Zenon Modulo: When Achilles Outruns the Tortoise using Deduction Modulo

November 18, 2013

David Delahaye
David.Delahaye@cnam.fr

Cnam / Inria, CPR / Deducteam, Paris, France

GDR GPL, GT LTP, LaBRI, Bordeaux, France
Proof Search in Axiomatic Theories

Current Trends

- Axiomatic theories (Peano arithmetic, set theory, etc.);
- Decidable fragments (Presburger arithmetic, arrays, etc.);
- Applications of formal methods in industrial settings.

Place of the Axioms?

- Leave axioms wandering among the hypotheses?
- Induce a combinatorial explosion in the proof search space;
- Do not bear meaning usable by automated theorem provers.
A Solution

- A cutting-edge combination between:
  - First order automated theorem proving method (resolution);
  - Theory-specific decision procedures (SMT approach).

Drawbacks

- Specific decision procedure for each given theory;
- Decidability constraint over the theories;
- Lack of automatability and genericity.
Proof Search in Axiomatic Theories

Use of Deduction Modulo

- Transform axioms into rewrite rules;
- Turn proof search among the axioms into computations;
- Avoid unnecessary blowups in the proof search;
- Shrink the size of proofs (record only meaningful steps).

This Talk

- Introduce the principles of deduction modulo;
- Present the results of an experiment with Zenon;
- Give an overview of the BWare project.
Principles of Deduction Modulo

Inclusion

\[ \forall a \forall b ((a \subseteq b) \iff (\forall x (x \in a \Rightarrow x \in b))) \]

Proof in Sequent Calculus

\[ \frac{\ldots, x \in A \vdash A \subseteq A, x \in A}{\vdash A \subseteq A, x \in A} \quad \text{Ax} \]
\[ \frac{\ldots \vdash A \subseteq A, x \in A \Rightarrow x \in A}{\vdash A \subseteq A, \forall x (x \in A \Rightarrow x \in A)} \quad \forall \text{R} \]
\[ \frac{\vdash A \subseteq A, \forall x (x \in A \Rightarrow x \in A)}{\ldots, A \subseteq A \vdash A \subseteq A} \quad \text{Ax} \]
\[ \frac{\ldots, A \subseteq A \vdash A \subseteq A}{\ldots, (\forall x (x \in A \Rightarrow x \in A)) \Rightarrow A \subseteq A \vdash A \subseteq A} \quad \forall \text{L} \]
\[ \frac{A \subseteq A \iff (\forall x (x \in A \Rightarrow x \in A)) \vdash A \subseteq A}{\forall \text{L} \times 2} \]
\[ \forall a \forall b ((a \subseteq b) \iff (\forall x (x \in a \Rightarrow x \in b))) \vdash A \subseteq A \]
Principles of Deduction Modulo

Inclusion

\[ \forall a \forall b ((a \subseteq b) \rightarrow (\forall x (x \in a \Rightarrow x \in b))) \]

Rewrite Rule

\[ (a \subseteq b) \rightarrow (\forall x (x \in a \Rightarrow x \in b)) \]

Proof in Deduction Modulo

\[
\begin{align*}
& x \in A \vdash x \in A \\
\Rightarrow & x \in A \Rightarrow x \in A \\
\Rightarrow & A \subseteq A
\end{align*}
\]

Ax

\[ \forall \Rightarrow \]

\begin{align*}
& A \subseteq A \rightarrow \forall x (x \in A \Rightarrow x \in A)
\end{align*}

\[ \Rightarrow R \]

\[ \forall R, A \subseteq A \rightarrow \forall x (x \in A \Rightarrow x \in A) \]
From Axioms to Rewrite Rules

Difficulties

- Confluence and termination of the rewrite system;
- Preservation of the consistency;
- Preservation of the cut-free completeness;
- Automation of the transformation.

An Example

- Axiom $A \iff (A \Rightarrow B)$;
- Transformed into $A \rightarrow A \Rightarrow B$;
- We want to prove: $B$. 
From Axioms to Rewrite Rules

An Example (Continued)

- In sequent calculus, we have a cut-free proof:

\[
\begin{align*}
\sim \Pi & \\
& \frac{A \Rightarrow (A \Rightarrow B), A \vdash B, B}{\Rightarrow R} \\
& \frac{A \Rightarrow (A \Rightarrow B) \vdash B, A \Rightarrow B}{\Rightarrow L} \\
& \frac{A \Rightarrow (A \Rightarrow B), (A \Rightarrow B) \Rightarrow A \vdash B}{\leftrightarrow L} \\
& \frac{A \leftrightarrow (A \Rightarrow B) \vdash B}{\leftrightarrow L} \\
\end{align*}
\]

Where \( \Pi \) is:

\[
\begin{align*}
& \frac{A \vdash B, A}{\text{ax}} \\
& \frac{A \vdash B, A}{\text{ax}} \\
& \frac{A, B \vdash B}{\Rightarrow L} \\
\end{align*}
\]
From Axioms to Rewrite Rules

An Example (Continued)

In deduction modulo, we have to cut $A$ to get a proof:

$$
\frac{\Pi}{A \vdash B} \quad \frac{\Pi}{A \vdash B} \quad \Rightarrow R, \quad A \rightarrow A \Rightarrow B
$$

Where $\Pi$ is:

$$
\frac{\Pi}{A \vdash A} \quad \frac{\Pi}{A \vdash A} \quad \frac{\Pi}{A, B \vdash B} \quad \Rightarrow L, \quad A \rightarrow A \Rightarrow B
$$

$$
\frac{A \vdash A}{A \vdash B} \quad \frac{A, A \vdash B}{A \vdash B} \quad \text{cut}
$$
The Zenon Automated Theorem Prover

Features of Zenon

- First order logic with equality;
- Tableau-based proof search method;
- Extensible by adding new deductive rules;
- Certifying, 3 outputs: Coq, Isabelle, Dedukti;
- Used by other systems: Focalize, TLA.

Zenon

- Reference:
- Freely available (BSD license);
- Developed by D. Doligez;
- Download: http://focal.inria.fr/zenon/
The Zenon Automated Theorem Prover

The Tableau Method

- We start from the negation of the goal (no clausal form);
- We apply the rules in a top-down fashion;
- We build a tree whose each branch must be closed;
- When the tree is closed, we have a proof of the goal.

Closure and Cut Rules

\[
\begin{align*}
\frac{}{\bot} & \quad \frac{\neg \top}{\bot} \\
\frac{\neg R_r(t, t)}{\bot} & \quad \frac{P}{\neg P} \\
\frac{P}{\neg P} & \quad \frac{R_s(a, b)}{\neg R_s(b, a)}
\end{align*}
\]

\[\text{cut}\]
The Zenon Automated Theorem Prover

Analytic Rules

\[
\begin{align*}
\frac{\neg\neg P}{P} & \quad \alpha_{\neg
} \\
\frac{P \land Q}{P, Q} & \quad \alpha_{\land} \\
\frac{P \lor Q}{P | Q} & \quad \beta_{\lor} \\
\frac{\exists x \ P(x)}{P(\epsilon(x).P(x))} & \quad \delta_{\exists} \\
\frac{\neg\forall x \ P(x)}{\neg P(\epsilon(x).\neg P(x))} & \quad \delta_{\forall}
\end{align*}
\]

\[
\begin{align*}
\frac{P \iff Q}{\neg P, \neg Q | P, Q} & \quad \beta_{\iff} \\
\frac{\neg(P \iff Q)}{\neg P, Q | P, \neg Q} & \quad \beta_{\neg\iff} \\
\frac{\neg(P \lor Q)}{\neg P, \neg Q} & \quad \alpha_{\neg\lor} \\
\frac{\neg(P \land Q)}{\neg P | \neg Q} & \quad \beta_{\neg\land} \\
\frac{P \implies Q}{P, \neg Q} & \quad \alpha_{\implies} \\
\frac{P \implies Q}{\neg P | Q} & \quad \beta_{\implies}
\end{align*}
\]
The Zenon Automated Theorem Prover

\( \gamma \)-Rules

\[
\begin{align*}
\forall x \ P(x) & \quad \rightarrow \quad P(X) \\
\forall x \ P(x) & \quad \rightarrow \quad P(t)
\end{align*}
\]

Relational Rules

- Equality, reflexive, symmetric, transitive rules;
- Are not involved in the computation of superdeduction rules.
Example of Proof Search

\[ \forall x \ (P(x) \lor Q(x)), \ \neg P(a), \ \neg Q(a) \]
The Zenon Automated Theorem Prover

Example of Proof Search

\[ \forall x (P(x) \lor Q(x)), \neg P(a), \neg Q(a) \]

\[ \forall M P(X) \lor Q(X) \quad \gamma M \]
The Zenon Automated Theorem Prover

Example of Proof Search

\[ \forall x \ (P(x) \lor Q(x)) \land \neg P(a) \land \neg Q(a) \]

\[ \frac{P(X) \lor Q(X)}{P(X) \lor Q(X)} \]

\[ \beta \lor \forall M \]

\[ \forall x \ (P(x) \lor Q(x)) \land \neg P(a) \land \neg Q(a) \]

\[ \frac{P(X) \lor Q(X)}{P(X) \lor Q(X)} \]

\[ \beta \lor \forall M \]
The Zenon Automated Theorem Prover

Example of Proof Search

\[
\forall x \ (P(x) \lor Q(x)) \ , \ \neg P(a) \ , \ \neg Q(a) \\
\frac{P(X) \lor Q(X)}{P(X) \lor Q(X)} \quad \beta \lor \\
\frac{P(X)}{Q(X)}
\]

\(\forall M\)
Example of Proof Search

\[\forall x (P(x) \lor Q(x)), \neg P(a), \neg Q(a)\]

\[\frac{P(X) \lor Q(X)}{P(a) \lor Q(a)} \gamma_{\forall M}
\]

\[\frac{P(X)}{Q(X)} \beta_{\lor}
\]

\[\frac{P(a)}{Q(X)} \gamma_{\forall inst}
\]
The Zenon Automated Theorem Prover

Example of Proof Search

\[
\forall x \ (P(x) \lor Q(x)), \ \neg P(a), \ \neg Q(a) \quad \gamma_{\forall M}
\]

\[
P(X) \lor Q(X) \\
\beta_{\lor}
\]

\[
P(X) \\
\gamma_{\forall \text{inst}}
\]

\[
Q(X) \\
\beta_{\lor}
\]

\[
P(a) \lor Q(a) \\
\gamma_{\forall \text{inst}}
\]

\[
P(a) \quad Q(a) \\
\beta_{\lor}
\]
Example of Proof Search

\[ \forall x (P(x) \lor Q(x)), \neg P(a), \neg Q(a) \quad \gamma_{\forall M} \]

\[ \frac{P(X) \lor Q(X)}{P(X)} \quad \beta_{\lor} \]

\[ \frac{P(X)}{P(a) \lor Q(a)} \quad \gamma_{\forall \text{inst}} \]

\[ \frac{P(a)}{\circ} \quad \frac{Q(a)}{\circ} \quad \beta_{\lor} \]
The Zenon Automated Theorem Prover

Example of Proof Search

\[
\forall x (P(x) \lor Q(x)) , \neg P(a) , \neg Q(a) \quad \gamma_{\forall M}
\]

\[
P(X) \lor Q(X) \quad \beta_{\lor}
\]

\[
P(X) \quad \gamma_{\forall \text{inst}}
\]

\[
P(a) \lor Q(a) \quad \beta_{\lor}
\]

\[
P(a) \quad \bigstar
\]

\[
Q(a) \quad \bigstar
\]
The Zenon Automated Theorem Prover

Example of Proof Search

\[
\forall x \ (P(x) \lor Q(x)) \quad \neg P(a) \quad \neg Q(a)
\]

\[
\forall \exists \quad P(X) \lor Q(X)
\]

\[
\forall \exists \quad P(a) \lor Q(a)
\]

\[
\forall \exists \quad P(a) \quad \exists \quad Q(a)
\]
The Zenon Automated Theorem Prover

Example of Proof Search

\[ \forall x \ (P(x) \lor Q(x)) , \neg P(a) , \neg Q(a) \]
\[ \frac{P(a) \lor Q(a)}{P(a) \quad Q(a)} \]
\[ \frac{\gamma \forall \text{inst}}{} \]

\[ \beta \lor \]

\[ P(a) \quad Q(a) \]

\[ \bullet \quad \bullet \]

\[ \epsilon \]
Integrating Deduction Modulo to Zenon

Goals

- Improve the proof search in axiomatic theories;
- Reduce the proof size;
- New tool: Zenon + Deduction Modulo = Zenon Modulo!

Compared to Super Zenon

- Extension of Zenon to superdeduction;
- Superdeduction: variant of deduction modulo;
- Freely available (GPL license);
- Collaboration Cnam and Siemens;
- Download: http://cedric.cnam.fr/~delahaye/super-zenon/
Integrating Deduction Modulo to Zenon

Goals

► Improve the proof search in axiomatic theories;
► Reduce the proof size;
► New tool: Zenon + Deduction Modulo = Zenon Modulo!

Compared to Super Zenon

► Extension of Zenon to superdeduction;
► Superdeduction: variant of deduction modulo;
► Freely available (GPL license);
► Collaboration Cnam and Siemens;
► Reference:

Integrating Deduction Modulo to Zenon

Goals

- Improve the proof search in axiomatic theories;
- Reduce the proof size;
- New tool: Zenon + Deduction Modulo = Zenon Modulo!

Compared to Super Zenon

- Compare deduction modulo and superdduction in practice;
- Rewrite rules over propositions and terms;
- Normalization strategies (efficiency);
- Light integration (metavariable management);
- No trace of computation in the proofs.
Class Rewrite System

Definition
A class rewrite system is a pair consisting of:

- $\mathcal{R}$: a set of proposition rewrite rules;
- $\mathcal{E}$: a set of term rewrite rules (and equational axioms).

Rewrite Rules

- Proposition rewrite rule: $l \rightarrow r$, where $l$ is an atomic proposition and $FV(r) \subseteq FV(l)$;
- Term rewrite rule: $l \rightarrow r$, where $FV(r) \subseteq FV(l)$.

Congruence

- $=_\mathcal{R}\mathcal{E} \equiv$ congruence generated by the set $\mathcal{R} \cup \mathcal{E}$. 

Rules of Zenon Modulo

Closure and Cut Rules

\[ \frac{P}{\neg Q} \quad \bullet \text{ if } P =_R \bot \]

\[ \frac{P}{\neg P} \quad \bullet \text{ if } P =_R \bot \]

\[ \frac{P}{\neg P} \quad \bullet \text{ if } P =_R \bot \]

\[ \frac{P}{\neg Q} \quad \bullet \text{ if } P =_R \bot \]

\[ \frac{P}{\neg P} \quad \bullet \text{ if } P =_R \bot \]

Where \( R_r \) is a reflexive relation, and \( R_s \) a symmetric relation.
Rules of Zenon Modulo

\(\alpha/\beta\)-Rules

\[\frac{\neg S}{P} \alpha_{\neg \iff} \text{ if } S \Rightarrow P \\]

\[\frac{S}{P, Q} \alpha_{\land} \text{ if } S \Leftarrow P \land Q \\]

\[\frac{S}{P \mid Q} \beta_{\lor} \text{ if } S \Leftarrow P \lor Q \\]

\[\frac{S}{\neg P \mid Q} \beta_{\Rightarrow} \text{ if } S \Leftarrow P \Rightarrow Q \\]

\[\frac{S}{\neg P, \neg Q \mid P, Q} \beta_{\Leftarrow} \text{ if } S \Leftarrow P \Leftarrow Q \\]

\[\frac{\neg S}{\neg P \mid \neg Q} \beta_{\Leftarrow} \text{ if } S \Leftarrow P \Leftarrow Q \\]
**Rules of Zenon Modulo**

**δ/γ-Rules**

\[
\frac{S}{P(\epsilon(x).P(x))} \quad \delta_\exists \text{ if } S = \text{Ref} \exists x P(x)
\]

\[
\frac{\neg S}{\neg P(\epsilon(x).\neg P(x))} \quad \delta_\forall \text{ if } S = \text{Ref} \forall x P(x)
\]

\[
\frac{S}{P(X)} \quad \gamma_{\forall M} \text{ if } S = \text{Ref} \forall x P(x)
\]

\[
\frac{\neg S}{\neg P(X)} \quad \gamma_{\exists M} \text{ if } S = \text{Ref} \exists x P(x)
\]

\[
\frac{S}{P(t)} \quad \gamma_{\forall \text{inst}} \text{ if } S = \text{Ref} \forall x P(x)
\]

\[
\frac{\neg S}{\neg P(t)} \quad \gamma_{\exists \text{inst}} \text{ if } S = \text{Ref} \exists x P(x)
\]
Example of Proof

Example with the Set Inclusion

- With regular rules of Zenon:

\[
\forall a \forall b ((a \subseteq b) \iff (\forall x (x \in a \Rightarrow x \in b))), \ A \not\subseteq A \\
(X \subseteq Y) \iff (\forall x (x \in X \Rightarrow x \in Y)) \\
X \subseteq Y, \ \forall x (x \in X \Rightarrow x \in Y) \\
(A \subseteq A) \iff (\forall x (x \in A \Rightarrow x \in A)) \\
A \subseteq A, \ \forall x (x \in A \Rightarrow x \in A)
\]

Where \( \Pi \) is:

\[
A \not\subseteq A, \ \neg \forall x (x \in A \Rightarrow x \in A) \\
\neg(\epsilon_x \in A \Rightarrow \epsilon_x \in A) \\
\epsilon_x \in A, \ \epsilon_x \not\in A
\]

with \( \epsilon_x = \epsilon(x). \neg(x \in A \Rightarrow x \in A) \)
Example of Proof

Example with the Set Inclusion

- With regular rules of Zenon:

\[
\forall a \forall b \ ((a \subseteq b) \iff (\forall x \ (x \in a \Rightarrow x \in b))), \ A \not\subseteq A
\]
\[
\begin{align*}
(A \subseteq A) & \iff (\forall x \ (x \in A \Rightarrow x \in A)) \\
A \subseteq A, \ \forall x \ (x \in A \Rightarrow x \in A) & \quad \beta \iff
\end{align*}
\]

Where \( \Pi \) is:

\[
A \not\subseteq A, \ \neg \forall x \ (x \in A \Rightarrow x \in A)
\]
\[
\neg (\epsilon_x \in A \Rightarrow \epsilon_x \in A)
\]
\[
\epsilon_x \in A, \ \epsilon_x \not\in A
\]

with \( \epsilon_x = \epsilon(x). \neg (x \in A \Rightarrow x \in A) \)
Example of Proof

Example with the Set Inclusion

- With the rules of Zenon Modulo:

\[
\begin{align*}
A & \not\subseteq A \\
\neg \forall x \ (x \in A \Rightarrow x \in A) & \Rightarrow A \subseteq A \rightarrow \forall x \ (x \in A \Rightarrow x \in A) \\
\neg (\epsilon_x \in A \Rightarrow \epsilon_x \in A) & \Rightarrow (\epsilon_x \in A, \epsilon_x \not\in A) \\
\epsilon_x & \in A, \epsilon_x \not\in A \\
\end{align*}
\]

with \( \epsilon_x = \epsilon(x). \neg(x \in A \Rightarrow x \in A) \)
Example of Proof

Example with the Set Inclusion

- With the rules of Zenon Modulo:

\[
\begin{align*}
A \not\subseteq A \\
\neg (\epsilon_x \in A \Rightarrow \epsilon_x \in A) \\
\epsilon_x \in A, \epsilon_x \not\in A \\
\end{align*}
\]

\[
\delta \forall x, A \subseteq A = \forall x (x \in A \Rightarrow x \in A)
\]

\[
\alpha \Rightarrow
\]

with \(\epsilon_x = \epsilon(x)\). \(\neg (x \in A \Rightarrow x \in A)\)
Zenon Modulo over the TPTP Library

For any First Order Theory

- Automated orientation of the theories;
- Not oriented axioms left as axioms.

Heuristic

- \( \forall \vec{x} \ (P \iff \varphi) \): \( P \longrightarrow \varphi \) is generated if \( \text{FV}(\varphi) \subseteq \text{FV}(P) \);
  Otherwise if \( \varphi \) literal and \( \text{FV}(P) \subset \text{FV}(\varphi) \) then apply heuristic to \( \forall \vec{x} \ (\varphi \iff P) \);

- \( \forall \vec{x} \ (\neg P \iff \varphi) \): \( P \longrightarrow \neg \varphi \) is generated if \( \text{FV}(\varphi) \subseteq \text{FV}(P) \);
  Otherwise if \( \varphi \) literal and \( \text{FV}(P) \subset \text{FV}(\varphi) \) then apply heuristic to \( \forall \vec{x} \ (\varphi \iff \neg P) \);

- \( \forall \vec{x} \ (s = t) \): \( s \longrightarrow t \) is generated if \( \text{FV}(t) \subseteq \text{FV}(s) \);
  Otherwise \( t \longrightarrow s \) if \( \text{FV}(s) \subset \text{FV}(t) \);
  In addition, commutativity axioms are excluded.
## Experimental Results

### Figures

<table>
<thead>
<tr>
<th>TPTP Category</th>
<th>Zenon</th>
<th>Zenon Mod. (Prop. Rew.)</th>
<th>Zenon Mod. (Term/Prop. Rew.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOF 6,659 prob.</td>
<td>1,586</td>
<td>1,626 (2.5%)</td>
<td>1,616 (1.9%)</td>
</tr>
<tr>
<td>SET 462 prob.</td>
<td>149</td>
<td>219 (47%)</td>
<td>222 (49%)</td>
</tr>
</tbody>
</table>

- TPTP Library v5.5.0;
- Intel Xeon X5650 2.67GHz;
- Timeout 300 s, memory limit 1 GB.
Experimental Results

Figures

<table>
<thead>
<tr>
<th>TPTP Category</th>
<th>Zenon</th>
<th>Zenon Mod. (Prop. Rew.)</th>
<th>Zenon Mod. (Term/Prop. Rew.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOF 6,659 prob.</td>
<td>1,586</td>
<td>1,626 (2.5%)</td>
<td>1,616 (1.9%)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+114 (7.2%)</td>
<td>+170 (10.7%)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-74 (4.7%)</td>
<td>-140 (8.8%)</td>
</tr>
<tr>
<td>SET 462 prob.</td>
<td>149</td>
<td>219 (47%)</td>
<td>222 (49%)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+78 (52.3%)</td>
<td>+86 (57.7%)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-8 (5.4%)</td>
<td>-13 (8.7%)</td>
</tr>
</tbody>
</table>

- 29 difficult problems (TPTP ranking);
- 29 with a ranking $\geq 0.7$;
- 9 with a ranking $\geq 0.8$;
- 1 with a ranking $\geq 0.9$. 
Proof Compression

Experiment

- 1,446 problems proved by both Zenon and Zenon Modulo;
- 624 FOF problems and 110 SET problems;
- Subset of proofs where rewriting occurs;
- Measure: number of proof nodes of the resulting proof.

Figures

<table>
<thead>
<tr>
<th>TPTP Category</th>
<th>Average Reduction</th>
<th>Maximum Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOF 624 problems</td>
<td>6.8%</td>
<td>91.4%</td>
</tr>
<tr>
<td>SET 110 problems</td>
<td>21.6%</td>
<td>84.6%</td>
</tr>
</tbody>
</table>

Experiment shown:
- 1,446 problems proved by both Zenon and Zenon Modulo;
- 624 FOF problems and 110 SET problems;
- Subset of proofs where rewriting occurs;
- Measure: number of proof nodes of the resulting proof.

<table>
<thead>
<tr>
<th>Figure</th>
<th>TPTP Category</th>
<th>Average Reduction</th>
<th>Maximum Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOF</td>
<td>624 problems</td>
<td>6.8%</td>
<td>91.4%</td>
</tr>
<tr>
<td>SET</td>
<td>110 problems</td>
<td>21.6%</td>
<td>84.6%</td>
</tr>
</tbody>
</table>
Proof Compression

Figures

![Graph showing average reduction with Zenon Modulo (Percent) vs. Zenon Proof Size (Min-Max Proof Nodes FOF/SET)]

- **FOF**
- **SET**

Zenon Proof Size ([Min-Max] Proof Nodes FOF/SET)

- [3-6]/[7-10]
- [6-8]/[10-13]
- [8-11]/[13-18]
- [11-16]/[18-22]
- [16-21]/[22-27]
- [21-28]/[27-31]
- [29-38]/[31-34]
- [39-68]/[36-53]
- [70-3474]/[54-132]
A Backend for Zenon Modulo

Using the Existing Backends

- Create special inference nodes for rewriting rules;
- Record rewrite steps in the proof traces;
- Extend the existing backends of Zenon;
- Prove the rewriting lemmas in Coq and Isabelle.

Problems of this Approach

- Possible large number of rewrite steps to record;
- May Lead to memory explosion;
- Against the Poincaré principle;
- Loss of deduction modulo benefits.
Using the Dedukti Universal Proof Checker

Features of Dedukti

- Universal proof checker for the $\lambda\Pi$-calculus modulo;
- Propositions/types and proofs/$\lambda$-terms (Curry-Howard);
- Native support of rewriting;
- Only need to provide the set of rewrite rules.

Dedukti

- Freely available (CeCILL-B license);
- Developed by Deducteam;
- Download: https://www rocq.inria.fr/deducteam/Dedukti/
Using the Dedukti Universal Proof Checker

From Zenon Modulo Proofs to Dedukti

- From classical to intuitionistic logic;
- Based on a double-negation translation;
- Optimized to minimize the number of double-negations;
- 54% of the TPTP proofs already intuitionistic.

Dedukti

- Freely available (CeCILL-B license);
- Developed by Deducteam;
- Download: 
  https://www rocq inria fr/deducteam/Dedukti/
Experimental Results over the TPTP Library

Figures

<table>
<thead>
<tr>
<th>FOF 624 prob.</th>
<th>Dedukti Success</th>
<th>Dedukti Failure</th>
<th>Backend Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problems</td>
<td>559</td>
<td>5</td>
<td>60</td>
</tr>
<tr>
<td>Rate</td>
<td>89.6%</td>
<td>0.8%</td>
<td>9.6%</td>
</tr>
</tbody>
</table>

Failures

- Dedukti: rewrite system (termination, confluence, etc.);
- Backend: minimization of the double-negations.
References for Zenon Modulo

Rules, Results, and Backend

- LPAR’13 paper:

Proof Certification and Compression

- IWIL’13 paper:
The BWare Project

The Project

- INS prog. of the French National Research Agency (ANR);
- Academics: Cnam, LRI, Inria;
- Companies: Mitsubishi, ClearSy, OCamlPro.

Goals

- Mechanized framework for automated verification of B PO;
- Generic platform (several automated deduction tools);
- First order tools and SMT solvers;
- Production of proof objects (certificates).
Extending Zenon to Deduction Modulo
David Delahaye

Introduction
Principles of Deduction Modulo
Overview of the Zenon ATP
Deduction Modulo for Zenon
Zenon Modulo over the TPTP Library
A Backend for Zenon Modulo
References for Zenon Modulo

Deduction Modulo for BWare

Conclusion
Deduction Modulo in the BWare Project

Tools

- Super Zenon, Zenon Modulo (extensions of Zenon);
- iProver Modulo (extension of iProver);
- Backend for these tools: Dedukti.

Adequacy of the Tools

- Build a B set theory modulo (manually);
- Comprehension scheme (higher order) hard-coded;
- Good results of Super Zenon for B proof rules;
- Good results of Zenon Modulo in the SET category of TPTP.
Conclusion

Deduction Modulo in Automated Tools

- Resolution: iProver Modulo (based on iProver);
- Tableaux: Super Zenon, Zenon Modulo (based on Zenon);
- Appropriate backend: Dedukti ($\lambda\Pi$-calculus modulo).

Experimental Results

- Performances increased for generic benchmarks (TPTP);
- Successful use in industrial settings (B method):
  - Collaboration Cnam/Siemens: verification of B proof rules;
  - BWare project: verification of B PO (work in progress).
Automated Deduction

Automated Generation of Theories Modulo

- Generation of theories modulo “on the fly”;
- Preservation of “good” properties (cut-free completeness);
- Difficulties for term rewrite rules (heuristics);
- Use of external tools to study the rewrite system;
- Integration of the equational axioms (rewriting modulo).

Set Theory Modulo

- Good experimental results for set theory;
- Results of Super Zenon (B), Zenon Modulo (TPTP);
- Ability to prove difficult problems in this domain;
- Promising for the BWare project;
- Problem of large formulas, large contexts (PO).
Proof Checking

Proof Checking for Automated Tools

- $\lambda\Pi$-calculus modulo appropriate to encode theories;
- Suitable framework to certify deduction modulo proofs;
- High quality proof certificates (size in particular);
- Dedukti as a backend for several automated tools:
  - Zenon Modulo (extension of Zenon);
  - iProver Modulo (extension of iProver).

Interoperability between Proof Systems

- Shallow embeddings of theories;
- Dedukti embeddings:
  - CoqInE (from Coq);
  - Holide (from HOL);
  - Focalide (from Focalize).