Encoding Lean's Type Theory in Dedukti

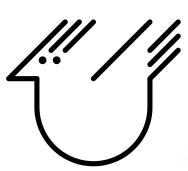
Frédéric Blanqui

joint work with Rishikesh Vaishnav



Proof assistants

- proof assistants are tools helping to write and check proofs
- many proof assistants have been developed over the years




HOL
Interactive Theorem Prover
LEAN
THEOREM PROVER
PVS

...

- they all duplicate basic libraries on integers, real numbers, ...
- some definitions or theorems are available in only one system

Can we reuse some development done in another system?

This is difficult:

- proof assistants greatly differ in their syntax and underlying logic, and are constantly evolving
- in contrast to programming languages, proof systems have different expressive power
- developing and maintaining translators between all pairs of proof assistants is not sustainable

Our approach: use a logical framework as pivot language

a logical framework is a formal language in which one can express deduction rules and proofs

examples: predicate logic, pure type systems, $\lambda\Pi$-calculus, Dedukti

The $\lambda\Pi$ logical framework

the $\lambda\Pi$-calculus is the simplest type system combining both simple and dependent types [de Bruijn, 1968]

$$\frac{\Gamma \vdash A \ominus \mathsf{Type} \quad \Gamma, x : A \vdash B \ominus \mathsf{Type}}{\Gamma \vdash (\Pi x : A, B) \ominus \mathsf{Type}} \qquad (\Pi x : A, B \text{ is } A \to B \text{ if } x \notin B)$$

$$\frac{\Gamma \vdash t \ominus (\Pi x : A, B) \quad \Gamma \vdash u \ominus A}{\Gamma \vdash tu \ominus B[x/u]}$$

can encode different logics [Harper, Honsell, Plotkin, LICS'87]

The Dedukti logical framework

- Dedukti is like the $\lambda\Pi$-calculus but on equivalence classes of types modulo some equational theory E:

$$\frac{\Gamma \vdash t : A \quad A =_E B}{\Gamma \vdash t : B}$$

- a Dedukti theory is a set of typed constants together with a set of typed equations E (2nd-order generalized algebraic theory)
- there exist Dedukti theories for many logics from (fragments of) first-order logic, arithmetic, Zermelo set theory, to higher-order logic, pure type systems,... [Dowek *et al*, years 2000]
- equations allow more shallow encodings (systems with implicit computations can be encoded in $=_E$)

problem: decidability of $=_E$?
find a convergent rewrite system R equivalent to E
(for instance using Knuth-Bendix completion)

Example: Dedukti theory for first-order logic (FOL)

constants:

I : Type — type of FO terms

$f : I \to \ldots \to I \to I$ — for each function symbol f of arity n

$Prop$: Type — type of FO formulas

$P : I \to \ldots \to I \to Prop$ — for each predicate symbol P of arity n

$\top : Prop$, $\neg : Prop \to Prop$, $\forall : (I \to Prop) \to Prop$, ...

$Prf : Prop \to$ Type — [Curry-de Bruijn-Howard]

equations:

$$Prf\,(A \Rightarrow B) = Prf\ A \to Prf\ B \qquad \text{[Brouwer-Heyting-Kolmogorov]}$$
$$Prf\,(\forall A) = \Pi x : I, Prf\,(A\,x)$$
$$Prf\,(A \wedge B) = \Pi C : Prop, (Prf\ A \to Prf\ B \to Prf\ C) \to Prf\ C$$
$$Prf\,\bot = \Pi C : Prop, Prf\ C$$
$$Prf\,(\neg A) = Prf\,A \to Prf\,\bot$$

...

theorem: π proof of A in FOL $\Leftrightarrow |\pi| \circ Prf\,|A|$ in Dedukti

Lean's type theory (1/2)

extends $\lambda\Pi$ with:

- instead of a single universe Type, a linear hierarchy of universes $(\mathsf{U}i)_{i\in\mathbb{N}}$ closed by dependent product (= inaccessible cardinals)

$$\frac{\Gamma \vdash A : \mathsf{U}i \quad \Gamma, x : A \vdash B : \mathsf{U}j}{\Gamma \vdash (\Pi x : A, B) : \mathsf{U}(\max\, i\, j)}$$

- the universe of propositions U0 is impredicative

$$\frac{\Gamma \vdash A : \mathsf{U}i \quad \Gamma, x : A \vdash B : \mathsf{U}0}{\Gamma \vdash (\Pi x : A, B) : \mathsf{U}0}$$

- quantification on types (polymorphism)

$$\frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash A : \mathsf{U}i}{\Gamma \vdash (\lambda x : A, t) : (\Pi x : A, B)}$$

Lean's type theory (2/2)

is also modulo some equational theory $=_{Lean}$ for:

- inductive types and primitive recursive functions
- primitive projections for records
- quotient types
- universe polymorphism (quantification on universe variables)

$$\frac{\mathbb{N} \models i = j}{\mathsf{U}i =_{Lean} \mathsf{U}j} \qquad i, j := x \mid 0 \mid i + 1 \mid \max\, i\, j \mid \mathrm{imax}\, i\, j$$
$$[\![\mathrm{imax}\, i\, j]\!] = \text{if } j = 0 \text{ then } 0 \text{ else } \max\, i\, j$$

- η for functions and records
- η for unit types and proofs (proof irrelevance)

$$\frac{\Gamma \vdash A \circledcirc \mathsf{U}0 \quad \Gamma \vdash t \circledcirc A \qquad \Gamma \vdash u \circledcirc A}{\Gamma \vdash t =_{Lean} u}$$

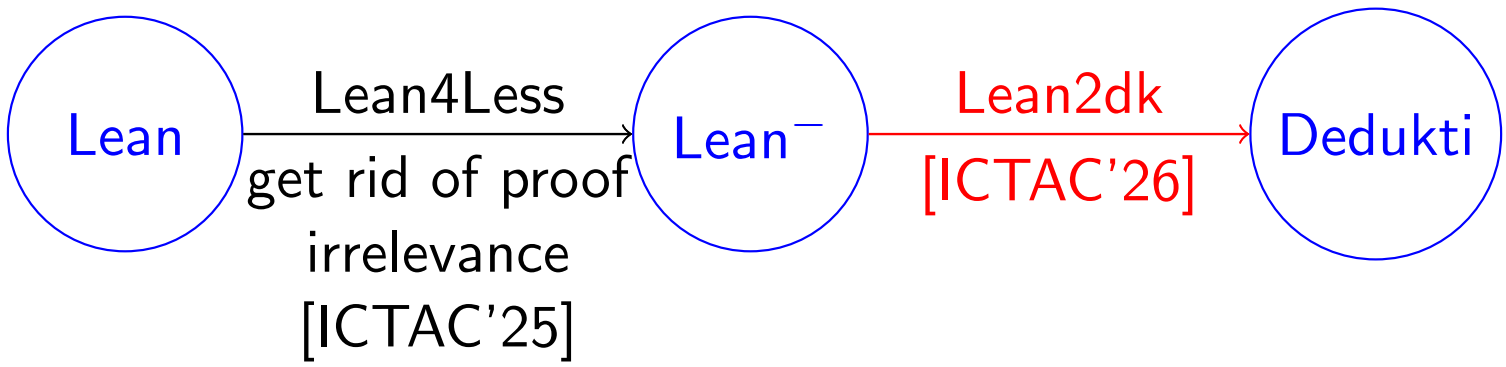
How to encode Lean's type theory in Dedukti?

problems:

- encoding of types? [Cousineau, Dowek, TLCA'07]
  - assumes that all the types of a term are equivalent
  - satisfied by Lean? open problem! but 2026 work by Carneiro *et al* using denotational semantics [Coquand, Huber, 2019]
- convergent rewrite system for Lean equivalence on universes?
- primitive recursive functions and quotient types?
- η for functions and records?
- η for unit types and proofs (proof irrelevance)?
  - unorientable equations
  - no head symbol
  - conditionally based on typing

  get rid of η for unit types and proofs first [Vaishnav, ICTAC'25]

How to encode Lean's type theory in Dedukti?



Convergent system for Lean equivalence on universes?

no universe variables: easy

$$\begin{aligned}
\max\ 0\, x &\hookrightarrow x \\
\max\ x\, 0 &\hookrightarrow x \\
\max\,(x+1)\,(y+1) &\hookrightarrow (\max\ x\, y)+1 \\
\mathrm{imax}\ x\, 0 &\hookrightarrow 0 \\
\mathrm{imax}\ x\,(y+1) &\hookrightarrow \max\ x\, y
\end{aligned}$$

difficulty with universe variables:

max satisfies unorientable equations (e.g. commutativity)

Case of max

[Voevodsky, 2014]: every expression with variables, 0, $+1$ and max is equivalent to an expression of the form $\max E$ where E is a set of expressions of the form $x + i$ or i ($i \in \mathbb{N}$)

there is a convergent rewrite system using matching modulo AC computing that normal form [Genestier, FSCD'20]

we can get rid of AC-matching by using a strategy alternating standard rewriting steps and AC-normalization steps [B., FSCD'22]

Case of imax

[Géran, CSL'26]: every expression with variables, 0, $+1$, max and imax is equivalent to an expression of the form $\max E$ where E is a set of expressions of the form $V\,(G \cup \{x\})\,x\,i$ or $C\,G\,(i+1)$ where

- G is a set of variables
- $[\![C\,G\,i]\!] = [\![V\,G\,x\,i]\!] = 0$ if $0 \in [\![G]\!]$
- $[\![C\,G\,i]\!] = i$ and $[\![V\,G\,x\,i]\!] = [\![x]\!] + i$ otherwise.

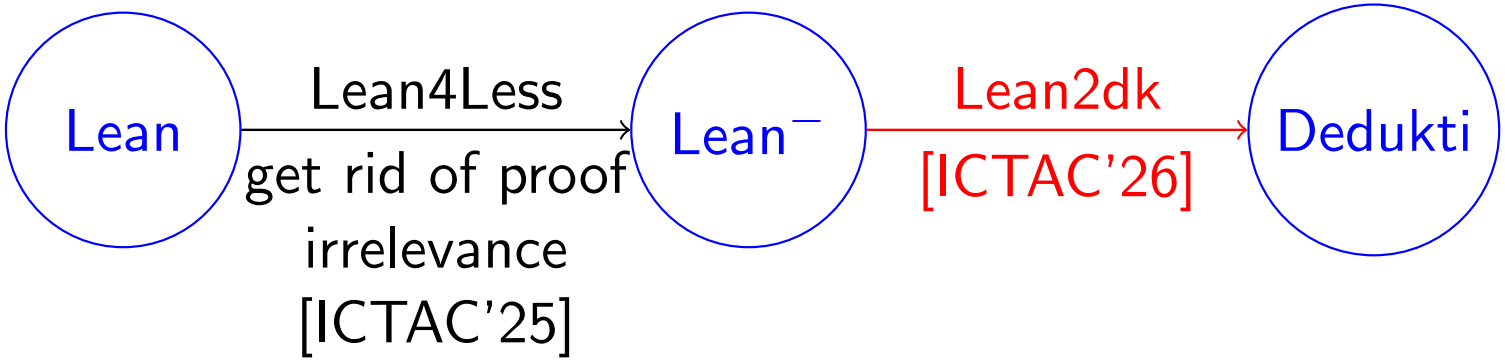
there is a convergent rewrite system computing that normal form when variables are encoded by natural numbers [ICTAC'26]

(find details in the paper)

Conclusion

problems:

- encoding of types? ✅ [ICTAC'26] extension of [Cousineau, Dowek, TLCA'07] to type systems with universe variables where all types of a term are equivalent
- convergent system for equivalence on universes? ✅ [ICTAC'26]
- primitive recursive functions and quotient types? ✅
- η for functions and records? ✅
- η for unit types and proofs (proof irrelevance)? ✅ using



- does this scale to large libraries? ❓ ongoing work
  https://github.com/Deducteam/lean2dk