Exact Learning Description Logic Ontologies from Data Retrieval Examples

Boris Konev[1], Ana Ozaki[1], and Frank Wolter[1]

Department of Computer Science, University of Liverpool, UK

Abstract. We investigate the complexity of learning description logic ontologies in Angluin et al.'s framework of exact learning via queries posed to an oracle. We consider membership queries of the form "is individual a a certain answer to a data retrieval query q of ABox $\mathcal{A}$ and the target TBox?" and equivalence queries of the form "is a given TBox equivalent to the target TBox?". We show that (i) DL-Lite TBoxes with role inclusions and $\mathcal{ELI}$ concept expressions on the right-hand side of inclusions and (ii) $\mathcal{EL}$ TBoxes without complex concept expressions on the right-hand side of inclusions can be learned in polynomial time. Both results are proved by a non-trivial reduction to learning from subsumption examples. We also show that arbitrary $\mathcal{EL}$ TBoxes cannot be learned in polynomial time.

1 Introduction

Building an ontology is prone to errors, time consuming, and costly. A large number of different research communities have addressed this problem by, for example, supplying tool support for editing ontologies [15, 4, 9], developing reasoning support for debugging ontologies [18], supporting modular ontology design [17], and investigating automated ontology generation from data or text [8, 6, 5, 14]. One major problem when building an ontology is the fact that domain experts are rarely ontology engineering experts and that, conversely, ontology engineers are typically not familiar with the domain of the ontology. An ontology building project therefore often relies on the successful communication between an ontology engineer (familiar with the semantics of ontology languages) and a domain expert (familiar with the domain of interest). In this paper, we consider a simple model of this communication process and analyse, within this model, the computational complexity of reaching a correct domain ontology. We assume that

- the domain expert knows the domain ontology and its vocabulary without being able to formalize or communicate this ontology;
- the domain expert is able to communicate the vocabulary of the ontology and shares it with the the ontology engineer. Thus, the domain expert and ontology engineer have a common understanding of the vocabulary of the ontology. The ontology engineer knows nothing else about the domain.
- the ontology engineer can pose queries to the domain expert which the domain expert answers truthfully. Assuming that the domain expert can interpret data in her area of expertise, the main queries posed by the ontology engineer are based on instance retrieval examples:

- assume a data instance $\mathcal{A}$ and a query $q(x)$ are given. Is the individual a a certain answer to query $q(x)$ in $\mathcal{A}$ and the ontology $\mathcal{O}$?

In addition, we require a way for the ontology engineer to find out whether she has reconstructed the target ontology already and, if this is not the case, to request an example illustrating the incompleteness of the reconstruction. We abstract from defining a communication protocol for this, but assume for simplicity that the following query can be posed by the ontology engineer:

- Is this ontology $\mathcal{H}$ complete? If not, return a data instance $\mathcal{A}$, a query $q(x)$, and an individual a such that a is a certain answer to $q(x)$ in $\mathcal{A}$ and the ontology $\mathcal{O}$ and it is not a certain answer to $q(x)$ in $\mathcal{A}$ and the ontology $\mathcal{H}$.

Given this model, our question is whether the ontology engineer can learn the target ontology $\mathcal{O}$ and which computational resources are required for this depending on the ontology language in which the ontology $\mathcal{O}$ and the hypothesis ontologies $\mathcal{H}$ are formulated. Our model obviously abstracts from a number of fundamental problems in building ontologies and communicating about them. In particular, it is not realistic to assume that the domain expert knows the domain ontology and its vocabulary (without being able to formalize it) as it is well known that finding an appropriate vocabulary for a domain of interest is a major problem in ontology design [8]. We make this assumption here in order to isolate the problem of communication about the logical relationships between known vocabulary items and its dependence on the ontology language within which the relationships can be formulated.

The model described above is an instance of Angluin et al.'s framework of exact learning via queries to an oracle [1]. The queries using instance retrieval examples can be regarded as membership queries posed by a learner to an oracle and the completeness query based on a hypothesis $\mathcal{H}$ can be regarded as an equivalence query by the learner to the oracle. Formulated in Angluin's terms we are thus interested in whether there exists a deterministic learning algorithm that poses membership and equivalence queries of the above form to an oracle and that learns an arbitrary ontology over a given ontology language in polynomial time. We consider polynomial learnability in three distinct DLs: we show that DL-Lite ontologies with role inclusions and arbitrary $\mathcal{ELI}$ concepts on the right-hand side of concept inclusions can be learned in polynomial time if database queries in instance retrieval examples are $\mathcal{ELI}$ instance queries (or, equivalently, acyclic conjunctive queries). We call this DL DL-Lite$_{\mathcal{R}}^{\exists}$ and note that it is the core of the web ontology language profile OWL2 QL. We also note that *without* complex $\mathcal{ELI}$ concepts on the right-hand side of concept inclusions, polynomial learnability would be trivial as only finitely many non-equivalent such TBoxes exist over a given vocabulary of concept and role names. The second DL we consider is $\mathcal{EL}$ which is the logic underpinning the web ontology language profile OWL2 EL. We show that $\mathcal{EL}$ TBoxes cannot be learned in polynomial time using the protocol above if the database queries in instance retrieval examples are $\mathcal{EL}$ instance queries. We then consider the fragment $\mathcal{EL}_{\mathsf{lhs}}$ of $\mathcal{EL}$ without complex concepts on the right-hand side of concept inclusions and prove that it can be learned in polynomial time using the above protocol with instance retrieval examples. The proofs of the positive learning results are by reduction to polynomial time learnability results for DL-Lite$_{\mathcal{R}}^{\exists}$ and $\mathcal{EL}_{\mathsf{lhs}}$ for the case in which *concept subsumptions* rather than instance retrieval examples as used in the

communication between the learner and the oracle [12]. Detailed proofs are provided in the full version at http://cgi.csc.liv.ac.uk/anaozaki/publ.html.

2 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$^{\exists}_{\mathcal{R}}$ of DL-Lite is defined as follows [7]. 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$^{\exists}_{\mathcal{R}}$ concept inclusion (CI) is of the form $B \sqsubseteq C$, where B is a basic concept expression and C is an $\mathcal{ELI}$ concept expression, that is, C is formed according to the rule $C, D := A \mid \top \mid C \sqcap D \mid \exists r.C \mid \exists r^-.C$ where A ranges over $\mathsf{N_C}$ and r ranges over $\mathsf{N_R}$. A DL-Lite$^{\exists}_{\mathcal{R}}$ TBox is a finite set of DL-Lite$^{\exists}_{\mathcal{R}}$ CIs and RIs. As usual, an $\mathcal{EL}$ *concept expression* is an $\mathcal{ELI}$ concept expression that does not use inverse roles, an $\mathcal{EL}$ *concept inclusion* has the form $C \sqsubseteq D$ with C and D $\mathcal{EL}$ concept expressions, and a *(general)* $\mathcal{EL}$ *TBox* is a finite set of $\mathcal{EL}$ concept inclusions [2]. We also 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. The *size* of a concept expression 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. A TBox *signature* is the set of concept and role names occurring in the TBox. The *size* of a TBox $\mathcal{T}$, denoted with $|\mathcal{T}|$, is $\sum_{C \sqsubseteq D \in \mathcal{T}} |C| + |D|$.

Let $\mathsf{N_I}$ be a countably infinite set of *individual names*. An ABox $\mathcal{A}$ is a finite non-empty set containing *concept name assertions* $A(a)$ and *role assertions* $r(a, b)$, where a, b are individuals in $\mathsf{N_I}$, A is a concept name and r is a role. The set of individuals that occur in $\mathcal{A}$ is denoted by $\mathsf{Ind}(\mathcal{A})$. We say that $\mathcal{A}$ is a *singleton* ABox if it contains only one ABox assertion. Assertions of the form $C(a)$ and $r(a, b)$, where $a, b \in \mathsf{N_I}$, C an $\mathcal{ELI}$ concept expression, and $r \in \mathsf{N_R}$, are called *instance assertions*. Note that instance assertions of the form $C(a)$ with C not a concept name nor $C = \top$ do not occur in ABoxes. The semantics of description logic is defined as usual [3]. We write $\mathcal{I} \models \alpha$ to say that an inclusion or assertion α is true in $\mathcal{I}$. An interpretation $\mathcal{I}$ is a *model* of a KB $(\mathcal{T}, \mathcal{A})$ if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{T} \cup \mathcal{A}$. $(\mathcal{T}, \mathcal{A}) \models \alpha$ means that $\mathcal{I} \models \alpha$ for all models $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$.

A *learning framework* $\mathfrak{F}$ is a triple $(X, \mathcal{L}, \mu)$, where X is a set of *examples* (also called domain or instance space), $\mathcal{L}$ is a set of *learning concepts*[1] and μ is a mapping from $\mathcal{L}$ to 2^X. The *subsumption* learning framework $\mathfrak{F}_{\mathcal{S}}$, studied in [12], is defined as $(X_{\mathcal{S}}, \mathcal{L}, \mu_{\mathcal{S}})$, where $\mathcal{L}$ is the set of all TBoxes that are formulated in a given DL; $X_{\mathcal{S}}$ is the set of *subsumption examples* of the form $C \sqsubseteq D$, where C, D are concept expressions of the DL under consideration; and $\mu_{\mathcal{S}}(\mathcal{T})$ is defined as $\{C \sqsubseteq D \in X_{\mathcal{S}} \mid \mathcal{T} \models C \sqsubseteq D\}$, for every $\mathcal{T} \in \mathcal{L}$. It should be clear that $\mu_{\mathcal{S}}(\mathcal{T}) = \mu_{\mathcal{S}}(\mathcal{T}')$ if, and only if, the TBoxes $\mathcal{T}$ and $\mathcal{T}'$ entail the same set of inclusions, that is, they are logically equivalent.

[1] In the learning literature (e.g., [1]), the term 'learning concept' is often defined as a set of examples. We do not distinguish between learning concepts and their representations and only consider representable learning concepts to emphasize on the task of identifying a TBox that is logically equivalent to the target TBox.

We study the *data retrieval* learning framework $\mathfrak{F}_{\mathcal{D}}$ defined as $(X_{\mathcal{D}}, \mathcal{L}, \mu_{\mathcal{D}})$, where $\mathcal{L}$ is same as in $\mathfrak{F}_{\mathcal{S}}$; X is the set of *data retrieval examples* of the form $(\mathcal{A}, D(a))$, where $\mathcal{A}$ is an ABox, $D(a)$ is a concept assertion of the DL under consideration, and $a \in \mathsf{Ind}(\mathcal{A})$; and $\mu(\mathcal{T}) = \{(\mathcal{A}, D(a)) \in X_{\mathcal{D}} \mid (\mathcal{T}, \mathcal{A}) \models D(a)\}$. As in the case of learning from subsumptions, $\mu_{\mathcal{S}}(\mathcal{T}) = \mu_{\mathcal{S}}(\mathcal{T}')$ if, and only if, the TBoxes $\mathcal{T}$ and $\mathcal{T}'$ are logically equivalent.

Given a learning framework $\mathfrak{F} = (X, \mathcal{L}, \mu)$, we are interested in the exact identification of a *target* learning concept $l \in \mathcal{L}$ by posing queries to oracles. Let $\mathsf{MEM}_{l,X}$ be the oracle that takes as input some $x \in X$ and returns 'yes' if $x \in \mu(l)$ and 'no' otherwise. We say that x is a *positive example* for l if $x \in \mu(l)$ and a *negative example* for l if $x \notin \mu(l)$. Then a *membership query* is a call to the oracle $\mathsf{MEM}_{l,X}$. Similarly, for every $l \in \mathcal{L}$, we denote by $\mathsf{EQ}_{l,X}$ the oracle that takes as input a *hypothesis* learning concept $h \in \mathcal{L}$ and returns 'yes', if $\mu(h) = \mu(l)$, or a *counterexample* $x \in \mu(h) \oplus \mu(l)$ otherwise, where $\oplus$ denotes the symmetric set difference. An *equivalence query* is a call to the oracle $\mathsf{EQ}_{l,X}$.

We say that a learning framework $(X, \mathcal{L}, \mu)$ is *exact learnable* if there is an algorithm A such that for any target $l \in \mathcal{L}$ the algorithm A always halts and outputs $l' \in \mathcal{L}$ such that $\mu(l) = \mu(l')$ using membership and equivalence queries answered by the oracles $\mathsf{MEM}_{l,X}$ and $\mathsf{EQ}_{l,X}$, respectively. A learning framework $(X, \mathcal{L}, \mu)$ is *polynomially* exact learnable if it is exact learnable by an algorithm A such that at every step[2] of computation the time used by A up to that step is bounded by a polynomial $p(|l|, |x|)$, where l is the target and $x \in X$ is the largest counterexample seen so far[3]. As argued in the introduction, for learning subsumption and data retrieval learning frameworks we additionally assume that the signature of the target TBox is always known to the learner.

An important class of learning algorithms—in particular, all algorithms presented in [12, 10, 16] fit in this class—always make equivalence queries with hypotheses h which are polynomial in the size of l and such that $\mu(h) \subseteq \mu(l)$, so that counterexamples returned by the $\mathsf{EQ}_{l,X}$ oracles are always positive. We say that such algorithms use *positive bounded equivalence queries*.

3 Polynomial Time Learnability

In this section we prove polynomial time exact learnability of the DL-Lite$^{\exists}_{\mathcal{R}}$ and $\mathcal{EL}_{\mathsf{lhs}}$ data retrieval learning frameworks. These frameworks are instances of the general definition given above, where the concept expression D in a data retrieval example $(\mathcal{A}, D(a))$ is an $\mathcal{ELI}$ concept expression in the DL-Lite$^{\exists}_{\mathcal{R}}$ framework and an $\mathcal{EL}$ concept expression in the $\mathcal{EL}_{\mathsf{lhs}}$ framework, respectively.

The proof is by reduction to learning from subsumptions. We illustrate its idea for $\mathcal{EL}_{\mathsf{lhs}}$. To learn a TBox from data retrieval examples we run a learning from subsumptions algorithm as a 'black box'. Every time the learning from subsumptions algorithm makes a membership or an equivalence query we rewrite the query into the data setting and pass it on to the data retrieval oracle. The oracle's answer, rewritten back to the subsumption

[2] We count each call to an oracle as one step of computation.

[3] We assume some natural notion of a length of an example x and a learning concept l, denoted $|x|$ and $|l|$, respectively.

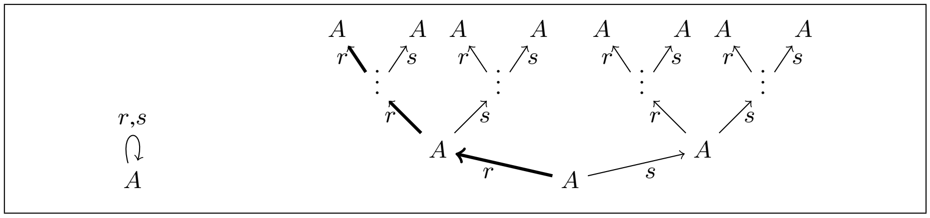


Fig. 1: An ABox $\mathcal{A} = \{r(a,a), s(a,a), A(a)\}$ and its unravelling up to level n.

setting, is given to the learning from subsumptions algorithm. When the learning from subsumptions algorithm terminates we return the learnt TBox. This reduction is made possible by the close relationship between data retrieval and subsumption examples. For every TBox $\mathcal{T}$ and inclusions $C \sqsubseteq D$, one can interpret a concept expression C as a labelled tree and encode this tree as an ABox $\mathcal{A}_C$ with root ρ_C such that $\mathcal{T} \models C \sqsubseteq D$ iff $(\mathcal{T}, \mathcal{A}_C) \models D(\rho_C)$.

Then, membership queries in the subsumption setting can be answered with the help of a data retrieval oracle due to the relation between subsumptions and instance queries described above. An inclusion $C \sqsubseteq D$ is a (positive) subsumption example for some target TBox $\mathcal{T}$ if, and only if, $(\mathcal{A}_C, D(\rho_C))$ is a (positive) data retrieval example for the same target $\mathcal{T}$. To handle equivalence queries, we need to be able to rewrite data retrieval counterexamples returned by the data retrieval oracle into the subsumption setting. For every TBox $\mathcal{T}$ and data retrieval query $(\mathcal{A}, D(a))$ one can construct a concept expression $C_\mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}) \models D(a)$ iff $\mathcal{T} \models C_\mathcal{A} \sqsubseteq D$. Such a concept expression $C_\mathcal{A}$ can be obtained by unravelling $\mathcal{A}$ into a tree-shaped ABox and representing it as a concept expression. This unravelling, however, can increase the ABox size exponentially. Thus, to obtain a polynomial bound on the running time of the learning process, $C_\mathcal{A} \sqsubseteq D$ cannot be simply returned as an answer to a subsumption equivalence query. For example, for a target TBox $\mathcal{T} = \{\exists r^n.A \sqsubseteq B\}$ and a hypothesis $\mathcal{H} = \emptyset$ the data retrieval query $(\mathcal{A}, B(a))$, where $\mathcal{A} = \{r(a,a), s(a,a), A(a)\}$, is a positive counterexample. The tree-shaped unravelling of $\mathcal{A}$ up to level n is a full binary tree of depth n, as shown in Fig. 1. On the other hand, the non-equivalence of $\mathcal{T}$ and $\mathcal{H}$ can already be witnessed by $(\mathcal{A}', B(a))$, where $\mathcal{A}' = \{r(a,a), A(a)\}$. The unravelling of $\mathcal{A}'$ up to level n produces a linear size ABox $\{r(a,a_2), r(a_2,a_3), \ldots, r(a_{n-1},a_n), A(a), A(a_2), \ldots, A(a_n)\}$, corresponding to the left-most path in Fig. 1, which, in turn, is linear-size w.r.t. the target inclusion $\exists r^n.A \sqsubseteq B$. Notice that $\mathcal{A}'$ is obtained from $\mathcal{A}$ by removing the $s(a,a)$ edge and checking, using membership queries, whether $(\mathcal{T}, \mathcal{A}') \models q$ still holds. In other words, one might need to ask further membership queries in order to rewrite answers to data retrieval equivalence queries given by the data retrieval oracle into the subsumption setting.

We address the need of rewriting counterexamples by introducing an abstract notion of reduction between different exact learning frameworks. To simplify notation, we assume that both learning frameworks use the same set of learning concepts $\mathcal{L}$ and only consider positive bounded equivalence queries. This definition of reduction can be easily extended to arbitrary learning frameworks and arbitrary queries.

We say that a learning framework $\mathfrak{F} = (X, \mathcal{L}, \mu)$ *polynomially reduces* to $\mathfrak{F}' = (X', \mathcal{L}, \mu')$ if for some polynomials $p_1(\cdot)$, $p_2(\cdot)$ and $p_3(\cdot,\cdot)$ there exist a function $f : X' \to X$ and a partial function $g : \mathcal{L} \times \mathcal{L} \times X \to X'$, defined for every (l, h, x) such that $|h| = p_1(|l|)$, $\mu(h) \subseteq \mu(l)$ and $x \in X$, for which the following conditions hold.

- For all $x' \in X'$ we have $x' \in \mu'(l)$ if, and only if, $f(x') \in \mu(l)$;
- For all $x \in X$ we have $x \in \mu(l) \setminus \mu(h)$ if, and only if, $g(l, h, x) \in \mu'(l) \setminus \mu'(h)$;
- $f(x')$ is computable in time $p_2(|x'|)$;
- $g(l, h, x)$ is computable in time $p_3(|l|, |x|)$ and l can only be accessed by calls to the membership oracle $\mathsf{MEM}_{l,X}$.

As in the case of learning algorithms, we consider every call to the oracle as one step of computation. Notice also that even though g takes h as input, the polynomial time bound on computing $g(l, h, x)$ does not depend on the size of h as g is only defined for h polynomial in the size of l.

Theorem 1. *Let $(X, \mathcal{L}, \mu)$ and $(X', \mathcal{L}, \mu')$ be learning frameworks. If there exists a polynomial reduction from $(X, \mathcal{L}, \mu)$ to $(X', \mathcal{L}, \mu')$ and a polynomial learning algorithm for $(X', \mathcal{L}, \mu')$ that uses membership queries and positive bounded equivalence queries then $(X, \mathcal{L}, \mu)$ is polynomially exact learnable.*

We use Theorem 1 to prove that DL-Lite$^{\exists}_{\mathcal{R}}$ and $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes can be learned in polynomial time from data retrieval examples. We employ the following result:

Theorem 2 ([12]). *The DL-Lite$^{\exists}_{\mathcal{R}}$ and $\mathcal{EL}_{\mathsf{lhs}}$ subsumption learning frameworks are polynomial time exact learnable with membership and positive bounded equivalence queries.*

As the existence of f is guaranteed by the following lemma, in what follows we prove the existence of g and establish the corresponding time bounds.

Lemma 1. *Let $L \in \{$DL-Lite$^{\exists}_{\mathcal{R}}, \mathcal{EL}_{\mathsf{lhs}}\}$ and let $C \sqsubseteq D$ be an L concept inclusion. Then $(\mathcal{T}, \mathcal{A}_C) \models D(\rho_C)$ if, and only if, $\mathcal{T} \models C \sqsubseteq D$.*

Polynomial Reduction for DL-Lite$^{\exists}_{\mathcal{R}}$ TBoxes We show for any target $\mathcal{T}$ and hypothesis $\mathcal{H}$ polynomial in the size of $\mathcal{T}$ that Algorithm 1 transforms every positive counterexample in polynomial time to a positive counterexample with a singleton ABox (i.e., of the form $\{A(a)\}$ or $\{r(a, b)\}$). Using the equivalences $(\mathcal{T}, \{A(a)\}) \models C(a)$ iff $\mathcal{T} \models A \sqsubseteq C$ and $(\mathcal{T}, \{r(a, b)\}) \models C(a)$ iff $\mathcal{T} \models \exists r.\top \sqsubseteq C$, we then obtain a positive subsumption counterexample, so $g(l, h, x)$ is computable in polynomial time.

Given a positive data retrieval counterexample $(\mathcal{A}, C(a))$, Algorithm 1 exhaustively applies the *role saturation* and *parent-child merging* rules introduced in [12]. We say that an instance assertion $C(a)$ is *role saturated* for $(\mathcal{T}, \mathcal{A})$ if $(\mathcal{T}, \mathcal{A}) \not\models C'(a)$ whenever C' is the result of replacing a role r by some role $s \in \mathsf{N_R} \cap \Sigma_{\mathcal{T}}$ with $\mathcal{T} \not\models r \sqsubseteq s$ and $\mathcal{T} \models s \sqsubseteq r$, where $\Sigma_{\mathcal{T}}$ is the signature of the target TBox $\mathcal{T}$ known to the learner. To define parent/child merging, we identify each $\mathcal{ELI}$ concept C with a finite tree T_C whose nodes are labeled with concept names and edges are labeled with roles in the standard way. For example, if $C = \exists t.(A \sqcap \exists r.\exists r^-.\exists r.B) \sqcap \exists s.\top$ then Fig. 2a illustrates

Algorithm 1 Reducing the positive counterexample

1: Let $C(a)$ be an instance assertion such that $(\mathcal{H}, \mathcal{A}) \not\models C(a)$ and $(\mathcal{T}, \mathcal{A}) \models C(a)$
2: **function** REDUCECOUNTEREXAMPLE($\mathcal{A}$, $C(a)$)
3: Find a role saturated and parent/child merged $C(a)$ (membership queries)
4: **if** $C = C_0 \sqcap ... \sqcap C_n$ **then**
5: Find C_i, $0 \leq i \leq n$, such that $(\mathcal{H}, \mathcal{A}) \not\models C_i(a)$
6: $C := C_i$
7: **if** $C = \exists r.C'$ and there is $r(a, b) \in \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}) \models C'(b)$ **then**
8: REDUCECOUNTEREXAMPLE($\mathcal{A}$, $C'(b)$)
9: **else**
10: Find a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models C(a)$ but
11: $(\mathcal{H}, \mathcal{A}') \not\models C(a)$ (membership queries)
12: **return** ($\mathcal{A}'$,$C(a)$)

T_C. Now, we say that an instance assertion $C(a)$ is *parent/child merged* for $\mathcal{T}$ and $\mathcal{A}$ if for nodes n_1, n_2, n_3 in T_C such that n_2 is an r-successor of n_1, n_3 is an s-successor of n_2 and $\mathcal{T} \models r^- \equiv s$ we have $(\mathcal{T}, \mathcal{A}) \not\models C'(a)$ if C' is the concept that results from identifying n_1 and n_3. For instance, the concept in Fig. 2c is the result of identifying the leaf labeled with B in Fig. 2b with the parent of its parent.

We present a run of Algorithm 1 for $\mathcal{T} = \{A \sqsubseteq \exists s.B, s \sqsubseteq r\}$ and $\mathcal{H} = \{s \sqsubseteq r\}$. Assume the oracle gives as counterexample $(\mathcal{A}, C(a))$, where $\mathcal{A} = \{t(a, b), A(b), s(a, c)\}$ and $C(a) = \exists t.(A \sqcap \exists r.\exists r^-.\exists r.B) \sqcap \exists s.\top(a)$ (Fig. 2a). Role saturation produces $C(a) = \exists t.(A \sqcap \exists s.\exists s^-.\exists s.B) \sqcap \exists s.\top(a)$ (Fig. 2b). Then, applying parent/child merging twice we obtain $C(a) = \exists t.(A \sqcap \exists s.B) \sqcap \exists s.\top(a)$ (Fig. 2c and 2d).

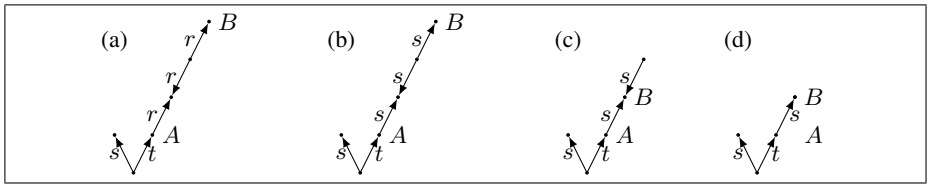


Fig. 2: Concept C being role saturated and parent/child merged.

Since $(\mathcal{H}, \mathcal{A}) \not\models \exists t.(A \sqcap \exists s.B)(a)$, after Lines 4-6, Algorithm 1 updates C by choosing the conjunct $\exists t.(A \sqcap \exists s.B)$. As C is of the form $\exists t.C'$ and there is $t(a, b) \in \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}) \models C'(b)$, the algorithm recursively calls the function "ReduceCounterExample" with $A \sqcap \exists s.B(b)$. Now, since $(\mathcal{H}, \mathcal{A}) \not\models \exists s.B(b)$, after Lines 4-6, C is updated to $\exists s.B$. Finally, C is of the form $\exists t.C'$ and there is no $t(b, c) \in \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}) \models C'(c)$. So the algorithm proceeds to Lines 11-12, where it chooses $A(b) \in \mathcal{A}$. Since $(\mathcal{T}, \{A(b)\}) \models \exists s.B(b)$ and $(\mathcal{H}, \{A(b)\}) \not\models \exists s.B(b)$ we have that $\mathcal{T} \models A \sqsubseteq \exists s.B$ and $\mathcal{H} \not\models A \sqsubseteq \exists s.B$.

Lemma 2. *Let $(\mathcal{A}, C(a))$ be a positive counterexample. Then the following holds:*

1. *if C is a basic concept then there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models C(a)$;*

Algorithm 2 Minimizing an ABox $\mathcal{A}$

1: Let $\mathcal{A}$ be an ABox such that $(\mathcal{T}, \mathcal{A}) \models A(a)$ but $(\mathcal{H}, \mathcal{A}) \not\models A(a)$, for $A \in \mathsf{N_C}$, $a \in \mathsf{Ind}(\mathcal{A})$.
2: **function** MINIMIZEABOX($\mathcal{A}$)
3: Concept saturate $\mathcal{A}$ with $\mathcal{H}$
4: **for** every $A \in \mathsf{N_C} \cap \Sigma_\mathcal{T}$ and $a \in \mathsf{Ind}(\mathcal{A})$ such that
5: $(\mathcal{T}, \mathcal{A}) \models A(a)$ and $(\mathcal{H}, \mathcal{A}) \not\models A(a)$ **do**
6: Domain Minimize $\mathcal{A}$ with $A(a)$
7: Role Minimize $\mathcal{A}$ with $A(a)$
8: **return** ($\mathcal{A}$)

2. *if C is of the form $\exists r.C'$ (or $\exists r^-.C'$) and C is role saturated and parent/child merged then either there is $r(a, b) \in \mathcal{A}$ (or $r(b, a) \in \mathcal{A}$) such that $(\mathcal{T}, \mathcal{A}) \models C'(b)$ or there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models C(a)$.*

Lemma 3. *For any target DL-Lite$^\exists_\mathcal{R}$ TBox $\mathcal{T}$ and hypothesis DL-Lite$^\exists_\mathcal{R}$ TBox $\mathcal{H}$ given a positive data retrieval counterexample $(\mathcal{A}, C(a))$, Algorithm 1 computes in time polynomial in $|\mathcal{T}|$, $|\mathcal{H}|$, $|\mathcal{A}|$ and $|C|$ a counterexample $C'(b)$ such that $(\mathcal{T}, \mathcal{A}') \models C'(b)$, where $\mathcal{A}' \subseteq \mathcal{A}$ is a singleton ABox.*

Proof. (Sketch) Let $(\mathcal{A}, C(a))$ be the input of "ReduceCounterExample". The number of membership queries in Line 3 is polynomial in $|C|$ and $|\mathcal{T}|$. If C has more than one conjunct then it is updated in Lines 4-6, so C becomes either (1) a basic concept or (2) of the form $\exists r.C'$ (or $\exists r^-.C'$). By Lemma 2 in case (1) there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models C(a)$, computed by Line 11 of Algorithm 1. In case (2) either there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models C(a)$, computed by Line 11 of Algorithm 1, or we obtain a counterexample with a refined C. Since the size of the refined counterexample is strictly smaller after every recursive call of "ReduceCounterExample", the total number of calls is bounded by $|C|$. ❑

Using Theorem 2 and Theorem 1 we obtain:

Theorem 3. *The DL-Lite$^\exists_\mathcal{R}$ data retrieval framework is polynomially exact learnable.*

Polynomial Reduction for $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes In this section we give a polynomial algorithm computing g for $\mathcal{EL}_{\mathsf{lhs}}$. First we note that the concept assertion in data retrieval counterexamples $(\mathcal{A}, D(a))$ can always be made atomic. Let $\Sigma_\mathcal{T}$ be the signature of the target TBox $\mathcal{T}$.

Lemma 4. *If $(\mathcal{A}, D(a))$ is a positive counterexample then by posing polynomially many membership queries one can find a concept name $A \in \Sigma_\mathcal{T}$ and an individual $b \in \mathsf{Ind}(\mathcal{A})$ such that $(\mathcal{A}, A(b))$ is also a counterexample.*

Thus it suffices to show that given a positive counterexample $(\mathcal{A}, D(a))$ with $D \in \mathsf{N_C}$, one can compute an $\mathcal{EL}$ concept expression C bounded in size by $|\mathcal{T}|$ such that $(\mathcal{T}, \{C(b)\}) \models A(b)$ and $(\mathcal{H}, \{C(b)\}) \not\models A(b)$, where $A \in \mathsf{N_C}$. As $(\mathcal{T}, \{C(b)\}) \models A(b)$ if and only if $\mathcal{T} \models C \sqsubseteq A$, we obtain a positive subsumption counterexample. Our algorithm for computing g is based on two operations: minimization, computed by

Algorithm 3 Computing a tree shaped ABox

function FINDTREE($\mathcal{A}$)
  MINIMIZEABOX($\mathcal{A}$)
  while there is a cycle c in $\mathcal{A}$ **do**
    Unfold $a \in \mathsf{Ind}(\mathcal{A})$ in cycle c
    MINIMIZEABOX($\mathcal{A}$)
  Let C be the concept expression corresponding to $\mathcal{A}$ with counterexample $A(a)$.
  return $(C(a),A(a))$

Algorithm 2, and unfolding. Algorithm 2 *minimizes* a given ABox with the following rules.

(Concept saturate $\mathcal{A}$ with $\mathcal{H}$) If $A(a) \notin \mathcal{A}$ and $(\mathcal{H}, \mathcal{A}) \models A(a)$ then replace $\mathcal{A}$ by $\mathcal{A} \cup \{A(a)\}$, where $A \in \mathsf{N_C} \cap \Sigma_{\mathcal{T}}$ and $a \in \mathsf{Ind}(\mathcal{A})$.

(Domain Minimize $\mathcal{A}$ with $A(a)$) If $A(a)$ is a counterexample and $(\mathcal{T}, \mathcal{A}^{-b}) \models A(a)$ then replace $\mathcal{A}$ by $\mathcal{A}^{-b}$, where $\mathcal{A}^{-b}$ is the result of removing from $\mathcal{A}$ all ABox assertions in which b occurs.

(Role Minimize $\mathcal{A}$ with $A(a)$) If $A(a)$ is a counterexample and $(\mathcal{T}, \mathcal{A}^{-r(b,c)}) \models A(a)$ then replace $\mathcal{A}$ by $\mathcal{A}^{-r(b,c)}$, where $\mathcal{A}^{-r(b,c)}$ be obtained by removing a role assertion $r(b, c)$ from $\mathcal{A}$.

Lemma 5. *Given a positive counterexample $(\mathcal{A}, D(a))$ with $D \in \mathsf{N_C}$, Algorithm 2 computes in polynomially many steps with respect to $|\mathcal{A}|$, $|\mathcal{H}|$, and $|\mathcal{T}|$ an ABox $\mathcal{A}'$ such that $|\mathsf{Ind}(\mathcal{A}')| \leq |\mathcal{T}|$ and $(\mathcal{A}', A(b))$ is a positive counterexample, for some $A \in \mathsf{N_C}$ and $b \in \mathsf{Ind}(\mathcal{A}')$.*

It remains to show that $\mathcal{A}$ can be made tree-shaped. We say that $\mathcal{A}$ has an (undirected) cycle if there is a finite sequence $a_0 \cdot r_1 \cdot a_1 \cdot \ldots \cdot r_k \cdot a_k$ such that (i) $a_0 = a_k$ and (ii) there are mutually distinct assertions of the form $r_{i+1}(a_i, a_{i+1})$ or $r_{i+1}(a_{i+1}, a_i)$ in $\mathcal{A}$, for $0 \leq i < k$. The *unfolding* of a cycle $c = a_0 \cdot r_1 \cdot a_1 \cdot \ldots \cdot r_k \cdot a_k$ in a given ABox $\mathcal{A}$ is obtained by replacing c by the cycle $c' = a_0 \cdot r_1 \cdot a_1 \cdot \ldots \cdot r_k \cdot a_{k-1} \cdot r_k \cdot \widehat{a}_0 \cdot r_1 \cdots \widehat{a}_{k-1} \cdot r_k \cdot a_0$, where $\widehat{a}_i$ are fresh individual names, $0 \leq i \leq k-1$, in such a way that (i) if $r(a_i, d) \in \mathcal{A}$, for an individual d not in the cycle, then $r(\widehat{a}_i, d) \in \mathcal{A}$; and (ii) if $A(a_i) \in \mathcal{A}$ then $A(\widehat{a}_i) \in \mathcal{A}$.

We prove in the appendix that after every unfolding-minimisation step in Algorithm 3 the ABox $\mathcal{A}$ on the one hand becomes strictly larger and on the other does not exceed the size of the target TBox $\mathcal{T}$. Thus Algorithm 3 terminates after a polynomial number of steps yielding a tree-shaped counterexample.

Lemma 6. *Algorithm 3 computes a minimal tree shaped ABox $\mathcal{A}$ with size polynomial in $|\mathcal{T}|$ and runs in polynomially many steps in $|\mathcal{T}|$ and $|\mathcal{A}|$.*

Using Theorem 2 and Theorem 1 we obtain:

Theorem 4. *The $\mathcal{EL}_{\mathsf{lhs}}$ data retrieval framework is polynomially exact learnable.*

4 Limits of Polynomial Time Learnability

Our proof of non-polynomial learnability of general $\mathcal{EL}$ TBoxes from data retrieval examples extends previous results on non-polynomial learnability of $\mathcal{EL}$ TBoxes from

subsumptions [12]. We start by giving a brief overview of the construction in [12], show that it fails in the data retrieval setting and then demonstrate how it can be modified.

The non-learnability proof in [12] proceeds as follows. A learner tries to exactly identify one of the possible target TBoxes $\{\mathcal{T}_L \mid L \in \mathfrak{L}_n\}$, for a superpolynomial in n set $\mathfrak{L}_n$ defined below. At every stage of computation the oracle maintains a set of TBoxes S, which the learner is unable to distinguish based on the answers given so far. Initially $S = \{\mathcal{T}_L \mid L \in \mathfrak{L}_n\}$. It has been proved that for any $\mathcal{EL}$ inclusion $C \sqsubseteq D$ either $\mathcal{T}_L \models C \sqsubseteq D$ for every $L \in \mathfrak{L}_n$ or the number of $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L \models C \sqsubseteq D$ does not exceed $|C|$. When a polynomial learner asks a membership query $C \sqsubseteq D$ the oracle answers 'yes' if $\mathcal{T}_L \models C \sqsubseteq D$ for every $L \in \mathfrak{L}_n$ and 'no' otherwise. In the latter case the oracle removes polynomially many $\mathcal{T}_L$ such that $\mathcal{T}_L \models C \sqsubseteq D$ from S. Similarly, for any equivalence query with hypothesis $\mathcal{H}$ asked by a polynomial learning algorithm there exists a polynomial size inclusion $C \sqsubseteq D$, which can be returned as a counterexample and that excludes only polynomially many TBoxes from S. Thus, every query to the oracle reduces the size of S at most polynomially in n, but then the learner cannot distinguish between the remaining TBoxes of our initial superpolynomial set S.

The set of indices $\mathfrak{L}_n$ and the target TBoxes $\mathcal{T}_L$ are defined as follows. Fix two role names r and s. An *n-tuple* L is a sequence of role sequences $(\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n)$, where every $\boldsymbol{\sigma}_i$ is a sequence of role names r and s, that is $\boldsymbol{\sigma}_i = \sigma_i^1 \sigma_i^2 \dots \sigma_i^n$ with $\sigma_i^j \in \{r, s\}$. Then $\mathfrak{L}_n$ is a set of n-tuples such that for 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 $i = j$. There are $N = \lfloor 2^n/n \rfloor$ different tuples in $\mathfrak{L}_n$. For every $n > 0$ and every n-tuple $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n)$ we define an acyclic $\mathcal{EL}$ TBox $\mathcal{T}_L$ as the union of $\mathcal{T}_0 = \{X_i \sqsubseteq \exists r.X_{i+1} \sqcap \exists s.X_{i+1} \mid 0 \leq i < n\}$ and the following inclusions:

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

$$A \equiv X_0 \sqcap \exists \boldsymbol{\sigma}_1.M \sqcap \dots \sqcap \exists \boldsymbol{\sigma}_n.M.$$

where the expression $\exists \boldsymbol{\sigma}.C$ stands for $\exists \sigma^1.\exists \sigma^2 \dots \exists \sigma^n.C$, M is a concept name that 'marks' a designated path given by $\boldsymbol{\sigma}$ and $\mathcal{T}_0$ 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.

In contrast to the subsumption framework, every $\mathcal{T}_L$ *can* be exactly identified using data retrieval queries. For example, as $X_0 \sqcap \exists \boldsymbol{\sigma}_1.M \sqcap \dots \sqcap \exists \boldsymbol{\sigma}_n.M \sqsubseteq A \in \mathcal{T}_L$, a learning from data retrieval queries algorithm can learn all the sequences in the n-tuple $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n)$, by defining an ABox $\mathcal{A} = \{X_0(a_1), r(a_1, a_2), s(a_1, a_2), \dots, r(a_{n-1}, a_n), s(a_{n-1}, a_n), M(a_n)\}$ and then proceeding with unfolding and minimizing $\mathcal{A}$ via membership queries of the form $(\mathcal{T}_L, \mathcal{A}) \models A(a_1)$.

To show the non-tractability for data retrieval queries, we first modify S in such a way that the concept expression which 'marks' the sequences in $L = (\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_n)$ is now given by the set $\mathfrak{B}_n$ of all conjunctions $F_1 \sqcap \dots \sqcap F_n$, where $F_i \in \{E_i, \bar{E}_i\}$, for $1 \leq i \leq n$. Intuitively, every member of $\mathfrak{B}_n$ encodes a binary string of length n with E_i encoding 1 and $\bar{E}_i$ encoding 0. For every $L \in \mathfrak{L}_n$ and every $\mathbf{B} \in \mathfrak{B}_n$ we define $\mathcal{T}_L^{\mathbf{B}}$ as the union of $\mathcal{T}_0$ and the concept inclusions defined above with $\mathbf{B}$ replacing M.

Then for any sequence $\boldsymbol{\sigma}$ of length n there exists at most one $L \in \mathfrak{L}_n$, at most one $1 \leq i \leq n$ and at most one $\mathbf{B} \in \mathfrak{B}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models A_i \sqsubseteq \exists \boldsymbol{\sigma}.\mathbf{B}$ and $\mathcal{T}_L^{\mathbf{B}} \models$

$B_i \sqsubseteq \exists \boldsymbol{\sigma}.\mathbf{B}$. Notice that the size of each $\mathcal{T}_L^{\mathbf{B}}$ is polynomial in n and so $\mathfrak{L}_n$ contains superpolynomially many n-tuples in the size of each $\mathcal{T}_L^{\mathbf{B}}$, with $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$. Every $\mathcal{T}_L^{\mathbf{B}}$ entails, among other inclusions, $\bigsqcap_{i=1}^{n} C_i \sqsubseteq A$, where every C_i is either A_i or B_i. Let Σ_n be the signature of the TBoxes $\mathcal{T}_L^{\mathbf{B}}$ and consider a TBox $\mathcal{T}^*$ defined as the following set of concept inclusions:

$$\exists r.(E_1 \sqcap \bar{E}_1) \sqsubseteq (E_1 \sqcap \bar{E}_1),\ (E_1 \sqcap \bar{E}_1) \sqsubseteq \exists r.(E_1 \sqcap \bar{E}_1),$$
$$\exists s.(E_1 \sqcap \bar{E}_1) \sqsubseteq (E_1 \sqcap \bar{E}_1),\ (E_1 \sqcap \bar{E}_1) \sqsubseteq \exists s.(E_1 \sqcap \bar{E}_1),$$
$$(E_i \sqcap \bar{E}_i) \sqsubseteq A \quad \text{for every } 1 \leq i \leq n \text{ and every } A \in \Sigma_n \cap \mathsf{N_C}$$

The basic idea of extending our TBoxes with $\mathcal{T}^*$ is that if $a \in (E_i \sqcap \bar{E}_i)^{\mathcal{I}_\mathcal{A}}$, for an ABox $\mathcal{A}$ and individual $a \in \mathsf{Ind}(\mathcal{A})$, then for all $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$, we have $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models D(b)$, where D is any $\mathcal{EL}$ concept expression over Σ_n and $b \in \mathsf{Ind}(\mathcal{A})$ is any successor or predecessor of a (or a itself). This means that for each individual in $\mathcal{A}$ at most one $\mathbf{B}$ of the 2^n binary strings in $\mathfrak{B}_n$ can be distinguished by data retrieval queries. The following lemma enables us to respond to membership queries without eliminating too many $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$ used to encode $\mathcal{T}_L^{\mathbf{B}}$ in the set of TBoxes that the learner cannot distinguish.

Lemma 7. *For any ABox $\mathcal{A}$, any $\mathcal{EL}$ concept assertion $D(a)$ over Σ_n, and any $a \in \mathsf{Ind}(\mathcal{A})$, if there is $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$ such that $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$ then:*

- *either $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$, for every $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$, or*
- *$(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$ for at most $|D|$ elements $L \in \mathfrak{L}_n$, or*
- *$(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$ for at most $|\mathcal{A}|$ elements $\mathbf{B} \in \mathfrak{B}_n$.*

The next lemma (proved in Appendix E) is immediate from Lemma 15 presented in [12]. It shows how the oracle can answer equivalence queries eliminating at most one $L \in \mathfrak{L}_n$ used to encode $\mathcal{T}_L^{\mathbf{B}}$ in the set S of TBoxes that the learner cannot distinguish.

Lemma 8. *For any $n > 1$ and any $\mathcal{EL}$ TBox $\mathcal{H}$ in Σ_n with $|\mathcal{H}| < 2^n$, there exists an ABox $\mathcal{A}$, an individual $a \in \mathsf{Ind}(\mathcal{A})$ and an $\mathcal{EL}$ concept expression D over Σ_n such that (i) the size of $\mathcal{A}$ plus the size of D does not exceed $6n$ and (ii) if $(\mathcal{H}, \mathcal{A}) \models D(a)$ then $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models D(a)$ for at most one $L \in \mathfrak{L}_n$ and if $(\mathcal{H}, \mathcal{A}) \not\models D(a)$ then for every $L \in \mathfrak{L}_n$ we have $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$.*

Then, by Lemmas 7 and 8, we have that: (i) any polynomial size membership query can distinguish at most polynomially many TBoxes from S; and (ii) if the learner's hypothesis is polynomial size then there exists a polynomial size counterexample that the oracle can give which distinguishes at most polynomially many TBoxes from S.

Theorem 5. *The $\mathcal{EL}$ data retrieval framework is not polynomially exact learnable.*

5 Future Work

We plan to consider an extension of the learning protocol in which arbitrary conjunctive queries are admitted in queries to the domain expert/oracle. We then still have polynomial time learnability for $\mathcal{EL}_{\mathsf{lhs}}$ but conjecture non-polynomial time learnability for DL-Lite$_{\mathcal{R}}^{\exists}$. Another extension is exact learnability for the Horn-extension of DL-Lite$_{\mathcal{R}}^{\exists}$ for which we conjecture that polynomial time learnability still holds.

Bibliography

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

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

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

[4] S. Bechhofer, I. Horrocks, C. Goble, and R. Stevens. Oiled: a reason-able ontology editor for the semantic web. In *KI 2001: Advances in Artificial Intelligence*, pages 396–408. Springer, 2001.

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

[6] P. Buitelaar, P. Cimiano, and B. Magnini, editors. *Ontology Learning from Text: Methods, Evaluation and Applications*. IOS Press, 2005.

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

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

[9] J. Day-Richter, M. A. Harris, M. Haendel, S. Lewis, et al. Obo-edit an ontology editor for biologists. *Bioinformatics*, 23(16):2198–2200, 2007.

[10] M. Frazier and L. Pitt. Learning From Entailment: An Application to Propositional Horn Sentences. In *ICML*, pages 120–127, 1993.

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

[12] B. Konev, C. Lutz, A. Ozaki, and F. Wolter. Exact learning of lightweight description logic ontologies. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014*, 2014.

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

[14] Y. Ma and F. Distel. Learning formal definitions for snomed CT from text. In *Artificial Intelligence in Medicine - 14th Conference on Artificial Intelligence in Medicine, AIME 2013, Murcia, Spain, May 29 - June 1, 2013. Proceedings*, pages 73–77, 2013.

[15] M. A. Musen. Protégé ontology editor. *Encyclopedia of Systems Biology*, pages 1763–1765, 2013.

[16] C. Reddy and P. Tadepalli. Learning First-Order Acyclic Horn Programs from Entailment. In *in Proceedings of the 15th International Conference on Machine Learning; (and Proceedings of the 8th International Conference on Inductive Logic Programming*, pages 23–37. Morgan Kaufmann, 1998.

[17] H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*. Springer, 2009.

[18] H. Wang, M. Horridge, A. Rector, N. Drummond, and J. Seidenberg. Debugging OWL-DL ontologies: A heuristic approach. In *The Semantic Web–ISWC 2005*, pages 745–757. Springer, 2005.

A Simulations and Canonical Models

The semantics of DL-Lite$_{\mathcal{R}}^{\exists}$ and $\mathcal{EL}_{\mathsf{lhs}}$ is given by interpretations. An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty set $\Delta^{\mathcal{I}}$ and a function $\cdot^{\mathcal{I}}$ that assigns each concept name A to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and each role name r to a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. We make the unique name assumption for the individuals of an ABox. To interpret an ABox $\mathcal{A}$, we consider interpretations $\mathcal{I}$ which assign to each $a \in \mathsf{Ind}(\mathcal{A})$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies an assertion $A(a) \in \mathcal{A}$ if $a^{\mathcal{I}} \in A^{\mathcal{I}}$ and an assertion $r(a,b) \in \mathcal{A}$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$. The *extension* $C^{\mathcal{I}}$ of an $\mathcal{ELI}$ concept expression C is inductively defined as follows:

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(\exists r.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists e \in C^{\mathcal{I}} \ : \ (d,e) \in r^{\mathcal{I}}\}$
- $(\exists r^{-}.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists e \in C^{\mathcal{I}} \ : \ (e,d) \in r^{\mathcal{I}}\}$

An interpretation $\mathcal{I}$ *satisfies*:

- a concept inclusion $C \sqsubseteq D$, in symbols $\mathcal{I} \models C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$;
- a role inclusion $r \sqsubseteq s$, in symbols $\mathcal{I} \models r \sqsubseteq s$, if $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$;
- an instance assertion $C(a)$, in symbols $\mathcal{I} \models C(a)$, if $a^{\mathcal{I}} \in C^{\mathcal{I}}$;
- a role assertion $r(a,b)$, in symbols $\mathcal{I} \models r(a,b)$, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$.

We say that an interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ (an ABox $\mathcal{A}$) if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{T}$ ($\alpha \in \mathcal{A}$). A CI (a RI) α *follows from* a TBox $\mathcal{T}$ if every model of $\mathcal{T}$ is a model of α, in symbols $\mathcal{T} \models \alpha$. $\models \alpha$ is used to denote that α follows from the empty TBox. A knowledge base (KB) is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ consisting of a TBox and an ABox. An instance assertion follows from $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if every individual name that occurs in α also occurs in $\mathsf{Ind}(\mathcal{A})$ and every model of $(\mathcal{T}, \mathcal{A})$ is a model of α, in symbols $(\mathcal{T}, \mathcal{A}) \models \alpha$. For DL-Lite$_{\mathcal{R}}^{\exists}$ KBs, we assume that the ABox $\mathcal{A}$ is closed under inverses, i.e., $r(a,b) \in \mathcal{A}$ iff $r^{-}(b,a) \in \mathcal{A}$. If the add or remove assertions from an ABox $\mathcal{A}$, we silently do this assuming that the resulting $\mathcal{A}$ is again closed under inverses. For $\mathcal{EL}_{\mathsf{lhs}}$ KBs, role assertions in $\mathcal{A}$ do not have inverse roles and we do not make $\mathcal{A}$ closed under inverses.

Homomorphisms and Simulations A *path* in an $\mathcal{ELI}$ concept expression C is a finite sequence $C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_k \cdot C_k$, where $C_0 = C$, $k \geq 0$, and $\exists r_{i+1}.C_{i+1}$ is a conjunct of C_i, for $0 \leq i < k$. The set $\mathsf{paths}(C)$ contains all paths in C. We also define $\mathsf{tail}(p) = \{A \mid A \text{ is a conjunct of } C_k\}$, where C_k is the last concept expression in path p.

Definition 1. *The tree interpretation $\mathcal{I}_C$ of C is defined as follows:*

- $\Delta^{\mathcal{I}_C} = \{p = C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_k \cdot C_k \mid p \in \mathsf{paths}(C)\ \}$
- $A^{\mathcal{I}_C} = \{p \mid A \in \mathsf{tail}(p)\}$
- $r^{\mathcal{I}_C} = \{(p, p') \mid p' = p \cdot r \cdot D\}$

The next two lemmas relate our tree shaped interpretations with homomorphisms. A homomorphism $h : \mathcal{I}_C \rightarrow \mathcal{I}$ between the tree interpretation of an $\mathcal{ELI}$ concept expression C and an interpretation $\mathcal{I}$ is defined as follows, for $p, p' \in \Delta^{\mathcal{I}_C}$:

- if $p \in A^{\mathcal{I}_C}$ then $h(p) \in A^{\mathcal{I}}$;
- if $(p, p') \in r^{\mathcal{I}_C}$ then $(h(p), h(p')) \in r^{\mathcal{I}}$.

Lemma 9. *Let C be an $\mathcal{ELI}$ concept expression. Denote as $\mathcal{I}_C$ the tree shaped interpretation of C. If there is a homomorphism $h : \mathcal{I}_C \rightarrow \mathcal{I}$, such that $h(\rho_C) = d$, where ρ_C is the root path of $\mathcal{I}_C$, then $d \in C^{\mathcal{I}}$.*

Proof. In the base case let C be a concept name A. Since $\rho_A \in A^{\mathcal{I}_A}$, if there is a homomorphism $h : \mathcal{I}_A \rightarrow \mathcal{I}$, such that $h(\rho_A) = d$, then $d \in A^{\mathcal{I}}$. Now we make a case distinction:

- For $C = C_1 \sqcap C_2$: Suppose the lemma is true for a homomorphism $h_1 : \mathcal{I}_{C_1} \rightarrow \mathcal{I}$ with $h_1(\rho_{C_1}) = d$ and a homomorphism $h_2 : \mathcal{I}_{C_2} \rightarrow \mathcal{I}$ with $h_2(\rho_{C_2}) = d$. Then $d \in C_1^{\mathcal{I}}$ and $d \in C_2^{\mathcal{I}}$. By semantics of $\cap$, $d \in (C_1 \sqcap C_2)^{\mathcal{I}}$.
- For $C = \exists r.C'$: We know that $\rho \in (\exists r.C')^{\mathcal{I}_C}$. By semantics of $\exists$, there is d such that $(\rho, d) \in r^{\mathcal{I}_C}$ and $d \in C'^{\mathcal{I}_C}$. If there is a homomorphism $h : \mathcal{I}_C \rightarrow \mathcal{I}$ then $(h(\rho), h(d)) \in r^{\mathcal{I}}$. Suppose the lemma is true for C'. Then, $h(d) \in C'^{\mathcal{I}}$ and we have that $h(\rho) \in (\exists r.C')^{\mathcal{I}}$.
- For $C = \exists r^-.C'$: We know that $\rho \in (\exists r^-.C')^{\mathcal{I}_C}$. By semantics of $\exists$, there is d such that $(d, \rho) \in r^{\mathcal{I}_C}$ and $d \in C'^{\mathcal{I}_C}$. If there is a homomorphism $h : \mathcal{I}_C \rightarrow \mathcal{I}$ then $(h(d), h(\rho)) \in r^{\mathcal{I}}$. Suppose the lemma is true for C'. Then, $h(d) \in C'^{\mathcal{I}}$ and we have that $h(\rho) \in (\exists r^-.C')^{\mathcal{I}}$.

❑

Lemma 10. *Let C be an $\mathcal{ELI}$ concept expression. Denote as $\mathcal{I}_C$ the tree shaped interpretation of C. If $d \in C^{\mathcal{I}}$ then there is a homomorphism $h : \mathcal{I}_C \rightarrow \mathcal{I}$, such that $h(\rho_C) = d$, where ρ_C is the root path of $\mathcal{I}_C$.*

Proof. In the base case let C be a concept name A. If $d \in A^{\mathcal{I}}$ then, since $\rho_A \in \mathcal{I}_A$, we have that $h(\rho_A) = d$ is a homomorphism $h : \mathcal{I}_A \rightarrow \mathcal{I}$. Now we make a case distinction:

– For $C = C_1 \sqcap C_2$: If $d \in (C_1 \sqcap C_2)^{\mathcal{I}}$ then, by semantics of $\cap$, $d \in C_1^{\mathcal{I}}$ and $d \in C_2^{\mathcal{I}}$. Suppose the lemma is true for $d_1 \in C_1^{\mathcal{I}}$ and $d_2 \in C_2^{\mathcal{I}}$. Then there is a homomorphism $h_1 : \mathcal{I}_{C_1} \to \mathcal{I}$ with $h_1(\rho_{C_1}) = d_1$ and a homomorphism $h_2 : \mathcal{I}_{C_2} \to \mathcal{I}$ with $h_2(\rho_{C_2}) = d_2$. Now, we need to show that the lemma is also true for $C = C_1 \sqcap C_2$. For $p \in \Delta^{\mathcal{I}_C}$, we define h as follows:

$$h(p) = \begin{cases} d & \text{if } p = \rho, \text{ where } \rho \text{ is the root of } \mathcal{I}_C \\ h_1(p') & \text{if } p = p', \text{ where } p' \in \Delta^{\mathcal{I}_{C_1}} \\ h_2(p') & \text{if } p = p', \text{ where } p' \in \Delta^{\mathcal{I}_{C_2}} \end{cases}$$

We know that $\rho \in C^{\mathcal{I}_C}$. If $d \in C^{\mathcal{I}}$ then $h(\rho) = d$ is a partial homomorphism. Since $C = C_1 \sqcap C_2$, we have that for all $p \in \Delta^{\mathcal{I}_C} \setminus \{\rho\}$, $p \in \mathsf{paths}(C_1)$ or $p \in \mathsf{paths}(C_2)$. By hypothesis h_1 and h_2 are homomorphisms, so $h : \mathcal{I}_C \to \mathcal{I}$ is also a homomorphism.

– For $C = \exists r.C'$: If $d \in (\exists r.C')^{\mathcal{I}}$ then, by semantics of $\exists$, there is a d' such that $(d, d') \in r^{\mathcal{I}}$ and $d' \in C'^{\mathcal{I}}$. Suppose the lemma is true for $d' \in C'^{\mathcal{I}}$. Then there is a homomorphism $h' : \mathcal{I}_{C'} \to \mathcal{I}$ with $h'(\rho_{C'}) = d'$. Now, we need to show that the lemma is also true for $C = \exists r.C'$. For $p \in \Delta^{\mathcal{I}_C}$, we define h as follows:

$$h(p) = \begin{cases} d & \text{if } p = \rho, \text{ where } \rho \text{ is the root of } \mathcal{I}_C \\ h'(p') & \text{if } p = \rho \cdot r \cdot p' \text{ and } p' \in \Delta^{\mathcal{I}_{C'}} \end{cases}$$

We know that $\rho \in C^{\mathcal{I}_C}$. If $d \in C^{\mathcal{I}}$ then $h(\rho) = d$ is a partial homomorphism. Since $C = \exists r.C'$, we have that for all $p \in \Delta^{\mathcal{I}_C} \setminus \{\rho\}$, $p = \rho \cdot r \cdot p'$ with $p' \in \mathsf{paths}(C')$. By hypothesis h' is a homomorphism, so $h : \mathcal{I}_C \to \mathcal{I}$ is also a homomorphism.

– For $C = \exists r^-.C'$: If $d \in (\exists r^-.C')^{\mathcal{I}}$ then, by semantics of $\exists$, there is a d' such that $(d', d) \in r^{\mathcal{I}}$ and $d' \in C'^{\mathcal{I}}$. Suppose the lemma is true for $d' \in C'^{\mathcal{I}}$. Then there is a homomorphism $h' : \mathcal{I}_{C'} \to \mathcal{I}$ with $h'(\rho_{C'}) = d'$. Now, we need to show that the lemma is also true for $C = \exists r^-.C'$. For $p \in \Delta^{\mathcal{I}_C}$, we define h as follows:

$$h(p) = \begin{cases} d & \text{if } p = \rho, \text{ where } \rho \text{ is the root of } \mathcal{I}_C \\ h'(p') & \text{if } p = \rho \cdot r^- \cdot p' \text{ and } p' \in \Delta^{\mathcal{I}_{C'}} \end{cases}$$

We know that $\rho \in C^{\mathcal{I}_C}$. If $d \in C^{\mathcal{I}}$ then $h(\rho) = d$ is a partial homomorphism. Since $C = \exists r^-.C'$, we have that for all $p \in \Delta^{\mathcal{I}_C} \setminus \{\rho\}$, $p = \rho \cdot r^- \cdot p'$ with $p' \in \mathsf{paths}(C')$. By hypothesis h' is a homomorphism, so $h : \mathcal{I}_C \to \mathcal{I}$ is also a homomorphism.

❑

Let $\mathcal{I}, \mathcal{J}$ be interpretations and Σ a signature, $d \in \Delta^{\mathcal{I}}$ and $e \in \Delta^{\mathcal{J}}$. A relation $S \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{J}}$ is a *Σ-simulation* from $(\mathcal{I}, d)$ to $(\mathcal{J}, e)$ if the following conditions are satisfied:

– for all concept names $A \in \Sigma$ and all $(d, e) \in S$, if $d \in A^{\mathcal{I}}$ then $e \in A^{\mathcal{J}}$;
– for all role names $r \in \Sigma$, all $(d, e) \in S$ and all $d' \in \Delta^{\mathcal{I}}$, if $(d, d') \in r^{\mathcal{I}}$ then there exists $e' \in \Delta^{\mathcal{J}}$ such that $(e, e') \in r^{\mathcal{J}}$ and $(d', e') \in S$.

We write $(\mathcal{I}, d) \Rightarrow_{\Sigma} (\mathcal{J}, e)$ if there is a Σ-simulation from $(\mathcal{I}, d)$ to $(\mathcal{J}, e)$. If $\Sigma = \mathsf{N_C} \cup \mathsf{N_R}$ we omit Σ and speak of a simulation $(\mathcal{I}, d) \Rightarrow (\mathcal{J}, e)$. Simulations preserve the membership of $\mathcal{EL}$ concept expressions [13]:

Lemma 11. *For all $\mathcal{EL}$ concept expressions C: if $d \in C^{\mathcal{I}}$ and $(\mathcal{I}, d) \Rightarrow (\mathcal{J}, e)$, then $e \in C^{\mathcal{J}}$.*

The Canonical Model of an ABox The canonical models of DL-Lite$_{\mathcal{R}}^{\exists}$ and $\mathcal{EL}_{\mathsf{lhs}}$ knowledge bases is given by the pair $(\mathcal{T}, \mathcal{A})$, where $\mathcal{A}$ is an ABox and $\mathcal{T}$ is a TBox with DL-Lite$_{\mathcal{R}}^{\exists}$ and $\mathcal{EL}_{\mathsf{lhs}}$ CIs, respectively. Before we proceed with knowledge bases, we present the canonical model of an ABox.

Definition 2. *The canonical model $\mathcal{I}_{\mathcal{A}} = (\Delta^{\mathcal{I}_{\mathcal{A}}}, \cdot^{\mathcal{I}_{\mathcal{A}}})$ of an ABox $\mathcal{A}$ is defined as follows:*

- $\Delta^{\mathcal{I}_{\mathcal{A}}} = \{a \mid a \in \mathsf{Ind}(\mathcal{A})\}$
- $A^{\mathcal{I}_{\mathcal{A}}} = \{a \mid A(a) \in \mathcal{A}, A \in \mathsf{N_C}\}$
- $r^{\mathcal{I}_{\mathcal{A}}} = \{(a, b) \mid r(a, b) \in \mathcal{A}, r \in \mathsf{N_R}\}$

A.1 The Canonical Model for DL-Lite$_{\mathcal{R}}^{\exists}$

The canonical model $\mathcal{I}_{\mathcal{T},\mathcal{A}} = \bigcup \mathcal{I}_{n \geq 0}$ of a DL-Lite$_{\mathcal{R}}^{\exists}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is defined by a sequence of interpretations $\mathcal{I}_n$. Let $\mathcal{I}_{\mathcal{A}}$ be the canonical interpretation of $\mathcal{A}$. In the base case, $\mathcal{I}_0 := \mathcal{I}_{\mathcal{A}}$. The domain $\Delta^{\mathcal{I}_n}$ contains, in addition to $\Delta^{\mathcal{I}_0}$, sequences $a_0 \cdot r_0 \cdot C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_m \cdot C_m$, where $a_0 \in \mathsf{Ind}(\mathcal{A})$. For $p = a_0 \cdot r_0 \cdot C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_m \cdot C_m$ and $q = C'_0 \cdot r'_1 \cdot C'_1 \cdot ... \cdot r'_{m'} \cdot C'_{m'}$ we define $p \cdot s \cdot q = a_0 \cdot r_0 \cdot C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_m \cdot C_m \cdot s \cdot C'_0 \cdot r'_1 \cdot C'_1 \cdot ... \cdot r'_{m'} \cdot C'_{m'}$, that is, the concatenation of p and q through the role s. Assume $\mathcal{I}_n$ has been defined. Let $k \leq n$ be minimal such that there is a $p \in \Delta^{\mathcal{I}_k}$ with $p \in B^{\mathcal{I}_k}$, $B \sqsubseteq D \in \mathcal{T}$ but $p \notin D^{\mathcal{I}_k}$. Let $D_0, D_1, ..., D_l$ be the conjuncts of D of the form $D_i = \exists s_i.E_i$, $0 \leq i \leq l$. Now, we define $\mathcal{I}_{n+1}$ as follows:

- $\Delta^{\mathcal{I}_{n+1}} = \Delta^{\mathcal{I}_n} \cup \{p \cdot s_i \cdot q \mid q \in \mathsf{paths}(E_i), 0 \leq i \leq l\}$;
- $A^{\mathcal{I}_{n+1}} = A^{\mathcal{I}_n} \cup \{p \cdot s_i \cdot q \mid A \in \mathsf{tail}(q), 0 \leq i \leq l\} \cup \{p \mid \text{if } A \text{ is a conjunct of } D\}$;
- $r^{\mathcal{I}_{n+1}} = r^{\mathcal{I}_n} \cup \{(p \cdot s_i \cdot q, p \cdot s_i \cdot q') \mid (q, q') \in r'^{\mathcal{I}_{E_i}}, \mathcal{T} \models r' \sqsubseteq r, 0 \leq i \leq l\}$ $\cup \{(p, p \cdot s_i \cdot E_i) \mid \mathcal{T} \models s_i \sqsubseteq r, 0 \leq i \leq l\}$.

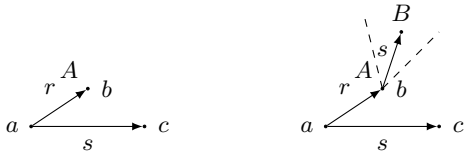
(The canonical model $\mathcal{I}_{\mathcal{T},C} = \bigcup \mathcal{I}_{n \geq 0}$ of an $\mathcal{ELI}$ concept expression C and a DL-Lite$_{\mathcal{R}}^{\exists}$ TBox $\mathcal{T}$ is defined analogously with $\mathcal{I}_0 = \mathcal{I}_C$, where $\mathcal{I}_C$ is the tree interpretation of C.)

As an example let $\mathcal{A} = \{r(a, b), A(b), s(a, c)\}$ and $\mathcal{T} = \{A \sqsubseteq \exists s.B\}$. Figures 3a and 3b shows the interpretations $\mathcal{I}_{\mathcal{A}}$ and $\mathcal{I}_{\mathcal{T},\mathcal{A}}$, respectively.

It follows from Lemma 13 that the sequence of interpretations which define $\mathcal{I}_{\mathcal{T},\mathcal{A}} = \bigcup \mathcal{I}_{n \geq 0}$ is the canonical model of a DL-Lite$_{\mathcal{R}}^{\exists}$ knowledge base $(\mathcal{T}, \mathcal{A})$. This lemma requires a technical lemma, presented below, which states that there is a homomorphism from $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ to an arbitrary model of $(\mathcal{T}, \mathcal{A})$.

Lemma 12. *Let $\mathcal{J}$ be a model of $(\mathcal{T}, \mathcal{A})$. Then there exists a homomorphism $h : \mathcal{I}_{\mathcal{T},\mathcal{A}} \to \mathcal{J}$ mapping $h(a) = a^{\mathcal{J}}$ for all $a \in \mathsf{Ind}(\mathcal{A})$ with the following properties:*

- *if $p \in A^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$ then $h(p) \in A^{\mathcal{J}}$;*
- *if $(p, p') \in r^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$ then $(h(p), h(p')) \in r^{\mathcal{J}}$, where $p, p' \in \Delta^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$.*



(a) The canonical model $\mathcal{I}_{\mathcal{A}}$ (b) The canonical model $\mathcal{I}_{\mathcal{T},\mathcal{A}}$

Fig. 3: Canonical Models with $\mathcal{A} = \{r(a,b), A(b), s(a,c)\}$ and $\mathcal{T} = \{A \sqsubseteq \exists s.B\}$

Proof. The proof is by induction on the sequence of interpretations for the canonical model $\mathcal{I}_{\mathcal{T},\mathcal{A}}$. We define $h = \bigcup_{n \geq 0} h_n$ and set $h_0 : \Delta^{\mathcal{I}_0} \rightarrow \Delta^{\mathcal{J}}$ with $h_0(a) = a^{\mathcal{J}}$ for $a \in \Delta^{\mathcal{I}_0}$ and $a^{\mathcal{J}} \in \Delta^{\mathcal{J}}$. By definition of $\mathcal{I}_0$, $A(a) \in \mathcal{A}$ iff $a \in A^{\mathcal{I}_0}$ and $r(a,b) \in \mathcal{A}$ iff $(a,b) \in r^{\mathcal{I}_0}$. Since $\mathcal{J}$ is a model of $(\mathcal{T},\mathcal{A})$, if $A(a) \in \mathcal{A}$ then $a \in A^{\mathcal{J}}$. Similarly, if $r(a,b) \in \mathcal{A}$ then $(a,b) \in r^{\mathcal{J}}$. So $a \in A^{\mathcal{I}_0}$ implies $h_0(a) \in A^{\mathcal{J}}$. Also, $(a,b) \in r^{\mathcal{I}_0}$ implies $(h_0(a), h_0(b)) \in r^{\mathcal{J}}$. Thus, h_0 is a homomorphism.

Suppose it was proven that $h_n : \mathcal{I}_n \rightarrow J$ is a homomorphism. Let $k \leq n$ be minimal such that there is a $p \in \Delta^{\mathcal{I}_k}$ with $p \in B^{\mathcal{I}_k}$, $B \sqsubseteq D \in \mathcal{T}$ but $p \notin D^{\mathcal{I}_k}$. As described above, for every conjunct D_i of $D = D_0 \sqcap ... \sqcap D_l$, $0 \leq i \leq l$, $\mathcal{I}_{n+1}$ is defined as:

1. if D_i is a concept name A then $\mathcal{I}_{n+1}$ is defined in the same way as $\mathcal{I}_n$ except that $A^{\mathcal{I}_{n+1}} = A^{\mathcal{I}_n} \cup \{p\}$. Since $\mathcal{J}$ is a model of $(\mathcal{T},\mathcal{A})$, if $h_n(p) \in B^{\mathcal{J}}$ and $B \sqsubseteq D \in \mathcal{T}$ then $h_n(p) \in D^{\mathcal{J}}$. Since $D_i = A$ is a conjunct of D, $h_n(p) \in A^{\mathcal{J}}$. So $h_{n+1} = h_n$ is a homomorphism.
2. otherwise D_i is of the form $\exists s.E$, where s can be an inverse role. Then a copy of the tree shaped interpretation $\mathcal{I}_E$ of the concept expression E connected by the role s is added to $\mathcal{I}_{n+1}$ following the way described above. By hypothesis, $h_n(p) \in B^{\mathcal{J}}$. Since $\mathcal{J}$ is a model of $(\mathcal{T},\mathcal{A})$, if $B \sqsubseteq D \in \mathcal{T}$ then $h_n(p) \in D^{\mathcal{J}}$. Since $D_i = \exists s.E$ is a conjunct of D, $h_n(p) \in (\exists s.E)^{\mathcal{J}}$. By semantics of $\exists$,
   - $(*)$ there is $d \in \Delta^{\mathcal{J}}$ such that $(h_n(p), d) \in s^{\mathcal{J}}$ and $d \in E^{\mathcal{J}}$.

   By Lemma 10, if $d \in E^{\mathcal{J}}$ then there is a homomorphism $h' : \mathcal{I}_E \rightarrow \mathcal{J}$ mapping ρ_E to d, where $\mathcal{I}_E$ is the tree interpretation of E rooted in ρ_E. Now, for every $p \cdot s \cdot q \in \Delta^{\mathcal{I}_{n+1}}$ such that $q \in \mathsf{paths}(E)$ we define $h_{n+1}(p \cdot s \cdot q) = h'(q)$. We want to show that h_{n+1} is a homomorphism.

   By definition of $\mathcal{I}_{n+1}$, $p \cdot s \cdot q \in A^{\mathcal{I}_{n+1}}$ iff $q \in A^{\mathcal{I}_E}$. If $q \in A^{\mathcal{I}_E}$ then (since $h' : \mathcal{I}_E \rightarrow \mathcal{J}$ is a homomorphism) $h'(q) \in A^{\mathcal{J}}$. So $p \cdot s \cdot q \in A^{\mathcal{I}_{n+1}}$ implies $h_{n+1}(p \cdot s \cdot q) = h'(q) \in A^{\mathcal{J}}$. Also, by definition of $\mathcal{I}_{n+1}$, $(p \cdot s \cdot q, p \cdot s \cdot q') \in r^{\mathcal{I}_{n+1}}$ iff $(q, q') \in r'^{\mathcal{I}_E}$ and $\mathcal{T} \models r' \sqsubseteq r$. If $(q, q') \in r'^{\mathcal{I}_E}$ then $(h'(q), h'(q')) \in r'^{\mathcal{J}}$. Since $\mathcal{J}$ is a model of $(\mathcal{T},\mathcal{A})$, if $\mathcal{T} \models r' \sqsubseteq r$ then $(h'(q), h'(q')) \in r^{\mathcal{J}}$. As $h_{n+1}(p \cdot s \cdot q) = h'(q)$ and $h_{n+1}(p \cdot s \cdot q') = h'(q')$, if $(p \cdot s \cdot q, p \cdot s \cdot q') \in r^{\mathcal{I}_{n+1}}$ then $(h_{n+1}(p \cdot s \cdot q), h_{n+1}(p \cdot s \cdot q')) \in r^{\mathcal{J}}$. Finally, notice that by definition of $\mathcal{I}_{n+1}$, $t^{\mathcal{I}_{n+1}} = t^{\mathcal{I}_n} \cup \{(p, p \cdot s \cdot E) \mid \mathcal{T} \models s \sqsubseteq t\}$. By $(*)$, we have that $(h_n(p), d) \in s^{\mathcal{J}}$. Since $\mathcal{J}$ is a model of $(\mathcal{T},\mathcal{A})$, if $\mathcal{T} \models s \sqsubseteq t$ then $(h_n(p), d) \in t^{\mathcal{J}}$. By definition of $h' : \mathcal{I}_E \rightarrow \mathcal{J}$, $h'(\rho_E) = d$. So $h_{n+1}(p \cdot s \cdot E) = d$. Then $(p, p \cdot s \cdot E) \in t^{\mathcal{I}_{n+1}}$ implies $(h_n(p), h_{n+1}(p \cdot s \cdot E)) \in t^{\mathcal{J}}$.

Thus, $h_{n+1} : \mathcal{I}_{n+1} \to \mathcal{J}$ is a homomorphism. Since $\mathcal{I}_{\mathcal{T},\mathcal{A}} = \bigcup \mathcal{I}_{n\geq 0}$, there exists a homomorphism $h : \mathcal{I}_{\mathcal{T},\mathcal{A}} \to \mathcal{J}$. ❑

Lemma 13. *Let α be an instance assertion, $(\mathcal{T}, \mathcal{A})$ a DL-Lite$^{\exists}_{\mathcal{R}}$ knowledge base and $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ defined as above. Then, $\mathcal{I}_{\mathcal{T},\mathcal{A}} \models \alpha$ if and only if $(\mathcal{T}, \mathcal{A}) \models \alpha$.*

Proof. ($\Rightarrow$) We make a case distinction.

- α is of the form $C(a)$: Let $\mathcal{J}$ be a model of $(\mathcal{T}, \mathcal{A})$. If $\mathcal{I}_{\mathcal{T},\mathcal{A}} \models C(a)$, that is, $a \in C^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$, then by Lemma 10 there is a homomorphism $h : \mathcal{I}_C \to \mathcal{I}_{\mathcal{T},\mathcal{A}}$ mapping ρ_C to a. By Lemma 12, there is a homomorphism $g : \mathcal{I}_{\mathcal{T},\mathcal{A}} \to \mathcal{J}$ mapping a to $a^{\mathcal{J}}$. Then there is $g \circ h : \mathcal{I}_C \to \mathcal{J}$, mapping ρ_C to $a^{\mathcal{J}}$. This means that (Lemma 10) $a^{\mathcal{J}} \in C^{\mathcal{J}}$. Since $\mathcal{J}$ is an arbitrary model of $(\mathcal{T}, \mathcal{A})$, $C(a)$ holds in every model of $(\mathcal{T}, \mathcal{A})$. That is, $(\mathcal{T}, \mathcal{A}) \models C(a)$.
- α is of the form $r(a, b)$: Let $\mathcal{J}$ be a model of $(\mathcal{T}, \mathcal{A})$. By Lemma 12, there is a homomorphism $h : \mathcal{I}_{\mathcal{T},\mathcal{A}} \to \mathcal{J}$ mapping $a^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$ to $a^{\mathcal{J}}$ and $b^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$ to $b^{\mathcal{J}}$. If $\mathcal{I}_{\mathcal{T},\mathcal{A}} \models r(a, b)$, that is, $(a, b) \in r^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$, then $(h(a), h(b)) \in r^{\mathcal{J}}$. Since $\mathcal{J}$ is an arbitrary model of $(\mathcal{T}, \mathcal{A})$, $r(a, b)$ holds in every model of $(\mathcal{T}, \mathcal{A})$. That is, $(\mathcal{T}, \mathcal{A}) \models r(a, b)$.

($\Leftarrow$) We show that $\mathcal{I}_{\mathcal{T},\mathcal{A}} \models (\mathcal{T}, \mathcal{A})$. By definition of $\mathcal{I}_0$, $a \in A^{\mathcal{I}_0}$ iff $A(a) \in \mathcal{A}$ and $a \in (\exists r.\top)^{\mathcal{I}_0}$ iff $r(a, b) \in \mathcal{A}$, where r can be inverse. So if $\mathcal{A} \models B(a)$ then $a \in B^{\mathcal{I}_0}$. For inclusions $B' \sqsubseteq D \in \mathcal{T}$, where $(\mathcal{T}, \mathcal{A}) \models B'(a)$, we suppose $a \in B'^{\mathcal{I}_n}$ and, by definition of $\mathcal{I}_{n+1}$, we have that $a \in D^{\mathcal{I}_{n+1}}$. Now since $\mathcal{I}_{\mathcal{T},\mathcal{A}} = \bigcup \mathcal{I}_{n\geq 0}$ is a model of $(\mathcal{T}, \mathcal{A})$, $(\mathcal{T}, \mathcal{A}) \models C(a)$ implies $\mathcal{I}_{\mathcal{T},\mathcal{A}} \models C(a)$. ❑

A.2 The Canonical Model for $\mathcal{EL}$

The canonical model $\mathcal{I}_{\mathcal{T},\mathcal{A}} = \bigcup \mathcal{I}_{n\geq 0}$ of an $\mathcal{EL}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is defined by a sequence of interpretations $\mathcal{I}_n$. Let $\mathcal{I}_{\mathcal{A}}$ be the canonical interpretation of $\mathcal{A}$. In the base case, $\mathcal{I}_0 := \mathcal{I}_{\mathcal{A}}$. The domain $\Delta^{\mathcal{I}_n}$ contains, in addition to $\Delta^{\mathcal{I}_0}$, sequences $a_0 \cdot r_0 \cdot C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_m \cdot C_m$, where $a_0 \in \mathsf{Ind}(\mathcal{A})$. For $p = a_0 \cdot r_0 \cdot C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_m \cdot C_m$ and $q = C'_0 \cdot r'_1 \cdot C'_1 \cdot ... \cdot r'_{m'} \cdot C'_{m'}$ we define $p \cdot s \cdot q = a_0 \cdot r_0 \cdot C_0 \cdot r_1 \cdot C_1 \cdot ... \cdot r_m \cdot C_m \cdot s \cdot C'_0 \cdot r'_1 \cdot C'_1 \cdot ... \cdot r'_{m'} \cdot C'_{m'}$, that is, the concatenation of p and q through the role s. Assume $\mathcal{I}_n$ has been defined. Let $k \leq n$ be minimal such that there is a $p \in \Delta^{\mathcal{I}_k}$ with $p \in C^{\mathcal{I}_k}$, $C \sqsubseteq D \in \mathcal{T}$ but $p \notin D^{\mathcal{I}_k}$. Let $D_0, D_1, ..., D_l$ be the conjuncts of D of the form $D_i = \exists s_i.E_i$, $0 \leq i \leq l$. Now, we define $\mathcal{I}_{n+1}$ as follows:

- $\Delta^{\mathcal{I}_{n+1}} = \Delta^{\mathcal{I}_n} \cup \{p \cdot s_i \cdot q \mid q \in \mathsf{paths}(E_i), 0 \leq i \leq l\}$;
- $A^{\mathcal{I}_{n+1}} = A^{\mathcal{I}_n} \cup \{p \cdot s_i \cdot q \mid A \in \mathsf{tail}(q), 0 \leq i \leq l\} \cup \{p \mid \text{if } A \text{ is a conjunct of } D\}$;
- $r^{\mathcal{I}_{n+1}} = r^{\mathcal{I}_n} \cup \{(p \cdot s_i \cdot q, p \cdot s_i \cdot q') \mid (q, q') \in r^{\mathcal{I}_{E_i}}, 0 \leq i \leq l\}$.

(The canonical model $\mathcal{I}_{\mathcal{T},C} = \bigcup \mathcal{I}_{n\geq 0}$ of an $\mathcal{EL}$ concept expression C and an $\mathcal{EL}$ TBox $\mathcal{T}$ is defined analogously with $\mathcal{I}_0 = \mathcal{I}_C$, where $\mathcal{I}_C$ is the tree interpretation of C.)

The following lemma states the main property of the canonical model defined above. It can be proved using lemmas similar to the ones in Appendix A.1.

Lemma 14. *Let α be an instance assertion, $(\mathcal{T}, \mathcal{A})$ an $\mathcal{EL}$ knowledge base and $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ defined as above. Then, $\mathcal{I}_{\mathcal{T},\mathcal{A}} \models \alpha$ if and only if $(\mathcal{T}, \mathcal{A}) \models \alpha$.*

B Proofs for Polynomial Time Learnability

Proof of Theorem 1. Let $(X, \mathcal{L}, \mu)$ and $(X', \mathcal{L}, \mu')$ be learning frameworks. If there exists a polynomial reduction from $(X, \mathcal{L}, \mu)$ to $(X', \mathcal{L}, \mu')$ and a polynomial learning algorithm for $(X', \mathcal{L}, \mu')$ that only uses positive bounded equivalence queries then $(X, \mathcal{L}, \mu)$ is polynomially exact learnable.

Proof. As $(X, \mathcal{L}, \mu)$ polynomially reduces to learning $(X', \mathcal{L}, \mu')$, there are functions $f : X' \to X$ and $g : \mathcal{L} \times \mathcal{L} \times X \to X'$ such that f transforms a membership query with $x' \in X'$ as input into a query with $x \in X$ and g transforms a counterexample $x \in X$ into a counterexample $x' \in X'$. Let $q_1(|x'|)$ and $q_2(|l|, |x|)$ be the polynomial upper bounds on the time needed to compute $f(x')$ and $g(l, h, x)$, respectively. Denote by y the largest $x \in X$ computed so far by function g. Since $(X', \mathcal{L}, \mu')$ is polynomially learnable by an algorithm A' that only uses positive bounded equivalence queries, at every step of computation the time used by A' up to that step is bounded by a polynomial $p(|l|, q_2(|l|, |y|))$, where $|l|$ is the size of a target $l \in \mathcal{L}$. Note that the largest counterexample seen so far by Algorithm A' is bounded by $q_2(|l|, |y|)$. Then, A' is computed in $p'(|l|, |y|) = p(|l|, q_2(|l|, |y|))$ many steps.

We use A' to learn $(X, \mathcal{L}, \mu)$ as follows. Whenever a membership query with $x' \in X'$ as input is called by A' we compute $f(x')$ and call the $\mathsf{MEM}_{l,X}$ oracle. We return 'yes' if $f(x') \in \mu(l)$ and 'no' otherwise. It should be clear that for every membership query with $x \in X'$ asked by A' we have $|x'| \leq p'(|l|, |y|)$. Whenever an equivalence query with $h \in \mathcal{L}$ as input is called by A' we pass it on to the $\mathsf{EQ}_{l,X}$ oracle. If it returns 'yes' then the learner succeeded. Otherwise it returns 'no' and provides a positive counterexample $x \in \mu(l) \setminus \mu(h)$. Then, we compute $g(l, h, x)$ and return it to A'.

By definition of f and g all the answers provided to A' are consistent with answers the oracles $\mathsf{MEM}_{l,X'}$ and $\mathsf{EQ}_{l,X'}$ would provide to A'. As Algorithm A' is polynomially computable in $p'(|l|, |y|)$ steps we have that $(X, \mathcal{L}, \mu)$ is exactly learned after at most $p'(|l|, |y|) \cdot q_1(p'(|l|, |y|)) \cdot q_2(|l|, |y|)$ steps, which is polynomial in $|l|$ and $|y|$, as required. ❑

Proof of Lemma 1. Let $L \in \{\text{DL-Lite}_{\mathcal{R}}^{\exists}, \mathcal{EL}_{\mathsf{lhs}}\}$ and let $C \sqsubseteq D$ be an L concept inclusion. Then $(\mathcal{T}, \mathcal{A}_C) \models D(\rho_C)$ if, and only if, $\mathcal{T} \models C \sqsubseteq D$.

Proof. Let $\mathcal{I}_C$ and $\mathcal{I}_D$ be the tree interpretations of C and D, respectively, and $\mathcal{I}_{\mathcal{A}_C}$ be the canonical model of $\mathcal{A}_C$. For $\mathcal{T}$ a DL-Lite$_{\mathcal{R}}^{\exists}$ (or an $\mathcal{EL}_{\mathsf{lhs}}$) TBox, let $\mathcal{I}_{C,\mathcal{T}}$ be the canonical model of C and $\mathcal{T}$. Similarly, let $\mathcal{I}_{\mathcal{A}_C,\mathcal{T}}$ be the canonical model of $\mathcal{A}_C$ and $\mathcal{T}$. Since $\mathcal{I}_C$ and $\mathcal{I}_{\mathcal{A}_C}$ are isomorphic, $\mathcal{I}_{C,\mathcal{T}}$ and $\mathcal{I}_{\mathcal{A}_C,\mathcal{T}}$ are also isomorphic. This means that there is a homomorphism $h : \mathcal{I}_D \to \mathcal{I}_{C,\mathcal{T}}$ mapping the root ρ_D of $\mathcal{I}_D$ to ρ_C iff there is a homomorphism $g : \mathcal{I}_D \to \mathcal{I}_{\mathcal{A}_C,\mathcal{T}}$ mapping ρ_D to ρ_C. So, $(\mathcal{T}, \mathcal{A}_C) \models D(\rho_C)$ iff $\mathcal{T} \models C \sqsubseteq D$. ❑

C Proofs for Positive Reduction for DL-Lite$_{\mathcal{R}}^{\exists}$ TBoxes

Proof of Lemma 2. Let $(\mathcal{A}, C(a))$ be a positive counterexample. Then the following holds:

1. if C is a basic concept then there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models C(a)$;
2. if C is of the form $\exists r.C'$ (or $\exists r^-.C'$) and C is role saturated and parent/child merged then either there is $r(a,b) \in \mathcal{A}$ (or $r(b,a) \in \mathcal{A}$) such that $(\mathcal{T}, \mathcal{A}) \models C'(b)$ or there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models C(a)$.

Proof. We first prove Point (1). Assume C is a basic concept B. By Lemma 13, if $(\mathcal{T}, \mathcal{A}) \models B(a)$ then $a \in B^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$, where $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ is the canonical model of $(\mathcal{T}, \mathcal{A})$. By construction of $\mathcal{I}_{\mathcal{T},\mathcal{A}} = \bigcup \mathcal{I}_{n \geq 0}$ we have that there is an n such that $a \in B^{\mathcal{I}_n}$. We now state a claim from which Point (1) of this lemma follows.

Claim 1 For all n, if $a \in B^{\mathcal{I}_n}$, where B is a basic concept, then there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $a \in B^{\mathcal{I}_{\mathcal{T},\mathcal{A}'}}$.

For $n = 0$, since $\mathcal{I}_0 = \mathcal{I}_{\mathcal{A}}$, B is either a concept name A with $A(a) \in \mathcal{A}$ or B is of the form $\exists r.\top$ (or $\exists r^-.\top$) with $r(a,b) \in \mathcal{A}$ (or $r(b,a) \in \mathcal{A}$). In all cases, the lemma is satisfied for $\mathcal{I}_n$ with $n = 0$. Suppose the claim holds for $n = k$. We want to show now for $n = k + 1$. Following the construction of $\mathcal{I}_{\mathcal{T},\mathcal{A}}$, assume $a \in B'^{\mathcal{I}_k}$, $a \notin D^{\mathcal{I}_k}$ and let $B' \sqsubseteq D \in \mathcal{T}$ be such that $a \in D^{\mathcal{I}_{k+1}}$. By induction hypothesis, there is a singleton ABox $\mathcal{A}'$ such that $a \in B'^{\mathcal{I}_{\mathcal{T},\mathcal{A}'}}$, where $\mathcal{I}_{\mathcal{T},\mathcal{A}'}$ is the canonical model of $(\mathcal{T}, \mathcal{A}')$. If $B' \sqsubseteq D \in \mathcal{T}$ then, as $a \in B'^{\mathcal{I}_{\mathcal{T},\mathcal{A}'}}$, we know that $a \in D^{\mathcal{I}_{\mathcal{T},\mathcal{A}'}}$. For every basic concept B'' such that $a \in B''^{\mathcal{I}_{k+1}}$ we have that either (a) $a \in B''^{\mathcal{I}_k}$ or (b) $\mathcal{R} \models D \sqsubseteq B''$. In case (a) we can apply the induction hypothesis. In case (b), as $a \in D^{\mathcal{I}_{\mathcal{T},\mathcal{A}'}}$, we know that $a \in B''^{\mathcal{I}_{\mathcal{T},\mathcal{A}'}}$. In both cases there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $a \in B''^{\mathcal{I}_{\mathcal{T},\mathcal{A}'}}$. Then, by Lemma 13, we have that $(\mathcal{T}, \mathcal{A}') \models B''(a)$.

We now prove Point (2). We show the lemma for $\exists r.C'(a)$, the case where the counterexample is of the form $\exists r^-.C'(a)$ is analogous. If $(\mathcal{T}, \mathcal{A}) \models \exists r.C'(a)$ then, by Lemma 13, $a \in (\exists r.C')^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$. By semantics of $\exists$, there is $d \in \Delta^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$ such that $(a,d) \in r^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$ and $d \in C^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$. By assumption there is no $r(a,b) \in \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}) \models C'(b)$. As $d \in C'^{\mathcal{I}_{\mathcal{T},\mathcal{A}}}$, we have that $d \notin \mathsf{Ind}(\mathcal{A})$. By Lemma 10, there is a homomorphism $h : \mathcal{I}_{C'} \to \mathcal{I}_{\mathcal{T},\mathcal{A}}$ mapping $\rho_{C'}$ to d. If there is $p \in \Delta^{\mathcal{I}'_C}$ such that $h(p) \in \mathsf{Ind}(\mathcal{A})$ then this contradicts the fact that $\exists r.C'(a)$ is role saturated and parent/child merged. Then we have that there is a singleton $\mathcal{A}' \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{A}') \models \exists r.C'(a)$. ❑

Proof of Lemma 3. For any target DL-Lite$_{\mathcal{R}}^{\exists}$ TBox $\mathcal{T}$ and hypothesis DL-Lite$_{\mathcal{R}}^{\exists}$ TBox $\mathcal{H}$ given a positive data retrieval counterexample $(\mathcal{A}, C(a))$, Algorithm 1 computes in time polynomial in $|\mathcal{T}|$, $|\mathcal{H}|$, $|\mathcal{A}|$ and $|C|$ a counterexample $C'(b)$ such that $(\mathcal{T}, \mathcal{A}') \models C'(b)$, where $\mathcal{A}' \subseteq \mathcal{A}$ is a singleton ABox.

Proof. Let $C(a)$ be the counterexample given to the function "ReduceCounterExample" (Line 2). In Line 3, Algorithm 1 exhaustively applies the following rules which rely on posing membership queries to the oracle:

(Role saturation) if $(\mathcal{T}, \mathcal{A}) \models C'(a)$ and C' is the result of replacing a role r in the tree representation of C to some role r', where $\mathcal{T} \models r' \sqsubseteq r$ and $r' \not\equiv_{\mathcal{T}} r$, then consider $C'(a)$ instead of $C(a)$ as a counterexample.

(Parent/child merging) if $(\mathcal{T}, \mathcal{A}) \models C'(a)$ and C' is the result of identifying p_1 and p_3 in the tree interpretation of C, where $p_1, p_2, p_3 \in \mathsf{paths}(C)$ with $p_2 = p_1 \cdot r \cdot C_1$, $p_3 = p_2 \cdot s \cdot C_2$ and $r^- \equiv_{\mathcal{T}} s$; then consider $C'(a)$ instead of $C(a)$ as a counterexample.

Let $D(a)$ denote the counterexample obtained by the application of the rules above. If D is of the form $D_0 \sqcap ... \sqcap D_n$ then there is a D_i, $0 \leq i \leq n$, such that $D_i(a)$ is also a counterexample. In Line 5, Algorithm 1, chooses a conjunct D_i such that $D_i(a)$ is a counterexample. Otherwise, D is a basic concept or D is of the form $\exists r.E$. In this case consider $D_i = D$. Now, we make a case distinction:

- D_i is a basic concept B: then Point (1) of Lemma 2 applies.
- D_i is of the form $\exists r.E$ and we make a case distinction:
  1. there is $r(a,b) \in \mathcal{A}$ such that $(\mathcal{T},\mathcal{A}) \models E(b)$. In this case, Algorithm 1 recursively calls the function "ReduceCounterExample" (Lines 8-9).
  2. otherwise, notice that since $D(a)$ is role saturated and parent/child merged, $D_i(a)$ also have the same properties. Then, Point (2) of Lemma 2 applies.

In case (1), with $E(b)$ as the counterexample, observe that $E(b)$ is a refined version of the initial counterexample $C(a)$. Since the size of the refined counterexample is strictly smaller in every recursive call of the function "ReduceCounterExample", the number of calls to this function is bounded by $|C|$. Then, conditions for Point (1) or Point (2) of Lemma 2 are satisfied after polynomially many recursive calls.

It remains to show that termination is in polynomial time with respect to $|\mathcal{T}|$, $|\mathcal{H}|$, $|C|$ and $|\mathcal{A}|$. Algorithm 1 makes membership queries in Lines 3, 8 and 11. In Line 3 the number of rule applications is bounded by $|C|$. In Line 8 it is verified for every $r(a,b) \in \mathcal{A}$ whether $(\mathcal{T},\mathcal{A}) \models C'(b)$. So the number of membership queries is bounded by $|\mathcal{A}_{r(a,b)}|$, where $\mathcal{A}_{r(a,b)} \subseteq \mathcal{A}$ is the set of assertions in $\mathcal{A}$ of the form $r(a,b)$, where b is some element in $\mathsf{Ind}(\mathcal{A})$ and r can be an inverse role. Finally, in Line 11 the number of membership queries is bounded by $|\mathcal{A}_a|$, where $\mathcal{A}_a \subseteq \mathcal{A}$ is the set of assertions in $\mathcal{A}$ where individual a occurs. Since the size of the counterexample is strictly smaller in every recursive call of the function "ReduceCounterExample", the number of calls to this function is bounded by $|C|$. ❑

D Proofs for Positive Reduction for $\mathcal{EL}_{\mathsf{lhs}}$ TBoxes

Since our target TBox $\mathcal{T}$ contains $\mathcal{EL}_{\mathsf{lhs}}$ concept inclusions, we can find counterexamples by posing atomic queries $(\mathcal{T},\mathcal{A}) \models A(b)$ to the oracle, with $A \in \mathsf{N_C} \cap \Sigma_{\mathcal{T}}$ and $b \in \mathsf{Ind}(\mathcal{A})$. We start our proofs in this section with a straightforward argument for this fact.

Proof of Lemma 4. If $(\mathcal{A}, D(a))$ is a positive counterexample then there exists a concept name A and an individual $b \in \mathsf{Ind}(\mathcal{A})$ such that $(\mathcal{A}, A(b))$ is also a counterexample.

Proof. As $(\mathcal{A}, D(a))$ is a positive counterexample, $(\mathcal{T},\mathcal{A}) \models D(a)$ and $(\mathcal{H},\mathcal{A}) \not\models D(a)$. Then, by Lemma 14, $\mathcal{I}_{\mathcal{T},\mathcal{A}} \models D(a)$ and $\mathcal{I}_{\mathcal{H},\mathcal{A}} \not\models D(a)$, where $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ and $\mathcal{I}_{\mathcal{H},\mathcal{A}}$ are the canonical models of $(\mathcal{T},\mathcal{A})$ and $(\mathcal{H},\mathcal{A})$, respectively. As $(\mathcal{T},\mathcal{A}) \models D(a)$ and $\mathcal{I}_{\mathcal{H},\mathcal{A}} \not\models D(a)$ we have that $\mathcal{I}_{\mathcal{H},\mathcal{A}} \not\models \mathcal{T}$. Then, there is a concept inclusion $C \sqsubseteq A \in \mathcal{T}$ and $b \in \Delta^{\mathcal{I}_{\mathcal{H},\mathcal{A}}}$ such that $b \in (C \setminus A)^{\mathcal{I}_{\mathcal{H},\mathcal{A}}}$. Thus, $(\mathcal{T},\mathcal{A}) \models A(b)$ and $(\mathcal{H},\mathcal{A}) \not\models A(b)$. ❑

In order to compute a tree interpretation of an $\mathcal{EL}$ concept C we first *minimize* an ABox $\mathcal{A}$ so that $|\mathcal{A}|$ is bounded by $|\mathcal{T}|$. Recall that we denote by $\mathcal{A}^{-a}$ the result of removing from $\mathcal{A}$ all ABox assertions where a occurs. That is, $\mathcal{A}^{-a} = \mathcal{A} \setminus \mathcal{A}^a$, where $\mathcal{A}^a = \{r(a,b) \mid b \in \mathsf{Ind}(\mathcal{A}), r \in \Sigma_\mathcal{T} \cap \mathsf{N_R}\} \cup \{r(b,a) \mid b \in \mathsf{Ind}(\mathcal{A}), r \in \Sigma_\mathcal{T} \cap \mathsf{N_R}\} \cup \{A(a) \mid A \in \Sigma_\mathcal{T} \cap \mathsf{N_C}\}$. Also, $\mathcal{A}^{-r(a,b)}$ is obtained by removing a role assertion $r(a,b)$ from $\mathcal{A}$. Let $\mathcal{I}_\mathcal{A}$ be the canonical model of an ABox $\mathcal{A}$. We say that $\mathcal{I}_\mathcal{A}$ is a *countermodel* if $\mathcal{I}_\mathcal{A} \not\models \mathcal{T}$ and $\mathcal{I}_\mathcal{A} \models \mathcal{H}$. We define $\mathcal{A}$ as *minimal* if the following conditions are satisfied:

1. $\mathcal{I}_\mathcal{A}$ is a countermodel;
2. $\mathcal{I}_{\mathcal{A}^{-a}} \models \mathcal{T}$; and
3. $\mathcal{I}_{\mathcal{A}^{-r(a,b)}} \models \mathcal{T}$.

The following lemma shows that the size of a minimal ABox is polynomial in $|\mathcal{T}|$.

Lemma 15. *Let $\mathcal{I}_\mathcal{A}$ be the canonical model of a minimal ABox $\mathcal{A}$. Then, $|\Delta^{\mathcal{I}_\mathcal{A}}| \leq |\mathcal{T}|$.*

Proof. By Condition 1 of minimal ABoxes, $\mathcal{I}_\mathcal{A}$ is a countermodel. So $\mathcal{I}_\mathcal{A} \not\models \mathcal{T}$. Then there is $C \sqsubseteq A \in \mathcal{T}$ such that $a \in (C \setminus A)^{\mathcal{I}_\mathcal{A}}$, for some $a \in \Delta^{\mathcal{I}_\mathcal{A}}$. If $a \in C^{\mathcal{I}_\mathcal{A}}$ then (by Lemma 10) there is a homomorphism $h : \mathcal{I}_C \to \mathcal{I}_\mathcal{A}$ mapping ρ_C to a, where ρ_C is the root of $\mathcal{I}_C$. We need to show that h is surjective. Suppose this is not the case. Then, there is $d \in \Delta^{\mathcal{I}_\mathcal{A}}$ such that $d \notin \mathsf{Im_h}$, where $\mathsf{Im_h} = \{e \in \Delta^{\mathcal{I}_\mathcal{A}} | e = h(p)$ for some $p \in \Delta^{\mathcal{I}_C}\}$. Now, denote as $\mathcal{I}_{\mathcal{A}^{-d}}$ the result of removing $d \notin \mathsf{Im_h}$ from $\mathcal{I}_\mathcal{A}$. Since $\mathcal{I}_{\mathcal{A}^{-d}}$ is a subinterpretation of $\mathcal{I}_\mathcal{A}$, if $a \notin A^{\mathcal{I}_\mathcal{A}}$ then $a \notin A^{\mathcal{I}_{\mathcal{A}^{-d}}}$. So $a \in (C \setminus A)^{\mathcal{I}_{\mathcal{A}^{-d}}}$, which means that $\mathcal{I}_{\mathcal{A}^{-d}} \not\models \mathcal{T}$. This contradicts the fact that $\mathcal{I}_{\mathcal{A}^{-d}} \models \mathcal{T}$ for any element d from $\Delta^{\mathcal{I}_\mathcal{A}}$ (Condition 3 of minimal ABoxes). Since $C \sqsubseteq A \in \mathcal{T}$, we know that $|\Delta^{\mathcal{I}_C}| \leq |\mathcal{T}|$. Thus, $|\Delta^{\mathcal{I}_\mathcal{A}}| \leq |\Delta^{\mathcal{I}_C}| \leq |\mathcal{T}|$. ❏

Algorithm 2, presented in Section 3, minimizes $\mathcal{A}$ to an $\mathcal{A}'$ with the properties described above. Since the Algorithm receives a positive counterexample, we know that $\mathcal{I}_\mathcal{A}$ is not a model of $\mathcal{T}$, that is, $\mathcal{I}_\mathcal{A} \not\models \mathcal{T}$. In order to satisfy Condition 1 above and reduce $\mathcal{A}$ (Conditions 2 and 3), Algorithm 2 applies rules 'Concept saturate', 'Domain Minimize' and 'Role Minimize', as described in Section 3.

Proof of Lemma 5. Given a positive counterexample $(\mathcal{A}, D(a))$ with $D \in \mathsf{N_C}$, Algorithm 2 computes in polynomially many steps with respect to $|\mathcal{A}|$, $|\mathcal{H}|$, and $|\mathcal{T}|$ an ABox $\mathcal{A}'$ such that $|\mathsf{Ind}(\mathcal{A}')| \leq |\mathcal{T}|$ and $(\mathcal{A}', A(b))$ is a positive counterexample, for some concept name A and individual $b \in \mathsf{Ind}(\mathcal{A}')$.

Proof. By Lemma 15, if $\mathcal{A}$ is a minimal ABox then $|\mathsf{Ind}(\mathcal{A})| \leq |\mathcal{T}|$. Then, we only need the following claims to show this lemma.

Claim 1 Algorithm 2 computes a minimal ABox $\mathcal{A}$.

For Condition 1, we have that, in Line 3, Algorithm 2 concept saturates $\mathcal{A}$ with $\mathcal{H}$. Then, after computing Line 3, we have that $\mathcal{I}_\mathcal{A} \models \mathcal{H}$. Since the minimization rules described above do not remove any concept name implied by $\mathcal{H}$, the ABox computed by the algorithm is a model of $\mathcal{H}$ in all steps that follow Line 3. By definition of the rules,

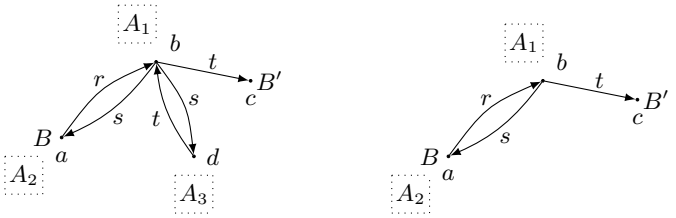
at least one counterexample is entailed by $(\mathcal{T}, \mathcal{A})$, which is the counterexample where the rules are being applied. So for all iterations of Algorithm 2, $\mathcal{I}_{\mathcal{A}} \not\models \mathcal{T}$.

For Condition 2, suppose there is $\mathcal{A}^{-d}$ such that $\mathcal{I}_{\mathcal{A}^{-d}} \not\models \mathcal{T}$. Then there is $C \sqsubseteq A \in \mathcal{T}$ and $a \in \Delta^{\mathcal{I}_{\mathcal{A}^{-d}}}$ such that $a \in (C \setminus A)^{\mathcal{I}_{\mathcal{A}^{-d}}}$. This contradicts the fact that, in Line 6, domain minimization was applied in $\mathcal{A}$ for all counterexamples. Thus, $\mathcal{I}_{\mathcal{A}^{-d}} \models \mathcal{T}$. The argument is similar for role minimization (Condition 3).

Claim 2 Algorithm 2 runs in polynomially many steps with respect to $|\mathcal{A}|$ and $|\mathsf{N_C} \cap \Sigma_{\mathcal{T}}|$, where $\mathsf{N_C} \cap \Sigma_{\mathcal{T}}$ are the concept names in the vocabulary.

We know that the number of possible concept name assertions in $\mathcal{A}$ is $|\mathsf{N_C} \cap \Sigma_{\mathcal{T}}| \cdot |\mathsf{Ind}(\mathcal{A})|$. So, in Line 3, the number of applications of the rule Concept Saturate with $\mathcal{H}$ is bounded by $|\mathsf{N_C} \cap \Sigma_{\mathcal{T}}| \cdot |\mathsf{Ind}(\mathcal{A})|$. Also, the number of iterations in Line 4 is at most $|\mathsf{N_C} \cap \Sigma_{\mathcal{T}}| \cdot |\mathsf{Ind}(\mathcal{A})|$. Since role and domain minimization is linear in $|\mathcal{A}|$, we have that Algorithm 2 is polynomial with respect to $|\mathcal{A}|$ and $|\mathsf{N_C} \cap \Sigma_{\mathcal{T}}|$. ❑

As an example, let $\mathcal{T} = \{\exists s.\exists r.(\exists s.B \sqcap \exists t.B') \sqsubseteq A_1, \exists r.(\exists t.B' \sqcap \exists s.B) \sqsubseteq A_2, \exists t.(\exists s.B \sqcap \exists t.B') \sqsubseteq A_3, \exists r.\exists t.B' \sqsubseteq B\}$, $\mathcal{H} = \{\exists r.\exists t.B' \sqsubseteq B\}$ and $\mathcal{A} = \{r(a,b), s(b,a), t(b,c), s(b,d), t(d,b), B'(c)\}$. After concept saturating $\mathcal{A}$ with $\mathcal{H}$ we have that $B(a)$ is added to $\mathcal{A}$. Then we have $A_1(b), A_2(a), A_3(d)$ as counterexamples (Figure 4a). Assume Algorithm 2 starts minimizing $\mathcal{A}$ in Line 4 with the counterexample $A_1(b)$. The algorithm eliminates $s(b,d)$ and $t(d,b)$ from $\mathcal{A}$. As a result, $A_3(d)$ is not a counterexample any more. In the next iteration, the algorithm tries to minimize $\mathcal{A}$ with $A_2(a)$, which does not eliminates any other assertion from $\mathcal{A}$. So $\mathcal{A}$ is now minimal. The result of minimizing $\mathcal{A}$ is shown by Figure 4b, it contains now only $A_1(b)$ and $A_2(a)$ as counterexamples.



(a) Initial $\mathcal{A}$ with the counterexamples $A_1(b), A_2(a), A_3(d)$

(b) Minimal $\mathcal{A}$ with the counterexamples $A_1(b), A_2(a)$

Fig. 4: Minimizing $\mathcal{A}$

We have seen that a minimal ABox $\mathcal{A}$ is a countermodel bounded by $|\mathcal{T}|$. Algorithm 3, presented in Section 3, is based on two operations (i) minimization, presented above, and (ii) *unfolding*. The unfolding operation doubles the length of a cycle in $\mathcal{A}$. By increasing the length of cycles and then minimizing, the algorithm proceeds unfolding elements until $\mathcal{A}$ is tree shaped. We say that $\mathcal{A}$ has a (undirected) cycle if there is a finite sequence $a_0 \cdot r_1 \cdot a_1 \cdot \ldots \cdot r_k \cdot a_k$ such that (i) $a_0 = a_k$ and (ii) there are mutually

distinct assertions of the form $r_{i+1}(a_i, a_{i+1})$ or $r_{i+1}(a_{i+1}, a_i)$ in $\mathcal{A}$, for $0 \leq i < k$. For a cycle $c = a_0 \cdot r_1 \cdot a_1 \cdot ... \cdot r_k \cdot a_k$, denote as $\mathsf{nodes}(c) = \{a_0, a_1, ..., a_{k-1}\}$ the set of individuals that occur in c. Also, $\mathsf{roles}(c) = \{r_1, r_2, ..., r_k\}$ is the set of roles that occur in c. We denote by $\widehat{a}$ the copy of an element a created by the unfolding operation described below. The set of copies of individuals that occur in c is denoted by $\mathsf{nodes}(\widehat{c}) = \{\widehat{a_0}, \widehat{a_1}, ..., \widehat{a}_{k-1}\}$. Let $\mathcal{I}_\mathcal{A}$ be the canonical interpretation of an ABox $\mathcal{A}$. An element $a \in \Delta^{\mathcal{I}_\mathcal{A}}$ is *folded* if there is a cycle $c = a_0 \cdot r_1 \cdot a_1 \cdot ... \cdot r_k \cdot a_k$ with $a = a_0 = a_k$. Without loss of generality we assume that $r_1(a_0, a_1) \in \mathcal{A}$. The *unfolding* of c is described below.

1. We first open the cycle by removing $r_1(a_0, a_1)$ from $\mathcal{A}$. So $r_1^{\mathcal{I}_\mathcal{A}} := r_1^{\mathcal{I}_\mathcal{A}} \setminus \{(a_0, a_1)\}$.
2. Then we create copies of the nodes in the cycle:
   - $\Delta^{\mathcal{I}_\mathcal{A}} := \Delta^{\mathcal{I}_\mathcal{A}} \cup \{\widehat{b} \mid b \in \mathsf{nodes}(c)\}$
   - $A^{\mathcal{I}_\mathcal{A}} := A^{\mathcal{I}_\mathcal{A}} \cup \{\widehat{b} \mid b \in A^{\mathcal{I}_\mathcal{A}}\}$
   - $r^{\mathcal{I}_\mathcal{A}} := r^{\mathcal{I}_\mathcal{A}} \cup \{(\widehat{b}, \widehat{d}) \mid (b, d) \in r^{\mathcal{I}_\mathcal{A}}\} \cup \{(\widehat{b}, e) \mid (b, e) \in r^{\mathcal{I}_\mathcal{A}}, e \notin \mathsf{nodes}(c)\}$
3. As a third step we close again the cycle, now with double size. So we update $r_1^{\mathcal{I}_\mathcal{A}} := r_1^{\mathcal{I}_\mathcal{A}} \cup \{(a_0, \widehat{a_1}), (\widehat{a_0}, a_1)\}$.

We now show that our unfolding maintains the invariant that if $A(a)$ is a counterexample for $(\mathcal{T}, \mathcal{A})$ relative to $(\mathcal{H}, \mathcal{A})$ then $A(a)$ will remain as a counterexample after applying this operation over an arbitrary cycle in $\mathcal{A}$. This is obtained by Lemmas 16 and 17.

Lemma 16. *Let $\mathcal{A}'$ be the result of unfolding a cycle c in $\mathcal{A}$. Then the following relation $S \subseteq \Delta^{\mathcal{I}_\mathcal{A}} \times \Delta^{\mathcal{I}_{\mathcal{A}'}}$ is a simulation $\mathcal{I}_\mathcal{A} \Rightarrow \mathcal{I}_{\mathcal{A}'}$:*

- *for $a \in \Delta^{\mathcal{I}_\mathcal{A}} \setminus \mathsf{nodes}(c)$, $(a^{\mathcal{I}_\mathcal{A}}, a^{\mathcal{I}_{\mathcal{A}'}}) \in S$;*
- *for $a \in \mathsf{nodes}(c)$, $(a^{\mathcal{I}_\mathcal{A}}, a^{\mathcal{I}_{\mathcal{A}'}}) \in S$ and $(a^{\mathcal{I}_\mathcal{A}}, \widehat{a}^{\mathcal{I}_{\mathcal{A}'}}) \in S$.*

Proof. We need to show that S is a simulation $\mathcal{I}_\mathcal{A} \Rightarrow \mathcal{I}_{\mathcal{A}'}$. That is, for $d, d_1 \in \Delta^{\mathcal{I}_\mathcal{A}}$ and $e, e_1 \in \Delta^{\mathcal{I}_{\mathcal{A}'}}$:

1. for all concept names $A \in \Sigma_\mathcal{T}$ and all $(d, e) \in S$, if $d \in A^{\mathcal{I}_\mathcal{A}}$ then $e \in A^{\mathcal{I}_{\mathcal{A}'}}$;
2. for all role names $r \in \Sigma_\mathcal{T}$, all $(d_1, e_1) \in S$ and all $d_2 \in \Delta^{\mathcal{I}_\mathcal{A}}$, if $(d_1, d_2) \in r^{\mathcal{I}_\mathcal{A}}$ then there exists $e_2 \in \Delta^{\mathcal{I}_{\mathcal{A}'}}$ such that $(e_1, e_2) \in r^{\mathcal{I}_{\mathcal{A}'}}$ and $(d_2, e_2) \in S$.

For Point 1 we have that by definition of the unfolding operation (Step 2), if $a \in A^{\mathcal{I}_\mathcal{A}}$ then $a \in A^{\mathcal{I}_{\mathcal{A}'}}$ and $\widehat{a} \in A^{\mathcal{I}_{\mathcal{A}'}}$. Point 2 follows from Claims 1 and 2 below.

Claim 1 If $a^{\mathcal{I}_\mathcal{A}}$ has an r-successor b then $a^{\mathcal{I}_{\mathcal{A}'}}$ has an r-successor d with $(b, d) \in S$.

By definition of the unfolding operation (Step 1), $r_1(a_0, a_1)$ is the only role assertion removed from $\mathcal{A}$. In Step 3 we include $(a_0, \widehat{a_1})$ to $r_1^{\mathcal{I}_\mathcal{A}}$. By definition of S, we have that $(a_1, \widehat{a_1}) \in S$. So if $a^{\mathcal{I}_\mathcal{A}}$ has an r-successor b then $a^{\mathcal{I}_{\mathcal{A}'}}$ has an r-successor d with $(b, d) \in S$.

Claim 2 If $a^{\mathcal{I}_\mathcal{A}}$ has an r-successor b then $\widehat{a}^{\mathcal{I}_{\mathcal{A}'}}$ has an r-successor e with $(b, e) \in S$.

For $(a_0, a_1) \in r_1^{\mathcal{I}_\mathcal{A}}$, in Step 3 we include $(\widehat{a}_0, a_1)$ to $r_1^{\mathcal{I}'_\mathcal{A}}$. By definition of S, $(a_1^{\mathcal{I}_{\mathcal{A}'}}, a_1^{\mathcal{I}_{\mathcal{A}'}}) \in S$. Otherwise, in Step 2 we have that for all r-successors $b^{\mathcal{I}_\mathcal{A}}$ of $a^{\mathcal{I}_\mathcal{A}}$ such that $b \notin \mathsf{nodes}(c)$, $(\widehat{a}^{\mathcal{I}_{\mathcal{A}'}}, b^{\mathcal{I}_{\mathcal{A}'}}) \in r^{\mathcal{I}_{\mathcal{A}'}}$. By definition of S, $(b^{\mathcal{I}_\mathcal{A}}, b^{\mathcal{I}_{\mathcal{A}'}}) \in S$. Also, in Step 2, for the r-successors $b^{\mathcal{I}_\mathcal{A}}$ of $a^{\mathcal{I}_\mathcal{A}}$ such that $b \in \mathsf{nodes}(c)$, we have that $(\widehat{a}^{\mathcal{I}_{\mathcal{A}'}}, \widehat{b}^{\mathcal{I}_{\mathcal{A}'}}) \in r^{\mathcal{I}_{\mathcal{A}'}}$, where $\widehat{b}^{\mathcal{I}_{\mathcal{A}'}}$ is the copy of $b^{\mathcal{I}_{\mathcal{A}'}}$. Again, by definition of S, $(b^{\mathcal{I}_\mathcal{A}}, \widehat{b}^{\mathcal{I}_{\mathcal{A}'}}) \in S$. ❑

Lemma 17. *Let $\mathcal{A}'$ be the result of unfolding a cycle c in $\mathcal{A}$. Let $h_* : \mathcal{I}_{\mathcal{A}'} \to \mathcal{I}_\mathcal{A}$ be the following mapping:*

- *for $a \in \Delta^{\mathcal{I}_\mathcal{A}} \setminus \mathsf{nodes}(c)$, $h_*(a^{\mathcal{I}_{\mathcal{A}'}}) = a^{\mathcal{I}_\mathcal{A}}$;*
- *for $a \in \mathsf{nodes}(c)$, $h_*(a^{\mathcal{I}_{\mathcal{A}'}}) = a^{\mathcal{I}_\mathcal{A}}$ and $h_*(\widehat{a}^{\mathcal{I}_{\mathcal{A}'}}) = a^{\mathcal{I}_\mathcal{A}}$.*

Then, $h_* : \mathcal{I}_{\mathcal{A}'} \to \mathcal{I}_\mathcal{A}$ is a homomorphism.

Proof. By definition of the unfolding operation, no concept name assertion is removed from $\mathcal{A}'$. So $a \in A^{\mathcal{I}_{\mathcal{A}'}}$ iff $a \in A^{\mathcal{I}_\mathcal{A}}$. Also, in Step 2 of the unfolding operation we have that $\widehat{a} \in A^{\mathcal{I}_{\mathcal{A}'}}$ iff $a \in A^{\mathcal{I}_\mathcal{A}}$. So if $a \in A^{\mathcal{I}_{\mathcal{A}'}}$ or $\widehat{a} \in A^{\mathcal{I}_{\mathcal{A}'}}$ then $h_*(a) = h_*(\widehat{a}) = a \in A^{\mathcal{I}_\mathcal{A}}$. Now, for $(a, b) \in r^{\mathcal{I}_{\mathcal{A}'}}$, we make a case distinction:

- $a, b \notin \mathsf{nodes}(\widehat{c})$: in this case, the unfolding operation does not include any new role assertion. Then, $(a, b) \in r^{\mathcal{I}_{\mathcal{A}'}}$ implies $(a, b) \in r^{\mathcal{I}_\mathcal{A}}$.
- $\widehat{a}, \widehat{b} \in \mathsf{nodes}(\widehat{c})$: by Step 2 of the unfolding operation, if $(\widehat{a}, \widehat{b}) \in r^{\mathcal{I}_{\mathcal{A}'}}$ then $(a, b) \in r^{\mathcal{I}_\mathcal{A}}$.
- $\widehat{a} \in \mathsf{nodes}(\widehat{c})$ and $b \notin \mathsf{nodes}(\widehat{c})$: for $(\widehat{a}_0, a_1) \in r_1^{\mathcal{I}_{\mathcal{A}'}}$ we know that $(a_0, a_1) \in r_1^{\mathcal{I}_\mathcal{A}}$. Otherwise, by Step 2, if $(\widehat{a}, b) \in r^{\mathcal{I}_{\mathcal{A}'}}$ then $(a, b) \in r^{\mathcal{I}_\mathcal{A}}$.
- $a \notin \mathsf{nodes}(\widehat{c})$ and $\widehat{b} \in \mathsf{nodes}(\widehat{c})$: by the definition of the unfolding operation there is only one case, in Step 3, which is $(a_0, \widehat{a}_1) \in r_1^{\mathcal{I}_{\mathcal{A}'}}$. In this case we know that $(a_0, a_1) \in r_1^{\mathcal{I}_\mathcal{A}}$.

In all cases we have that for $a, b \in \Delta^{\mathcal{I}_{\mathcal{A}'}}$, $(a, b) \in r^{\mathcal{I}_{\mathcal{A}'}}$ implies $(h_*(a), h_*(b)) \in r^{\mathcal{I}_\mathcal{A}}$. ❑

Before we show Lemma 6 we need the following lemma, which shows the progress of our unfolding operations.

Lemma 18. *Let $\mathcal{I}_n$ be the minimal ABox computed in the n-th iteration in Line 5 of Algorithm 3. Assume $\mathcal{I}_n$ has a cycle. For all $n \geq 0$, $|\Delta^{\mathcal{I}_{n+1}}| > |\Delta^{\mathcal{I}_n}|$.*

Proof. By assumption $\mathcal{I}_n$ has a cycle c. Let $\mathcal{I}'_n$ be the result of unfolding c and $\mathcal{I}_{n+1}$ be the result of minimizing $\mathcal{I}'_n$. Let $h_* : \mathcal{I}'_n \to \mathcal{I}_n$ be the homomorphism defined in Lemma 17. Let $g = h_*|_{\Delta^{\mathcal{I}_{n+1}}}$ be h_* restricted to $\Delta^{\mathcal{I}_{n+1}} \subseteq \Delta^{\mathcal{I}'_n}$. Since $\mathcal{I}_{n+1}$ is a subinterpretation of $\mathcal{I}'_n$, $g : \mathcal{I}_{n+1} \to \mathcal{I}_n$ is a homomorphism.

Claim 1 $g : \mathcal{I}_{n+1} \to \mathcal{I}_n$ is a surjective homomorphism.

Suppose g is not surjective. Since $\mathcal{I}_{n+1}$ is a countermodel (Condition 1 of minimal ABoxes) there is $C \sqsubseteq A \in \mathcal{T}$ such that $a \in (C \setminus A)^{\mathcal{I}_{n+1}}$, with $a \in \Delta^{\mathcal{I}_{n+1}}$. Let $\mathcal{J}$ be the subinterpretation of $\mathcal{I}_n$ determined by the range of g. By the unfolding definition,

$a \in A^{\mathcal{I}_{n+1}}$ iff $g(a) \in A^{\mathcal{I}_n}$. Then $g(a) \in (C \setminus A)^{\mathcal{J}}$. Since $\mathcal{I}_n$ is minimal, if g is not surjective then this contradicts Condition 2 of minimal ABoxes.

Claim 2 Suppose $g : \mathcal{I}_{n+1} \to \mathcal{I}_n$ is an injective homomorphism. Then, for $d_1, d_2 \in \Delta^{\mathcal{I}_{n+1}}$, $(g(d_1), g(d_2)) \in r^{\mathcal{I}_n}$ implies $(d_1, d_2) \in r^{\mathcal{I}_{n+1}}$.

Suppose this is not the case and there is $d_1, d_2 \in \Delta^{\mathcal{I}_{n+1}}$ such that $(g(d_1), g(d_2)) \in r^{\mathcal{I}_n}$ and $(d_1, d_2) \notin r^{\mathcal{I}_{n+1}}$. Let $\mathcal{J}$ be the result of removing $(g(d_1), g(d_2))$ from $r^{\mathcal{I}_n}$. Since g is injective, $g : \mathcal{I}_{n+1} \to \mathcal{J}$ is also a homomorphism. As $\mathcal{I}_{n+1}$ is a countermodel (Condition 1 of minimal ABoxes) there is $C \sqsubseteq A \in \mathcal{T}$ such that $a \in (C \setminus A)^{\mathcal{I}_{n+1}}$, with $a \in \Delta^{\mathcal{I}_{n+1}}$. Then, $g'(a) \in C^{\mathcal{J}}$. By the unfolding definition, $a \in A^{\mathcal{I}_{n+1}}$ iff $g(a) \in A^{\mathcal{I}_n}$. By definition of $\mathcal{J}$, $g(a) \in A^{\mathcal{I}_n}$ iff $g'(a) \in A^{\mathcal{J}}$. Then $g'(a) \in (C \setminus A)^{\mathcal{J}}$. Since $\mathcal{J}$ is $\mathcal{I}_n$ with $(g(d_1), g(d_2))$ removed from $r^{\mathcal{I}_n}$, this contradicts the fact that $\mathcal{I}_n$ is role minimal (Condition 3 of minimal ABoxes).

Claim 3 $g : \mathcal{I}_{n+1} \to \mathcal{I}_n$ is not an injective homomorphism.

As g is surjective (Claim 1), a or $\widehat{a}$ is in $\Delta^{\mathcal{I}_{n+1}}$. Suppose that g is injective. Then, for each $a \in \mathsf{nodes}(c)$, only one of $\{a, \widehat{a}\}$ are in $\Delta^{\mathcal{I}_{n+1}}$. Recall that cycle c is a sequence $a_0 \cdot r_1 \cdot a_1 \cdot ... \cdot r_k \cdot a_k$, with $a_0 = a_k$, where we defined w.l.g. that $(a_0, a_1) \in r_1^{\mathcal{I}_n}$. Assume $a_0 \in \Delta^{\mathcal{I}_{n+1}}$ (the case where $\widehat{a}_0 \in \Delta^{\mathcal{I}_{n+1}}$ is similar). Now, we make a case distinction:

- $k = 1$: in this case, the cycle is a reflexive element. That is, $a_0 = a_1$ and, then, $(a_0, a_0) \in r_1^{\mathcal{I}_n}$. By definition of the unfolding operation, $(a_0, a_0) \notin r_1^{\mathcal{I}_{n+1}} \subseteq r_1^{\mathcal{I}'_n}$. So if $a_0 \in \Delta^{\mathcal{I}_{n+1}}$ then $(g(a_0), g(a_0)) \in r_1^{\mathcal{I}_n}$ and $(a_0, a_0) \notin r_1^{\mathcal{I}_{n+1}}$, which contradicts Claim 2.
- $k > 1$: By definition of the unfolding operation $r_1(a_0, \widehat{a}_1)$, $r_1(\widehat{a}_0, a_1)$ are the only role assertions between elements in $\mathsf{nodes}(c)$ and $\mathsf{nodes}(\widehat{c})$ in $\mathcal{I}'_n$, where $\mathcal{I}'_n$ is $\mathcal{I}_n$ with cycle c unfolded. This means that (*) neither $(\widehat{a}_i, a_{i+1})$ or $(a_{i+1}, \widehat{a}_i)$ are in $r_{i+1}^{\mathcal{I}_{n+1}} \subseteq r_{i+1}^{\mathcal{I}'_n}$, for $1 \leq i < k$. By assumption $a_0 \in \Delta^{\mathcal{I}_{n+1}}$. Then, for $1 \leq i < k$, $\widehat{a}_i \in \Delta^{\mathcal{I}_{n+1}}$, otherwise we would obtain a contradiction with Claim 2. Since $a_0 = a_k \in \Delta^{\mathcal{I}_{n+1}}$ and g is injective, $\widehat{a}_0 = \widehat{a}_k \notin \Delta^{\mathcal{I}_{n+1}}$. By the same argument, as $\widehat{a}_{k-1} \in \Delta^{\mathcal{I}_{n+1}}$ we have that $a_{k-1} \notin \Delta^{\mathcal{I}_{n+1}}$. By definition of c, either (a_{k-1}, a_k) or (a_k, a_{k-1}) are in $r_k^{\mathcal{I}_n}$. Together with the fact (*) that neither $(\widehat{a}_{k-1}, a_k)$ or $(a_k, \widehat{a}_{k-1})$ are in $r_k^{\mathcal{I}_{n+1}} \subseteq r_k^{\mathcal{I}'_n}$, this contradicts Claim 2.

Then g is not injective. Since g is surjective (Claim 1) and not injective (Claim 3), $|\Delta^{\mathcal{I}_{n+1}}| > |\Delta^{\mathcal{I}_n}|$. ❑

Proof of Lemma 6. Algorithm 3 computes a minimal tree shaped ABox $\mathcal{A}$ with size polynomial in $|\mathcal{T}|$ and runs in polynomially many steps in $|\mathcal{T}|$ and $|\mathcal{A}|$.

Proof. The fact that the computed ABox is tree shaped follows from Line 3. Also, by Lemma 5 the size of the ABox is bounded by $|\mathcal{T}|$. So it remains to show that Algorithm 3 terminates after at polynomially many steps in $|\mathcal{T}|$ and $|\mathcal{A}|$. By Lemma 5, Lines 2 and

5 is polynomial in $|\mathcal{A}|$ and $|\mathsf{N_C} \cap \Sigma_\mathcal{T}|$. Also, unfolding a cycle c in Line 4 is linear in $|\mathcal{A}|$. It remains to show that the number of iterations is bounded by $|\mathcal{T}|$. Let $\mathcal{I}_n$ be the minimal ABox computed in the n-th iteration in Line 5 of Algorithm 3. By Lemma 5, for all n iterations of Algorithm 3, in Line 5 $|\Delta^{\mathcal{I}_n}|$ is bounded by $|\mathcal{T}|$. By Lemma 18, after each $n+1$-th iteration of the algorithm, $|\Delta^{\mathcal{I}_{n+1}}|$ increases by at least one element with respect to $|\Delta^{\mathcal{I}_n}|$. So the number of iterations is bounded by $|\mathcal{T}|$. ❑

E Proofs for Limits of Polynomial Time Learnability

To show Lemma 7, we first show Lemma 22, which uses Lemmas 20 and 21 from [12]. We also require the following lemma from [11], which characterizes concept inclusions entailed by acyclic $\mathcal{EL}$ TBoxes.

Lemma 19 ([11]). *Let $\mathcal{T}$ be an acyclic $\mathcal{EL}$ TBox, r a role name and D an $\mathcal{EL}$ concept expression. 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 $\mathcal{EL}$ concept expressions 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 expression 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 20. *Let $\mathbf{B} = F_1 \sqcap ... \sqcap F_n$, where $F_i \in \{E_i, \bar{E}_i\}$. 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 expression C over Σ_n, if $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq \exists \boldsymbol{\sigma}.\mathbf{B}$ 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 expression C'; or*
2. *$\models C \sqsubseteq \exists \boldsymbol{\sigma}.\mathbf{B}$.*

Proof. We prove the proposition by induction on m. Since for all F_i occurring in $\mathbf{B}$, $\mathcal{T}_L^{\mathbf{B}}$ does not contain an inclusion $F_i \equiv C$, where C is an $\mathcal{EL}$ concept expression, by Lemma 19, there is a concept name Z such that $\mathcal{T}_L^{\mathbf{B}} \models Z \sqsubseteq F_i$. Then, for $m = 0$, C is of the form $Z \sqcap C'$, where Z is a concept name, C' is an $\mathcal{EL}$ concept expression and $\mathcal{T}_L^{\mathbf{B}} \models Z \sqsubseteq F_i$. This is only possible if Z is F_i itself. As this holds for all F_i, we have that $\models C \sqsubseteq \mathbf{B}$.
For $m > 0$. By Lemma 19 we have one of the following two cases:

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

❑

Lemma 21 ([12]). *For any acyclic $\mathcal{EL}$ TBox $\mathcal{T}$, any inclusion $A \sqsubseteq C \in \mathcal{T}$ and any concept expression 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 for Lemma 22.

Lemma 22. *For all $\mathcal{EL}$ concept inclusions $C \sqsubseteq D$ over Σ_n where $\mathbf{B}$ is not a subconcept of C:*

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

Proof. To prove this lemma we argue by induction on the structure of D and show the following.

Claim 1 For all $\mathcal{EL}$ concept inclusions $C \sqsubseteq D$ over Σ_n where $\mathbf{B} \in \mathfrak{B}_n$ is not a subconcept of C, if there is $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$ then:

- either $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$ for every $L \in \mathfrak{L}_n$ and every $\mathbf{B} \in \mathfrak{B}_n$ or
- for each $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$ there is $\boldsymbol{\sigma}$ in L and a sequence of roles $t_1, \ldots, t_m$, $m \geq 0$, such that $\models D \sqsubseteq \exists t_1. \cdots \exists t_m.\exists \boldsymbol{\sigma}.\top$, where $t_j \in \{r, s\}$, $1 \leq j \leq m$.

We assume throughout the proof that in all cases $\mathbf{B}$ is not a subconcept of C and that there exists some $L_0 \in \mathfrak{L}_n$ such that $\mathcal{T}_{L_0}^{\mathbf{B}} \models C \sqsubseteq D$.

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

- D is one of X_i, A_i, B_i, E_i or $\bar{E}_i$ for $1 \leq i \leq n$. By Lemma 19, C is of the form $Z \sqcap C'$, for some concept name Z, and $\mathcal{T}_{L_0}^{\mathbf{B}} \models Z \sqsubseteq D$. If D is one of X_i, A_i, B_i, E_i or $\bar{E}_i$, then this can only be the case if $Z = D$. But then for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$.
- D is X_0. By Lemma 19, C is of the form $Z \sqcap C'$, for some concept name Z, and $\mathcal{T}_{L_0}^{\mathbf{B}} \models Z \sqsubseteq X_0$. This is the case if either $Z = X_0$, or Z is one of A, A_i, B_i, $1 \leq i \leq n$. In either case, for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq X_0$.
- D is A. If C is of the form $A \sqcap C'$ or, for all i, $1 \leq i \leq n$, A_i or B_i is a conjunct of C, then for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq A$. 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 $L = (\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$ be such that $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq A$. Notice that $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq A$, for $L = (\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$, if, and only if, $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq X_0 \sqcap \exists \boldsymbol{\sigma}_1.\mathbf{B} \sqcap \cdots \sqcap \exists \boldsymbol{\sigma}_n.\mathbf{B}$. By Lemma 20, for such a $\mathcal{T}_L^{\mathbf{B}}$ we must have $\models C \sqsubseteq \exists \boldsymbol{\sigma}_j.\mathbf{B}$, but then this is not possible as $\mathbf{B}$ is not a subconcept of C.

Thus if D is a concept name then either for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$ or there exists no $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$, where $\mathbf{B}$ is not a subconcept of C.

Induction step. If $D = D_1 \sqcap D_2$, then $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$ if, and only if, $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D_i$, $i \in \{1, 2\}$. So the lemma follows from the induction hypothesis.

For $D = \exists t.D'$, suppose that there is $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq D$. Then, by Lemma 19, either (i) there exists a conjunct Z of C, Z a concept name, such that $\mathcal{T}_L^{\mathbf{B}} \models Z \sqsubseteq \exists t.D'$ or (ii) there exists a conjunct $\exists t.C'$ of C with $\mathcal{T}_L^{\mathbf{B}} \models C' \sqsubseteq D'$. Consider cases (i) and (ii).

(i) Let Z be a conjunct of C such that Z is a concept name and $\mathcal{T}_L^{\mathbf{B}} \models Z \sqsubseteq \exists t.D'$. Notice that Z cannot be E_i or $\bar{E}_i$ as for no $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models E_i \sqsubseteq \exists t.D'$ or $\mathcal{T}_L^{\mathbf{B}} \models \bar{E}_i \sqsubseteq \exists t.D'$. Consider the remaining possibilities.
- Z is one of X_i, $0 \leq i \leq n$. It is easy to see that for $L, L' \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models X_i \sqsubseteq \exists t.D'$ if, and only if $\mathcal{T}_{L'}^{\mathbf{B}} \models X_i \sqsubseteq \exists t.D'$. Thus, for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models Z \sqsubseteq \exists t.D'$.
- Z is one of A_i, B_i for $1 \leq i \leq n$. By Lemma 21, $\mathcal{T}_L^{\mathbf{B}} \models Z \sqsubseteq \exists t.D'$ if, and only if, $\mathcal{T}_L^{\mathbf{B}} \models X_0 \sqcap \exists \boldsymbol{\sigma}_i.\mathbf{B} \sqsubseteq \exists t.D'$. By Lemma 19, either $\mathcal{T}_L^{\mathbf{B}} \models X_0 \sqsubseteq \exists t.D'$ or $\mathcal{T}_L^{\mathbf{B}} \models \exists \boldsymbol{\sigma}_i.\mathbf{B} \sqsubseteq \exists t.D'$. If $\mathcal{T}_L^{\mathbf{B}} \models X_0 \sqsubseteq \exists t.D'$ then for every $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq \exists t.D'$. Now, suppose that $\exists t.D'$ is such that $\mathcal{T}_L^{\mathbf{B}} \not\models X_0 \sqsubseteq \exists t.D'$ and $\mathcal{T}_L^{\mathbf{B}} \models \exists \boldsymbol{\sigma}_i.\mathbf{B} \sqsubseteq \exists t.D'$. By inductive applications of Lemma 19, this is only possible when $\models \exists t.D' \sqsubseteq \exists \boldsymbol{\sigma}_i.\top$. Notice that since all $\boldsymbol{\sigma}_i$ are unique, there exists exactly one $L \in \mathfrak{L}_n$ (namely, L is L_0) such that $\mathcal{T}_L^{\mathbf{B}} \models Z \sqsubseteq \exists \boldsymbol{\sigma}_i.F$, where $\models \mathbf{B} \sqsubseteq F$.
- Z is A. Suppose that for some $L = (\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_n) \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models A \sqsubseteq \exists t.D'$, equivalently $\mathcal{T}_L^{\mathbf{B}} \models X_0 \sqcap \exists \boldsymbol{\sigma}_1.\mathbf{B} \sqcap \ldots \exists \boldsymbol{\sigma}_n.\mathbf{B} \sqsubseteq \exists t.D'$. By Lemma 19, either $\mathcal{T}_L^{\mathbf{B}} \models X_0 \sqsubseteq \exists t.D'$ or $\mathcal{T}_L^{\mathbf{B}} \models \exists \boldsymbol{\sigma}_i.\mathbf{B} \sqsubseteq \exists t.D'$, for some $i : 1 \leq i \leq n$, so, as above, unless $\mathcal{T}_L^{\mathbf{B}} \models X_0 \sqsubseteq \exists t.D'$ we have that $\models \exists t.D' \sqsubseteq \exists \boldsymbol{\sigma}_i.\top$, as required.

(ii) Let $\exists t.C'$ be a conjunct of C with $\mathcal{T}_L^{\mathbf{B}} \models C' \sqsubseteq D'$. The induction hypothesis implies that either (a) for every $L \in \mathfrak{L}_n$ we have that $\mathcal{T}_L^{\mathbf{B}} \models C' \sqsubseteq D'$ or (b) for each $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models C' \sqsubseteq D'$ there is $\boldsymbol{\sigma}$ in L and a sequence of roles $t_1, \ldots, t_m$, $m \geq 0$, such that $\models D' \sqsubseteq \exists t_1. \cdots \exists t_m.\exists \boldsymbol{\sigma}.\top$, where $t_j \in \{r, s\}$, $1 \leq j \leq m$. In case (a), we have that for every $L \in \mathfrak{L}_n$, $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq \exists t.D'$. In case (b), if for each $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models C' \sqsubseteq D'$ there is $\boldsymbol{\sigma}$ such that $\models D' \sqsubseteq \exists t_1 \ldots \exists t_m.\exists \boldsymbol{\sigma}.\top$ then same happens with $\exists t.D'$ (notice that for every $L \in \mathfrak{L}_n$ and every $\mathbf{B} \in \mathfrak{B}_n$ we have that $\mathcal{T}_L^{\mathbf{B}} \models C' \sqsubseteq D'$ iff $\mathcal{T}_L^{\mathbf{B}} \models \exists t.C' \sqsubseteq \exists t.D'$).

To summarize, either $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq \exists t.D'$ for every $L \in \mathfrak{L}_n$ and every $\mathbf{B} \in \mathfrak{B}_n$ or $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq \exists t.D'$ implies that $\models \exists t.D' \sqsubseteq \exists t_0. \ldots \exists t_m.\exists \boldsymbol{\sigma}.\top$, $m \geq 0$, for some $\boldsymbol{\sigma}$ in L. Since all $\boldsymbol{\sigma}$ are unique for each $L \in \mathfrak{L}_n$, the number of different $L \in \mathfrak{L}_n$ such that $\mathcal{T}_L^{\mathbf{B}} \models C \sqsubseteq \exists t.D'$ does not exceed $|D|$. ❑

Before we proceed to the proof of Lemma 7, we need Lemma 23.

Definition 3. *The unravelling $\mathcal{A}^u$ of $\mathcal{A}$ into a (possibly infinite) tree is defined as:*

- $\mathsf{Ind}(\mathcal{A}^u)$ *is the set of sequences $b_0 r_0 \cdots r_{n-1} b_n$ with $b_0, \ldots, b_n \in \mathsf{Ind}(\mathcal{A})$, $r_0, \ldots, r_{n-1} \in \mathsf{N_R}$ and $r_i(b_i, b_{i+1}) \in \mathcal{A}$;*
- *for each $A(b) \in \mathcal{A}$ and $\alpha = b_0 r_0 \cdots r_{n-1} \cdot b_n \in \mathsf{Ind}(\mathcal{A}^u)$ with $b_n = b$, we have $A(\alpha) \in \mathcal{A}^u$;*
- *for each $\alpha = b_0 r_0 \cdots r_{n-1} b_n \in \mathsf{Ind}(\mathcal{A}^u)$ with $n > 0$, we have $r_{n-1}(b_0 r_0 \cdots r_{n-1} b_{n-1}, \alpha) \in \mathcal{A}^u$.*

Lemma 23. *For any ABox $\mathcal{A}$ and $\mathcal{EL}$ concept expression D over Σ_n there is a concept expression $C_\mathcal{A}$ such that $a \in C_\mathcal{A}^{\mathcal{I}_\mathcal{A}}$ and, for every $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$:*

$$(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models D(a) \quad \textit{iff} \quad \mathcal{T}_L^{\mathbf{B}} \models C_\mathcal{A} \sqsubseteq D.$$

Proof. Let $\mathcal{A}^u$ be the unravelling of $\mathcal{A}$. Let $\mathcal{T}_L^{\mathbf{B}}$ be a TBox for some arbitrary $\mathbf{B} \in \mathfrak{B}_n$ and $L \in \mathfrak{L}_n$. By definition of $\mathcal{A}^u$ we have that $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models D(a)$ iff $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}^u) \models D(a)$. Denote as $\mathcal{A}_a^{u,k}$ the subtree of $\mathcal{A}^u$ which is rooted in $a \in \mathsf{Ind}(\mathcal{A}^u)$ and has depth $k \in \mathbb{N}$. Let $\mathcal{T}_L^{\mathbf{B}'}$ be the result of removing $X_0 \sqcap \exists \boldsymbol{\sigma}_1.\mathbf{B} \sqcap \cdots \sqcap \exists \boldsymbol{\sigma}_n.\mathbf{B} \sqsubseteq A$ from $\mathcal{T}_L^{\mathbf{B}}$. Then, $(\mathcal{T}_L^{\mathbf{B}'}, \mathcal{A}^u) \models D(a)$ iff $(\mathcal{T}_L^{\mathbf{B}'}, \mathcal{A}_a^{u,|D|}) \models D(a)$. Let $\mathcal{I}_{\mathcal{T}_L^{\mathbf{B}'}, \mathcal{A}^u}$ be the canonical model of $\mathcal{T}_L^{\mathbf{B}'}$ and $\mathcal{A}^u$. By definition of $\mathcal{T}_L^{\mathbf{B}}$, one can make it a canonical model of $\mathcal{T}_L^{\mathbf{B}}$ and $\mathcal{A}^u$ by including $d \in A^{\mathcal{I}_{\mathcal{T}_L^{\mathbf{B}'}, \mathcal{A}^u}}$ whenever $d \in (X_0 \sqcap \exists \boldsymbol{\sigma}_1.\mathbf{B} \sqcap \cdots \sqcap \exists \boldsymbol{\sigma}_n.\mathbf{B})^{\mathcal{I}_{\mathcal{T}_L^{\mathbf{B}'}, \mathcal{A}^u}}$. Then, $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}^u) \models D(a)$ iff $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}_a^{u,|D|+n}) \models D(a)$. Let $C_\mathcal{A}$ be the concept expression corresponding to the tree interpretation of $\mathcal{A}_a^{u,|D|+n}$ rooted in a. We have that, for every $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$, $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models D(a)$ iff $\mathcal{T}_L^{\mathbf{B}} \models C_\mathcal{A} \sqsubseteq D$. ❑

We can now proceed to the proof of Lemma 7. We say that an $\mathcal{EL}$ concept expression C *occurs* in an ABox $\mathcal{A}$ if there exists $a \in \mathsf{Ind}(\mathcal{A})$ such that $\mathcal{A} \models C(a)$. For $a, b \in \mathsf{Ind}(\mathcal{A})$, a *role chain* from a to b is a sequence $a_0 \cdot t_0 \cdot \ldots \cdot t_{n-1} \cdot a_n$ with $a_0 = a$, $a_n = b$ and $t_i(a_i, a_{i+1}) \in \mathcal{A}$, where $0 \leq i \leq n-1$ and $t_i \in \{r, s\}$.

Proof of Lemma 7. For any ABox $\mathcal{A}$, any $\mathcal{EL}$ concept assertion $D(a)$ over Σ_n, and any $a \in \mathsf{Ind}(\mathcal{A})$, if there is $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$ such that $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$ then:

- either $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$, for every $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$, or
- $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$ for at most $|D|$ elements $L \in \mathfrak{L}_n$, or
- $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$ for at most $|\mathcal{A}|$ elements $\mathbf{B} \in \mathfrak{B}_n$.

Proof. We make a case distinction:

1. for all i, $1 \leq i \leq n$, $E_i \sqcap \bar{E}_i$ does not occur in $\mathcal{A}$: first notice that in this case, for every $\mathcal{EL}$ concept expression C over Σ_n, $a \in \mathsf{Ind}(\mathcal{A})$ and $\mathcal{T}_L^{\mathbf{B}} \in S$:

   $$(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models C(a) \quad \text{iff} \quad (\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models C(a).$$

   For any $\mathcal{A}$ and $\mathcal{EL}$ concept expression D over Σ_n, by Lemma 23, there is a concept expression $C_\mathcal{A}$ such that $a \in C_\mathcal{A}^{\mathcal{I}_\mathcal{A}}$ and, for every $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$:

   $$(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models D(a) \quad \text{iff} \quad \mathcal{T}_L^{\mathbf{B}} \models C_\mathcal{A} \sqsubseteq D.$$

   If there is no $\mathbf{B} \in \mathfrak{B}_n$ such that $\mathbf{B}$ occurs in $\mathcal{A}$ then the Lemma follows from Corollary 22. Notice that although our construction of $C_\mathcal{A}$ is not polynomial, Corollary 22 does not impose any restriction in the size of $C_\mathcal{A}$. Otherwise, since for all i, $1 \leq i \leq n$, $E_i \sqcap \bar{E}_i$ does not occur in $\mathcal{A}$, we have that the number of $\mathbf{B} \in \mathfrak{B}_n$ such that $\mathbf{B}$ occurs in $\mathcal{A}$ is linear in the size of $\mathcal{A}$. So the number of $\mathbf{B} \in \mathfrak{B}_n$ such that $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$ does not exceed the size of $\mathcal{A}$.

2. there is i, $1 \leq i \leq n$, such that $E_i \sqcap \bar{E}_i$ occurs in $\mathcal{A}$: let $E_i \sqcap \bar{E}_{i\mathcal{A}}$ be the set of individuals $b \in \mathsf{Ind}(\mathcal{A})$ such that $E_i \sqcap \bar{E}_i(b) \in \mathcal{A}$. By construction of $\mathcal{T}^*$, for every ABox $\mathcal{A}$ and every $\mathcal{EL}$ concept expression D over Σ_n we have that $(\mathcal{T}^*, \mathcal{A}) \models D(b)$, where $b \in E_i \sqcap \bar{E}_{i\mathcal{A}}$. Then, in particular, for every $L \in \mathfrak{L}_n$ we have that $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(b)$. For $a \in \mathsf{Ind}(\mathcal{A}) \setminus E_i \sqcap \bar{E}_{i\mathcal{A}}$ we make a case distinction:
   - there is a role chain from a to some $b \in E_i \sqcap \bar{E}_{i\mathcal{A}}$: by definition of $\mathcal{T}^*$, as $(E_i \sqcap \bar{E}_i) \sqsubseteq A$ for every $1 \leq i \leq n$ and every $A \in \Sigma_n \cap \mathsf{N_C}$, we have that $(\mathcal{T}^*, \mathcal{A}) \models (E_1 \sqcap \bar{E}_1)(b)$. Then, since $\{\exists r.(E_1 \sqcap \bar{E}_1) \sqsubseteq (E_1 \sqcap \bar{E}_1), \exists s.(E_1 \sqcap \bar{E}_1) \sqsubseteq (E_1 \sqcap \bar{E}_1)\} \subseteq \mathcal{T}^*$, we have that $(\mathcal{T}^*, \mathcal{A}) \models (E_1 \sqcap \bar{E}_1)(a)$. In this case, by the argument above, for every $L \in \mathfrak{L}_n$ and every $\mathcal{EL}$ concept expression D over Σ_n, we have that $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$.
   - for all $b \in E_i \sqcap \bar{E}_{i\mathcal{A}}$, there is no role chain from a to b: let $\mathcal{A}' = \mathcal{A} \setminus \{E_i(b), \bar{E}_i(b) \mid b \in E_i \sqcap \bar{E}_{i\mathcal{A}}\}$. Since in this case, for all $b \in E_i \sqcap \bar{E}_{i\mathcal{A}}$, there is no role chain from a to b, we have that, for every $\mathcal{EL}$ concept expression D, $\mathcal{A} \models D(a)$ iff $\mathcal{A}' \models D(a)$. By definition of $\mathcal{A}'$, $E_i \sqcap \bar{E}_i$ does not occur in $\mathcal{A}'$, then the lemma follows as in Case 1.

❑

The next lemma from [12] prepares the proof of Lemma 8.

Lemma 24 ([12]). *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 $t_1, \ldots t_l$ 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 of Lemma 8. For any $n > 1$ and any $\mathcal{EL}$ TBox $\mathcal{H}$ in Σ_n with $|\mathcal{H}| < 2^n$, there exists an ABox $\mathcal{A}$, an individual $a \in \mathsf{Ind}(\mathcal{A})$ and an $\mathcal{EL}$ concept expression D over Σ_n such that (i) the size of $\mathcal{A}$ plus the size of D does not exceed $6n$ and (ii) if $(\mathcal{H}, \mathcal{A}) \models D(a)$ then $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}) \models D(a)$ for at most one $L \in \mathfrak{L}_n$ and if $(\mathcal{H}, \mathcal{A}) \not\models D(a)$ then for every $L \in \mathfrak{L}_n$ we have $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$.

Proof. As $\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^* \models C \sqsubseteq D$ iff $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}_{C,a}) \models D(a)$, where $\mathcal{A}_{C,a}$ is an ABox with canonical model isomorphic to the tree interpretation of C with root ρ_C mapped to $a \in \mathsf{Ind}(\mathcal{A})$, to prove this lemma we show the following claim.

Claim 1 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 $\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^* \models C \sqsubseteq D$ 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 $\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^* \models C \sqsubseteq D$.

We define an exponentially large TBox $\mathcal{T}_\cap$ and use it to prove that one can select an $\mathcal{EL}$ concept inclusion $C \sqsubseteq D$ in such a way that either $\mathcal{H} \models C \sqsubseteq D$ and $\mathcal{T}_\cap \not\models C \sqsubseteq D$, or vice versa. Then, the oracle can return $(\mathcal{A}_{C,a}, D(a))$ as a counterexample, where $\mathcal{A}_{C,a}$ is the tree shaped ABox corresponding to the $\mathcal{EL}$ concept expression C rooted in $a \in \mathsf{Ind}(\mathcal{A})$.

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

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

Let $\mathcal{A}_{C_{\boldsymbol{b}}}$ be the ABox corresponding to a concept expression $C_{\boldsymbol{b}}$, as defined above. Since, for all i, $1 \leq i \leq n$, $E_i \sqcap \bar{E}_i$ does not occur in $\mathcal{A}_{C_{\boldsymbol{b}}}$, we have that $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}_{C_{\boldsymbol{b}}}) \models C(a)$ iff $(\mathcal{T}_L^{\mathbf{B}}, \mathcal{A}_{C_{\boldsymbol{b}}}) \models C(a)$. Then, in the following we only consider $\mathcal{T}_L^{\mathbf{B}}$. Consider the 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 $\mathcal{T}_L^{\mathbf{B}}$, for every $L \in \mathfrak{L}_n$, 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 expression 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 24, 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 $\mathcal{T}_L^{\mathbf{B}} \models C_{\boldsymbol{b}} \sqsubseteq \exists t_1 \cdots \exists t_l.Y$ for at most one $L \in \mathfrak{L}_n$.

Suppose for some $L \in \mathfrak{L}_n$ we have $\mathcal{T}_L^{\mathbf{B}} \models C_{\boldsymbol{b}} \sqsubseteq \exists t_1. \cdots \exists t_l.Y$. By Lemma 19, there is A_j or B_j such that $\mathcal{T}_L^{\mathbf{B}} \models A_j \sqsubseteq \exists t_1. \cdots \exists t_l.Y$ (or $\mathcal{T}_L^{\mathbf{B}} \models B_j \sqsubseteq \exists t_1. \cdots \exists t_l.Y$, 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 implied by $\mathbf{B}$. Since every $\boldsymbol{\sigma}_j$ is unique, for every $L' \in \mathfrak{L}_n$ such that $L' \neq L$ we have $\mathcal{T}_{L'}^{\mathbf{B}} \not\models C_{\boldsymbol{b}} \sqsubseteq \exists \boldsymbol{\sigma}_j.Y$.

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 expression 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 $C = Z_1 \sqcap \dots \sqcap Z_m \sqcap \exists t_1.C'_1 \sqcap \dots \sqcap \exists t_l.C'_l$, where $Z_1, \dots, Z_m$ 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 $j = 1, \dots l$. As $\mathcal{H} \models \mathcal{T}_\cap$ we have $\mathcal{H} \models X_0 \sqsubseteq \exists t_j.C'_j$, for $j = 1, \dots l$. So $\mathcal{H} \models Z_1 \sqcap \dots \sqcap Z_m \sqcap X_0 \sqsubseteq A$.

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 $\mathcal{T}_L^{\mathbf{B}} \not\models Z_1 \sqcap \dots \sqcap Z_m \sqcap X_0 \sqsubseteq A$, for any $L \in \mathfrak{L}_n$. Notice that in the worst case $Z_1 \sqcap \dots \sqcap Z_m$ contains the conjunction of all Σ_n-concept names, except A_i, B_i, so the size of $Z_1 \sqcap \dots \sqcap Z_m \sqcap X_0 \sqsubseteq A$ does not exceed $6n$, and $Z_1 \sqcap \dots \sqcap Z_m \sqcap X_0 \sqsubseteq A$ is as required.

Assume that $Z_0 \sqcap \dots \sqcap Z_m \sqcap X_0$ 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 $\mathcal{T}_L^{\mathbf{B}} \models C_{\boldsymbol{b}} \sqsubseteq B_i$. The size of $C_{\boldsymbol{b}} \sqsubseteq B_i$ does not exceed $6n$, so it is as required.

Assume that $Z_0 \sqcap \dots \sqcap Z_m \sqcap X_0$ 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 $\mathcal{T}_L^{\mathbf{B}} \models C_{\boldsymbol{b}} \sqsubseteq A_i$. 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.

❑

Proof of Theorem 5. The $\mathcal{EL}$ data retrieval framework is not polynomially exact learnable.

Proof. Assume that TBoxes are polynomial time learnable in the open data model. 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 $S_1 = \mathfrak{L}_n$ and $S_2 = \mathfrak{B}_n$. We follow Angluin's strategy of removing elements from S_1 and S_2 in such a way that the learner cannot distinguish between any of the remaining $\mathcal{T}_L^{\mathbf{B}}$ TBoxes encoded by $L \in S_1$ and $\mathbf{B} \in S_2$. The strategy is as follows.

Given an membership query $(\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^*, \mathcal{A}) \models D(a)$, with $\mathcal{A} \models C(a)$, if $\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^* \models C \sqsubseteq D$ for every $L \in \mathfrak{L}_n$ and every $\mathbf{B} \in \mathfrak{B}_n$, then the answer is 'yes'; otherwise the answer is 'no' and all $L \in \mathfrak{L}_n$ and $\mathbf{B} \in \mathfrak{B}_n$ with $\mathcal{T}_L^{\mathbf{B}} \cup \mathcal{T}^* \models C \sqsubseteq D$ are removed from S_1 and S_2, respectively. By Lemma 7, at most the size of D elements can be removed from S_1 or at most the size of $\mathcal{A}$ elements can be removed from S_2. Given an equivalence query with $\mathcal{H}$, the answer is 'no' and a concept inclusion $C \sqsubseteq D$ not entailed by $\mathcal{H}$ such that $\mathcal{T}_{L'}^{\mathbf{B}} \cup \mathcal{T}^* \models C \sqsubseteq D$ for at most one $L' \in \mathfrak{L}_n$ is guaranteed by Lemma 8. Then a counterexample $(\mathcal{T}, \mathcal{A}) \models D(a)$ with $\mathcal{A} \models C(a)$ and bounded by $6n$ is produced (consider the size of a query or a counterexample $(\mathcal{T}, \mathcal{A}) \models D(a)$ as being the size of $\mathcal{A}$ plus the size of concept expression D).

As all counterexamples produced are bounded by $6n$, 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 7 and 8, at most $(p(n, 6n))^2$ elements are removed from S_1 and S_2 during the run of the algorithm. But then the algorithm cannot distinguish between any TBoxes $\mathcal{T}_L^{\mathbf{B}}$ and $\mathcal{T}_{L'}^{\mathbf{B}'}$ for $L \neq L' \in S_1$ and $\mathbf{B} \neq \mathbf{B}' \in S_2$ based on the given answers and we have derived a contradiction.

❑