Term Rewriting Systems

Franz Baader
Theoretical Computer Science
TU Dresden
Germany

1. Motivation and basic definitions and results.
2. Equational Problems: the word problem and term rewriting
3. Termination of term rewriting systems
4. Confluence of term rewriting systems
5. Completion of term rewriting systems

Dresden

Goal

given a finite set of identities E, try to construct a decision procedure for the word problem for E

Theorem

If E is a set of identities and R is a finite, convergent TRS such that $\overset{*}{\leftrightarrow}_E = \overset{*}{\leftrightarrow}_R$, then the word problem for E decidable.

First approach

Show termination: Try to find a reduction order $>$ that can orient the identities of E into a terminating set of rules.

This succeeds if, for all $(s \approx t) \in E$, we have $s > t$ or $t > s$.

Show confluence: Decide confluence of the TRS R obtained this way by computing all critical pairs and testing whether they are joinable.

Dresden

Example

where this simple approach succeeds

We consider

$$E := \{x + 0 \approx x,\ x + s(y) \approx s(x + y)\}$$

Show termination: If we use the lexicographic path order $>_{lpo}$ induced by

$$+ > s,$$

then we have $x + 0 > x$ and $x + s(y) > s(x + y)$.

The rewrite system

$$R := \{x + 0 \to x,\ x + s(y) \to s(x + y)\}$$

is thus terminating.

Show confluence: It is also confluent since there are no non-trivial critical pairs.

Dresden

Example

where this simple approach does not succeed

We consider again $E = \{x + 0 \approx x,\ x + s(y) \approx s(x + y)\}$

Show termination: But now we use the lexicographic path order $>_{lpo}$ induced by $s > +$.

Then we have $x + 0 > x$ and $s(x + y) > x + s(y)$.

The rewrite system

$$R := \{x + 0 \to x,\ s(x + y) \to x + s(y)\}$$

is thus also terminating.

Show confluence: It is however not confluent since the following critical pair is not joinable:

$$x + s(0) \leftarrow s(x + 0) \rightarrow s(x)$$

Dresden

Main ideas underlying completion:

- If the critical pair $\langle s,t\rangle$ of R is not joinable, then there are distinct normal forms $\widehat{s},\widehat{t}$ of s,t.
- The identity $\widehat{s}\approx\widehat{t}$ is obviously an equational consequence of R since $\widehat{s}\overset{*}{\leftrightarrow}_R\widehat{t}$.
- Thus, adding one of the rules

  $$\widehat{s}\rightarrow\widehat{t} \text{ or } \widehat{t}\rightarrow\widehat{s}$$

  to R does not change the generated equational theory.
- In the extended system, $\langle s,t\rangle$ is now joinable.
- To obtain a terminating new system, we need

  $$\widehat{s}>\widehat{t} \text{ or } \widehat{t}>\widehat{s}$$

Dresden

The basic completion procedure

Input:
A finite set E of Σ-identities and a reduction order $>$ on $T(\Sigma,V)$.

Output:
A finite convergent TRS R that is equivalent to E, if the procedure terminates successfully;

"`Fail`", if the procedure terminates unsuccessfully.

Initialization:
If there exists $(s\approx t)\in E$ such that $s\neq t$, $s\not> t$ and $t\not> s$, then terminate with output `Fail`.

Otherwise, $i:=0$ and

$$R_0:=\{l\rightarrow r\mid (l\approx r)\in E\cup E^{-1}\wedge l>r\}.$$

Dresden

The basic completion procedure

`repeat` $R_{i+1}:=R_i$;

`for all` $\langle s,t\rangle\in CP(R_i)$ `do`

(a) Reduce s,t to some R_i-normal forms $\widehat{s},\widehat{t}$;

(b) If $\widehat{s}\neq\widehat{t}$ and neither $\widehat{s}>\widehat{t}$ nor $\widehat{t}>\widehat{s}$, then terminate with output `Fail`;

(c) If $\widehat{s}>\widehat{t}$, then $R_{i+1}:=R_{i+1}\cup\{\widehat{s}\rightarrow\widehat{t}\}$;

(d) If $\widehat{t}>\widehat{s}$, then $R_{i+1}:=R_{i+1}\cup\{\widehat{t}\rightarrow\widehat{s}\}$;

`od`

$i:=i+1$;

`until` $R_i=R_{i-1}$;

`output` R_i;

Dresden

The basic completion procedure may show three different types of behaviour, depending on the particular input E and $>$:

1. It may terminate with failure because one of the nontrivial input identities cannot be ordered using $>$, or the normal forms of the terms in one of the critical pairs are distinct and cannot be ordered using $>$.

   In this case, not much is gained. One could, however, try to run the procedure again, using another reduction order.

2. It may terminate successfully with output R_n because in the nth step of the iteration all critical pairs are joinable.

   In this case, the output R_n is a finite convergent TRS that is equivalent to E.

   This system can be used to decide the word problem for E.

3. It may run for ever since infinitely many new rules are generated.

   In this case, $R_\infty:=\bigcup_{i\geq 0}R_i$ is an infinite convergent TRS that is equivalent to E.

   Since R_∞ is infinite, it only yields a semidecision procedure for $\approx_E$.

Dresden

Example — the procedure terminates successfully

Input:

$E := \{f(f(x)) \approx g(x)\}$

LPO $>_{lpo}$ induced by $f > g$

$R_0 := \{f(f(x)) \to g(x)\}$ has the following non-joinable critical pair:

$$g(f(x)) \leftarrow f(f(f(x))) \to f(g(x))$$

$R_1 := \{f(f(x)) \to g(x),\ f(g(x)) \to g(f(x))\}$ is confluent, since all its critical pairs are now joinable, i.e. $R_2 = R_1$.

Output:

$R_2 = \{f(f(x)) \to g(x),\ f(g(x)) \to g(f(x))\}$

Example — the procedure terminates with failure

Input:

$E := \{x * (y + z) \approx (x * y) + (x * z),\ (u + v) * w \approx (u * w) + (v * w)\}$

LPO $>_{lpo}$ induced by $* > +$

$R_0 := \{x * (y + z) \to (x * y) + (x * z),\ (u + v) * w \to (u * w) + (v * w)\}$ has the following non-joinable critical pair:

$$(u + v) * (y + z)$$

$$((u + v) * y) + ((u + v) * z) \qquad (u * (y + z)) + (v * (y + z))$$

$$((u * y) + (v * y)) + ((u * z) + (v * z)) \quad \not> \ \neq \ \not< \quad ((u * y) + (u * z)) + ((v * y) + (v * z))$$

Example — the procedure does not terminate

Input:

$E = \{x + 0 \approx x,\ x + s(y) \approx s(x + y)\}$

LPO $>_{lpo}$ induced by $s > +$

$$R_0 = \{x + 0 \to x,\ s(x + y) \to x + s(y)\}$$
$$R_1 = R_0 \cup \{x + s(0) \to s(x)\}$$

R_1 is not confluent since the following critical pair is not joinable:

$$x + s(s(0)) \leftarrow s(x + s(0)) \to s(s(x))$$

In each step of the iteration a new rule of the form

$$x + s^n(0) \to s^n(x)$$

is generated.

The basic completion procedure may show different types of behaviour, depending on the particular input E and $>$:

1. It may terminate with failure because an identity cannot be ordered using the reduction order $>$.
2. It may terminate successfully with output R_n because in the nth step of the iteration all critical pairs are joinable.
3. It may run for ever since infinitely many new rules are generated.
4. The procedure crashes because it needs too much space.
5. The procedure terminates, but it takes 10 000 years.

An improved completion procedure

extends the basic completion by simplification of rules by other rules:

- yields smaller rules
- if the left- and right-hand sides reduce to the same term, the rule can be removed

Example

$R = \{f(f(x,y),z) \to f(x,f(y,z)),\ \ f(x,f(y,z)) \to f(x,z)\}$

$$f(f(x,y),z) \longrightarrow f(x,f(y,z)) \downarrow f(x,z)$$

Simpler set of rules:

$R = \{f(f(x,y),z) \to f(x,z),\ \ f(x,f(y,z)) \to f(x,z)\}$

Dresden

An improved completion procedure

- described by a set of inference rules that covers a wide range of different specific completion procedures
- specific completion procedure is obtained from this set of rules by fixing a strategy for application of the inference rules
- works on pairs (E,R) where E is a finite set of identities and R is a finite set of rewrite rules
- E contains input identities or critical pairs not yet oriented with the input reduction order $>$
- R set of rewrite rules oriented with $>$
- Goal: to transform an initial pair $(E_0,\emptyset)$ into a pair $(\emptyset,R)$ such that R is convergent and equivalent to E_0.

Dresden

The inference rules for completion

DEDUCE $\quad \dfrac{E,R}{E\cup\{s\approx t\},R} \quad$ if $s \leftarrow_R u \rightarrow_R t$

ORIENT $\quad \dfrac{E\cup\{s \dot{\approx} t\},R}{E,R\cup\{s\to t\}} \quad$ if $s>t$

DELETE $\quad \dfrac{E\cup\{s\approx s\},R}{E,R}$

SIMPLIFY-IDENTITY $\quad \dfrac{E\cup\{s \dot{\approx} t\},R}{E\cup\{u\approx t\},R} \quad$ if $s \rightarrow_R u$

R-SIMPLIFY-RULE $\quad \dfrac{E,R\cup\{s\to t\}}{E,R\cup\{s\to u\}} \quad$ if $t \rightarrow_R u$

L-SIMPLIFY-RULE $\quad \dfrac{E,R\cup\{s\to t\}}{E\cup\{u\approx t\},R} \quad$ if $s \overset{\sqsupset}{\rightarrow}_R u$

Dresden

L-SIMPLIFY-RULE $\quad \dfrac{E,R\cup\{s\to t\}}{E\cup\{u\approx t\},R} \quad$ if $s \overset{\sqsupset}{\rightarrow}_R u$

The restriction $s \overset{\sqsupset}{\rightarrow}_R u$ says that s is reduced by a rule $l\to r\in R$ such that l cannot be reduced by $s\to t$.

Thus, if

$$R := \{f(x,x)\to x,\ f(x,y)\to x\},$$

then L-SIMPLIFY-RULE can be applied to $f(x,x)\to x$.

If

$$R := \{f(x,y)\to x,\ f(x,y)\to y\}$$

then L-SIMPLIFY-RULE cannot be applied.

This restriction is needed to ensure that the final set of rules R is indeed equivalent to the input E_0.

Dresden

We write

$$(E, R) \vdash_{\mathcal{C}} (E', R')$$

to indicate that (E, R) can be transformed to (E', R') by applying one of the inference rules.

Lemma (termination of R)

If $R \subseteq >$ and $(E, R) \vdash_{\mathcal{C}} (E', R')$, then $R' \subseteq >$.

Thus, the rewrite system R in the pair (E, R) is terminating if this pair has been obtained from an initial pair of the form $(E_0, \emptyset)$ by application of the inference rules.

Lemma (soundness of inference rules)

$(E_1, R_1) \vdash_{\mathcal{C}} (E_2, R_2)$ implies $\approx_{E_1 \cup R_1} = \approx_{E_2 \cup R_2}$.

Definition (completion procedure)

A completion procedure is a program that accepts as input a finite set of identities E_0 and a reduction order $>$, and uses the inference rules to generate a (finite or infinite) sequence

$$(E_0, \emptyset) \vdash_{\mathcal{C}} (E_1, R_1) \vdash_{\mathcal{C}} (E_2, R_2) \vdash_{\mathcal{C}} (E_3, R_3) \vdash_{\mathcal{C}} \cdots$$

This sequence is called a run of the completion procedure on input E_0 and $>$.

We extend every finite run $(E_0, R_0) \vdash_{\mathcal{C}} \cdots \vdash_{\mathcal{C}} (E_n, R_n)$ to an infinite one by setting $(E_{n+i}, R_{n+i}) := (E_n, R_n)$ for all $i \geq 1$.

Result of a run: persistent identities and rules

$$E_\omega := \bigcup_{i \geq 0} \bigcap_{j \geq i} E_j \quad \text{and} \quad R_\omega := \bigcup_{i \geq 0} \bigcap_{j \geq i} R_j$$

Note: if the run is finite, then $E_\omega = E_n$ and $R_\omega = R_n$.

Definition (success, failure, correctness)

A run on input E_0 of a completion procedure

succeeds iff $E_\omega = \emptyset$ and R_ω is convergent and equivalent to E_0.

fails iff $E_\omega \neq \emptyset$.

A completion procedure is correct iff every run that does not fail succeeds.

Basic completion procedure:

- failure occurs if an input identity cannot be oriented or the normal forms of a critical pair are distinct (i.e. cannot be removed using DELETE) and cannot be oriented using $>$ (i.e. cannot be transformed from an identity into a rule).
- the other two cases (terminates successfully, does not terminate) are successful in the sense of the above definition.

An arbitrary completion procedure may also have infinite failing runs.

Without an additional fairness assumption, an arbitrary completion procedure need not be correct.

Obviously, a necessary condition for correctness is that all the relevant critical pairs are computed.

Definition (fairness)

A run of a completion procedure is called fair iff

$CP(R_\omega) \subseteq \bigcup_{i \geq 0} E_i$.

A completion procedure is fair iff every non-failing run is fair.

This condition is also sufficient for correctness:

Theorem (correctness)

Every fair completion procedure is correct.

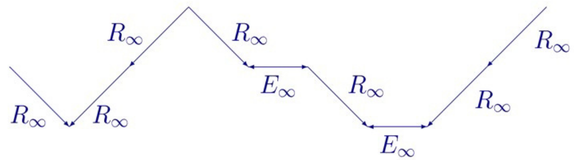
Main idea

underlying the correctness proof:
turning mountains into valleys

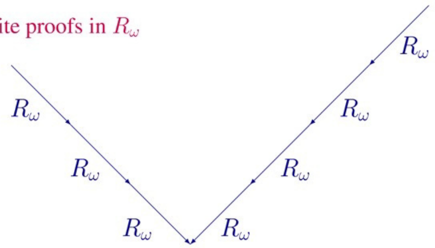
Show that one can transform arbitrary proofs in $E_\infty \cup R_\infty$

$R_\infty := \bigcup_{i\geq 1} R_i$

$E_\infty := \bigcup_{i\geq 0} E_i$



into rewrite proofs in R_ω



using well-founded induction w.r.t. a well-founded order (proof order) on proofs.

Example

Input:

$E_0 := \{f(x) \approx g(x),\ f(f(f(x))) \approx h(x),\ f(f(x)) \approx x\}$

$>_{lpo}$ induced by $f > h > g$

Apply ORIENT:

$E_1 = \{f(f(f(x))) \approx h(x),\ f(f(x)) \approx x\}$

$R_1 = \{f(x) \to g(x)\}$

Apply SIMPLIFY-IDENTITY (3 times):

$E_4 = \{g(g(g(x))) \approx h(x),\ f(f(x)) \approx x\}$

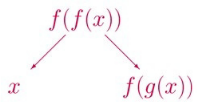
$R_4 = \{f(x) \to g(x)\}$

Apply ORIENT:

$E_5 = \{g(g(g(x))) \approx h(x)\}$

$R_5 = \{f(x) \to g(x),\ f(f(x)) \to x\}$

Apply DEDUCE:



$E_6 = \{g(g(g(x))) \approx h(x),\ x \approx f(g(x))\}$

$R_6 = \{f(x) \to g(x),\ f(f(x)) \to x\}$

Apply L-SIMPLIFY-RULE:

$E_7 = \{g(g(g(x))) \approx h(x),\ x \approx f(g(x)),\ g(f(x)) \approx x\}$

$R_7 = \{f(x) \to g(x)\}$

Apply SIMPLIFY-IDENTITY (2 times):

$E_9 = \{g(g(g(x))) \approx h(x),\ g(g(x)) \approx x\}$

$R_9 = \{f(x) \to g(x)\}$

Apply ORIENT:

$E_{10} = \{g(g(g(x))) \approx h(x)\}$

$R_{10} = \{f(x) \to g(x),\ g(g(x)) \to x\}$

Apply ORIENT:

$E_{11} = \emptyset$

$R_{11} = \{f(x) \to g(x),\ g(g(x)) \to x,\ h(x) \to g(g(g(x)))\}$

Apply R-SIMPLIFY-RULE:

$E_{12} = \emptyset$

$R_{12} = \{f(x) \to g(x),\ g(g(x)) \to x,\ h(x) \to g(x)\}$

Since there are no non-trivial critical pairs between these rules, this is a fair and non-failing run.

Consequently, R_{12} is convergent and equivalent to E_0.