Automatically Finding Non-CR Examples in Term Rewriting

Hans Zantema

Technische Universiteit Eindhoven and Radboud Universiteit Nijmegen

IWC, TU Eindhoven, June 28, 2013
Our goal:
Our goal:

Given a set of abstract rewrite properties, automatically find a TRS satisfying these properties
Our goal:

Given a set of abstract rewrite properties, automatically find a TRS satisfying these properties (preferrably by exploiting the power of SAT / SMT / constraint solving)
Our goal:

Given a set of abstract rewrite properties, automatically find a TRS satisfying these properties (preferably by exploiting the power of SAT / SMT / constraint solving)

Last year at IWC in Nagoya we did this for finite ARSs, yielding some remarkable examples for which finding them by hand would be a hard job
Our goal:

Given a set of abstract rewrite properties, automatically find a TRS satisfying these properties (preferrably by exploiting the power of SAT / SMT / constraint solving)

Last year at IWC in Nagoya we did this for finite ARSs, yielding some remarkable examples for which finding them by hand would be a hard job

However, for many sets of properties no finite ARS exists, but a TRS exists
Our goal:

Given a set of abstract rewrite properties, automatically find a TRS satisfying these properties (preferably by exploiting the power of SAT / SMT / constraint solving)

Last year at IWC in Nagoya we did this for finite ARSs, yielding some remarkable examples for which finding them by hand would be a hard job

However, for many sets of properties no finite ARS exists, but a TRS exists

For instance, the single rule $f(a) \to a$ is terminating and its reverse $a \to f(a)$ is non-terminating, while no finite ARS has this combination of properties
Leading example

Find a TRS that is locally confluent (WCR), but not confluent (CR), and for which the reverse is terminating (SN)
Leading example

Find a TRS that is locally confluent (WCR), but not confluent (CR), and for which the reverse is terminating (SN)

Solution:

\[ a \rightarrow b \]
\[ a \rightarrow f(a) \]
\[ b \rightarrow f(f(b)) \]
Find a TRS that is locally confluent (WCR), but not confluent (CR), and for which the reverse is terminating (SN)

Solution:

\[
\begin{align*}
    a & \rightarrow b \\
    a & \rightarrow f(a) \\
    b & \rightarrow f(f(b))
\end{align*}
\]
Our goal: find such an example fully automatically
Our goal: find such an example fully automatically

In which space of TRSs are we looking?
Our goal: find such an example fully automatically

In which space of TRSs are we looking?

- Set of finite ARSs (= TRSs in which every lhs and rhs is a constant)?
Our goal: find such an example fully automatically

In which space of TRSs are we looking?

- Set of finite ARSs (= TRSs in which every lhs and rhs is a constant)?

   Too restrictive: if reverse of $R$ is terminating then $R$ is terminating too, and no such example exists by Newman’s Lemma
Our goal: find such an example fully automatically

In which space of TRSs are we looking?

- Set of finite ARSs (= TRSs in which every lhs and rhs is a constant)?
  
  Too restrictive: if reverse of $R$ is terminating then $R$ is terminating too, and no such example exists by Newman’s Lemma

- Set of all TRSs?
Our goal: find such an example fully automatically

In which space of TRSs are we looking?

- Set of finite ARSs (= TRSs in which every lhs and rhs is a constant)?
  
  Too restrictive: if reverse of $R$ is terminating then $R$ is terminating too, and no such example exists by Newman’s Lemma

- Set of all TRSs?
  
  Far too large
Our goal: find such an example fully automatically

In which space of TRSs are we looking?

- Set of finite ARSs (= TRSs in which every lhs and rhs is a constant)?
  
  Too restrictive: if reverse of $R$ is terminating then $R$ is terminating too, and no such example exists by Newman’s Lemma

- Set of all TRSs?

  Far too large

We choose the smallest class of TRSs strictly including finite ARSs and allowing terminating TRSs for which the reverse is not terminating:
Our goal: find such an example fully automatically

In which space of TRSs are we looking?

- Set of finite ARSs (= TRSs in which every lhs and rhs is a constant)?

  Too restrictive: if reverse of $R$ is terminating then $R$ is terminating too, and no such example exists by Newman’s Lemma

- Set of all TRSs?

  Far too large

We choose the smallest class of TRSs strictly including finite ARSs and allowing terminating TRSs for which the reverse is not terminating:

- Ground TRSs over a single unary symbol $f$ and a finite set of constants
Ground TRSs over a single unary symbol $f$ and a finite set of constants
Ground TRSs over a single unary symbol $f$ and a finite set of constants

So, we only allow rules of the shape

$$f^n(a) \rightarrow f^k(b)$$
Ground TRSs over a single unary symbol $f$ and a finite set of constants

So, we only allow rules of the shape

$$f^n(a) \rightarrow f^k(b)$$

By adding extra constants all such rules can be mimicked by a combination of rules of the shape

$$a \rightarrow b, \ f(a) \rightarrow b, \ a \rightarrow f(b)$$
Ground TRSs over a single unary symbol $f$ and a finite set of constants

So, we only allow rules of the shape

$$f^n(a) \rightarrow f^k(b)$$

By adding extra constants all such rules can be mimicked by a combination of rules of the shape

$$a \rightarrow b, \ f(a) \rightarrow b, \ a \rightarrow f(b)$$

Fixing a number $n$ of constants, let $T_1$ be the TRS of all $3n^2$ such rules
Ground TRSs over a single unary symbol $f$ and a finite set of constants

So, we only allow rules of the shape

$$f^n(a) \rightarrow f^k(b)$$

By adding extra constants all such rules can be mimicked by a combination of rules of the shape

$$a \rightarrow b, \ f(a) \rightarrow b, \ a \rightarrow f(b)$$

Fixing a number $n$ of constants, let $T_1$ be the TRS of all $3n^2$ such rules

Our search space will consist of the subTRSs of $T_1$
In order to express composition we also consider $T_2$ being the TRS of all $6n^2$ rules of the shape

$$a \rightarrow b, f(a) \rightarrow b, a \rightarrow f(b), a \rightarrow f(f(b)), f(f(a)) \rightarrow b, f(a) \rightarrow f(b)$$
In order to express composition we also consider $T_2$ being the TRS of all $6n^2$ rules of the shape

$$a \rightarrow b, f(a) \rightarrow b, a \rightarrow f(b), a \rightarrow f(f(b)), f(f(a)) \rightarrow b, f(a) \rightarrow f(b)$$

For instance, if $a \rightarrow f(b) \in R$ and $b \rightarrow f(c) \in S$, then $a \rightarrow_{R \cdot S} f(f(c))$
In order to express composition we also consider $T_2$ being the TRS of all $6n^2$ rules of the shape

$$a \rightarrow b, f(a) \rightarrow b, a \rightarrow f(b), a \rightarrow f(f(b)), f(f(a)) \rightarrow b, f(a) \rightarrow f(b)$$

For instance, if $a \rightarrow f(b) \in R$ and $b \rightarrow f(c) \in S$, then $a \rightarrow_{R \cdot S} f(f(c))$

Let $\text{comp}(R, S)$ be the subTRS of $T_2$ mimicking all such compositions, so in this case $a \rightarrow f(f(c)) \in \text{comp}(R, S)$
In order to express composition we also consider $T_2$ being the TRS of all $6n^2$ rules of the shape

$$a \rightarrow b, f(a) \rightarrow b, a \rightarrow f(b), a \rightarrow f(f(b)), f(f(a)) \rightarrow b, f(a) \rightarrow f(b)$$

For instance, if $a \rightarrow f(b) \in R$ and $b \rightarrow f(c) \in S$, then $a \rightarrow_{R \cdot S} f(f(c))$

Let $\text{comp}(R, S)$ be the subTRS of $T_2$ mimicking all such compositions, so in this case $a \rightarrow f(f(c)) \in \text{comp}(R, S)$

**Theorem**

1. If $R, S \subseteq T_1$ then $\rightarrow \text{comp}(R, S) = \rightarrow R \cdot \rightarrow S$.
2. If $R, S \subseteq T_2$ then $\rightarrow \text{comp}(R, S) \subseteq \rightarrow R \cdot \rightarrow S$. 
Local confluence

Hans Zantema

Automatically Finding Non-CR Examples in Term Rewriting
Define inverse and reflexive closure:

\[
\text{inv}(R) = \{ r \rightarrow \ell \mid \ell \rightarrow r \in R \}
\]

\[
\text{rc}(R) = R \cup \{ a \rightarrow a \mid a \in A \}
\]
Local confluence

Define inverse and reflexive closure:

$$\text{inv}(R) = \{ r \to \ell \mid \ell \to r \in R \}$$

$$\text{rc}(R) = R \cup \{ a \to a \mid a \in A \}$$

**Theorem**

Let $R \subseteq T_1$, $R_1 = \text{rc}(R)$, $R_{i+1} = \text{comp}(R_i, R_i)$ for $i > 0$, and

$$\text{comp}(\text{inv}(R), R) \subseteq \text{comp}(R_i, \text{inv}(R_i))$$

for some $i > 0

Then $\text{WCR}(R)$
Local confluence

Define inverse and reflexive closure:

\[
\text{inv}(R) = \{ r \to \ell \mid \ell \to r \in R \}
\]

\[
\text{rc}(R) = R \cup \{ a \to a \mid a \in A \}
\]

**Theorem**

Let \( R \subseteq T_1 \), \( R_1 = \text{rc}(R) \), \( R_{i+1} = \text{comp}(R_i, R_i) \) for \( i > 0 \), and

\[
\text{comp}(\text{inv}(R), R) \subseteq \text{comp}(R_i, \text{inv}(R_i))
\]

for some \( i > 0 \)

Then \( \text{WCR}(R) \)

If \( \text{WCR} \) is required, we express this by the slightly stronger requirement \( \text{comp}(\text{inv}(R), R) \subseteq \text{comp}(R_i, \text{inv}(R_i)) \) for \( i = 2 \) or \( i = 3 \)
Projection to finite ARS

Rough idea: identify $f(f(x))$ with $x$
Projection to finite ARS

Rough idea: identify \( f(f(x)) \) with \( x \)

More precisely: for every constant \( a \) define new constants \( a_0, a_1 \)

\[
\pi(a \rightarrow b) = \{ a_0 \rightarrow b_0, a_1 \rightarrow b_1 \}
\]

\[
\pi(a \rightarrow f(b)) = \{ a_0 \rightarrow b_1, a_1 \rightarrow b_0 \}
\]

\[
\pi(f(a) \rightarrow b) = \{ a_0 \rightarrow b_1, a_1 \rightarrow b_0 \}
\]
Projection to finite ARS

Rough idea: identify $f(f(x))$ with $x$

More precisely: for every constant $a$ define new constants $a_0, a_1$

$$
\pi(a \to b) = \{a_0 \to b_0, a_1 \to b_1\}
$$

$$
\pi(a \to f(b)) = \{a_0 \to b_1, a_1 \to b_0\}
$$

$$
\pi(f(a) \to b) = \{a_0 \to b_1, a_1 \to b_0\}
$$

**Theorem**

*If $CR(R)$, then $CR(\pi(R))$*
Projection to finite ARS

Rough idea: identify \( f(f(x)) \) with \( x \)

More precisely: for every constant \( a \) define new constants \( a_0, a_1 \)

\[ \pi(a \rightarrow b) = \{a_0 \rightarrow b_0, a_1 \rightarrow b_1\} \]

\[ \pi(a \rightarrow f(b)) = \{a_0 \rightarrow b_1, a_1 \rightarrow b_0\} \]

\[ \pi(f(a) \rightarrow b) = \{a_0 \rightarrow b_1, a_1 \rightarrow b_0\} \]

**Theorem**

*If \( CR(R) \), then \( CR(\pi(R)) \)*

If \( \neg CR(R) \) is required, we express this by the stronger requirement \( \neg CR(\pi(R)) \), which is about finite ARS, so can be expressed by earlier techniques
Termination

Where for WCR and $\neg$CR we only found approximations, termination can be expressed exactly.
Where for WCR and ¬CR we only found approximations, termination can be expressed exactly

**Theorem**

A ground TRS $R$ over $\{f\} \cup A$ is terminating if and only if a map $W : A \to R$ exists such that $W(a) + n > W(b) + k$ for every $f^n(a) \to f^k(b) \in R$.
Where for WCR and ¬CR we only found approximations, termination can be expressed exactly

**Theorem**

A ground TRS $R$ over $\{f\} \cup A$ is terminating if and only if a map $W : A \rightarrow R$ exists such that $W(a) + n > W(b) + k$ for every $f^n(a) \rightarrow f^k(b) \in R$

This criterium for SN is easily expressed in SMT, satisfiability modulo theory of linear inequalities, where until now everything was in propositional SAT
Implementation

Hans Zantema

Automatically Finding Non-CR Examples in Term Rewriting
For finite ARSs we already developed a tool CARPA: Counter examples of Abstract Rewriting Produced Automatically
Implementation

For finite ARSs we already developed a tool CARPA: Counter examples of Abstract Rewriting Produced Automatically

We modified the CARPA input language to deal with the new TRS features
For finite ARSs we already developed a tool CARPA: Counter examples of Abstract Rewriting Produced Automatically

We modified the CARPA input language to deal with the new TRS features

Example

\[
x_1 = \text{inv}(1) \quad \text{x1 is inverse of basic TRS 1}
\]

\[
\text{sn}(x_1) \quad \text{x1 is terminating}
\]
For finite ARSs we already developed a tool CARPA: Counter examples of Abstract Rewriting Produced Automatically

We modified the CARPA input language to deal with the new TRS features

**Example**

\[
\begin{align*}
\text{x1} &= \text{inv}(1) & \text{x1 is inverse of basic TRS 1} \\
\text{sn(x1)} &= & \text{x1 is terminating} \\
\text{x2} &= \text{peak}(1,1) \\
\text{x3} &= \text{rc}(1) \\
\text{x3} &= \text{comp(x3,x3)} \\
\text{x3} &= \text{val(x3,x3)} & \text{describes WCR(1) as discussed:} \\
\text{subs(x2,x3)} &= & \text{peak contained in valley}
\end{align*}
\]
For finite ARSs we already developed a tool CARPA: Counter examples of Abstract Rewriting Produced Automatically

We modified the CARPA input language to deal with the new TRS features

**Example**

\[ x_1 = \text{inv}(1) \quad x_1 \text{ is inverse of basic TRS 1} \]
\[ s_1(x_1) \quad x_1 \text{ is terminating} \]
\[ x_2 = \text{peak}(1,1) \]
\[ x_3 = \text{rc}(1) \]
\[ x_3 = \text{comp}(x_3,x_3) \]
\[ x_3 = \text{val}(x_3,x_3) \quad \text{describes WCR}(1) \text{ as discussed:} \]
\[ \text{subs}(x_2,x_3) \quad \text{peak contained in valley} \]
\[ x_1 = \text{mod2}(1) \quad x_1 \text{ is projection to finite ARS of 1} \]
\[ ncr(x_1) \quad \text{non-confluent by earlier techniques} \]
Applying our tool to this input, and specify \( \#A = 3 \)
Applying our tool to this input, and specify $\#A = 3$

- generates a formula expressing all these ingredients
Applying our tool to this input, and specify $\#A = 3$

- generates a formula expressing all these ingredients
- calls the SMT solver YICES
Applying our tool to this input, and specify $\#A = 3$

- generates a formula expressing all these ingredients
- calls the SMT solver YICES
- inspects the solution found by YICES, and transforms this to the following output

1 $\rightarrow$ 2
1 $\rightarrow$ f(1)
2 $\rightarrow$ f(3)
3 $\rightarrow$ f(2)
Applying our tool to this input, and specify \( \#A = 3 \)

- generates a formula expressing all these ingredients
- calls the SMT solver YICES
- inspects the solution found by YICES, and transforms this to the following output

\[
\begin{align*}
1 & \rightarrow 2 \\
1 & \rightarrow f(1) \\
2 & \rightarrow f(3) \\
3 & \rightarrow f(2)
\end{align*}
\]

which is indeed a TRS satisfying the given requirements
Conclusions
Conclusions

We developed a method for automatically finding TRSs having a given set of properties, in particular $\text{WCR}(\to)$, $\neg\text{CR}(\to)$, $\text{SN}(\leftarrow)$
We developed a method for automatically finding TRSs having a given set of properties, in particular $WCR(\rightarrow)$, $\neg CR(\rightarrow)$, $SN(\leftarrow)$

The real work is done by an SMT solver
Conclusions

- We developed a method for automatically finding TRSs having a given set of properties, in particular $\text{WCR}(\rightarrow), \neg \text{CR}(\rightarrow), \text{SN}(\leftarrow)$

- The real work is done by an SMT solver

- We restricted to ground TRSs over constants and one single unary symbol: both restrictive and a substantial extension compared to finite ARSs
Conclusions

- We developed a method for automatically finding TRSs having a given set of properties, in particular $\text{WCR}(\rightarrow)$, $\neg \text{CR}(\rightarrow)$, $\text{SN}(\leftarrow)$

- The real work is done by an SMT solver

- We restricted to ground TRSs over constants and one single unary symbol: both restrictive and a substantial extension compared to finite ARSs

- If $\rightarrow^*$ comes in: arbitrary number of steps, then we approximate, or project to a finite ARS, loosing completeness
Conclusions

- We developed a method for automatically finding TRSs having a given set of properties, in particular WCR(→), ¬CR(→), SN(←)

- The real work is done by an SMT solver

- We restricted to ground TRSs over constants and one single unary symbol: both restrictive and a substantial extension compared to finite ARSs

- If →* comes in: arbitrary number of steps, then we approximate, or project to a finite ARS, losing completeness

- For this restricted class WCR and CR are decidable; we believe encoding corresponding algorithms in SAT/SMT will not better serve our goal
Conclusions

- We developed a method for automatically finding TRSs having a given set of properties, in particular WCR($\rightarrow$), $\neg$CR($\rightarrow$), SN($\leftarrow$)

- The real work is done by an SMT solver

- We restricted to ground TRSs over constants and one single unary symbol: both restrictive and a substantial extension compared to finite ARSs

- If $\rightarrow^*$ comes in: arbitrary number of steps, then we approximate, or project to a finite ARS, losing completeness

- For this restricted class WCR and CR are decidable; we believe encoding corresponding algorithms in SAT/SMT will not better serve our goal

- Termination is expressed exactly