External Provers In Proof Assistants

Frédéric Blanqui, Inria Saclay
Guillaume Burel, SAMOVAR
Yacine El Haddad, LSV

Abstract

To widen their use, proof assistants needs more automation. This can be performed by calling external automated theorem provers. Centered around Dedukti, a proof checker for the universal logical framework $\lambda\Pi$-calculus modulo theory, the objective is to design a proof environment that is able to call external provers to build part of the proof. The global architecture must emphasize the need to have a correct and exhaustive proof once the portions proved externally are glued together. From Dedukti to external provers, proof obligations in the $\lambda\Pi$-calculus modulo theory must be passed in an efficient way, either by encoding them or by extending the external prover to accept them. Reciprocally, proofs or proof traces produced by these tools needs to be reconstructed back into Dedukti.

Context

Among techniques for verifying software, those based on proofs are very promising.
(Cf. CompCert in Coq.)

One can distinguish between proof assistants

- very expressive
- interaction with the user

and automated theorem provers (ATPs)

- fully automatic
- finely tuned, optimization hacks

The main challenges are the following:

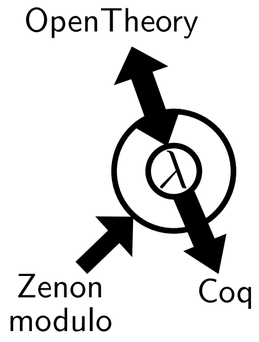
- lack of automation of proof assistants
- lack of trust in ATPs
- lack of interoperability between proof systems

Dedukti

Logical framework based on the $\lambda\Pi$-calculus modulo rewriting

Can express many logics

- Import from Matita, OpenTheory, FoCaLize, ...
- Export to Coq, Matita, PVS, Lean, OpenTheory (HOL Light, HOL4, ...)
- ATPs with a Dedukti output: iProverModulo, Zenon modulo, ArchSat



Dedukti [2] is an universal framework for proof interoperability [3]

Traces vs. Proofs

Veracity of a theorem can be obtained by checking complete proof, e.g. in Dedukti format.

Some ATPs produces Dedukti proofs:

- Zenon Modulo, iProverModulo

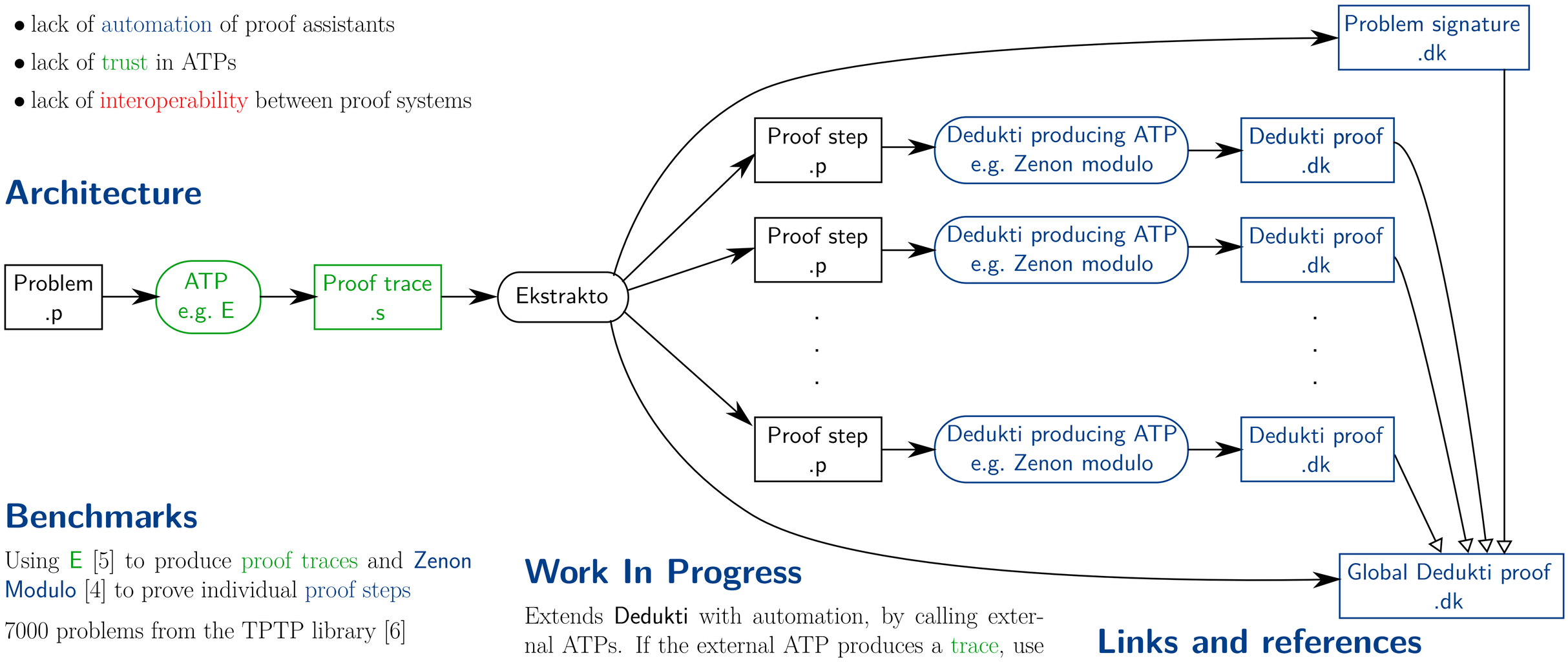
Many others only provide only proof traces

- partial informations, only coarse-grained steps

Tools generating complete proofs are in general less efficient than tools producing proof traces.

How to reconstruct a formal proof checkable by Dedukti from a proof trace?

Architecture



Benchmarks

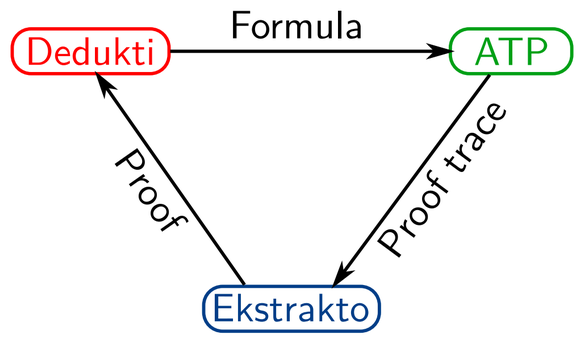
Using E [5] to produce proof traces and Zenon Modulo [4] to prove individual proof steps

7000 problems from the TPTP library [6]

	E	Zenon Modulo	E+Zenon Modulo
Proved	55%	15%	34%
Checkable	0%	15%	**34%**

Work In Progress

Extends Dedukti with automation, by calling external ATPs. If the external ATP produces a trace, use Ekstrakto to get a complete proof.



Some proof steps are not provable, they only preserves provability (e.g. Skolemization)
⇒ need a special handling for them

Gain

- Complete proofs
- Fast generating (parallel computation of proof steps)
- Agnostic wrt the ATPs that are used

Links and references

[1] https://github.com/elhaddadyacine/ekstrakto

[2] https://deducteam.github.io/

[3] Ali Assaf et al., Dedukti: a logical framework based on the $\lambda\Pi$-calculus modulo theory. Submitted, 2016.

[4] http://deducteam.gforge.inria.fr/zenonmodulo/

[5] http://www.eprover.org/

[6] http://tptp.cs.miami.edu/

This research was partially supported by Labex DigiCosme (project ANR11LABEX0045DIGICOSME) operated by ANR as part of the program « Investissement d'Avenir» Idex ParisSaclay (ANR11IDEX000302).

Contact : guillaume.burel@ensiie.fr