14]. Our equivalence proof establishes a close link between query evaluation in $\mathcal{ALC}$ and $\mathcal{ALCI}$ and CSP that is relevant for DL research also beyond the stated dichotomy problem. Note that, in contrast to the proof of the Feder-Vardi conjecture, the dichotomy proof for TBoxes of depth one (stated as Point 1 above) is much more elementary. Also, it covers functional roles and establishes equivalence between PTIME query evaluation and rewritability into monadic Datalog, which fails for $\mathcal{ALCI}$-TBoxes of unrestricted depth even when monadic Datalog is replaced with Datalog; this is a consequence of the link to CSPs establishes in this paper.

3. There is no PTIME/CONP-dichotomy for query evaluation w.r.t. $\mathcal{ALCF}$-TBoxes (unless PTIME = NP).

This is proved by showing that, for every problem in CONP, there is an $\mathcal{ALCF}$-TBox for which query evaluation has the same complexity (up to polynomial time reductions); it then remains to apply Ladner's Theorem, which guarantees the existence of NP-intermediate problems. Consequently, we cannot expect an exhaustive classification of the complexity of query evaluation w.r.t. $\mathcal{ALCF}$-TBoxes. Variations of the proof of Point 3 allow us to establish also the following:

4. For $\mathcal{ALCF}$-TBoxes, the following problems are undecidable: PTIME-hardness of query evaluation, CONP-hardness of query evaluation, and rewritability into monadic Datalog and into Datalog (with inequalities).

To prove the results listed above, we introduce two new notions that are of independent interest and general utility. The first one is *materializability* of a TBox $\mathcal{T}$, which means that evaluating a query over an ABox $\mathcal{A}$ w.r.t. $\mathcal{T}$ can be reduced to query evaluation in a single model of $\mathcal{A}$ and $\mathcal{T}$ (a *materialization*). Note that such models play a crucial role in the context of Horn DLs, where they are often called canonical models or universal models. In contrast to the Horn DL case, however, we only require the *existence* of such a model without making any assumptions about its form or construction.

5. If an $\mathcal{ALCFI}$-TBox $\mathcal{T}$ is not materializable, then CQ-evaluation w.r.t. $\mathcal{T}$ is CONP-hard.

We also investigate the nature of materializations. It turns out that if a TBox is materializable for one of the considered query languages, then it is materializable also for all others. The concrete materializations, however, need not agree. To obtain these results, we characterize CQ-materializations in terms of homomorphisms and ELIQ-materializations in terms of simulations (an ELIQ is an $\mathcal{ELI}$-instance query, thus the DL version of an acyclic CQ, with a single answer variable).

Perhaps in contrary to the intuitions that arise from the experience with Horn DLs, materializability of a TBox $\mathcal{T}$ is *not* a sufficient condition for query evaluation w.r.t. $\mathcal{T}$ to be in PTIME (unless PTIME = NP) since the existing materialization might be hard to compute. This leads us to study the notion of *unraveling tolerance* of a TBox $\mathcal{T}$, meaning that answers to acyclic CQs over an ABox $\mathcal{A}$ w.r.t. $\mathcal{T}$ are preserved under unraveling the ABox $\mathcal{A}$. In CSP, unraveling tolerance corresponds to the existence of tree obstructions, a notion that characterizes the well known

arc consistency condition and rewritability into monadic Datalog [FV98, Kro10a]. It can be shown that every TBox formulated in Horn-$\mathcal{ALCFI}$ (the intersection of $\mathcal{ALCFI}$ and Horn-$\mathcal{SHIQ}$) is unraveling tolerant and that there are unraveling tolerant TBoxes which are not equivalent to any Horn-$\mathcal{ALCFI}$-TBox. Thus, the following result yields a rather general (and uniform!) PTIME upper bound for CQ-evaluation.

6. If an $\mathcal{ALCFI}$-TBox $\mathcal{T}$ is unraveling tolerant, then query evaluation w.r.t. $\mathcal{T}$ is in PTIME.

Although the above result is rather general, unraveling tolerance of a TBox $\mathcal{T}$ is *not* a necessary condition for CQ-evaluation w.r.t. $\mathcal{T}$ to be in PTIME (unless PTIME = NP). However, for $\mathcal{ALCFI}$-TBoxes $\mathcal{T}$ *of depth one*, being materializable and being unraveling tolerant turns out to be equivalent. For such TBoxes, we thus obtain that CQ-evalutation w.r.t. $\mathcal{T}$ is in PTIME iff $\mathcal{T}$ is materializable iff $\mathcal{T}$ is unraveling tolerant while, otherwise, CQ-evaluation w.r.t. $\mathcal{T}$ is CONP-hard. This establishes the first main result above.

Our framework also allows one to formally capture some intuitions and beliefs commonly held in the context of CQ-answering in DLs. For example, we show that for every $\mathcal{ALCFI}$-TBox $\mathcal{T}$, CQ-evaluation is in PTIME iff PEQ-evaluation is in PTIME iff ELIQ-evaluation is in PTIME, and the same is true for CONP-hardness and for rewritability into Datalog and into monadic Datalog. In fact, the use of multiple query languages and in particular of $\mathcal{ELI}$-instance queries does not only yield additional results, but is at the heart of our proof strategies. Another interesting observation in this spirit is that an $\mathcal{ALCFI}$-TBox is materializable iff it is convex, a condition that is also called the disjunction property and plays a central role in attaining PTIME complexity for standard reasoning in Horn DLs such as $\mathcal{EL}$, DL-Lite, and Horn-$\mathcal{SHIQ}$; see for example [BBL05, KL07] for more details.

This paper is a significantly extended and revised version of the conference publication [LW12].

Related Work. An early reference on data complexity in DLs is [Sch93], showing CONP-hardness of ELQs in the fragment $\mathcal{ALE}$ of $\mathcal{ALC}$ (an ELQ is an ELIQ in which all edges are directed away from the answer variable). A CONP upper bound for ELIQs in the much more expressive DL $\mathcal{SHIQ}$ was obtained in [HMS07] and generalized to CQs in [GLHS08]. Horn-$\mathcal{SHIQ}$ was first defined in [HMS07], where also a PTIME upper bound for ELIQs is established; the generalization to CQs can be found in [EGOS08]. See also [KL07, Ros07, OCE08, CDL$^+$13] and references therein for the data complexity in DLs and [BGO10, BMRT11] for related work on the guarded fragment and on existential rules.

To the best of our knowledge, the conference version of this paper was first to initiate the study of data complexity in ontology-mediated querying at the level of individual TBoxes and the first to observe a link between this area and CSP. There is, however, a certain technical similarity to the link between view-based query processing for regular path queries (RPQs) and CSP found in [CGLV00, CGLV03b, CGLV03a]. In this case, the recognition problem for perfect rewritings for RPQs can be polynomially reduced to non-uniform CSP and vice versa. On the level of OMQs, the data complexity of ontology-mediated querying with DLs has been studied in [BtCLW14], see also [FKL17]; also here, a connection to CSP plays a central role. In [LSW13, LSW15], the non-uniform data complexity of ontology-mediated query answering is studied in the case where the TBox is formulated in an inexpressive DL of the DL-Lite or $\mathcal{EL}$ family and where individual predicates in the data can be given a closed-world reading, which also gives rise to CONP-hardness of query evaluation; while [LSW13] is considering the level of TBoxes, [LSW15] treats the level of OMQs, establishing a connection to surjective CSPs. Rewritability into Datalog for atomic queries and at the level of OMQs has also been studied in [KNC16]. Finally, we mention [LS17] where a complete

classification of the data complexity of OMQs (also within PTIME) is achieved when the TBox is formulated in $\mathcal{EL}$ and the actual queries are atomic queries.

Recently, the data complexity at the level of TBoxes has been studied also in the guarded fragment and in the two-variable guarded fragment of FO with counting [HLPW17a]. This involves a generalization of the notions of materializability and unraveling tolerance and leads to a variety of PTIME/CONP-dichotomy results. In particular, our dichotomy between Datalog-rewritability and CONP is extended from $\mathcal{ALCIF}$-TBoxes of depth one to $\mathcal{ALCHIF}$-TBoxes of depth two. Using a variant of Ladner's Theorem, several non-dichotomy results for weak fragments of the two-variable guarded fragment with counting of depth two are established and it is shown that PTIME data complexity of query evaluation is undecidable. For $\mathcal{ALCHIQ}$-TBoxes of depth one, though, PTIME data complexity of query evaluation and, equivalently, rewritability into Datalog (with inequalities) is proved to be decidable. In [HLPW17b], the results presented in this paper have been used to show that whenever an $\mathcal{ALCIF}$-TBox of depth one enjoys PTIME query evaluation, then it can be rewritten into a Horn-$\mathcal{ALCIF}$-TBox that gives the same answers to CQs (the converse is trivial). It is also proved that this result does not hold in other cases such as for $\mathcal{ALCHIF}$-TBoxes of depth one.

The work on CSP dichotomies started with Schaefer's PTIME/NP-dichotomy theorem, stating that every CSP defined by a two element template is in PTIME or NP-hard [Sch78]. Schaefer's theorem was followed by dichotomy results for CSPs with (undirected) graph templates [HN90] and several other special cases, leading to the widely known Feder-Vardi conjecture which postulates a PTIME/NP-dichotomy for all CSPs, independently of the size of the template [FV98]. The conjecture has recently been confirmed [Bul17, Zhu17] using an approach to studying the complexity of CSPs via universal algebra [BJK05]. Interesting results have also been obtained for other complexity classes such as AC^0 [ABI$^+$05, LLT07].

2. PRELIMINARIES

We introduce the relevant description logics and query languages, define the fundamental notions studied in this paper, and illustrate them with suitable examples.

We shall be concerned with the DL $\mathcal{ALC}$ and its extensions $\mathcal{ALCI}$, $\mathcal{ALCF}$, and $\mathcal{ALCFI}$. Let N_C, N_R, and N_I denote countably infinite sets of *concept names*, *role names*, and *individual names*, respectively. $\mathcal{ALC}$-*concepts* are constructed according to the rule

$$C, D \ := \ \top \mid \bot \mid A \mid C \sqcap D \mid C \sqcup D \mid \neg C \mid \exists r.C \mid \forall r.C$$

where A ranges over N_C and r ranges over N_R. $\mathcal{ALCI}$-*concepts* admit, in addition, *inverse roles* from the set $N_R^- = \{r^- \mid r \in N_R\}$, which can be used in place of role names. Thus, $A \sqcap \exists r^-.\forall s.B$ is an example of an $\mathcal{ALCI}$-concept. To avoid heavy notation, we set $r^- := s$ if $r = s^-$ for a role name s; in particular, we thus have $(r^-)^- = r$.

In DLs, ontologies are formalized as TBoxes. An $\mathcal{ALC}$-*TBox* is a finite set of *concept inclusions (CIs)* $C \sqsubseteq D$, where C, D are $\mathcal{ALC}$ concepts, and $\mathcal{ALCI}$-TBoxes are defined analogously. An $\mathcal{ALCF}$-*TBox (resp. $\mathcal{ALCFI}$-TBox)* is an $\mathcal{ALC}$-TBox (resp. $\mathcal{ALCI}$-TBox) that additionally admits functionality assertions $func(r)$, where $r \in N_R$ (resp. $r \in N_R \cup N_R^-$), declaring that r is interpreted as a partial function. Note that there is no such thing as an $\mathcal{ALCF}$-concept or an $\mathcal{ALCFI}$-concept, as the extension with functional roles does not change the concept language.

An *ABox* $\mathcal{A}$ is a non-empty finite set of assertions of the form $A(a)$ and $r(a, b)$ with $A \in N_C$, $r \in N_R$, and $a, b \in N_I$. In some cases, we drop the finiteness condition on ABoxes and then

explicitly speak about *infinite ABoxes*. We use $\mathsf{Ind}(\mathcal{A})$ to denote the set of individual names used in the ABox $\mathcal{A}$ and sometimes write $r^-(a,b) \in \mathcal{A}$ instead of $r(b,a) \in \mathcal{A}$.

The semantics of DLs is given by *interpretations* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set and $\cdot^{\mathcal{I}}$ maps each concept name $A \in \mathsf{N_C}$ to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and each role name $r \in \mathsf{N_R}$ to a binary relation $r^{\mathcal{I}}$ on $\Delta^{\mathcal{I}}$. The extension $(r^-)^{\mathcal{I}}$ of r^- under the interpretation $\mathcal{I}$ is defined as the converse relation $(r^{\mathcal{I}})^{-1}$ of $r^{\mathcal{I}}$ and the extension $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ of concepts under the interpretation $\mathcal{I}$ is defined inductively as follows:

$$
\begin{aligned}
\top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} &= \emptyset \\
(\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\exists r.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \exists d' \in \Delta^{\mathcal{I}} : (d, d') \in r^{\mathcal{I}} \text{ and } d' \in C^{\mathcal{I}}\} \\
(\forall r.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \forall d' \in \Delta^{\mathcal{I}} : (d, d') \in r^{\mathcal{I}} \text{ implies } d' \in C^{\mathcal{I}}\}
\end{aligned}
$$

An interpretation $\mathcal{I}$ *satisfies* a CI $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, an assertion $A(a)$ if $a \in A^{\mathcal{I}}$, an assertion $r(a,b)$ if $(a,b) \in r^{\mathcal{I}}$, and a functionality assertion $\mathsf{func}(r)$ if $r^{\mathcal{I}}$ is a partial function. Note that we make the *standard name assumption*, that is, individual names are not interpreted as domain elements (like first-order constants), but as themselves. This assumption is common both in DLs and in database theory. The results in this paper do not depend on it.

An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ if it satisfies all CIs in $\mathcal{T}$ and $\mathcal{I}$ is a *model* of an ABox $\mathcal{A}$ if all individual names from $\mathcal{A}$ are in in $\Delta^{\mathcal{I}}$ and $\mathcal{I}$ satisfies all assertions in $\mathcal{A}$. We call an ABox $\mathcal{A}$ *consistent w.r.t. a TBox* $\mathcal{T}$ if $\mathcal{A}$ and $\mathcal{T}$ have a joint model.

We consider several query languages. A *positive existential query (PEQ)* $q(\vec{x})$ is a first-order formula with free variables $\vec{x} = x_1, \ldots, x_n$ constructed from atoms $A(x)$ and $r(x,y)$ using conjunction, disjunction, and existential quantification, where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and x, y are variables. The variables in $\vec{x}$ are the *answer variables* of $q(\vec{x})$. A PEQ without answer variables is *Boolean*. An *assignment π for $q(\vec{x})$ in an interpretation $\mathcal{I}$* is a mapping from the variables that occur in $q(\vec{x})$ to $\Delta^{\mathcal{I}}$. A tuple $\vec{a} = a_1, \ldots, a_n$ in $\mathsf{Ind}(\mathcal{I})$ is an *answer to $q(\vec{x})$ in $\mathcal{I}$* if there exists an assigment π for $q(\vec{x})$ in $\mathcal{I}$ such that $\mathcal{I} \models^\pi q(\vec{x})$ (in the standard first-order sense) and $\pi(x_i) = a_i$ for $1 \leq i \leq n$. In this case, we write $\mathcal{I} \models q(\vec{a})$. A tuple $\vec{a} \in \mathsf{Ind}(\mathcal{A})$, $\mathcal{A}$ an ABox, is a *certain answer to $q(\vec{x})$ in $\mathcal{A}$ w.r.t.* a TBox $\mathcal{T}$, in symbols $\mathcal{T}, \mathcal{A} \models q(\vec{a})$, if $\mathcal{I} \models q(\vec{a})$ for all models $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$. Computing certain answers to a query in the sense just defined is the main querying problem we are interested in. Although this paper focusses on the theoretical aspects of query answering, we given a concrete example that illustrates the usefulness of query answering with DL ontologies.

Example 1. *Let*

$$
\begin{aligned}
\mathcal{T} &= \{\mathsf{Professer} \sqsubseteq \mathsf{Academic}, \quad \mathsf{Professor} \sqsubseteq \exists \mathsf{gives}.\mathsf{Course}\} \\
\mathcal{A} &= \{\mathsf{Student}(\mathsf{john}), \mathsf{supervisedBy}(\mathsf{john}, \mathsf{mark}), \mathsf{Professor}(\mathsf{mark})\} \\
q(x,y) &= \exists z \, \mathsf{Student}(x) \wedge \mathsf{supervisedBy}(x,y) \wedge \mathsf{Academic}(y) \wedge \mathsf{gives}(y,z) \wedge \mathsf{Course}(z)
\end{aligned}
$$

Thus the query asks to return all pairs that consist of a student x and an academic y such that x is supervised by y and y gives a course. Although this information is not directly present in the ABox, because of the TBox it is easy to see that (john, mark) *is a certain answer.*

Apart from PEQs, we also study several fragments thereof. A *conjunctive query (CQ)* is a PEQ without disjunction. We generally assume that a CQ $q(\vec{x})$ takes the form $\exists \vec{y}\, \varphi(\vec{x}, \vec{y})$, where $\varphi(\vec{x}, \vec{y})$ is a conjunction of atoms of the form $A(x)$ and $r(x, y)$. It is easy to see that every PEQ $q(\vec{x})$ is equivalent to a disjunction $\bigvee_{i \in I} q_i(\vec{x})$, where each $q_i(\vec{x})$ is a CQ (such a disjunction is often called a *union of conjunctive queries, or UCQ*).

To introduce simple forms of CQs that play a crucial role in this paper, we recall two further DLs that we use here for mainly querying purposes. $\mathcal{EL}$-*concepts* are constructed from $\mathsf{N_C}$ and $\mathsf{N_R}$ according to the syntax rule

$$C, D := \top \mid A \mid C \sqcap D \mid \exists r.C$$

and $\mathcal{ELI}$-*concepts* additionally admit inverse roles. An $\mathcal{EL}$-*TBox* is a finite set of concept inclusions $C \sqsubseteq D$ with C and D $\mathcal{EL}$-concepts, and likewise for $\mathcal{ELI}$-*TBoxes*.

We now use $\mathcal{EL}$ and $\mathcal{ELI}$ to define restricted classes of CQs. If C is an $\mathcal{ELI}$-concept and x a variable, then $C(x)$ is called an $\mathcal{ELI}$ *query (ELIQ)*; if C is an $\mathcal{EL}$-concept, then $C(x)$ is called an $\mathcal{EL}$ *query (ELQ)*. Note that every ELIQ can be regarded as an acyclic CQ with one answer variable, and indeed this is an equivalent definition of ELIQs; in the case of ELQs, it is additionally the case that all edges are directed away from the answer variable. For example, the ELIQ $\exists r.(A \sqcap \exists s^-.B)(x)$ is equivalent to the acyclic CQ

$$\exists y_1 \exists y_2 (r(x, y_1) \wedge A(y_1) \wedge s(y_2, y_1) \wedge B(y_2)).$$

In what follows, we will not distinguish between an ELIQ and its translation into an acyclic CQ with one answer variable and freely apply notions introduced for PEQs also to ELIQs and ELQs. We also sometimes slightly abuse notation and use PEQ to denote the set of all positive existential queries, and likewise for CQ, ELIQ, and ELQ.

Example 2.
(1) Let $\mathcal{T}_{\exists,r} = \{A \sqsubseteq \exists r.A\}$ and $q(x) = \exists r.A(x)$. Then we have for any ABox $\mathcal{A}$, $\mathcal{T}_{\exists,r}, \mathcal{A} \models q(a)$ iff $A(a) \in \mathcal{A}$ or there are $r(a, b), A(b) \in \mathcal{A}$.

(2) Let $\mathcal{T}_{\exists,l} = \{\exists r.A \sqsubseteq A\}$ and $q(x) = A(x)$. For any ABox $\mathcal{A}$, $\mathcal{T}_{\exists,l}, \mathcal{A} \models q(a)$ iff there is an r-path in $\mathcal{A}$ from a to some b with $A(b) \in \mathcal{A}$; that is, there are $r(a_0, a_1), \ldots, r(a_{n-1}, a_n) \in \mathcal{A}$, $n \geq 0$, with $a_0 = a$, $a_n = b$, and $A(b) \in \mathcal{A}$.

(3) Consider an undirected graph G represented as an ABox $\mathcal{A}$ with assertions $r(a, b), r(b, a) \in \mathcal{A}$ iff there is an edge between a and b. Let $A_1, \ldots, A_k, M$ be concept names. Then G is k-colorable iff $\mathcal{T}_k, \mathcal{A} \not\models \exists x\, M(x)$, where

$$\begin{aligned} \mathcal{T}_k \;=\; & \{A_i \sqcap A_j \sqsubseteq M \mid 1 \leq i < j \leq k\} \cup \\ & \{A_i \sqcap \exists r.A_i \sqsubseteq M \mid 1 \leq i \leq k\} \cup \\ & \{\top \sqsubseteq \bigsqcup_{1 \leq i \leq k} A_i\}. \end{aligned}$$

Instead of actually computing certain answers to queries, we concentrate on the query evaluation problem, which is the decision problem version of query answering. We next introduce this problem along with associated notions of complexity. An *ontology-mediated query (OMQ)* is a pair $(\mathcal{T}, q(\vec{x}))$ with $\mathcal{T}$ a TBox $\mathcal{T}$ and $q(\vec{x})$ a query. The *query evaluation problem for* $(\mathcal{T}, q(\vec{x}))$ is to decide, given an ABox $\mathcal{A}$ and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A})$, whether $\mathcal{T}, \mathcal{A} \models q(\vec{a})$. We shall typically be interested in joint complexity bounds for evaluating *all* OMQs formulated in a query language $\mathcal{Q}$ of interest w.r.t. a given TBox $\mathcal{T}$.

Definition 3. Let $\mathcal{T}$ be an $\mathcal{ALCFI}$-TBox and let $\mathcal{Q} \in \{\text{CQ}, \text{PEQ}, \text{ELIQ}, \text{ELQ}\}$. Then

- $\mathcal{Q}$-*evaluation w.r.t.* $\mathcal{T}$ *is in* PTIME if for every $q(\vec{x}) \in \mathcal{Q}$, the query evaluation problem for $(\mathcal{T}, q(\vec{x}))$ is in PTIME.
- $\mathcal{Q}$-*evaluation w.r.t.* $\mathcal{T}$ *is* CONP-*hard* if there exists $q(\vec{x}) \in \mathcal{Q}$ such that the query evaluation problem for $(\mathcal{T}, q(\vec{x}))$ is CONP-hard.

Note that one should not think of '$\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$' as a decision problem since, informally, this is a collection of infinitely many decision problems, one for each query in $\mathcal{Q}$. Instead, one should think of '$\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$ to be in PTIME' (or CONP-hard) as a property of $\mathcal{T}$.

Example 4.

(1) PEQ-evaluation w.r.t. the TBoxes $\mathcal{T}_{\exists,r}$ and $\mathcal{T}_{\exists,l}$ from Example 2 is in PTIME. *This follows from the fact that these TBoxes are $\mathcal{EL}$-TBoxes (TBoxes using only $\mathcal{EL}$-concepts) and it is well known that PEQ-evaluation w.r.t. $\mathcal{EL}$-TBoxes is in* PTIME *[KL07].*

(2) Consider the TBoxes $\mathcal{T}_k$ from Example 2 that express k-colorability using the query $\exists x\, M(x)$. For $k \geq 3$, CQ-evaluation w.r.t. $\mathcal{T}_k$ is CONP-hard since k-colorability is NP-hard. However, in contrast to the tractability of 2-colorability, CQ-evaluation w.r.t. $\mathcal{T}_2$ is still CONP-hard. This follows from Theorem 18 below and, intuitively, is the case because $\mathcal{T}_2$ 'entails a disjunction': for $\mathcal{A} = \{B(a)\}$, we have $\mathcal{T}_2, \mathcal{A} \models A_1(a) \vee A_2(a)$, but neither $\mathcal{T}_2, \mathcal{A} \models A_1(a)$ nor $\mathcal{T}_2, \mathcal{A} \models A_2(a)$.

In addition to the classification of TBoxes according to whether query evaluation is in PTIME or CONP-hard, we are also interested in whether OMQs based on the TBox are rewritable into more classical database querying languages, in particular into Datalog and into monadic Datalog.

A *Datalog rule* ρ has the form $S(\vec{x}) \leftarrow R_1(\vec{y}_1) \wedge \cdots \wedge R_n(\vec{y}_n)$ where $n > 0$, S is a relation symbol, and $R_1, \ldots, R_n$ are relation symbols, that is, concept names and role names. We refer to $S(\vec{x})$ as the *head* of ρ and $R_1(\vec{y}_1) \wedge \cdots \wedge R_n(\vec{y}_n)$ as its *body*. Every variable in the head of ρ is required to occur also in its body. A *Datalog program* Π is a finite set of Datalog rules with a selected *goal relation* goal that does not occur in rule bodies. Relation symbols that occur in the head of at least one rule are called *intensional relation symbols (IDBs)*, the remaining symbols are called *extensional relation symbols (EDBs)*. Note that, by definition, goal is an IDB. The *arity* of the program is the arity of the goal relation. Programs of arity zero are called *Boolean*. A Datalog program that uses only IDBs of arity one, with the possible exception of the goal relation, is called *monadic*.

For an ABox $\mathcal{A}$, a Datalog program Π, and $\vec{a}$ from $\text{Ind}(\mathcal{A})$ of the same length as the arity of goal, we write $\mathcal{A} \models \Pi(\vec{a})$ if Π returns $\vec{a}$ as an answer on $\mathcal{A}$, defined in the usual way [CGT89]. A (monadic) Datalog program Π is a *(monadic) Datalog-rewriting* of an OMQ $(\mathcal{T}, q(\vec{x}))$ if for all ABoxes $\mathcal{A}$ and $\vec{a}$ from $\text{Ind}(\mathcal{A})$, $\mathcal{T}, \mathcal{A} \models q(\vec{a})$ iff $\mathcal{A} \models \Pi(\vec{a})$. In this case the OMQ $(\mathcal{T}, q(\vec{x}))$ is called *(monadic) Datalog-rewritable*. When working with DLs such as $\mathcal{ALCFI}$ that include functional roles, it is more natural to admit the use of inequalities in the bodies of Datalog rules instead of working with 'pure' programs. We refer to such extended programs as (monadic) Datalog$^{\neq}$ programs and accordingly speak of (monadic) Datalog$^{\neq}$-rewritability.

Example 5.
(1) The OMQ $(\mathcal{T}_{\exists,l}, A(x))$ from Example 2 expressing a form of reachability is rewritable into the monadic Datalog program

$$\text{goal}(x) \leftarrow P(x), \quad P(x) \leftarrow A(x), \quad P(x) \leftarrow r(x,y) \wedge P(y).$$

(2) The OMQ $(\mathcal{T}_k, \exists x\, M(x))$ from Example 2 is Datalog-rewritable when $k = 2$ since non-2-colorability can be expressed by a Datalog program (but not as a monadic one). For $k \geq 3$, non-k-colorability cannot be expressed by a Datalog program (in fact, not even by a $\text{Datalog}^{\neq}$ program) [ACY91].

(3) The OMQ $(\{\mathsf{func}(r)\}, \exists x\, M(x))$ is rewritable into the monadic $\text{Datalog}^{\neq}$ program

$$\mathsf{goal}() \leftarrow r(x, y_1) \wedge r(x, y_2) \wedge y_1 \neq y_2, \quad \mathsf{goal}() \leftarrow M(x)$$

but is not rewritable into pure Datalog.

Definition 6. Let $\mathcal{T}$ be an $\mathcal{ALCFI}$-TBox and let $\mathcal{Q} \in \{\mathrm{CQ}, \mathrm{PEQ}, \mathrm{ELIQ}, \mathrm{ELQ}\}$. Then $\mathcal{T}$ is *(monadic) $\text{Datalog}^{\neq}$-rewritable* for $\mathcal{Q}$ if $(\mathcal{T}, q(\vec{x}))$ is (monadic) $\text{Datalog}^{\neq}$-rewritable for every $q(\vec{x}) \in \mathcal{Q}$.

We would like to stress that the extension of Datalog to $\text{Datalog}^{\neq}$ makes sense only in the presence of functional roles. In fact, it follows from the CSP connection established in Section 6 and the results in [FV03] that for $\mathcal{ALCI}$-TBoxes, $\text{Datalog}^{\neq}$-rewritability for $\mathcal{Q}$ agrees with Datalog-rewritability for $\mathcal{Q}$, for all query classes $\mathcal{Q}$ considered in this paper.

Example 7. *It is folklore that every $\mathcal{EL}$-TBox is monadic Datalog-rewritable for ELQ, ELIQ, CQs, and PEQs. Thus, this applies in particular to the $\mathcal{EL}$-TBoxes $\mathcal{T}_{\exists,l}$ and $\mathcal{T}_{\exists,r}$ from Example 2. A concrete construction of Datalog-rewritings for ELIQs can be found in the proof of Theorem 25 below. In contrast, the $\mathcal{ALC}$-TBox $\mathcal{T}_k$ from Example 2 is not $\text{Datalog}^{\neq}$-rewritable for ELQ when $k \geq 3$ since the OMQ $(\mathcal{T}_k, \exists x\, M(x))$ is not Datalog-rewritable, by Example 5 (2).*

$\text{Datalog}^{\neq}$-programs can be evaluated in PTIME [CGT89] in data complexity, and thus $\text{Datalog}^{\neq}$-rewritability for $\mathcal{Q}$ of a TBox $\mathcal{T}$ implies that $\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$ is in PTIME in data complexity. We shall see later that the converse direction does not hold in general.

We will often be concerned with homomorphisms between ABoxes and between interpretations, defined next. Let $\mathcal{A}$ and $\mathcal{B}$ be ABoxes. A function $h : \mathsf{Ind}(\mathcal{A}) \to \mathsf{Ind}(\mathcal{B})$ is a *homomorphism from $\mathcal{A}$ to $\mathcal{B}$* if it satisfies the following conditions:

(1) $A(a) \in \mathcal{A}$ implies $A(h(a)) \in \mathcal{B}$ and

(2) $r(a, b) \in \mathcal{A}$ implies $r(h(a), h(b)) \in \mathcal{B}$.

We say that h *preserves* $I \subseteq \mathsf{N}_\mathsf{I}$ if $h(a) = a$ for all $a \in I$. Homomorphisms from an interpretation $\mathcal{I}$ to an interpretation $\mathcal{J}$ are defined analogously as functions $h : \Delta^{\mathcal{I}} \to \Delta^{\mathcal{J}}$. Note that these two notions are in fact identical since, up to presentation, ABoxes and finite interpretations are the same thing. In what follows we will not always distinguish between the two presentations.

3. MATERIALIZABILITY

We introduce materializability as a central notion for analyzing the complexity and rewritability of TBoxes. A materialization of a TBox $\mathcal{T}$ and ABox $\mathcal{A}$ for a class of queries $\mathcal{Q}$ is a (potentially infinite) model of $\mathcal{T}$ and $\mathcal{A}$ that gives the same answers to queries in $\mathcal{Q}$ as $\mathcal{T}$ and $\mathcal{A}$ do. It is not difficult to see that a materialization for ELIQs is not necessarily a materialization for CQs and that a materialization for ELQs is not necessarily a materialization for ELIQs. We shall call a TBox $\mathcal{T}$ *materializable* for a query language $\mathcal{Q}$ if for every ABox $\mathcal{A}$ that is consistent w.r.t. $\mathcal{T}$, there is a materialization of $\mathcal{T}$ and $\mathcal{A}$ for $\mathcal{Q}$. Interestingly, we show that materializability of $\mathcal{ALCFI}$-TBoxes does *not* depend on whether one considers ELIQs, CQs, or PEQs. This result allows us to simply talk about materializable TBoxes, independently of the query language considered. The

fundamental result linking materializability of a TBox to the complexity of query evaluation is that ELIQ-evaluation is CONP-hard w.r.t. non-materializable $\mathcal{ALCFI}$-TBoxes. As another application of materializability, we show that for $\mathcal{ALCFI}$-TBoxes, PTIME query evaluation, CONP-hardness of query evaluation, and Datalog$^{\neq}$-rewritability also do not depend on the query language. In the case of $\mathcal{ALCF}$, materializability for ELIQs additionally coincides with materializability for ELQs.

Definition 8. Let $\mathcal{T}$ be an $\mathcal{ALCFI}$-TBox and $\mathcal{Q} \in \{\text{CQ}, \text{PEQ}, \text{ELIQ}, \text{ELQ}\}$. Then

(1) a model $\mathcal{I}$ of $\mathcal{T}$ and an ABox $\mathcal{A}$ is a *$\mathcal{Q}$-materialization of $\mathcal{T}$ and $\mathcal{A}$* if for all queries $q(\vec{x}) \in \mathcal{Q}$ and $\vec{a} \subseteq \text{Ind}(\mathcal{A})$, we have $\mathcal{I} \models q(\vec{a})$ iff $\mathcal{T}, \mathcal{A} \models q(\vec{a})$;

(2) $\mathcal{T}$ is *$\mathcal{Q}$-materializable* if for every ABox $\mathcal{A}$ that is consistent w.r.t. $\mathcal{T}$, there exists a $\mathcal{Q}$-materialization of $\mathcal{T}$ and $\mathcal{A}$.

In Point (1) of Definition 8, it is important that the materialization $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ is a model of $\mathcal{T}$ and $\mathcal{A}$. In fact, for an ABox $\mathcal{A}$ that is consistent w.r.t. $\mathcal{T}$, we can *always* find an interpretation $\mathcal{I}$ such that for every CQ $q(\vec{x})$ and $\vec{a} \subseteq \text{Ind}(\mathcal{A})$, $\mathcal{I} \models q(\vec{a})$ iff $\mathcal{T}, \mathcal{A} \models q(\vec{a})$. In particular, the direct product of all (up to isomorphisms) countable models of $\mathcal{T}$ and $\mathcal{A}$ can serve as such an $\mathcal{I}$. However, the interpretation is, in general, not a model of $\mathcal{T}$.

Note that a $\mathcal{Q}$-materialization can be viewed as a more abstract version of the *canonical* or *minimal* or *universal* model as often used in the context of 'Horn DLs' such as $\mathcal{EL}$ and DL-Lite [LTW09, KLT$^+$10, BO15] and more expressive ontology languages based on tuple-generating dependencies (tgds) [CGK13] as well as in data exchange [FKMP05]. In fact, the ELQ-materialization in the next example is exactly the 'compact canonical model' from [LTW09].

Example 9.

(1) Let $\mathcal{T}_{\exists,l} = \{\exists r.A \sqsubseteq A\}$ be as in Example 2 and let $\mathcal{A}$ be an ABox. Let $\mathcal{I}$ be the interpretation obtained from $\mathcal{A}$ by adding to $A^{\mathcal{I}}$ all $a \in \text{Ind}(\mathcal{A})$ such that there exists an r-path from a to some b with $A(b) \in \mathcal{A}$. Then $\mathcal{I}$ is a PEQ-materialization of $\mathcal{T}$ and $\mathcal{A}$ and so $\mathcal{T}$ is PEQ-materializable.

(2) Let $\mathcal{T}_{\exists,r} = \{A \sqsubseteq \exists r.A\}$ be as in Example 2 and let $\mathcal{A}$ be an ABox with at least one assertion of the form $A(a)$. To obtain an ELQ-materialization $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$, start with $\mathcal{A}$ as an interpretation, add a fresh domain element d_r to $\Delta^{\mathcal{I}}$ and to $A^{\mathcal{I}}$, and extend $r^{\mathcal{I}}$ with (a, d_r) and (d_r, d_r) for all $A(a) \in \mathcal{A}$. Thus $\mathcal{T}_{\exists,r}$ is ELQ-materializable.

(3) The TBox $\mathcal{T} = \{A \sqsubseteq A_1 \sqcup A_2\}$ is not ELQ-materializable. To see this let $\mathcal{A} = \{A(a)\}$. Then no model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ is an ELQ-materialization of $\mathcal{T}$ and $\mathcal{A}$ as it satisfies $a \in A_1^{\mathcal{I}}$ or $a \in A_2^{\mathcal{I}}$ but neither $\mathcal{T}, \mathcal{A} \models A_1(a)$ nor $\mathcal{T}, \mathcal{A} \models A_2(a)$.

Trivially, every PEQ-materialization is a CQ-materialization, every CQ-materialization is an ELIQ-materialization and every ELIQ-materialization is an ELQ-materialization. Conversely, it follows directly from the fact that each PEQ is equivalent to a disjunction of CQs that every CQ-materialization is also a PEQ-materialization. In contrast, the following example demonstrates that ELQ-materializations are different from ELIQ-materializations. A similar argument separates ELIQ-materializations from CQ-materializations.

Example 10. *Let $\mathcal{T}_{\exists,r}$ be as in Example 9,*

$$\mathcal{A} = \{B_1(a), B_2(b), A(a), A(b)\} \text{ and}$$
$$q(x) = (B_1 \sqcap \exists r.\exists r^-.B_2)(x),$$

Then the ELQ-materialization $\mathcal{I}$ from Example 9 (2) is not a $\mathcal{Q}$-materialization for any $\mathcal{Q}$ from the set of query languages ELIQ, CQ, PEQ. For example, we have $\mathcal{I} \models q(a)$, but $\mathcal{T}, \mathcal{A} \not\models q(a)$. An ELIQ/CQ/PEQ-materialization of $\mathcal{T}$ and $\mathcal{A}$ is obtained by unfolding $\mathcal{I}$ (see below): instead of using

only one additional domain element d_r as a witness for $\exists r.A$, we attach to both a and b an infinite r-path of elements that satisfy A. Note that every CQ/PEQ-materialization of $\mathcal{T}_{\exists,r}$ and $\mathcal{A}$ must be infinite.

We will sometimes restrict our attention to materializations $\mathcal{I}$ that are countable and *generated*, i.e, every $d \in \Delta^{\mathcal{I}}$ is reachable from some $a \in \Delta^{\mathcal{I}} \cap N_I$ in the undirected graph

$$G_{\mathcal{I}} = (\Delta^{\mathcal{I}}, \{\{d, d'\} \mid (d, d') \in \bigcup_{r \in N_R} r^{\mathcal{I}}\}).$$

The following lemma shows that we can make that assumption without loss of generality.

Lemma 11. *Let $\mathcal{T}$ be an $\mathcal{ALCFI}$-TBox, $\mathcal{A}$ an ABox, and $\mathcal{Q} \in \{CQ, PEQ, ELIQ, ELQ\}$. If $\mathcal{I}$ is a $\mathcal{Q}$-materialization of $\mathcal{T}$ and $\mathcal{A}$, then there exists a subinterpretation $\mathcal{J}$ of $\mathcal{I}$ that is a countable and generated $\mathcal{Q}$-materialization of $\mathcal{T}$ and $\mathcal{A}$.*

Proof. Let $\mathcal{I}$ be a $\mathcal{Q}$-materialization of $\mathcal{T}$ and $\mathcal{A}$. To construct $\mathcal{J}$ we apply a standard selective filtration procedure to $\mathcal{I}$. More precisely, we identify a sequence $\mathsf{Ind}(\mathcal{A}) = S_0 \subseteq S_1 \subseteq \cdots \subseteq \Delta^{\mathcal{I}}$ and then define $\mathcal{J}$ to be the restriction of $\mathcal{I}$ to $\bigcup_i S_i$. Let $\mathcal{C}$ be the set of all concepts of the form $\exists r.C$ that occur in $\mathcal{T}$ and of all concepts $\exists r.\neg C$ such that $\forall r.C$ occurs in $\mathcal{T}$. Assume S_i has already been defined. Then define S_{i+1} as the union of S_i and, for every $d \in S_i$ and concept $\exists r.C \in \mathcal{C}$ with $d \in (\exists r.C)^{\mathcal{I}}$, an arbitrary $d' \in \Delta^{\mathcal{I}}$ with $(d, d') \in r^{\mathcal{I}}$ and $d' \in C^{\mathcal{I}}$ (unless such a d' exists already in S_i). It is easy to see that $\mathcal{J}$ is a countable and generated $\mathcal{Q}$-materialization of $\mathcal{T}$ and $\mathcal{A}$. ❏

3.1. **Model-Theoretic Characterizations of Materializability.** We characterize materializations using simulations and homomorphisms. This sheds light on the nature of materializations and establishes a close connection between materializations and initial models as studied in model theory, algebraic specification, and logic programming [Mal71, MG85, Mak87].

   A *simulation from an interpretation $\mathcal{I}_1$ to an interpretation $\mathcal{I}_2$* is a relation $S \subseteq \Delta^{\mathcal{I}_1} \times \Delta^{\mathcal{I}_2}$ such that

(1) for all $A \in N_C$: if $d_1 \in A^{\mathcal{I}_1}$ and $(d_1, d_2) \in S$, then $d_2 \in A^{\mathcal{I}_2}$;
(2) for all $r \in N_R$: if $(d_1, d_2) \in S$ and $(d_1, d_1') \in r^{\mathcal{I}_1}$, then there exists $d_2' \in \Delta^{\mathcal{I}_2}$ such that $(d_1', d_2') \in S$ and $(d_2, d_2') \in r^{\mathcal{I}_2}$;
(3) for all $a \in \Delta^{\mathcal{I}_1} \cap N_I$: $a \in \Delta^{\mathcal{I}_2}$ and $(a, a) \in S$.

Note that, by Condition (3), domain elements that are individual names need to be respected by simulations while other domain elements need not. In database parlance, the latter are thus treated as *labeled nulls*, that is, while their existence is important, their identity is not.

   We call a simulation S an *i-simulation* if Condition (2) is satisfied also for inverse roles. Note that S is a homomorphism preserving $\Delta^{\mathcal{I}_1} \cap N_I$ if S is a function with domain $\Delta^{\mathcal{I}}$. We remind the reader of the following characterizations of ELQs using simulations, ELIQs using i-simulations, and CQs using homomorphisms (see e.g. [LW10]). An interpretation $\mathcal{I}$ has *finite outdegree* if the undirected graph $G_{\mathcal{I}}$ has finite outdegree.

Lemma 12. *Let $\mathcal{I}$ and $\mathcal{J}$ be interpretations such that $\Delta^{\mathcal{I}} \cap N_I$ is finite, $\mathcal{I}$ is countable and generated, and $\mathcal{J}$ has finite outdegree. Then the following conditions are equivalent (where none of the assumed conditions on $\mathcal{I}$ and $\mathcal{J}$ is required for (2) $\Rightarrow$ (1)).*

(1) For all ELIQs $C(x)$ and $a \in \Delta^{\mathcal{I}} \cap N_I$: if $\mathcal{I} \models C(a)$, then $\mathcal{J} \models C(a)$;
(2) There is an i-simulation from $\mathcal{I}$ to $\mathcal{J}$.

The same equivalence holds when ELIQs and i-simulations are replaced by ELQs and simulations, respectively. Moreover, the following conditions are equivalent (where none of the assumed conditions on $\mathcal{I}$ and $\mathcal{J}$ is required for (5) $\Rightarrow$ (3)).

    (3) For all PEQs $q(\vec{x})$ and $\vec{a} \subseteq \Delta^{\mathcal{I}} \cap \mathsf{N_I}$: if $\mathcal{I} \models q(\vec{a})$, then $\mathcal{J} \models q(\vec{a})$;

    (4) For all CQs $q(\vec{x})$ and $\vec{a} \subseteq \Delta^{\mathcal{I}} \cap \mathsf{N_I}$: if $\mathcal{I} \models q(\vec{a})$, then $\mathcal{J} \models q(\vec{a})$;

    (5) There is a homomorphism from $\mathcal{I}$ to $\mathcal{J}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$.

Proof. We prove the equivalence of (3)-(5). The equivalence of (1) and (2) is similar (both for ELIQs and ELQs) but simpler and left to the reader. The implication (3) $\Rightarrow$ (4) is trivial. For the proof of (4) $\Rightarrow$ (5), assume that $\mathcal{I}$ is countable and generated and let $\mathcal{J}$ have finite outdegree. We first assume that only a finite set Σ of concept and role names have a non-empty interpretation in $\mathcal{I}$ and then generalize the result to arbitrary $\mathcal{I}$. Assume that (4) holds. First observe that for every finite subset X of $\Delta^{\mathcal{I}}$ there is a homomorphism h_X preserving $X \cap \mathsf{N_I}$ from the subinterpretation $\mathcal{I}_{\restriction X}$ of $\mathcal{I}$ induced by X into $\mathcal{J}$: associate with every $d \in X$ a variable x_d and regard $\mathcal{I}_{\restriction X}$ as the CQ

$$q_X(\vec{x}) = \exists \vec{y} \bigwedge_{d \in X \cap A^{\mathcal{I}}} A(x_d) \wedge \bigwedge_{(d,d') \in (X \times X) \cap r^{\mathcal{I}}} r(x_d, x_{d'}),$$

where $\vec{x}$ comprises the variables in $\{x_a \mid a \in X \cap \mathsf{N_I}\}$ and $\vec{y}$ comprises the variables x_d with $d \in X \setminus \mathsf{N_I}$ (q_X is a CQ by our assumption that only finitely many concept and role names have non-empty interpretation). For the assignment $\pi(x_d) = d$, we have $\mathcal{I} \models_\pi \varphi(\vec{x}, \vec{y})$. Thus $\mathcal{I} \models_\pi q_X(\vec{x})$ and so, by (2), $\mathcal{J} \models_\pi q_X(\vec{x})$. Consequently, there exists an assignment π' for $q_X(\vec{x})$ in $\mathcal{J}$ which coincides with π on $\{x_a \mid a \in X \cap \mathsf{N_I}\}$ such that $\mathcal{J} \models_{\pi'} \varphi(\vec{x}, \vec{y})$. Let $h_X(d) = \pi'(d)$ for $d \in X$. Then h_X is a homomorphism from $\mathcal{I}_{\restriction X}$ to $\mathcal{J}$ preserving $X \cap \mathsf{N_I}$, as required.

    We now lift the homomorphisms h_X to a homomorphism h from $\mathcal{I}$ to $\mathcal{J}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$. Since $\mathcal{I}$ is countable and generated, there exists a sequence $X_0 \subseteq X_1 \subseteq \cdots$ of finite subsets of $\Delta^{\mathcal{I}}$ such that $X_0 = \Delta^{\mathcal{I}} \cap \mathsf{N_I}$, $\bigcup_{i \geq 0} X_i = \Delta^{\mathcal{I}}$, and for all $d \in X_i$ there exists a path in X_i from some $a \in X_0$ to d.

    By the observation above, we find homomorphisms h_{X_i} from $\mathcal{I}_{\restriction X_i}$ to $\mathcal{J}$ preserving $X_i \cap \mathsf{N_I}$, for $i \geq 0$. Let $d_0, d_1 \ldots$ be an enumeration of $\Delta^{\mathcal{I}}$. We define the required homomorphism h as the limit of a sequence $h_0 \subseteq h_1 \subseteq \cdots$, where each h_n has domain $\{d_0, \ldots, d_n\}$ and where we ensure for each h_n and all $d \in \{d_0, \ldots, d_n\}$ that there are infinitely many j with $h_n(d) = h_{\restriction X_j}(d)$. Observe that since $\mathcal{J}$ has finite outdegree and since for all $d \in X_i$, there exists a path in X_i from some $a \in X_0$ to d, for each $d \in \Delta^{\mathcal{I}}$ there exist only finitely many distinct values in $\{h_{\restriction X_i}(d) \mid i \geq 0\}$. By the pigeonhole principle, there thus exist infinitely many j with the same value $h_{X_j}(d)$. For $h_0(d_0)$ we take such a value for d_0. Assume h_n has been defined and assume that the set $I = \{j \mid h_n(d) = h_{\restriction X_j}(d)$ for all $d \in \{d_0, \ldots, d_n\}\}$ is infinite. Again by the pigeonhole principle, we find a value $e \in \Delta^{\mathcal{J}}$ such that $h_{X_j}(d_{n+1}) = e$ for infinitely many $j \in I$. We set $h_{n+1}(d_{n+1}) = e$. The function $h = \bigcup_{i \geq 0} h_0$ is a homomorphism from $\mathcal{I}$ to $\mathcal{J}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$, as required.

    To lift this result to arbitrary interpretations $\mathcal{I}$, it is sufficient to prove that the homomorphisms h_X still exist. This can be shown using again the pigeonhole principle. Let $X \subseteq \Delta^{\mathcal{I}}$ be finite. We may assume that for each $d \in X$, there exists a path in X from some $a \in X \cap \mathsf{N_I}$, to d. We have shown that for each finite set Σ of concept and role names, there exists a homomorphism h_X^Σ from the Σ-reduct $\mathcal{I}_X^\Sigma$ of $\mathcal{I}_X$ to $\mathcal{J}$ ($\mathcal{I}_X^\Sigma$ interprets only the symbols in Σ as non-empty). Since $\mathcal{J}$ has finite outdegree, infinitely many h_X^Σ coincide. A straightforward modification of the pigeonhole argument above can now be used to construct the required homomorphism h_X.

For the proof of (5) $\Rightarrow$ (3), assume $\mathcal{I} \models q(\vec{a})$ and let h be a homomorphism from $\mathcal{I}$ to $\mathcal{J}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$. Let π be an assignment for $q(\vec{x})$ in $\mathcal{I}$ witnessing $\mathcal{I} \models q(\vec{a})$. Then the composition $h \circ \pi$ is an assignment for $q(\vec{x})$ in $\mathcal{J}$ witnessing $\mathcal{J} \models q(\vec{a})$. ❑

For the next steps, we need some observations regarding the unfolding of interpretations into forest-shaped interpretations. Let us first make precise what we mean by unfolding. The *i-unfolding* of an interpretation $\mathcal{I}$ is an interpretation $\mathcal{J}$ defined as follows. The domain $\Delta^{\mathcal{J}}$ of $\mathcal{J}$ consists of all words $d_0 r_1 \ldots r_n d_n$ with $n \geq 0$, each d_i from $\Delta^{\mathcal{I}}$ and each r_i a (possibly inverse) role such that

- (a) $d_i \in \mathsf{N_I}$ iff $i = 0$;
- (b) $(d_i, d_{i+1}) \in r_{i+1}^{\mathcal{I}}$ for $0 \leq i < n$;
- (c) if $r_i^- = r_{i+1}$, then $d_{i-1} \neq d_{i+1}$ for $0 < i < n$.

For $d_0 \cdots d_n \in \Delta^{\mathcal{J}}$, we set $\mathsf{tail}(d_0 \cdots d_n) = d_n$. Now set

$$
\begin{aligned}
A^{\mathcal{J}} &= \{w \in \Delta^{\mathcal{J}} \mid \mathsf{tail}(w) \in A^{\mathcal{I}}\} &&\text{for all } A \in \mathsf{N_C} \\
r^{\mathcal{J}} &= (r^{\mathcal{I}} \cap (\mathsf{N_I} \times \mathsf{N_I})) \cup \\
&\quad \{(\sigma, \sigma r d) \mid \sigma, \sigma r d \in \Delta^{\mathcal{J}}\} \cup \{(\sigma r^- d, \sigma) \mid \sigma, \sigma r^- d \in \Delta^{\mathcal{J}}\} &&\text{for all } r \in \mathsf{N_R}.
\end{aligned}
$$

We say that an interpretation $\mathcal{I}$ is *i-unfolded* if it is isomorphic to its own i-unfolding. Clearly, every i-unfolding of an interpretation is i-unfolded.

For $\mathcal{ALCF}$-TBoxes, it is not required to unfold along inverse roles. This is reflected in the *unfolding* of an interpretation $\mathcal{I}$, where in contrast to the i-unfolding we use as the domain the set of all words $d_0 r_1 \ldots r_n d_n$ with $n \geq 0$, each d_i from $\Delta^{\mathcal{I}}$, and each r_i a *role name* such that Conditions (a) and (b) above are satisfied. The interpretation of concept and role names remains the same. We call an interpretation $\mathcal{I}$ *unfolded* if it is isomorphic to its own unfolding. The following lemma summarizes the main properties of unfoldings. Its proof is straightforward and left to the reader.

Lemma 13. *Let $\mathcal{I}$ be an interpretation, $\mathcal{I}^i$ its i-unfolding, and $\mathcal{I}^u$ its unfolding. Then for every interpretation $\mathcal{J}$, the following conditions are satisfied:*

- *(1) the function $f(w) := \mathsf{tail}(w)$, $w \in \Delta^{\mathcal{I}^i}$, is a homomorphism from $\mathcal{I}^i$ to $\mathcal{I}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$;*
- *(2) the function $f(w) := \mathsf{tail}(w)$, $w \in \Delta^{\mathcal{I}^u}$, is a homomorphism from $\mathcal{I}^u$ to $\mathcal{I}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$;*
- *(3) if there is an i-simulation from $\mathcal{I}$ to $\mathcal{J}$, then there is a homomorphism from $\mathcal{I}^i$ to $\mathcal{J}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$;*
- *(4) if there is a simulation from $\mathcal{I}$ to $\mathcal{J}$, then there is a homomorphism from $\mathcal{I}^u$ to $\mathcal{J}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$;*
- *(5) if $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$ with $\mathcal{T}$ an $\mathcal{ALCFI}$-TBox, then $\mathcal{I}^i$ is a model of $\mathcal{T}$ and $\mathcal{A}$;*
- *(6) if $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$ with $\mathcal{T}$ an $\mathcal{ALCF}$-TBox, then $\mathcal{I}^u$ is a model of $\mathcal{T}$ and $\mathcal{A}$.*

An interpretation $\mathcal{I}$ is called *hom-initial in a class $\mathbb{K}$ of interpretations* if for every $\mathcal{J} \in \mathbb{K}$, there exists a homomorphism from $\mathcal{I}$ to $\mathcal{J}$ preserving $\Delta^{\mathcal{I}} \cap \mathsf{N_I}$. $\mathcal{I}$ is called *sim-initial (i-sim-initial) in a class $\mathbb{K}$ of interpretations* if for every $\mathcal{J} \in \mathbb{K}$, there exists a simulation (i-simulation) from $\mathcal{I}$ to $\mathcal{J}$. The following theorem provides the announced characterization of materializations in terms of simulations and homomorphisms. In the following, the class of all models of $\mathcal{T}$ and $\mathcal{A}$ is denoted by $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$.

Theorem 14. *Let $\mathcal{T}$ be an $\mathcal{ALCFI}$-TBox, $\mathcal{A}$ an ABox, and let $\mathcal{I} \in \mathsf{Mod}(\mathcal{T}, \mathcal{A})$ be countable and generated. Then $\mathcal{I}$ is*

(1) an ELIQ-materialization of $\mathcal{T}$ and $\mathcal{A}$ iff it is i-sim-initial in $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$;

(2) a CQ-materialization of $\mathcal{T}$ and $\mathcal{A}$ iff it is a PEQ-materialization of $\mathcal{T}$ and $\mathcal{A}$ iff it is hom-initial in $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$;

(3) an ELQ-materialization of $\mathcal{T}$ and $\mathcal{A}$ iff it is sim-initial in $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$, provided that $\mathcal{T}$ is an $\mathcal{ALCF}$-TBox.

The 'only if' directions of all three points hold without any of the assumed conditions on $\mathcal{I}$.

Proof. We show that (1) follows from Lemma 12 and Lemma 13; (2) and (3) can be proved similarly.

We start with the direction from right to left. Assume that $\mathcal{I}$ is i-sim-initial in $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$. Since $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$, we have $\mathcal{I} \models C(a)$ whenever $\mathcal{T}, \mathcal{A} \models C(a)$ for any ELIQ $C(x)$ and $a \in \mathsf{Ind}(\mathcal{A})$. Conversely, if $\mathcal{T}, \mathcal{A} \not\models C(a)$ then there exists a model $\mathcal{J}$ of $\mathcal{T}$ and $\mathcal{A}$ such that $\mathcal{J} \not\models C(a)$. There is an i-simulation from $\mathcal{I}$ to $\mathcal{J}$. Thus, by the implication (2) $\Rightarrow$ (1) from Lemma 12, we have $\mathcal{I} \not\models C(a)$ as required.

For the direction from left to right, assume that $\mathcal{I}$ is a materialization of $\mathcal{T}$ and $\mathcal{A}$ and take a model $\mathcal{J}$ of $\mathcal{T}$ and $\mathcal{A}$. We have to construct an i-simulation from $\mathcal{I}$ to $\mathcal{J}$. It actually suffices to construct an i-simulation from $\mathcal{I}$ to the i-unfolding $\mathcal{J}^i$ of $\mathcal{J}$: by Point (3) of Lemma 13, there is a homomorphism from $\mathcal{J}^i$ to $\mathcal{J}$ and the composition of an i-simulation with a homomorphism is again an i-simulation.

To obtain an i-simulation from $\mathcal{I}$ to $\mathcal{J}^i$, we first identify a subinterpretation $\mathcal{J}'$ of $\mathcal{J}^i$ that has finite outdegree and is still a model of $\mathcal{T}$ and $\mathcal{A}$. By the implication (1) $\Rightarrow$ (2) from Lemma 12 and since $\mathcal{I}$ is a materialization, there must then be an i-simulation from $\mathcal{I}$ to $\mathcal{J}'$. Clearly this is also an i-simulation from $\mathcal{I}$ to $\mathcal{J}^i$ and we are done.

It thus remains to construct $\mathcal{J}'$, which is done by applying selective filtration to $\mathcal{J}^i$ in exactly the same way as in the proof of Lemma 11. It can be verified that the outdegree of the resulting subinterpretation $\mathcal{J}'$ of $\mathcal{J}^i$ is bounded by $|\mathcal{T}| + |\mathcal{A}|$ and, therefore, finite. By construction, $\mathcal{J}' \in \mathsf{Mod}(\mathcal{T}, \mathcal{A})$. ❏

The following example shows that the generatedness condition in Theorem 14 cannot be dropped. We leave it open whether the same is true for countability.

Example 15. *Let $\mathcal{T} = \{A \sqsubseteq \exists r.A, B \sqsubseteq A\}$ and $\mathcal{A} = \{B(a)\}$ and consider the interpretation $\mathcal{I}$ defined by*

$$\begin{aligned}
\Delta^{\mathcal{I}} &= \{a\} \cup \{0, 1, 2 \ldots\} \cup \{\ldots, -2', -1', 0', 1', 2', \ldots\} \\
A^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
B^{\mathcal{I}} &= \{a\} \\
r^{\mathcal{I}} &= \{(a, 0)\} \cup \{(n, n+1) \mid n \in \mathbb{N}\} \cup \{(n', n'+1) \mid n \in \mathbb{Z}\}.
\end{aligned}$$

Then $\mathcal{I}$ is a PEQ-materialization of $\mathcal{T}$ and $\mathcal{A}$, but it is not hom-initial (and in fact not even sim-initial) since the restriction of $\mathcal{I}$ to domain $\{a\} \cup \{0, 1, 2 \ldots\}$ is also a model of $\mathcal{T}$ and $\mathcal{A}$, but there is no homomorphism (and no simulation) from $\mathcal{I}$ to this restriction preserving $\{a\}$.

As an application of Theorem 14, we now show that materializability coincides for the query languages PEQ, CQ, and ELIQ (and that for $\mathcal{ALCF}$-TBoxes, all these also coincide with ELQ-materializability).

Theorem 16. *Let $\mathcal{T}$ be an $\mathcal{ALCFI}$-TBox. Then the following conditions are equivalent:*

(1) $\mathcal{T}$ is PEQ-materializable;

(2) $\mathcal{T}$ is CQ-materializable;

(3) $\mathcal{T}$ is ELIQ-materializable;

(4) $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$ *contains an i-sim-initial* $\mathcal{I}$*, for every ABox* $\mathcal{A}$ *that is consistent w.r.t.* $\mathcal{T}$*;*

(5) $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$ *contains a hom-initial* $\mathcal{I}$*, for every ABox* $\mathcal{A}$ *that is consistent w.r.t.* $\mathcal{T}$*.*

If $\mathcal{T}$ *is an* $\mathcal{ALCF}$*-TBox, then the above is the case iff* $\mathcal{T}$ *is ELQ-materializable iff* $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$ *contains a sim-initial* $\mathcal{I}$*, for every ABox* $\mathcal{A}$ *that is consistent w.r.t.* $\mathcal{T}$*.*

Proof. The implications $(1) \Rightarrow (2)$ and $(2) \Rightarrow (3)$ are trivial. For $(3) \Rightarrow (4)$, let $\mathcal{I}$ be an ELIQ-materialization of $\mathcal{T}$ and an ABox $\mathcal{A}$. By Lemma 11, we may assume that $\mathcal{I}$ is countable and generated. By Lemma 14, $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$ contains an i-sim-initial interpretation. For $(4) \Rightarrow (5)$, assume that $\mathcal{I} \in \mathsf{Mod}(\mathcal{T}, \mathcal{A})$ is i-sim-initial. By Points (3) and (5) of Lemma 13, the i-unfolding of $\mathcal{I}$ is hom-initial in $\mathsf{Mod}(\mathcal{T}, \mathcal{A})$ and (5) follows. $(5) \Rightarrow (1)$ follows from Theorem 14. The implications for $\mathcal{ALCF}$-TBoxes are proved similarly. ❑

Because of Theorem 16, we sometimes speak of *materializability* without reference to a query language and of *materializations* instead of PEQ-materializations.

3.2. **Materializability and** CONP**-hardness.** We show that non-materializability of a TBox $\mathcal{T}$ implies CONP-hardness of ELIQ-evaluation w.r.t. $\mathcal{T}$. To this end, we first establish that materializability is equivalent to the disjunction property, which is sometimes also called convexity and plays a central role in attaining PTIME complexity for subsumption in DLs [BBL05], and for attaining PTIME data complexity for query answering with DL TBoxes [KL07].

Let $\mathcal{T}$ be a TBox. For an ABox $\mathcal{A}$, individual names $a_0, \ldots, a_k \in \mathsf{Ind}(\mathcal{A})$, and ELIQs $C_0(x), \ldots, C_k(x)$, we write $\mathcal{T}, \mathcal{A} \models C_0(a_0) \vee \cdots \vee C_k(a_k)$ if for every model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$, $\mathcal{I} \models C_i(a_i)$ holds for some $i \leq k$. We say that $\mathcal{T}$ has the *ABox disjunction property for ELIQ (resp. ELQ)* if for all ABoxes $\mathcal{A}$, individual names $a_0, \ldots, a_k \in \mathsf{Ind}(\mathcal{A})$, and ELIQs (resp. ELQs) $C_0(x), \ldots, C_k(x)$, $\mathcal{T}, \mathcal{A} \models C_0(a_0) \vee \cdots \vee C_k(a_k)$ implies $\mathcal{T}, \mathcal{A} \models C_i(a_i)$ for some $i \leq k$.

Theorem 17. *An* $\mathcal{ALCFI}$*- (*$\mathcal{ALCF}$*-)TBox* $\mathcal{T}$ *is materializable iff it has the ABox disjunction property for ELIQs (ELQs).*

Proof. For the nontrivial "if" direction, let $\mathcal{A}$ be an ABox that is consistent w.r.t. $\mathcal{T}$ and such that there is no ELIQ-materialization of $\mathcal{T}$ and $\mathcal{A}$. Then $\mathcal{T} \cup \mathcal{A} \cup \Gamma$ is not satisfiable, where

$$\Gamma = \{\neg C(a) \mid \mathcal{T}, \mathcal{A} \not\models C(a), a \in \mathsf{Ind}(\mathcal{A}), C(x) \text{ ELIQ}\}.$$

In fact, any satisfying interpretation would be an ELIQ-materialization. By compactness, there is a finite subset Γ' of Γ such that $\mathcal{T} \cup \mathcal{A} \cup \Gamma'$ is not satisfiable, i.e. $\mathcal{T}, \mathcal{A} \models \bigvee_{\neg C(a) \in \Gamma'} C(a)$. Since $\Gamma' \subseteq \Gamma$, we have $\mathcal{T}, \mathcal{A} \not\models C(a)$ for all $\neg C(a) \in \Gamma'$. Thus, $\mathcal{T}$ lacks the ABox disjunction property. ❑

Based on Theorems 16 and 17, we now establish that materializability is a necessary condition for query evaluation to be it PTIME.

Theorem 18. *If an* $\mathcal{ALCFI}$*-TBox* $\mathcal{T}$ *(*$\mathcal{ALCF}$*-TBox* $\mathcal{T}$*) is not materializable, then ELIQ-evaluation (ELQ-evaluation) w.r.t.* $\mathcal{T}$ *is* CONP*-hard.*

Proof. The proof is by reduction of 2+2-SAT, a variant of propositional satisfiability that was first introduced by Schaerf as a tool for establishing lower bounds for the data complexity of query answering in a DL context [Sch93]. A *2+2 clause* is of the form $(p_1 \vee p_2 \vee \neg n_1 \vee \neg n_2)$, where each of p_1, p_2, n_1, n_2 is a propositional letter or a truth constant 0, 1. A *2+2 formula* is a finite conjunction of 2+2 clauses. Now, 2+2-SAT is the problem of deciding whether a given 2+2 formula is satisfiable. It is shown in [Sch93] that 2+2-SAT is NP-complete.

We first show that if an $\mathcal{ALCFI}$-TBox $\mathcal{T}$ is not materializable, then UELIQ-evaluation w.r.t. $\mathcal{T}$ is coNP-hard, where a UELIQ is a disjunction $C_0(x) \vee \cdots \vee C_k(x)$, with each $C_i(x)$ an ELIQ. We then sketch the modifications necessary to lift the result to ELIQ-evaluation w.r.t. $\mathcal{T}$.

Since $\mathcal{T}$ is not materializable, by Theorem 17 it does not have the ABox disjunction property. Thus, there is an ABox $\mathcal{A}_\vee$, individual names $a_0, \ldots, a_k \in \mathsf{Ind}(\mathcal{A})$, and ELIQs $C_0(x), \ldots, C_k(x)$, $k \geq 1$, such that $\mathcal{T}, \mathcal{A}_\vee \models C_0(a_0) \vee \cdots \vee C_k(a_k)$, but $\mathcal{T}, \mathcal{A}_\vee \not\models C_i(a_i)$ for all $i \leq k$. Assume w.l.o.g. that this sequence is minimal, i.e., $\mathcal{T}, \mathcal{A}_\vee \not\models C_0(a_0) \vee \cdots \vee C_{i-1}(a_{i-1}) \vee C_{i+1}(a_{i+1}) \vee \cdots \vee C_k(a_k)$ for all $i \leq k$. This clearly implies that for all $i \leq k$,

($*$) there is a model $\mathcal{I}_i$ of $\mathcal{T}$ and $\mathcal{A}_\vee$ with $\mathcal{I} \models C_i(a_i)$ and $\mathcal{I} \not\models C_j(a_j)$ for all $j \neq i$.

We will use $\mathcal{A}_\vee$, the individual names $a_1, \ldots, a_k$, and the ELIQs $C_0(x), \ldots, C_k(x)$ to generate truth values for variables in the input 2+2 formula.

Let $\varphi = c_0 \wedge \cdots \wedge c_n$ be a 2+2 formula in propositional letters $z_0, \ldots, z_m$, and let $c_i = p_{i,1} \vee p_{i,2} \vee \neg n_{i,1} \vee \neg n_{i,2}$ for all $i \leq n$. Our aim is to define an ABox $\mathcal{A}_\varphi$ with a distinguished individual name f and a UELIQ $q(x)$ such that φ is unsatisfiable iff $\mathcal{T}, \mathcal{A}_\varphi \models q(f)$. To start, we represent the formula φ in the ABox $\mathcal{A}_\varphi$ as follows:

- the individual name f represents the formula φ;
- the individual names $c_0, \ldots, c_n$ represent the clauses of φ;
- the assertions $c(f, c_0), \ldots, c(f, c_n)$, associate f with its clauses, where c is a role name that does not occur in $\mathcal{T}$;
- the individual names $z_0, \ldots, z_m$ represent variables, and the individual names $0, 1$ represent truth constants;
- the assertions
$$\bigcup_{i \leq n} \{p_1(c_i, p_{i,1}), p_2(c_i, p_{i,2}), n_1(c_i, n_{i,1}), n_2(c_i, n_{i,2})\}$$
  associate each clause with the variables/truth constants that occur in it, where p_1, p_2, n_1, n_2 are role names that do not occur in $\mathcal{T}$.

We further extend $\mathcal{A}_\varphi$ to enforce a truth value for each of the variables z_i. To this end, add to $\mathcal{A}_\varphi$ copies $\mathcal{A}_0, \ldots, \mathcal{A}_m$ of $\mathcal{A}_\vee$ obtained by renaming individual names such that $\mathsf{Ind}(\mathcal{A}_i) \cap \mathsf{Ind}(\mathcal{A}_j) = \emptyset$ whenever $i \neq j$. As a notational convention, let a_j^i be the name used for the individual name $a_j \in \mathsf{Ind}(\mathcal{A}_\vee)$ in $\mathcal{A}_i$ for all $i \leq m$ and $j \leq k$. Intuitively, the copy $\mathcal{A}_i$ of $\mathcal{A}$ is used to generate a truth value for the variable z_i. To actually connect each individual name z_i to the associated ABox $\mathcal{A}_i$, we use role names $r_0, \ldots, r_k$ that do not occur in $\mathcal{T}$. More specifically, we extend $\mathcal{A}_\varphi$ as follows:

- link variables z_i to the ABoxes $\mathcal{A}_i$ by adding assertions $r_j(z_i, a_j^i)$ for all $i \leq m$ and $j \leq k$; thus, truth of z_i means that the concept $\exists r_0.C_0$ is true at z_i and falsity means that one of the concepts $\exists r_j.C_j, j \leq k$, is true at z_i;
- to ensure that 0 and 1 have the expected truth values, add a copy of $C_0(x)$ viewed as an ABox with root $0'$ and a copy of $C_1(x)$ viewed as an ABox with root $1'$; add $r_0(0, 0')$ and $r_1(1, 1')$.

Consider the query $q_0(x) = C(x)$ with
$$C = \exists c.(\exists p_1.\mathsf{ff} \sqcap \exists p_2.\mathsf{ff} \sqcap \exists n_1.\mathsf{tt} \sqcap \exists n_2.\mathsf{tt})$$

which describes the existence of a clause with only false literals and thus captures falsity of φ, where tt is an abbreviation for $\exists r_0.C_0$ and ff an abbreviation for the $\mathcal{ELU}$-concept $\exists r_1.C_1 \sqcup \cdots \sqcup \exists r_k.C_k$.

It is straightforward to show that φ is unsatisfiable iff $\mathcal{T}, \mathcal{A} \models q_0(f)$. To obtain the desired UELIQ $q(x)$, it remains to take $q_0(x)$ and distribute disjunction to the outside.

We now show how to improve the result from UELIQ-evaluation to ELIQ-evaluation. Our aim is to change the encoding of falsity of a variable z_i from satisfaction of $\exists r_1.C_1 \sqcup \cdots \sqcup \exists r_k.C_k$ at z_i to satisfaction of $\exists h.(\exists r_1.C_1 \sqcap \cdots \sqcap \exists r_k.C_k)$, at z_i, where h is an additional role that does not occur in $\mathcal{T}$. We can then replace the concept ff in the query q_0 with $\exists h.(\exists r_1.C_1 \sqcap \cdots \sqcap \exists r_k.C_k)$, which gives the desired ELIQ $q(x)$.

It remains to modify $\mathcal{A}_\varphi$ to support the new encoding of falsity. The basic idea is that each z_i has k successors $b_1^i, \ldots, b_k^i$ reachable via h such that for $1 \le j \le k$,

- $\exists r_\ell.C_\ell$ is satisfied at b_j^i for all $\ell = 1, \ldots, j-1, j+1, \ldots, k$ and
- the assertion $r_j(b_j^i, a_j^i)$ is in $\mathcal{A}_\varphi$.

Thus, $\exists r_1.C_1 \sqcap \cdots \sqcap \exists r_k.C_k$ is satisfied at b_j^i iff C_j is satisfied at a_j^i, for all j with $1 \le j \le k$. In detail, the modification of $\mathcal{A}_\varphi$ is as follows:

- for $1 \le j \le k$, add to $\mathcal{A}_\varphi$ a copy of C_j viewed as an ABox, where the root individual name is d_j;
- for all $i \le m$, replace the assertions $r_j(z_i, a_j^i)$, $1 \le j \le k$, with the following:
  - $h(z_i, b_1^i), \ldots, h(z_i, b_k^i)$ for all $i \le m$;
  - $r_j(b_j^i, a_j^i), r_1(b_j^i, d_1), \ldots, r_{j-1}(b_j^i, d_{j-1})$,
    $r_{j+1}(b_j^i, d_{j+1}), \ldots, r_k(b_j^i, d_k)$ for all $i \le m$ and $1 \le j \le k$.

This finishes the modified construction. Again, it is not hard to prove correctness.

It remains to note that, when $\mathcal{T}$ is an $\mathcal{ALCF}$-TBox, then the above construction of q yields an ELQ instead of an ELIQ.   ❏

The converse of Theorem 18 fails, i.e., there are TBoxes that are materializable, but for which ELIQ-evaluation is CONP-hard. In fact, materializations of such a TBox $\mathcal{T}$ and ABox $\mathcal{A}$ are guaranteed to exist, but cannot always be computed in PTIME (unless PTIME = CONP). Technically, this follows from Theorem 33 later on which states that for every non-uniform CSP, there is a materializable $\mathcal{ALC}$-TBox for which Boolean CQ-answering has the same complexity, up to complementation of the complexity class.

3.3. **Complexity of TBoxes for Different Query Languages.** We make use of our results on materializability to show that PTIME query evaluation w.r.t. $\mathcal{ALCFI}$-TBoxes does not depend on whether we consider PEQs, CQs, or ELIQs, and the same is true for CONP-hardness, for Datalog$^{\ne}$-rewritability, and for monadic Datalog$^{\ne}$-rewritability. For $\mathcal{ALCF}$-TBoxes, we can add ELQs to the list. Theorem 19 below gives a uniform explanation for the fact that, in the traditional approach to data complexity in OBDA, the complexity of evaluating PEQs, CQs, and ELIQs has turned out to be identical for almost all DLs. It allows us to (sometimes) speak of the 'complexity of query evaluation' without reference to a concrete query language.

Theorem 19. *For all $\mathcal{ALCFI}$-TBoxes $\mathcal{T}$,*

(1) PEQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME iff CQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME iff ELIQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME;

(2) $\mathcal{T}$ is (monadic) Datalog$^{\ne}$-rewritable for PEQ iff it is Datalog$^{\ne}$-rewritable for CQ iff it is (monadic) Datalog$^{\ne}$-rewritable for ELIQ (unless PTIME = CONP);

(3) PEQ-evaluation w.r.t. $\mathcal{T}$ is CONP-hard iff CQ-evaluation w.r.t. $\mathcal{T}$ is CONP-hard iff ELIQ-evaluation w.r.t. $\mathcal{T}$ is CONP-hard.

If $\mathcal{T}$ is an $\mathcal{ALCF}$-TBox, then ELIQ can be replaced by ELQ in (1), (2), and (3). If $\mathcal{T}$ is an $\mathcal{ALCI}$-TBox, then Datalog$^{\neq}$-rewritability can be replaced by Datalog-rewritability in (2).

Proof. We start with Points (1) and (2), for which the "only if" directions are trivial. For the converse directions, we may assume by Theorem 18 that the TBox $\mathcal{T}$ is materializable. The implications from CQ to PEQ in Points (1) and (2) follow immediately from this assumption: one can first transform a given PEQ $q(\vec{x})$ into an equivalent disjunction of CQs $\bigvee_{i \in I} q_i(\vec{x})$. CQ-materializability of $\mathcal{T}$ implies that, for any ABox $\mathcal{A}$ and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A})$, $\mathcal{T}, \mathcal{A} \models q(\vec{a})$ iff there exists $i \in I$ such that $\mathcal{T}, \mathcal{A} \models q_i(\vec{a})$. Thus if CQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME, evaluation of $(\mathcal{T}, q(\vec{x}))$ is in PTIME. The same holds for (monadic) Datalog$^{\neq}$-rewritability because the class of Datalog$^{\neq}$-queries is closed under finite union.

We now consider the implications from ELIQ to CQ (and from ELQ to CQ if $\mathcal{T}$ is a $\mathcal{ALCF}$-TBox) in Points (1) and (2). The following claim is the main step of the proof. It states that for any CQ $q(\vec{x})$, we can reduce the evaluation of $q(\vec{x})$ w.r.t. $\mathcal{T}$ on an ABox $\mathcal{A}$ to evaluating quantifier-free CQs and ELIQs $C(x)$ w.r.t. $\mathcal{T}$ (ELQs if $\mathcal{T}$ is an $\mathcal{ALCF}$-TBox), both on $\mathcal{A}$.

Claim 1. For any materializable TBox $\mathcal{T}$ and CQ $q(\vec{x})$ with $\vec{x} = x_1 \cdots x_n$, one can construct a finite set $\mathcal{Q}$ of pairs $(\varphi(\vec{x}, \vec{y}), \mathcal{C})$, where

- $\varphi(\vec{x}, \vec{y})$ is a (possibly empty) conjunction of atoms of the form $x = y$ or $r(x, y)$, where r is a role name in $q(\vec{x})$ and
- $\mathcal{C}$ is a finite set of ELIQs

such that the following conditions are equivalent for any ABox $\mathcal{A}$ and $\vec{a} = a_1 \cdots a_n$ from $\mathsf{Ind}(\mathcal{A})$:

  (i) $\mathcal{T}, \mathcal{A} \models q(\vec{a})$;
  (ii) there exists $(\varphi(\vec{x}, \vec{y}), \mathcal{C}) \in \mathcal{Q}$ and an assignment π in $\mathsf{Ind}(\mathcal{A})$ with $\pi(x_i) = a_i$ for $1 \leq i \leq n$, $\mathcal{A} \models_\pi \varphi(\vec{x}, \vec{y})$, and $\mathcal{T}, \mathcal{A} \models C(\pi(x))$ for all $C(x) \in \mathcal{C}$.

If $\mathcal{T}$ is an $\mathcal{ALCF}$-TBox, then one can choose ELQs instead of ELIQs in each $\mathcal{C}$ in $\mathcal{Q}$.

Before we prove Claim 1, we show how the desired results follow from it. Let a CQ $q(\vec{x})$ be given and let $\mathcal{Q}$ be the set of pairs from Claim 1.

- Assume that ELIQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME. Then $\mathcal{T}, \mathcal{A} \models q(\vec{a})$ can be decided in polynomial time since there are only polynomially many assignments π and for any such π, $\mathcal{A} \models_\pi \varphi(\vec{x}, \vec{y})$ can be checked in polynomial time (using a naive algorithm) and $\mathcal{T}, \mathcal{A} \models C(\pi(x))$ can be checked in polynomial time for each ELIQ $C(x) \in \mathcal{C}$.
- Assume that $\mathcal{T}$ is (monadic) Datalog$^{\neq}$-rewritable for ELIQ. Let $p = (\varphi(\vec{x}, \vec{y}), \mathcal{C}) \in \mathcal{Q}$. For each $C(x) \in \mathcal{C}$, choose a (monadic) Datalog$^{\neq}$-rewriting $\Pi_C(x)$ of $(\mathcal{T}, C(x))$, assume w.l.o.g that none of the chosen programs share any IDB relations, and that the goal relation of $\Pi_C(x)$ is goal_C. Let Π_p be the (monadic) Datalog$^{\neq}$ program that consists of the rules of all the chosen programs, plus the following rule:

$$\mathsf{goal}(\vec{x}) \quad \leftarrow \quad \varphi(\vec{x}, \vec{y}) \wedge \bigwedge_{C(x) \in \mathcal{C}} \mathsf{goal}_C(x).$$

  The desired (monadic) Datalog$^{\neq}$-rewriting of $(\mathcal{T}, q(\vec{x}))$ is obtained by taking the union of all the constructed (monadic) Datalog$^{\neq}$ queries.

The implications from ELQs to CQs for $\mathcal{ALCF}$-TBoxes in Points (1) and (2) follow in the same way since, then, each $\mathcal{C}$ in $\mathcal{Q}$ consists of ELQs only.

For the proof of Claim 1, we first require a technical observation that allows us to deal with subqueries that are not connected to an answer variable in the CQ $q(\vec{x})$. To illustrate, consider the query $q_0 = \exists x\, B(x)$. To prove Claim 1 for q_0, we have to find a set $\mathcal{Q}$ of pairs $(\varphi(\vec{y}), \mathcal{C})$ satisfying

Conditions (i) and (ii). Clearly, in this case the components $\varphi(\vec{y})$ will be empty and so we have to construct a finite set $\mathcal{C}$ of ELIQs such that for any ABox $\mathcal{A}$, $\mathcal{T}, \mathcal{A} \models \exists x\, B(x)$ iff there exists an ELIQ $C(x) \in \mathcal{C}$ and an assignment π in $\mathsf{Ind}(\mathcal{A})$ such that $\mathcal{T}, \mathcal{A} \models C(\pi(x))$. An *infinite* set $\mathcal{C}$ with this property is given by the set of all ELIQs $\exists \vec{r}.B(x)$, where $\vec{r}$ is a sequence $r_1 \cdots r_n$ of roles r_i in $\mathcal{T}$ and $\exists \vec{r}.B$ stands for $\exists r_1 \cdots \exists r_n.B$—this follows immediately from the assumption that $\mathcal{T}$ is materializable and that, by Lemma 11, for any ABox $\mathcal{A}$ that is consistent w.r.t. $\mathcal{T}$, there exists a generated CQ-initial model of $\mathcal{T}$ and ABox $\mathcal{A}$. The following result states that it is sufficient to include in $\mathcal{C}$ the set of all $\exists \vec{r}.B(x)$ with $\vec{r}$ of length bounded by $n_0 := 2^{(2(|\mathcal{T}|+|C|)} \cdot 2|\mathcal{T}| + 1$.

Claim 2. Let C be an $\mathcal{ELI}$-concept and assume that $\mathcal{T}, \mathcal{A} \models \exists x\, C(x)$. If $\mathcal{T}$ is materializable, then there exists a sequence of roles $\vec{r} = r_1 \cdots r_n$ with r_i in $\mathcal{T}$ and of length $n \leq n_0$ and an $a \in \mathsf{Ind}(\mathcal{A})$ such that $\mathcal{T}, \mathcal{A} \models \exists \vec{r}.C(a)$. If C is an $\mathcal{EL}$-concept and $\mathcal{T}$ an $\mathcal{ALCF}$-TBox, then the sequence $\vec{r}$ consists of role names in $\mathcal{T}$.

Proof of Claim 2. Let $\mathcal{I}$ be a CQ-materialization of $\mathcal{T}$ and $\mathcal{A}$. By Points (3) and (5) of Lemma 13, we may assume that $\mathcal{I}$ is i-unfolded. From $\mathcal{T}, \mathcal{A} \models \exists x\, C(x)$, we obtain $C^{\mathcal{I}} \neq \emptyset$. Let n be minimal such that there are $a \in \mathsf{Ind}(\mathcal{A})$ and $d \in C^{\mathcal{I}}$ with $n = \mathsf{dist}_{\mathcal{I}}(d, a)$ where $\mathsf{dist}_{\mathcal{I}}(d, a)$ denotes the length of the shortest path from d to a in the undirected graph $G_{\mathcal{I}}$. If $n \leq n_0$, we are done. Otherwise fix an $a \in \mathsf{Ind}(\mathcal{A})$ and denote by M the set all $e \in C^{\mathcal{I}}$ with $n = \mathsf{dist}_{\mathcal{I}}(e, a)$. Let $d_0, \ldots, d_n$ with $a = d_0$, $d_n = d$, and $(d_i, d_{i+1}) \in r_{i+1}^{\mathcal{I}}$ for $i < n$ be the shortest path from a to d. Observe that $\mathcal{T}, \mathcal{A} \models \exists \vec{r}.C(a)$ for $\vec{r} = r_0 \cdots r_{n-1}$ since $\mathcal{I}$ is a materialization of $\mathcal{T}$ and $\mathcal{A}$. We now employ a pumping argument to show that this leads to a contradiction. Let $\mathsf{cl}(\mathcal{T}, C)$ denote the closure under single negation of the set of subconcepts of concepts in $\mathcal{T}$ and C. Set

$$t_{\mathcal{T},C}^{\mathcal{I}}(e) = \{D \in \mathsf{cl}(\mathcal{T}, C) \mid e \in D^{\mathcal{I}}\}.$$

As $n > n_0$, then there exist d_i and d_{i+j} with $j > 1$ and $i + j < n$ such that

$$t_{\mathcal{T},C}^{\mathcal{I}}(d_i) = t_{\mathcal{T},C}^{\mathcal{I}}(d_{i+j}), \quad t_{\mathcal{T},C}^{\mathcal{I}}(d_{i+1}) = t_{\mathcal{T},C}^{\mathcal{I}}(d_{i+j+1}), \quad r_{i+1} = r_{i+j+1}$$

Now replace in $\mathcal{I}$ the interpretation induced by the subtree rooted at d_{i+j+1} by the interpretation induced by the subtree rooted at d_{i+1}. We do the same construction for all elements of M and denote the resulting interpretation by $\mathcal{J}$. One can easily show by induction that $\mathcal{J}$ is still a model of $\mathcal{T}$ and $\mathcal{A}$, but now $\mathcal{J} \not\models \exists \vec{r}.C(a)$ and so $\mathcal{T}, \mathcal{A} \not\models \exists \vec{r}.C(a)$. This contradiction finishes the proof of Claim 2. For $\mathcal{EL}$-concepts and $\mathcal{ALCF}$-TBoxes, Claim 2 can be proved similarly by using an unfolded (rather than i-unfolded) materialization, which exists by Points (4) and (6) of Lemma 13.

Proof of Claim 1. Assume that $\mathcal{T}$ and $q(\vec{x}) = \exists \vec{x}\, \psi(\vec{x}, \vec{y})$ are given. We have to construct a set $\mathcal{Q}$ such that Conditions (i) and (ii) are satisfied. Let $\mathcal{A}$ be an ABox with $\mathcal{T}, \mathcal{A} \models q(\vec{a})$, $\mathcal{I}$ a materialization of $\mathcal{T}$ and $\mathcal{A}$ that is i-unfolded, and π an assignment in $\mathcal{I}$ such that $\mathcal{I} \models_\pi \psi(\vec{x}, \vec{y})$. We define a corresponding pair $p = (\varphi(\vec{x}, \vec{z}), \mathcal{C})$ to be included in $\mathcal{Q}$ (and these are the only pairs in $\mathcal{Q}$).

For identifying $\varphi(\vec{x}, \vec{z})$, set $x \sim y$ if $\pi(x) = \pi(y)$ and denote by $[x]$ the equivalence class of x w.r.t. "$\sim$". Let φ_0 be the set of all atoms $r([x], [y])$ such that $\pi(x), \pi(y) \in \mathsf{Ind}(\mathcal{A})$ and there are $x' \in [x]$ and $y' \in [y]$ with $r(x', y')$ in ψ. We obtain $\varphi(\vec{x}, \vec{y})$ by selecting an answer variable $y \in [x]$ for every $[x]$ that contains such a variable and then replacing $[x]$ by y in φ_0, adding $x_i = x_j$ to $\varphi(\vec{x}, \vec{z})$ for any two (selected) answer variables x_i, x_j with $x_i \sim x_j$, and by regarding the remaining equivalences classes $[y]$ that do not contain answer variables as variables in $\vec{z}$.

We now identify $\mathcal{C}$. Assume w.l.o.g. that $\mathcal{I}$ uses the naming scheme of i-unravelings. Let $a \in \mathsf{Ind}(\mathcal{A})$. By $\mathcal{I}_a$, we denote the subinterpretation of $\mathcal{I}$ induced by the set of all elements

$ar_1d_1 \cdots r_nd_n \in \Delta^{\mathcal{I}}$. Let M be a maximal connected component of $\Delta^{\mathcal{I}_a} \cap \{\pi(y) \mid y \in \mathsf{var}(\psi)\}$. We associate with M an ELIQ to be included in $\mathcal{C}$ (and these are the only ELIQs in $\mathcal{C}$).

The conjunctive query φ_M consists of all atoms $r([x], [y])$ such that $\pi(x), \pi(y) \in M$ and there are $x' \in [x], y' \in [y]$ with $r(x', y')$ in ψ and all atoms $A([x])$ such that $\pi(x) \in M$ and there is $x' \in [x]$ with $A(x')$ in ψ. We again assume that equivalence classes $[x]$ that contain an answer variable (there is at most one such class in φ_M) are replaced with an answer variable from $[x]$ and regard the remaining equivalences classes as variables. Note that φ_M is tree-shaped since $\mathcal{I}_a$ is. We can thus pick a variable x_0 with $\pi(x_0) \in M$ such that $\mathsf{dist}_{\mathcal{I}}(a, \pi(x_0))$ is minimal. Let x be $[x_0]$ if $[x_0]$ contains no answer variable and, otherise, let x be the answer variable that $[x_0]$ has been replaced with. Let $[y_1], \ldots, [y_m]$ be the variables in φ_M that are distinct from x and consider the ELIQ $\exists[y_1] \cdots \exists[y_m]\varphi_M(x, [y_1], \ldots, [y_m])$, which we write as $C_M(x)$ where C_M is an appropriate $\mathcal{ELI}$-concept. We now distinguish the following cases:

- $\pi(x) = a$. In this case, we include $C_M(x)$ in $\mathcal{C}$;
- otherwise, we still know that $\mathcal{T}, \mathcal{A} \models \exists x\, C(x)$. Thus, by Claim 2 there is a sequence of roles $\vec{r} = r_1 \cdots r_n$ with r_i in $\mathcal{T}$ and $n \leq n_0$ such that $\mathcal{T}, \mathcal{A} \models \exists \vec{r}.C_M(a)$ for some $a \in \mathsf{Ind}(\mathcal{A})$. In this case, we include $\exists\vec{r}.C_M(y)$ in $\mathcal{C}$ for some fresh variable y.

This finishes the construction of $\mathcal{C}$ and thus of $\mathcal{Q}$. Clearly, $\mathcal{Q}$ is finite. The stated properties of $\mathcal{Q}$ follow directly from its construction. For $\mathcal{ALCF}$-TBoxes and ELQs, Claim 1 can be proved similarly using an unfolded materialization (instead of an i-unfolded one) and the observation that in this case all $C_M([x])$ and $\exists\vec{r}.C_M(y)$ are ELQs ($\vec{r}$ uses role names only by Claim 2).

We now turn our attention to Point (3). Here, the "if" directions are trivial and we prove the "only if" part. It suffices to show that if PEQ-evaluation w.r.t. a TBox $\mathcal{T}$ is CONP-hard, then so is ELIQ-evaluation. We start with showing the slightly simpler result that CONP-hardness of CQ-evaluation w.r.t. $\mathcal{T}$ implies CONP-hardness of UELIQ-evaluation, and then sketch the modifications required to strengthen the proof to attain the original statement.

Thus assume that evaluating the CQ $q(\vec{x})$ w.r.t. $\mathcal{T}$ is CONP-hard. We shall exhibit an UELIQ $q'(x)$ such that for every ABox $\mathcal{A}$ and all $\vec{a} \in \mathsf{Ind}(\mathcal{A})$, one can produce in polynomial time an ABox $\mathcal{A}'$ with a distinguished individual name a_0 such that $\mathcal{T}, \mathcal{A} \models q(\vec{a})$ iff $\mathcal{T}, \mathcal{A}' \models q'(a_0)$. Instead of constructing $q'(\vec{x})$ right away, we will start with describing the translation of $\mathcal{A}$ to $\mathcal{A}'$. Afterwards, it will be clear how to construct $q'(\vec{x})$.

Thus, let $\mathcal{A}$ be an ABox and $\vec{a}$ from $\mathsf{Ind}(\mathcal{A})$. The construction of $\mathcal{A}'$ builds on Claim 1 above. Let $\mathcal{Q}$ be the set of pairs from that claim and reserve a fresh individual name a_0. To obtain the desired ABox $\mathcal{A}'$, we extend $\mathcal{A}$ for every pair $p = (\varphi(\vec{x}, \vec{y}), \mathcal{C})$ in $\mathcal{Q}$. Let $\mathcal{C} = C_{p,1}(x_1), \ldots, C_{p,k_p}(x_{k_p})$. Then

- introduce a fresh individual name a_p and fresh role names $r_p, r, r_{p,1}, \ldots, r_{p,k_p}$;
- add the assertion $r_p(a_0, a_p)$;
- for every assignment π in $\mathsf{Ind}(\mathcal{A})$ with $\mathcal{A} \models_\pi \varphi(\vec{x}, \vec{y})$ and $\pi(\vec{x}) = \vec{a}$, introduce
  - a fresh individual name $a_{p,\pi}$ and the assertion $r(a_p, a_{p,\pi})$;
  - the assertion $r_{p,i}(a_p, \pi(x_i))$ for $1 \leq i \leq k_p$.

From Claim 1, it is immediate that $\mathcal{T}, \mathcal{A} \models q(\vec{a})$ iff $\mathcal{T}, \mathcal{A}' \models q'(x)$ where $q'(x)$ is the UELIQ $\bigsqcup_{p \in Q} q'(x)$ with $C_p = \exists r_p.\exists r. \bigsqcap_{1 \leq i \leq k_p} \exists r_{p,i}.C_{p,i}$. Thus, evaluating $q'(x)$ w.r.t. $\mathcal{T}$ is CONP-hard, as required. It remains to modify the reduction by replacing CQs with PEQs and UELIQs with ELIQs. The former is straightforward: every PEQ is equivalent to a finite disjunction of CQs, and thus we can construct $\mathcal{A}'$ and $q'(x)$ in essentially the same way as before; instead of considering all pairs from $\mathcal{Q}$ for a single CQ, we now use the union of all sets $\mathcal{Q}$ for the finitely many CQs in

question. Finally, we can replace UELIQs with ELIQs by using the same construction as in the proof of Theorem 18: after adding some straightforward auxiliary structure to $\mathcal{A}'$, one can replace a disjunction of ELIQs by (essentially) their conjunction, which is again an ELIQ. Details are left to the reader. ❏

We remark that Theorem 19 can be extended to also cover rewritability into first-order (FO) queries, and that the proof is almost identical to that for Datalog$^{\neq}$-rewritability.

4. Unraveling Tolerance

We develop a condition on TBoxes, called unraveling tolerance, that is sufficient for the TBox to be monadic Datalog$^{\neq}$-rewritable for PEQ, and thus also sufficient for PEQ-evaluation w.r.t. the TBox being in PTIME. Unraveling tolerance strictly generalizes syntactic 'Horn conditions' such as the ones used to define the DL Horn-$\mathcal{SHIQ}$, which was designed as a (syntactically) maximal DL with PTIME query evaluation [HMS07, EGOS08].

Unraveling tolerance is based on an unraveling operation on ABoxes, in the same spirit as the unfolding of an interpretation into a tree interpretation we discussed above. Formally, the *unraveling* $\mathcal{A}^u$ of an ABox $\mathcal{A}$ is the following (possibly infinite) ABox:

- $\mathsf{Ind}(\mathcal{A}^u)$ is the set of sequences $b_0 r_0 b_1 \cdots r_{n-1} b_n$, $n \geq 0$, with $b_0, \ldots, b_n \in \mathsf{Ind}(\mathcal{A})$ and $r_0, \ldots, r_{n-1} \in \mathsf{N_R} \cup \mathsf{N_R^-}$ such that for all $i < n$, we have $r_i(b_i, b_{i+1}) \in \mathcal{A}$ and $(b_{i-1}, r_{i-1}^-) \neq (b_{i+1}, r_i)$ when $i > 0$;
- for each $C(b) \in \mathcal{A}$ and $\alpha = b_0 r_0 \cdots r_{n-1} b_n \in \mathsf{Ind}(\mathcal{A}^u)$ with $b_n = b$, $C(\alpha) \in \mathcal{A}^u$;
- for each $\alpha = b_0 r_0 \cdots r_{n-1} b_n \in \mathsf{Ind}(\mathcal{A}^u)$ with $n > 0$, $r_{n-1}(b_0 r_0 \cdots r_{n-2} b_{n-1}, \alpha) \in \mathcal{A}^u$.

For all $\alpha = b_0 \cdots b_n \in \mathsf{Ind}(\mathcal{A}^u)$, we write $\mathsf{tail}(\alpha)$ to denote b_n. Note that the condition $(b_{i-1}, r_{i-1}^-) \neq (b_{i+1}, r_i)$ is needed to ensure that functional roles can still be interpreted in a functional way after unraveling.

Definition 20 (Unraveling Tolerance). A TBox $\mathcal{T}$ is *unraveling tolerant* if for all ABoxes $\mathcal{A}$ and ELIQs q, we have that $\mathcal{T}, \mathcal{A} \models q$ implies $(\mathcal{T}, \mathcal{A}^u) \models q$.

It is not hard to prove that the converse direction '$\mathcal{T}, \mathcal{A}^u \models q$ implies $\mathcal{T}, \mathcal{A} \models q$' is true for *all* $\mathcal{ALCFI}$-TBoxes. Note that it is pointless to define unraveling tolerance for queries that are not necessarily tree shaped, such as CQs.

Example 21.

(1) The $\mathcal{ALC}$-TBox $\mathcal{T}_1 = \{A \sqsubseteq \forall r.B\}$ is unraveling tolerant. This can be proved by showing that (i) for any (finite or infinite) ABox $\mathcal{A}$, the interpretation $\mathcal{I}_\mathcal{A}^+$ that is obtained from $\mathcal{A}$ by extending $B^{\mathcal{I}_\mathcal{A}^+}$ with all $a \in \mathsf{Ind}(\mathcal{A})$ that satisfy $\exists r^-.A$ in $\mathcal{A}$ (when viewed as an interpretation) is an ELIQ-materialization of $\mathcal{T}_1$ and $\mathcal{A}$; and (ii) $\mathcal{I}_\mathcal{A}^+ \models C(a)$ iff $\mathcal{I}_{\mathcal{A}^u}^+ \models C(a)$ for all ELIQs $C(x)$ and $a \in \mathsf{Ind}(\mathcal{A})$. The proof of (ii) is based on a straightforward induction on the structure of the $\mathcal{ELI}$-concept C. As illustrated by the ABox $\mathcal{A} = \{r(a,b), A(a)\}$ and the fact that $\mathcal{A}^u, \mathcal{T} \models B(b)$, the use of inverse roles in the definition of $\mathcal{A}^u$ is crucial here despite the fact that $\mathcal{T}_1$ does not use inverse roles.

(2) A simple example for an $\mathcal{ALC}$-TBox that is not unraveling tolerant is

$$\mathcal{T}_2 = \{A \sqcap \exists r.A \sqsubseteq B, \neg A \sqcap \exists r.\neg A \sqsubseteq B\}.$$

For $\mathcal{A} = \{r(a,a)\}$, it is easy to see that we have $\mathcal{T}_2, \mathcal{A} \models B(a)$ (use a case distinction on the truth value of A at a), but $\mathcal{T}_2, \mathcal{A}^u \not\models B(a)$.

Before we show that unraveling tolerance indeed implies PTIME query evaluation, we first demonstrate the generality of this property by relating it to Horn-$\mathcal{ALCFI}$, the $\mathcal{ALCFI}$-fragment of Horn-$\mathcal{SHIQ}$. Different versions of Horn-$\mathcal{SHIQ}$ have been proposed in the literature, giving rise to different versions of Horn-$\mathcal{ALCFI}$ [HMS07, KRH07, EGOS08, Kaz09]. As the original definition from [HMS07] based on polarity is rather technical, we prefer to work with the following equivalent and less cumbersome definition. A *Horn-$\mathcal{ALCFI}$-TBox* $\mathcal{T}$ is a finite set of concept inclusions $L \sqsubseteq R$ and functionality assertions where L and R are built according to the following syntax rules:

$$R, R' ::= \top \mid \bot \mid A \mid \neg A \mid R \sqcap R' \mid L \to R \mid \exists r.R \mid \forall r.R$$
$$L, L' ::= \top \mid \bot \mid A \mid L \sqcap L' \mid L \sqcup L' \mid \exists r.L$$

with r ranging over $\mathsf{N_R} \cup \mathsf{N_R^-}$ and $L \to R$ abbreviating $\neg L \sqcup R$. Whenever convenient, we may assume w.l.o.g. that $\mathcal{T}$ contains only a single concept inclusion $\top \sqsubseteq C_\mathcal{T}$ where $C_\mathcal{T}$ is built according to the topmost rule above.

By applying some simple transformations, it is not hard to show that every Horn-$\mathcal{ALCFI}$-TBox according to the original polarity-based definition is equivalent to a Horn-$\mathcal{ALCFI}$-TBox of the form introduced here. Although not important in our context, we note that even a polynomial time transformation is possible.

Theorem 22.
Every Horn-$\mathcal{ALCFI}$-TBox is unraveling tolerant.

Proof. As a preliminary, we give a characterization of the entailment of ELIQs in the presence of Horn-$\mathcal{ALCFI}$-TBoxes which is in the spirit of the chase procedure as used in database theory [FKMP05, CGK13] and of consequence-driven algorithms as used for reasoning in Horn description logics such as $\mathcal{EL}^{++}$ and Horn-$\mathcal{SHIQ}$ [BBL05, Kaz09, Krö10b].

We use *extended ABoxes*, i.e., finite sets of assertions $C(a)$ and $r(a, b)$ with C a *potentially compound* concept. An $\mathcal{ELIU}_\bot$-*concept* is a concept that is formed according to the second syntax rule in the definition of Horn-$\mathcal{ALCFI}$. For an extended ABox $\mathcal{A}'$ and an assertion $C(a)$, C an $\mathcal{ELIU}_\bot$-concept, we write $\mathcal{A}' \vdash C(a)$ if $C(a)$ *has a syntactic match* in $\mathcal{A}'$, formally:

- $\mathcal{A}' \vdash \top(a)$ is unconditionally true;
- $\mathcal{A}' \vdash \bot(a)$ if $\bot(b) \in \mathcal{A}'$ for some $b \in \mathsf{Ind}(\mathcal{A})$;
- $\mathcal{A}' \vdash A(a)$ if $A(a) \in \mathcal{A}'$;
- $\mathcal{A}' \vdash C \sqcap D(a)$ if $\mathcal{A}' \vdash C(a)$ and $\mathcal{A}' \vdash D(a)$;
- $\mathcal{A}' \vdash C \sqcup D(a)$ if $\mathcal{A}' \vdash C(a)$ or $\mathcal{A}' \vdash D(a)$;
- $\mathcal{A}' \vdash \exists r.C(a)$ if there is an $r(a, b) \in \mathcal{A}'$ such that $\mathcal{A}' \vdash C(b)$.

Now for the announced characterization. Let $\mathcal{T} = \{\top \sqsubseteq C_\mathcal{T}\}$ be a Horn-$\mathcal{ALCFI}$-TBox and $\mathcal{A}$ a potentially infinite ABox (so that the characterization also applies to unravelings of ABoxes). We produce a sequence of extended ABoxes $\mathcal{A}_0, \mathcal{A}_1, \ldots$, starting with $\mathcal{A}_0 = \mathcal{A}$. In what follows, we use additional individual names of the form $ar_1C_1 \cdots r_kC_k$ with $a \in \mathsf{Ind}(\mathcal{A}_0)$, $r_1, \ldots, r_k$ roles that occur in $\mathcal{T}$, and $C_1, \ldots, C_k$ subconcepts of concepts in $\mathcal{T}$. We assume that $\mathsf{N_I}$ contains such names as needed and use the symbols $a, b, \ldots$ also to refer to individual names of this compound form. Each extended ABox $\mathcal{A}_{i+1}$ is obtained from $\mathcal{A}_i$ by applying the following rules in a fair way:

R1 if $a \in \mathsf{Ind}(\mathcal{A}_i)$, then add $C_\mathcal{T}(a)$.
R2 if $C \sqcap D(a) \in \mathcal{A}_i$, then add $C(a)$ and $D(a)$;
R3 if $C \to D(a) \in \mathcal{A}_i$ and $\mathcal{A}_i \vdash C(a)$, then add $D(a)$;
R4 if $\exists r.C(a) \in \mathcal{A}_i$ and $\mathsf{func}(r) \notin \mathcal{T}$, then add $r(a, arC)$ and $C(arC)$;
R5 if $\exists r.C(a) \in \mathcal{A}_i$, $\mathsf{func}(r) \in \mathcal{T}$, and $r(a, b) \in \mathcal{A}_i$, then add $C(b)$;

R6 if $\exists r.C(a) \in \mathcal{A}_i$, $\mathsf{func}(r) \in \mathcal{T}$, and there is no $r(a,b) \in \mathcal{A}_i$, then add $r(a, arC)$ and $C(arC)$;

R7 if $\forall r.C(a) \in \mathcal{A}_i$ and $r(a,b) \in \mathcal{A}_i$, then add $C(b)$.

Let $\mathcal{A}_c = \bigcup_{i \geq 0} \mathcal{A}_i$ be the *completion* of the original ABox $\mathcal{A}$.[2] Note that $\mathcal{A}_c$ may be infinite even if $\mathcal{A}$ is finite, and that none of the above rules is applicable in $\mathcal{A}_c$. We write '$\mathcal{A}_c \vdash \bot$' instead of '$\mathcal{A}_c \vdash \bot(a)$ for some $a \in \mathsf{N_C}$'. If $\mathcal{A} \nvdash \bot$, then $\mathcal{A}_c$ corresponds to an interpretation $\mathcal{I}_c$ in the standard way, i.e.,

$$
\begin{aligned}
\Delta^{\mathcal{I}_c} &= \mathsf{Ind}(\mathcal{A}_c) \\
A^{\mathcal{I}_c} &= \{a \mid A(a) \in \mathcal{A}_c\} && \text{for all } A \in \mathsf{N_C} \\
r^{\mathcal{I}_c} &= \{r(a,b) \mid r(a,b) \in \mathcal{A}_c\} && \text{for all } r \in \mathsf{N_R}
\end{aligned}
$$

where in $\mathcal{I}_c$ we assume that only the individual names in $\mathsf{Ind}(\mathcal{A})$ are elements of $\mathsf{N_I}$.

Claim 1. If $\mathcal{A}_c \nvdash \bot$, then $\mathcal{I}_c$ is a PEQ-materialization of $\mathcal{T}$ and $\mathcal{A}$.

To prove Claim 1 it suffices to show that there is a homomorphism h preserving $\mathsf{Ind}(\mathcal{A})$ from $\mathcal{I}_c$ into every model $\mathcal{J}$ of $\mathcal{T}$ and $\mathcal{A}$ and that $\mathcal{I}_c$ is a model of $\mathcal{T}$ and $\mathcal{A}$. The latter is immediate by construction of $\mathcal{A}_c$. Regarding the former, the desired homomorphism h can can be constructed inductively starting with the ABox $\mathcal{A}_0$ and then extending to $\mathcal{A}_1, \mathcal{A}_2, \ldots$. Using Claim 1 and the easily proved fact that $\mathcal{A}_c \nvdash \bot$ iff $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$ one can now show the following.

Claim 2. $\mathcal{T}, \mathcal{A} \models C(a)$ iff $\mathcal{A}_c \vdash C(a)$ or $\mathcal{A}_c \vdash \bot$, for all ELIQs $C(x)$ and $a \in \mathsf{Ind}(\mathcal{A})$.

We now turn to the actual proof of Theorem 22. Consider the application of the above completion construction to both the original ABox $\mathcal{A}$ and its unraveling $\mathcal{A}^u$. Recall that individual names in $\mathcal{A}^u$ are of the form $a_0 r_0 a_1 \cdots r_{n-1} a_n$. Consequently, individual names in $\mathcal{A}_c^u$ take the form $a_0 r_0 a_1 \cdots r_{n-1} a_n s_1 C_1 \cdots s_k C_k$. For $a \in \mathsf{Ind}(\mathcal{A}_c)$ and $\alpha \in \mathsf{Ind}(\mathcal{A}_c^u)$, we write $a \sim \alpha$ if a and α are of the form $a_n s_1 C_1 \cdots s_k C_k$ and $a_0 r_0 a_1 \cdots r_{n-1} a_n s_1 C_1 \cdots s_k C_k$, respectively, with $k \geq 0$. Note that, in particular, $a \sim a$ for all $a \in \mathsf{Ind}(\mathcal{A})$. The following claim can be shown by induction on i.

Claim 3. For all $a \in \mathsf{Ind}(\mathcal{A}_i)$ and $\alpha \in \mathsf{Ind}(\mathcal{A}_i^u)$ with $a \sim \alpha$, we have

(1) $\mathcal{A}_i \vdash C(a)$ iff $\mathcal{A}_i^u \vdash C(\alpha)$ for all $\mathcal{ELI}$-concepts C;

(2) $C(a) \in \mathcal{A}_i$ iff $C(\alpha) \in \mathcal{A}_i^u$ for all subconcepts C of concepts in $\mathcal{T}$.

Now, unraveling tolerance of $\mathcal{T}$ follows from Claims 2 and 3. ❏

Theorem 22 shows that unraveling tolerance and Horn logic are closely related. Yet, the next example demonstrates that there are unraveling tolerant $\mathcal{ALCFI}$-TBoxes that are not equivalent to any Horn sentence of FO. Since any Horn-$\mathcal{ALCFI}$-TBox is equivalent to such a sentence, it follows that unraveling tolerant $\mathcal{ALCFI}$-TBoxes strictly generalize Horn-$\mathcal{ALCFI}$-TBoxes. This increased generality will pay off in Section 5 when we establish a dichotomy result for TBoxes of depth one.

Example 23. *Take the $\mathcal{ALC}$-TBox*

$$\mathcal{T} = \{\exists r.(A \sqcap \neg B_1 \sqcap \neg B_2) \sqsubseteq \exists r.(\neg A \sqcap \neg B_1 \sqcap \neg B_2)\}.$$

One can show as in Example 21 (1) that $\mathcal{T}$ is unraveling tolerant; here, the materialization is actually $\mathcal{A}$ itself rather than some extension thereof, i.e., as far as ELIQ (and even PEQ) evaluation is concerned, $\mathcal{T}$ cannot be distinguished from the empty TBox.

It is well-known that FO Horn sentences are preserved under direct products of interpretations [CK90]. *To show that $\mathcal{T}$ is not equivalent to any such sentence, it thus suffices to show that $\mathcal{T}$ is not*

---

[2] Order of rule application has an impact on the shape of $\mathcal{A}_c$, but is irrelevant for the remainder of the proof.

preserved under direct products. This is simple: let $\mathcal{I}_1$ and $\mathcal{I}_2$ consist of a single r-edge between elements d and e, and let $e \in (A \sqcap B_1 \sqcap \neg B_2)^{\mathcal{I}_1}$ and $e \in (A \sqcap \neg B_1 \sqcap B_2)^{\mathcal{I}_2}$; then the direct product $\mathcal{I}$ of $\mathcal{I}_1$ and $\mathcal{I}_2$ still has the r-edge between (d,d) and (e,e) and satisfies $(e,e) \in (A \sqcap \neg B_1 \sqcap \neg B_2)^{\mathcal{I}}$, thus is not a model of $\mathcal{T}$.

We next show that unraveling tolerance is indeed a sufficient condition for monadic Datalog$^{\neq}$-rewritability (and thus for PTIME query evaluation). In Section 6, we will establish a connection between query evaluation under DL TBoxes and constraint satisfaction problems (CSPs). The monadic Datalog$^{\neq}$ programs that we construct resemble the canonical monadic Datalog programs for CSPs [FV98].

Let $\mathcal{T}$ be an unraveling tolerant $\mathcal{ALCFI}$-TBox and $q = C_0(x)$ an ELIQ. We show how to construct a Datalog$^{\neq}$-rewriting of the OMQ $(\mathcal{T}, q(x))$. Using the construction from the proof of Theorem 19, one can extend this construction from ELIQs to PEQs. Recall from the proof of Theorem 19 that $\mathsf{cl}(\mathcal{T}, C_0)$ denotes the closure under single negation of the set of subconcepts of $\mathcal{T}$ and C_0. For an interpretation $\mathcal{I}$ and $d \in \Delta^{\mathcal{I}}$, we use $t_{\mathcal{T},q}^{\mathcal{I}}(d)$ to denote the set of concepts $C \in \mathsf{cl}(\mathcal{T}, C_0)$ such that $d \in C^{\mathcal{I}}$. A $\mathcal{T},q$-*type* is a subset $t \subseteq \mathsf{cl}(\mathcal{T}, C_0)$ such that for some model $\mathcal{I}$ of $\mathcal{T}$, we have $t = t_{\mathcal{T},q}^{\mathcal{I}}(d)$. We use $\mathsf{tp}(\mathcal{T}, q)$ to denote the set of all $\mathcal{T},q$-types. Observe that one can construct the set $\mathsf{tp}(\mathcal{T}, q)$ in exponential time as the set of all $t \subseteq \mathsf{cl}(\mathcal{T}, C_0)$ such that for any concept $\neg C \in \mathsf{cl}(\mathcal{T}, C_0)$ either $C \in t$ or $\neg C \in t$ and the concept $\bigsqcap_{C \in t} C$ is satisfiable in a model of $\mathcal{T}$.

For $t, t' \in \mathsf{tp}(\mathcal{T}, q)$ and r a role, we write $t \leadsto_r t'$ if there are a model $\mathcal{I}$ of $\mathcal{T}$ and $d, d' \in \Delta^{\mathcal{I}}$ such that $t_{\mathcal{T},q}^{\mathcal{I}}(d) = t$, $t_{\mathcal{T},q}^{\mathcal{I}}(d') = t'$, and $(d,d') \in r^{\mathcal{I}}$. One can construct the set of all (t, t', r) such that $t \leadsto_r t'$ in exponential time by checking for each candidate tuple (t, t', r) whether the concept

$$\left(\bigsqcap_{C \in t} C\right) \sqcap \exists r.\left(\bigsqcap_{C \in t'} C\right)$$

is satisfiable in a model of $\mathcal{T}$.

Introduce, for every set $T \subseteq \mathsf{tp}(\mathcal{T}, C_0)$ a unary IDB relation P_T. Let Π be the monadic Datalog$^{\neq}$ program that contains the following rules:

(1) $P_T(x) \leftarrow A(x)$ for all concept names $A \in \mathsf{cl}(\mathcal{T}, C_0)$ and $T = \{t \in \mathsf{tp}(\mathcal{T}, q) \mid A \in t\}$;
(2) $P_T(x) \leftarrow P_{T_0}(x) \wedge r(x,y) \wedge P_{T_1}(y)$ for all $T_0, T_1 \subseteq \mathsf{tp}(\mathcal{T}, q)$ and all role names r that occur in $\mathsf{cl}(\mathcal{T}, C_0)$ and their inverses, where $T = \{t \in T_0 \mid \exists t' \in T_1 : t \leadsto_r t'\}$;
(3) $P_{T_0 \cap T_1}(x) \leftarrow P_{T_0}(x) \wedge P_{T_1}(x)$ for all $T_0, T_1 \subseteq \mathsf{tp}(\mathcal{T}, q)$;
(4) $\mathsf{goal}(x) \leftarrow P_T(x)$ for all $T \subseteq \mathsf{tp}(\mathcal{T}, q)$ such that $t \in T$ implies $C_0 \in T$;
(5) $\mathsf{goal}(x) \leftarrow P_{\emptyset}(y)$;
(6) $\mathsf{goal}(x) \leftarrow r(y, z_1) \wedge r(y, z_2) \wedge z_1 \neq z_2$ for all $\mathsf{func}(r) \in \mathcal{T}$.

To show that Π is a rewriting of the OMQ $(\mathcal{T}, C_0(x))$, it suffices to establish the following lemma.

Lemma 24. $\mathcal{A} \models \Pi(a_0)$ iff $\mathcal{T}, \mathcal{A} \models C_0(a_0)$, for all ABoxes $\mathcal{A}$ and $a_0 \in \mathsf{Ind}(\mathcal{A})$.

Proof. The "if" direction is straightforward: by induction on the number of rule applications, one can show that whenever Π derives $P_T(a)$, then every model of $\mathcal{T}$ and $\mathcal{A}$ satisfies $t_{\mathcal{T},q}^{\mathcal{I}}(a) \in T$. By definition of the goal rules of Π, $\mathcal{A} \models \Pi(a_0)$ thus implies that every model of $\mathcal{T}$ and $\mathcal{A}$ makes $C_0(a_0)$ true or that $\mathcal{A}$ is inconsistent w.r.t. $\mathcal{T}$. Consequently, $\mathcal{T}, \mathcal{A} \models C_0(a_0)$.

For the "only if" direction, it suffices to show that $\mathcal{A} \not\models \Pi(a_0)$ implies $\mathcal{T}, \mathcal{A}^u \not\models C_0(a_0)$ since $\mathcal{T}$ is unraveling tolerant. Because of the rules in Π of the form (3), for every $a \in \mathsf{Ind}(\mathcal{A})$ we can find a unique minimal T_a such that $P_{T_a}(a)$ is derived by Π. Observe that, $A(\alpha) \in \mathcal{A}^u$, $\mathsf{tail}(\alpha) = a$, and $t \in T_a$ implies $A \in t$ because of the rules of the form (1) in Π and by construction of $\mathcal{A}^u$.

We first associate with every $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$ a concrete $\mathcal{T}, q$-type $t_\alpha \in T_{\mathsf{tail}(\alpha)}$. To start, we choose $t_a \in T_a$ arbitrarily for all $a \in \mathsf{Ind}(\mathcal{A})$. Now assume that t_α has already been chosen and that $\beta = \alpha r b \in \mathsf{Ind}(\mathcal{A}^u)$. Then $r(\mathsf{tail}(\alpha), b) \in \mathcal{A}$. Because of the rules in Π of the form (2) and (5), we can thus choose $t_\beta \in T_b$ such that $t_\alpha \rightsquigarrow_r t_\beta$. In this way, all types t_α will eventually be chosen. We now construct an interpretation $\mathcal{I}$, starting with

$$\begin{aligned}
\Delta^{\mathcal{I}} &= \mathsf{Ind}(\mathcal{A}^u) \\
A^{\mathcal{I}} &= \{\alpha \mid A \in t_\alpha\} \text{ for all concept names } A \\
r^{\mathcal{I}} &= \{(\alpha, \beta) \mid r(\alpha, \beta) \in \mathcal{A}^u\} \text{ for all role names } r.
\end{aligned}$$

Next, consider every $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$ and every $\exists r.C \in t_\alpha$ such that $\mathcal{A}^u$ does not contain an assertion $r(\alpha, \beta)$ with $C \in t_\beta$. First assume that $\mathsf{func}(r) \notin \mathcal{T}$. There must be a $\mathcal{T}, q$-type t such that $t_\alpha \rightsquigarrow_r t$ and $C \in t$. Choose a model $\mathcal{J}_{\alpha, \exists r.C}$ of $\mathcal{T}$ and $D = \sqcap t_a \sqcap \exists r. \sqcap t$, a $d \in D^{\mathcal{J}_{\alpha, \exists r.C}}$, and an $e \in (\sqcap t)^{\mathcal{J}_{\alpha, \exists r.C}}$ with $(d, e) \in r^{\mathcal{J}_{\alpha, \exists r.C}}$. W.l.o.g., we can assume that $\mathcal{J}_{\alpha, \exists r.C}$ is tree-shaped with root d. Let $\mathcal{J}_{\alpha, \exists r.C}^-$ be obtained from $\mathcal{J}_{\alpha, \exists r.C}$ by dropping the subtree rooted at e. Now disjointly add $\mathcal{J}_{\alpha, \exists r.C}^-$ to $\mathcal{I}$, additionally including (a, d) in $r^{\mathcal{I}}$. Now assume that $\mathsf{func}(r) \in \mathcal{T}$. Then, if there exists $r(\alpha, \beta) \in \mathcal{A}^u$, then $C \in t_\beta$ as otherwise we do not have $t_\alpha \rightsquigarrow_r t_\beta$. Thus, assume there is no $r(\alpha, \beta) \in \mathcal{A}^u$. There must be a $\mathcal{T}, q$-type t such that $t_\alpha \rightsquigarrow_r t$ and $C \in t$. We then have $D \in t$ for all $\exists r.D \in t_\alpha$ and so construct only a single $\mathcal{J}_{\alpha, \exists r.C}^-$ for the role r and disjointly add $\mathcal{J}_{\alpha, \exists r.C}^-$ to $\mathcal{I}$, additionally including (a, d) in $r^{\mathcal{I}}$. This finishes the construction of $\mathcal{I}$. The following claim can be proved by induction on C, details are omitted.

Claim. For all $C \in \mathsf{cl}(\mathcal{T}, C_0)$:
  (a) $\alpha \in C^{\mathcal{I}}$ iff $C \in t_\alpha$ for all $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$ and
  (b) $d \in C^{\mathcal{J}_{\alpha, \exists r.D}}$ iff $d \in C^{\mathcal{I}}$ for all $\mathcal{J}_{\alpha, \exists r.D}$ and all $d \in \Delta^{\mathcal{J}_{\alpha, \exists r.D}^-}$.

By construction of $\mathcal{I}$ and since $A(\alpha) \in \mathcal{A}^u$ implies $A \in t_\alpha$, $\mathcal{I}$ is a model of $\mathcal{A}$. Due to the rules in Π that are of the form (4), Point (a) of the claim yields $\mathcal{I} \not\models C_0(a_0)$. Finally, we observe that $\mathcal{I}$ is a model of $\mathcal{T}$. The concept inclusions in $\mathcal{T}$ are satisfied by the above claim, since $C \sqsubseteq D \in \mathcal{T}$ means that $C \in t$ implies $D \in t$ for every $\mathcal{T}, q$-type t, and since each $\mathcal{J}_{\alpha, \exists r.C}$ is a model of $\mathcal{T}$. Due to the rules in Π that are of the form (6) and since each $\mathcal{J}_{\alpha, \exists r.C}$ is a model of $\mathcal{T}$, all functionality assertions in $\mathcal{T}$ are satisfied as well. Summing up, we have shown that $\mathcal{T}, \mathcal{A}^u \not\models C_0(a_0)$, as required. ❑

Together with Theorem 19, we have established the following result.

Theorem 25. *Every unraveling tolerant $\mathcal{ALCFI}$-TBox is monadic Datalog$^{\neq}$-rewritable for PEQ.*

Together with Theorems 19 and 22, Theorem 25 also reproves the known PTIME upper bound for the data complexity of CQ-evaluation in Horn-$\mathcal{ALCFI}$ [EGOS08]. Note that it is not clear how to attain a proof of Theorem 25 via the CSP connection established in Section 6 since functional roles break this connection.

By Theorems 18 and 25, unraveling tolerance implies materializability unless PTIME $=$ NP. Based on the disjunction property, this implication can also be proved without the side condition.

Theorem 26. *Every unraveling tolerant $\mathcal{ALCFI}$-TBox is materializable.*

Proof. We show the contrapositive using a proof strategy that is very similar to the second step in the proof of Theorem 18. Thus, take an $\mathcal{ALCFI}$-TBox $\mathcal{T}$ that is not materializable. By Theorem 16, $\mathcal{T}$ does not have the disjunction property. Thus, there are an ABox $\mathcal{A}_\vee$, ELIQs $C_0(x_0), \ldots, C_k(x_k)$, and $a_1, \ldots, a_k \in \mathsf{Ind}(\mathcal{A}_\vee)$ such that $\mathcal{T}, \mathcal{A}_\vee \models C_0(a_0) \vee \cdots \vee C_k(a_k)$, but $\mathcal{T}, \mathcal{A}_\vee \not\models C_i(a_i)$ for all $i \leq k$. Let $\mathcal{A}_i$ be C_i viewed as a tree-shaped ABox with root b_i, for all $i \leq k$. Assume w.l.o.g.

that none of the ABoxes $\mathcal{A}_\vee, \mathcal{A}_0, \ldots, \mathcal{A}_k$ share any individual names and reserve fresh individual names $c_0, \ldots, c_k$ and fresh role names $r, r_0, \ldots, r_k$. Let the ABox $\mathcal{A}$ be the union of

$$\mathcal{A}_\vee \cup \mathcal{A}_0 \cup \cdots \cup \mathcal{A}_k \cup \{r(c, c_0), \ldots, r(c, c_k)\}$$

and

$$\{r_0(c_j, b_0), \ldots, r_{j-1}(c_j, b_{j-1}), r_j(c_j, a_j), r_{j+1}(c_j, b_{j+1}), \ldots, r_k(c_j, b_k)\}$$

for $1 \leq j \leq k$. Consider the ELIQ

$$q = \exists r.(\exists r_0.C_0 \sqcap \cdots \sqcap \exists r_k.C_k)(x).$$

By the following claim, $\mathcal{A}$ and q witness that $\mathcal{T}$ is not unraveling tolerant.

Claim. $\mathcal{T}, \mathcal{A} \models q(c)$, but $\mathcal{T}, \mathcal{A}^u \not\models q(c)$.

Proof. "$\mathcal{T}, \mathcal{A} \models q(c)$". Take a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$. By construction of $\mathcal{A}$, we have $a_i^{\mathcal{I}} \in (\exists r_j.C_j)^{\mathcal{I}}$ whenever $i \neq j$. Due to the edges $r_0(c_0, a_0), \ldots, r_k(c_k, a_k)$ and since $\mathcal{T}, \mathcal{A}_\vee \models C_0(a_0) \vee \cdots \vee C_k(a_k)$, we thus find at least one c_i such that $c_i^{\mathcal{I}} \in (\exists r_i.C_i)^{\mathcal{I}}$. Consequently, $\mathcal{I} \models q(c)$.

"$\mathcal{T}, \mathcal{A}^u \not\models q(c)$" (sketch). Consider the elements $crc_ir_ia_i$ in $\mathcal{A}^u$. Each such element is the root of a copy of the unraveling $\mathcal{A}_\vee^u$ of $\mathcal{A}_\vee$, restricted to those individual names in $\mathcal{A}_\vee$ that are reachable from a_i. Since $\mathcal{T}, \mathcal{A}_\vee \not\models C_i(a_i)$, we find a model $\mathcal{I}_i$ of $\mathcal{T}$ and $\mathcal{A}_\vee$ with $a_i \notin C_i^{\mathcal{I}_i}$. By unraveling $\mathcal{I}_i$, we obtain a model $\mathcal{I}_i^u$ of $\mathcal{T}$ and $\mathcal{A}_\vee^u$ with $a_i \notin C_i^{\mathcal{I}_i^u}$. Combining the models $\mathcal{I}_0^u, \ldots, \mathcal{I}_k^u$ in a suitable way, one can craft a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}_\vee^u$ such that $crc_ir_ia_i \notin C_i^{\mathcal{I}}$ for all $i \leq k$ and the 'role edges of $\mathcal{I}$' that concern the roles $r, r_0, \ldots, r_k$ are exactly those in $\mathcal{A}$. This implies $\mathcal{I} \not\models q(c)$ as desired.

In some more detail, $\mathcal{I}$ is obtained as follows. We can assume w.l.o.g. that the domains of $\mathcal{I}_0^u, \ldots, \mathcal{I}_k^u$ are disjoint. Take the disjoint union of $\mathcal{I}_0^u, \ldots, \mathcal{I}_k^u$, renaming a_i in $\mathcal{I}_i^u$ to $crc_ir_ia_i$ for all i. Now take copies $\mathcal{J}, \mathcal{J}_0, \ldots, \mathcal{J}_k$ of any model of $\mathcal{T}$, make sure that their domains are disjoint and that they are also disjoint from the domain of the model constructed so far. Additionally make sure that $c \in \Delta^{\mathcal{J}}$ and $c_i \in \Delta^{\mathcal{J}_i}$ for all i. Disjointly add these models to the model constructed so far. It can be verified that the model constructed up to this point is a model of $\mathcal{T}$. Add all role edges from $\mathcal{A}$ that concern the roles $r, r_0, \ldots, r_k$ to the resulting model, which has no impact on the satisfaction of $\mathcal{T}$ since $r, r_0, \ldots, r_k$ do not occur in $\mathcal{T}$. It can be verified that $\mathcal{I}$ is as required. ❑

5. DICHOTOMY FOR $\mathcal{ALCFI}$-TBOXES OF DEPTH ONE

In practical applications, the concepts used in TBoxes are often of very limited quantifier depth. Motivated by this observation, we consider TBoxes of *depth one* which are sets of CIs $C \sqsubseteq D$ such that no restriction $\exists r.E$ or $\forall r.E$ in C and D is in the scope of another restriction of the form $\exists r.E$ or $\forall r.E$. To confirm that this is indeed a practically relevant case, we have analyzed the 429 ontologies in the BioPortal repository,[3] finding that after removing all constructors that are not part of $\mathcal{ALCFI}$, more than 80% of them are of depth one. The main result of this section is a dichotomy between PTIME and CONP for TBoxes of depth one which is established by proving a converse of Theorem 26, that is, showing that materializability implies unraveling tolerance (and thus PTIME query evaluation and even monadic Datalog$^{\neq}$-rewritability by Theorem 25) for TBoxes of depth one. Together with Theorem 18, which says that non-materializability implies CONP-hardness, this yields the dichotomy.

We remark that the same strategy cannot be used to obtain a dichotomy in the case of unrestricted depth. In particular, the well-known technique of rewriting a TBox into depth one by introducing fresh concept names can change its complexity because it enables querying for concepts such as $\neg A$ or $\forall r.A$ which are otherwise 'invisible' to (positive existential) queries. For TBoxes of unrestricted depth (and even in $\mathcal{ALC}$) it is in fact neither the case that PTIME query evaluation implies unraveling tolerance (or even Datalog$^{\neq}$-rewritability) nor that materializability implies PTIME query evaluation. This is formally established in Section 6.

Theorem 27. *Every materializable $\mathcal{ALCFI}$-TBox of depth one is unraveling tolerant.*

Proof. Let $\mathcal{T}$ be a materializable TBox of depth one, $\mathcal{A}$ an ABox, $C_0(x)$ an ELIQ, and $a_0 \in \mathsf{Ind}(\mathcal{A})$ such that $\mathcal{T}, \mathcal{A}^u \not\models C_0(a_0)$. We have to show that $\mathcal{T}, \mathcal{A} \not\models C_0(a_0)$. It follows from $\mathcal{T}, \mathcal{A}^u \not\models C_0(a_0)$ that $\mathcal{A}^u$ is consistent w.r.t. $\mathcal{T}$. There must thus be a materialization $\mathcal{I}^u$ for $\mathcal{T}$ and $\mathcal{A}^u$, despite the fact that $\mathcal{A}^u$ is infinite: by Theorem 27, $\mathcal{T}$ has the disjunction property and the argument from the proof of Theorem 27 that the disjunction property implies materializability goes through without modification also for infinite ABoxes. Our aim is to turn $\mathcal{I}^u$ into a model $\mathcal{I}$ of $\mathcal{A}$ and $\mathcal{T}$ such that $\mathcal{I} \not\models C_0(a_0)$. To achieve this, we first uniformize $\mathcal{I}^u$ in a suitable way.

We assume w.l.o.g. that $\mathcal{I}^u$ has forest-shape, i.e., that $\mathcal{I}^u$ can be constructed by selecting a tree-shaped interpretation $\mathcal{I}_\alpha$ with root α for each $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$, then taking the disjoint union of all these interpretations, and finally adding role edges (α, β) to $r^{\mathcal{I}^u}$ whenever $r(\alpha, \beta) \in \mathcal{A}^u$. In fact, to achieve the desired shape we can take the i-unfolding of $\mathcal{I}^u$ defined and analysed in Lemmas 13 and 14, where we start the i-unfolding from the elements of $\mathsf{Ind}(\mathcal{A}^u) \subseteq \Delta^{\mathcal{I}^u}$.

We start with exhibiting a self-similarity inside the unraveled ABox $\mathcal{A}^u$ and inside $\mathcal{I}^u$.

Claim 1. For all $\alpha, \beta \in \mathsf{Ind}(\mathcal{A}^u)$ with $\mathsf{tail}(\alpha) = \mathsf{tail}(\beta)$,

    (1) $\mathcal{A}^u \models C(\alpha)$ iff $\mathcal{A}^u \models C(\beta)$ for all ELIQs $C(x)$;

    (2) $\alpha \in C^{\mathcal{I}^u}$ iff $\beta \in C^{\mathcal{I}^u}$ for all concepts C built only from concept names, $\neg$, and $\sqcap$.

To establish Point (1), take $\alpha, \beta \in \mathsf{Ind}(\mathcal{A}^u)$ such that $\mathsf{tail}(\alpha) = \mathsf{tail}(\beta)$ and $\mathcal{A}^u \not\models C(\alpha)$. Then there is a model $\mathcal{I}$ of $\mathcal{A}^u$ and $\mathcal{T}$ such that $\mathcal{I} \not\models C(\alpha)$. One can find a model $\mathcal{J}$ of $\mathcal{A}^u$ and $\mathcal{T}$ such that $\mathcal{J} \not\models C(\beta)$, as follows. By construction of $\mathcal{A}^u$, there is an isomorphism $\iota : \mathsf{Ind}(\mathcal{A}^u) \to \mathsf{Ind}(\mathcal{A}^u)$ with $\iota(\alpha) = \beta$ such that $A(\gamma) \in \mathcal{A}^u$ iff $A(\iota(\gamma)) \in \mathcal{A}^u$ and $r(\gamma, \gamma') \in \mathcal{A}^u$ iff $r(\iota(\gamma), \iota(\gamma')) \in \mathcal{A}^u$ for all $\gamma \in \mathsf{Ind}(\mathcal{A}^u)$, all concept names A, and all role names r. We obtain $\mathcal{J}$ from $\mathcal{I}$ by renaming each $\gamma \in \mathsf{Ind}(\mathcal{A}^u)$ with $\iota(\gamma)$. Point (2) can be proved by a straightforward induction on C. The base case uses Point (1) and the fact that $\mathcal{I}^u$ is a materialization of $\mathcal{T}$ and $\mathcal{A}$. This finishes the proof of Claim 1.

Now for the announced uniformization of $\mathcal{I}^u$. What we want to achieve is that for all $\alpha, \beta \in \mathsf{Ind}(\mathcal{A}^u)$, $\mathsf{tail}(\alpha) = \mathsf{tail}(\beta)$ implies $\mathcal{I}_\alpha = \mathcal{I}_\beta$ (recall that $\mathcal{I}_\alpha$ is the tree component of $\mathcal{I}^u$ rooted at α, and likewise for $\mathcal{I}_\beta$). Construct the interpretation $\mathcal{J}^u$ as follows:

- for each $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$ with $\mathsf{tail}(\alpha) = a$, take a copy $\mathcal{J}_\alpha$ of $\mathcal{I}_a$ with the root a renamed to α;
- then $\mathcal{J}^u$ is the disjoint union of all interpretations $\mathcal{J}_\alpha$, $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$, extended with a role edge $(\alpha, \beta) \in r^{\mathcal{J}^u}$ whenever $r(\alpha, \beta) \in \mathcal{A}^u$.

Our next aim is to show that $\mathcal{J}^u$ is as required, that is, it is a model of $\mathcal{T}$ and $\mathcal{A}^u$ and satisfies $\mathcal{J}^u \not\models C_0(a_0)$.

It is indeed straightforward to verify that $\mathcal{J}^u$ is a model of $\mathcal{A}^u$: all role assertions are satisfied by construction; moreover, $A(\alpha) \in \mathcal{A}^u$ implies $A(a) \in \mathcal{A}^u$ where $a = \mathsf{tail}(\alpha)$, thus $a \in A^{\mathcal{I}_u}$ and $\alpha \in A^{\mathcal{J}_u}$.

Next, we show that $\mathcal{J}^u$ is a model of $\mathcal{T}$. Let $f : \Delta^{\mathcal{J}^u} \to \Delta^{\mathcal{I}^u}$ be a mapping that assigns to each domain element of $\mathcal{J}^u$ the original element in $\mathcal{I}^u$ of which it is a copy.

Claim 2. $d \in C^{\mathcal{J}^u}$ iff $f(d) \in C^{\mathcal{I}^u}$ for all $d \in \Delta^{\mathcal{J}^u}$ and $\mathcal{ALCI}$-concepts C of depth one.

The proof of claim 2 is by induction on the structure of C. We assume w.l.o.g. that C is built only from the constructors $\neg$, $\sqcap$, and $\exists r.C$. The base case, where C is a concept name, is an immediate consequence of the definition of $\mathcal{J}^u$. The case where $C = \neg D$ and $C = D_1 \sqcap D_2$ is routine. It thus remains to consider the case $C = \exists r.D$, where D is built from $\neg$ and $\sqcap$ only.

First let $d \in C^{\mathcal{J}^u}$. Then there is a $(d, e) \in r^{\mathcal{J}^u}$ with $e \in D^{\mathcal{J}^u}$. First assume that the edge (d, e) was added to $r^{\mathcal{J}^u}$ because $d = \alpha$ and $e = \beta$ for some $\alpha, \beta \in \mathsf{Ind}(\mathcal{A}^u)$ with $r(\alpha, \beta) \in \mathcal{A}^u$. Let $\mathsf{tail}(\alpha) = a$ and $\mathsf{tail}(\beta) = b$. Then we have $f(\alpha) = a$ and $f(\beta) = b$. By construction of $\mathcal{A}^u$, $r(\alpha, \beta) \in \mathcal{A}^u$ implies that $\beta = \alpha r b$ or $\alpha = \beta r^- a$. In both cases we have $r(a, b) \in \mathcal{A}$, thus $r(a, arb) \in \mathcal{A}^u$, thus $(a, arb) \in r^{\mathcal{I}^u}$. Since $\beta = e \in D^{\mathcal{J}^u}$, induction hypothesis yields that $b \in D^{\mathcal{I}^u}$. From Point (2) of Claim 1, we obtain $arb \in D^{\mathcal{I}^u}$ and we are done. Now assume that there is an $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$ such that $(d, e) \in \mathcal{J}_\alpha$. By construction of $\mathcal{J}^u$, we then have $(f(d), f(e)) \in r^{\mathcal{I}^u}$ and induction hypothesis yields $f(e) \in D^{\mathcal{I}^u}$.

Now let $f(d) \in C^{\mathcal{I}^u}$. Then there is an $(f(d), e) \in r^{\mathcal{I}^u}$ with $e \in D^{\mathcal{I}^u}$. First assume that $f(d) = \alpha$ and $e = \beta$ for some $\alpha, \beta \in \mathsf{Ind}(\mathcal{A}^u)$ with $r(\alpha, \beta) \in \mathcal{A}^u$. Since $f(d) \in \mathsf{Ind}(\mathcal{A}^u)$, we must have $d = \gamma \in \mathsf{Ind}(\mathcal{A}^u)$ and $f(d) = a \in \mathsf{Ind}(\mathcal{A})$ with $\mathsf{tail}(\gamma) = a$. By construction of $\mathcal{A}^u$, $r(a, \beta) \in \mathcal{A}^u$ implies that $\beta = arb$, thus $r(a, b) \in \mathcal{A}$, thus $r(\gamma, \delta) \in \mathcal{A}^u$ with (i) $\delta = \gamma r b$ or (ii) $\gamma = \delta r^- a$ and $\mathsf{tail}(\delta) = b$. Since $arb = e \in D^{\mathcal{I}^u}$, Point (2) of Claim 1 yields $b \in D^{\mathcal{I}_u}$. Since $\mathsf{tail}(\delta) = b$ implies $f(\delta) = b$, induction hypothesis yields $\delta \in D^{\mathcal{J}^u}$ and we are done. Now assume that there is an $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$ such that $(f(d), e) \in \mathcal{I}_\alpha$. By construction of $\mathcal{J}^u$, $f(d)$ being in $\mathcal{I}_\alpha$ implies that $\alpha = a$ for some $a \in \mathsf{Ind}(\mathcal{A})$ and that there is an $\alpha' \in \mathsf{Ind}(\mathcal{A}^u)$ such that d is in $\mathcal{J}_{\alpha'}$ and $\mathsf{tail}(\alpha') = a$. Again by construction of $\mathcal{J}^u$, we thus find an e' in $\mathcal{J}_{\alpha'}$ with $f(e') = e$ and $(d, e') \in r^{\mathcal{J}_{\alpha'}} \subseteq r^{\mathcal{J}^u}$. Induction hypothesis yields $e' \in D^{\mathcal{J}^u}$. This finishes the proof of Claim 2.

It follows from Claim 2 that $\mathcal{J}^u$ satisfies all CIs in $\mathcal{T}$. To show that $\mathcal{J}^u$ is a model of $\mathcal{T}$, it remains to show that $\mathcal{J}^u$ satisfies all functionality assertions. Thus, let $\mathsf{func}(r) \in \mathcal{T}$ and $d \in \Delta^{\mathcal{J}^u}$. If $d \notin \mathsf{Ind}(\mathcal{A}^u)$, then it is straightforward to verify that, by construction of $\mathcal{J}^u$, d has at most one r-successor in $\mathcal{J}^u$. Now assume $d = \alpha \in \mathsf{Ind}(\mathcal{A}^u)$ and let $\mathsf{tail}(\alpha) = a$. By construction of $\mathcal{J}^u$ and $\mathcal{A}^u$, α has the same number of r-successors in $\mathcal{J}^u$ as a in $\mathcal{I}^u$. Since $\mathcal{I}^u$ satisfies $\mathsf{func}(r)$, α can have at most one r-successor in $\mathcal{J}^u$.

The final condition that $\mathcal{J}^u$ should satisfy is $\mathcal{J}^u \not\models C_0(a_0)$. Assume to the contrary. We view $C_0(x_0)$ as a (tree-shaped) CQ. Take a homomorphism h from $C_0(x_0)$ to $\mathcal{J}^u$ with $h(x_0) = a_0$. (In this proof we consider homomorphisms that do not have to preserve any individual names.) Let the CQ $q(x)$ be obtained from $C_0(x_0)$ by identifying variables y_1, y_2 whenever $h(y_1) = h(y_2)$. To achieve a contradiction, it suffices to exhibit a homomorphism h' from $q(x_0)$ to $\mathcal{I}^u$ with $h'(x_0) = a_0$. We start with setting $h'(x) = h(x)$ whenever $h(x) \in \mathsf{Ind}(\mathcal{A}^u)$. Let q' be obtained from $q(x_0)$ by dropping all role atoms $r(x, y)$ with $h'(x)$ and $h'(y)$ already defined (which are satisfied under h' by construction of $\mathcal{J}^u$ and since $\mathcal{I}^u$ is a model of $\mathcal{A}$). Because of the forest shape of $\mathcal{J}^u$ and by construction, q' is a disjoint union of ELIQs such that, in each ELIQ $C(x)$ contained in q', h' is defined for the root x of $C(x)$, but not for any other variable in it. Consequently, it suffices to show that whenever $\mathcal{J}_\alpha \models C(\alpha)$ for some ELIQ $C(x)$ and $\alpha \in \mathsf{Ind}(\mathcal{A}^u)$, then $\mathcal{I}^u \models C(\alpha)$; the remaining part of h' can then be constructed in a straightforward way. Now $\mathcal{J}_\alpha \models C(\alpha)$ implies $\mathcal{I}_a \models C(a)$ where $a = \mathsf{tail}(\alpha)$ by choice of $\mathcal{J}_\alpha$, which yields $\mathcal{I}^u \models C(a)$ and thus $\mathcal{I}^u \models C(\alpha)$ by Point (1) of Claim 1.

This finishes the construction and analysis of the uniform model $\mathcal{J}^u$. It remains to convert $\mathcal{J}^u$ into a model $\mathcal{I}$ of $\mathcal{T}$ and the original ABox $\mathcal{A}$ such that $\mathcal{I} \not\models C_0(a_0)$:

- take the disjoint union of the components $\mathcal{J}_a$ of $\mathcal{J}^u$, for each $a \in \mathsf{Ind}(\mathcal{A})$;

- add the edge (a, b) to $r^{\mathcal{I}}$ whenever $r(a, b) \in \mathcal{A}$.

It is straightforward to verify that $\mathcal{I}$ is a model of $\mathcal{A}$: all role assertions are satisfied by construction of $\mathcal{I}$; moreover, $A(a) \in \mathcal{A}$ implies $A(a) \in \mathcal{A}^u$ implies $a \in A^{\mathcal{J}^u}$ implies $a \in A^{\mathcal{I}}$. To show that $\mathcal{I}$ is a model of $\mathcal{T}$ and that $\mathcal{I} \not\models C_0(a_0)$, we first observe the following. A *bisimulation between interpretations* $\mathcal{I}_1$ *and* $\mathcal{I}_2$ is a relation $S \subseteq \Delta^{\mathcal{I}_1} \times \Delta^{\mathcal{I}_2}$ such that

(1) for all $A \in \mathsf{N_C}$ and $(d_1, d_2) \in S$: $d_1 \in A^{\mathcal{I}_1}$ iff $d_2 \in A^{\mathcal{I}_2}$;
(2) for all $r \in \mathsf{N_R} \cup \{s^- \mid s \in \mathsf{N_R}\}$: if $(d_1, d_2) \in S$ and $(d_1, d_1') \in r^{\mathcal{I}_1}$, then there exists $d_2' \in \Delta^{\mathcal{I}_2}$ such that $(d_1', d_2') \in S$ and $(d_2, d_2') \in r^{\mathcal{I}_2}$;
(3) for all $r \in \mathsf{N_R} \cup \{s^- \mid s \in \mathsf{N_R}\}$: if $(d_1, d_2) \in S$ and $(d_2, d_2') \in r^{\mathcal{I}_2}$, then there exists $d_1' \in \Delta^{\mathcal{I}_1}$ such that $(d_1', d_2') \in S$ and $(d_1, d_1') \in r^{\mathcal{I}_2}$.

Recall that, whenever there is a bisimulation S between $\mathcal{I}_1$ and $\mathcal{I}_2$ with $(d, e) \in S$, then $d \in C^{\mathcal{I}_1}$ iff $e \in C^{\mathcal{I}_2}$ for all $\mathcal{ALCI}$-concepts C [GO07, LPW11].

Claim 3. There is a bisimulation S between $\mathcal{J}^u$ and $\mathcal{I}$ such that $(a, a) \in S$ for all $a \in \mathsf{Ind}(\mathcal{A})$.

Since $\mathcal{J}^u$ is uniform in the sense that $\mathcal{J}_\alpha$ is isomorphic to $\mathcal{J}_\beta$ whenever $\mathsf{tail}(\alpha) = \mathsf{tail}(\beta)$, we find a bisimulation between $\mathcal{J}_\alpha$ and $\mathcal{J}_a$ whenever $\mathsf{tail}(\alpha) = a$. It can be verified that the union of all these bisimulations qualifies as the desired bisimulation S for Claim 3. Thus, Claim 3 is proved.

It follows from Claim 3 that $\mathcal{I}$ satisfies all concept inclusions in $\mathcal{T}$, and that $\mathcal{I} \not\models C_0(a_0)$. It thus remains to verify that $\mathcal{I}$ also satisfies all functionality assertions in $\mathcal{T}$. This can be done in the same way in which we have verified that $\mathcal{J}^u$ satisfies all those assertions. ❏

The desired dichotomy follows: If an $\mathcal{ALCFI}$-TBox $\mathcal{T}$ of depth one is materializable, then PEQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME and monadic Datalog$^{\neq}$-rewritable by Theorems 27 and 25. Otherwise, ELIQ-evaluation w.r.t. $\mathcal{T}$ is CONP-complete by Theorem 18.

Theorem 28 (Dichotomy). *For every $\mathcal{ALCFI}$-TBox $\mathcal{T}$ of depth one, one of the following is true:*

- *$\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$ is in* PTIME *for any $\mathcal{Q} \in \{PEQ,CQ,ELIQ\}$ (and monadic Datalog$^{\neq}$-rewritable);*
- *$\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$ is* CONP-*complete for any $\mathcal{Q} \in \{PEQ,CQ,ELIQ\}$.*

For example of depth one TBoxes for which query evaluation is in PTIME and for which it is CONP-hard, please see Example 9; there, cases (1) and (2) are materializable and thus in PTIME while case (3) is not materializable and thus CONP-hard.

6. QUERY EVALUATION IN $\mathcal{ALC}/\mathcal{ALCI}$ = CSP

We drop functional roles and consider TBoxes formulated in $\mathcal{ALC}$ and in $\mathcal{ALCI}$ showing that query evaluation w.r.t. TBoxes from these classes has the same computational power as non-uniform CSPs, in the following sense:

(1) for every OMQ $(\mathcal{T}, q)$ with $\mathcal{T}$ an $\mathcal{ALCI}$-TBox and q an ELIQ, there is a CSP such that the complement of the CSP and the query evaluation problem for the $(\mathcal{T}, q)$ are reducible to each other in polynomial time;
(2) for every CSP, there is an $\mathcal{ALC}$-TBox $\mathcal{T}$ such that the CSP is equivalent to the complement of evaluating an OMQ $(\mathcal{T}, \exists x\, M(x))$ and, conversely, for *every* OMQ $(\mathcal{T}, q)$ query evaluation can be reduced in polynomial time to the CSP's complement.

This result has many interesting consequences. In particular, the PTIME/NP-dichotomy for non-uniform CSPs [Bul17, Zhu17], formerly also known as the Feder-Vardi conjecture, yields a PTIME/CONP-dichotomy for query evaluation w.r.t. $\mathcal{ALC}$-TBoxes (equivalently: w.r.t. $\mathcal{ALCI}$-TBoxes). Remarkably, all this is true already for *materializable* TBoxes. By Theorem 19 and since we carefully choose the appropriate query language in each technical result below, it furthermore holds for any of the query languages ELIQ, CQ, and PEQ (and ELQ for $\mathcal{ALC}$-TBoxes).

We begin by introducing CSPs. Since every CSP is equivalent to a CSP with a single predicate that is binary, up to polynomial time reductions [FV98], we consider CSPs over unary and binary predicates (concept names and role names) only. A *signature* Σ is a finite set of concept and role names. We use $\mathsf{sig}(\mathcal{T})$ to denote the set of all concept and role names that occur in the TBox $\mathcal{T}$. An ABox $\mathcal{A}$ is a Σ-*ABox* if all concept and role names in $\mathcal{A}$ are in Σ. Moreover, we write $\mathcal{A}|_\Sigma$ to denote the restriction of an ABox $\mathcal{A}$ to the assertions that use a symbol from Σ. For two finite Σ-ABoxes $\mathcal{A}$ and $\mathcal{B}$, we write $\mathcal{A} \to \mathcal{B}$ if there is a homomorphism from $\mathcal{A}$ to $\mathcal{B}$ that does not have to preserve any individual names. A Σ-ABox $\mathcal{B}$ gives rise to the following (non-uniform) constraint satisfaction problem $\mathsf{CSP}(\mathcal{B})$: given a finite Σ-ABox $\mathcal{A}$, decide whether $\mathcal{A} \to \mathcal{B}$. $\mathcal{B}$ is called the *template* of $\mathsf{CSP}(\mathcal{B})$. Many problems in NP can be given in the form $\mathsf{CSP}(\mathcal{B})$. For example, k-colorability is $\mathsf{CSP}(\mathcal{C}_k)$, where $\mathcal{C}_k$ is the $\{r\}$-ABox that contains $r(i, j)$ for all $1 \leq i \neq j \leq k$.

We now formulate and prove Points (1) and (2) from above, starting with (1). The following is proved in [BtCLW14].

Theorem 29. *For every $\mathcal{ALCI}$-TBox $\mathcal{T}$ and ELIQ $C(x)$, one can compute a template $\mathcal{B}$ in exponential time such that the query evaluation problem for the OMQ $(\mathcal{T}, C(x))$ and the complement of $\mathsf{CSP}(\mathcal{B})$ are polynomial time reducible to each other.*

The proof of Theorem 29 given in [BtCLW14] proceeds in two steps. To deal with answer variables, it uses *generalized CSPs with constants*, defined by a finite set of templates (instead of a single one) and admitting the inclusion of constant symbols in the signature of the CSP (instead of only relation symbols). One shows that (i) for every OMQ $(\mathcal{T}, C(x))$, one can construct a generalized CSP with a single constant whose complement is mutually reducible in polynomial time with the query evaluation problem for $(\mathcal{T}, C(x))$ and (ii) every generalized CSP with constants is mutually reducible in polynomial time with some standard CSP. For the reader's convenience, we illustrate the construction of the template from a given OMQ, concentrating on *Boolean ELIQs* which are of the form $\exists x\, C(x)$ with $C(x)$ an ELIQ. In this special case, one can avoid the use of generalized CSPs with constants.

Theorem 30. *Let $\mathcal{T}$ be an $\mathcal{ALCI}$-TBox, $q = \exists x\, C(x)$ with $C(x)$ an ELIQ, and Σ the signature of $\mathcal{T}$ and q. Then one can construct (in time single exponential in $|\mathcal{T}| + |C|$) a Σ-template $\mathcal{B}_{\mathcal{T},q}$ such that for all ABoxes $\mathcal{A}$:*

$$(\mathsf{HomDual}) \quad \mathcal{T}, \mathcal{A} \models q \ \textit{ iff } \ \mathcal{A}|_\Sigma \not\to \mathcal{B}_{\mathcal{T},q}$$

Proof. Assume $\mathcal{T}$ and $q = \exists x\, C(x)$ are given. We use the notation from the proof of Theorem 25. Thus, $\mathsf{cl}(\mathcal{T}, C)$ denotes the closure under single negation of the set of subconcepts of $\mathcal{T}$ and C, $\mathsf{tp}(\mathcal{T}, q)$ denotes the set of $\mathcal{T}, q$-types and for $t, t' \in \mathsf{tp}(\mathcal{T}, q)$ we write $t \leadsto_r t'$ if t and t' can be satisfied in domain elements of a model of $\mathcal{T}$ that are related by r. Now, a $\mathcal{T}, q$-type t *omits* q if it is satisfiable in a model $\mathcal{I}$ of $\mathcal{T}$ with $C^{\mathcal{I}} = \emptyset$. Let $\mathcal{B}_{\mathcal{T},q}$ be the set of assertions $A(t)$ such t omits q and $A \in t$ and $r(t, t')$ such that t and t' omit q and $t \leadsto_r t'$. It is not difficult to show that condition (HomDual) holds for all ABoxes $\mathcal{A}$. Observe that $\mathcal{B}_{\mathcal{T},q}$ can be constructed in exponential time since the set of $\mathcal{T}, q$-types omitting q can be constructed in exponential time. ❑

Example 31. *Let $\mathcal{T} = \{A \sqsubseteq \forall r.B\}$ and define $q = \exists x\, B(x)$. Then up to isomorphism, $\mathcal{B}_{\mathcal{T},q}$ is $\{r(a, a), r(a, b), A(b), r(a, c)\}$.*

As a consequence of Theorem 29, we obtain the following dichotomy result.

Theorem 32 (Dichotomy). *For every $\mathcal{ALCI}$-TBox $\mathcal{T}$, one of the following is true:*

- *$\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$ is in PTIME for any $\mathcal{Q} \in \{PEQ,CQ,ELIQ\}$;*
- *$\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$ is CONP-complete for any $\mathcal{Q} \in \{PEQ,CQ,ELIQ\}$.*

For $\mathcal{ALC}$-TBoxes, this dichotomy additionally holds for ELQs.

Proof. Assume to the contrary of what is to be shown that there is an $\mathcal{ALCI}$-TBox $\mathcal{T}$ such that $\mathcal{Q}$-evaluation w.r.t. $\mathcal{T}$ is neither in PTIME nor CONP-hard, for some $\mathcal{Q} \in \{PEQ,CQ,ELIQ\}$. Then by Theorem 19, the same holds for ELIQ-evaluation w.r.t. $\mathcal{T}$. It follows that there is a concrete ELIQ q such that query evaluation for $(\mathcal{T}, q)$ is CONP-intermediate. By Theorem 29, there is a template $\mathcal{B}$ such that evaluating $(\mathcal{T}, q)$ is mutually reducible in polynomial time with the complement of $\mathsf{CSP}(\mathcal{B})$. Thus $\mathsf{CSP}(\mathcal{B})$ is NP-intermediate, a contradiction to the fact that there are no such CSPs [Bul17, Zhu17]. ❏

We now establish Point (2) from the beginning of the section. In a sense, the following provides a converse to Theorem 29.

Theorem 33. *For every template $\mathcal{B}$ over signature Σ, one can construct in polynomial time a materializable $\mathcal{ALC}$-TBox $\mathcal{T_B}$ such that, for a distinguished concept name M, the following hold:*

(1) *$\mathsf{CSP}(\mathcal{B})$ is equivalent to the complement of the OMQ $(\mathcal{T_B}, \exists x\, M(x))$ in the sense that for every Σ-ABox $\mathcal{A}$, $\mathcal{A} \to \mathcal{B}$ iff $\mathcal{T_B}, \mathcal{A} \not\models \exists x\, M(x)$;*

(2) *the query evaluation problem for $(\mathcal{T_B}, q)$ is polynomial time reducible to the complement of $\mathsf{CSP}(\mathcal{B})$, for all PEQs q.*

Note that the equivalence formulated in Point (1) implies polynomial time reducibility of $\mathsf{CSP}(\mathcal{B})$ to the complement of $(\mathcal{T_B}, \exists x\, M(x))$ and vice versa, but is much stronger than that.

Our approach to proving Theorem 33 is to generalize the reduction of k-colorability to query evaluation w.r.t. $\mathcal{ALC}$-TBoxes discussed in Examples 2 and 4, where the main challenge is to overcome the observation from Example 4 that PTIME CSPs such as 2-colorability may be translated into CONP-hard TBoxes. Note that this is due to the disjunction in the TBox $\mathcal{T}_k$ of Example 2, which causes non-materializability. Our solution is to replace the concept names $A_1, \ldots, A_k$ in $\mathcal{T}_k$ with suitable compound concepts that are 'invisible' to the (positive existential) query. Unlike the original depth one TBox $\mathcal{T}_k$, the resulting TBox is of depth three. This 'hiding' of concept names also plays a crucial role in the proofs of non-dichotomy and undecidability presented in Section 7.[4] We now formally develop this idea and establish some crucial properties of the TBoxes that are obtained by hiding concept names (which are called enriched abstractions below). We return to the proof of Theorem 33 afterwards.

Let $\mathcal{T}$ be an $\mathcal{ALCI}$-TBox and $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ a signature that contains all role names in $\mathcal{T}$. Our aim is to hide all concept names that are not in Σ. For $B \in \mathsf{N_C} \setminus \Sigma$, let Z_B be a fresh concept name and let r_B and s_B be fresh role names. The *abstraction of B* is the $\mathcal{ALC}$-concept $H_B := \forall r_B.\exists s_B.\neg Z_B$. The Σ-*abstraction C' of a (potentially compound) concept C* is obtained from C by replacing every $B \in \mathsf{N_C} \setminus \Sigma$ with H_B. The Σ-*abstraction of a TBox $\mathcal{T}$* is obtained from $\mathcal{T}$ by replacing all concepts in $\mathcal{T}$ with their Σ-abstractions. We associate with $\mathcal{T}$ and Σ an auxiliary TBox

$$\mathcal{T}^\exists = \{\top \sqsubseteq \exists r_B.\top, \top \sqsubseteq \exists s_B.Z_B \mid B \in \Sigma\}$$

---

[4]The 'hiding technique' introduced here has been adopted in [BLR⁺16] in the context of query inseparability.

Finally, $\mathcal{T}' \cup \mathcal{T}^\exists$ is called the *enriched Σ-abstraction* of $\mathcal{T}$ and Σ. To hide the concept names that are not in Σ, we can replace a TBox $\mathcal{T}$ with its enriched abstraction. The following example shows that the TBox $\mathcal{T}^\exists$ is crucial for this: without $\mathcal{T}^\exists$, disjunctions in $\mathcal{T}$ over concept names from Σ can still induce disjunctions in the Σ-abstraction.

Example 34. *Let $\mathcal{T} = \{A \sqsubseteq \neg B_1 \sqcup \neg B_2\}$ and $\Sigma = \{A\}$. Then $\mathcal{T}' = \{A \sqsubseteq \neg H_{B_1} \sqcup \neg H_{B_2}\}$ is the Σ-abstraction of $\mathcal{T}$. For $\mathcal{A} = \{A(a)\}$, we derive $\mathcal{T}', \mathcal{A} \models \exists r_{B_1}.\top(a) \vee \exists r_{B_2}.\top(a)$ but $\mathcal{T}', \mathcal{A} \not\models \exists r_{B_1}.\top(a)$ and $\mathcal{T}', \mathcal{A} \not\models \exists r_{B_2}.\top(a)$. Thus $\mathcal{T}'$ does not have the ABox disjunction property and is not materializable. In contrast, it follows from Lemma 35 below that $\mathcal{T}' \cup \mathcal{T}^\exists$ is materializable and, in fact, $\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models q(a)$ iff $\mathcal{T}^\exists, \mathcal{A} \models q(a)$ holds for all PEQs q.*

In the proof of Theorem 33 and in Section 7, we work with TBoxes that enjoy two crucial properties which ensure a good behaviour of enriched Σ-abstractions. We introduce these properties next.

A TBox $\mathcal{T}$ *admits trivial models* if the singleton interpretation $\mathcal{I}$ with $X^\mathcal{I} = \emptyset$ for all $X \in \mathsf{N_C} \cup \mathsf{N_R}$ is a model of $\mathcal{T}$. It is Σ-*extensional* if for every Σ-ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{T}$, there exists a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ such that $\Delta^\mathcal{I} = \mathsf{Ind}(\mathcal{A})$, $A^\mathcal{I} = \{a \mid A(a) \in \mathcal{A}\}$ for all concept names $A \in \Sigma$, and $r^\mathcal{I} = \{(a, b) \mid r(a, b) \in \mathcal{A}\}$ for all role names $r \in \Sigma$.

The following lemma summarizes the main properties of abstractions. Point (1) relates consistency of ABoxes w.r.t. a TBox $\mathcal{T}$ to consistency w.r.t. their enriched Σ-abstraction $\mathcal{T}' \cup \mathcal{T}^\exists$. Note that the ABox $\mathcal{A}$ might contain the fresh symbols from $\mathcal{T}'$ but these have no impact on consistency (as witnessed by the use of $\mathcal{A}|_\Sigma$ rather than $\mathcal{A}$ on the left-hand side of the equivalence). Point (2) is similar to Point (1) but concerns the evaluation of OMQs based on $\mathcal{T}$ and based on $\mathcal{T}' \cup \mathcal{T}^\exists$; we only consider a restricted form of actual queries that are sufficient for the proofs in Section 7. Points (3) and (4) together state that evaluating OMQs $(\mathcal{T}' \cup \mathcal{T}^\exists, q)$ with q a PEQ is tractable on ABoxes whose Σ-part is consistent w.r.t. $\mathcal{T}$.

Lemma 35. *Let $\mathcal{T}$ be an $\mathcal{ALCI}$-TBox, $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ contain all role names in $\mathcal{T}$, and assume that $\mathcal{T}$ is Σ-entensional and admits trivial models. Let $\mathcal{T}' \cup \mathcal{T}^\exists$ be the enriched Σ-abstraction of $\mathcal{T}$. Then for every ABox $\mathcal{A}$ and all concept names A (that are not among the fresh symbols in $\mathcal{T}'$):*

(1) $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{T}$ iff $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$;

(2) for all $a \in \mathsf{Ind}(\mathcal{A})$ and the Σ-abstraction A' of A:

$$\mathcal{T}, \mathcal{A}|_\Sigma \models A(a) \quad \textit{iff} \quad \mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models A'(a)$$

and

$$\mathcal{T}, \mathcal{A}|_\Sigma \models \exists x\, A(x) \quad \textit{iff} \quad \mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models \exists x\, A'(x);$$

(3) $\mathcal{T}^\exists$ is monadic Datalog$^{\neq}$-rewritable for PEQs;

(4) if $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{T}$, then

$$\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models q(\vec{a}) \quad \textit{iff} \quad \mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$$

for all PEQs q and all $\vec{a}$.

Proof. (1) Assume first that $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$. We show that $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{T}$. Take a model $\mathcal{I}$ of $\mathcal{T}' \cup \mathcal{T}^\exists$ and $\mathcal{A}$. Define an interpretation $\mathcal{J}$ in the same way as $\mathcal{I}$ except that $B^\mathcal{J} := H_B^\mathcal{I}$ for all $B \in \mathsf{N_C} \setminus \Sigma$. It is straightforward to show by induction for all $\mathcal{ALCI}$-concepts D not using the fresh symbols from Σ-abstractions and their Σ-abstractions D': $d \in D^\mathcal{J}$ iff $d \in D'^\mathcal{I}$, for all $d \in \Delta^\mathcal{I}$. Thus $\mathcal{J}$ is a model of $\mathcal{T}$ and $\mathcal{A}|_\Sigma$ and it follows that $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{T}$.

Now assume that $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{T}$. We show that $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$. Take a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}|_\Sigma$. Construct a model $\mathcal{J}$ of $\mathcal{T}' \cup \mathcal{T}^\exists$ and $\mathcal{A}$ as follows: $\Delta^\mathcal{J}$ is the set of words

$w = dv_1 \cdots v_n$ such that $d \in \Delta^{\mathcal{I}}$ and $v_i \in \{r_B, s_B, \bar{s}_B \mid B \in \mathsf{N_C} \setminus \Sigma\}$ where $v_i \neq \bar{s}_B$ if (i) $i > 2$ or (ii) $i = 2$ and ($d \notin H_B^{\mathcal{I}}$ or $v_1 \neq r_B$). Now let

$$
\begin{aligned}
A^{\mathcal{J}} &= A^{\mathcal{I}} \text{ for all } A \in \mathsf{N_C} \cap \Sigma \\
B^{\mathcal{J}} &= \{d \in \mathsf{Ind}(\mathcal{A}) \mid B(d) \in \mathcal{A}\} \text{ for all } B \in \mathsf{N_C} \setminus \Sigma \\
Z_B^{\mathcal{J}} &= Z_B^{\mathcal{I}} \cup \{w \mid \mathsf{tail}(w) = s_B\} \text{ for all } B \in \mathsf{N_C} \setminus \Sigma \\
r^{\mathcal{J}} &= r^{\mathcal{I}} \text{ for all } r \in \mathsf{N_C} \cap \Sigma \\
r_B^{\mathcal{J}} &= r_B^{\mathcal{I}} \cup \{(w, wr_B) \mid wr_B \in \Delta^{\mathcal{J}}\} \text{ for all } B \in \mathsf{N_C} \setminus \Sigma \\
s_B^{\mathcal{J}} &= s_B^{\mathcal{I}} \cup \{(w, ws_B) \mid wr_B \in \Delta^{\mathcal{J}}\} \cup \{(w, w\bar{s}_B) \mid w\bar{s}_B \in \Delta^{\mathcal{J}}\} \text{ for all } B \in \mathsf{N_C} \setminus \Sigma
\end{aligned}
$$

It follows directly from the construction of $\mathcal{J}$ that $H_B^{\mathcal{J}} = B^{\mathcal{I}}$, for all $B \in \mathsf{N_C} \setminus \Sigma$. Thus, for all concepts D (not using fresh symbols from Σ-abstractions) and their Σ-abstractions D' and all $d \in \Delta^{\mathcal{I}}$: $d \in D'^{\mathcal{J}}$ iff $d \in D^{\mathcal{I}}$. Thus, the CIs of $\mathcal{T}'$ hold in all $d \in \Delta^{\mathcal{I}}$ since the CIs of $\mathcal{T}$ hold in all $d \in \Delta^{\mathcal{I}}$. The CIs of $\mathcal{T}'$ also hold in all $d \in \Delta^{\mathcal{J}} \setminus \Delta^{\mathcal{I}}$ since $\mathcal{T}$ admits trivial models. Thus, $\mathcal{J}$ is a model of $\mathcal{T}'$. Since $\mathcal{J}$ is a model of $\mathcal{T}^{\exists}$ by construction, it follows that $\mathcal{J}$ is a model of $\mathcal{T}' \cup \mathcal{T}^{\exists}$.

(2) can be proved using the models constructed in the proof of (1).

(3) is a consequence of the fact that $\mathcal{T}^{\exists}$ can be viewed as a TBox formulated in the description logic DL-Lite$_{\mathcal{R}}$ and that any OMQ $(\mathcal{T}, q)$ with $\mathcal{T}$ a DL-Lite$_{\mathcal{R}}$-TBox and q a PEQ is known to be rewritable into a union of CQs [CDGL$^+$07, ACKZ09].

(4) Assume that $\mathcal{A}|_{\Sigma}$ is consistent w.r.t. $\mathcal{T}$ and that $\mathcal{T}^{\exists}, \mathcal{A} \not\models q(\vec{a})$. We show $\mathcal{T}' \cup \mathcal{T}^{\exists}, \mathcal{A} \not\models q(\vec{a})$. Note first that one can construct a hom-initial model $\mathcal{I}_{\mathcal{A}}^{\exists}$ of $\mathcal{T}^{\exists}$ and $\mathcal{A}$ in the same way as $\mathcal{J}$ was constructed from $\mathcal{I}$ in the proof of Point (2) (by replacing $\mathcal{I}$ with the interpretation $\mathcal{I}_{\mathcal{A}}$ corresponding to $\mathcal{A}$ and not using the symbols $\bar{s}_B$ in the construction). Thus, $\Delta^{\mathcal{I}_{\mathcal{A}}^{\exists}}$ is the set of words $w = av_1 \cdots v_n$ such that $a \in \mathsf{Ind}(\mathcal{A})$ and $v_i \in \{r_B, s_B, \mid B \in \mathsf{N_C} \setminus \Sigma\}$. We have $\mathcal{I}_{\mathcal{A}}^{\exists} \not\models q(\vec{a})$. Now, as $\mathcal{T}$ is Σ-extensional, there is a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}|_{\Sigma}$ with $\Delta^{\mathcal{I}} = \mathsf{Ind}(\mathcal{A})$ and $A^{\mathcal{I}} = \{a \mid A(a) \in \mathcal{A}\}$ for all $A \in \Sigma$, and $r^{\mathcal{I}} = \{(a, b) \mid r(a, b) \in \mathcal{A}\}$ for all role names $r \in \Sigma$. Construct the model $\mathcal{J}$ of $\mathcal{T}' \cup \mathcal{T}^{\exists}$ and $\mathcal{A}$ in the same way as in the proof of Point (2). Define a mapping $h : \mathcal{J} \to \mathcal{I}_{\mathcal{A}}^{\exists}$ by setting $h(w) = w'$, where w' is obtained from w by replacing every $\bar{s}_B$ by s_B. One can show that h is a homomorphism. Thus $\mathcal{J} \not\models q(\vec{a})$ and so $\mathcal{T}' \cup \mathcal{T}^{\exists}, \mathcal{A} \not\models q(\vec{a})$, as required. The converse direction is trivial. ❏

We are now ready to prove Theorem 33.

Proof of Theorem 33. Assume a Σ-template $\mathcal{B}$ is given. We construct the TBox $\mathcal{T}_{\mathcal{B}}$ in two steps. First take for any $d \in \mathsf{Ind}(\mathcal{B})$ a concept name A_d and define a TBox $\mathcal{H}_{\mathcal{B}}$ with the following CIs:

$$
\begin{aligned}
\mathsf{dom} &\sqsubseteq \bigsqcup_{d \in \mathsf{Ind}(\mathcal{B})} A_d \\
A_d \sqcap A_e &\sqsubseteq \bot \qquad \text{for all } d, e \in \mathsf{Ind}(\mathcal{B}), d \neq e \\
A_d \sqcap \exists r.A_e &\sqsubseteq \bot \qquad \text{for all } d, e \in \mathsf{Ind}(\mathcal{B}), r \in \Sigma, r(d, e) \notin \mathcal{B} \\
A_d \sqcap A &\sqsubseteq \bot \qquad \text{for all } d \in \mathsf{Ind}(\mathcal{B}), A \in \Sigma, A(d) \notin \mathcal{B}.
\end{aligned}
$$

Here $\mathsf{dom} \sqsubseteq \bigsqcup_{d \in \mathsf{Ind}(\mathcal{B})} A_d$ stands for the set of CIs

$$
\exists r.\top \sqsubseteq \bigsqcup_{d \in \mathsf{Ind}(\mathcal{B})} A_d, \quad A \sqsubseteq \bigsqcup_{d \in \mathsf{Ind}(\mathcal{B})} A_d, \quad \top \sqsubseteq \forall r.(\bigsqcup_{d \in \mathsf{Ind}(\mathcal{B})} A_d)
$$

where r and A range over all role and concept names in Σ, respectively. We use a CI of the form $\mathsf{dom} \sqsubseteq C$ rather than $\top \sqsubseteq C$ to ensure that the TBox $\mathcal{H}_{\mathcal{B}}$ admits trivial models. It should also

be clear that $\mathcal{T}$ is Σ-extensional. Now let M be a fresh concept name. Then the following can be proved in a straightforward way.

Claim 1. For any ABox $\mathcal{A}$ the following conditions are equivalent:

(1) $\mathcal{H}_\mathcal{B}, \mathcal{A}|_\Sigma \not\models \exists x\, M(x)$;
(2) $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{H}_\mathcal{B}$;
(3) $\mathcal{A}|_\Sigma \rightarrow \mathcal{B}$.

Thus, $\mathsf{CSP}(\mathcal{B})$ and the complement of the query evaluation problem for $(\mathcal{H}_\mathcal{B}, \exists x\, M(x))$ are reducible to each other in polynomial time. Because of the disjunctions, however, the query evaluation problem w.r.t $\mathcal{H}_\mathcal{B}$ is typically coNP-hard even if $\mathsf{CSP}(\mathcal{B})$ is in PTIME.

In the second step, we thus 'hide' the concept names A_d by replacing them with their abstractions H_{A_d}. Let $\mathcal{H}'_\mathcal{B} \cup \mathcal{T}^\exists$ be the enriched Σ-abstraction of $\mathcal{H}_\mathcal{B}$. From Claim 1 and Lemma 35 (1) according to which $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{H}_\mathcal{B}$ iff $\mathcal{A}$ is consistent w.r.t. $\mathcal{H}'_\mathcal{B} \cup \mathcal{T}^\exists$, we obtain

Claim 2. For any ABox $\mathcal{A}$ not containing the concept name M, the following conditions are equivalent:

(1) $\mathcal{H}'_\mathcal{B} \cup \mathcal{T}^\exists, \mathcal{A} \not\models \exists x\, M(x)$;
(2) $\mathcal{A}$ is consistent w.r.t. $\mathcal{H}'_\mathcal{B} \cup \mathcal{T}^\exists$;
(3) $\mathcal{A}|_\Sigma \rightarrow \mathcal{B}$.

Let $\mathcal{T}_\mathcal{B} = \mathcal{H}'_\mathcal{B} \cup \mathcal{T}^\exists$ be the enriched Σ-abstraction of $\mathcal{H}_\mathcal{B}$. We show that $\mathcal{T}_\mathcal{B}$ is as required to prove Theorem 33. The theorem comprises two points:

(1) We have to show that $\mathsf{CSP}(\mathcal{B})$ is equivalent to the complement of the OMQ $(\mathcal{T}_\mathcal{B}, \exists x\, M(x))$. This is an immediate consequence of Claim 2.

(2) For the converse reduction, let q be a PEQ. We have to show that the query evaluation problem for $(\mathcal{T}_\mathcal{B}, q)$ is reducible in polynomial time to the complement of $\mathsf{CSP}(\mathcal{B})$. Let $\mathcal{A}$ be an ABox and $\vec{a}$ from $\mathsf{Ind}(\mathcal{A})$. We show that the following are equivalent:

(a) $\mathcal{T}_\mathcal{B}, \mathcal{A} \models q(\vec{a})$;
(b) $\mathcal{A}|_\Sigma \not\rightarrow \mathcal{B}$ or $\mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$.

Regarding (b), we remark that checking whether $\mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$ can be part of the reduction since, by Lemma 35 (3), it needs only polynomial time. First assume that (a) holds. If $\mathcal{A}|_\Sigma \rightarrow \mathcal{B}$, then by Claim 1 the ABox $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{H}_\mathcal{B}$. By Lemma 35 (4), we obtain $\mathcal{T}_\mathcal{B}, \mathcal{A} \models q(\vec{a})$ iff $\mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$ for all PEQs q and all $\vec{a}$, as required.

Conversely, assume (b) holds. If $\mathcal{A}|_\Sigma \not\rightarrow \mathcal{B}$, then by Claim 2 $\mathcal{A}$ is not consistent w.r.t. $\mathcal{T}_\mathcal{B}$ and so $\mathcal{T}_\mathcal{B}, \mathcal{A} \models q(\vec{a})$. If $\mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$, then $\mathcal{T}_\mathcal{B}, \mathcal{A} \models q(\vec{a})$ since $\mathcal{T}^\exists \subseteq \mathcal{T}_\mathcal{B}$.   ❏

We close this section by illustrating an example consequence of Theorem 33. It was proved in [FV98] that there are CSPs that are in PTIME yet not rewritable into Datalog, and in fact also not into Datalog$^{\neq}$ due to the results in [FV03]. This was strengthened to CSPs that contain no relations of arity larger than two in [Ats08]. It was also observed in [FV98] that there are CSPs that are rewritable into Datalog, but not into monadic Datalog (such as the CSP expressing 2-colorability). This again extends to Datalog$^{\neq}$ and applies to CSPs with relations of arity at most two. With this in mind, the following is a consequence of Theorems 33 and 19.

Theorem 36.

(1) There are $\mathcal{ALC}$-TBoxes $\mathcal{T}$ such that PEQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME, but $\mathcal{T}$ is not Datalog-rewritable for ELIQs;

(2) *there are $\mathcal{ALC}$-TBoxes that are Datalog-rewritable for PEQs, but not monadic Datalog-rewritable for ELIQs.*

7. Non-Dichotomy and Undecidability in $\mathcal{ALCF}$

We show that the complexity landscape of query evaluation w.r.t. $\mathcal{ALCF}$-TBoxes is much richer than for $\mathcal{ALCI}$, and in fact too rich to be fully manageable. In particular, we prove that for CQ-evaluation, there is no dichotomy between PTIME and CONP (unless PTIME = NP). We also establish that materializability, (monadic) Datalog$^{\neq}$-rewritability, PTIME query evaluation, and CONP-hardness of query evaluation are undecidable. We start with the undecidability proofs, which are by reduction of an undecidable rectangle tiling problem and reuse the 'hidden concepts' introduced in the previous section. Next, the TBox from that reduction is adapted to prove the non-dichotomy result by an encoding of the computations of nondeterministic polynomial time Turing machines (again using hidden concepts). The basis for the technical development in this section is a TBox constructed in [BBLW16] to prove the undecidability of *query emptiness* in $\mathcal{ALCF}$.

An instance of the finite rectangle tiling problem is given by a triple $\mathfrak{P} = (\mathfrak{T}, H, V)$ with $\mathfrak{T}$ a finite set of *tile types* including an *initial tile* T_{init} to be placed on the lower left corner and a *final tile* T_{final} to be placed on the upper right corner, $H \subseteq \mathfrak{T} \times \mathfrak{T}$ a *horizontal matching relation*, and $V \subseteq \mathfrak{T} \times \mathfrak{T}$ a *vertical matching relation*. A *tiling* for $(\mathfrak{T}, H, V)$ is a map $f : \{0, \ldots, n\} \times \{0, \ldots, m\} \to \mathfrak{T}$ such that $n, m \geq 0$, $f(0,0) = T_{\mathsf{init}}$, $f(n, m) = T_{\mathsf{final}}$, $(f(i,j), f(i+1, j)) \in H$ for all $i < n$, and $(f(i,j), f(i, j+1)) \in V$ for all $i < m$. We say that $\mathfrak{P}$ *admits a tiling* if there exists a map f that is a tiling for $\mathfrak{P}$. It is undecidable whether an instance of the finite rectangle tiling problem admits a tiling.

Now let $\mathfrak{P} = (\mathfrak{T}, H, V)$ be a finite rectangle tiling problem with $\mathfrak{T} = \{T_1, \ldots, T_p\}$. We regard the tile types in $\mathfrak{T}$ as concept names and set $\Sigma_g = \{T_1, \ldots, T_p, x, y, \hat{x}, \hat{y}\}$, where x, y, $\hat{x}$, and $\hat{y}$ are *functional* role names. The TBox $\mathcal{T}_{\mathfrak{P}}$ is defined as the following set of CIs, where (T_i, T_j, T_ℓ) range over all triples from $\mathfrak{T}$ such that $(T_i, T_j) \in H$ and $(T_i, T_\ell) \in V$ and where for $e \in \{c, x, y\}$ the concept B_e ranges over all conjunctions $L_1 \sqcap L_2$ with $L_i \in \{Z_{e,i}, \neg Z_{e,i}\}$, for concept names $Z_{e,i}$ $(i = 1, 2)$:

$$T_{\mathsf{final}} \sqsubseteq Y \sqcap U \sqcap R$$

$$\exists x.(U \sqcap Y \sqcap T_j) \sqcap I_x \sqcap T_i \sqsubseteq U \sqcap Y$$

$$\exists y.(R \sqcap Y \sqcap T_\ell) \sqcap I_y \sqcap T_i \sqsubseteq R \sqcap Y$$

$$\exists x.(T_j \sqcap Y \sqcap \exists y.Y) \sqcap \exists y.(T_\ell \sqcap Y \sqcap \exists x.Y) \sqcap I_x \sqcap I_y \sqcap C \sqcap T_i \sqsubseteq Y$$

$$Y \sqcap T_{\mathsf{init}} \sqsubseteq A$$

$$B_x \sqcap \exists x.\exists \hat{x}.B_x \sqsubseteq I_x$$

$$B_y \sqcap \exists y.\exists \hat{y}.B_y \sqsubseteq I_y$$

$$\exists x.\exists y.B_c \sqcap \exists y.\exists x.B_c \sqsubseteq C$$

$$\bigsqcup_{1 \leq s < t \leq p} T_s \sqcap T_t \sqsubseteq \bot$$

$$U \sqsubseteq \forall y.\bot \quad R \sqsubseteq \forall x.\bot \quad U \sqsubseteq \forall x.U \quad R \sqsubseteq \forall y.R$$

$$Y \sqcap T_{\mathsf{init}} \sqsubseteq D \sqcap L \quad D \sqsubseteq \forall \hat{y}.\bot \quad L \sqsubseteq \forall \hat{x}.\bot \quad D \sqsubseteq \forall x.D \sqcap \forall \hat{x}.D \quad L \sqsubseteq \forall y.L \sqcap \forall \hat{y}.L$$

With the exception of the CIs in the last line, the TBox $\mathcal{T}_{\mathfrak{P}}$ has been defined and analyzed in [BBLW16]. Here, we briefly give intuition and discuss its main properties. The role names x and y are used to represent horizontal and vertical adjacency of points in a rectangle. The role

names $\hat{x}$ and $\hat{y}$ are used to simulate the inverses of x and y. The concept names in $\mathcal{T}_{\mathfrak{P}}$ serve the following puroposes:

- U, R, L, and D mark the upper, right, left, and lower ('down') border of the rectangle.
- In the B_c concepts, the concept names $Z_{c,1}$ and $Z_{c,2}$ serve as second-order variables and ensure that a flag C is set at positions where the grid cell is closed.
- In the concepts B_x and B_y, the concept names $Z_{x,1}, Z_{x,2}, Z_{y,1}, Z_{y,2}$ also serve as second-order variables and ensure that flags I_x and I_y are set at positions where x and $\hat{x}$ as well as y and $\hat{y}$ are inverse to each other.
- The concept name Y is propagated through the grid from the upper right corner to the lower left one, ensuring that these flags are set everywhere, that every position of the grid is labeled with at least one tile type, and that the horizontal and vertical matching conditions are satisfied.
- Finally, when the lower left corner of the grid is reached, the concept name A is set as a flag.

Because of the use of the concepts B_e, CQ evaluation w.r.t. $\mathcal{T}_{\mathfrak{P}}$ is coNP-hard: we leave it as an exercise to the reader to verify that $\mathcal{T}_{\mathfrak{P}}$ does not have the ABox disjunction property. $\mathcal{T}_{\mathfrak{P}}$ without the three CIs involving the concepts B_e, however, is (equivalent to) a Horn-$\mathcal{ALCF}$ TBox and thus enjoys PTIME CQ-evaluation. Call a Σ_g-ABox $\mathcal{A}$ an *grid ABox (with initial individual a)* if $\mathcal{A}$ represents a grid along with a proper tiling for $\mathfrak{P}$. In detail, we require that there is a tiling f for $\mathfrak{P}$ with domain $\{0, \ldots, n\} \times \{0, \ldots, m\}$ and a bijection $g : \{0, \ldots, n\} \times \{0, \ldots, m\} \to \mathsf{Ind}(\mathcal{A})$ with $g(0,0) = a$ such that

- for all $j < n$, $k \leq m$: $T(g(j,k)) \in \mathcal{A}$ iff $T = f(j,k)$;
- for all $b_1, b_2 \in \mathsf{Ind}(\mathcal{A})$: $x(b_1, b_2) \in \mathcal{A}$ iff $\hat{x}(b_2, b_1) \in \mathcal{A}$ iff there are $j < n$, $k \leq m$ such that $(b_1, b_2) = (g(j,k), g(j+1,k))$;
- for all $b_1, b_2 \in \mathsf{Ind}(\mathcal{A})$: $y(b_1, b_2) \in \mathcal{A}$ iff $\hat{y}(b_2, b_1) \in \mathcal{A}$ iff there are $j \leq n$, $k < m$ such that $(b_1, b_2) = (g(j,k), g(j,k+1))$.

Clearly, if $\mathfrak{P}$ admits a tiling then a grid ABox exists and for any grid ABox $\mathcal{A}$, $\mathcal{T}_{\mathfrak{P}}, \mathcal{A} \models A(a)$ for the (uniquely determined) initial individual a of $\mathcal{A}$. The following summarizes relevant properties of Σ_g-ABoxes that follow almost directly from the analysis of $\mathcal{T}_{\mathfrak{P}}$ in [BBLW16]. We say that an ABox $\mathcal{A}$ *contains* an ABox $\mathcal{A}'$ if $\mathcal{A}' \subseteq \mathcal{A}$ and that $\mathcal{A}$ *contains a closed ABox* $\mathcal{A}$ if, additionally, $r(a,b) \in \mathcal{A}$ and $a \in \mathsf{Ind}(\mathcal{A}')$ implies $r(a,b) \in \mathcal{A}'$ for $r \in \{x, y, \hat{x}, \hat{y}\}$. Moreover, we say that *inconsistency of (Σ-)ABoxes w.r.t. a TBox $\mathcal{T}$ is monadic Datalog$^{\neq}$-rewritable* if there is a Boolean monadic Datalog$^{\neq}$-program Π such that for any (Σ-)ABox $\mathcal{A}$, $\mathcal{A} \models \Pi()$ iff $\mathcal{A}$ is inconsistent w.r.t. $\mathcal{T}$.

Lemma 37. *Let $\mathfrak{P}$ be a finite rectangle tiling problem. Then the following holds.*

(1) $\mathcal{T}_{\mathfrak{P}}$ admits trivial models and is Σ_g-extensional.

(2) Inconsistency of Σ_g-ABoxes w.r.t. $\mathcal{T}_{\mathfrak{P}}$ is monadic Datalog$^{\neq}$-rewritable.

(3) If a Σ_g-ABox $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}_{\mathfrak{P}}$, then $\mathcal{A}$ contains

- *closed grid ABoxes $\mathcal{A}_1, \ldots, \mathcal{A}_n$, $n \geq 0$, with mutually disjoint sets $\mathsf{Ind}(\mathcal{A}_i)$ and*
- *a (possibly empty) Σ_g-ABox $\mathcal{A}'$ disjoint from $\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n$*

such that $\mathcal{A} = \mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n \cup \mathcal{A}'$ and $\mathcal{T}_{\mathfrak{P}}, \mathcal{A} \models A(a)$ iff a is the initial node of some $\mathcal{A}_i$. Moreover, there is a model $\mathcal{I}$ of $\mathcal{A}$ witnessing Σ_g-extensionality of $\mathcal{T}_{\mathfrak{P}}$ that satisfies $a \in A^{\mathcal{I}}$ iff a is the initial node of some $\mathcal{A}_i$.

Proof. (1) is a straightforward consequence of the definition of $\mathcal{T}_{\mathfrak{P}}$. (2) Assume a Σ_g-ABox $\mathcal{A}$ is given. Apply the following rules exhaustively to $\mathcal{A}$:

(a) add $I_x(a)$ to $\mathcal{A}$ if there exists b with $x(a,b), \hat{x}(b,a) \in \mathcal{A}$;

(b) add $I_y(a)$ to $\mathcal{A}$ if there exists b with $y(a,b), \hat{y}(b,a) \in \mathcal{A}$;

(c) add $C(a)$ to $\mathcal{A}$ if there exist a_1, a_2, b with $x(a,a_1), y(a,a_2), y(a_1,b), x(a_2,b) \in \mathcal{A}$.

Denote the resulting ABox by $\mathcal{A}^\dagger$. Now remove the three CIs involving the concepts B_e from $\mathcal{T}_\mathfrak{P}$ and denote by $\mathcal{T}_\mathfrak{P}^\dagger$ the resulting TBox. Using the analysis of the CIs involving the concepts B_e in [BBLW16], one can show that $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}_\mathfrak{P}$ iff $\mathcal{A}^\dagger$ is consistent w.r.t. $\mathcal{T}_\mathfrak{P}^\dagger$. Since the latter is a Horn-$\mathcal{ALCF}$-TBox, it is unraveling tolerant and one can build a monadic Datalog$^{\neq}$-rewriting of the inconsistency of Σ_g-ABoxes w.r.t. $\mathcal{T}_\mathfrak{P}^\dagger$, essentially as in the proof of Theorem 25. Finally, the obtained program can be modified so as to behave as if started on $\mathcal{A}^\dagger$ when started on $\mathcal{A}$, by implementing rules (a) to (c) as monadic Datalog rules.

(3) This is almost a direct consequence of the properties established in [BBLW16]. In particular, one finds the desired model $\mathcal{I}$ from the 'moreover' part by applying the three rules from the proof of Point (2) and then applying the CIs in $\mathcal{T}_\mathfrak{P}^\dagger$ as rules. The only condition on the decomposition of the ABox $\mathcal{A}$ into $\mathcal{A} = \mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n \cup \mathcal{A}'$ that does not follow from [BBLW16] is that the containment of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ in $\mathcal{A}$ is closed also for the role names $\hat{x}$ and $\hat{y}$. To ensure this condition, we use the CIs that mention L and D that were not present in the TBox used in that paper. In fact, the following two properties follow directly from these CIs: (i) the individuals c reachable along an x-path in $\mathcal{A}$ from some a with $\mathcal{T}_\mathfrak{P}, \mathcal{A} \models A(a)$ all satisfy $\mathcal{T}_\mathfrak{P}, \mathcal{A} \models D(c)$ and so do not have an $\hat{x}$-successor; and (ii) the individuals c reachable along a y-path in $\mathcal{A}$ from some a with $\mathcal{T}_\mathfrak{P}, \mathcal{A} \models A(a)$ all satisfy $\mathcal{T}_\mathfrak{P}, \mathcal{A} \models L(c)$ and so do not have a $\hat{y}$-successor. (i) and (ii) together with the properties established in [BBLW16] entail that the containment of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ in $\mathcal{A}$ is closed also for the role names $\hat{x}$ and $\hat{y}$, as required. ❑

Note that it follows from Lemma 37 (3) that if $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$, then the sequence $\mathcal{A}_1, \ldots, \mathcal{A}_n$ is empty iff $\mathcal{T}_\mathfrak{P}, \mathcal{A} \not\models \exists x\, A(x)$ (and $\mathcal{A}'$ is non-empty since ABoxes are non-empty). In particular this must be the case when $\mathfrak{P}$ does not admit a tiling. In the proof Lemma 38 below, this is actually all we need from Lemma 37 (3). In full generality, it will only be used in the proof of non-dichotomy later on. We also remark that the decomposition $\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n \cup \mathcal{A}'$ of $\mathcal{A}$ in Lemma 37 (3) is unique.

Let $\mathcal{T} = \mathcal{T}_\mathfrak{P} \cup \{A \sqsubseteq B_1 \sqcup B_2\}$, where B_1 and B_2 are fresh concept names. Set $\Sigma = \Sigma_g \cup \{B_1, B_2\}$ and let $\mathcal{T}' \cup \mathcal{T}^\exists$ be the enriched Σ-abstraction of $\mathcal{T}$.

Lemma 38.

   (1) *If $\mathfrak{P}$ admits a tiling, then CQ-evaluation w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$ is CONP-hard and $\mathcal{T}' \cup \mathcal{T}^\exists$ is not materializable.*

   (2) *If $\mathfrak{P}$ does not admit a tiling, then CQ-evaluation w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$ is monadic Datalog$^{\neq}$-rewritable and $\mathcal{T}' \cup \mathcal{T}^\exists$ is materializable.*

Proof. (1) If $\mathfrak{P}$ admits a tiling, then there is a grid ABox $\mathcal{A}$ with initial node a. $\mathcal{A}$ uses symbols from Σ_g, only. We have $\mathcal{T}_\mathfrak{P}, \mathcal{A} \models A(a)$ and $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}_\mathfrak{P}$. By Lemma 35 (2), $\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models A'(a)$ where A' is the Σ-abstraction of A. Since $\mathcal{T}'$ contains $A' \sqsubseteq B_1 \sqcup B_2$ and B_1 and B_2 do not occur elsewhere in $\mathcal{T}' \cup \mathcal{T}^\exists$, it is clear that $\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models B_1(a) \vee B_2(a)$ but $\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \not\models B_i(a)$ for $i = 1, 2$. Thus $\mathcal{T}' \cup \mathcal{T}^\exists$ does not have the ABox disjunction property. It follows that $\mathcal{T}' \cup \mathcal{T}^\exists$ is not materializable and that CQ-evaluation w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$ is CONP-hard.

(2) Assume that $\mathfrak{P}$ does not admit a tiling. Let q be a PEQ. We show how to construct a monadic Datalog$^{\neq}$-rewriting Π of $(\mathcal{T}' \cup \mathcal{T}^\exists, q)$. On ABoxes $\mathcal{A}$ that are inconsistent w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$, Π is supposed to return all tuples over $\mathsf{Ind}(\mathcal{A})$ of the same arity as q. By Lemma 35 (1), an ABox $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$ iff $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{T}$. It thus follows from Lemma 37 (2) that

inconsistency of ABoxes w.r.t. $\mathcal{T}' \cup \mathcal{T}^{\exists}$ is monadic Datalog$^{\neq}$-rewritable. From a concrete rewriting, we can build a monadic Datalog$^{\neq}$-program Π_0 that checks inconsistency and, if successful, returns all answers.

Now for ABoxes $\mathcal{A}$ that are consistent w.r.t. $\mathcal{T}' \cup \mathcal{T}^{\exists}$. By Lemma 35 (1), $\mathcal{A}|_{\Sigma}$ is consistent w.r.t. $\mathcal{T}_{\mathfrak{P}}$. Since $\mathfrak{P}$ does not admit a tiling and by Lemma 37 (2), $\mathcal{T}_{\mathfrak{P}}, \mathcal{A}|_{\Sigma} \not\models \exists x\, A(x)$. Thus, by Lemma 37 (1) we find a model $\mathcal{I}$ of $\mathcal{T}_{\mathfrak{P}}$ and $\mathcal{A}|_{\Sigma}$ such that $\Delta^{\mathcal{I}} = \mathsf{Ind}(\mathcal{A})$, $B^{\mathcal{I}} = \{a \mid B(a) \in \mathcal{A}\}$ for all $B \in \Sigma$, $r^{\mathcal{I}} = \{(a, b) \mid r(a, b) \in \mathcal{A}\}$ for all $r \in \Sigma$, and $A^{\mathcal{I}} = \emptyset$. Since $A^{\mathcal{I}} = \emptyset$, $\mathcal{I}$ is a model of $\mathcal{T}$. From Lemma 35 (4), we thus obtain that $\mathcal{T}' \cup \mathcal{T}^{\exists}, \mathcal{A} \models q(\vec{a})$ iff $\mathcal{T}^{\exists}, \mathcal{A} \models q(\vec{a})$, for all $\vec{a}$. By Lemma 35 (3), $(\mathcal{T}^{\exists}, q)$ is monadic Datalog$^{\neq}$-rewritable into a program Π_1.

The desired program Π is simply the union of Π_0 and Π_1, assuming disjointness of IDB relations. ❑

Lemma 38 implies the announced undecidability results.

Theorem 39. *For $\mathcal{ALCF}$-TBoxes $\mathcal{T}$, the following problems are undecidable (Points 1 to 4 are subject to the side condition that* PTIME $\neq$ NP*):*

(1) CQ-evaluation w.r.t. $\mathcal{T}$ is in PTIME*;*
(2) CQ-evaluation w.r.t. $\mathcal{T}$ is CONP*-hard;*
(3) $\mathcal{T}$ is monadic Datalog$^{\neq}$-rewritable;
(4) $\mathcal{T}$ is Datalog$^{\neq}$-rewritable;
(5) $\mathcal{T}$ is materializable.

We now come to the proof of non-dichotomy.

Theorem 40 (Non-Dichotomy)**.** *For every language $L \in$ CONP, there exists an $\mathcal{ALCF}$-TBox $\mathcal{T}$ such that, for a distinguished concept name M_0, the following holds:*

(1) L is polynomial time reducible to the evaluation of $(\mathcal{T}, \exists x\, M_0(x))$;
(2) the evaluation of $(\mathcal{T}, q)$ is polynomial time reducible to L, for all PEQs q.

To prove Theorem 40 let $L \in$ CONP. Take a non-deterministic polynomial time Turing Machine M that recognizes the complement of L. Let $M = (Q, \Gamma_0, \Gamma_1, \Delta, q_0, q_a, q_r)$ with Q a finite set of states, Γ_0 and Γ_1 finite input and tape alphabets such that $\Gamma_0 \subseteq \Gamma_1$ and $\Gamma_1 \setminus \Gamma_0$ contains the blank symbol β, $q_0 \in Q$ the starting state, $\Delta \subseteq Q \times \Gamma_1 \times Q \times \Gamma_1 \times \{L, R\}$ the transition relation, and $q_a, q_r \in Q$ the accepting and rejecting states. Denote by $L(M)$ the language recognized by M. We can assume w.l.o.g. that there is a fixed symbol $\gamma_0 \in \Gamma_0$ such that all words accepted by M are of the form $\gamma_0 v$ with $v \in (\Gamma_0 \setminus \{\gamma_0\})^*$; in fact, it is easy to modify M to satisfy this property without changing the complexity of its word problem. We also assume that for any input $v \in \Gamma_0^*$, M uses exactly $|v|^{k_1}$ cells for the computation, halts after exactly $|v|^{k_2}$ steps in the accepting or rejecting state, and does not move to the left of the starting cell.

To represent inputs to M and to provide the space for simulating computations, we use grid ABoxes as in the proof of Theorem 39, where the tiling of the bottom row represents the input word followed by blank symbols. As the set of tile types, we use $\mathfrak{T} = \Gamma_0 \cup \{\beta, T, T_{\mathsf{final}}\}$ where T is a 'default tile' that labels every position except those in the bottom row and the upper right corner. Identify T_{init} with γ_0 and let $\Sigma_g = \Gamma_0 \cup \{\beta, T, T_{\mathsf{final}}\} \cup \{x, y, \hat{x}, \hat{y}\}$. Consider the TBox $\mathcal{T}_{\mathfrak{P}_M}$ defined above, for $\mathfrak{P}_M = (\mathfrak{T}, H, V)$ with

$$H = \{(\gamma_0, \gamma), (\gamma, \gamma'), (\gamma, \beta), (\beta, \beta), (T, T), (T, T_{\mathsf{final}}) \mid \gamma, \gamma' \in \Gamma_0 \setminus \{\gamma_0\}\}$$
$$V = \{(\gamma, T), (\beta, T), (T, T), (T, T_{\mathsf{final}}), (\gamma, T_{\mathsf{final}}), (\beta, T_{\mathsf{final}}) \mid \gamma \in \Gamma_0\}.$$

Recall that $\mathcal{T}_{\mathfrak{P}_M}$ checks whether a given Σ_g-ABox contains a grid structure with a tiling that respects H, V, T_{init}, and T_{final}, and derives the concept name A at the lower left corner of such grids. We

now construct a TBox $\mathcal{T}_M$ that, after the verification has finished, initiates a computation of M on the grid. In addition to the concept names in $\mathcal{T}_{\mathfrak{P}_M}$, $\mathcal{T}_M$ uses concept names A_γ and $A_{q,\gamma}$ for all $\gamma \in \Gamma_1$ and $q \in Q$ to represent symbols written during the computation (in contrast to the elements of Γ_1 as concept names, used to encode the input word) and to represent the state and head position. In detail, $\mathcal{T}_M$ contains the following CIs:

- When the verification of the grid has finished, A floods the ABox:
$$A \sqsubseteq \forall r.A \quad \text{for all } r \in \{x, y, \hat{x}, \hat{y}\}.$$

- The initial configuration is as required:
$$\gamma_0 \sqcap A \sqsubseteq A_{q_0,\gamma} \qquad \gamma \sqcap A \sqsubseteq A_\gamma \quad \text{for all } \gamma \in (\Gamma_0 \cup \{\beta\}) \setminus \{\gamma_0\}.$$

- For every $(q, \gamma) \in Q \times \Gamma_1$, the transition relation of M is satisfied:
$$A_{q,\gamma} \sqcap A \quad \sqsubseteq \quad \bigsqcup_{(q,\gamma,q',\gamma',L)\in\Delta} \exists y.(A_{\gamma'} \sqcap \bigsqcup_{\gamma''\in\Gamma_1} \exists \hat{x}.A_{q',\gamma''}) \sqcup$$
$$\bigsqcup_{(q,\gamma,q',\gamma',R)\in\Delta} \exists y.(A_{\gamma'} \sqcap \bigsqcup_{\gamma''\in\Gamma_1} \exists x.A_{q',\gamma''}).$$

- The representations provided by $A_{q,\gamma}$ and A_γ for symbols in Γ_1 coincide:
$$A_{q,\gamma} \sqcap A_{\gamma'} \sqsubseteq A_{q,\gamma'} \sqcap A_\gamma, \quad \text{for all } \gamma, \gamma' \in \Gamma_1$$

- The symbol written on a cell does not change if the head is not on the cell:
$$A_\gamma \sqcap A \sqsubseteq \forall y.A_\gamma \quad \text{for all } \gamma \in \Gamma_1$$

- The rejecting state is never reached:
$$A_{q_r,\gamma} \sqcap A \sqsubseteq \bot \quad \text{for all } \gamma \in \Gamma_1.$$

Let $\mathcal{T} = \mathcal{T}_{\mathfrak{P}_M} \cup \mathcal{T}_M$. We are going to show that an appropriate extended abstraction of $\mathcal{T}$ satisfies Conditions (1) and (2) of Theorem 40. We start with the following lemma which summarizes two important properties of $\mathcal{T}$.

Lemma 41.

(1) $\mathcal{T}$ admits trivial models and is Σ_g-extensional.
(2) For any Σ_g-ABox $\mathcal{A}$, $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$ iff the following two conditions hold:
  (a) $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}_{\mathfrak{P}_M}$;
  (b) let $\mathcal{A} = \mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n \cup \mathcal{A}'$ be the decomposition of $\mathcal{A}$ given in Lemma 37 (2) and assume that $\mathcal{A}_i$ is the $n_i \times m_i$-grid ABox with input v_i for $1 \leq i \leq n$. Then the following hold for $1 \leq i \leq n$:
    (i) $n_i \geq |v_i|^{k_1}$ and $m_i \geq |v_i|^{k_2}$ and
    (ii) $v_i \in L(M)$.

Proof. Since (1) follows directly from the construction of $\mathcal{T}$, we concentrate on (2). Assume first that $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$. Then $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}_{\mathfrak{P}_M}$ and so we can assume that there is a decomposition $\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n \cup \mathcal{A}'$ of $\mathcal{A}$ as in Lemma 37 (3). By definition, each $\mathcal{A}_i$ is an $n_i \times m_i$-grid ABox with input v_i. Since $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$, there is a model $\mathcal{I}$ of $\mathcal{A}$ and $\mathcal{T}$. By the first CIs of $\mathcal{T}_M$ and since the initial node a of each $\mathcal{A}_i$ must be in $A^\mathcal{I}$ by Lemma 37, $\mathsf{Ind}(\mathcal{A}_i) \subseteq A^\mathcal{I}$ for each i. Thus the restriction of $\mathcal{I}$ to $\mathsf{Ind}(\mathcal{A}_i)$ simulates an accepting computation of M starting with v_i. But since every computatation of M starting with a word of length n requires at least n^{k_1} space and m^{k_2} time and the containment of $\mathcal{A}_i$ in $\mathcal{A}$ is closed for the role names x and y, this is impossible if $n_i < |v_i|^{k_1}$ or $m_i < |v_i|^{k_2}$ and also if $v_i \notin L(M)$. Thus (i) and (ii) hold, as required.

For the converse direction, assume that (a) and (b) hold. Since $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}_{\mathfrak{P}_M}$, there is a decomposition $\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n \cup \mathcal{A}'$ of $\mathcal{A}$ as in Lemma 37 (3). Also by Lemma 37 (3), there is a model $\mathcal{I}$ of $\mathcal{A}$ that witnesses Σ_g-extensionality of $\mathcal{T}_{\mathfrak{P}_M}$ such that $a \in A^\mathcal{I}$ iff a is the initial node of some $\mathcal{A}_i$. We construct a model $\mathcal{I}'$ of $\mathcal{T}$ by modifying $\mathcal{I}$ as follows: with the exception of A, the symbols of $\mathcal{T}_{\mathfrak{P}_M}$ are interpreted in the same way as in $\mathcal{I}$ and thus $\mathcal{I}'$ is a model of $\mathcal{T}_{\mathfrak{P}_M}$. To satisfy the first CI of $\mathcal{T}_M$, we set $A^{\mathcal{I}'} = \mathsf{Ind}(\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n)$. Note that this suffices since the containment of each $\mathcal{A}_i$ in $\mathcal{A}$ is closed for the role names x and y. The remaining symbols from $\mathcal{T}_M$ are now interpreted in such a way that they describe on each $\mathcal{A}_i$ an accepting computation for v_i. This is possible since $v_i \in L(M)$, $n_i \geq |v_i|^{k_1}$ and $m_i \geq |v_i|^{k_2}$, and each computation of M starting with a word v of length n requires at most n^{k_1} space and m^{k_2} time. It can be verified that $\mathcal{I}'$ is a model of $\mathcal{T}_M$; note that since A is a conjunct of the left hand side of every CI in $\mathcal{T}_M$, the CIs in $\mathcal{T}_M$ are trivially satisfied in every node $d \in \Delta^\mathcal{I} \setminus \mathsf{Ind}(\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n)$. Thus $\mathcal{I}'$ satisfies $\mathcal{T}$ and $\mathcal{A}$ and we have proved consistency of $\mathcal{A}$ w.r.t. $\mathcal{T}$, as required.   ❑

We are now in the position to prove Theorem 40.

Proof of Theorem 40. Let $L \in \text{CONP}$ and let M and $\mathcal{T}$ be the TM and TBox from above. Set $\Sigma = \Sigma_g \cup \{M_0\}$ where M_0 is a fresh concept name and let $\mathcal{T}' \cup \mathcal{T}^\exists$ be the enriched Σ-abstraction of $\mathcal{T}$. We show that $\mathcal{T}$ satisfies Points (1) and (2) from Theorem 40.

(1) It suffices to give a polynomial time reduction from $L(M)$ to the complement of evaluating $(\mathcal{T}' \cup \mathcal{T}^\exists, \exists x\, M_0(x))$ (note that M_0 does not occur in any of the involved TBoxes). Assume that an input word v for M is given. If v is not from $\gamma_0(\Gamma_0 \setminus \{\gamma_0\})^*$, then reject. Otherwise, construct in polynomial time the $|v|^{k_1} \times |v|^{k_2}$-grid ABox $\mathcal{A}$ with input v. Observe that $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}_{\mathfrak{P}_M}$ and has the trivial decomposition $\mathcal{A} = \mathcal{A}_1$ in Lemma 37 (3). Thus Lemma 41 (2) implies that $v \in L(M)$ iff $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$. The latter condition is equivalent to $\mathcal{T}, \mathcal{A} \not\models \exists x\, M_0(x)$ since M_0 does not occur in $\mathcal{A}$ or $\mathcal{T}$. Since $\mathcal{T}$ admits trivial models, Lemma 35 (2) thus yields $v \in L(M)$ iff $\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \not\models \exists x\, M_0(x)$.

(2) We first make the following observation.

Claim 1. $\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$ iff

- $\mathcal{A}|_\Sigma$ is not consistent w.r.t. $\mathcal{T}$ or
- $\mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$.

For the 'only if' direction, observe that if $\mathcal{A}|_\Sigma$ is consistent w.r.t. $\mathcal{T}$, then by Lemma 35 (4), $\mathcal{T}' \cup \mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$ iff $\mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$. For the 'if' direction, observe that if $\mathcal{A}|_\Sigma$ is not consistent w.r.t. $\mathcal{T}$, then by Lemma 35 (1) $\mathcal{A}$ is not consistent w.r.t. $\mathcal{T}' \cup \mathcal{T}^\exists$. This finishes the proof of the claim.

By Lemma 35 (3), $\mathcal{T}^\exists, \mathcal{A} \models q(\vec{a})$ can be decided in polynomial time. Thus, Claim 1 implies that it suffices to give a polynomial time reduction of ABox consistency w.r.t. $\mathcal{T}$ to $L(M)$. But Lemma 41 (2) provides a polynomial reduction of ABox consistency w.r.t. $\mathcal{T}$ to $L(M)$ since

- Condition (a) of Lemma 41 (2) can be checked in polynomial time (by Lemma 37 (1));
- the decomposition of $\mathcal{A}$ in Condition (b) of Lemma 41 (2) as well as the words v_i, $1 \leq i \leq n$, can be computed in polynomial time;
- and Point (i) of Condition (b) can be checked in polynomial time.

It thus remains to check whether $v_i \in L(M)$ for $1 \leq i \leq n$. This finishes the proof of Theorem 40.   ❑

Theorems 40 and 19 imply that that there is no PTIME/CONP-dichotomy for query evaluation w.r.t. $\mathcal{ALCF}$-TBoxes, unless PTIME $=$ NP.

Observe that the TBoxes constructed in the undecidability and the non-dichotomy proof are both of depth four. This can be easily reduced to depth three: recall that the TBoxes of depth four are obtained from TBoxes $\mathcal{T}$ of depth two by taking their enriched Σ-abstractions. One can obtain a TBox of depth three (for which query evaluation has the same complexity up to polynomial time reductions) by first replacing in $\mathcal{T}$ compound concepts C in the scope of a single value or existential restriction by fresh concept names A_C and adding $A_C \equiv C$ to $\mathcal{T}$. Then the fresh concept names are added to the signature Σ and one constructs the enriched abstraction of the resulting TBox for the extended signature. This TBox is as required. Thus, our undecidability and non-dichotomy results hold for $\mathcal{ALCF}$-TBoxes of depth three already.

8. DISCUSSION

We have studied the complexity of query evaluation in the presence of an ontology formulated in a DL between $\mathcal{ALC}$ and $\mathcal{ALCFI}$, focussing on the boundary between PTIME and CONP. For $\mathcal{ALCFI}$-TBoxes of depth one, we have established a dichotomy between PTIME and CONP and shown that it can be precisely characterized in terms of unraveling tolerance and materializability. Moreover and unlike in the general case, PTIME complexity coincides with rewitability into Datalog$^{\neq}$. The case of higher or unrestricted depth is harder to analyze. We have shown that for arbitrary $\mathcal{ALC}$- and $\mathcal{ALCI}$-TBoxes there is a dichotomy between PTIME and CONP. The proof is by a reduction to the recently confirmed PTIME/NP-dichotomy for CSPs. For $\mathcal{ALCF}$ TBoxes of depth three we have shown that there is no dichotomy unless PTIME = NP and that deciding whether a given TBox admits PTIME query evaluation is undecidable, and so are related questions.

Several interesting research questions remain. We briefly discuss three possible directions.

(1) Is it decidable whether a given $\mathcal{ALC}$- or $\mathcal{ALCI}$-TBox admits PTIME query evaluation and, closely related, whether it is unraveling tolerant and whether it is materializable? First results for TBoxes of depth one have been obtained in [HLPW17a], but the general problem remains open. It is interesting to point out that unraveling tolerance is decidable for OMQs whose TBox is formulated in $\mathcal{ALCI}$ (where a concrete query is given, unlike in the case of unraveling tolerance of TBoxes); in that case, unraveling tolerance is equivalent to rewritability into monadic Datalog [FKL17]. It would also be interesting to study more general notions of unraveling tolerance based on unravelings into structures of bounded treewidth rather than into real trees.

(2) It would be interesting to study additional complexity classes such as LOGSPACE, NLOGSPACE, and AC^0. It is known that all these classes occur even for $\mathcal{ALC}$-TBoxes of depth one, see e.g. [CDL$^+$13] and the recent [LS17] which establishes a full complexity complexity classification of OMQs that are based on an $\mathcal{EL}$-TBox and an ELIQ. For example, CQ-evaluation w.r.t. the depth one $\mathcal{EL}$-TBox $\{\exists r.A \sqsubseteq A\}$, which encodes reachability in directed graphs, is NLOGSPACE-complete. It would thus be interesting to identify further dichotomies such as between NLOGSPACE and PTIME. We conjecture that for $\mathcal{ALCFI}$-TBoxes of depth one, it is decidable whether query evaluation is in PTIME, NLOGSPACE, LOGSPACE, and AC^0.

(3) Apart from Datalog, rewritability into FO queries is also of interest. In the context of OMQs where the actual query is fixed rather than quantified, several results have been obtained, see e.g. [BLW13, HLSW15, BHLW16] for FO-rewritability of OMQs whose TBox is formulated in a Horn DL and [BtCLW14, FKL17] for FO- and Datalog-rewritability of OMQs whose TBox is formulated in $\mathcal{ALC}$ or an extension thereof. When the query is quantified (as in the current paper), a first relevant result has been established in [LW11] where it is shown that that FO-rewritability is decidable

for materializable $\mathcal{ALCFI}$-TBoxes of depth one. This underlines the importance of deciding materializability, which would allow to lift this result to (otherwise unrestricted) $\mathcal{ALCFI}$-TBoxes of depth one.

References

[ABI+05]   Eric Allender, Michael Bauland, Neil Immerman, Henning Schnoor, and Heribert Vollmer. The complexity of satisfiability problems: Refining Schaefer's theorem. In *MFCS*, pages 71–82, 2005.

[ACKZ09]   Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The DL-Lite family and relations. *J. Artif. Intell. Res. (JAIR)*, 36:1–69, 2009.

[ACY91]   Foto N. Afrati, Stavros S. Cosmadakis, and Mihalis Yannakakis. On datalog vs. polynomial time. In *PODS*, pages 13–25, 1991.

[Ats08]   Albert Atserias. On digraph coloring problems and treewidth duality. *Eur. J. Comb.*, 29(4):796–820, 2008.

[Bar14]   Libor Barto. Constraint satisfaction problem and universal algebra. *SIGLOG News*, 1(2):14–24, 2014.

[BBLW16]   Franz Baader, Meghyn Bienvenu, Carsten Lutz, and Frank Wolter. Query and predicate emptiness in ontology-based data access. *J. Artif. Intell. Res. (JAIR)*, 56:1–59, 2016.

[BBL05]   Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In *IJCAI*, pages 364–369. Professional Book Center, 2005.

[BGO10]   Vince Barany, Georg Gottlob, and Martin Otto. Querying the guarded fragment. In *LICS*, pages 1–10, 2010.

[BHLW16]   Meghyn Bienvenu, Peter Hansen, Carsten Lutz, and Frank Wolter. First order-rewritability and containment of conjunctive queries in Horn description logics. In *IJCAI*, pages 965–971, 2016.

[Bul17]   Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *FOCS*, 2017.

[BJK05]   Andrei A. Bulatov, Peter Jeavons, and Andrei A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM J. Comput.*, 34(3):720–742, 2005.

[BLW13]   Meghyn Bienvenu, Carsten Lutz, and Frank Wolter. First-order rewritability of atomic queries in horn description logics. In *IJCAI*, pages 754–760, 2013.

[BMRT11]   Jean-Francois Baget, Marie-Laure Mugnier, Sebastian Rudolph, and Michael Thomazo. Walking the complexity lines for generalized guarded existential rules. In *IJCAI*, pages 712–717, 2011.

[BO15]   Meghyn Bienvenu and Magdalena Ortiz. Ontology-mediated query answering with data-tractable description logics. In *Reasoning Web*, volume 9203 of *LNCS*, pages 218–307. Springer, 2015.

[BtCLW14]   Meghyn Bienvenu, Balder ten Cate, Carsten Lutz, and Frank Wolter. Ontology-based data access: A study through disjunctive datalog, CSP, and MMSNP. *ACM Trans. Database Syst.*, 39(4):33:1–33:44, 2014.

[BLR+16]   Elena Botoeva, Carsten Lutz, Vladislav Ryzhikov, Frank Wolter and Michael Zakharyaschev. Query-Based Entailment and Inseparability for ALC Ontologies In *IJCAI*, pages 1001–1007, 2016.

[Bul02]   Andrei A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *FOCS*, pages 649–658, 2002.

[Bul11]   Andrei A. Bulatov. On the CSP dichotomy conjecture. In *CSR*, pages 331–344, 2011.

[CDGL+07]   Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. of Autom. Reasoning*, 39(3):385–429, 2007.

[CDL+13]   Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Data complexity of query answering in description logics. *Artificial Intelligence*, 195:335–360, 2013.

[CGK13]   Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res. (JAIR)*, 48:115–174, 2013.

[CGLV00]   Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. View-based query processing and constraint satisfaction. In *LICS*, pages 361–371, 2000.

[CGLV03a]   Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Reasoning on regular path queries. *SIGMOD Record*, 32(4):83–92, 2003.

[CGLV03b]   Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. View-based query containment. In *PODS*, pages 56–67, 2003.

[CGT89]   Stefano Ceri, Georg Gottlob, and Letizia Tanca. What you always wanted to know about datalog (and never dared to ask). *IEEE Trans. Knowl. Data Eng.*, 1(1):146–166, 1989.

[CK90]   C. C. Chang and H. Jerome Keisler. *Model Theory*, volume 73 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, 1990.

[EGOS08]   Thomas Eiter, Georg Gottlob, Magdalena Ortiz, and Mantas Simkus. Query answering in the description logic Horn-$\mathcal{SHIQ}$. In *JELIA*, pages 166–179, 2008.

[FKL17]   Cristina Feier, Antti Kuusisto, and Carsten Lutz. Rewritability in monadic disjunctive datalog, MMSNP, and expressive description logics. In *ICDT*, pages 1–17, 2017.

[FKMP05]   Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.

[FV98]   Tomás Feder and Moshe Y. Vardi. The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory *SIAM J. Comput.*, 28(1):57–104, 1998.

[FV03]   Tomás Feder and Moshe Y. Vardi. Homomorphism closed vs. existential positive. In *LICS*, pages 311–320, 2003.

[GLHS08]   Birte Glimm, Carsten Lutz, Ian Horrocks, and Ulrike Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *JAIR*, 31:157–204, 2008.

[GO07]   Valentin Goranko and Martin Otto. Model theory of modal logic. In *Handbook of Modal Logic*, pages 249–329. Elsevier, 2007.

[HLSW15]   Peter Hansen, Carsten Lutz, Inanç Seylan, and Frank Wolter. Efficient query rewriting in the description logic $\mathcal{EL}$ and beyond. In *IJCAI*, pages 3034–3040, 2015.

[HLPW17a] André Hernich, Carsten Lutz, Fabio Papacchini, Frank Wolter. Dichotomies in Ontology-Mediated Querying with the Guarded Fragment. In *PODS*, pages 185–199, 2017.

[HLPW17b] André Hernich, Carsten Lutz, Fabio Papacchini, Frank Wolter. Horn Rewritability vs PTime Query Answering for Description Logic TBoxes. In *Description Logics*, 2017.

[HMS07]   Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reasoning in description logics by a reduction to disjunctive datalog. *J. Autom. Reasoning*, 39(3):351–384, 2007.

[HN90]   Pavol Hell and Jaroslav Nesetril. On the complexity of *h*-coloring. *J. Comb. Theory, Ser. B*, 48(1):92–110, 1990.

[KNC16]   Mark Kaminski, Yavor Nenov, and Bernardo Cuenca Grau, Datalog rewritability of Disjunctive Datalog programs and non-Horn ontologies. *Artificial Intelligence*, 236: 90–118, 2016.

[Kaz09]   Yevgeny Kazakov. Consequence-driven reasoning for Horn-$\mathcal{SHIQ}$ ontologies. In Craig Boutilier, editor, *IJCAI*, pages 2040–2045, 2009.

[KL07]   Adila Krisnadhi and Carsten Lutz. Data complexity in the $\mathcal{EL}$ family of description logics. In *LPAR*, pages 333–347, 2007.

[KLT$^+$10] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. The combined approach to query answering in DL-Lite. In *KR*, 2010.

[KRH07]   Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler. Complexity boundaries for Horn description logics. In *AAAI*, pages 452–457, 2007.

[Kro10a]   Andrei A. Krokhin. Tree dualities for constraint satisfaction. In *CSL*, pages 32–33, 2010.

[Krö10b]   Markus Krötzsch. Efficient inferencing for OWL EL. In *JELIA*, pages 234–246, 2010.

[KS09]   Gábor Kun and Mario Szegedy. A new line of attack on the dichotomy conjecture. In *STOC*, pages 725–734, 2009.

[KZ14]   Roman Kontchakov and Michael Zakharyaschev. An introduction to description logics and query rewriting. In *Reasoning Web*, volume 8714 of *LNCS*, pages 195–244. Springer, 2014.

[LLT07]   Benoit Larose, Cynthia Loten, and Claude Tardif. A characterisation of first-order constraint satisfaction problems. *Logical Methods in Computer Science*, 3(4), 2007.

[LPW11]   Carsten Lutz, Robert Piro, and Frank Wolter. Description logic tboxes: Model-theoretic characterizations and rewritability. In *IJCAI*, 2011.

[LSW13]   Carsten Lutz, Inanç Seylan, and Frank Wolter. Ontology-based data access with closed predicates is inherently intractable (sometimes). In *IJCAI*, pages 1024–1030. IJCAI/AAAI, 2013.

[LS17]   Carsten Lutz and Leif Sabellek. Ontology-Mediated Querying with the Description Logic $\mathcal{EL}$: Trichotomy and Linear Datalog Rewritability. In *IJCAI*, pages 1181–1187. IJCAI/AAAI, 2017.

[LSW15]   Carsten Lutz, Inanç Seylan, and Frank Wolter. Ontology-mediated queries with closed predicates. In *IJCAI*, pages 3120–3126. AAAI Press, 2015.

[LTW09]   Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In *IJCAI*, pages 2070–2075, 2009.

[LW10]   Carsten Lutz and Frank Wolter. Deciding inseparability and conservative extensions in the description logic
            $\mathcal{EL}$. *J. Symb. Comput.*, 45(2):194–228, 2010.
[LW11]   Carsten Lutz and Frank Wolter. Non-uniform data complexity of query answering in description logics. In
            Description Logics, 2011.
[LW12]   Carsten Lutz and Frank Wolter. Non-uniform data complexity of query answering in description logics. In
            KR. AAAI Press, 2012.
[Mak87]   Johann A. Makowsky. Why Horn formulas matter in computer science: Initial structures and generic ex-
            amples. *J. Comput. Syst. Sci.*, 34(2/3):266–292, 1987.
[Mal71]   Anatoli I. Malcev. *The metamathematics of algebraic systems, collected papers:1936-1967*. North-Holland,
            1971.
[MG85]   Jose Meseguer and Joseph A. Goguen. Initiality, induction, and computability. In *Algebraic Methods in
            Semantics*, pages 459–541. Cambridge University Press, 1985.
[OCE08]   Magdalena Ortiz, Diego Calvanese, and Thomas Eiter. Data complexity of query answering in expressive
            description logics via tableaux. *J. of Autom. Reasoning*, 41(1):61–98, 2008.
[PLC$^+$08] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and
            Riccardo Rosati. Linking data to ontologies. *J. Data Semantics*, 10:133–173, 2008.
[Ros07]   Riccardo Rosati. The limits of querying ontologies. In *ICDT*, volume 4353 of *LNCS*, pages 164–178.
            Springer, 2007.
[Sch78]   Thomas J. Schaefer. The complexity of satisfiability problems. In *STOC*, pages 216–226, 1978.
[Sch93]   Andrea Schaerf. On the complexity of the instance checking problem in concept languages with existential
            quantification. *J. of Intel. Inf. Systems*, 2:265–278, 1993.
[Zhu17]   Dmitriy Zhuk. The Proof of CSP Dichotomy Conjecture. In *FOCS*, 2017.