ExactLearner: a Tool for Exact Learning of $\mathcal{EL}$ Ontologies

Ricardo Duarte, Boris Konev, Ana Ozaki
Center for Advancing Electronics Dresden (cfaed), TU Dresden, Germany
University of Liverpool, United Kingdom
Free University of Bozen-Bolzano, Italy

Abstract

We present ExactLearner, a tool for exactly learning and teaching $\mathcal{EL}$ terminologies. The learning protocol follows Angluin's exact learning model, where an ontology engineer tries to identify an ontology by interacting with a domain expert by asking queries. We implement the learning process as a question-answer game between two components of our system, the learner and the teacher. We evaluate ExactLearner's performance on $\mathcal{EL}$ ontologies from the Oxford ontology repository and demonstrate that despite the algorithm being exponential, it successfully terminates for small and medium size ontologies. We investigate the impact of various learner and teacher features and identify those most useful for learning.

Introduction

Authoring ontologies is a laborious task that requires a combined expertise of domain experts, who know the vocabulary of terms used in a particular subject area and have an understanding of the conceptual relationships between them, and of knowledge engineers, who can formalise these relations in an appropriate ontology definition language. In (Konev et al. 2014; 2018) the dialogue between an expert and a knowledge engineer is formalised as an instance of Angluin's exact learning framework in which a *learner* tries to exactly identify an ontology by asking queries to the teacher, seen as an *oracle*. It is assumed that the vocabulary of terms is communicated directly to the learner and the emphasis of the learning process is on identifying the logical relations between the terms.

The contributions of this paper are twofold. First, we build on results of (Konev et al. 2018) and give an algorithm for learning $\mathcal{EL}$ terminologies, which is exponential in the size of concept expressions and its vocabulary but not in the size of the whole terminology. This result complements previous results showing that there is no polynomial time algorithm which can exactly learn (even acyclic) $\mathcal{EL}$ terminologies (Konev et al. 2014). We then introduce ExactLearner, a tool for exactly learning and teaching $\mathcal{EL}$ terminologies, which contains an implementation of our learning algorithm as well as a teacher. We evaluate ExactLearner's performance on $\mathcal{EL}$ ontologies from the Oxford ontology repository (Oxford) and demonstrate that despite the algorithm being exponential, it successfully terminates for small and medium size ontologies. We investigate the impact of various learner and teacher features and identify those most useful for learning. The missing proofs can be found in the full version of this paper available at `https://exactlearner.github.io`.

Related work. Most relevant to our work are: the DL-Learner (Lehmann 2009), which learns concept expressions (but not ontologies) in various fragments of description logic, using refinement operators; and systems based on the exact learning model, such as: Logan-H (Arias, Khardon, and Maloberti 2007) for learning function-free first order Horn sentences from interpretations; and EIRENE (Alexe et al. 2011), for learning schema mappings. For a more detailed discussion of related work, see (Konev et al. 2018).

Preliminaries

Description logic. Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be countably infinite sets of *concept* and *role* names. An $\mathcal{EL}$ concept expression C is formed according to the rule: $C, D := A \mid \top \mid C \sqcap D \mid \exists r.C$, where A ranges over $\mathsf{N_C}$ and r ranges over $\mathsf{N_R}$. A (general) $\mathcal{EL}$ *concept inclusion* has the form $C \sqsubseteq D$, where C and D are $\mathcal{EL}$ concept expressions. An $\mathcal{EL}$ *ontology* is a finite set of $\mathcal{EL}$ concept inclusions (Baader, Brandt, and Lutz 2005). We call an $\mathcal{EL}$ ontology $\mathcal{O}$ a *terminology* if for all $C \sqsubseteq D \in \mathcal{O}$ either C or D is a concept name and $\mathcal{O}$ has at most one[1] inclusion of the form $A \sqsubseteq C$ for every $A \in \mathsf{N_C}$. $\mathcal{EL}_{lhs}$ is the class of $\mathcal{EL}$ terminologies consisting only of inclusions of the form $C \sqsubseteq A$, while $\mathcal{EL}_{rhs}$ only of inclusions of the form $A \sqsubseteq C$.

The *size* $|C|$ of a concept expression C is the length of the string that represents it, where concept names

[1] In the literature, the term *terminology* commonly refers to sets of concept inclusions $A \sqsubseteq C$ and concept definitions $A \equiv C$, with no concept name occurring more than once on the left. As $A \equiv C$ can be equivalently rewritten as $A \sqsubseteq C$ and $C \sqsubseteq A$, our definition is a natural extension of this one.

and role names are considered to be of length one. An ontology *vocabulary* is the set of concept and role names occurring in the ontology. The size of a concept inclusion $C \sqsubseteq D$, denoted $|C \sqsubseteq D|$, is $|C| + |D|$ and the size of an ontology $\mathcal{O}$, denoted $|\mathcal{O}|$, is $\sum_{C \sqsubseteq D \in \mathcal{O}} |C \sqsubseteq D|$. The semantics of $\mathcal{EL}$ is defined as usual (Baader et al. 2003). We write $\mathcal{I} \models \alpha$ to say that a concept inclusion α is true in $\mathcal{I}$. An interpretation $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$ if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{O}$. $\mathcal{O} \models \alpha$ means that $\mathcal{I} \models \alpha$ for all models $\mathcal{I}$ of $\mathcal{O}$; and $\mathcal{O} \equiv \mathcal{O}'$ means that $\mathcal{O} \models \alpha$ if and only if $\mathcal{O}' \models \alpha$ for all concept inclusions α.

Subsumption learning framework. Given a class of ontologies $\mathcal{L}$ (for example all ontologies in a particular DL, $\mathcal{EL}$ terminologies etc), we are interested in the exact identification of a *target ontology* $\mathcal{O} \in \mathcal{L}$ by posing queries to an oracle. We assume that the vocabulary of the target terminology $\Sigma_{\mathcal{O}}$ is known to the learner. A *membership query* is a call to the oracle to test for an inclusion $C \sqsubseteq D$, where C, D are $\Sigma_{\mathcal{O}}$-concept expressions of the DL under consideration, if $\mathcal{O} \models C \sqsubseteq D$. An inclusion $C \sqsubseteq D$ is a *positive example* w.r.t. a target $\mathcal{O}$ if $\mathcal{O} \models C \sqsubseteq D$ and a *negative example* else. An *equivalence query* is a call to the oracle to check if a *hypothesis* ontology $\mathcal{H}$ is equivalent to the target $\mathcal{O}$. If it is the case, the oracle responds 'yes', otherwise the oracle returns a positive example $C \sqsubseteq D$ with $\mathcal{H} \not\models C \sqsubseteq D$ or a negative example $E \sqsubseteq F$ with $\mathcal{H} \models E \sqsubseteq F$. Such a positive example $C \sqsubseteq D$ (negative example $E \sqsubseteq F$) is called a *positive counterexample* (a *negative counterexample*, resp.) to $\mathcal{H}$ being equivalent to $\mathcal{O}$. For a formal definition of the *subsumption learning framework* and a discussion of how this definition relates to Angluin's exact learning model see (Konev et al. 2018).

We say that a class of ontologies $\mathcal{L}$ is *exactly learnable* if there is an algorithm, which halts for any target $\mathcal{O} \in \mathcal{L}$ and computes, using membership and equivalence queries, $\mathcal{H} \in \mathcal{L}$ with $\mathcal{H} \equiv \mathcal{O}$. An ontology class is exactly learnable in polynomial time if it is exactly learnable by an algorithm A such that at every step[2] of computation the time used by A up to that step is bounded by a polynomial $p(|\mathcal{O}|, |C \sqsubseteq D|)$, where $\mathcal{O}$ is the target and $C \sqsubseteq D$ is the largest counterexample seen so far. $\mathcal{EL}_{lhs}$ and $\mathcal{EL}_{rhs}$ are known to be exactly learnable in polynomial time, while the class of all $\mathcal{EL}$ ontologies is not learnable in polynomial time (Konev et al. 2014).

Learning $\mathcal{EL}$ Ontologies

In this section we present Algorithm 1, which can exactly learn $\mathcal{EL}$ terminologies in time exponential in $|C_{\mathcal{O}}|$, the size of the largest concept expression in $\mathcal{O}$, and $|\Sigma_{\mathcal{O}}|$, the size of the ontology vocabulary, but not in the size of the whole ontology.

In the main loop of the algorithm the learner poses an equivalence query to the oracle. If the oracle answers "yes" then the algorithm returns $\mathcal{H}$ equivalent to $\mathcal{O}$. Otherwise, it receives a counterexample $C \sqsubseteq D$. It is easy to see that at all times $\mathcal{O} \models \mathcal{H}$ so the counterexample is always positive.

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

Algorithm 1 The learning algorithm for $\mathcal{EL}$

Input: An $\mathcal{EL}$ terminology $\mathcal{O}$ given to the oracle; $\Sigma_{\mathcal{O}}$ given to the learner
Output: An $\mathcal{EL}$ terminology $\mathcal{H}$ computed by the learner such that $\mathcal{O} \equiv \mathcal{H}$

1: Set $\mathcal{H} = \{A \sqsubseteq B \mid \mathcal{O} \models A \sqsubseteq B,\ A, B \in \Sigma_{\mathcal{O}}\}$
2: **while** $\mathcal{H} \not\equiv \mathcal{O}$ **do**
3: Let $C \sqsubseteq D$ be the returned positive counterexample for $\mathcal{T}$ relative to $\mathcal{H}$
4: Compute $C' \sqsubseteq D'$ with C' or D' in $\Sigma_{\mathcal{O}} \cap \mathsf{N_C}$
5: **if** $C' \in \Sigma_{\mathcal{O}} \cap \mathsf{N_C}$ **then**
6: Compute a right $\mathcal{O}$-essential α from $C' \sqsubseteq D' \sqcap \prod_{C' \sqsubseteq F' \in \mathcal{H}} F'$
7: **else**
8: Compute a left $\mathcal{O}$-essential α from $C' \sqsubseteq D'$
9: **end if**
10: Add α to $\mathcal{H}$
11: **end while**
12: **return** $\mathcal{H}$

As $\mathcal{O}$ is a terminology, complex C and D in the counterexample can only "connect" via a concept name, which can be identified by asking membership queries. This is formalised by the following lemma, which is an extension of Theorem 16 in (Konev et al. 2012) and is proved by the canonical model construction.

Lemma 1. *Given a positive counterexample $C \sqsubseteq D$, one can construct, by posing membership queries, a positive counterexample $C' \sqsubseteq D'$ such that $|C' \sqsubseteq D'| \leq |C \sqsubseteq D|$ and either C' or D' is a concept name in time polynomial in $|\mathcal{H}|$, $|C|$ and $|\Sigma_{\mathcal{O}}|$.*

Having transformed the counterexample to the case of a concept name on the left or on the right, the algorithm tries to minimise the size of the counterexample. If C' is a concept name then Algorithm 1 merges D' with the right-hand sides of all inclusions in $\mathcal{H}$ with C' on the left (if they exist) and computes a so called *right $\mathcal{O}$-essential* counterexample. Otherwise, D' is a concept name, and the algorithm computes a *left $\mathcal{O}$-essential* counterexample. It then adds the resulting $\mathcal{O}$-essential concept inclusion α to $\mathcal{H}$.

To explain the left and right $\mathcal{O}$-essential counterexamples, following (Konev et al. 2014; 2018), we identify in the obvious way each $\mathcal{EL}$ concept expression C with a finite tree T_C whose nodes are labelled with sets of concept names and whose edges are labelled with roles.

Right $\mathcal{O}$-essential concept inclusion α is computed by applying exhaustively the following rules to $A \sqsubseteq C$:

Concept saturation for $\mathcal{O}$: If $\mathcal{O} \models A \sqsubseteq C'$ and C' results from C by adding a concept name A' to the label of some node, then replace $A \sqsubseteq C$ by $A \sqsubseteq C'$.

Sibling merging for $\mathcal{O}$: If $\mathcal{O} \models A \sqsubseteq C'$ and C' is the result of identifying in C two r-successors of the same node then replace $A \sqsubseteq C$ by $A \sqsubseteq C'$.

Decomposition on the right for $\mathcal{O}$: If d' is an r-successor of d in C, A' is in the node label of d, and $\mathcal{O} \models A' \sqsubseteq \exists r.C_{d'}$ plus $A' \not\equiv_{\mathcal{O}} A$ if d is the root of C, then replace $A \sqsubseteq C$ by

(a) $A' \sqsubseteq \exists r.C_{d'}$ if $\mathcal{H} \not\models A' \sqsubseteq \exists r.C_{d'}$; or

(b) $A \sqsubseteq C|^{-}_{d'\downarrow}$, otherwise, where

C_d is the concept corresponding to the subtree rooted in d and $C|^{-}_{d\downarrow}$ is the concept corresponding to the result of removing the subtree rooted in d from C.

We illustrate the transformation rules with examples.

1. For $\mathcal{H} = \emptyset$ and $\mathcal{O} = \{\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Human}\}$ the oracle can return an arbitrary long $\mathsf{hasParent}$ chain starting at Human as a counterexample, for instance, $\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\exists \mathsf{hasParent}.\top$ is a chain of length two. With concept saturation, this counterexample can be strengthened to $\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.(\mathsf{Human} \sqcap \exists \mathsf{hasParent}.\mathsf{Human})$, which is equivalent to $\mathcal{O}$.
2. For $\mathcal{O} = \{\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.(\mathsf{Human} \sqcap \mathsf{Male})\}$ and $\mathcal{H} = \{\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Human}\}$, upon receiving a counterexample $\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Male}$, the learner merges its right hand side with the right hand side of the inclusion in $\mathcal{H}$ to form $\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Male} \sqcap \exists \mathsf{hasParent}.\mathsf{Human}$ and then strengthens it by sibling merging to form the inclusion in $\mathcal{O}$.
3. For $\mathcal{H} = \emptyset$ and $\mathcal{O} = \{\mathsf{Woman} \sqsubseteq \mathsf{Human}, \mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Human}\}$, even with concept saturation, there exist infinitely many chain counterexamples; $\mathsf{Woman} \sqsubseteq \mathsf{Human} \sqcap \exists \mathsf{hasParent}.(\mathsf{Human} \sqcap \exists \mathsf{hasParent}.\mathsf{Human})$ is one of them. This inclusion can be decomposed at the root into (a) $\mathsf{Human} \sqsubseteq \mathsf{Woman}$ and (b) $\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.(\mathsf{Human} \sqcap \exists \mathsf{hasParent}.\mathsf{Human})$. Picking either of them allows the learner make progress.

Left $\mathcal{O}$-essential concept inclusion α is computed by applying exhaustively the following rules to $C \sqsubseteq A$.

Concept saturation for $\mathcal{H}$: If $\mathcal{H} \models C \sqsubseteq C'$ and C' results from C by adding a concept name A' to the label of some node, then replace $C \sqsubseteq A$ by $C' \sqsubseteq A$.

Decomposition on the left for $\mathcal{O}$: If d is a non-root node such that $\mathcal{O} \models C|^{-}_{d\downarrow} \sqsubseteq A'$, for some $A' \in \Sigma_{\mathcal{O}}$, then replace $C \sqsubseteq A$ by $C_d \sqsubseteq A'$, otherwise, if $\mathcal{O} \models C_d \sqsubseteq A'$ and $\mathcal{H} \not\models C_d \sqsubseteq A'$, for some $A' \in \Sigma_{\mathcal{O}}$, then replace $C \sqsubseteq A$ by $C_d \sqsubseteq A'$.

The applicability of a rule may depend on the application of another rule. For example, for $\mathcal{H} = \{\exists \mathsf{hasParent}.\top \sqsubseteq \mathsf{Human}\}$ and $\mathcal{O} = \mathcal{H} \cup \{\exists \mathsf{hasChild}.\mathsf{Human} \sqsubseteq \mathsf{Human}\}$ a counterexample could be $\exists \mathsf{hasChild}.\exists \mathsf{hasParent}.\top \sqsubseteq \mathsf{Human}$, which can only be decomposed on the left for $\mathcal{O}$ if we apply concept saturation for $\mathcal{H}$ first.

Our proof of termination of Algorithm 1 and its complexity bound is based on the following lemma. To simplify the presentation we use $\sharp_{\mathcal{O}}$ to denote $|C_{\mathcal{O}}| \cdot |\Sigma_{\mathcal{O}}| + 1$.

	p	# timouts	avg CE	avg max C
Test 2:	0.01	3	17.2	27.7
	0.5	25	107.8	26.6
	1.0	26	190.4	19.5
Test 4:	0.01	2	5.6	31.7
	0.5	3	6.1	31.6
	1.0	3	6.3	31.9

Table 1: Learner against the adversarial teacher.

Lemma 2. *Given a positive counterexample $C \sqsubseteq D$ for $\mathcal{O}$ relative to $\mathcal{H}$, one can construct a positive counterexample $C' \sqsubseteq D'$ such that $|C' \sqsubseteq D'| \leq \sharp_{\mathcal{O}}$ in polynomial time in $|C \sqsubseteq D|$, $|\Sigma_{\mathcal{O}}|$ and $|\mathcal{H}|$.*

Since there are at most $|\Sigma_{\mathcal{O}}|^{\sharp_{\mathcal{O}}}$ many inclusions over $\Sigma_{\mathcal{O}}$ of size $\sharp_{\mathcal{O}}$, at most $|\Sigma_{\mathcal{O}}|^{\sharp_{\mathcal{O}}}$ counterexamples get added to $\mathcal{H}$ over the run of the algorithm. Thus we obtain the following theorem.

Theorem 1. *The class of $\mathcal{EL}$ terminologies is exactly learnable by Algorithm 1 in $O(|\Sigma_{\mathcal{O}}|^{2|C_{\mathcal{O}}| \cdot |\Sigma_{\mathcal{O}}|+2} \cdot (|C \sqsubseteq D|)^2)$ time, where $\Sigma_{\mathcal{O}}$ is the vocabulary of the target terminology $\mathcal{O}$, $C_{\mathcal{O}}$ is largest concept expression in $\mathcal{O}$ and $C \sqsubseteq D$ is the largest counterexample seen so far by the algorithm.*

Concept saturation, sibling merging and decomposition on the right are all essential—hence the name—steps of the polynomial learning algorithm for DL-Lite$^{\exists}_{\mathcal{R}}$, which extends $\mathcal{EL}_{rhs}$ with inverse roles and role hierarchies (Konev et al. 2014; 2018). Indeed, Algorithm 1 polynomially learns $\mathcal{EL}_{rhs}$.

Theorem 2. *$\mathcal{EL}_{rhs}$ is exactly learnable in polynomial time by Algorithm 1.*

Evaluation

We have implemented our learning algorithm in the ExactLearner system, available at `https://github.com/ExactLearner/ExactLearner`, in Java using the OWL API (Horridge and Bechhofer 2011) and the ELK reasoner (Kazakov, Krötzsch, and Simancik 2014). ExactLearner has two main components: a learner and a teacher.

The learner supports (1) "Concept Saturation", (2) "Sibling Merging", (3) "Decomposition", applied on the right side of inclusions, and (4) "Concept Desaturation", (5) "Sibling Branching" and (6) "Decomposition", applied on the left. Operations (1), (2), (3) and (6) have already been described. In addition, we have also implemented (4) and (5), which act as heuristics to construct smaller, more informative counterexamples. Concept desaturation tries to remove concept names from nodes in the left of counterexamples to make them logically stronger. Sibling branching tries to strengthen a counterexample by splitting paths on the left. For example, for $\mathcal{O} = \{\exists \mathsf{hasDegree}.\mathsf{BSc} \sqcap \exists \mathsf{hasDegree}.\mathsf{MSc} \sqsubseteq \mathsf{PG}\}$ and $\mathcal{H} = \emptyset$, the inclusion $\exists \mathsf{hasDegree}.(\mathsf{BSc} \sqcap \mathsf{MSc} \sqcap \mathsf{PhD}) \sqsubseteq$

PG is a counterexample, from which desaturation removes the irrelevant PhD and then sibling branching strengthens it to the one in $\mathcal{O}$.

We have evaluated ExactLearner's performance on $\mathcal{EL}$ ontologies from the Oxford ontology repository (Oxford). As a first experiment we ran the learner against a naïve teacher, which presents the target ontology inclusions one by one without modification. This experiment aims at estimating the overheads of the learning process under the best possible conditions. In this first experiment, for 50 out of 797 $\mathcal{EL}$ ontologies computations concluded within 1 hour. We selected these ontologies for further experiments. The selected ontologies range in size from 9 to 11 177 inclusions with signature sizes ranging from 23 to 9334 concept names and from 2 to 25 role names. The average size of counterexamples produced by the teacher was 5.48 while the average size of the largest concept in $\mathcal{O}$ was 2.7. The average size of the largest concept in $\mathcal{H}$ was 31.3, an increase caused by concept saturation on the right side of inclusions. The performance bottlenecks in our system are checking if the presented inclusion is a counterexample w.r.t. the current hypothesis ontology at the server side and entailment checks in the learner.

To further challenge the learner, we have introduced an adversarial teacher, which forces the learner to apply particular operations from (1)–(6) above by manipulating the counterexamples. For instance, to force the learner to perform concept saturation on the right of $A \sqsubseteq C$, the teacher exhaustively tries to remove concept names from every node in the tree representation of C, while ensuring that the modified inclusion is still a counterexample. All in all, the adversarial teacher can apply: (7) "Concept Desaturation" on the right, which we have just described; (8) "Sibling Branching" on the right, which weakens counterexamples of the form $A \sqsubseteq \exists r.(C \sqcap D)$ into $A \sqsubseteq \exists r.C \sqcap \exists r.D$ (provided the latter is still a counterexample); (9) "Concept Saturation" on the left; and (10) "Sibling Merging" on the left, which are the opposite of learner's concept desaturation and sibling branching. We also substitute concept definitions into counterexamples, for instance, if $A \sqsubseteq \exists r.B$ is a counterexample and $B \sqsubseteq C \in \mathcal{T}$ we test $A \sqsubseteq \exists r.C$ for being a counterexample as well. We call this operation (11) "Composition on the right". (12) "Composition on the left" is its counterpart. Operations (7)–(12) are applied at random with set probabilities so that the level of difficulty could be controlled.

Table 1 presents statistics of running the learner against the adversarial teacher. In Test 2 the teacher was applying transformations (7)–(12) with probability p of 0.01, 0.5 and 1.0. The learner can cope with a small distortion of examples ($p = 0.01$) but a significant distortion leads to a big increase in the number of time-outs. Figure 1 shows the percentage of the rules applied by the learner in Test 2 for $p = 0.01$. As the chart indicates, the most frequently applied rule (42%) is desaturation in the left. This number grows to 94% when $p = 0.5$ and to 96% when $p = 1.0$.

In Test 3 we have disabled rule (9) at the oracle side as well as rules (8) and (10) as they almost never applied, see Figure 1, yet took up a significant time in our tests. This led to a significant drop in the failure rate even though other adversarial teacher operations were applied with a high probability. This suggests that the main cause of time-outs is the exponential explosion in the size of the signature rather than the size of concepts in the target ontology.

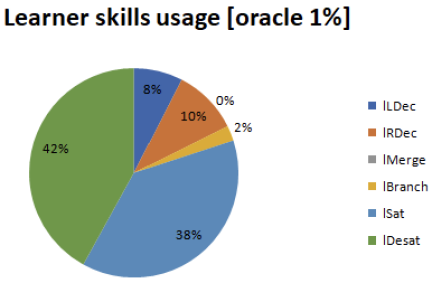

Figure 1: Usage of rules (1)–(6) by the learner.

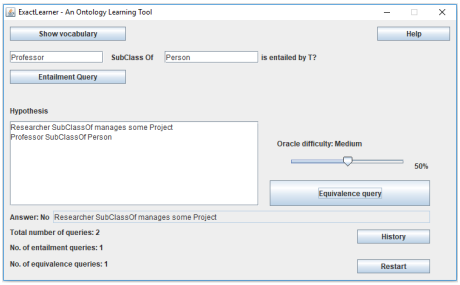

Figure 2: Learning game after 2 queries.

Playing with the Teacher. Our prototype teacher component can also be accessed via a graphical interface allowing a user to play the game of learning an ontology by posing as few membership and equivalence queries as possible. Fig. 2 presents a screenshot of the game after 2 queries.

Conclusion

We presented ExactLearner, a prototype tool for exactly learning, and teaching, $\mathcal{EL}$ ontologies. We demonstrated its applicability to small and medium size ontologies. We identified the size of the ontology signature as the main cause of the performance bottleneck.

As future work, we plan to transform our learning algorithm in the Probably Approximately Correct with membership queries setting, so that our algorithm can also run without the teacher. We also plan to investigate the complexity of exactly learning of $\mathcal{EL}$ terminologies in the PAC learning setting under different probability distributions.

Acknowledgements. We would like to thank Frank Wolter for fruitful discussions and Liyi Zhao for her contribution to an earlier version of ExactLearner.

References

Alexe, B.; ten Cate, B.; Kolaitis, P. G.; and Tan, W. C. 2011. EIRENE: interactive design and refinement of schema mappings via data examples. *PVLDB* 4(12):1414–1417.

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

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

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

Horridge, M., and Bechhofer, S. 2011. The OWL API: A java API for OWL ontologies. *Semant. web* 2(1):11–21.

Kazakov, Y.; Krötzsch, M.; and Simancik, F. 2014. The incredible ELK - from polynomial procedures to efficient reasoning with EL ontologies. *J. Autom. Reasoning* 53(1):1–61.

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

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

Konev, B.; Lutz, C.; Ozaki, A.; and Wolter, F. 2018. Exact learning of lightweight description logic ontologies. *Journal of Machine Learning Research*. (in press). Available from `http://www.csc.liv.ac.uk/~frank/publ/jmlr18.pdf`.

Lehmann, J. 2009. Dl-learner: Learning concepts in description logics. *Journal of Machine Learning Research* 10:2639–2642.

Oxford. Information systems group ontologies. Retrieved from `https://www.cs.ox.ac.uk/isg/ontologies/`. Accessed: 18 May 2018.

Proofs for Section

To show Lemma 2 we first need the following technical lemma, which uses the canonical model of a concept and an ontology. The *canonical model* $\mathcal{I}_{C,\mathcal{O}}$ of a concept expression C and an ontology $\mathcal{O}$ is defined as follows. If $\mathcal{O} = \emptyset$, then we want $\mathcal{I}_{C,\mathcal{O}}$ to be T_C (defined in the Preliminaries) viewed as a tree-shaped interpretation which we denote by $\mathcal{I}_C$ rather than by $\mathcal{I}_{C,\mathcal{O}}$. In detail, the domain of $\mathcal{I}_C$ is the set of nodes of T_C and for all $A \in \mathsf{N_C}$ and all $r \in \mathsf{N_R}$:

$$d \in A^{\mathcal{I}_C} \quad \text{iff} \quad A \in l(d), \text{ for all } d \in \Delta^{\mathcal{I}_C};$$
$$(d, d') \in r^{\mathcal{I}_C} \quad \text{iff} \quad r = l(d, d'), \text{ for all } d, d' \in \Delta^{\mathcal{I}_C}.$$

We call the root ρ_C of T_C the *root of* $\mathcal{I}_C$. If $\mathcal{O} \neq \emptyset$, then $\mathcal{I}_{C,\mathcal{O}}$ is obtained by extending $\mathcal{I}_C$ so that the CIs in $\mathcal{O}$ are satisfied. More precisely, $\mathcal{I}_{C,\mathcal{O}}$ is defined as the limit of a sequence $\mathcal{I}_0, \mathcal{I}_1, \ldots$ of interpretations, where $\mathcal{I}_0 = \mathcal{I}_C$. For the inductive definition of the sequence, assume that $\mathcal{I}_n$ has been defined. Then obtain $\mathcal{I}_{n+1}$ by applying one of the following rules once:

1. if $C \sqsubseteq D \in \mathcal{O}$ and $d \in C^{\mathcal{I}_n}$ but $d \notin D^{\mathcal{I}_n}$, then take the interpretation $\mathcal{I}_D$ and add it to $\mathcal{I}_n$ by identifying its root ρ_C with d. In more detail, assume that $\Delta^{\mathcal{I}_n} \cap \Delta^{\mathcal{I}_C} = \{d\}$ and $d = \rho_C$ and define $\mathcal{I}_{n+1}$ by setting, for all concept names A and role names r:

$$\Delta^{\mathcal{I}_{n+1}} = \Delta^{\mathcal{I}_n} \cup \Delta^{\mathcal{I}_C},$$
$$A^{\mathcal{I}_{n+1}} = A^{\mathcal{I}_n} \cup A^{\mathcal{I}_C}, \quad r^{\mathcal{I}_{n+1}} = r^{\mathcal{I}_n} \cup r^{\mathcal{I}_C}.$$

We assume that rule application is fair, that is, if a rule is applicable in a certain place, then it will indeed eventually be applied there. If for some $n > 0$ no rule is applicable then we set $\mathcal{I}_{n+1} = \mathcal{I}_n$. We obtain $\mathcal{I}_{C,\mathcal{O}}$ by setting for all concept names A and role names r:

$$\Delta^{\mathcal{I}_{C,\mathcal{O}}} = \bigcup_{n \geq 0} \Delta^{\mathcal{I}_n}, \; A^{\mathcal{I}_{C,\mathcal{O}}} = \bigcup_{n \geq 0} A^{\mathcal{I}_n}, \; r^{\mathcal{I}_{C,\mathcal{O}}} = \bigcup_{n \geq 0} r^{\mathcal{I}_n}.$$

Lemma 3. *Let $\mathcal{O}$ be an $\mathcal{EL}$ terminology. If $\mathcal{O} \models C \sqsubseteq \exists r.F$ then either:*

- *C has a conjunct of the form $\exists r.F'$ such that $\mathcal{O} \models F' \sqsubseteq F$ or;*
- *$\mathcal{O} \models C \sqsubseteq A$ and $\mathcal{O} \models A \sqsubseteq \exists r.F$, for some $A \in \Sigma_{\mathcal{O}}$.*

Proof. Let $\mathcal{I}_{C,\mathcal{O}}$ be the canonical model of C and $\mathcal{O}$. The interpretation $\mathcal{I}_{C,\mathcal{O}}$ has the following property (which can be proved using the construction of $\mathcal{I}_{C,\mathcal{O}}$):

† for any $d \in \Delta^{\mathcal{I}_C}$ and $\mathcal{EL}$ concept expression of the form $\exists r.F$: if there exists a homomorphism h from the labeled tree corresponding to $\exists r.F$ into $\mathcal{I}_{C,\mathcal{O}}$ such that $h(a_{\exists r.F}) = d$ and all nodes in the labeled tree corresponding to F are mapped to $\Delta^{\mathcal{I}_{C,\mathcal{O}}} \setminus \Delta^{\mathcal{I}_C}$, then there exists a concept name E with $d \in E^{\mathcal{I}_C}$ such that $\mathcal{O} \models E \sqsubseteq \exists r.F$.

If $\mathcal{O} \models C \sqsubseteq \exists r.F$ then by (†), there is a homomorphism $h : \mathcal{I}_{\exists r.F} \to \mathcal{I}_{C,\mathcal{O}}$ mapping the root of $\mathcal{I}_{\exists r.F}$ to the root of $\mathcal{I}_{C,\mathcal{O}}$. Let d be the node corresponding to the child of $\rho_{\exists r.F}$. By construction of $\mathcal{I}_{C,\mathcal{O}}$ we have that $\Delta^{\mathcal{I}_C} \subseteq \Delta^{\mathcal{I}_{C,\mathcal{O}}}$. We make a case distinction:

- $h(d) \in \Delta^{\mathcal{I}_C}$: let F' be the concept corresponding to the subtree rooted in $h(d)$. Then there is a homomorphism $h : \mathcal{I}_F \to \mathcal{I}_{F',\mathcal{O}}$. This means that $\mathcal{O} \models F' \sqsubseteq F$. So C has a conjunct $\exists r.F'$ such that $\mathcal{O} \models F' \sqsubseteq F$.
- $h(d) \notin \Delta^{\mathcal{I}_C}$: then, by construction of $\mathcal{I}_{C,\mathcal{O}}$, there is a concept name A such that $\rho \in A^{\mathcal{I}_{C,\mathcal{O}}}$, that is $\mathcal{O} \models C \sqsubseteq A$, and $\mathcal{O} \models A \sqsubseteq \exists r.F$.

We split the proof of Lemma 2 into three parts: Lemma 4 shows that given a counterexample α we can compute a counterexample with size bounded by $|\alpha|$ where one of the sides is a concept name; and Lemmas 5 and 6 show that given a counterexample of the form $A \sqsubseteq C$ or the form $C \sqsubseteq A$, respectively, one can compute a counterexample with size bounded by $|C_{\mathcal{O}}| + 1$.

Lemma 4. *Given a positive counterexample $C \sqsubseteq D$ for $\mathcal{O}$ relative to $\mathcal{H}$, one can construct a positive counterexample $C' \sqsubseteq D'$ such that $|C' \sqsubseteq D'| \leq |C \sqsubseteq D|$ and either C' or D' is a concept name in polynomial time in $|\mathcal{H}|$, $|C|$ and $|\Sigma_{\mathcal{O}}|$.*

Proof. If (a) C is a concept name or (b) there is a concept name $A \in \Sigma_{\mathcal{O}} \cap \mathsf{N_C}$ such that $C \sqsubseteq A$ is a positive counterexample then we are done. If (b) is not the case then:

‡ for all $A \in \mathsf{N_C}$, $\mathcal{O} \models C \sqsubseteq A$ iff $\mathcal{H} \models C \sqsubseteq A$.

If D is of the form $D_1 \sqcap ... \sqcap D_n$ then we know that there is D_i, $1 \leq i \leq n$, such that $C \sqsubseteq D_i$ is a positive counterexample. By (‡) we can assume this conjunct to be of the form $\exists r.F$. Now, we make a case distinction:

1. If C has a conjunct $\exists r.F'$ such that $\mathcal{O} \models F' \sqsubseteq F$ then we can apply this lemma with $F' \sqsubseteq F$ as positive counterexample. Notice that $\mathcal{H} \not\models F' \sqsubseteq F$, because if not then $C \sqsubseteq \exists r.F$ would not be a positive counterexample.
2. Otherwise, by Lemma 3 and (‡), there is a concept name $A \in \Sigma_{\mathcal{O}} \cap \mathsf{N_C}$ such that $A \sqsubseteq \exists r.F$ is a positive counterexample.

Since each time case 1 happens the concept expression C is strictly smaller, this can happen at most $|C|$ times. This means that either (a) or (b) happens or there is a concept name A such that $A \sqsubseteq \exists r.F$ is a positive counterexample, where $\exists r.F$ is a conjunct of D. In all cases the lemma holds.

Lemma 5. *Given a positive counterexample $A \sqsubseteq C$ for $\mathcal{O}$ relative to $\mathcal{H}$, one can construct a positive counterexample $A' \sqsubseteq C'$ such that $|\Delta^{\mathcal{I}_{C'}}| \leq |\Delta^{\mathcal{I}_{C_{\mathcal{O}}}}|$, where $C_{\mathcal{O}}$ is the largest concept expression in $\mathcal{O}$, in polynomial time in $|\mathcal{H}|$, $|C|$ and $|\Sigma_{\mathcal{O}}|$.*

Proof. Let $A \sqsubseteq C$ be a positive counterexample for $\mathcal{O}$ relative to $\mathcal{H}$. We exhaustively apply the rules concept saturation for $\mathcal{O}$, sibling merging for $\mathcal{O}$ and decomposition on the right for $\mathcal{O}$, which rely on posing membership queries to the oracle.

We can see that the number of rule applications is bounded by $|C|^2 \cdot |\Sigma_{\mathcal{O}}|$. The fact that $A \sqsubseteq C$ is a positive counterexample for $\mathcal{O}$ relative to $\mathcal{H}$ is straightforward for the first two rules. If $A \sqsubseteq C$ is replaced by $A \sqsubseteq C'$ then $\models C' \sqsubseteq C$. Hence $\mathcal{H} \not\models A \sqsubseteq C'$. Regarding the rule decomposition on the right for $\mathcal{O}$, we have $\{A \sqsubseteq C|^-_{d'\downarrow}, A' \sqsubseteq \exists r.C'\} \models A \sqsubseteq C$. Thus, one of the inclusions $A \sqsubseteq C|^-_{d'\downarrow}$ and $A' \sqsubseteq \exists r.C'$ is not entailed by $\mathcal{H}$, and this is the concept inclusion resulting from the rule application. It remains to show that $A \sqsubseteq C$ is such that $|\Delta^{\mathcal{I}_C}| \leq |\Delta^{\mathcal{I}_{C_{\mathcal{O}}}}|$. The proof follows the same lines of Lemma 4 in (Konev et al. 2014), which we briefly sketch here. First we define

$$A^{\mathcal{O}} = \{A\} \cup \{D \mid \mathcal{O} \models A \sqsubseteq B, B \sqsubseteq D \in \mathcal{O}\}$$

and construct the canonical model $\mathcal{I}_{D_0,\mathcal{O}}$ of $D_0 = \bigsqcap_{D \in A^{\mathcal{O}}} D$ and $\mathcal{O}$. The interpretation $\mathcal{I}_{D_0,\mathcal{O}}$ has the following properties:

1. for every $\mathcal{EL}$ concept expression F: $\rho_{D_0} \in F^{\mathcal{I}_{D_0,\mathcal{O}}}$ iff $\mathcal{O} \models A \sqsubseteq F$;
2. for any $d \in \Delta^{\mathcal{I}_{D_0}}$ and $\mathcal{EL}$ concept expression of the form $F = \exists r.F'$: if there exists a homomorphism h from the labeled tree corresponding to F into $\mathcal{I}_{D_0,\mathcal{O}}$ such that $h(a_F) = d$ and all nodes in the labeled tree corresponding to F' are mapped to $\Delta^{\mathcal{I}_{D_0,\mathcal{O}}} \setminus \Delta^{\mathcal{I}_{D_0}}$, then there exists a concept name E with $d \in E^{\mathcal{I}_{D_0}}$ such that $\mathcal{O} \models E \sqsubseteq F$.

Then, one can show that there is an injective homomorphism h from the labeled tree T_C of C into the restriction $\mathcal{J}$ of $\mathcal{I}_{D_0,\mathcal{O}}$ to $\Delta^{\mathcal{I}_0}$ which maps a_C to ρ_{D_0}. By definition of $A^{\mathcal{O}}$, there is $B \sqsubseteq D \in \mathcal{O}$ such that $|\Delta^{\mathcal{I}_C}| \leq |\Delta^{\mathcal{I}_D}|$, which means that $|\Delta^{\mathcal{I}_C}| \leq |\Delta^{\mathcal{I}_{C_{\mathcal{O}}}}|$, where $C_{\mathcal{O}}$ is the largest concept expression in $\mathcal{O}$. The main observation here is that the argument holds for $\mathcal{EL}$ terminologies, that is, in the presence of concept inclusions of the form $C \sqsubseteq A$. Such inclusions do not interfere in the argument since concept saturation ensures that nodes have all concept names implied by $\mathcal{O}$ in their labels.

Lemma 6. *Given a positive counterexample $C \sqsubseteq A$ for $\mathcal{O}$ relative to $\mathcal{H}$, one can construct a positive counterexample $C' \sqsubseteq A'$ such that $|\Delta^{\mathcal{I}_{C'}}| \leq |\Delta^{\mathcal{I}_{C_{\mathcal{O}}}}|$, where $C_{\mathcal{O}}$ is the largest concept expression in $\mathcal{O}$, in polynomial time in $|\mathcal{H}|$, $|C|$ and $|\Sigma_{\mathcal{O}}|$.*

Proof. Let $C \sqsubseteq A$ be a positive counterexample for $\mathcal{O}$ relative to $\mathcal{H}$. We exhaustively apply the rules concept saturation for $\mathcal{H}$, and decomposition on the left for $\mathcal{O}$, which rely on posing membership queries to the oracle.

We can see that the number of rule applications is bounded by $|C| \cdot |\Sigma_{\mathcal{O}}|$. The fact that $C \sqsubseteq A$ is a positive counterexample for $\mathcal{O}$ relative to $\mathcal{H}$ is straightforward for the first rule. Regarding decomposition on the left for $\mathcal{O}$, either we take $C|^-_{d\downarrow} \sqsubseteq A$, which means that $\models C \sqsubseteq C|^-_{d\downarrow}$, or we take $C_d \sqsubseteq A'$ such that $\mathcal{O} \models C_d \sqsubseteq A'$ and $\mathcal{H} \not\models C_d \sqsubseteq A'$. In all cases, the resulting concept inclusion is a positive counterexample. It remains show that $C \sqsubseteq A$ is such that $|\Delta^{\mathcal{I}_C}| \leq |\Delta^{\mathcal{I}_{C_{\mathcal{O}}}}|$.

If there is $E \in \Sigma_{\mathcal{O}}$ such that $\mathcal{O} \models C \sqsubseteq E$ (and $C \sqsubseteq E$ is not a tautology) then there is $D \sqsubseteq E \in \mathcal{O}$ such that $\rho_C \in D^{\mathcal{I}_{C,\mathcal{O}}}$. Let $(C,\mathcal{O})^{\mathcal{I}_C} = \{D \mid \exists E \in \Sigma_{\mathcal{O}} : \rho_C \in D^{\mathcal{I}_{C,\mathcal{O}}}, \rho_C \notin E^{\mathcal{I}_C}, D \sqsubseteq E \in \mathcal{O}\}$. Then, for all $D \in (C,\mathcal{O})^{\mathcal{I}_C}$ there is a homomorphism $h : \mathcal{I}_D \to \mathcal{I}_{C,\mathcal{O}}$ mapping the root of $\mathcal{I}_D$ to the root of $\mathcal{I}_{C,\mathcal{O}}$. Also, there is $D' \in (C,\mathcal{O})^{\mathcal{I}_C}$ such that:

1. there is a partial homomorphism $h : \mathcal{I}_{D'} \to \mathcal{I}_C$ mapping the root of $\mathcal{I}_{D'}$ to the root of $\mathcal{I}_C$, for $e \in \Delta^{\mathcal{I}_{D'}}$ with $h(e) \in \Delta^{\mathcal{I}_C}$;
2. if D' has a conjunct $\exists r.F$, with $e_1 \in F^{\mathcal{I}_{D'}}$ as the corresponding r-successor of ρ_D, and $h(e_1) \notin \Delta^{\mathcal{I}_C}$ then there is $A \in \mathsf{N_C}$ such that $\rho_C \in A^{\mathcal{I}_C}$ and $\mathcal{O} \models A \sqsubseteq \exists r.F$.

The fact that $|\Delta^{\mathcal{I}_C}| \leq |\Delta^{\mathcal{I}_{D'}}|$ for some $D' \sqsubseteq E \in \mathcal{O}$ is a result of the following claim.

Claim 1. For all $d \in \Delta^{\mathcal{I}_C}$, there is $e \in \Delta^{\mathcal{I}_{D'}}$ such that $d = h(e)$.

Suppose to the contrary that there is $d' \in \Delta^{\mathcal{I}_C}$ such that d' is outside the image of $h : \mathcal{I}_{D'} \rightarrow \mathcal{I}_{C,\mathcal{O}}$. Let $G = C|^{-}_{d'\downarrow}$ be the concept corresponding to the result of removing the subtree rooted in d'. Since C is concept saturated for $\mathcal{H}$, If $\rho_G \in D'^{\mathcal{I}_{G,\mathcal{O}}}$ and $\rho_G \notin E^{\mathcal{I}_G}$ then this contradicts the fact that decomposition on the left for $\mathcal{O}$ was exhaustively applied. To show the contradiction, we argue that h is a homomorphism from $\mathcal{I}_{D'}$ into the restriction $\mathcal{I}_{G,\mathcal{O}}$ of $\mathcal{I}_{C,\mathcal{O}}$. Our construction has the following properties:

3. for $e \in \Delta^{\mathcal{I}_{D'}}$ with $h(e) \in \Delta^{\mathcal{I}_C}$, $h(e) \in \Delta^{\mathcal{I}_G}$;
4. for all $d \in \Delta^{\mathcal{I}_C} \setminus \{\rho_C\}$, $d \in A^{\mathcal{I}_C}$ iff $d \in A^{\mathcal{I}_{C,\mathcal{O}}}$, for all $A \in \mathsf{N_C}$.

We obtain Point (3) by the fact that since d' is outside the image of $h : \mathcal{I}_{D'} \rightarrow \mathcal{I}_{C,\mathcal{O}}$, all descendants of d' are outside the image of $h : \mathcal{I}_{D'} \rightarrow \mathcal{I}_{C,\mathcal{O}}$ as well. Point (4) follows from decomposition of the left for $\mathcal{O}$ and concept saturation for $\mathcal{H}$.

For $e \in \Delta^{\mathcal{I}_{D'}}$ with $h(e) \in \Delta^{\mathcal{I}_C}$, by Points (1) and (3), there is a partial homomorphism from $\mathcal{I}_{D'}$ to $\mathcal{I}_G$. For $e \in \Delta^{\mathcal{I}_{D'}}$ with $h(e) \notin \Delta^{\mathcal{I}_C}$, there is $e_1 \in \Delta^{\mathcal{I}_{D'}}$ such that: (i) $h(e_1) \notin \Delta^{\mathcal{I}_C}$; (ii) e is in the subtree rooted in e_1; and (iii) e_1 has minimal distance from $\rho_{D'}$. This means that the parent e_2 of e_1 is such that $h(e_2) \in \Delta^{\mathcal{I}_C}$. Let F be the concept corresponding to the subtree rooted in e_1 and let s be the role between e_2 and e_1. That is, $(e_2, e_1) \in s^{\mathcal{I}_{D'}}$. Since $h(e_1) \notin \Delta^{\mathcal{I}_C}$ we have that all descendants of e_1 are not mapped to $\Delta^{\mathcal{I}_C}$. Then, by construction of the canonical model $\mathcal{I}_{C,\mathcal{O}}$ of C and the $\mathcal{EL}$ terminology $\mathcal{O}$, one can show that there is a $A' \in \mathsf{N_C}$ such that $h(e_2) \in A'^{\mathcal{I}_{C,\mathcal{O}}}$ and $\mathcal{O} \models A' \sqsubseteq \exists s.F$. So it remains to show that $h(e_2) \in A'^{\mathcal{I}_{G,\mathcal{O}}}$, for some $A' \in \mathsf{N_C}$ such that $\mathcal{O} \models A' \sqsubseteq \exists s.F$. We make a case distinction:

- for $h(e_2) \neq \rho_C$: by Point (4), $h(e_2) \in A'^{\mathcal{I}_{C,\mathcal{O}}}$ implies $h(e_2) \in A'^{\mathcal{I}_C}$. As $h(e_2) \in \Delta^{\mathcal{I}_C}$, by Point (3), we have that $h(e_2) \in \Delta^{\mathcal{I}_G}$. So $h(e_2) \in A'^{\mathcal{I}_G}$, and thus, $h(e_2) \in A'^{\mathcal{I}_{G,\mathcal{O}}}$.
- for $h(e_2) = \rho_C$: by Point (2), if $e_1 \in F^{\mathcal{I}_{D'}}$ and $h(e_1) \notin \Delta^{\mathcal{I}_C}$ then there is a concept name A' such that $\rho_C \in A'^{\mathcal{I}_C}$ and $\mathcal{O} \models A' \sqsubseteq \exists s.F$. So $\rho_G \in A'^{\mathcal{I}_G}$, which means that $\rho_G = h(e_2) \in A'^{\mathcal{I}_{G,\mathcal{O}}}$.

Since we showed for arbitrary $e \in \Delta^{\mathcal{I}_{D'}}$ with $h(e) \in \Delta^{\mathcal{I}_C}$ and $h(e) \notin \Delta^{\mathcal{I}_C}$ that there is $d \in \Delta^{\mathcal{I}_{G,\mathcal{O}}}$ such that $d = h(e)$, we have that $h : \mathcal{I}_{D'} \rightarrow \mathcal{I}_{G,\mathcal{O}}$ is a homomorphism. Thus, there is $D' \in (C, \mathcal{O})^{\mathcal{I}_G}$ such that $\rho_G \in D'^{\mathcal{I}_{G,\mathcal{O}}}$ and $\rho_G \notin E^{\mathcal{I}_G}$ for some $D' \sqsubseteq E \in \mathcal{O}$, which contradicts the fact that decomposition on the left for $\mathcal{O}$ was exhaustively applied, so our claim follows.

Lemma 7 is used to show Theorem 2.

Lemma 7. *[Adaptation of (Konev et al. 2014)] Assume that $A \sqsubseteq C_1$ and $A \sqsubseteq C_2$ are right $\mathcal{O}$-essential. Then one can construct a right $\mathcal{O}$-essential $A \sqsubseteq C$ such that $\emptyset \models C \sqsubseteq C_1 \sqcap C_2$ in polynomial time in $|C_1| + |C_2|$.*

Proof. If $A \sqsubseteq C_1$ and $A \sqsubseteq C_2$ are right $\mathcal{O}$-essential then one can show following the same lines of Lemma 30 of (Konev et al. 2014) that $A \sqsubseteq C_1 \sqcap C_2$ is concept saturated and decomposed on the right. The only rule which can be applicable is sibling merging. To construct the desired $A \sqsubseteq C$, one has to exhaustively apply sibling merging. Again following the same lines of Lemma 2 of (Konev et al. 2014), one can show that the resulting inclusion is concept saturated and decomposed on the right after applying sibling merging. Again the main observation here is that this lemma holds for $\mathcal{EL}$ terminologies, which may have concept inclusions of the form $C \sqsubseteq A$. As we already mentioned, such inclusions do not interfere in the argument since concept saturation ensures that nodes have all concept names implied by $\mathcal{O}$ in their labels.