Automated certification of termination proofs

Frédéric Blanqui

INRIA & LORIA

TYPES’07
Outline

Extensions of proof assistant kernels

CoLoR: a Coq Library on Rewriting and termination

Rainbow: a termination proof certifier
Proof assistant kernels

\[(\text{conv}) \quad \frac{\Gamma \vdash t : T \quad T \simeq_{\Gamma} T'}{\Gamma \vdash t : T'}\]

- type checking
- type equivalence checking
  \[\Rightarrow \simeq_{\Gamma} \text{ must be decidable}\]
Equivalence on types/propositions

bigger is $\simeq_\Gamma$:

- bigger can be the deduction steps
  - closer are we to the mathematical practice
  - less uninteresting technical lemmas are needed
  - faster are developments

- smaller can be the proofs
  - more proofs can be type-checked

- more terms can be typable
  - easier are developments with dependent types

- closer are we to undecidability
Extensions of the Calculus of Constructions

<table>
<thead>
<tr>
<th>$\simeq_\Gamma$ decidable</th>
<th>$\simeq_\Gamma$ not decidable</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta$ (CC)</td>
<td>$\beta_\iota$ (CIC)</td>
</tr>
</tbody>
</table>
The Calculus of Constructions with extensionality
Hofmann [95] Oury [05]

\[
\text{(beta)} \quad \frac{T \downarrow^\beta U}{T \simeq_\Gamma U}
\]

\[
\text{(ext)} \quad \frac{\Gamma \vdash p : T = U}{T \simeq_\Gamma U}
\]
The Open Calculus of Constructions (OCC)
Stehr [02]

type checking \[ \Gamma \vdash t : T \]
type inference \[ \Gamma \vdash t \rightarrow: T \]
computational equality \[ \Gamma \vdash \text{!!} t = u \] (using rewriting)
structural equality \[ \Gamma \vdash \| t = u \] (same term representation)
provability \[ \Gamma \vdash \text{??} T \] (using logic programming)

(Red) \[ \begin{array}{c} \Gamma \vdash z \rightarrow: \text{!!} \forall x : V, P \Rightarrow t = u \\ \Gamma \vdash v : V \end{array} \]
\[ \Gamma \vdash \text{??} P^v_x \]
\[ \Gamma \vdash \text{!!}(t = u)^v_x \]

implemented in first-order rewriting logic (Maude)
taking $\simeq_\Gamma = \downarrow_\beta R$ with a set $R$ of higher-order rewrite rules includes:

- CIC: $\nu$-reduction is an orthogonal constructor-based HORS
- functions with non-structurally smaller recursive calls
- (non-linear) matching on various arguments at the same time
- matching on defined symbols: $x + (y + z) \rightarrow (x + y) + z$
- non-free data types: $s(p(x)) \rightarrow x$ and $p(s(x)) \rightarrow x$ (integers)
- decision procedures like the `ring` tactic in Coq by applying rules with matching modulo associativity and commutativity

prototype of Coq+R using CiME (SVN branch “recriture”, B. [03])
Termination of $\rightarrow_\beta^R$?

- when $R$ is any terminating first-order rewrite system
  (Breazu-Tannen Okada [89] Barbanera [90] Dougherty [91])

- general schema / computability closure
  (B. Jouannaud Okada [99] B. [01-05])

- HORPO
  (Jouannaud Rubio [99] Walukiewicz [03])

- size-based termination
  (Xi [01] Abel [03-06] Barthe+ [04-06] B. [04-05] B. Riba [06])
  (prototype for CIC by Grégoire [05])
\( \simeq^\Gamma \) includes the equational theory generated by the closed equational hypotheses of \( \Gamma \):

\[
\begin{align*}
  h_1 &: \ x = y + 2, \\
  h_2 &: \ f(x) = g(x - 2) \vdash refl\_equal \ f(x) : f(x) = g(y)
\end{align*}
\]

Two possible implementations:

- using rewriting since, by Knuth-Bendix completion, equational hypotheses can be turned into a terminating confluent system
- using an external specialized tool for congruence closure

Prototype implemented in Maude (Strub [03])
\[ \simeq_{\Gamma} \text{ includes the equational theory generated by the } \textit{closed} \text{ equational hypotheses of } \Gamma \text{ and } \textit{any Nelson-Oppen combination of stably infinite theories} \text{ (linear arithmetic, lists, arrays, etc.)} \]

\[ \Rightarrow \text{ implementation using an external specialized tool like haRVey, Ergo, Simplify, Omega, etc.} \]
Going further?

...take into account *quantified* equational hypotheses present in \( \Gamma \):

\[
N : \text{Type}, \ 0 : N, \ s : N \Rightarrow N, \ + : N \Rightarrow N \Rightarrow N,
+_{0} : \forall x : N, x + 0 = x,
+_{s} : \forall xy : N, x + (sy) = s(x + y),
+_{c} : \forall xy : N, x + y = y + x,
+_{a} : \forall xyz : N, x + (y + z) = (x + y) + z, \ldots
h_{1} : x = y + 2, \ h_{2} : f(x) = g(x − 2) \vdash \text{refl\_equal} \ f(x) : f(x) = g(y)
\]

⇒ application in the definition of functors

⇒ requires *Knuth-Bendix completion*
Conclusion

replacing in kernels \( \iota \) by \( R \) has many advantages but:

▶ how to check termination, confluence, etc. ?
   ⇒ use available automated termination provers
   like TPA, AProVE, TTT, CiME, etc.

▶ how to be sure that these tools are correct ?
   ⇒ certify their source code ?
   difficult, long, must be done every time the code is changed

▶ make them provide a certificate and certify the certificates !
   ⇒ certify termination criteria used by the tools
   ⇒ provide tactics checking criteria conditions
   ⇒ provide a program generating proofs from certificates

\[
\begin{array}{c}
R \rightarrow \text{Termin. Prover} \\
\rightarrow \text{cert} \\
\rightarrow \text{Prf Generator + PA Library} \\
\rightarrow \text{prf} \\
\rightarrow \text{Proof Assistant} \\
\rightarrow \text{y/n}
\end{array}
\]
Outline

Extensions of proof assistant kernels

CoLoR: a Coq Library on Rewriting and termination

Rainbow: a termination proof certifier
CoLoR: a Coq Library on Rewriting and termination

- project started in March 2004, first release in March 2005
- contributors: B., Coupet-Grimal, Delobel, Hinderer, Koprowski, Le Roux
- free download on http://color.loria.fr/
- publications: JFLA'06, WST'06, RTA'06
CoLoR content (1/2)

- Mathematical structures:
  - relations
  - semi-rings

- Data structures:
  - lists
  - vectors
  - integer polynomials with multiple variables
  - finite multisets
  - matrices

- Term structures:
  - algebraic terms with symbols of fixed arity
  - varyadic terms
  - simply typed lambda-terms
CoLoR content (2/2)

- Transformation techniques:
  - conversion from algebraic to varyadic terms
  - arguments filtering
  - rule elimination
  - dependency pairs

- Termination criteria:
  - polynomial interpretations
  - multiset ordering
  - recursive path ordering
  - higher-order recursive path ordering
  - dependency graph cycles
  - matrix interpretations
CoLoR figures

- 41,400 lines of Coq:
  - half of the size of Coq standard library
  - 5% of Coq contribs
  - Mathematical structures: 10%
  - Data structures: 29%
  - Term structures: 44%
  - Termination criteria: 17%
- 36 inductive types or predicates (Inductive)
- 115 recursive functions (Fixpoint)
- 533 non-recursive definitions (Definition)
- 2149 lemmas and theorems (Lemma)
Outline

Extensions of proof assistant kernels

CoLoR: a Coq Library on Rewriting and termination

Rainbow: a termination proof certifier

Rainbow: a termination proof certifier

Termination proof grammar:

\[
\text{proof} = \text{MANNA\_NESS red\_ord} \mid \text{DP proof} \mid \ldots \\
\text{red\_ord} = \text{POLY\_INT poly\_int} \mid \text{RPO prec status} \\
\quad \mid \text{LEX red\_ord+} \mid \ldots \\
\text{poly\_int} = \text{symbol polynom poly\_int} \\
\text{polynom} = \ldots
\]
XML Schema for termination proofs

```xml
<complexType name="ReductionOrdering">
  <choice>
    <element name="rpo" type="prf:RPO"/>
    <element name="poly_int" type="prf:PolynomialInterpretation">
      <element name="lex" type="prf:LexicographicCombination"/>
    </element>
  </choice>
</complexType>

<complexType name="Proof">
  <choice>
    <element name="manna_ness" type="prf:ReductionOrdering"/>
  </choice>
</complexType>

<element name="proof" type="prf:Proof"/>
```
Example of termination problem

<?xml version="1.0" encoding="ISO-8859-1"?>

<problem
    xmlns="urn:rainbow.problem.format"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:rainbow.problem.format problem.xsd">

    <algebra>
        <signature>
            <mapping><fun>plus</fun><arity>2</arity></mapping>
            <mapping><fun>succ</fun><arity>1</arity></mapping>
            <mapping><fun>zero</fun><arity>0</arity></mapping>
        </signature>
    </algebra>

    <theory/>

Frédéric Blanqui  Automated certification of termination proofs
<strategy><none/></strategy>

<rules>
  <rule> <!-- plus zero v0 = v0 -->
    <lhs>
      <app>
        <fun>plus</fun>
        <arg><app><fun>zero</fun></app></arg>
        <arg><var>0</var></arg>
      </app>
    </lhs>
    <rhs>
      <var>0</var>
    </rhs>
  </rule>
  ...
</rules>

</problem>
Example of termination proof

<?xml version="1.0" encoding="ISO-8859-1"?>

<proof
    xmlns="urn:rainbow.proof.format"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:rainbow.proof.format proof.xsd">
    <manna_ness>
        <poly_int>
            <mapping>
                <fun>plus</fun>
                <polynomial>
                    <!-- 2.x_1 + 1.x_2 + 2 -->
                    <monomial>
                        <coef>2</coef>
                        <var>1</var>
                        <var>0</var>
                    </monomial>
                    <monomial>
                        <coef>1</coef>
                        <var>0</var>
                        <var>1</var>
                    </monomial>
                    <monomial>
                        <coef>2</coef>
                        <var>0</var>
                        <var>0</var>
                    </monomial>
            </polynomial>
        </mapping>
    </poly_int>
</manna_ness>
</proof>
Inductive symbol : Set :=
  | plus : symbol
  | succ : symbol
  | zero : symbol.

Lemma eq_symbol_dec : forall f g : symbol, {f=g}+{~f=g}.
Proof. decide equality. Qed.

Definition arity (s : symbol) :=
  match s with
  | plus => 2
  | succ => 1
  | zero => 0
  end.
Require Import ASignature.

Definition sig := mkSignature arity eq_symbol_dec.
Definition S_plus x2 x1 := (@Fun sig plus (Vcons x2 (Vcons x1 Vnil)));
Definition S_succ x1 := (@Fun sig succ (Vcons x1 Vnil));
Definition S_zero := (@Fun sig zero Vnil).

Require Import ATrs.

Definition rules :=
    mkRule (S_plus S_zero (Var 0))
    (Var 0)
:: mkRule (S_plus (S_succ (Var 1)) (Var 0))
    (S_succ (S_plus (Var 1) (Var 0)))
:: nil.
Require Import Polynom.

Definition poly (f : symbol) : poly (arity f) :=
  match f as f return poly (arity f) with
  | plus => (2%Z, (Vcons 1 (Vcons 0 Vnil)))
      :: (1%Z, (Vcons 0 (Vcons 1 Vnil)))
      :: (2%Z, (Vcons 0 (Vcons 0 Vnil)))
      :: nil
  | succ => (1%Z, (Vcons 1 Vnil))
      :: (1%Z, (Vcons 0 Vnil))
      :: nil
  | zero => (1%Z, Vnil)
      :: nil
  end.
Require Import MonotonePolynom.

Lemma monotony : forall f, pmonotone (poly f).

Proof.
pmonotone.
Qed.

Require Import APolyInt.

Definition PI := mkPolyInterpretation sig poly monotony.

Lemma termination : wf (red rules).

Proof.
poly_int PI.
Qed.
(* check that, for all l->r in R, all the coefficients of the polynomial \([l]-[r]+1\) are \(\geq 0\) *)

Lemma polyInterpretationTermination : forall R,
    lforall (fun lr => coef_pos (rulePoly_gt lr)) R -> WF (red R).

Proof. ... Qed.

Ltac poly_int PI :=
    match goal with
    |- WF (red ?R) =>
        apply (polyInterpretationTermination PI R);
        vm_compute; intuition; discriminate
    end.
Towards a certified termination competition

annual international termination competition (deadline May 21)
http://www.lri.fr/~marche/termination-competition/
categories: rewrite systems, logic programs, functional programs

new option this year: certified proofs!

in 2006, $3/11 = 27\%$ tools were discovered to be bugged
Current results

competition database: 1875 problems (some are open!)
in standard rewriting: 865

<table>
<thead>
<tr>
<th>certified</th>
<th>TPA</th>
<th>19%</th>
<th>with polynomial interpretations</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Rainbow</td>
<td>27%</td>
<td>with matrix interpretations</td>
<td></td>
</tr>
<tr>
<td>+ CoLoR</td>
<td>32%</td>
<td>with both</td>
<td></td>
</tr>
<tr>
<td>not certified</td>
<td>Jambox</td>
<td>62%</td>
<td>matrix interp + dependency pairs</td>
</tr>
<tr>
<td></td>
<td>AProVE</td>
<td>73%</td>
<td></td>
</tr>
</tbody>
</table>

TPA is developed by Adam Koprowski
http://www.win.tue.nl/tpa/
Related work

- A3PAT project for certifying CiME (Contejean [04])

- Isabelle/HOL termination checker (Bulwahn Krauss Nipkow [07])
Put CoLoR in your life! :-)

http://color.loria.fr/

Contributions are welcome!