Gilles Dowek (1966-2025)

Frédéric Blanqui

The prehistory of Dedukti: Deduction Modulo (1999)

Gilles introduced “Deduction Modulo” in his Habilitation Thesis

This is Natural Deduction on equivalence classes of formulas

He showed that several logics can be seen as instances of this general framework: higher-order logic, arithmetic, Z set theory

and proved results valid whatever is the equivalence relation

At the origin of Dedukti (2007)

In 2007, Gilles and his PhD student Denis Cousineau proved that:

> any terminating functional pure type system can be faithfully encoded in the $\lambda\Pi$-calculus modulo rewriting

Remarks:

- the $\lambda\Pi$-calculus is the simplest functional pure type system featuring simple and dependent types
- it does not feature polymorphism but polymorphic systems (e.g. Girard' system F or Huet-Coquand's calculus of constructions) can be encoded in it by using type-level rewrite rules

The birth of Dedukti (2009-2011)

Gilles then got a PhD student, Mathieu Boespflug, who developed in Haskell a type-checker for the $\lambda\Pi$-calculus modulo rewriting called Dedukti ("deduction" in Esperanto)

Remarks:

- Together with Maxime Dénès and Benjamin Grégoire, Mathieu Boespflug's work helped in greatly improving the efficiency of the Coq proof assistant (dividing the checking time of Gonthier's proof of the 4-colors theorem from 3 hours to 30 minutes)
- Other checkers for Dedukti files have been developed later:
  - dkcheck, written in OCaml by Ronan Saillard in 2013-2015
  - lambdapi, initially written by Rodolphe Lepigre in 2017-2018
  - kontroli, written in Rust by Michael Färber in 2019-2020

Dedukti, a universal proof checker (2012-)

Gilles then got several PhD students and colleagues who developed translations from various proof systems to Dedukti:

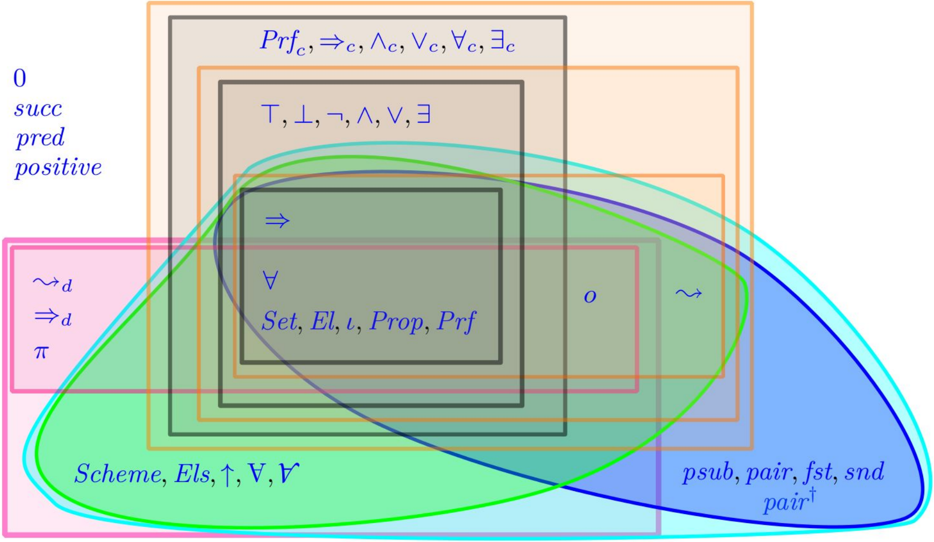
- Coq (Boespflug, Burel, 2012; Férey, 2021)
- OpenTheory (Assaf, Burel, 2012)
- ZenonModulo (Delahaye,Doligez,Gilbert,Halmagrand,Hermant,2013)
- Matita (Assaf, 2015)
- FoCaLiZe (Cauderlier, 2015)
- TSTP (El Haddad, Burel, B., 2019; Sutciffe, B., Burel, 2023)
- PVS (Hondet, B., 2020)
- Agda (Genestier, 2020)
- K (Ledein, 2022)
- Metamath (Ledein, 2023)
- Alethe (Coltellacci, 2023)
- Eunoia (Dunne, 2024)
- Lean (Vaishnav, 2024)
- LEO-III (Taprogge, 2024)
- Vampire (Petković Komel, Rawson, Suda, 2025)

Each prover comes with its own challenges

- Matita: universe variables
- Coq: universe cumulativity
- Agda: universe polymorphism
- PVS: subtyping
- Lean: proof irrelevance
- HOL-Light: huge non-computational low-level proofs
- ...

But encoding is not enough: we need to compare systems

The theory U: a modular construction of type theories (B., Dowek, Grienenberger, Hondet, Thiré, 2021)



$Prf_c, \Rightarrow_c, \wedge_c, \vee_c, \forall_c, \exists_c$
0
$succ$
$pred$
$positive$
$\top, \bot, \neg, \wedge, \vee, \exists$
$\Rightarrow$
$\forall$
$Set, El, \iota, Prop, Prf$
o
$\rightsquigarrow$
$\rightsquigarrow_d$
$\Rightarrow_d$
π
$Scheme, Els, \uparrow, \forall, \not\forall$
$psub, pair, fst, snd$
$pair^\dagger$

Towards proof systems interoperability (2018-)

going from Dedukti/HOL to various proof systems:

- Coq (Thiré, 2018; B. 2024)
- Matita (Thiré, 2018)
- Lean (Thiré, 2018)
- PVS (Thiré, 2018))
- OpenTheory (Thiré, 2018)
- Agda (Felicissimo, 2023)
- Lean (Vaishnav, B., 2025)

But translation is not enough: we need concept alignments

- assume that there exists on some structure S a theorem $T(S)$
- S is defined both in system A and in system B, giving S_A and S_B
- but T is proved in system B only, giving $T_B(S_B)$
- the translation of $T_B(S_B)$ gives $T_A(S'_B)$ and not $T_A(S_A)$
- to obtain $T_A(S_A)$ we need to prove that S'_B and S_A are indeed isomorphic

The Logipedia project proposal (2020)

Thank you Gilles