# Explaining Description Logic Entailments in Practice with Evee and Evonne

Christian Alrabbaa[1], Stefan Borgwardt[1], Tom Friese[1], Patrick Koopmann[1], Julián Méndez[2], and Alexej Popovič[1]

[1] Institute of Theoretical Computer Science, TU Dresden, Germany
[2] Interactive Media Lab Dresden, TU Dresden, Germany
firstname.lastname@tu-dresden.de, julian.mendez2@tu-dresden.de

**Keywords:** Description Logics · OWL · Proofs · Protégé · Visualization · Uniform Interpolation

## 1 Introduction

Description logics (DLs) are a family of languages for specifying terminological knowledge in the form of *ontologies*, which are used in many areas such as biology, medicine, and the semantic web [7]. A central use case for DL ontologies is to derive implicit information using a description logic reasoner, for example to obtain a subsumption hierarchy of the terms specified in the ontology. Since the entailments derived by a reasoner are not always obvious, explanation services are necessary for ontology engineers and users. Most research on explaining DL entailments in the past focused on determining *which* axioms of the ontology are responsible for creating the entailment: this is done using *axiom pinpointing*, which computes a set of *justifications*—minimal subsets of the ontology that are sufficient for creating the entailment [16]. However, *how* the axioms in a justification produce the entailment is not always obvious. *Proofs* are a way to explain this, by connecting the axioms of the justification through a series of inference steps to the entailment to be explained (see Figures 1 and 2). In a series of papers, we investigated proofs for description logic entailments theoretically [2,3], practically [2,1,4] and cognitively [5,4]. However, proofs only help users if accessed through a user interface.

The widely-used ontology editor Protégé [15] includes a built-in service to explain inferences using justifications. There exists a plugin to show proofs generated using the description logic reasoner Elk [10,11], which already brings a great advantage over the standard explanation service of Protégé. However, this plugin is limited to DLs that Elk supports (focused on the OWL 2 EL profile), and can sometimes generate very large proofs.

For more explainable DL reasoning, we developed Evee-libs,[3] a Java library implementing different proof generation methods, which is used by two user interfaces: 1) Evee-protege is a collection of plugins for Protégé for the different proof generation methods, using the infrastructure also used by the

[3] https://github.com/de-tu-dresden-inf-lat/evee, version 0.1

Elk proofs plugin, and thus gives users a familiar visualization of proofs that is well-integrated into Protégé; 2) Evonne[4] is a more advanced proof exploration tool that comes as a standalone application.

## 2 Proof Generation with Evee-libs

Evee-libs implements methods for three different types of proofs, and a technique for condensing proofs based on a signature.

**Optimized Elk proofs** are, as with the existing proof plugin, generated using Elk, but optimized based on different, user-definable, optimality criteria. We implemented the Dijkstra-based method for finding proofs of minimal size presented in [3], which finds a proof that is minimal according to a *proof measure*. Such a proof measure associates to every proof a rational number that needs to be minimized. The algorithm requires the proof measure to be *recursive* (see [3] for a formal definition), which intuitively means that measure can be defined by induction on the tree unraveling of the proof. Evee-libs comes with three such measures predefined: 1) tree proof size, the number of nodes in the tree unraveling of the proof, 2) weighted tree proof size, weighting each node with the size of the axiom on that node, and 3) proof depth, the length of the longest path from the root to a leaf.

**Elimination proofs** are generated using the *forgetting-based approach* first presented in [2]. Such proofs do not rely on a given calculus of rules, and can be defined independently of any logic. Specifically, inferences in an elimination proof have the following form:

$$\frac{\alpha_1 \qquad \dots \qquad \alpha_n}{\beta}\ \text{eliminate } X_1, \dots, X_n$$

where $X_1, \dots, X_n$ are predicate (concept or role) names, $\{\alpha_1, \dots, \alpha_n\} \models \beta$, no subset of $\{\alpha_1, \dots, \alpha_n\}$ entails $\beta$, and $X_1, \dots, X_n$ do not occur in $\beta$. An example of an elimination proof is shown in Figure 1. Elimination proofs are computed using an external library for *uniform interpolation*, in our case either Lethe [12] or Fame [18]. Other implementations can be used via an interface that is part of the library. The logic supported by this method only depends on the uniform interpolation tool: in our case, we support $\mathcal{ALCH}$ via Lethe, and $\mathcal{ALC}$ via Fame. Evee-libs implements three approaches for computing elimination proofs 1) the first method uses a heuristic to determine in which order names are eliminated, 2) the second method optimizes for the number of names eliminated overall, and 3) the third method optimizes the (weighted) tree size of the resulting proof.

**Detailed Lethe-Based Proofs** provide more detailed proofs by tracking the actual inferences performed by Lethe when deriving the entailment to be explained. In contrast to the elimination proofs, which give a more high-level perspective, the detailed proofs give longer proofs with smaller proof steps, which can be helpful in case the elimination proof contains inferences that are not immediately easy to understand.

[4] `https://imld.de/evonne`

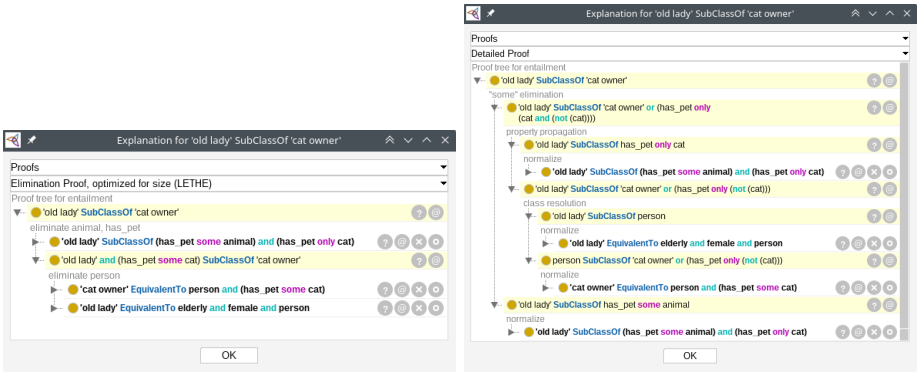


**Fig. 1.** Elimination proof (left) and detailed Lethe-based proof (right) for an entailment in the People ontology,[6] as displayed with Evee-protege.

**Signature-Based Proof Condensation** For the first 2 methods, it is possible to additionally supply a signature of *known terms*. Proofs are then generated in such a way that all entailed axioms that only use those terms are considered "known" and are shown without a sub-proof [1].

Details on the different proof generation methods can be found in [2,1,4], where we also provide a quantitative evaluation of the proofs generated by the different proof methods. In [4], we also performed a small qualitative user study to get an idea of user preferences, which confirmed that, depending on the user type and the situation, different proof generation methods may be preferred, which is why we let users choose between the different techniques.

## 3 Proof Exploration with Evee-protege and Evonne

For users of the ontology IDE Protégé, the easiest way to use our proof generation methods is by putting the plugins of Evee-protege [4] available on the website into the plugin folder of Protégé. Proofs can then be accessed via the question mark that is shown in Protégé next to any subsumption relation that has been computed by the reasoner. After clicking the question mark, users can choose between being shown a justification or a proof generated by one of the supported methods. Some of the proof generation methods (especially for the optimized elimination proofs) can take some time, in which case a progress window is shown to the user. If the computation takes too long, the user can cancel the computation, and is then shown the best proof found so far, provided at least one could be computed. With the Protégé proof visualization, inferences can be unfolded step-wise until the full proof is shown; see Figure 1.

More advanced proof exploration is possible with Evonne. Evonne can be tried online on the web page, and locally using the available Docker images.

[6] `http://owl.man.ac.uk/2006/07/sssw/people.owl`

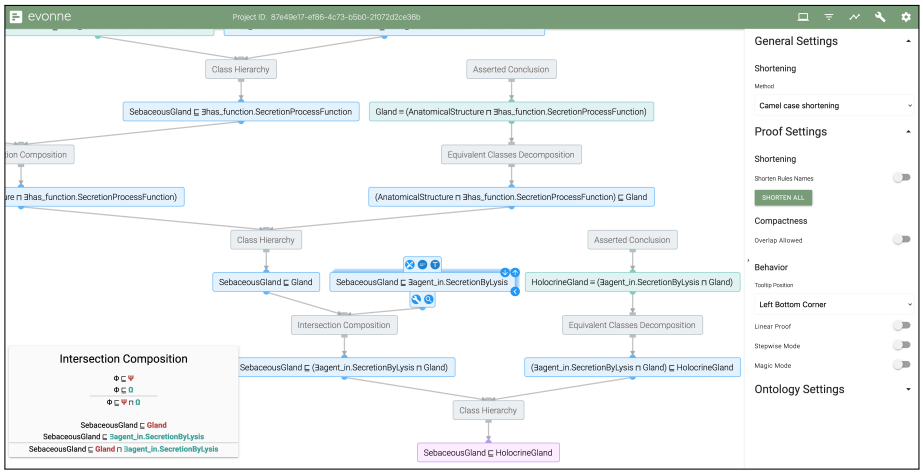


**Fig. 2.** Evonne showing a proof from the Skin Physiology Ontology (SPO).[7]

Upon starting, users specify the ontology and select a concept inclusion they would like to have explained, as well as the proof generation method to use, and optionally a signature file if they want to see a signature-condensed proof. They are then greeted with an interactive visualization of the proof next to a visual representation of a relevant part of the ontology. When turning off the ontology view, we see a screen as in Fig. 2. The proof visualization has a range of layout and interaction components. For example, sub-proofs can be hidden, concepts abbreviated, axioms formulated in natural language, and there is a "magic" mode that allows exploring the proof in a bi-directional manner, starting from both the conclusion and the justification axioms. An early prototype of Evonne is described in [8], and some information on the proof exploration with Evonne can be found in [1]. A detailed publication is currently under preparation.

## 4 Outlook

We plan to extend Evee-libs to also generate proofs based on the calculus for classification in $\mathcal{ALCH}$ presented in [17], and to compare all methods in a more in-depth user study. With Evee and Evonne being so far tailored towards explaining entailments, in the future we also want to support explanations of *missing* entailments, for which we have developed methods based on counterinterpretations [6] and abduction [13,9,14], but no graphical user interfaces yet.

## 5 Acknowledgments

This work was supported by DFG grant 389792660 as part of TRR 248 – CPEC, and the DFG Research Training Group QuantLA, GRK 1763.

[7] `https://bioportal.bioontology.org/ontologies/SPO`

## References


1. Alrabbaa, C., Baader, F., Borgwardt, S., Dachselt, R., Koopmann, P., Méndez, J.: Evonne: Interactive proof visualization for description logics (system description). In: Proc. IJCAR 2022. LNCS, Springer (2022), https://lat.inf.tu-dresden.de/research/papers/2022/ALBABODAKOME-IJCAR22.pdf, to appear
2. Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: Finding small proofs for description logic entailments: Theory and practice. In: Proc. LPAR 2020. EasyChair (2020). https://doi.org/10.29007/nhpp
3. Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: Finding good proofs for description logic entailments using recursive quality measures. In: Proc. CADE 28. LNCS, vol. 12699, pp. 291–308. Springer (2021)
4. Alrabbaa, C., Borgwardt, S., Friese, T., Koopmann, P., Méndez, J., Popovič, A.: On the eve of true explainability for OWL ontologies: Description logic proofs with Evee and Evonne. In: Proc. DL'22. CEUR-WS.org (2022), https://arxiv.org/abs/2206.07711, to appear
5. Alrabbaa, C., Borgwardt, S., Knieriemen, N., Kovtunova, A., Rothermel, A.M., Wiehr, F.: In the hand of the beholder: Comparing interactive proof visualizations. In: Proc. DL 2021. vol. 2954. CEUR-WS.org (2021)
6. Alrabbaa, C., Hieke, W., Turhan, A.: Counter model transformation for explaining non-subsumption in EL. In: Proc. FCR'21. CEUR Workshop Proceedings, vol. 2961, pp. 9–22. CEUR-WS.org (2021)
7. Baader, F., Horrocks, I., Lutz, C., Sattler, U.: An Introduction to Description Logic. Cambridge University Press (2017)
8. Flemisch, T., Langner, R., Alrabbaa, C., Dachselt, R.: Towards designing a tool for understanding proofs in ontologies through combined node-link diagrams. In: Proc. VOILA'20. vol. 2778, pp. 28–40. CEUR-WS.org (2020)
9. Haifani, F., Koopmann, P., Tourret, S., Weidenbach, C.: Connection-minimal abduction in $\mathcal{EL}$ via translation to FOL. In: Proc. IJCAR 2022. Lecture Notes in Computer Science, Springer (2022), to appear.
10. Kazakov, Y., Klinov, P., Stupnikov, A.: Towards reusable explanation services in protege. In: Proc. DL'17. vol. 1879. CEUR-WS.org (2017)
11. Kazakov, Y., Krötzsch, M., Simancik, F.: The incredible ELK - From polynomial procedures to efficient reasoning with $\mathcal{EL}$ ontologies. J. Autom. Reason. **53**(1), 1–61 (2014). https://doi.org/10.1007/s10817-013-9296-3
12. Koopmann, P.: LETHE: Forgetting and uniform interpolation for expressive description logics. Künstliche Intell. **34**(3), 381–387 (2020)
13. Koopmann, P.: Signature-based abduction with fresh individuals and complex concepts for description logics. In: Proc. IJCAI 2021. pp. 1929–1935. ijcai.org (2021). https://doi.org/10.24963/ijcai.2021/266
14. Koopmann, P., Del-Pinto, W., Tourret, S., Schmidt, R.A.: Signature-based abduction for expressive description logics. In: Proc. KR 2020. pp. 592–602 (2020). https://doi.org/10.24963/kr.2020/59
15. Musen, M.A.: The Protégé project: A look back and a look forward. AI Matters **1**(4), 4–12 (2015). https://doi.org/10.1145/2757001.2757003
16. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proc. IJCAI-03. Morgan Kaufmann (2003)
17. Simancik, F., Kazakov, Y., Horrocks, I.: Consequence-based reasoning beyond Horn ontologies. In: Proc. IJCAI 2011. pp. 1093–1098. IJCAI/AAAI (2011)

18. Zhao, Y., Schmidt, R.A.: FAME: An automated tool for semantic forgetting in expressive description logics. In: Proc. IJCAR 2018. LNCS, vol. 10900, pp. 19–27. Springer (2018)