Explaining Subsumption and Patching Non-Subsumption with Tableaux Methods

Thorsten Liebig and Stephan Scheele and Julian Lambertz
Ulm University, D-89069 Ulm, Germany
{thorsten.liebig|stephan.scheele|julian.lambertz}@uni-ulm.de

May 15, 2007

Explaining Subsumption and Unsatisfiability

Our approach follows the idea of adapting a tableaux algorithm for explaining subsumption and unsatisfiability as suggested in [1, 4]. The explanation follows the proof and uses the structure of a tableaux tree for the production of natural-language explanations. Our method covers the DL $\mathcal{SHIN}$ including GCIs. We rely on a data structure for annotated terms and nodes to keep book of dependencies between expressions and tableaux nodes and to identify those that contribute to a clash (are relevant). We saturate the tableaux tree to search for alternative explanation candidates. An evaluation function then chooses the best option according to the shortest length. Future work aims at evaluation functions based on empirical knowledge to differentiate explanation options w.r.t. explanation quality. Irrelevant expressions are hidden in the explanation. Furthermore we have developed a technique to optimize an explanation by aggregating a sequence of similar explanation steps into one assertion that a user can drill down if desired. Concerning role hierarchies we have implemented a general methodology to explain the inheritance of properties between roles. Our prototype Tabex is implemented in Lisp (CLOS) and capable of explaining within $\mathcal{SHF}$ including GCIs. Tabex improves the explanation quality by aggregating explanation steps and utilizing the idea of drill-down and roll-up. Optionally it can utilize RacerPro for optimization purposes.

Explaining and Patching Non-Subsumption

When applying the techniques from above, a non-subsumption results in at least one model for the corresponding negated subsumption query. Each of this models is caused by an unclosed tableaux path and can be interpreted as counter examples wrt. the subsumption query. The idea of **explaining non-subsumption** is quite similar to the approach for subsumption. All explanations of unclosed paths build up the explanation that ends when there are no more successors generated and constrained by the subsumer **and** subsumee.

The problem with **patching a non-subsumption** is the infinite search space caused by the infinite many ways of closing at least one of the potentially many unclosed tableaux paths. Therefore, our idea is to constrain the search space by concentrating on a set of common errors of unexperienced users studied in [5]. The errors from [5] considered in this approach are: allquantification instead of existential quantification, wrong use of negation in combination with a quantor ($\neg\forall r.(..)$ vs. $\forall r.(\neg..)$ and $\neg\exists r.(..)$ vs. $\exists r.(\neg..)$) and use of a partial instead of a complete definition. Our strategy looks for symptoms of the described errors whithin an unclosed path. If we find such symptoms in a node, we investigate all involved concept definitions in order to identify the error.

Although concentrating on the mentioned errors reduces the search space significantly, the approach often leads to many suggestions. A first idea to rate suggestions is to leave suggestions leading to a trivial (partial) subsumption unconsidered as mentioned in [3]. A second idea is that the user wants to patch the non-subsumption but does not want the class hierarchy to change elsewhere (compare with [2]). As a first simple metric we compute the differences a suggestion would cause within the direct sub- and superconcepts for all concepts of the KB with help of an external reasoner. A suggestion with little impact on the hierarchy is rated better than one with a high impact. Furthermore, a suggestion patching the non-subsumption is preferred than one which does not.

Conclusion and Outlook

We argue that tableaux-based methods are valuable for explaining as well as providing clues for repairing an unwanted non-subsumption. In comparison to axiom pinpointing our approach currently is restricted in language expressivity, but more robust with respect to large amounts of axioms and provides more detailed explanations. An extension to ABox explanations seems possible.

References

[1] A. Borgida, E. Franconi, I. Horrocks, D. McGuinness, and P. F. Patel-Schneider. Explaining $\mathcal{ALC}$ subsumption. In *Proc. of DL99*, pages 37–40, 1999.

[2] C. Elsenbroich, O. Kutz, and U. Sattler. A Case for Abductive Reasoning over Ontologies. In *Proc. of OWL: Experiences and Directions WS*, 2006.

[3] A. Kalyanpur. *Debugging and Repair of OWL Ontologies.* PhD thesis, University of Maryland College Park, 2006.

[4] T. Liebig and M. Halfmann. Explaining Subsumption in $\mathcal{ALEHF}_{R^+}$ TBoxes. In *Proc. of DL05*, pages 144–151, 2005.

[5] A. L. Rector, N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang, and C. Wroe. OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns. In *EKAW*, pages 63–81, 2004.