Ekstrakto and SKonverto

TPTP Tea Party

Guillaume Burel

Thursday July 13th, 2023

Samovar, ENSIIE

Goals

Increasing Trust in Automated Theorem Provers

- checkable proofs
- reproducibility

Enabling Cooperation between tools

- between provers/solvers
- with proof assistants

Use Dedukti for interoperability and proof checking

Trusting automated theorem provers

Automated theorem provers:

- quite big piece of software
- complex proof calculi
- finely tuned, optimization hacks

Trust?

- Originally, only answer "yes" / "no" (more often, "maybe")
- More and more, produce at least proof traces (*i.e.* big steps)

Outline

- Introduction
- Ekstrakto
- SKonverto
- Conclusion

ATPs and Proofs

ATPs producing Dedukti proofs

- can trust the result
- not so efficient
- Zenon Modulo, ArchSAT, (iProverModulo)

Efficient ATPs

- coarse-grain proofs, i.e. TSTP
- checking such proofs?
- E, Vampire, ...

Directly outputting Dedukti proofs?

Provers can be hard to instrument to produce exact Dedukti proofs

- large piece of software
- developers not expert in $\lambda\Pi$-calculus modulo theory
- non stable and quite big proof calculus

Proof trace

But often, provers produce at least a proof trace:

- list of formulas that were derived to obtain the proof
- sometimes with more informations
  - premises
  - name of the inference rules
  - theory
  - ...

Example of trace: TSTP format

Output format of E, Vampire, Zipperposition, ...

List of formulas

- each annotated by an inference tree whose leafs are other formulas

```
cnf(c_0_60,plain,
    ( join(X1,join(X2,X3)) = join(X2,join(X1,X3)) ),
    inference(rw,[status(thm)],
      [inference(spm,[status(thm)],[c_0_30,c_0_18]),
       c_0_30])).
```

Example of trace: TSTP format

Output format of E, Vampire, Zipperposition, ...

List of formulas

- each annotated by an inference tree whose leafs are other formulas

```
cnf(c_0_60,plain,
    ( join(X1,join(X2,X3)) = join(X2,join(X1,X3)) ),
    inference(rw,[status(thm)],
      [inference(spm,[status(thm)],[c_0_30,c_0_18]),
       c_0_30])).
```

Independent of the proof calculus

Proof reconstruction

Use the content of the proof trace to reconstruct a Dedukti proof

Idea:

- Reprove each step using a Dedukti producing tool
- Combine the proofs of the steps to get a proof of the original formula

Try to be agnostic:

- w.r.t. the prover that produces the trace
- w.r.t. the prover that reprove the steps

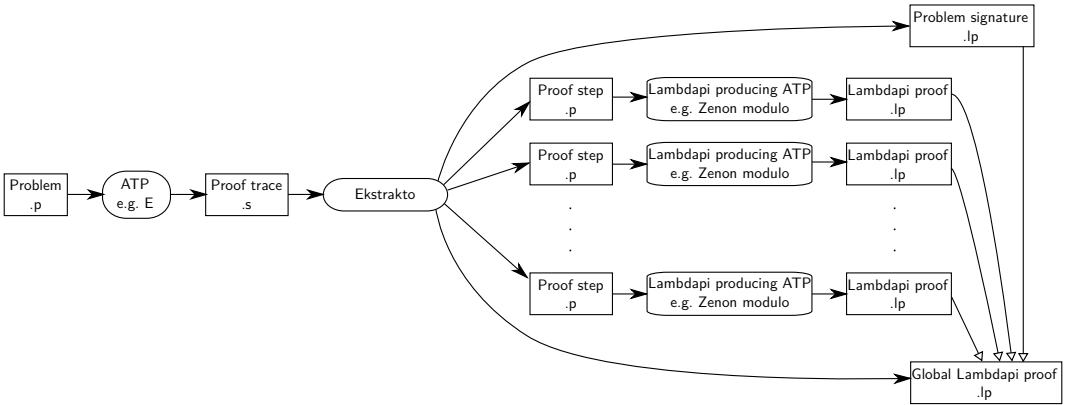
Ekstrakto

[El Haddad 2021]

- Input: TSTP proof trace
- Output: Reconstructed Lambdapi proof

`https://github.com/Deducteam/ekstrakto`

Ekstrakto architecture



Step problems

problem.s:

```
cnf(c_1, plain, A, ...).
cnf(c_2, plain, B, ...).
cnf(c_3, plain, C, inference(... c_1 ... c_2 ...)).
```

Step problems

problem.s:

cnf(c_1, plain, A, ...).
cnf(c_2, plain, B, ...).
cnf(c_3, plain, C, inference(... c_1 ... c_2 ...)).

lemmas/c_3.p:

fof(c_3, conjecture, A => B => C).

Step problems

problem.s:

cnf(c_1, plain, A, ...).
cnf(c_2, plain, B, ...).
cnf(c_3, plain, C, inference(... c_1 ... c_2 ...)).

lemmas/c_3.p:

fof(c_3, conjecture, A => B => C).

lemmas/c_3.lp:

symbol delta : Prf ($|A|$) → Prf ($|B|$) → Prf ($|C|$) :=
  ...

Recombining proof steps

proof_problem.lp:

```
symbol c_1 : Prf (|A|) := ...;
symbol c_2 : Prf (|B|) := ...;
symbol c_3 : Prf (|C|) := c_3.delta c_1 c_2;
```

Experimental evaluation

Benchmark:

- CNF problems of TPTP v7.4.0 (8118 files)

Trace producers:

- E and Vampire

Step provers:

- Zenon modulo and ArchSat

Results

Percentage of Lambdapi proofs on the extracted TPTP files

Prover	% **E**	% VAMPIRE
ZenonModulo	87%	60%
ArchSAT	92%	81%
ZenonModulo ∪ *ArchSAT*	95%	85%

Percentage of complete Lambdapi proofs

Prover	% **E** TSTP	% VAMPIRE TSTP
ZenonModulo	45%	54%
ArchSAT	56%	74%
ZenonModulo ∪ *ArchSAT*	69%	83%

Outline

- Introduction
- Ekstrakto
- SKonverto
- Conclusion

Non provable steps

Problem:

- some steps are not provable
  their conclusion is not a logical consequence of their premises
- OK because they preserve provability
- but Ekstrakto cannot work for them

Non provable steps

Problem:

- some steps are not provable
  their conclusion is not a logical consequence of their premises
- OK because they preserve provability
- but Ekstrakto cannot work for them

Main instance: Skolemization

$$\Gamma, \forall \vec{x}, \exists y, A[\vec{x}, y] \vdash B \text{ iff } \Gamma, \forall \vec{x}, A[\vec{x}, f(\vec{x})] \vdash B \text{ for a fresh } f$$

Present in the CNF transformation used by almost all ATPs

Skonverto

[El Haddad 2021]

Inputs:

- an axiom and its Skolemized version
- a Lambdapi proof using the latter

Output:

- a Lambdapi proof using the non-Skolemized axiom

Implementation of a constructive proof of Skolem theorem by [Dowek and Werner 2005]

- in the context of first-order natural deduction

```
symbol axiom : Prf (∀ (λ X, ∃ (λ Y, (p X (s Y)))));

symbol goal
  (ax_tran : Prf (∀ (λ X1 : El ι, ∀ (λ X2 : El ι, ∀ (λ X3 : El
     (p X1 X2) ⇒ ((p X2 X3) ⇒ (p X1 X3)))))))
  (ax_step : Prf (∀ (λ X1 : El ι, (p X1 (s (f X1))))))
  (ax_congr : Prf (∀ (λ X1 : El ι, ∀ (λ X2 : El ι,
     (p X1 X2) ⇒ (p (s X1) (s X2))))))
  (ax_goal : Prf (¬ (∃ (λ X4 : El ι, ((p a (s (s X4))))))))
  : Prf ⊥
≔  ax_goal  (∃I (λ X4 : El ι, p a (s (s X4))) (f (f a))
   (ax_tran a (s (f a)) (s (s (f (f a))))
     (ax_step a)
     (ax_congr (f a) (s (f (f a))) (ax_step (f a)))));
```

```
symbol goal
  (ax_tran : Prf (∀ (λ X1 : El ι, ∀ (λ X2 : El ι, ∀ (λ X3 : El
     (p X1 X2) ⇒ ((p X2 X3) ⇒ (p X1 X3)))))))
  (ax_step : Prf (∀ (λ X, ∃ (λ Y, (p X (s Y))))))
  (ax_congr : Prf (∀ (λ X1 : El ι, ∀ (λ X2 : El ι,
     (p X1 X2) ⇒ (p (s X1) (s X2))))))
  (ax_goal : Prf (¬ (∃ (λ X4 : El ι, ((p a (s (s X4))))))))
  : Prf ⊥
≔  ax_goal (λ r h, ∃E (λ z, p a (s z)) (ax_step a) r
       (λ z a1, ∃E (λ z0, p z (s z0)) (ax_step z) r
       (λ z0 a2, h z0 (ax_tran a (s z) (s (s z0)) a1
           (ax_congr z (s z0) a2)))));
```

Outline

Conclusion

Instrumenting a prover to produce Dedukti proofs

- good if you start your prover from scratch

Reconstructing proofs

- more adapted for existing provers
- cannot reconstruct all proofs
- also for proof assistants
  - PVS, Atelier B

More integration with the TPTP infrastructure

- use of GDV