# SAT-Based PAC Learning of Description Logic Concepts

**Balder ten Cate**[1] , **Maurice Funk**[2,3] , **Jean Christoph Jung**[4] and **Carsten Lutz**[2,3]
[1]ILLC, University of Amsterdam
[2]Leipzig University
[3]Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI)
[4]TU Dortmund University
b.d.tencate@uva.nl, maurice.funk@uni-leipzig.de, jean.jung@tu-dortmund.de, carsten.lutz@uni-leipzig.de

## Abstract

We propose *bounded fitting* as a scheme for learning description logic concepts in the presence of ontologies. A main advantage is that the resulting learning algorithms come with theoretical guarantees regarding their generalization to unseen examples in the sense of PAC learning. We prove that, in contrast, several other natural learning algorithms fail to provide such guarantees. As a further contribution, we present the system SPELL which efficiently implements bounded fitting for the description logic $\mathcal{ELH}^r$ based on a SAT solver, and compare its performance to a state-of-the-art learner.

## 1 Introduction

In knowledge representation, the manual curation of knowledge bases (KBs) is time consuming and expensive, making learning-based approaches to knowledge acquisition an attractive alternative. We are interested in description logics (DLs) where *concepts* are an important class of expressions, used for querying KBs and also as central building blocks for ontologies. The subject of learning DL concepts from labeled data examples has received great interest, resulting in various implemented systems such as DL-Learner, DL-Foil, and YINYANG [Bühmann *et al.*, 2016; Fanizzi *et al.*, 2018; Iannone *et al.*, 2007]. These systems take a set of positively and negatively labeled examples and an ontology $\mathcal{O}$, and try to construct a concept that fits the examples w.r.t. $\mathcal{O}$. The related *fitting problem*, which asks to decide the existence of a fitting concept, has also been studied intensely [Lehmann and Hitzler, 2010; Funk *et al.*, 2019; Jung *et al.*, 2021].

The purpose of this paper is to propose a new approach to concept learning in DLs that we call *bounded fitting*, inspired by both bounded model checking as known from systems verification [Biere *et al.*, 1999] and by Occam algorithms from computational learning theory [Blumer *et al.*, 1989]. The idea of bounded fitting is to search for a fitting concept of bounded size, iteratively increasing the size bound until a fitting is found. This approach has two main advantages, which we discuss in the following.

First, it comes with formal guarantees regarding the generalization of the returned concept from the training data to previously unseen data. This is formalized by Valiant's framework of *probably approximately correct (PAC) learning* [Valiant, 1984]. Given sufficiently many data examples sampled from an unknown distribution, bounded fitting returns a concept that with high probability $\delta$ has a classification error bounded by some small $\epsilon$. It is well-known that PAC learning is intimately linked to Occam algorithms which guarantee to find a hypothesis of small size [Blumer *et al.*, 1989; Board and Pitt, 1992]. By design, algorithms following the bounded fitting paradigm are Occam, and as a consequence the number of examples needed for generalization depends only linearly on $1/\delta$, $1/\epsilon$, and the size of the target concept to be learned. This generalization guarantee holds independently of the DL used to formulate concepts and ontologies. In contrast, no formal generalization guarantees have been established for DL concept learning approaches.

The second advantage is that, in important cases, bounded fitting enables learning based on SAT solvers and thus leverages the practical efficiency of these systems. We consider ontologies formulated in the description logic $\mathcal{ELH}^r$ and concepts formulated in $\mathcal{EL}$, which may be viewed as a core of the ontology language OWL 2 EL. In this case, the *size-restricted fitting problem*, which is defined like the fitting problem except that the maximum size of fitting concepts to be considered is given as an additional input (in unary), is NP-complete; it is thus natural to implement bounded fitting using a SAT solver. For comparison, we mention that the unbounded fitting problem is ExpTime-complete in this case [Funk *et al.*, 2019].

As a further contribution of the paper, we analyze the generalization ability of other relevant approaches to constructing fitting $\mathcal{EL}$-concepts. We start with algorithms that return fittings that are 'prominent' from a logical perspective in that they are most specific or most general or of minimum quantifier depth among all fittings. Algorithms with such characteristics and their applications are discussed in [ten Cate *et al.*, 2023]. Notably, constructing fittings via direct products of positive examples yields most specific fittings [Zarrieß and Turhan, 2013; Jung *et al.*, 2020]. Our result is that, even without ontologies, these types of algorithms are not *sample-efficient*, that is, no polynomial amount of positive and negative examples is sufficient to achieve generalization in the PAC sense.

We next turn to algorithms based on so-called downward refinement operators which underlie all implemented DL learning systems that we are aware of. We consider two natural such operators that are rather similar to one another and combine them with a breadth-first search strategy. The first operator can be described as exploring 'most-general specializations' of the current hypotheses and the second one does the same, but is made 'artificially Occam' (with, most likely, a negative impact on practicality). We prove that while the first operator does not lead to a not sample-efficient algorithm (even without ontologies), the second one does. This leaves open whether or not implemented systems based on refinement operators admit generalization guarantees, as they implement complex heuristics and optimizations.

As our final contribution we present SPELL, a SAT-based system that implements bounded fitting of $\mathcal{EL}$-concepts under $\mathcal{ELH}^r$-ontologies. We evaluate SPELL on several datasets and compare it to the only other available learning system for $\mathcal{EL}$ that we are aware of, the *$\mathcal{EL}$ tree learner (ELTL)* incarnation of the *DL-Learner* system [Bühmann *et al.*, 2016]. We find that the running time of SPELL is almost always significantly lower than that of ELTL. Since, as we also show, it is the size of the target concept that has most impact on the running time, this means that SPELL can learn larger target queries than ELTL. We also analyze the relative strengths and weaknesses of the two approaches, identifying classes of inputs on which one of the systems performs significantly better than the other one. Finally, we make initial experiments regarding generalization, where both systems generalize well to unseen data, even on very small samples. While this is expected for SPELL, for ELTL it may be due to the fact that some of the heuristics prefer fittings of small size, which might make ELTL an Occam algorithm.

Proof details are provided in the appendix.

**Related work.** Cohen and Hirsh identified a fragment of the early DL CLASSIC that admits sample-efficient PAC learning, even in polynomial time [Cohen and Hirsh, 1994]. For several DLs such as $\mathcal{EL}$ and CLASSIC, concepts are learnable in polynomial time in Angluin's framework of exact learning with membership and equivalence queries [Frazier and Pitt, 1996; ten Cate and Dalmau, 2021; Funk *et al.*, 2021; Funk *et al.*, 2022b]. The algorithms can be transformed in a standard way into sample-efficient polynomial time PAC learning algorithms that, however, additionally use membership queries to an oracle [Angluin, 1987]. It is known that sample-efficient PAC learning under certain assumptions implies the existence of Occam algorithms [Board and Pitt, 1992]. These assumptions, however, do not apply to the learning tasks studied here.

## 2 Preliminaries

**Concepts, ontologies, queries.** Let $\mathsf{N_C}$, $\mathsf{N_R}$, and $\mathsf{N_I}$ be countably infinite sets of *concept names*, *role names*, and *individual names*, respectively. An *$\mathcal{EL}$-concept* is formed according to the syntax rule

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

where $A$ ranges over $\mathsf{N_C}$ and $r$ over $\mathsf{N_R}$. A concept of the form $\exists r.C$ is called an *existential restriction* and the *quantifier depth* of a concept is the maximum nesting depth of existential restrictions in it. An *$\mathcal{ELH}^r$-ontology* $\mathcal{O}$ is a finite set of *concept inclusions (CIs)* $C \sqsubseteq D$, *role inclusions* $r \sqsubseteq s$, and *range assertions* $\mathsf{ran}(r) \sqsubseteq C$ where $C$ and $D$ range over $\mathcal{EL}$-concepts and $r, s$ over role names. An *$\mathcal{EL}$-ontology* is an $\mathcal{ELH}^r$-ontology that uses neither role inclusions nor range assertions. We also sometimes mention *$\mathcal{ELI}$-concepts* and *$\mathcal{ELI}$-ontologies*, which extend their $\mathcal{EL}$-counterparts with inverse roles $r^-$ that can be used in place of role names. See [Baader *et al.*, 2017] for more information. A *database* $\mathcal{D}$ (also called *ABox* in a DL context) is a finite set of *concept assertions* $A(a)$ and *role assertions* $r(a, b)$ where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and $a, b \in \mathsf{N_I}$. We use $\mathsf{adom}(\mathcal{D})$ to denote the set of individual names that are used in $\mathcal{D}$. A *signature* is a set of concept and role names, in this context uniformly referred to as *symbols*. For any syntactic object $O$, such as a concept or an ontology, we use $\mathsf{sig}(O)$ to denote the set of symbols used in $O$ and $||O||$ to denote the *size* of $O$, that is, the number of symbols used to write $O$ encoded as a word over a finite alphabet, with each occurrence of a concept or role name contributing a single symbol.

The semantics is defined in terms of *interpretations* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ where $\Delta^\mathcal{I}$ is the *domain* of $\mathcal{I}$ and $\cdot^\mathcal{I}$ assigns a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ to every $A \in \mathsf{N_C}$ and a binary relation $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ to every $r \in \mathsf{N_R}$. The *extension* $C^\mathcal{I}$ of $\mathcal{EL}$-concepts $C$ is then defined as usual [Baader *et al.*, 2017]. An interpretation $\mathcal{I}$ *satisfies* a concept or role inclusion $\alpha \sqsubseteq \beta$ if $\alpha^\mathcal{I} \subseteq \beta^\mathcal{I}$, a range assertion $\mathsf{ran}(r) \sqsubseteq C$ if the projection of $r^\mathcal{I}$ to the second component is contained in $C^\mathcal{I}$, a concept assertion $A(a)$ if $a \in A^\mathcal{I}$, and a role assertion $r(a, b)$ if $(a, b) \in r^\mathcal{I}$. We say that $\mathcal{I}$ is a *model* of an ontology/database if it satisfies all inclusions/assertions in it.

An $\mathcal{EL}$-concept $C$ can be viewed as an *$\mathcal{EL}$-query (ELQ)* $q$, as follows. Let $\mathcal{D}$ be a database and $\mathcal{O}$ an $\mathcal{ELH}^r$-ontology. Then $a \in \mathsf{adom}(\mathcal{D})$ is an *answer* to $q$ on $\mathcal{D}$ w.r.t. $\mathcal{O}$ if $a \in C^\mathcal{I}$ for all models $\mathcal{I}$ of $\mathcal{D}$ and $\mathcal{O}$. In a similar way, we may view $\mathcal{ELI}$-concepts as *$\mathcal{ELI}$-queries (ELIQs)*. We will from now on mostly view $\mathcal{EL}$-concepts as ELQs. This does not, however, restrict their use, which may be as actual queries or as concepts used as building blocks for ontologies.

An *ontology-mediated query (OMQ) language* is a pair $(\mathcal{L}, \mathcal{Q})$ with $\mathcal{L}$ an ontology language and $\mathcal{Q}$ a query language, such as $(\mathcal{ELH}^r, \text{ELQ})$ and $(\mathcal{ELI}, \text{ELIQ})$. For a query language $\mathcal{Q}$ and signature $\Sigma$, we use $\mathcal{Q}_\Sigma$ to denote the set of all queries $q \in \mathcal{Q}$ with $\mathsf{sig}(q) \subseteq \Sigma$. All query languages considered in this paper are unary, that is, they return a subset of $\mathsf{adom}(\mathcal{D})$ as answers. We use $q(\mathcal{D} \cup \mathcal{O})$ to denote the set of answers to $q$ on $\mathcal{D}$ w.r.t. $\mathcal{O}$. For an $\mathcal{L}$-ontology $\mathcal{O}$ and queries $q_1, q_2$, we write $\mathcal{O} \models q_1 \sqsubseteq q_2$ if for all databases $\mathcal{D}$, $q_1(\mathcal{D} \cup \mathcal{O}) \subseteq q_2(\mathcal{D} \cup \mathcal{O})$. We say that $q_1$ and $q_2$ are *equivalent* w.r.t. $\mathcal{O}$, written $\mathcal{O} \models q_1 \equiv q_2$, if $\mathcal{O} \models q_1 \sqsubseteq q_2$ and $\mathcal{O} \models q_2 \sqsubseteq q_1$. When $\mathcal{O} = \emptyset$, we write $q_1 \sqsubseteq q_2$ and $q_1 \equiv q_2$.

Every ELQ $q$ may be viewed as a database $\mathcal{D}_q$ in an obvious way, e.g. $q = \exists r.\exists s.A$ as $\mathcal{D}_q = \{r(a_q, a_1), s(a_1, a_2), A(a_2)\}$. Let $\mathcal{D}_1, \mathcal{D}_2$ be databases and $\Sigma$ a signature. A *$\Sigma$-simulation* from $\mathcal{D}_1$ to $\mathcal{D}_2$ is a relation $S \subseteq \mathsf{adom}(\mathcal{D}_1) \times \mathsf{adom}(\mathcal{D}_2)$ such that for all $(a_1, a_2) \in S$:

1. if $A(a_1) \in \mathcal{D}_1$ with $A \in \Sigma$, then $A(a_2) \in \mathcal{D}_2$;
2. if $r(a_1, b_1) \in \mathcal{D}_1$ with $r \in \Sigma$, there is $r(a_2, b_2) \in \mathcal{D}_2$ such that $(b_1, b_2) \in S$.

For $a_1 \in \mathsf{adom}(\mathcal{D}_1)$ and $a_2 \in \mathsf{adom}(\mathcal{D}_2)$, we write

$(\mathcal{D}_1, a_1) \preceq_\Sigma (\mathcal{D}_2, a_2)$ if there is a $\Sigma$-simulation $S$ from $\mathcal{D}_1$ to $\mathcal{D}_2$ with $(a_1, a_2) \in S$. We generally drop the mention of $\Sigma$ in case that $\Sigma = \mathsf{N_C} \cup \mathsf{N_R}$. The following well-known lemma links simulations to ELQs.

**Lemma 1.** *For all ELQs $q$, databases $\mathcal{D}$, and $a \in \mathsf{adom}(\mathcal{D})$: $a \in q(\mathcal{D})$ iff $(\mathcal{D}_q, a_q) \preceq (\mathcal{D}, a)$. Consequently, for all ELQs $q, p$: $q \sqsubseteq p$ iff $(\mathcal{D}_p, a_p) \preceq (\mathcal{D}_q, a_q)$.*

**Fitting.** A *pointed database* is a pair $(\mathcal{D}, a)$ with $\mathcal{D}$ a database and $a \in \mathsf{adom}(\mathcal{D})$. A *labeled data example* takes the form $(\mathcal{D}, a, +)$ or $(\mathcal{D}, a, -)$, the former being a *positive example* and the latter a *negative example*.

Let $\mathcal{O}$ be an ontology, $\mathcal{Q}$ a query language, and $E$ a collection of labeled data examples. A query $q \in \mathcal{Q}$ *fits* $E$ w.r.t. $\mathcal{O}$ if $a \in q(\mathcal{D} \cup \mathcal{O})$ for all $(\mathcal{D}, a, +) \in E$ and $a \notin q(\mathcal{D} \cup \mathcal{O})$ for all $(\mathcal{D}, a, -) \in E$. We then call $E$ a *$q$-labeled data example w.r.t. $\mathcal{O}$*. We say that $q$ is a *most specific fitting* if $\mathcal{O} \models q \sqsubseteq q'$ for every $q' \in \mathcal{Q}$ that fits $E$, and that it is *most general* if $\mathcal{O} \models q' \sqsubseteq q$ for every $q' \in \mathcal{Q}$ that fits $E$.

**Example 1.** *Consider the collection $E_0$ of examples $(\{r(a,a), A(a), B(a)\}, a, +)$, $(\{A(a), r(a,b), B(b)\}, a, +)$, $(\{r(a,b)\}, b, -)$. It has several ELQ fittings, the most specific one being $A \sqcap \exists r.B$. There is no most general fitting ELQ as both $A$ and $\exists r.B$ fit, but no common generalization does.*

A *fitting algorithm* for an OMQ language $(\mathcal{L}, \mathcal{Q})$ is an algorithm that takes as input an $\mathcal{L}$-ontology $\mathcal{O}$ and a collection of labeled data examples $E$ and returns a query $q \in \mathcal{Q}$ that fits $E$ w.r.t. $\mathcal{O}$, if such a $q$ exists, and otherwise reports non-existence or does not terminate. The *size-restricted fitting problem* for $(\mathcal{L}, \mathcal{Q})$ means to decide, given a collection of labeled data examples $E$, an $\mathcal{L}$-ontology $\mathcal{O}$, and an $s \geq 1$ in unary, whether there is a query $q \in \mathcal{Q}$ with $||q|| \leq s$ that fits $E$ w.r.t. $\mathcal{O}$.

It is well-known that for every database $\mathcal{D}$ and $\mathcal{ELH}^r$-ontology $\mathcal{O}$, we can compute in polynomial time a database $\mathcal{U}_{\mathcal{D},\mathcal{O}}$ that is *universal for ELQs* in the sense that $a \in q(\mathcal{D} \cup \mathcal{O})$ iff $a \in q(\mathcal{U}_{\mathcal{D},\mathcal{O}})$ for all ELQs $q$ and $a \in \mathsf{adom}(\mathcal{D})$ [Lutz *et al.*, 2009]. Given a collection of labeled data examples $E$ and an $\mathcal{ELH}^r$-ontology $\mathcal{O}$, we denote with $E_\mathcal{O}$ the collection obtained from $E$ by replacing each (positive or negative) example $(\mathcal{D}, a, \cdot)$ with $(\mathcal{U}_{\mathcal{D},\mathcal{O}}, a, \cdot)$. The following proposition shows that a fitting algorithm for ELQ without ontologies also gives rise to a fitting algorithm for $(\mathcal{ELH}^r, \text{ELQ})$ with at most a polynomial increase in running time. It is immediate from the definition of universality.

**Proposition 1.** *An ELQ $q$ fits a collection of labeled examples $E$ w.r.t. an $\mathcal{ELH}^r$-ontology $\mathcal{O}$ iff $q$ fits $E_\mathcal{O}$ w.r.t. $\emptyset$.*

We remark that in contrast to ELQs, finite databases that are universal for ELIQs need not exist [Funk *et al.*, 2022a].

**PAC learning.** We recall the definition of PAC learning, in a formulation that is tailored towards OMQ languages. Let $P$ be a probability distribution over pointed databases and let $q_T$ and $q_H$ be queries, the target and the hypothesis. The error of $q_H$ relative to $q_T$ and $P$ is

$$\mathsf{error}_{P,q_T}(q_H) = \Pr_{(\mathcal{D},a)\sim P}(a \in q_H(\mathcal{D} \cup \mathcal{O}) \,\Delta\, q_T(\mathcal{D} \cup \mathcal{O}))$$

where $\Delta$ denotes symmetric difference and $\mathsf{Pr}_{(\mathcal{D},a)\sim P}\, X$ is the probability of $X$ when drawing $(\mathcal{D}, a)$ randomly according to $P$.

**Definition 1.** *A* PAC learning algorithm *for an OMQ language $(\mathcal{L}, \mathcal{Q})$ is a (potentially randomized) algorithm $\mathfrak{A}$ associated with a function $m : \mathbb{R}^2 \times \mathbb{N}^4 \to \mathbb{N}$ such that*

- *$\mathfrak{A}$ takes as input an $\mathcal{L}$-ontology $\mathcal{O}$ and a collection of labeled data examples $E$;*
- *for all $\epsilon, \delta \in (0, 1)$, all $\mathcal{L}$-ontologies $\mathcal{O}$, all finite signatures $\Sigma$, all $s_Q, s_E \geq 0$, all probability distributions $P$ over pointed databases $(\mathcal{D}, c)$ with $\mathsf{sig}(\mathcal{D}) \subseteq \Sigma$ and $||\mathcal{D}|| \leq s_E$, and all $q_T \in \mathcal{Q}_\Sigma$ with $||q_T|| \leq s_Q$, the following holds: when running $\mathfrak{A}$ on $\mathcal{O}$ and a collection $E$ of at least $m(1/\delta, 1/\epsilon, ||\mathcal{O}||, |\Sigma|, s_Q, s_E)$ labeled data examples that are $q_T$-labeled w.r.t. $\mathcal{O}$ and drawn according to $P$, it returns a hypothesis $q_H$ such that with probability at least $1 - \delta$ (over the choice of $E$), we have $\mathsf{error}_{P,q_T}(q_H) \leq \epsilon$.*

*We say that* $\mathfrak{A}$ has sample size $m$ *and call* $\mathfrak{A}$ sample-efficient *if $m$ is a polynomial.*

Note that a PAC learning algorithm is not required to terminate if no fitting query exists. It would be desirable to even attain *efficient* PAC learning which additionally requires $\mathfrak{A}$ to be a polynomial time algorithm. However, ELQs are known to not be efficiently PAC learnable even without ontologies, unless RP = NP [Kietz, 1993; ten Cate *et al.*, 2022]. The same is true for ELIQs and any other class of conjunctive queries that contains all ELQs.

## 3 Bounded Fitting and Generalization

We introduce bounded fitting and analyze when fitting algorithms are PAC learning algorithms.

**Definition 2.** *Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language and let $\mathfrak{A}$ be an algorithm for the size-restricted fitting problem for $(\mathcal{L}, \mathcal{Q})$. Then* BOUNDED-FITTING$_\mathfrak{A}$ *is the algorithm that, given a collection of labeled data examples $E$ and an $\mathcal{L}$-ontology $\mathcal{O}$, runs $\mathfrak{A}$ with input $(E, \mathcal{O}, s)$ to decide whether there is a $q \in \mathcal{Q}$ with $||q|| \leq s$ that fits $E$ w.r.t. $\mathcal{O}$, for $s = 1, 2, 3 \ldots$, returning a fitting query as soon as it finds one.*

**Example 2.** *Consider again Example 1. For $s = 1$, bounded fitting tries the candidates $\top, A, B, \exists r.\top$ and returns the fitting $A$. If started on $E_0$ extended with $(\{A(a)\}, a, -)$, it finds one of the fitting ELQs $A \sqcap \exists r.\top$ and $\exists r.B$ in Round 2.*

In spirit, bounded fitting focusses on finding fitting queries when they exist, and not on deciding the existence of a fitting query. This is in analogy with bounded model checking, which focusses on finding counterexamples rather than on proving that no such examples exist. If an upper bound on the size of fitting queries is known, however, we can make bounded fitting terminate by reporting non-existence of a fitting query once the bound is exceeded. This is more of theoretical than of practical interest since the size bounds tend to be large. For ELQs without ontologies and for $(\mathcal{EL}, \text{ELQ})$, for instance, it is double exponential [Funk, 2019]. It thus seems more realistic to run an algorithm that decides the existence of a fitting in parallel to bounded fitting and to report the result as soon as one of the algorithms terminates. There are also important cases where fitting existence is undecidable, such as for the OMQ language $(\mathcal{ELI}, \text{ELIQ})$ [Funk *et al.*, 2019].

Bounded fitting may be used also in such cases as long as the size-restricted fitting problem is still decidable. This is the case for $(\mathcal{ELI}, \text{ELIQ})$, as a direct consequence of query evaluation to be decidable in this OMQ language [Baader *et al.*, 2008], see Appendix H.

A major advantage of bounded fitting is that it yields a sample-efficient PAC learning algorithm with sample size linear in the size of the target query. This is because bounded fitting is an Occam algorithm which essentially means that it produces a fitting query that is at most polynomially larger than the fitting query of minimal size [Blumer *et al.*, 1989].[1]

**Theorem 1.** *Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language. Every bounded fitting algorithm for $(\mathcal{L}, \mathcal{Q})$ is a (sample-efficient) PAC learning algorithm with sample size $O\big(\frac{1}{\epsilon} \cdot \log\big(\frac{1}{\epsilon}\big) \cdot \log\big(\frac{1}{\delta}\big) \cdot \log|\Sigma| \cdot ||q_T||\big)$.*

We remark that bounded fitting is *robust* in that other natural measures of query size (such as the number of existential restrictions) and enumeration sequences such as $s = 1, 2, 4, 8, \ldots$ also lead to sample-efficient PAC learning algorithms. This results in some flexibility in implementations.

We next show that many other fitting algorithms are not sample-efficient when used as PAC learning algorithms. We start with algorithms that return fittings which are most specific or most general or of minimum quantifier depth. No such algorithm is a sample-efficient PAC learning algorithm, even without ontologies.

**Theorem 2.** *If $\mathfrak{A}$ is a fitting algorithm for ELQs that satisfies one of the conditions below, then $\mathfrak{A}$ is not a sample-efficient PAC learning algorithm.*

1. *$\mathfrak{A}$ always produces a most specific fitting, if it exists;*
2. *$\mathfrak{A}$ always produces a most general fitting, if it exists;*
3. *$\mathfrak{A}$ produces a fitting of minimal quantifier depth, if a fitting exists.*

The proof of Theorem 2 relies on duals of finite relational structures, which are widely known in the form of homomorphism duals [Nesetril and Tardif, 2000]. Here, we introduce the new notion of *simulation* duals.

Let $(\mathcal{D}, a)$ be a pointed database and $\Sigma$ a signature. A set $M$ of pointed databases is a $\Sigma$-*simulation dual* of $(\mathcal{D}, a)$ if for all pointed databases $(\mathcal{D}', a')$, the following holds:

$$(\mathcal{D}, a) \preceq_\Sigma (\mathcal{D}', a') \quad \text{iff} \quad (\mathcal{D}', a') \not\preceq_\Sigma (\mathcal{D}'', a'') \quad \text{for all } (\mathcal{D}'', a'') \in M.$$

For illustration, consider the simulation dual $M$ of $(\mathcal{D}_q, a_q)$ for an ELQ $q$. Then every negative example for $q$ has a simulation into an element of $M$ and $q$ is the most general ELQ that fits $\{(\mathcal{D}, a, -) \mid (\mathcal{D}, a) \in M\}$. We exploit this in the proof of Theorem 2. Moreover, we rely on the fact that ELQs have simulation duals of polynomial size. In contrast, (non-pointed) homomorphism duals of tree-shaped databases may become exponentially large [Nesetril and Tardif, 2005].

[1]A precise definition of Occam algorithms is based on the notion of VC-dimension; it is not crucial to the main part of the paper, details can be found in the appendix.

**Theorem 3.** *Given an ELQ $q$ and a finite signature $\Sigma$, a $\Sigma$-simulation dual $M$ of $(\mathcal{D}_q, a_q)$ of size $||M|| \leq 3 \cdot |\Sigma| \cdot ||q||^2$ can be computed in polynomial time. Moreover, if $\mathcal{D}_q$ contains only a single $\Sigma$-assertion that mentions $a_q$, then $M$ is a singleton.*

The notion of simulation duals is of independent interest and we develop it further in the appendix. We show that Theorem 3 generalizes from databases $\mathcal{D}_q$ to all pointed databases $(\mathcal{D}, a)$ such that the directed graph induced by the restriction of $\mathcal{D}$ to the individuals reachable (in a directed sense) from $a$ is a DAG. Conversely, databases that are not of this form do not have finite simulation duals. We find it interesting to recall that DAG-shaped databases do in general not have finite homomorphism duals [Nesetril and Tardif, 2000].

Using Theorem 3, we now prove Point 2 of Theorem 2. Points 1 and 3 are proved in the appendix.

*Proof.* To highlight the intuitions, we leave out some minor technical details that are provided in the appendix. Assume to the contrary of what we aim to show that there is a sample-efficient PAC learning algorithm that produces a most general fitting ELQ, if it exists, with associated polynomial function $m\colon \mathbb{R}^2 \times \mathbb{N}^4$ as in Definition 1. As target ELQs $q_T$, we use concepts $C_i$ where $C_0 = \top$ and $C_i = \exists r.(A \sqcap B \sqcap C_{i-1})$. Thus, $C_i$ is an $r$-path of length $i$ in which every non-root node is labeled with $A$ and $B$.

Choose $\Sigma = \{A, B, r\}$, $\delta = \epsilon = 0.5$, and $n$ large enough so that $2^n > 2m(1/\delta, 1/\epsilon, 0, |\Sigma|, 3n, 3 \cdot |\Sigma| \cdot ||C_n||^2)$. Further choose $q_T = C_n$.

We next construct negative examples; positive examples are not used. Define a set of ELQs $S = S_n$ where

$$S_0 = \{\top\} \quad S_i = \{\exists r.(\alpha \sqcap C) \mid C \in S_{i-1}, \alpha \in \{A, B\}\}.$$

Note that the ELQs in $S$ resemble $q_T$ except that every node is labeled with only one of the concept names $A, B$. Now consider any $q \in S$. Clearly, $q_T \sqsubseteq q$. Moreover, the pointed database $(\mathcal{D}_q, a_q)$ contains a single assertion that mentions $a_q$. By Theorem 3, $q$ has a singleton $\Sigma$-simulation dual $\{(\mathcal{D}'_q, a'_q)\}$ with $||\mathcal{D}'_q|| \leq 3 \cdot |\Sigma| \cdot ||C_n||^2$. We shall use these duals as negative examples.

The two crucial properties of $S$ are that for all $q \in S$,

1. $q$ is the most general ELQ that fits $(\mathcal{D}'_q, a'_q, -)$;
2. for all $T \subseteq S$, $q \notin T$ implies $\bigsqcap_{p \in T} p \not\sqsubseteq q$.

By Point 1 and since $q_T \sqsubseteq q$, each $(\mathcal{D}'_q, a'_q)$ is also a negative example for $q_T$.

Let the probability distribution $P$ assign probability $\frac{1}{2^n}$ to all $(\mathcal{D}'_q, a'_q)$ with $q \in S$ and probability 0 to all other pointed databases. Now assume that the algorithm is started on a collection of $m(1/\delta, 1/\epsilon, 0, |\Sigma|, 3n, 3 \cdot |\Sigma| \cdot ||C_n||^2)$ labeled data examples $E$ drawn according to $P$. It follows from Point 1 that $q_H = \bigsqcap_{(\mathcal{D}'_q, a'_q) \in E} q$ is the most general ELQ that fits $E$. Thus, (an ELQ equivalent to) $q_H$ is output by the algorithm.

To obtain a contradiction, it suffices to show that with probability $1 - \delta$, we have $\mathsf{error}_{P, q_T}(q_H) > \epsilon$. We argue that, in fact, $q_H$ violates all (negative) data examples that are not in the sample $E$, that is, $a_q \in q_H(\mathcal{D}_p)$ for all $p \in S$ with $(\mathcal{D}'_p, a'_p) \notin E$. The definition of $P$ and choice of $n$ then yield that with probability 1, $\mathsf{error}_{P, q_T}(q_H) = \frac{|S| - |E|}{|S|} > \frac{1}{2}$.

Thus consider any $p \in S$ such that $(\mathcal{D}'_p, a'_p) \notin E$. It follows from Point 2 that $q_H \not\sqsubseteq p$ and the definition of duals may now be used to derive $a'_p \in q_H(\mathcal{D}'_p)$ as desired. □

## 4 Refinement Operators

We discuss fitting algorithms based on refinement operators, used in implemented systems such as ELTL, and show that the generalization abilities of such algorithms subtly depend on the exact operator (and strategy) used.

Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language. A *(downward) refinement* of a query $q \in \mathcal{Q}$ w.r.t. an $\mathcal{L}$-ontology $\mathcal{O}$ is any $p \in \mathcal{Q}$ such that $\mathcal{O} \models p \sqsubseteq q$ and $\mathcal{O} \not\models q \sqsubseteq p$. A *(downward) refinement operator* for $(\mathcal{L}, \mathcal{Q})$ is a function $\rho$ that associates every $q \in Q_\Sigma$, $\mathcal{L}$-ontology $\mathcal{O}$, and finite signature $\Sigma$ with a set $\rho(q, \mathcal{O}, \Sigma)$ of downward refinements $p \in \mathcal{Q}_\Sigma$ of $q$ w.r.t. $\mathcal{O}$. The operator $\rho$ is *ideal* if it is finite and complete where $\rho$ is

1. *finite* if $\rho(q, \mathcal{O}, \Sigma)$ is finite for all $q$, $\mathcal{O}$, and finite $\Sigma$, and
2. *complete* if for all finite signatures $\Sigma$ and all $q, p \in \mathcal{Q}_\Sigma$, $\mathcal{O} \models p \sqsubseteq q$ implies that there is a finite $\rho, \mathcal{O}, \Sigma$-*refinement sequence* from $q$ to $p$, that is, a sequence of queries $q_1, \ldots, q_n$ such that $q = q_1$, $q_{i+1} \in \rho(q_i, \mathcal{O}, \Sigma)$ for $1 \leq i < n$, and $\mathcal{O} \models q_n \equiv p$.

When $\mathcal{O}$ is empty, we write $\rho(q, \Sigma)$ in place of $\rho(q, \mathcal{O}, \Sigma)$.

For $(\mathcal{EL}, \text{ELQ})$ and thus also for $(\mathcal{ELH}^r, \text{ELQ})$, it is known that no ideal refinement operator exists [Kriegel, 2019]. This problem can be overcome by making use of Proposition 1 and employing an ideal refinement operator for ELQs without ontologies, which does exist [Lehmann and Haase, 2009]. But also these refinement operators are not without problems. It was observed in [Kriegel, 2021] that for any such operator, non-elementarily long refinement sequences exist, potentially impairing the practical use of such operators. We somewhat relativize this by the following observation. A refinement operator $\rho$ for $(\mathcal{L}, \mathcal{Q})$ is $f$*-depth bounded*, for $f : \mathbb{N} \to \mathbb{N}$, if for all $q, p \in \mathcal{Q}$ and all $\mathcal{L}$-ontologies $\mathcal{O}$ with $\mathcal{O} \models p \sqsubseteq q$, there exists a $\rho, \mathcal{O}, \Sigma$-refinement sequence from $q$ to $p$ that is of length at most $f(||p||)$.

**Theorem 4.** *Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ-language. If $(\mathcal{L}, \mathcal{Q})$ has an ideal refinement operator, then it has a $2^{O(n)}$-depth bounded ideal refinement operator.*

The depth bounded operator in Theorem 4 is obtained by starting with some operator $\rho$ and adding to each $\rho(q, \mathcal{O}, \Sigma)$ all $p \in \mathcal{Q}_\Sigma$ such that $\mathcal{O} \models p \sqsubseteq q$, $\mathcal{O} \not\models q \sqsubseteq p$, and $||p|| \leq ||q||$. Note that the size of queries is used in an essential way, as in Occam algorithms.

A refinement operator by itself is not a fitting algorithm as one also needs a strategy for applying the operator. We use breadth-first search as a simple yet natural such strategy.

We consider two related refinement operators $\rho_1$ and $\rho_2$ for ELQs. The definition of both operators refers to (small) query size, inspired by Occam algorithms. Let $q$ be an ELQ. Then $\rho_1(q, \Sigma)$ is the set of all $p \in \text{ELQ}_\Sigma$ such that $p \sqsubseteq q$, $q \not\sqsubseteq p$, and $||p|| \leq 2||q|| + 1$. The operator $\rho_2$ is defined like $\rho_1$ except that we include in $\rho_2(q, \Sigma)$ only ELQs $p$ that are a *(downward) neighbor* of $q$, that is, for all ELQs $p'$, $p \sqsubseteq p' \sqsubseteq q$ implies $p' \sqsubseteq p$ or $q \sqsubseteq p'$. The following lemma shows that $\rho_2(q, \Sigma)$ actually contains *all* neighbors of $q$ with $\text{sig}(q) \subseteq \Sigma$, up to equivalence. An ELQ $q$ is *minimal* if there is no ELQ $p$ such that $||p|| < ||q||$ and $p \equiv q$.

**Lemma 2.** *For every ELQ $q$ and minimal downward neighbor $p$ of $q$, we have $||p|| \leq 2||q|| + 1$.*

Both $\rho_1$ and $\rho_2$ can be computed by brute force. For more elaborate approaches to computing $\rho_2$, see [Kriegel, 2021] where downward neighbors of ELQs are studied in detail.

**Lemma 3.** *$\rho_1$ and $\rho_2$ are ideal refinement operators for ELQ.*

We next give more details on what we mean by breadth-first search. Started on a collection of labeled data examples $E$, the algorithm maintains a set $M$ of candidate ELQs that fit all positive examples $E^+$ in $E$, beginning with $M = \{\top\}$ and proceeding in rounds. If any ELQ $q$ in $M$ fits $E$, then we return such a fitting $q$ with $||q||$ smallest. Otherwise, the current set $M$ is replaced with the set of all ELQs from $\bigcup_{q \in M} \rho(q, \text{sig}(E))$ that fit $E^+$, and the next round begins. For $i \in \{1, 2\}$, let $\mathfrak{A}_i$ be the version of this algorithm that uses refinement operator $\rho_i$. Although $\rho_1$ and $\rho_2$ are defined quite similarly, the behavior of the algorithms $\mathfrak{A}_1$ and $\mathfrak{A}_2$ differs.

**Theorem 5.** *$\mathfrak{A}_1$ is a sample-efficient PAC learning algorithm, but $\mathfrak{A}_2$ is not.*

To prove Theorem 5, we show that $\mathfrak{A}_1$ is an Occam algorithm while $\mathfrak{A}_2$ produces a most general fitting (if it exists), which allows us to apply Theorem 2.

The above is intended to provide a case study of refinement operators and their generalization abilities. Implemented systems use refinement operators and strategies that are more complex and include heuristics and optimizations. This makes it difficult to analyze whether implemented refinement-based systems constitute a sample-efficient PAC learner.

We comment on the ELTL system that we use in our experiments. ELTL is based on the refinement operator for $(\mathcal{ELH}^r, \text{ELQ})$ presented in [Lehmann and Haase, 2009]. That operator, however, admits only $\mathcal{ELH}^r$ ontologies of a rather restricted form: all CIs must be of the form $A \sqsubseteq B$ with $A, B$ concept *names*. Since no ideal refinement operators for unrestricted $(\mathcal{EL}, \text{ELQ})$ exist and ELTL does not eliminate ontologies in the spirit of Proposition 1, it remains unclear whether and how ELTL achieves completeness (i.e., finding a fitting whenever there is one).

## 5 The SPELL System

We implemented bounded fitting for the OMQ language $(\mathcal{ELH}^r, \text{ELQ})$ in the system SPELL (for *SAT-based PAC $\mathcal{EL}$ concept Learner*).[2] SPELL takes as input a knowledge base in OWL RDF/XML format that contains both an $\mathcal{ELH}^r$ ontology $\mathcal{O}$ and a collection $E$ of positive and negative examples, and it outputs an ELQ represented as a SPARQL query. SPELL is implemented in Python 3 and uses the PySat library to interact with the Glucose SAT solver. It provides integration into the SML-Bench benchmark framework [Westphal *et al.*, 2019].

In the first step, SPELL removes the ontology $\mathcal{O}$ by replacing the given examples $E$ with $E_\mathcal{O}$ as per Proposition 1.

[2]Available at https://github.com/spell-system/SPELL.

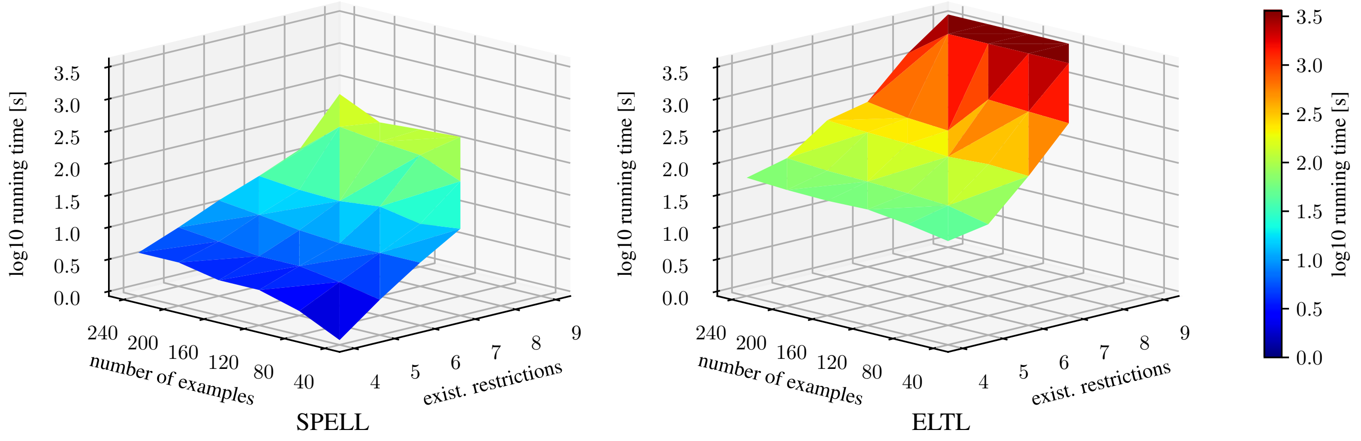

Figure 1: Yago experiment, dark red area indicates timeout (60min)

It then runs bounded fitting in the variant where in each round $n$, fitting ELQs with at most $n-1$ existential restrictions are considered (rather than fitting ELQs $q$ with $||q|| \leq n$). The existence of such a fitting is checked using the SAT solver. Also this variant of bounded fitting results in a sample-efficient PAC learning algorithm, with sample size $O\big(\frac{1}{\epsilon} \cdot \log\big(\frac{1}{\epsilon}\big) \cdot \log\big(\frac{1}{\delta}\big) \cdot |\Sigma| \cdot ||q_T||\big)$, see the appendix. We prefer this variant for implementation because it admits a more natural reduction to SAT, described next.

From $E_\mathcal{O}$ and the bound $n$, we construct a propositional formula $\varphi = \varphi_1 \wedge \varphi_2$ that is satisfiable if and only if there is an ELQ $q$ over $\Sigma = \mathsf{sig}(E_\mathcal{O})$ with at most $n-1$ existential restrictions that fits $E_\mathcal{O}$. Indeed, any model of $\varphi$ returned by the SAT solver uniquely represents a fitting ELQ $q$. More precisely, $\varphi_1$ ensures that such a model represents $\mathcal{EL}$-concepts $C_1, \dots, C_n$ where each $C_i$ only contains existential restrictions of the form $\exists r.C_j$ with $j > i$, and we take $q$ to be $C_1$. We use variables of the form $c_{i,A}$ to express that the concept name $A$ is a conjunct of $C_i$, and variables $x_{j,r}$ and $y_{i,j}$ to express that $\exists r.C_j$ is a conjunct of $C_i$. Then $\varphi_2$ enforces that the represented ELQ fits $E_\mathcal{O}$. Let $\mathcal{D}$ be the disjoint union of all databases that occur in an example in $E_\mathcal{O}$. We use variables $s_{i,a}$, with $1 \leq i \leq n$ and $a \in \mathsf{adom}(\mathcal{D})$, to express that $a \in C_i(\mathcal{D})$; the exact definition of $\varphi_2$ uses simulations and relies on Lemma 1. The number of variables in $\varphi$ is $O\big(n^2 \cdot |\mathcal{D}|\big)$, thus linear in $|\mathcal{D}|$.

We have implemented several improvements over this basic reduction of which we describe two. The first improvement is based on the simple observation that for computing a fitting ELQ with $n-1$ existential restrictions, for every example $(\mathcal{D}', a, \pm) \in E_\mathcal{O}$ it suffices to consider individuals that can be reached via at most $n-1$ role assertions from $a$. Moreover, we may restrict $\Sigma$ to symbols that occur in all $n-1$-reachable parts of the positive examples. The second improvement is based on the observation that the search space for satisfying assignments of $\varphi$ contains significant *symmetries* as the same ELQ $q$ may be encoded by many different arrangements of concepts $C_1, \dots C_n$. We add constraints to $\varphi$ so that the number of possible arrangements is reduced, breaking many symmetries. For details see the appendix.

## 6 Experimental Evaluation

We evaluate SPELL on several benchmarks[3] and compare it to the ELTL component of the DL-Learner system [Bühmann *et al.*, 2016]. Existing benchmarks do not suit our purpose as they aim at learning concepts that are formulated in more expressive DLs of the $\mathcal{ALC}$ family. As a consequence, a fitting $\mathcal{EL}$ concept almost never exists. This is the case, for example, in the often used Structured Machine Learning Benchmark [Westphal *et al.*, 2019]. We thus designed several new benchmarks leveraging various existing knowledge bases, making sure that a fitting $\mathcal{EL}$ concept always exists. We hope that our benchmarks will provide a basis also for future experimental evaluations of $\mathcal{EL}$ learning systems.

**Performance evaluation.** We carried out two experiments that aim at evaluating the performance of SPELL. The main questions are: Which parameters have most impact on the running time? And how does the running time compare to that of ELTL?

The first experiment uses the Yago 4 knowledge base which combines the concept classes of schema.org with data from Wikidata [Tanon *et al.*, 2020]. The smallest version of Yago 4 is still huge and contains over 40 million assertions. We extracted a fragment of 12 million assertions assertions that focusses on movies and famous persons. We then systematically vary the number of labeled examples and the size of the target ELQs. The latter take the form $C_n = \exists\text{actor}. \bigsqcap_{i=1}^{n} r_i.\top$ where each $r_i$ is a role name that represents a property of actors in Yago and $n$ is increased to obtain larger queries. The positive examples are selected by querying Yago with $C_n$ and the negative examples by querying Yago with generalizations of $C_n$. The results are presented in Figure 1. They show that the size of the target query has a strong impact on the running time whereas the impact of the number of positive and negative examples is much more modest. We also find that SPELL performs ∼1.5 orders of magnitude better than ELTL, meaning in particular that it can handle larger target queries.

[3]Available at https://github.com/spell-system/benchmarks.

| Sample Size | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 | 70 | 75 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ELTL | 0.77 | 0.78 | 0.85 | 0.85 | 0.86 | 0.89 | 0.90 | 0.96 | 0.96 | 0.96 | 0.96 | 0.98 | 0.98 | 0.98 | 0.98 |
| SPELL | 0.80 | 0.81 | 0.84 | 0.85 | 0.86 | 0.86 | 0.89 | 0.97 | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 |

Table 1: Generalization experiment accuracies

| | o2b-1 | o2b-2 | o2b-3 | o2b-4 | o2b-5 | o2b-6 |
|---|---|---|---|---|---|---|
| ELTL | TO | TO | 274 | 580 | 28 | 152 |
| SPELL | $< 1$ | $< 1$ | $< 1$ | $< 1$ | $< 1$ | $< 1$ |

Table 2: OWL2Bench running times [s], TO: >60min

Since Yago has only a very restricted ontology that essentially consists of inclusions $A \sqsubseteq B$ with $A, B$ concept names, we complement the above experiment with a second one based on OWL2Bench. OWL2Bench is a benchmark for ontology-mediated querying that combines a database generator with a hand-crafted ontology which extends the University Ontology Benchmark [Singh *et al.*, 2020; Zhou *et al.*, 2013]. The ontology is formulated in OWL 2 EL and we extracted its $\mathcal{ELH}^r$ fragment which uses all aspects of this DL and comprises 142 concept names, 83 role names, and 173 concept inclusions. We use datasets that contain 2500-2600 individuals and 100-200 examples, generated as in the Yago case. We designed 6 ELQs with 3-5 occurrences of concept and role names and varying topology. The results are shown in Table 2. The difference in running time is even more pronounced in this experiment, with SPELL returning a fitting ELQ almost instantaneously in all cases.[4]

**Strengths and weaknesses.** In this experiment, we aim to highlight the respective strengths and weaknesses of SPELL and ELTL or, more generally, of bounded fitting versus refinement-operator based approaches. We anticipated that the performance of bounded fitting would be most affected by the number of existential restrictions in the target query whereas the performance of refinement would be most affected by the (unique) length of the sequence $C_1, \dots, C_k$ such that $C_1 = \top$, $C_{i+1}$ is a downward neighbor of $C_i$ for $1 \leq i < k$, and $C_k$ is the target query. Let us call this the *depth* of $C_k$. The number of existential restrictions and depth are orthogonal parameters. In the $k$*-path* benchmark, we use target ELQs of the form $\exists r^k.\top$, $k \geq 1$. These should be difficult for bounded fitting when the number $k$ of existential restrictions gets large, but easy for refinement as the depth of $\exists r^k.\top$ is only $k$. In the $k$*-1-conj* benchmark, we use ELQs of the form $\exists r. \bigsqcap_{i=1}^k A_i$, $k \geq 1$. These have only one existential restriction and depth $2^k$. ELQs in the $k$*-2-conj* benchmark take the form $\exists r. \exists r. \bigsqcap_{i=1}^k A_i$ and even have depth $2^{2^k}$ [Kriegel, 2021]. These should be difficult for refinement when $k$ gets large, but easy for SPELL. There is no ontology and we use only a single positive and a single negative example, which are the target ELQ and its unique upwards neighbor (defined in analogy with downwards neighbors). The results in Table 3 confirm our expectations, with ELTL arguably degrading faster than SPELL.

[4]ELTL crashes on this benchmark unless one option ('useMinimizer') is switched off. We thus ran ELTL without useMinimizer.

| | $k$-path | | $k$-1-conj | | $k$-2-conj | |
|---|---|---|---|---|---|---|
| $k$ | ELTL | SPELL | ELTL | SPELL | ELTL | SPELL |
| 4 | 1 | <1 | 1 | <1 | 1 | <1 |
| 6 | 1 | <1 | 2 | <1 | 394 | <1 |
| 8 | 1 | <1 | 20 | <1 | TO | <1 |
| 10 | 1 | <1 | TO | <1 | TO | <1 |
| 12 | 1 | 26 | TO | <1 | TO | <1 |
| 14 | 1 | 30 | TO | <1 | TO | <1 |
| 16 | 1 | 68 | TO | <1 | TO | <1 |
| 18 | 1 | TO | TO | <1 | TO | <1 |

Table 3: Strengths/weaknesses running time [s], TO: >10min

**Generalization.** We also performed initial experiments to evaluate how well the constructed fittings generalize to unseen data. We again use the Yago benchmark, but now split the examples into training data and testing data (assuming a uniform probability distribution). Table 1 lists the median accuracies of returned fittings (over 20 experiments) where the number of examples in the training data ranges from 5 to 75. As expected, fittings returned by SPELL generalize extremely well, even when the number of training examples is remarkably small. To our surprise, ELTL exhibits the same characteristics. This may be due to the fact that some heuristics of ELTL prefer fittings of smaller size, which might make ELTL an Occam algorithm. It would be interesting to carry out more extensive experiments on this aspect.

## 7 Conclusion and Future Work

We have introduced the bounded fitting paradigm along with the SAT-based implementation SPELL for $(\mathcal{ELH}^r, \text{ELQ})$, with competitive performance and formal generalization guarantees. A natural next step is to extend SPELL to other DLs such as $\mathcal{ELI}$, $\mathcal{ALC}$, or $\mathcal{ELU}$, both with and without ontologies. We expect that, in the case without ontology, a SAT encoding of the size-restricted fitting problem will often be possible. The case with ontology is more challenging; e.g., size-restricted fitting is ExpTime-complete for $(\mathcal{ELI}, \text{ELIQ})$, see Appendix H for additional discussion. It is also interesting to investigate query languages beyond DLs such as conjunctive queries (CQs). Note that the size-restricted fitting problem for CQs is $\Sigma_2^p$-complete [Gottlob *et al.*, 1999] and thus beyond SAT solvers; one could resort to using an ASP solver or to CQs of bounded treewidth.

It would also be interesting to investigate settings in which input examples may be labeled erroneously or according to a target query formulated in different language than the query to be learned. In both cases, one has to admit non-perfect fittings and the optimization features of SAT solvers and Max-SAT solvers seem to be promising for efficient implementation.

## Acknowledgements

Balder ten Cate is supported by the European Union's Horizon 2020 research and innovation programme under grant MSCA-101031081 and Carsten Lutz by the DFG Collaborative Research Center 1320 EASE and by BMBF in DAAD project 57616814 (SECAI).

## References

[Angluin, 1987] Dana Angluin. Queries and concept learning. *Mach. Learn.*, 2(4):319–342, 1987.

[Baader *et al.*, 2008] Franz Baader, Carsten Lutz, and Sebastian Brandt. Pushing the EL envelope further. In *Proc. of OWLED*. CEUR-WS.org, 2008.

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

[Biere *et al.*, 1999] Armin Biere, Alessandro Cimatti, Edmund M. Clarke, and Yunshan Zhu. Symbolic model checking without BDDs. In *Proc. of TACAS*, pages 193–207. Springer, 1999.

[Blumer *et al.*, 1989] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, 1989.

[Board and Pitt, 1992] Raymond A. Board and Leonard Pitt. On the necessity of Occam algorithms. *Theor. Comput. Sci.*, 100(1):157–184, 1992.

[Bühmann *et al.*, 2016] Lorenz Bühmann, Jens Lehmann, and Patrick Westphal. DL-Learner - A framework for inductive learning on the semantic web. *J. Web Sem.*, 39:15–24, 2016.

[Cohen and Hirsh, 1994] William W. Cohen and Haym Hirsh. The learnability of description logics with equality constraints. *Mach. Learn.*, 17(2-3):169–199, 1994.

[Dutton and Brigham, 1986] Ronald D. Dutton and Robert C. Brigham. Computationally efficient bounds for the catalan numbers. *European Journal of Combinatorics*, 7(3):211–213, 1986.

[Fanizzi *et al.*, 2018] Nicola Fanizzi, Giuseppe Rizzo, Claudia d'Amato, and Floriana Esposito. DLFoil: Class expression learning revisited. In *Proc. of EKAW*, pages 98–113, 2018.

[Frazier and Pitt, 1996] Michael Frazier and Leonard Pitt. Classic learning. *Mach. Learn.*, 25(2-3):151–193, 1996.

[Funk *et al.*, 2019] Maurice Funk, Jean Christoph Jung, Carsten Lutz, Hadrien Pulcini, and Frank Wolter. Learning description logic concepts: When can positive and negative examples be separated? In *Proc. of IJCAI*, pages 1682–1688, 2019.

[Funk *et al.*, 2021] Maurice Funk, Jean Christoph Jung, and Carsten Lutz. Actively learning concept and conjunctive queries under $\mathcal{EL}^r$-ontologies. In *Proc. of IJCAI*, pages 1887–1893, 2021.

[Funk *et al.*, 2022a] Maurice Funk, Jean Christoph Jung, and Carsten Lutz. Exact learning of $\mathcal{ELI}$ queries in the presence of DL-Lite-Horn ontologies. In *Proc. of DL*. CEUR-WS.org, 2022.

[Funk *et al.*, 2022b] Maurice Funk, Jean Christoph Jung, and Carsten Lutz. Frontiers and exact learning of $\mathcal{ELI}$ queries under DL-Lite ontologies. In *Proc. of IJCAI*, 2022.

[Funk, 2019] Maurice Funk. Concept-by-Example in $\mathcal{EL}$ Knowledge Bases. Master's thesis, University of Bremen, 2019.

[Gottlob *et al.*, 1999] Georg Gottlob, Nicola Leone, and Francesco Scarcello. On the complexity of some inductive logic programming problems. *New Gener. Comput.*, 17(1):53–75, 1999.

[Iannone *et al.*, 2007] Luigi Iannone, Ignazio Palmisano, and Nicola Fanizzi. An algorithm based on counterfactuals for concept learning in the semantic web. *Appl. Intell.*, 26(2):139–159, 2007.

[Jung *et al.*, 2019] Jean Christoph Jung, Fabio Papacchini, Frank Wolter, and Michael Zakharyaschev. Model comparison games for Horn description logics. In *Proc. of LICS*, pages 1–14. IEEE, 2019.

[Jung *et al.*, 2020] Jean Christoph Jung, Carsten Lutz, and Frank Wolter. Least general generalizations in description logic: Verification and existence. In *Proc. of AAAI*, pages 2854–2861. AAAI Press, 2020.

[Jung *et al.*, 2021] Jean Christoph Jung, Carsten Lutz, Hadrien Pulcini, and Frank Wolter. Separating data examples by description logic concepts with restricted signatures. In *Proc. of KR*, pages 390–399, 2021.

[Kietz, 1993] Jörg-Uwe Kietz. Some lower bounds for the computational complexity of inductive logic programming. In *Proc. of ECML*, pages 115–123, 1993.

[Kriegel, 2019] Francesco Kriegel. *Constructing and Extending Description Logic Ontologies using Methods of Formal Concept Analysis*. PhD thesis, TU Dresden, 2019.

[Kriegel, 2021] Francesco Kriegel. Navigating the $\mathcal{EL}$ subsumption hierarchy. In *Proc. of DL*. CEUR-WS.org, 2021.

[Lehmann and Haase, 2009] Jens Lehmann and Christoph Haase. Ideal downward refinement in the $\mathcal{EL}$ description logic. In *Proc. of ILP*, pages 73–87, 2009.

[Lehmann and Hitzler, 2010] Jens Lehmann and Pascal Hitzler. Concept learning in description logics using refinement operators. *Mach. Learn.*, 78:203–250, 2010.

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

[Nesetril and de Mendez, 2012] Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012.

[Nesetril and Tardif, 2000] Jaroslav Nesetril and Claude Tardif. Duality theorems for finite structures (characterising gaps and good characterisations). *J. Comb. Theory, Ser. B*, 80(1):80–97, 2000.

[Nesetril and Tardif, 2005] Jaroslav Nesetril and Claude Tardif. Short answers to exponentially long questions: Extremal aspects of homomorphism duality. *SIAM J. Discret. Math.*, 19(4):914–920, 2005.

[Singh *et al.*, 2020] Gunjan Singh, Sumit Bhatia, and Raghava Mutharaju. OWL2Bench: A benchmark for OWL 2 reasoners. In *Proc. of ISWC*, pages 81–96. Springer, 2020.

[Tanon *et al.*, 2020] Thomas Pellissier Tanon, Gerhard Weikum, and Fabian M. Suchanek. YAGO 4: A reasonable knowledge base. In *Proc. of ESWC*, pages 583–596. Springer, 2020.

[ten Cate and Dalmau, 2021] Balder ten Cate and Victor Dalmau. Conjunctive queries: Unique characterizations and exact learnability. In *Proc. of ICDT*, volume 186 of *LIPIcs*, pages 9:1–9:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[ten Cate *et al.*, 2022] Balder ten Cate, Maurice Funk, Jean Christoph Jung, and Carsten Lutz. On the non-efficient PAC learnability of acyclic conjunctive queries. *CoRR*, abs/2208.10255, 2022.

[ten Cate *et al.*, 2023] Balder ten Cate, Victor Dalmau, Maurice Funk, and Carsten Lutz. Extremal fitting problems for conjunctive queries. In *Proc. of PODS*, 2023.

[Valiant, 1984] Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.

[Westphal *et al.*, 2019] Patrick Westphal, Lorenz Bühmann, Simon Bin, Hajira Jabeen, and Jens Lehmann. SML-bench - A benchmarking framework for structured machine learning. *Semantic Web*, 10(2):231–245, 2019.

[Zarrieß and Turhan, 2013] Benjamin Zarrieß and Anni-Yasmin Turhan. Most specific generalizations w.r.t. general $\mathcal{EL}$-TBoxes. In *Proc. of IJCAI*, pages 1191–1197, 2013.

[Zhou *et al.*, 2013] Yujiao Zhou, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, and Jay Banerjee. Making the most of your triple store: query answering in OWL 2 using an RL reasoner. In *WWW*, pages 1569–1580. ACM, 2013.

## A Additional Preliminaries

We make precise how ELQs can be viewed as databases as announced in Section 2 of the main paper. Formally, we inductively associate to every $q$ a pointed database $(\mathcal{D}_q, a_q)$ with $\mathcal{D}_q$ tree-shaped (recall that a database is tree-shaped if the directed graph $G_\mathcal{D} = (\mathsf{adom}(\mathcal{D}), \{(a,b) \mid r(a,b) \in \mathcal{D}\})$ is a tree), as follows:

- If $q = \top$, then $\mathcal{D}_q$ contains the single fact $\top(a_q)$;[5]
- if $q = A$, then $\mathcal{D}_q$ contains the single fact $A(a_q)$;
- if $q = q_1 \sqcap q_2$, then $\mathcal{D}_q$ is obtained from $(\mathcal{D}_{q_1}, d_{q_1}), (\mathcal{D}_{q_2}, d_{q_2})$ by first taking the disjoint union of $\mathcal{D}_{q_1}$ and $\mathcal{D}_{q_2}$ and then identifying $a_{q_1}$ and $a_{q_2}$ to $a_q$;
- if $q = \exists r.p$, then $\mathcal{D}_q$ is obtained from $(\mathcal{D}_p, a_p)$ by taking $\mathcal{D}_q = \mathcal{D}_p \cup \{r(a_q, a_p)\}$ for a fresh individual $a_q$.

Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be interpretations. The *direct product* of $\mathcal{I}_1$ and $\mathcal{I}_2$, denoted $\mathcal{I}_1 \times \mathcal{I}_2$, is the interpretation with domain $\Delta^{\mathcal{I}_1} \times \Delta^{\mathcal{I}_2}$ and such that for all concept names $A$ and role names $r$:

$$\begin{aligned} A^{\mathcal{I}_1 \times \mathcal{I}_2} &= A^{\mathcal{I}_1} \times A^{\mathcal{I}_2} \\ r^{\mathcal{I}_1 \times \mathcal{I}_2} &= r^{\mathcal{I}_1} \times r^{\mathcal{I}_2}. \end{aligned}$$

## B Occam Algorithms and Proof of Theorem 1

Let $\mathcal{O}$ be an ontology and $\mathcal{Q}$ a class of queries. For a set of pointed databases $S$, we say that $\mathcal{Q}$ *shatters* $S$ w.r.t. $\mathcal{O}$ if for every subset $S' \subseteq S$, there is a $q \in \mathcal{Q}$ such that $S' = \{(\mathcal{D}, a) \in S \mid a \in q(\mathcal{D} \cup \mathcal{O})\}$. The *VC-dimension* of $\mathcal{Q}$ w.r.t. $\mathcal{O}$ is the cardinality of the largest set of pointed databases $S$ that is shattered by $\mathcal{Q}$ w.r.t. $\mathcal{O}$.

Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language and $\mathfrak{A}$ a fitting algorithm for $(\mathcal{L}, \mathcal{Q})$. For an $\mathcal{L}$-ontology $\mathcal{O}$, a finite signature $\Sigma$, and $s, m \geq 1$, we use $\mathcal{H}^\mathfrak{A}(\mathcal{O}, \Sigma, s, m)$ to denote the set of all outputs that $\mathfrak{A}$ makes when started on $\mathcal{O}$ and a collection of $m$ data examples $E$ such that $\mathsf{sig}(E) \subseteq \Sigma$ and $E$ is $q_T$-labeled w.r.t. $\mathcal{O}$ according to some $q_T \in \mathcal{Q}_\Sigma$ with $||q_T|| \leq s$. This is called an *effective hypothesis space of* $\mathfrak{A}$. We say that $\mathfrak{A}$ is an *Occam algorithm* if there exists a polynomial $p$ and a constant $\alpha \in [0,1)$ such that for all $\mathcal{L}$-ontologies $\mathcal{O}$, finite signatures $\Sigma$, and $s, m \geq 1$, the VC-dimension of $\mathcal{H}^\mathfrak{A}(\mathcal{O}, \Sigma, s, m)$ w.r.t. $\mathcal{O}$ is bounded by $p(s, |\Sigma|) \cdot m^\alpha$.

Theorem 3.2.1 of [Blumer *et al.*, 1989] then implies the following.

**Lemma 4.** *If $\mathfrak{A}$ is an Occam algorithm with the VC-dimension of effective hypothesis spaces bounded by $p(s, |\Sigma|) \cdot m^\alpha$, then $\mathfrak{A}$ is a PAC learning algorithm with sample size*

$$m(1/\epsilon, 1/\delta, n) = \max\Big(\frac{4}{\epsilon}\log\frac{2}{\delta}, \Big(\frac{8p(s,|\Sigma|)}{\epsilon}\log\frac{13}{\epsilon}\Big)^{1/(1-\alpha)}\Big).$$

There are certain difference between the setup used in this paper and the setup in [Blumer *et al.*, 1989]. We comment on why we still obtain Lemma 4 from Theorem 3.2.1 of [Blumer *et al.*, 1989]. The aim of [Blumer *et al.*, 1989] is to study the learning of *concept classes* which are defined in a general way as a set $\mathcal{C}$ of *concepts* $C \subseteq X$ where $X$ is a fixed set of *examples*. Consequently, their definition of PAC algorithms refers to concept classes and, in contrast to Definition 1, does neither mention ontologies nor signatures. However, when fixing an $\mathcal{L}$-ontology $\mathcal{O}$ and signature $\Sigma$, we obtain an associated concept class $\mathcal{C}_{\mathcal{O},\Sigma}$ by taking $X$ to be the set of all pointed $\Sigma$-databases and each query $q \in \mathcal{Q}$ as the concept that consists of all pointed $\Sigma$-databases that are positive examples for $q$. Moreover, by simply fixing $\mathcal{O}$ and $\Sigma$, any fitting algorithm $\mathfrak{A}$ for $(\mathcal{L}, \mathcal{Q})$ turns into a learning algorithm for $\mathcal{C}_{\mathcal{O},\Sigma}$ in the sense of [Blumer *et al.*, 1989]. Here, 'fixing' means that we promise to only run $\mathfrak{A}$ on input ontology $\mathcal{O}$ and collections of labeled data examples $E$ such that $\mathsf{sig}(E) \subseteq \Sigma$ and $E$ is $q_T$-labeled w.r.t. $\mathcal{O}$ according to some $q_T \in \mathcal{Q}_\Sigma$. The definition of Occam algorithms in [Blumer *et al.*, 1989] refers to effective hypothesis spaces $\mathcal{H}^\mathfrak{A}(s, m)$ and requires that their VC-dimension is bounded by $p(s) \cdot m^\alpha$ (where $||\mathcal{O}||$ and $|\Sigma|$ are considered constants). If $\mathfrak{A}$ is Occam in our sense, then $\mathfrak{A}$ with $\mathcal{O}$ and $\Sigma$ fixed is Occam in the sense of [Blumer *et al.*, 1989]. Theorem 3.2.1 of that paper then gives that $\mathfrak{A}$ with $\mathcal{O}$ and $\Sigma$ fixed is a PAC learning algorithm for $\mathcal{C}_{\mathcal{O},\Sigma}$ with the bound stated in Lemma 4.

We remark that the precondition of Theorem 3.2.1 in [Blumer *et al.*, 1989] actually demands that the algorithm runs in polynomial time, but an analysis of the proof shows that this assumption is not used. Then, by Definition 1, every fitting algorithm $\mathfrak{A}$ that is a PAC learning algorithm when restricted to $\mathcal{O}$ and $\Sigma$, for *any* $\mathcal{O}$ and $\Sigma$ and with the *same function* $m$ describing the sample size, is a PAC learning algorithm for $(\mathcal{L}, \mathcal{Q})$.

A final small difference is that, in [Blumer *et al.*, 1989], the function $m$ in the definition of PAC algorithms does not depend on the size of the examples. Our version is a standard variation and does not impair the application of Blumer's Theorem 3.2.1: to see this, it suffices to observe that we do not use this parameter in the definition of effective hypothesis spaces and thus our Occam algorithms (with fixed $\mathcal{O}$ and $\Sigma$) are also Occam algorithms in the sense of Blumer. Moreover, every PAC algorithm in the sense of Blumer is a PAC algorithm in our sense. Intuitively, having the example size as a parameter in the function $m$ makes the lower bounds (results on algorithms not being non-sample PAC learners) stronger as it makes it impossible to use examples of excessive size. It is also more generous regarding the upper bounds (developing PAC algorithms), but we do not make use of that generosity.

**Theorem 1.** *Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language. Every bounded fitting algorithm for $(\mathcal{L}, \mathcal{Q})$ is a (sample-efficient) PAC learning algorithm with sample size $O\big(\frac{1}{\epsilon} \cdot \log\big(\frac{1}{\epsilon}\big) \cdot \log\big(\frac{1}{\delta}\big) \cdot \log|\Sigma| \cdot ||q_T||\big)$.*

*Proof.* Let $\mathfrak{B} = \textsc{Bounded-Fitting}_\mathfrak{A}$ be a bounded fitting algorithm for $(\mathcal{L}, \mathcal{Q})$. Let $\mathcal{O}$ be an $\mathcal{L}$-ontology, $\Sigma$ a finite signature, and $s, m \geq 0$. We show that the VC-dimension of $\mathcal{H}^\mathfrak{B}(\mathcal{O}, \Sigma, s, m)$ is at most $O(s \cdot \log|\Sigma|)$.

It is immediate from Definition 2 that when started on $\mathcal{O}$ and a collection of $m$ data examples $E$ such that $\mathsf{sig}(E) \subseteq \Sigma$ and $E$ is $q_T$-labeled w.r.t. $\mathcal{O}$ according to some $q_T \in Q_\Sigma$ with $||q_T|| \leq s$, then $\mathfrak{B}$ returns a fitting $q \in \mathcal{Q}$ for $E$ w.r.t. $\mathcal{O}$ whose size $||q||$ is smallest among all fitting queries. Consequently, $\mathcal{H}^\mathfrak{B}(\mathcal{O}, \Sigma, s, m)$ consists only of queries $q \in \mathcal{Q}$ with $||q|| \leq s$.

[5]We allow facts of the form $\top(a)$ for convenience.

There are at most $(|\Sigma|+c+1)^s$ such queries for some constant[6] $c$ and since $2^{|S|}$ queries are needed to shatter a set $S$, the VC-dimension of $\mathcal{H}^{\mathfrak{B}}(\mathcal{O}, \Sigma, s, m)$ is at most $\log((|\Sigma|+c+1)^s) \in O(s \cdot \log|\Sigma|)$, as desired. It remains to apply Lemma 4. $\square$

We next comment on the fact that, in the $i$-th round of the SPELL system, we try to fit ELQs that have at most $i-1$ existential quantifiers, rather than ELQs of size at most $i$. By using the following lemma and applying the same arguments as in the proof of Theorem 1, we obtain that our SAT-based approach yields a PAC learning algorithm with sample size $O\big(\frac{1}{\epsilon} \cdot \log\big(\frac{1}{\epsilon}\big) \cdot \log\big(\frac{1}{\delta}\big) \cdot |\Sigma| \cdot ||q_T||\big)$.

**Lemma 5.** *Let $\mathcal{O}$ be an $\mathcal{ELH}^r$-ontology, $n \geq 1$, and $\text{ELQ}^{\exists}_{(n)}$ the set of ELQs that have at most $n$ existential restrictions. Then the VC-dimension of $\text{ELQ}^{\exists}_{(n)}$ w.r.t. $\mathcal{O}$ is at most $2(|\Sigma|+1)n$.*

*Proof.* Let $n \geq 1$. We first observe that the number of concepts in $\text{ELQ}^{\exists}_{(n)}$ is bounded from above by $m_n = 4^{(|\Sigma|+1)n}$. To see this, note that the number of rooted, directed, unlabeled trees with $n$ nodes is bounded from above by the $n$-th Catalan number, which in turn is bounded from above by $4^n$ [Dutton and Brigham, 1986]. Each such tree gives rise to an ELQ by assigning a unique role name from $\Sigma$ to each of the at most $n-1$ edges of the tree and a set of concept names from $\Sigma$ to each of the at most $n$ nodes of the tree. This clearly yields the stated bound $m_n$. Then trivially, the VC-dimension of $\text{ELQ}^{\exists}_{(n)}$ w.r.t. the empty ontology is at most $\log m_n$, thus $2(|\Sigma|+1)n$. Making the ontology non-empty may only decrease the VC-dimension as it may make non-equivalent concepts equivalent, but not vice versa. $\square$

It is easy to see that (the proof of) Lemma 5 applies also to other DLs such as $\mathcal{ELI}$ and $\mathcal{ALCI}$.

## C Proof of Theorem 2

We split the three Points in Theorem 2 into three separate theorems.

**Theorem 6.** *Let $\mathfrak{A}$ be a fitting algorithm for ELQs that always produces a most specific fitting, if it exists. Then $\mathfrak{A}$ is not a sample-efficient PAC learning algorithm.*

*Proof.* Assume to the contrary of what we aim to show that there is a sample-efficient PAC learning algorithm that produces a most specific fitting concept, if it exists, with polynomial function $m : \mathbb{R}^2 \times \mathbb{N}^4 \to \mathbb{N}$ as in Definition 1. Choose $\Sigma = \{A, r\}$, $q_T = A$, $\delta = \epsilon = 0.5$, and $n$ even and large enough such that

$$\binom{n}{n/2} > 2m(1/\epsilon, 1/\delta, 0, |\Sigma|, ||q_T||, n(n+1)).$$

We next construct positive examples; negative examples are not used. Let $\mathcal{S}$ denote the set of subsets of $\{1, \dots, n\}$ and let $\mathcal{S}^{\frac{1}{2}}$ be defined likewise, but include only sets of cardinality exactly $n/2$. With every $S \in \mathcal{S}$, we associate the database

$$\mathcal{D}_S = \{r(b_0, b_1), \dots, r(b_{n-1}, b_n)\} \cup \{A(b_i) \mid i \in S\}$$

as well as the pointed database $(\mathcal{D}'_S, a_0)$ that can be obtained by starting with $\{A(a_0)\}$ and then taking, for every $i \in S$, a disjoint copy of $\mathcal{D}_{\{1,\dots,n\} \setminus \{i\}}$ and identifying the root $b_0$ with $a_0$. Note that every $(\mathcal{D}'_S, a_0)$ is a positive example for $q_T$.

A crucial property of $(\mathcal{D}'_S, a_0)$ is that it is a simulation dual of $(\mathcal{D}_S, a_0)$ restricted to structures $(\mathcal{D}_{S'}, a_0)$, meaning the following.[7]

**Claim.** For all $S, S' \in \mathcal{S}$:

$$(\mathcal{D}_S, a_0) \preceq_\Sigma (\mathcal{D}_{S'}, a_0) \text{ iff } (\mathcal{D}_{S'}, a_0) \not\preceq_\Sigma (\mathcal{D}'_S, a_0).$$

The claim is easy to verify. We do not work with unrestricted simulation duals here because we want the databases $(\mathcal{D}'_S, a_0)$ to be acyclic, and unrestricted simulation duals are not.

Let $P$ be the probability distribution that assigns probability $1/|\mathcal{S}^{\frac{1}{2}}|$ to every $(\mathcal{D}'_S, a_0)$ with $S \in \mathcal{S}^{\frac{1}{2}}$, and probability 0 to all other pointed databases.

Now assume that the algorithm is started on a collection of $m(1/\epsilon, 1/\delta, 0, |\Sigma|, ||q_T||, n(n+1))$ labeled data examples $E$. Since all examples are acyclic, the most specific fitting $q_H$ exists [Jung *et al.*, 2020] and is output by the algorithm. It is not important to make explicit at this point the exact details of $q_H$, but it can be thought of as the direct product of all the examples in $E$, viewed as an ELQ.

To obtain a contradiction, it suffices to show that with probability at least $1 - \delta = 0.5$, we have $\text{error}_{P,q_T}(q_H) > \epsilon = 0.5$. We argue that, in fact, $q_H$ violates all data examples $(\mathcal{D}'_S, a_0)$ with $S \in \mathcal{S}^{\frac{1}{2}}$ that are not in the sample $E$. The definition of $P$ and choice of $n$ then yield that with probability 1, $\text{error}_{P,q_T}(q_H) > 0.5$.

Thus take $S \in \mathcal{S}^{\frac{1}{2}}$ with $(\mathcal{D}'_S, a_0) \notin E$. To show that $a_0 \notin q_H(\mathcal{D}'_S)$, it suffices to prove the following:

1. $(\mathcal{D}_S, a_0) \preceq (\mathcal{D}_{q_H}, a_{q_H})$;
   Let $q_S$ be $(\mathcal{D}_S, a_0)$ viewed as an ELQ. We show that $q_S$ is a fitting of $E$. Take any $(\mathcal{D}_{S'}, a_0) \in E$. Then $S \neq S'$ and thus $(\mathcal{D}_{S'}, a_0) \not\preceq (\mathcal{D}_S, a_0)$. The claim yields $(\mathcal{D}_S, a_0) \preceq (\mathcal{D}'_{S'}, a_0)$. Thus $a_0 \in q_S(\mathcal{D}'_{S'})$, and we are done.
   Since $q_H$ is the most specific fitting of $E$, it follows from $q_S$ being a fitting that $q_H \sqsubseteq q_S$, which yields $(\mathcal{D}_S, a_0) \preceq (\mathcal{D}_{q_H}, a_{q_H})$ as desired.
2. $(\mathcal{D}_S, a_0) \not\preceq (\mathcal{D}'_S, a_0)$.
   Follows from the claim and the fact that $(\mathcal{D}_S, a_0) \preceq (\mathcal{D}_S, a_0)$.

Now, $a_0 \notin q_H(\mathcal{D}'_S)$ follows from $(\mathcal{D}_{q_H}, a_{q_H}) \preceq (\mathcal{D}'_S, a_0)$ which is ruled out by Points 1 and 2 above and the fact that the composition of two simulations is again a simulation. $\square$

**Theorem 7.** *Let $\mathfrak{A}$ be a fitting algorithm for ELQs that always produces a most general fitting, if it exists. Then $\mathfrak{A}$ is not a sample-efficient PAC learning algorithm.*

[6]The number of symbols from the finite alphabet used to encode syntactic objects as a word from the definition of $||\cdot||$.

[7]For homomorphisms, the notion of a restricted duality is well-established, see for example [Nesetril and de Mendez, 2012].

*Proof.* We only provide the missing details from the proof in the main part, that is, the proof of Points 1 and 2 stated in the main part and the place of the proof that say "it follows from Point 1 that":

1. $q$ is the most general ELQ that fits $(\mathcal{D}'_q, a'_q, -)$;

   Let $p$ be an ELQ such that $(\mathcal{D}'_q, a'_q)$ is a negative example for $p$. We have to show that $p \sqsubseteq q$.

   $(\mathcal{D}'_q, a'_q)$ being a negative example for $p$ means that $a'_q \notin p(\mathcal{D}'_q)$ and thus $(\mathcal{D}_p, a_p) \not\preceq (\mathcal{D}'_q, a'_q)$ by Lemma 1. The definition of duals thus yields $(\mathcal{D}_q, a_q) \preceq (\mathcal{D}_p, a_p)$ and Lemma 1 gives $p \sqsubseteq q$, as desired.

2. For all $T \subseteq S$, $q \notin T$ implies $p_T \not\sqsubseteq q$ where $p_T = \bigsqcap_{p \in T} p$.

   Consider the database $\mathcal{D}_{p_T}$. Then clearly $a_{p_T} \in p_T(\mathcal{D}_{p_T})$. But since $q \notin T$, $\mathcal{D}_{p_T}$ contains no $r$-path outgoing from $a_{p_T}$ that contains the $A/B$-labeling of $q$, and thus $a_{p_T} \notin q(\mathcal{D}_{p_T})$.

3. It follows from Point 1 that $q_H = \bigsqcap_{(\mathcal{D}'_q, a'_q) \in E} q$ is the most general ELQ that fits $E$.

   Clearly, $q_H \sqsubseteq q$ for every $(\mathcal{D}'_q, a'_q) \in E$, and thus $(\mathcal{D}_q, a_q) \preceq (\mathcal{D}_{q_H}, a_{q_H})$. By Point 1, $(\mathcal{D}_q, a_q) \not\preceq (\mathcal{D}'_q, a'_q)$. Since the composition of two simulations is a simulation, this implies $(\mathcal{D}_{q_H}, a_{q_H}) \not\preceq (\mathcal{D}'_q, a'_q)$. It follows that $q_H$ fits $E$.

   It remains to show that $q_H$ is most general. Assume that some ELQ $p$ fits $E$. Then $a'_q \notin p(\mathcal{D}'_q)$ for all $(\mathcal{D}'_q, a'_q) \in E$ and thus $(\mathcal{D}_p, a_p) \not\preceq (\mathcal{D}'_q, a'_q)$. By definition of duals, $(\mathcal{D}_q, a_q) \preceq (\mathcal{D}_p, a_p)$ and this is witnessed by some simulation $S_q$. But then $\bigcup_q S_q$ is a simulation showing $(\mathcal{D}_{q_H}, a_{q_H}) \preceq (\mathcal{D}_p, a_p)$, and thus $p \sqsubseteq q_H$ as desired.

□

**Theorem 8.** *Let $\mathfrak{A}$ be a fitting algorithm for ELQs that always produces a fitting of minimum quantifier depth. Then $\mathfrak{A}$ is not a sample-efficient PAC learning algorithm.*

*Proof.* Assume to the contrary of what we aim to show that there is a sample-efficient learning algorithm that produces a most shallow fitting concept, if it exists, with associated polynomial function $m : \mathbb{R}^2 \times \mathbb{N}^4 \to \mathbb{N}$ as in Definition 1. We are going to use target queries of the form $q_T = \exists t^{n+1}.\top$.

Choose $\Sigma = \{r, s, t\}$, $\delta = 0.5$, $\epsilon = 0.4$, and $n$ large enough such that

$$\frac{2^n!}{2^{np(n)}(2^n - p(n))!} > 1 - \delta \qquad (*)$$

where $p(n)$ is the polynomial

$$p(n) = m(\frac{1}{\delta}, \frac{1}{\epsilon}, 0, |\Sigma|, n+1, p'(n))$$

and $p'$ is a fixed polynomial that describes the size of the examples that we are going to use. Lemma 6 below shows that such an $n$ always exists, regardless of the precise polynomial $p'$. The meaning of the expression on the left-hand side of $(*)$ will be explained later.

Recall that the target query $q_T = \exists t^{n+1}.\top$ is of quantifier depth $n+1$. We construct (both positive and negative) examples such that with high probability, the drawn examples admit a fitting of quantifier depth $n$ that, however, does not generalize well. Define a set of ELQs

$$S = \{\exists r_1. \dots \exists r_n.\top \mid r_i \in \{r, s\},\ 1 \le i \le n\}.$$

By Theorem 3, each $q \in S$ has a polynomially sized $\Sigma$-dual that consists of a single element $(P_q, a)$. By duality, $(P_q, a)$ is a positive example for $q_T$. Also by Theorem 3, $q_T$ has a polynomially sized $\Sigma$-dual that contains a single element $(\mathcal{D}_T, a)$. For each $q \in S$, we construct a negative example $(N_q, a)$ by taking

$$(N_q, a) = (P_q, a) \times (\mathcal{D}_T, a)$$

where $\times$ denotes the direct product of two pointed databases. Since the direct product is of polynomial size, both the positive examples and the negative examples are of size polynomial in $n$. We let $p'(n)$ be any polynomial that bounds (from above) the size of the examples.

Note that by the properties of duals and products, for all $q \in S$ and for all $\Sigma$-ELQs $q'$, we have

(i) $a \in q'(P_q)$ iff $q' \not\sqsubseteq q$, and

(ii) $a \in q'(N_q)$ iff $q' \not\sqsubseteq q$ and $q' \not\sqsubseteq q_T$.

To see Point (i) note that $a \in q'(P_q)$ iff (by Lemma 1) $(\mathcal{D}_{q'}, a_{q'}) \preceq (P_q, a)$ iff (by duality) $(\mathcal{D}_q, a_q) \not\preceq (\mathcal{D}_{q'}, a_{q'})$ iff (by Lemma 1) $q' \not\sqsubseteq q$. Point (ii) can be shown similar and uses that $(\mathcal{D}, a) \preceq (\mathcal{D}_1, a_1) \times (\mathcal{D}_2, a_2)$ iff $(\mathcal{D}, a) \preceq (\mathcal{D}_1, a_1)$ and $(\mathcal{D}_a) \preceq (\mathcal{D}_2, a_2)$, for all pointed databases $(\mathcal{D}, a), (\mathcal{D}_1, a_1), (\mathcal{D}_2, a_2)$.

Let $P$ be the probability distribution that assigns probability $\frac{1}{2^{n+1}}$ to every $(P_q, a)$ and $(N_q, a)$, and probability 0 to all other pointed databases. Now, assume that the algorithm is started on a collection of $k = m(1/\delta, 1/\epsilon, 0, |\Sigma|, n+1, p'(n))$ pointed databases $E$ labeled according to $q_T$ and outputs a hypothesis $q_H$.

Note that the probability of sampling $\ell$ *different* objects from an $N$-element set is the ratio of those sequences of length $\ell$ that contain pairwise distinct elements in the set of all sequences of length $\ell$, that is,

$$\frac{\prod_{i=0}^{\ell-1}(N - \ell)}{N^\ell} = \frac{N!}{N^\ell \cdot (N - \ell)!}.$$

We apply this observation to $N = 2^n$ and $\ell = k$. By choice of $n$, with probability $> 1 - \delta$ we have that for no $q \in S$, both $(N_q, a) \in E$ and $(P_q, a) \in E$. To derive a contradiction, we show that the error of $q_H$ is strictly larger than $\epsilon$ if this is the case.

Consider the ELQ $q' = \bigsqcap_{(N_p, a, -) \in E} p$. We claim that $q'$ fits $E$. Note that $q \not\sqsubseteq q_T$ for any $q \in S$. Point (ii) then implies $a \notin q(N_q)$ for all $q \in S$ and thus $q'$ fits all negative examples. Together with our assumption that for no $q \in S$, both $(N_q, a) \in E$ and $(P_q, a) \in E$, Point (i) implies that $a \in p(P_q)$ for all $(N_p, a, -) \in E$ and $(P_p, a, +) \in E$.

Since $q'$ is a fitting of depth $n$ and the algorithm finds a fitting of minimal depth, $q_H$ must have depth at most $n$, which implies that $q_H \not\sqsubseteq q_T$.

Consider all $q \in S$. It must be that either $q_H \sqsubseteq q$ or $q_H \not\sqsubseteq q$. In the first case, Point (i) implies $a \notin q_H(P_q)$, hence $q_H$ labels the (positive) example $(P_q, a)$ incorrectly. In the second case, Point (ii) implies $a \in q_H(N_q)$, hence $q_H$ labels the (negative) example $(N_q, a)$ incorrectly. Therefore, $\mathsf{error}_{P,q_T}(q_H) \geq 0.5 > \epsilon$. $\square$

**Lemma 6.** *For every polynomial $p(n)$,*

$$\lim_{n\to\infty}\left(\frac{2^n!}{2^{np(n)}(2^n-p(n))!}\right) = 1.$$

*Proof.* As argued in the proof of Theorem 8, the term inside the limit is a probability, so the limit is at most 1. We start with bounding the expression inside the limit from below.

$$\begin{aligned}\frac{2^n!}{2^{np(n)}(2^n-p(n))!} &= \frac{2^n\cdot(2^n-1)\cdot\cdots\cdot(2^n-p(n)+1)}{(2^n)^{p(n)}}\\ &\geq \frac{(2^n-p(n)+1)^{p(n)}}{(2^n)^{p(n)}}\\ &= \left(1-\frac{p(n)+1}{2^n}\right)^{p(n)}\end{aligned}$$

It clearly suffices to show that the limit of the last expression is 1. In order to do so, we reformulate the expression to avoid the $p(n)$ in the exponent.

$$\begin{aligned}&\lim_{n\to\infty}\left(\left(1-\frac{p(n)+1}{2^n}\right)^{p(n)}\right)\\ =&\lim_{n\to\infty}\left(\exp\left(\ln\left(\left(1-\frac{p(n)+1}{2^n}\right)^{p(n)}\right)\right)\right)\\ =&\exp\left(\lim_{n\to\infty}\left(\ln\left(\left(1-\frac{p(n)+1}{2^n}\right)^{p(n)}\right)\right)\right)\\ =&\exp\left(\lim_{n\to\infty}\left(p(n)\cdot\ln\left(1-\frac{p(n)+1}{2^n}\right)\right)\right).\end{aligned}$$

To determine the limit of a product where one factor $p(n)$ converges to $\infty$ and the other $\ln(\cdot)$ converges to 0, we apply l'Hôpital's rule. Set $f(n) = \ln(1-\frac{p(n)+1}{2^n})$ and $g(n) = 1/p(n)$, so $\lim_{n\to\infty}\frac{f(n)}{g(n)}$ is exactly the limit we want to determine (inside the $\exp(\cdot)$). L'Hôpital's rule says that if $\lim_{n\to\infty}\frac{f'(n)}{g'(n)}$ exists, then $\lim_{n\to\infty}\frac{f'(n)}{g'(n)} = \lim_{n\to\infty}\frac{f(n)}{g(n)}$. The derivations $f'(n)$ and $g'(n)$ of $f(n)$ and $g(n)$ are:

$$\begin{aligned}f'(n) &= \frac{\ln(2)(p(n)+1)-p'(n)}{2^n-p(n)-1}\\ g'(n) &= \frac{-p'(n)}{q(n)}\ \text{for some polynomial } q(n)\end{aligned}$$

It remains to observe that $f'(n)/g'(n)$ is an expression that has an exponential $2^n$ in its numerator and everywhere else only polynomials. Thus, $\lim_{n\to\infty}\frac{f'(n)}{g'(n)} = 0 = \lim_{n\to\infty}\frac{f(n)}{g(n)}$, which yields $\exp(0) = 1$ as desired. $\square$

## D Proof of Theorem 3 and Simulation Duals

Instead of proving Theorem 3, we directly prove the more general version in which databases $\mathcal{D}_q$ for an ELQ $q$ are replaced with DAG-shaped databases. A database $\mathcal{D}$ is *DAG-shaped* if the directed graph $G_\mathcal{D} = (\mathsf{adom}(\mathcal{D}), \{(u,v) \mid r(u,v) \in \mathcal{D}\})$ is a DAG.

Let $\mathcal{D}$ be a database and $a, b \in \mathsf{adom}(\mathcal{D})$. A *path from $a$ to $b$ in $\mathcal{D}$* is a finite sequence $a_1, r_1, a_2, \ldots, r_{k-1}, a_k$ such that $a_1 = a$, $a_k = b$, and $r_i(a_i, a_{i+1}) \in \mathcal{D}$, for $1 \leq i < k$. Note that role assertions may be traveled forwards, but not backwards. The *codepth* of an individual $a$ in a DAG-shaped database $\mathcal{D}$ is the length of the longest path starting in $a$; the codepth of an individual $a$ such that there is no assertion $r(a,b) \in \mathcal{D}$ is defined to be 0.

**Theorem 9.** *Let $\Sigma$ be a finite signature and $(\mathcal{D}, a)$ be a pointed database such that $\mathcal{D}$ is DAG-shaped. Then, we can compute in polynomial time a $\Sigma$-simulation dual $M$ of $(\mathcal{D}, a)$ such that*

- *$||M|| \leq 3\cdot|\Sigma|\cdot||\mathcal{D}||^3$, and*
- *if $\mathcal{D}$ is tree-shaped, then $||M|| \leq 3\cdot|\Sigma|\cdot||\mathcal{D}||^2$.*

*Moreover, if $\mathcal{D}$ contains exactly one $\Sigma$-assertion that mentions $a$, then the computed $M$ is actually a singleton set.*

*Proof.* Let $(\mathcal{D}, a)$ be a pointed database with $\mathcal{D}$ DAG-shaped and $\Sigma$ a finite signature. We construct a $\Sigma$-simulation dual of $(\mathcal{D}, a)$ as follows. First, we define a database $\mathcal{D}^*$ with domain

$$\begin{aligned}\mathsf{adom}(\mathcal{D}^*) = \{b^\top\} \cup{}&\\ &\{\langle b, A(b)\rangle \mid A(b) \in \mathcal{D}, A \in \Sigma\} \cup\\ &\{\langle b, r(b,c)\rangle \mid r(b,c) \in \mathcal{D}, r \in \Sigma\}\end{aligned}$$

and include the following assertions, for all $\langle b, A(b)\rangle \in \mathsf{adom}(\mathcal{D}^*)$ and $\langle b, r(b,c)\rangle \in \mathsf{adom}(\mathcal{D}^*)$:

(i) $B(b^\top)$ for all $B \in \Sigma \cap \mathsf{N_C}$;
(ii) $s(b^\top, b^\top)$ for all $s \in \Sigma \cap \mathsf{N_R}$;
(iii) $B(\langle b, A(b)\rangle)$ for all $B \in \Sigma \cap \mathsf{N_C}$ with $B \neq A$;
(iv) $s(\langle b, A(b)\rangle, b^\top)$ for all $s \in \Sigma \cap \mathsf{N_R}$;
(v) $B(\langle b, r(b,c)\rangle)$ for all $B \in \Sigma \cap \mathsf{N_C}$;
(vi) $s(\langle b, r(b,c)\rangle, b^\top)$ for all $s \in \Sigma \cap \mathsf{N_R}$ with $s \neq r$;
(vii) $r(\langle b, r(b,c)\rangle, \langle c, \alpha\rangle)$ for all $\langle c, \alpha\rangle \in \mathsf{adom}(\mathcal{D}^*)$.

We prove two auxiliary claims.

*Claim 1.* For all $b \in \mathsf{adom}(\mathcal{D})$ and $\langle b, \alpha\rangle \in \mathsf{adom}(\mathcal{D}^*)$, $(\mathcal{D}, b) \not\preceq_\Sigma (\mathcal{D}^*, \langle b, \alpha\rangle)$.

*Proof of Claim 1.* We prove the claim by induction on the codepth of $b$ in $\mathcal{D}$. If $b$ has codepth 0, then $\alpha$ is of the form $A(b)$, for $A(b) \in \mathcal{D}$. By Point (iii) in the definition of $\mathcal{D}^*$, $A(\langle b, A(b)\rangle) \notin \mathcal{D}^*$, and thus $(\mathcal{D}, b) \not\preceq_\Sigma (\mathcal{D}^*, \langle b, \alpha\rangle)$.

Now, let $b$ have codepth greater than 0. We distinguish cases on the shape of $\alpha$.

- If $\alpha$ is of the form $A(b)$ for some $A(b) \in \mathcal{D}$, then we can argue as in the base case that $(\mathcal{D}, b) \not\preceq_\Sigma (\mathcal{D}^*, \langle b, \alpha\rangle)$.
- If $\alpha$ is of the form $r(b,c)$ for some $r(b,c) \in \mathcal{D}$, assume for contradiction that there is a $\Sigma$-simulation $S$ from $\mathcal{D}$ to $\mathcal{D}^*$ with $(b, \langle b, r(b,c)\rangle \in S)$. Since $S$ is a simulation and $c$ is an $r$-successor of $b$ in $\mathcal{D}$, there has to be

an $r$-successor $c'$ of $\langle b, r(b,c)\rangle$ in $\mathcal{D}^*$ with $(c,c') \in S$. By Point (vi) and (vii), $c'$ is of shape $\langle c, \alpha\rangle$. But then $(\mathcal{D}, c) \preceq_\Sigma (\mathcal{D}', \langle c, \alpha\rangle)$, contradicting the induction hypothesis.

*Claim 2.* For all $b \in \mathsf{adom}(\mathcal{D})$ and pointed databases $(\mathcal{D}', c)$, if $(\mathcal{D}, b) \not\preceq_\Sigma (\mathcal{D}', c)$ then there is a $\langle b, \alpha\rangle \in \mathsf{adom}(\mathcal{D}^*)$ such that $(\mathcal{D}', c) \preceq_\Sigma (\mathcal{D}^*, \langle b, \alpha\rangle)$.

*Proof of Claim 2.* We prove the claim by induction on the codepth of $b$ in $\mathcal{D}$. If $b$ has codepth 0 and $(\mathcal{D}, b) \not\preceq_\Sigma (\mathcal{D}', c)$, then there is a concept name $A \in \Sigma$ such that $A(b) \in \mathcal{D}$ and $A(c) \notin \mathcal{D}'$. It can be verified using Points (i)–(iii) above that the relation

$$S = \{(c, \langle b, A(b)\rangle)\} \cup \{(c', b^\top) \mid c' \in \mathsf{adom}(\mathcal{D}')\}$$

is a $\Sigma$-simulation from $\mathcal{D}'$ to $\mathcal{D}^*$ with $(c, \langle b, A(b)\rangle) \in S$ as required.

Now, let $b$ have codepth greater than 0 and assume $(\mathcal{D}, b) \not\preceq_\Sigma (\mathcal{D}', c)$. We distinguish cases on why the latter is the case:

- If there is a concept name $A \in \Sigma$ such that $A(b) \in \mathcal{D}$ and $A(c) \notin \mathcal{D}'$, we can argue as in the base case that $(\mathcal{D}', c) \preceq_\Sigma (\mathcal{D}^*, \langle b, A(b)\rangle)$.
- If there is an assertion $r(b,b') \in \mathcal{D}$ such that for all $r(c,c') \in \mathcal{D}'$, $(\mathcal{D}, b') \not\preceq_\Sigma (\mathcal{D}', c')$. We show that $(\mathcal{D}', c) \preceq_\Sigma (\mathcal{D}^*, \langle b, r(b,b')\rangle)$.
  The induction hypothesis implies that for all $r(c,c') \in \mathcal{D}'$ there is an $\langle b', \beta\rangle \in \mathsf{adom}(\mathcal{D}^*)$ and a simulation $S_{c'}$ from $\mathcal{D}'$ to $\mathcal{D}^*$ with $(c', \langle b', \beta\rangle) \in S_{c'}$. It can be verified using Points (v)-(vii) above that
  $$S = \{(b, \langle b, r(b,b')\rangle)\} \cup \{(c', b^\top) \mid c' \in \mathsf{adom}(\mathcal{D}')\} \cup \bigcup_{r(c,c') \in \mathcal{D}'} S_{c'}$$
  is a simulation from $\mathcal{D}'$ to $\mathcal{D}^*$ with $(b, \langle b, r(b,b')\rangle) \in S$.

This completes the proofs of Claims 1 and 2. The next claim shows how to read off a simulation dual of $(\mathcal{D}, a)$ from $\mathcal{D}^*$.

*Claim 3.* The set

$$M_a = \{(\mathcal{D}^*, \langle a, \alpha\rangle) \mid \langle a, \alpha\rangle \in \mathsf{adom}(\mathcal{D}^*)\}$$

is a $\Sigma$-simulation dual of $(\mathcal{D}, a)$.

*Proof of Claim 3.* Suppose $(\mathcal{D}, a) \not\preceq_\Sigma (\mathcal{D}', a')$ for some $(\mathcal{D}', a')$. Then Claim 2 implies that there is some $\langle a, \alpha\rangle \in \mathsf{adom}(\mathcal{D}^*)$ with $(\mathcal{D}', a') \preceq (\mathcal{D}^*, \langle a, \alpha\rangle)$. It remains to note that $(\mathcal{D}^*, \langle a, \alpha\rangle) \in M_a$. Conversely, suppose that $(\mathcal{D}, a) \preceq_\Sigma (\mathcal{D}', a')$ and assume for showing a contradiction that $(\mathcal{D}', a') \preceq_\Sigma (\mathcal{D}^*, \langle a, \alpha\rangle)$ for some $\langle a, \alpha\rangle \in \mathsf{adom}(\mathcal{D}^*)$. Since $\preceq_\Sigma$ is transitive, we obtain $(\mathcal{D}, a) \preceq_\Sigma (\mathcal{D}^*, \langle a, \alpha\rangle)$, in contradiction to Claim 1. This finishes the proof of Claim 3.

Clearly, $M_a$ is a singleton set if $\mathcal{D}$ contains only a single $\Sigma$-assertion mentioning $a$. It remains to analyze $||M_a||$. We start with analyzing $||\mathcal{D}^*||$. Points (i) and (ii) together contribute $|\Sigma|$ assertions. Points (iii) and (iv) contribute together $|\Sigma| \cdot n_C$ assertions where $n_C$ denotes the number of assertions of shape $A(b)$ in $\mathcal{D}$. Points (v) and (vi) contribute $|\Sigma| \cdot n_R$ assertions where $n_R$ denotes the number of assertions of shape $r(b,c)$ in $\mathcal{D}$. Finally, Point (vii) contributes $|\mathcal{D}|^2$ assertions. Overall, we obtain

$$||\mathcal{D}^*|| \leq |\Sigma| + |\Sigma| \cdot n_C + |\Sigma| \cdot n_R + |\mathcal{D}|^2 \leq 3 \cdot |\Sigma| \cdot |\mathcal{D}|^2.$$

Thus, $||M_a|| \leq |\mathcal{D}| \cdot 3 \cdot |\Sigma| \cdot |\mathcal{D}|^2 \leq 3 \cdot |\Sigma| \cdot ||\mathcal{D}||^3$ as required.

If $\mathcal{D}$ is tree-shaped, then the bound on the number of assertions that Point (vii) contributes can be improved. Only a single incoming assertion is added for each $\langle c, \alpha\rangle$, resulting in $|\mathcal{D}|$ assertions. This improves the overall bounds to

$$||\mathcal{D}^*|| \leq |\Sigma| + |\Sigma| \cdot n_C + |\Sigma| \cdot n_R + |\mathcal{D}| \leq 3 \cdot |\Sigma| \cdot |\mathcal{D}|.$$

Thus, $||M_a|| \leq |\mathcal{D}| \cdot 3 \cdot |\Sigma| \cdot |\mathcal{D}| \leq 3 \cdot |\Sigma| \cdot ||\mathcal{D}||^2$ as required. □

We next characterize the pointed databases that admit finite simulation duals. We need some additional notation.

We say that $b$ is *reachable* from $a$ if there is a path from $a$ to $b$. We use $\mathcal{D}^{\downarrow a}$ to denote the database that consists of all facts $A(b), r(b,b') \in \mathcal{D}$ with $b, b'$ reachable from $a$. We note that, for some $a \in \mathsf{adom}(\mathcal{D})$, $\mathcal{D}^{\downarrow a}$ might be empty (namely, if there are no assertions of the form $A(a), r(a,b) \in \mathcal{D}$). In a slight abuse of notation, we then allow to write $(\emptyset, a)$ and mean the pointed database $(\{\top(a)\}, a)$. We use $\mathcal{D}_\Sigma$ to denote the restriction of a database $\mathcal{D}$ to its $\Sigma$-assertions, for any signature $\Sigma$. Hence, $\mathcal{D}_\Sigma^{\downarrow a}$ is the restriction of $\mathcal{D}^{\downarrow a}$ to $\Sigma$.

The proof of the characterization relies on the (standard) notion of unravelings. Let $\mathcal{D}$ be a database and $a \in \mathsf{adom}(\mathcal{D})$. The *unraveling of $\mathcal{D}$ at $a$* is the (possibly infinite) database $\mathcal{U}$ whose domain $\mathsf{adom}(\mathcal{U})$ consists of all paths starting in $a$ and that contains the following assertions for every $p = a_1, r_1, a_2, \ldots, r_{k-1}, a_k \in \mathsf{adom}(\mathcal{U})$:

- $A(p)$ for all $A(a_k) \in \mathcal{D}$ and
- $r_{k-1}(p', p)$ for $p' = a_1, r_1, \ldots, a_{k-1}$.

**Theorem 10.** *Let $\Sigma$ be a finite signature and $(\mathcal{D}, a)$ a pointed database. Then, $(\mathcal{D}, a)$ has a finite $\Sigma$-simulation dual iff $\mathcal{D}_\Sigma^{\downarrow a}$ is DAG-shaped.*

*Proof.* For the "if"-direction, suppose that $\mathcal{D}_\Sigma^{\downarrow a}$ is DAG-shaped. It should be clear that both $(\mathcal{D}, a) \preceq_\Sigma (\mathcal{D}_\Sigma^{\downarrow a}, a)$ and $(\mathcal{D}_\Sigma^{\downarrow a}, a) \preceq_\Sigma (\mathcal{D}, a)$, and thus $(\mathcal{D}, a)$ and $(\mathcal{D}_\Sigma^{\downarrow a}, a)$ have the same $\Sigma$-simulation duals. Theorem 9 implies the existence of a finite $\Sigma$-simulation for $(\mathcal{D}_\Sigma^{\downarrow a}, a)$ and thus of $(\mathcal{D}, a)$.

For "only if", we assume that $\mathcal{D}_\Sigma^{\downarrow a}$ is not DAG-shaped and show that there cannot be a finite $\Sigma$-simulation dual. Assume to the contrary of what is to be shown that $M$ is a finite $\Sigma$-simulation dual of $(\mathcal{D}, a)$. Let $\mathcal{U}$ be the unraveling of $\mathcal{D}_\Sigma^{\downarrow a}$ at $a$. Note that $\mathcal{U}$ is an infinite (and tree-shaped) database as $\mathcal{D}_\Sigma^{\downarrow a}$ is not DAG-shaped. Let, moreover, $\mathcal{U}_i$ denote the restriction of $\mathcal{U}$ to individuals that have distance at most $i$ from the root $a$, for $i \geq 0$. Clearly, we have:

(i) $(\mathcal{D}, a) \preceq_\Sigma (\mathcal{U}, a)$, and

(ii) $(\mathcal{D}, a) \not\preceq_\Sigma (\mathcal{U}_i, a)$, for all $i \geq 0$.

By duality and Point (ii), for every $i \geq 0$ there exists some $(\mathcal{D}', a') \in M$ with $(\mathcal{U}_i, a) \preceq_\Sigma (\mathcal{D}', a')$. Since $M$ is finite, there is some $(\mathcal{D}^*, a^*) \in M$ such that $(\mathcal{U}_i, a) \preceq_\Sigma (\mathcal{D}^*, a^*)$

for infinitely many $i \geq 0$. Using a standard "simulation skipping" argument, we can inductively construct a simulation $S$ witnessing $(\mathcal{U}, a) \preceq_\Sigma (\mathcal{D}^*, a^*)$. By duality, we obtain $(\mathcal{D}, a) \not\preceq_\Sigma (\mathcal{U}, a)$, which is in contradiction to Point (i) above.

Let us now give some details regarding the simulation skipping argument. Let $I$ be an infinite set such that $(\mathcal{U}_i, a) \preceq_\Sigma (\mathcal{D}^*, a^*)$ for all $i \in I$, and let $(S_i)_{i\in I}$ be a family of $\Sigma$-simulations witnessing that. We provide an infinite family $(S_i^*)_{i\geq 0}$ of relations $\mathsf{adom}(\mathcal{U}_i) \times \mathsf{adom}(\mathcal{D}^*)$ such that, for every $i \geq 0$:

(a) $S_i^*$ is a $\Sigma$-simulation witnessing $(\mathcal{U}_i, a) \preceq_\Sigma (\mathcal{D}^*, a^*)$, and

(b) $S_i^* \subseteq S_j$, for infinitely many $j \in I$.

We start with setting $S_0^* = \{(a, a)\}$ which clearly satisfies Points (a) and (b). To obtain $S_{i+1}^*$ from $S_i^*$, let $B = \mathsf{adom}(\mathcal{U}_{i+1}) \setminus \mathsf{adom}(\mathcal{U}_i)$. Note that $B$ is finite. By Point (b) applied to $S_i^*$, there is an infinite set $J \subseteq I$ such that $S_i^* \subseteq S_j$ for every $j \in J$. Since both $B$ and $\mathcal{D}^*$ are finite, we can pick an infinite subset $J' \subseteq J$ such that for every $b \in B$, every $d \in \mathsf{adom}(\mathcal{D}^*)$, and every $j, j' \in J'$, we have

$$(b, d) \in S_j \quad \text{iff} \quad (b, d) \in S_{j'}.$$

Obtain $S_{i+1}^*$ from $S_i^*$ by adding $(b, d) \in B \times \mathsf{adom}(\mathcal{D}^*)$ in case $(b, d) \in S_j$ for all $j \in J'$. It is routine to verify that $S_{i+1}^*$ satisfies Points (a) and (b), and that

$$S = \bigcup_{i\geq 0} S_i^*$$

witnesses $(\mathcal{U}, a) \preceq_\Sigma (\mathcal{D}^*, a^*)$, as desired. □

## E Proof of Theorem 4

Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language, $\mathcal{O}$ an $\mathcal{L}$-ontology, and $\Sigma$ a finite signature. A *(downward) frontier* for a query $q \in Q$ with respect to $\mathcal{O}$ and $\Sigma$ is a finite set $F \subseteq \mathcal{Q}$ such that

1. each $p \in F$ is a downward refinement of $q$ w.r.t. $\mathcal{O}$ and
2. for each $p \in \mathcal{Q}_\Sigma$ that is a downward refinement of $q$ w.r.t. $\mathcal{O}$, there is some $p' \in F$ such that $\mathcal{O} \models p \sqsubseteq p'$.

Note that for both refinement operators $\rho_1$ and $\rho_2$ defined in the main part of the paper and any ELQ $q$ and signature $\Sigma$, $\rho_i(q, \Sigma)$ is a downward frontier for $q$ with respect to the empty ontology and $\Sigma$.

The following clearly implies Theorem 4.

**Theorem 11.** *Let $(\mathcal{L}, \mathcal{Q})$ be an OMQ language. The following are equivalent:*

1. *$(\mathcal{L}, \mathcal{Q})$ has an ideal downward refinement operator,*
2. *$(\mathcal{L}, \mathcal{Q})$ has an ideal downward refinement operator that is $2^{O(n)}$-depth bounded,*
3. *for all $\mathcal{L}$-ontologies $\mathcal{O}$ and all finite signatures $\Sigma$, each $q \in \mathcal{Q}$ has a downward frontier w.r.t. $\mathcal{O}$ and $\Sigma$.*

*Proof. From 1 to 3.* Let $\rho$ be a downward refinement operator. We claim that, for each $\mathcal{L}$-ontology $\mathcal{O}$ and $q \in \mathcal{Q}$ and finite signature $\Sigma$, $\rho(q, \mathcal{O}, \Sigma)$ is a downward frontier of $q$ w.r.t. $\mathcal{O}$ and $\Sigma$. For Point 1 in the definition of downward frontier, note that any $p \in \rho(q, \mathcal{O}, \Sigma)$ is a downward refinement of $q$ w.r.t. $\mathcal{O}$. For Point 2, let $p \in \mathcal{Q}_\Sigma$ be any downward refinement of $q$ w.r.t. $\mathcal{O}$. By completeness, there is a finite sequence $q_1, \ldots, q_n$ with $q_1 = q$, $q_n = p$, and $q_{i+1} = \rho(q_i, \mathcal{O}, \Sigma)$ for all $i$. Note that, necessarily, $n \geq 2$. It follows that $q_2 \in \rho(q, \mathcal{O}, \Sigma)$ and $\mathcal{O} \models p \sqsubseteq q_2$.

*From 3 to 2.* Take any $\mathcal{L}$-ontology $\mathcal{O}$ and finite signature $\Sigma$. For each $q \in \mathcal{Q}$, let $F(q, \mathcal{O}, \Sigma)$ be a downward frontier for $q$ w.r.t. $\mathcal{O}$ and $\Sigma$. Let $\rho(q, \mathcal{O}, \Sigma)$ be the union of $F(q, \mathcal{O}, \Sigma)$ with the set of all downwards refinements $q' \in Q_\Sigma$ of $q$ with $||q'|| \leq ||q||$. Clearly, $\rho$ is a finite downward refinement operator. To show that $\rho$ is complete, consider any pair of queries $(q, p)$ from $\mathcal{Q}$ such that $\mathcal{O} \models p \sqsubseteq q$. Suppose for the sake of a contradiction that there is no downward $\rho, \mathcal{O}, \Sigma$-refinement sequence starting in $q$ and ending in a query $p'$ with $\mathcal{O} \models p \equiv p'$. It then follows from the properties of downward frontiers that there exists an infinite sequence of (pairwise non-equivalent) queries

$$q_1, q_2, \ldots$$

with $q_1 = q$ and $q_{i+1} \in F(q_i, \mathcal{O}, \Sigma)$ for all $i \geq 0$, such that $p$ is a downward refinement of each $q_i$ w.r.t. $\mathcal{O}$. Let $k > 0$ be minimal with $||q_k|| \geq ||p||$. Clearly, $k = 2^{O(||p||)}$. Moreover, $q_1, \ldots, q_k, p$ is a $\rho, \mathcal{O}, \Sigma$-refinement sequence starting in $q$ and ending in $p$, a contradiction. Hence, $\rho$ is an ideal downward refinement operator. Furthermore, $\rho$ is $2^{O(n)}$-depth bounded.

The implication from 2 to 1 is trivial. □

## F Proof of Lemma 2 and Theorem 5

Before proving Lemma 2 and Theorem 5, we recall some important properties of minimal ELQs. Recall that an ELQ $q$ is *minimal* if there is no ELQ $p$ with $p \equiv q$ and $||p|| < ||q||$. Due to the correspondence of ELQs and $\mathcal{EL}$-concepts, we may speak of minimal $\mathcal{EL}$-concepts. Minimal ELQs (and thus, minimal $\mathcal{EL}$-concepts) can be characterized in terms of functional simulations, where a simulation $S$ between $\mathcal{D}_1$ and $\mathcal{D}_2$ is called *functional* if for every $d \in \mathsf{adom}(\mathcal{D}_1)$, there is at most one $e \in \mathsf{adom}(\mathcal{D}_2)$ with $(d, e) \in S$.

**Lemma 7.** *An ELQ $q$ is minimal iff the only functional simulation $S$ from $\mathcal{D}_q$ to $\mathcal{D}_q$ with $(a_q, a_q) \in S$ is the identity.*

The following lemma shows several ways how to refine a minimal $\mathcal{EL}$-concept. Its proof is straightforward using simulations and Lemmas 1 and 7, details are left to the reader.

**Lemma 8.** *Let $C$ be a minimal $\mathcal{EL}$-concept and $D = A_1 \sqcap \cdots \sqcap A_k \sqcap \exists r_1.D_1 \sqcap \cdots \sqcap \exists r_\ell.D_\ell$ a subconcept of $C$. Then the following hold:*

1. *For all $C'$ that can be obtained from $C$ by replacing $D$ with $D \sqcap A$ for some concept name $A \notin \{A_1, \ldots, A_k\}$, we have $C' \sqsubseteq C$ and $C \not\sqsubseteq C'$.*
2. *For all $C'$ that can be obtained from $C$ by replacing $D$ with $D \sqcap \exists r.\top$ for some role name $r \notin \{r_1, \ldots, r_\ell\}$, we have $C' \sqsubseteq C$ and $C \not\sqsubseteq C'$.*
3. *For all $C'$ that can be obtained from $C$ by replacing a $D$ with $D \sqcap \exists r.\widehat{D}$ for some role name $r \in \{r_1, \ldots, r_\ell\}$ and a concept $\widehat{D}$ such that $D_j \not\sqsubseteq \widehat{D}$ for all $j$ with $r = r_j$, we have $C' \sqsubseteq C$ and $C \not\sqsubseteq C'$.*

The following is a slight strengthening of [Jung *et al.*, 2020, Lemma 6 in the full paper]. Recall that $\mathcal{D}^{\downarrow a}$ denotes the set of all assertions $A(b), r(b, b')$ in $\mathcal{D}$ such that $b, b'$ are reachable from $a$, c.f. Section D.

**Lemma 9.** *Let* $(\mathcal{D}_1, a_1)$ *and* $(\mathcal{D}_2, a_2)$ *be pointed databases with* $\mathcal{D}_2$ *tree-shaped. If* $(\mathcal{D}_1, a_1) \not\preceq (\mathcal{D}_2, a_2)$*, then there exists a set* $\mathcal{D} \subseteq \mathcal{D}_1$ *with* $a_1 \in \mathsf{adom}(\mathcal{D}_1)$ *such that* $|\mathcal{D}| \leq |\mathcal{D}_2^{\downarrow a_2}|+1$ *and* $(\mathcal{D}, a_1) \not\preceq (\mathcal{D}_2, a_2)$.

*Proof.* The proof is by induction on the depth of $\mathcal{D}_2^{\downarrow a_2}$.

Assume first that $\mathcal{D}_2^{\downarrow a_2}$ has depth 0. If there exists a concept name $A \in \mathsf{N_C}$ with $A(a_1) \in \mathcal{D}_1$ but $A(a_2) \notin \mathcal{D}_2$, then $\mathcal{D} = \{A(a_1)\}$ is as required. Otherwise there exists a role name $r \in \mathsf{N_R}$ and $a'$ with $r(a_1, a') \in \mathcal{D}_1$. Then, $\mathcal{D} = \{r(a_1, a')\}$ is as required.

Now, suppose that $\mathcal{D}_2^{\downarrow a_2}$ has depth $k > 0$ and assume $(\mathcal{D}_1, a_1) \not\preceq (\mathcal{D}_2, a_2)$. If there exists a concept name $A \in \mathsf{N_C}$ with $A(a_1) \in \mathcal{D}_1$ but $A(a_2) \notin \mathcal{D}_2$, then $\mathcal{D} = \{A(a_1)\}$ is as required. Otherwise there exists a role name $r \in \mathsf{N_R}$ and some $r(a_1, a') \in \mathcal{D}_1$ such that for all $b$ with $(a_2, b) \in \mathcal{D}_2$, we have $(\mathcal{D}_1, a') \not\preceq (\mathcal{D}_2, b)$. Fix $a'$. By induction hypothesis, we can fix for every $b$ with $r(a_2, b) \in \mathcal{D}_2$, a subset $\mathcal{D}_b \subseteq \mathcal{D}_1$ with $a' \in \mathsf{adom}(\mathcal{D}_b)$ such that $|\mathcal{D}_b| \leq |\mathcal{D}_2^{\downarrow b}| + 1$ and $(\mathcal{D}_b, a') \not\preceq (\mathcal{D}_2, b)$. Let $\mathcal{D}$ be the union of $\{r(a, a')\}$ and all $\mathcal{D}_b$ with $r(a_2, b) \in \mathcal{D}_2$. Then $\mathcal{D}$ is as required. □

**Lemma 2.** *For every ELQ* $q$ *and minimal downward neighbor* $p$ *of* $q$*, we have* $||p|| \leq 2||q|| + 1$.

*Proof.* Let $p, q$ be ELQs such that $p$ is a minimal downward neighbor of $q$, that is, $p \sqsubseteq q$, $q \not\sqsubseteq p$, and for all $p'$ with $p \sqsubseteq p' \sqsubseteq q$, we have $p' \equiv p$ or $q \equiv p'$. Since $||q|| \geq ||q'||$ for every minimal ELQ $q'$ with $q' \equiv q$, we may assume that also $q$ is minimal.

Let $(D_p, a_p)$ and $(D_q, a_q)$ be the pointed databases associated with $p$ and $q$, respectively. By Lemma 1, there is a simulation $S$ from $\mathcal{D}_q$ to $\mathcal{D}_p$ with $(a_q, a_p) \in S$. We can w.l.o.g. assume $S$ to be functional. Clearly, the inverse $S^-$ of $S$ is not a simulation from $\mathcal{D}_p$ to $\mathcal{D}_q$ since $q \not\sqsubseteq p$. We distinguish two cases.

*Case 1.* There is $(a, a') \in S$ such that $A(a') \in \mathcal{D}_p$, but $A(a) \notin \mathcal{D}_q$. Obtain $\mathcal{D}_{q'}$ from $\mathcal{D}_q$ by adding $A(a)$, and let $q'$ be the corresponding ELQ. Clearly, $S$ is a simulation from $\mathcal{D}_{q'}$ to $\mathcal{D}_p$, hence $p \sqsubseteq q'$. Moreover, by construction and Point 1 of Lemma 8, we have $q' \sqsubseteq q$ and $q \not\sqsubseteq q'$. Since $p$ is a downward neighbor of $q$ and $p \sqsubseteq q' \sqsubseteq q$, we thus have $p \equiv q'$. Since $q'$ is obtained from $q$ by adding a single atom and $||p|| \leq ||q'||$, we obtain $||p|| \leq 2||q|| + 1$ as required.

*Case 2.* Case 1 does not apply and there is $(a, a') \in S$ and an assertion $r(a', b') \in \mathcal{D}_p$ such that there is no $b$ with $(b, b') \in S$. Choose such an $(a, a')$ such that $a'$ has maximal distance from the root $a_p$. We distinguish two subcases.

(a) If $a$ does not have an $r$-successor in $\mathcal{D}_q$, obtain $\mathcal{D}_{q'}$ from $\mathcal{D}_q$ by adding an atom $r(a, b)$, for some fresh $b$.

Clearly, $S' = S \cup \{(b, b')\}$ is a functional simulation from $\mathcal{D}_{q'}$ to $\mathcal{D}_p$ with $(a_q, a_p) \in S'$, hence $p \sqsubseteq q'$. Moreover, by construction and Point 2 of Lemma 8, $q' \sqsubseteq q$ and $q \not\sqsubseteq q'$. Since $p$ is a downward neighbor of $q$, we have $q' \equiv p$. Since $p$ is obtained from $q$ by adding a single atom and $||p|| \leq ||q'||$, we obtain $||p|| \leq 2||q|| + 1$ as required.

(b) Otherwise, $a$ has $r$-successors $a_1, \ldots, a_k$ in $\mathcal{D}_q$. Let $b_1, \ldots, b_k$ be the (uniquely defined) elements with $(a_i, b_i) \in S$ for every $i$. In particular, $b' \notin \{b_1, \ldots, b_k\}$. Note that $(\mathcal{D}_p, b') \not\preceq (\mathcal{D}_q, a_i)$ for every $i \in \{1, \ldots, k\}$: otherwise, we would have $(\mathcal{D}_p, b') \preceq (\mathcal{D}_p, b_i)$ for some $i$ in contradiction to minimality of $p$.

Let $\widehat{\mathcal{D}}$ be a minimal subset of $\mathcal{D}_p$ such that $b' \in \mathsf{adom}(\widehat{\mathcal{D}})$ and $(\widehat{\mathcal{D}}, b') \not\preceq (\mathcal{D}_q, a_i)$ for all $i$. By Lemma 9, $|\widehat{\mathcal{D}}| \leq (n_1 + 1) + \cdots + (n_k + 1)$ where $n_i$ is the number of assertions in the tree rooted at $a_i$. It follows that $|\widehat{\mathcal{D}}| \leq |\mathcal{D}_q|$.

Now, obtain $\mathcal{D}_{q'}$ from $\mathcal{D}_q$ by adding $\widehat{\mathcal{D}}$ (assuming that the individuals in $\mathcal{D}_q$ and $\widehat{\mathcal{D}}$ are disjoint) as well as the assertion $r(a, b')$. Note that $||q'|| = |\mathcal{D}_{q'}| \leq |\mathcal{D}_q| + |\widehat{\mathcal{D}}| + 1 \leq 2|\mathcal{D}_q| + 1 = 2||q|| + 1$.

Clearly, $S$ can be extended to a functional simulation from $\mathcal{D}_{q'}$ to $\mathcal{D}_p$, hence $p \sqsubseteq q'$. Moreover, by construction and Point 3 of Lemma 8, $q' \sqsubseteq q$ and $q \not\sqsubseteq q'$. Since $p$ is a downward neighbor of $q$, this implies $q' \equiv p$. Hence, $||p|| \leq ||q'|| \leq 2||q|| + 1$ as required.

This finishes the proof of the lemma. □

Let $\Sigma$ be a finite signature and $p, q \in \text{ELQ}_\Sigma$. A $\Sigma$-*specialization sequence from* $p$ *to* $q$ is a sequence $q_1, \ldots, q_k$ of queries from $\text{ELQ}_\Sigma$ such that $q_1 = p$, $q_k = q$, and $q_{i+1}$ is a neighbor of $q_i$, that is, $q_{i+1} \in \rho_2(q_i, \Sigma)$, for $1 \leq i < k$.[8] We recall two useful properties of the $\mathcal{EL}$-subsumption lattice [Kriegel, 2019, Corollary 5.2.3], namely that for all ELQs $p, q \in \text{ELQ}_\Sigma$ with $p \sqsubseteq q$,

(I) there is a (finite) $\Sigma$-specialization sequence from $q$ to $p$ and

(II) all $\Sigma$-specialization sequences from $q$ to $p$ have the same length (*Jordan-Dedekind chain condition*).

**Lemma 3.** $\rho_1$ *and* $\rho_2$ *are ideal refinement operators for ELQ.*

*Proof.* Both $\rho_1$ and $\rho_2$ are finite by definition. It follows from Property (I) above and the definition of $\rho_2$, and Lemma 2 that $\rho_2$ is complete. The same is then true for $\rho_1$ as it contains $\rho_2$ in the sense that $\rho_1(q, \Sigma) \subseteq \rho_2(q, \Sigma)$ for every ELQ $q$ and finite signature $\Sigma$. □

**Theorem 5.** $\mathfrak{A}_1$ *is a sample-efficient PAC learning algorithm, but* $\mathfrak{A}_2$ *is not.*

*Proof.* We start with analyzing $\mathfrak{A}_2$. Let $E$ be the input to $\mathfrak{A}_2$ and let $\Sigma = \mathsf{sig}(E)$. By Theorem 2, it suffices to show that $\mathfrak{A}_2$ produces a most general fitting of $E$ (if it exists). By the Jordan-Dedekind chain condition, we can assign a *level* to every ELQ $q$ defined as the length of $\Sigma$-specialization sequences from $\top$ to $q$. Let $M_1, M_2, \ldots$ be the sequence of

[8]Note the difference to refinement sequences where refinements are used in place of neighbors.

sets $M$ constructed by the breadth-first search algorithm $\mathfrak{A}_2$, that is, $M_1 = \{\top\}$ and $M_{i+1}$ is obtained from $M_i$ by applying the refinement operator $\rho_2$. It is easy to show that for all $i \geq 0$, the set $M_i$ contains precisely the ELQs of level $i$ that fit the positive examples $E^+$ in $E$.

So suppose that a most general fitting $q^*$ exists and that $\mathfrak{A}_2$ returns some ELQ $q$ after $n$ rounds. Then $q \in M_n$. Since $q$ is a fitting and $q^*$ is a most general fitting, we have $q \sqsubseteq q^*$. By Property (I) above, there is a non-empty $\Sigma$-specialization sequence from $q^*$ to $q$. Thus, the level of $q^*$ is strictly smaller than that of $q$. But then there is an $m < n$ such that the set $M_m$ contains $q^*$, in contradiction to $q$ being output by $\mathfrak{A}_2$.

To show that $\mathfrak{A}_1$ is a sample-efficient PAC learning algorithm we show that it is an Occam algorithm. Let $M_1, M_2, \ldots$ be the sequence of sets $M$ constructed by the breadth-first search algorithm $\mathfrak{A}_1$. We show below that

(i) for all $i \geq 1$, $q \in M_i$ implies $||q|| \leq 2^i - 1$, and

(ii) if $q$ is an ELQs with $||q|| \leq s$ that fits the positive examples $E^+$ in $E$, then there is an $i \leq 2\log(s)$ with $q \in M_i$.

Points (i) and (ii) imply that $\mathfrak{A}_1$ returns a fitting ELQ that is only polynomially larger than a smallest fitting ELQ and is thus Occam. Indeed, let a smallest fitting ELQ $q^*$ be of size $||q^*|| = s$ and let $q$ be the ELQ returned by $\mathfrak{A}_1$. By (ii), $\mathfrak{A}_1$ discovers $q^*$ after $2\log(s)$ rounds, which by definition of $\mathfrak{A}_1$ implies that $q$ is returned after at most $2\log(s)$ rounds. It thus follows from (i) that the returned ELQ $q$ satisfies

$$||q|| \leq 2^{2\log(s)} - 1 \in O(s^2).$$

Consequently, there is a polynomial $p$ such that $\mathcal{H}^{\mathfrak{A}_1}(\mathcal{O}, \Sigma, s, m)$ contains only ELQs $q$ with $||q|| \leq p(s)$. There are at most $|\Sigma|^{p(s)}$ such queries and since $2^{|S|}$ queries are needed to shatter a set $S$, the VC-dimension of $\mathcal{H}^{\mathfrak{A}_1}(\mathcal{O}, \Sigma, s, m)$ is at most $\log(|\Sigma|^{p(s)}) = p(s) \cdot \log(|\Sigma|)$. It then follows from Lemma 4 that $\mathfrak{A}_1$ is a sample-efficient PAC learning algorithm.

It thus remains to prove Points (i) and (ii). Point (i) can be shown by induction on $i$. For the induction start, it suffices to recall that $M_1 = \{\top\}$ and $||\top|| = 1$. For $i > 1$, $M_i$ consists of ELQs $p$ with $||p|| \leq 2||q|| + 1$ for some ELQ $q \in M_{i-1}$. Applying the induction hypothesis, we obtain

$$||p|| \leq 2 \cdot \left(2^{i-1} - 1\right) + 1 = 2^i - 1.$$

For Point (ii), let $q$ be an ELQ with $||q|| \leq s$ fitting $E^+$. It suffices to show that there is a $\rho_1, \Sigma$-refinement sequence $q_1, \ldots, q_k$ from $\top$ to $q$ with $k \leq 2\log(s)$.

Let $p_1, \ldots, p_m$ be a $\Sigma$-specialization sequence from $\top$ to $q$, that is, $p_{i+1}$ is a downward neighbor of $p_i$ (equivalently: $p_{i+1} \in \rho_2(p_i, \Sigma)$), for all $i$. Inductively define $q_1, q_2 \ldots$ as follows:

- $q_1 = p_1 = \top$, and
- for even numbers $j \geq 2$, let $\ell$ be maximal with $||p_\ell|| \leq 2||q_{j-1}|| + 1$, and set:
  - $q_j = p_\ell$, and
  - $q_{j+1} = p_{\ell+1}$ if $\ell < m$.

  Stop if $q_j$ or $q_{j+1}$ is $p_m$.

By construction, $||q_j|| \leq 2||q_{j-1}|| + 1$ for even $j \geq 2$. Moreover, for odd $j \geq 2$, $q_j \in \rho_2(q_{j-1}, \Sigma)$ and thus Lemma 2 implies that $||q_j|| \leq 2||q_{j-1}|| + 1$. Finally, observe that the construction ensures that $||q_{j+2}|| > 2||q_j||$, for all odd $j$ and thus the process stops after at most $2\log(s)$ steps. $\square$

## G Details for Section 5

From $E_\mathcal{O}$ and the bound $n$, SPELL constructs a propositional $\varphi = \varphi_1 \wedge \varphi_2$ that is satisfiable if and only if there is an ELQ $q$ over $\Sigma = \mathsf{sig}(E_\mathcal{O})$ with $n-1$ existential restrictions that fits $E_\mathcal{O}$.

Intuitively, $\varphi_1$ makes sure that every model of $\varphi$ encodes an ELQ $q$ in the variables $c_{i,A}, x_{i,r}, y_{i,j}$ as described in the main part. Recall that we use an arrangement of $n$ concepts $C_1, \ldots, C_n$. In what follows, we let $q$ denote the encoded ELQ and assume that $\mathcal{D}_q$ has individuals $1, \ldots, n$ (as indicated by $C_1, \ldots, C_n$) with 1 being the root. For encoding a proper arrangement, $\varphi_1$, it contains the following clauses for each $i$ with $2 \leq i \leq n$:

$$\bigvee_{j=1}^{i-1} y_{j,i} \tag{1}$$

$$\neg y_{j_1,i} \vee \neg y_{j_2,i} \quad \text{for all } j_1, j_2 \text{ with } 1 \leq j_1 < j_2 < i \tag{2}$$

$$\bigvee_{r \in \Sigma \cap \mathsf{N_R}} x_{i,r} \tag{3}$$

$$\neg x_{i,r} \vee \neg x_{i,r'} \quad \text{for all } r, r' \in \Sigma \cap \mathsf{N_R} \text{ with } r \neq r' \tag{4}$$

Clauses (1) and (2) ensure that $C_j$ appears in exactly one $C_i$, $i < j$ as a conjunct of the form $\exists r.C_j$ for some role name $r$. Clauses (3) and (4) ensure that there is a unique such role name.

The formula $\varphi_2$ makes sure that $q$ fits $E_\mathcal{O}$ by enforcing that

$(*)$ the variables $s_{i,a}$ are true in a model of $\varphi$ iff $a \in C_i(\mathcal{D})$ iff $(\mathcal{D}_q, i) \preceq (\mathcal{D}, a)$,

where $\mathcal{D}$ is the disjoint union of all databases that occur in $E_\mathcal{O}$. To achieve this, we implement the properties of simulations in terms of clauses. The challenge is to capture both directions of the "iff" in $(*)$ in an efficient way.

For all $a \in \mathsf{adom}(\mathcal{D})$, $\mathsf{type}(a)$ is the set $\{A \in \mathsf{N_C} \mid A(a) \in \mathcal{D}\}$. Let $\mathsf{TP} = \{\mathsf{type}(a) \mid a \in \mathsf{adom}(\mathcal{D})\}$ be the set of all types in that occur $\mathcal{D}$. We introduce auxiliary variables $t_{i,\tau}$, for every $1 \leq i \leq n$ and $\tau \in \mathsf{TP}$ with the intuition that $t_{i,\tau}$ is true iff all concept names that occur as a conjunct in $C_i$ are contained in $\tau$. This is enforced by including in $\varphi_2$ the following clauses for all $i$ with $1 \leq i \leq n$ and all types $\tau \in \mathsf{TP}$:

$$\neg t_{i,\tau} \vee \neg c_{i,A} \quad \text{for all } A \in (\Sigma \cap \mathsf{N_C} \setminus \tau) \tag{5}$$

$$t_{i,\tau} \vee \bigvee_{A \in (\Sigma \cap \mathsf{N_C} \setminus \tau)} c_{i,A} \tag{6}$$

The simulation condition on concept names is now enforced by the following clauses, for $i$ with $1 \leq i \leq n$ and all $a \in \mathsf{adom}(\mathcal{D})$:

$$\neg s_{i,a} \vee t_{i,\mathsf{type}(a)} \tag{7}$$

This captures, however, only the "only if"-direction of the "iff" in $(*)$. To implement the other direction and the simulation condition for role names, we introduce further auxiliary variables $d_{i,j,a}$ ($d$ as in *defect* to indicate non-simulation) with the intuitive meaning that $d_{i,j,a}$ is true iff $(\mathcal{D}_q, i) \not\preceq (\mathcal{D}, a)$ and there is an $r$-successor to $j$ that is not simulated in any $r$-successor of $a$ (the $r$ is uniquely determined by $j$ by Clauses (3) and (4)). This is achieved by the following clauses for all $i, j$ with $1 \leq i < j \leq n$, and $r \in \Sigma \cap \mathsf{N_R}$, $a \in \mathsf{adom}(\mathcal{D})$, and all $r(a, b) \in \mathcal{D}$:

$$s_{i,a} \vee \neg t_{i,\mathsf{type}(a)} \vee \bigvee_{k=i+1}^{n} d_{i,k,a} \tag{8}$$

$$d_{i,j,a} \vee \neg y_{i,j} \vee \neg x_{j,r} \vee \bigvee_{r(a,c) \in \mathcal{D}} s_{j,c} \tag{9}$$

$$\neg s_{i,a} \vee \neg d_{i,j,a} \tag{10}$$

$$\neg d_{i,j,a} \vee y_{i,j} \tag{11}$$

$$\neg d_{i,j,a} \vee \neg x_{j,r} \vee \neg s_{j,b} \tag{12}$$

As an example, Clause (11) can be read as follows: if there is a defect $d_{i,j,a}$, then $y_{i,j}$ must be true, meaning that $\exists r.C_j$ occurs as a conjunct in $C_i$.

It can be verified that the number of variables is $O(n^2 + n \cdot |\mathcal{D}|)$, the number of clauses is $O(n^3 \cdot |\Sigma| \cdot |\mathsf{adom}(\mathcal{D})|)$ and that the overall size of the formula is $O(n^3 \cdot |\Sigma| \cdot |\mathcal{D}|)$ as well.

Next, we give details on the additional clauses that break some symmetries in $\varphi$. As an example for these symmetries, consider the ELQ $\exists r.\exists s.\top \sqcap \exists t.\top$. In our encoding, it may be represented by the concepts

$$C_1 = \exists r.C_2 \sqcap \exists t.C_3,\ C_2 = \exists s.C_4,\ C_3 = C_4 = \top$$

or equivalently by the concepts

$$C_1' = \exists r.C_2' \sqcap \exists t.C_4',\ C_2' = \exists s.C_3',\ C_3' = C_4' = \top.$$

These different representations correspond to different models of $\varphi_1$. Consider the underlying graphs $G_{\mathcal{D}_C} = (\mathsf{adom}(\mathcal{D}_C), \{(a, b) \mid r(a, b) \in \mathcal{D}_C\})$ of concepts, where $\mathcal{D}_C$ is the concept $C$ viewed as a pointed database. Note that $G_{\mathcal{D}_{C_1}} = G_{\mathcal{D}_{C_1'}}$. The only difference between the arrangements $C_1, \ldots, C_4$ and $C_1', \ldots, C_4'$ comes from assigning them in a different way to the vertices of $G_{\mathcal{D}_{C_1}}$.

To avoid this, we add in Round $n$ of bounded fitting clauses that permit for every tree-shaped graph $G$ with $n$ vertices only a single canonical assignment of the concepts $C_1, \ldots C_n$ to the vertices of $G$. It suffices to consider tree-shaped graphs since $G_C$ is tree-shaped for every $\mathcal{EL}$-concept $C$. To produce the clauses, we enumerate (outside the SAT solver) all possible tree-shaped graphs with $n$ vertices. For each such graph $G$, we introduce a propositional variable $x_G$ and encode (in a straightforward way) that $x_G$ is true iff $C_1, \ldots, C_n$ are assigned to the vertices of $G$ in the canonical way. We then assert (with a big disjunction) that one of the $x_G$ has to be satisfied. However, note that there are exponentially many possible graphs and therefore we only add these clauses if $n < 12$, to avoid spending too much time and undoing the benefit of breaking this symmetry. It is an interesting research question how to break even more symmetries.

## H Size-restricted fitting for $\mathcal{EL}$ and $\mathcal{ELI}$

We analyze the complexity of the size-restricted fitting problem for ELQs, for ELIQs, and for the OMQ language $(\mathcal{ELI}, \text{ELIQ})$. Recall that universal databases in the sense defined before Proposition 1 do not exist for the latter, and in fact not even for $(\mathcal{EL}, \text{ELIQ})$. We discuss this a bit further at the end of this section. Recall that we generally assume unary coding of the input $k$ to the size-restricted fitting problem.[9] An investigation of the problem under binary coding is left as future work; a good starting point for this are results in [Jung *et al.*, 2019; Jung *et al.*, 2020].

**Lemma 10.** *The following problems are* NP*-complete:*

- *the size-restricted fitting problem for ELQs;*
- *the problem of deciding given a set of labeled examples $E$ and a number $k$, whether there is an ELQ that fits $E$ and that uses at most $k$ existential restrictions.*

*Proof.* The arguments are essentially identical, so we give the proof only for the size-restricted fitting problem.

For the NP upper bound, let $E, k$ be an input to the size-restricted fitting problem. Observe that we can guess in polynomial time an ELQ $q$ with $||q|| \leq k$ and verify in polynomial time that $a \in q(\mathcal{D})$ for all $(\mathcal{D}, a, +) \in E$ and $a \notin q(\mathcal{D})$ for all $(\mathcal{D}, a, -) \in E$. The latter is true since query evaluation of ELQs is possible in PTIME.

For NP-hardness, recall that the fitting problem for every class of unary conjunctive queries that includes all ELQs is NP-hard [ten Cate *et al.*, 2022]. The proof of that statement is by reduction from 3CNF-satisfiability. In more detail, a given 3CNF-formula $\varphi$ with $m$ variables is translated to a collection of labeled data examples $E$ such that $\varphi$ is satisfiable iff $E$ has a fitting ELQ of size $p(m)$ for some fixed polynomial $p$. Thus, it actually constitutes a reduction to the size-restricted fitting problem for ELQs. ☐

**Theorem 12.** *The size-restricted fitting problem is* NP*-complete for ELIQs and* EXPTIME*-complete for $(\mathcal{ELI}, \text{ELIQ})$.*

*Proof.* We start with the case without ontologies. For the NP upper bound, let $E, k$ be an input to the size-restricted fitting problem. Observe that we can guess in polynomial time an ELIQ $q$ with $||q|| \leq k$ and verify in polynomial time that $a \in q(\mathcal{D})$ for all $(\mathcal{D}, a, +) \in E$ and $a \notin q(\mathcal{D})$ for all $(\mathcal{D}, a, -) \in E$. The latter is true since query evaluation of ELIQs is possible in PTIME.

For NP-hardness, recall that the fitting problem for every class of unary conjunctive queries that includes all ELQs is NP-hard [ten Cate *et al.*, 2022]. The proof of that statement is by reduction from 3CNF-satisfiability. In more detail, a 3CNF-formula $\varphi$ with $m$ variables is translated to a collection of labeled data examples $E$ such that $\varphi$ is satisfiable iff $E$ has a fitting ELIQ of size $p(m)$ for some fixed polynomial $p$.

[9]This seems more relevant in the context of the current paper: it suffices for the size-restricted fitting problem to be in NP with $k$ coded in unary to enable a SAT approach to bounded fitting with the SAT formulas being of size polynomial in (the size of the data examples and) $k$.

Thus, it actually constitutes a reduction to the size-restricted fitting problem.

We now consider the OMQ language $(\mathcal{ELI}, \text{ELIQ})$. We show ExpTime-hardness by reduction from subsumption w.r.t. $\mathcal{ELI}$-ontologies which is known to be ExpTime-hard [Baader *et al.*, 2008]. Let $\mathcal{O}, A, B$ be an input to the subsumption problem, that is, the question is to decide whether $\mathcal{O} \models A \sqsubseteq B$. Construct a copy $\mathcal{O}'$ of $\mathcal{O}$ by replacing every concept name $X \in \text{sig}(\mathcal{O}) \setminus \{B\}$ with a fresh concept name $X'$ and every role name $r \in \text{sig}(\mathcal{O})$ with a fresh role name $r'$. Clearly, $\mathcal{O} \models A \sqsubseteq B$ iff $\mathcal{O}' \models A' \sqsubseteq B$. Then let $E$ consist of the two labeled examples

$$(\{A(a)\}, a, +) \qquad \text{and} \qquad (\{A'(a)\}, a, +).$$

Then $\mathcal{O} \models A \sqsubseteq B$ iff there is a fitting of $E$ w.r.t. $\mathcal{O} \cup \mathcal{O}'$ iff $B$ is a fitting of $E$ w.r.t. $\mathcal{O} \cup \mathcal{O}'$.

For the ExpTime-upper bound let $\mathcal{O}, E, k$ be an input to the size-restricted fitting problem. We provide a Turing reduction to subsumption w.r.t. $\mathcal{ELI}$-ontologies. In the reduction, we enumerate all ELIQs $q$ with $||q|| \leq k$, and test for each whether it fits $E$ w.r.t. $\mathcal{O}$ using an oracle for answering instance queries over $\mathcal{ELI}$ knowledge bases. Since there are only exponentially many candidates $q$, each test whether $q$ fits $E$ w.r.t. $\mathcal{O}$ uses only $|E|$ calls to the oracle. Since $k$ is encoded in unary, the inputs to the oracle are of polynomial size. Finally, as the oracle itself runs in exponential time [Baader *et al.*, 2008], the ExpTime-upper bound follows. $\square$

Let us return to the issue of universal databases in $(\mathcal{ELI}, \text{ELIQ})$. As mentioned above, universal databases as defined in the main body of the paper do not exist for this OMQ language. For bounded fitting, however, one might consider a weaker notion. For every database $\mathcal{D}$, $\mathcal{ELI}$-ontology $\mathcal{O}$, and $k \geq 1$, one can compute a database $\mathcal{U}_{\mathcal{D},\mathcal{O},k}$ that is *$k$-universal for ELIQs* in the sense that $a \in q(\mathcal{D} \cup \mathcal{O})$ iff $a \in q(\mathcal{U}_{\mathcal{D},\mathcal{O},k})$ for all ELIQs $q$ with at most $k$ existential restrictions (or of size at most $||k||$, a stronger condition) and all $a \in \mathsf{adom}(\mathcal{D})$. We do not give a detailed construction here, but only mention that such a database can be obtained from an infinite tree-shaped universal database by cutting off at depth $k$. What this means is that while we do not have available a universal database that works for *all* rounds of bounded fitting, for each single round $k$ we can compute a $k$-universal database to be used in that round. In contrast to the case of $(\mathcal{EL}, \text{ELQ})$, these $k$-universal databases may become exponential in size. One may hope, though, that their size is still manageable in practical cases. Note that keeping the ontology and treating it in the SAT encoding is not an option due to the ExpTime-hardness identified by Theorem 12.