



# Formalising Metatheoretical Proofs in Nominal Isabelle

Peter Chapman, supervised by Roy Dyckhoff  
School of Computer Science, University of St Andrews

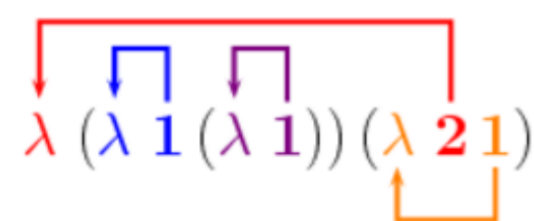
[pc@cs.st-andrews.ac.uk](mailto:pc@cs.st-andrews.ac.uk)

## Introduction

**In the lambda calculus [1], and in other formalisations dealing with variable binding, we have a process known as  $\alpha$ -conversion. This is where we can, if necessary, change the name of a variable, from say  $x$  to  $y$ , wherever it appears in certain kinds of formulae, and the resulting formula will be equivalent. For instance, the identity function  $\lambda x.x$  is always the identity function, regardless of whether it is written as  $\lambda y.y$  or  $\lambda \rho.\rho$ . Whilst it is easy to understand, it is difficult to formalise effectively; a naïve implementation would usually insist that  $\lambda x.x$  and  $\lambda y.y$  are not equivalent.**



More complicated implementations are usually difficult to read, as they often hide away the important parts of a proof in a sea of notation, most of which is specifically there to deal with  $\alpha$ -conversion. Indeed, most of a development is devoted to proving trivial facts about substitutions, that would be completed on paper in a few lines. To give a solution to this problem, an extension to the Isabelle proof assistant was developed called *Nominal Isabelle* [2].

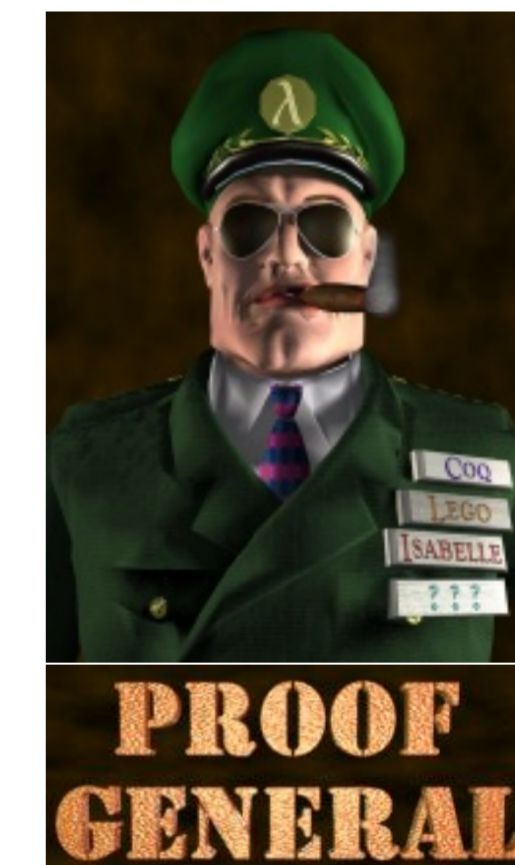


The above term is given in de Bruijn notation, where each lambda is paired with a variable by a number. For large terms, this becomes unreadable very quickly. However, de Bruijn notation was frequently used in implementations before *Nominal Isabelle* was developed. In *Nominal Isabelle*, we could simply write the above term as  $\lambda[z].(\lambda[y].y(\lambda[x].x))(\lambda[w].zw)$ , which is easier to interpret than the given representation using de Bruijn indices, and is almost identical to the informal pen-and-paper approach.

## Example – Craig’s Interpolation Theorem

We give some details about a particular metatheoretical result, called *Craig’s Interpolation Theorem*. Informally speaking, this is a result about implication. If the formula  $A \supset B$  is valid, then we can find an interpolant  $C$  such that  $A \supset C$  and  $C \supset B$  are valid, and moreover that  $C$  is expressed in the common language of  $A$  and  $B$ .

```
File Edit View Cmds Tools Options Buffers Proof-General X-Symbol Isabelle
State Context Retract Undo Next Use Goto Find Command Stop Restart Help
NominalCraig.thy | Scratch.thy
(* case 1, an axiom *)
lemma craigIntAxi:
  fixes  $\Gamma_1 \Gamma_2 :: \text{"form set"}$ 
  and  $P :: \text{"form"}$ 
  assumes As1: " $P \in \Gamma_1$ "
  and As2: " $\text{finite } \Gamma_1 \wedge \text{finite } \Gamma_2$ "
  shows " $\exists A \text{ dl dr. } (\text{dl} \vdash (\Gamma_1 \Rightarrow A))$ 
         $\wedge (\text{dr} \vdash ((\Gamma_2 \cup \{A\}) \Rightarrow P))$ 
         $\wedge (A \in \Gamma_1 \Gamma_2 P)$ "
proof-
  from As1 As2 have dl:"Init ( $\Gamma_1 \Rightarrow P$ )  $\vdash (\Gamma_1 \Rightarrow P)$ " by force
  from As2 have dr: "Init ( $\Gamma_2 \cup \{P\} \Rightarrow P$ )  $\vdash (\Gamma_2 \cup \{P\} \Rightarrow P)$ " by force
  from As1 have "ccn P  $\subseteq$  ccnIn  $\Gamma_1$ " by force
  moreover have "ccn P  $\subseteq$  ccnIn  $\Gamma_2 \cup \text{ccn } P$ " by force
  ultimately have cqn: " $P \in \Gamma_1 \Gamma_2 P$ " using As1 by force
  from dl dr cqn show ?thesis by blast
qed
IS08-----XEmacs: NominalCraig.thy (Isar script XS:isabelle/s Font Scripting)----52
proof (prove): step 19
using this:
  Init ( $\Gamma_1 \Rightarrow P$ )  $\vdash (\Gamma_1 \Rightarrow P)$ 
  Init ( $\Gamma_2 \cup \{P\} \Rightarrow P$ )  $\vdash (\Gamma_2 \cup \{P\} \Rightarrow P)$ 
  P  $\in \Gamma_1 \Gamma_2 P$ 
goal (1 subgoal):
  1.  $\exists A \text{ dl dr. } \text{dl} \vdash (\Gamma_1 \Rightarrow A) \wedge \text{dr} \vdash ((\Gamma_2 \cup \{A\}) \Rightarrow P) \wedge A \in \Gamma_1 \Gamma_2 P$ 
```



## Further Work

- Much of the current work in formalising metatheory using sequent calculi is done using *sets*. However, for the most part, the original work was presented via *multisets*. In an effort to unify as far as possible the two approaches, we wish to begin to use multisets more often in the formal methods.
- There are other problems which we could use our development on. Most notable amongst these is Cut Admissibility [4], which can be seen as the opposite direction of interpolation; given  $A \supset C$  and  $C \supset B$ , we can cut  $C$  to derive  $A \supset B$ , and still be able to prove the same set of theorems. In particular, Cut admissibility is a result that says that no more theorems are provable with the presence of the Cut rule, than without it.
- We could create extensions to *Isabelle* that would automate a lot of the uninteresting steps in such metatheoretical proofs. The important steps are the derivations. Ideally, we would like to specify the rules of the system, the problem to be solved, and the derivations that provide a solution, and *Isabelle*, with the machinery of *Proof General* would decide whether this constituted a proof or not.

## References

- [1] Barendregt, H. The Lambda Calculus: Its Syntax and Semantics, Revised Edition. Studies in Logic and the Foundations of Mathematics, Volume 103, Elsevier, 1984.
- [2] Tasson, C and Urban, C. Nominal Techniques in Isabelle/HOL. Lecture Notes in Computer Science, Volume 3632, Springer-Verlag, 2005.
- [3] Ridge, T. Craig’s Interpolation Theorem formalised and mechanised in Isabelle/HOL, available from <http://www.cl.cam.ac.uk/~tjr22>, 2006.
- [4] Troelstra, A.S. and Schwictenberg, H. Basic Proof Theory, Second Edition. Cambridge Tracts in Theoretical Computer Science, 2000.