Explanation for defeasible entailment

Victoria Chama[1] and Thomas Meyer[1,2]

[1] Department of Computer Science, University of Cape Town
chmvic006@myuct.ac.za
[2] Center for Artificial Intelligence Research
tmeyer@cs.uct.ac.za

Description Logics (DLs) are well-known formalisms for reasoning about information in a given domain. DLs have many advantages such as being decidable fragments of First-Order Logic, and having a clear semantics and well-defined reasoning procedures which can be automated [7, 2]. Take the classic penguin example, and consider a knowledge base containing the statements: "penguins are birds", "robins are birds", "penguins do not fly", "birds fly" and "birds have wings".We can use the well-defined syntax and semantics of DLs to define entailment which allows us to derive implicit knowledge that can be made explicit through inferences [2]. For example using the information above we can query the knowledge base, ask "do robins have wings", and the answer would be YES.

DLs employ various reasoning services such as concept satisfiability, subsumption, consistency checking and instance checking which can be used to derive useful implicit information from knowledge bases. Reductions between reasoning services also enable only one reasoning procedure to be implemented which alleviates the need of creating tools to perform each and every reasoning service [9, 10]. Various reasoning techniques/algorithms have been developed to solve some of the reasoning problems highlighted above. The most widely used technique, the tableau-based approach, have been shown to be efficient in practice for real knowledge bases [2].

The DLs services mentioned above can be more useful by adding explanations to the conclusions that DLs systems can draw. Using the example above, the answer to our query "do robins have wings" was YES. However, it is more beneficial to users if the DL system can also provide an explanation of how it came to the conclusion. In this example an explanation to the query is that "we know that robins are birds, and birds have wings, therefore we can conclude that robins have wings". Explanation facilities are useful in understanding entailments, debugging and repairing information declared in knowledge bases and also knowledge base comprehension. In our example above our knowledge base is very small with only five statements. In reality knowledge bases can contain ten of thousand of statements and without automated support for explanation, it can be difficult to identify the statements that give rise to entailments [3, 6]. There are various algorithms to compute justifications, and implementations of these algorithms for the DL case are available through the ontology editor Protégé [6].

Classical DLs cannot deal with exceptional cases. For this reason, there have been numerous proposals to define non-monotonic reasoning systems. One such approach is the KLM approach to defeasible reasoning, which was originally

defined for propositional logic, but has been lifted to the case for DLs [5] . Rational closure is a specific method within the KLM approach that has a well-defined semantics [4] and an implementation [8] for the ontology editor Prot*égé*.

The rational closure algorithm performs reasoning by first constructing a ranking where every defeasible condition has a rank [4]. A defeasible conditional is exceptional if its antecedent is exceptional with respect to the knowledge base. In order to determine the rank of a statement, you first have to check how exceptional it is.

Let us extend the example above to the defeasible case. Suppose the knowledge base contains the following the statements: "penguins are birds", "robins are birds", "penguins do not fly", "birds typically fly" and "birds typically have wings".

When we query this knowledge base and ask "do penguins typically have wings", the answer to the query using rational closure is NO. The reason the system returns NO is not as straightforward as one might think. Initially, a user might use the same arguments that were used in our initial version of our example but that justification leads to errors. Thus, it is useful if defeasible reasoning is extended to include explanations.

In this work, we combine explanations with defeasible reasoning. Generally, when you look at classical DLs, the explanations can be derived from using DL reasoning services. But here we need to take defeasibility into account, and the way in which the explanations are obtained are more complex because we have to consider the ranking of statements.

For instance, in order to obtain the explanation to the query "do penguins typically have wings", we first look at all justifications that support the answer YES. The statements "penguins are birds" and "birds typically have wings" can be used to justify the answer. However, according to the information in our knowledge base we know that penguins can not fly thus the logical consequence of this is that there are no penguins since the existence of penguins will cause a conflict. As a result using rational closure, the statements "birds typically fly" and "birds typically have wings" are discarded meaning we can no longer use them in our justification, thus giving the explanation of why the answer to the query is NO.

Generally from an algorithmic perspective what is happening is if the answer to a query is YES, look at all the minimal sets which include only the statements required for the entailment to hold. Then check for the ones that occur as part of the ranked statements. That will be the real explanation. However, note that an entailment can have more than one justification and other justifications can be used to build up the explanation. If the answer is NO, then check to see if there are any minimal subsets that entail the negation [1] of the query and build the explanation from there which is how we came up with the explanation for the query above. Thus, looking at the example above it is clear that explanation for defeasible reasoning will have the same advantages explanation has for classical reasoning systems.

References

1. Baader, F., Hollunder, B.: How to prefer more specific defaults in terminological default logic. In: Bajcsy, R. (ed.) IJCAI 1993. pp. 669–675. Morgan Kaufmann Publishers
2. Baader, F., Calvanese, D., McGuinness, D., Patel-Schneider, P., Nardi, D.: The description logic handbook: Theory, implementation and applications. Cambridge university press, 2 edn. (2007)
3. Bail, S.P., Parsia, B., Sattler, U.: The justificatory structure of OWL ontologies. PhD Thesis, University of Manchester (2013)
4. Britz, K., Casini, G., Meyer, T., Varzinczak, I.: A klm perspective on defeasible reasoning for description logics. In: Description Logic, Theory Combination, and All That, pp. 147–173. Springer (2019)
5. Casini, G., Meyer, T., Moodley, K., Sattler, U., Varzinczak, I.: Introducing defeasibility into OWL ontologies. In: International Semantic Web Conference. pp. 409–426. Springer (2015)
6. Horridge, M.: Justification based explanation in ontologies. PhD Thesis, University of Manchester (2011)
7. Horrocks, I., Patel-Schneider, P.F., Van Harmelen, F.: From shiq and rdf to owl: The making of a web ontology language. Web semantics: science, services and agents on the World Wide Web **1**(1), 7–26 (2003)
8. Moodley, K., Meyer, T., Varzinczak, I.: A protégé plug-in for defeasible reasoning. In: Proceedings of the 25th International Workshop on Description Logics. pp. 486–496 (2012)
9. Rudolph, S.: Foundations of description logics. In: Reasoning Web International Summer School. pp. 76–136. Springer (2011)
10. Sikos, L.F.: Description logics in multimedia reasoning (2017)