Alethe: Towards a Generic SMT Proof Format

PxTP 2021

Hans-Jörg Schurr[1], Mathias Fleury[2], Haniel Barbosa[3], Pascal Fontaine[4]

[1]University of Lorraine, CNRS, Inria, and LORIA

[2]Johannes Kepler University Linz, Austria

[3]Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

[4]Université de Liège

July 11, 2021

Some History

Within veriT

- First: Ad-hoc (📜 TACAS 2006)
- Later: Redesigned (📜 PxTP 2011)
- Syntax changed over time

SMTCoq

- One of the first users
- Verified checker (📜 CPP 2011)
- Base for automation in Coq (📜 CAV 2017, PxTP -3h20)

Fine-Grained Proofs and Isabelle

- Support for reasoning with bound variables (📜 CADE 2017, JAR 2020)
- Typical for pre-processing in SMT
- Isabelle/HOL integration (📜 CADE +49h40)

Now!

💀 Proofonomicon

👼 Speculative Specification

🪶 It's now Alethe! (📜 PxTP -0h05)

Some History

Within veriT

- First: Ad-hoc (📜 TACAS 2006)
- Later: Redesigned (📜 PxTP 2011)
- Syntax changed over time

SMTCoq

- One of the first users
- Verified checker (📜 CPP 2011)
- Base for automation in Coq (📜 CAV 2017, PxTP -3h20)

Fine-Grained Proofs and Isabelle

- Support for reasoning with bound variables (📜 CADE 2017, JAR 2020)
- Typical for pre-processing in SMT
- Isabelle/HOL integration (📜 CADE +49h40)

Now!

💀 Proofonomicon

👼 Speculative Specification

🪶 It's now Alethe! (📜 PxTP -0h05)

Some History

Within veriT

- First: Ad-hoc (📜 TACAS 2006)
- Later: Redesigned (📜 PxTP 2011)
- Syntax changed over time

SMTCoq

- One of the first users
- Verified checker (📜 CPP 2011)
- Base for automation in Coq (📜 CAV 2017, PxTP -3h20)

Fine-Grained Proofs and Isabelle

- Support for reasoning with bound variables (📜 CADE 2017, JAR 2020)
- Typical for pre-processing in SMT
- Isabelle/HOL integration (📜 CADE +49h40)

Now!

💀 Proofonomicon

👼 Speculative Specification

🪶 It's now Alethe! (📜 PxTP -0h05)

Some History

Within veriT

- First: Ad-hoc (📜 TACAS 2006)
- Later: Redesigned (📜 PxTP 2011)
- Syntax changed over time

SMTCoq

- One of the first users
- Verified checker (📜 CPP 2011)
- Base for automation in Coq (📜 CAV 2017, PxTP -3h20)

Fine-Grained Proofs and Isabelle

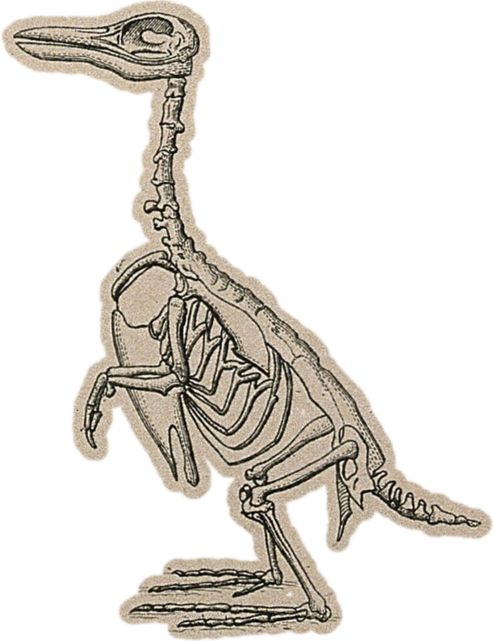
- Support for reasoning with bound variables (📜 CADE 2017, JAR 2020)
- Typical for pre-processing in SMT
- Isabelle/HOL integration (📜 CADE +49h40)

Now!

💀 Proofonomicon

👼 Speculative Specification

🪶 It’s now Alethe! (📜 PxTP -0h05)



Basic Structur

$$\frac{t_2}{t_3}$$

$$\vdots$$

$$\frac{t_1 \qquad \neg t_1}{\bot}\ \texttt{resolution}$$

```
(assume a0 t1)
(assume a1 t2)
(step s1  (cl t3)
        :premises (a1)     :rule rule1)
   ...
(step s20 (cl (not t1))
        :premises (s19)    :rule rule2)
(step s21 (cl )
        :premises (a0 s20) :rule resolution)
```

Subproofs With Assumptions

$$\dfrac{\dfrac{t_1}{t_2} \qquad \dfrac{\begin{matrix}[t_2]\\ \vdots \\ t_3\end{matrix}}{\neg t_2, t_3}\ \texttt{subproof}}{t_3}\ \texttt{resolution}$$

```
(assume a0 t1)
(step s1 (cl t2)
        :premises (a0)    :rule rule1)
(anchor :step s2)
  (assume s2.a1      t2)
   ...
  (step   s2.s10 (cl t3)
        :premises (s2.s9) :rule rule2)
(step s2 (cl (not t2) t3) :rule subproof)
(step s3 (cl t3)
        :premises (s1 s2) :rule resolution)
```

Reasoning With Binders

$$\dfrac{\dfrac{\dfrac{}{x \mapsto y \rhd x = y}\,\texttt{refl}}{x \mapsto y \rhd f(x) = f(y)}\,\texttt{cong}}{\forall x.\, f(x) = \forall y.\, f(y)}\,\texttt{bind}$$

```
(anchor :step s2 :args ((:= (x S) y)))
  (step s2.s1 (cl (= x y))    :rule refl)
  (step s2.s2 (cl (= (f x) (f y)))
                              :rule cong)
(step s2 (cl (= (forall ((x S)) (f x))
                (forall ((y S)) (f y)))
                              :rule bind)
```

Rules

Current State

- Overall 90 rules, mostly simple tautologies
- Seven categories – with overlaps
- Some historic overhead

🏗️ Cleanup and normalization

How can we accommodate different solvers?

- Some solvers might be able to use rules more strictly.
- Example:
  - $a = b \wedge b = c \rightarrow a = c$
  - $c = b \wedge a = b \rightarrow a = c$

💡 Have an optional annotation to mark restricted usage.

Rules

Current State

- Overall 90 rules, mostly simple tautologies
- Seven categories – with overlaps
- Some historic overhead
- 🏗️ Cleanup and normalization

How can we accommodate different solvers?

- Some solvers might be able to use rules more strictly.
- Example:
  - $a = b \wedge b = c \rightarrow a = c$
  - $c = b \wedge a = b \rightarrow a = c$
- 💡 Have an optional annotation to mark restricted usage.

Tools

A Checker and Elaborator

- “A second pair of eyes”.
- Small, independent codebase – in Rust.
- Long term: rewrite steps to their stricter form, framework to replace non-standard rules by standard rules.

👨 Bruno Andreotti

Support in *cvc5*

- Part of a wider effort to overhaul the proof module of *cvc5*.
- Will add more theories to Alethe.

👩 Hanna Lachnitt and the wider *cvc5* team.

Tools

A Checker and Elaborator

- “A second pair of eyes”.
- Small, independent codebase – in Rust.
- Long term: rewrite steps to their stricter form, framework to replace non-standard rules by standard rules.

👨 Bruno Andreotti

Support in *cvc5*

- Part of a wider effort to overhaul the proof module of *cvc5*.
- Will add more theories to Alethe.

👩 Hanna Lachnitt and the wider *cvc5* team.

Speculative Specification

http://www.verit-solver.org/alethe.pdf

Feedback

https://gitlab.uliege.be/verit/alethe