Formalized Meta-Theory of Sequent Calculi for Substructural Logics: an abstract

Kaustuv Chaudhuri\textsuperscript{1}, Leonardo Lima\textsuperscript{2}, and Giselle Reis\textsuperscript{3}

\textsuperscript{1}kaustuv.chaudhuri@inria.fr – Inria & LIX/École polytechnique, France
\textsuperscript{2}leonardo.alfs@gmail.com – Technische Universität Dresden
\textsuperscript{3}giselle@cmu.edu – Carnegie Mellon University in Qatar

Sequent calculus proof systems are perhaps the most standard technique used to formulate logics. New logics are nearly always proposed in terms of a sequent calculus. Such proposals are usually also accompanied by certain meta-theorems about the calculi. Cut-elimination is normally one of the first things to be established, as it entails the system’s consistency and makes it suitable for automated proof search. Other meta-theorems include identity reduction, which shows internal completeness of the proof system; rule permutations and inversion lemmas to establish the polarities of connectives; and focusing theorems that establish the existence of normal forms. These proofs involve a number of cases which is sometimes exponential in the number of rules in the system, with many of them being very similar. The development of those proofs by hand is tedious and error prone. The situation is worsened for the case of substructural logics: the selective admissibility of contraction, weakening and exchange may result in more cases with subtle differences. A context which was previously considered as a set, for example, may become a multiset or a sequence. This means that proofs will need to take into account the multiplicity of elements and/or their position. The repetitive and detail-intensive nature of these proofs makes them good candidates for computerization.

We have formalized the meta-theory for several sequent calculi for various fragments of linear logic in the proof assistant Abella. The implementation can be found online at:

\url{https://github.com/meta-logic/abella-reasoning}

This formalization requires the formalization of details that are generally left implicit in informal proofs. These include lemmas on sets and multisets which we take as standard (and invisible) background. At first sight, the
development of this infrastructure may seem as tedious as proving the meta-
theorems by hand – with the additional hurdle of needing to learn a new
technology. Our experience has shown that there are many options for
designing the background theory and with a good encoding of multisets
and their properties, the formalization of particular meta-theorems can be
completed quickly. Moreover, such infrastructure needed to be developed
only once, and was used for encoding contexts in all fragments considered.
The proofs use only elementary theorem proving techniques that can be
explained to and carried out by undergraduate students. They follow the
usual textbook inductive proofs on the rank of the cut formula and/or proof
heights.

We expect that most proof assistants available today are able to handle
this formalization. Only for the first order systems have we used a more
specialized feature: a two-level logic encoding. This facilitates the treatment
of binders by avoiding a few height preserving lemmas, but in principle the
lemmas could have been proved again in Abella at the cost of having explicit
size measures on some definitions.

Given our results and the added certainty that a formalization brings
to a theorem, one is left to wonder why they are not carried out more
frequently. Our hypothesis is that the amount of boilerplate in the proofs
and the non-trivial design of the infrastructure makes the trade-off not worth
for the average proof theorist. Having the infrastructure of contexts available
as libraries (of sets, multisets, sequences, etc) is already a big step. Those
libraries can be tailored for the encoding of proof systems, as to facilitate
proofs of meta-theorems. Reducing the amount of boilerplate in the proofs
is more challenging, as it most likely requires the increase of automation of
proof assistants. We are constantly investigating better ways to deal with
the tedious and repetitive parts of proofs.