Finding Proofs for Description Logic Entailments in Practice
(Extended Abstract)*

Christian Alrabbaa, Stefan Borgwardt,
Patrick Koopmann, and Alisa Kovtunova

Institute of Theoretical Computer Science, TU Dresden, Germany

Abstract. Approaches in knowledge representation based on formal logics and rule-based formalisms have the advantage that each step of reasoning can be understood by a user. However, often many of these inference steps are necessary, leading to a proof of the final conclusion that is hard to understand. If the set of inference rules is small, proofs may also look repetitious. Implemented systems (including the description logic (DL) reasoner ELK) that are able to produce formal proofs, are often bound to such predefined sets.
In this work we present a black-box approach to generate proofs for expressive description logics (DLs) that are not based on a set of inference rules. Instead, it exploits the non-standard inference *forgetting* to generate intermediate proof steps. We have evaluated this approach on a set of realistic ontologies and compared the proofs obtained using the existing forgetting tools LETHE and FAME with ones generated by ELK.

1 Introduction

Consequence-based (CB) reasoning for description logics [3] remains an active field of research after more than a decade [2]. The various CB calculi that have been developed and implemented for lightweight and expressive DL were recently surveyed in [5]. CB reasoning has many benefits: it derives formulae entailed by the ontology and these formulae can be organised in a formal proof represented as a labeled, directed hypergraph whose hyperedges correspond to single sound derivation steps [1]. Because of a nice goal-oriented behaviour, it is incorporated in leading DL reasoners, e.g. ELK [8, 9], Snorocket [14], and Konclude [20]. One can see the resulting formal proofs as good candidates for an explanation of an entailment. However, proofs still may be very large and repetitious, and thus inappropriate for immediate human consumption: it may be hard to comprehend why the overall entailment holds even if each single derivation step is easy to follow. Therefore, the research on how to create shorter, better understandable proofs by pruning away unimportant parts [13, 17] or extending sets of rules [4, 15, 18] is still ongoing.

* This is an abstract of the paper [1] published in full at LPAR-23.

In the spirit of justification oriented proofs [7] and concept interpolation [19], we present an orthogonal approach that produces proofs which are not based on a set of inference rules. The depth of such proofs is bounded by the size of the signature of the ontology. The new approach incorporates a forgetting tool and a reasoner in a black-box fashion. We compare the proofs obtained using the existing forgetting tools LETHE and FAME with proofs generated by the DL reasoner ELK for entailment tasks from the ORE benchmark ontologies.

2 Forgetting-Based Proofs

We assume a basic familiarity with description logics [3]. We first explain the idea on an arbitrary DL, where we focus on the terminological part, i.e., the TBox. Later, we instantiate this assumption for the implementation.

The main idea comes from the observation that the premises of a proof usually contain more symbols than the conclusion. For example, in the proof $\frac{A \sqsubseteq C \quad C \sqsubseteq B}{A \sqsubseteq B}$ the concept name C is eliminated by the inference step. Hence, the problem of finding a proof looks like a forgetting problem [11, 12], where the symbols that do not occur in the conclusion should be removed. In the example, $\{A \sqsubseteq B\}$ can be seen as the result of forgetting C in the original TBox $\{A \sqsubseteq C,\ C \sqsubseteq B\}$. More generally, a *forgetting-based proof* is composed of multiple steps that correspond to forgetting single symbols.

Definition 1. *Given a TBox $\mathcal{T}$ and a concept or role name X, the result of* forgetting X in $\mathcal{T}$ *is another TBox $\mathcal{T}'$ such that $\mathsf{sig}(\mathcal{T}') \subseteq \mathsf{sig}(\mathcal{T}) \setminus \{X\}$ and for every sentence α with $\mathsf{sig}(\alpha) \subseteq \mathsf{sig}(\mathcal{T}) \setminus \{X\}$ we have $\mathcal{T}' \models \alpha$ iff $\mathcal{T} \models \alpha$.*

The result of forgetting might not always exist in typical description logics, and may require the use of additional constructs such as fixpoint operators [11,12,21].

Our *forgetting-based approach (FBA)* for constructing proofs can be summarized as follows. Given a TBox $\mathcal{T}$ and a sentence η that is of the form $A \sqsubseteq B$ or $A \equiv B$, the goal is to forget all symbols except A and B from $\mathcal{T}$. Since rarely the whole TBox is needed for a proof, we first extract a justification $\mathcal{J} \subseteq \mathcal{T}$ for η, which already gives us the premises of the final proof. We then construct a sequence of TBoxes $\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_n$, where $\mathcal{T}_0 = \mathcal{J}$ and each $\mathcal{T}_i$, $1 \leq i \leq n$, is the result of forgetting one symbol from $\mathcal{T}_{i-1}$ (except A or B), and then extracting a justification for η. This process finishes when $\mathsf{sig}(\mathcal{T}_n) \subseteq \{A, B\}$. If one forgetting step fails, we try to forget a different symbol instead. If this fails for all symbols in the current signature, the process terminates early and we set $\mathcal{T}_n := \{\eta\}$.

To reconstruct the actual proof (represented as a labeled, directed hypergraph whose hyperedges correspond to single sound derivation steps) from these TBoxes, we start with a vertex labeled with η and select a justification $\mathcal{J}_\eta$ for η in $\mathcal{T}_{n-1}$, which gives us a first proof step that derives η from $\mathcal{J}_\eta$. We add a new vertex for each element of $\mathcal{J}_\eta$, and a hyperedge connecting these vertices to the sink. We then recursively justify each element of $\mathcal{J}_\eta$ in $\mathcal{T}_{n-2}$, and continue in this manner until we reach the sentences from the original TBox in $\mathcal{T}_0 = \mathcal{J} \subseteq \mathcal{T}$.

Each justification step for an intermediate sentence then corresponds to one inference step in the proof, which can be further annotated by the symbol that was forgotten in the corresponding forgetting operation.

Note that the precise result of FBA depends on the choices of justifications in each step as well as the order in which symbols are forgotten. However, regardless of these choices, FBA is sound and complete [1] if it employs a sound and complete reasoner to compute justifications and a sound forgetting tool.

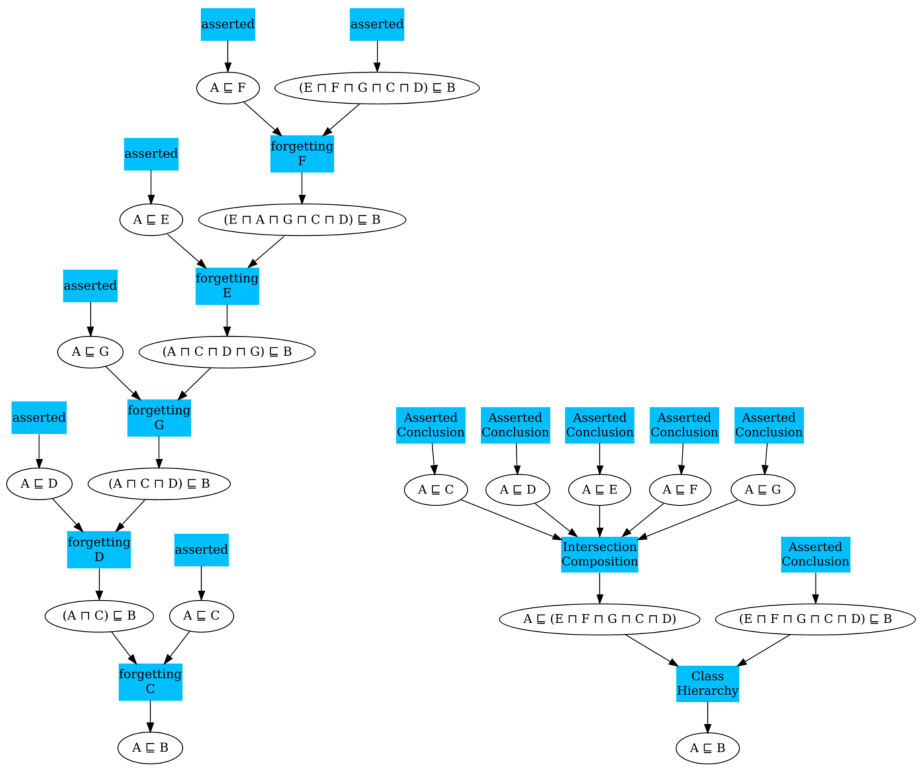

Fig. 1: A proof generated by Elk (right) vs. one for the same entailment generated via FBA (left; using either Lethe or Fame). Using the n-ary “Intersection Composition” rule, Elk allows to formulate a cleaner proof, while the iterative forgetting approach uses separate steps to eliminate each symbol.

To evaluate the approach, we used a dataset of entailment tasks extracted from the 2015 OWL Reasoner Evaluation (ORE) competition [16]. We generate proofs using both Elk and FBA, with either Lethe [10] or Fame [21] as black-box forgetters[1] (and HermiT [6] as reasoner). Of Lethe, we used the latest

[1] Both Lethe and Fame may generate fixpoint expressions that are not expressible in usual DLs. We omit any such steps from the generated proofs.

version 0.6,[2] and for Fame, we used the $\mathcal{ALCOI}$-variant of Fame 1.0 available on their website,[3] since other versions often fail to produce forgetting results that can be handled by the OWL API. Apart from the supported DL, a major difference between Lethe and Fame is that Lethe is theoretically guaranteed to compute a sound and complete forgetting result, while Fame is incomplete and may fail to forget a given name. Note that Lethe supports $\mathcal{ALCH}$ and Fame supports $\mathcal{ALCOI}$, but we only considered $\mathcal{ELH}$ reasoning tasks since we wanted to compare with Elk.

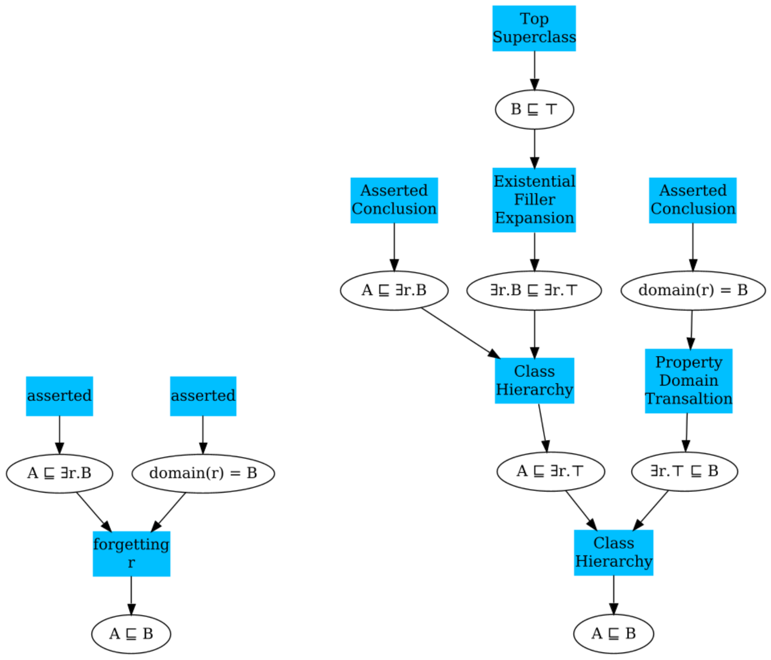

Fig. 2: The Elk proof (right) makes many inference steps, which are summarized by Lethe and Fame (left) into a single step. Assuming basic knowledge about description logics, the latter is arguably preferable.

The images in Figures 1–2 were generated automatically, and use blue boxes to denote inference steps, with incoming and outgoing arrows indicating premises and conclusions, respectively, and a label indicating the type of inference rule (for Elk) or the forgotten symbol(s) (for FBA). Figure 1 depicts a case where the Elk proofs are arguably better than the ones generated by FBA, while Figure 2 illustrates advantages of FBA. Note that we chose quite small proofs as examples, because larger ones would not easily fit on these pages.

[2] `https://lat.inf.tu-dresden.de/~koopmann/LETHE`
[3] `http://www.cs.man.ac.uk/~schmidt/sf-fame`

3 Discussion

We note that proofs generated using LETHE and FAME have the advantage that they can use inference steps in more expressive logics than $\mathcal{ELH}$, which may result in more compact proofs. However, understanding inference steps formulated in $\mathcal{ALCH}$ or $\mathcal{ALCOI}$ is inherently harder than for pure $\mathcal{ELH}$ proofs.

The main advantage of FBA is that it works in a black-box fashion, using any forgetting tool and reasoner to compute justifications (as long as they support the chosen DL). This means that we can find proofs for inferences in expressive logics like $\mathcal{ALCH}$ or $\mathcal{ALCOI}$, for which no consequence-based proof generators exist, using forgetting tools such as LETHE and FAME.

Acknowledgements This work was partially supported by the DFG in grant 389792660 as part of TRR 248 (`https://perspicuous-computing.science`), and QuantLA, GRK 1763 (`https://lat.inf.tu-dresden.de/quantla`).

References

1. Christian Alrabbaa, Franz Baader, Stefan Borgwardt, Patrick Koopmann, and Alisa Kovtunova. Finding small proofs for description logic entailments: Theory and practice. In Elvira Albert and Laura Kovacs, editors, *LPAR23. LPAR-23: 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 73 of *EPiC Series in Computing*, pages 32–67. EasyChair, 2020. URL: `https://easychair.org/publications/paper/qgX6`, `doi:10.29007/nhpp`.
2. Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05)*, pages 364–369. Professional Book Center, 2005. URL: `http://ijcai.org/Proceedings/09/Papers/053.pdf`.
3. Franz Baader, Ian Horrocks, Carsten Lutz, and Uli Sattler. *An Introduction to Description Logic.* Cambridge University Press, 2017. `doi:10.1017/9781139025355`.
4. Alexander Borgida, Enrico Franconi, and Ian Horrocks. Explaining $\mathcal{ALC}$ subsumption. In *ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, August 20-25, 2000*, pages 209–213, 2000. URL: `http://www.frontiersinai.com/ecai/ecai2000/pdf/p0209.pdf`.
5. David Tena Cucala, Bernardo Cuenca Grau, and Ian Horrocks. 15 years of consequence-based reasoning. In *Description Logic, Theory Combination, and All That - Essays Dedicated to Franz Baader on the Occasion of His 60th Birthday*, pages 573–587, 2019. `doi:10.1007/978-3-030-22102-7_27`.
6. Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: an OWL 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.
7. Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Justification oriented proofs in OWL. In *The Semantic Web - ISWC 2010 - 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part I*, pages 354–369, 2010. `doi:10.1007/978-3-642-17746-0_23`.
8. Yevgeny Kazakov, Pavel Klinov, and Alexander Stupnikov. Towards reusable explanation services in protege. In Alessandro Artale, Birte Glimm, and Roman Kontchakov, editors, *Proc. of the 30th Int. Workshop on Description Logics (DL'17)*, volume 1879 of *CEUR Workshop Proceedings*, 2017. URL: `http://www.ceur-ws.org/Vol-1879/paper31.pdf`.

9. Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. The incredible ELK – from polynomial procedures to efficient reasoning with $\mathcal{EL}$ ontologies. *J. Autom. Reasoning*, 53(1):1–61, 2014. `doi:10.1007/s10817-013-9296-3`.
10. Patrick Koopmann. LETHE: Forgetting and uniform interpolation for expressive description logics. *KI - Künstliche Intelligenz*, 2020. `doi:10.1007/s13218-020-00655-w`.
11. Patrick Koopmann and Renate A. Schmidt. Forgetting concept and role symbols in $\mathcal{ALCH}$-ontologies. In *Logic for Programming, Artificial Intelligence, and Reasoning - LPAR-19*, volume 8312 of *LNCS*, pages 552–567. Springer, 2013. `doi:10.1007/978-3-642-45221-5_37`.
12. Carsten Lutz and Frank Wolter. Foundations for uniform interpolation and forgetting in expressive description logics. In *Proceedings of IJCAI 2011*, pages 989–995. IJCAI/AAAI, 2011. `doi:10.5591/978-1-57735-516-8/IJCAI11-170`.
13. Deborah L. McGuinness. *Explaining Reasoning in Description Logics*. PhD thesis, Rutgers University, NJ, USA, 1996. `doi:10.7282/t3-q0c6-5305`.
14. Alejandro Metke-Jimenez and Michael Lawley. Snorocket 2.0: Concrete domains and concurrent classification. In Samantha Bail, Birte Glimm, Rafael S. Gonçalves, Ernesto Jiménez-Ruiz, Yevgeny Kazakov, Nicolas Matentzoglu, and Bijan Parsia, editors, *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, volume 1015 of *CEUR Workshop Proceedings*, pages 32–38. CEUR-WS.org, 2013. URL: `http://ceur-ws.org/Vol-1015/paper_3.pdf`.
15. Tu Anh Thi Nguyen, Richard Power, Paul Piwek, and Sandra Williams. Measuring the understandability of deduction rules for OWL. In *Proceedings of the First International Workshop on Debugging Ontologies and Ontology Mappings, WoDOOM 2012, Galway, Ireland, October 8, 2012.*, pages 1–12, 2012. URL: `http://www.ida.liu.se/~patla/conferences/WoDOOM12/papers/paper4.pdf`.
16. Bijan Parsia, Nicolas Matentzoglu, Rafael S. Gonçalves, Birte Glimm, and Andreas Steigmiller. The OWL Reasoner Evaluation (ORE) 2015 competition report. *J. Autom. Reasoning*, 59(4):455–482, 2017. `doi:10.1007/s10817-017-9406-8`.
17. Marvin R. G. Schiller and Birte Glimm. Towards explicative inference for OWL. In *Informal Proceedings of the 26th International Workshop on Description Logics, Ulm, Germany, July 23 - 26, 2013*, pages 930–941, 2013. URL: `http://ceur-ws.org/Vol-1014/paper_36.pdf`.
18. Marvin R. G. Schiller, Florian Schiller, and Birte Glimm. Testing the adequacy of automated explanations of EL subsumptions. In *Proceedings of the 30th International Workshop on Description Logics, Montpellier, France, July 18-21, 2017.*, 2017. URL: `http://ceur-ws.org/Vol-1879/paper43.pdf`.
19. Stefan Schlobach. Explaining subsumption by optimal interpolation. In José Júlio Alferes and João A. Leite, editors, *Proc. of the 9th Eur. Conf. on Logics in Artificial Intelligence (JELIA'04)*, volume 3229 of *Lecture Notes in Computer Science*, pages 413–425. Springer-Verlag, 2004. `doi:10.1007/978-3-540-30227-8_35`.
20. Andreas Steigmiller and Birte Glimm. Pay-as-you-go description logic reasoning by coupling tableau and saturation procedures. *Journal of Artificial Intelligence Research*, 54:535–592, 2015. `doi:10.1613/jair.4897`.
21. Yizheng Zhao and Renate A. Schmidt. FAME: an automated tool for semantic forgetting in expressive description logics. In *Automated Reasoning - 9th International Joint Conference, IJCAR 2018*, volume 10900 of *LNCS*, pages 19–27. Springer, 2018. `doi:10.1007/978-3-319-94205-6_2`.