Exact Learning of Lightweight Description Logic Ontologies

Boris Konev
University of Liverpool
United Kingdom

Carsten Lutz
Universität Bremen
Germany

Ana Ozaki
University of Liverpool
United Kingdom

Frank Wolter
University of Liverpool
United Kingdom

Abstract

We study learning of description logic TBoxes in Angluin et al.'s framework of exact learning via queries. We admit entailment queries ("is a given subsumption entailed by the target TBox?") and equivalence queries ("is a given TBox equivalent to the target TBox?"), assuming that the signature and logic of the target TBox are known. We present three main results: (1) TBoxes formulated in DL-Lite with role inclusions and $\mathcal{ELI}$ concepts on the right-hand side of concept inclusions can be learned in polynomial time; (2) $\mathcal{EL}$ TBoxes with only concept names on the right-hand side of concept inclusions can be learned in polynomial time; and (3) $\mathcal{EL}$ TBoxes cannot be learned in polynomial time. It follows that non-polynomial time learnability of $\mathcal{EL}$ TBoxes is caused by the interaction between existential restrictions on the right- and left-hand sides of concept inclusions. We also show that neither entailment nor equivalence queries alone are sufficient in cases (1) and (2) above.

Successfully deploying a description logic (DL) in a concrete application requires to carefully capture the relevant domain knowledge in a DL ontology. This is a subtle, error-prone, and time consuming task, which is further hindered by the fact that domain experts are rarely experts in ontology engineering and, conversely, ontology engineers are often not sufficiently familiar with the domain to be modeled. From its beginnings, DL research was driven by the aim to provide various forms of support for ontology engineers, assisting them in the design of high-quality ontologies. Examples include the ubiquitous task of ontology classification (Baader et al. 2003), bootstrapping ontology design from examples, data, or text (Baader and Molitor 2000; Borchmann and Distel 2011; Ma and Distel 2013) and checking the completeness of the modeling in a systematic way (Baader et al. 2007).

Machine learning (ML) techniques are natural candidates to support ontology engineering by partly automatizing the design process, and in fact ML approaches have been proposed for ontology engineering in non-DL-contexts. There, the main focus is on mining the relevant terms of the application domain from bodies of text and semi-structured data with the aim of including them in the ontology. Some approaches also support learning of subsumptions between the identified terms, but not between compound logical expressions based on these terms (Cimiano, Hotho, and Staab 2005; Buitelaar, Cimiano, and Magnini 2005). In the context of DL ontologies, though, the main objective is to develop a detailed modeling of the relevent concepts of the domain using the logical operators provided by the DL at hand (Voelker 2009). In this paper, we thus concentrate on learning the full logical structure of a DL TBox. In particular, we study the foundations of learning ontologies that are formulated in several DLs which underly the three tractable *profiles* of the OWL2 ontology language.

We adopt Angluin et al.'s framework of exact learning via queries (Angluin 1987c). In particular, we are interested in learning a target TBox $\mathcal{T}$ that is formulated in a given DL and whose signature Σ (the concept and role names that occur in $\mathcal{T}$) is also known, by posing queries to an oracle. Intuitively, the oracle can be thought of as a domain expert who interacts with the learning algorithm. We consider two forms of oracle queries:

- *entailment queries*: does $\mathcal{T}$ entail a given Σ concept inclusion $C \sqsubseteq D$? These queries are answered by the oracle with 'yes' or 'no'.
- *equivalence queries*: is a given Σ-TBox $\mathcal{H}$ (the hypothesis) equivalent to $\mathcal{T}$? The oracle answers 'yes' if $\mathcal{H}$ and $\mathcal{T}$ are logically equivalent and 'no' otherwise. It then returns a Σ-inclusion $C \sqsubseteq D$ such that $\mathcal{T} \models C \sqsubseteq D$ and $\mathcal{H} \not\models C \sqsubseteq D$ (a *positive counterexample*), or vice versa (a *negative counterexample*).

We generally assume that the inclusions used in entailment queries and returned by equivalence queries are of the same form as the inclusions allowed in the target TBox. While, in general, answering an equivalence query of the above form can be expected to be rather challenging also for a domain expert and an ontology engineer, we emphasize that the actual algorithms developed in this paper use equivalence queries only in a very restricted way. In particular, our algorithms always ensure that the hypothesis TBox $\mathcal{H}$ that they maintain is a logical consequence of the target TBox $\mathcal{T}$, and then use equivalence queries only to ask 'is the TBox $\mathcal{H}$ learned so far complete and if not, please give me a missing inclusion $C \sqsubseteq D$'.

Following Angluin, our aim is to find (deterministic) learning algorithms that run in polynomial time and consequently also make only polynomially many oracle queries. More precisely, we require that there is a two-variable polynomial $p(n, m)$ such that at every point of the algorithm execution, the time spent up to that point is bounded by $p(n, m)$, where n is the size of the target TBox $\mathcal{T}$ to be learned and m is the size of the largest (positive or negative) counterexample that was returned by the oracle until the considered point of the execution. If such an algorithm exists, then $\mathcal{L}$ TBoxes are *polynomial time learnable*.

Within the setup laid out above, we study three different DLs, which are significant fragments of the three profiles of OWL2 (Motik et al. 2009). The popular description logic $\mathcal{EL}$ can be viewed as a logical core of the OWL2 EL profile of OWL2. It is known that propositional Horn formulas, which correspond to $\mathcal{EL}$ TBoxes without existential restrictions, can be learned in polynomial time when both entailment and equivalence queries are available, but not when one of these types of queries is disallowed (Frazier and Pitt 1993; Angluin, Frazier, and Pitt 1992; Angluin 1987a), see also (Arias and Balcázar 2011). We show that, in contrast, $\mathcal{EL}$ TBoxes are not polynomial time learnable. The results for the other two DLs that we study will shed some light on the reasons for this behaviour.

The DL-Lite family of DLs underlies the OWL2 QL profile of OWL2 (Calvanese et al. 2007). In many versions of DL-Lite such as in DL-Lite$_\mathcal{R}$ (DL-Lite with role inclusions), the number of Σ-concept inclusions is polynomial in the size of Σ. In this case, TBoxes are trivially learnable in polynomial time, even when only entailment queries (but no equivalence queries) are available or vice versa. We thus consider the more interesting DL-Lite$_\mathcal{R}^\exists$ dialect, that is, the extension of DL-Lite$_\mathcal{R}$ that allows any $\mathcal{ELI}$ concept on the right-hand side of concept inclusions. Note that DL-Lite$_\mathcal{R}^\exists$ can be viewed as a logical core of OWL2 QL. Here, the number of Σ-concept inclusions is infinite, but we are nevertheless able to establish that DL-Lite$_\mathcal{R}^\exists$ TBoxes are polynomial time learnable. In practice, many $\mathcal{EL}$ TBoxes actually fall within (the intersection of $\mathcal{EL}$ and) DL-Lite$_\mathcal{R}^\exists$. We also show that DL-Lite$_\mathcal{R}^\exists$ TBoxes cannot be learned in polynomial time with entailment queries alone or with equivalence queries alone.

We then consider the fragment $\mathcal{EL}_{\mathsf{lhs}}$ of $\mathcal{EL}$ where only concept names (but no compound concepts) are admitted on the right-hand side of concept inclusions. $\mathcal{EL}_{\mathsf{lhs}}$ is a significant fragment of the OWL2 RL profile of OWL2, and also a fragment of datalog. We prove that $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes can be learned in polynomial time using a non-trivial extension of Angluin's polynomial time agorithm for learning propositional Horn theories (1992). Note that the symmetric fragment $\mathcal{EL}_{\mathsf{rhs}}$ that allows only concept names on the *right*-hand side of concept inclusions is a fragment of DL-Lite$_\mathcal{R}^\exists$. Together, our results for DL-Lite$_\mathcal{R}^\exists$ and $\mathcal{EL}_{\mathsf{lhs}}$ thus show that non-polynomial time learnability of $\mathcal{EL}$ TBoxes is caused by the interaction between existential restrictions on the right- and left-hand sides of concept inclusions.

To improve readability, we first present the upper bounds (polynomial time learnability for DL-Lite$_\mathcal{R}^\exists$ and $\mathcal{EL}_{\mathsf{lhs}}$) and then the lower bounds. Proof details are provided in the appendix of this paper.

Related Work. In the terminology of the learning literature, entailment queries are a kind of membership queries, which can take many different forms (Raedt 1997). Learning using membership and equivalence queries appears to be the most successful protocol for exact learning in Angluin's framework. Apart from propositional Horn formulas, polynomial time learnable classes for this protocol include regular sets (Angluin 1987b) and monotone DNF (Angluin 1987c). Exploring the possibilities of extending the learning algorithm for propositional Horn formulas to (fragments) of first-order Horn logic has been a major topic in exact learning (Reddy and Tadepalli 1999; Arias and Khardon 2002; Arias, Khardon, and Maloberti 2007; Selman and Fern 2011). Note that $\mathcal{EL}_{\mathsf{lhs}}$ can be regarded as a fragment of first-order (FO) Horn logic. Existing approaches to polynomial time learning of FO Horn formulas either disallow recursion, bound the number of individual variables per Horn clause, or admit additional queries in the learning protocol. None of this is the case in our setup. Also related to our work is exact learning of schema mappings in data exchange, as recently studied in (ten Cate, Dalmau, and Kolaitis 2012). In this and some other studies mentioned above, membership queries take the form of interpretations ("is a given interpretation a model of the target theory?"). In DL, exact learning has been studied for CLASSIC in (Frazier and Pitt 1996) where it is shown that single CLASSIC concepts (but not TBoxes) can be learned in polynomial time (here membership queries are concepts). Learning single concepts using refinement operators has been studied in (Lehmann and Hitzler 2010).

Preliminaries

Let $\mathsf{N_C}$ be a countably infinite set of *concept names* and $\mathsf{N_R}$ a countably infinite set of *role names*. The dialect DL-Lite$_\mathcal{R}^\exists$ of DL-Lite is defined as follows (Calvanese et al. 2007). A *role* is a role name or an inverse role r^- with $r \in \mathsf{N_R}$. A *role inclusion (RI)* is of the form $r \sqsubseteq s$, where r and s are roles. A *basic concept* is either a concept name or of the form $\exists r.\top$, with r a role. A DL-Lite$_\mathcal{R}^\exists$ concept inclusion (CI) is of the form $B \sqsubseteq C$, where B is a basic concept and C is an $\mathcal{ELI}$ concept, that is, C is formed according to the rule

$$C, D \quad := \quad A \quad | \quad \top \quad | \quad C \sqcap D \quad | \quad \exists r.C \quad | \quad \exists r^-.C$$

where A ranges over $\mathsf{N_C}$ and r ranges over $\mathsf{N_R}$. A DL-Lite$_\mathcal{R}^\exists$ TBox is a finite set of DL-Lite$_\mathcal{R}^\exists$ CIs and RIs.[1]

As usual, an *$\mathcal{EL}$ concept* is an $\mathcal{ELI}$ concept that does not use inverse roles, an *$\mathcal{EL}$ concept inclusion* has the form $C \sqsubseteq D$ with C and D $\mathcal{EL}$ concepts, and a *(general) $\mathcal{EL}$ TBox* is a finite set of $\mathcal{EL}$ concept inclusions (Baader, Brandt, and Lutz 2005). We use $C \equiv D$ as an abbreviation for the inclusions $C \sqsubseteq D$ and $D \sqsubseteq C$. An $\mathcal{EL}$ TBox $\mathcal{T}$ is *acyclic* if it consists of inclusions $A \sqsubseteq C$ and

[1]For simplicity, we consider DL-Lite without disjointness axioms. All our results also hold for the extension of DL-Lite$_\mathcal{R}^\exists$ with disjointness.

$A \equiv C$ such that A is a concept name, no concept names occurs more than once on the left hand side of an inclusion in $\mathcal{T}$, and $\mathcal{T}$ contains no cycles (Baader et al. 2003; Konev et al. 2012). The semantics of concepts and TBoxes is defined as usual in terms of interpretations (Baader et al. 2003). For a TBox $\mathcal{T}$ and concept inclusion $C \sqsubseteq D$, we write $\mathcal{T} \models C \sqsubseteq D$ if every model $\mathcal{I}$ of $\mathcal{T}$ also satisfies $C \sqsubseteq D$. $\mathcal{T}$ is omitted if it is empty, that is, we then simply write $\models C \sqsubseteq D$. We say that concepts C and D are *equivalent w.r.t.* $\mathcal{T}$ and write $C \equiv_{\mathcal{T}} D$ if both $\mathcal{T} \models C \sqsubseteq D$ and $\mathcal{T} \models D \sqsubseteq C$; if r, s are roles, $r \equiv_{\mathcal{T}} s$ is defined analogously. A *signature* Σ is a finite set of concept and role names. The *size* of a concept C, denoted with $|C|$, is the length of the string that represents it, where concept names and role names are considered to be of length one. The *size* of a TBox $\mathcal{T}$, denoted with $|\mathcal{T}|$, is $\sum_{C \sqsubseteq D \in \mathcal{T}} |C| + |D|$.

Learning DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes in Polynomial Time

We prove that DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes can be learned in polynomial time. The signature Σ of the target TBox $\mathcal{T}$ is known to the learner. To simplify presentation, we assume that the target TBox is in *named form*, that is, it contains for each role r a concept name A_r such that $A_r \equiv \exists r.\top \in \mathcal{T}$. This assumption is without loss of generality since any (polynomial time) learning algorithm for TBoxes in named form can be transformed into one for unrestricted TBoxes: the algorithm still uses the concept names A_r in its internal representations (although they are no longer included in the target signature Σ), and replaces each A_r with $\exists r.\top$ in queries to the oracle and when ultimately returning the TBox that it has learned.

In an initial phase, the learning algorithm for DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes determines, using at most $\mathcal{O}(|\Sigma|^2)$ entailment queries, all CIs $B_1 \sqsubseteq B_2$ with $\mathcal{T} \models B_1 \sqsubseteq B_2$ and B_1, B_2 basic concepts and the set of all RIs $r \sqsubseteq s$ with $\mathcal{T} \models r \sqsubseteq s$. A concept or role inclusion is in *reduced form* if all basic concepts that occur in it are concept names. We assume without further notice that all concept inclusions considered by the learner, except those that have been determined in the initial phase, are in reduced form. In particular, counterexamples returned by the oracle are immediately converted into this form.

To formulate the learning algorithm, it is useful to identify each $\mathcal{ELI}$ concept C with a finite tree T_C whose nodes are labeled with sets of concept names and whose edges are labeled with roles. In detail, if A is a concept name, then T_A has a single node d with label $l(d) = \{A\}$; if $C = \exists r.D$, then T_C is obtained from T_D by adding a new root d_0 and an edge from d_0 to the root d of T_D with label $l(d_0, d) = r$ (we then call d an *r-successor* of d_0); if $C = D_1 \sqcap D_2$, then T_C is obtained by identifying the roots of T_{D_1} and T_{D_2}. Conversely, every tree T of the described form gives rise to an $\mathcal{ELI}$ concept C_T in the obvious way. We will not always distinguish explicitly between C and its tree representation T_C which allows us to speak, for example, about the nodes and subtrees of an $\mathcal{ELI}$ concept.

The following notion plays a central role in the learning algorithm. Let $\mathcal{T}$ be a DL-Lite$^{\exists}_{\mathcal{R}}$ TBox. A DL-Lite$^{\exists}_{\mathcal{R}}$ inclusion $A \sqsubseteq C$ is *$\mathcal{T}$-essential* if it is in reduced form, $\mathcal{T} \models A \sqsubseteq C$, and the following conditions are satisfied:

1. $A \sqsubseteq C$ is *concept saturated for* $\mathcal{T}$: if C' results from C by adding a concept name A' to the label of some node, then $\mathcal{T} \not\models A \sqsubseteq C'$.
2. $A \sqsubseteq C$ is *role saturated for* $\mathcal{T}$: if C' results from C by replacing a role r by a role r' with $r' \not\equiv_{\mathcal{T}} r$ and $\mathcal{T} \models r' \sqsubseteq r$, then $\mathcal{T} \not\models A \sqsubseteq C'$.
3. $A \sqsubseteq C$ is *sibling-merged for* $\mathcal{T}$: if C' is the result of identifying in C an r-successor and an s-successor of the same node where $r \equiv_{\mathcal{T}} s$, then $\mathcal{T} \not\models A \sqsubseteq C'$.[2]
4. $A \sqsubseteq C$ is *parent/child-merged for* $\mathcal{T}$: if d' is an r-successor of d in C and d'' is an s-successor of d' with $r^- \equiv_{\mathcal{T}} s$, then $\mathcal{T} \not\models A \sqsubseteq C'$ where C' results from C by identifying d and d''.
5. $A \sqsubseteq C$ is *minimal for* $\mathcal{T}$: if d' is an r-successor of d in C and A' is in the node label of d, then $\mathcal{T} \not\models A' \sqsubseteq \exists r.C'$ where C' corresponds to the subtree rooted at d' (where $A' \not\equiv_{\mathcal{T}} A$ if d is the root of C).

Note that Points 1 to 5 carefully mix conditions that 'maximize' (Points 1 and 2) and 'minimize' (Point 5) the concept C. Points 3 and 4 are a form of maximization.

Algorithm 1 The learning algorithm for DL-Lite$^{\exists}_{\mathcal{R}}$

```
 1: Compute                                        (entailment queries)
        $\mathcal{H}_{basic} = \{r \sqsubseteq s \mid \mathcal{T} \models r \sqsubseteq s\} \cup$
            $\{B_1 \sqsubseteq B_2 \mid \mathcal{T} \models B_1 \sqsubseteq B_2,\ B_1, B_2$ basic$\}$
 2: Set $\mathcal{H}_{add} = \emptyset$
 3: while Equivalent $(\mathcal{H}_{basic} \cup \mathcal{H}_{add})$? returns "no" do
 4:     Let $A \sqsubseteq C$ be the returned positive counterexample
            for $\mathcal{T}$ relative to $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$
 5:     Find a $\mathcal{T}$-essential inclusion $A' \sqsubseteq C'$ with
            $\mathcal{H}_{basic} \cup \mathcal{H}_{add} \not\models A' \sqsubseteq C'$   (entailment queries)
 6:     if there is $A' \sqsubseteq C'' \in \mathcal{H}_{add}$ then
 7:         Find $\mathcal{T}$-essential inclusion $A' \sqsubseteq C^*$ such that
                $\models C^* \sqsubseteq C \sqcap C'$          (entailment queries)
 8:         Replace $A' \sqsubseteq C''$ by $A' \sqsubseteq C^*$ in $\mathcal{H}_{add}$
 9:     else
10:         add $A' \sqsubseteq C'$ to $\mathcal{H}_{add}$
11:     end if
12: end while
13: Set $\mathcal{H} = \mathcal{H}_{basic} \cup \mathcal{H}_{add}$.
```

The algorithm for learning DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes in named form is given as Algorithm 1, where the implementation of the lines marked with "(entailment queries)" requires the execution of entailment queries as detailed below. Observe that, in Line 4, the assumption that a positive counterexample is returned by the oracle (i.e., CI entailed by $\mathcal{T}$ but not by $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$) is justified by the construction of $\mathcal{H}_{basic}$ and $\mathcal{H}_{add}$, that is, we only include inclusions entailed by the

[2]The node that results from the identification is then either an r-successor or an s-successor of its parent, which is equivalent.

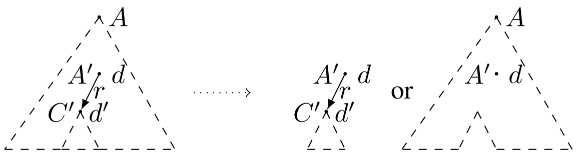

Figure 1: Minimization.

target TBox $\mathcal{T}$ in our hypothesis TBox $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$ and thus, at all times, $\mathcal{T} \models \mathcal{H}_{basic} \cup \mathcal{H}_{add}$. We now show in more detail how Lines 5 and 7 can be implemented, and that only polynomially many entailment queries are required. The next lemma addresses Line 5.

Lemma 1 *Given a positive counterexample $A \sqsubseteq C$ for $\mathcal{T}$ relative to $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$, one can construct a $\mathcal{T}$-essential such counterexample using only polynomially many entailment queries in $|C| + |\mathcal{T}|$.*

Proof. Let $A \sqsubseteq C$ be a positive counterexample for $\mathcal{T}$ relative to $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$. We may assume that $A \sqsubseteq C$ is in reduced form and exhaustively apply the following rules, which rely on posing entailment queries to the oracle:

(Concept saturation) if $\mathcal{T} \models A \sqsubseteq C'$ and C' results from C by adding a concept name A' to the label of some node, then replace $A \sqsubseteq C$ by $A \sqsubseteq C'$.

(Role saturation) if $\mathcal{T} \models A \sqsubseteq C'$ and C' results from C by replacing a role r by a r' with $r \not\equiv_{\mathcal{T}} r'$ and $\mathcal{T} \models r' \sqsubseteq r$, then replace $A \sqsubseteq C$ by $A \sqsubseteq C'$.

(Sibling merging) if $\mathcal{T} \models A \sqsubseteq C'$ and C' is the result of identifying in C an r-successor and an s-successor of the same node where $r \equiv_{\mathcal{T}} s$, then replace $A \sqsubseteq C$ by $A \sqsubseteq C'$.

(Parent/child merging) if d' is an r-successor of d in C, d'' is an s-successor of d' with $r^- \equiv_{\mathcal{T}} s$, and $\mathcal{T} \models A \sqsubseteq C'$ where C' results from C by identifying d and d'', then replace $A \sqsubseteq C$ by $A \sqsubseteq C'$.

(Minimization) if d' is an r-successor of d, A' is in the node label of d, and C' corresponds to the subtree rooted at d', $\mathcal{T} \models A' \sqsubseteq \exists r.C'$ (plus $A' \not\equiv_{\mathcal{T}} A$ if d is the root of C), then replace $A \sqsubseteq C$ by

(a) $A' \sqsubseteq \exists r.C'$ if $\mathcal{H}_{basic} \cup \mathcal{H}_{add} \not\models A' \sqsubseteq \exists r.C'$;

(b) $A \sqsubseteq C|^-_{d'\downarrow}$, where $C|^-_{d'\downarrow}$ is obtained from C by removing the subtree generated by d' from C, otherwise.

The last rule is illustrated in Figure 1. It follows directly from the definition of the rules that the final concept inclusion is $\mathcal{T}$-essential. Let us verify that it is also a positive counterexample for $\mathcal{T}$ relative to $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$. It suffices to show that the CI resulting from each rule application is entailed by $\mathcal{T}$, but not by $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$. The former is straightforward for all five rules. Regarding the latter, in the first four rules we have $\mathcal{H}_{basic} \models C' \sqsubseteq C$ if $A \sqsubseteq C$ is replaced by $A \sqsubseteq C'$. Hence $\mathcal{H}_{basic} \cup \mathcal{H}_{add} \not\models A \sqsubseteq C'$. In the last rule, we have $\{A \sqsubseteq C|^-_{d'\downarrow}, A' \sqsubseteq \exists r.C'\} \models A \sqsubseteq C$. Thus, one of the inclusions $A \sqsubseteq C|^-_{d'\downarrow}$ and $A' \sqsubseteq \exists r.C'$ is not entailed by $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$, and this is the CI resulting from the rule application.

It is not hard to see that the number of rule applications is bounded by $|C|^2 \cdot |\Sigma|$. Note that termination of the algorithm requires the condition in the last rule that $A' \not\equiv_{\mathcal{T}} A$ if d is the root of C. Otherwise, this rule could 'replace' $A \sqsubseteq C$ by any $A' \sqsubseteq C$ with A equivalent to A' w.r.t. $\mathcal{T}$. With the mentioned condition and since all CIs are in reduced form, each time when d is the root of C and $A \sqsubseteq C$ is replaced by $A' \sqsubseteq \exists r.C'$, then we strictly descend in subsumption order, that is, $\mathcal{T} \models A \sqsubseteq A'$ and $\mathcal{T} \not\models A' \sqsubseteq A$. ❏

The following lemma addresses Line 7.

Lemma 2 *Assume that $A \sqsubseteq C_1$ and $A \sqsubseteq C_2$ are $\mathcal{T}$-essential. Then one can construct a $\mathcal{T}$-essential $A \sqsubseteq C$ such that $\models C \sqsubseteq C_1 \sqcap C_2$ using polynomially many entailment queries in $|C_1| + |C_2|$.*

Proof. Starting with $A \sqsubseteq C_1 \sqcap C_2$, we exhaustively apply the rule for sibling merging from above and denote the result by $A \sqsubseteq C$. Observe that since $A \sqsubseteq C_1$ and $A \sqsubseteq C_2$ are both $\mathcal{T}$-essential, the concept inclusion $A \sqsubseteq C_1 \sqcap C_2$ is (i) concept-saturated for $\mathcal{T}$, (ii) role saturated for $\mathcal{T}$, (iii) parent/child-merged for $\mathcal{T}$, and (iv) minimal for $\mathcal{T}$. The only property of $\mathcal{T}$-essential inclusions that can fail for $A \sqsubseteq C_1 \sqcap C_2$ is being sibling-merged for $\mathcal{T}$. Now one can show that an inclusion with properties (i)–(iv) still has those properties after applying the rule (sibling merging) above. Thus $A \sqsubseteq C$ is $\mathcal{T}$-essential, as required. ❏

Example 3 *Suppose the target TBox $\mathcal{T}$ is*[3]

$$\begin{aligned}\mathsf{Prof} \sqsubseteq\ & \mathsf{Academic} \sqcap \\ & \exists \mathsf{worksFor}.(\exists \mathsf{supports}.\mathsf{Teaching} \sqcap \\ & \qquad \exists \mathsf{supports}.\mathsf{Research}) \\ \mathsf{Academic} \sqsubseteq\ & \exists \mathsf{advisor}.\mathsf{Academic}\end{aligned}$$

The algorithm first computes $\mathcal{H}_{\mathsf{basic}}$ as

$$\begin{array}{ll}\mathsf{Prof} \sqsubseteq \mathsf{Academic} & \mathsf{Prof} \sqsubseteq \exists \mathsf{worksFor}.\top \\ \mathsf{Prof} \sqsubseteq \exists \mathsf{advisor}.\top & \mathsf{Academic} \sqsubseteq \exists \mathsf{advisor}.\top\end{array}$$

and then poses $\mathcal{H}_{\mathsf{basic}}$ as the first equivalence query. Assume that the oracle returns the positive counterexample

$$\mathsf{Prof} \sqsubseteq \exists \mathsf{advisor}.\mathsf{Academic}.$$

which is not concept saturated for $\mathcal{T}$; saturating it yields

$$\mathsf{Prof} \sqsubseteq \mathsf{Academic} \sqcap \exists \mathsf{advisor}.\mathsf{Academic}$$

which is not minimal for $\mathcal{T}$; minimizing it yields

$$\mathsf{Academic} \sqsubseteq \exists \mathsf{advisor}.\mathsf{Academic}.$$

The above CI constitutes the new $\mathcal{H}_{\mathsf{add}}$ and the algorithm asks $\mathcal{H}_{\mathsf{basic}} \cup \mathcal{H}_{\mathsf{add}}$ as an equivalence query. Assume the oracle returns the positive counterexample

$$\begin{aligned}\mathsf{Prof} \sqsubseteq\ & \exists \mathsf{worksFor}.(\exists \mathsf{supports}.\mathsf{Teaching} \sqcap \\ & \qquad \exists \mathsf{worksFor}^-.\mathsf{Academic})\end{aligned}$$

[3]To facilitate presentation, we use a TBox $\mathcal{T}$ that is not in named form; for this concrete example, however, named form does not make any difference.

which is not parent/child-merged for $\mathcal{T}$; merging it results in the following CI, which is added to $\mathcal{H}_{\mathsf{add}}$:

$$\mathsf{Prof} \sqsubseteq \mathsf{Academic} \sqcap \exists \mathsf{worksFor}.\exists \mathsf{supports}.\mathsf{Teaching}.$$

For the next equivalence query, the oracle returns the positive counterexample

$$\mathsf{Prof} \sqsubseteq \exists \mathsf{worksFor}.\exists \mathsf{supports}.\mathsf{Research}.$$

Conjoining the right-hand side of this CI with that of the existing CI for Prof *in $\mathcal{H}_{\mathsf{add}}$ and applying sibling merging yields the first CI from $\mathcal{T}$, and the algorithm has succeeded to learn $\mathcal{T}$.*

If the algorithm terminates, then it obviously has found a TBox $\mathcal{H}_{basic} \cup \mathcal{H}_{add}$ that is logically equivalent to $\mathcal{T}$. It thus remains to show that the algorithm terminates in polynomially time. Observe that $\mathcal{H}_{add}$ contains at most one concept inclusion $A \sqsubseteq C$ for each concept name A. At each step in the while loop, either some $A \sqsubseteq C$ is added to $\mathcal{H}_{add}$ such that no inclusion with A on the left-hand side existed in $\mathcal{H}_{add}$ before or an existing inclusion $A \sqsubseteq C$ in $\mathcal{H}_{add}$ is replaced by a fresh $A \sqsubseteq C'$ with $\models C' \sqsubseteq C$ and $\not\models C \sqsubseteq C'$. It is thus not hard to see that we can prove termination in polynomial time by establishing the following lemma.

Lemma 4 *The number of replacements of an existing CI $A \sqsubseteq C$ in $\mathcal{H}_{add}$ is bounded polynomially in $|\mathcal{T}|$.*

Proof.(sketch) We first observe that, as a consequence of the fact that $A \sqsubseteq C$ and its replacement $A \sqsubseteq C'$ are both $\mathcal{T}$-essential, the tree that corresponds to C is obtained from the tree that corresponds to C' by removing subtrees. Let n_C denote the number of nodes in the tree representation of C and let $A^{\mathcal{T}} = \{B \mid A \sqsubseteq B \in \mathcal{H}_{basic}\} \cup \{D \mid A \equiv_{\mathcal{T}} B, B \sqsubseteq D \in \mathcal{T}\}$. Then it suffices to prove the following

Claim. If $A \sqsubseteq C$ is $\mathcal{T}$-essential, then $n_C \leq \sum_{D \in A^{\mathcal{T}}} n_D$.

The claim is proved in the appendix using canonical models for DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes. ❑

We have obtained the following main result of this section.

Theorem 5 *DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes are polynomial time learnable using entailment and equivalence queries.*

In the appendix, we provide additional examples showing that if any of the five conditions for being $\mathcal{T}$-essential is removed, then Algorithm 1 no longer runs in polynomial time or does not even terminate. As an example, assume that minimality is omitted from the definition of being $\mathcal{T}$-essential and let $\mathcal{T} = \{A \sqsubseteq B, B \sqsubseteq \exists r.B\}$. Then the oracle can provide for the n-th equivalence query the positive counterexample $A \sqsubseteq \exists r^n.B$, $n \geq 1$. The algorithm does not terminate.

Learning $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes in Polynomial Time

We consider the restriction $\mathcal{EL}_{\mathsf{lhs}}$ of general $\mathcal{EL}$ TBoxes where only concept names are allowed on the right-hand side of concept inclusions. We assume that CIs used in entailment queries and in equivalence queries and those returned as counterexamples are also of this restricted form. Our learning algorithm is a non-trivial extension of the polynomial time agorithm for learning propositional Horn theories presented in (Angluin, Frazier, and Pitt 1992; Arias and Balcázar 2011).

We first introduce some notation. An interpretation $\mathcal{I}$ is a *tree interpretation* if the directed graph $(\Delta^{\mathcal{I}}, \bigcup_{r \in \mathsf{N_R}} r^{\mathcal{I}})$ is a tree and $r^{\mathcal{I}} \cap s^{\mathcal{I}} = \emptyset$ for all distinct $r, s \in \mathsf{N_R}$. We generally denote the root of a tree interpretation $\mathcal{I}$ with $\rho_{\mathcal{I}}$. The *product* of two interpretations $\mathcal{I}$ and $\mathcal{J}$ is the interpretation $\mathcal{I} \times \mathcal{J}$ with $\Delta^{\mathcal{I} \times \mathcal{J}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{J}}$, $(d, e) \in A^{\mathcal{I} \times \mathcal{J}}$ if $d \in A^{\mathcal{I}}$ and $e \in A^{\mathcal{J}}$, and $((d, e), (d', e')) \in r^{\mathcal{I} \times \mathcal{J}}$ if $(d, d') \in r^{\mathcal{I}}$ and $(e, e') \in r^{\mathcal{J}}$, for any concept name A and role name r. Products preserve the truth of $\mathcal{EL}$ concept inclusions (Lutz, Piro, and Wolter 2011):

Lemma 6 *For all $\mathcal{EL}$ concepts C: $d \in C^{\mathcal{I}}$ and $e \in C^{\mathcal{J}}$ iff $(d, e) \in C^{\mathcal{I} \times \mathcal{J}}$.*

One can show that the product of tree interpretations is a disjoint union of tree interpretations. If $\mathcal{I}$ and $\mathcal{J}$ are tree interpretations, we denote by $\mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ the tree interpretation that is contained in $\mathcal{I} \times \mathcal{J}$ with root $(\rho_{\mathcal{I}}, \rho_{\mathcal{J}})$.

Let $\mathcal{I}, \mathcal{J}$ be interpretations, $d \in \Delta^{\mathcal{I}}$ and $e \in \Delta^{\mathcal{J}}$. A relation $\sim \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{J}}$ is a *simulation from* $(\mathcal{I}, d)$ *to* $(\mathcal{J}, e)$ if the following conditions are satisfied: (i) $d \sim e$; (ii) $d \in A^{\mathcal{I}}$ and $d \sim e$ implies $e \in A^{\mathcal{J}}$; (iii) $(d, d') \in r^{\mathcal{I}}$ and $d \sim e$ implies $d' \sim e'$ for some $e' \in \Delta^{\mathcal{J}}$ with $(e, e') \in r^{\mathcal{J}}$. We write $(\mathcal{I}, d) \Rightarrow (\mathcal{J}, e)$ if there is a simulation from $(\mathcal{I}, d)$ to $(\mathcal{J}, e)$ and if $\mathcal{I}$ and $\mathcal{J}$ are tree interpretations then we write $\mathcal{I} \Rightarrow \mathcal{J}$ as a shorthand for $(\mathcal{I}, \rho_{\mathcal{I}}) \Rightarrow (\mathcal{J}, \rho_{\mathcal{J}})$. Simulations preserve the membership to $\mathcal{EL}$-concepts (Lutz, Piro, and Wolter 2011):

Lemma 7 *For all $\mathcal{EL}$ concepts C: if $d \in C^{\mathcal{I}}$ and $(\mathcal{I}, d) \Rightarrow (\mathcal{J}, e)$, then $e \in C^{\mathcal{J}}$.*

For an $\mathcal{EL}$ concept C, we write $\mathcal{I}_C$ to denote the tree interpretation that is obtained by viewing C as an interpretation in the natural way, that is, $\mathcal{I}_C$ is the tree T_C introduced in the context of our learning algorithm for DL-Lite$^{\exists}_{\mathcal{R}}$, represented as an interpretation. Conversely, every tree interpretation $\mathcal{I}$ can be viewed as an $\mathcal{EL}$ concept $C_{\mathcal{I}}$ in a straightforward way.

Let $\mathcal{T}$ be the target TBox to be learned, and assume again that its signature Σ is known to the learner. An interpretation $\mathcal{I}$ is a *$\mathcal{T}$-countermodel* if $\mathcal{I} \not\models \mathcal{T}$. We will now describe a class of $\mathcal{T}$-countermodels that are in a sense minimal and central to our learning algorithm. For a tree interpretation $\mathcal{I}$, we use $\mathcal{I}|^{-}_{\mathsf{root}}$ to denote the interpretation obtained from $\mathcal{I}$ by removing the root $\rho_{\mathcal{I}}$ of $\mathcal{I}$. For any element d of $\Delta^{\mathcal{I}}$, we use $\mathcal{I}|^{-}_{d\downarrow}$ to denote $\mathcal{I}$ with the subtree rooted at d removed. A $\mathcal{T}$-countermodel is *essential* if the following conditions are satisfied:

1. $\mathcal{I}|^{-}_{\mathsf{root}} \models \mathcal{T}$;
2. $\mathcal{I}|^{-}_{d\downarrow} \models \mathcal{T}$ for all $d \in \Delta^{\mathcal{I}} \setminus \{\rho_{\mathcal{I}}\}$.

Intuitively, Condition 1 states that $\mathcal{I}$ contradicts $\mathcal{T}$ only at the root, that is, the only reason for why $\mathcal{I}$ does not satisfy $\mathcal{T}$ is that for at least one CI $C \sqsubseteq A \in \mathcal{T}$, we have that $\rho_{\mathcal{I}} \in C^{\mathcal{I}}$

Algorithm 2 The learning algorithm for $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes

```
1: Set $\mathfrak{I}$ to the empty sequence (of tree interpretations)
2: Set $\mathcal{H} = \emptyset$
3: while Equivalent($\mathcal{H}$)? returns "no" do
4:     Let $C \sqsubseteq A$ be the returned positive counterexample
           for $\mathcal{T}$ relative to $\mathcal{H}$
5:     Find an essential $\mathcal{T}$-countermodel $\mathcal{I}$ with $\mathcal{I} \models \mathcal{H}$
                                                (entailment queries)
6:     if there is a $\mathcal{J} \in \mathfrak{I}$ such that $\mathcal{J} \not\Rightarrow (\mathcal{I} \times_{\mathsf{r}} \mathcal{J})$ and
               $\mathcal{I} \times_{\mathsf{r}} \mathcal{J} \not\models \mathcal{T}$ then
7:         Let $\mathcal{J}$ be the first such element of $\mathfrak{I}$
8:         Find an essential $\mathcal{T}$-countermodel $\mathcal{J}' \subseteq \mathcal{I} \times_{\mathsf{r}} \mathcal{J}$
                                                (entailment queries)
9:         replace $\mathcal{J}$ in $\mathfrak{I}$ with $\mathcal{J}'$
10:    else
11:        append $\mathcal{I}$ to $\mathfrak{I}$
12:    end if
13:    Construct $\mathcal{H} = \{C_{\mathcal{I}} \sqsubseteq A \mid \mathcal{I} \in \mathfrak{I}, \mathcal{T} \models C_{\mathcal{I}} \sqsubseteq A\}$
                                                (entailment queries)
14: end while
```

and $\rho_{\mathcal{I}} \notin A^{\mathcal{I}}$. Condition 2 is a minimality condition which states that for any such $C \sqsubseteq A$ (which needs not be unique), $\rho_{\mathcal{I}}$ is no longer in $C^{\mathcal{I}}$ if we remove any node from $\mathcal{I}$.

The algorithm for learning $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes is given as Algorithm 2, where the implementation of the lines marked with "(entailment queries)" requires the execution of entailment queries as detailed below. It maintains a sequence $\mathfrak{I}$ of tree interpretations that intuitively represents the TBox $\mathcal{H}$ constructed in Line 13; however, tree interpretations are easier to work with for our purposes than the $\mathcal{EL}$ concepts that they represent. In Line 8, we write $\mathcal{J}' \subseteq \mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ as shorthand for: $\mathcal{J}'$ is a subinterpretation of $\mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ that is obtained from $\mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ by removing subtrees. Note that the assumption in Line 4 that a *positive* counterexample is returned is justified by the construction of $\mathcal{H}$ in Lines 2 and 13, which ensures that, at all times, $\mathcal{T} \models \mathcal{H}$.

We now provide additional details on how to realize the three lines marked with "(entailment queries)". Line 13 is easiest: We simply use entailment queries to find all CIs $C_{\mathcal{I}} \sqsubseteq A$ with $\mathcal{I} \in \mathfrak{I}$ and A a concept name from Σ. We will later show that the length of $\mathfrak{I}$ is bounded polynomially in $|\mathcal{T}|$, therefore polynomially many entailment queries suffice. Lines 5 and 8 are addressed by Lemmas 8 and 9 below.

Lemma 8 *Given a positive counterexample $C \sqsubseteq A$ for $\mathcal{T}$ relative to $\mathcal{H}$, one can construct an essential $\mathcal{T}$-countermodel $\mathcal{I}$ with $\mathcal{I} \models \mathcal{H}$ using only polynomially many entailment queries in $|\mathcal{T}| + |C|$.*

Proof. $\mathcal{I}$ is constructed by applying the following rules to $\mathcal{I} := \mathcal{I}_C$.

1. Saturate $\mathcal{I}$ by exhaustively applying the CIs from $\mathcal{H}$ as rules: if $D \sqsubseteq B \in \mathcal{H}$ and $d \in D^{\mathcal{I}}$, then add d to $B^{\mathcal{I}}$.
2. Replace $\mathcal{I}$ by a minimal subtree of $\mathcal{I}$ refuting $\mathcal{T}$ to address Condition 1 of essential $\mathcal{T}$-countermodels: replace $\mathcal{I}$ by $\mathcal{I}|_d$ if $\mathcal{I}|_d$ is minimal with $\mathcal{I}|_d \not\models \mathcal{T}$ (checked using entailment queries), where $\mathcal{I}|_d$ denotes the substree of $\mathcal{I}$ rooted at d.
3. Exhaustively remove subtrees from $\mathcal{I}$ until Condition 2 of essential $\mathcal{T}$-countermodels is also satisfied: if $\mathcal{I}|^{-}_{d\downarrow} \not\models \mathcal{T}$ (checked using entailment queries), then replace $\mathcal{I}$ by $\mathcal{I}|^{-}_{d\downarrow}$.

The resulting interpretation $\mathcal{I}$ is as required. Details are presented in the appendix. ❏

Lemma 9 *Given essential $\mathcal{T}$-countermodels $\mathcal{I}$ and $\mathcal{J}$ with $\mathcal{I} \times \mathcal{J} \not\models \mathcal{T}$, one can construct an essential $\mathcal{T}$-countermodel $\mathcal{J}' \subseteq \mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ using only polynomially many entailment queries in $|\mathcal{T}| + |\mathcal{I}| + |\mathcal{J}|$.*

Proof. Let $\mathcal{I}$ and $\mathcal{J}$ be essential $\mathcal{T}$-countermodels with $\mathcal{I} \times_{\mathsf{r}} \mathcal{J} \not\models \mathcal{T}$. Set $\mathcal{J}' = \mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ and then exhaustively apply Rule 3 from the proof of Lemma 8 to $\mathcal{J}'$. Clearly, the new $\mathcal{J}'$ is a $\mathcal{T}$-countermodel. Moreover, it is essential:

(1) $\mathcal{J}'|^{-}_{\mathsf{root}} \models \mathcal{T}$: by Lemma 6, $\mathcal{I}|^{-}_{\mathsf{root}} \models \mathcal{T}$ and $\mathcal{J}|^{-}_{\mathsf{root}} \models \mathcal{T}$ imply $\mathcal{I}|^{-}_{\mathsf{root}} \times \mathcal{J}|^{-}_{\mathsf{root}} \models \mathcal{T}$. Now $\mathcal{J}'|^{-}_{\mathsf{root}}$ can be obtained from $\mathcal{I}|^{-}_{\mathsf{root}} \times \mathcal{J}|^{-}_{\mathsf{root}}$ by removing subtrees, and removing subtrees clearly preserves being a model of an $\mathcal{EL}_{\mathsf{lhs}}$ TBox.

(2) $\mathcal{J}'|^{-}_{d\downarrow} \models \mathcal{T}$ for all $d \in \Delta^{\mathcal{J}'} \setminus \{\rho_{\mathcal{J}'}\}$: otherwise, the subtree rooted at d would have been removed during the construction of $\mathcal{J}'$. ❏

Example 10 *Suppose that the target TBox is*

$$\exists \mathsf{parent}.\mathsf{Mouse} \sqsubseteq \mathsf{Mouse}$$
$$\exists \mathsf{parent}.\mathsf{Rabbit} \sqsubseteq \mathsf{Rabbit}$$
$$\exists \mathsf{parent}.\top \sqsubseteq \mathsf{Living}$$

and assume that the oracle returns to the first equivalence query the positive counterexample $C \sqsubseteq \mathsf{Living}$ with

$$C = \exists \mathsf{parent}.\mathsf{Mouse} \sqcap \exists \mathsf{parent}.\mathsf{Rabbit}$$

$\mathcal{I}_C$ is a $\mathcal{T}$-countermodel that violates Condition 2 of being essential which leads to its rewriting into $\mathcal{I}_{C'}$ with

$$C' = \exists \mathsf{parent}.\mathsf{Mouse}$$

(Alternatively, it could also lead to $C' = \exists \mathsf{parent}.\mathsf{Rabbit}$.) The algorithm inserts $\mathcal{I}_{C'}$ into $\mathfrak{I}$ and asks the next equivalence query, to which the oracle returns $D \sqsubseteq \mathsf{Living}$ with

$$D = \exists \mathsf{parent}.\mathsf{Rabbit}.$$

$\mathcal{I}_D$ is an essential $\mathcal{T}$-countermodel and we have $\mathcal{I}_{C'} \not\Rightarrow (\mathcal{I}_D \times_{\mathsf{r}} \mathcal{I}_{C'})$, where $\mathcal{I}_D \times_{\mathsf{r}} \mathcal{I}_{C'}$ corresponds to the concept

$$D' = \exists \mathsf{parent}.\top.$$

$\mathcal{I}_{D'}$ is $\mathcal{T}$-essential and thus it is inserted into $\mathfrak{I}$ in place of $\mathcal{I}_{C'}$. At this point, the algorithm has learnt $\mathcal{H} = \{\exists \mathsf{parent}.\top \sqsubseteq \mathsf{Living}\}$. The algorithm asks the next equivalence query, to which the oracle returns $E \sqsubseteq \mathsf{Rabbit}$ with

$$E = \exists \mathsf{parent}.\exists \mathsf{parent}.\mathsf{Rabbit}$$

$\mathcal{I}_E$ violates Conditions 1 and 2 of being essential, which leads to its rewriting into $\mathcal{I}_{E'}$ with

$$E' = \mathsf{Living} \sqcap \exists \mathsf{parent}.\mathsf{Rabbit}.$$

We have $\mathcal{I}_{D'} \Rightarrow \mathcal{I}_{E'} \times_{\mathsf{r}} \mathcal{I}_{D'}$ and so $\mathcal{I}_{E'}$ is appended to the list $\mathfrak{I}$. At this point, the algorithm has learnt $\mathcal{H} = \{\mathsf{Living} \sqcap \exists \mathsf{parent}.\mathsf{Rabbit} \sqsubseteq \mathsf{Rabbit}, \exists \mathsf{parent}.\top \sqsubseteq \mathsf{Living}\}$, which is equivalent to the TBox that consists of the second and third CI in $\mathcal{T}$. The algorithm can now proceed to learn $\mathsf{Living} \sqcap \exists \mathsf{parent}.\mathsf{Mouse} \sqsubseteq \mathsf{Mouse}$.

If the algorithm terminates, then it obviously returns a TBox $\mathcal{H}$ that is equivalent to the target TBox $\mathcal{T}$. It thus remains to prove that the algorithm terminates after polynomially many steps, which is a consequence of the following lemma.

Lemma 11 *Let $\mathfrak{I}$ be a sequence computed at some point of an execution of Algorithm 2. Then (i) the length of $\mathfrak{I}$ is bounded by the number of CIs in $\mathcal{T}$ and (ii) each interpretation in each position of $\mathfrak{I}$ is replaced only $|\mathcal{T}| + |\mathcal{T}|^2$ often with a new interpretation.*

Proof.(sketch) Assume that at each point of the execution of the algorithm, $\mathfrak{I}$ has the form $\mathcal{I}_0, \dots, \mathcal{I}_k$ for some $k \geq 0$. To establish Point (i), we closely follow (Angluin, Frazier, and Pitt 1992) and show that (iii) for every $\mathcal{I}_i$, there is a $D_i \sqsubseteq A_i \in \mathcal{T}$ with $\mathcal{I}_i \not\models D_i \sqsubseteq A_i$ and (iv) if $i \neq j$, then $D_i \sqsubseteq A_i$ and $D_j \sqsubseteq A_j$ are not identical.

The proof of Points (iii) and (iv), in turn requires a careful analysis of the concept inclusions in the target TBox that are violated at the root of the $\mathcal{T}$-countermodels $\mathcal{I}_0, \dots, \mathcal{I}_k$.

For the proof of Point (ii), we show that $|\Delta^{\mathcal{I}}| \leq |\mathcal{T}|$ for any essential $\mathcal{T}$-countermodel $\mathcal{I}$ and analyze the relationship between the essential $\mathcal{T}$-countermodels produced from $\mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ when $\mathcal{J} \not\Rightarrow (\mathcal{I} \times_{\mathsf{r}} \mathcal{J})$ (using Lemma 6 and 7). ❏

We have thus etablished the main result of this section.

Theorem 12 *$\mathcal{EL}_{\mathsf{lhs}}$ TBoxes are polynomial time learnable using entailment and equivalence queries.*

It can again be shown that all adopted conditions are indeed necessary. Consider for example the first condition of being $\mathcal{T}$-essential. Without it, for the target TBox $\mathcal{T} = \{\exists r.A \sqsubseteq A\}$ the oracle can return the infinite set of positive counterexamples $\exists r^n.A \sqsubseteq A$, n a prime number. Then the learning algorithm does not terminate.

Limits of Polynomial Time Learnability

The main result of this section is that $\mathcal{EL}$ TBoxes cannot be learned in polynomial time using entailment and equivalence queries. We also show that DL-Lite$_{\mathcal{R}}^{\exists}$ TBoxes cannot be learned in polynomial time using entailment or equivalence queries alone. This holds for $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes as well and follows from the fact that propositional Horn theories cannot be learned in polynomial time using entailment or equivalence queries alone (Frazier and Pitt 1993; Angluin, Frazier, and Pitt 1992; Angluin 1987a).

We start by proving the non-learnability result for $\mathcal{EL}$ TBoxes. Our proof shows that even *acyclic* target TBoxes cannot be learned in polynomial time when general $\mathcal{EL}$ concept inclusions are admitted in entailment queries, as counterexamples returned by the oracle, and in TBoxes used as equivalence queries. On our way, we also prove non-learnability of DL-Lite$_{\mathcal{R}}^{\exists}$ TBoxes using entailment queries only. The proof is inspired by Angluin's lower bound for the following abstract learning problem (1987c): a learner aims to identify one of N distinct sets $L_1, \dots, L_N$ which have the property that there exists a set $L_\cap$ for which $L_i \cap L_j = L_\cap$, for any $i \neq j$. It is assumed that $L_\cap$ is not a valid argument to an equivalence query. The learner can pose membership queries "$x \in L$?" and equivalence queries "$H = L$?". Then in the worst case it takes at least $N - 1$ membership and equivalence queries to exactly identify a hypothesis L_i from $L_1, \dots, L_N$. The proof proceeds as follows. At every stage of computation, the oracle (which here should be viewed as an adversary) maintains a set of hypotheses S, which the learner is not able to distinguish based on the answers given so far. Initially, $S = \{L_1, \dots, L_N\}$. When the learner asks a membership query x, the oracle returns 'Yes' if $x \in L_\cap$ and 'No' otherwise. In the latter case, the (unique) L_i such that $x \in L_i$ is removed from S. When the learner asks an equivalence query H, the oracle returns 'No' and a counterexample $x \in L_\cap \oplus H$ (the symmetric difference of $L_\cap$ and H). This always exists as $L_\cap$ is not a valid query. If the counterexample x is not a member of $L_\cap$, (at most one) $L_i \in S$ such that $x \in L_i$ is eliminated from S. In the worst case, the learner has to reduce the cardinality of S to one to exactly identify a hypothesis, which takes $N - 1$ queries.

Similarly to the method outlined above, in our proof we maintain a set of acyclic $\mathcal{EL}$ TBoxes S whose members the learning algorithm is not able to distinguish based on the answers obtained so far. We start with $S = \{\mathcal{T}_1, \dots, \mathcal{T}_N\}$, where N is superpolynomial in the size of every TBox $\mathcal{T}_i$, and describe an oracle that responds to entailment and equivalence queries. For didactic purposes, we first present a set of acyclic TBoxes $\mathcal{T}_1, \dots, \mathcal{T}_N$, for which the oracle can respond to entailment queries in the way described above but which is polynomial time learnable when equivalence queries are also allowed. We then show how the TBoxes can be modified to obtain a family of acyclic TBoxes that is not polynomial time learnable using entailment and equivalence queries.

To present the TBoxes in S, fix two role names r and s. For any sequence $\sigma = \sigma^1\sigma^2 \dots \sigma^n$ with $\sigma_i \in \{r, s\}$, the expression $\exists\boldsymbol{\sigma}.C$ stands for $\exists\sigma^1.\exists\sigma^2 \dots \exists\sigma^n.C$. For every such sequence $\boldsymbol{\sigma}$, of which there are $N = 2^n$ many, consider the acyclic $\mathcal{EL}$ TBox $\mathcal{T}_\sigma$ defined as

$$\begin{aligned} \mathcal{T}_{\boldsymbol{\sigma}} &= \{A \sqsubseteq \exists\boldsymbol{\sigma}.M \sqcap X_0\} \cup \mathcal{T}_0 \text{ with} \\ \mathcal{T}_0 &= \{X_i \sqsubseteq \exists r.X_{i+1} \sqcap \exists s.X_{i+1} \mid 0 \leq i < n\} \end{aligned}$$

where $\mathcal{T}_0$ generates a full binary tree whose edges are labelled with the role names r and s and with X_0 at the root, X_1 at level 1 and so on. M is a concept name that 'marks' a particular path in this tree given by the sequence $\boldsymbol{\sigma}$. One can use Angluin's strategy to show that TBoxes from the set S of all such TBoxes $\mathcal{T}_{\boldsymbol{\sigma}}$ cannot be learned in polynomial time using entailment queries only: notice that for no sequence $\boldsymbol{\sigma}' \neq \boldsymbol{\sigma}$ of length n, we have $\mathcal{T}_{\boldsymbol{\sigma}} \models A \sqsubseteq \exists\boldsymbol{\sigma}'.M$. Thus an entailment query of the form $A \sqsubseteq \exists\boldsymbol{\sigma}.M$ eliminates at most one TBox from the set of TBoxes that the learner cannot distinguish. This observation can be generalized to arbitary entailment queries $C \sqsubseteq D$ in $\mathcal{EL}$ since one can prove, similarly to the proof of Lemma 14 below, that for any $\mathcal{EL}$ concept inclusion $C \sqsubseteq D$ either $\{A \sqsubseteq X_0\} \cup \mathcal{T}_0 \models C \sqsubseteq D$ (thus $C \sqsubseteq D$ is entailed by all TBoxes in S) or at most one TBox from S entails $C \sqsubseteq D$. It follows that acyclic $\mathcal{EL}$ TBoxes are not polynomial time learnable using entailment queries, only. Observe that the TBoxes $\mathcal{T}_{\boldsymbol{\sigma}}$ are actually formulated in DL-Lite$^{\exists}_{\mathcal{R}}$. We show in the appendix that all arguments above are true also when entailment queries are formulated in DL-Lite$^{\exists}_{\mathcal{R}}$ (which include inverse roles) and thus obtain the following result.

Theorem 13 *DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes are not polynomial time learnable using entailment queries, only.*

Interestingly, a single equivalence query is sufficient to learn any TBox from S in two steps: given the equivalence query $\{A \sqsubseteq X_0\} \cup \mathcal{T}_0$, the oracle has no other option but to reveal the target TBox $\mathcal{T}_{\boldsymbol{\sigma}}$ as $A \sqsubseteq \exists\boldsymbol{\sigma}.M$ can be found 'inside' every counterexample. Our strategy to rule out this option for the oracle is to modify $\mathcal{T}_1, \dots, \mathcal{T}_N$ in such a way that although a TBox $\mathcal{T}_\cap$ axiomatizing the intersection over the set of consequences of each $\mathcal{T}_i$, $i \leq N$, exists, its size is superpolynomial and so cannot be used as an equivalence query by a polynomial time learning algorithm.

For every $n > 0$ and every n-tuple $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n)$, where every $\boldsymbol{\sigma}_i$ is a role sequence of length n as above, we define an acyclic $\mathcal{EL}$ TBox $\mathcal{T}_L$ as the union of $\mathcal{T}_0$ and the following inclusions:[4]

$$\begin{array}{lcl} A_1 \sqsubseteq \exists\boldsymbol{\sigma}_1.M \sqcap X_0 & & A_n \sqsubseteq \exists\boldsymbol{\sigma}_n.M \sqcap X_0 \\ B_1 \sqsubseteq \exists\boldsymbol{\sigma}_1.M \sqcap X_0 & \cdots & B_n \sqsubseteq \exists\boldsymbol{\sigma}_n.M \sqcap X_0 \end{array}$$
$$A \equiv X_0 \sqcap \exists\boldsymbol{\sigma}_1.M \sqcap \cdots \sqcap \exists\boldsymbol{\sigma}_n.M.$$

Let $\mathfrak{L}_n$ be a set of n-tuples such that for $1 \leq i \leq n$ and every $L, L' \in \mathfrak{L}_n$ with $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n)$, $L' = (\boldsymbol{\sigma}'_1, \dots, \boldsymbol{\sigma}'_n)$, if $\boldsymbol{\sigma}_i = \boldsymbol{\sigma}'_j$ then $L = L'$ and $i = j$. Then for any sequence $\boldsymbol{\sigma}$ of length n there exists at most one $L \in \mathfrak{L}_n$ and at most one $i \leq n$ such that $\mathcal{T}_L \models A_i \sqsubseteq \exists\boldsymbol{\sigma}.M$ and $\mathcal{T}_L \models B_i \sqsubseteq \exists\boldsymbol{\sigma}.M$. We can choose $\mathfrak{L}_n$ such that there are $N = \lfloor 2^n/n \rfloor$ different tuples in $\mathfrak{L}_n$. Notice that the size of each $\mathcal{T}_L$ with $L \in \mathfrak{L}_n$ is polynomial in n and so N is superpolynomial in the size of each $\mathcal{T}_L$ with $L \in \mathfrak{L}_n$.

Every $\mathcal{T}_L$, for $L \in \mathfrak{L}_n$, entails, among other inclusions, $\bigsqcap_{i=1}^{n} C_i \sqsubseteq A$, where every C_i is either A_i or B_i. There are 2^n different such inclusions, which indicates that every representation of the 'intersection TBox' requires superpolynomially many axioms. It follows from Lemma 15 below that this is indeed the case.

The following lemma (proved in the appendix) enables us to respond to entailment queries without eliminating too many TBoxes from the list S of TBoxes that the learner cannot distinguish. We use Σ_n to denote the signature of $\mathcal{T}_L$.

[4]In fact, to prove non-polynomial learnability, it suffices to consider $\exists\boldsymbol{\sigma}_1.M \sqcap \cdots \sqcap \exists\boldsymbol{\sigma}_n.M \sqsubseteq A$ in place of the concept equality; however, inclusions of this form are not allowed in acyclic TBoxes.

Lemma 14 *For all $\mathcal{EL}$ concept inclusions $C \sqsubseteq D$ over Σ_n:*

- *either $\mathcal{T}_L \models C \sqsubseteq D$ for every $L \in \mathfrak{L}_n$ or*
- *the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C \sqsubseteq D$ does not exceed $|C|$.*

To illustrate the result, consider two TBoxes $\mathcal{T}_L$ and $\mathcal{T}_{L'}$, where $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n)$ and $L' = (\boldsymbol{\sigma}'_1, \dots, \boldsymbol{\sigma}'_n)$. Then the inclusion $X_0 \sqcap \exists\boldsymbol{\sigma}_1.M \sqcap \exists\boldsymbol{\sigma}'_1.M \sqcap A_2 \sqcap \cdots \sqcap A_n \sqsubseteq A$ is entailed by both $\mathcal{T}_L$ and $\mathcal{T}_{L'}$ but not by any other $\mathcal{T}_{L''}$ with $L \in \mathfrak{L}_n$.

We now show how the oracle can answer equivalence queries, aiming to show that for any polynomial size equivalence query $\mathcal{H}$, the oracle can return a counterexample $C \sqsubseteq D$ such that either (i) $\mathcal{H} \models C \sqsubseteq D$ and $\mathcal{T}_L \models C \sqsubseteq D$ for at most one $L \in \mathfrak{L}_n$ or (ii) $\mathcal{H} \not\models C \sqsubseteq D$ and for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D$. Thus, such a counterexample eliminates at most one $\mathcal{T}_L$ from the set S of TBoxes that the learner cannot distinguish. In addition, however, we have to take extra care of the size of counterexamples as the learning algorithm is allowed to run in time polynomial not only in the size of the target TBox but also in the size of the counterexamples returned by the oracle. For instance, if the hypothesis TBox $\mathcal{H}$ contains an inclusion $C \sqsubseteq D$ which is not entailed by any $\mathcal{T}_L$, one cannot simply return $C \sqsubseteq D$ as a counterexample since the learner will be able to 'pump up' its running time by asking for equivalence of the target TBox to a sequence of hypotheses $\mathcal{H}_i = \{C_i \sqsubseteq D_i\}$ such that the size of $C_{i+1} \sqsubseteq D_{i+1}$ is twice the size of $C_i \sqsubseteq D_i$. Then at every stage in a run of the learning algorithm, the running time will be polynomial in the size of the input and the size of the largest counterexample received so far, but the overall running time will be exponential in the size of input. The following lemma addresses this issue.

Lemma 15 *For any $n > 1$ and any $\mathcal{EL}$ TBox $\mathcal{H}$ in Σ_n with $|\mathcal{H}| < 2^n$, there exists an $\mathcal{EL}$ CI $C \sqsubseteq D$ over Σ_n such that (i) the size of $C \sqsubseteq D$ does not exceed $6n$ and (ii) if $\mathcal{H} \models C \sqsubseteq D$ then $\mathcal{T}_L \models C \sqsubseteq D$ for at most one $L \in \mathfrak{L}_n$ and if $\mathcal{H} \not\models C \sqsubseteq D$ then for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D$.*

Then we have the following.

Theorem 16 *$\mathcal{EL}$ TBoxes are not polynomial time learnable using entailment and equivalence queries.*

Proof. Assume that TBoxes are polynomial time learnable. Then there exists a learning algorithm whose running time is bounded at any stage by a polynomial $p(n, m)$. Choose n such that $\lfloor 2^n/n \rfloor > (p(n, 6n))^2$ and let $S = \{\mathcal{T}_L \mid L \in \mathfrak{L}_n\}$. We follow Angluin's strategy of letting the oracle remove TBoxes from S in such a way that the learner cannot distinguish between any of the remaining TBoxes. Given an entailment query $C \sqsubseteq D$, if $\mathcal{T}_L \models C \sqsubseteq D$ for every $L \in \mathfrak{L}_n$, then the answer is 'yes'; otherwise the answer is 'no' and all $\mathcal{T}_L$ with $\mathcal{T}_L \models C \sqsubseteq D$ are removed from S (by Lemma 14, there are at most $|C|$ such TBoxes). Given an equivalence query $\mathcal{H}$, the answer is 'no', a counterexample $C \sqsubseteq D$ guaranteed by Lemma 15 is produced, and (at most one) $\mathcal{T}_L$ such that $\mathcal{T}_L \models C \sqsubseteq D$ is removed from S.

As all counterexamples produced are smaller than $6n$, the overall running time of the algorithm is bounded by $p(n, 6n)$. Hence, the learner asks no more than $p(n, 6n)$ queries and the size of every query does not exceed $p(n, 6n)$. By Lemmas 14 and 15, at most $(p(n, 6n))^2$ TBoxes are removed from S during the run of the algorithm. But then, the algorithm cannot distinguish between any remaining TBoxes and we have derived a contradiction. ❏

We now show that DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes cannot be learnt in polynomial time using equivalence queries only. We use the following result on non-learnability of monotone DNF formulas using equivalence queries due to Angluin (1990). Here, equivalence queries take a hypothesis ψ in the form of a monotone DNF formula and return as a counterexample either a truth assignment that satisfies ψ but not the target formula ϕ or vice versa. Let $M(n, t, s)$ denote the set of all monotone DNF formulas whose variables are $x_1, \ldots, x_n$, that have exactly t conjunctions, and where each conjunction contains exactly s variables.

Theorem 17 (Angluin) *For any polynomial $q(\cdot)$ there exist constants t_0 and s_0 and a strategy for the oracle $\mathfrak{O}$ to answer equivalence queries posed by a learning algorithm in such a way that for sufficiently large n any learning algorithm that asks at most $q(n)$ equivalence queries, each bounded in size by $q(n)$, cannot exactly identify elements of $M(n, t_0, s_0)$.*

To employ Theorem 17, we associate with every monotone DNF formula $\phi = \bigvee_{i=1}^{t}(x^i_1 \wedge \cdots \wedge x^i_{s^i})$ with variables $x_1, \ldots, x_n$ a DL-Lite$^{\exists}_{\mathcal{R}}$ TBox $\mathcal{T}_\phi$ as follows. With each conjunct $x^i_1 \wedge \cdots \wedge x^i_{s^i}$ we associate a concept $C_i := \exists\rho^i_1.\exists\rho^i_2.\ldots\exists\rho^i_n.\top$ where $\rho^i_j = r$ if x_j occurs in $x^i_1 \wedge \cdots \wedge x^i_{s^i}$ and $\rho^i_j = \bar{r}$ otherwise (r and $\bar{r}$ are role names). Let A be a concept name and set

$$\mathcal{T}_\phi = \{A \sqsubseteq \textstyle\bigsqcap_{i=1}^{t} C_i, \quad \bar{r} \sqsubseteq r\}.$$

For example, for $n = 4$ and $\phi = (x_1 \wedge x_4) \vee x_2$ we have

$$\mathcal{T}_\phi = \{A \sqsubseteq \exists r.\exists\bar{r}.\exists\bar{r}.\exists r.\top,\ A \sqsubseteq \exists\bar{r}.\exists r.\exists\bar{r}.\exists\bar{r}.\top,\ \bar{r} \sqsubseteq r\}.$$

A truth assignment I (for the variables $x_1 \ldots, x_n$) also corresponds to a concept $C_I := \exists\rho^i_1.\exists\rho^i_2.\ldots\exists\rho^i_n.\top$, where $\rho^i_j = r$ if I makes x_j true and $\rho^i_j = \bar{r}$ otherwise. Then $I \models \phi$ iff $\mathcal{T}_\phi \models A \sqsubseteq C_I$ holds for all truth assignments I.

Note that $\bar{r}$ represents that a variable is false and r that a variable is true. Thus, the role inclusion $\bar{r} \sqsubseteq r$ captures the monotonicity of the DNF formulas considered. For any fixed values n, s and t, we set $T(n, t, s) = \{\mathcal{T}_\phi \mid \phi \in M(n, t, s)\}$.

Theorem 18 *The class of DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes is not polynomial time learnable using equivalence queries.*

Proof. We sketch the proof for the case when the TBoxes $\mathcal{H}$ from any equivalence query represent a monotone DNF formula in the variables $x_1, \ldots, x_n$, that is, if all equivalence queries $\mathcal{H}$ are of the form

$$\{A \sqsubseteq \textstyle\bigsqcap_{\rho_1\cdots\rho_n\in\Gamma} \exists\rho_1.\exists\rho_2.\ldots\exists\rho_n.\top, \quad \bar{r} \sqsubseteq r\},$$

for some $\Gamma \subseteq \{r, \bar{r}\}^n$. Arbitrary DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes in equivalence queries are considered in the appendix.

For a proof by contradiction, suppose that the running time of a learning algorithm $\mathfrak{A}$ for DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes in $\Sigma = \{A, r, \bar{r}\}$ is bounded at every stage of computation by a polynomial $p(x, y)$, where x is the size of the target TBox, and y is the maximal size of a counterexample returned by the oracle up to the current stage of computation. Let $q(n) = p(n^2, 4n + 2)$, and let t_0 and s_0 be the constants from the strategy of the oracle in Theorem 17. Let n be sufficiently large so that the claim of Theorem 17 holds.

Consider the oracle $\mathfrak{O}'$ that responds to every equivalence query $\mathcal{H}$ that represents a monotone DNF-formula ψ in n variables by returning the counterexample $A \sqsubseteq C_I$ corresponding to the truth assignment I that $\mathfrak{O}$ would return for the equivalence query ψ. We show that $\mathfrak{A}$ cannot distinuish between certain TBoxes in $T(n, t_0, s_0)$ and thus obtain a contradiction.

The largest counterexample returned by $\mathfrak{O}'$ is of the form $A \sqsubseteq \exists\rho_1.\cdots\exists\rho_n.\top$, so for sufficiently large n the maximal size of any counterexample returned by $\mathfrak{O}'$ in any run of the algorithm is bounded by $4n + 2$. Similarly, the size of every potential target TBox $\mathcal{T}_\phi \in T(n, t_0, s_0)$ does not exceed $t_0 \cdot (4n + 2)$ and, as t_0 is a constant, for sufficiently large n it is bounded by n^2. Thus, for sufficiently large n the total running time of $\mathfrak{A}$ for any target TBox in $T(n, t_0, s_0)$ is bounded by $p(n^2, 4n + 2)$. So, the size of a monotone DNF equivalence query forwarded to the strategy $\mathfrak{O}$ is bounded by $q(n)$, and there will be at most $q(n)$ queries forwarded. But then $\mathfrak{O}$ returns answers such that some ϕ and ψ from $M(n, t_0, s_0)$ are not distinguished. It remains to observe that $\mathfrak{A}$ does not distinguish $\mathcal{T}_\phi$ and $\mathcal{T}_\psi$. ❏

Future Work

We have presented the first study of learnability of DL TBoxes in Angluin et al's framework of learning via queries. Many research questions remain to be explored. An immediate question is whether acyclic $\mathcal{EL}$ TBoxes can be learned in polynomial time using queries and counterexamples of the form $A \equiv C$ and $A \sqsubseteq C$ only. Note that our non-learnability result for acyclic $\mathcal{EL}$ TBoxes relies heavily on counterexamples that are not of this form. Another immediate question is whether the extension of $\mathcal{EL}_{\mathsf{lhs}}$ with inverse roles (which is a better approximation of OWL2 RL than $\mathcal{EL}_{\mathsf{lhs}}$ itself) can still be learned in polynomial time. Other interesting research directions are non-polynomial time learning algorithms for $\mathcal{EL}$ TBoxes and the admission of different types of membership queries and counterexamples in the learning protocol. For example, one could replace concept inclusions as counterexamples with interpretations. In an OBDA context, one could allow membership queries that speak about certain answers to queries over an ABox and relative to the target TBox. Our results provide a good starting point for studying such variations.

Acknowledgements Konev and Wolter were supported by the EPSRC project EP/H043594/1. Lutz was supported by the DFG project Prob-DL (LU1417/1-1). Ozaki is supported by the Science without Borders scholarship programme.

References

Angluin, D.; Frazier, M.; and Pitt, L. 1992. Learning conjunctions of Horn clauses. *Machine Learning* 9:147–164.

Angluin, D. 1987a. Learning propositional Horn sentences with hints. Technical report, Yale University.

Angluin, D. 1987b. Learning regular sets from queries and counterexamples. *Inf. Comput.* 75(2):87–106.

Angluin, D. 1987c. Queries and concept learning. *Machine Learning* 2(4):319–342.

Angluin, D. 1990. Negative results for equivalence queries. *Machine Learning* 5:121–150.

Arias, M., and Balcázar, J. L. 2011. Construction and learnability of canonical Horn formulas. *Machine Learning* 85(3):273–297.

Arias, M., and Khardon, R. 2002. Learning closed Horn expressions. *Inf. Comput.* 178(1):214–240.

Arias, M.; Khardon, R.; and Maloberti, J. 2007. Learning horn expressions with logan-h. *Journal of Machine Learning Research* 8:549–587.

Baader, F., and Molitor, R. 2000. Building and structuring description logic knowledge bases using least common subsumers and concept analysis. In *ICCS*, 292–305.

Baader, F.; Calvanese, D.; McGuiness, D.; Nardi, D.; and Patel-Schneider, P. 2003. *The Description Logic Handbook: Theory, implementation and applications*. Cambridge University Press.

Baader, F.; Ganter, B.; Sertkaya, B.; and Sattler, U. 2007. Completing description logic knowledge bases using formal concept analysis. In *IJCAI*, 230–235.

Baader, F.; Brandt, S.; and Lutz, C. 2005. Pushing the $\mathcal{EL}$ envelope. In *IJCAI*, 364–369. Professional Book Center.

Borchmann, D., and Distel, F. 2011. Mining of $\mathcal{EL}$-GCIs. In *The 11th IEEE International Conference on Data Mining Workshops*. Vancouver, Canada: IEEE Computer Society.

Buitelaar, P.; Cimiano, P.; and Magnini, B., eds. 2005. *Ontology Learning from Text: Methods, Evaluation and Applications*. IOS Press.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *Journal of Automated reasoning* 39(3):385–429.

Cimiano, P.; Hotho, A.; and Staab, S. 2005. Learning concept hierarchies from text corpora using formal concept analysis. *J. Artif. Intell. Res. (JAIR)* 24:305–339.

Frazier, M., and Pitt, L. 1993. Learning from entailment: An application to propositional horn sentences. In *ICML*, 120–127.

Frazier, M., and Pitt, L. 1996. Classic learning. *Machine Learning* 25(2-3):151–193.

Konev, B.; Ludwig, M.; Walther, D.; and Wolter, F. 2012. The logical difference for the lightweight description logic EL. *J. Artif. Intell. Res. (JAIR)* 44:633–708.

Konev, B.; Lutz, C.; Ozaki, A.; and Wolter, F. 2013. Appendix of: Exact learning of lightweight description logic ontologies. http://cgi.csc.liv.ac.uk/~frank/publ/publ.html.

Kontchakov, R.; Lutz, C.; Toman, D.; Wolter, F.; and Zakharyaschev, M. 2010. The combined approach to query answering in DL-Lite. In *KR*.

Lehmann, J., and Hitzler, P. 2010. Concept learning in description logics using refinement operators. *Machine Learning* 78(1-2):203–250.

Lutz, C.; Piro, R.; and Wolter, F. 2011. Description logic TBoxes: Model-theoretic characterizations and rewritability. In *IJCAI*, 983–988.

Ma, Y., and Distel, F. 2013. Learning formal definitions for snomed ct from text. In *AIME*, 73–77.

Motik, B.; Grau, B. C.; Horrocks, I.; Wu, Z.; Fokoue, A.; and Lutz, C. 2009. Owl 2 web ontology language: Profiles. W3C Recommendation.

Raedt, L. D. 1997. Logical settings for concept-learning. *Artif. Intell.* 95(1):187–201.

Reddy, C., and Tadepalli, P. 1999. Learning Horn definitions: Theory and an application to planning. *New Generation Comput.* 17(1):77–98.

Selman, J., and Fern, A. 2011. Learning first-order definite theories via object-based queries. In *ECML/PKDD (3)*, 159–174.

ten Cate, B.; Dalmau, V.; and Kolaitis, P. G. 2012. Learning schema mappings. In *ICDT*, 182–195.

Voelker, J. 2009. *Learning Expressive Ontologies*. Ph.D. Dissertation, Universitaet Karlsruhe.

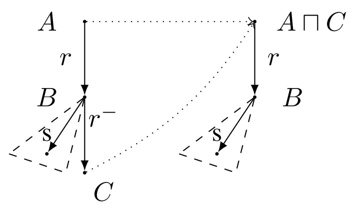

Figure 2: Parent/child merging

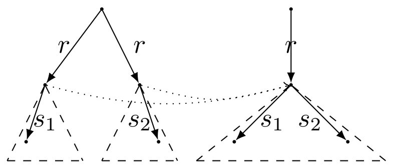

Figure 3: Sibling merging

Proofs for: Learning DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes in Polynomial Time

The reader can find illustrations of parent/child merging and sibling merging in Figures 2 and 3, respectively.

We show that if any of the first four conditions for being $\mathcal{T}$-essential is removed, then Algorithm 1 requires at least exponential time or does not even terminate. An example showing that minimality (Condition 5) is required has been given already in the paper.

1. Assume concept saturation is omitted from the definition of being $\mathcal{T}$-essential. Let $\mathcal{T} = \{A \sqsubseteq \exists r.A\}$. Then the oracle can provide for the n-th equivalence query in the while loop of Algorithm 1 the positive counterexample $A \sqsubseteq \exists r^n.\top$, $n \geq 2$. The algorithm does not terminate.
2. Assume role saturation is omitted from the definition of being $\mathcal{T}$-essential. Let $\mathcal{T} = \{A \sqsubseteq \exists e_1.\exists e_2.\ldots\exists e_n.\top\} \cup \{e_1 \sqsubseteq r_1, e_1 \sqsubseteq s_1, e_2 \sqsubseteq r_2, e_2 \sqsubseteq s_2, ..., e_n \sqsubseteq r_n, e_n \sqsubseteq s_n\}$. For $M \subseteq \{1,\ldots,n\}$ (with $\overline{M} = \{1,\ldots,n\} \setminus M$), set $C_M = \exists t_1.\exists t_2.\ldots\exists t_n.\top$, where $t_i = r_i$ if $i \in M$ and $t_i = s_i$ if $i \in \overline{M}$, $1 \leq i \leq n$. Then the oracle can provide for the first 2^n equivalence queries in the while loop of Algorithm 1 a fresh positive counterexample $A \sqsubseteq C_M$ by always choosing a fresh $M \subseteq \{1,\ldots,n\}$.
3. Assume sibling-merged is omitted from the definition of being $\mathcal{T}$-essential. Define concepts C_i as follows:
$$C_1 = \exists r.\top \sqcap \exists s.\top, \quad C_{i+1} = C_1 \sqcap \exists e.C_i$$
and let $\mathcal{T}_n = \{A \sqsubseteq \exists e.C_n\}$. For $M \subseteq \{1,\ldots,n\}$ set $C_1^M = \exists r.\top$ if $1 \in M$ and $C_1^M = \exists s.\top$ if $1 \in \overline{M}$. Also, let $C_{i+1}^M = \exists r.\top \sqcap \exists e.C_i^M$ if $i+1 \in M$ and $C_{i+1}^M = \exists s.\top \sqcap \exists e.C_i^M$ if $i+1 \in \overline{M}$, $1 \leq i \leq n$. Then the oracle can provide for the first 2^n equivalence queries in the while loop of Algorithm 1 the fresh positive counterexample $A \sqsubseteq \exists e.C_n^M$ by always choosing a fresh $M \subseteq \{1,\ldots,n\}$.
4. Assume parent/child-merged is omitted from the definition of $\mathcal{T}$-essential. Let $\mathcal{T} = \{A \sqsubseteq \exists r.\top \sqcap \exists s.\top \sqcap \exists e.B\}$. For $M \subseteq \{1,\ldots,n\}$ (with $\overline{M} = \{1,\ldots,n\} \setminus M$), set $C_M = \exists t_1.\exists t_1^-.\exists t_2.\exists t_2^-.\ldots\exists t_n.\exists t_n^-.\exists e.\top$), where $t_i = r$ if $i \in M$ and $t_i = s$ if $i \in \overline{M}$, $1 \leq i \leq n$. Then the oracle can provide for the first 2^n equivalence queries in the while loop of Algorithm 1 a fresh positive counterexample $A \sqsubseteq C_M$ by always choosing a fresh $M \subseteq \{1,\ldots,n\}$.

We provide proofs for the two claims left unproven in the proof sketch of Lemma 4. In what follows we often use the fact that for any interpretation $\mathcal{J}$, $d \in \Delta^{\mathcal{J}}$, and $\mathcal{ELI}$ concept F, $d \in F^{\mathcal{J}}$ iff there exists a homomorphism h from the labelled tree corresponding to F into $\mathcal{J}$ with $h(a_D) = d$.

Lemma 4 The number of times that an existing CI $A \sqsubseteq C$ is replaced in $\mathcal{H}_{add}$ by a fresh CI $A \sqsubseteq C'$ is bounded polynomially in $|\mathcal{T}|$.

Proof. We first show the following

Claim 1. If $A \sqsubseteq C$ is $\mathcal{T}$-essential, $\mathcal{T} \models A \sqsubseteq C'$, and $\models C' \sqsubseteq C$, then the tree corresponding to C is obtained from the tree corresponding to C' by removing subtrees.

Proof of Claim 1. Assume that $A \sqsubseteq C$ is $\mathcal{T}$-essential, $\mathcal{T} \models A \sqsubseteq C'$, and $\models C' \sqsubseteq C$. We show that then there is an injective homomorphism h that maps the labelled tree T_C to the labeled tree $T_{C'}$ such that

- the root of T_C is mapped to the root of $T_{C'}$;
- $A \in l(d)$ if $A \in l(h(d))$, for all concept names A and all d in T_C.

We can regard $T_{C'}$ as an interpretation $\mathcal{I}_{C'}$. This interpretation satisfies C' in its root and so, since $\models C' \sqsubseteq C$, it satisfies C in its root. Thus, there is a homomorphism h from T_C to $T_{C'}$ mapping the root of T_C to the root of $T_{C'}$. h is injective since $A \sqsubseteq C$ is sibling and parent/child merged and $\mathcal{T} \models A \sqsubseteq C'$. Finally $A \in l(d)$ if $A \in h(l(d))$, for all concept names A and all d in T_C, follows from concept saturatedness for $\mathcal{T}$ of $A \sqsubseteq C$ and $\mathcal{T} \models A \sqsubseteq C'$. This finishes the proof of Claim 1.

Recall that n_C denotes the number of nodes in the tree representation of C and that $A^{\mathcal{T}} = \{B \mid A \sqsubseteq B \in \mathcal{H}_{basic}\} \cup \{D \mid A \equiv_{\mathcal{T}} B, B \sqsubseteq D \in \mathcal{T}\}$. We show the following

Claim 2. If $A \sqsubseteq C$ is $\mathcal{T}$-essential, then $n_C \leq \sum_{D \in A^{\mathcal{T}}} n_D$.

The proof of Claim 2 employs canonical models for DL-Lite TBoxes (Kontchakov et al. 2010). Assume $A \sqsubseteq C$ is $\mathcal{T}$-essential. We may assume that $\mathcal{T} = \mathcal{H}_{basic} \cup \mathcal{T}'$, where $\mathcal{T}'$ consists of role inclusions and concept inclusions in reduced form.

We define the canonical model $\mathcal{I}_{A,\mathcal{T}}$ as the union of a sequence of interpretations $\mathcal{I}_0, \mathcal{I}_1, \ldots$. Let $\mathcal{I}_0$ be the tree shaped interpretation defined by the labelled tree corresponding to $D_0 := \bigsqcap_{D \in A^{\mathcal{T}}} D$. Its root is denoted by ρ_A. Now assume $\mathcal{I}_n$ has been defined. To define $\mathcal{I}_{n+1}$,

- let $i \leq n$ be minimal such that there exist $d, d' \in r^{\mathcal{I}_i}$ with $(d, d') \in r^{\mathcal{I}_i}$ but $(d, d') \notin s^{\mathcal{I}_n}$ for some role s with

$\mathcal{T} \models r \sqsubseteq s$. Then define $\mathcal{I}_{n+1}$ in the same way as $\mathcal{I}_n$ except that $s^{\mathcal{I}_{n+1}} = s^{\mathcal{I}_n} \cup \{(d,d')\}$ if s is a role name and $s_0^{\mathcal{I}_{n+1}} = s_0^{\mathcal{I}_n} \cup \{(d',d)\}$ if $s = s_0^-$ for a role name s_0.

- Otherwise let $i \leq n$ be minimal such that there exists $\exists r.\top \sqsubseteq E \in \mathcal{H}_{basic}$ with E a concept name and $d \in (\exists r.\top)^{\mathcal{I}_i}$ with $d \notin E^{\mathcal{I}_n}$. Then define $\mathcal{I}_{n+1}$ in the same way as $\mathcal{I}_n$ except that $E^{\mathcal{I}_{n+1}} := E^{\mathcal{I}_n} \cup \{d\}$.
- Otherwise let $i \leq n$ be minimal such that there exists $E \sqsubseteq F \in \mathcal{T}$ with E a concept name and $d \in E^{\mathcal{I}_i}$ such that $d \notin F^{\mathcal{I}_n}$. Then add to $\mathcal{I}_n$ a copy of the tree shaped interpretation $\mathcal{I}_{d,F}$ defined by the labelled tree corresponding to F and identify d with the root of $\mathcal{I}_{d,F}$.
- Otherwise let $\mathcal{I}_{A,\mathcal{T}} := \mathcal{I}_n$.

If Point 4 never applies in the definition of the sequence $\mathcal{I}_0, \mathcal{I}_1, \ldots$, then let $\mathcal{I}_{A,\mathcal{T}} := \bigcup_{n\geq 0} \mathcal{I}_n$. The interpretation $\mathcal{I}_{A,\mathcal{T}}$ has the following properties:

1. $\mathcal{I}_{A,\mathcal{T}}$ is a model of $\mathcal{T}$;
2. for every $\mathcal{ELI}$ concept F: $\rho_A \in F^{\mathcal{I}_{A,\mathcal{T}}}$ iff $\mathcal{T} \models A \sqsubseteq F$;
3. for any $d \in \Delta^{\mathcal{I}_0}$ and $\mathcal{ELI}$ concept of the form $F = \exists r.F'$: if there exists a homomorphism h from the labelled tree corresponding to F into $\mathcal{I}_{\mathcal{A},\mathcal{T}}$ such that $h(a_F) = d$ and all nodes in the labelled tree corresponding to F' are mapped to $\Delta^{\mathcal{I}} \setminus \Delta^{\mathcal{I}_0}$, then there exists a concept name E with $d \in E^{\mathcal{I}}$ such that $\mathcal{T} \models E \sqsubseteq F$.

The only non-standard condition is Point 3 which can be proved using the construction of $\mathcal{I}_{A,\mathcal{T}}$ and the assumption that $\mathcal{T} = \mathcal{H}_{basic} \cup \mathcal{T}'$ for $\mathcal{T}'$ containing concept inclusions in reduced form only. We now show the following claim (from which Claim 2 follows immediately):

Claim 3. There is an injective homomorphism from the labelled tree corresponding to C into the restriction $\mathcal{J}$ of $\mathcal{I}_{A,\mathcal{T}}$ to $\Delta^{\mathcal{I}_0}$ which maps a_C to ρ_A.

By Point 2 above and since $\mathcal{T} \models A \sqsubseteq C$, there is a homomorphism h from the labelled tree T_C corresponding to C into $\mathcal{I}_{A,\mathcal{T}}$ which maps a_C to ρ_A. It follows from Point 2 again and the condition that $A \sqsubseteq C$ is role-saturated, sibling merged, and parent/child merged for $\mathcal{T}$ that h is injective. It thus remains to be proved that h is into $\mathcal{J}$ (rather than $\mathcal{I}_{A,\mathcal{T}}$). By concept saturatedness of $A \sqsubseteq C$ and Point 2, we have $(*)$: $h(d) \in E^{\mathcal{I}_{A,\mathcal{T}}}$ iff $E \in l(d)$ for every node d in T_C and every concept name E.

Assume now that $h(d') \notin \Delta^{\mathcal{I}_0}$ for some node d' in T_C. Assume that d' has minimal distance from d_C with this property - thus the parent d of d' in T_C is mapped to $\Delta^{\mathcal{I}_0}$. Let $l(d,d') = r$. We make a case distinction:

- the whole subtree generated by d' is mapped into $\Delta^{\mathcal{I}_{A,\mathcal{T}}} \setminus \Delta^{\mathcal{I}_0}$. Let $C' = \exists r.C''$, where C'' corresponds to the subtree generated by d' in C. By Point 3, there exists a concept name E with $h(d) \in E^{\mathcal{I}_{A,\mathcal{T}}}$ such that $\mathcal{T} \models E \sqsubseteq C'$. By $(*)$, $E \in l(d)$. We make a case distinction:
  - $d \neq \rho_A$. Then $A \sqsubseteq C$ is not minimal for $\mathcal{T}$ since C contains the edge (d,d') such that E is in the node label of d, $l(d,d') = r$, and $\mathcal{T} \models E \sqsubseteq \exists r.C''$. We have derived a contradiction.
  - $d = \rho_A$ and $A \not\equiv_{\mathcal{T}} E$. Then $A \sqsubseteq C$ is not minimal for $\mathcal{T}$ since C contains the edge (ρ_A, d'), E is in the node label of ρ_A, $E \not\equiv_{\mathcal{T}} A$, $l(d,d') = r$, and $\mathcal{T} \models E \sqsubseteq \exists r.C''$. We have derived a contradiction.
  - $d = \rho_A$ and no E from the node label of ρ_A that is non-$\mathcal{T}$-equivalent to A can be chosen such that $\mathcal{T} \models E \sqsubseteq \exists r.C''$. We show that then $d' \in \Delta^{\mathcal{I}_0}$ which is a contradiction and so our claim follows. Asume $d' \in \Delta^{\mathcal{I}_{A,\mathcal{T}}} \setminus \Delta^{\mathcal{I}_0}$. By construction of $\mathcal{I}_{A,\mathcal{T}}$, there exists an inclusion $E_0 \sqsubseteq D \in \mathcal{T}$ with E_0 a concept name such that $\rho_A \in E_0^{\mathcal{I}_{A,\mathcal{T}}}$ and d' is in the copy of the tree shaped interpretation $\mathcal{I}_{\rho_A,D}$ which was attached to ρ_A in the construction of $\mathcal{I}_{A,\mathcal{T}}$. Then $E_0 \not\equiv_{\mathcal{T}} A$ because otherwise $\mathcal{I}_{\rho_A,D}$ would be already part of $\mathcal{I}_0$. But that contradicts our assumption that no E from the node label of ρ_A that is non-$\mathcal{T}$-equivalent to A can be chosen such that $\mathcal{T} \models E \sqsubseteq \exists r.C''$.
- the subtree generated by d' is not mapped into $\Delta^{\mathcal{I}_{A,\mathcal{T}}} \setminus \Delta^{\mathcal{I}_0}$. Then h is not injective and we have again derived a contradiction.

Claim 1 and 2 together directly imply Lemma 4. ❑

Proofs for: Learning $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes in Polynomial Time

We begin with the proof that the interpretation $\mathcal{I}$ constructed in the proof sketch for Lemma 8 satisfies the conditions required in Lemma 8.

Lemma 8. Given a positive counterexample $C \sqsubseteq A$ for $\mathcal{T}$ relative to $\mathcal{H}$, one can construct an essential $\mathcal{T}$-countermodel $\mathcal{I}$ with $\mathcal{I} \models \mathcal{H}$ only polynomially many entailment queries in $|\mathcal{T}| + |C|$.

Proof. Let $C \sqsubseteq A$ be a positive counterexample for $\mathcal{T}$ relative to $\mathcal{H}$. First observe that $\mathcal{I} \not\models \mathcal{T}$: since $\mathcal{H} \not\models C \sqsubseteq A$, we know that A does not occur as a top-level conjunct in C. Consequently, $\rho_{\mathcal{I}_C} \in C^{\mathcal{I}_C} \setminus A^{\mathcal{I}_C}$ and thus $\mathcal{I}_C \not\models \mathcal{T}$.

Now we show that the interpretation $\mathcal{I}$ constructed in the proof sketch of Lemma 8 has the required properties:

- $\mathcal{I} \models \mathcal{H}$: clearly, the interpretation $\mathcal{J}$ constructed in Step 1 is a model of $\mathcal{H}$. Taking subtress and removing subtrees from $\mathcal{I}$ preserves being a model of $\mathcal{H}$, and so $\mathcal{I} \models \mathcal{H}$.
- $\mathcal{I} \not\models \mathcal{T}$: the interpretation $\mathcal{J}$ constructed in Step 1 is not a model of $\mathcal{T}$. In fact, we can use $C_{\mathcal{J}} \sqsubseteq A$ as a positive counterexample for $\mathcal{T}$ relative to $\mathcal{H}$ instead of $C \sqsubseteq A$. Observe that $\emptyset \models C_{\mathcal{J}} \sqsubseteq C$, and thus $\mathcal{T} \models C \sqsubseteq A$ implies $\mathcal{T} \models C_{\mathcal{J}} \sqsubseteq A$. On the other hand, $\rho_{\mathcal{J}} \in B^{\mathcal{J}}$ implies $\mathcal{H} \models C \sqsubseteq B$ for all concept names B. Consequently and since $\mathcal{H} \not\models C \sqsubseteq A$, we have $\rho_{\mathcal{J}} \notin A^{\mathcal{J}}$. Thus $\mathcal{J} \not\models \mathcal{T}$. Steps 2 and 3 preserve the condition that $\mathcal{J}$ is not a model of $\mathcal{T}$ and so $\mathcal{I} \not\models \mathcal{T}$.
- $\mathcal{I}$ satisfies Condition 1 for $\mathcal{T}$-essential models because of Step 2.
- $\mathcal{I}$ satisfies Condition 2 for $\mathcal{T}$-essential models because of Step 3.

❑

Lemma 11. Let $\mathfrak{I}$ be a sequence computed at some point of an execution of Algorithm 2. Then (i) the length of $\mathfrak{I}$ is bounded by the number of CIs in $\mathcal{T}$ and (ii) each interpretation in each position of $\mathfrak{I}$ is replaced only $|\mathcal{T}| + |\mathcal{T}|^2$ often with a new interpretation.

The rest of the section is devoted to proving Lemma 11. For easy reference, assume that at each point of the execution of the algorithm, $\mathfrak{I}$ has the form $\mathcal{I}_0, \dots, \mathcal{I}_k$ for some $k \geq 0$. To establish Point (i) above, we closely follow (Angluin, Frazier, and Pitt 1992) and show that

(iii) for every $\mathcal{I}_i$, there is a $D_i \sqsubseteq A_i \in \mathcal{T}$ with $\mathcal{I}_i \not\models D_i \sqsubseteq A_i$ and

(iv) if $i \neq j$, then $D_i \sqsubseteq A_i$ and $D_j \sqsubseteq A_j$ are not identical.

In fact, Point (iii) is immediate since whenever a new $\mathcal{I}_i$ is added to $\mathfrak{I}$ in the algorithm, then $\mathcal{I}_i$ is a $\mathcal{T}$-countermodel. To prove Point (iv), we first establish the intermediate Lemma 19 below. For a tree-shaped interpretation $\mathcal{I}$ and a concept inclusion $C \sqsubseteq A$, we write $\mathcal{I} \models^{\mathsf{r}} C \sqsubseteq A$ if $\rho_\mathcal{I} \notin C^\mathcal{I}$ or $\rho_\mathcal{I} \in A^\mathcal{I}$; that is, the inclusion $C \sqsubseteq A$ is satisfied at the root of $\mathcal{I}$, but not necessarily at other points in $\mathcal{I}$. It is easy to see that if some interpretation $\mathcal{I}$ is a $\mathcal{T}$-countermodel, then there is a $C \sqsubseteq A \in \mathcal{T}$ such that $\mathcal{I} \not\models^{\mathsf{r}} C \sqsubseteq A$.

Lemma 19 *If the interpretation $\mathcal{I}$ constructed in Line 5 of the algorithm satisfies $\mathcal{I} \not\models^{\mathsf{r}} C \sqsubseteq A \in \mathcal{T}$ and $\rho_{\mathcal{I}_j} \in C^{\mathcal{I}_j}$ for some j, then $\mathcal{J} = \mathcal{I}_i$ is replaced with $\mathcal{J}'$ in Line 9 for some $i \leq j$.*

Proof. Assume that the interpretation $\mathcal{I}$ constructed in Line 5 of the algorithm satisfies $\mathcal{I} \not\models^{\mathsf{r}} C \sqsubseteq A \in \mathcal{T}$ and that there is some j with $\rho_{\mathcal{I}_j} \in C^{\mathcal{I}_j}$. If there is some $i < j$ such that $\mathcal{I}_i \not\Rightarrow (\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_i)$ and $\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_i \not\models \mathcal{T}$, then $\mathcal{J} = \mathcal{I}_{i'}$ will be replaced with $\mathcal{J}'$ in Line 9 for some $i' \leq i$ and we are done. Thus assume that there is no such i. We aim to show that $\mathcal{J} = \mathcal{I}_j$ is replaced with $\mathcal{J}'$ in Line 9. To this end, it suffices to prove that $\mathcal{I}_j \not\Rightarrow (\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_j)$ and $\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_j \not\models \mathcal{T}$. The latter is a consequence of $\mathcal{I} \not\models^{\mathsf{r}} C \sqsubseteq A$ and $\rho_{\mathcal{I}_j} \in C^{\mathcal{I}_j}$.

Assume to the contrary of what we have to show that $\mathcal{I}_j \Rightarrow (\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_j)$. We establish a contradiction against $\mathcal{I} \models \mathcal{H}$ (which holds by construction of $\mathcal{I}$ in the algorithm) by showing that

1. $\mathcal{I} \not\models^{\mathsf{r}} C_{\mathcal{I}_j} \sqsubseteq A$ and
2. $C_{\mathcal{I}_j} \sqsubseteq A \in \mathcal{H}$.

For Point 1, $\mathcal{I}_j \Rightarrow (\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_j)$ and $\rho_{\mathcal{I}_j} \in (C_{\mathcal{I}_j})^{\mathcal{I}_j}$ imply $\rho_{\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_j} \in (C_{\mathcal{I}_j})^{\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_j}$, which gives $\rho_\mathcal{I} \in (C_{\mathcal{I}_j})^\mathcal{I}$, by Lemma 7 and Lemma 6. It remains to observe that $\mathcal{I} \not\models^{\mathsf{r}} C \sqsubseteq A$ implies $\rho_\mathcal{I} \notin A^\mathcal{I}$.

In view of the construction of $\mathcal{H}$ in the algorithm, Point 2 can be established by showing that $\mathcal{T} \models C_{\mathcal{I}_j} \sqsubseteq A$. Since $C \sqsubseteq A \in \mathcal{T}$, it suffices to prove that $C_{\mathcal{I}_j} \sqsubseteq C$. This, however, is an immediate consequence of the fact that $\rho_{\mathcal{I}_j} \in C^{\mathcal{I}_j}$ and the definition of $C_{\mathcal{I}_j}$. ❑

Now, Point (iv) above is a consequence of the following.

Lemma 20 *At any time of the algorithm execution, the following condition holds: if $\mathcal{I}_i \not\models^{\mathsf{r}} C \sqsubseteq A \in \mathcal{T}$ and $j < i$, then $\rho_{\mathcal{I}_j} \notin C^{\mathcal{I}_j}$.*

Proof. We prove the invariant formulated in Lemma 20 by induction on the number of iterations of the while loop. Clearly, the invariant is satisfied before the loop is entered. We now consider the two places where $\mathfrak{I}$ is modified, that is, Line 9 and Line 11, starting with the latter.

In Line 11, $\mathcal{I}$ is appended to $\mathfrak{I}$. Assume that $\mathcal{I} \not\models^{\mathsf{r}} C \sqsubseteq A \in \mathcal{T}$. We have to show that, before $\mathcal{I}$ was added to $\mathfrak{I}$, there was no $\mathcal{I}_i \in \mathfrak{I}$ with $\rho_{\mathcal{I}_i} \in C^{\mathcal{I}_j}$. This, however, is immediate by Lemma 19.

Now assume that $\mathcal{J}$ was replaced in Line 9 with $\mathcal{J}'$. We have to show two properties:

1. If $\mathcal{J}' = \mathcal{I}_i \not\models^{\mathsf{r}} C \sqsubseteq A \in \mathcal{T}$ and $j < i$, then $\rho_{\mathcal{I}_j} \notin C^{\mathcal{I}_j}$.

   Assume to the contrary that $\rho_{\mathcal{I}_j} \in C^{\mathcal{I}_j}$. Since $\mathcal{J}'$ is obtained from $\mathcal{I} \times \mathcal{J}$ by removing subtrees (see Lemma 9), $\mathcal{J}' \not\models^{\mathsf{r}} C \sqsubseteq A$ implies $\mathcal{I} \times \mathcal{J} \not\models^{\mathsf{r}} C \sqsubseteq A$. Consequently, $\mathcal{I} \not\models^{\mathsf{r}} C \sqsubseteq A$ or $\mathcal{J} \not\models^{\mathsf{r}} C \sqsubseteq A$. The former and $\rho_{\mathcal{I}_j} \in C^{\mathcal{I}_j}$ yields $i \leq j$ by Lemma 19, in contradiction to $j < i$. In the latter case, since $\mathcal{I}_i = \mathcal{J}$ before the replacement of $\mathcal{J}$ with $\mathcal{J}'$, we have a contradiction against the induction hypothesis.

2. If $\mathcal{J}' = \mathcal{I}_j$ and $\mathcal{I}_i \not\models^{\mathsf{r}} C \sqsubseteq A \in \mathcal{T}$ with $i > j$, then $\rho_{\mathcal{I}_j} \notin C^{\mathcal{I}_j}$.

   Assume to the contrary that $\rho_{\mathcal{I}_j} \in C^{\mathcal{I}_j}$. Since $\mathcal{J}'$ is obtained from $\mathcal{I} \times_{\mathsf{r}} \mathcal{J}$ by removing subtrees, we then have $\rho_{\mathcal{I} \times_{\mathsf{r}} \mathcal{J}} \in C^{\mathcal{I} \times_{\mathsf{r}} \mathcal{J}}$, thus $\rho_\mathcal{J} \in C^\mathcal{J}$. Since $\mathcal{I}_j = \mathcal{J}$ before the replacement of $\mathcal{J}$ with $\mathcal{J}'$, we have a contradiction against the induction hypothesis.

❑

We now turn towards proving Point (ii) above. It is a consequence of Lemma 22 below.

Lemma 21 *If $\mathcal{I}$ is an essential $\mathcal{T}$-countermodel, then $|\Delta^\mathcal{I}| \leq |\mathcal{T}|$.*

Proof. Let $\mathcal{I}$ be an essential $\mathcal{T}$-counttermodel. Then $\mathcal{I} \not\models \mathcal{T}$, but $\mathcal{I}|^-_{\mathsf{root}} \models \mathcal{T}$. It follows that there is a $C \sqsubseteq A \in \mathcal{T}$ such that $\rho_\mathcal{I} \in C^\mathcal{I} \setminus A^\mathcal{I}$. Consequently, there is a simulation $\sim$ from $(\mathcal{I}_C, \rho_{\mathcal{I}_C})$ to $(\mathcal{I}, \rho_\mathcal{I})$. Since $\mathcal{I}_C$ is tree-shaped, we can assume w.l.o.g. that $\sim$ is a total function (that is, a homomorphism). To show that $|\mathcal{I}| \leq |C|$ and thus $|\mathcal{I}| \leq |\mathcal{T}|$ as required, it clearly suffices to show that $\sim$ is surjective. Assume that this is not the case, and let $d \in \Delta^\mathcal{I}$ be outside the range of $\sim$ and $\mathcal{J} = \mathcal{I}|^-_{d\downarrow}$. All descendants of d must be outside the range of $\sim$ as well and thus $\sim$ is a simulation from $(\mathcal{I}_C, \rho_{\mathcal{I}_C})$ to $(\mathcal{J}, \rho_\mathcal{J})$. Therefore, $\rho^\mathcal{J} \in C^\mathcal{J}$, which implies $\mathcal{J} \not\models C \sqsubseteq A$, in contradiction to $\mathcal{I}$ being an essential $\mathcal{T}$-countermodel: Property 2 of being essential requires $\mathcal{J} \models \mathcal{T}$. ❑

Lemma 22 *Let $\mathcal{I}_0, \dots, \mathcal{I}_n$ be a sequence of interpretations such that $\mathcal{I}_{i+1}$ replaces $\mathcal{I}_i$ in Line 9 for all $i < n$. Then $n \leq |\mathcal{T}| + |\mathcal{T}|^2$.*

Proof. We first show that for every $i < n$ either

1. there is a concept name A such that $\rho_{\mathcal{I}_i} \in A^{\mathcal{I}_i}$ and $\rho_{\mathcal{I}_{i+1}} \notin A^{\mathcal{I}_{i+1}}$ or
2. $\mathcal{I}_{i+1} \Rightarrow \mathcal{I}_i$ via a surjective simulation.

For a proof by contradiction assume that there is $i < n$ such that neither Point 1 nor Point 2 holds. Since $\mathcal{I}_{i+1}$ is a subinterpretation of some $\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_i$ and $\mathcal{I} \times_{\mathsf{r}} \mathcal{I}_i \Rightarrow \mathcal{I}_i$ we obtain that $\mathcal{I}_{i+1} \Rightarrow \mathcal{I}_i$. Since $\mathcal{I}_{i+1}$ is an essential $\mathcal{T}$-countermodel, there is a $C \sqsubseteq A \in \mathcal{T}$ such that $\mathcal{I}_{i+1} \not\models^{\mathsf{r}} C \sqsubseteq A$. Let $\mathcal{J}$ be the subinterpretation of $\mathcal{I}_i$ determined by the range of the simulation $\sim$ from $\mathcal{I}_{i+1}$ to $\mathcal{I}_i$. Then $\rho_{\mathcal{J}} \in C^{\mathcal{J}}$ and so, since $\rho_{\mathcal{J}} \notin A^{\mathcal{J}}$ because Point 1 does not hold, $\mathcal{J} \not\models^{\mathsf{r}} C \sqsubseteq A$. $\mathcal{I}_i$ is an essential $\mathcal{T}$-countermodel and so $\mathcal{J} = \mathcal{I}$. But then $\sim$ is surjective and we have derived a contradiction.

In addition to the property stated above, we also have:

- for all concept names A, if $\rho_{\mathcal{I}_i} \in A^{\mathcal{I}_{i+1}}$, then $\rho_{\mathcal{I}_{i+1}} \in A^{\mathcal{I}_i}$;
- $\mathcal{I}_i \not\Rightarrow \mathcal{I}_{i+1}$.

By Lemma 21 we have $|\Delta^{\mathcal{I}_i}| \leq |\mathcal{T}|$ for all $i \leq n$. Thus, any convex subsequence $\mathcal{I}_j, \dots, \mathcal{I}_k$ of the sequence $\mathcal{I}_0, \dots, \mathcal{I}_n$ such that $\rho_{\mathcal{I}_i} \in A^{\mathcal{I}_i}$ iff $\rho_{\mathcal{I}_{i+1}} \in A^{\mathcal{I}_{i+1}}$ holds for all concept names A and all $j \leq i < k$ has length bounded by $|\mathcal{T}|^2$. ❑

Proofs for: Limits of Polynomial Time Learnability

To prove Lemma 14, we first show two technical lemmas. We also require the following lemma from (Konev et al. 2012) that characterizes concept inclusions entailed by acyclic $\mathcal{EL}$ TBoxes.

Lemma 23 *Let $\mathcal{T}$ be an acyclic $\mathcal{EL}$ TBox, r a role name and D an $\mathcal{EL}$ concept. Suppose that $\mathcal{T} \models \bigsqcap_{1 \leq i \leq n} A_i \sqcap \bigsqcap_{1 \leq j \leq m} \exists r_j.C_j \sqsubseteq D$, where A_i are concept names for $1 \leq i \leq n$, C_j are $\mathcal{EL}$ concepts for $1 \leq j \leq m$, and $m, n \geq 0$, then*

- *if D is a concept name such that $\mathcal{T}$ does not contain an inclusion $D \equiv C$, for some concept C, then there exists A_i, $1 \leq i \leq n$, such that $\mathcal{T} \models A_i \sqsubseteq D$;*
- *if D is of the form $\exists r.D'$ then either (i) there exists A_i, $1 \leq i \leq n$, such that $\mathcal{T} \models A_i \sqsubseteq \exists r.D'$ or (ii) there exists r_j, $1 \leq j \leq m$, such that $r_j = r$ and $\mathcal{T} \models C_j \sqsubseteq D'$.*

Lemma 24 *For any $0 \leq m \leq n$, any sequence of role names $\boldsymbol{\sigma} = \sigma^1 \dots \sigma^m$, any $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$ and any $\mathcal{EL}$ concept C over Σ_n: if $\mathcal{T}_L \models C \sqsubseteq \exists\boldsymbol{\sigma}.M$ then either*

1. *$m = n$, $\boldsymbol{\sigma} = \boldsymbol{\sigma}_i$, for some $1 \leq i \leq n$ and C is of the form $A \sqcap C'$, $A_i \sqcap C'$ or $B_i \sqcap C'$, for some $\mathcal{EL}$ concept C'; or*
2. *$\models C \sqsubseteq \exists\boldsymbol{\sigma}.M$.*

Proof. We prove the proposition by induction on m. If $m = 0$, by Lemma 23, C is of the form $Z \sqcap C'$, for some concept name Z and concept C', and $\mathcal{T}_L \models Z \sqsubseteq M$. This is only possible if Z is M itself, so $\models C \sqsubseteq M$.

Let $m > 0$. By Lemma 23 we have one of the following two cases:

- C is of the form $X \sqcap C'$, for some concept name X and concept C' such that $\mathcal{T}_L \models X \sqsubseteq \exists\boldsymbol{\sigma}.M$. It is easy to see that this is only possible if $m = n$, $\boldsymbol{\sigma} = \boldsymbol{\sigma}_i$ and X is one of A, A_i or B_i.
- C is of the form $\exists\sigma^1.C' \sqcap C''$ for some concepts C' and C'' such that $\mathcal{T}_L \models C' \sqsubseteq \exists\sigma^2.\cdots\exists\sigma^m.M$. By induction hypothesis, $\models C' \sqsubseteq \exists\sigma^2.\cdots\exists\sigma^m.M$. But then $\models C \sqsubseteq \exists\boldsymbol{\sigma}.M$.

❑

Finally, we require the following observation.

Lemma 25 *For any acyclic $\mathcal{EL}$ TBox $\mathcal{T}$, any inclusion $A \sqsubseteq C \in \mathcal{T}$ and any concept of the form $\exists t.D$ we have $\mathcal{T} \models A \sqsubseteq \exists t.D$ if, and only if, $\mathcal{T} \models C \sqsubseteq \exists t.D$.*

We are now ready to prove Lemma 14.

Lemma 14 *For every $\mathcal{EL}$ concept inclusion $C \sqsubseteq D$ over Σ_n:*

- *either for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D$ or*
- *the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C \sqsubseteq D$ does not exceed $|C|$.*

Proof. We prove the lemma by induction on the size of D. We assume throughout the proof that there exists some $L_0 \in \mathfrak{L}_n$ such that $\mathcal{T}_{L_0} \models C \sqsubseteq D$.

Base case: D is a concept name. We make the following case distinction.

- D is one of X_i, A_i, B_i or M, for $i \geq 1$. By Lemma 23, C is of the form $Z \sqcap C'$, for some concept name Z, and $\mathcal{T}_{L_0} \models Z \sqsubseteq D$. If D is one of X_i, A_i, B_i or M, then this can only be the case if $Z = D$. But then for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D$.
- $D = X_0$. By Lemma 23, C is of the form $Z \sqcap C'$, for some concept name Z, and $\mathcal{T}_{L_0} \models Z \sqsubseteq X_0$. This is the case if either $Z = X_0$, or Z is one of $A, A_1, B_1, \dots, A_n, B_n$. In either case, for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq X_0$.
- $D = A$. If C is form $A \sqcap C'$ or for for all i such that $1 \leq i \leq n$, A_i or B_i is a conjunct of C, then for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq A$. Assume now that C is not of this form. Then for some j such that $1 \leq j \leq n$, C is neither of the form $A \sqcap C'$ nor of the form $A_j \sqcap C'$ nor of the form $B_j \sqcap C'$. Let $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$ be such that $\mathcal{T}_L \models C \sqsubseteq A$. Notice that $\mathcal{T}_L \models C \sqsubseteq A$, for $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$, if, and only if, $\mathcal{T}_L \models C \sqsubseteq X_0 \sqcap \exists\boldsymbol{\sigma}_1.M \sqcap \cdots \sqcap \exists\boldsymbol{\sigma}_n.M$. By Lemma 24, for such a $\mathcal{T}_L$ we must have $\models C \sqsubseteq \exists\boldsymbol{\sigma}_j.M$. Clearly, the number of different $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$ with $\models C \sqsubseteq \exists\boldsymbol{\sigma}_j.M$ does not exceed $|C|$.

  Thus, either for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq A$ or the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C \sqsubseteq A$ does not exceed $|C|$.

Induction step. If $D = D_1 \sqcap D_2$, then $\mathcal{T}_L \models C \sqsubseteq D$ if, and only if, $\mathcal{T} \models C \sqsubseteq D_i$, $i = 1, 2$. By induction hypothesis, for $i = 1, 2$ either for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D_i$, or there exist at most $|C|$ different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C \sqsubseteq D_i$. Thus either for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D$, or the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C \sqsubseteq D$ also does not exceed $|C|$.

Let $D = \exists t.D'$. Suppose that for some $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D$. Then, by Lemma 23, either there exists a conjunct Z of C, Z a concept name, such that $\mathcal{T}_L \models Z \sqsubseteq \exists t.D'$ or there exists a conjunct $\exists t.C'$ of C with $\mathcal{T}_L \models C' \sqsubseteq D'$. We analyse for every conjunct of C of the form Z or $\exists t.C'$ for how many different $L \in \mathfrak{L}_n$ it holds that $\mathcal{T}_L \models Z \sqsubseteq \exists t.D'$ (or $\mathcal{T}_L \models \exists t.C' \sqsubseteq \exists t.D'$ respectively).

(i) Let Z be a conjunct of C such that Z is a concept name and $\mathcal{T}_L \models Z \sqsubseteq \exists t.D'$. Notice that Z cannot be M as for no $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models M \sqsubseteq \exists t.D'$. Consider the remaining possibilities.

- Z is one of X_i, $i \geq 0$. It is easy to see that for $L, L' \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models X_i \sqsubseteq \exists t.D'$ if, and only if $\mathcal{T}_{L'} \models X_i \sqsubseteq \exists t.D'$. Thus, for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models Z \sqsubseteq \exists t.D'$.
- Z is one of A_i, B_i for $i \geq 1$. By Lemma 25, $\mathcal{T}_L \models Z \sqsubseteq \exists t.D'$ if, and only if, $\mathcal{T}_L \models X_0 \sqcap \exists \boldsymbol{\sigma}_i.M \sqsubseteq \exists t.D'$. By Lemma 23, either $\mathcal{T}_L \models X_0 \sqsubseteq \exists t.D'$ or $\mathcal{T}_L \models \exists \boldsymbol{\sigma}_i.M \sqsubseteq \exists t.D'$. If $\mathcal{T}_L \models X_0 \sqsubseteq \exists t.D'$ then, as above, for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq \exists t.D'$. Suppose that $\exists t.D'$ is such that $\mathcal{T}_L \not\models X_0 \sqsubseteq \exists t.D'$ and $\mathcal{T}_L \models \exists \boldsymbol{\sigma}_i.M \sqsubseteq \exists t.D'$. By inductive applications of Lamma 23, this is only possible when $\exists t.D'$ is $\exists \boldsymbol{\sigma}_i.M$. Notice that all $\boldsymbol{\sigma}_i$ are unique so there exists exactly one $L \in \mathfrak{L}_n$ (namely, L is L_0) such that $\mathcal{T}_L \models Z \sqsubseteq \exists \boldsymbol{\sigma}_i.M$.
- Z is A. Suppose that for some $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models A \sqsubseteq \exists t.D'$, equivalently $\mathcal{T}_L \models X_0 \sqcap \exists \boldsymbol{\sigma}_1.M \sqcap \dots \boldsymbol{\sigma}_n.M \sqsubseteq \exists t.D'$. By Lemma 23, either $\mathcal{T}_L \models X_0 \sqsubseteq \exists t.D'$ or $\mathcal{T}_L \models \exists \boldsymbol{\sigma}_i.M \sqsubseteq \exists t.D'$, for some $i : 1 \leq i \leq n$, so, as above, unless $\mathcal{T}_L \models X_0 \sqsubseteq \exists t.D'$ we have $\exists t.D'$ is $\exists \boldsymbol{\sigma}_i.M$. But then $L = L_0$.

(ii) Let $\exists t.C'$ be a conjunct of C with $\mathcal{T}_L \models C' \sqsubseteq D'$. The induction hypothesis implies that the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C' \sqsubseteq D'$ does not exceed $|C'|$.

To summarize, either $\mathcal{T}_L \models C \sqsubseteq \exists t.D'$ for every $L \in \mathfrak{L}_n$ or for every conjunct C_0 of C of the form Z or $\exists t.C'$, the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C_0 \sqsubseteq \exists t.D'$ does not exceed $|C_0|$. Hence the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C \sqsubseteq \exists t.D'$ does not exceed $|C|$. ❑

The next lemma prepares the proof of Lemma 15.

Lemma 26 *For any $0 \leq i \leq n$ and Σ_n-concept D, if $\mathcal{T}_0 \not\models X_i \sqsubseteq D$ then there exists a sequence of role names $t_1, \dots t_l$ such that $\models D \sqsubseteq \exists t_1. \cdots \exists t_l.Y$ and $\mathcal{T}_0 \not\models X_i \sqsubseteq \exists t_1. \cdots \exists t_l.Y$, where Y is either $\top$ or a concept name, $0 \leq l \leq n - i + 1$.*

Proof. We prove the lemma by induction on i. The base case is $i = n$. Then $\mathcal{T}_0 \not\models X_n \sqsubseteq D$ either if $\models D \sqsubseteq \exists t.\top$, for some role name t, or $\models D \sqsubseteq Y$, for some concept name $Y \neq X_n$.

Suppose that the proposition is proved for $0 < j \leq n$ and let $i = j - 1$. We proceed by induction on the structure of D. If D is a concept name, we are done as for no concept name $Z \neq X_i$ we have $\mathcal{T}_0 \models X_i \sqsubseteq Z$. If D is of the form $\exists t.D'$, where t is one of r, s, we, obviously, have $\mathcal{T}_0 \not\models X_{i+1} \sqsubseteq D'$, and so, by induction hypothesis, there exists a sequence of role names $t_1, \dots, t_l$, with $l \leq n - i$, such that $\mathcal{T}_0 \not\models X_{i+1} \sqsubseteq \exists t_1. \cdots \exists t_l.Y$ and $\models D' \sqsubseteq \exists t_1. \cdots \exists t_l.Y$. But then, by Lemma 25 and Lemma 23, $\mathcal{T}_0 \not\models X_i \sqsubseteq \exists t.\exists t_1. \cdots \exists t_l.Y$ and $\models \exists t.D' \sqsubseteq \exists t.\exists t_1. \cdots \exists t_l.Y$. If D is of the form $D = D_1 \sqcap D_2$, for one of D_i, $i = 1, 2$, we have $\mathcal{T}_0 \not\models X_i \sqsubseteq D_i$ and the proposition holds by induction. ❑

Lemma 15 *For any $n > 1$ and any $\mathcal{EL}$ TBox $\mathcal{H}$ in Σ_n with $|\mathcal{H}| < 2^n$ there exists a Σ_n-concept inclusion $C \sqsubseteq D$ such that (i) the size of $C \sqsubseteq D$ does not exceed $6n$ and (ii) if $\mathcal{H} \models C \sqsubseteq D$ then $\mathcal{T}_L \models C \sqsubseteq D$ for at most one $L \in \mathfrak{L}_n$ and if $\mathcal{H} \not\models C \sqsubseteq D$ then for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C \sqsubseteq D$.*

Proof. We define an exponentially large TBox $\mathcal{T}_\cap$ and use it to prove that one can select the required $\mathcal{EL}$ inclusion $C \sqsubseteq D$ is such a way that either $\mathcal{H} \models C \sqsubseteq D$ and $\mathcal{T}_\cap \not\models C \sqsubseteq D$, or vice versa.

To define $\mathcal{T}_\cap$, for any sequence $\boldsymbol{b} = b_1 \dots b_n$, where every b_i is either 0 or 1, we denote by $C_{\boldsymbol{b}}$ the conjunction $\bigsqcap_{i \leq n} C_i$, where $C_i = A_i$ if $b_i = 1$ and $C_i = B_i$ if $b_i = 0$. Then we define

$$\mathcal{T}_\cap = \mathcal{T}_0 \cup \{C_{\boldsymbol{b}} \sqsubseteq A \sqcap X_0 \mid \boldsymbol{b} \in \{0, 1\}^n\}.$$

Consider possibilities for $\mathcal{H}$ and $\mathcal{T}_\cap$.

(1) If $\mathcal{H} \not\models \mathcal{T}_\cap$ then there exists an inclusion $C \sqsubseteq D \in \mathcal{T}_\cap$ such that $\mathcal{H} \not\models C \sqsubseteq D$. Clearly, $C \sqsubseteq D$ is entailed by every $\mathcal{T}_L$, for $L \in \mathfrak{L}_n$, and the size of $C \sqsubseteq D$ does not exceed $6n$, so $C \sqsubseteq D$ is as required.

(2) Suppose that for some $\boldsymbol{b} \in \{0, 1\}^n$ and a concept of the form $\exists t.D'$ we have $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq \exists t.D'$ and $\mathcal{T}_\cap \not\models C_{\boldsymbol{b}} \sqsubseteq \exists t.D'$. To 'minimise' $C_{\boldsymbol{b}} \sqsubseteq \exists t.D'$, notice that $\mathcal{T}_0 \not\models X_0 \sqsubseteq \exists t.D'$. Then, by Lemma 26, there exists a sequence of role names $t_1, \dots, t_l$, for $0 \leq l \leq n + 1$ and Y being $\top$ or a concept name such that $\models \exists t.D' \sqsubseteq \exists t_1. \cdots \exists t_l.Y$, so $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq \exists t_1. \cdots \exists t_l.Y$, and $\mathcal{T}_0 \not\models X_0 \sqsubseteq \exists t_1. \cdots \exists t_l.Y$. Clearly, the size of $C_{\boldsymbol{b}} \sqsubseteq \exists t_1. \cdots \exists t_l.Y$ does not exceed $6n$. It remains to prove that $\mathcal{T}_L \models C_{\boldsymbol{b}} \sqsubseteq \exists t_1 \cdots \exists t_l.Y$ for at most one $L \in \mathfrak{L}_n$.

Suppose for some $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C_{\boldsymbol{b}} \sqsubseteq \exists t_1. \cdots \exists t_l.Y$. By Lemma 23, there exists A_j or B_j such that $\mathcal{T}_L \models A_j \sqsubseteq \exists t_1. \cdots \exists t_l.Y$ (or $\mathcal{T}_L \models B_j \sqsubseteq \exists t_1. \cdots \exists t_l.Y$, respectively). As $\mathcal{T}_0 \not\models X_0 \sqsubseteq \exists t_1. \cdots \exists t_l.Y$ it is easy to see that this is only possible when $l = n$, $(t_1, t_2, \dots, t_n) = \boldsymbol{\sigma}_j$, and Y is M. Since every $\boldsymbol{\sigma}_j$ is unique, for every $L' \in \mathfrak{L}_n$ such that $L' \neq L$ we have $\mathcal{T}_{L'} \not\models C_{\boldsymbol{b}} \sqsubseteq \exists \boldsymbol{\sigma}_j.M$.

Thus, $C_{\boldsymbol{b}} \sqsubseteq \exists t_1. \cdots \exists t_l.Y$ is as required.

(3) Finally, suppose that Case 1 and 2 above do not apply. Then $\mathcal{H} \models \mathcal{T}_\cap$ and for every $\boldsymbol{b} \in \{0, 1\}^n$ and every $\mathcal{EL}$ concept over Σ_n of the form $\exists t.D'$: if $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq \exists t.D'$

then $\mathcal{T}_0 \models X_0 \sqsubseteq \exists t.D'$. We show that unless there exists an inclusion $C \sqsubseteq D$ satisfying the conditions of the lemma, $\mathcal{H}$ contains at least 2^n different inclusions. Thus, we have derived a contradiction.

Fix $\boldsymbol{b} \in \{0,1\}^n$. As $\mathcal{H} \models \mathcal{T}_\cap$ we have $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq A$. Then there must exist an (at least one) inclusion $C \sqsubseteq A \sqcap D \in \mathcal{H}$ such that $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq C$ and $\not\models C \sqsubseteq A$. Let $C = Z_1 \sqcap \cdots \sqcap Z_m \sqcap \exists t_1.C_1' \sqcap \cdots \sqcap \exists t_l.C_l'$, where $Z_1, \ldots, Z_m$ are different concept names. As $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq \exists t_j.C_j'$ we have $\mathcal{T}_0 \models X_0 \sqsubseteq \exists t_j.C_j'$, for $j = 1, \ldots l$. As $\mathcal{H} \models \mathcal{T}_\cap$ we have $\mathcal{H} \models X_0 \sqsubseteq \exists t_j.C_j'$, for $j = 1, \ldots l$. So $\mathcal{H} \models Z_1 \sqcap \cdots \sqcap Z_m \sqcap X_0 \sqsubseteq A$.

Suppose that for some $i : 1 \leq i \leq n$ there exists no $j : 1 \leq j \leq m$ such that Z_j is either A_i or B_i. Then we have $\mathcal{T}_L \not\models Z_1 \sqcap \cdots \sqcap Z_m \sqcap X_0 \sqsubseteq A$, for any $L \in \mathfrak{L}_n$. Notice that in the worst case $Z_1 \sqcap \cdots \sqcap Z_m$ contains the conjunction of all Σ_n-concept names, except A_i, B_i, so the size of $Z_1 \sqcap \cdots \sqcap Z_m \sqcap X_0 \sqsubseteq A$ does not exceed $6n$, and $Z_1 \sqcap \cdots \sqcap Z_m \sqcap X_0 \sqsubseteq A$ is as required.

Assume that $Z_0 \sqcap \cdots \sqcap Z_m \sqcap X_0$ contains a conjunct B_i such that $b_i \neq 0$. Then $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq B_i$ and for no $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C_{\boldsymbol{b}} \sqsubseteq B_i$. The size of $C_{\boldsymbol{b}} \sqsubseteq B_i$ does not exceed $6n$, so it is as required.

Assume that $Z_0 \sqcap \cdots \sqcap Z_m \sqcap X_0$ contains a conjunct A_i such that $b_i \neq 1$. Then $\mathcal{H} \models C_{\boldsymbol{b}} \sqsubseteq A_i$ and for no $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L \models C_{\boldsymbol{b}} \sqsubseteq A_i$. The size of $C_{\boldsymbol{b}} \sqsubseteq A_i$ does not exceed $6n$, so it is as required.

The only remaining option is that $Z_1 \sqcap \cdots \sqcap Z_m \sqcap X_0$ contains exactly the A_i with $b_i = 1$ and exactly the B_i with $b_i = 0$.

This argument applies to arbitrary $\boldsymbol{b} \in \{0,1\}^n$. Thus if there exists no inclusion $C \sqsubseteq D$ satisfying the conditions of the lemma then $\mathcal{H}$ contains at least 2^n inclusions. ❑

We come to the proof of Theorem 13. Theorem 13 follows immediately with Angluin's strategy from the following lemma. Let $\Gamma_n = \{r, s, A, M, X_0, \ldots, X_n\}$.

Lemma 27 *For every DL-Lite$_{\mathcal{R}}^{\exists}$ concept inclusion $B \sqsubseteq D$ over Γ_n:*

- *either for every $\mathcal{T}_{\boldsymbol{\sigma}} \in S$ we have $\mathcal{T}_{\boldsymbol{\sigma}} \models B \sqsubseteq D$;*
- *or there is at most one $\mathcal{T}_{\boldsymbol{\sigma}} \in S$ such that $\mathcal{T}_{\boldsymbol{\sigma}} \models B \sqsubseteq D$.*

Proof. Assume $B \sqsubseteq D$ is given. If $B \neq A$ or M does not occur in D, then the claim can be readily checked. Thus, we assume that $B = A$ and M occurs in D. Assume there exists $\boldsymbol{\sigma}_0$ such that $\mathcal{T}_{\boldsymbol{\sigma}_0} \models A \sqsubseteq D$ (if no such $\boldsymbol{\sigma}_0$ exists, we are done). For any $\boldsymbol{\sigma}$, let $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}}}$ be the canonical model constructed in the same way as in the proof of Lemma 4. Note that $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}}}$ is a tree interpretation with root ρ_A. Observe that the following conditions are equivalent for every $\boldsymbol{\sigma}$:

- $\mathcal{T}_{\boldsymbol{\sigma}} \models A \sqsubseteq D$;
- $\rho_A \in D^{\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}}}}$;
- there is a homomorphism from the interpretation $\mathcal{I}_D$ corresponding to D to $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}}}$ which maps the root d_D of $\mathcal{I}_D$ to ρ_A.

We consider a restricted form of parent-child merging for D. Apply the following merging operation exhaustively to D:

- if there are nodes $d, d_1, d_2 \in T_D$ with $l(d_1, d) = \sigma$ and $l(d, d_2) = \sigma^-$ for some $\sigma \in \{r, s\}$, then replace D by the resulting concept after d_1 and d_2 are merged in D.

Let D' be the resulting concept. It is readily checked that any homomorphism h from $\mathcal{I}_D$ to $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}}}$ which maps the root of $\mathcal{I}_D$ to ρ_A factors through $\mathcal{I}_{D'}$ and that D' is an $\mathcal{EL}$ concept. Thus, if there is an additional $\boldsymbol{\sigma}' \neq \boldsymbol{\sigma}_0$ such that $\mathcal{T}_{\boldsymbol{\sigma}'} \models A \sqsubseteq D$, then there are two homomorphisms $h_{\boldsymbol{\sigma}_0}$ and $h_{\boldsymbol{\sigma}'}$ from $\mathcal{I}_{D'}$ to $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}_0}}$ and from $\mathcal{I}_{D'}$ to $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}'}}$, mapping the root of $\mathcal{I}_{D'}$ to the roots of $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}_0}}$ and $\mathcal{I}_{A,\mathcal{T}_{\boldsymbol{\sigma}'}}$, respectively. Moreover, M occurs in D'. Since D' is a $\mathcal{EL}$-concept we find a sequence $D' = D_0, \ldots, D_m$ such that $\exists s_{i+1}.D_{i+1}$ is a top-level conjunct of D_i for $s_i \in \{r, s\}$ and all $i < m$. But then $s_1 \cdots s_m = \boldsymbol{\sigma}_0$ and $s_1 \cdots s_m = \boldsymbol{\sigma}'$ and we have derived a contradiction to the assumption that $\boldsymbol{\sigma}_0$ and $\boldsymbol{\sigma}'$ are distinct. ❑

We come to the proof of Theorem 18.

Theorem 18 The class of DL-Lite$_{\mathcal{R}}^{\exists}$ TBoxes is not polynomial time learnable using equivalence queries.

In the paper we have designed a strategy for the oracle with the following property: if all TBoxes $\mathcal{H}$ in equivalence queries represent monotone DNF formulas in n variables, then the learner cannot distinguish all TBoxes in $T(n, s, t)$ in polynomial time (for sufficiently large n). We now extend this strategy to cover TBoxes in equivalence queries that do not represent monotone DNF formulas.

We say that a TBox $\mathcal{T}$ *has a DNF-representation for* n if it is obtained by the translation of a monotone DNF-formula with n variables; that is, if $\mathcal{T}$ is of the following form, for some $\Gamma \subseteq \{r, \bar{r}\}^n$:

$$\{A \sqsubseteq \bigsqcap_{\rho_1 \cdots \rho_n \in \Gamma} \exists \rho_1.\exists \rho_2.\ldots \exists \rho_n.\top, \quad \bar{r} \sqsubseteq r\}.$$

Observe that the TBoxes in $T(n, s, t)$ are exactly those TBoxes that have a DNF-representation for n and satisfy additionally the conditions that the DNF represented by $\mathcal{T}_\phi$ has exactly t conjunctions each conjunction of which has exactly s variables.

We describe the strategy for the oracle $\mathfrak{O}'$ to answer equivalence queries so that no learning algorithm is able to exactly identify members of $T(n, s, t)$ based on the answers to polynomially many equivalence queries of polynomial size. If the TBox in the equivalence query is 'obviously' not within the class $T(n, t, s)$, then we will explicitly give the counterexample that the oracle returns. If, on the other hand, the TBox $\mathcal{H}$ from the equivalence query is 'similar' to TBoxes that have a DNF-representation for n, then we approximate $\mathcal{H}$ by a TBox $\mathcal{H}'$ that has a DNF-representation for n and return the counterexample $A \sqsubseteq C_I$ corresponding to the truth assignment I that the oracle $\mathfrak{O}$ from Theorem 17 would return when given ψ.

In detail the strategy is as follows. Assume q is the given polynomial in Theorem 17 and that t_0, s_0 and the strategy of the oracle $\mathfrak{O}$ are chosen so that for sufficiently large n no learning algorithm for DNF formulas that asks at most $q(n)$

equivalence queries, each bounded in size by $q(n)$, can distinguish all members of $M(n, t_0, s_0)$. Choose a sufficiently large n. Let $\mathcal{H}$ be an equivalence TBox query issued by a learning algorithm. Then $\mathfrak{O}'$ does the following:

1. If $\mathcal{H}$ entails some $A \sqsubseteq \exists\rho_1.\exists\rho_2.\dots\exists\rho_{n+1}.\top$ with $\rho_i \in \{r, \bar{r}\}$ for $1 \leq i \leq n+1$, then return this inclusion as a negative counterexample;
2. If $\mathcal{H}$ entails some $\exists\rho_1.\top \sqsubseteq \exists\rho_2.\top$ such that $\{\rho_1, \rho_2\} \subseteq \{r, \bar{r}, r^-, \bar{r}^-\}$ and $\{\bar{r} \sqsubseteq r\} \not\models \exists\rho_1.\top \sqsubseteq \exists\rho_2.\top$, then return this inclusion as a negative counterexample;
3. If $\mathcal{H} \models \exists\rho_1.\top \sqsubseteq \exists\rho_2.\exists\rho_3.\top$ such that $\{\rho_1, \rho_2, \rho_3\} \subseteq \{r, \bar{r}\}$, then return this inclusion as a negative counterexample;
4. If there exists no $\rho_1, \dots, \rho_n \in \{r, \bar{r}\}^n$ such that $\mathcal{H} \models A \sqsubseteq \exists\rho_1.\cdots\exists\rho_n.\top$ then return $A \sqsubseteq \underbrace{\exists r \cdots \exists r}_{n}.\top$ as a positive counterexample.
5. Suppose now that none of the above applies. We say that a sequence $\rho_1, \dots, \rho_n \in \{r, \bar{r}\}^n$ is *r-minimal for $\mathcal{H}$* if $\mathcal{H} \models A \sqsubseteq \exists\rho_1.\cdots\exists\rho_n.\top$ and whenever $\rho_i = r$, for $1 \leq i \leq n$, we have $\mathcal{H} \not\models \exists\rho_1.\cdots\exists\rho_{i-1}.\exists\bar{r}.\exists\rho_{i+1}.\cdots\exists\rho_n.\top$. We obtain a TBox $\mathcal{H}'$ with a DNF representation by setting

$$\mathcal{H}' = \{A \sqsubseteq \bigsqcap_{\substack{\rho_1,\dots,\rho_n \text{ is} \\ r\text{-minimal for } \mathcal{H}}} \exists\rho_1.\cdots\exists\rho_n.\top, \quad \bar{r} \sqsubseteq r\}.$$

   Observe that for any sequence $\rho_1, \dots, \rho_n \in \{r, \bar{r}\}^n$ we have $\mathcal{H} \models A \sqsubseteq \exists\rho_1.\cdots\exists\rho_n.\top$ iff $\mathcal{H}' \models A \sqsubseteq \exists\rho_1.\cdots\exists\rho_n.\top$. We convert $\mathcal{H}'$ into its corresponding monotone DNF formula $\phi_{\mathcal{H}'}$ by reversing the translation from monotone DNF formulas into DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes of the above form, in the obvious way. Note that the size of $\phi_{\mathcal{H}'}$ is linear in the size of $\mathcal{H}'$. Given $\phi_{\mathcal{H}'}$ the oracle $\mathfrak{O}$ returns a (positive or negative) counterexample (a truth assignment) I. Then return the counterexample in the in the form of the CI $A \sqsubseteq C_I$.

Observe that the answers given in Points 1 to 3 are correct in the sense that if an inclusion α is returned as a negative example then $\mathcal{T} \not\models \alpha$ for any $\mathcal{T} \in T(n, t, s)$. Point 4 is trivially correct, since any monotone DNF is satisfied by the truth assignment that makes every variable true. We first analyse the size of the TBox $\mathcal{H}'$ computed in Point 5.

Lemma 28 *Assume that Points 1 to 4 do not apply to $\mathcal{H}$. Then the number of sequences $\rho_1, \dots, \rho_n \in \{r, \bar{r}\}^n$ which are r-minimal for $\mathcal{H}$ is bounded by $|\mathcal{H}|$.*

Proof. We first show that if $\rho_1, \dots, \rho_n \in \{r, \bar{r}\}^n$ is r-minimal for $\mathcal{H}$, then there exists an inclusion $A \sqsubseteq C \in \mathcal{H}$ such that

($*$) there are concepts $C = C_0, \dots, C_n$ with $\exists\rho_{i+1}.C_{i+1}$ a top-level conjunct of C_i, for $i < n$.

For the proof we require the canonical model construction given in the proof of Lemma 4. Here we start with the interpretation $\mathcal{I}_0$ which has domain $\{\rho_A\}$ and in which $A^{\mathcal{I}_0} = \{\rho_A\}$ and $X^{\mathcal{I}_0} = \emptyset$ for all symbols $X \neq A$. Then one applies the Expansion Rules 1-3 to construct $\mathcal{I}_{A,\mathcal{H}}$ in exactly the same way as in the proof of Lemma 4. Let $\rho_1, \dots, \rho_n \in \{r, \bar{r}\}^n$ be r-minimal for $\mathcal{H}$. Then there are $\rho_A = d_0, \dots, d_n$ in $\Delta^{\mathcal{I}_{A,\mathcal{H}}}$ such that $(d_i, d_{i+1}) \in \rho_i^{\mathcal{I}_{A,\mathcal{H}}}$ for all $i < n$.

Claim 1. For all $i > 0$: $d_i \notin A^{\mathcal{I}_{A,\mathcal{H}}}$.

Proof of Claim 1. Assume $d_i \in A^{\mathcal{I}_{A,\mathcal{H}}}$. Then $\mathcal{H} \models A \sqsubseteq \exists\rho_1 \cdots \exists\rho_i.A$. But then $\mathcal{H} \models A \sqsubseteq \exists(\rho_1 \cdots \rho_i)^n.\top$ for all $n > 0$ which contradicts the assumption that Point 1 does not apply to $\mathcal{H}$.

Using the fact Points 2 to 3 do not apply to $\mathcal{H}$, it follows that the only possible reason for having a sequence $\rho_0 = d_0, \dots, d_n$ in $\Delta^{\mathcal{I}_{A,\mathcal{H}}}$ such that $(d_i, d_{i+1}) \in \rho_i^{\mathcal{I}_{A,\mathcal{H}}}$ with $\rho_i \in \{r, \bar{r}\}$ for $i < n$ in $\mathcal{I}_{A,\mathcal{H}}$ is that there is an $A \sqsubseteq C \in \mathcal{H}$ such that ($*$) holds.

It follows that the number of distinct r-minimal sequences is bounded by the number of distinct sequences $C = C_0, \dots, C_n$ with $A \sqsubseteq C \in \mathcal{H}$ and $\exists\rho_{i+1}.C_{i+1}$ a top-level conjunct of C_i for all $i < n$. Thus, the number of distinct r-minimal sequences is bounded by $|\mathcal{H}|$. ❑

It follows from Lemma 28 that the size of the TBox $\mathcal{H}'$ computed in Point 5 is bounded by $4n|\mathcal{H}| + 2$.

We are now in the position to prove the main result in general form.

Proof of Theorem 18. Suppose that the running time of a learning algorithm $\mathfrak{A}$ for DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes in $\Sigma = \{A, r, \bar{r}\}$ is bounded at every stage of computation by a polynomial $p(x, y)$, where x is the size of the target TBox, and y is the maximal size of a counterexample returned by the oracle up to the current stage of computation. Let $q(n) = (p(n^2, 4n+6))^2$, and let constants t_0 and s_0 be as guaranteed by Lemma 17. We claim that, for sufficiently large n, $\mathfrak{A}$ cannot distinguish some $\mathcal{T}_\phi$ and $\mathcal{T}_\psi$ for $\phi, \psi \in M(n, t_0, s_0)$.

Assuming that $n > 11$ (the maximal size of counterexamples given under Point 2 and 3), the largest counterexample returned by our strategy described above is of the form $A \sqsubseteq \exists\rho_1.\cdots\exists\rho_{n+1}.\top$, so for sufficiently large n the maximal size of any counterexample in any run of $\mathfrak{A}$ is bounded by $4n + 6 = 4(n+1) + 2$. Similarly, the size of every potential target TBox $\mathcal{T}_\phi \in T(n, t_0, s_0)$ does not exceed $t_0 \cdot (4n + 2)$ and, as t_0 is a constant, for sufficiently large n it is bounded by n^2. Thus, for sufficiently large n the total running time of $\mathfrak{A}$ on any input from $T(n, t_0, s_0)$ is bounded by $p(n^2, 4n+6)$. Obviously, the size of each query is bounded by the running time of the learning algorithm. So, the size of a DNF equivalence query forwarded to the strategy $\mathfrak{O}$ guaranteed by Lemma 17 is bounded by $4n \times p(n^2, 4n+6) + 2 \leq q(n)$, and there will be at most $q(n)$ queries forwarded. But then $\mathfrak{O}$ can return answers such that some ϕ and ψ from $M(n, t_0, s_0)$ cannot be distinguished. It remains to observe that $\mathfrak{A}$ cannot distinguish $\mathcal{T}_\phi$ and $\mathcal{T}_\psi$.