Explaining $\mathcal{ALC}$ Subsumption

Alex Borgida
Dept. of Computer Science
Rutgers University, USA
borgida@cs.rutgers.edu

Enrico Franconi and **Ian Horrocks**
Dept. of Computer Science
Univ. of Manchester, UK
{franconi|horrocks}@cs.man.ac.uk

Deborah McGuinness
Dept. of Computer Science
Stanford University, USA
dlm@ksl.stanford.edu

Peter F. Patel-Schneider
Database Systems Research Dept.
Bell Labs, USA
pfps@research.bell-labs.com

1 Introduction

The usability of Description Logic (DL) based knowledge representation systems would be considerably enhanced by the ability to explain subsumption inferences to non-sophisticated users [McGuinness, 1996]. Explanation is relatively natural for systems based on structural subsumption algorithms [McGuinness and Borgida, 1995], but such algorithms are unable to deal with a more complex language such as $\mathcal{ALC}$. Tableaux based systems, on the other hand, can deal with $\mathcal{ALC}$ (and much more complex languages), but the reasoning method does not lead to a natural explanation of subsumption inferences. For example, non-sophisticated users would probably not find it useful to have the subsumption $\forall R.(C \sqcap D) \sqsubseteq \forall R.C$ explained by the fact that $(\forall R.(C \sqcap D)) \sqcap \exists R.\neg C$ is not satisfiable.[1]

Sequent calculi seem to offer both the ability to reason with languages such as $\mathcal{ALC}$, and a more natural way of explaining subsumption inferences to users: sequent rules are meant to axiomatise the entailment relation, and this has an obvious parallel with the subsumption relation. However, most sequent calculi also include reasoning rules that may be difficult for non-sophisticated users to follow, in particular the negation rules. It would also be undesirable to add explanation to tableaux based systems by either replacing or duplicating the tableaux reasoner: in the first case efficiency may suffer, while in the second case implementation and maintenance costs would be greatly increased.

[1] The explanation of *non*-subsumption is more natural for tableaux based systems (a counter-example can be generated) but is much less interesting for users.

The proposed solution is to use a modified sequent calculus based on a larger but more easily explainable set of rules (Section 2), and to demonstrate that a sequent proof using these rules can be generated by a (slightly extended) tableaux algorithm (Section 4).

2 A Sequent Calculus for $\mathcal{ALC}$

We briefly introduce here a sequent calculus for $\mathcal{ALC}$ subsumption. Sequent calculi for $\mathcal{ALC}$ are well known from the modal logic literature [Fitting, 1983]: they exploit the correspondence between $\mathcal{ALC}$ and the multi-modal logic $\boldsymbol{K_{(m)}}$, with the subsumption relation being encoded as the entailment relation in a sequent. There have been already attempts (e.g., [Royer and Quantz, 1992]) to reuse standard sequent calculi from the modal logic community in Description Logics, but only with the goal of extending them with the operators which are peculiar to Description Logics—like, for example, number restrictions or ABoxes. Our goal, however, is different: we want a set of sequent rules which are "easily" explainable, and which parallel the steps of a tableaux-based proof.

If we consider a standard sequent calculus, the main undesirable thing we notice is the presence of $\neg$-*rules*, which move formulæ from antecedents of sequents to succedents and vice versa. If a sequent proof is used as an explanation of subsumption, this results in shifts of subsumers to subsumees and vice versa, which may confuse the user. In order to get rid of the $\neg$-*rules*, we propose to provide additional rules which explicitly consider negation in front of every construct. Figure 1 exemplifies how $\wedge$-*rules* (first line) are doubled by adding their dual negated rules (second line). The $\vee$-*rules* are

$$\frac{X, a, b \vdash Y}{X, a \sqcap b \vdash Y}\ (l\wedge) \qquad \frac{X \vdash a, Y \qquad X \vdash b, Y}{X \vdash a \sqcap b, Y}\ (r\wedge)$$

$$\frac{X, \neg a \vdash Y \qquad X, \neg b \vdash Y}{X, \neg(a \sqcap b) \vdash Y}\ (l\neg\wedge) \qquad \frac{X \vdash \neg a, \neg b, Y}{X \vdash \neg(a \sqcap b), Y}\ (r\neg\wedge)$$

$$\frac{X, a \vdash Y \qquad X, b \vdash Y}{X, a \sqcup b \vdash Y}\ (l\vee) \qquad \frac{X \vdash a, b, Y}{X \vdash a \sqcup b, Y}\ (r\vee)$$

$$\frac{X, \neg a, \neg b \vdash Y}{X, \neg(a \sqcup b) \vdash Y}\ (l\neg\vee) \qquad \frac{X \vdash \neg a, Y \qquad X \vdash \neg b, Y}{X \vdash \neg(a \sqcup b), Y}\ (r\neg\vee)$$

$$\frac{X, a \vdash Y}{X, \neg\neg a \vdash Y}\ (l\neg\neg) \qquad \frac{X \vdash a, Y}{X \vdash \neg\neg a, Y}\ (r\neg\neg)$$

Figure 1: Rules for the propositional formulæ

$$\frac{X' \vdash b, Y'}{X \vdash \forall r.b, Y}\ (r\Box) \qquad \frac{X', b \vdash Y'}{X, \exists r.b \vdash Y}\ (l\Diamond)$$

$$\frac{X', \neg b \vdash Y'}{X, \neg\forall r.b \vdash Y}\ (l\neg\Box) \qquad \frac{X' \vdash \neg b, Y'}{X \vdash \neg\exists r.b, Y}\ (r\neg\Diamond)$$

where $X' = \{a \mid \forall r.a \in X\} \cup \{\neg a \mid \neg\exists r.a \in X\}$, *and*
$Y' = \{a \mid \exists r.a \in Y\} \cup \{\neg a \mid \neg\forall r.a \in Y\}$

Figure 2: Rules for the modal formulæ

simply the dual of the $\wedge$-*rules*. The same can be done for the $\Box$-*rule* (top left rule in Figure 2): the rule is doubled by considering explicit negation (bottom left rule in Figure 2). Again, the $\Diamond$-*rules* are simply the dual of the $\Box$-*rules* (right rules in Figure 2). Another advantage of having explicit negated rules is the avoidance of "normalisation" steps, which may confuse the user by changing the appearance of the problem. In some cases, however, it could be useful to re-introduce a normalisation sub-step during an explanation proof, if this increases the clarity of the explanation of that particular step (see the example in Section 3).

Another non standard feature of our calculus is the way we explicitly consider the applicability condition of the $\Box$- and $\Diamond$-*rules*. The condition states that the rule is applicable if all the homologous universal and existential formulæ are "gathered" together on the left and right hand sides of the sequent in the precondition; the rule is then applied only once. The peculiarity of these rules is that the inference system is still complete without any *weakening* rule (the discarding antecedent or succedent formulæ). The calculus was designed in this way in order to parallel the behaviour of the $\forall$- and $\exists$-*rules* in the tableaux calculus (see Section 4). The application of these rules, and in particular the gathering process, may require some additional explanation to the user, for example by explaining that the conjunction of $\exists R.C$ and $\forall R.D$ implies $\exists R.(C \sqcap D)$. In some cases it may be useful to enhance the quality of explanation by re-introducing a weakening sub-step in the proof, or by introducing a new formula implied by the existing formulæ, as in the above explanation of gathering.

The proposed set of sequent calculus rules now has the nice property that formulæ are never shifted from antecedents of sequents to succedents or vice versa. This is why additional termination axioms are needed (see Figure 3). In fact, in the standard sequent calculus all the termination axioms can be reduced to $X, a \vdash a, Y$ by applying the $\neg$-*rules*, but the absence of these rules forces an explicit treatment of all the cases.

$X, a \vdash a, Y$	$X, \neg a \vdash \neg a, Y$
$X, a, \neg a \vdash Y$	$X \vdash a, \neg a, Y$
$X, \bot \vdash Y$	$X \vdash \top, Y$

Figure 3: Termination axioms

3 Example

As an example, we show now how a sequent proof explaining the following subsumption could be obtained from the devised calculus:[2]

$$\exists\mathtt{child}.\top \sqcap \forall\mathtt{child}.\neg((\exists\mathtt{child}.\neg\mathtt{Doctor}) \sqcup (\exists\mathtt{child}.\mathtt{Lawyer}))$$
$$\sqsubseteq \exists\mathtt{child}.\forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$$

Informally, the explanation for a sequent

$$\mathtt{a}_1, \ldots, \mathtt{a}_n \vdash \mathtt{b}_1, \ldots, \mathtt{b}_m$$

will usually be based on the existence of some $\mathtt{a_i}$ and $\mathtt{b_j}$ such that $\mathtt{a_i} \sqsubseteq \mathtt{b_j}$. The case when non-deterministic rules are involved is more complex and may lead to reasoning by case; however, in each case the explanation reduces to the above deterministic one.

We see no reason why the user should understand the sequent representation, and as we are explaining subsumption we have used $\sqsubseteq$ instead of $\vdash$ in the explanation of the sequent steps. Moreover, since we feel that the different meanings of a comma on the two sides of $\vdash$ may be confusing to the naive user, we make explicit the fact that on the left hand side, it is conjunction, while on the right hand side it is disjunction. Explanations of the sequent $(l\wedge)$ and $(r\vee)$ rules therefore seem superfluous, and may be omitted.

In our example, the first step of the sequent proof—an application of the $(l\wedge)$ rule—is just such a case, and can be omitted from the explanation. Therefore, as far as the user is concerned, the proof starts by applying the $(l\diamond)$ rule, which leads to following judgement:

$$\top \sqcap \neg((\exists\mathtt{child}.\neg\mathtt{Doctor}) \sqcup (\exists\mathtt{child}.\mathtt{Lawyer}))$$
$$\sqsubseteq \forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$$

This step can be explained as: *In order to check that the combination of an $\exists R.A$ concept and an $\forall R.B$ is subsumed by an $\exists R.C$ concept, we can check whether the conjunction of A and B is subsumed by C.* This step is clearly more complex than the proceeding ones, and some users may require a more detailed explanation, possibly utilising the fact that from the conjunction of $\exists R.A$ and $\forall R.B$ we can derive $\exists R.(A \sqcap B)$; such detail is, however, beyond the scope of this document.

Then, by applying the $(l\neg\vee)$ rule, we obtain the following judgement:

$$\top \sqcap \neg(\exists\mathtt{child}.\neg\mathtt{Doctor}) \sqcap \neg(\exists\mathtt{child}.\mathtt{Lawyer})$$
$$\sqsubseteq \forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$$

This step can be explained as: *Applying de Morgan's laws, we propagate negation inward..* An explanation of de Morgan's laws would of course be available if required.

[2]The explanation is not intended to be definitive, merely to show that the calculus provides a reasonable basis for an explanation.

Next, by applying the $(r\Box)$ rule, we obtain the following judgement:

$$\neg\neg\mathtt{Doctor} \sqcap \neg\mathtt{Lawyer} \sqsubseteq \mathtt{Rich} \sqcup \mathtt{Doctor}$$

This step can be explained as: *Let us first apply de Morgan's laws to the two existential quantifiers at the left to give $\forall\mathtt{child}.\neg\neg\mathtt{Doctor}$ and $\forall\mathtt{child}.\neg\mathtt{Lawyer}$. In order to check that the combination of an $\forall R.A$ concept and an $\forall R.B$ concept is subsumed by an $\forall R.C$ concept, we can check whether the conjunction of A and B is subsumed by C.* Again, this is a more complex step that may require further explanation, possibly utilising the fact that from the conjunction of $\forall R.A$ and $\forall R.B$ we can derive $\forall R.(A \sqcap B)$.

Next, by applying the $(l\neg\neg)$ rule, we obtain the following judgement:

$$\mathtt{Doctor} \sqcap \neg\mathtt{Lawyer} \sqsubseteq \mathtt{Rich} \sqcup \mathtt{Doctor}$$

This step can be explained as: *Apply the double negation elimination rule.*

Application of the sequent $(r\vee)$ rule can, as mentioned above, be omitted from the explanation, and so we obtain the termination axiom,

$$\mathtt{Doctor} \sqsubseteq \mathtt{Doctor}$$

which can be explained as: *Obviously, a concept subsumes itself.*

4 Correspondence with Tableaux

It is possible to obtain a sequent proof directly from a standard tableaux algorithm, where applications of tableaux rules correspond with steps in the sequent proof, and clash detections correspond with termination axioms. However, such a proof would begin with an application of the *¬-rule* (e.g., $C \sqsubseteq D$ iff $C \sqcap \neg D \sqsubseteq \bot$), followed by a sequence of normalisation steps: exactly those sequent rules we wish to avoid and which we have eliminated in our modified calculus.

These problems can be solved using *lazy unfolding* and *tagging*. Firstly, the tableaux algorithm can use the lazy unfolding optimisation [Horrocks, 1998], which treats compound concepts in the same way as atomic concepts. This means that normalisation steps are performed only as required by the progress of the tableaux expansion, i.e., in order to apply a tableaux expansion rule to a negated compound concept. A normalisation step can then be combined with the subsequent rule application to give a single sequent step. For example, normalising $\neg(a \sqcap b)$ to $\neg a \sqcup \neg b$, followed by an application of the tableaux *⊔-rule*, corresponds with an application of the sequent $(\neg\wedge)$ rule.

Secondly, by tagging the subsumer concept D derived from the initial transformation of the subsumption problem $C \sqsubseteq D$ into the satisfiability problem $C \sqcap \neg D$, and by consistently tagging all concepts derived from it by applications of tableaux rules, it is always possible to determine whether a concept in a particular stage of the tableaux corresponds with a (negated) succedent of a sequent, i.e., its negation plays the role of subsumer in the explanation step. Thus, if in the previous example the concept $\neg(a \sqcap b)$ had been tagged (had been derived from the initially tagged $\neg D$ by a sequence of tableaux expansion steps), then its tableaux expansion would have been taken to correspond with an application of the sequent $(r\wedge)$ rule.

In the example from the previous section, the tableaux algorithm would be used to demonstrate the unsatisfiability of $\exists\mathtt{child}.\top \sqcap \forall\mathtt{child}.\neg((\exists\mathtt{child}.\neg\mathtt{Doctor}) \sqcup (\exists\mathtt{child}.\mathtt{Lawyer}))$ and $\neg\exists\mathtt{child}.\forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$, where $\neg\exists\mathtt{child}.\forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$ is tagged. An application of the tableaux $\sqcap$-*rule* rule to the first concept corresponds with the sequent $(l\wedge)$ rule, because this concept is not tagged.

The tableaux algorithm would then normalise $\neg\exists\mathtt{child}.\forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$ and apply the $\exists$-*rule* to $\exists\mathtt{child}.\top$ to generate the sub problem consisting of $\top$, $\neg((\exists\mathtt{child}.\neg\mathtt{Doctor}) \sqcup (\exists\mathtt{child}.\mathtt{Lawyer}))$ and $\neg\forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$, where $\neg\forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$ is tagged because it was derived from a tagged concept. This corresponds with the sequent $(l\Diamond)$ rule because the triggering concept, $\exists\mathtt{child}.\top$, is not tagged.

The next step in the tableaux algorithm would be a normalisation of $\neg((\exists\mathtt{child}.\neg\mathtt{Doctor}) \sqcup (\exists\mathtt{child}.\mathtt{Lawyer}))$ followed by an application of the $\sqcap$-*rule* to give $\forall\mathtt{child}.\neg\neg\mathtt{Doctor}$ and $\forall\mathtt{child}.\neg\mathtt{Lawyer}$. The combination of these two steps corresponds with the sequent $(l\neg\vee)$ rule.

The next step would be a normalisation of $\neg\forall\mathtt{child}.(\mathtt{Rich} \sqcup \mathtt{Doctor})$ followed by an application of the $\exists$-*rule* to give $\neg(\mathtt{Rich} \sqcup \mathtt{Doctor})$, $\neg\neg\mathtt{Doctor}$ and $\neg\mathtt{Lawyer}$, where $\neg(\mathtt{Rich} \sqcup \mathtt{Doctor})$ is tagged. The fact that the triggering concept was tagged means that this expansion corresponds with one of the sequent right rules, and the preceeding normalisation step means that it corresponds with the sequent $(r\Box)$ rule.

The tableaux algorithm would then proceed with a normalisation of $\neg\neg\mathtt{Doctor}$, corresponding with the sequent $(l\neg\neg)$ rule, and a normalisation of $\neg(\mathtt{Rich} \sqcup \mathtt{Doctor})$ followed by an application of the $\sqcap$-*rule* to give $\neg\mathtt{Rich}$ and $\neg\mathtt{Doctor}$, where both are tagged. The combination of the triggering concept being tagged and the normalisation step means that this expansion corresponds with the sequent $(r\vee)$ rule.

Finally, the tableau algorithm detects a clash between $\mathtt{Doctor}$ and $\neg\mathtt{Doctor}$. Because $\neg\mathtt{Doctor}$ is tagged, this corresponds with the sequent termination axiom $\mathtt{Doctor} \vdash \mathtt{Doctor}$.

5 Discussion

We have proposed a methodology for explaining subsumption relationships between $\mathcal{ALC}$ concepts based on a modified sequent calculus, and shown how the sequence of sequent rule applications can be derived from an optimised implementation of a standard tableaux algorithm. We do not claim that the sequent rules constitute an explanation in themselves, but that they form a reasonable foundation on which to base an explanation. Their main advantages in this regard is that they preserve the original structure of the two concepts and that concepts are never shifted from subsumer to subsumee or vice versa.

Future work will include a study of methods to enhance the quality of explanations, for example by introducing sub-steps such as weakening or the derivation of new concepts. The extension of this methodology to more expressive DLs will also be investigated.

References

[Fitting, 1983] Melvin Fitting. *Proof Methods for Modal and Intuitionistic Logics.* Kluwer, 1983.

[Horrocks, 1998] I. Horrocks. Using an expressive description logic: FaCT or fiction? In *Proc. of the 6 th International Conference on Principles of Knowledge Representation and Reasoning*, pages 636–647, Trento, Italy, 1998.

[McGuinness and Borgida, 1995] D. McGuinness and A. Borgida. Explaining subsumption in description logics. In *Proc. of the IJCAI'95*, pages 816–821, 1995.

[McGuinness, 1996] D. McGuinness. *Explaining Reasoning in Description Logics.* PhD thesis, Rutgers University, 1996.

[Royer and Quantz, 1992] Veronique Royer and Joachim Quantz. Deriving inference rules for terminological logics. In D. Pearce and G. Wagner, editors, *Logics in AI, Proceedings of JELIA'92*, pages 84–105. Springer, 1992.