Two Ways of Explaining Negative Entailments in Description Logics Using Abduction *

Patrick Koopmann

Institute for Theoretical Computer Science, Technische Universität Dresden, Germany

Abstract. We discuss two ways of using abduction to explain missing entailments from description logic knowledge bases, one more common, one more unusual, and then have a closer look at how current results/implementations on abduction could be used towards generating such explanations, and what still needs to be done.

1 Introduction

The importance of services to explain inferences performed by description logic (DL) reasoners is long understood. In fact, understanding and debugging large DL ontologies such as SNOMED CT (350,000 axioms) would be much more challenging as it is now without the service of justifications [5,23,12] explaining reasoner results and their direct support through the standard ontology editor Protégé [13]. Recently, explaining positive entailments (things that logically follow from the ontology) has been further improved by the possibility of showing proofs created by the $\mathcal{EL}$ reasoner ELK [14]. How to explain ontology entailments using proofs, and how to select good proofs, has since then been the subject of further research [1,2]. But what is the situation for explaining negative entailments, that is, logical consequences that do not follow from an ontology? Here, usually two solutions are suggested: showing a counter example in form of a DL interpretation [6], or using abduction [8,9]. However, literature on abduction in DLs often just takes this motivation as given, and then goes on to provide complexity results or methods, without further elaborating *how* this can help. In particular, many approaches for abduction look at first sight more well-suited for the task of *diagnosis*: providing a plausible explanation for some observation that does not logically follow from the known facts [9,20], which is indeed the traditional motivation for abduction [21]. In this abstract, we want to clarify this a bit by elaborating, based on a simple example, on two possible ways in which abduction can help, possibly in combination with proof services, to explain missing entailments from a DL knowledge base. In particular, we argue also for the use of abduction for the generation of counter examples. We then review some recent results under the light of those explanation services, highlighting both possible extensions and open challenges towards this direction. Towards the end, the paper makes some claims that have not been discussed in the literature yet. The interested reader can find detailed arguments for these in the accompanying technical report [16].

* This work was supported by the DFG in grant 389792660 as part of TRR 248.

2 Preliminaries

Description Logics For detailed information on DLs and their semantics, we refer to [3], and only give the syntax and central notions of the classical DL $\mathcal{ALC}$. $\mathcal{ALC}$ concepts are built from countably infinite sets $\mathsf{N_C}$ and $\mathsf{N_R}$ of respectively *concept* and *role names* according to the following syntax rule, where $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$:

$$C ::= A \mid \neg C \mid C \sqcup C \mid C \sqcap C \mid \exists r.C \mid \forall r.C$$

A TBox is a set of axioms of the form $C \sqsubseteq D$ and $C \equiv D$, where C and D are concepts, and an ABox is a set of axioms of the form $C(a)$ and $r(a, b)$, where C is a concept, $r \in \mathsf{N_R}$, and a and b are picked from a countably infinite set $\mathsf{N_I}$ of *individual names*. We call an ABox $\mathcal{A}$ *flat* if for every $C(a) \in \mathcal{A}$, C is a concept name. A *knowledge base* (KB) is a union of a TBox and an ABox, and thus also generalises both notions. We say that a KB $\mathcal{K}$ *entails* an axiom α, in symbols $\mathcal{K} \models \alpha$, if every model $\mathcal{I}$ of $\mathcal{K}$ is also a model of α.

Fixpoint Operators We also briefly recall the less classical DL $\mathcal{ALC}^\mu$, which extends $\mathcal{ALC}$ with *least fixpoint* concepts of the form $\mu X.C[X]$, where $C[X]$ is a concept in which X occurs like a concept name, but only under an even number of negation symbols [17]. Given such a concept $C[X]$ and another concept D, we denote by $C[D]$ the result of replacing X in $C[X]$ by D. Intuitively, the fixpoint concept $\mu X.C[X]$ corresponds to the infinite disjunction $C[\bot] \sqcup C[C[\bot]] \sqcup C[C[C[\bot]]] \sqcup \ldots$ [7].

Abduction We call a subset $\Sigma \subseteq \mathsf{N_C} \cup \mathsf{N_R}$ a *signature*, and denote by $\mathsf{sig}(\mathcal{K})$ the signature that consists of all concept and role names that occur in the KB $\mathcal{K}$. We focus on the following notion of abduction.

Definition 1. *Let $\mathcal{L}$ be a DL. A* signature-based $\mathcal{L}$ abduction problem *is given by a triple $\mathfrak{A} = \langle \mathcal{K}, \Phi, \Sigma \rangle$, with an $\mathcal{L}$ KB $\mathcal{K}$ of* background knowledge*, an $\mathcal{L}$ KB Φ as* observation*, and a signature $\Sigma \subseteq \mathsf{N_C} \cup \mathsf{N_R}$ of* abducibles*; and asks whether there exists a* hypothesis for $\mathfrak{A}$*, i.e. an $\mathcal{L}$ KB $\mathcal{H}$ satisfying*

$$\textbf{\textit{A1}}\ \ \mathcal{K} \cup \mathcal{H} \not\models \bot, \quad \textbf{\textit{A2}}\ \ \mathcal{K} \cup \mathcal{H} \models \Phi, \textit{ and } \quad \textbf{\textit{A3}}\ \ \mathsf{sig}(\mathcal{H}) \subseteq \Sigma.$$

If Φ and $\mathcal{H}$ are additionally required to be TBoxes/ABoxes/flat ABoxes, we speak of a TBox/ABox/flat ABox abduction problem.

Because of **A3**, this kind of abduction is called *signature-based abduction* [15,17,22,8]. Other variants of abduction usually use a different condition instead to avoid trivial answers: for instance that $\mathcal{H}$ contains axioms picked from a predefined set of *abducible axioms* [9,22], has a specified shape [10], or satisfies a relevance criterion [11].

3 Motivating and Introducing Type 1 and Type 2 Explanations

We illustrate our idea with a simplified toy example on the pizza ontology,[1] where the aim is to explain a missing entailed TBox axiom. Our notions of explanations generalise

[1] https://protege.stanford.edu/ontologies/pizza/pizza.owl

to other settings such as in explaining missing ABox inferences or query answers [8]. The pizza ontology is a toy ontology that is commonly used in tutorials on the OWL ontology language, and it provides definitions of different pizza types (Margherita, SalamiPizza), categories of pizzas (VegetarianPizza, SpicyPizza), together with various types of pizza ingredients and their properties. Specifically, it provides the following definition of VegetarianPizza:

$$\mathsf{VegetarianPizza} \equiv \mathsf{Pizza} \sqcap \forall \mathsf{hasTopping}.\mathsf{VegetarianTopping}$$

Assume an ontology engineer wants to extend the pizza ontology with a definition of the Pizza Marinara, for which they would use the following axioms:

$$\begin{aligned}\mathsf{PizzaMarinara} \equiv \mathsf{Pizza} &\sqcap \exists \mathsf{hasTopping}.\mathsf{Tomato} \\ &\sqcap \exists \mathsf{hasTopping}.\mathsf{Oregano} \\ &\sqcap \exists \mathsf{hasTopping}.\mathsf{Garlic}\end{aligned}$$

An unexperienced ontology engineer might wonder, why their PizzaMarinara is not classified as VegetarianPizza, even though they only listed ingredients that are subsumed by VegetarianTopping. Specifically, they would like to see an explanation for the *negative* entailment $\mathcal{T} \not\models \mathsf{PizzaMarinara} \sqsubseteq \mathsf{VegetarianPizza}$, and ask "Why is not every Pizza Marinara a vegetarian pizza?". This question may have two readings, depending on the context. One reading is more pragmatic: "What did I do wrong?/What do I have to change to create this entailment?". The other reading aims more at understanding the problem: "Could you show me a Pizza Marinara that is not vegetarian, and explain to me why it is not?".

To answer the pragmatic question, we could use TBox abduction to obtain a set of axioms $\mathcal{H}$ s.t. $\mathcal{K} \cup \mathcal{H} \models \mathsf{PizzaMarinara} \sqsubseteq \mathsf{VegetarianPizza}$. Of course, not any such set of axioms is equally helpful: ideally, those axioms would add to a definition of PizzaMarinara, that is, they should use terminology that one would use when specifying knowledge on a pizza, i.e. the pizza itself and its ingredients. We would thus employ signature-based abduction with the signature $\Sigma = \{\mathsf{PizzaMarinara}, \mathsf{hasTopping}, \mathsf{Tomato}, \mathsf{Oregano}, \mathsf{Garlic}\}$, to which the solution would be:

$$\mathcal{H}_1 = \{\ \mathsf{PizzaMarinara} \sqsubseteq \forall \mathsf{hasTopping}.(\mathsf{Tomato} \sqcup \mathsf{Oregano} \sqcup \mathsf{Garlic})\ \},$$

the so-called *closure-axiom* they forgot to add to its definition [24]. In the following, we call this pragmatic type of explanations *type 1 explanation*.

Definition 2 (Type 1 Explanation). *Let $\mathcal{K}$ and Φ be such that $\mathcal{K} \not\models \Phi$, and Σ a signature. A* type 1 explanation for $\mathcal{K} \not\models \Phi$ in Σ *is a KB $\mathcal{H}_1$ in Σ s.t. $\mathcal{K} \cup \mathcal{H}_1 \not\models \bot$ and $\mathcal{K} \cup \mathcal{H}_1 \models \Phi$.*

Note that this notion is equivalent to that of signature-based abduction hypothesis.

While this helps the engineer to fix their definition, it might be insufficient in helping them understand *why* the axiom is necessary. We could additionally provide a proof tree as in [1] to explain $\mathcal{T} \cup \mathcal{H}_1 \models \mathsf{MarinaraPizza} \sqsubseteq \mathsf{VegetarianPizza}$, and mark the points where the suggested axiom is necessary. However, still as an explanation it seems unsatisfying to just point at things that are not there rather than what is there.

Now consider the reading "Show me a Pizza Marinara that is not Vegetarian". This approach is usually answered by providing a counter model: a model of the knowledge base in which there is some domain element d that satisfies PizzaMarinara but not VegetarianPizza. The downside of this solution is however that such a model might contain a lot of information that is not related to the property of being vegetarian and only distracts from the real reason why the axiom is not entailed: for instance, the pizza ontology requires that every pizza is associated a base, and that every topping is associated a spiciness level. Spiciness levels and pizza base would thus need to be part of the counter model, but they would not contribute to the explanation. In more realistic examples, this is likely to become a serious problem. We could try to use a signature again to avoid this problem, for instance by filtering out elements that are not in a desired signature. However, we would then obtain something that is not even a model, and the relation to the TBox $\mathcal{T}$ might not be clear anymore. Note also that we cannot use proof trees in combination with interpretations, as proofs usually work on the level of axioms. Finally, the concept of interpretations might not be so familiar to the ontology engineer, as information is usually expressed in terms of axioms and assertions that are interpreted under the open world assumption.

A more natural approach is to instead show the engineer an ABox $\mathcal{H}_2$ consistent with the TBox, with an individual a s.t. $\mathcal{T} \cup \mathcal{H}_2 \models \mathsf{PizzaMarinara}(a)$ and $\mathcal{K} \cup \mathcal{H}_2 \models \neg\mathsf{VegetarianPizza}(a)$, and which does so by using terminology used for specifying pizzas. An example would be:

$$\{\ \mathsf{Pizza}(a),\ \mathsf{hasTopping}(a,b),\ \mathsf{Tomato}(b),\ \mathsf{hasTopping}(a,c),\ \mathsf{Oregano}(c),\ \mathsf{hasTopping}(a,d),\ \mathsf{Garlic}(d),\ \mathsf{hasTopping}(a,e),\ \mathsf{Chicken}(e)\ \}$$

We call this type of explanation *type 2 explanation*.

Definition 3 (Type 2 Explanation). *Let $\mathcal{K}$ and Φ be such that $\mathcal{K} \not\models \Phi$, and Σ a signature. A* type 2 explanation for $\mathcal{K} \not\models \Phi$ in Σ *is a KB $\mathcal{H}_2$ in Σ s.t. $\mathcal{K} \cup \mathcal{H}_2 \not\models \bot$ and for every model $\mathcal{I}$ of $\mathcal{K} \cup \mathcal{H}_2$, $\mathcal{I} \not\models \Phi$.*

Type 2 explanations can also be computed using abduction: here, we would use the observation $\{(\mathsf{PizzaMarinara} \sqcap \neg\mathsf{VegetarianPizza})(a)\}$. To help the engineer understand this explanation, we can again use a proof for $\mathcal{T} \cup \mathcal{H}_2 \models \Phi$. We can even use $\mathcal{H}_1$ in combination with the type 1 explanation, by providing a formal proof for $\mathcal{T} \cup \mathcal{H}_1 \cup \mathcal{H}_2 \models \bot$, thus illustrating in which way $\mathcal{H}_1$ avoids the counter example.

4 State-of-the-Art and Challenges

For both types of explanations, restricting abductive solutions using a signature is key to controlling its output to something useful. Most existing work on abduction either do not consider such a restriction, or they fix the set of axioms to be used, rather than the signature. In those cases, abduction boils down to finding a minimal subset from this set of abducibles that is sufficient for entailing the observation, something that could for instance be solved using axiom pinpointing [12]. The problem with this approach is that we need to know beforehand which axioms we are looking for: as we show below, the solution is otherwise simply too large. We focus in the following on the case of TBox entailments to be explained, as in the example.

4.1 Computing Explanations of Type 1

For explanations of Type 1, we need to perform signature-based TBox abduction, for which we presented a method for $\mathcal{ALC}$ in [17]. However, in the general case, this method does not compute a single $\mathcal{ALC}$ hypothesis for the given abduction, but it computes a hypothesis in form of a Boolean $\mathcal{ALCOI}^{\mu}$-KB that is general enough to cover every possible $\mathcal{ALC}$ hypothesis within the signature. If the observation consists only of TBox axioms, the result would be a Boolean combination Φ of $\mathcal{ALC}^{\mu}$ GCIs, which are combined using disjunction and conjunction, but without use of negation. From this, one can always extract an $\mathcal{ALC}$ hypothesis by dropping disjuncts and unfolding fixpoint expressions up to a certain depth, as shown in the extended version of this paper. However, we have not investigated yet how we can bound the depth of the unfolding.

The method for computing hypotheses in $\mathcal{ALCOI}^{\mu}$-KBs is based on uniform interpolation [18], and may thus produce solutions that are triple exponential in the size of the abduction problem [19]. While this may sound like bad news, such a blow-up is in general unavoidable: as illustrated in the extended version, it can be shown by a simple modification of a proof in [15] that there exists a family of $\mathcal{ALC}$ TBox abduction problems for which every hypothesis is of size triple exponential in the size of the abduction problem. On the positive side, the evaluation in [17] indicates that fixpoint operators are not often introduced into the solution. A more realistic investigation of this is future work.

4.2 Computing Explanations of Type 2

For explanations of type 2, we need to perform ABox abduction for an observation of the form $C(a)$. Moreover, as the previous example illustrates, hypotheses become more visual if they use fresh individual names, instead of encoding all the information into a single complex concept. Indeed, explanations that are flat ABoxes, or use complex concepts only where necessary, seem to be more convenient. The method from [17] is therefore not immediately useful, as it never computes a solution with fresh individual names. Signature-based ABox abduction for flat ABox hypotheses can be performed by the AAA ABox Abduction Solver [22], however, this tool does not introduce fresh individual names either. An approach could be to fix a sufficiently high number of individuals before running the solver. We have shown in [15] that the number of required individual names for flat ABox hypotheses can become exponential in the worst case, and if we allow for complex concepts, hypotheses may require axioms of triple exponential size. Furthermore, we have shown that deciding whether there exists a solution of size n, where n is given in binary, is NEXPTIME-complete for $\mathcal{EL}_{\bot}$ and NEXPTIME$^{\text{NP}}$-complete for $\mathcal{ALCI}$. Fixing the number of abducibles beforehand is thus not feasible in general. Therefore, a more directed way of generating fresh individual names is needed. We are currently investigating a method for ABox abduction in $\mathcal{EL}$ that ideas from a method for ABox repairs presented in [4], where broken erroneous entailments are repaired by introducing anonymous copies of existing individuals first. However, $\mathcal{EL}$ does not allow for negated concepts, which would be part of the abduction input if we want to compute explanations of Type 2. Consequently, the next challenge would be how to integrate more expressive DLs.

References

1. Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: Finding small proofs for description logic entailments: Theory and practice. In: Albert, E., Kovacs, L. (eds.) LPAR-23: 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning. EPiC Series in Computing, vol. 73, pp. 32–67. EasyChair (2020). https://doi.org/10.29007/nhpp
2. Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: Finding good proofs for description logic entailments using recursive quality measures. In: Platzer, A., Sutcliffe, G. (eds.) Automated Deduction - CADE 28 - 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12699, pp. 291–308. Springer (2021). https://doi.org/10.1007/978-3-030-79876-5_17
3. Baader, F., Horrocks, I., Lutz, C., Sattler, U.: An Introduction to Description Logic. Cambridge University Press (2017)
4. Baader, F., Koopmann, P., Kriegel, F., Nuradiansyah, A.: Computing optimal repairs of quantified aboxes w.r.t. static $\mathcal{EL}$ tboxes. In: Platzer, A., Sutcliffe, G. (eds.) Automated Deduction - CADE 28 - 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12699, pp. 309–326. Springer (2021). https://doi.org/10.1007/978-3-030-79876-5_18
5. Baader, F., Suntisrivaraporn, B.: Debugging SNOMED CT using axiom pinpointing in the description logic $\mathcal{EL}^+$. In: Proc. of the 3rd Conference on Knowledge Representation in Medicine (KR-MED'08): Representing and Sharing Knowledge Using SNOMED. CEUR-WS, vol. 410 (2008), http://ceur-ws.org/Vol-410/Paper01.pdf
6. Bauer, J., Sattler, U., Parsia, B.: Explaining by example: Model exploration for ontology comprehension. In: Grau, B.C., Horrocks, I., Motik, B., Sattler, U. (eds.) Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK, July 27-30, 2009. CEUR Workshop Proceedings, vol. 477. CEUR-WS.org (2009), http://ceur-ws.org/Vol-477/paper_37.pdf
7. Calvanese, D., De Giacomo, G., Lenzerini, M.: Reasoning in expressive description logics with fixpoints based on automata on infinite trees. In: Proceedings of IJCAI 99. pp. 84–89 (1999), http://ijcai.org/Proceedings/99-1/Papers/013.pdf
8. Calvanese, D., Ortiz, M., Simkus, M., Stefanoni, G.: Reasoning about explanations for negative query answers in DL-Lite. J. Artif. Intell. Res. **48**, 635–669 (2013). https://doi.org/10.1613/jair.3870
9. Ceylan, İ.İ., Lukasiewicz, T., Malizia, E., Molinaro, C., Vaicenavicius, A.: Explanations for negative query answers under existential rules. In: Calvanese, D., Erdem, E., Thielscher, M. (eds.) Proceedings of KR 2020. pp. 223–232. AAAI Press (2020). https://doi.org/10.24963/kr.2020/23
10. Du, J., Wan, H., Ma, H.: Practical TBox abduction based on justification patterns. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. pp. 1100–1106 (2017), http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14402
11. Elsenbroich, C., Kutz, O., Sattler, U.: A case for abductive reasoning over ontologies. In: Proceedings of the OWLED'06 Workshop on OWL: Experiences and Directions (2006), http://ceur-ws.org/Vol-216/submission_25.pdf
12. Horridge, M.: Justification based explanation in ontologies. Ph.D. thesis, University of Manchester, UK (2011), http://www.manchester.ac.uk/escholar/uk-ac-man-scw:131699
13. Horridge, M., Parsia, B., Sattler, U.: Explanation of OWL entailments in protege 4. In: Bizer, C., Joshi, A. (eds.) Proceedings of the Poster and Demonstration Session at the 7th International Semantic Web Conference (ISWC2008), Karlsruhe, Germany, October 28, 2008.

CEUR Workshop Proceedings, vol. 401. CEUR-WS.org (2008), http://ceur-ws.org/Vol-401/iswc2008pd_submission_47.pdf
14. Kazakov, Y., Klinov, P., Stupnikov, A.: Towards reusable explanation services in protege. In: Artale, A., Glimm, B., Kontchakov, R. (eds.) Proceedings of the 30th International Workshop on Description Logics, Montpellier, France, July 18-21, 2017. CEUR Workshop Proceedings, vol. 1879. CEUR-WS.org (2017), http://ceur-ws.org/Vol-1879/paper31.pdf
15. Koopmann, P.: Signature-based abduction with fresh individuals and complex concepts for description logics. In: Zhou, Z. (ed.) Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021. pp. 1929–1935. ijcai.org (2021). https://doi.org/10.24963/ijcai.2021/266
16. Koopmann, P.: Two ways of explaining negative entailments in description logics using abduction (extended version). LTCS-Report 21-03, Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany (2021)
17. Koopmann, P., Del-Pinto, W., Tourret, S., Schmidt, R.A.: Signature-based abduction for expressive description logics. In: Calvanese, D., Erdem, E., Thielscher, M. (eds.) Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020. pp. 592–602. AAAI Press (2020). https://doi.org/10.24963/kr.2020/59
18. Koopmann, P., Schmidt, R.A.: Uniform interpolation and forgetting for $\mathcal{ALC}$ ontologies with ABoxes. In: Bonet, B., Koenig, S. (eds.) Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA. pp. 175–181. AAAI Press (2015), http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9981
19. Lutz, C., Wolter, F.: Foundations for uniform interpolation and forgetting in expressive description logics. In: Walsh, T. (ed.) IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011. pp. 989–995. IJCAI/AAAI (2011). https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-170
20. Obeid, M., Obeid, Z., Moubaiddin, A., Obeid, N.: Using description logic and Abox abduction to capture medical diagnosis. In: Wotawa, F., Friedrich, G., Pill, I., Koitz-Hristov, R., Ali, M. (eds.) Advances and Trends in Artificial Intelligence. From Theory to Practice. pp. 376–388. Springer International Publishing, Cham (2019)
21. Peirce, C.S.: Deduction, induction, and hypothesis. Popular science monthly **13**, 470–482 (1878)
22. Pukancová, J., Homola, M.: The AAA ABox abduction solver. Künstliche Intell. **34**(4), 517–522 (2020). https://doi.org/10.1007/s13218-020-00685-4
23. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Gottlob, G., Walsh, T. (eds.) Proc. of the 18th Int. Joint Conf. on Artificial Intelligence (IJCAI 2003). pp. 355–362. Morgan Kaufmann, Acapulco, Mexico (2003)
24. Stevens, R.: Closing down the open world: Covering axioms and closure axioms. http://ontogenesis.knowledgeblog.org/1001 (2011), http://ontogenesis.knowledgeblog.org/1001