# Gilles Dowek's technique for proving the termination of the embedding of HOL in Dedukti

Frédéric Blanqui


Deduc⊢eam

# Dedukti, a type-checker for $\lambda\Pi/\beta\mathcal{R}$

$\lambda\Pi$-calculus modulo a set $\mathcal{R}$ of user-defined rewrite rules

$\lambda$-calculus with:

- dependent types: $\Pi p : \mathbb{N}.\mathbb{L}p \to \Pi q : \mathbb{N}.\mathbb{L}q \to \mathbb{L}(p+q)$
- simple types: $A \to B$ shorthand for $\Pi x : A.B$ with $x \notin \mathrm{FV}(B)$
- objects defined by rewrite rules: $(x+y)+z \to_{\mathcal{R}} x+(y+z)$
- types defined by rewrite rules: $\mathsf{F}0 \to_{\mathcal{R}} \mathbb{N}$, $\mathsf{F}(x+1) \to_{\mathcal{R}} \mathbb{N} \to \mathsf{F}x$
- types equivalent modulo $\beta\mathcal{R}$ are identified

# $\lambda\Pi/\beta\mathcal{R}$ as extended pure type system (PTS)

- sorts $s \in \{Type, Kind\}$
- axiom $Type : Kind$
- product rule $\dfrac{U : Type \quad V : s}{\Pi x : U.V : s}$
  (one can have type variables but cannot quantify over them)
- conversion rule $\dfrac{t : T \quad T =_{\beta\mathcal{R}} T' \quad T' : s}{t : T'}$

# $\lambda\Pi/\beta\mathcal{R}$ is a versatile framework

for each functional PTS $\Lambda$,
there are $\mathcal{R}_\Lambda$ and $\varphi, \psi : \Lambda \to \lambda\Pi/\beta\mathcal{R}_\Lambda$ such that:

- if $\Gamma \vdash_\Lambda t : T$ then $\varphi(\Gamma) \vdash_{\lambda\Pi/\beta\mathcal{R}_\Lambda} \psi(t) : \varphi(T)$
  [Cousineau & Dowek, 2007]
- if $\varphi(\Gamma) \vdash_{\lambda\Pi/\beta\mathcal{R}_\Lambda} u : \varphi(T)$ then there is $t$ such that $\Gamma \vdash_\Lambda t : T$
  [Assaf 2015]

polymorphism can be encoded using type-level rewriting!

# Encoding of functional PTSs

- $s : \mathit{Type}$ for each sort $s$ (type of codes for terms of type $s$)
- $\varepsilon_s : s \to \mathit{Type}$ for each sort $s$ (decoding function)
- $c_{s_1} : s_2$ for each axiom $s_1 : s_2$
- $\varepsilon_{s_2} c_{s_1} \to_{\mathcal{R}} s_1$ for each axiom $s_1 : s_2$
- $\pi_{s_1,s_2} : \Pi x : s_1.(\varepsilon_{s_1} x \to s_2) \to s_3$ for each rule $(s_1, s_2, s_3)$
- $\varepsilon_{s_3}(\pi_{s_1,s_2} xy) \to_{\mathcal{R}} \Pi z : \varepsilon_{s_1} x.\varepsilon_{s_2}(yz)$ for each rule $(s_1, s_2, s_3)$

# Dedukti: towards a universal proof translator/checker?

Dedukti comes with various companion tools translating terms and proofs of other systems into $\lambda\Pi$ modulo some rewrite systems

- **Holide** translates OpenTheory files (generated from HOL-Light, HOL4 or ProofPower) into Dedukti files
- **Coqine** translates Coq vo files into Dedukti files
- **Krajono** translates Matita vo files into Dedukti files
- **iProverModulo** and **ZenonModulo** are automated theorem provers generating Dedukti files

# Encoding of simple type theory (HOL)

with a shallow embedding of simple types:

- $\iota : \mathit{Type}$ (type of HOL objects)
- $o : \mathit{Type}$ (type of HOL propositions)
- $\Rightarrow : o \to o \to o$
- $\forall_A : (A \to o) \to o$ for every simple type $A$
- $\varepsilon : o \to \mathit{Type}$ (proposition-as-type decoding function)
- $\varepsilon(x \Rightarrow y) \to_{\mathcal{R}} \varepsilon x \to \varepsilon y$
- $\varepsilon(\forall_A y) \to_{\mathcal{R}} \Pi z : A.\varepsilon(yz)$

problem: termination of $\to_\beta \cup \to_{\mathcal{R}}$?

# Finite encoding (with a deep embedding of simple types)

- $\mathrm{type} : \mathit{Type}$ (type of HOL type codes)
- $c_\iota : \mathrm{type}$
- $c_o : \mathrm{type}$
- $c_\to : \mathrm{type} \to \mathrm{type} \to \mathrm{type}$
- $\eta : \mathrm{type} \to \mathit{Type}$ (decoding function of HOL type codes)
- $\eta(c_\to xy) \to_{\mathcal{R}} \eta x \to \eta y$
- $o = \eta c_o$

- $\Rightarrow : o \to o \to o$
- $\forall : \Pi x : \mathrm{type}.(\eta x \to o) \to o$
- $\varepsilon : o \to \mathit{Type}$
- $\varepsilon(x \Rightarrow y) \to_{\mathcal{R}} \varepsilon x \to \varepsilon y$
- $\varepsilon(\forall xy) \to_{\mathcal{R}} \Pi z : \eta x.\varepsilon(yz)$

problem: termination of $\to_\beta \cup \to_{\mathcal{R}}$?

# Applicability of reducibility-based methods?

problem: find $[\![\varepsilon]\!] : \mathcal{T}/\beta\mathcal{R} \to \mathit{RED}$ such that

$$[\![\varepsilon]\!](\forall_A v) = [\![\Pi z : A.\varepsilon(vz)]\!] = \{t \mid \forall u \in [\![A]\!], tu \in [\![\varepsilon]\!](vu)\}$$

$vu < \forall_A v$?

problem: $\forall_A : (A \to o) \to o$ not always positive (e.g. if $A = o$)

# Gilles Dowek's technique (2016)

1. prove that $\beta$ terminates using a reducibility-based method
2. prove that $\beta \cup \mathcal{R}$ terminates using an adhoc technique

# $SN(\beta)$

Barthe's result "relevance of proof irrevelance" (ICALP'98) does not apply because of type-level rewriting

for each $\mathcal{R}$, we need to find a model compatible with $\beta\mathcal{R}$

# $SN(\beta) \Rightarrow SN(\beta\mathcal{R})$

let $t \notin SN(\beta\mathcal{R})$ minimal wrt $\beta$, $\mathcal{R}$ and size
then $t = \varepsilon(\forall_A u) \rightarrow_{\mathcal{R}} \Pi x : A.\varepsilon(ux)$ and $\varepsilon(ux) \notin SN(\beta\mathcal{R})$
then either:

- $u = \lambda x : A.v$ and $t = \varepsilon(\forall_A(\lambda x : A.v)) \rightarrow_{\mathcal{R}} \rightarrow_{\beta} \Pi x : A.\varepsilon v$
- $u = \forall_B$ and $t = \varepsilon(\forall_{B \rightarrow o} \forall_B) \rightarrow_{\mathcal{R}} \rightarrow_{\mathcal{R}} \Pi x : B \rightarrow o.\Pi y : B.\varepsilon(xy)$
- $u = (v \Rightarrow)$ and $t = \varepsilon(\forall_o(v \Rightarrow)) \rightarrow_{\mathcal{R}} \rightarrow_{\mathcal{R}} \Pi x : o.\varepsilon v \rightarrow \varepsilon x$

# Possible directions of research

- there is already a model for HOL and CC: find a general model construction?
- find a class of rewrite systems $\mathcal{R}_2$ such that $SN(\beta\mathcal{R}_1) \Rightarrow SN(\beta\mathcal{R}_1\mathcal{R}_2)$