When is Ontology-Mediated Querying Efficient?

Pablo Barceló
DCC, U of Chile & IMFD Chile
pbarcelo@dcc.uchile.cl

Cristina Feier
University of Bremen
feier@uni-bremen.de

Carsten Lutz
University of Bremen
clu@uni-bremen.de

Andreas Pieris
University of Edinburgh
apieris@inf.ed.ac.uk

***Abstract*—In ontology-mediated querying, description logic (DL) ontologies are used to enrich incomplete data with domain knowledge which results in more complete answers to queries. However, the evaluation of ontology-mediated queries (OMQs) over relational databases is computationally hard. This raises the question when OMQ evaluation is efficient, in the sense of being tractable in combined complexity or fixed-parameter tractable. We study this question for a range of ontology-mediated query languages based on several important and widely-used DLs, using unions of conjunctive queries as the actual queries. For the DL $\mathcal{ELHI}_\bot$, we provide a characterization of the classes of OMQs that are fixed-parameter tractable. For its fragment $\mathcal{ELH}^{dr}_\bot$, which restricts the use of inverse roles, we provide a characterization of the classes of OMQs that are tractable in combined complexity. Both results are in terms of equivalence to OMQs of bounded tree width and rest on a reasonable assumption from parameterized complexity theory. They are similar in spirit to Grohe's seminal characterization of the tractable classes of conjunctive queries over relational databases. We further study the complexity of the meta problem of deciding whether a given OMQ is equivalent to an OMQ of bounded tree width, providing several completeness results that range from NP to 2ExpTime, depending on the DL used. We also consider the DL-Lite family of DLs, including members that, unlike $\mathcal{ELHI}_\bot$, admit functional roles.**

I. Introduction

An ontology-mediated query (OMQ) is a database query enriched with an ontology that contains domain knowledge [11], [13], [14]. Adding the ontology serves the purpose of delivering more complete answers to queries and of enriching the vocabulary available for querying. Ontologies are often formulated in description logics (DLs), a family of ontology languages that has emerged from artificial intelligence, underlies the OWL 2 recommendation for ontology languages on the web, and whose members can be seen as decidable fragments of (two-variable guarded) first-order logic [4]. The actual queries in OMQs are typically conjunctive queries (CQs), unions of CQs (UCQs), or fragments thereof, query languages that are at the heart of relational databases [1].

An *OMQ language* is a pair $(\mathcal{L}, \mathcal{Q})$ with $\mathcal{L}$ an ontology language and $\mathcal{Q}$ a query language [13]. Depending on the OMQ language chosen, the computational cost of evaluating OMQs can be high. Both the combined complexity and the data complexity of OMQ evaluation have received considerable interest in the literature, where data complexity means that the OMQ is fixed while the database is treated as an input, in line with the standard setup from database theory. The combined complexity ranges from PTime [8], [9], [12] to at least 2ExpTime [21], [31], [34]. Regarding the data complexity, there is an important divide between DLs that include negation or disjunction and induce coNP-hardness, and DLs that do not [16], [20], [27], [29]. Studying the data complexity of the former has turned out to be closely related to the complexity of constraint satisfaction problems (CSPs) [13].

In this paper, we explore the frontiers of two important notions of OMQ tractability, PTime combined complexity and fixed-parameter tractability (FPT) where the parameter is the size of the OMQ. We believe these to be more realistic than PTime data complexity given that ontologies can get large. In fact, hundreds or thousands of logical statements are not unusual in real world ontologies, and this can even go up to hundreds of thousands in extreme but important cases such as SNOMED CT [40]. However, there are only few OMQ languages that have PTime combined complexity or are FPT without imposing serious restrictions on the shape of the query or the ontology. An example for the former is $(\mathcal{ELH}^{dr}_\bot, \text{AQ})$ where $\cdot^{dr}$ stands for domain and range restrictions and AQ refers to the class of atomic queries of the form $A(x)$, A a concept name; this result is implicit in [32]. An example for FPT is $(\mathcal{ELHI}_\bot, \text{AQ})$; we are not aware of this being stated explicitly anywhere, but it is not too hard to prove using standard means. Both $\mathcal{ELH}^{dr}_\bot$ and $\mathcal{ELHI}_\bot$ are widely used DLs that underpin profiles of the OWL 2 recommendation [36]. One should think of the former as an important DL without negation and disjunction, and of the latter as an important DL in which basic reasoning problems such as subsumption are still in PTime. Note that the unrestricted use of CQs and UCQs rules out both of the considered complexities since (U)CQ-evaluation is NP-complete in complexity and W[1]-hard, thus most likely not fixed-parameter tractable [24].

A remarkable result by Grohe precisely characterizes the (recursively enumerable) classes of CQs over schemas of bounded arity that can be evaluated in PTime [25]: this is the case if and only if for some k, every CQ in the class is equivalent to a CQ of tree width k, unless the assumption from parameterized complexity theory that FPT $\neq$ W[1] fails. Grohe's result also establishes that PTime complexity and FPT coincide for evaluating CQs. A generalization to UCQs is in [17], more details are given in Sections II and III.

Our main contribution is to establish the following, under the widely-held assumption that FPT $\neq$ W[1]:

1) a precise characterization of classes of OMQs from $(\mathcal{ELHI}_\bot, \text{UCQ})$ for which evaluation is in FPT as those

978-1-7281-3608-0/19/$31.00 ©2019 IEEE

classes in which each OMQ is equivalent to an OMQ of bounded tree width (Section IV);

2) a precise characterization of classes of OMQs from $(\mathcal{ELH}^{dr}_\bot, \text{UCQ})$ that admit PTIME evaluation as those classes in which each OMQ is equivalent to an OMQ of bounded tree width (Section V),

where an OMQ has bounded tree width if the actual query in it has. Regarding Point 1, we also observe that the runtime of the FPT algorithm can be made single exponential in the parameter. In Point 2, we work under the assumption that the ontology does not introduce relations beyond those admitted in the database; such additional relations are introduced to enrich the vocabulary available for querying, bearing a similarity to views in relational databases [30]. Given that $\mathcal{ELH}^{dr}_\bot$ is a fragment of $\mathcal{ELHI}_\bot$, Points 1 and 2 imply that PTIME complexity and FPT coincide in $(\mathcal{ELH}^{dr}_\bot, \text{UCQ})$. To prove the 'upper bound' of Point 2, we use existential pebble games adapted in a careful way to OMQs. For the rather non-trivial 'lower bound', we build on Grohe's result. Here, the fact that OMQs can introduce additional relations results in serious challenges; in fact, several fundamental techniques that are standard in relational databases must be replaced by more subtle ones.

Related to our second main result, it has been shown in in [12] that whenever $\mathcal{Q}$ is a class of CQs that can be evaluated in PTIME, then the same is true for OMQs from $(\mathcal{ELH}, \mathcal{Q})$. In particular, $\mathcal{Q}$ might be the class of CQs of tree width bounded by some k. Our tractability results are stronger than this: adding an ontology can lower the complexity of a (U)CQ and it is in fact not hard to see that there are classes of OMQs from $(\mathcal{EL}, \text{CQ})$ that can be evaluated in PTIME, but the class of CQs used in them cannot. In our characterizations, equivalence to an OMQ Q of bounded tree width includes the case that Q uses a different ontology than the original OMQ. We also show, however, that in most of the studied cases there is no benefit in changing the ontology. More loosely related studies of the combined complexity of OMQs in which the ontology is formulated in fragments of $\mathcal{ELHI}_\bot$ such as DL-Lite$^\mathcal{R}$ and DL-Lite$^\mathcal{R}_\mathsf{horn}$, important in ontology-based data integration, and where the queries have bounded tree width are in [8], [9].

We further study the complexity of the meta problem of deciding whether a given OMQ is equivalent to an OMQ of bounded tree width (Section VI). Decidability is needed for the characterizations described above, but we also consider this question interesting in its own right. Our results range from Π^p_2 between (DL-Lite$^\mathcal{R}$, CQ) and (DL-Lite$^\mathcal{R}_\mathsf{horn}$, UCQ) via EXPTIME between $(\mathcal{EL}, \text{CQ})$ and $(\mathcal{ELH}^{dr}_\bot, \text{UCQ})$ to 2EXPTIME between $(\mathcal{ELI}, \text{CQ})$ and $(\mathcal{ELHI}_\bot, \text{UCQ})$; all these are completeness results. As an important special case, we consider the full database schema, meaning that the ontology cannot introduce additional relations. There, the complexity drops considerably, to NP, NP, and EXPTIME, respectively. The case of the full schema is also interesting because it admits constructions that are close to the case of relational databases, such as (a suitably adapted version of) retracts. We remark that when the schema is full, the problems studied here are closely related to the evaluation of (U)CQs of bounded tree width over relational databases with integrity constraints [6]. However, the constraints languages considered there are different from ontology languages and the connection breaks when the schema is not full.

Finally, we take a first glimpse at ontology languages that include a form of counting, more precisely at DL-Lite$^\mathcal{F}$, in which some binary relations can be declared to be partial functions (Section VII). This turns out to be closely related to the evaluation of UCQs over relational databases in the presence of key dependencies, as studied by Figueira [23]. We show that evaluating OMQs that are equivalent to an OMQ of tree width bounded by some k is in FPT and even in PTime when $k = 1$, and that the meta problem of deciding whether an OMQ belongs to this class is decidable in 3EXPTIME and NP-complete when $k = 1$. In this part, we assume the full database schema and Boolean queries. For the case $k > 1$, we additionally assume that the ontology cannot be changed.

Most of the proof details are deferred to the appendix, available at http://www.informatik.uni-bremen.de/tdki.

II. PRELIMINARIES

A. Databases and Queries

Databases. Let $\mathsf{N_C}$, $\mathsf{N_R}$, and $\mathbf{C}$ be countably infinite sets of *concept names*, *role names*, and *constants*, respectively. A *database* $\mathcal{D}$ is a finite set of *facts* of the form $A(a)$ and $r(a, b)$ where, here and in the remainder of the paper, A ranges over $\mathsf{N_C}$, r ranges over $\mathsf{N_R}$, and a, b range over $\mathbf{C}$. We denote by $\mathsf{dom}(\mathcal{D})$ the set of constants used in $\mathcal{D}$ and sometimes write $r^-(a, b) \in \mathcal{D}$ in place of $r(b, a) \in \mathcal{D}$. A *schema* $\mathbf{S}$ is a set of concept and role names. An $\mathbf{S}$-*database* is a database that uses only concept and role names from $\mathbf{S}$. Note that as usual in the context of DLs, databases can only refer to unary and binary relations, i.e., concept and role names, but not to relations of higher (or lower) arity. We shall sometimes consider also infinite databases and then say so explicitly.

A *homomorphism* from database $\mathcal{D}_1$ to database $\mathcal{D}_2$ is a function $h : \mathsf{dom}(\mathcal{D}_1) \to \mathsf{dom}(\mathcal{D}_2)$ such that $A(h(a)) \in \mathcal{D}_2$ for every $A(a) \in \mathcal{D}_1$, and $r(h(a), h(b)) \in \mathcal{D}_2$ for every $r(a, b) \in \mathcal{D}_1$. We write $\mathcal{D}_1 \to \mathcal{D}_2$ if there is a homomorphism from $\mathcal{D}_1$ to $\mathcal{D}_2$. For a database $\mathcal{D}$ and tuple (or set) $\mathbf{a}$ of constants, we write $\mathcal{D}|_\mathbf{a}$ to denote the restriction of $\mathcal{D}$ to facts that involve only constants from $\mathbf{a}$.

Conjunctive Queries. A *conjunctive query (CQ)* is of the form $q = \exists \mathbf{y}\, \varphi(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ and $\mathbf{y}$ are tuples of variables and $\varphi(\mathbf{x}, \mathbf{y})$ is a conjunction of *atoms* of the form $A(x)$ and $r(x, y)$ with x, y variables. We call $\mathbf{x}$ the *answer variables* of q, $\mathbf{y}$ the *quantified variables*, and use $\mathsf{var}(q)$ to denote $\mathbf{x} \cup \mathbf{y}$. We take the liberty to write $\alpha \in q$ to indicate that α is an atom in q and sometimes write $r^-(x, y) \in q$ in place of $r(y, x) \in q$. We neither admit equality atoms nor constants in CQs, but all results in this paper remain valid when both are admitted.

Every CQ q can be seen as a database $\mathcal{D}_q$ by dropping the existential quantifier prefix and viewing variables as constants. A *homomorphism* from q to a database $\mathcal{D}$ is a homomorphism from $\mathcal{D}_q$ to $\mathcal{D}$. We write $\mathcal{D} \models q(\mathbf{a})$ and call the tuple of constants $\mathbf{a}$ an *answer to q on $\mathcal{D}$* if there is a homomorphism h from q to $\mathcal{D}$ with $h(\mathbf{x}) = \mathbf{a}$, $\mathbf{x}$ the answer variables of q. Moreover, $q(\mathcal{D})$ denotes the set of all answers to q on $\mathcal{D}$.

A *union of conjunctive queries (UCQ)* q is a disjunction of one or more CQs that all have the same answer variables. *Answers* to a UCQ q are defined in the expected way, and so is $q(\mathcal{D})$. The *arity* of a (U)CQ q is defined as the number of answer variables in it, and we use the term *Boolean* interchangeably with 'arity zero'.

A *homomorphism* from a CQ $q_1(\mathbf{x})$ to a CQ $q_2(\mathbf{x})$ is a homomorphism h from $\mathcal{D}_{q_1}$ to $\mathcal{D}_{q_2}$ such that $h(\mathbf{x}) = \mathbf{x}$. We write $q_1 \rightarrow q_2$ if such a homomorphism exists. For a CQ q and tuple (or set) $\mathbf{z}$ of variables, the restriction $q|_{\mathbf{z}}$ of q to the variables in $\mathbf{z}$ is defined in the expected way (it might involve a change of arity).

Tree Width. The *evaluation problem for CQs* takes as input a CQ $q(\mathbf{x})$, a database $\mathcal{D}$, and a candidate answer $\mathbf{a}$, and asks whether $\mathbf{a} \in q(\mathcal{D})$. While in general the evaluation problem for CQs is NP-complete, it becomes tractable for CQs of *bounded tree width* [18]. The notion of tree width of a CQ, which is central to our work, is introduced next.

A *tree decomposition* of an undirected graph $G = (V, E)$ is a triple $D = (V_D, E_D, \mu)$ where (V_D, E_D) is an undirected tree and $\mu : V_D \rightarrow 2^V$ a function such that

- $\bigcup_{t \in V_D} \mu(t) = V$;
- if $\{v_1, v_2\} \in E$, then $v_1, v_2 \in \mu(t)$ for some $t \in V_D$;
- for every $v \in V$, the subgraph of (V_D, E_D) induced by the vertex set $\{t \in V_D \mid v \in \mu(t)\}$ is connected.

The *width* of D is $\max_{t \in V_D} |\mu(t)| - 1$ and the *tree width* of G is the smallest width of any tree decomposition of G.

Each database $\mathcal{D}$ is associated with an undirected graph (without self loops) $G_{\mathcal{D}}$, its *Gaifman graph*, defined as follows: the nodes of $G_{\mathcal{D}}$ are the constants in $\mathcal{D}$ and there is an edge $\{a, b\}$ iff $\mathcal{D}$ contains a fact that involves both a and b. Each CQ q is associated with the directed graph $G_q = G_{\mathcal{D}_q}$. We can thus use standard terminology from graph theory for databases and CQs, e.g. saying that a database $\mathcal{D}$ is connected and speaking about the tree width of $\mathcal{D}$ (when $\mathcal{D}$ contains no binary facts, we define its tree width to be 1). There is an exception, though. The *tree width of a CQ* $q = \exists \mathbf{y}\, \varphi(\mathbf{x}, \mathbf{y})$ is defined in a more liberal way, namely as the tree width of $G_{q|_{\mathbf{y}}}$ (and 1 if $q|_{\mathbf{y}}$ contains no binary atoms). For every $k \geq 0$, a *CQ$_k$* is a CQ of tree width at most k and a *UCQ$_k$* is a union of CQ$_k$s with the same answer variables.

B. Description Logics and Ontology-Mediated Queries

Concepts and Ontologies. We introduce several widely used description logics, see [4] for more details. An *$\mathcal{ELI}_\bot$-concept* is formed according to the syntax rule

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

An expression r^- is an *inverse role* and a *role* is a role name or an inverse role. As usual, we identify $(r^-)^-$ with r. An *$\mathcal{EL}_\bot$-concept* is an $\mathcal{ELI}_\bot$-concept with no inverse roles.

$$\begin{array}{rlrl}
\top^{\mathcal{I}} = & \!\!\mathsf{dom}(\mathcal{I}) & \bot^{\mathcal{I}} = & \!\!\emptyset \\
A^{\mathcal{I}} = & \!\!\{d \mid A(d) \in \mathcal{I}\} & r^{\mathcal{I}} = & \!\!\{(d, e) \mid r(d, e) \in \mathcal{I}\} \\
(C \sqcap D)^{\mathcal{I}} = & \!\!C^{\mathcal{I}} \cap D^{\mathcal{I}} & (r^-)^{\mathcal{I}} = & \!\!\{(e, d) \mid r(d, e) \in \mathcal{I}\} \\
\multicolumn{4}{c}{(\exists r.C)^{\mathcal{I}} = \{d \in \mathsf{dom}(\mathcal{I}) \mid \exists e : (d, e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}}
\end{array}$$

Fig. 1. Semantics of $\mathcal{ELI}_\bot$-concepts

An *$\mathcal{ELHI}_\bot$-ontology* is a finite set of *$\mathcal{ELI}_\bot$-concept inclusions* of the form $C \sqsubseteq D$, with C, D $\mathcal{ELI}_\bot$-concepts, and *role inclusions* of the form $r \sqsubseteq s$, with r, s roles. In the name $\mathcal{ELHI}_\bot$, the letter $\mathcal{H}$ indicates that role inclusions are admitted, $\mathcal{I}$ indicates that inverse roles are admitted, and $\cdot_\bot$ indicates that the $\bot$-concept may be used. It should thus also be clear what we mean by an *$\mathcal{EL}$-ontology*, an *$\mathcal{ELH}_\bot$-ontology*, and so on. An *$\mathcal{ELH}^{dr}_\bot$-ontology* is an $\mathcal{ELH}_\bot$-ontology that additionally admits *range restrictions* $\exists r^-.\top \sqsubseteq C$ with r a role name and C an $\mathcal{EL}_\bot$-concept. Note that $\cdot^{dr}$ stands for domain and range restrictions, where domain restrictions are simply $\mathcal{EL}_\bot$-concept inclusions of the form $\exists r.\top \sqsubseteq C$. We assume without loss of generality, and without further notice, that the $\bot$-concept occurs only in concept inclusions of the form $C \sqsubseteq \bot$, where C does not contain $\bot$.

Semantics. The semantics of ontologies is defined based on interpretations, relational structures that interpret only relations of arity one and two. We choose a presentation here that is slightly nonstandard, but equivalent to the usual one [4]: an *interpretation* is a finite or infinite database $\mathcal{I}$ with $\mathsf{dom}(\mathcal{I}) \neq \emptyset$. Each $\mathcal{ELI}_\bot$-concept C and role r is associated with an *extension* $C^{\mathcal{I}}$, resp. $r^{\mathcal{I}}$, according to Figure 1.

An interpretation $\mathcal{I}$ *satisfies* a concept inclusion $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, and a role inclusion $r \sqsubseteq s$ if $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$. It is a *model* of an $\mathcal{ELHI}_\bot$-ontology $\mathcal{O}$ if it satisfies all the inclusions in $\mathcal{O}$, and of a database $\mathcal{D}$ if $\mathcal{D} \subseteq \mathcal{I}$. A database $\mathcal{D}$ is *consistent with* $\mathcal{O}$ if $\mathcal{D}$ and $\mathcal{O}$ have a common model.

Note that standard reasoning tasks, e.g., the consistency of a given database with a given ontology, are in PTIME in $\mathcal{ELH}^{dr}_\bot$ and EXPTIME-complete between $\mathcal{ELI}_\bot$ and $\mathcal{ELHI}_\bot$ [4]. Note also that all the DLs defined above can be translated into (two-variable guarded) first-order logic in a standard way [4].

Ontology-Mediated Queries. An *ontology-mediated query (OMQ)* takes the form $Q = (\mathcal{O}, \mathbf{S}, q)$ with $\mathcal{O}$ an ontology, $\mathbf{S}$ a schema (which indicates that Q will be evaluated over $\mathbf{S}$-databases), and q a query. The *arity* of Q is the arity of q. We write $Q(\mathbf{x})$ to emphasize that the answer variables of q are $\mathbf{x}$. When $\mathbf{S} = \mathsf{N_C} \cup \mathsf{N_R}$, then we denote it with $\mathbf{S}_{\mathsf{full}}$ and speak of the *full schema*. It makes perfect sense to use a non-full schema $\mathbf{S}$ while referring to concept and role names from outside $\mathbf{S}$ in both the ontology $\mathcal{O}$ and query q. In fact, enriching the schema with additional symbols is one main application of ontologies in querying [3]. This is similar to the distinction between extensional and intensional relations in Datalog [1].

Consider an OMQ $Q(\mathbf{x})$ and an $\mathbf{S}$-database $\mathcal{D}$. A tuple $\mathbf{a} \in \mathsf{dom}(\mathcal{D})^{|\mathbf{x}|}$ is an *answer* to Q on $\mathcal{D}$, written $\mathcal{D} \models Q(\mathbf{a})$, if $\mathcal{I} \models q(\mathbf{a})$ for all models $\mathcal{I}$ of $\mathcal{D}$ and $\mathcal{O}$. We write $Q(\mathcal{D})$ for the set of answers to Q on $\mathcal{D}$.

An OMQ $Q = (\mathcal{O}, \mathbf{S}, q)$ is *empty* if, for all $\mathbf{S}$-databases $\mathcal{D}$ consistent with $\mathcal{O}$, there is no answer to Q on $\mathcal{D}$, i.e., $Q(\mathcal{D})$ is empty. Let Q_1, Q_2 be OMQs, $Q_i = (\mathcal{O}_i, \mathbf{S}, q_i)$ for $i \in \{1, 2\}$. Then Q_1 is *contained* in Q_2, written $Q_1 \subseteq Q_2$, if $Q_1(\mathcal{D}) \subseteq Q_2(\mathcal{D})$ for all $\mathbf{S}$-databases $\mathcal{D}$. Further, Q_1 and Q_2 are *equivalent*, written $Q_1 \equiv Q_2$, if $Q_1 \subseteq Q_2$ and $Q_2 \subseteq Q_1$. We use $(\mathcal{L}, \mathcal{Q})$ to refer to the *OMQ language* in which the ontology is formulated in $\mathcal{L}$ and where the actual queries are from $\mathcal{Q}$, e.g., $(\mathcal{EL}_\bot, \mathrm{CQ})$ and $(\mathcal{ELHI}_\bot, \mathrm{UCQ})$. As usual, we write $|O|$ for the *size* of a syntactic object O such as an OMQ, an ontology, or a conjunctive query, that is, the number of symbols needed to write O where concept names, role names, variables names, and the like count as one.

The Chase. The chase is a widely used tool in database theory that allows us, whenever a database is consistent with an $\mathcal{ELHI}_\bot$-ontology $\mathcal{O}$, to construct a *universal model* of $\mathcal{D}$ and $\mathcal{O}$ that enjoys many good properties; cf., [1], [33].

Let $\mathcal{O}$ be an $\mathcal{ELHI}_\bot$-ontology. Intuitively, *the chase of $\mathcal{D}$ with respect to $\mathcal{O}$*, denoted $\mathsf{ch}_\mathcal{O}(\mathcal{D})$, is the potentially infinite interpretation $\mathcal{I}$ that is obtained in the limit of recursively applying the following two rules on $\mathcal{D}$, based on the inclusions in $\mathcal{O}$:

1) if $a \in C^\mathcal{I}$, $C \sqsubseteq D \in \mathcal{O}$, and $D \neq \bot$, then add $D(a)$ to $\mathcal{I}$;
2) if $(a, b) \in r^\mathcal{I}$ and $r \sqsubseteq s \in \mathcal{O}$, then add $s(a, b)$ to $\mathcal{I}$.

In Rule 1, 'add $D(a)$ to $\mathcal{I}$' means to add to $\mathcal{I}$ a finite tree-shaped database that represents the $\mathcal{ELI}$-concept D, identifying its root with a. For example, the concept $A \sqcap \exists r.(B \sqcap \exists s.\top)$ corresponds to the database $\{A(a), r(a, b), B(b), s(b, c)\}$. Our chase is oblivious, a formal definition can be found in the appendix. We sometimes apply the chase directly to a CQ q, implicitly meaning its application to the database $\mathcal{D}_q$. The following lemma summarizes the main properties of the chase.

Lemma 1. *Let $\mathcal{D}$ be a database and $Q = (\mathcal{O}, \mathbf{S}, q)$ an OMQ from $(\mathcal{ELHI}_\bot, UCQ)$. Then*

1) *$\mathcal{D}$ is inconsistent with $\mathcal{O}$ iff there is an $a \in \mathsf{dom}(\mathsf{ch}_\mathcal{O}(\mathcal{D}))$ and a $C \sqsubseteq \bot \in \mathcal{O}$ such that $a \in C^{\mathsf{ch}_\mathcal{O}(\mathcal{D})}$;*
2) *$Q(\mathcal{D}) = q(\mathsf{ch}_\mathcal{O}(\mathcal{D}))$, if $\mathcal{D}$ is consistent with $\mathcal{O}$;*
3) *$\mathsf{ch}_\mathcal{O}(\mathcal{D}) \to \mathcal{I}$ via a homomorphism that is the identity on $\mathsf{dom}(\mathcal{D})$, for every model $\mathcal{I}$ of $\mathcal{D}$ and $\mathcal{O}$;*
4) *the tree width of $\mathcal{D}$ and of $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ are identical.*

C. Parameterized Complexity

We study the evaluation problem for OMQs (defined below) both in terms of a traditional complexity analysis and in terms of its parameterized complexity; cf., [24]. A *parameterized problem* over an alphabet Σ is a pair (P, κ), with $P \subseteq \Sigma^*$ a decision problem and κ a *parameterization* of P, that is, a PTime computable function $\kappa : \Sigma^* \to \mathbb{N}$. A prime example is p-CLIQUE, where P is the set of all pairs (G, k) with G an undirected graph that contains a k-clique and $\kappa(G, k) = k$.

A problem (P, κ) is *fixed-parameter tractable (fpt)* if there is a computable function $f : \mathbb{N} \to \mathbb{N}$ and an algorithm that decides P in time $|x|^{O(1)} \cdot f(\kappa(x))$, where x denotes the input. We use FPT to denote the class of all parameterized problems that are fpt. Notice that FPT corresponds to a relaxation of the usual notion of tractability: a problem in PTime is also in FPT, but the latter class also contains some NP-complete problems.

An *fpt-reduction* from a problem (P_1, κ_1) over Σ_1 to a problem (P_2, κ_2) over Σ_2 is a function $\rho : \Sigma_1^* \to \Sigma_2^*$ such that, for some computable functions $f, g : \mathbb{N} \to \mathbb{N}$,

1) $x \in P_1$ iff $\rho(x) \in P_2$, for all $x \in \Sigma_1^*$;
2) $\rho(x)$ is computable in time $|x|^{O(1)} \cdot f(\kappa_1(x))$, for $x \in \Sigma_1^*$;
3) $\kappa_2(\rho(x)) \leq g(\kappa_1(x))$, for all $x \in \Sigma_1^*$.

An important parameterized complexity class is $\mathrm{W}[1] \supseteq$ FPT. Hardness for W[1] is defined in terms of fpt-reductions. It is believed that FPT $\neq$ W[1], the status of this problem being comparable to that of PTime $\neq$ NP. Hence, if a parameterized problem (P, κ) is W[1]-hard then (P, κ) is not fpt unless FPT $=$ W[1]. A well-known W[1]-hard problem is precisely p-CLIQUE [19].

III. OMQ Evaluation and Semantic Tree-likeness

A. OMQ Evaluation

The main concern of this work is the *evaluation problem* for classes of OMQs $\mathbf{Q}$, defined as follows:

PROBLEM :	Evaluation($\mathbf{Q}$)
INPUT :	An OMQ $Q = (\mathcal{O}, \mathbf{S}, q(\mathbf{x}))$ from $\mathbf{Q}$, an $\mathbf{S}$-database $\mathcal{D}$, a tuple $\mathbf{a} \in \mathsf{dom}(\mathcal{D})^{\lvert\mathbf{x}\rvert}$
QUESTION :	Is it the case that $\mathbf{a} \in Q(\mathcal{D})$?

We are particularly interested in classifying the complexity of Evaluation($\mathbf{Q}$) for *all* subsets $\mathbf{Q}$ of an OMQ language $(\mathcal{L}, \mathcal{Q})$ of interest, where we view the latter as a set of OMQs.

We are also interested in the parameterized version of this problem, with the parameter being the size $|Q|$ of the OMQ Q, as customary in the database literature [37], which we call p-Evaluation($\mathbf{Q}$). In particular, if p-Evaluation($\mathbf{Q}$) is in FPT, then it can be solved in time $|\mathcal{D}|^{O(1)} \cdot f(|Q|)$, for a computable function $f : \mathbb{N} \to \mathbb{N}$. In general, the evaluation problem for CQs is NP-hard, and its parameterized version W[1]-hard [37]. Therefore, the same holds for the OMQ evaluation problem.

Proposition 1. *For any of the DLs $\mathcal{L}$ introduced above,*

1) Evaluation($\mathcal{L}$, CQ) *is* NP-*hard;*
2) p-Evaluation($\mathcal{L}$, CQ) *is* W[1]-*hard.*

The above hold even when the ontology is empty.

On the other hand, CQ evaluation is tractable if restricted to CQs of tree width bounded by k, for any k. As established by Bienvenu et al., this positive behavior extends to OMQ evaluation in $(\mathcal{ELH}, \mathrm{CQ}_k)$ [12], and it is not hard to extend their result to $(\mathcal{ELH}^{dr}_\bot, \mathrm{UCQ}_k)$. We refrain from giving details.

Proposition 2. Evaluation($\mathcal{ELH}^{dr}_\bot, \mathrm{UCQ}_k$) *is in* PTime *for each fixed $k \geq 1$.*

Adding inverse roles, however, destroys this property. In fact, evaluation is ExpTime-complete already in $(\mathcal{ELI}, \mathrm{CQ})$, with the lower bound being a consequence of the fact that the subsumption problem in $\mathcal{ELI}$ is ExpTime-hard [5]. Even with inverse roles, however, evaluating OMQs in which the actual queries are of bounded tree width is still fixed-parameter tractable.

Proposition 3. p-EVALUATION$(\mathcal{ELHI}_\bot, \mathrm{UCQ}_k)$ *is in* FPT, *for any* $k \geq 1$, *with single exponential running time in the parameter.*

B. Semantic Tree-likeness for OMQs

Recall that CQs q and q' over schema $\mathbf{S}$ are *equivalent* if $q(\mathcal{D}) = q'(\mathcal{D})$, for every $\mathbf{S}$-database $\mathcal{D}$. Grohe's Theorem establishes that, under the assumption FPT $\neq$ W[1], the classes of CQs that can be evaluated in PTIME over $\mathbf{S}$-databases are precisely those of bounded tree width *modulo equivalence*. Also, fixed-parameter tractability does not add anything to standard tractability in this scenario.

Theorem 1 (Grohe's Theorem [25])**.** *Let* $\mathbf{Q}$ *be a recursively enumerable class of CQs over a schema* $\mathbf{S}$. *The following are equivalent, assuming* FPT $\neq$ W[1]*:*

- *the evaluation problem for CQs in* $\mathbf{Q}$ *is in* PTIME*;*
- *the evaluation problem for CQs in* $\mathbf{Q}$ *is in* FPT*;*
- *there is a* $k \geq 1$ *such that every* $q \in \mathbf{Q}$ *is equivalent to a CQ* q' *in* CQ_k.

Interestingly, the notion that characterizes tractability in this case, namely, being of bounded tree width modulo equivalence, is decidable. Recall that a *retract* of a CQ q is a homomorphic image q' of q that is also equivalent to q, and a *core* of q is a maximum retract of it, i.e., a retract that admits no further retractions [26]. It can be proved that a CQ q is equivalent to a CQ q' in CQ_k, for $k \geq 1$, iff the core of q is in CQ_k. This problem is NP-complete, for each $k \geq 1$ [18]. There is also a natural generalization of this characterization and of Theorem 1 to the class of UCQs [17].

At this point, it is natural to ask whether it is possible to obtain a characterization of the classes of OMQs that can be efficiently evaluated, in the style of Theorem 1 and, in particular, whether a suitably defined notion of "being equivalent to a query of small tree width" for OMQs exhausts tractability or FPT for OMQ evaluation, as is the case in Grohe's Theorem. The following definition introduces such a notion. Notice that equivalence is applied no longer on the level of the (U)CQ, but to the whole OMQ.

Definition 1 (**UCQ_k-equivalence**)**.** *Let* $\mathcal{L}$ *be one of the DLs introduced above. An OMQ* $Q = (\mathcal{O}, \mathbf{S}, q)$ *from* $(\mathcal{L}, \mathit{UCQ})$ *is* UCQ_k-equivalent *if there exists an OMQ* $Q' = (\mathcal{O}', \mathbf{S}, q')$ *from* $(\mathcal{L}, \mathit{UCQ}_k)$ *such that* $Q \equiv Q'$. *If even* $\mathcal{O} = \mathcal{O}'$, *then we say that* Q *is* UCQ_k-equivalent while preserving the ontology.

Likewise, we define *CQ_k-equivalence* and *CQ_k-equivalence while preserving the ontology*. In informal contexts, we may refer to (U)CQ_k-equivalence as *semantic tree-likeness*. We denote by $(\mathcal{L}, \mathcal{Q})^{\equiv}_{\mathcal{Q}'_k}$, where $\mathcal{Q}, \mathcal{Q}' \in \{\mathrm{CQ}, \mathrm{UCQ}\}$, the class of OMQs from $(\mathcal{L}, \mathcal{Q})$ that are $\mathcal{Q}'_k$-equivalent. For example, $(\mathcal{ELHI}_\bot, \mathrm{CQ})^{\equiv}_{\mathrm{UCQ}_k}$ is the restriction of $(\mathcal{ELHI}_\bot, \mathrm{CQ})$ to OMQs that are equivalent to an OMQ from $(\mathcal{ELHI}_\bot, \mathrm{UCQ}_k)$.

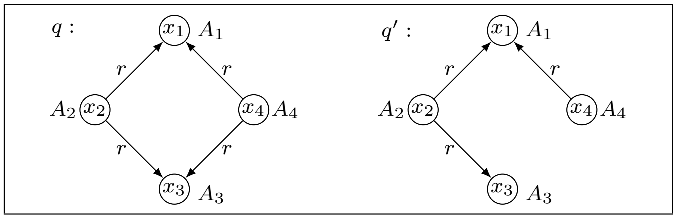

Fig. 2. CQs for Example 1

Example 1. We first illustrate that the ontology can have an impact on tree width. To this end, consider the OMQ $Q_1 = (\mathcal{O}_1, \mathbf{S}_{\mathsf{full}}, q)$ from $(\mathcal{EL}, \mathrm{CQ})$ given by

$$\begin{aligned}\mathcal{O}_1 &= \{A_2 \sqsubseteq A_4\}\\ q() &= r(x_2,x_1) \wedge r(x_4,x_1) \wedge r(x_2,x_3) \wedge r(x_4,x_3) \wedge\\ &\qquad A_1(x_1) \wedge A_2(x_2) \wedge A_3(x_3) \wedge A_4(x_4),\end{aligned}$$

see also Figure 3. Then q is a core of tree width 2, and thus not equivalent to a CQ of tree width 1. Yet Q is from $(\mathcal{EL}, \mathrm{CQ})^{\equiv}_{\mathrm{CQ}_1}$ as it is equivalent to the OMQ $(\mathcal{O}_1, \mathbf{S}_{\mathsf{full}}, q|_{\{x_1,x_2,x_3\}})$ in which the CQ has tree width 1.

We next show that the schema can have an impact as well. This is in a sense trivial as every OMQ based on the empty schema has tree width 1. The following example is more interesting. Let $Q_2 = (\mathcal{O}_2, \mathbf{S}_{\mathsf{full}}, q)$ where

$$\begin{aligned}\mathcal{O}_2 \;=\; \{ \quad & B_1 \sqsubseteq A_1, \quad B_2 \sqsubseteq A_1,\\ & \exists r.B_1 \sqsubseteq A_4, \quad B_2 \sqsubseteq A_3 \quad \}.\end{aligned}$$

Then it is not hard to see that Q_2 is not in $(\mathcal{ELHI}_\bot, \mathrm{CQ})^{\equiv}_{\mathrm{UCQ}_1}$. If, however, the concept name A_1 is omitted from the schema, then Q_2 is equivalent to the OMQ $(\mathcal{O}_2, \mathbf{S}_{\mathsf{full}} \setminus \{A_1\}, q')$ where

$$\begin{aligned}q'() = r(x_2,x_1) \wedge r(x_4,x_1) \wedge r(x_4,x_3) \wedge\\ A_1(x_1) \wedge A_2(x_2) \wedge A_3(x_3) \wedge A_4(x_4)\end{aligned}$$

and thus in $(\mathcal{EL}, \mathrm{CQ})^{\equiv}_{\mathrm{CQ}_1}$. To see this, take a homomorphism h from q' to $\mathcal{I} = \mathsf{ch}_{\mathcal{O}_2}(\mathcal{D})$ for any $\mathbf{S}_{\mathsf{full}} \setminus \{A_1\}$-database $\mathcal{D}$. Then $h(x_1) \in B_1^{\mathcal{I}}$ or $h(x_1) \in B_2^{\mathcal{I}}$. In the former case, we obtain from h a homomorphism from q to $\mathcal{I}$ by setting $h(x_4) = h(x_2)$, in the latter case we set $h(x_3) = h(x_1)$. □

In general, CQ_k-equivalence and UCQ_k-equivalence do not coincide, i.e., sometimes it is possible to rewrite into a disjunction of tree-like CQs, but not into a single one.

Proposition 4. *In* $(\mathcal{ELI}, \mathit{CQ})$, *the notions of* CQ_1*-equivalence while preserving the ontology and* UCQ_1*-equivalence while preserving the ontology do not coincide.*

On the other hand, CQ_k-equivalence and UCQ_k-equivalence coincide in $(\mathcal{ELIH}_\bot, \mathrm{UCQ})$, for all $k \geq 1$, when we restrict our attention to the full schema (see Corollary 2 below).

A Characterization of Semantic Tree-likeness. We provide a characterization of when an OMQ Q is UCQ_k-equivalent. But first we need some auxiliary terminology.

A CQ q is a *contraction* of a CQ q' if it can be obtained from q' by identifying variables. When an answer variable x is identified with a non-answer variable y, the resulting variable is x; the identification of two answer variables is not allowed.

Let $Q = (\mathcal{O}, \mathbf{S}, q) \in (\mathcal{ELHI}_\bot, \text{UCQ})$ and $k \geq 1$. The *UCQ$_k$-approximation of* Q is the OMQ $Q_a = (\mathcal{O}, \mathbf{S}, q_a)$, where q_a denotes the UCQ that consists of all contractions of a CQ from q of tree width at most k. By construction, $Q_a \subseteq Q$, and in this sense Q_a is an approximation of Q from below. The following result gives two central properties of Q_a, in particular that it is the best possible such approximation.

Theorem 2. *Let Q be an OMQ from $(\mathcal{ELHI}_\bot, UCQ)$, $k \geq 1$, and Q_a the UCQ$_k$-approximation of Q. Then*

1) *$Q(\mathcal{D}) = Q_a(\mathcal{D})$ for any $\mathbf{S}$-database $\mathcal{D}$ of treewidth $\leq k$;*
2) *$Q' \subseteq Q_a$ for every $Q' \in (\mathcal{ELHI}_\bot, UCQ_k)$ with $Q' \subseteq Q$.*

Let $Q = (\mathcal{O}, \mathbf{S}, q)$. The proof of Point 1 uses the fact that a homomorphism from q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ gives rise to a collapsing of q whose tree width is not larger than that of $\mathcal{D}$. For Point 2, we 'unravel' the input database into a database of tree width at most k and apply Point 1.

We obtain the following key corollary; '3 ⇒ 2' and '2 ⇒ 1' are immediate, while '1 ⇒ 3' follows from Theorem 2.

Corollary 1. *Let Q be an OMQ from $(\mathcal{ELHI}_\bot, UCQ)$ and $k \geq 1$. The following are equivalent:*

1) *Q is UCQ$_k$-equivalent;*
2) *Q is UCQ$_k$-equivalent while preserving the ontology;*
3) *Q is equivalent to its UCQ$_k$-approximation.*

In $(\mathcal{ELHI}_\bot, \text{UCQ})$, the notion of UCQ$_k$-equivalence thus coincides with UCQ$_k$-equivalence while preserving the ontology. Moreover, Corollary 1 implies decidability of UCQ$_k$-equivalence since OMQ containment is decidable in the OMQ languages considered in this paper [7]. This is further elaborated in Section VI.

Full Schema. We now study the case of OMQs based on the full schema, which admits constructions that are close to the case without ontologies. Recall that in the latter case, a CQ is equivalent to a CQ$_k$ iff its core has tree width at most k. The core, in turn, is defined as the maximum retract. When ontologies are added, there is no longer an equivalent of the core that enjoys good properties. We show, however, that when the schema is full, we can develop a notion of maximum retracts (whose definition involves the chase) such that, for the purposes of this paper, any maximum retract can play the role that the core plays in the case without ontologies.

Let $q(\mathbf{x})$ be a CQ and $\mathcal{O}$ an $\mathcal{ELHI}_\bot$-ontology. An *$\mathcal{O}$-retraction on* q is a homomorphism h from q to $\mathsf{ch}_{\mathcal{O}}(q)$ such that h is the identity on $\mathbf{x}$ and on all variables in the range of h. We use $\mathsf{ran}^+(h)$ to denote the range of h extended with all those $x \in \mathsf{var}(q)$ such that some fresh constant in the subdatabase of tree width 1 that the chase has generated below x is in the range of h. When h is an $\mathcal{O}$-retraction on q, then the restriction q_h of q to $\mathsf{ran}^+(h)$ is an *$\mathcal{O}$-retract* of q.

Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$. A *rewriting* of Q is an OMQ $Q' = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q')$ where q' can be constructed as follows:

1) choose an $\mathcal{O}$-retract q_h of q and set $q' = q_h$;
2) for each $C \sqsubseteq D \in \mathcal{O}$ and $x \in C^{\mathcal{D}_q} \cap \mathsf{dom}(q')$, let q_C be C viewed as a CQ using fresh variables and add q_C to q', identifying x with the root of q_C.

We say that Q' is *based on* q_h. We call Q' a *full* rewriting if q_h has no proper $\mathcal{O}$-retract, that is, the only such retract is q_h itself.

In what follows, we show that full rewritings of OMQs can play the role that the core plays for CQs without an ontology when analyzing semantic tree-likeness. We first observe, however, that full rewritings need not be unique.

Example 2. Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ with $\mathcal{O} = \{A \sqsubseteq \exists r.B,\ B \sqsubseteq A \sqcap \exists r.B\}$ and $q() = \exists x \exists y\, (A(x) \land r(x,y) \land B(y))$. Then both $q_1() = \exists x\, A(x)$ and $q_2() = \exists x\, B(x)$ are $\mathcal{O}$-retracts of q. Moreover, both $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_1)$ and $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_2')$ are full rewritings of Q, where $q_2' = \exists x\, (A(x) \land B(x))$. □

We observe next that an OMQs is equivalent to any of its rewritings.

Lemma 2. *Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ be an OMQ from $(\mathcal{ELHI}_\bot, CQ)$ and $Q' = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q')$ a rewriting of Q. Then $Q \equiv Q'$.*

We now establish the main property of rewritings: an OMQ from $(\mathcal{ELHI}_\bot, \text{CQ})$ based on the full schema is UCQ$_k$-equivalent iff some or all of its rewritings (which is equivalent) fall into $(\mathcal{ELHI}_\bot, \text{CQ}_k)$. In this sense, rewritings behave like a core for CQs without an ontology.

Theorem 3. *Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ be a non-empty OMQ from $(\mathcal{ELHI}_\bot, CQ)$ and $k \geq 1$. The following are equivalent:*

1) *Q is UCQ$_k$-equivalent;*
2) *Q has a rewriting that falls within $(\mathcal{ELHI}_\bot, CQ_k)$;*
3) *some full rewriting of Q falls within $(\mathcal{ELHI}_\bot, CQ_k)$;*
4) *all full rewritings of Q fall within $(\mathcal{ELHI}_\bot, CQ_k)$.*

The interesting part of the proof is '1 ⇒ 4'. It works by showing that if $Q' = (\mathcal{O}, \mathbf{S}, q') \in (\mathcal{ELHI}_\bot, \text{UCQ}_k)$ is equivalent to Q and $Q_f = (\mathcal{O}, \mathbf{S}, q_f)$ is a full rewriting of Q, then there is an injective homomorphism from q_f to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}_p)$ for some CQ p in the UCQ q', and thus the tree width of q_f is bounded by that of q'. We obtain the following corollary.

Corollary 2. *In $(\mathcal{ELHI}_\bot, CQ)$ based on the full schema, CQ$_k$-equivalence and UCQ$_k$-equivalence coincide, for $k \geq 1$.*

IV. Fixed-Parameter Tractability

The aim of this section is to establish the following theorem.

Theorem 4. *For any recursively enumerable class of OMQs $\mathbf{Q} \subseteq (\mathcal{ELHI}_\bot, UCQ)$, the following are equivalent, unless* FPT = W[1]*:*

1) p-Evaluation($\mathbf{Q}$) *is in* FPT*;*
2) *$\mathbf{Q} \subseteq (\mathcal{ELHI}_\bot, UCQ)^{\equiv}_{UCQ_k}$ for some $k \geq 1$.*

If either statement is false, p-Evaluation($\mathbf{Q}$) *is* W[1]*-hard.*

We remark that Theorem 4 also covers OMQs where the ontology is formulated in DL-Lite$^{\mathcal{R}}_{\mathsf{horn}}$, introduced in Section VI. Below, we state the two directions of Theorem 4 as separate theorems, starting with the much simpler '2 ⇒ 1' direction.

Theorem 5. p-EVALUATION$((\mathcal{ELHI}_\bot, UCQ)^{\equiv}_{\overline{UCQ}_k})$ *is in* FPT*, for any* $k \geq 1$*, with single exponential running time in the parameter.*

The above follows from Corollary 1, which states that an OMQ from $(\mathcal{ELHI}_\bot, \text{UCQ})^{\equiv}_{\overline{\text{UCQ}}_k}$ is equivalent to its UCQ$_k$-approximation $Q_a \in (\mathcal{ELHI}_\bot, \text{UCQ}_k)$, and Proposition 3.

Now for the rather non-trivial '1 ⇒ 2' direction, which we consider a main achievement of this paper.

Theorem 6. *Let* $\mathbf{Q} \subseteq (\mathcal{ELHI}_\bot, UCQ)$ *be a recursively enumerable class of OMQs such that, for any* $k \geq 1$*,* $\mathbf{Q} \not\subseteq (\mathcal{ELHI}_\bot, UCQ)^{\equiv}_{\overline{UCQ}_k}$*. Then* p-EVALUATION$(\mathbf{Q})$ *is* W[1]*-hard.*

As stated in Theorem 1, Grohe established a characterization of those classes of Boolean CQs that can be evaluated in PTIME combined complexity [25], a special case of Theorem 4 where ontologies are empty and schemas are full. The 'lower bound part' of Grohe's proof is by fpt-reduction from p-CLIQUE, a W[1]-hard problem. We prove Theorem 6 by following the same approach, carefully reusing a central construction from [25]. For $k, \ell \geq 1$, the *$k \times \ell$-grid* is the graph with vertex set $\{(i, j) \mid 1 \leq i \leq k \text{ and } 1 \leq j \leq \ell\}$ and an edge between (i, j) and (i', j') iff $|i - i'| + |j - j'| = 1$. A *minor* of an undirected graph is defined in the usual way, see, e.g., [25]. When k is understood from the context, we use K to denote $\binom{k}{2}$. The following is what we use from Grohe's proof.

Theorem 7 (Grohe)**.** *Given an undirected graph* $G = (V, E)$*, a* $k > 0$*, and a connected* $\mathbf{S}$*-database* $\mathcal{D}$ *such that* $G_\mathcal{D}$ *contains the* $k \times K$*-grid as a minor, one can construct in time* $f(k) \cdot \mathsf{poly}(|G|, |\mathcal{D}|)$ *an* $\mathbf{S}$*-database* $\mathcal{D}_G$ *such that:*

1) *there is a surjective homomorphism* h_0 *from* $\mathcal{D}_G$ *to* $\mathcal{D}$ *such that for every edge* $\{a, b\}$ *in the Gaifman graph of* $\mathcal{D}_G$*:* $s(a, b) \in \mathcal{D}_G$ *iff* $s(h_0(a), h_0(b)) \in \mathcal{D}$ *for all roles* s*;*
2) G *contains a* k*-clique iff there is a homomorphism* h *from* $\mathcal{D}$ *to* $\mathcal{D}_G$ *such that* $h_0(h(\cdot))$ *is the identity.*

A careful analysis of [25] reveals that the proof given there establishes Theorem 7 without the 'such that' part of Condition (1), which we need to deal with role inclusions. That part, however, can be attained by first suitably switching from the original schema to a schema that is based on sets of relations from the original one, then applying Grohe, and then switching back.

To avoid overly messy notation, we first prove Theorem 6 for the case where $\mathbf{Q} \subseteq (\mathcal{ELHI}_\bot, \text{CQ})$ consists only of Boolean OMQs. In the appendix, we explain how to extend the proof to the non-Boolean case, and from CQs to UCQs.

For the fpt-reduction from p-CLIQUE, assume that G is an undirected graph and $k \geq 1$ a clique size, given as an input to the reduction. By Robertson and Seymour's Excluded Grid Theorem, there is an ℓ such that every graph of tree width exceeding ℓ contains the $k \times K$-grid as a minor [38]. By our assumption on $\mathbf{Q}$, we find an OMQ $Q = (\mathcal{O}, \mathbf{S}, q)$ from $\mathbf{Q}$ such that $Q \notin (\mathcal{ELHI}_\bot, \text{CQ})^{\equiv}_{\overline{\text{UCQ}}_\ell}$. Since the choice of Q is independent of G and since it is decidable whether an OMQ from $(\mathcal{ELHI}_\bot, \text{CQ})$ belongs to $(\mathcal{ELHI}_\bot, \text{CQ})^{\equiv}_{\overline{\text{UCQ}}_\ell}$ by Theorem 11 in Section VI, we can simply enumerate the OMQs from $\mathbf{Q}$ until we find Q.

Let Q_a be the UCQ$_\ell$-approximation of Q. Note that any $\mathbf{S}$-database $\mathcal{D}$ with $\mathcal{D} \models Q$ and $\mathcal{D} \not\models Q_a$ must be of tree width exceeding ℓ since Q_a is equivalent to Q on $\mathbf{S}$-databases of tree width at most ℓ by Theorem 2. Thus $\mathcal{D}$ contains the $k \times K$-grid as a minor, which enables the application of Theorem 7. We could find such $\mathcal{D}$ by brute force enumeration and then hope to show that $\mathcal{D}_G \models Q$ iff there is a homomorphism h from $\mathcal{D}$ to $\mathcal{D}_G$ such that $h_0(h(\cdot))$ is the identity and thus, by Theorem 7, iff G contains a k-clique. This would in fact be easy if $\mathcal{O}$ was empty and $\mathbf{S}$ was full since then we could assume $\mathcal{D}$ to be isomorphic to q, but neither of this is guaranteed. As we show in the following, however, it is possible to construct $\mathcal{D}$ in a very careful way so that its relational structure is sufficiently tightly linked to q to enable the reduction.

A. The Construction of the Database

Injective homomorphisms are an important ingredient to identifying $\mathcal{D}$ since they link a CQ much closer to a database than non-injective homomorphisms. In fact, a main idea is to construct $\mathcal{D}$ such that for some contraction q_c of q: if q_c maps to $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G)$ but only in terms of *injective* homomorphisms, then the same is true for q_c and $\mathsf{ch}_\mathcal{O}(\mathcal{D})$.

For a database $\mathcal{D}$ and a Boolean CQ p, we write $\mathcal{D} \models^{io} p$ if $\mathcal{D} \models p$ and all homomorphisms h from p to $\mathcal{D}$ are injective. Here, 'io' stands for 'injectively only'. We start with a simple observation.

Lemma 3. *If* $\mathcal{D} \models p$*, for* $\mathcal{D}$ *a potentially infinite database and* p *a CQ, then* $\mathcal{D} \models^{io} p_c$ *for some contraction* p_c *of* p*.*

Let $q_1, \dots, q_n$ be the maximal connected components of q. For $1 \leq i \leq n$, let $Q_i = (\mathcal{O}, \mathbf{S}, q_i)$. We can assume w.l.o.g. that $Q_i \not\subseteq Q_j$ for all $i \neq j$ because if this is not the case, then we can drop the component q_j from q and the resulting OMQ is equivalent to Q. Since $Q \notin (\mathcal{ELHI}_\bot, \text{CQ})^{\equiv}_{\overline{\text{UCQ}}_\ell}$, it is clear that $Q_w \notin (\mathcal{ELHI}_\bot, \text{CQ})^{\equiv}_{\overline{\text{UCQ}}_\ell}$ for some w, with $1 \leq i \leq n$. From now on, we use Q_a to denote the UCQ$_\ell$-approximation of Q_w (rather than of Q), which we also compute as part of the reduction.

To achieve the desiderata for $\mathcal{D}$ mentioned above, we next identify an $\mathbf{S}$-database $\mathcal{D}$ such that $\mathcal{D} \models Q_w$ and $\mathcal{D} \not\models Q_a$ and, additionally, if $\mathsf{ch}_\mathcal{O}(\mathcal{D}) \models^{io} q_c$ for a contraction q_c of q_w, then there is no 'less constrained' contraction that does the same, even in databases that homomorphically map to $\mathcal{D}$. Here a contraction q_c of q_w is *less constrained* than a contraction q'_c of q_w, written $q_c \prec q'_c$, when q'_c is a proper contraction of q_c. We write $q_c \preceq q'_c$ when $q_c \prec q'_c$ or $q_c = q'_c$.

Lemma 4. *There is an* $\mathbf{S}$*-database* $\mathcal{D}$ *such that the following conditions are satisfied:*

1) $\mathcal{D} \models Q_w$ *and* $\mathcal{D} \not\models Q_a$*;*
2) *if* $\mathsf{ch}_\mathcal{O}(\mathcal{D}) \models^{io} q_c$*, for* q_c *a contraction of* q_w*, then there is no* $\mathbf{S}$*-database* $\mathcal{D}'$ *and contraction* q'_c *of* q_w *such that* $\mathcal{D}' \to \mathcal{D}$*,* $\mathsf{ch}_\mathcal{O}(\mathcal{D}') \models^{io} q'_c$*, and* $q'_c \prec q_c$*.*

Proof. Since $Q \notin (\mathcal{EL}, \mathrm{CQ})^{\equiv}_{\mathrm{UCQ}_\ell}$ and $Q_a \subseteq Q_w$, we find an $\mathbf{S}$-database $\mathcal{D}_0$ such that $\mathcal{D}_0 \models Q_w$, but $\mathcal{D}_0 \not\models Q_a$. The database $\mathcal{D}_0$ does not necessarily satisfy Condition 2, though. We thus replace it by a more suitable database $\mathcal{D}$, which we identify in an iterative process. Start with $\mathcal{D} = \mathcal{D}_0$ and as long as there are an $\mathbf{S}$-database $\mathcal{D}'$ and contractions q_c, q_c' of q such that $\mathsf{ch}_\mathcal{O}(\mathcal{D}) \models^{io} q_c$, $\mathcal{D}' \rightarrow \mathcal{D}$, $\mathsf{ch}_\mathcal{O}(\mathcal{D}') \models^{io} q_c'$, and q_c is a proper contraction of q_c', replace $\mathcal{D}$ by $\mathcal{D}'$.

It is clear that the resulting $\mathcal{D}$ satisfies Condition (2). Condition (1) is satisfied as well: we have $\mathsf{ch}_\mathcal{O}(\mathcal{D}) \models^{io} q_c$ for some contraction q_c of q_w, thus $\mathcal{D} \models Q_w$; further, $\mathcal{D} \rightarrow \mathcal{D}_0$ and $\mathcal{D}_0 \not\models Q_a$ yield $\mathcal{D} \not\models Q_a$. We prove in the appendix that this iterative process terminates. □

The conditions in Lemma 4 are decidable. It can be shown that it suffices to consider databases $\mathcal{D}'$ of a certain 'pseudo tree shape' (c.f. [7]) which enables a reduction to satisfiability of *monadic second-order logic* (MSO) sentences on trees.

Lemma 5. *Given an $\mathbf{S}$-database $\mathcal{D}$ and an OMQ Q from $(\mathcal{ELHI}_\bot, CQ)$, it is decidable whether Conditions 1 and 2 from Lemma 4 hold.*

Let $\mathcal{D}_0$ be the $\mathbf{S}$-database from Lemma 4. Since the properties of $\mathcal{D}_0$ are independent of G, and due to Lemma 5, we can find $\mathcal{D}_0$ by enumeration. However, $\mathcal{D}_0$ is still not as required and needs to be manipulated further to make it suitable for the reduction. We start with some preliminaries about unravelings.

For each $a \in \mathsf{dom}(\mathcal{D}_0)$, let $\mathcal{D}_0^a$ be the unraveling of $\mathcal{D}_0$ into a database of tree width 1 starting at a, defined in the appendix. The proof of the following lemma is omitted.

Lemma 6. $\mathcal{D}_0 \models (\mathcal{O}, \mathbf{S}, p)(a)$ *iff* $\mathcal{D}_0^a \models (\mathcal{O}, \mathbf{S}, p)(a)$ *for all unary CQs p with $\mathcal{D}_p$ of tree width one.*[1]

Of course, $\mathcal{D}_0^a$ can be infinite. By compactness, however, there is a finite $\mathcal{D}_a \subseteq \mathcal{D}_0^a$ such that Lemma 6 is satisfied for all (finitely many) p that use only symbols from $\mathcal{O}$ and q and satisfy $|p| \leq \max\{|\mathcal{O}|, |q|\}$. For brevity, we say that $\mathcal{D}_a$ *satisfies Lemma 6 for all relevant CQs*. We can find $\mathcal{D}_a$ by constructing $\mathcal{D}_0^a$ level by level and deciding after each such extension whether we have found the desired database, by checking the condition in Lemma 6 for all relevant CQs.

Now for the further manipulation of $\mathcal{D}_0$. We show that $\mathcal{D}_0$ can be replaced with a database $\mathcal{D}^+$ that is more closely linked to q_w than $\mathcal{D}_0$ is. For every $\mathcal{D} \subseteq \mathcal{D}_0$, let $\mathcal{D}^+$ denote the result of starting with $\mathcal{D}$ and then disjointly adding a copy of $\mathcal{D}_a$, identifying the root of this copy with a, for each $a \in \mathsf{dom}(\mathcal{D})$. For what follows, choose $\mathcal{D} \subseteq \mathcal{D}_0$ minimal such that $\mathcal{D}^+ \models Q_w$. Note that $\mathcal{D}$ contains only *binary facts* of the form $r(a, b)$ with $a \neq b$, but no *unary facts* of the form $A(a)$ or $r(a, a)$ since the latter can be made part of $\mathcal{D}_a$. We can find $\mathcal{D}$ by considering all subsets of $\mathcal{D}_0$.

Lemma 7.

1) $\mathcal{D}^+$ satisfies Conditions 1 and 2 of Lemma 4;

2) $\mathcal{D}$ has tree width exceeding ℓ.

By Point 2 of Lemma 7 and choice of ℓ, we have that $\mathcal{D}$ contains the $k \times K$-grid as a minor. We can thus apply Theorem 7 to G, k, and $\mathcal{D}$, obtaining an $\mathbf{S}$-database $\mathcal{D}_G$ and a homomorphism h_0 from $\mathcal{D}_G$ to $\mathcal{D}$ such that Points 1 and 2 of that theorem are satisfied. Recall that $q_1, \ldots, q_n$ are the maximal connected components of q, giving rise to OMQs $Q_1, \ldots, Q_n$, and that $Q_i \not\subseteq Q_j$ for all $i \neq j$. As a consequence, for each $i \neq w$ we can choose an $\mathbf{S}$-database $\mathcal{D}_i$ with $\mathcal{D}_i \models Q_i$ and $\mathcal{D}_i \not\models Q_w$. Let

1) $\mathcal{D}_G^+$ be obtained by starting with $\mathcal{D}_G$ and then disjointly adding, for each $a \in \mathsf{dom}(\mathcal{D}_G)$, a copy of $\mathcal{D}_{h_0(a)}$ identifying the root of this copy with a;
2) $\mathcal{D}_G^*$ be obtained by further disjointly adding $\mathcal{D}_1, \ldots, \mathcal{D}_{w-1}, \mathcal{D}_{w+1}, \ldots, \mathcal{D}_n$.

The fpt reduction of p-CLIQUE consists then in computing Q and $\mathcal{D}_G^*$ from G and $k \geq 1$.

B. Correctness of the Reduction

We show in the subsequent lemma that $\mathcal{D}_G^* \models Q$ if and only if G has a k-clique. For a CQ p, we use $nt(p)$ to denote the result of removing all 'dangling trees' from p, where trees might include reflexive loops and multi-edges and 'nt' stands for 'no trees'. Formally, $nt(p)$ is the maximal subset of p (viewed as a set of atoms) such that there is no articulation point $x \in \mathsf{var}(p)$ that separates $nt(p)$ into components p_1, p_2 with p_2 of tree width 1. It should be clear that $nt(p)$ is uniquely defined when p is connected and contains a non-tree part, that is, the tree width of p exceeds 1.

Lemma 8. *G has a k-clique iff $\mathcal{D}_G^* \models Q$.*

Proof. The 'only if' direction is easy. If G has a k-clique, then $\mathcal{D} \rightarrow \mathcal{D}_G$ by Point 2 of Theorem 7. It is straightforward to extend a witnessing homomorphism to one from $\mathcal{D}^+$ to $\mathcal{D}_G^+$, and thus $\mathcal{D}^+ \rightarrow \mathcal{D}_G^+$. Consequently, $\mathcal{D}^+ \models Q_w$ implies $\mathcal{D}_G^+ \models Q_w$. By construction of $\mathcal{D}^*$ it holds that $\mathcal{D}_G^* \models Q$.

For the 'if' direction, assume that $\mathcal{D}_G^* \models Q$. By choice of the components in $\mathcal{D}^* \setminus \mathcal{D}^+$, this means that $\mathcal{D}_G^+ \models Q_w$. Then $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+) \models q_w$ and by Lemma 3, we find a contraction q_c of q_w such that $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+) \models^{io} q_c$. We have $\mathcal{D}_G \rightarrow \mathcal{D}$ via the homomorphism h_0 from Theorem 7 and it is straightforward to extend h_0 so that it yields $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+) \rightarrow \mathsf{ch}_\mathcal{O}(\mathcal{D}^+)$. It follows that $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+) \models q_c$. Thus we find a contraction q_c' of q_c such that $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+) \models^{io} q_c'$. We must have $q_c = q_c'$ since $\mathcal{D}^+$ satisfies Condition 2 of Lemma 4, via Lemma 7. Let h be a homomorphism from q_c to $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+)$. Then the composition $h_0(h(\cdot))$ is a homomorphism from q_c to $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+)$. Since $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+) \models^{io} q_c$, this homomorphism must be injective. Let g be its restriction to the variables in $nt(q_c)$.[2]

The range of g must fall into $\mathsf{dom}(\mathcal{D})$ since g is injective: if the range of g involved an element from a tree width 1 part of $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+)$, added by the transition from $\mathcal{D}$ to $\mathcal{D}^+$ or by the

[1] Note that this is a stricter requirement than p being of tree width 1 because answer variables are omitted from tree decompositions.

[2] This is uniquely defined since q_c is clearly connected and, moreover, has tree width exceeding ℓ because $\mathcal{D}^+ \not\models Q_a$ and thus q_c is not a CQ in the UCQ q_a in Q_a.

chase, then because of the injectivity of g this gives rise to an articulation point in $nt(q_c)$ that separates $nt(q_c)$ into two components q_1, q_2 with G_{q_2} a tree, but such an articulation point does not exist. Moreover, the elements of $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+)$ that have not been added by the $\cdot^+$-construction or by the chase are precisely those in $\mathsf{dom}(\mathcal{D})$.

Moreover, g must satisfy a certain ontoness condition regarding the subset $\mathcal{D}$ of $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+)$. When we speak of an *edge in* $\mathcal{D}$, we mean an edge $e = \{a, b\} \subseteq \mathsf{dom}(\mathcal{D})$ in the Gaifman graph of $\mathcal{D}$. We say that g *maps an atom* $r(x, y) \in nt(q_c)$ *to* e if $\{g(x), g(y)\} = \{a, b\}$. It can be verified that

(†) for every edge e in $\mathcal{D}$, there is an atom in $nt(q_c)$ that g maps to e.

Assume to the contrary that g maps no atom in $nt(q_c)$ to an edge $\{a, b\} \in \mathcal{D}$. We show in the appendix that, then the database $\mathcal{D}_1$ obtained from $\mathcal{D}$ by removing all binary facts that involve a and b is such that $\mathcal{D}_1^+ \models Q$, contradicting the choice of $\mathcal{D}$.

We are now ready to finish the proof. At this point, we know that g is a restriction of $h_0(h(\cdot))$, that it is injective, and that its range is a subset of $\mathsf{dom}(\mathcal{D})$. In fact, the range of g must be exactly $\mathsf{dom}(\mathcal{D})$, by (†) and since $\mathcal{D}$ contains only binary facts. As a consequence, the inverse h_0^- of h_0 is an injective total function from $\mathsf{dom}(\mathcal{D})$ to $\mathsf{dom}(\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+))$. We next argue that its range actually falls within $\mathsf{dom}(\mathcal{D}_G)$, that is, it does not hit any tree width 1 parts of $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+)$, added by the $\cdot^+$-construction or by the chase. Any constant from $\mathsf{dom}(\mathcal{D})$ occurs in a fact of the form $r(a, b)$. By the ontoness condition (†), g maps some atom $r(x, y) \in nt(q_c)$ to the edge $\{a, b\}$ in $\mathcal{D}$. But then $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+)$ must contain the fact $r(h(x), h(y))$ and, moreover, $\{h(x), h(y)\} = \{h_0^-(a), h_0^-(b)\}$. But $r(h(x), h(y))$ cannot be in any of the tree width 1 parts of $\mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+)$: since h is injective, this would give rise to an articulation point in $nt(q_c)$ that separates $nt(q_c)$ into two components q_1, q_2 with G_{q_2} a tree. With $\{h(x), h(y)\} = \{h_0^-(a), h_0^-(b)\}$, we obtain $h_0^-(a), h_0^-(b) \in \mathsf{dom}(\mathcal{D}_G)$ as desired.

Thus, h_0^- is a function from $\mathsf{dom}(\mathcal{D})$ to $\mathsf{dom}(\mathcal{D}_G)$. We show that it is a homomorphism from $\mathcal{D}$ to $\mathcal{D}_G$, and thus Point 2 of Theorem 7 yields that G contains a k-clique, finishing the proof. Let $r(a, b) \in \mathcal{D}$. By the ontoness condition (†), we have that g maps some atom $s(x, y) \in nt(q_c)$ to the edge $\{a, b\}$. We have already argued that $\{h(x), h(y)\} = \{h_0^-(a), h_0^-(b)\} \subseteq \mathsf{dom}(\mathcal{D}_G)$. Since $s(h(x), h(y)) \in \mathsf{ch}_\mathcal{O}(\mathcal{D}_G^+)$ and $\{h(x), h(y)\} \subseteq \mathsf{dom}(\mathcal{D}_G)$, there must be some fact $s'(h(x), h(y)) \in \mathcal{D}_G$. By the 'such that' part of Point 1 of Theorem 7 and since $\{h(x), h(y)\} = \{h_0^-(a), h_0^-(b)\}$, we have $r(a, b) \in \mathcal{D}_G$. □

We explain in the appendix how to extend the above proof to the case where OMQs need not be Boolean, which essentially amounts to choosing also concrete answers along with databases, and then removing and reading the constants from the answers at the right places in the proof. We also explain how to extend the proof from CQs to UCQs. A difficulty lies in identifying a *connected* component of some CQ in the UCQ q that can play the role of q_w in the original proof, despite the presence of the other disjuncts in q. We overcome this be viewing q as a disjunction of conjunctions of connected CQs and rewriting q into an equivalent conjunction of disjunctions of connected CQs.

V. PTime Combined Complexity

The aim of this section is to establish the following theorem.

Theorem 8. *For any recursively enumerable class of OMQs* $\mathbf{Q} \subseteq (\mathcal{ELH}^{dr}_\bot, \mathit{UCQ})$ *based on the full schema, the following are equivalent, unless* FPT = W[1]*:*

1) Evaluation($\mathbf{Q}$) *is in* PTime *combined complexity;*
2) p-Evaluation($\mathbf{Q}$) *is in* FPT*;*
3) $\mathbf{Q} \subseteq (\mathcal{ELH}^{dr}_\bot, \mathit{UCQ})^{\equiv}_{\mathit{UCQ}_k}$ *for some* $k \geq 1$.

If either statement is false, p-Evaluation($\mathbf{Q}$) *is* W[1]*-hard.*

The '1 ⇒ 2' direction is trivial. For showing '2 ⇒ 3', observe that $\mathbf{Q} \subseteq (\mathcal{ELHI}_\bot, \mathrm{UCQ})$. Thus, by Theorem 6 and the hypothesis FPT ≠ W[1], $\mathbf{Q} \subseteq (\mathcal{ELHI}_\bot, \mathrm{UCQ})^{\equiv}_{\mathrm{UCQ}_k}$. Therefore, for every $Q \in \mathbf{Q}$, there exists $Q' \in (\mathcal{ELHI}_\bot, \mathrm{UCQ}_k)$ such that $Q \equiv Q'$. Since, by Corollary 1, UCQ$_k$-equivalence coincides with UCQ$_k$-equivalence while preserving the ontology, we can assume that $Q' \in (\mathcal{ELH}^{dr}_\bot, \mathrm{UCQ}_k)$. This implies that $Q \in (\mathcal{ELH}^{dr}_\bot, \mathrm{UCQ})^{\equiv}_{\mathrm{UCQ}_k}$. It thus remains to address the '3 ⇒ 1' direction, that is, to prove the following.

Theorem 9. Evaluation($(\mathcal{ELH}^{dr}_\bot, \mathit{UCQ})^{\equiv}_{\mathit{UCQ}_k}$) *based on the full schema is in* PTime *combined complexity, for any* $k \geq 1$.

Evaluating an OMQ $(\mathcal{O}, \mathbf{S}, q)$ from $(\mathcal{ELH}^{dr}_\bot, \mathrm{UCQ})$ is the same as evaluating every OMQ $(\mathcal{O}, \mathbf{S}, p)$, p a CQ in q, and taking the union of the answer sets. To establish Theorem 9, it thus suffices to prove that Evaluation($(\mathcal{ELH}^{dr}_\bot, \mathrm{CQ})^{\equiv}_{\mathrm{UCQ}_k}$) based on the full schema is in PTime.

We do this by using a suitable form of existential pebble game. Such games are also employed in the case of CQ evaluation over relational databases, that is, in the special case of Theorem 9 when the ontology is empty [18], [25]. In that case, the game is played on the CQ q and the input database $\mathcal{D}$, details are given later. When the ontology $\mathcal{O}$ is non-empty, a natural idea is to play the pebble game on q and $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ instead, which can be shown to give the correct result. However, $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ need not be finite. There is a way to compute in polynomial time a finite representation of $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ [32], but using that representation in place of $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ requires to rewrite q in a way that might increase the tree width and as a consequence there is no guarantee that the resulting game delivers the correct result. We thus start by giving a novel characterization of answers to OMQs from $(\mathcal{ELH}^{dr}_\bot, \mathrm{CQ})$ that is tailored towards being verified by existential pebble games.

A. Characterization of OMQ Answers

We require several definitions and preliminaries.

Let q be a CQ. A database $\mathcal{D}$ is a *ditree* if the directed graph $(\mathsf{dom}(\mathcal{D}), \{(a, b) \mid r(a, b) \in \mathcal{D}\})$ is a tree. Note that multi-edges are admitted while reflexive loops are not. We say that q *is a homomorphic preimage of a ditree* if there is a homomorphism from q to a ditree database $\mathcal{D}$. Consider the

CQ q' obtained from q by exhaustively identifying variables x_1 and x_2 whenever there are atoms $r(x_1, y)$ and $s(x_2, y)$. It can be verified that q is a homomorphic preimage of a ditree if and only if $\mathcal{D}_{q'}$ is a ditree. This also means that it is decidable in PTIME whether a given q is a homomorphic preimage of a ditree. If this is the case, then $\mathcal{D}_{q'}$ is *initial* among all ditrees $\mathcal{D}$ that q is a homomorphic preimage of, that is, $\mathcal{D}_{q'}$ admits a homomorphism to any such $\mathcal{D}$. We use $\mathsf{dtree}_q(x_0)$ to denote $\mathcal{D}_{q'}$ viewed as a CQ, constants corresponding to variables, in which the root constant is the only answer variable x_0 and all other variables are quantified. If q is not a homomorphic preimage of a ditree, then dtree_q is undefined.

A pair of variables x, y from q is *guarded* if they are linked by an edge in the Gaifman graph of $\mathcal{D}_q$. Let G_2^q be the set of all guarded pairs of variables from q. For every $(x, y) \in G_2^q$ with y quantified and for every $i \geq 0$, define $\mathsf{reach}^i(x, y)$ to be the smallest set such that

1) $x \in \mathsf{reach}^0(x, y)$ and $y \in \mathsf{reach}^1(x, y)$;
2) if $z \in \mathsf{reach}^i(x, y)$, $i > 0$, and $r(z, u) \in q$, then $u \in \mathsf{reach}^{i+1}(x, y)$;
3) if $y \in \mathsf{reach}^{i+1}(x, y)$ and $r(z, y) \in q$, then $z \in \mathsf{reach}^i(x, y)$.

Moreover, $\mathsf{reach}(x, y) = \bigcup_i \mathsf{reach}^i(x, y)$. A guarded pair (x, y) is $\exists$-*eligible* if $q|_{\mathsf{reach}(x,y)}$ is a homomorphic preimage of a ditree. We use $\mathsf{dtree}_{(x,y)}$ as a shorthand for $\mathsf{dtree}_{q|_{\mathsf{reach}(x,y)}}$.

Informally, (x, y) being $\exists$-eligible means that in a homomorphism h from q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$, for some database $\mathcal{D}$ and some $\mathcal{ELH}_\bot^{dr}$-ontology $\mathcal{O}$, atoms $r(x, y) \in q$ can 'cross the boundary' between $\mathcal{D}$ and the part of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ generated by the chase in the sense that $h(x) \in \mathsf{dom}(\mathcal{D})$ and $h(y)$ is mapped to a constant that was introduced by the chase. Note that the chase generates only structures that are ditrees.

Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ be an OMQ from $(\mathcal{ELH}_\bot^{dr}, \mathrm{CQ})$ and let $\mathcal{D}$ be an $\mathbf{S}$-database that is consistent with $\mathcal{O}$. We now define the central notion underlying the announced characterization, called $\mathcal{D}$-labeling of q, which (partially) represents a homomorphism from the CQ q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$.

An $\exists$-*MCC* is a subquery $p \subseteq q$ that constitutes a maximally connected component of q and contains only quantified variables. For an $\mathbf{S}$-database $\mathcal{D}$, we use $\mathsf{ch}^-_{\mathcal{O}}(\mathcal{D})$ to denote the restriction of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ to the constants in $\mathsf{dom}(\mathcal{D})$. A $\mathcal{D}$-*labeling* of q is a function $\ell : \mathsf{var}(q) \to \mathsf{dom}(\mathcal{D}) \cup \{\exists\} \cup (G_2^q \times \mathsf{dom}(\mathcal{D}))\}$ such that the following conditions are satisfied:

1) $\ell(x) \in \mathsf{dom}(\mathcal{D})$ for every answer variable x;
2) the restriction of ℓ to the variables in $V := \{x \mid \ell(x) \in \mathsf{dom}(\mathcal{D})\}$ is a homomorphism from $q|_V$ to $\mathsf{ch}^-_{\mathcal{O}}(\mathcal{D})$;
3) if $r(x, y) \in q$ and $\ell(y) \in \mathsf{dom}(\mathcal{D})$, then $\ell(x) \in \mathsf{dom}(\mathcal{D})$;
4) if $(x, y) \in G_2^q$, $\ell(x) \in \mathsf{dom}(\mathcal{D})$, $\ell(y) \notin \mathsf{dom}(\mathcal{D})$, then
   a) (x, y) is $\exists$-eligible,
   b) $\mathcal{D} \models (\mathcal{O}, \mathbf{S}, \mathsf{dtree}_{(x,y)})(\ell(x))$, and
   c) $\ell(y) = ((x', y'), \ell(x))$ where $x' \in \mathsf{reach}^0(x, y)$ and $y' \in \mathsf{reach}^1(x, y)$;
5) if $r(x, y) \in q$ and $\ell(x) = ((x', y'), a)$, then $\ell(y) = \ell(x)$;
6) if $r(x, y) \in q$, $\ell(y) = ((x', y'), a)$, and $y \notin \mathsf{reach}^0(x', y')$, then $\ell(x) = \ell(y)$;
7) if $r(x, y) \in q$, $\ell(y) = ((x', y'), a)$, and $y \in \mathsf{reach}^0(x', y')$, then $\ell(x) = a$;
8) if q' is an $\exists$-MCC of q such that $\ell(x) \notin \mathsf{dom}(\mathcal{D})$ for every variable x in q', then q' is a homomorphic preimage of a ditree and $\mathcal{D} \models (\mathcal{O}, \mathbf{S}, \exists x_0\, \mathsf{dtree}_{q'})$.

A $\mathcal{D}$-labeling of q represents a homomorphism h from q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ with the following conditions. If $\ell(z) = a \in \mathsf{dom}(\mathcal{D})$, then $h(z) = a$. If $\ell(z) = \exists$, then z is in an $\exists$-MCC of q and mapped to a constant generated by the chase. Finally, if $\ell(z) = ((x, y), a)$, then (the same is true or) $h(x) = a \in \mathsf{dom}(\mathcal{D})$, $h(y)$ is a constant generated by the chase, and $h(z)$ is in the tree-shaped sub-database of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ rooted at $h(y)$.

Lemma 9. *For every* $\mathbf{a} \in \mathsf{dom}(\mathcal{D})^{|\mathbf{x}|}$, $\mathcal{D} \models Q(\mathbf{a})$ *iff there is a* $\mathcal{D}$*-labeling* ℓ *of* $q(\mathbf{x})$ *such that* $\ell(\mathbf{x}) = \mathbf{a}$*.*

Note that Conditions 1 to 8 can all be verified in polynomial time, essentially because the evaluation of OMQs from $(\mathcal{ELH}_\bot^{dr}, \mathrm{CQ})$ is in PTIME when the actual CQ is tree-shaped; this is implicit in [32], see also [12].

Lemma 10. *For an OMQ* $Q = (\mathcal{O}, \mathbf{S}, q)$ *from* $(\mathcal{ELH}_\bot^{dr}, CQ)$*, an* $\mathbf{S}$*-database* $\mathcal{D}$*, and a mapping* $\ell : \mathsf{var}(q) \to \mathsf{dom}(\mathcal{D}) \cup \{\exists\} \cup (G_2^q \times \mathsf{dom}(\mathcal{D}))\}$*, the problem of deciding whether* ℓ *is a* $\mathcal{D}$*-labeling of* q *is in* PTIME*.*

B. Existential Pebble Games

We now describe the polynomial time algorithm for evaluating OMQs from $(\mathcal{ELH}_\bot^{dr}, \mathrm{CQ})^{\equiv}_{\mathrm{UCQ}_k}$ based on the full schema, first recalling the existential $k+1$-pebble games from [18], in a form that does not make pebbles explicit. The game is played between two players, Spoiler and Duplicator, on a CQ $q(\mathbf{x})$, a database $\mathcal{D}$, and a candidate answer $\mathbf{a}$. The positions are pairs (V, h) that consist of a set V of quantified variables from q of size at most $k+1$ and a mapping $h : V \cup \mathbf{x} \to \mathsf{dom}(\mathcal{D})$ such that $h(\mathbf{x}) = \mathbf{a}$. The initial position is $(\emptyset, \emptyset)$. In each round of the game, Spoiler chooses a new set V of size at most $k+1$. Then Duplicator chooses a new mapping $h : V \to \mathsf{dom}(\mathcal{D})$ such that if (V', h') was the previous position, then $h(x) = h'(x)$ for all $x \in V \cap V'$. Spoiler wins when h is not a homomorphism from $q|_V$ to $\mathcal{D}$. Duplicator wins if she has a winning strategy, that is, if she can play forever without Spoiler ever winning. It is known that when q is of tree width bounded by k, then Duplicator has a winning strategy if and only if there is a homomorphism from q to $\mathcal{D}$. This remains true if q is equivalent to a CQ of tree width bounded by k. The existence of a winning strategy for Duplicator can be decided in polynomial time by a straightforward elimination procedure: start with the set of all positions, exhaustively eliminate those from which Duplicator loses in one round, and then check whether $(\emptyset, \emptyset)$ has survived.

Now let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ be an OMQ from $(\mathcal{ELH}_\bot^{dr}, \mathrm{CQ})$, $\mathcal{D}$ an $\mathbf{S}$-database, and $\mathbf{a}$ a candidate answer. To decide whether $\mathcal{D} \models Q(\mathbf{a})$, we can assume that $\mathcal{D}$ is consistent with $\mathcal{O}$ since this property is decidable in polynomial time (implicit in [32]) and the result is clear on inconsistent databases. By Lemma 9, it suffices to find a $\mathcal{D}$-labeling of q. This is achieved by a version of the existential $k+1$-pebble game in which positions take the form (V,ℓ) where V is as before and ℓ is a mapping from $V \cup \mathbf{x}$ to $\mathsf{dom}(\mathcal{D}) \cup \{\exists\} \cup (G^q_2(q) \times \mathsf{dom}(\mathcal{D}))$ such that $\ell(\mathbf{x}) = \mathbf{a}$. The moves of Spoiler and Duplicator are as before. Spoiler wins if ℓ is not a $\mathcal{D}$-labeling of $q|_V$ and the winning condition for Duplicator remains unchanged. The existence of a winning strategy for Duplicator can be decided in polynomial time by the same elimination procedure because, by Lemma 10, it can be decided in polynomial time whether a given mapping ℓ is a $\mathcal{D}$-labeling. The following can be proved in the same way as without ontologies, see [18], [25].

Lemma 11. *If q is of tree width at most k, then Duplicator has a winning strategy if and only if there is a $\mathcal{D}$-labeling of q.*

The remaining obstacle on the way to prove Theorem 9 is that q needs not be of tree width k. It would be convenient to play on a full rewriting of q instead, which by Theorem 3 is of tree width bounded by k. However, we have no way of computing a full rewriting in PTIME. The solution is to first extend q with additional atoms as in the second step of the construction of rewritings in Section III and to then play on the resulting CQ q'. It can be shown that this gives the correct result because when $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_f)$ is a full rewriting of Q, then q_f must syntactically be a subquery of q'.

VI. Deciding Semantic Tree-Likeness

We study the complexity of deciding whether a given OMQ is UCQ$_k$-equivalent. Apart from $\mathcal{ELHI}_\bot$ and its fragments introduced in Section II, we also consider the additional fragments DL-Lite$^\mathcal{R}$ and DL-Lite$^\mathcal{R}_{\mathsf{horn}}$, which are prominent in ontology-based data integration [2], [15].

A *basic concept* is a concept name or of one of the forms $\top$, $\bot$, $\exists r.\top$, and $\exists r^-.\top$. A *DL-Lite$^\mathcal{R}_{\mathsf{horn}}$-ontology* is a finite set of statements of the form

$$B_1 \sqcap \dots \sqcap B_n \sqsubseteq B \quad r \sqsubseteq s \quad r \sqsubseteq s^- \quad r_1 \sqcap \dots \sqcap r_n \sqsubseteq \bot$$

where $B_1, \dots, B_n, B$ range over basic concepts and $r, s, r_1, \dots, r_n$ range over role names. A *DL-Lite$^\mathcal{R}$-ontology $\mathcal{O}$* is a *DL-Lite$^\mathcal{R}_{\mathsf{horn}}$-ontology* such that whenever $B_1 \sqcap \dots \sqcap B_n \sqsubseteq B \in \mathcal{O}$, then $n = 1$ or $B = \bot$.

A. Non-full Schema

We first concentrate on the case where the schema is non-full. The next result provides lower bounds.

Theorem 10. *For any $k \geq 1$, UCQ$_k$-equivalence is*
1) ExpTime*-hard in $(\mathcal{EL}, CQ)$;*
2) 2ExpTime*-hard in $(\mathcal{ELI}, CQ)$;*
3) Π^p_2*-hard in (DL-Lite$^\mathcal{R}$, CQ).*

The same lower bounds apply to CQ$_k$-equivalence, both while preserving the ontology and in the general case.

Point 1 is proved by reduction from the problem whether a given OMQ $(\mathcal{O}, \mathbf{S}, A(x))$ from $(\mathcal{EL}_\bot, \text{CQ})$ is empty (as defined in Section II), which is known to be ExpTime-hard [3]. Point 2 is shown by reducing the word problem of an exponentially space bounded alternating Turing machine M, with work alphabet of size at least $k+1$, to the complement of (U)CQ$_k$-equivalence. Our reduction carefully makes use of a construction from [7], where it is shown that containment in $(\mathcal{ELI}, \text{CQ})$ is 2ExpTime-hard. For Point 3 we provide a reduction from $\forall\exists$-QBF, building on and extending an NP-hardness proof for the combined complexity of (a restricted version of) query evaluation in (DL-Lite$^\mathcal{R}$, CQ) on databases of the form $\{A(a)\}$ given in [28].

The next result establishes matching upper bounds.

Theorem 11. *For any $k \geq 1$, UCQ$_k$-equivalence is*
1) *in* ExpTime *in $(\mathcal{ELH}^{dr}_\bot, UCQ)$;*
2) *in* 2ExpTime *in $(\mathcal{ELHI}_\bot, UCQ)$;*
3) *in Π^p_2 in (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$, UCQ).*

The proof rests on Corollary 1, that is, we compute the UCQ$_k$-approximation Q_a of the input OMQ Q and check whether $Q \subseteq Q_a$ (the converse holds unconditionally). It was shown in [7] that OMQ containment is ExpTime-complete in $(\mathcal{ELH}_\bot, \text{CQ})$ and 2ExpTime-complete in $(\mathcal{ELHI}_\bot, \text{CQ})$ and in [10]. that OMQ-containment is Π^p_2-complete in (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$, CQ). These results extend to the OMQ languages in Points 1 and 2 of Theorem 11. We show that with a bit of care we can obtain the same overall complexities despite the fact that the UCQ in Q_a consisting of exponentially many (polynomial size) CQs.

B. Full Schema

We now focus on the special case of the full schema, where the complexity of deciding semantic tree-likeness turns out to be identical to that of query evaluation.

Theorem 12. *For any $k \geq 1$, and OMQs based on the full schema, (U)CQ$_k$-equivalence is complete for*
1) NP *between $(\mathcal{EL}, CQ)$ and $(\mathcal{ELH}^{dr}_\bot, UCQ)$;*
2) ExpTime *between $(\mathcal{ELI}, CQ)$ and $(\mathcal{ELHI}_\bot, UCQ)$;*
3) NP *between (DL-Lite$^\mathcal{R}$, CQ) and (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$).*

The NP lower bounds are inherited from the case where the ontology is empty [18], while the ExpTime lower bound is proved by a reduction from the subsumption problem in $\mathcal{ELI}$ [5]. The upper bounds rest on Theorem 3, that is, given an input OMQ $Q = (\mathcal{O}, \mathbf{S}, q)$ we first extend q to a CQ q' based on $\mathcal{O}$ as in the second step of the construction of retracts, then guess a subquery q'' of q', and finally check whether $Q \subseteq (\mathcal{O}, \mathbf{S}, q'')$. This yields the upper bounds in Theorem 12 as containment of two OMQs that share the same ontology and are based on the full schema trivially reduces to OMQ evaluation, which is of the stated complexities [28], [32], [35].

VII. Dealing with Functional Roles

Until now we considered description logics that do not admit functional roles, an important feature for ontology modeling. Here we take a first look, focussing on DL-Lite$^\mathcal{F}$ as a basic yet prominent such DL [15]. While a main observation is that functional roles result in serious technical complications for tree widths exceeding one, we are also able to obtain some interesting initial results. Throughout the section, we focus on Boolean CQs (BCQs).

Recall that a *basic concept* is a concept name or of one of the forms $\top$, $\bot$, $\exists r.\top$, and $\exists r^-.\top$. A *DL-Lite$^{\mathcal{F}}$-ontology* is a finite set of statements of the form

$$B_1 \sqsubseteq B_2 \quad B_1 \sqcap \cdots \sqcap B_n \sqsubseteq \bot \quad r_1 \sqcap \cdots \sqcap r_n \sqsubseteq \bot \quad (\mathsf{funct}\ r)$$

where $B_1, \dots, B_n$ range over basic concepts and $r_1, \dots, r_n, r$ range over role names. An interpretation *satisfies* $(\mathsf{funct}\ r)$ if whenever $(d, e_1) \in r^{\mathcal{I}}$ and $(d, e_2) \in r^{\mathcal{I}}$, then $e_1 = e_2$.

In the DL-Lite$^{\mathcal{F}}$ case, our problems are tightly related to the semantic tree-likeness of BCQs over relational databases in the presence of key dependencies studied by Figueira [23]. We argue in the appendix that the results of [23] can easily be generalized to unions of BCQs (UBCQ). This entails an interesting statement about DL-Lite$^{\mathcal{F}}_{=}$, the fragment of DL-Lite$^{\mathcal{F}}$ that admits only functionality assertions, but no inclusions of any kind.

Theorem 13 (Figueira)**.** *For OMQs from (DL-Lite$^{\mathcal{F}}_{=}$, UBCQ) based on the full schema, UBCQ$_k$-equivalence while preserving the ontology is in* 2ExpTime, *for any $k \geq 1$. Moreover, an equivalent OMQ from (DL-Lite$^{\mathcal{F}}_{=}$, UBCQ$_k$) can be constructed in double exponential time (if it exists).*

The above result relies on a sophisticated argument based on tree walking automata. It is open whether the 2ExpTime upper bound is optimal. The best known lower bound is NP, from the case without functionality assertions [18].

We now use Theorem 13 to obtain results for DL-Lite$^{\mathcal{F}}$. Let $(\text{DL-Lite}^{\mathcal{F}}, \text{UBCQ})^{\equiv,po}_{\text{UBCQ}_k}$ denote the class of OMQs from (DL-Lite$^{\mathcal{F}}$, UBCQ) that are UBCQ$_k$-equivalent while preserving the ontology.

Theorem 14. *For any $k \geq 1$,*

1) p-Evaluation$((\textit{DL-Lite}^{\mathcal{F}}, \textit{UBCQ})^{\equiv,po}_{\textit{UBCQ}_k})$ *based on the full schema is in* FPT.
2) *For OMQs from (DL-Lite$^{\mathcal{F}}$, UBCQ) based on the full schema, UBCQ$_k$-equivalence while preserving the ontology is in* 3ExpTime.

The proof of Theorem 14 uses first-order rewritability. For a DL-Lite$^{\mathcal{F}}$-ontology $\mathcal{O}$, let $\mathcal{O}^{=}$ be the set of functionality assertions in $\mathcal{O}$. We show that every OMQ Q from (DL-Lite$^{\mathcal{F}}$, UBCQ) can be rewritten into a UBCQ $\mathsf{rew}(Q)$ such that, for every database $\mathcal{D}$ that satisfies $\mathcal{O}^{=}$, $Q(\mathcal{D}) = \mathsf{rew}(Q)(\mathcal{D})$. Further, our rewriting procedure preserves the tree width, that is, for every $Q \in (\text{DL-Lite}^{\mathcal{F}}, \text{UBCQ}_k)$, $\mathsf{rew}(Q) \in \text{UBCQ}_k$. We obtain the following.

Lemma 12. *Fix $k \geq 1$. For an OMQ $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q) \in$ (DL-Lite$^{\mathcal{F}}$, UBCQ) the following are equivalent:*

1) *Q is UBCQ$_k$-equivalent while preserving the ontology;*
2) *$(\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q))$ is UBCQ$_k$-equivalent while preserving the ontology.*

Since the UBCQ $\mathsf{rew}(Q)$ can be constructed in time single exponential in the size of Q [15], Theorem 14 is a consequence of Theorem 13 and Lemma 12.

As already observed in [6], [23], the case $k = 1$, is more well-behaved than the general case. The main reason is the fact that the chase procedure for DL-Lite$^{\mathcal{F}}_{=}$, which identifies terms according to the functionality assertions in the ontology, preserves tree width 1 in the sense that when a database $\mathcal{D}$ is of tree width 1, then so is $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$. This fails for tree widths larger than 1. It has, in fact, been observed by Figueira (Lemma 4.3 in [23]) that when starting with a database of tree width $k > 1$, then the chase can arbitrarily increase the tree width even if $\mathcal{O}$ consists only of a single functionality assertion.

By exploiting the above property, we can strengthen Theorem 14 for the case $k = 1$, using an approach that does not rely on Theorem 13. In fact, we show that every OMQ $Q = (\mathcal{O}, \mathbf{S}, q)$ from (DL-Lite$^{\mathcal{F}}$, UBCQ) can be rewritten in polynomial time into an OMQ $Q' = (\mathcal{O}^{\sqsubseteq}, \mathbf{S}, q')$, where $\mathcal{O}^{\sqsubseteq}$ denotes the result of removing from $\mathcal{O}$ all functionality assertions, such that Q is UBCQ$_1$-equivalent iff Q' is; the proof of the latter again involves first-order rewritability. This allows us to apply results from previous sections, in particular Corollary 1, Theorem 8, and Theorem 12.

Theorem 15.

1) *In (DL-Lite$^{\mathcal{F}}$, UBCQ), UBCQ$_1$-equivalence coincides with UBCQ$_1$-equivalence while preserving the ontology.*
2) Evaluation$((\textit{DL-Lite}^{\mathcal{F}}, \textit{UBCQ})^{\equiv}_{\textit{UBCQ}_1})$ *based on the full schema is in* PTime.
3) *For OMQs from (DL-Lite$^{\mathcal{F}}$, UBCQ) based on the full schema, UBCQ$_1$-equivalence is* NP*-complete.*

The upper bounds in Theorems 14 and 15 extend to DL-Lite$^{\mathcal{F}}_{\mathsf{horn}}$ in a straightforward way, that is, to the extension of DL-Lite$^{\mathcal{F}}$ with statements of the form $B_1 \sqcap \cdots \sqcap B_n \sqsubseteq B$. Moreover, Theorem 15 extends to the non-Boolean case.

VIII. Conclusion

An intriguing open problem that emerges from this paper is whether Theorem 8 can be generalized to the case where the schema is not required to be full, that is, whether OMQ evaluation is in PTime in $(\mathcal{EL}, \text{CQ})^{\equiv}_{\text{UCQ}_k}$ and modest extensions thereof such as $(\mathcal{ELH}^{dr}_{\bot}, \text{UCQ})^{\equiv}_{\text{UCQ}_k}$. Note that the companion Theorem 4 does not assume the full schema. Related to this seems to be the problem whether CQ$_k$-equivalence coincides with UCQ$_k$-equivalence in $(\mathcal{EL}, \text{CQ})$; recall that this is not the case in $(\mathcal{ELI}, \text{CQ})$ by Proposition 4.

Being a bit more adventurous, one could be interested in classifying PTime combined complexity within $(\mathcal{ELI}, \text{CQ})$ and related languages, and in classifying PTime combined complexity and FPT for DLs with functional roles, existential rule languages such as (frontier-)guarded rules, and DLs that include negation and disjunction such as $\mathcal{ALC}$ and $\mathcal{SHIQ}$.

Acknowledgments. We thank the anonymous reviewers for their helpful comments. This research was supported by ERC consolidator grant 647289 CODA, by EPSRC grant EP/S003800/1 EQUID, by the Millennium Institute for Foundational Research on Data, and Fondecyt grant 1170109.

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

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

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

[4] Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[5] Franz Baader, Carsten Lutz, and Sebastian Brandt. Pushing the $\mathcal{EL}$ envelope further. In *Proc. of OWLED*, volume 496 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[6] Pablo Barceló, Georg Gottlob, and Andreas Pieris. Semantic acyclicity under constraints. In *PODS*, pages 343–354, 2016.

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

[8] Meghyn Bienvenu, Stanislav Kikot, Roman Kontchakov, Vladimir V. Podolskii, Vladislav Ryzhikov, and Michael Zakharyaschev. The complexity of ontology-based data access with OWL 2 QL and bounded treewidth queries. In *Proc. of PODS*, pages 201–216. ACM, 2017.

[9] Meghyn Bienvenu, Stanislav Kikot, Roman Kontchakov, Vladimir V. Podolskii, and Michael Zakharyaschev. Ontology-mediated queries: Combined complexity and succinctness of rewritings via circuit complexity. *J. ACM*, 65(5):28:1–28:51, 2018.

[10] Meghyn Bienvenu, Carsten Lutz, and Frank Wolter. Query containment in description logics reconsidered. In *Proc. of KR*. AAAI Press, 2012.

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

[12] Meghyn Bienvenu, Magdalena Ortiz, Mantas Simkus, and Guohui Xiao. Tractable queries for lightweight description logics. In *Proc. of IJCAI*, pages 768–774. IJCAI/AAAI, 2013.

[13] 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.

[14] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, and Riccardo Rosati. Ontologies and databases: The DL-Lite approach. In *Proc. of Reasoning Web*, pages 255–356, 2009.

[15] 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. Autom. Reasoning*, 39(3):385–429, 2007.

[16] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Data complexity of query answering in description logics. *Artif. Intell.*, 195:335–360, 2013.

[17] Hubie Chen. On the complexity of existential positive queries. *ACM Trans. Comput. Log.*, 15(1):9:1–9:20, 2014.

[18] Víctor Dalmau, Phokion G. Kolaitis, and Moshe Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *CP*, pages 310–326, 2002.

[19] Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness II: on completeness for W[1]. *Theor. Comput. Sci.*, 141(1&2):109–131, 1995.

[20] Thomas Eiter, Georg Gottlob, Magdalena Ortiz, and Mantas Simkus. Query answering in the description logic Horn-$\mathcal{SHIQ}$. In *Proc. of JELIA*, volume 5293 of *LNCS*, pages 166–179. Springer, 2008.

[21] Thomas Eiter, Carsten Lutz, Magdalena Ortiz, and Mantas Simkus. Query answering in description logics with transitive roles. In *Proc. of IJCAI*, pages 759–764, 2009.

[22] Cristina Feier, David Carral, Giorgio Stefanoni, Bernardo Cuenca Grau, and Ian Horrocks. The combined approach to query answering beyond the owl 2 profiles. In *Proc. of IJCAI*, pages 2971–2977, 2015.

[23] Diego Figueira. Semantically acyclic conjunctive queries under functional dependencies. In *Proc. of LICS*, pages 847–856, 2016.

[24] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006.

[25] Martin Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1:1–1:24, 2007.

[26] Pavol Hell and Jaroslav Nesetril. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[27] 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.

[28] Stanislav Kikot, Roman Kontchakov, and Michael Zakharyaschev. On (in)tractability of OBDA with OWL 2 QL. In *Proc. of DL2011*, volume 745 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

[29] Adila Krisnadhi and Carsten Lutz. Data complexity in the $\mathcal{EL}$ family of description logics. In *Proc. of LPAR*, volume 4790 of *LNCS*, pages 333–347. Springer, 2007.

[30] Alon Y Levy, Alberto O Mendelzon, Yehoshua Sagiv, and Divesh Srivastava. Answering queries using views. In *Proc of PODS*, pages 95–104, 1995.

[31] Carsten Lutz. The complexity of conjunctive query answering in expressive description logics. In *Proc. of IJCAR*, volume 5195 of *LNCS*, pages 179–193. Springer, 2008.

[32] Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In *Proc. of IJCAI09*. AAAI Press, 2009.

[33] David Maier, Alberto O. Mendelzon, and Yehoshua Sagiv. Testing implications of data dependencies. *ACM Trans. Database Syst.*, 4(4):455–469, 1979.

[34] Nhung Ngo, Magdalena Ortiz, and Mantas Simkus. Closed predicates in description logics: Results on combined complexity. In *Proc. of KR*, pages 237–246. AAAI Press, 2016.

[35] Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In *Proc. of KR*. AAAI Press, 2010.

[36] W3C OWL Working Group. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 2009. Available at http://www.w3.org/TR/owl2-overview/.

[37] Christos H. Papadimitriou and Mihalis Yannakakis. On the complexity of database queries. *J. Comput. Syst. Sci.*, 58(3):407–427, 1999.

[38] Neil Robertson and Paul D. Seymour. Graph minors. V. excluding a planar graph. *J. Comb. Theory, Ser. B*, 41(1):92–114, 1986.

[39] Yehoshua Sagiv and Mihalis Yannakakis. Equivalences among relational expressions with the union and difference operators. *J. ACM*, 27(4):633–655, 1980.

[40] Kent A. Spackman, Keith E. Campbell, and Roger A. Côté. SNOMED RT: a reference terminology for health care. In *Proc. of AMIA*. AMIA, 1997.

APPENDIX

TECHNICAL LEMMAS AND CONSTRUCTIONS

We give some fundamental lemmas and technical constructions used in proofs throughout the paper. The following facts about homomorphisms, databases, and OMQs are well-known, see for example [13].

Lemma 13. *Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be databases, h a homomorphism from $\mathcal{D}_1$ to $\mathcal{D}_2$, and $Q = (\mathcal{O}, \mathbf{S}, q)$ an OMQ from $(\mathcal{ELHI}_\bot, UCQ)$. Then*

1) *if $\mathcal{D}_2$ is consistent with $\mathcal{O}$, then so is $\mathcal{D}_1$;*
2) *if $\mathcal{D}_1 \models Q(\mathbf{a})$, then $\mathcal{D}_2 \models Q(h(\mathbf{a}))$ for all tuples $\mathbf{a}$ over $\mathsf{dom}(\mathcal{D}_1)$.*

The chase. We next introduce a version of the well-known chase procedure, see for example [33]. The constants in $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ that are not in $\mathcal{D}$ are called *fresh*.

Let $\mathcal{O}$ be an $\mathcal{ELHI}_\bot$-ontology. Consider the following two chase rules that extend an interpretation $\mathcal{I}$ based on the inclusions in $\mathcal{O}$:

1) if $a \in C^\mathcal{I}$, $C \sqsubseteq D \in \mathcal{O}$, and $D \neq \bot$, then add $D(a)$ to $\mathcal{I}$;
2) if $(a, b) \in r^\mathcal{I}$ and $r \sqsubseteq s \in \mathcal{O}$, then add $s(a, b)$ to $\mathcal{I}$.

In Rule 1, 'add $D(a)$ to $\mathcal{I}$' means to add to $\mathcal{I}$ a finite database that represents the $\mathcal{ELI}$-concept D, identifying its root with a. For example, the concept $A \sqcap \exists r.(B \sqcap \exists s.\top)$ corresponds to the database $\{A(a), r(a, b), B(b), s(b, c)\}$.

Let $\mathcal{D}$ be a database. The *chase of $\mathcal{D}$ with $\mathcal{O}$*, denoted $\mathsf{ch}_\mathcal{O}(\mathcal{D})$, is the potentially infinite interpretation that is obtained as the limit of any sequence $\mathcal{I}_0, \mathcal{I}_1, \dots$, where

- $\mathcal{I}_0 = \mathcal{D}$,
- $\mathcal{I}_{i+1}$ is obtained from $\mathcal{I}_i$ by applying both chase rules in all possible ways, and
- rule application is *fair*, which intuitively means that if an assertion $C \sqsubseteq D$ with $D \neq \bot$, or $r \sqsubseteq s$ in $\mathcal{O}$ is not satisfied at some point during the construction of the chase, then eventually it will be satisfied.

Since our chase is *oblivious*, that is, Rule 1 adds a $D(a)$ to $\mathcal{I}_i$ even if $a \in D^{\mathcal{I}_i}$, the result of all these sequences is identical up to isomorphism.

Containment under full schema. The following lemma characterizes containment between OMQs based on UCQs and the full schema. It is an extension of a classic lemma for containment of UCQs [39].

Lemma 14. *Let $Q_i = (\mathcal{O}_i, \mathbf{S}_{\mathsf{full}}, q_i) \in (\mathcal{EL}_\bot, UCQ)$, $i \in \{1, 2\}$. Then $Q_1 \subseteq Q_2$ iff for every CQ q_1' in q_1 such that $\mathcal{D}_{q_1'}$ is consistent with $\mathcal{O}_1$, there is a CQ q_2' in q_2 $q_2' \to \mathsf{ch}_{\mathcal{O}_2}(q_1')$.*

Unravelings. We next describe the unraveling of a database into a (potentially infinite) database of bounded tree width. Let $\mathcal{D}$ be a database and $k \geq 1$. The *k-unraveling of $\mathcal{D}$* is obtained as the limit of a (potentially) infinite sequence of databases $\mathcal{D}_0, \mathcal{D}_1, \dots$. Along with this sequence, we define additional sequences $(V_0, E_0, \mu_0), (V_1, E_1, \mu_1), \dots$ and $\pi_0, \pi_1, \dots$ such that (V_i, E_i, μ_i) is a tree decomposition of $\mathcal{D}_i$ of width at most k and π_i is a function from $\mathsf{dom}(\mathcal{D}_i) \to \mathsf{dom}(\mathcal{D})$ such that for each $v \in V_i$ we have that $\pi_i|_{\mu_i(v)}$ is an isomorphism between $\mathcal{D}_i|_{\mu_i(v)}$ and an induced subdatabase of $\mathcal{D}$. The initial database $\mathcal{D}_0$ is empty and V_0 consists of a single vertex v with $\mu_0(v) = \emptyset$. For each $i \geq 0$, we obtain $\mathcal{D}_{i+1}$, $(V_{i+1}, E_{i+1}, \mu_{i+1})$, and π_{i+1} by extending $\mathcal{D}_i$, (V_i, E_i, μ_i), and π_i as follows: for every $v \in V_i$ that is a leaf in the tree (V_i, E_i) and every non-empty subdatabase $\mathcal{D}'$ of $\mathcal{D}$ with $\mathsf{dom}(\mathcal{D}') \leq k + 1$, add a successor v' of v; as $\mu_{i+1}(v')$, use the isomorphic copy of $\mathcal{D}'$ obtained by replacing $a \in \mathsf{dom}(\mathcal{D}')$ with $b \in \mu_i(v)$ if $\pi_i(b) = a$ and with a fresh constant otherwise. These renamings also define π_{i+1}, in the obvious way. Let $\mathcal{D}^u$, (V, E, μ), and π be the limits of the constructed sequences. It is clear that (V, E, μ) is an (infinite) tree decomposition of $\mathcal{D}$ of width at most k.

For a tuple $\mathbf{a}$ over $\mathsf{dom}(\mathcal{D})$, we define the *$k$-unraveling of $\mathcal{D}$ up to $\mathbf{a}$*, denoted $\mathcal{D}^u_\mathbf{a}$, to be the database obtained by starting with the unraveling $\mathcal{D}|^u_{\overline{\mathbf{a}}}$ of the restriction $\mathcal{D}|_{\overline{\mathbf{a}}}$ of $\mathcal{D}$ to those facts that do not involve a constant from $\mathbf{a}$ and then adding the following facts:

1) all facts from $\mathcal{D}$ that involve only constants from $\mathbf{a}$;
2) for every $r(a, b) \in \mathcal{D}$ with $\{a, b\} \cap \mathbf{a} = \{a\}$ and every $c \in \mathsf{dom}(\mathcal{D}|^u_{\overline{\mathbf{a}}})$ with $\pi(c) = b$, the fact $r(a, c)$;
3) for every $r(b, a) \in \mathcal{D}$ with $\{a, b\} \cap \mathbf{a} = \{a\}$ and every $c \in \mathsf{dom}(\mathcal{D}|^u_{\overline{\mathbf{a}}})$ with $\pi(c) = b$, the fact $r(c, a)$.

The following lemma summarizes the main properties of k-unravelings.

Lemma 15. *Let $\mathcal{D}$ be a database, $k \geq 1$, $\mathbf{a} \in \mathsf{dom}(\mathcal{D})^n$, and $\mathcal{D}^u_\mathbf{a}$ be the k-unraveling of $\mathcal{D}$ up to $\mathbf{a}$. Then*

1) *$\mathcal{D}^u_\mathbf{a} \to \mathcal{D}$ via a homomorphism that is the identity on $\mathbf{a}$;*
2) *$\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a}) \to \mathsf{ch}_\mathcal{O}(\mathcal{D})$ via a homomorphism that is the identity on $\mathbf{a}$, for all $\mathcal{ELHI}_\bot$-ontologies $\mathcal{O}$;*
3) *$\mathcal{D}$ is consistent with $\mathcal{O}$ iff $\mathcal{D}^u_\mathbf{a}$ is consistent with $\mathcal{O}$, for all $\mathcal{ELHI}_\bot$-ontologies $\mathcal{O}$;*
4) *$\mathcal{D} \models Q(\mathbf{a})$ iff $\mathcal{D}^u_\mathbf{a} \models Q(\mathbf{a})$, for all OMQs Q from $(\mathcal{ELHI}_\bot, UCQ_k)$.*

Proof. (1) The function $\pi : \mathsf{dom}(\mathcal{D}|^u_{\overline{\mathbf{a}}}) \to \mathsf{dom}(\mathcal{D}|_{\overline{\mathbf{a}}})$ introduced during the construction of $\mathcal{D}|^u_{\overline{\mathbf{a}}}$ is a homomorphism from $\mathcal{D}|^u_{\overline{\mathbf{a}}}$ to $\mathcal{D}|_{\overline{\mathbf{a}}}$. We can extend π to a homomorphism π' from $\mathcal{D}^u_\mathbf{a}$ to $\mathcal{D}$ by setting $\pi(a) = a$ for all $a \in \mathbf{a}$.

(2) We argue that π can be extended to a homomorphism f from $\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a})$ to $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ that is the identity on $\mathbf{a}$. For every $b \in \mathsf{dom}(\mathcal{D}^u_\mathbf{a})$, let $(\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a}))_b$ be the subdatabase of $\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a})$ induced by the constants from $\mathsf{dom}(\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a}))$ which are reachable from b in the Gaifman graph of $\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a})$ and are not from $\mathsf{dom}(\mathcal{D}^u_\mathbf{a})$ (except for b). That is, $(\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a}))_b$ is the anonymous part of the chase that 'starts' at b. To prepare for the proofs of Points (3) and (4), we take care that f satisfies the following additional properties:

1) f is surjective;
2) $a \in C^{\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a})}$ iff $f(a) \in C^{\mathsf{ch}_\mathcal{O}(\mathcal{D})}$ for all $a \in \mathsf{dom}(\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a}))$ and $\mathcal{ELI}$-concepts C;
3) for every $b \in \mathsf{dom}(\mathcal{D}^u_\mathbf{a})$, $(\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a}))_b$ and $(\mathsf{ch}_\mathcal{O}(\mathcal{D}))_{h(b)}$ are isomorphic, with $f|^-_{\mathsf{dom}((\mathsf{ch}_\mathcal{O}(\mathcal{D}))_{h(b)})}$ being a bijective homomorphism from $(\mathsf{ch}_\mathcal{O}(\mathcal{D}))_{h(b)}$ to $(\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a}))_b$.

Let $\mathcal{I}_0, \mathcal{I}_1, \ldots$ and $\mathcal{J}_0, \mathcal{J}_1, \ldots$ be the chase sequences for the construction of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}^u_{\mathbf{a}})$ and $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}_{\mathbf{a}})$, respectively. We construct $f_0, f_1, \ldots$, each f_i being a homomorphism from $\mathcal{I}_i$ to $\mathcal{J}_i$ such that Properties 1-3 above are satisfied (with $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}^u_{\mathbf{a}})$ replaced by $\mathcal{I}_i$ and $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ replaced by $\mathcal{J}_i$).

To start, set $f = \pi$. It can be verified that Properties 1-3 hold for the base case, where $\mathcal{I}_0 = \mathcal{D}^u_{\mathbf{a}}$ and $\mathcal{J}_0 = \mathcal{D}_{\mathbf{a}}$. For $i > 0$, we make a case distinction regarding which of the two chase rules is applied. In case of Rule 1, it is straightforward to extend f_{i-1} to the freshly introduced constants in $\mathcal{I}_i$, and to verify that Properties 1 to 3 above are still satisfied. In case of Rule 2, it is not even necessary to extend f_{i-1}.

The desired homomorphism f is obtained as $\bigcup_i f_i$.

(3) The 'only if' direction follows from Point 1 and Lemma 13. For the 'if' direction, we exploit Property 2 of the homomorphism f constructed in the proof of Point 2 above: for every concept inclusion $C \sqsubseteq \bot$ in $\mathcal{O}'$, there exists some $a \in \mathsf{dom}(\mathsf{ch}_{\mathcal{O}}(\mathcal{D}^u_{\mathbf{a}}))$ with $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}^u_{\mathbf{a}}) \models C(a)$ iff there exists some $b \in \mathsf{dom}(\mathsf{ch}_{\mathcal{O}}(\mathcal{D}_{\mathbf{a}}))$ with $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}_{\mathbf{a}}) \models C(b)$.

(4) The 'if' direction follows from Point 1 and Lemma 13. For 'only if', let $\mathcal{D} \models Q(\mathbf{a})$, with $Q = (\mathcal{O}, \mathbf{S}, q)$. If $\mathcal{D}$ is inconsistent with $\mathcal{O}$, then $\mathcal{D}^u_{\mathbf{a}}$ is inconsistent as well by Point 3, and thus $\mathcal{D}^u_{\mathbf{a}} \models Q(\mathbf{a})$.

Assume that $\mathcal{D}$ is consistent with $\mathcal{O}$. Since $\mathcal{D} \models Q(\mathbf{a})$, there is a homomorphism h from some CQ $p(\mathbf{x})$ in q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ such that $h(\mathbf{x}) = \mathbf{a}$. It suffices to show that there is a homomorphism g from $p(\mathbf{x})$ to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}^u_{\mathbf{a}})$ such that $g(\mathbf{x}) = \mathbf{a}$. We assume w.l.o.g. that p is connected (otherwise we can treat one maximal connected component at a time).

We distinguish between two cases. In the first case, the range of h does not contain any constant from $\mathcal{D}$. In this case p is Boolean and it maps completely into the anonymous part of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$. Then, there must be some $b \in \mathsf{dom}(\mathcal{D})$ such that the image of p under h is a subdatabase of $(\mathsf{ch}_{\mathcal{O}}(\mathcal{D}))_b$. Let $b' \in \mathsf{dom}(\mathcal{D}^u_{\mathbf{a}})$ be such that $f(b') = b$, where f is the homomorphism described at Point 2 above. Note that such a b' always exists as f is surjective. Then, from Property 3 of f mentioned at Point 2 above, it follows that $(\mathsf{ch}_{\mathcal{O}}(\mathcal{D}))_b \rightarrow (\mathsf{ch}_{\mathcal{O}}(\mathcal{D}^u_{\mathbf{a}}))_{b'}$ via the homomorphism $f|^-_{\mathsf{dom}((\mathsf{ch}_{\mathcal{O}}(\mathcal{D}))_{h(b)})}$. Then $f|^-_{\mathsf{dom}((\mathsf{ch}_{\mathcal{O}}(\mathcal{D}))_{h(b)})} \circ h$ is a homomorphism from p to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}^u_{\mathbf{a}})$.

If the range of h does contain some constant from $\mathsf{dom}(\mathcal{D})$, let $\mathbf{z}$ be the set of those variables z of $\mathsf{var}(p)$ for which $h(z) \in \mathsf{dom}(\mathcal{D}|_{\overline{\mathbf{a}}})$. If $\mathbf{z}$ is not empty, let (V', E', μ') be some tree decomposition of $p|_{\mathbf{z}}$ of width at most k (as $\mathbf{z}$ cannot contain any answer variable, $p|_{\mathbf{z}}$ has tree width at most k). Also let (V, E, μ) be the tree decomposition of width k of $\mathcal{D}|^u_{\mathbf{a}}$ resulting from the construction of $\mathcal{D}|^u_{\mathbf{a}}$. We proceed inductively on each bag from V', starting with an arbitrary bag v'. We have that the image of v' under h is a subdatabase $\mathcal{D}'$ of $\mathcal{D}$ of size at most $k+1$. Then, there must be some $v \in V$ such that the restriction of $\mathcal{D}|^u_{\mathbf{a}}$ to $\mu(v)$ is isomorphic to $\mathcal{D}'$ (by construction of $\mathcal{D}|^u_{\mathbf{a}}$) and $f|^-_{\mathsf{dom}(\mathcal{D}')}$ is a homomorphism from $\mathcal{D}'$ to that restriction. We set $g(z) = f|^-_{\mathsf{dom}(\mathcal{D}')}(h(z))$, for every $z \in \mu'(v')$.

At each subsequent induction step, we consider all bags which intersect with a given bag $v' \in V'$ and extend g accordingly (this is possible due to the fact that (V', E', μ') has width at most k).

We next define $g(z) = h(z)$, for every variable $y \in \mathsf{var}(p)$ for which $h(y) \in \mathbf{a}$. Note that at this point, g is defined on all variables y from $\mathsf{var}(p)$ for which $h(y) \in \mathsf{dom}(\mathcal{D})$. Further on, g has the property that $f(g(y)) = h(y)$, for every such variable y. The remaining variables from $\mathsf{var}(p)$ are mapped by h into the 'existential part' of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$. By exploiting the above mentioned property of g and using a technique similar to the one employed in the case where all variables from p were mapped into the existential part of the chase, we can extend g to a full homomorphism from $p(\mathbf{x})$ to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}|^u_{\mathbf{a}})$ such that $g(\mathbf{x}) = \mathbf{a}$.

□

We also need a slightly different version of unraveling a database $\mathcal{D}$ into a database of tree width 1, which we introduce next. The difference to the unravelings above is that we choose a constant $a \in \mathsf{dom}(\mathcal{D})$ at which to start the unraveling.

For a constant $a \in \mathsf{dom}(\mathcal{D})$, the *1-unraveling of $\mathcal{D}$ at a* is defined similarly to the 1-unraveling of $\mathcal{D}$ except that we start the construction with the (potentially empty) restriction of $\mathcal{D}$ to facts that involve only the constant a, with the tree decomposition (V_0, E_0, μ_0) where V_0 contains a single vertex v with $\mu_0(v) = \{a\}$ rather than $\mu_0(v) = \emptyset$, and with the function π_0 that is the identity on a.

Proofs for Section III

Proposition 3. *p-*Evaluation$(\mathcal{ELHI}_\bot, \text{UCQ}_k)$ *is in* FPT*, for any $k \geq 1$, with single exponential running time in the parameter.*

Proof. (sketch) Let the following be given: an OMQ $Q = (\mathcal{O}, \mathbf{S}, q)$ from $(\mathcal{ELHI}_\bot, \text{UCQ}_k)$, an $\mathbf{S}$-database $\mathcal{D}_0$, and a candidate answer $\mathbf{a} \in \mathsf{dom}(\mathcal{D}_0)$.

Our algorithm consists of three different steps. We start with some preliminaries. We assume w.l.o.g. that the ontology is in normal form, that is, all concept inclusions in it are of one of the forms

$$\top \sqsubseteq A, \quad A \sqsubseteq \bot, \quad A_1 \sqcap A_2 \sqsubseteq A, \quad \exists r.A \sqsubseteq B, \quad A \sqsubseteq \exists r.B$$

where A, B, A_1, A_2 range over concept names and r ranges over roles. It is well-known that every can be converted into normal form in linear time with the resulting ontology being equivalent up to the introduction of fresh concept names [4].

An *extended database* is a database that might additionally contain *concept facts* of the form $C(a)$ with C an $\mathcal{ELI}$-concept. An interpretation $\mathcal{I}$ *satisfies* a concept fact $C(a)$ if $a \in C^{\mathcal{I}}$. It is a *model* of an extended database $\mathcal{B}$ if $\mathcal{I}$ is a model of all facts in $\mathcal{B}$, including the concept facts. We write $\mathcal{B}, \mathcal{O} \models C(a)$ if every model of $\mathcal{B}$ and $\mathcal{O}$ satisfies $C(a)$. With $\mathsf{sub}(\mathcal{O})$, we denote the set of all subconcepts of concepts that occur in $\mathcal{O}$. For $\mathcal{ELI}_\bot$-concepts C, D, we write $\mathcal{O} \models C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{O}$. Whether $\mathcal{O} \models C \sqsubseteq D$ can be decided in single exponential time [4].

Step 1. We exhaustively apply the following chase rules, starting with the database $\mathcal{D} = \mathcal{D}_0$ and producing a sequence of extended databases:

1) if $C_1(a), \ldots, C_n(a) \in \mathcal{D}$, $\mathcal{O} \models C_1 \sqcap \cdots \sqcap C_n \sqsubseteq C$ with $C \in \mathsf{sub}(\mathcal{O})$, and $C(a) \notin \mathcal{D}$, then add $C(a)$ to $\mathcal{D}$;
2) if $r(a,b), C(b) \in \mathcal{D}$, $\exists r.C \sqsubseteq D \in \mathcal{O}$, and $D(a) \notin \mathcal{D}$, then add $D(a)$ to $\mathcal{D}$;
3) if $r(a,b) \in \mathcal{D}$, $r \sqsubseteq s \in \mathcal{O}$, and $s(a,b) \notin \mathcal{D}$, then add $s(a,b)$ to $\mathcal{D}$.

It is clear that no chase rule is applicable after at most polynomially many steps. From now on, we use $\mathcal{D}$ to refer to the extended database resulting from the chase. It is not difficult to show the following.

Claim. For all $a \in \mathsf{dom}(\mathcal{D}_0)$ and $C \in \mathsf{sub}(\mathcal{O})$, $\mathcal{D}_0, \mathcal{O} \models C(a)$ iff $C(a) \in \mathcal{D}$.

Therefore, if $\mathcal{D}$ contains a concept fact of the form $\bot(a)$, then $\mathcal{D}_0$ is inconsistent with $\mathcal{O}$ and we can answer 'true'. Otherwise, $\mathcal{D}_0$ is consistent with $\mathcal{O}$.

Step 2. A *type* is a subset $t \subseteq \mathsf{sub}(\mathcal{O})$. For each interpretation $\mathcal{I}$ and $a \in \mathsf{dom}(\mathcal{D})$, the type realized at a in $\mathcal{I}$ is $t_a^{\mathcal{I}} = \{C \in \mathsf{sub}(\mathcal{O}) \mid a \in C^{\mathcal{I}}\}$. For types, t_1, t_2, we write $\mathcal{O} \models t_1 \rightsquigarrow t_2$ if every model of $\mathcal{O}$ that realizes t_1 also realizes t_2. For every role r, we write $\mathcal{O} \models^{\mathsf{max}} t_1 \sqsubseteq \exists r.t_2$ if $\mathcal{O} \models t_1 \sqsubseteq \exists r.t_2$ and t_2 is maximal with this property regarding set inclusion. It is not hard to see that both $\mathcal{O} \models t_1 \rightsquigarrow t_2$ and $\mathcal{O} \models^{\mathsf{max}} t_1 \sqsubseteq \exists r.t_2$ can also be decided in single exponential time. For example, the former is equivalent to $\mathcal{B}, \mathcal{O}' \models A(a)$ where $\mathcal{B}$ is the extended database $\{C(a) \mid C \in t_1\}$, $\mathcal{O}'$ is $\mathcal{O}$ extended with $\bigsqcap t_2 \sqsubseteq \bot$, and A is a fresh concept name.

In Step 2, we first extend $\mathcal{D}$ as follows: for every $a \in \mathsf{dom}(\mathcal{D})$, every type t with $\mathcal{O} \models t_a^{\mathcal{D}} \rightsquigarrow t$, and every $C \in t$, add $C(a_t)$. Note that this extension relies on Step 1 to make explicit the types of the constants in $\mathsf{dom}(\mathcal{D}_0)$. We then apply the following chase rule exactly $|q|$ times:

($*$) for every $a \in \mathsf{dom}(\mathcal{D})$ and every role r and type t such that $\mathcal{O} \models^{\mathsf{max}} t_a^{\mathcal{D}} \sqsubseteq \exists r.t$ and there is no $b \in \mathsf{dom}(\mathcal{D})$ with $r(a,b) \in \mathcal{D}$ and $t_b^{\mathcal{D}} \supseteq t$, add $r(a,b)$ and $C(b)$ for every $C \in t$, with b a fresh constant.

This second chase generates a tree of depth up to $|q|$ below each constant from $\mathsf{dom}(\mathcal{D}_0)$ and we again denote the result with $\mathcal{D}$. If we chased forever rather than only $|q|$ steps, we would obtain a (potentially infinite) universal model of $\mathcal{D}_0$ and $\mathcal{O}$. The truncated version $\mathcal{D}$ constructed here is not universal, but every neighborhood of diameter at most $|q|$ in the universal model is also present in $\mathcal{D}$, due to the extension of $\mathcal{D}$ carried out before starting the second chase. Thus, $\mathcal{D}_0 \models Q(\mathbf{a})$ iff there is a homomorphism from $q(\mathbf{x})$ to $\mathcal{D}$ with $h(\mathbf{x}) = \mathbf{a}$ iff $\mathcal{D} \models q(\mathbf{a})$ in the standard sense of relational databases.

Step 3. Apply the polynomial time algorithm for evaluating CQs of tree width bounded by k (see Theorem 1) to q and $\mathcal{D}$.

The correctness of the algorithm follows from what was said above. Regarding the overall running time, it is not hard to see that $|\mathsf{dom}(\mathcal{D})| \leq |D_0| + |D_0| \cdot |\mathcal{O}|^{|q|}$ and thus $|D_0| \leq p(|D_0| \cdot |\mathcal{O}|^{|q|})$, p a polynomial. Since the check in Step 3 needs only polynomial time, this gives the desired $2^{p(|Q|)} \cdot p(|\mathcal{D}_0|)$ bound on the running time where p is again a polynomial. $\square$

Proposition 4. *In $(\mathcal{ELI}, CQ)$, the notions of CQ$_1$-equivalence while preserving the ontology and UCQ$_1$-equivalence while preserving the ontology do not coincide.*

Proof. Let $Q = (\mathcal{O}, \mathbf{S}, q)$ be an OMQ from ($\mathcal{ELI}$, CQ) with:

$$\begin{array}{rclllccl}
\mathcal{O} & = & \{B_1 & \sqsubseteq & A_1 & B_1 \sqcap \exists r^-.B_4 & \sqsubseteq & A_3 \\
 & & B_2 & \sqsubseteq & A_2 & B_2 \sqcap \exists r.C_1 & \sqsubseteq & A_4 \\
 & & B_3 & \sqsubseteq & A_3 & \exists r.B_3 \sqcap C_2 & \sqsubseteq & A_4 \\
 & & B_4 & \sqsubseteq & A_4 & B_4 \sqcap \exists r.C_3 & \sqsubseteq & A_2 \\
 & & C_1 & \sqsubseteq & A_1 & \exists r.C_1 \sqcap C_4 & \sqsubseteq & A_2 \\
 & & C_2 & \sqsubseteq & A_2 & C_2 \sqcap \exists r.B_1 & \sqsubseteq & A_4 \\
 & & C_3 & \sqsubseteq & A_3 & \exists r.C_3 \sqcap B_2 & \sqsubseteq & A_4 \\
 & & C_4 & \sqsubseteq & A_4 & C_4 \sqcap \exists r.B_3 & \sqsubseteq & A_2\}, \\
\mathbf{S} & = & \multicolumn{6}{l}{\{B_1, B_2, B_3, B_4, C_1, C_2, C_3, C_4, r\}.}
\end{array}$$

and q the CQ from Example 1.

Then Q is UCQ$_1$-equivalent, but not CQ$_1$-equivalent while preserving the ontology. We have that $Q \equiv (\mathcal{O}, \mathbf{S}, \exists x_1 \exists x_2 \exists x_3 \exists x_4\, (\varphi_1 \vee \varphi_2))$, where:

$$\varphi_1 = r(x_2,x_1) \wedge r(x_2,x_3) \wedge A_1(x_1) \wedge A_2(x_2) \wedge A_4(x_2) \wedge A_3(x_3)$$

and

$$\varphi_2 = r(x_2,x_1) \wedge r(x_4,x_1) \wedge A_1(x_1) \wedge A_3(x_1) \wedge A_2(x_2) \wedge A_4(x_4).$$

To see that Q is not CQ$_1$-equivalent while preserving the ontology, assume the contrary. Then, there exists some OMQ $Q' = (\mathcal{O}, \mathbf{S}, q')$ with q' from CQ$_1$ such that $Q \equiv Q'$. Let

$$\begin{array}{rcl}
\mathcal{D}_1 & = & \{r(b_2,b_1), r(b_2,b_3), r(b_4,b_3), r(b_4,b_1), \\
 & & B_1(b_1), B_2(b_2), B_3(b_3), B_4(b_4)\} \\
\mathcal{D}_2 & = & \{r(c_2,c_1), r(c_2,c_3), r(c_4,c_3), r(c_4,c_1), \\
 & & C_1(c_1), C_2(c_2), C_3(c_3), C_4(c_4)\}.
\end{array}$$

Then $\mathcal{D}_1 \models Q$ and $\mathcal{D}_2 \models Q$. Thus $\mathcal{D}_1 \models Q'$ and $\mathcal{D}_2 \models Q'$ and consequently $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}_1) \times \mathsf{ch}_{\mathcal{O}}(\mathcal{D}_2) \models q'$ where '$\times$' denotes the direct product. It can be verified that $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}_1) \times \mathsf{ch}_{\mathcal{O}}(\mathcal{D}_2)$ is isomorphic to $\mathcal{D}_q$. As q' is from CQ$_1$, it follows that $\mathcal{D}_q^u \models q'$, where $\mathcal{D}_q^u$ is the 1-unraveling of $\mathcal{D}_q$, that is,

$$\begin{array}{rcl}
\mathcal{D}_q^u & = & \{C_1(x_1), C_2(x_2), C_3(x_3), C_4(x_4), C_1(x_5), \ldots \\
 & & r(x_2,x_1), r(x_2,x_3), r(x_4,x_3), r(x_4,x_5), \ldots\}
\end{array}$$

Let $Q^u = (\mathcal{O}, \mathbf{S}, q^u)$, q^u a Boolean CQ such that $D_{q^u} = D_q^u$. Then, $Q^u \subseteq Q' \equiv Q$ and $Q \subseteq Q^u$. Thus, $Q \equiv Q^u$. Now consider the database

$$\begin{array}{rcl}
\mathcal{D}_{12} & = & \{r(x_2,x_1), r(x_2,x_3), r(x_4,x_3), r(x_4,x_5), \ldots, \\
 & & B_1(x_1), B_2(x_2), B_3(x_3), B_4(x_4), \\
 & & C_1(x_5), C_2(x_6), C_3(x_7), C_4(x_8), \\
 & & B_1(x_9), \ldots, \\
 & & \ldots \qquad\qquad \}
\end{array}$$

We have that $\mathcal{D}_{12} \models Q^u$, but $\mathcal{D}_{12} \not\models Q$ – contradiction. □

Theorem 2. *Let Q be an OMQ from $(\mathcal{ELHI}_\bot, UCQ)$, $k \geq 1$, and Q_a the UCQ_k-approximation of Q. Then*

1) $Q(\mathcal{D}) = Q_a(\mathcal{D})$ for any $\mathbf{S}$-database $\mathcal{D}$ of treewidth $\leq k$;
2) $Q' \subseteq Q_a$ for every $Q' \in (\mathcal{ELHI}_\bot, UCQ_k)$ with $Q' \subseteq Q$.

Proof. Let $Q = (\mathcal{O}, \mathbf{S}, q)$ and $Q_a = (\mathcal{O}, \mathbf{S}, q_a)$. For Point 1, further let $\mathcal{D}$ be an $\mathbf{S}$-database of tree width at most k. We can assume that $\mathcal{D}$ is consistent with $\mathcal{O}$. We have $Q_a(\mathcal{D}) \subseteq Q(\mathcal{D})$ by construction of Q_a, independently of the tree width of $\mathcal{D}$. For the converse, let $\mathbf{a} \in Q(\mathcal{D})$. By Lemma 1, we find a CQ $p(\mathbf{x})$ in q and a homomorphism h from p to $\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a})$ with $h(\mathbf{x}) = \mathbf{a}$. Let $\widehat{p}(\mathbf{x})$ be the contraction of p obtained by identifying any two variables z_1 and z_2 with $h(z_1) = h(z_2)$. Since $\mathcal{D}$ has tree width at most k, by Lemma 1 so has $\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a})$. Consequently, also $\widehat{p}$ has tree width at most k and thus is a CQ in q_a. It follows that $\mathbf{a} \in Q_a(\mathcal{D})$.

For Point 2, let $Q' = (\mathcal{O}', \mathbf{S}, q')$. Take an $\mathbf{S}$-database $\mathcal{D}$ such that $\mathcal{D} \models Q'(\mathbf{a})$. We can assume that $\mathcal{D}$ is consistent with $\mathcal{O}$. Consider $\mathcal{D}^u_\mathbf{a}$, the k-unraveling of $\mathcal{D}$ up to $\mathbf{a}$. Lemma 15 yields $\mathcal{D}^u_\mathbf{a} \models Q'(\mathbf{a})$ and consequently $\mathcal{D}^u_\mathbf{a} \models Q(\mathbf{a})$. From Point 1, we thus get $\mathcal{D}^u_\mathbf{a} \models Q_a(\mathbf{a})$. By Lemma 15, there is also a homomorphism g from $\mathsf{ch}_\mathcal{O}(\mathcal{D}^u_\mathbf{a})$ to $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ with $h(\mathbf{a}) = \mathbf{a}$. Lemma 13 yields $\mathcal{D} \models Q_a(\mathbf{a})$ as required. □

Before proceeding to the proof of Theorem 3, we provide some preliminary results. First of all, the main purpose of Step 2 in the definition of the notion of rewriting of $\mathcal{O}$ is to deal with the fact that $\mathcal{O}$-retracts are defined with respect to the chase of the original query rather than the resulting $\mathcal{O}$-retract. It enables the following basic lemma.

Lemma 16. *Let $Q = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q)$ be an OMQ from $(\mathcal{ELHI}_\bot, CQ)$ and $Q' = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q')$ a rewriting of Q based on the $\mathcal{O}$-retract q_h. Then h is a homomorphism from q to $\mathsf{ch}_\mathcal{O}(q')$.*

Proof. By definition of $\mathcal{O}$-retracts, h is a homomorphism from q to $\mathsf{ch}_\mathcal{O}(q)$ such that all variables from $\mathsf{ran}^+(h)$ are also in q_h, thus in q'. Due to Step 2 in the construction of q', for each $x \in \mathsf{ran}^+(h)$, we have $x \in C^{\mathcal{D}_q}$ iff $x \in C^{\mathcal{D}_{q'}}$ for all concepts C that occur on the left-hand side of a concept inclusion in $\mathcal{O}$. Using the definition of the chase, the following is easy to verify:

Claim. If $\mathcal{D}_1, \mathcal{D}_2$ are databases, $a_i \in \mathcal{D}_i$ for $i \in \{1, 2\}$ and $a_1 \in C^{\mathcal{D}_1}$ iff $a_2 \in C^{\mathcal{D}_2}$ for all concepts C that occur on the left-hand side of a concept inclusion in $\mathcal{O}$, then the subdatabase of tree width 1 that the chase generates in $\mathsf{ch}_\mathcal{O}(\mathcal{D}_1)$ below a_1 is isomorphic to the one that it generates in $\mathsf{ch}_\mathcal{O}(\mathcal{D}_2)$ below a_2.

Consequently, h is a homomorphism from q to $\mathsf{ch}_\mathcal{O}(q')$. □

We prove next that an OMQs is in fact equivalent to any of its rewritings.

Lemma 2. *Let $Q = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q)$ be an OMQ from $(\mathcal{ELHI}_\bot, CQ)$ and $Q' = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q')$ a rewriting of Q. Then $Q \equiv Q'$.*

Proof. Let Q' be based on the $\mathcal{O}$-retract q_h.

'$Q \subseteq Q'$'. Assume that $\mathcal{D} \models Q(\mathbf{a})$. We can assume w.l.o.g. that $\mathcal{D}$ is consistent with $\mathcal{O}$. Then there is a homomorphism g from $q(\mathbf{x})$ to $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ with $g(\mathbf{x}) = \mathbf{a}$. Since q_h is a subquery of q, g also demonstrates that $\mathcal{D} \models Q_h(\mathbf{a})$ where $Q_h = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q_h)$. Further, we can extend g to show that $\mathcal{D} \models Q'(\mathbf{a})$. Let $C \sqsubseteq D \in \mathcal{O}$ and $x \in C^{\mathcal{D}_q}$. Then a fresh copy q_C of C viewed as a CQ was added to q_h in the construction of q', identifying x with the root of q_C. But since $x \in C^{\mathcal{D}_q}$, we must have $g(x) \in C^\mathcal{D}$. As a consequence, we can extend g to q_C.

'$Q' \subseteq Q$'. Assume that $\mathcal{D} \models Q'(\mathbf{a})$. We can assume w.l.o.g. that $\mathcal{D}$ is consistent with $\mathcal{O}$. Then there is a homomorphism g from $q'(\mathbf{x})$ to $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ with $g(\mathbf{x}) = \mathbf{a}$. It is possible to extend g to a homomorphism from $\mathsf{ch}_\mathcal{O}(q')$ to $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ by inductively following the applications of the chase rules. By Lemma 16, h is a homomorphism from q to $\mathsf{ch}_\mathcal{O}(q')$. Composing h with g yields a homomorphism h' from $q(\mathbf{x})$ to $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ with $h'(\mathbf{x}) = \mathbf{a}$ and thus $\mathcal{D} \models Q(\mathbf{a})$. □

Theorem 3. *Let $Q = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q)$ be a non-empty OMQ from $(\mathcal{ELHI}_\bot, CQ)$ and $k \geq 1$. The following are equivalent:*

1) Q is UCQ_k-equivalent;
2) Q has a rewriting that falls within $(\mathcal{ELHI}_\bot, CQ_k)$;
3) some full rewriting of Q falls within $(\mathcal{ELHI}_\bot, CQ_k)$;
4) all full rewritings of Q fall within $(\mathcal{ELHI}_\bot, CQ_k)$.

Proof. '3 ⇒ 2' and '2 ⇒ 1' are immediate, and so is '4 ⇒ 3' since a full rewriting clearly always exists. We show '1 ⇒ 4'. For q a CQ we say that we first q is *fully $\mathcal{O}$-retracted* if it has no proper $\mathcal{O}$-retract. We first establish the following claim about the surjectivity of homomorphisms from q to $\mathsf{ch}_\mathcal{O}(q)$ when q is fully $\mathcal{O}$-retracted.

Claim. Let q be a CQ and $\mathcal{O}$ an $\mathcal{ELHI}_\bot$-ontology such that q is fully $\mathcal{O}$-retracted, and let h be a homomorphism from $q(\mathbf{x})$ to $\mathsf{ch}_\mathcal{O}(q)$ with $h(\mathbf{x}) = \mathbf{x}$. Then $\mathsf{ran}^+(h) = \mathsf{var}(q)$.

Proof of claim. Assume to the contrary that $\mathsf{ran}^+(h) \subsetneq \mathsf{var}(q)$. We can extend h to a homomorphism h' from $\mathsf{ch}_\mathcal{O}(q)$ to $\mathsf{ch}_\mathcal{O}(q)$ by inductively following the application of the chase rule. Let $n = |\mathsf{var}(q)|$ and let g be the $n!$-fold composition of h' with itself, restricted to $\mathsf{var}(q)$. It can be verified that g is a homomorphism from q to $\mathsf{ch}_\mathcal{O}(q)$ that is the identity on all answer variables and variables in its range, thus an $\mathcal{O}$-retraction. Moreover, $\mathsf{ran}^+(h) \subsetneq \mathsf{var}(q)$ implies $\mathsf{ran}^+(g) \subsetneq \mathsf{var}(q)$ and thus g induces a proper $\mathcal{O}$-retract of q. This contradicts the fact that q is fully $\mathcal{O}$-retracted, and thus the claim is established.

Now assume that $Q = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q) \in (\mathcal{ELHI}_\bot, \text{CQ})$ is UCQ_k-equivalent. By Corollary 1, this is witnessed by an equivalent OMQ $Q' = (\mathcal{O}, \mathbf{S}_\mathsf{full}, q')$ from $(\mathcal{ELHI}_\bot, \text{UCQ}_k)$. Since Q is non-empty, so is Q' and we can assume w.l.o.g.

that $\mathcal{D}_p$ is consistent with $\mathcal{O}$ for any CQ p in q'. Let $Q_f = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_f)$ be any full rewriting of Q. Assume w.l.o.g. that q, q', and q_f all have the same answer variables $\mathbf{x}$. By Lemma 2, Q_f and Q' are equivalent and by Lemma 14 and since $\mathcal{D}_{q'}$ is consistent with $\mathcal{O}$, there must be a CQ p in q' and homomorphisms

- h_1 from q_f to $\mathsf{ch}_{\mathcal{O}}(p)$ with $h_1(\mathbf{x}) = \mathbf{x}$ and
- h_2 from p to $\mathsf{ch}_{\mathcal{O}}(q_f)$ with $h_2(\mathbf{x}) = \mathbf{x}$.

We can extend h_2 to a homomorphism from $\mathsf{ch}_{\mathcal{O}}(p)$ to $\mathsf{ch}_{\mathcal{O}}(q_f)$, by inductively following the applications of the chase rules. Let h denote the composition $h_2 \circ h_1$. By the claim, we must have $\mathsf{ran}^+(h) = \mathsf{var}(q)$. As a consequence, h_1 must be injective. But p has tree width at most k, and consequently so has $\mathsf{ch}_{\mathcal{O}}(p)$. It thus follows from h_1 being injective that q_f also has tree width at most k. □

Proofs for Section IV

Theorem 7 (Grohe)**.** Given an undirected graph $G = (V, E)$, a $k > 0$, and a connected $\mathbf{S}$-database $\mathcal{D}$ such that $G_{\mathcal{D}}$ contains the $k \times K$-grid as a minor, one can construct in time $f(k) \cdot \mathsf{poly}(|G|, |\mathcal{D}|)$ an $\mathbf{S}$-database $\mathcal{D}_G$ such that:

1) there is a surjective homomorphism h_0 from $\mathcal{D}_G$ to $\mathcal{D}$ such that for every edge $\{a, b\}$ in the Gaifman graph of $\mathcal{D}_G$: $s(a, b) \in \mathcal{D}_G$ iff $s(h_0(a), h_0(b)) \in \mathcal{D}$ for all roles s;
2) G contains a k-clique iff there is a homomorphism h from $\mathcal{D}$ to $\mathcal{D}_G$ such that $h_0(h(\cdot))$ is the identity.

Proof. A careful analysis of [25] reveals that the proof given there establishes Theorem 7 without the 'such that' condition in Point 1. That condition, however, can easily be attained as follows. Assume that G, k, and $\mathcal{D}$ are given. First rewrite $\mathcal{D}$ into a new schema $\mathbf{S}'$ that consists of the concept names in $\mathbf{S}$ and a fresh role name r_R for every set $R \subset \{r, r^- \mid r \text{ role name in } \mathbf{S}\}$, by replacing every maximal set $\{r_0(a, b), \ldots, r_n(a, b)\}$, $r_1, \ldots, r_n$ (potentially inverse) roles, with the fact $r_{r_0,\ldots,r_n}(a, b)$. Then apply Theorem 7 without the 'such that' condition in Point 1, obtaining D'_G and h_0. Now revert back D'_G to schema $\mathbf{S}$ in the obvious way. It can be verified that the resulting database D_G and h_0 satisfy Conditions 1 and 2, also with the 'such that' condition in Point 1. □

Lemma 3. *If $\mathcal{D} \models p$, for $\mathcal{D}$ a potentially infinite database and p a CQ, then $\mathcal{D} \models^{io} p_c$ for some contraction p_c of p.*

Proof. We can find p_c by the following iterative process: start with $p_c = p$ and then exhaustively take a homomorphism h from p_c to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ and if h is not injective, replace p_c with the contraction p'_c of p_c induced by h, that is, identify quantified variables y_1 and y_2 if $h(y_1) = h(y_2)$. Clearly, this process must terminate since the number of variables in p_c decreases with each step. □

Lemma 4. *There is an $\mathbf{S}$-database $\mathcal{D}$ such that the following conditions are satisfied:*

1) *$\mathcal{D} \models Q_w$ and $\mathcal{D} \not\models Q_a$;*
2) *if $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}) \models^{io} q_c$, for q_c a contraction of q_w, then there is no $\mathbf{S}$-database $\mathcal{D}'$ and contraction q'_c of q_w such that $\mathcal{D}' \to \mathcal{D}$, $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}') \models^{io} q'_c$, and $q'_c \prec q_c$.*

Proof. It remains to argue that the iterative process described in the main part of the paper terminates. For an $\mathbf{S}$-database $\mathcal{B}$ let $\mathcal{Q}_{\mathcal{B}}$ denote the set of contractions q_c of q_w such that $\mathsf{ch}_{\mathcal{O}}(\mathcal{B}) \models^{io} q_c$. The partial order '$\preceq$' lifts to sets $\mathcal{Q}_1, \mathcal{Q}_2$ of contractions of q in the obvious way: $\mathcal{Q}_1 \preceq \mathcal{Q}_2$ if for every $q_c \in \mathcal{Q}_1$, there is a $q'_c \in \mathcal{Q}_2$ with $q_c \preceq q'_c$. As usual, we write $\mathcal{Q}_1 \prec \mathcal{Q}_2$ if $\mathcal{Q}_1 \preceq \mathcal{Q}_2$ and $\mathcal{Q}_1 \not\succeq \mathcal{Q}_2$.

Observe that whenever $\mathcal{D}$ is replaced by $\mathcal{D}'$ in the iterative process, then $\mathcal{Q}_{\mathcal{D}'} \prec \mathcal{Q}_{\mathcal{D}}$:

1) $\mathcal{Q}_{\mathcal{D}'} \preceq \mathcal{Q}_{\mathcal{D}}$: Let $q_c \in \mathcal{Q}_{\mathcal{D}'}$. Using $\mathcal{D}' \to \mathcal{D}$, one can prove that $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}') \to \mathsf{ch}_{\mathcal{O}}(\mathcal{D})$, following the applications of the chase rules. From $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}') \models q_c$, we thus obtain $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}) \models q_c$. By Lemma 3, we find a q'_c with $q_c \preceq q'_c$ and $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}) \models^{io} q'_c$, thus $q'_c \in \mathcal{Q}_{\mathcal{D}}$.
2) $\mathcal{Q}_{\mathcal{D}} \not\preceq \mathcal{Q}_{\mathcal{D}'}$: Assume that $\mathcal{D}$ was replaced by $\mathcal{D}'$ because of the contractions q_c and q'_c with $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}) \models^{io} q_c$, $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}') \models^{io} q'_c$ and $q'_c \preceq q_c$. Then $q_c \in \mathcal{Q}_{\mathcal{D}}$. We show that there is no $q''_c \in \mathcal{Q}_{\mathcal{D}'}$ with $q_c \preceq q''_c$. Assume to the contrary that there is such a q''_c. Then $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}') \models^{io} q''_c$ and thus there is a homomorphism h from q''_c to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}')$. We also have $q_c \to q''_c$ and $q'_c \to q_c$ via a non-injective homomorphism since q_c is a proper contraction of q'_c. By composition, we obtain a non-injective homomophism h' from q'_c to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}')$, in contradiction to $q'_c \in \mathcal{Q}_{\mathcal{D}'}$.

Note that '$\preceq$' is a partial order on the contractions of q_w. It remains to note then that when '$\prec$' is interpreted on sets of contractions is trivially well-founded. This is because $\prec$ is strict and there are only finitely many contractions of q_w. □

Lemma 5. *Given an $\mathbf{S}$-database $\mathcal{D}$ and an OMQ Q from $(\mathcal{ELHI}_\bot, \text{CQ})$, it is decidable whether Conditions 1 and 2 from Lemma 4 hold.*

Proof. Condition 1 is decidable since OMQ evaluation is. For Condition 2, we first show that it suffices to look at databases $\mathcal{D}'$ of a certain restricted shape called pseudo trees [7] and then argue that this enables a reduction to the satisfiability of MSO sentences on trees. An alternative for the second step is a reduction to the emptiness problem of alternating tree automata, which we conjecture to deliver a 2ExpTime upper bound.

Pseudo trees. A database $\mathcal{D}$ is a *pseudo tree* if it is the union of databases $\mathcal{D}_0, \ldots, \mathcal{D}_m$ that satisfy the following conditions:

1) $\mathcal{D}_1, \ldots, \mathcal{D}_m$ are of tree width one;
2) $\mathsf{dom}(\mathcal{D}_i) \cap \mathsf{dom}(\mathcal{D}_j) = \emptyset$, for $1 \le i < j \le k$;
3) $\mathcal{D}_i \cap \mathcal{D}_0$ is a singleton for $1 \le i \le k$.

The *width* of $\mathcal{D}$ is $|\mathsf{dom}(\mathcal{D}_0)|$.

Claim. The condition in Point 2 of Lemma 4 is equivalent to the same condition when '$\mathbf{S}$-database $\mathcal{D}'$' is replaced with 'pseudo tree $\mathbf{S}$-database $\mathcal{D}'$ of width at most $|q_w|$.

Proof of claim. Assume that if $\mathsf{ch}_{\mathcal{O}}(\mathcal{D}) \models^{io} q_c$, q_c a contraction of q_w, and that there is an $\mathbf{S}$-database $\mathcal{D}'$ and a contraction q'_c of q_w such that $\mathcal{D}' \to \mathcal{D}$, $\mathsf{ch}_\mathcal{O}(\mathcal{D}') \models^{io} q'_c$, and $q'_c \prec q_c$. Let h be an (injective) homomorphism from q'_c to $\mathsf{ch}_\mathcal{O}(\mathcal{D}')$. Construct a pseudo tree database $\mathcal{D}''$ as the union of the restriction $\mathcal{D}_0$ of $\mathcal{D}$ to the range of h and the databases $\mathcal{D}_1, \dots, \mathcal{D}_m$ which are the 1-unravelings of some constant a from $\mathcal{D}_0$. It can be verified that $\mathcal{D}'' \to \mathcal{D}'$, thus $\mathcal{D}' \to \mathcal{D}$. Moreover, $\mathsf{ch}_\mathcal{O}(\mathcal{D}'') \models^{io} q'_c$. We have thus replaced $\mathcal{D}'$ with the pseudo tree database $\mathcal{D}''$.

Reduction to MSO. It thus suffices to decide the existence of a pseudo-tree database that satisfies the conditions for $\mathcal{D}'$ in Point 2 of Lemma 4. Let Σ be a finite alphabet. A Σ-*labeled tree* takes the form (T, ℓ) where $T \subseteq S^*$, S a set of any cardinality, is closed under prefixes and $\ell : T \to \Sigma$ is a node labeling function. The satisfiability problem for monadic second-order logic (MSO) on Σ-labeled trees is decidable. It is not hard to encode pseudo-tree $\mathbf{S}$-databases into Σ-labeled trees for an appropriately chosen Σ, see [7]. Note that the entire $\mathcal{D}_0$-component can be encoded as a single node label since it contains only boundedly many constants. Further, it is possible to express the conditions for $\mathcal{D}'$ in Point 2 of Lemma 4 as MSO sentences. This is technically very closely related to the tree automata constructions in [7]. We refrain from going into technical detail. □

Lemma 7.

1) *$\mathcal{D}^+$ satisfies Conditions 1 and 2 of Lemma 4;*
2) *$\mathcal{D}$ has tree width exceeding ℓ.*

Proof. For Condition 1 of Lemma 4, it is clear that $\mathcal{D}^+ \models Q_w$. Further, recall that $\mathcal{D}_0 \not\models Q_a$. It is straightfoward to show that $\mathcal{D}^+ \to \mathcal{D}_0$ and thus also $\mathcal{D}^+ \not\models Q_a$.

For Condition 2 of Lemma 4, assume to the contrary that there are an $\mathbf{S}$-database $\mathcal{D}'$ and contractions q_c, q'_c of q_w such that $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+) \models^{io} q_c$, $\mathcal{D}' \to \mathcal{D}^+$, $\mathsf{ch}_\mathcal{O}(\mathcal{D}') \models^{io} q'_c$, and $q'_c \prec q_c$. We use a case distinction:

- $\mathsf{ch}_\mathcal{O}(\mathcal{D}_0) \models^{io} q_c$.
  From $\mathcal{D}' \to \mathcal{D}^+$ and $\mathcal{D}^+ \to \mathcal{D}_0$, we obtain $\mathcal{D}' \to \mathcal{D}_0$. This together with $\mathsf{ch}_\mathcal{O}(\mathcal{D}_0) \models^{io} q_c$ and $\mathsf{ch}_\mathcal{O}(\mathcal{D}') \models^{io} q'_c$ yields a contradiction to $\mathcal{D}_0$ satisfying Condition 2 of Lemma 4.
- $\mathsf{ch}_\mathcal{O}(\mathcal{D}_0) \not\models^{io} q_c$. From $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+) \models^{io} q_c$ and $\mathcal{D}^+ \to \mathcal{D}_0$, we obtain $\mathsf{ch}_\mathcal{O}(\mathcal{D}_0) \models q_c$. By Lemma 3, there is a contraction q'_c of q_c with $\mathsf{ch}_\mathcal{O}(\mathcal{D}_0) \models^{io} q'_c$. But then we must have $q'_c = q_c$ as otherwise we obtain a contradiction to $\mathcal{D}_0$ satisfying Condition 2 of Lemma 4 (instantiated with $\mathcal{D} = \mathcal{D}' = \mathcal{D}_0$). Contradiction.

Now for Point 2 of Lemma 7. From the fact that $\mathcal{D}^+ \not\models Q_a$ and that Q_a is equivalent to Q on $\mathbf{S}$-databases of tree width at most ℓ, we obtain that the tree width of $\mathcal{D}^+$ exceeds ℓ. But by construction of $\mathcal{D}^+$, the tree width of $\mathcal{D}^+$ cannot be higher than that of $\mathcal{D}$. □

Lemma 8. *G has a k-clique iff $\mathcal{D}^*_G \models Q$.*

Proof. By what was said in the main body of the paper, it remains to prove that

(†) for every edge e in $\mathcal{D}$, there is an atom in $nt(q_c)$ that g maps to e.

In fact, assume to the contrary that g maps no atom in $nt(q_c)$ to some edge $\{a, b\} \in \mathcal{D}$. We argue that, then the database $\mathcal{D}_1$ obtained from $\mathcal{D}$ by removing all binary facts that involve a and b is such that $\mathcal{D}^+_1 \models Q$, contradicting the choice of $\mathcal{D}$. It suffices to show that there is a homomorphism h_1 from q_c to $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+_1)$. We obtain h_1 by manipulating the homomorphism $h_0(h(\cdot))$ from q_c to $\mathsf{ch}_\mathcal{O}(\mathcal{D}^+)$ in a suitable way. If $h_0(h(\cdot))$ does not map any atom in q_c to $\{a, b\}$, then there is nothing to do. Otherwise, iterate over all atoms $r(x, y) \in q_c$ that are mapped to $\{a, b\}$. Then $r(x, y) \notin nt(q_c)$, that is, this atom must be in a part of q_c that is of tree width 1. Assume w.l.o.g. that $h_0(h(x)) = a$ and that y is a successor of x in that part, that is, y is further away from the root than x; the cases that $h_0(h(x)) = b$ and/or x is a successor of y can be treated analogously and the case that $x = y$ cannot occur since $\mathcal{D}$ contains no facts of the form $r(c, c)$. Let q_y be the tree width 1 subquery of q_c rooted at y. Since $h_0(h(\cdot))$ is injective, none of the atoms in q_y is mapped to $\{a, b\}$. In $\mathcal{D}^+_1$, a copy of the tree width 1 database $\mathcal{D}_a$ has been added, the root identified with a. We can use $h_0(h(\cdot))$ to (re)define h_1 for all variables in q_y. Informally, whenever $h_0(h(z))$ is (a copy of) $c \in \mathsf{dom}(\mathcal{D}_0)$ in $\mathcal{D}^+$ for any variable z from q_y, then we can choose for $h_1(z)$ a corresponding copy of c in the mentioned copy of $\mathcal{D}_a$ in $\mathcal{D}^+_1$. This establishes (†). □

A. Non-Boolean OMQs

We next consider the case where $\mathcal{Q} \subseteq (\mathcal{ELHI}_\bot, \text{CQ})$ might contain non-Boolean OMQs. We start with a lemma which strengthens the result from Point (1) of Theorem 2 and whose proof follows from the proof of the same theorem:

Lemma 17. *Let Q be an n-ary OMQ from $(\mathcal{ELHI}_\bot, UCQ)$, $k \geq 1$, and Q_a the UCQ$_k$-approximation of Q. Then for every $\mathbf{S}$-database $\mathcal{D}$ and n-tuple $\mathbf{a}$ over $\mathsf{dom}(\mathcal{D})$ such that $D|_{\overline{\mathbf{a}}}$ is of tree width k, it is the case that: $\mathcal{D} \models Q(\mathbf{a})$ iff $\mathcal{D} \models Q_a(\mathbf{a})$.*

Let $Q = (\mathcal{O}, \mathbf{S}, q) \notin (\mathcal{ELHI}_\bot, \text{UCQ})^{\equiv}_{\text{UCQ}_\ell}$ of arity n. Instead of considering the maximal connected components of q as in the Boolean case, we consider the connected components $q_1, \dots, q_p$ of the Boolean CQ $q|_\mathbf{y}$, where $\mathbf{y}$ are the quantified variables of q. For each i, let q^+_i be the restriction of q to the variables $\mathbf{x} \cup \mathsf{var}(q_i)$. Note that it is not guaranteed that all variables from $\mathbf{x}$ occur in some atom of q^+_i. We nevertheless assume that the answer variables of q^+_i are exactly $\mathbf{x}$, which can be achieved e.g. by admitting dummy atoms of the form $\mathsf{adom}(x_i)$ where adom is assumed to be true for all constants in the input database. Also, let $Q_i = (\mathcal{O}, \mathbf{S}, q^+_i)$.

As $Q = (\mathcal{O}, \mathbf{S}, q) \notin (\mathcal{ELHI}_\bot, \text{UCQ})^{\equiv}_{\text{UCQ}_\ell}$, it must be the case that there is some w such that $Q_w \notin (\mathcal{ELHI}_\bot, \text{UCQ})^{\equiv}_{\text{UCQ}_\ell}$. Let $q^-(\mathbf{x}) = \bigwedge_{1 \leq i \neq w \leq p} q^+_i$ and $Q^- = (\mathcal{O}, \mathbf{S}, q^-)$. It can be assumed that $Q \not\equiv Q^-$ (otherwise Q could be replaced by Q^-). Let D^- be a database such that $D^- \models Q^-(\mathbf{a})$, but $D^- \not\models Q(\mathbf{a})$, for some tuple $\mathbf{a}$ over $\mathsf{dom}(D^-)$. Then $D^- \not\models Q_w(\mathbf{a})$. We assume w.l.o.g. that all constants in $\mathbf{a}$ are distinct. If this is not the case, we can duplicate constants from $\mathsf{dom}(D^-)$ to obtain a tuple with distinct elements which is an answer to Q^-.

Let $Q_a = (\mathcal{O}, \mathbf{S}, q_a)$ be the ℓ-approximation of Q_w. Furthermore, let $\mathcal{B}$ be a database such that $\mathcal{B} \models Q_w(\mathbf{b})$, but $\mathcal{B} \not\models Q_a(\mathbf{b})$ for some tuple $\mathbf{b}$ over $\mathsf{dom}(\mathcal{B})$. We can assume again w.l.o.g. that all constants in $\mathbf{b}$ are distinct and also that the domains of D^- and $\mathcal{B}$ are disjoint. This also means that we can rename $\mathbf{b}$ as $\mathbf{a}$ in $\mathcal{B}$ and obtain that: $\mathcal{B} \models Q_w(\mathbf{a})$, but $\mathcal{B} \not\models Q_a(\mathbf{a})$.

For a contraction $q_c(\mathbf{x})$ of q, a database $\mathcal{D}$, and an $\mathbf{a} \in \mathsf{dom}(\mathcal{D})^{|\mathbf{x}|}$, we write $\mathcal{D} \models^{io} q_c(\mathbf{a})$ if all homomorphims h from q_c to $\mathcal{D}$ with $h(\mathbf{x}) = \mathbf{a}$ are injective. We also use again a modified version of Lemma 4, established by the same proof.

Lemma 18. *Let $Q = (\mathcal{O}, \mathbf{S}, q)$ be an OMQ from $(\mathcal{ELHI}_\bot, CQ)$ and $\mathcal{D}$ an $\mathbf{S}$-database with $\mathcal{D} \models Q(\mathbf{a})$. Then there is an $\mathbf{S}$-database $\mathcal{D}'$ such that the following conditions are satisfied:*

1) *$\mathcal{D}' \to \mathcal{D}$, $\mathcal{D}' \models Q(\mathbf{a})$ and*
2) *if $\mathsf{ch}_\mathcal{O}(\mathcal{D}') \models^{io} q_c(\mathbf{a})$, q_c a contraction of q, then it is not the case that there is an $\mathbf{S}$-database $\mathcal{D}''$ and a contraction q_c' of q such that $\mathcal{D}'' \to \mathcal{D}'$, $\mathsf{ch}_\mathcal{O}(\mathcal{D}'') \models^{io} q_c'(\mathbf{a})$, and $q_c' \prec q_c$.*

We consider now the database $\mathcal{D}_0$ obtained from Lemma 18 when Q is assumed to be Q_w, $\mathcal{D}$ to be $\mathcal{B}$ and $\mathbf{a}$ to be itself. We consider subsets $\mathcal{D}$ of $\mathcal{D}_0$ and construct $\mathcal{D}^+$ as in the Boolean case. We take such a $\mathcal{D}$ which is subset-minimal with the property that $\mathcal{D}^+ \models Q_w(\mathbf{a})$. As $\mathcal{D}^+ \to \mathcal{D}_0$, $\mathcal{D}_0 \to \mathcal{B}$ and $\mathcal{B} \not\models Q_a(\mathbf{a})$, it follows that $\mathcal{D}^+ \not\models Q_a(\mathbf{a})$. Further on, from Lemma 17 it follows that $(\mathcal{D}^+)_{\overline{\mathbf{a}}}$ has tree width exceeding ℓ. From the construction of $\mathcal{D}^+$ it follows that the tree width of $\mathcal{D}|_{\overline{\mathbf{a}}}$ exceeds ℓ, so it must contain the $k \times K$ grid as a minor. We apply Theorem 7 to $\mathcal{D}|_{\overline{\mathbf{a}}}$ and obtain a new database $(\mathcal{D}|_{\overline{\mathbf{a}}})_G$ and a homomorphism h_0 from $(\mathcal{D}|_{\overline{\mathbf{a}}})_G$ to $\mathcal{D}|_{\overline{\mathbf{a}}}$ such that Points 1 and 2 of that theorem are satisfied. We construct a new database $\mathcal{D}_G$ by reattaching the $\mathbf{a}$-part of $\mathcal{D}$ to $(\mathcal{D}|_{\overline{\mathbf{a}}})_G$ as follows:

- for every atom $A(a) \in \mathcal{D}$ with $a \in \mathbf{a}$, we add $A(a)$ to $(\mathcal{D}|_{\overline{\mathbf{a}}})_G$;
- for every atom $R(a_1, a_2) \in \mathcal{D}$ with $a_1, a_2 \in \mathbf{a}$, we add $R(a_1, a_2)$ to $(\mathcal{D}|_{\overline{\mathbf{a}}})_G$;
- for every atom $R(a, b)$ or $R(b, a)$ in $\mathcal{D}$ with $a \in \mathbf{a}$ and $b \in \mathsf{dom}(\mathcal{B}')$, and every $b' \in \mathsf{dom}((\mathcal{D}|_{\overline{\mathbf{a}}})_G)$ such that $h_0(b') = b$ we add $R(a, b')$ or $R(b', a)$ to $(\mathcal{D}|_{\overline{\mathbf{a}}})_G$;

It can be checked that $\mathcal{D}_G$ maps into $\mathcal{D}$ with an extension of the homomorphism h_0 with the identity homomorphism on $\mathbf{a}$. We construct $\mathcal{D}_G^+$ as in the original proof and than let $\mathcal{D}_G^*$ be the union of $\mathcal{D}^-$ and $\mathcal{D}_G^+$. It can be shown then that G has a k-clique iff $\mathcal{D}_G^* \models Q(\mathbf{a})$.

B. From CQs to UCQs

We next explain how to extend the proof from CQs to UCQs, that is, to the case where $\mathcal{Q} \subseteq (\mathcal{ELHI}_\bot, \text{UCQ})$. Let $Q = (\mathcal{O}, \mathbf{S}, q) \notin (\mathcal{ELHI}_\bot, \text{UCQ})^{\equiv}_{\text{UCQ}_\ell}$. Note that q is a disjunction of conjunctions of connected Boolean CQs (*conCQs*, for short), and that we can find an equivalent conjunction of disjunctions of conCQs q'. The conjuncts of $q' = q_1 \wedge \cdots \wedge q_n$ are disjunctions (unions) of connected Boolean CQs (*UconCQs*, for short). Let $Q_i = (\mathcal{O}, \mathbf{S}, q_i)$, for $1 \leq i \leq n$. We can assume w.l.o.g. that $Q_i \not\subseteq Q_j$ for all $i \neq j$. Also, some Q_v must be non-equivalent to its UCQ$_\ell$-approximation. In the remainder of the proof, the UconCQs $q_1, \ldots, q_{v-1}, q_{v+1}, \ldots, q_n$ play exactly the role of the conCQs $q_1, \ldots, q_{w-1}, q_{w+1}, \ldots, q_n$ in the original proof. Let us look more closely at the UconCQ q_v. Let $q_v = p_1 \vee \cdots \vee p_m$ and let $P_i = (\mathcal{O}, \mathbf{S}, p_i)$ for $1 \leq i \leq m$. We can assume w.l.o.g. that $P_i \not\subseteq P_j$ for all $i \neq j$. We also replace any p_i with its UCQ$_\ell$-approximation p_i^a whenever the OMQ

$$P_i^a := (\mathcal{O}, \mathbf{S}, p_1 \vee \cdots \vee p_{i-1} \vee p_i^a \vee p_{i+1} \vee \cdots \vee p_m)$$

is equivalent to Q_v. Since Q_v is not equivalent to its UCQ$_\ell$-approximation, these assumptions imply that there is a w such that $P_w \not\subseteq P_w^a$. Let P_a be the UCQ$_\ell$-approximation of P_w. Since $P_w \not\subseteq P_w^a$, there must be an $\mathbf{S}$-database $\mathcal{D}$ such that $\mathcal{D} \models P_w$, $\mathcal{D} \not\models P^a$, and $\mathcal{D} \not\models P_i$ for all $i \neq w$. In the remainder of the proof, the conCQ p_w plays the role of the conCQ q_w in the original proof. We next observe that the proof of Lemma 4 actually establishes a slightly stronger result, namely that for every OMQ Q^* from $(\mathcal{ELHI}_\bot, \text{CQ})$ over schema $\mathbf{S}$ and *every* $\mathbf{S}$-database $\mathcal{D}$ with $\mathcal{D} \models Q^*$, there is an $\mathbf{S}$-database $\mathcal{D}'$ such that $\mathcal{D}' \to \mathcal{D}$, $\mathcal{D}' \models Q^*$, and Condition 2 of Lemma 4 is satisfied (with Q_w replaced by Q^*).

For the remaining proof, we start with the database $\mathcal{D}_0$ that is obtained as $\mathcal{D}'$ when invoking the strengthened lemma with $Q = P_w$ and the database $\mathcal{D}$ that we had identified before. Then, $\mathcal{D}_0 \models P_w$, and from the fact that $\mathcal{D}_0 \to \mathcal{D}$, it follows that: $\mathcal{D}_0 \not\models P^a$, and $\mathcal{D}_0 \not\models P_i$, for every i, with $i \neq w$. The rest of the proof goes through with essentially no change.

Proofs for Section V

Towards a proof of Lemma 9, we first characterize answers to OMQs based on a weaker form of $\mathcal{D}$-labelings. Arguably, these are more intuitive, but also less 'local' than the $\mathcal{D}$-labelings defined in Section V.

A *weak $\mathcal{D}$-labeling* of q is a function $\ell : \mathsf{var}(q) \to \mathsf{dom}(\mathcal{D}) \cup \{\exists\}$ such that the following conditions are satisfied:

1) $\ell(x) \in \mathsf{dom}(\mathcal{D})$ for every answer variable x;
2) the restriction of ℓ to the variables in $V := \{x \mid \ell(x) \in \mathsf{dom}(\mathcal{D})\}$ is a homomorphism from $q|_V$ to $\mathsf{ch}^-_\mathcal{O}(\mathcal{D})$;
3) if $r(x, y) \in q$ and $\ell(y) \in \mathsf{dom}(\mathcal{D})$, then $\ell(x) \in \mathsf{dom}(\mathcal{D})$;
4) if $(x, y) \in G_2$, $\ell(x) \in \mathsf{dom}(\mathcal{D})$, and $\ell(y) = \exists$, then
   a) (x, y) is $\exists$-eligible;
   b) $\mathcal{D} \models (\mathcal{O}, \mathbf{S}, \mathsf{dtree}_{(x,y)})(\ell(x))$; and
   c) $\ell(x) = \ell(x')$ for all $x' \in \mathsf{reach}^0(x, y)$.
5) if q' is an $\exists$-MCC of q such that $\ell(x) \notin \mathsf{dom}(\mathcal{D})$ for every variable x in q', then q' is a homomorphic preimage of a ditree and $\mathcal{D} \models (\mathcal{O}, \mathbf{S}, \exists x_0 \, \mathsf{dtree}_{q'})$.

Lemma 19. *Let $\mathcal{D}$ be an $\mathbf{S}$-database that is consistent with $\mathcal{O}$ and $\mathbf{a} \in \mathsf{dom}(\mathcal{D})^{|\mathbf{x}|}$. Then $\mathcal{D} \models Q(\mathbf{a})$ iff there is a weak $\mathcal{D}$-labeling ℓ of $q(\mathbf{x})$ such that $\ell(\mathbf{x}) = \mathbf{a}$.*

Proof. (sketch) 'if'. Assume that ℓ is a weak $\mathcal{D}$-labeling of $q(\mathbf{x})$ such that $\ell(\mathbf{x}) = \mathbf{a}$. To show that $\mathcal{D} \models Q(\mathbf{a})$, it suffices to construct a homomorphism h from q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ with $h(\mathbf{x}) = \mathbf{a}$. We start by putting $h(x) = \ell(x)$ whenever $\ell(x) \in \mathsf{dom}(\mathcal{D})$. Next, consider every $(x,y) \in G_2$ such that $\ell(x) \in \mathsf{dom}(\mathcal{D})$ and $\ell(y) = \exists$. We extend h to all variables in $\mathsf{reach}(x,y)$ by using the homomorphism from $q|_{\mathsf{reach}(x.y)}$ to $\mathsf{dtree}_{(x,y)}$ (existence guaranteed by definition of $\exists$-eligible) and the homomorphism from $\mathsf{dtree}_{(x,y)}$ to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ that maps the root of $\mathsf{dtree}_{(x,y)}$ to $\ell(x)$ (existence guaranteed by Condition 4b). It remains to treat all $\exists$-MCCs q' of q. Here, we combine the homomorphism from q' to $\mathsf{dtree}_{q'}$ and from $\mathsf{dtree}_{q'}$ to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ (existence guaranteed by Condition 5). It can be verified that h is indeed a homomorphism.

'only if'. Assume that $\mathcal{D} \models Q(\mathbf{a})$. Then there is a homomorphism h from q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ with $h(\mathbf{x}) = \mathbf{a}$. For all variables x in q, define

$$\ell(x) = \begin{cases} h(x) & \text{if } h(x) \in \mathsf{dom}(\mathcal{D}); \\ \exists & \text{otherwise.} \end{cases}$$

It can be verified that ℓ is a weak $\mathcal{D}$-labeling of q with $\ell(\mathbf{x}) = \mathbf{a}$. For Condition 4, one uses that when $(x,y) \in G_2$, $\ell(x) \in \mathsf{dom}(\mathcal{D})$, and $\ell(y) = \exists$, then there is a homomorphism from $q|_{\mathsf{reach}(x,y)}$ to the database that the chase has generated below $\ell(x)$, which takes the form of a ditree with multi-edges; this shows that (x,y) is $\exists$-eligible. Condition 4b then follows from the choice of $\mathsf{dtree}_{(x,y)}$. □

The problem with weak $\mathcal{D}$-labelings is that Condition 4c is not yet sufficiently 'local', that is, the variables x and x' mentioned there can be arbitrarily far apart in a tree decomposition of q. This is rectified in (non-weak) $\mathcal{D}$-labelings as defined in Section V. We are now ready to prove correctness of the characterization of OMQ answers in terms of the latter.

Lemma 9. *For every $\mathbf{a} \in \mathsf{dom}(\mathcal{D})^{|\mathbf{x}|}$, $\mathcal{D} \models Q(\mathbf{a})$ iff there is a $\mathcal{D}$-labeling ℓ of $q(\mathbf{x})$ such that $\ell(\mathbf{x}) = \mathbf{a}$.*

Proof. (sketch) 'if'. Assume that there is a $\mathcal{D}$-labeling ℓ of $q(\mathbf{x})$ such that $\ell(\mathbf{x}) = \mathbf{a}$. Let ℓ' be obtained from ℓ by setting $\ell'(x) = \exists$ iff $\ell(x) \notin \mathsf{dom}(\mathcal{D})$. It suffices to show that ℓ' is a weak $\mathcal{D}$-labeling of q. The only condition that is not immediate is Condition 4c of weak $\mathcal{D}$-labelings. So assume that $(x,y) \in G_2$, $\ell(x) \in \mathsf{dom}(\mathcal{D})$, and $\ell(y) = \exists$. Let $x' \in \mathsf{reach}^0(x,y)$. By Condition 4c of $\mathcal{D}$-labelings, $\ell(y) = ((x'',y''),\ell(x))$ where $x'' \in \mathsf{reach}^0(x,y)$ and $y'' \in \mathsf{reach}^1(x,y)$. By Conditions 5 to 7 of $\mathcal{D}$-labelings and since it can be easily proved that $\mathsf{reach}^j(x,y) = \mathsf{reach}^j(x'',y'')$ for all j, we obtain $\ell(x') = \ell(x)$ as required.

'only if'. Assume that $\mathcal{D} \models Q(\mathbf{a})$. Then there is a homomorphism h from q to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$ with $h(\mathbf{x}) = \mathbf{a}$. For each variable z such that $h(z) \notin \mathsf{dom}(\mathcal{D})$ and there is an $(x,y) \in G_2$ for which $h(x) \in \mathsf{dom}(\mathcal{D})$, $h(y) \notin \mathsf{dom}(\mathcal{D})$, Conditions 4a and 4b of $\mathcal{D}$-labelings are satisfied, and $z \in \mathsf{reach}(y)$, choose such an (x,y), and denote it with (u_z,v_z) For all variables x in q, define

$$\ell(x) = \begin{cases} h(x) & \text{if } h(x) \in \mathsf{dom}(\mathcal{D}); \\ ((u_z,v_z),h(u_z)) & \text{if } h(x) \notin \mathsf{dom}(\mathcal{D}) \\ & \quad \text{and } u_z, v_z \text{ are defined} \\ \exists & \text{otherwise.} \end{cases}$$

It can be verified that ℓ is a $\mathcal{D}$-labeling of q with $\ell(\mathbf{x}) = \mathbf{a}$. □

Theorem 9. EVALUATION$((\mathcal{ELH}^{dr}_{\bot}, UCQ)^{\equiv}_{UCQ_k})$ *based on the full schema is in* PTIME *combined complexity, for any $k \geq 1$.*

Proof. Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ be from $(\mathcal{ELH}^{dr}_{\bot}, \mathrm{CQ})^{\equiv}_{\mathrm{UCQ}_k}$, $Q_f = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_f)$ a full rewriting of Q, $\mathcal{D}$ the input database that is consistent with $\mathcal{O}$, and $\mathbf{a}$ a candidate answer. By Theorem 3, q_f is of tree width bounded by k. Ideally, we would like to play the modified game on q_f and answer 'yes' if Duplicator has a winning strategy for any of these CQs.

However, we do not have a full rewriting in our hands as we have no way of computing one in PTIME. To solve this problem, we first extend q as follows: for each variable x in q and each concept inclusion $C \sqsubseteq D \in \mathcal{O}$ with $\mathcal{D}_q \models C(x)$, x viewed as a constant, take a fresh copy q_C of C viewed as a CQ and add q_C to q, identifying x with the root of q_C. Note that Q is equivalent to $Q^+ = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q^+)$ and that by construction, q_f syntactically is a subquery of the resulting CQ q^+. We play the modified game on q^+ rather than on q_f.

We have to argue that Duplicator has a winning strategy on q^+ iff $\mathcal{D} \models Q(\mathbf{a})$. The 'if' direction is clear since $\mathcal{D} \models Q(\mathbf{a})$ implies $\mathcal{D} \models \mathcal{Q}^+(\mathbf{a})$, thus Lemma 9 yields a $\mathcal{D}$-labeling ℓ of q^+ with $\ell(\mathbf{x}) = \mathbf{a}$, and ℓ clearly gives rise to a winning strategy for Duplicator on q^+. Conversely, a winning strategy for Duplicator on q^+ also gives such a strategy on any subquery of q^+, such as q_f. Thus, there is a $\mathcal{D}$-labeling ℓ of q_f, which means that $\mathcal{D} \models Q_f(\mathbf{a})$ and thus $\mathcal{D} \models Q(\mathbf{a})$. □

PROOFS FOR SECTION VI

Theorem 10. *For any $k \geq 1$, UCQ_k-equivalence is*

1) EXPTIME*-hard in $(\mathcal{EL}, CQ)$;*
2) 2EXPTIME*-hard in $(\mathcal{ELI}, CQ)$;*
3) Π^p_2-hard in (DL-Lite$^{\mathcal{R}}$, CQ).

The same lower bounds apply to CQ_k-equivalence, both while preserving the ontology and in the general case.

Proof. Point 1 is proved by reduction from the following problem: given an OMQ of the form $Q = (\mathcal{O}, \mathbf{S}, A(x))$ with $\mathcal{O}$ formulated in $\mathcal{EL}_{\bot}$, is Q empty? For Q to be empty there must be no $\mathbf{S}$-database $\mathcal{D}$ that is consistent with $\mathcal{O}$ and which satisfies $\mathcal{D} \models Q(a)$ with $a \in \mathsf{dom}(\mathcal{D})$. This problem is known to be EXPTIME-hard [3].

We start with the case $k = 1$, that is, we consider UCQ$_1$-equivalence and CQ$_1$-equivalence, the latter both while preserving the ontology and in the general case. We use the same reduction for all three cases and afterwards explain how to generalize to $k > 1$.

Let $Q = (\mathcal{O}, \mathbf{S}, A(x))$ be as stated above. Reserve fresh concept names B, B_1, B_2 and a fresh role name r. Let $\mathcal{O}^*$ be

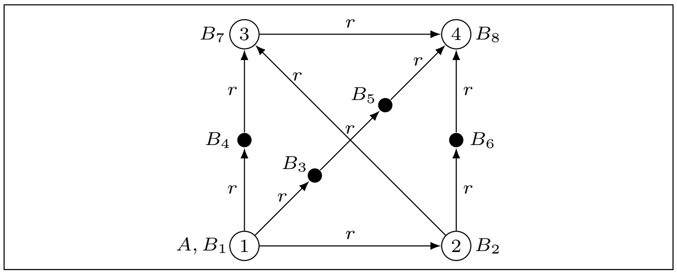

Fig. 3. CQ for $k = 2$

obtained from $\mathcal{O}$ by replacing every concept inclusion of the form $C \sqsubseteq \bot$ with $C \sqsubseteq B$. Now define

$$\begin{array}{rcl} \mathcal{O}' & = & \mathcal{O}^* \cup \{\exists s.B \sqsubseteq B \mid s \text{ role name in } \mathcal{O}\} \cup \\ & & \{B \sqsubseteq A \sqcap \exists r.(B_1 \sqcap B_2 \sqcap \exists r.\top)\} \\ \mathbf{S}' & = & \mathbf{S} \cup \{r, B_1, B_2\} \\ q(x) & = & \exists y_1 \exists y_2 \exists z\, A(x) \wedge r(x, y_1) \wedge r(x, y_2) \wedge B_1(y_1) \wedge \\ & & B_2(y_2) \wedge r(y_1, z) \wedge r(y_2, z) \end{array}$$

and set $Q' = (\mathcal{O}', \mathbf{S}', q)$.

Claim. Q is empty iff Q' is (U)CQ$_1$-equivalent (while preserving the ontology or not).

For (the contrapositive of) 'if', assume that Q is non-empty and take an $\mathbf{S}$-database $\mathcal{D}_0$ that is consistent with $\mathcal{O}$ and satisfies $\mathcal{D}_0 \models A(a)$. Glue to a in $\mathcal{D}_0$ a copy of q without the atom $A(x)$ and call the resulting $\mathbf{S}'$-database $\mathcal{D}$. Clearly, $\mathcal{D} \models Q'(a)$. Let $\mathcal{D}^u_a$ be the 1-unraveling of $\mathcal{D}$ up to a. Then $\mathcal{D}^u_a \not\models Q'(a)$ since the copy of q that we have glued to $\mathcal{D}_0$ has been 'broken' by unraveling, B_1, B_2 do not occur in $\mathcal{O}^*$, and the concept inclusion $B \sqsubseteq A \sqcap \exists r.(B_1 \sqcap B_2 \sqcap \exists r.\top)$ in $\mathcal{O}'$ cannot fire since consistency of $\mathcal{D}_0$ with $\mathcal{O}$ and Lemma 15 imply that $\mathcal{O}$ derives B at a in $\mathcal{D}^u_a$. But by Lemma 15, no OMQ from $(\mathcal{ELHI}_\bot, \text{UCQ}_1)$ can distinguish a in $\mathcal{D}$ from a in $\mathcal{D}^u_a$. Consequently, Q' is not UCQ$_1$-equivalent.

For 'only if', assume that Q is empty. We show that Q' is equivalent to $(\mathcal{O}', \mathbf{S}', B(x))$ and thus CQ$_1$-equivalent while preserving the ontology. The containment $(\mathcal{O}', \mathbf{S}', B(x)) \subseteq Q'$ is immediate by construction of Q'. For the converse, let $\mathcal{D}$ be an $\mathbf{S}'$-database with $\mathcal{D} \models Q'(a)$. This clearly implies that $\mathcal{O}$ derives A at a in $\mathcal{D}$. Since Q is empty, this is only possible when B is also derived at a. Thus $\mathcal{D} \models (\mathcal{O}', \mathbf{S}', B(x))(a)$.

Sketch for UCQ$_k$-case, $k > 1$. The main properties of CQ q in the above proof are that it is of tree width at least $k+1$, homomorphically maps into a directed tree (that can be generated by an $\mathcal{EL}$-concept), and does not admit a homomorphism to its k-unraveling. To achieve the same for $k > 1$, we can use as q the undirected $k+2$-clique whose vertices we assume w.l.o.g. to be $\{1, \dots, k+2\}$, orient each edge $\{i, j\}$ in the direction from i to j if $i < j$, subdivide each edge (i, j) into $j - i$ edges by introducing intermediate points, represent directed edges as r-atoms, add the atom $A(1)$, and finally label each vertex with a different concept name B_i, $i \geq 1$. For the case $k = 2$, the resulting CQ is displayed in Figure 3. The reduction can then be adapted by replacing q as described, replacing B_1, B_2 in $\mathbf{S}'$ with the set of all fresh concept names B_i in q, and replacing the concept $A \sqcap \exists r.(B_1 \sqcap B_2 \sqcap \exists r.\top)$ on the right-hand side of the concept inclusion in the second line of the definition of $\mathcal{O}'$ with some $\mathcal{EL}$-concept C such that $a \in C^\mathcal{I}$ implies $\mathcal{I} \models q(a)$ for all interpretations $\mathcal{I}$. Such a concept can be obtained by identifying all vertices that are reachable from vertex 1 in exactly i steps, for each i. In the example case $k = 2$, it takes the form

$$A \sqcap B_1 \sqcap \exists r.(B_2 \sqcap B_3 \sqcap B_4 \sqcap \exists r.(B_5 \sqcap B_6 \sqcap B_7 \sqcap \exists r.B_8)).$$

In the proof of the 'if' direction of the claim, 1-unravelings are replaced with k-unravelings. Note that q does not admit a homomorphism to its k-unraveling: since q has tree width at least $k+1$, such a homomorphism would have to be non-injective and thus the k-unraveling of q would have to comprise atoms $B_i(x), B_j(x)$ for some i, j with $i \neq j$.

Now for Point 2 of Theorem 10. The following was established in the proof of Theorem 48 in [7].

Theorem 16. *Let $M = (Q, \Sigma, \Gamma, q_0, \Delta)$ be an exponentially space bounded alternating Turing machine and let $k = |\Gamma| - 1$. Given an input w to M, one can construct in polynomial time a Boolean OMQ $Q_w = (\mathcal{O}_w, \mathbf{S}_w, q_w)$ from $(\mathcal{ELI}, CQ)$ such that for a selected concept name $A^* \notin \mathbf{S}_w$, the following conditions are satisfied:*

1) *M accepts w iff there is an $\mathbf{S}_w$-database $\mathcal{D}$ and an $a \in \mathsf{dom}(\mathcal{D})$ such that $\mathcal{D} \models (\mathcal{O}_w, \mathbf{S}_w, A^*(x))(a)$ and $\mathcal{D} \not\models Q_w$;*
2) *q_w is connected, uses only symbols from $\mathbf{S}_w$, is of tree width k, and not equivalent to any CQ of smaller tree width;*
3) *the restriction of $\mathsf{ch}_{\mathcal{O}_w}(\mathcal{D}_{q_w})$ to symbols in $\mathbf{S}_w$ is $\mathcal{D}_{q_w}$.*

We remark that tree width is not explicitly mentioned in [7]. However, the CQ q_w constructed there contains a subquery p whose Gaifman graph contains as a minor the complete balanced bipartite graph $K_{k+1,k+1}$ with exactly one adjacent edge dropped from each vertex. It is not hard to verify that such a graph has degeneracy at least k, thus tree width at least k. Moreover, the Gaifman graph of p is a core, thus p is not equivalent to a CQ of tree width smaller than k.

Let $k \geq 1$. We can choose $M = (Q, \Sigma, \Gamma, q_0, \Delta)$ to have a 2ExpTime-hard word problem and satisfying $|\Gamma| > k + 1$. We reduce the word problem for M, as follows. Let w be an input to M, and let $Q_w = (\mathcal{O}_w, \mathbf{S}_w, q_w)$ be as in Theorem 16. Define $q(x) = A^*(x) \wedge q_w$ where x is a fresh variable and let $Q = (\mathcal{O}_w, \mathbf{S}_w, q)$.

Claim M accepts w iff Q is not (U)CQ$_k$-equivalent (while preserving the ontology or not).

For 'only if', assume that M accepts w. By Point 1 of Theorem 16, there is an $\mathbf{S}_w$-database $\mathcal{D}_0$ and an $a \in \mathsf{dom}(\mathcal{D}_0)$ such that $\mathcal{D}_0 \models (\mathcal{O}_w, \mathbf{S}_w, A^*(x))(a)$ and $\mathcal{D}_0 \not\models Q_w$. Let $\mathcal{D}$ be the disjoint union of $\mathcal{D}_0$ and $\mathcal{D}_{q_w}$. Then $\mathcal{D} \models Q(a)$. Now assume to the contrary of what is to be shown that Q is UCQ$_k$-equivalent. Then Q is equivalent to its UCQ$_k$-approximation $Q_a = (\mathcal{O}_w, \mathbf{S}_w, q_a)$. Clearly, we can remove from q_a any CQ in which the variable x from q is identified with any other variable, without compromising equivalence or altering tree width. Thus each CQ in q_a takes the form $A^*(x) \wedge q'_w$ where q'_w is a contraction of q_w. Since q_w is connected, so is q'_w. Let Q'_a be the Boolean OMQ $(\mathcal{O}_w, \mathbf{S}_w, q'_a)$ where q'_a consists of the q'_w-parts of the CQs in q_a. Clearly, Q'_a is just the UCQ$_k$-approximation of Q_w. From $\mathcal{D} \models Q(a)$ and $\mathcal{D}_0 \not\models Q_w$, we thus obtain $\mathcal{D} \models Q_a$ and $\mathcal{D}_0 \not\models Q'_a$. Consequently, $\mathcal{D}_{q_w} \models Q'_a$. Since q_w and thus q'_w uses only symbols from $\mathbf{S}_w$ and by Point 3 of Theorem 16, this implies that there is a homomorphism from q'_w to q_w. This means that q'_w is equivalent to q_w, but q'_w is of tree width k, in contradiction to Point 2 of Theorem 16.

For 'if', assume that M does not accept w. By Point 1 of Theorem 16, $\mathcal{D} \models (\mathcal{O}_w, \mathbf{S}_w, A^*(x))(a)$ implies $\mathcal{D} \models Q_w$ for any $\mathbf{S}_w$-database $\mathcal{D}$ and $a \in \mathsf{dom}(\mathcal{D})$. Consequently, Q is equivalent to $(\mathcal{O}_w, \mathbf{S}_w, A^*(x))$ and thus CQ$_1$-equivalent while preserving the ontology.

We finally address Point 3 of Theorem 10. It is proved in [10] that containment in (DL-Lite$^\mathcal{R}$, CQ) is Π^p_2-complete. However, it is rather unclear how to utilize the hardness proof given in that paper for our purposes. We instead give a direct proof by reduction from $\forall\exists$-QBF, building on and extending an NP-hardness proof for the combined complexity of (a restricted version of) query evaluation in (DL-Lite$^\mathcal{R}$, CQ) given in [28]. To make the presentation more digestible, we first show that containment between unary OMQs $(\mathcal{O}, \mathbf{S}, C(x))$ and $(\mathcal{O}, \mathbf{S}, q(x))$, where $\mathcal{O}$ is a DL-Lite$^\mathcal{R}$-ontology and C a conjunction of concept names, is Π^p_2-hard.

Thus let $\varphi = \forall \mathbf{x} \exists \mathbf{y}\, (C_1 \vee \cdots \vee C_\ell)$ be a sentence of $\forall\exists$-QBF where each $C_1, \dots, C_\ell$ is a clause. Further let $\mathbf{x} = x_1 \cdots x_n$ and $\mathbf{y} = y_1 \cdots y_m$. Let the DL-Lite$^\mathcal{R}$-ontology $\mathcal{O}$ contain the following:

$$\begin{array}{rcll}
T_i & \sqsubseteq & X_i & \text{for } 1 \leq i \leq n \\
F_i & \sqsubseteq & X_i & \text{for } 1 \leq i \leq n \\
T_i & \sqsubseteq & C_j & \text{if } x_i \in C_j \\
F_i & \sqsubseteq & C_i & \text{if } \neg x_i \in C_j \\
L_i & \sqsubseteq & \exists r.(L_{i+1} \sqcap Y_{i+1}) & \text{for } 0 \leq i \leq m \\
L_i & \sqsubseteq & \exists r.(L_{i+1} \sqcap \overline{Y}_{i+1}) & \text{for } 0 \leq i \leq m \\
Y_i & \sqsubseteq & \exists r^-.C^+_j & \text{if } y_i \in C_j \\
\overline{Y}_i & \sqsubseteq & \exists r^-.C^+_j & \text{if } \neg y_i \in C_j \\
C^+_i & \sqsubseteq & C_i & \text{for } 1 \leq i \leq \ell \\
C^+_i & \sqsubseteq & \exists r^-.C^+_i & \text{for } 1 \leq i \leq \ell
\end{array}$$

where, as usual in DL-Lite$^\mathcal{R}$, $A \sqsubseteq \exists r.(B_1 \sqcap \cdots \sqcap B_n)$ abbreviates $A \sqsubseteq \exists r_0.\top, r_0 \sqsubseteq r, \exists r_0^-.\top \sqsubseteq B_1, \dots, \exists r_0^-.\top \sqsubseteq B_n$, where r_0 is a fresh role name. Set

$$\mathbf{S} = \{L_0, T_1, \dots, T_n, F_1, \dots, F_n\}$$

and define the CQ $q(x_0)$ to be as follows, all variables except x_0 existentially quantified:

$$\begin{array}{c}
r(x_0, x_1) \wedge \cdots \wedge r(x_{m-1}, x_m) \wedge \\
C_1(z^1_0) \wedge r(z^1_0, z^1_1) \wedge \cdots \wedge r(z^1_{m-1}, x_m) \wedge \\
\cdots \\
C_\ell(z^\ell_0) \wedge r(z^\ell_0, z^\ell_1) \wedge \cdots \wedge r(z^\ell_{m-1}, x_m).
\end{array}$$

Let $Q_1 = (\mathcal{O}, \mathbf{S}, X_1(x) \wedge \cdots \wedge X_n(x))$ and $Q_2 = (\mathcal{O}, \mathbf{S}, q)$. Then we have the following, which implies Π^p_2-hardness of the containment question mentioned above.

Claim 1. $Q_1 \subseteq Q_2$ iff φ is true.

Proof of claim. For the 'if' direction, assume that $Q_1 \subseteq Q_2$. Let π be a truth assignment for the variables $\mathbf{x}$. We have to show that π can be extended to the variables in $\mathbf{y}$ such that $C_1 \vee \cdots \vee C_\ell$ is satisfied. Let $\mathcal{D}_\pi$ be the $\mathbf{S}$-database that contains the fact $T_i(a)$ if $\pi(x_i) = 1$ and $F_i(a)$ if $\pi(x_i) = 0$. Clearly, $\mathcal{D}_\pi \models Q_1(a)$, and thus $\mathcal{D}_\pi \models Q_2(a)$, that is, there is a homomorphism h from q to $\mathsf{ch}_\mathcal{O}(\mathcal{D}_\pi)$ with $h(x_0) = a$. It is easy to see that $\mathsf{ch}_\mathcal{O}(\mathcal{D}_\pi)$ takes the form of a binary tree of depth m with root a in which every path p corresponds to a truth assignment π_p to the variables in $\mathbf{y}$, and vice versa: the node on level i is labeled with (exactly one of) Y_i or $\overline{Y}_i$ and $\pi_p(y_i) = 1$ iff the former is the case. By construction of q, the variable x_m of q must be mapped to the final node of a path p, and we extend π with π_p. It can be verfied that the use of the concept names C_j in $\mathcal{O}$ and q imply that π satisfies all clauses from the QBF.

For the 'only if' direction, assume that φ is true. Let $\mathcal{D}$ be an $\mathbf{S}$-database with $\mathcal{D} \models Q_1(a)$. We produce a homomorphism h from q to $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ such that $h(x_0) = a$. Since $\mathcal{D} \models Q_1(a)$ and $X_1, \dots, X_\ell \notin \mathbf{S}$, we must have $T_i(a)$ or $F_i(a)$ (or both) in $\mathcal{D}$ for $1 \leq i \leq n$. Let $\pi_\mathcal{D}$ be a truth assignment such that $\pi_\mathcal{D}(x_i) = 1$ implies $T_i(a) \in \mathcal{D}$ and $\pi_\mathcal{D}(x_i) = 0$ implies $F_i(a) \in \mathcal{D}$. Since φ is true, we can extend $\pi_\mathcal{D}$ to a truth assignment to $\mathbf{y}$ such that all clauses are satisfied. Thus truth assignment identifies a path in $\mathsf{ch}_\mathcal{O}(\mathcal{D})$ in the subtree database rooted at a. Then h can map the variables $x_0, \dots, x_m$ from q to that path. Since all clauses are satisfied, the remaining paths in q can also be mapped.

We next modify the Π^p_2-hardness proof just given so that it applies to UCQ$_k$-equivalence in (DL-Lite$^\mathcal{R}_\mathsf{horn}$, CQ). Fix some $k \geq 1$. We would like to reuse essentially the same CQ as before, but now we have to make sure that it is of tree width exceeding k. To achieve this, we mix in the CQ from the proof of Point 1 of Theorem 10 that we had obtained by starting with the $k+2$-clique based on vertices $\{1, \dots, k+2\}$, orienting each edge $\{i, j\}$ in the direction from i to j if $i < j$, subdividing each edge (i, j) into $j - i$ edges by introducing intermediate points, representing directed edges as r-atoms, and finally labeling each vertex with a different concept name B_i, $i \geq 1$. All variables are unquantified, that is, the resulting CQ p is just a set of atoms. By identifying all vertices that are reachable from vertex 1 in exactly i steps, for each i, we obtain a path-shaped contraction of p that can be represented as an $\mathcal{EL}$-concept

$$C = C_1 \sqcap \exists r.(C_2 \sqcap \exists r.(C_3 \sqcap \cdots \sqcap \exists r.C_{k+2}) \cdots)$$

such that $a \in C^{\mathcal{I}}$ implies $\mathcal{I} \models p(a)$ for all interpretations $\mathcal{I}$. Let $\mathcal{O}$ be the ontology from the previous reduction extended by the following concept inclusions:

$$\begin{array}{rcll} I_i & \sqsubseteq & C_i \sqcap \exists r.I_{i+1} & \text{for } 1 \leq i \leq k+1 \\ I_{k+2} & \sqsubseteq & L_0 & \\ B & \sqsubseteq & C_i & \text{for } 1 \leq i \leq \ell \\ B & \sqsubseteq & \exists r.B & \\ B & \sqsubseteq & \exists r^-.B & \end{array}$$

Let $\mathbf{S}$ consist of $\{I_1, T_1, \ldots, T_n, F_1, \ldots, F_n, r, B\}$ and all concept names B_i introduced in the construction of the CQ p. Assume that the variable in p that corresponds to vertex 1 from the original clique is x_0 and the vertex that corresponds to vertex $k+2$ is x_{k+1}. Define the CQ $q(x_0)$ to be as follows, all variables except x_0 existentially quantified, $w = m+k+2$:

$$\begin{array}{l} I_1(x_0) \wedge X_1(x_0) \wedge \cdots \wedge X_n(x_0) \wedge \\ p \wedge \\ r(x_{k+1}, x_{k+2}) \wedge \cdots \wedge r(x_{w-1}, x_w) \wedge \\ C_1(z_0^1) \wedge r(z_0^1, z_1^1) \wedge \cdots \wedge r(z_{w-1}^1, x_w) \wedge \\ \qquad\qquad\qquad \cdots \\ C_\ell(z_0^\ell) \wedge r(z_0^\ell, z_1^\ell) \wedge \cdots \wedge r(z_{w-1}^\ell, x_w). \end{array}$$

Claim 2. $Q = (\mathcal{O}, \mathbf{S}, q)$ is UCQ$_k$-equivalent iff φ is true.

In fact, it can be verified that Q is equivalent to $(\mathcal{O}, \mathbf{S}, q')$ if φ is true, where q' is the CQ that consists of the first line of the definition of q. Conversely, if φ is false, then consider the following CQ, viewed as a database $\mathcal{D}$:

$$I_1(x_0) \wedge X_1(x_0) \wedge \cdots \wedge X_n(x_0) \ \wedge p \wedge B(x_{k+1}).$$

It is easy to see that $\mathcal{D}$ has tree width $k+1$ and that $\mathcal{D} \models Q(x_0)$, x_0 meaning the constant of the same name in $\mathcal{D}$ here. It can also be verified that the k-unraveling $\mathcal{D}'$ of $\mathcal{D}$ is such that $\mathcal{D}' \not\models Q(x_0)$. Together with Corollary 1 and Theorem 2, this implies that Q is not UCQ$_k$-equivalent. □

In the following proof, we are going to make use of complexity results for OMQ containment from the literature. Recall that, in this paper, $Q_1 \subseteq Q_2$ with $Q_i = (\mathcal{O}_i, \mathbf{S}, q_i)$ if $Q_1(\mathcal{D}) \subseteq Q_2(\mathcal{D})$ for all $\mathbf{S}$-databases $\mathcal{D}$ including those that are inconsistent with $\mathcal{O}_1$ or $\mathcal{O}_2$. In the literature on containment, in contrast, it is common to consider only those $\mathcal{D}$ that are consistent with both $\mathcal{O}_1$ and $\mathcal{O}_2$. We refer to this as *consistent containment*. In the proof that follows, the actual queries are UCQs and $\mathcal{O}_1$ and $\mathcal{O}_2$ are the same ontology $\mathcal{O}$. In this case, the gap between containment and consistent containment is unproblematic since we can reduce containment to consistent containment in polynomial time, as follows. Let $\mathcal{O}'$ be obtained from $\mathcal{O}$ by

1) replacing every CI $C \sqsubseteq \bot$ with $C \sqsubseteq B$
2) adding $\top \sqsubseteq A$

where A and B are fresh concept names. Moreover, if $\mathbf{x} = x_1 \cdots x_n$ are the answer variables in q_1 and q_2 (which we can w.l.o.g. assume to be identical), then let q_i' be obtained from q_i by adding as an additional disjunct the CQ $A(x_1) \wedge \cdots \wedge A(x_n) \wedge \exists y\, B(y)$, for $i \in \{1, 2\}$. It can be shown that $Q_1 \subseteq Q_2$ if $(\mathcal{O}', \mathbf{S}, q_1')$ is consistently contained in $(\mathcal{O}', \mathbf{S}, q_2')$. A crucial observation is that every $\mathbf{S}$-database is consistent with $\mathcal{O}'$.

Theorem 11. *For any $k \geq 1$, UCQ$_k$-equivalence is*

1) in ExpTime *in $(\mathcal{ELH}_\bot^{dr}, UCQ)$;*
2) in 2ExpTime *in $(\mathcal{ELHI}_\bot, UCQ)$;*
3) in Π_2^p in (DL-Lite$^{\mathcal{R}}_{\mathsf{horn}}$, UCQ).

Proof. Points 1 and 2 are proved in a uniform way. By Corollary 1, it suffices to construct the UCQ$_k$-approximation $Q_a = (\mathcal{O}, \mathbf{S}, q_a)$ of the input query $Q = (\mathcal{O}, \mathbf{S}, q)$, and check whether $Q \subseteq Q_a$ (the converse containment holds by construction of Q_a). An approach based on alternating tree automata has been used in [7] to show that OMQ containment is ExpTime-complete in $(\mathcal{ELH}_\bot, \text{CQ})$ and 2ExpTime-complete in $(\mathcal{ELHI}_\bot, \text{CQ})$. It is an easy exercise, and does not require any new ideas, to extend this approach from CQs to UCQs, and from $\mathcal{ELH}_\bot$ to $\mathcal{ELH}_\bot^{dr}$. Applying the resulting decision procedures for containment as a black box, we obtain a 2ExpTime upper bound for $(\mathcal{ELH}_\bot^{dr}, \text{UCQ})$ and a 3ExpTime upper bound for $(\mathcal{ELHI}_\bot, \text{UCQ})$. To lower these bounds by one exponential, we have to address the fact that q_a has exponentially many disjuncts (each of polynomial size). This requires another minor change in the decision procedure for containment, exploiting that every collapsing of a CQ in q_a is also a collapsing of q. We give more details in what follows.

The central relevant statement from [7] is as follows.

Theorem 17. *For every OMQ $Q = (\mathcal{O}, \Sigma, q)$ from $(\mathcal{ELHI}_\bot, CQ)$ with q Boolean, there is a two-way alternating parity tree automaton $\mathfrak{A}_Q$ that accepts a $(|\mathcal{O}| \cdot |q|)$-ary $\Sigma_\varepsilon \cup \Sigma_N$-labeled tree (T, L) iff it is proper, $\mathcal{D}_{(T,L)}$ is consistent with $\mathcal{O}$ and $\mathcal{D}_{(T,L)} \models Q$. $\mathfrak{A}_Q$ has at most $2^{p(|q|+\log(|\mathcal{O}|))}$ states, and at most $p(|q| + |\mathcal{O}|)$ states if $\mathcal{O}$ is an $\mathcal{ELH}_\bot^{dr}$-ontology, p a polynomial. It can be constructed in time polynomial in its size.*

Here, $\Sigma_\varepsilon \cup \Sigma_N$ are suitable alphabets such that, among other things, a $\Sigma_\varepsilon \cup \Sigma_N$-labeled tree (T, L) represents an (almost) tree-shaped database $\mathcal{D}_{(T,L)}$. The term ‘proper’ refers to a technical condition that need not bother us here. The construction of the automaton $\mathfrak{A}_Q$ from Theorem 17 relies on the notion of forest decompositions, which partitions a query into a center part and several tree-shaped parts.

A *forest decomposition* of q is a tuple $F = (q_0, q_1, x_1, \ldots, q_k, x_k, \mu)$ where $(q_0, q_1, \ldots, q_k)$ is a partition of (the atoms of) a contraction of q, $x_1, \ldots, x_k$ are variables from q_0, and μ is a mapping from $\mathsf{var}(q_0)$ to a fixed set of constants dom_0 such that the following conditions are satisfied for $1 \leq i, j \leq k$;

1) q_0 is non-empty;
2) q_i is weakly tree-shaped with root x_i, that is, G_q is a tree (multi-edges allowed);
3) $\mathsf{var}(q_i) \cap \mathsf{var}(q_0) = \{x_i\}$;
4) $\mathsf{var}(q_i) \cap \mathsf{var}(q_j) \subseteq \mathsf{var}(q_0)$ if $i \neq j$;
5) q_i contains no atom $A(x_i)$;
6) x_i has a single successor in q_i.

The central property of forest decompositions is then as follows.

Lemma 20. *Let $\mathcal{O}$ be an $\mathcal{ELHI}_\bot$-ontology, (T, L) a proper $\Sigma_\varepsilon \cup \Sigma_N$-labeled tree, $\mathcal{C}$ the part of $\mathcal{D}_{(T,L)}$ represented by the root vertex of T, and q a Boolean connected CQ. Then the following are equivalent:*

1) *there is a homomorphism h from q to $\mathsf{ch}_\mathcal{O}(\mathcal{D}_{(T,L)})$ whose range has a non-empty intersection with $\mathsf{dom}(\mathcal{C})$;*
2) *there is a forest decomposition $F = (q_0, q_1, x_1, \dots, q_k, x_k, \mu)$ of q such that*
   - *μ is a homomorphism from q_0 to $\mathsf{ch}_\mathcal{O}(\mathcal{D}_{(T,L)})$ whose range falls within $\mathsf{dom}(\mathcal{C})$;*
   - *there is a homomorphism h_i from q_i to $\mathsf{ch}_\mathcal{O}(\mathcal{D}_{(T,L)})$ such that $h_i(x_i) = \mu(x_i)$, for $1 \leq i \leq k$.*

In the construction of $\mathfrak{A}_Q$, the transition relation contains a disjunction over all forest decompositions of the input query q. We, however, are not interested in the CQ q_a rather than in q. But this is easy to achieve: instead of using all forest decompositions of q in the mentioned disjunction, we use all forest decompositions of a CQ from q_a. Because of the use of contractions in the definition of forest decompositions, each such decomposition is also a forest decomposition of q, and consequently no further modifications of the construction are required.

For Point 3, we again have to check whether $Q \subseteq Q_a = (\mathcal{O}, \mathbf{S}, q_a)$. It was shown in [10] that containment in (DL-Lite$_{\mathsf{horn}}$, CQ) is Π_2^p-complete and the proof extends to (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$, UCQ). It thus again remains to deal with the fact that q_a consists of exponentially many CQs (of polynomial size). We sketch the proof of a Σ_2^p upper bound for checking non-containment, that is $Q \not\subseteq Q_a$. We will make use of the fact that OMQ evaluation in (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$, UCQ) is NP-complete [15], [22].

A pair $(\mathcal{D}, \mathbf{a})$ with $\mathcal{D}$ an $\mathbf{S}$-database and $\mathbf{a}$ a tuple over $\mathsf{dom}(\mathcal{D})$ is a *witness* for $Q \not\subseteq Q_a$ if $\mathcal{D} \models Q(\mathbf{a})$ and $\mathcal{D} \not\models Q_a(\mathbf{a})$. It is observed in [10] that it suffices to consider witnesses $(\mathcal{D}, \mathbf{a})$ where the number of constants in $\mathcal{D}$ is bounded by $|q| \cdot (|\Sigma|+1)$. To decide whether $Q \not\subseteq Q_a$, we guess a witness $(\mathcal{D}, \mathbf{a})$ of such dimension. We then verify in NP that $\mathcal{D} \models Q(\mathbf{a})$, co-guess a CQ p from q_a, and verify in coNP that $\mathcal{D} \not\models (\mathcal{O}, \mathbf{S}, p)(\mathbf{a})$. With co-guessing a CQ p' in q_a, we mean to co-guess an equivalence relation on q that represents variable identifications, to then produce p' in polynomial time, and to verify in polynomial time that it has tree width k (note that k is fixed). The overall algorithm clearly runs in Σ_2^p, as desired. □

Theorem 12. *For any $k \geq 1$, and OMQs based on the full schema, (U)CQ$_k$-equivalence is complete for*

1) NP *between ($\mathcal{EL}$, CQ) and ($\mathcal{ELH}^{dr}_\bot$, UCQ);*
2) ExpTime *between ($\mathcal{ELI}$, CQ) and ($\mathcal{ELHI}_\bot$, UCQ);*
3) NP *between (DL-Lite$^\mathcal{R}$, CQ) and (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$).*

Proof. The NP lower bounds are inherited from the case where the ontology is empty [18], while the ExpTime lower bound is proved by a reduction from the subsumption problem in $\mathcal{ELI}$, namely given an $\mathcal{ELI}$-ontology $\mathcal{O}$ and concept names A, B, is A subsumed by B w.r.t. $\mathcal{O}$ (written $\mathcal{O} \models A \sqsubseteq B$), that is, is $A^\mathcal{I} \subseteq B^\mathcal{I}$ in every model $\mathcal{I}$ of $\mathcal{O}$? This problem is known to be ExpTime-complete [5]. We start with the case $k = 1$. Thus, let $\mathcal{O}$, A, and B be as stated. Define

$$\begin{aligned} \mathcal{O}' &= \mathcal{O} \cup \{B \sqsubseteq \exists r.(B_1 \sqcap B_2 \sqcap \exists r.\top)\} \\ q(x) &= \exists y_1 \exists y_2 \exists z\, A(x) \wedge r(x, y_1) \wedge r(x, y_2) \wedge B_1(y_1) \wedge \\ &\qquad B_2(y_2) \wedge r(y_1, z) \wedge r(y_2, z) \end{aligned}$$

where r is a fresh role name, and set $Q = (\mathcal{O}', \mathbf{S}_{\mathsf{full}}, q)$. Notice the similarity of this construction to the proof of Point 1 of Theorem 10.

Claim. $\mathcal{O} \models A \sqsubseteq B$ iff Q is (U)CQ$_1$-equivalent.

For the 'if' direction, it suffices to note that when $\mathcal{O} \not\models A \sqsubseteq B$ then Q is a full rewriting of itself. For the 'only if' direction, from $\mathcal{O} \models A \sqsubseteq B$ it follows that $Q = (\mathcal{O}', \mathbf{S}_{\mathsf{full}}, A(x))$ is a full rewriting of Q. The generalization to the case $k > 1$ is as in the proof of Point 1 of Theorem 10, details are omitted.

Let us focus on the upper bounds. We first argue that instead of proving the results for ($\mathcal{ELH}^{dr}_\bot$, UCQ), ($\mathcal{ELHI}_\bot$, UCQ), and (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$, UCQ), it suffices to establish them for the corresponding OMQ languages based on CQs. In fact, we can assume w.l.o.g. that, when an OMQ $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ is given as input and $q(\mathbf{x}) = \bigvee_{1 \leq i \leq n} p_i$, then the OMQs $Q_i = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p_i)$, $1 \leq i \leq n$, are pairwise incomparable regarding containment. The reason is that, when the schema is full, OMQ containment trivially reduces to OMQ evaluation, which means that containment in ($\mathcal{ELH}^{dr}_\bot$, CQ) and (DL-Lite$^\mathcal{R}_{\mathsf{horn}}$, CQ) is in NP, and in ExpTime in ($\mathcal{ELHI}_\bot$, CQ) [28], [32], [35]. We can thus remove a disjunct p_i from q if there is a p_j, $j < i$, such that $Q_j \subseteq Q_i$. We can also assume that $\mathcal{D}_{p_i}$ is consistent with $\mathcal{O}$, for $1 \leq i \leq n$, since otherwise we can remove the disjunct p_i and if all disjuncts are removed then Q is trivially UCQ$_k$-equivalent; moreover, checking consistency of $\mathcal{D}_{p_i}$ with $\mathcal{O}$ also reduces easily to OMQ evaluation.

Claim. Q is UCQ$_k$-equivalent iff every Q_i is.

Proof of claim. For the non-trivial 'only if' direction, assume that Q is UCQ$_k$-equivalent and let $Q' = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q')$ be an equivalent OMQ with q' from UCQ$_k$. Consider some Q_i. Clearly, $Q_i \subseteq Q'$ implies $\mathcal{D}_{p_i} \models Q'(\mathbf{x})$ where the answer variables $\mathbf{x}$ of p_i are viewed as constants. But then $\mathcal{D}_{p_i} \models Q''$ where $Q'' = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p')$ for some CQ p' in q', and thus $Q_i \subseteq Q''$. Since $Q'' \subseteq Q$, we can argue analogously that

$Q'' \subseteq Q_j$ for some j. Thus $Q_i \subseteq Q_j$ implies that $i = j$ since otherwise Q_i and Q_j would be comparable regarding containment. But then Q_i is equivalent to Q'' and thus CQ$_k$-equivalent. This finishes the proof of the claim.

It thus suffices to prove upper bounds for UCQ$_k$-equivalence in $(\mathcal{ELH}^{dr}_\bot, \text{CQ})$, $(\mathcal{ELHI}_\bot, \text{CQ})$, and (DL-Lite$^{\mathcal{R}}_{\mathsf{horn}}$, CQ). All these bounds are established in a uniform way. Assume that the OMQ $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ is given where q is a CQ. We first check whether Q is empty (using a containment check) and if it is then we return that Q is UCQ$_k$-equivalent. Otherwise, $\mathcal{D}_q$ must be consistent with $\mathcal{O}$. We then extend q to a CQ q' as follows, paralelling Step 2 in the construction of rewritings: for each $C \sqsubseteq D \in \mathcal{O}$ and $x \in C^{\mathcal{D}_q}$, add a fresh copy q_C of C viewed as a CQ and add q_C to q, identifying x with the root of q_C. We then guess a subquery q'' of q' of tree width at most k and check whether Q is equivalent to $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q'')$. Equivalence can be implemented as two containment checks; see above for the relevant complexities.

If we are able to guess correctly, then clearly Q is UCQ$_k$-equivalent. Conversely, if Q is UCQ$_k$-equivalent, then by Theorem 3 there is a full rewriting $Q' = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p)$ of Q with $p \in \text{CQ}_k$. It is easy to verify that p is a subquery of q'.

It can be verified that in all the considered cases, the above procedure yields the stated upper bounds. □

Proofs for Section VII

The result that we can immediately inherit from [23] is the following that talks about BCQs:

Theorem 18 (Figueira). *For OMQs from (DL-Lite$^{\mathcal{F}}_{=}$, BCQ) based on the full schema, BCQ$_k$-equivalence while preserving the ontology is in* 2ExpTime, *for any $k \geq 1$. Moreover, an equivalent OMQ from (DL-Lite$^{\mathcal{F}}_{=}$, BCQ$_k$) can be constructed in double exponential time (if it exists).*

We need to show that the above result can be stated for UBCQs (Theorem 13 in the main body of the paper). To this end, we establish the following technical result.

Lemma 21. *Consider an OMQ $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ from (DL-Lite$^{\mathcal{F}}$, UBCQ). The following are equivalent:*

1) Q is UBCQ$_k$-equivalent while preserving the ontology.
2) For each q' in q, (i) $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q')$ is BCQ$_k$-equivalent while preserving the ontology, or (ii) there exists q'' in q such that $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q') \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q'')$.

Proof. $(1) \Rightarrow (2)$. By hypothesis, there exists a UBCQ$_k$ $\hat{q}$ such that $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q) \equiv (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, \hat{q})$. Consider an arbitrary BCQ q' in q. Since $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q) \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, \hat{q})$, we get that there exists p in $\hat{q}$ such that $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q') \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p)$. But since $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, \hat{q}) \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$, we get that there exists p' in q such that $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p) \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p')$. Therefore,

$$(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q') \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p) \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p'),$$

where p belongs to $\hat{q}$ and p' belongs to q. We consider two cases: $q' = p'$, which implies that $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q') \equiv (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p)$, and condition (i) holds since p is from BCQ$_k$, and $q' \neq p'$ but since $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q') \subseteq (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, p')$ condition (ii) holds.

$(2) \Rightarrow (1)$. This direction is clear. □

It is easy to verify that Theorem 13 follows from Theorem 18 and Lemma 21.

Lemma 12. *Fix $k \geq 1$. For an OMQ $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q) \in$ (DL-Lite$^{\mathcal{F}}$, UBCQ) the following are equivalent:*

1) Q is UBCQ$_k$-equivalent while preserving the ontology;
2) $(\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q))$ is UBCQ$_k$-equivalent while preserving the ontology.

For showing the above lemma, we first need to establish the following technical lemma:

Lemma 22. *Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q) \in$ (DL-Lite$^{\mathcal{F}}$, UBCQ). Then there is a UBCQ $\mathsf{rew}(Q)$ such that for every database $\mathcal{D}$ that satisfies $\mathcal{O}^{=}$, $\mathcal{D} \models Q$ iff $\mathcal{D} \models \mathsf{rew}(Q)$. Furthermore, if $q \in$ UBCQ$_k$ for some $k \geq 1$, then $\mathsf{rew}(Q) \in$ UBCQ$_k$.*

Proof. We start with introducing some useful notions. Let q be a BCQ. A BCQ $p \subseteq q$ is a *tree in q with root x* if $x \in \mathsf{var}(q)$ is an articulation point that separates q into components q', p with p of the form $r(x, y) \wedge \varphi(\mathbf{y})$ where $\varphi(\mathbf{y})$ is a tree with root y. Let at be an atom of the form $A(x)$, $S(x, z)$, or $S(z, x)$ with z a fresh variable. For an ontology $\mathcal{O}$, we say that at *$\mathcal{O}$-generates* p if $\{\mathsf{at}\}, \mathcal{O} \models p(x)$ where the variables in at, including x, are viewed as constants in the database $\{\mathsf{at}\}$. Moreover, p is *$\mathcal{O}$-generatable* if there is an atom that $\mathcal{O}$-generates p. Further, at *detachedly $\mathcal{O}$-generates* a BCQ p if $\{\mathsf{at}\}, \mathcal{O} \models p$ and p is *detached $\mathcal{O}$-generatable* if there is an atom that detachedly $\mathcal{O}$-generates p.

Now let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q) \in$ (DL-Lite$^{\mathcal{F}}$, UBCQ) as in the lemma. We define $\mathsf{rew}(Q)$ to be the disjunction of all BCQs that can be obtained in the following way:

1) choose a BCQ p in q, a contraction p' of p, and a set S of trees in p' and are $\mathcal{O}$-generatable, and remove those trees from p';
2) if for the resulting BCQ p'', $G_{p''}$ is not a minor of G_p, then stop;
3) otherwise, for each $p_t \in S$ choose an atom at that $\mathcal{O}$-generates p_t and add at;
4) choose a set of maximal connected components of the resulting BCQ that are detachedly $\mathcal{O}$-generatable, for each such component choose an atom at that detachedly $\mathcal{O}$-generates it, and add at.

With this definition of $\mathsf{rew}(Q)$, the "Furthermore" part of the lemma is trivially satisfied since the class of structures of tree width k is minor closed [24] and the additional modifications in Steps 3 and 4 clearly cannot increase tree width. It thus remains to show the following.

Claim. For every database $\mathcal{D}$ that satisfies $\mathcal{O}^{=}$, $\mathcal{D} \models Q$ iff $\mathcal{D} \models \mathsf{rew}(Q)$.

The 'if' direction is easy to show using the construction of $\mathsf{rew}(Q)$, we omit details. For the 'only if' direction, let $\mathcal{D}$ be a database that satisfies $\mathcal{O}^{=}$ and such that $\mathcal{D} \models Q$. Then there is a CQ p in q and a homomorphism h from p to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$. We show how to use h to guide the choices in Steps 1 to 4 above so as to obtain a BCQ $\widehat{q}$ in $\mathsf{rew}(Q)$ such that h can be extended to a homomorphism from $\widehat{q}$ to $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$.

Let V_2 denote the set of pairs $(x_1, x_2) \in \mathsf{var}(p)^2$ such that $h(x_1) = h(x_2) \in \mathsf{dom}(\mathcal{D})$ and, additionally, p contains atoms

$$r_1(y_1, y_2), \dots, r_{n-1}(y_{n-1}, y_n) \qquad (\dagger)$$

such that $y_0 = x_1$, $y_n = x_1$, and $h(y_i) \notin \mathsf{dom}(\mathcal{D})$ for $1 < i < n$. In Step 1, we choose as p' the contraction of p that is obtained by identifying x_1 and x_2 whenever $(x_1, x_2) \in V_2$. We further choose as S be the set of all trees $p_t(x) = r(x, y) \wedge \varphi(\mathbf{y})$ in p' such that $h(x) \in \mathsf{dom}(\mathcal{D})$ and $h(y) \notin \mathsf{dom}(\mathcal{D})$. We must then have $h(z) \notin \mathsf{dom}(\mathcal{D})$ for all $z \in \mathbf{y}$ as otherwise x would have been identified with some variable in $\mathbf{y}$. By the construction of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$, there must thus be a fact F in $\mathcal{D}$ such that $\{F\}, \mathcal{D} \models p_t(h(x))$. As a consequence of this and the semantics of DL-Lite, we find an atom at of the form $A(x)$, $S(x, z)$, or $S(z, x)$, with z a fresh variable, such that at $\mathcal{O}$-generates p_t and h extends to a homomorphism from $\{\mathsf{at}\}$ to $\mathcal{D}$. By the former, p_t is $\mathcal{O}$-generatable.

Let p'' be the result of removing all trees in S from p'. We next argue that p'' is a minor of p and thus we do not stop in Step 2. We work with the definition of minors from [24], that is, an undirected graph G_1 is a minor of an undirected graph G_2 if G_1 can be obtained from a (not necessarily induced) subgraph of G_2 by contracting edges. We start with the subquery p^* of p that is the result of

1) taking the restriction of p to all variables y such that $h(y) \in \mathsf{dom}(\mathcal{D})$ or y is in a maximal connected component of p all of whose variables are mapped by h to outside of $\mathsf{dom}(\mathcal{D})$ and then
2) adding back a path of the form (†) for all $(x_1, x_2) \in V_2$.

Clearly, G_{p^*} is a subgraph of G_p. We can obtain p'' from p^* by contracting all edges in G_{p^*} that are induced by the paths that have been added back. Thus p'' is a minor of p.

The choices in Steps 3 and 4 can also be guided by h. We have already argued that all $p_t \in S$ are $\mathcal{O}$-generatable, and that we can find replacing atoms at that are 'compatible with h', which we choose in Step 3. For Step 4, we choose those maximal connected components all of whose variables h maps to outside of $\mathsf{dom}(\mathcal{D})$. By construction of $\mathsf{ch}_{\mathcal{O}}(\mathcal{D})$, for every such component there must be an atom at of one of the three relevant forms that detachedly $\mathcal{O}$-generates it and such that h extends to a homomorphism from $\{\mathsf{at}\}$ to $\mathcal{D}$.

Let $\widehat{p}$ be the CQ generated by guiding Steps 1 to 4 with h as described above. By construction, we clearly find an extension h' of h to the fresh variables in $\widehat{p}$ such that h' is a homomorphism from $\widehat{p}$ to $\mathcal{D}$. Thus, $\mathcal{D} \models \mathsf{rew}(Q)$. □

We can now give the proof of Lemma 12.

Proof. $\underline{(1) \Rightarrow (2).}$ By hypothesis, there exists an OMQ $Q' = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q')$ from (DL-Lite$^{\mathcal{F}}$, UBCQ$_k$) such that $Q \equiv Q'$. By Lemma 22, we can conclude that

$$(\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q)) \;\equiv\; (\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q')).$$

By Lemma 22, $\mathsf{rew}(Q') \in \text{UBCQ}_k$, and (2) follows.

$\underline{(2) \Rightarrow (1).}$ By hypothesis, there is $q' \in \text{UBCQ}_k$ such that

$$(\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q)) \;\equiv\; (\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, q').$$

By Lemma 22, $Q \equiv (\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q))$. It is not difficult to show that $(\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q)) \equiv (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q))$, which implies that $Q \equiv (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q))$. It is also easy to verify that $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, \mathsf{rew}(Q)) \equiv (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q')$. Therefore, $Q \equiv (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q')$, and (1) follows since $q' \in \text{UBCQ}_k$. □

Theorem 15.

1) *In (DL-Lite$^{\mathcal{F}}$, UBCQ), UBCQ$_1$-equivalence coincides with UBCQ$_1$-equivalence while preserving the ontology.*
2) $\textsc{Evaluation}((\textit{DL-Lite}^{\mathcal{F}}, \textit{UBCQ})^{\equiv}_{\textit{UBCQ}_1})$ *based on the full schema is in* PTime.
3) *For OMQs from (DL-Lite$^{\mathcal{F}}$, UBCQ) based on the full schema, UBCQ$_1$-equivalence is* NP*-complete.*

Before giving the proof of the above result, we first present the reduction of UBCQ$_1$-equivalence in (DL-Lite$^{\mathcal{F}}$, UBCQ) to UBCQ$_1$-equivalence in (DL-Lite, UBCQ) announced in the paper, where DL-Lite denotes the DL obtained from DL-Lite$^{\mathcal{F}}$ after dropping the functionality assertions.

For a UBCQ q, we use $\mathsf{id}_F(q)$ to denote the result of contracting each BCQ in q in a minimal way such that the functionality assertions in F are respected. It is important to observe that if q is of tree width 1, then so is $\mathsf{id}_F(q)$. This is not the case for any higher tree width.

Lemma 23. *Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ be an OMQ from (DL-Lite$^{\mathcal{F}}$, UBCQ). On databases that are consistent with $\mathcal{O}^{=}$, Q is equivalent to any of the following: $(\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q)$, $(\mathcal{O}, \mathbf{S}_{\mathsf{full}}, \mathsf{id}_{\mathcal{O}^{=}}(q))$, $(\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, \mathsf{id}_{\mathcal{O}^{=}}(q))$.*

Lemma 24. *Let $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ be an OMQ from (DL-Lite$^{\mathcal{F}}$, UBCQ). There exists a UBCQ q' such that*

1) *Q is equivalent to $(\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, q')$;*
2) *$(\emptyset, \mathbf{S}_{\mathsf{full}}, q')$ is equivalent to $(\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q')$;*
3) *If $q \in$ UBCQ$_k$ for some $k \geq 1$, then $q' \in$ UBCQ$_k$.*

Proof. Let $Q^{\sqsubseteq} = (\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q)$ and assume that $q' = \mathsf{rew}(Q^{\sqsubseteq})$ is the UBCQ provided by Lemma 22.

Then Point 1 is satisfied. Indeed, if $\mathcal{D}$ is inconsistent with $\mathcal{O}^{=}$, then $\mathcal{D} \models Q$ and $\mathcal{D} \models (\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, q')$. Otherwise, $\mathcal{D} \models Q$ iff $\mathcal{D} \models Q^{\sqsubseteq}$ (by Lemma 23) iff $\mathcal{D} \models q'$ (by Lemma 22).

For Point 2, it suffices to show that $(\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q') \subseteq (\emptyset, \mathbf{S}_{\mathsf{full}}, q')$. Assume that $\mathcal{D} \models (\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q')$. Then there is a homomorphism h from q' to $\mathsf{ch}_{\mathcal{O}^{\sqsubseteq}}(\mathcal{D})$. Let $\mathcal{D}'$ be the restriction of $\mathcal{D}$ to the constants in the range of h. By construction, $\mathcal{D}' \models q'$. Thus $\mathcal{D}' \models Q^{\sqsubseteq}$, that is, there is a homomorphism g from q to $\mathsf{ch}_{\mathcal{O}^{\sqsubseteq}}(\mathcal{D}')$. Since $\mathcal{D}'$ is a subset of $\mathsf{ch}_{\mathcal{O}^{\sqsubseteq}}(\mathcal{D})$, there is also a homomorphism from q to $\mathsf{ch}_{\mathcal{O}^{\sqsubseteq}}(\mathcal{D})$. We obtain that $\mathcal{D} \models Q^{\sqsubseteq}$, and thus $\mathcal{D} \models (\emptyset, \mathbf{S}_{\mathsf{full}}, q')$.

Point 3 follows from Lemma 22. □

Now for the reduction. Let $Q_1 = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_1)$ be an OMQ from (DL-Lite$^{\mathcal{F}}$, UBCQ), and let $Q_2 = (\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q_2)$, where $q_2 = \mathsf{id}_{\mathcal{O}^{=}}(q)$. We can show the following.

Lemma 25. *Q_1 is UBCQ$_1$-equivalent iff Q_2 is UBCQ$_1$-equivalent. Moreover, if Q_2 is equivalent to $Q_2' = (\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q_2')$ with $q_2' \in$ UBCQ$_1$, then $Q_1 \equiv (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_2')$.*

Proof. The 'if' direction is implied by the "Moreover" part. To prove the "Moreover" part, assume that Q_2 is equivalent to $Q_2' = (\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q_2')$ with $q_2' \in \text{UBCQ}_1$. We have to show that Q_1 and $Q_1' = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q_2')$ give the same answers on all databases $\mathcal{D}$. If $\mathcal{D}$ is inconsistent with $\mathcal{O}$, then $\mathcal{D} \models Q_1$ and $\mathcal{D} \models Q_1'$. If $\mathcal{D}$ is consistent with $\mathcal{O}^{=}$, then $\mathcal{D} \models Q_1$ iff $\mathcal{D} \models Q_2$ by Lemma 23. Moreover, $\mathcal{D} \models Q_2$ iff $\mathcal{D} \models Q_2'$ iff $\mathcal{D} \models Q_1'$, the latter by Lemma 23.

For the 'only if' direction, assume that Q_1 is equivalent to an OMQ Q_1' from $(\text{DL-Lite}^{\mathcal{F}}, \text{UBCQ}_1)$. By Lemma 24, we can assume Q_1' to be of the form $(\mathcal{O}^{=}, \mathbf{S}_{\mathsf{full}}, q_1')$ with all BCQs in q_1' of tree width 1. We can assume w.l.o.g. that

(∗) $\mathcal{D}_p$ satisfies all functionality assertions in $\mathcal{O}^{=}$, for each BCQ p in q_1.

In fact, we can achieve (∗) by replacing each BCQ p in q_1' with $\mathsf{id}_{\mathcal{O}^{=}}(p)$ which preserves tree width 1 and, by Lemma 23, is also equivalence preserving regarding Q_1'.

We now show that $Q_2' = (\emptyset, \mathbf{S}_{\mathsf{full}}, q_1')$ is equivalent to Q_2.

'$\subseteq$'. Let $\mathcal{D}$ be a database with $\mathcal{D} \models Q_2'$. Then there is a homomorphism from some BCQ p in q_1' to $\mathcal{D}$. Clearly $\mathcal{D}_p \models Q_2'$ and by construction of Q_2', this implies $\mathcal{D}_p \models Q_1'$ and thus also $\mathcal{D}_p \models Q_1$. By (∗), $\mathcal{D}_p$ satisfies all the functionality assertions in $\mathcal{O}^{=}$. By Lemma 23, we thus have $\mathcal{D}_p \models Q_2$. Since there is a homomorphism from $\mathcal{D}_p$ to $\mathcal{D}$, this yields $\mathcal{D} \models Q_2$ as required.

'$\supseteq$'. Let $\mathcal{D}$ be a database with $\mathcal{D} \models Q_2$. Then there is a homomorphism from some BCQ p in q_2 to $\mathsf{ch}_{\mathcal{O}^{\sqsubseteq}}(\mathcal{D})$. Clearly, we have $\mathcal{D}_p \models Q_2$. By construction of q_2, $\mathcal{D}_p$ satisfies all the functionality assertions in $\mathcal{O}^{=}$. Lemma 23 thus yields $\mathcal{D}_p \models Q_1$ and, consequently, $\mathcal{D}_p \models Q_1'$. Since $\mathcal{D}_p$ satisfies all the functionality assertions in $\mathcal{O}^{=}$, this means $\mathcal{D}_p \models Q_2'$. The homomorphism from p to $\mathsf{ch}_{\mathcal{O}^{\sqsubseteq}}(\mathcal{D})$ yields $\mathcal{D} \models (\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q_1')$. By Point 2 of Lemma 24, this implies $\mathcal{D} \models Q_2'$, as required. □

We can now give the proof of Theorem 15.

Proof. Point 1 follows from the "Moreover" part of Lemma 25 and Corollary 1.

To prove Point 2, assume that an OMQ $Q = (\mathcal{O}, \mathbf{S}_{\mathsf{full}}, q)$ from $(\text{DL-Lite}^{\mathcal{F}}, \text{UBCQ})^{\equiv}_{\text{UBCQ}_1}$ is given as an input, along with a database $\mathcal{D}$. We can check in PTIME whether $\mathcal{D}$ is consistent with $\mathcal{O}^{=}$. If it is not, then we answer 'true'. Otherwise, by Lemma 23 it suffices to evaluate $(\mathcal{O}^{\sqsubseteq}, \mathbf{S}_{\mathsf{full}}, q)$ in place of Q. This OMQ, however, is from $(\text{DL-Lite}, \text{UBCQ})^{\equiv}_{\text{UBCQ}_1}$ and thus we can invoke Theorem 8.

The upper bound in Point 3 is an immediate consequence of Lemma 25 and Point 3 of Theorem 12. The lower bound is inherited from the case where the ontology is empty. □