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 δ has a classification error bounded by some small ϵ. 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.*, 2023a]. 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 [ten Cate *et al.*, 2023b].

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 Σ, 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 Σ a signature. A *Σ-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 Σ-simulation S from $\mathcal{D}_1$ to $\mathcal{D}_2$ with $(a_1, a_2) \in S$. We generally drop the mention of Σ 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

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

where Δ denotes symmetric difference and $\Pr_{(\mathcal{D},a)\sim P} X$ 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 $m : \mathbb{R}^2 \times \mathbb{N}^4 \rightarrow \mathbb{N}$ 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 Σ, 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 of [ten Cate *et al.*, 2023b].

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 Σ a signature. A set M of pointed databases is a Σ-*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].

Theorem 3. *Given an ELQ q and a finite signature Σ, a Σ-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 Σ-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 [ten Cate *et al.*, 2023b]. 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 [ten Cate *et al.*, 2023b].

Proof. To highlight the intuitions, we leave out some minor technical details that are provided in [ten Cate *et al.*, 2023b]. 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 Σ-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, $\mathsf{error}_{P, q_T}(q_H) = \frac{|S| - |E|}{|S|} > \frac{1}{2}$.

[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 [ten Cate *et al.*, 2023b].

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. $\square$

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 ρ that associates every $q \in Q_\Sigma$, $\mathcal{L}$-ontology $\mathcal{O}$, and finite signature Σ 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 ρ is *ideal* if it is finite and complete where ρ is

1. *finite* if $\rho(q, \mathcal{O}, \Sigma)$ is finite for all q, $\mathcal{O}$, and finite Σ, and
2. *complete* if for all finite signatures Σ 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 ρ 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 ρ 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 ρ_1 and ρ_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 ρ_2 is defined like ρ_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 ρ_1 and ρ_2 can be computed by brute force. For more elaborate approaches to computing ρ_2, see [Kriegel, 2021] where downward neighbors of ELQs are studied in detail.

Lemma 3. *ρ_1 and ρ_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 ρ_i. Although ρ_1 and ρ_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.

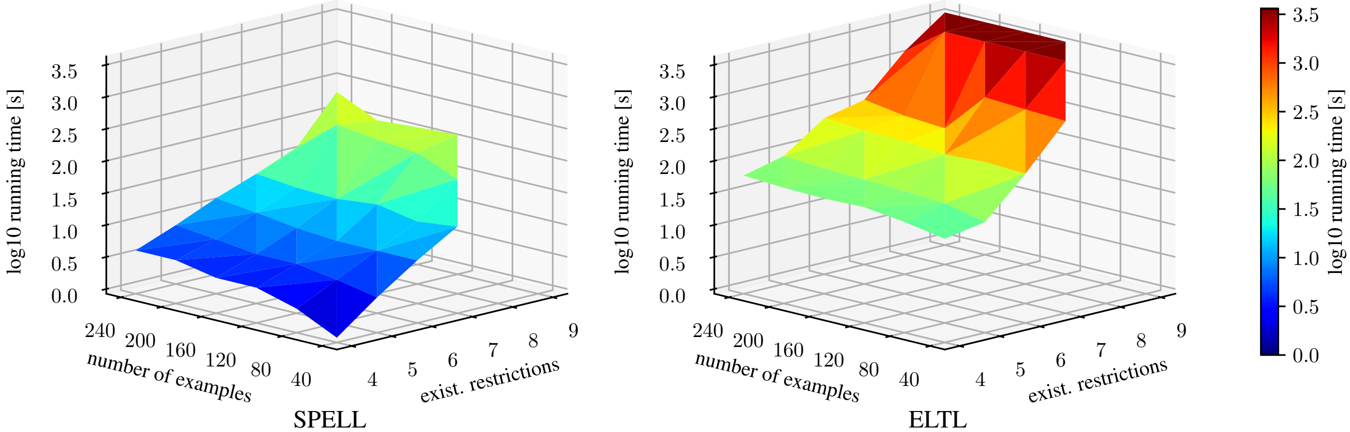

Figure 1: Yago experiment, dark red area indicates timeout (60min)

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 [ten Cate *et al.*, 2023b] . 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 $\Sigma = \text{sig}(E_{\mathcal{O}})$ with at most $n-1$ existential restrictions that fits $E_{\mathcal{O}}$. Indeed, any model of φ returned by the SAT solver uniquely represents a fitting ELQ q. More precisely, φ_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 φ_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 φ_2 uses simulations and relies on Lemma 1. The number of variables in φ 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 Σ 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 φ 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 φ so that the number of possible arrangements is reduced, breaking many symmetries. For details see [ten Cate *et al.*, 2023b].

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 of [ten Cate *et al.*, 2023b] 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 Σ_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.

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

[ten Cate *et al.*, 2023b] Balder ten Cate, Maurice Funk, Jean Christoph Jung, and Carsten Lutz. SAT-based PAC learning of description logic concepts. *CoRR*, abs/2305.08511, 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.