Certification of Constrained Rewriting

René Thiemann
joint work with Christian Sternagel

Computational Logic, University of Innsbruck

05 July 2012

Outline

- Task 5: Certification
- IsaFoR + CeTA
- Progress in Task 5

Outline

- Task 5: Certification
- IsaFoR + CeTA
- Progress in Task 5

Overview of Task 5

- abstract goal: certification of constrained rewriting
- 3 concrete goals:

Overview of Task 5

- abstract goal: certification of constrained rewriting
- 3 concrete goals:
  1. formalization and certification of loops w.r.t. strategy
     (does a loop $t \to^+ C[t\mu]$ respect strategy w.r.t. forbidden patterns?)

Overview of Task 5

- abstract goal: certification of constrained rewriting
- 3 concrete goals:
  1. formalization and certification of loops w.r.t. strategy
     (does a loop $t \to^+ C[t\mu]$ respect strategy w.r.t. forbidden patterns?)

  2. formalization and certification of soundness and completeness results of unraveling transformation from conditional to unconditional rewriting

Overview of Task 5

- abstract goal: certification of constrained rewriting
- 3 concrete goals:
  1. formalization and certification of loops w.r.t. strategy
     (does a loop $t \to^+ C[t\mu]$ respect strategy w.r.t. forbidden patterns?)
  2. formalization and certification of soundness and completeness results of unraveling transformation from conditional to unconditional rewriting
  3. formalization and certification of constrained rewriting steps, taking output of general rewrite tool of Task 1

Overview of Task 5

- abstract goal: certification of constrained rewriting
- 3 concrete goals:
  1. formalization and certification of loops w.r.t. strategy (does a loop $t \to^+ C[t\mu]$ respect strategy w.r.t. forbidden patterns?)
  2. formalization and certification of soundness and completeness results of unraveling transformation from conditional to unconditional rewriting
  3. formalization and certification of constrained rewriting steps, taking output of general rewrite tool of Task 1
- working plan: late start

	1st year	2nd year	3rd year
Thiemann	1M	2M	1M
Zankl		1M	1M
PD2		6M	12M
Sternagel			2M

Goal 1: Loops w.r.t. strategy

Consider leftmost-outermost rewriting: $\xrightarrow{\mathsf{lo}}$

$$\begin{aligned} x == y &\to \mathsf{eq}(\mathsf{chk}(x), \mathsf{chk}(y)) \\ \mathsf{eq}(x, x) &\to \mathsf{true} \\ \mathsf{chk}(x) &\to \mathsf{false} \\ \mathsf{eq}(\mathsf{false}, y) &\to \mathsf{false} \end{aligned}$$

Above TRS can be used to check equality of arbitrary terms in constant number of reductions

- $t == t \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{chk}(t), \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{true}$

Goal 1: Loops w.r.t. strategy

Consider leftmost-outermost rewriting: $\xrightarrow{\mathsf{lo}}$

$$\begin{aligned} x == y &\to \mathsf{eq}(\mathsf{chk}(x), \mathsf{chk}(y)) \\ \mathsf{eq}(x, x) &\to \mathsf{true} \\ \mathsf{chk}(x) &\to \mathsf{false} \\ \mathsf{eq}(\mathsf{false}, y) &\to \mathsf{false} \end{aligned}$$

Above TRS can be used to check equality of arbitrary terms in constant number of reductions

- $t == t \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{chk}(t), \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{true}$
- $s == t \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{chk}(s), \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{false}, \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{false}$ if $s \neq t$

Goal 1: Loops w.r.t. strategy

Consider leftmost-outermost rewriting: $\xrightarrow{\mathsf{lo}}$

$$
\begin{aligned}
x == y &\to \mathsf{eq}(\mathsf{chk}(x), \mathsf{chk}(y)) \\
\mathsf{eq}(x, x) &\to \mathsf{true} \\
\mathsf{chk}(x) &\to \mathsf{false} \\
\mathsf{eq}(\mathsf{false}, y) &\to \mathsf{false}
\end{aligned}
$$

Above TRS can be used to check equality of arbitrary terms in constant number of reductions

- $t == t \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{chk}(t), \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{true}$
- $s == t \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{chk}(s), \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{false}, \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{false}$ if $s \neq t$

If strategy is ignored, unwanted behaviour can occur

- $t == t \to \mathsf{eq}(\mathsf{chk}(t), \mathsf{chk}(t)) \to \mathsf{eq}(\mathsf{false}, \mathsf{chk}(t)) \to \mathsf{false}$
- $s == t \to \mathsf{eq}(\mathsf{chk}(s), \mathsf{chk}(t)) \to \mathsf{eq}(\mathsf{chk}(t), \mathsf{false}) \to \mathsf{eq}(\mathsf{false}, \mathsf{false}) \to \mathsf{true}$

Goal 1: Loops w.r.t. strategy

Consider leftmost-outermost rewriting: $\xrightarrow{\mathsf{lo}}$

$$\begin{aligned} x == y &\to \mathsf{eq}(\mathsf{chk}(x), \mathsf{chk}(y)) \\ \mathsf{eq}(x, x) &\to \mathsf{true} \\ \mathsf{chk}(x) &\to \mathsf{false} \\ \mathsf{eq}(\mathsf{false}, y) &\to \mathsf{false} \end{aligned}$$

Above TRS can be used to check equality of arbitrary terms in constant number of reductions

- $t == t \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{chk}(t), \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{true}$
- $s == t \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{chk}(s), \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{eq}(\mathsf{false}, \mathsf{chk}(t)) \xrightarrow{\mathsf{lo}} \mathsf{false}$ if $s \neq t$

If strategy is ignored, unwanted behaviour can occur

- $t == t \to \mathsf{eq}(\mathsf{chk}(t), \mathsf{chk}(t)) \to \mathsf{eq}(\mathsf{false}, \mathsf{chk}(t)) \to \mathsf{false}$
- $s == t \to \mathsf{eq}(\mathsf{chk}(s), \mathsf{chk}(t)) \to \mathsf{eq}(\mathsf{chk}(t), \mathsf{false}) \to \mathsf{eq}(\mathsf{false}, \mathsf{false}) \to \mathsf{true}$

$\Rightarrow$ (Non-)Termination analysis has to be strategy aware

Goal 1: Loops w.r.t. strategy

Considered strategies for rewrite step $s \rightarrow_p t$

- innermost (no redex strictly below position p)
- outermost (no redex strictly above position p)
- forbidden patterns generalizes innermost and outermost strategies (includes flexible combination of no redex below/at/above position p)

Goal 1: Loops w.r.t. strategy

Considered strategies for rewrite step $s \to_p t$

- innermost (no redex strictly below position p)
- outermost (no redex strictly above position p)
- forbidden patterns generalizes innermost and outermost strategies (includes flexible combination of no redex below/at/above position p)

Loop under strategy

- loop $t \to \dots u \to_p v \to \dots C[t\mu]_q$ respects strategy iff reduction

$$\underbrace{C[C[C[\dots C[}_{n}u\,\underbrace{\mu]\dots\mu]\mu]\mu]}_{n} \to_{q\dots qp} \underbrace{C[C[C[\dots C[}_{n}v\,\underbrace{\mu]\dots\mu]\mu]\mu]}_{n}$$

respects strategy for all n and all steps $u \to_p v$ in the loop

Goal 1: Loops w.r.t. strategy

Considered strategies for rewrite step $s \to_p t$

- innermost (no redex strictly below position p)
- outermost (no redex strictly above position p)
- forbidden patterns generalizes innermost and outermost strategies (includes flexible combination of no redex below/at/above position p)

Loop under strategy

- loop $t \to \dots u \to_p v \to \dots C[t\mu]_q$ respects strategy iff reduction

$$\underbrace{C[C[C[\dots C[}_{n} u \underbrace{\mu]\dots\mu]\mu]\mu]}_{n} \to_{q\dots qp} \underbrace{C[C[C[\dots C[}_{n} v \underbrace{\mu]\dots\mu]\mu]\mu]}_{n}$$

  respects strategy for all n and all steps $u \to_p v$ in the loop
- decidable for
  - context-sensitive rewriting (trivial)
  - innermost strategy (difficult)
  - outermost strategy (even more difficult)
  - forbidden patterns (most difficult)

Goal 2: Unraveling

- Unraveling is transformation $\mathcal{U}$ from conditional to standard rewriting
- Evaluation of conditions is encoded using U-symbols
- Variations in how many auxiliary variables are used as arguments of U

Goal 2: Unraveling

- Unraveling is transformation $\mathcal{U}$ from conditional to standard rewriting
- Evaluation of conditions is encoded using U-symbols
- Variations in how many auxiliary variables are used as arguments of U
- Example:

$$
\begin{array}{ll}
\mathsf{div}(x, \mathsf{s}(y)) \to 0 & \mid x \leq y \to^* \mathsf{true} \\
\mathsf{div}(x, \mathsf{s}(y)) \to \mathsf{s}(\mathsf{div}(x - \mathsf{s}(y), \mathsf{s}(y))) & \mid x \leq y \to^* \mathsf{false}
\end{array}
$$

becomes

$$
\begin{array}{ll}
\mathsf{div}(x, \mathsf{s}(y)) \to \mathsf{U}_1(x \leq y, {\color{red}x}, {\color{red}y}) & \mathsf{U}_1(\mathsf{true}, {\color{red}x}, {\color{red}y}) \to 0 \\
\mathsf{div}(x, \mathsf{s}(y)) \to \mathsf{U}_2(x \leq y, x, y) & \mathsf{U}_2(\mathsf{false}, x, y) \to \mathsf{s}(\mathsf{div}(x - \mathsf{s}(y), \mathsf{s}(y)))
\end{array}
$$

red variables are subject to optimizations

Goal 2: Unraveling

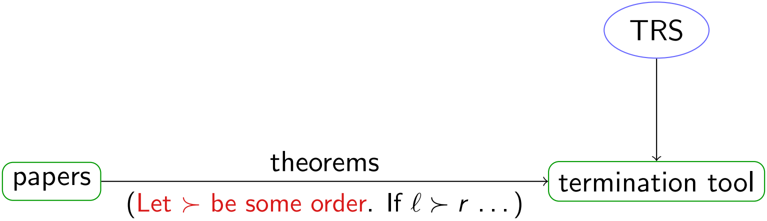
- Completeness of unraveling is easy: $s \rightarrow_{\mathcal{R}} t$ implies $s \rightarrow^{+}_{\mathcal{U}(\mathcal{R})} t$
- Soundness is much harder and does not always hold: if $s \rightarrow^{*}_{\mathcal{U}(\mathcal{R})} t$ then $s \rightarrow^{*}_{\mathcal{R}} t$ for s and t not containing U-symbols sufficient conditions like
  - ultra left linearity (for optimized and non-optimized unraveling)
  - ultra weak left linearity

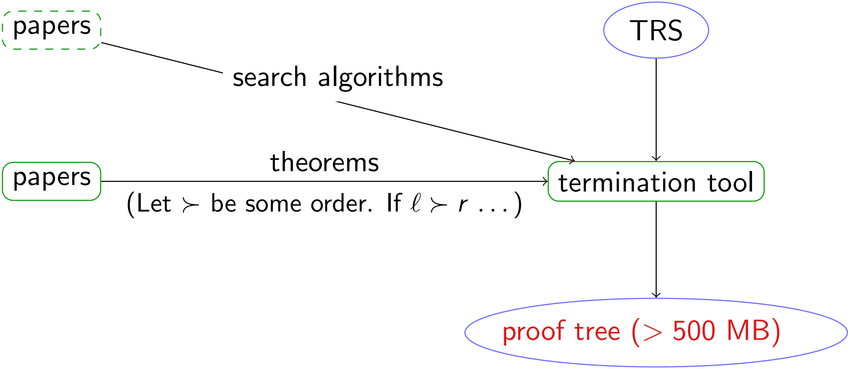
Goal 3: Certifying constrained rewriting

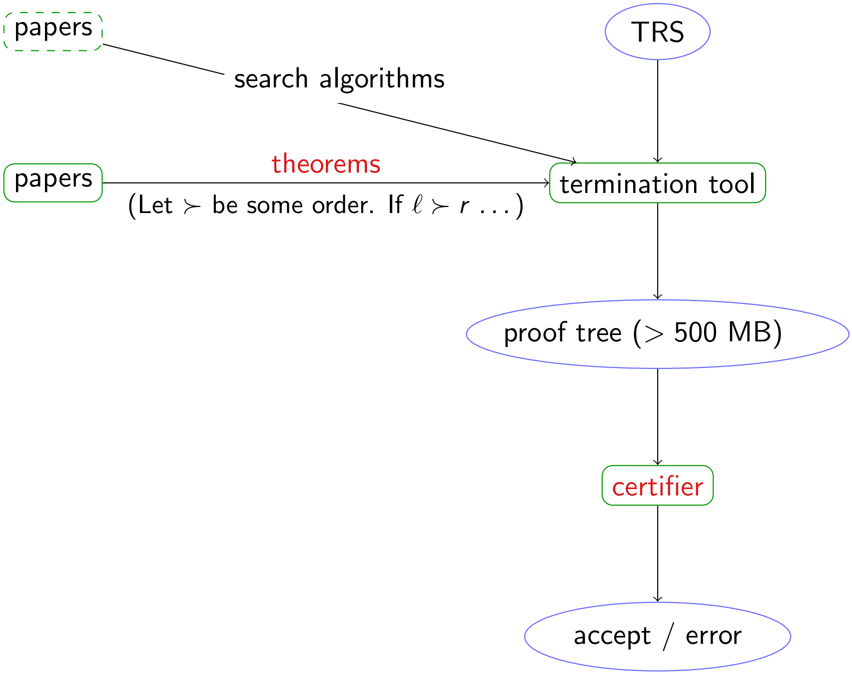
- Completely depends on results of Task 1:
  - what kinds of constraints?
  - definition of rewrite relation?
  - ...

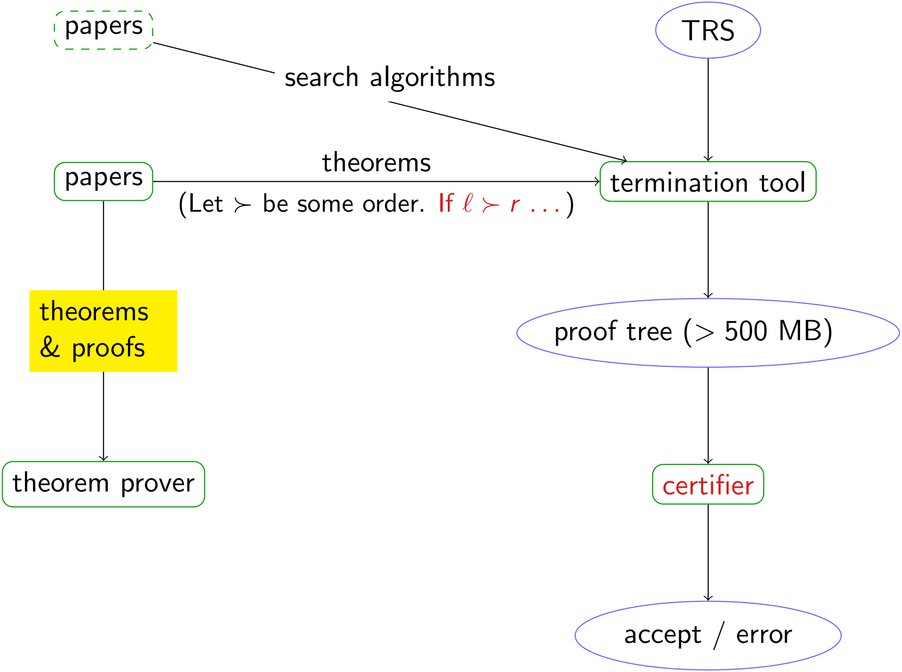
Outline

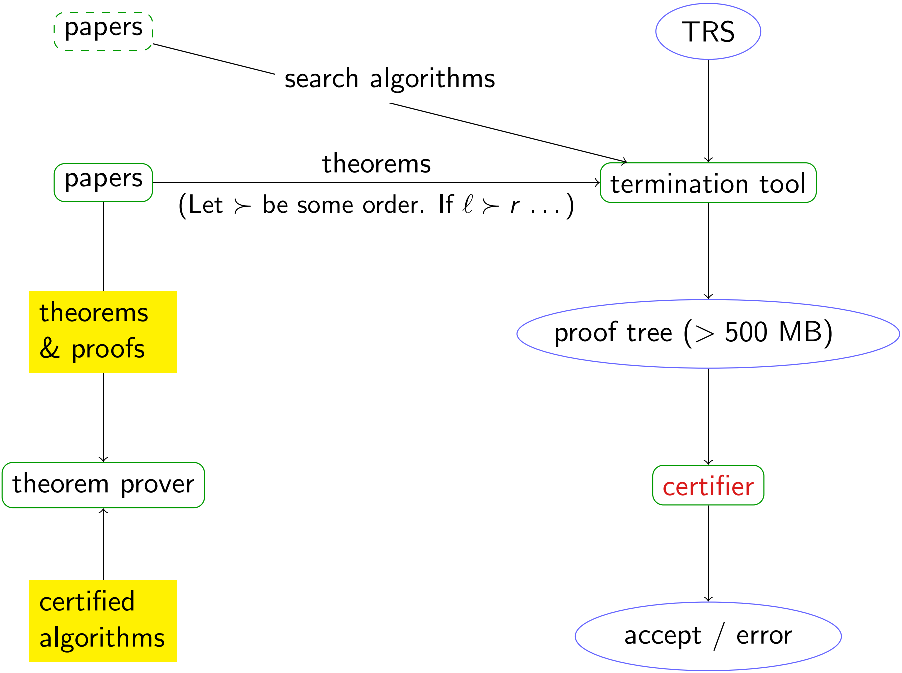
- Task 5: Certification
- IsaFoR + CeTA
- Progress in Task 5

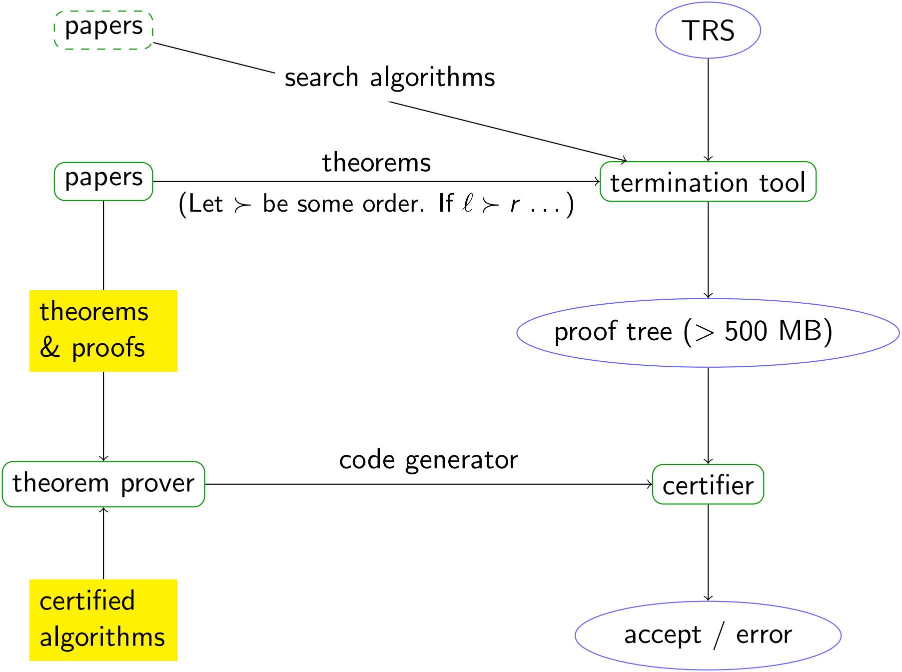

TRS
papers
theorems
(Let $\succ$ be some order. If $\ell \succ r$ ...)
termination tool

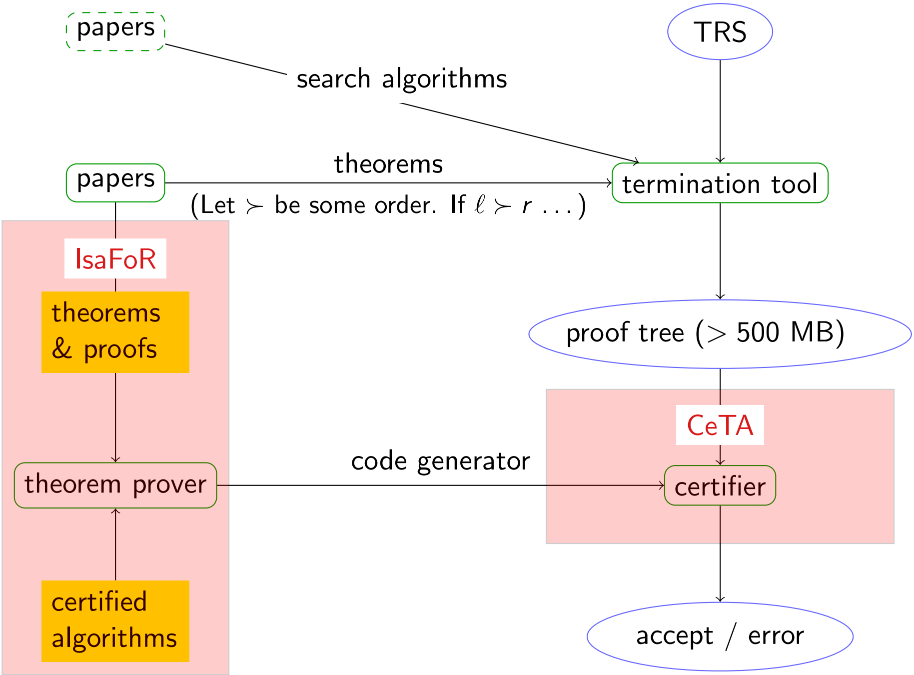

papers
TRS
search algorithms
theorems
papers
termination tool
(Let $\succ$ be some order. If $\ell \succ r$ ...)
proof tree ($>$ 500 MB)


papers
search algorithms
TRS
papers
theorems
(Let $\succ$ be some order. If $\ell \succ r$ . . .)
termination tool
proof tree (> 500 MB)
certifier
accept / error


papers
search algorithms
TRS
papers
theorems
termination tool
(Let $\succ$ be some order. If $\ell \succ r$ ...)
theorems
& proofs
proof tree (> 500 MB)
theorem prover
certifier
accept / error


papers
TRS
search algorithms
papers
theorems
termination tool
(Let ≻ be some order. If ℓ ≻ r . . .)
theorems
& proofs
proof tree (> 500 MB)
theorem prover
certifier
certified
algorithms
accept / error


papers
search algorithms
TRS
papers
theorems
termination tool
(Let $\succ$ be some order. If $\ell \succ r$ . . .)
theorems
& proofs
proof tree ($>$ 500 MB)
code generator
theorem prover
certifier
certified
algorithms
accept / error


papers
TRS
search algorithms
papers
theorems
termination tool
(Let ≻ be some order. If ℓ ≻ r . . .)
IsaFoR
theorems
& proofs
proof tree (> 500 MB)
CeTA
code generator
theorem prover
certifier
certified
algorithms
accept / error

Isabelle/HOL

- Isabelle/HOL: Interactive theorem proving for higher order logic
- HOL: functional programming + theorem proving

Isabelle/HOL

- Isabelle/HOL: Interactive theorem proving for higher order logic
- HOL: functional programming + theorem proving

(∗ *programming* ∗)
datatype 'a list = *Nil* | *Cons 'a ('a list)*
fun insert where
 insert x Nil = *Cons x Nil*
 insert x (Cons y ys) = *if* $x \leq y$ *then Cons x (Cons y ys) else* ...

Isabelle/HOL

- Isabelle/HOL: Interactive theorem proving for higher order logic
- HOL: functional programming + theorem proving

$(*$ *programming* $*)$
datatype $'a\ list = Nil \mid Cons\ 'a\ ('a\ list)$
fun $insert$ *where*
$insert\ x\ Nil = Cons\ x\ Nil$
$insert\ x\ (Cons\ y\ ys) = if\ \ x \leq y\ \ then\ \ Cons\ x\ (Cons\ y\ ys)\ \ else \ldots$

$(*$ *specifying* $*)$
$fun\ sorted\ \ where$
$sorted\ \ Nil = True$
$sorted\ (Cons\ x\ Nil) = True$
$sorted\ (Cons\ x\ (Cons\ y\ ys)) = x \leq y \wedge sorted\ (Cons\ y\ ys)$

Isabelle/HOL

- Isabelle/HOL: Interactive theorem proving for higher order logic
- HOL: functional programming + theorem proving

$(*\ \mathit{programming}\ *)$
$\mathit{datatype}\ {}'a\ \mathit{list} = \mathit{Nil} \mid \mathit{Cons}\ {}'a\ ({}'a\ \mathit{list})$
$\mathit{fun}\ \mathit{insert}\ \mathit{where}$
$\ \mathit{insert}\ x\ \mathit{Nil} = \mathit{Cons}\ x\ \mathit{Nil}$
$\ \mathit{insert}\ x\ (\mathit{Cons}\ y\ \mathit{ys}) = \mathit{if}\ x \leq y\ \mathit{then}\ \mathit{Cons}\ x\ (\mathit{Cons}\ y\ \mathit{ys})\ \mathit{else} \ldots$

$(*\ \mathit{specifying}\ *)$
$\mathit{fun}\ \mathit{sorted}\ \mathit{where}$
$\ \mathit{sorted}\ \mathit{Nil} = \mathit{True}$
$\ \mathit{sorted}\ (\mathit{Cons}\ x\ \mathit{Nil}) = \mathit{True}$
$\ \mathit{sorted}\ (\mathit{Cons}\ x\ (\mathit{Cons}\ y\ \mathit{ys})) = x \leq y \wedge \mathit{sorted}\ (\mathit{Cons}\ y\ \mathit{ys})$

$(*\ \mathit{proving}\ *)$
$\mathit{lemma} : \mathit{sorted}\ \mathit{xs} \Longrightarrow \mathit{sorted}\ (\mathit{insert}\ x\ \mathit{xs})$
$\mathit{proof}\,(\mathit{induction}\ \mathit{xs} \ldots)$

IsaFoR: Isabelle/HOL Formalization of Rewriting

- library on rewriting, deep embedding (TRS is data object)

$$\begin{array}{ll} \mathit{datatype}\ ('f,'v)\mathit{term} & = \mathit{Var}\ 'v \mid \mathit{Fun}\ 'f\ (('f,'v)\mathit{term}\ \mathit{list}) \\ \mathit{type\_synonym}\ ('f,'v)\mathit{trs} & = (('f,\ 'v)\mathit{term} \times ('f,'v)\mathit{term})\ \mathit{set} \end{array}$$

IsaFoR: Isabelle/HOL Formalization of Rewriting

- library on rewriting, deep embedding (TRS is data object)

$$\begin{aligned} &\mathit{datatype}\ ('f,'v)\mathit{term} && = \mathit{Var}\ 'v \mid \mathit{Fun}\ 'f\ (('f,'v)\mathit{term}\ \mathit{list}) \\ &\mathit{type\_synonym}\ ('f,'v)\mathit{trs} && = (('f,\ 'v)\mathit{term} \times ('f,'v)\mathit{term})\ \mathit{set} \end{aligned}$$

- supported constraints:

IsaFoR: Isabelle/HOL Formalization of Rewriting

- library on rewriting, deep embedding (TRS is data object)

$$\begin{aligned} &datatype\ ('f,'v)term &&= Var\ 'v \mid Fun\ 'f\ (('f,'v)term\ list) \\ &type\_synonym\ ('f,'v)trs &&= (('f,\ 'v)term \times ('f,'v)term)\ set \end{aligned}$$

- supported constraints:
  - innermost strategy (many results)

IsaFoR: Isabelle/HOL Formalization of Rewriting

- library on rewriting, deep embedding (TRS is data object)

$$\begin{aligned} &\textit{datatype } ('f,'v)\textit{term} &&= \textit{Var } 'v \mid \textit{Fun } 'f \ (('f,'v)\textit{term list}) \\ &\textit{type\_synonym } ('f,'v)\textit{trs} &&= (('f,\ 'v)\textit{term} \times ('f,'v)\textit{term}) \textit{ set} \end{aligned}$$

- supported constraints:
  - innermost strategy (many results)
  - substitution constraint *nfs* to allow free variables in right-hand sides:

$$C[\ell\sigma] \overset{\mathit{nfs}}{\rightarrow}{}^{\mathcal{Q}}_{\mathcal{R}} C[r\sigma] \text{ for } \ell \to r \in \mathcal{R} \text{ if}$$

  - all arguments of $\ell\sigma$ are in normal form w.r.t. $\to_{\mathcal{Q}}$ (innermost)
  - if *nfs* then $x\sigma$ is in normal form w.r.t. $\to_{\mathcal{Q}}$ for all $x \in \mathcal{V}(\ell) \cup \mathcal{V}(r)$

IsaFoR: Isabelle/HOL Formalization of Rewriting

- library on rewriting, deep embedding (TRS is data object)

$$\begin{array}{ll} \textit{datatype}\ ('f,'v)\textit{term} & = \textit{Var}\ 'v \mid \textit{Fun}\ 'f\ (('f,'v)\textit{term list}) \\ \textit{type\_synonym}\ ('f,'v)\textit{trs} & = ((('f,\ 'v)\textit{term} \times ('f,'v)\textit{term})\ \textit{set} \end{array}$$

- supported constraints:
  - innermost strategy (many results)
  - substitution constraint *nfs* to allow free variables in right-hand sides:

$$C[\ell\sigma] \overset{\mathit{nfs}}{\rightarrow}{}^{\mathcal{Q}}_{\mathcal{R}} C[r\sigma] \text{ for } \ell \to r \in \mathcal{R} \text{ if}$$

    - all arguments of $\ell\sigma$ are in normal form w.r.t. $\to_{\mathcal{Q}}$ (innermost)
    - if *nfs* then $x\sigma$ is in normal form w.r.t. $\to_{\mathcal{Q}}$ for all $x \in \mathcal{V}(\ell) \cup \mathcal{V}(r)$
  - outermost strategy (few results)

IsaFoR: Isabelle/HOL Formalization of Rewriting

- library on rewriting, deep embedding (TRS is data object)

$$\begin{aligned} &\mathit{datatype}\ ('f,'v)\mathit{term} &&= \mathit{Var}\ 'v \mid \mathit{Fun}\ 'f\ (('f,'v)\mathit{term}\ \mathit{list}) \\ &\mathit{type\_synonym}\ ('f,'v)\mathit{trs} &&= (('f,\ 'v)\mathit{term} \times ('f,'v)\mathit{term})\ \mathit{set} \end{aligned}$$

- supported constraints:
  - innermost strategy (many results)
  - substitution constraint *nfs* to allow free variables in right-hand sides:

$$C[\ell\sigma] \overset{\mathit{nfs}}{\to}{}^{\mathcal{Q}}_{\mathcal{R}} C[r\sigma] \text{ for } \ell \to r \in \mathcal{R} \text{ if}$$

    - all arguments of $\ell\sigma$ are in normal form w.r.t. $\to_{\mathcal{Q}}$ (innermost)
    - if *nfs* then $x\sigma$ is in normal form w.r.t. $\to_{\mathcal{Q}}$ for all $x \in \mathcal{V}(\ell) \cup \mathcal{V}(r)$
  - outermost strategy (few results)
- focus on termination criteria

IsaFoR: Isabelle/HOL Formalization of Rewriting

- library on rewriting, deep embedding (TRS is data object)

$$\begin{aligned} &\mathit{datatype}\ ('f,'v)\mathit{term} &&= \mathit{Var}\ 'v \mid \mathit{Fun}\ 'f\ (('f,'v)\mathit{term}\ \mathit{list}) \\ &\mathit{type\_synonym}\ ('f,'v)\mathit{trs} &&= (('f,\ 'v)\mathit{term} \times ('f,'v)\mathit{term})\ \mathit{set} \end{aligned}$$

- supported constraints:
  - innermost strategy (many results)
  - substitution constraint *nfs* to allow free variables in right-hand sides:

$$C[\ell\sigma] \overset{\mathit{nfs}}{\to}{}^{\mathcal{Q}}_{\mathcal{R}} C[r\sigma] \text{ for } \ell \to r \in \mathcal{R} \text{ if}$$

    - all arguments of $\ell\sigma$ are in normal form w.r.t. $\to_{\mathcal{Q}}$ (innermost)
    - if *nfs* then $x\sigma$ is in normal form w.r.t. $\to_{\mathcal{Q}}$ for all $x \in \mathcal{V}(\ell) \cup \mathcal{V}(r)$
  - outermost strategy (few results)
- focus on termination criteria
- recently added results on confluence, complexity, completion, and equational reasoning

Supported termination techniques

- Dependency Pairs
- Dependency Pair Framework
- (Innermost) Dependency Graph Processor
- (Innermost) DP Transformations Narrowing, Rewriting, and (Forward)-Instantiation
- Flat Context Closure
- Innermost Switch
- Innermost- and Outermost-Loops
- Match- and roof-bounds
- Reduction Pair Processors (poly, matrix, LPO, RPO, KBO, ...)
- Root-Labeling
- Semantic Labeling
- Size-Change Termination
- String Reversal
- Subterm Criterion
- Uncurrying
- (Innermost) Usable Rules

CeTA: Certified Termination Analysis

- CeTA is executable proof checker within IsaFoR

CeTA: Certified Termination Analysis

- CeTA is executable proof checker within IsaFoR
  - load IsaFoR in Isabelle

CeTA: Certified Termination Analysis

- CeTA is executable proof checker within IsaFoR
  - load IsaFoR in Isabelle
  - `export-code`
    `(check-proof :: string -> certification-result)`
    `in Haskell`

CeTA: Certified Termination Analysis

- CeTA is executable proof checker within IsaFoR
  - load IsaFoR in Isabelle
  - `export-code (check-proof :: string -> certification-result) in Haskell`
  - write Haskell file Main.hs to load file and call `check-proof`

CeTA: Certified Termination Analysis

- CeTA is executable proof checker within IsaFoR
  - load IsaFoR in Isabelle
  - `export-code`
    `(check-proof :: string -> certification-result)`
    `in Haskell`
  - write Haskell file Main.hs to load file and call `check-proof`
  - invoke Haskell-compiler and obtain CeTA as binary (outside Isabelle)

CeTA: Certified Termination Analysis

- CeTA is executable proof checker within IsaFoR
  - load IsaFoR in Isabelle
  - `export-code`
    `(check-proof :: string -> certification-result)`
    `in Haskell`
  - write Haskell file Main.hs to load file and call `check-proof`
  - invoke Haskell-compiler and obtain CeTA as binary (outside Isabelle)
- both IsaFoR and CeTA are easily available (repository or distribution)

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

Outline

- Task 5: Certification
- IsaFoR + CeTA
- Progress in Task 5

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Innermost Loops: original proof works in three phases

- Example: lhs $\mathsf{eq}(z, z)$, redex $\mathsf{f}(\mathsf{eq}(x, y))$, $\mu = \{x/\mathsf{f}(y), y/\mathsf{f}(x)\}$

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Innermost Loops: original proof works in three phases
  1. setting up matching problems:

$s\mu^n$ never matches lhs for any n

- Example: lhs $\mathsf{eq}(z, z)$, redex $\mathsf{f}(\mathsf{eq}(x, y))$, $\mu = \{x/\mathsf{f}(y), y/\mathsf{f}(x)\}$
  1. for $s = \mathsf{eq}(x, y)$ or $s = f(x)$ or $s = f(y)$, $s\mu^n$ does not match $\mathsf{eq}(z, z)$

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Innermost Loops: original proof works in three phases
  1. setting up matching problems:

     $$s\mu^n \text{ never matches lhs for any } n$$

  2. simplified to identity problems using matching algorithm:

     $$s\mu^n \neq t\mu^n \qquad \text{for all } n$$

- Example: lhs $\mathsf{eq}(z,z)$, redex $\mathsf{f}(\mathsf{eq}(x,y))$, $\mu = \{x/\mathsf{f}(y), y/\mathsf{f}(x)\}$
  1. for $s = \mathsf{eq}(x,y)$ or $s = f(x)$ or $s = f(y)$, $s\mu^n$ does not match $\mathsf{eq}(z,z)$
  2. $x\mu^n \neq y\mu^n$ for all n

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Innermost Loops: original proof works in three phases
  1. setting up matching problems:

     $$s\mu^n \text{ never matches lhs for any } n$$

  2. simplified to identity problems using matching algorithm:

     $$s\mu^n \neq t\mu^n \qquad \text{for all } n$$

  3. complex algorithm to decide identity problems
- Example: lhs $\mathsf{eq}(z, z)$, redex $\mathsf{f}(\mathsf{eq}(x, y))$, $\mu = \{x/\mathsf{f}(y), y/\mathsf{f}(x)\}$
  1. for $s = \mathsf{eq}(x, y)$ or $s = f(x)$ or $s = f(y)$, $s\mu^n$ does not match $\mathsf{eq}(z, z)$
  2. $x\mu^n \neq y\mu^n$ for all n
  3. True (the variable in both terms is toggling)

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Innermost Loops: original proof works in three phases
  1. setting up matching problems:

     $$s\mu^n \text{ never matches lhs for any } n$$

  2. simplified to identity problems using matching algorithm:

     $$s\mu^n \neq t\mu^n \qquad \text{for all } n$$

  3. complex algorithm to decide identity problems
- Example: lhs $\mathsf{eq}(z, z)$, redex $\mathsf{f}(\mathsf{eq}(x, y))$, $\mu = \{x/\mathsf{f}(y), y/\mathsf{f}(x)\}$
  1. for $s = \mathsf{eq}(x, y)$ or $s = f(x)$ or $s = f(y)$, $s\mu^n$ does not match $\mathsf{eq}(z, z)$
  2. $x\mu^n \neq y\mu^n$ for all n
  3. True (the variable in both terms is toggling)
- everything formalized, deviation from original proof in 3. step: improved complex algorithm, Kruskal no longer required

Goal 1: Loops under strategies for $t \to^{+} C[t\mu]$

- Outermost Loops: original proof works in three phases

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Outermost Loops: original proof works in three phases
  1. setting up matching and extended matching problems:

$$D[s(C,\mu)^k]\mu^n \text{ never matches lhs for any } n, k$$

$(C,\mu)^k$: k-times application of $u \mapsto C[u\mu]$

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Outermost Loops: original proof works in three phases
  1. setting up matching and extended matching problems:

     $$D[s(C,\mu)^k]\mu^n \text{ never matches lhs for any } n, k$$

     $(C,\mu)^k$: k-times application of $u \mapsto C[u\mu]$
  2. simplified to identity and extended identity problems:

     $$D[s(C,\mu)^k]\mu^n \neq t\mu^n \qquad \text{for all } n, k \tag{$\star$}$$

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Outermost Loops: original proof works in three phases
  1. setting up matching and extended matching problems:

     $$D[s(C,\mu)^k]\mu^n \text{ never matches lhs for any } n, k$$

     $(C,\mu)^k$: k-times application of $u \mapsto C[u\mu]$
  2. simplified to identity and extended identity problems:

     $$D[s(C,\mu)^k]\mu^n \neq t\mu^n \qquad \text{for all } n, k \qquad (\star)$$

  3. complex algorithm to decide extended identity problems

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- Outermost Loops: original proof works in three phases
  1. setting up matching and extended matching problems:

     $$D[s(C,\mu)^k]\mu^n \text{ never matches lhs for any } n, k$$

     $(C,\mu)^k$: k-times application of $u \mapsto C[u\mu]$
  2. simplified to identity and extended identity problems:

     $$D[s(C,\mu)^k]\mu^n \neq t\mu^n \qquad \text{for all } n, k \tag{$\star$}$$
  3. complex algorithm to decide extended identity problems
- main result formalized without any formalization of 3. step: for extended identity problems from outermost loops k in $(\star)$ can be fixed to 0
  $\Longrightarrow$ get non-extended identity problem

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- forbidden patterns: original proof works in two phases

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- forbidden patterns: original proof works in two phases
  1. setting up matching and extended matching problems (much more involved than in innermost and outermost case)

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- forbidden patterns: original proof works in two phases
    1. setting up matching and extended matching problems (much more involved than in innermost and outermost case)
    2. solve (extended) matching problems

Goal 1: Loops under strategies for $t \to^+ C[t\mu]$

- forbidden patterns: original proof works in two phases
  1. setting up matching and extended matching problems (much more involved than in innermost and outermost case)
  2. solve (extended) matching problems
- several open tasks
  - formalization of rewrite relation with forbidden patterns

Goal 1: Loops under strategies for $t \to^{+} C[t\mu]$

- forbidden patterns: original proof works in two phases
  1. setting up matching and extended matching problems (much more involved than in innermost and outermost case)
  2. solve (extended) matching problems
- several open tasks
  - formalization of rewrite relation with forbidden patterns
  - complete formalization of 1. step

Goal 1: Loops under strategies for $t \to^{+} C[t\mu]$

- forbidden patterns: original proof works in two phases
  1. setting up matching and extended matching problems (much more involved than in innermost and outermost case)
  2. solve (extended) matching problems
- several open tasks
  - formalization of rewrite relation with forbidden patterns
  - complete formalization of 1. step
  - figure out whether trick for outermost loops (fix $k = 0$) is also possible for extended identity problems from forbidden patterns;
    if not, formalize complex algorithm for extended identity problems

Goal 2: Unraveling

- last week: nothing formalized in IsaFoR

Goal 2: Unraveling

- last week: nothing formalized in IsaFoR
- this week: formalization of easy results
  - conditional rewrite relation, quasi-decreasingness

Goal 2: Unraveling

- last week: nothing formalized in IsaFoR
- this week: formalization of easy results
  - conditional rewrite relation, quasi-decreasingness
  - generic unraveling-transformation:
    for $\ell \to r \Leftarrow s_1 = t_1, \dots s_n = t_n$ unraveled rules are

$$\begin{aligned} \ell &\to C_1[s_1] \\ C_1[t_1] &\to C_2[s_2] \\ \dots &\to \dots \\ C_n[t_n] &\to r \end{aligned}$$

C_i are arbitrary, standard unraveling by choosing $C_1 = U_1(\,\cdot\,, \mathcal{V}(\ell)), \dots$

Goal 2: Unraveling

- last week: nothing formalized in IsaFoR
- this week: formalization of easy results
  - conditional rewrite relation, quasi-decreasingness
  - generic unraveling-transformation:
    for $\ell \to r \Leftarrow s_1 = t_1, \ldots s_n = t_n$ unraveled rules are

$$\begin{aligned} \ell &\to C_1[s_1] \\ C_1[t_1] &\to C_2[s_2] \\ \ldots &\to \ldots \\ C_n[t_n] &\to r \end{aligned}$$

    C_i are arbitrary, standard unraveling by choosing $C_1 = U_1(\cdot, \mathcal{V}(\ell)), \ldots$
  - completeness result:

$$SN(\mathcal{U}_{C_i}(\mathcal{R})) \Longrightarrow \mathit{quasi\text{-}decreasing}(\mathcal{R}) \Longrightarrow SN(\to_{\mathcal{R}})$$

Goal 2: Unraveling

- last week: nothing formalized in IsaFoR
- this week: formalization of easy results
  - conditional rewrite relation, quasi-decreasingness
  - generic unraveling-transformation:
    for $\ell \to r \Leftarrow s_1 = t_1, \dots s_n = t_n$ unraveled rules are

$$\begin{aligned} \ell &\to C_1[s_1] \\ C_1[t_1] &\to C_2[s_2] \\ \dots &\to \dots \\ C_n[t_n] &\to r \end{aligned}$$

    C_i are arbitrary, standard unraveling by choosing $C_1 = U_1(\cdot, \mathcal{V}(\ell)), \dots$
  - completeness result:

$$SN(\mathcal{U}_{C_i}(\mathcal{R})) \Longrightarrow \mathit{quasi\text{-}decreasing}(\mathcal{R}) \Longrightarrow SN(\to_{\mathcal{R}})$$

- several open tasks
  - certification algorithm for unraveling and suitable format

Goal 2: Unraveling

- last week: nothing formalized in IsaFoR
- this week: formalization of easy results
  - conditional rewrite relation, quasi-decreasingness
  - generic unraveling-transformation:
    for $\ell \to r \Leftarrow s_1 = t_1, \dots s_n = t_n$ unraveled rules are

$$\begin{aligned} \ell &\to C_1[s_1] \\ C_1[t_1] &\to C_2[s_2] \\ \dots &\to \dots \\ C_n[t_n] &\to r \end{aligned}$$

    C_i are arbitrary, standard unraveling by choosing $C_1 = U_1(\cdot, \mathcal{V}(\ell)), \dots$
  - completeness result:

$$SN(\mathcal{U}_{C_i}(\mathcal{R})) \Longrightarrow \mathit{quasi\text{-}decreasing}(\mathcal{R}) \Longrightarrow SN(\to_{\mathcal{R}})$$

- several open tasks
  - certification algorithm for unraveling and suitable format
  - algorithm to compute $\to_{\mathcal{R}}$ (for quasi-decreasing $\mathcal{R}$)

Goal 2: Unraveling

- last week: nothing formalized in IsaFoR
- this week: formalization of easy results
  - conditional rewrite relation, quasi-decreasingness
  - generic unraveling-transformation:
    for $\ell \to r \Leftarrow s_1 = t_1, \dots s_n = t_n$ unraveled rules are

$$\begin{aligned} \ell &\to C_1[s_1] \\ C_1[t_1] &\to C_2[s_2] \\ \dots &\to \dots \\ C_n[t_n] &\to r \end{aligned}$$

    C_i are arbitrary, standard unraveling by choosing $C_1 = U_1(\cdot, \mathcal{V}(\ell)), \dots$
  - completeness result:

$$SN(\mathcal{U}_{C_i}(\mathcal{R})) \Longrightarrow \textit{quasi-decreasing}(\mathcal{R}) \Longrightarrow SN(\to_{\mathcal{R}})$$

- several open tasks
  - certification algorithm for unraveling and suitable format
  - algorithm to compute $\to_{\mathcal{R}}$ (for quasi-decreasing $\mathcal{R}$)
  - soundness results for unravelings

Goal 3: Constraint rewriting

- wait for outcome of Task 1
- then there will be several open tasks

Task 5: Certification, Summary

- 1. goal: certifying loops
  - large parts already done (innermost, outermost)
  - generalization to forbidden patters currently open
- 2. goal: unraveling
  - only completeness results in IsaFoR at the moment
- 3. goal: constrained rewriting
  - not clear what to do at the moment

- open questions
  - which tools provide proofs to certify (TTT2?, VMTL?, Nagoya?)
  - which soundness result for unraveling (Nagoya?, Vienna?, combined?)