

Certification of Termination Proofs for Term Rewriting

A short story of a long battle...

Adam Koprowski

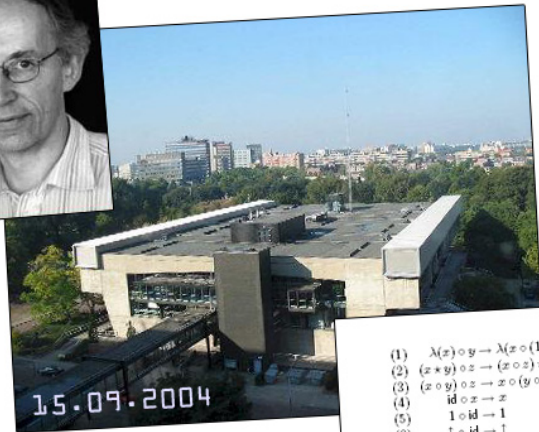
Radboud University Nijmegen
Foundations group, Intelligent Systems, ICIS

16 December 2008

Who am I?



Who am I?

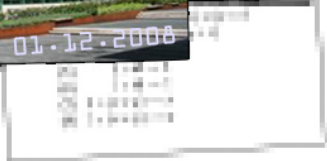
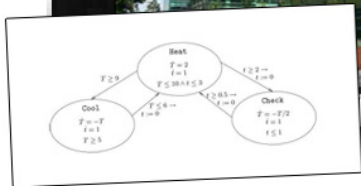


- (1) $\lambda(x) \circ y \rightarrow \lambda(x \circ (1 * (y \circ 1)))$
- (2) $(x * y) \circ z \rightarrow (x \circ z) * (y \circ z)$
- (3) $(x \circ y) \circ z \rightarrow x \circ (y \circ z)$
- (4) $\text{id} \circ x \rightarrow x$
- (5) $1 \circ \text{id} \rightarrow 1$
- (6) $\uparrow \circ \text{id} \rightarrow \uparrow$
- (7) $1 \circ (x * y) \rightarrow x$
- (8) $\uparrow \circ (x * y) \rightarrow y$

Who am I?



Who am I?



- 1 Background: termination of term rewriting
- 2 CoLoR project: certification of termination proofs
 - Why?... motivation
 - How?... CoLoR's approach to certification
 - When?... history of the project
 - What?... overview of the content
 - Related work
 - Certified competition
- 3 Conclusions... sort of

- 1 Background: termination of term rewriting
- 2 CoLoR project: certification of termination proofs
 - Why?... motivation
 - How?... CoLoR's approach to certification
 - When?... history of the project
 - What?... overview of the content
 - Related work
 - Certified competition
- 3 Conclusions... sort of

Example (Quick sort)

$\text{qsort}(\text{nil}) \rightarrow \text{nil}$

$\text{qsort}(x :: xs) \rightarrow \text{append}(\text{qsort}(\text{filterLe}(x, xs)), x :: \text{qsort}(\text{filterGe}(x, xs)))$

$\text{append}(\text{nil}, l) \rightarrow l$

$\text{append}(x :: xs, l) \rightarrow x :: \text{append}(xs, l)$

$\text{filterLe}(n, \text{nil}) \rightarrow \text{nil}$

$\text{filterLe}(n, x :: xs) \rightarrow \text{filter}(\text{le}(x, n), x, \text{filterLe}(n, xs))$

$\text{filterGe}(n, \text{nil}) \rightarrow \text{nil}$

$\text{filterGe}(n, x :: xs) \rightarrow \text{filter}(\text{ge}(x, n), x, \text{filterGe}(n, xs))$

$\text{ge}(x, y) \rightarrow \text{le}(y, x)$

$\text{filter}(\text{false}, x, xs) \rightarrow xs$

$\text{le}(0, y) \rightarrow \text{true}$

$\text{filter}(\text{true}, x, xs) \rightarrow x :: xs$

$\text{le}(s(x), 0) \rightarrow \text{false}$

$\text{le}(s(x), s(y)) \rightarrow \text{le}(x, y)$

Example (Quick sort)

$\text{qsort}(\text{nil}) \rightarrow \text{nil}$

$\text{qsort}(x :: xs) \rightarrow \text{append}(\text{qsort}(\text{filterLe}(x, xs)), x :: \text{qsort}(\text{filterGe}(x, xs)))$

$\text{append}(\text{nil } l) \rightarrow l$

Example (Collatz conjecture)

$\text{collatz}(s(s(x))) \rightarrow f(\text{even}(x), s(s(x)))$

$\text{even}(0) \rightarrow \text{true}$

$f(\text{true}, x) \rightarrow \text{collatz}(\text{half}(x))$

$\text{even}(s(0)) \rightarrow \text{false}$

$f(\text{false}, x) \rightarrow \text{collatz}(s(\text{triple}(x)))$

$\text{even}(s(s(x))) \rightarrow \text{even}(x)$

$\text{half}(0) \rightarrow 0$

$\text{triple}(0) \rightarrow 0$

$\text{half}(s(s(x))) \rightarrow s(\text{half}(x))$

$\text{triple}(s(x)) \rightarrow s(s(s(\text{triple}(x))))$

$\text{filter}(\text{false}, x, xs) \rightarrow xs$

$\text{le}(0, y) \rightarrow \text{true}$

$\text{filter}(\text{true}, x, xs) \rightarrow x :: xs$

$\text{le}(s(x), 0) \rightarrow \text{false}$

$\text{le}(s(x), s(y)) \rightarrow \text{le}(x, y)$

Introduction to term rewriting

Example (Quick sort)

$\text{qsort}(\text{nil}) \rightarrow \text{nil}$

$\text{qsort}(x :: xs) \rightarrow \text{append}(\text{qsort}(\text{filterLe}(x, xs)), x :: \text{qsort}(\text{filterGe}(x, xs)))$

$\text{append}(\text{nil}, l) \rightarrow l$

Example (Collatz conjecture)

$\text{collatz}(s(s(x))) \rightarrow f(\text{true}, s(s(x)))$ $\text{collatz}(0) \rightarrow \text{true}$

$f(\text{true}, s(s(x)))$

$f(\text{false}, s(s(x)))$

Definition

A TRS is **terminating** iff it does not admit infinite reductions.

$\text{half}(0) \rightarrow 0$

$\text{half}(s(s(x))) \rightarrow s(\text{half}(x))$

$\text{triple}(0) \rightarrow 0$

$\text{triple}(s(x)) \rightarrow s(s(s(\text{triple}(x))))$

$\text{filter}(\text{false}, x, xs) \rightarrow xs$

$\text{filter}(\text{true}, x, xs) \rightarrow x :: xs$

$\text{le}(0, y) \rightarrow \text{true}$

$\text{le}(s(x), 0) \rightarrow \text{false}$

$\text{le}(s(x), s(y)) \rightarrow \text{le}(x, y)$

Termination of rewriting

Termination of rewriting:

- **Is undecidable.**
- Is an important topic in term rewriting.
- Many methods exist and new ones are constantly being developed.
- Recently the emphasis is on automation.
- There exists a number of tools for proving termination.
- Stimulated by the termination competition.
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

Termination of rewriting

Termination of rewriting:

- Is undecidable.
- **Is an important topic in term rewriting.**
- Many methods exist and new ones are constantly being developed.
- Recently the emphasis is on automation.
- There exists a number of tools for proving termination.
- Stimulated by the termination competition.
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

Termination of rewriting

Termination of rewriting:

- Is undecidable.
- Is an important topic in term rewriting.
- **Many methods exist and new ones are constantly being developed.**
- Recently the emphasis is on automation.
- There exists a number of tools for proving termination.
- Stimulated by the termination competition.
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

Termination of rewriting

Termination of rewriting:

- Is undecidable.
- Is an important topic in term rewriting.
- Many methods exist and new ones are constantly being developed.
- **Recently the emphasis is on automation.**
- There exists a number of tools for proving termination.
- Stimulated by the termination competition.
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

Termination of rewriting

Termination of rewriting:

- Is undecidable.
- Is an important topic in term rewriting.
- Many methods exist and new ones are constantly being developed.
- Recently the emphasis is on automation.
- **There exists a number of tools for proving termination.**
- Stimulated by the termination competition.
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

Termination of rewriting

Termination of rewriting:

- Is undecidable.
- Is an important topic in term rewriting.
- Many methods exist and new ones are constantly being developed.
- Recently the emphasis is on automation.
- There exists a number of tools for proving termination.
- **Stimulated by the termination competition.**
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

Termination of rewriting

Termination of rewriting:

- Is undecidable.
- Is an important topic in term rewriting.
- Many methods exist and new ones are constantly being developed.
- Recently the emphasis is on automation.
- There exists a number of tools for proving termination.
- Stimulated by the termination competition.
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

Termination of rewriting

Termination of rewriting:

- Is undecidable.
- Is an important topic in term rewriting.
- Many methods exist and new ones are constantly being developed.
- Recently the emphasis is on automation.
- There exists a number of tools for proving termination.
- Stimulated by the termination competition.
- Tools (and proofs that they produce) are getting more and more complex, so reliability is an issue (tools disqualifications in the competition).
- In 2007 a new category of certified termination introduced in the competition.

- 1 Background: termination of term rewriting
- 2 CoLoR project: certification of termination proofs
 - Why?... motivation
 - How?... CoLoR's approach to certification
 - When?... history of the project
 - What?... overview of the content
 - Related work
 - Certified competition
- 3 Conclusions... sort of

CoLoR

`http://color.loria.fr`

CoLoR: Coq Library on Rewriting and Termination.

Goal: certification of termination proofs produced by various termination provers.

- Increasing reliability of termination provers.
- Common proof format for termination provers:
 - common tools (proof presentation, manipulation, ...).
 - control language for provers (integration of tools)
- Extension of proof assistance kernels.

CoLoR

`http://color.loria.fr`

CoLoR: Coq Library on Rewriting and Termination.

Goal: certification of termination proofs produced by various termination provers.

- **Increasing reliability of termination provers.**
- Common proof format for termination provers:
 - common tools (proof presentation, manipulation, ...).
 - control language for provers (integration of tools)
- Extension of proof assistance kernels.

CoLoR

`http://color.loria.fr`

CoLoR: Coq Library on Rewriting and Termination.

Goal: certification of termination proofs produced by various termination provers.

- Increasing reliability of termination provers.
- **Common proof format for termination provers:**
 - common tools (proof presentation, manipulation, ...),
 - control language for provers (integration of tools)
- Extension of proof assistance kernels.

CoLoR

`http://color.loria.fr`

CoLoR: Coq Library on Rewriting and Termination.

Goal: certification of termination proofs produced by various termination provers.

- Increasing reliability of termination provers.
- Common proof format for termination provers:
 - **common tools (proof presentation, manipulation, ...)**,
 - control language for provers (integration of tools)
- Extension of proof assistance kernels.

CoLoR

`http://color.loria.fr`

CoLoR: Coq Library on Rewriting and Termination.

Goal: certification of termination proofs produced by various termination provers.

- Increasing reliability of termination provers.
- Common proof format for termination provers:
 - common tools (proof presentation, manipulation, ...),
 - **control language for provers (integration of tools)**
- Extension of proof assistance kernels.

CoLoR

`http://color.loria.fr`

CoLoR: Coq Library on Rewriting and Termination.

Goal: certification of termination proofs produced by various termination provers.

- Increasing reliability of termination provers.
- Common proof format for termination provers:
 - common tools (proof presentation, manipulation, ...),
 - control language for provers (integration of tools)
- Extension of proof assistance kernels.

How to certify termination results?

- **Possibility: certification of tools source code.**
⇒ difficult, tool dependent, extra work with every change, ...
- **CoLoR's approach:**
 - TPG: common format for termination proofs.
 - Tools output proofs in TPG format.
 - CoLoR: a Coq library of results on termination.
 - Rainbow: a tool for translation from proofs in TPG format to Coq proofs, using results from CoLoR.

How to certify termination results?

- Possibility: certification of tools source code.
⇒ difficult, tool dependent, extra work with every change, ...
- **CoLoR's approach:**
 - TPG: common format for termination proofs.
 - Tools output proofs in TPG format.
 - CoLoR: a Coq library of results on termination.
 - Rainbow: a tool for translation from proofs in TPG format to Coq proofs, using results from CoLoR.

How to certify termination results?

- Possibility: certification of tools source code.
⇒ difficult, tool dependent, extra work with every change, ...
- CoLoR's approach:
 - TPG: common format for termination proofs.
 - Tools output proofs in TPG format.
 - CoLoR: a Coq library of results on termination.
 - Rainbow: a tool for translation from proofs in TPG format to Coq proofs, using results from CoLoR.

How to certify termination results?

- Possibility: certification of tools source code.
⇒ difficult, tool dependent, extra work with every change, ...
- CoLoR's approach:
 - TPG: common format for termination proofs.
 - Tools output proofs in TPG format.
 - CoLoR: a Coq library of results on termination.
 - Rainbow: a tool for translation from proofs in TPG format to Coq proofs, using results from CoLoR.

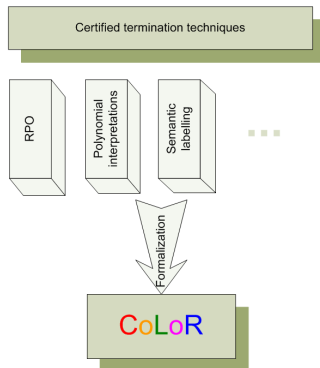
How to certify termination results?

- Possibility: certification of tools source code.
⇒ difficult, tool dependent, extra work with every change, ...
- CoLoR's approach:
 - TPG: common format for termination proofs.
 - Tools output proofs in TPG format.
 - CoLoR: a Coq library of results on termination.
 - Rainbow: a tool for translation from proofs in TPG format to Coq proofs, using results from CoLoR.

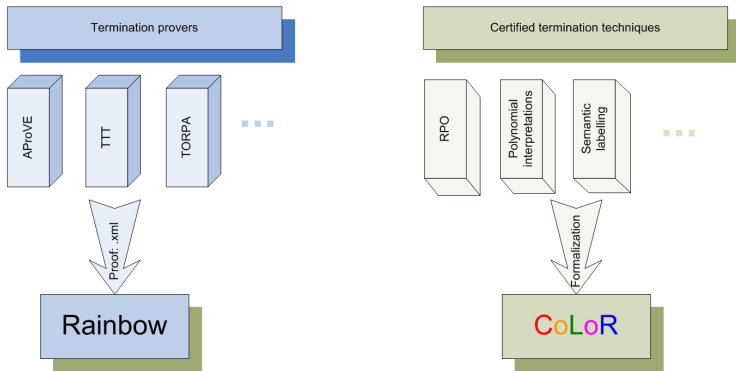
How to certify termination results?

- Possibility: certification of tools source code.
⇒ difficult, tool dependent, extra work with every change, ...
- CoLoR's approach:
 - TPG: common format for termination proofs.
 - Tools output proofs in TPG format.
 - CoLoR: a Coq library of results on termination.
 - Rainbow: a tool for translation from proofs in TPG format to Coq proofs, using results from CoLoR.

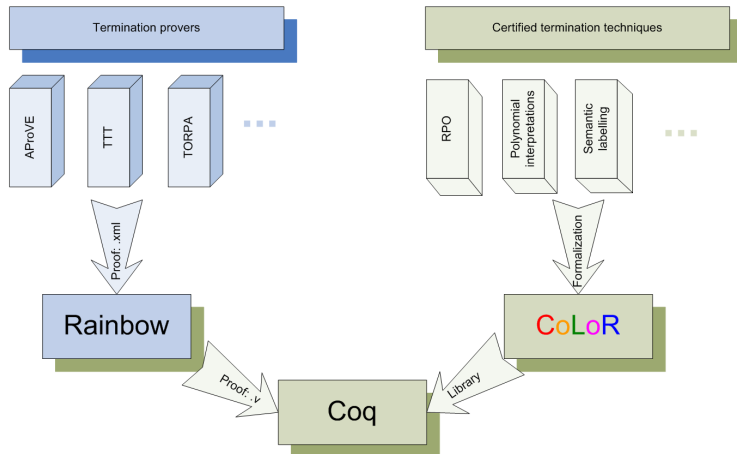
CoLoR's architecture overview



CoLoR's architecture overview



CoLoR's architecture overview



- Project started (Blanqui) March 2004
- First release March 2005
- First certified proofs July 2006
- First certification workshop May 2007
- First certified competition June 2007

- Project started (Blanqui) March 2004
- **First release** **March 2005**
- First certified proofs July 2006
- First certification workshop May 2007
- First certified competition June 2007

- Project started (Blanqui) March 2004
- First release March 2005
- **First certified proofs** July 2006
- First certification workshop May 2007
- First certified competition June 2007

- Project started (Blanqui) March 2004
- First release March 2005
- First certified proofs July 2006
- **First certification workshop** **May 2007**
- First certified competition June 2007

- Project started (Blanqui) March 2004
- First release March 2005
- First certified proofs July 2006
- First certification workshop May 2007
- **First certified competition June 2007**

- Termination criteria:

- **polynomial interpretations** [Hinderer]
- multiset ordering [Koprowski]
- recursive path ordering [Coupet-Grimal, Delobel]
- higher-order recursive path ordering [Koprowski]
- dependency graph cycles [Blanqui]
- **matrix interpretations** [Koprowski, Zantema]
- **arctic interpretations** [Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs [Blanqui]
- dependency graph decomposition [Lucas, Blanqui]
- arguments filtering [Blanqui]
- term conversions [Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations**

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs
 - dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering**

- recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations**

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs
 - dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

[Hinderer]

- multiset ordering

[Koprowski]

- recursive path ordering**

[Coupet-Grimal, Delobel]

- higher-order recursive path ordering

[Koprowski]

- dependency graph cycles

[Blanqui]

- matrix interpretations**

[Koprowski, Zantema]

- arctic interpretations**

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs

[Blanqui]

- dependency graph decomposition

[Lucas, Blanqui]

- arguments filtering

[Blanqui]

- term conversions

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering**
 - dependency graph cycles
 - matrix interpretations
 - arctic interpretations

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs
 - dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations
 - arctic interpretations

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs
 - dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs
 - dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations**

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs
 - dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations**

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs
 - dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations**

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs**

- dependency graph decomposition
 - arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- **polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - **matrix interpretations**
 - **arctic interpretations**

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- **dependency pairs**

- **dependency graph decomposition**

- arguments filtering
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations**

[Hinderer]

[Koprowski]

[Coupet-Grimal, Delobel]

[Koprowski]

[Blanqui]

[Koprowski, Zantema]

[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs**
 - dependency graph decomposition**
 - arguments filtering**
 - term conversions

[Blanqui]

[Lucas, Blanqui]

[Blanqui]

[Blanqui]

- Termination criteria:

- polynomial interpretations**

- multiset ordering
 - recursive path ordering
 - higher-order recursive path ordering
 - dependency graph cycles
 - matrix interpretations**
 - arctic interpretations**

[Hinderer]
[Koprowski]
[Coupet-Grimal, Delobel]
[Koprowski]
[Blanqui]
[Koprowski, Zantema]
[Koprowski, Waldmann]

- Transformation techniques:

- dependency pairs**
 - dependency graph decomposition**
 - arguments filtering
 - term conversions**

[Blanqui]
[Lucas, Blanqui]
[Blanqui]
[Blanqui]

- Term structures:

- simply typed lambda-terms [Koprowski]
- varyadic terms [Blanqui]
- algebraic terms with symbols of fixed arity [Hinderer, Blanqui]

- General libraries and algorithms:

- matrices [Koprowski]
- semi-rings [Koprowski, Zantema]
- finite multisets [Koprowski]
- integer polynomials with multiple variables [Hinderer]
- computation of strongly connected components (SCCs) [Ducas]
- lists, vectors, relations, etc.

- Term structures:

- simply typed lambda-terms

- varyadic terms

- algebraic terms with symbols of fixed arity

[Koprowski]

[Blanqui]

[Hinderer, Blanqui]

- General libraries and algorithms:

- matrices

[Koprowski]

- semi-rings

[Koprowski, Zantema]

- finite multisets

[Koprowski]

- integer polynomials with multiple variables

[Hinderer]

- computation of strongly connected components (SCCs)

[Ducas]

- lists, vectors, relations, etc.

- Term structures:

- simply typed lambda-terms
- **varyadic terms**
- algebraic terms with symbols of fixed arity

[Koprowski]

[Blanqui]

[Hinderer, Blanqui]

- General libraries and algorithms:

- matrices
- semi-rings
- finite multisets
- integer polynomials with multiple variables
- computation of strongly connected components (SCCs)
- lists, vectors, relations, etc.

[Koprowski]

[Koprowski, Zantema]

[Koprowski]

[Hinderer]

[Ducas]

- Term structures:

- simply typed lambda-terms
- varyadic terms
- algebraic terms with symbols of fixed arity

[Koprowski]

[Blanqui]

[Hinderer, Blanqui]

- General libraries and algorithms:

- matrices
- semi-rings
- finite multisets
- integer polynomials with multiple variables
- computation of strongly connected components (SCCs)
- lists, vectors, relations, etc.

[Koprowski]

[Koprowski, Zantema]

[Koprowski]

[Hinderer]

[Ducas]

- Term structures:

- simply typed lambda-terms
- varyadic terms
- algebraic terms with symbols of fixed arity

[Koprowski]

[Blanqui]

[Hinderer, Blanqui]

- General libraries and algorithms:

- matrices
- semi-rings
- finite multisets
- integer polynomials with multiple variables
- computation of strongly connected components (SCCs)
- lists, vectors, relations, etc.

[Koprowski]

[Koprowski, Zantema]

[Koprowski]

[Hinderer]

[Ducas]

- Term structures:
 - simply typed lambda-terms [Koprowski]
 - varyadic terms [Blanqui]
 - algebraic terms with symbols of fixed arity [Hinderer, Blanqui]
- General libraries and algorithms:
 - **matrices** [Koprowski]
 - semi-rings [Koprowski, Zantema]
 - finite multisets [Koprowski]
 - integer polynomials with multiple variables [Hinderer]
 - computation of strongly connected components (SCCs) [Ducas]
 - lists, vectors, relations, etc.

- Term structures:
 - simply typed lambda-terms [Koprowski]
 - varyadic terms [Blanqui]
 - algebraic terms with symbols of fixed arity [Hinderer, Blanqui]
- General libraries and algorithms:
 - matrices [Koprowski]
 - semi-rings [Koprowski, Zantema]
 - finite multisets [Koprowski]
 - integer polynomials with multiple variables [Hinderer]
 - computation of strongly connected components (SCCs) [Ducas]
 - lists, vectors, relations, etc.

- Term structures:
 - simply typed lambda-terms [Koprowski]
 - varyadic terms [Blanqui]
 - algebraic terms with symbols of fixed arity [Hinderer, Blanqui]
- General libraries and algorithms:
 - matrices [Koprowski]
 - semi-rings [Koprowski, Zantema]
 - **finite multisets** [Koprowski]
 - integer polynomials with multiple variables [Hinderer]
 - computation of strongly connected components (SCCs) [Ducas]
 - lists, vectors, relations, etc.

- Term structures:
 - simply typed lambda-terms [Koprowski]
 - varyadic terms [Blanqui]
 - algebraic terms with symbols of fixed arity [Hinderer, Blanqui]
- General libraries and algorithms:
 - matrices [Koprowski]
 - semi-rings [Koprowski, Zantema]
 - finite multisets [Koprowski]
 - **integer polynomials with multiple variables** [Hinderer]
 - computation of strongly connected components (SCCs) [Ducas]
 - lists, vectors, relations, etc.

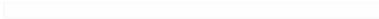

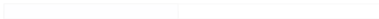
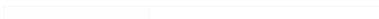
- Term structures:
 - simply typed lambda-terms [Koprowski]
 - varyadic terms [Blanqui]
 - algebraic terms with symbols of fixed arity [Hinderer, Blanqui]
- General libraries and algorithms:
 - matrices [Koprowski]
 - semi-rings [Koprowski, Zantema]
 - finite multisets [Koprowski]
 - integer polynomials with multiple variables [Hinderer]
 - **computation of strongly connected components (SCCs)** [Ducas]
 - lists, vectors, relations, etc.

- Term structures:
 - simply typed lambda-terms [Koprowski]
 - varyadic terms [Blanqui]
 - algebraic terms with symbols of fixed arity [Hinderer, Blanqui]
- General libraries and algorithms:
 - matrices [Koprowski]
 - semi-rings [Koprowski, Zantema]
 - finite multisets [Koprowski]
 - integer polynomials with multiple variables [Hinderer]
 - computation of strongly connected components (SCCs) [Ducas]
 - **lists, vectors, relations, etc.**

In total:

- ≈ 50.000 lines of code.
- ≈ 1.000 definitions and ≈ 3.000 lemmas.
- Only 20% of that is the code for actual termination methods!

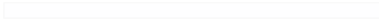

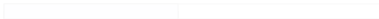
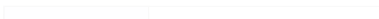
Size comparison with other libraries:

- Coq standard library 
- C-CoRN 
- COMPCERT 
- CoLoR 

In total:

- ≈ 50.000 lines of code.
- ≈ 1.000 definitions and ≈ 3.000 lemmas.
- Only 20% of that is the code for actual termination methods!

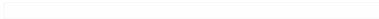

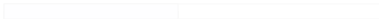
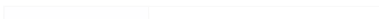
Size comparison with other libraries:

- Coq standard library 
- C-CoRN 
- COMPCERT 
- CoLoR 

In total:

- ≈ 50.000 lines of code.
- ≈ 1.000 definitions and ≈ 3.000 lemmas.
- **Only 20% of that is the code for actual termination methods!**

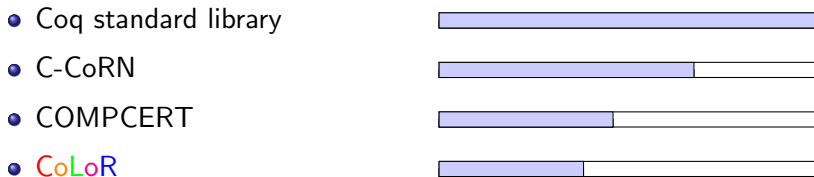
Size comparison with other libraries:

- Coq standard library 
- C-CoRN 
- COMPCERT 
- CoLoR 

In total:

- ≈ 50.000 lines of code.
- ≈ 1.000 definitions and ≈ 3.000 lemmas.
- Only 20% of that is the code for actual termination methods!

Size comparison with other libraries:



- CoLoR project
Authors: Blanqui, ...
Proof assistant: Coq
- A3PAT project
Authors: Contejean, ...
Proof assistant: Coq
- Isabelle/HOL termination checker
Authors: Bulwahn, Krauss, Nipkow, ...
Proof assistant: Isabelle/HOL

- **CoLoR project**
Authors: Blanqui, ...
Proof assistant: Coq
- **A3PAT project**
Authors: Contejean, ...
Proof assistant: Coq
- **Isabelle/HOL termination checker**
Authors: Bulwahn, Krauss, Nipkow, ...
Proof assistant: Isabelle/HOL

- **CoLoR project**
Authors: Blanqui, ...
Proof assistant: Coq
- **A3PAT project**
Authors: Contejean, ...
Proof assistant: Coq
- **Isabelle/HOL termination checker**
Authors: Bulwahn, Krauss, Nipkow, ...
Proof assistant: Isabelle/HOL

Certified competition

- In the 2007 termination competition a new “certified” category was introduced.
- Participants 2007 (975 problems):

- TPA + CoLoR
- C/ME + A3PAT
- T_1T_2 + CoLoR

- Participants 2008 (1391 problems):

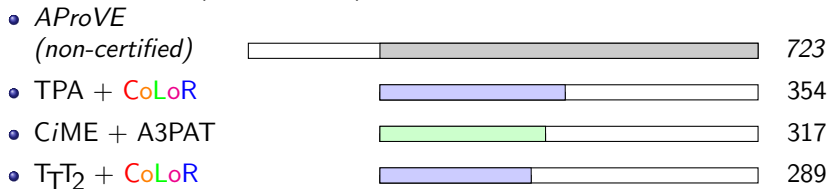
- APoVE + C/ME + A3PAT
- APoVE + CoLoR
- APoVE + A3PAT
- C/ME + A3PAT
- TPA + CoLoR

Certified competition

- In the 2007 termination competition a new “certified” category was introduced.
- Participants 2007 (975 problems):
 - TPA + CoLoR
 - CiME + A3PAT
 - $T_T T_2$ + CoLoR
- Participants 2008 (1391 problems):

Certified competition

- In the 2007 termination competition a new “certified” category was introduced.
- Participants 2007 (975 problems):

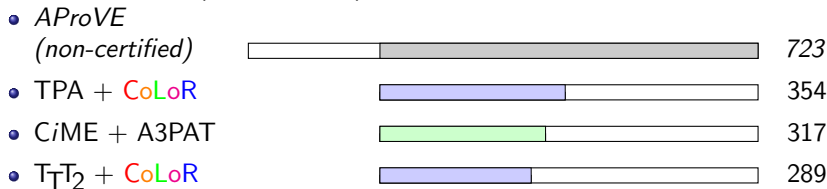


- Participants 2008 (1391 problems):

Certified competition

- In the 2007 termination competition a new “certified” category was introduced.

- Participants 2007 (975 problems):



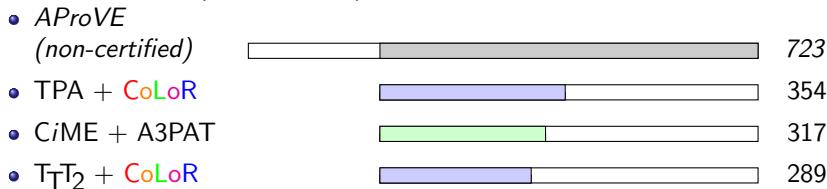
- Participants 2008 (1391 problems):

- AProVE + CoLoR + A3PAT
- AProVE + CoLoR
- AProVE + A3PAT
- CiME3 + A3PAT
- Matchbox + CoLoR

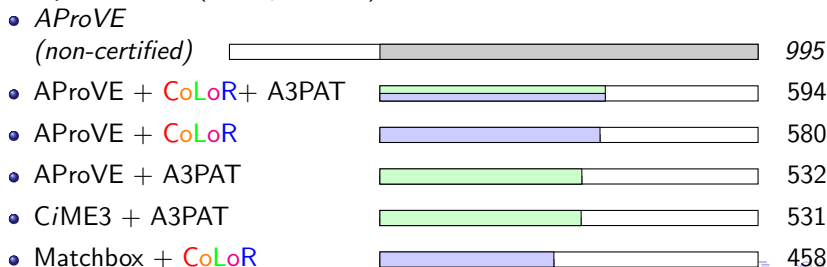
Certified competition

- In the 2007 termination competition a new “certified” category was introduced.

- Participants 2007 (975 problems):



- Participants 2008 (1391 problems):



- 1 Background: termination of term rewriting
- 2 CoLoR project: certification of termination proofs
 - Why?... motivation
 - How?... CoLoR's approach to certification
 - When?... history of the project
 - What?... overview of the content
 - Related work
 - Certified competition
- 3 Conclusions... sort of

- Lesson 1

If it is possible do (involved) computations/reasoning in an unsafe setting and verify the results in Coq a posteriori.

That requires some notion of a certificate.

Proof search is usually much more complex than proof verification.

We see that even in theorem provers — proof checking VS proof searching.

- Lesson 1

If it is possible do (involved) computations/reasoning in an unsafe setting and verify the results in Coq a posteriori.

That requires some notion of a certificate.

Proof search is usually much more complex than proof verification.

We see that even in theorem provers — proof checking VS proof searching.

- Lesson 1

If it is possible do (involved) computations/reasoning in an unsafe setting and verify the results in Coq a posteriori.

That requires some notion of a certificate.

Proof search is usually much more complex than proof verification.

We see that even in theorem provers — proof checking VS proof searching.

- Lesson 1

If it is possible do (involved) computations/reasoning in an unsafe setting and verify the results in Coq a posteriori.

That requires some notion of a certificate.

Proof search is usually much more complex than proof verification.

We see that even in theorem provers — proof checking VS proof searching.

- **Lesson 2**

It is not unusual for software projects to be behind schedule / run out of budget.

It is even more so for Coq projects.

Why?

- algorithm \mapsto program
- paper proof \mapsto formal proof in Coq
- Lack of libraries.
- Proof engineering is not yet as mature as software engineering (re-usability, re-factoring etc.)

- Lesson 2

It is not unusual for software projects to be behind schedule / run out of budget.

It is even more so for Coq projects.

Why?

- algorithm \mapsto program
- paper proof \mapsto formal proof in Coq
- Lack of libraries.
- Proof engineering is not yet as mature as software engineering (re-usability, re-factoring etc.)

- Lesson 2

It is not unusual for software projects to be behind schedule / run out of budget.

It is even more so for Coq projects.

Why?

- algorithm \mapsto program
- paper proof \mapsto formal proof in Coq
- Lack of libraries.
- Proof engineering is not yet as mature as software engineering (re-usability, re-factoring etc.)

- Lesson 2

It is not unusual for software projects to be behind schedule / run out of budget.

It is even more so for Coq projects.

Why?

- algorithm \mapsto program
- paper proof \mapsto formal proof in Coq
- Lack of libraries.
- Proof engineering is not yet as mature as software engineering (re-usability, re-factoring etc.)

- Lesson 2

It is not unusual for software projects to be behind schedule / run out of budget.

It is even more so for Coq projects.

Why?

- algorithm \mapsto program
- paper proof \mapsto formal proof in Coq
- Lack of libraries.
- Proof engineering is not yet as mature as software engineering (re-usability, re-factoring etc.)

- Lesson 3

When writing your definitions there is usually plenty of choice.

You want to make the right choices. **You really do.**

Because that will have a tremendous impact on the reasoning about those definitions that you are going to do for long hours afterwards.

- Lesson 3

When writing your definitions there is usually plenty of choice.

You want to make the right choices. *You really do.*

Because that will have a tremendous impact on the reasoning about those definitions that you are going to do for long hours afterwards.

- Lesson 3

When writing your definitions there is usually plenty of choice.

You want to make the right choices. **You really do.**

Because that will have a tremendous impact on the reasoning about those definitions that you are going to do for long hours afterwards.

- Lesson 3

When writing your definitions there is usually plenty of choice.

You want to make the right choices. **You really do.**

Because that will have a tremendous impact on the reasoning about those definitions that you are going to do for long hours afterwards.

<http://color.loria.fr>



Thank you for your attention.