Interoperability between arithmetic proofs using Dedukti

Francois Thiré
DEDUC-EAM (INRIA)
francois.thire@inria.fr

Motivations & objectives

- There exist many logics and many proof checkers for these logics
- Some logics are more powerful than others (e.g., quantify over proofs)
- Theorems and proofs are not shared between different proof checkers: Well-known theorems and proofs are proved manually for each new proof checker/logic
- There is no standard for these logics
- Our objective is to translate automatically a small logic of arithmetic proofs from an expressive logic called The Calculus of Inductive Constructions (CiC), to a less expressive logic, The Higher Order Logic (HOL).

Dedukti is handy for interoperability

- Dedukti implements the \(\lambda\Pi\)-calculus modulo theory. It is a logical framework
- Logical frameworks are a kind of proof checker that allows to embed several logics
- Logical frameworks are good systems to make interoperability easier
- \(\lambda\Pi\)-calculus modulo theory is a simple logic that combines dependent types and rewrite rules
- The substantial advantage of Dedukti for interoperability is that the encoding of a logical system \(L\) to Dedukti is shallow:
  - Use of Higher-order Abstract syntax
  - Type preservation: \(\Gamma \vdash t : T \Rightarrow \Gamma \vdash \tilde{t} : \tilde{T}\)
  - Computation preservation: \(t_1 \rightarrow_L t_2 \Rightarrow \tilde{t}_1 \rightarrow_{\tilde{L}} \tilde{t}_2\)

Dedukti[CiC] to Dedukti[HOL]

- This translation is not always possible: In CiC, it is possible to quantify over proofs, not in HOL or in CiC, there is an infinite hierarchy of universes that does not exist in HOL.
- But there should be a translation for arithmetic proofs: one does not need universes nor to quantify over proofs
- Features to remove:
  - Universes
  - Dependent types
  - Inductive definitions and recursive definitions (encoded in \(\lambda\Pi\)-calculus modulo theory by rewrite rules)

Implementation

- Implementation in OCaml (5000 lines)
- Compilation time with Ediloh (Fermat little theorem): 20s
- Several independent tools:
  - Universo (remove universes)
  - Deduktipli (remove dependent types)
  - Ediloh (the compiler from Dedukti[HOL] to OpenTheory)

Future Work

- Automatize the translation
- Extend that arithmetic library to other provers such as Coq, Matita, PVS...
- Create the W3P (W3C of proofs) in order to create the first standards for proofs
- OCaml is not really handy to write such compilers especially for binders. A joint work with Prof. Brigitte Pientka is to look at a new logical/programming system that would be handy to express proof transformations
- Embed other logical systems like CubicalTT (C. Leena Subramaniam)
- Extend Dedukti to rewrite modulo a congruence (G. Ferey)