Computational Logic

Certification of Nontermination Proofs using Strategies and Nonlooping Derivations

Julian Nagele René Thiemann **Sarah Winkler**

Computational Logic Group
University of Innsbruck

VSTTE, July 18, 2014

Outline

Motivation

Approach and Techniques

Forbidden Pattern Loops
- The Theory Side
- The IsaFoR Side
- The Tool and Certification Sides

Experiments and Conclusion

Why Certify Nontermination?

Example (Buggy map function)

term rewrite system (TRS) $\mathcal{R}$

$$\mathsf{map}\ f\ xs \to \mathsf{if}\ (\mathsf{empty}\ xs)\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ xs))\ (\mathsf{map}\ f\ xs))$$

$$\mathsf{hd}\ (:\ x\ xs) \to x \qquad \mathsf{if}\ \mathsf{true}\ t\ e \to t \qquad \mathsf{empty}\ (:\ x\ xs) \to \mathsf{false}$$

$$\mathsf{tl}\ (:\ x\ xs) \to xs \qquad \mathsf{if}\ \mathsf{false}\ t\ e \to e \qquad \mathsf{empty}\ \mathsf{nil} \to \mathsf{true}$$

Why Certify Nontermination?

Example (Buggy map function)

term rewrite system (TRS) $\mathcal{R}$

$$\mathsf{map}\ f\ xs \to \mathsf{if}\ (\mathsf{empty}\ xs)\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ xs))\ (\mathsf{map}\ f\ xs))$$

$$\mathsf{hd}\ (:\ x\ xs) \to x \qquad \mathsf{if}\ \mathsf{true}\ t\ e \to t \qquad \mathsf{empty}\ (:\ x\ xs) \to \mathsf{false}$$

$$\mathsf{tl}\ (:\ x\ xs) \to xs \qquad \mathsf{if}\ \mathsf{false}\ t\ e \to e \qquad \mathsf{empty}\ \mathsf{nil} \to \mathsf{true}$$

$$\underline{\mathsf{map}\ f\ \mathsf{nil}} \to \mathsf{if}\ (\mathsf{empty}\ \mathsf{nil})\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ \mathsf{nil}))\ (\underline{\mathsf{map}\ f\ \mathsf{nil}})) \to \ldots$$

Why Certify Nontermination?

Example (Buggy map function)

term rewrite system (TRS) $\mathcal{R}$

$$\mathsf{map}\ f\ xs \to \mathsf{if}\ (\mathsf{empty}\ xs)\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ xs))\ (\mathsf{map}\ f\ xs))$$

$$\mathsf{hd}\ (:\ x\ xs) \to x \qquad \mathsf{if}\ \mathsf{true}\ t\ e \to t \qquad \mathsf{empty}\ (:\ x\ xs) \to \mathsf{false}$$

$$\mathsf{tl}\ (:\ x\ xs) \to xs \qquad \mathsf{if}\ \mathsf{false}\ t\ e \to e \qquad \mathsf{empty}\ \mathsf{nil} \to \mathsf{true}$$

is nonterminating: it admits infinite derivation

$$\underline{\mathsf{map}\ f\ \mathsf{nil}} \to \mathsf{if}\ (\mathsf{empty}\ \mathsf{nil})\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ \mathsf{nil}))\ (\underline{\mathsf{map}\ f\ \mathsf{nil}})) \to \ldots$$

Why Certify Nontermination?

Example (Buggy map function)

term rewrite system (TRS) $\mathcal{R}$

$$\mathsf{map}\ f\ xs \to \mathsf{if}\ (\mathsf{empty}\ xs)\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ xs))\ (\mathsf{map}\ f\ xs))$$

$$\mathsf{hd}\ (:\ x\ xs) \to x \qquad \mathsf{if}\ \mathsf{true}\ t\ e \to t \qquad \mathsf{empty}\ (:\ x\ xs) \to \mathsf{false}$$

$$\mathsf{tl}\ (:\ x\ xs) \to xs \qquad \mathsf{if}\ \mathsf{false}\ t\ e \to e \qquad \mathsf{empty}\ \mathsf{nil} \to \mathsf{true}$$

is nonterminating: it admits infinite derivation

$$\underline{\mathsf{map}\ f\ \mathsf{nil}} \to \mathsf{if}\ (\mathsf{empty}\ \mathsf{nil})\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ \mathsf{nil}))\ (\underline{\mathsf{map}\ f\ \mathsf{nil}})) \to \ldots$$

- termination tools (AProVE, $\mathsf{T_TT_2}$, CiME, ...) give nontermination proof: useful to find bugs

Why Certify Nontermination?

Example (Buggy map function)

term rewrite system (TRS) $\mathcal{R}$

$$\mathsf{map}\ f\ xs \to \mathsf{if}\ (\mathsf{empty}\ xs)\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ xs))\ (\mathsf{map}\ f\ xs))$$

$$\mathsf{hd}\ (:\ x\ xs) \to x \qquad \mathsf{if}\ \mathsf{true}\ t\ e \to t \qquad \mathsf{empty}\ (:\ x\ xs) \to \mathsf{false}$$

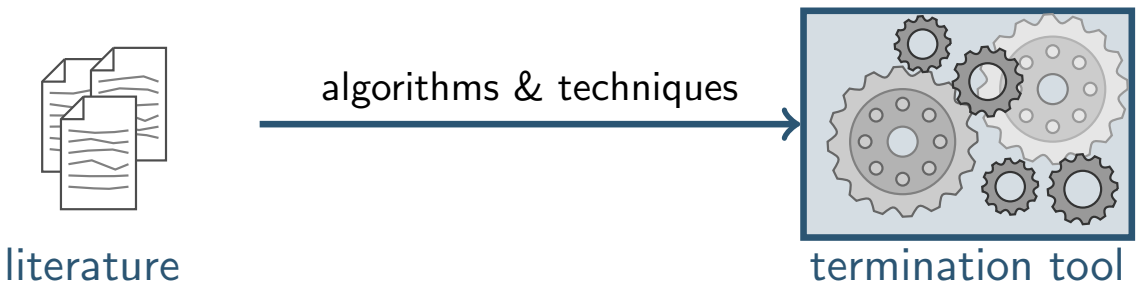
$$\mathsf{tl}\ (:\ x\ xs) \to xs \qquad \mathsf{if}\ \mathsf{false}\ t\ e \to e \qquad \mathsf{empty}\ \mathsf{nil} \to \mathsf{true}$$

is nonterminating: it admits infinite derivation

$$\underline{\mathsf{map}\ f\ \mathsf{nil}} \to \mathsf{if}\ (\mathsf{empty}\ \mathsf{nil})\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ \mathsf{nil}))\ (\underline{\mathsf{map}\ f\ \mathsf{nil}})) \to \ldots$$

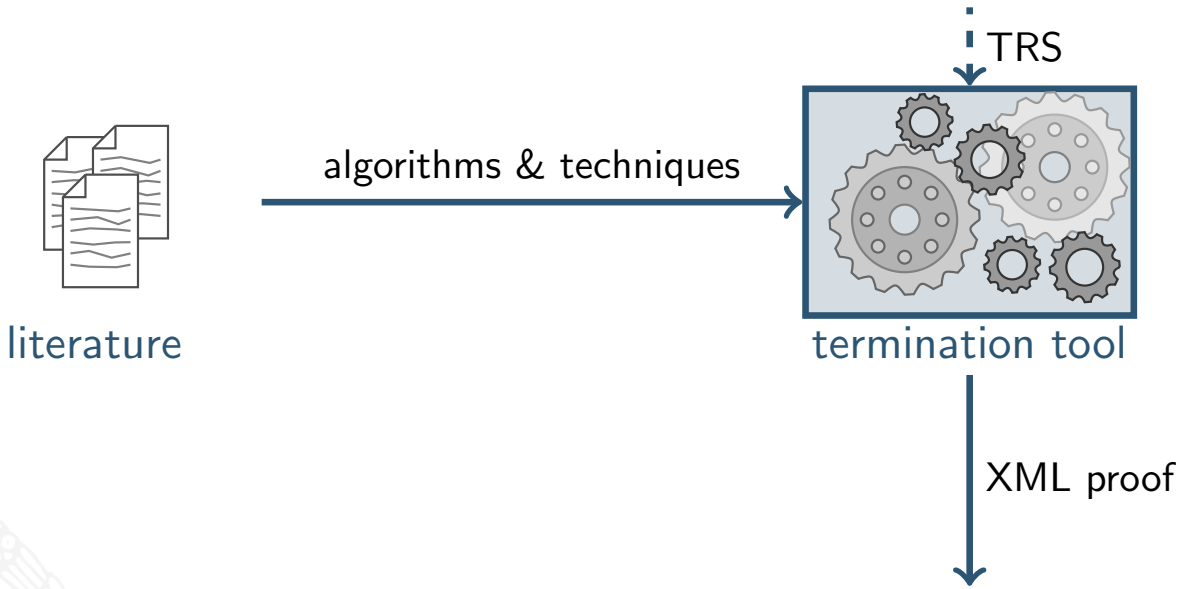
- termination tools (AProVE, $\mathsf{T_TT_2}$, CiME, ...) give nontermination proof: useful to find bugs
- but termination tools are complex and error-prone: certification can guarantee correctness of proofs

Approach Outline



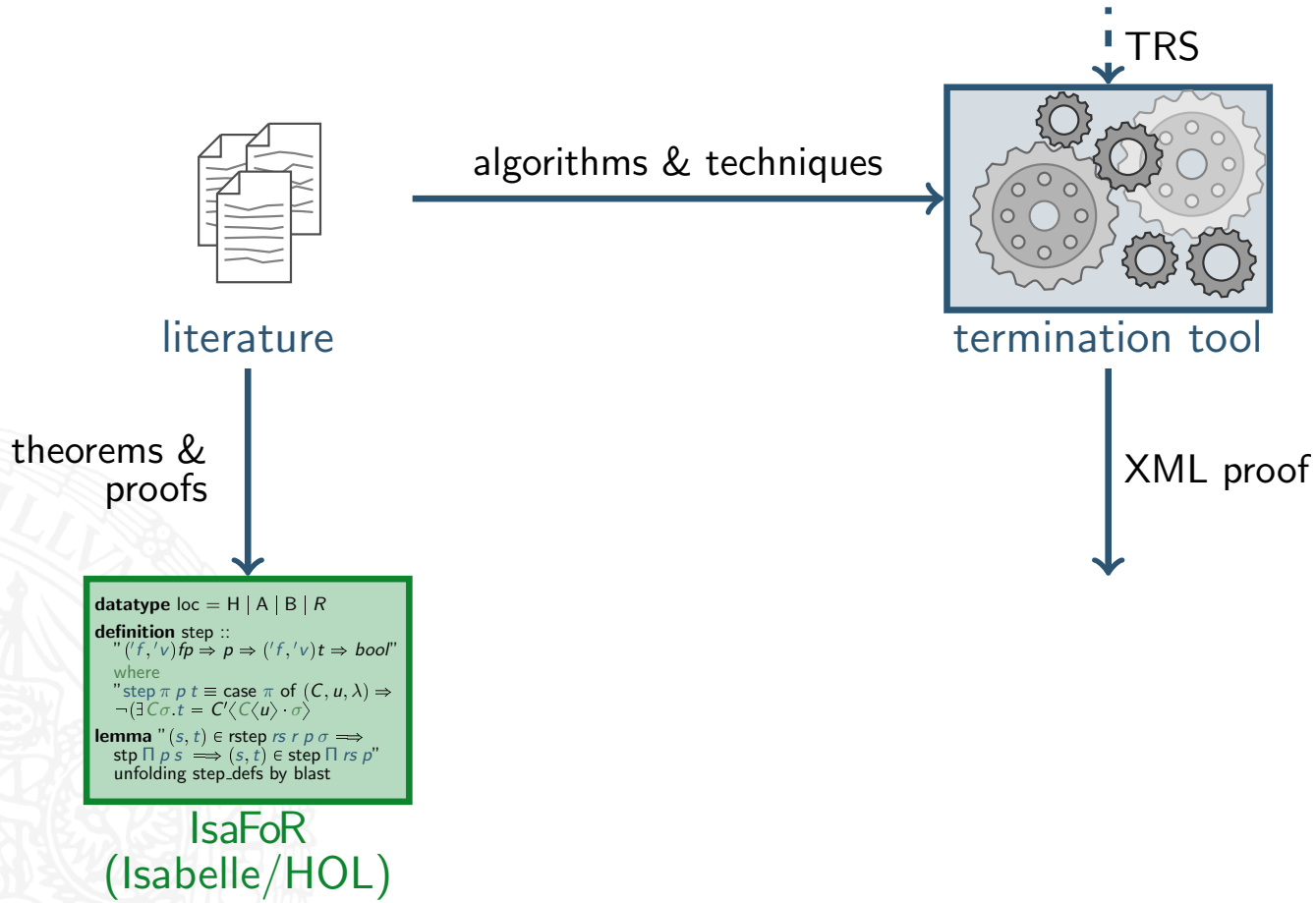
Approach Outline

- termination tools provide evidence of nontermination



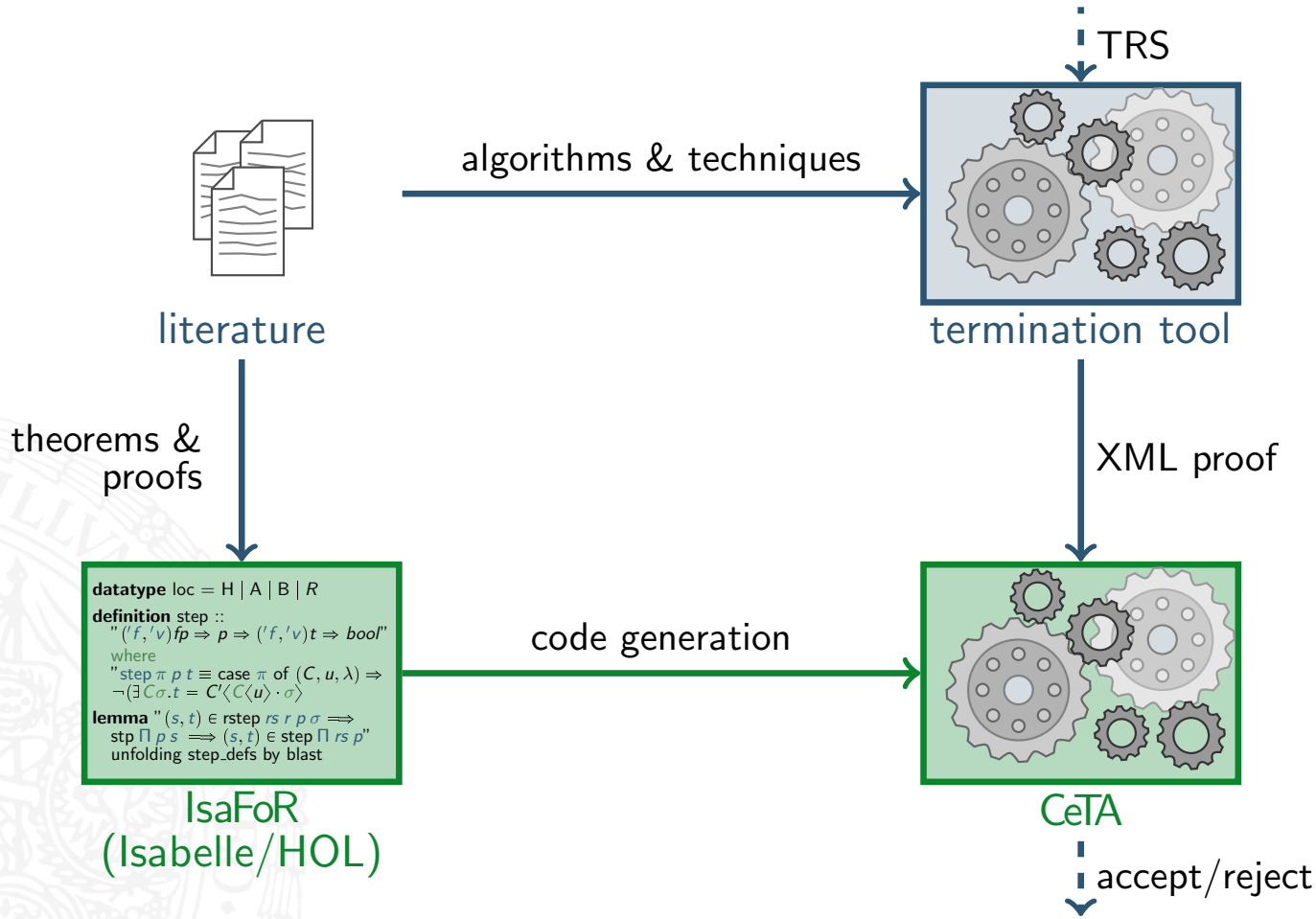
Approach Outline

- termination tools provide evidence of nontermination

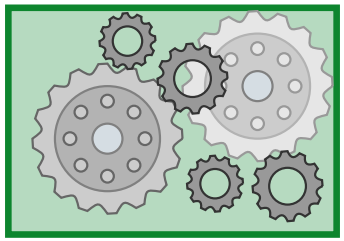


Approach Outline

- termination tools provide evidence of nontermination
- CeTA automatically certifies correctness of evidence



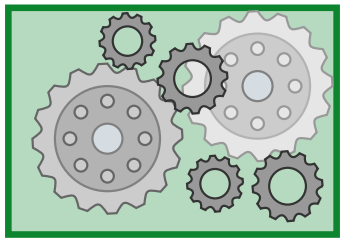
CeTA's Nontermination Machinery 2013



- (innermost) loops

R. Thiemann and C. Sternagel.
Certification of Nontermination Proofs.
Proc. ITP 2012, pp. 266–282, 2012.

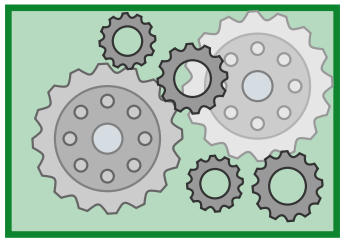
CeTA's Nontermination Machinery 2014



- (innermost) loops
- loops with various strategies (forbidden pattern loops) ①

R. Thiemann, C. Sternagel, J. Giesl, and P. Schneider-Kamp.
Loops under Strategies ... Continued.
Proc. IWS 2010, pp. 51–65, 2010.

CeTA's Nontermination Machinery 2014

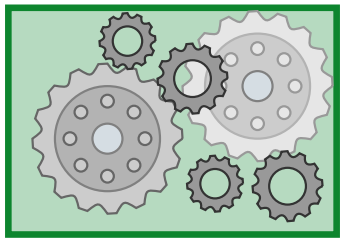


- (innermost) loops
- loops with various strategies (forbidden pattern loops) ①
- nonlooping derivations
  - from innermost termination to termination ②

B. Gramlich.

Abstract Relations between Restricted Termination and Confluence Properties of Rewrite Systems.
Fundamenta Informaticae, 24:3–23, 1995.

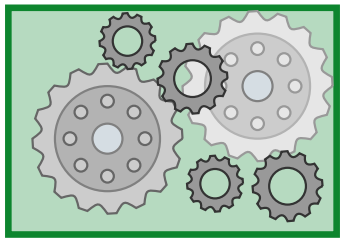
CeTA's Nontermination Machinery 2014



- (innermost) loops
- loops with various strategies (forbidden pattern loops) ①
- nonlooping derivations
  - from innermost termination to termination ②
  - rewriting/narrowing dependency pairs ③

J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke.
Mechanizing and Improving Dependency Pairs.
Journal of Automated Reasoning, 37(3):155–203, 2006.

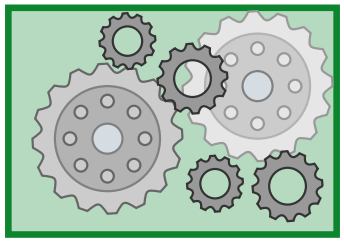
CeTA's Nontermination Machinery 2014



- (innermost) loops
- loops with various strategies (forbidden pattern loops) ①
- nonlooping derivations
  - from innermost termination to termination ②
  - rewriting/narrowing dependency pairs ③
  - detect nonterminating derivations using pattern rules ④

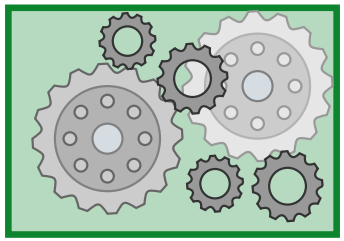
F. Emmes, T. Enger, and J. Giesl.
Proving non-looping non-termination automatically.
Proc. IJCAR 2012, pp. 225–240, 2012.

CeTA's Nontermination Machinery 2014



- (innermost) loops
- loops with various strategies (forbidden pattern loops) ①
- nonlooping derivations
  - from innermost termination to termination ②
  - rewriting/narrowing dependency pairs ③
  - detect nonterminating derivations using pattern rules ④
- partial nontermination proofs ⑤

CeTA's Nontermination Machinery 2014



- (innermost) loops
- loops with various strategies (forbidden pattern loops) ① this talk
- nonlooping derivations
  - from innermost termination to termination ②
  - rewriting/narrowing dependency pairs ③
  - detect nonterminating derivations using pattern rules ④
- partial nontermination proofs ⑤

Outline

Motivation

Approach and Techniques

Forbidden Pattern Loops
- The Theory Side
- The IsaFoR Side
- The Tool and Certification Sides

Experiments and Conclusion

Forbidden Pattern Loops

Example (Buggy map function)

$$\text{map}\ f\ xs \to \text{if}\ (\text{empty}\ xs)\ \text{nil}\ (:\ (f\ (\text{hd}\ xs))\ (\text{map}\ f\ xs))$$

$$\text{hd}\ (:\ x\ xs) \to x \qquad \text{if true}\ t\ e \to t \qquad \text{empty}\ (:\ x\ xs) \to \text{false}$$

$$\text{tl}\ (:\ x\ xs) \to xs \qquad \text{if false}\ t\ e \to e \qquad \text{empty nil} \to \text{true}$$

admits loop

$$\text{map}\ f\ \text{nil} \to \text{if}\ (\text{empty nil})\ \text{nil}\ (:\ (f\ (\text{hd nil}))\ (\underline{\text{map}\ f\ \text{nil}}))$$

Forbidden Pattern Loops

Example (Buggy map function)

$$\mathsf{map}\ f\ xs \to \mathsf{if}\ (\mathsf{empty}\ xs)\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ xs))\ (\mathsf{map}\ f\ xs))$$

$$\mathsf{hd}\ (:\ x\ xs) \to x \qquad \mathsf{if}\ \mathsf{true}\ t\ e \to t \qquad \mathsf{empty}\ (:\ x\ xs) \to \mathsf{false}$$

$$\mathsf{tl}\ (:\ x\ xs) \to xs \qquad \mathsf{if}\ \mathsf{false}\ t\ e \to e \qquad \mathsf{empty}\ \mathsf{nil} \to \mathsf{true}$$

admits loop

$$\mathsf{map}\ f\ \mathsf{nil} \to \mathsf{if}\ (\mathsf{empty}\ \mathsf{nil})\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ \mathsf{nil}))\ (\underline{\mathsf{map}\ f\ \mathsf{nil}}))$$

- this loop would not actually occur in practice

Forbidden Pattern Loops

Example (Buggy map function)

$$\text{map}\ f\ xs \to \text{if}\ (\text{empty}\ xs)\ \text{nil}\ (:\ (f\ (\text{hd}\ xs))\ (\text{map}\ f\ xs))$$

$$\text{hd}\ (:\ x\ xs) \to x \qquad \text{if true}\ t\ e \to t \qquad \text{empty}\ (:\ x\ xs) \to \text{false}$$

$$\text{tl}\ (:\ x\ xs) \to xs \qquad \text{if false}\ t\ e \to e \qquad \text{empty nil} \to \text{true}$$

admits loop

$$\text{map}\ f\ \text{nil} \to \text{if}\ (\text{empty nil})\ \text{nil}\ (:\ (f\ (\text{hd nil}))\ (\underline{\text{map}\ f\ \text{nil}}))$$

- this loop would not actually occur in practice
- desired evaluation strategy:
  disallow reductions in *then* and *else* branches

Forbidden Pattern Loops

Example (Buggy map function)

$$\mathsf{map}\ f\ xs \to \mathsf{if}\ (\mathsf{empty}\ xs)\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ xs))\ (\mathsf{map}\ f\ xs))$$

$$\mathsf{hd}\ (:\ x\ xs) \to x \qquad \mathsf{if}\ \mathsf{true}\ t\ e \to t \qquad \mathsf{empty}\ (:\ x\ xs) \to \mathsf{false}$$

$$\mathsf{tl}\ (:\ x\ xs) \to xs \qquad \mathsf{if}\ \mathsf{false}\ t\ e \to e \qquad \mathsf{empty}\ \mathsf{nil} \to \mathsf{true}$$

admits loop

$$\mathsf{map}\ f\ \mathsf{nil} \to \mathsf{if}\ (\mathsf{empty}\ \mathsf{nil})\ \mathsf{nil}\ (:\ (f\ (\mathsf{hd}\ \mathsf{nil}))\ (\underline{\mathsf{map}\ f\ \mathsf{nil}}))$$

- this loop would not actually occur in practice
- desired evaluation strategy:
  disallow reductions in *then* and *else* branches

We want to check loops in presence of strategies!

Definition (Forbidden Pattern)

(ℓ, o, λ) is forbidden pattern if $o \in \mathcal{P}\mathrm{os}(\ell)$ for term ℓ and $\lambda \in \{\mathsf{H}, \mathsf{A}, \mathsf{B}, \mathsf{R}\}$

Definition (Forbidden Pattern)

(ℓ, o, λ) is forbidden pattern if $o \in \mathcal{P}\mathrm{os}(\ell)$ for term ℓ and $\lambda \in \{\mathsf{H}, \mathsf{A}, \mathsf{B}, \mathsf{R}\}$

Definition (Rewriting with Forbidden Patterns)

for TRS R, terms s, t, position q and set of forbidden patterns Π have

$$s \xrightarrow[R,q]{\Pi} t$$

Definition (Forbidden Pattern)

(ℓ, o, λ) is forbidden pattern if $o \in \mathcal{P}os(\ell)$ for term ℓ and $\lambda \in \{\mathsf{H}, \mathsf{A}, \mathsf{B}, \mathsf{R}\}$

Definition (Rewriting with Forbidden Patterns)

for TRS R, terms s, t, position q and set of forbidden patterns Π have

$$s \xrightarrow[R,q]{\Pi} t$$

if $s \to_q t$ in R and there exists no $(\ell, o, \lambda) \in \Pi$, $o' \in \mathcal{P}os(s)$, substitution σ such that $s|_{o'} = \ell\sigma$ and

- $q = o'o$ if $\lambda = \mathsf{H}$ (here)

Definition (Forbidden Pattern)

(ℓ, o, λ) is forbidden pattern if $o \in \mathcal{P}os(\ell)$ for term ℓ and $\lambda \in \{\mathsf{H}, \mathsf{A}, \mathsf{B}, \mathsf{R}\}$

Definition (Rewriting with Forbidden Patterns)

for TRS R, terms s, t, position q and set of forbidden patterns Π have

$$s \xrightarrow[R,q]{\Pi} t$$

if $s \to_q t$ in R and there exists no $(\ell, o, \lambda) \in \Pi$, $o' \in \mathcal{P}os(s)$, substitution σ such that $s|_{o'} = \ell\sigma$ and

- $q = o'o$ if $\lambda = \mathsf{H}$ (here)
- $q > o'o$ if $\lambda = \mathsf{B}$ (below)

Definition (Forbidden Pattern)

(ℓ, o, λ) is forbidden pattern if $o \in \mathcal{P}\mathrm{os}(\ell)$ for term ℓ and $\lambda \in \{\mathsf{H}, \mathsf{A}, \mathsf{B}, \mathsf{R}\}$

Definition (Rewriting with Forbidden Patterns)

for TRS R, terms s, t, position q and set of forbidden patterns Π have

$$s \xrightarrow[R,q]{\Pi} t$$

if $s \to_q t$ in R and there exists no $(\ell, o, \lambda) \in \Pi$, $o' \in \mathcal{P}\mathrm{os}(s)$, substitution σ such that $s|_{o'} = \ell\sigma$ and

- $q = o'o$ if $\lambda = \mathsf{H}$ (here)
- $q > o'o$ if $\lambda = \mathsf{B}$ (below)
- $q < o'o$ if $\lambda = \mathsf{A}$ (above)

Definition (Forbidden Pattern)

(ℓ, o, λ) is forbidden pattern if $o \in \mathcal{P}os(\ell)$ for term ℓ and $\lambda \in \{\mathsf{H}, \mathsf{A}, \mathsf{B}, \mathsf{R}\}$

Definition (Rewriting with Forbidden Patterns)

for TRS R, terms s, t, position q and set of forbidden patterns Π have

$$s \xrightarrow[R,q]{\Pi} t$$

if $s \to_q t$ in R and there exists no $(\ell, o, \lambda) \in \Pi$, $o' \in \mathcal{P}os(s)$, substitution σ such that $s|_{o'} = \ell\sigma$ and

- $q = o'o$ if $\lambda = \mathsf{H}$ (here)
- $q > o'o$ if $\lambda = \mathsf{B}$ (below)
- $q < o'o$ if $\lambda = \mathsf{A}$ (above)
- q is right of o' if $\lambda = \mathsf{R}$ (right of)

Definition (Forbidden Pattern)

(ℓ, o, λ) is forbidden pattern if $o \in \mathcal{P}os(\ell)$ for term ℓ and $\lambda \in \{\mathsf{H}, \mathsf{A}, \mathsf{B}, \mathsf{R}\}$

Definition (Rewriting with Forbidden Patterns)

for TRS R, terms s, t, position q and set of forbidden patterns Π have

$$s \xrightarrow[R,q]{\Pi} t$$

if $s \to_q t$ in R and there exists no $(\ell, o, \lambda) \in \Pi$, $o' \in \mathcal{P}os(s)$, substitution σ such that $s|_{o'} = \ell\sigma$ and

- $q = o'o$ if $\lambda = \mathsf{H}$ (here)
- $q > o'o$ if $\lambda = \mathsf{B}$ (below)
- $q < o'o$ if $\lambda = \mathsf{A}$ (above)
- q is right of o' if $\lambda = \mathsf{R}$ (right of)

Example (Buggy map function)

intended evaluation strategy for if is obtained by

$$\Pi = \{(\text{if } b\ t\ e, p, \lambda) \mid p \in \{12, 2\} \text{ and } \lambda \in \{\mathsf{H}, \mathsf{B}\}$$

Looking At Loops

Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Looking At Loops

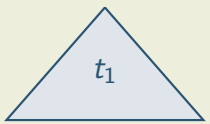
Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* C[t_1\mu] \to_R^* C[C[t_1\mu]\mu] \to_R^* C[C[C[t_1\mu]\mu]\mu] \to_R^* \ldots$$



$\to_R^*$

Looking At Loops

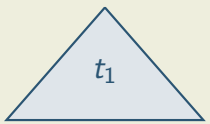
Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* C[t_1\mu] \to_R^* C[C[t_1\mu]\mu] \to_R^* C[C[C[t_1\mu]\mu]\mu] \to_R^* \ldots$$



$\to_R^*$

Looking At Loops

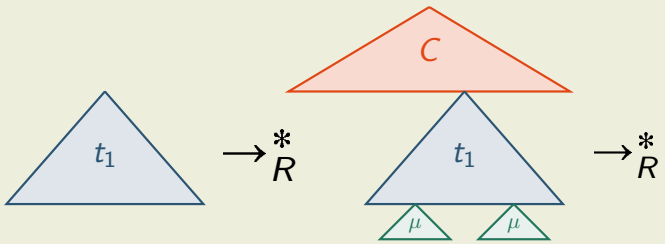
Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* C[t_1\mu] \to_R^* C[C[t_1\mu]\mu] \to_R^* C[C[C[t_1\mu]\mu]\mu] \to_R^* \ldots$$



Looking At Loops

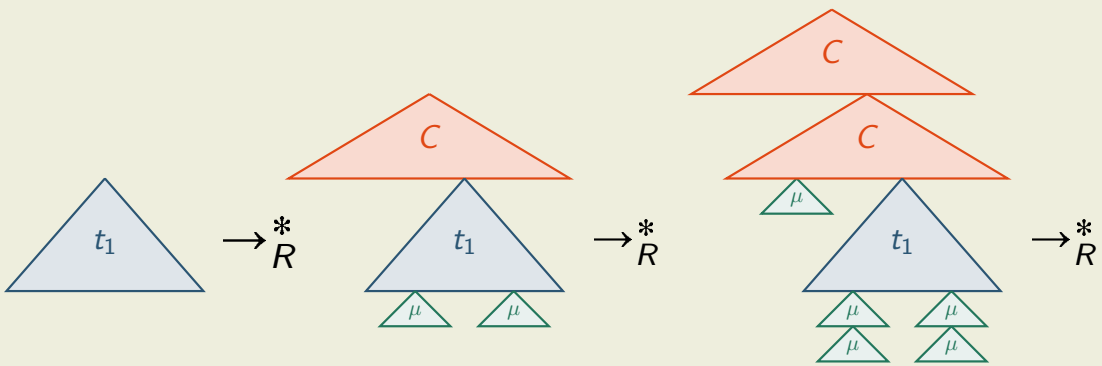
Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* C[t_1\mu] \to_R^* C[C[t_1\mu]\mu] \to_R^* C[C[C[t_1\mu]\mu]\mu] \to_R^* \ldots$$



Looking At Loops

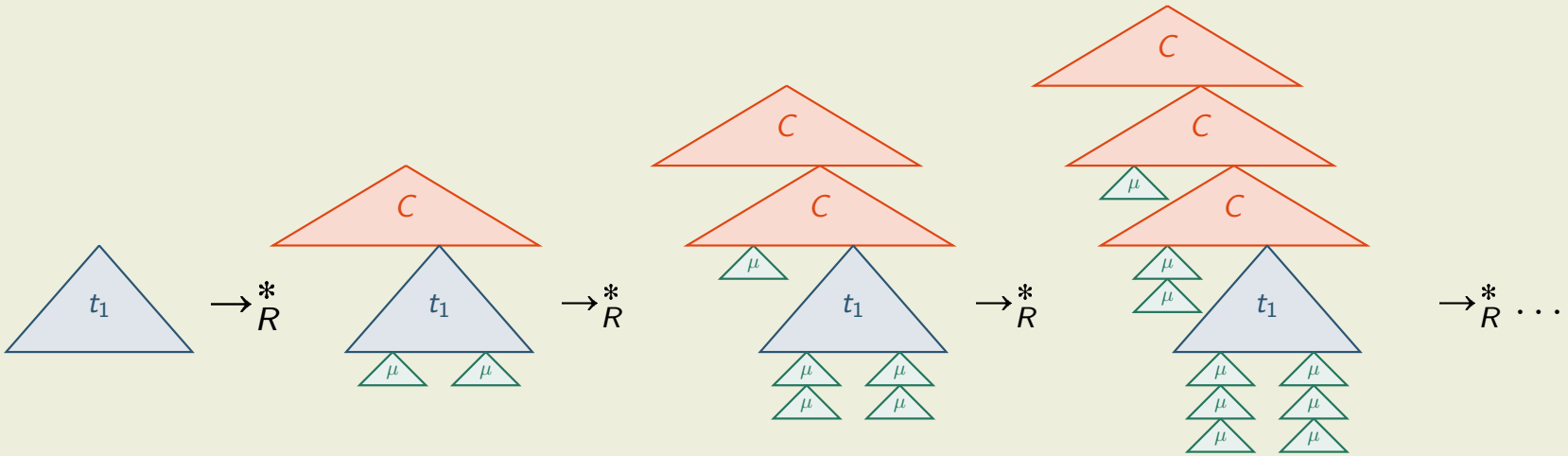
Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* C[t_1\mu] \to_R^* C[C[t_1\mu]\mu] \to_R^* C[C[C[t_1\mu]\mu]\mu] \to_R^* \ldots$$



Looking At Loops

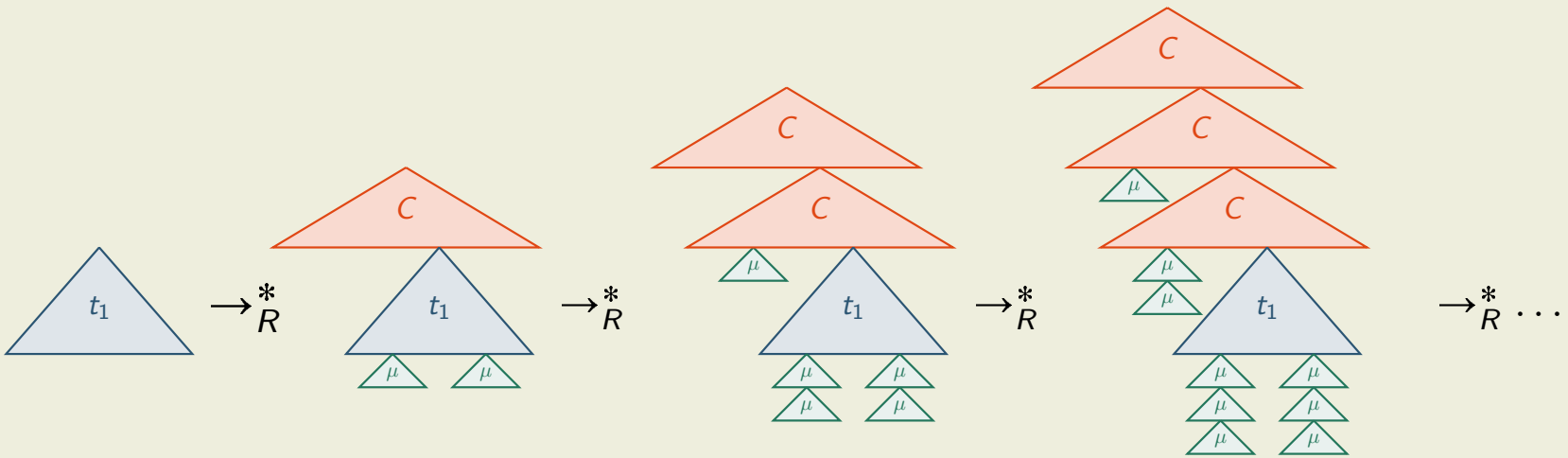
Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* t_1(C,\mu)^1 \to_R^* t_1(C,\mu)^2 \to_R^* t_1(C,\mu)^3 \to_R^* \ldots$$



Looking At Loops

Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* t_1(C,\mu)^1 \to_R^* t_1(C,\mu)^2 \to_R^* t_1(C,\mu)^3 \to_R^* \ldots$$

Definition (Π-Loop)

Loop is Π-loop if every step in infinite rewrite sequence is valid Π-step.

Looking At Loops

Definition (Loop)

Rewrite sequence $t_1 \to_{q_1} t_2 \to_{q_2} \cdots \to_{q_{l-1}} t_l$ is loop if $t_l = C[t_1\mu]$.

Remark

Loop gives rise to infinite rewrite sequence

$$t_1 \to_R^* t_1(C,\mu)^1 \to_R^* t_1(C,\mu)^2 \to_R^* t_1(C,\mu)^3 \to_R^* \ldots$$

Definition (Π-Loop)

Loop is Π-loop if every step in infinite rewrite sequence is valid Π-step.

... to check whether loop constitutes Π-loop:
check every step $t_i \to t_{i+1}$ against every $\pi \in \Pi$

Decision Problem (Case H)

($\star$) Given loop step $t_i \to_q t_{i+1}$ with $C|_p = \Box$ and pattern (ℓ, o, H):
are there n, o', σ such that $t_i(C, \mu)^n|_{o'} = \ell\sigma$ and $p^n q = o' o$?

Decision Problem (Case H)

($\star$) Given loop step $t_i \to_q t_{i+1}$ with $C|_p = \Box$ and pattern (ℓ, o, H):
are there n, o', σ such that $t_i(C, \mu)^n|_{o'} = \ell\sigma$ and $p^n q = o' o$?

Definition (Matching Problem)

Matching problem is pair $(t \trianglerighteq \ell, \mu)$ for terms t, ℓ and substitution μ.

Decision Problem (Case H)

($\star$) Given loop step $t_i \to_q t_{i+1}$ with $C|_p = \Box$ and pattern (ℓ, o, H):
are there n, o', σ such that $t_i(C,\mu)^n|_{o'} = \ell\sigma$ and $p^n q = o' o$?

Definition (Matching Problem)

Matching problem is pair $(t \trianglerighteq \ell, \mu)$ for terms t, ℓ and substitution μ.
It is solvable if there are n, σ such that $t\mu^n = \ell\sigma$.

Decision Problem (Case H)

($\star$) Given loop step $t_i \to_q t_{i+1}$ with $C|_p = \Box$ and pattern (ℓ, o, H):
are there n, o', σ such that $t_i(C, \mu)^n|_{o'} = \ell\sigma$ and $p^n q = o' o$?

Definition (Matching Problem)

Matching problem is pair $(t \trianglerighteq \ell, \mu)$ for terms t, ℓ and substitution μ.
It is solvable if there are n, σ such that $t\mu^n = \ell\sigma$.

Theorem (Thiemann *et al* '08, '09)

Solvability of matching problems is decidable.

Decision Problem (Case H)

($\star$) Given loop step $t_i \to_q t_{i+1}$ with $C|_p = \Box$ and pattern (ℓ, o, H):
are there n, o', σ such that $t_i(C,\mu)^n|_{o'} = \ell\sigma$ and $p^n q = o' o$?

Definition (Matching Problem)

Matching problem is pair $(t \trianglerighteq \ell, \mu)$ for terms t, ℓ and substitution μ.
It is solvable if there are n, σ such that $t\mu^n = \ell\sigma$.

Theorem (Thiemann *et al* '08, '09)

Solvability of matching problems is decidable.

Lemma (Case H)

Answer to ($\star$) is "no" iff $M_{\mathsf{H}} = \{(t(C,\mu)^{n_0}|_{o_0} \trianglerighteq \ell, \mu)\}$ has no solution.

Decision Problem (Case H)

($\star$) Given loop step $t_i \to_q t_{i+1}$ with $C|_p = \Box$ and pattern (ℓ, o, H):
are there n, o', σ such that $t_i(C,\mu)^n|_{o'} = \ell\sigma$ and $p^n q = o' o$?

Definition (Matching Problem)

Matching problem is pair $(t \trianglerighteq \ell, \mu)$ for terms t, ℓ and substitution μ.
It is solvable if there are n, σ such that $t\mu^n = \ell\sigma$.

Theorem (Thiemann *et al* '08, '09)

Solvability of mat

similarly, cases for A, B, and R can be decided
using appropriate sets of (extended) matching problems

Lemma (Case H)

Answer to ($\star$) is "no" iff $M_{\mathsf{H}} = \{(t(C,\mu)^{n_0}|_{o_0} \trianglerighteq \ell, \mu)\}$ has no solution.

The IsaFoRSide

- forbidden pattern loops cover in particular loops for innermost, outermost, context-sensitive and leftmost strategies
- formalization mostly follows paper proofs
- simplified procedure to solve extended matching problems

The IsaFoRSide

- forbidden pattern loops cover in particular loops for innermost, outermost, context-sensitive and leftmost strategies
- formalization mostly follows paper proofs
- simplified procedure to solve extended matching problems

Definition (Extended Matching Problem)

Extended matching problem is tuple $(D \rhd \ell, C, t, \mathcal{M}, \mu)$ for contexts D, C, terms u, ℓ, $\mathcal{M} = \{s_1 \rhd \ell_1, \ldots, s_m \rhd \ell_m\}$ and substitution μ.

The IsaFoRSide

- forbidden pattern loops cover in particular loops for innermost, outermost, context-sensitive and leftmost strategies
- formalization mostly follows paper proofs
- simplified procedure to solve extended matching problems

Definition (Extended Matching Problem)

Extended matching problem is tuple $(D \rhd \ell, C, t, \mathcal{M}, \mu)$ for contexts D, C, terms u, ℓ, $\mathcal{M} = \{s_1 \rhd \ell_1, \ldots, s_m \rhd \ell_m\}$ and substitution μ.
It is solvable if $D[t(C,\mu)^m]\mu^k = \ell\sigma$ and $s_i\mu^k = \ell_i\sigma$ for some m, k, σ.

The IsaFoRSide

- forbidden pattern loops cover in particular loops for innermost, outermost, context-sensitive and leftmost strategies
- formalization mostly follows paper proofs
- simplified procedure to solve extended matching problems

Definition (Extended Matching Problem)

Extended matching problem is tuple $(D \rhd \ell, C, t, \mathcal{M}, \mu)$ for contexts D, C, terms u, ℓ, $\mathcal{M} = \{s_1 \rhd \ell_1, \ldots, s_m \rhd \ell_m\}$ and substitution μ.
It is solvable if $D[t(C,\mu)^m]\mu^k = \ell\sigma$ and $s_i\mu^k = \ell_i\sigma$ for some m, k, σ.

3 Phases of Solving Procedure

1. transform to solved form
2. build respective (extended) identity problems
3. solve (extended) identity problems

The IsaFoRSide

- forbidden pattern loops cover in particular loops for innermost, outermost, context-sensitive and leftmost strategies
- formalization mostly follows paper proofs
- simplified procedure to solve extended matching problems

Definition (Extended Matching Problem)

Extended matching problem is tuple $(D \trianglerighteq \ell, C, t, \mathcal{M}, \mu)$ for contexts D, C, terms u, ℓ, $\mathcal{M} = \{s_1 \trianglerighteq \ell_1, \ldots, s_m \trianglerighteq \ell_m\}$ and substitution μ.
It is solvable if $D[t(C,\mu)^m]\mu^k = \ell\sigma$ and $s_i\mu^k = \ell_i\sigma$ for some m, k, σ.

3 Phases of Solving Procedure

1. transform to solved form
2. build respective (extended) identity problems
3. solve (extended) identity problems

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathsf{MP} = (D \Rrightarrow \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathsf{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathsf{MP} = (D \succcurlyeq \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathsf{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

(i) $\mathsf{MP} \Rightarrow (D_{i'} \succcurlyeq \ell'_{i'}, C, t, \mathcal{M} \cup \{t_i \succcurlyeq \ell'_i \mid 1 \leqslant i \leqslant m', i \neq i'\}, \mu)$ if $j = 0$ and $D = f(t_1, \ldots, D_{i'}, \ldots, t_{m'})$.

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathsf{MP} = (D \trianglerighteq \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathsf{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

(i) $\mathsf{MP} \Rightarrow (D_{i'} \trianglerighteq \ell'_{i'}, C, t, \mathcal{M} \cup \{t_i \trianglerighteq \ell'_i \mid 1 \leqslant i \leqslant m', i \neq i'\}, \mu)$ if $j = 0$ and $D = f(t_1, \ldots, D_{i'}, \ldots, t_{m'})$.

(iii) $\mathsf{MP} \Rightarrow \bot$ if $j = 0$ and $D = g(\ldots)$ where $f \neq g$.

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathrm{MP} = (D \trianglerighteq \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathsf{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

(i) $\mathrm{MP} \Rightarrow (D_{i'} \trianglerighteq \ell'_{i'}, C, t, \mathcal{M} \cup \{t_i \trianglerighteq \ell'_i \mid 1 \leqslant i \leqslant m', i \neq i'\}, \mu)$ if $j = 0$ and $D = f(t_1, \ldots, D_{i'}, \ldots, t_{m'})$.

(iii) $\mathrm{MP} \Rightarrow \bot$ if $j = 0$ and $D = g(\ldots)$ where $f \neq g$.

(vi) $\mathrm{MP} \Rightarrow (D\mu \trianglerighteq \ell_0, C\mu, t\mu, \{s_i\mu \trianglerighteq \ell_i \mid 1 \leqslant i \leqslant m\}, \mu)$ if $s_j \in \mathcal{V}_{\mathsf{incr},\mu}$

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathsf{MP} = (D \trianglerighteq \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathsf{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

(i) $\mathsf{MP} \Rightarrow (D_{i'} \trianglerighteq \ell'_{i'}, C, t, \mathcal{M} \cup \{t_i \trianglerighteq \ell'_i \mid 1 \leqslant i \leqslant m', i \neq i'\}, \mu)$ if $j = 0$ and $D = f(t_1, \ldots, D_{i'}, \ldots, t_{m'})$.

(iii) $\mathsf{MP} \Rightarrow \bot$ if $j = 0$ and $D = g(\ldots)$ where $f \neq g$.

(vi) $\mathsf{MP} \Rightarrow (D\mu \trianglerighteq \ell_0, C\mu, t\mu, \{s_i\mu \trianglerighteq \ell_i \mid 1 \leqslant i \leqslant m\}, \mu)$ if $s_j \in \mathcal{V}_{\mathsf{incr},\mu}$

In IsaFoR: Using Isabelle's `function` Package

challenge: get both termination and efficiency

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathsf{MP} = (D \trianglerighteq \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathsf{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

(i) $\mathsf{MP} \Rightarrow (D_{i'} \trianglerighteq \ell'_{i'}, C, t, \mathcal{M} \cup \{t_i \trianglerighteq \ell'_i \mid 1 \leqslant i \leqslant m', i \neq i'\}, \mu)$ if $j = 0$ and $D = f(t_1, \ldots, D_{i'}, \ldots, t_{m'})$.

(iii) $\mathsf{MP} \Rightarrow \bot$ if $j = 0$ and $D = g(\ldots)$ where $f \neq g$.

(vi) $\mathsf{MP} \Rightarrow (D\mu \trianglerighteq \ell_0, C\mu, t\mu, \{s_i\mu \trianglerighteq \ell_i \mid 1 \leqslant i \leqslant m\}, \mu)$ if $s_j \in \mathcal{V}_{\mathsf{incr},\mu}$

In IsaFoR: Using Isabelle's

i.e., do not recompute $\mathcal{V}_{\mathsf{incr},\mu}$ in every iteration

challenge: get both termination and efficiency

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathrm{MP} = (D \trianglerighteq \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathsf{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

(i) $\mathrm{MP} \Rightarrow (D_{i'} \trianglerighteq \ell'_{i'}, C, t, \mathcal{M} \cup \{t_i \trianglerighteq \ell'_i \mid 1 \leqslant i \leqslant m', i \neq i'\}, \mu)$ if $j = 0$ and $D = f(t_1, \ldots, D_{i'}, \ldots, t_{m'})$.

(iii) $\mathrm{MP} \Rightarrow \bot$ if $j = 0$ and $D = g(\ldots)$ where $f \neq g$.

(vi) $\mathrm{MP} \Rightarrow (D\mu \trianglerighteq \ell_0, C\mu, t\mu, \{s_i\mu \trianglerighteq \ell_i \mid 1 \leqslant i \leqslant m\}, \mu)$ if $s_j \in \mathcal{V}_{\mathsf{incr},\mu}$

In IsaFoR: Using Isabelle's `function` Package

challenge: get both termination and efficiency

- previously: two functions

Phase 1: Transformation to Solved Form

On Paper: Inference System

for $\mathrm{MP} = (D \trianglerighteq \ell, C, t, \mathcal{M}, \mu)$ and $\mathcal{V}_{\mathrm{incr},\mu} = \{x \in \mathcal{V} \mid \exists n : x\mu^n \notin \mathcal{V}\}$,

(i) $\mathrm{MP} \Rightarrow (D_{i'} \trianglerighteq \ell'_{i'}, C, t, \mathcal{M} \cup \{t_i \trianglerighteq \ell'_i \mid 1 \leqslant i \leqslant m', i \neq i'\}, \mu)$ if $j = 0$ and $D = f(t_1, \ldots, D_{i'}, \ldots, t_{m'})$.

(iii) $\mathrm{MP} \Rightarrow \bot$ if $j = 0$ and $D = g(\ldots)$ where $f \neq g$.

(vi) $\mathrm{MP} \Rightarrow (D\mu \trianglerighteq \ell_0, C\mu, t\mu, \{s_i\mu \trianglerighteq \ell_i \mid 1 \leqslant i \leqslant m\}, \mu)$ if $s_j \in \mathcal{V}_{\mathrm{incr},\mu}$

In IsaFoR: Using Isabelle's `function` Package

challenge: get both termination and efficiency

- previously: two functions
- new: pass (μ, V_i) as argument, encapsulated in type with invariant $V_i = \mathcal{V}_{\mathrm{incr},\mu}$, and provide selectors to both μ and V_i
  quotient construction with `lifting & transfer` package

The Tool Side

- AProVE supports innermost and outermost loops

The Tool Side

- AProVE supports innermost and outermost loops
- using Isabelle's code exportation feature to OCaml, we added support for forbidden pattern loops to $\mathsf{T_TT_2}$

The Tool Side

- AProVE supports innermost and outermost loops
- using Isabelle's code exportation feature to OCaml, we added support for forbidden pattern loops to $\mathsf{T_TT_2}$

The Certification Side

Demo

Outline

Motivation

Approach and Techniques

Forbidden Pattern Loops
- The Theory Side
- The IsaFoR Side
- The Tool and Certification Sides

Experiments and Conclusion

Experiments

Testbed

all 596 first-order TRSs from TPDB 8.0.7 where at least one tool generated nontermination proof in termination competition 2013

Experiments

Testbed

all 596 first-order TRSs from TPDB 8.0.7 where at least one tool generated nontermination proof in termination competition 2013

Tools

- AProVE'13 and T$_T$T$_2$'13 are versions restricted to (non)termination techniques that could be certified by CeTA in 2013
- AProVE'14 and T$_T$T$_2$'14 are full versions

Experiments

Testbed

all 596 first-order TRSs from TPDB 8.0.7 where at least one tool generated nontermination proof in termination competition 2013

Tools

- AProVE'13 and T$_T$T$_2$'13 are versions restricted to (non)termination techniques that could be certified by CeTA in 2013
- AProVE'14 and T$_T$T$_2$'14 are full versions

Results

	AProVE'13	AProVE'14
# nontermination proofs	276	575
# certified proofs	276	563

Experiments

Testbed

all 596 first-order TRSs from TPDB 8.0.7 where at least one tool generated nontermination proof in termination competition 2013

Tools

- AProVE'13 and T$_T$T$_2$'13 are versions restricted to (non)termination techniques that could be certified by CeTA in 2013
- AProVE'14 and T$_T$T$_2$'14 are full versions

Results

	AProVE'13	AProVE'14	T$_T$T$_2$'13	T$_T$T$_2$'14
# nontermination proofs	276	575	221	417
# certified proofs	276	563	221	417

Experiments

Testbed

all 596 first-order TRSs from TPDB 8.0.7 where at least one tool generated nontermination proof in termination competition 2013

Tools

- AProVE'13 and T$_T$T$_2$'13 are versions restricted to (non)termination techniques that could be certified by CeTA in 2013
- AProVE'14 and T$_T$T$_2$'14 are full versions

Results

certify all steps which apply known techniques

			T$_T$T$_2$'13	T$_T$T$_2$'14
# nontermination pr			221	417
# certified proofs	276	563	221	417
# partially certified proofs	–	12	–	–

Conclusion

Contributions

certification

- CeTA certifies 98% of nontermination proofs
- with partial certification, 70% of remaining proofs

Conclusion

Contributions

certification

- CeTA certifies 98% of nontermination proofs
- with partial certification, 70% of remaining proofs

theory

- drastically simplified algorithm to solve extended matching problems in (1)
- relax preconditions for technique (2)

Conclusion

Contributions

certification	▸ CeTA certifies 98% of nontermination proofs ▸ with partial certification, 70% of remaining proofs
theory	▸ drastically simplified algorithm to solve extended matching problems in (1) ▸ relax preconditions for technique (2)
software	▸ integrate executable functions from certifier into T_TT_2 ▸ detected bug in AProVE wrt method (3)

Conclusion

Contributions

certification	▸ CeTA certifies 98% of nontermination proofs ▸ with partial certification, 70% of remaining proofs
theory	▸ drastically simplified algorithm to solve extended matching problems in (1) ▸ relax preconditions for technique (2)
software	▸ integrate executable functions from certifier into $T_T T_2$ ▸ detected bug in AProVE wrt method (3)

Formalization & Details

`http://cl-informatik.uibk.ac.at/software/ntcert/`