The mapping from function symbols to polynomials can be extended to terms: A term t containing the variables $x_1, \dots, x_n$ yields a polynomial P_t with indeterminates $X_1, \dots, X_n$ (where X_i corresponds to $\beta(x_i)$).

Example:

$\Omega = \{b/0,\ f/1,\ g/3\}$
$P_b = 3, \quad P_f(X_1) = X_1^2, \quad P_g(X_1, X_2, X_3) = X_1 + X_2X_3.$

Let $t = g(f(b), f(x), y)$, then $P_t(X, Y) = 9 + X^2Y$.

If P, Q are polynomials in $\mathbb{N}[X_1, \dots, X_n]$, we write $P > Q$ if $P(a_1, \dots, a_n) > Q(a_1, \dots, a_n)$ for all $a_1, \dots, a_n \in U_\mathcal{A}$.

Clearly, $s \succ_\mathcal{A} t$ iff $P_s > P_t$ iff $P_s - P_t > 0$.

Question: Can we check $P_s - P_t > 0$ automatically?

Hilbert's 10th Problem:

Given a polynomial $P \in \mathbb{Z}[X_1, \dots, X_n]$ with integer coefficients, is $P = 0$ for some n-tuple of natural numbers?

Theorem 4.23 *Hilbert's 10th Problem is undecidable.*

Proposition 4.24 *Given a polynomial interpretation and two terms s, t, it is undecidable whether $P_s > P_t$.*

Proof. By reduction of Hilbert's 10th Problem. $\square$

One easy case:

If we restrict to linear polynomials, deciding whether $P_s - P_t > 0$ is trivial:

$\sum k_i a_i + k > 0$ for all $a_1, \dots, a_n \geq 1$ if and only if

$k_i \geq 0$ for all $i \in \{1, \dots, n\}$,

and $\sum k_i + k > 0$

Another possible solution:

Test whether $P_s(a_1, \dots, a_n) > P_t(a_1, \dots, a_n)$ for all $a_1, \dots, a_n \in \{\, x \in \mathbb{R} \mid x \geq 1 \,\}$.

This is decidable (but hard). Since $U_\mathcal{A} \subseteq \{\, x \in \mathbb{R} \mid x \geq 1 \,\}$, it implies $P_s > P_t$.

Alternatively:

Use fast overapproximations.

Simplification Orderings

The *proper subterm ordering* $\rhd$ is defined by $s \rhd t$ if and only if $s|_p = t$ for some position $p \neq \varepsilon$ of s.

A rewrite ordering $\succ$ over $\mathrm{T}_\Sigma(X)$ is called *simplification ordering,* if it has the *subterm property:* $s \rhd t$ implies $s \succ t$ for all $s, t \in \mathrm{T}_\Sigma(X)$.

Example:

Let R_{emb} be the rewrite system $R_{\text{emb}} = \{\, f(x_1, \dots, x_n) \to x_i \mid f/n \in \Omega,\ 1 \leq i \leq n \,\}$.

Define $\rhd_{\text{emb}} = \to^{+}_{R_{\text{emb}}}$ and $\unrhd_{\text{emb}} = \to^{*}_{R_{\text{emb}}}$ (*"homeomorphic embedding relation"*).

$\rhd_{\text{emb}}$ is a simplification ordering.

Lemma 4.25 *If $\succ$ is a simplification ordering, then $s \rhd_{\text{emb}} t$ implies $s \succ t$ and $s \unrhd_{\text{emb}} t$ implies $s \succeq t$.*

Proof. Since $\succ$ is transitive and $\succeq$ is transitive and reflexive, it suffices to show that $s \to_{R_{\text{emb}}} t$ implies $s \succ t$. By definition, $s \to_{R_{\text{emb}}} t$ if and only if $s = s[l\sigma]$ and $t = s[r\sigma]$ for some rule $l \to r \in R_{\text{emb}}$. Obviously, $l \rhd r$ for all rules in R_{emb}, hence $l \succ r$. Since $\succ$ is a rewrite relation, $s = s[l\sigma] \succ s[r\sigma] = t$. □

Goal:

Show that every simplification ordering is well-founded (and therefore a reduction ordering).

Note: This works only for *finite* signatures!

To fix this for infinite signatures, the definition of simplification orderings and the definition of embedding have to be modified.

Theorem 4.26 ("Kruskal's Theorem") *Let Σ be a finite signature, let X be a finite set of variables. Then for every infinite sequence $t_1, t_2, t_3, \dots$ there are indices $j > i$ such that $t_j \unrhd_{\text{emb}} t_i$. ($\unrhd_{\text{emb}}$ is called* a well-partial-ordering (wpo).*)*

Proof. See Baader and Nipkow, page 113–115. □

Theorem 4.27 (Dershowitz) *If Σ is a finite signature, then every simplification ordering $\succ$ on $\mathrm{T}_\Sigma(X)$ is well-founded (and therefore a reduction ordering).*

Proof. Suppose that $t_1 \succ t_2 \succ t_3 \succ \ldots$ is an infinite descending chain.

First assume that there is an $x \in \mathrm{var}(t_{i+1}) \setminus \mathrm{var}(t_i)$. Let $\sigma = \{x \mapsto t_i\}$, then $t_{i+1}\sigma \trianglerighteq x\sigma = t_i$ and therefore $t_i = t_i\sigma \succ t_{i+1}\sigma \succeq t_i$, contradicting irreflexivity.

Consequently, $\mathrm{var}(t_i) \supseteq \mathrm{var}(t_{i+1})$ and $t_i \in \mathrm{T}_\Sigma(V)$ for all i, where V is the finite set $\mathrm{var}(t_1)$. By Kruskal's Theorem, there are $i < j$ with $t_i \trianglelefteq_{\mathrm{emb}} t_j$. Hence $t_i \preceq t_j$, contradicting $t_i \succ t_j$. □

There are reduction orderings that are not simplification orderings and terminating TRSs that are not contained in any simplification ordering.

Example:

Let $R = \{f(f(x)) \to f(g(f(x)))\}$.

R terminates and $\to_R^+$ is therefore a reduction ordering.

Assume that $\to_R$ were contained in a simplification ordering $\succ$. Then $f(f(x)) \to_R f(g(f(x)))$ implies $f(f(x)) \succ f(g(f(x)))$, and $f(g(f(x))) \trianglerighteq_{\mathrm{emb}} f(f(x))$ implies $f(g(f(x))) \succeq f(f(x))$, hence $f(f(x)) \succ f(f(x))$.

Path Orderings

Let $\Sigma = (\Omega, \Pi)$ be a finite signature, let $\succ$ be a strict partial ordering *("precedence")* on Ω.

The *lexicographic path ordering* $\succ_{\mathrm{lpo}}$ on $\mathrm{T}_\Sigma(X)$ induced by $\succ$ is defined by: $s \succ_{\mathrm{lpo}} t$ iff

(1) $t \in \mathrm{var}(s)$ and $t \neq s$, or

(2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

- (a) $s_i \succeq_{\mathrm{lpo}} t$ for some i, or
- (b) $f \succ g$ and $s \succ_{\mathrm{lpo}} t_j$ for all j, or
- (c) $f = g$, $s \succ_{\mathrm{lpo}} t_j$ for all j, and $(s_1, \ldots, s_m)\ (\succ_{\mathrm{lpo}})_{\mathrm{lex}}\ (t_1, \ldots, t_n)$.

where $(\succ_{\mathrm{lpo}})_{\mathrm{lex}}$ is the m-fold lexicographic combination of $\succ_{\mathrm{lpo}}$
(note that $f = g$ implies $m = n$).

Lemma 4.28 *$s \succ_{\mathrm{lpo}} t$ implies* $\mathrm{var}(s) \supseteq \mathrm{var}(t)$.

Proof. By induction on $|s| + |t|$ and case analysis. □

Theorem 4.29 $\succ_{\text{lpo}}$ *is a simplification ordering on* $\mathrm{T}_\Sigma(X)$.

Proof. Show transitivity, subterm property, stability under substitutions, compatibility with Σ-operations, and irreflexivity, usually by induction on the sum of the term sizes and case analysis. Details: Baader and Nipkow, page 119/120. $\square$

Theorem 4.30 *If the precedence* $\succ$ *is total, then the lexicographic path ordering* $\succ_{\text{lpo}}$ *is total on ground terms, i. e., for all* $s, t \in \mathrm{T}_\Sigma(\emptyset)$*:* $s \succ_{\text{lpo}} t \lor t \succ_{\text{lpo}} s \lor s = t$.

Proof. By induction on $|s| + |t|$ and case analysis. $\square$

Recapitulation:

Let $\Sigma = (\Omega, \Pi)$ be a finite signature, let $\succ$ be a strict partial ordering *("precedence")* on Ω. The *lexicographic path ordering* $\succ_{\text{lpo}}$ on $\mathrm{T}_\Sigma(X)$ induced by $\succ$ is defined by: $s \succ_{\text{lpo}} t$ iff

(1) $t \in \text{var}(s)$ and $t \neq s$, or

(2) $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$, and

  (a) $s_i \succeq_{\text{lpo}} t$ for some i, or

  (b) $f \succ g$ and $s \succ_{\text{lpo}} t_j$ for all j, or

  (c) $f = g$, $s \succ_{\text{lpo}} t_j$ for all j, and $(s_1, \dots, s_m)\ (\succ_{\text{lpo}})_{\text{lex}}\ (t_1, \dots, t_n)$.

There are several possibilities to compare subterms in (2)(c):

- compare list of subterms lexicographically left-to-right (*"lexicographic path ordering (lpo)"*, Kamin and Lévy)
- compare list of subterms lexicographically right-to-left (or according to some permutation π)
- compare multiset of subterms using the multiset extension (*"multiset path ordering (mpo)"*, Dershowitz)
- to each function symbol $f/n \in \Omega$ with $n \geq 1$ associate a status $\in \{mul\} \cup \{\, lex_\pi \mid \pi : \{1, \dots, n\} \to \{1, \dots, n\}\,\}$ and compare according to that status (*"recursive path ordering (rpo) with status"*)

The Knuth-Bendix Ordering

Let $\Sigma = (\Omega, \Pi)$ be a finite signature, let $\succ$ be a strict partial ordering *("precedence")* on Ω, let $w : \Omega \cup X \to \mathbb{R}_0^+$ be a *weight function*, such that the following admissibility conditions are satisfied:

$w(x) = w_0 \in \mathbb{R}^+$ for all variables $x \in X$; $w(c) \geq w_0$ for all constants $c \in \Omega$.

If $w(f) = 0$ for some $f/1 \in \Omega$, then $f \succ g$ for all $g/n \in \Omega$ with $f \neq g$.

The weight function w can be extended to terms recursively:

$$w(f(t_1, \dots, t_n)) = w(f) + \sum_{1 \leq i \leq n} w(t_i)$$

or alternatively

$$w(t) = \sum_{x \in \mathrm{var}(t)} w(x) \cdot \#(x, t) + \sum_{f \in \Omega} w(f) \cdot \#(f, t).$$

where $\#(a, t)$ is the number of occurrences of a in t.

The *Knuth-Bendix ordering* $\succ_{\mathrm{kbo}}$ on $\mathrm{T}_\Sigma(X)$ induced by $\succ$ and w is defined by: $s \succ_{\mathrm{kbo}} t$ iff

(1) $\#(x, s) \geq \#(x, t)$ for all variables x and $w(s) > w(t)$, or

(2) $\#(x, s) \geq \#(x, t)$ for all variables x, $w(s) = w(t)$, and

  (a) $t = x$, $s = f^n(x)$ for some $n \geq 1$, or

  (b) $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$, and $f \succ g$, or

  (c) $s = f(s_1, \dots, s_m)$, $t = f(t_1, \dots, t_m)$, and $(s_1, \dots, s_m)\ (\succ_{\mathrm{kbo}})_{\mathrm{lex}}\ (t_1, \dots, t_m)$.

Theorem 4.31 *The Knuth-Bendix ordering induced by $\succ$ and w is a simplification ordering on $\mathrm{T}_\Sigma(X)$.*

Proof. Baader and Nipkow, pages 125–129. □

Remark

If $\Pi \neq \emptyset$, then all the term orderings described in this section can also be used to compare non-equational atoms by treating predicate symbols like function symbols.

4.6 Knuth-Bendix Completion

Completion:

Goal: Given a set E of equations, transform E into an equivalent convergent set R of rewrite rules.
(If R is finite: decision procedure for E.)

Knuth-Bendix Completion: Idea

How to ensure termination?

Fix a reduction ordering $\succ$ and construct R in such a way that $\rightarrow_R \subseteq \succ$ (i. e., $l \succ r$ for every $l \rightarrow r \in R$).

How to ensure confluence?

Check that all critical pairs are joinable.

Note: Every critical pair $\langle s, t\rangle$ can be *made* joinable by adding $s \rightarrow t$ or $t \rightarrow s$ to R.

(Actually, we first add $s \approx t$ to E and later try to turn it into a rule that is contained in $\succ$; this gives us some additional degree of freedom.)

Knuth-Bendix Completion: Inference Rules

The completion procedure is presented as a set of inference rules working on a set of equations E and a set of rules R: $E_0, R_0 \vdash E_1, R_1 \vdash E_2, R_2 \vdash \dots$

At the beginning, $E = E_0$ is the input set and $R = R_0$ is empty. At the end, E should be empty; then R is the result.

For each step $E, R \vdash E', R'$, the equational theories of $E \cup R$ and $E' \cup R'$ agree: $\approx_{E\cup R} = \approx_{E'\cup R'}$.

Notations:

The formula $s \mathrel{\dot{\approx}} t$ denotes either $s \approx t$ or $t \approx s$.

CP(R) denotes the set of all critical pairs between rules in R.

Orient:

$$\frac{E \cup \{s \mathrel{\dot{\approx}} t\},\quad R}{E,\quad R \cup \{s \to t\}} \qquad \text{if } s \succ t$$

Note: There are equations $s \approx t$ that cannot be oriented, i. e., neither $s \succ t$ nor $t \succ s$.

Trivial equations cannot be oriented – but we don't need them anyway:

Delete:

$$\frac{E \cup \{s \approx s\},\quad R}{E,\quad R}$$

Critical pairs between rules in R are turned into additional equations:

Deduce:

$$\frac{E,\quad R}{E \cup \{s \approx t\},\quad R} \qquad \text{if } \langle s,t\rangle \in \mathrm{CP}(R).$$

Note: If $\langle s,t\rangle \in \mathrm{CP}(R)$ then $s \leftarrow_R u \rightarrow_R t$ and hence $R \models s \approx t$.

The following inference rules are not absolutely necessary, but very useful (e. g., to get rid of joinable critical pairs and to deal with equations that cannot be oriented):

Simplify-Eq:

$$\frac{E \cup \{s \mathrel{\dot{\approx}} t\},\quad R}{E \cup \{u \approx t\},\quad R} \qquad \text{if } s \rightarrow_R u.$$

Simplification of the right-hand side of a rule is unproblematic:

R-Simplify-Rule:

$$\frac{E,\quad R \cup \{s \to t\}}{E,\quad R \cup \{s \to u\}} \qquad \text{if } t \rightarrow_R u.$$

Simplification of the left-hand side may influence orientability and orientation. Therefore, it yields an *equation*:

L-Simplify-Rule:

$$\frac{E,\quad R \cup \{s \to t\}}{E \cup \{u \approx t\},\quad R} \qquad \begin{array}{l}\text{if } s \rightarrow_R u \text{ using a rule } l \to r \in R\\ \text{such that } s \sqsupset l \text{ (see below).}\end{array}$$

For technical reasons, the lhs of $s \to t$ may only be simplified using a rule $l \to r$, if $l \to r$ *cannot* be simplified using $s \to t$, that is, if $s \sqsupset l$, where the *encompassment quasi-ordering* $\underset{\sim}{\sqsupset}$ is defined by

$$s \underset{\sim}{\sqsupset} l \ \text{ if } \ s|_p = l\sigma \text{ for some } p \text{ and } \sigma$$

and $\sqsupset = \underset{\sim}{\sqsupset} \setminus \underset{\sim}{\sqsubset}$ is the strict part of $\underset{\sim}{\sqsupset}$.

Lemma 4.32 $\sqsupset$ *is a well-founded strict partial ordering.*

Lemma 4.33 *If* $E, R \vdash E', R'$, *then* $\approx_{E \cup R} = \approx_{E' \cup R'}$.

Lemma 4.34 *If* $E, R \vdash E', R'$ *and* $\to_R \subseteq \succ$, *then* $\to_{R'} \subseteq \succ$.

Note: Like in ordered resolution, simplification should be preferred to deduction:

- Simplify/delete whenever possible.
- Otherwise, orient an equation, if possible.
- Last resort: compute critical pairs.

Knuth-Bendix Completion: Correctness Proof

If we run the completion procedure on a set E of equations, different things can happen:

(1) We reach a state where no more inference rules are applicable and E is not empty.
$\Rightarrow$ Failure (try again with another ordering?)

(2) We reach a state where E is empty and all critical pairs between the rules in the current R have been checked.

(3) The procedure runs forever.

In order to treat these cases simultaneously, we need some definitions.

A (finite or infinite sequence) $E_0, R_0 \vdash E_1, R_1 \vdash E_2, R_2 \vdash \ldots$ with $R_0 = \emptyset$ is called a *run* of the completion procedure with input E_0 and $\succ$.

For a run, $E_\infty = \bigcup_{i \geq 0} E_i$ and $R_\infty = \bigcup_{i \geq 0} R_i$.

The sets of *persistent equations or rules* of the run are $E_* = \bigcup_{i \geq 0} \bigcap_{j \geq i} E_j$ and $R_* = \bigcup_{i \geq 0} \bigcap_{j \geq i} R_j$.

Note: If the run is finite and ends with E_n, R_n, then $E_* = E_n$ and $R_* = R_n$.

A run is called *fair*, if $CP(R_*) \subseteq E_\infty$ (i. e., if every critical pair between persisting rules is computed at some step of the derivation).

Goal:

Show: If a run is fair and E_* is empty, then R_* is convergent and equivalent to E_0.

In particular: If a run is fair and E_* is empty, then $\approx_{E_0} = \approx_{E_\infty \cup R_\infty} = \leftrightarrow^*_{E_\infty \cup R_\infty} = \downarrow_{R_*}$.

General assumptions from now on:

$E_0, R_0 \vdash E_1, R_1 \vdash E_2, R_2 \vdash \ldots$ is a fair run.

R_0 and E_* are empty.

A *proof* of $s \approx t$ in $E_\infty \cup R_\infty$ is a finite sequence $(s_0, \ldots, s_n)$ such that $s = s_0$, $t = s_n$, and for all $i \in \{1, \ldots, n\}$:

(1) $s_{i-1} \leftrightarrow_{E_\infty} s_i$, or

(2) $s_{i-1} \rightarrow_{R_\infty} s_i$, or

(3) $s_{i-1} \leftarrow_{R_\infty} s_i$.

The pairs (s_{i-1}, s_i) are called *proof steps*.

A proof is called a *rewrite proof in R_**, if there is a $k \in \{0, \ldots, n\}$ such that $s_{i-1} \rightarrow_{R_*} s_i$ for $1 \leq i \leq k$ and $s_{i-1} \leftarrow_{R_*} s_i$ for $k + 1 \leq i \leq n$

Idea (Bachmair, Dershowitz, Hsiang):

Define a well-founded ordering on proofs, such that for every proof that is not a rewrite proof in R_* there is an equivalent smaller proof.

Consequence: For every proof there is an equivalent rewrite proof in R_*.

We associate a *cost* $c(s_{i-1}, s_i)$ with every proof step as follows:

(1) If $s_{i-1} \leftrightarrow_{E_\infty} s_i$, then $c(s_{i-1}, s_i) = (\{s_{i-1}, s_i\}, -, -)$, where the first component is a multiset of terms and $-$ denotes an arbitrary (irrelevant) term.

(2) If $s_{i-1} \rightarrow_{R_\infty} s_i$ using $l \rightarrow r$, then $c(s_{i-1}, s_i) = (\{s_{i-1}\}, l, s_i)$.

(3) If $s_{i-1} \leftarrow_{R_\infty} s_i$ using $l \rightarrow r$, then $c(s_{i-1}, s_i) = (\{s_i\}, l, s_{i-1})$.

Proof steps are compared using the lexicographic combination of the multiset extension of the reduction ordering $\succ$, the encompassment ordering $\sqsupset$, and the reduction ordering $\succ$.

The cost $c(P)$ of a proof P is the multiset of the costs of its proof steps.

The *proof ordering* $\succ_C$ compares the costs of proofs using the multiset extension of the proof step ordering.

Lemma 4.35 *$\succ_C$ is a well-founded ordering.*

Lemma 4.36 *Let P be a proof in $E_\infty \cup R_\infty$. If P is not a rewrite proof in R_*, then there exists an equivalent proof P' in $E_\infty \cup R_\infty$ such that $P \succ_C P'$.*

Proof. If P is not a rewrite proof in R_*, then it contains

(a) a proof step that is in E_∞, or
(b) a proof step that is in $R_\infty \setminus R_*$, or
(c) a subproof $s_{i-1} \leftarrow_{R_*} s_i \rightarrow_{R_*} s_{i+1}$ (peak).

We show that in all three cases the proof step or subproof can be replaced by a smaller subproof:

Case (a): A proof step using an equation $s \mathrel{\dot{\approx}} t$ is in E_∞. This equation must be deleted during the run.

If $s \mathrel{\dot{\approx}} t$ is deleted using *Orient*:

$$\dots s_{i-1} \leftrightarrow_{E_\infty} s_i \dots \quad \Longrightarrow \quad \dots s_{i-1} \rightarrow_{R_\infty} s_i \dots$$

If $s \mathrel{\dot{\approx}} t$ is deleted using *Delete*:

$$\dots s_{i-1} \leftrightarrow_{E_\infty} s_{i-1} \dots \quad \Longrightarrow \quad \dots s_{i-1} \dots$$

If $s \mathrel{\dot{\approx}} t$ is deleted using *Simplify-Eq*:

$$\dots s_{i-1} \leftrightarrow_{E_\infty} s_i \dots \quad \Longrightarrow \quad \dots s_{i-1} \rightarrow_{R_\infty} s' \leftrightarrow_{E_\infty} s_i \dots$$

Case (b): A proof step using a rule $s \rightarrow t$ is in $R_\infty \setminus R_*$. This rule must be deleted during the run.

If $s \rightarrow t$ is deleted using *R-Simplify-Rule*:

$$\dots s_{i-1} \rightarrow_{R_\infty} s_i \dots \quad \Longrightarrow \quad \dots s_{i-1} \rightarrow_{R_\infty} s' \leftarrow_{R_\infty} s_i \dots$$

If $s \rightarrow t$ is deleted using *L-Simplify-Rule*:

$$\dots s_{i-1} \rightarrow_{R_\infty} s_i \dots \quad \Longrightarrow \quad \dots s_{i-1} \rightarrow_{R_\infty} s' \leftrightarrow_{E_\infty} s_i \dots$$

Case (c): A subproof has the form $s_{i-1} \leftarrow_{R_*} s_i \rightarrow_{R_*} s_{i+1}$.

If there is no overlap or a non-critical overlap:

$$\dots s_{i-1} \leftarrow_{R_*} s_i \rightarrow_{R_*} s_{i+1} \dots \Longrightarrow \dots s_{i-1} \rightarrow_{R_*}^{*} s' \leftarrow_{R_*}^{*} s_{i+1} \dots$$

If there is a critical pair that has been added using *Deduce*:

$$\dots s_{i-1} \leftarrow_{R_*} s_i \rightarrow_{R_*} s_{i+1} \dots \Longrightarrow \quad \dots s_{i-1} \leftrightarrow_{E_\infty} s_{i+1} \dots$$

In all cases, checking that the replacement subproof is smaller than the replaced subproof is routine. □

Theorem 4.37 *Let $E_0, R_0 \vdash E_1, R_1 \vdash E_2, R_2 \vdash \ldots$ be a fair run and let R_0 and E_* be empty. Then*

(1) every proof in $E_\infty \cup R_\infty$ is equivalent to a rewrite proof in R_*,

(2) R_* is equivalent to E_0, and

(3) R_* is convergent.

Proof. (1) By well-founded induction on $\succ_C$ using the previous lemma.

(2) Clearly $\approx_{E_\infty \cup R_\infty} = \approx_{E_0}$. Since $R_* \subseteq R_\infty$, we get $\approx_{R_*} \subseteq \approx_{E_\infty \cup R_\infty}$. On the other hand, by (1), $\approx_{E_\infty \cup R_\infty} \subseteq \approx_{R_*}$.

(3) Since $\rightarrow_{R_*} \subseteq \succ$, R_* is terminating. By (1), R_* is confluent. $\square$

4.7 Unfailing Completion

Classical completion:

Try to transform a set E of equations into an equivalent convergent TRS.

Fail, if an equation can neither be oriented nor deleted.

Unfailing completion (Bachmair, Dershowitz and Plaisted):

If an equation cannot be oriented, we can still use *orientable instances* for rewriting.

Note: If $\succ$ is total on ground terms, then every *ground instance* of an equation is trivial or can be oriented.

Goal: Derive a *ground convergent* set of equations.

Let E be a set of equations, let $\succ$ be a reduction ordering.

We define the relation $\rightarrow_{E^\succ}$ by

$s \rightarrow_{E^\succ} t$ iff there exist $(u \approx v) \in E$ or $(v \approx u) \in E$,
$p \in \mathrm{pos}(s)$, and $\sigma : X \rightarrow \mathrm{T}_\Sigma(X)$,
such that $s|_p = u\sigma$ and $t = s[v\sigma]_p$ *and* $u\sigma \succ v\sigma$.

Note: $\rightarrow_{E^\succ}$ is terminating by construction.

From now on let $\succ$ be a reduction ordering that is total on ground terms.

E is called ground convergent w.r.t. $\succ$, if for all ground terms s and t with $s \leftrightarrow_E^* t$ there exists a ground term v such that $s \rightarrow_{E^\succ}^* v \leftarrow_{E^\succ}^* t$. (Analogously for $E \cup R$.)

As for standard completion, we establish ground convergence by computing critical pairs.

However, the ordering $\succ$ is not total on non-ground terms. Since $s\theta \succ t\theta$ implies $s \not\preceq t$, we approximate $\succ$ on ground terms by $\not\preceq$ on arbitrary terms.

Let $u_i \stackrel{.}{\approx} v_i$ $(i = 1, 2)$ be equations in E whose variables have been renamed such that $\mathrm{var}(u_1 \stackrel{.}{\approx} v_1) \cap \mathrm{var}(u_2 \stackrel{.}{\approx} v_2) = \emptyset$. Let $p \in \mathrm{pos}(u_1)$ be a position such that $u_1|_p$ is not a variable, σ is an mgu of $u_1|_p$ and u_2, and $u_i\sigma \not\preceq v_i\sigma$ $(i = 1, 2)$. Then $\langle v_1\sigma, (u_1\sigma)[v_2\sigma]_p\rangle$ is called a *semi-critical pair* of E with respect to $\succ$.

The set of all semi-critical pairs of E is denoted by $\mathrm{SP}_\succ(E)$.

Semi-critical pairs of $E \cup R$ are defined analogously. If $\rightarrow_R \subseteq \succ$, then $\mathrm{CP}(R)$ and $\mathrm{SP}_\succ(R)$ agree.

Note: In contrast to critical pairs, it may be necessary to consider overlaps of an equation with itself at the top. For instance, if $E = \{f(x) \approx g(y)\}$, then $\langle g(y), g(y')\rangle$ is a non-trivial semi-critical pair.

The *Deduce* rule takes now the following form:

Deduce:

$$\frac{E,\ \ R}{E \cup \{s \approx t\},\ \ R} \qquad \text{if } \langle s, t\rangle \in \mathrm{SP}_\succ(E \cup R).$$

Moreover, the fairness criterion for runs is replaced by

$$\mathrm{SP}_\succ(E_* \cup R_*) \subseteq E_\infty$$

(i. e., if every semi-critical pair between persisting rules or equations is computed at some step of the derivation).

Analogously to Thm. 4.37 we obtain now the following theorem:

Theorem 4.38 *Let $E_0, R_0 \vdash E_1, R_1 \vdash E_2, R_2 \vdash \ldots$ be a fair run; let $R_0 = \emptyset$. Then*

(1) $E_* \cup R_*$ is equivalent to E_0, and

(2) $E_* \cup R_*$ is ground convergent.

Moreover one can show that, whenever there exists a *reduced* convergent R such that $\approx_{E_0} = \downarrow_R$ and $\rightarrow_R \in \succ$, then for every fair *and simplifying* run $E_* = \emptyset$ and $R_* = R$ up to variable renaming.

Here R is called reduced, if for every $l \rightarrow r \in R$, both l and r are irreducible w. r. t. $R \setminus \{l \rightarrow r\}$. A run is called simplifying, if R_* is reduced, and for all equations $u \approx v \in E_*$, u and v are incomparable w. r. t. $\succ$ and irreducible w. r. t. R_*.

Unfailing completion is refutationally complete for equational theories:

Theorem 4.39 *Let E be a set of equations, let $\succ$ be a reduction ordering that is total on ground terms. For any two terms s and t, let $\hat{s}$ and $\hat{t}$ be the terms obtained from s and t by replacing all variables by Skolem constants. Let $eq/2$, $true/0$ and $false/0$ be new operator symbols, such that true and false are smaller than all other terms. Let $E_0 = E \cup \{\,eq(\hat{s},\hat{t}) \approx true,\ eq(x,x) \approx false\}$. If $E_0, \emptyset \vdash E_1, R_1 \vdash E_2, R_2 \vdash \ldots$ be a fair run of unfailing completion, then $s \approx_E t$ iff some $E_i \cup R_i$ contains $true \approx false$.*

Outlook:

Combine ordered resolution and unfailing completion to get a calculus for equational clauses:

compute inferences between (strictly) maximal literals as in ordered resolution,
compute overlaps between maximal sides of equations as in unfailing completion

$\Rightarrow$ Superposition calculus.

5 Termination Revisited

So far: Termination as a subordinate task for entailment checking.

TRS is generated by some saturation process; ordering must be chosen before the saturation starts.

Now: Termination as a main task (e. g., for program analysis).

TRS is fixed and known in advance.

Literature:

Nao Hirokawa and Aart Middeldorp: Dependency Pairs Revisited, RTA 2004, pp. 249-268 (in particular Sect. 1–4).

Thomas Arts and Jürgen Giesl: Termination of Term Rewriting Using Dependency Pairs, Theoretical Computer Science, 236:133-178, 2000.

5.1 Dependency Pairs

Invented by T. Arts and J. Giesl in 1996, many refinements since then.

Given: finite TRS R over $\Sigma = (\Omega, \emptyset)$.

$T_0 := \{\, t \in \mathrm{T}_\Sigma(X) \mid \text{there is an infinite derivation } t \to_R t_1 \to_R t_2 \to_R \dots \}$.

$T_\infty := \{\, t \in T_0 \mid \forall p > \varepsilon : t|_p \notin T_0 \,\}$ = minimal elements of T_0 w. r. t. $\rhd$.

$t \in T_0 \Rightarrow$ there exists a $t' \in T_\infty$ such that $t \trianglerighteq t'$.

R is non-terminating iff $T_0 \neq \emptyset$ iff $T_\infty \neq \emptyset$.

Assume that $T_\infty \neq \emptyset$ and consider some non-terminating derivation starting from $t \in T_\infty$. Since all subterms of t allow only finite derivations, at some point a rule $l \to r \in R$ must be applied at the root of t (possibly preceded by rewrite steps below the root):

$$t = f(t_1, \dots, t_n) \xrightarrow{>\varepsilon}{}^{*}_{R} f(s_1, \dots, s_n) = l\sigma \xrightarrow{\varepsilon}_{R} r\sigma.$$

In particular, $\mathrm{root}(t) = \mathrm{root}(l)$, so we see that the root symbol of any term in T_∞ must be contained in $D := \{\, \mathrm{root}(l) \mid l \to r \in R \,\}$. D is called the set of *defined symbols* of R; $C := \Omega \setminus D$ is called the set of *constructor symbols* of R.

The term $r\sigma$ is contained in T_0, so there exists a $v \in T_\infty$ such that $r\sigma \trianglerighteq v$.

If v occurred in $r\sigma$ at or below a variable position of r, then $x\sigma|_p = v$ for some $x \in \mathrm{var}(r) \subseteq \mathrm{var}(l)$, hence $s_i \trianglerighteq x\sigma$ and there would be an infinite derivation starting from some t_i. This contradicts $t \in T_\infty$, though.

Therefore, $v = u\sigma$ for some non-variable subterm u of r. As $v \in T_\infty$, we see that $\mathrm{root}(u) = \mathrm{root}(v) \in D$. Moreover, u cannot be a proper subterm of l, since otherwise again there would be an infinite derivation starting from some t_i.

Putting everything together, we obtain

$$t = f(t_1, \dots, t_n) \xrightarrow{>\varepsilon}{}_R^* f(s_1, \dots, s_n) = l\sigma \xrightarrow{\varepsilon}{}_R r\sigma \trianglerighteq u\sigma$$

where $r \trianglerighteq u$, u is not a variable, $\mathrm{root}(u) \in D$, $l \not\triangleright u$.

Since $u\sigma \in T_\infty$, we can continue this process and obtain an infinite sequence.

If we define $S := \{\, l \to u \mid l \to r \in R,\ r \trianglerighteq u,\ u \notin X,\ \mathrm{root}(u) \in D,\ l \not\triangleright u \,\}$, we can combine the rewrite step at the root and the subterm step and obtain

$$t \xrightarrow{>\varepsilon}{}_R^* l\sigma \xrightarrow{\varepsilon}{}_S u\sigma.$$

To get rid of the superscripts ε and $>\varepsilon$, it turns out to be useful to introduce a new set of function symbols $f^\sharp$ that are only used for the root symbols of this derivation:

$$\Omega^\sharp := \{\, f^\sharp/n \mid f/n \in \Omega \,\}.$$

For a term $t = f(t_1, \dots, t_n)$ we define $t^\sharp := f^\sharp(t_1, \dots, t_n)$; for a set of terms T we define $T^\sharp := \{\, t^\sharp \mid t \in T \,\}$.

The set of *dependency pairs* of a TRS R is then defined by

$$\mathrm{DP}(R) := \{\, l^\sharp \to u^\sharp \mid l \to r \in R,\ r \trianglerighteq u,\ u \notin X,\ \mathrm{root}(u) \in D,\ l \not\triangleright u \,\}.$$

For $t \in T_\infty$, the sequence using the S-rule corresponds now to

$$t^\sharp \to_R^* l^\sharp\sigma \to_{\mathrm{DP}(R)} u^\sharp\sigma$$

where $t^\sharp \in T_\infty^\sharp$ and $u^\sharp\sigma \in T_\infty^\sharp$.

(Note that rules in R do not contain symbols from $\Omega^\sharp$, whereas all roots of terms in $\mathrm{DP}(R)$ come from $\Omega^\sharp$, so rules from R can only be applied below the root and rules from $\mathrm{DP}(R)$ can only be applied at the root.)

Since $u^\sharp\sigma$ is again in $T_\infty^\sharp$, we can continue the process in the same way. We obtain: R is non-terminating iff there is an infinite sequence

$$t_1 \to_R^* t_2 \to_{\mathrm{DP}(R)} t_3 \to_R^* t_4 \to_{\mathrm{DP}(R)} \cdots$$

with $t_i \in T_\infty^\sharp$ for all i.

Moreover, if there exists such an infinite sequence, then there exists an infinite sequence in which all DPs that are used are used infinitely often. (If some DP is used only finitely often, we can cut off the initial part of the sequence up to the last occurrence of that DP; the remainder is still an infinite sequence.)

Dependency Graphs

Such infinite sequences correspond to “cycles” in the “dependency graph”:

Dependency graph $\mathrm{DG}(R)$ of a TRS R:

directed graph

nodes: dependency pairs $s \to t \in \mathrm{DP}(R)$

edges: from $s \to t$ to $u \to v$ if there are σ, τ such that $t\sigma \to_R^* u\tau$.

Intuitively, we draw an edge between two dependency pairs, if these two dependency pairs can be used after another in an infinite sequence (with some R-steps in between). While this relation is undecidable in general, there are reasonable overapproximations:

The functions cap and ren are defined by:

$$\begin{aligned}\mathrm{cap}(x) &= x\\ \mathrm{cap}(f(t_1,\dots,t_n)) &= \begin{cases} y & \text{if } f \in D\\ f(\mathrm{cap}(t_1),\dots,\mathrm{cap}(t_n)) & \text{if } f \in C \cup D^\sharp\end{cases}\end{aligned}$$

$$\begin{aligned}\mathrm{ren}(x) &= y, \quad y \text{ fresh}\\ \mathrm{ren}(f(t_1,\dots,t_n)) &= f(\mathrm{ren}(t_1),\dots,\mathrm{ren}(t_n))\end{aligned}$$

The overapproximated dependency graph contains an edge from $s \to t$ to $u \to v$ if $\mathrm{ren}(\mathrm{cap}(t))$ and u are unifiable.

A *cycle* in the dependency graph is a non-empty subset $K \subseteq \mathrm{DP}(R)$ such that there is a non-empty path in K from every DP in K to every DP in K (the two DPs may be identical).

Let $K \subseteq \mathrm{DP}(R)$. An infinite rewrite sequence in $R \cup K$ of the form

$$t_1 \to_R^* t_2 \to_K t_3 \to_R^* t_4 \to_K \dots$$

with $t_i \in T_\infty^\sharp$ is called K-minimal, if all rules in K are used infinitely often.

R is non-terminating iff there is a cycle $K \subseteq \mathrm{DP}(R)$ and a K-minimal infinite rewrite sequence.

5.2 Subterm Criterion

Our task is to show that there are no K-minimal infinite rewrite sequences.

Suppose that every dependency pair symbol $f^\sharp$ in K has positive arity (i. e., no constants). A *simple projection* π is a mapping $\pi : \Omega^\sharp \to \mathbb{N}$ such that $\pi(f^\sharp) = i \in \{1,\dots,\mathrm{arity}(f^\sharp)\}$.

We define $\pi(f^\sharp(t_1,\dots,t_n)) = t_{\pi(f^\sharp)}$.

Theorem 5.1 (Hirokawa and Middeldorp) *Let K be a cycle in* $\mathrm{DG}(R)$*. If there is a simple projection π for K such that $\pi(l) \trianglerighteq \pi(r)$ for every $l \to r \in K$ and $\pi(l) \triangleright \pi(r)$ for some $l \to r \in K$, then there are no K-minimal sequences.*

Proof. Suppose that

$$t_1 \to_R^* u_1 \to_K t_2 \to_R^* u_2 \to_K \dots$$

is a K-minimal infinite rewrite sequence. Apply π to every t_i:

Case 1: $u_i \to_K t_{i+1}$. There is an $l \to r \in K$ such that $u_i = l\sigma$, $t_{i+1} = r\sigma$. Then $\pi(u_i) = \pi(l)\sigma$ and $\pi(t_{i+1}) = \pi(r)\sigma$. By assumption, $\pi(l) \trianglerighteq \pi(r)$. If $\pi(l) = \pi(r)$, then $\pi(u_i) = \pi(t_{i+1})$. If $\pi(l) \triangleright \pi(r)$, then $\pi(u_i) = \pi(l)\sigma \triangleright \pi(r)\sigma = \pi(t_{i+1})$. In particular, $\pi(u_i) \triangleright \pi(t_{i+1})$ for infinitely many i (since every DP is used infinitely often).

Case 2: $t_i \to_R^* u_i$. Then $\pi(t_i) \to_R^* \pi(u_i)$.

By applying π to every term in the K-minimal infinite rewrite sequence, we obtain an infinite $(\to_R \cup \triangleright)$-sequence containing infinitely many $\triangleright$-steps. Since $\triangleright$ is well-founded, there must also exist infinitely many $\to_R$-steps (otherwise the infinite sequence would have an infinite tail consisting only of $\triangleright$-steps, contradicting well-foundedness.)

Now note that $\triangleright \circ \to_R \subseteq \to_R \circ \triangleright$. Therefore we can commute $\triangleright$-steps and $\to_R$-steps and move all $\to_R$-steps to the front. We obtain an infinite $\to_R$-sequence that starts with $\pi(t_1)$. However $t_1 \trianglerighteq \pi(t_1)$ and $t_1 \in T^\sharp_\infty$, so there cannot be an infinite $\to_R$-sequence starting from $\pi(t_1)$. □

Problem: The number of cycles in $\mathrm{DG}(R)$ can be exponential.

Better method: Analyze strongly connected components (SCCs).

SCC of a graph: maximal subgraph in which there is a non-empty path from every node to every node. (The two nodes can be identical.)[3]

Important property: Every cycle is contained in some SCC.

Idea: Search for a simple projection π such that $\pi(l) \trianglerighteq \pi(r)$ for all DPs $l \to r$ in the SCC. Delete all DPs in the SCC for which $\pi(l) \triangleright \pi(r)$ (by the previous theorem, there cannot be any K-minimal infinite rewrite sequences using these DPs). Then re-compute SCCs for the remaining graph and re-start.

No SCCs left $\Rightarrow$ no cycles left $\Rightarrow$ R is terminating.

Example: See Ex. 13 from Hirokawa and Middeldorp.

[3]There are several definitions of SCCs that differ in the treatment of edges from a node to itself.

5.3 Reduction Pairs and Argument Filterings

Goal: Show the non-existence of K-minimal infinite rewrite sequences

$$t_1 \rightarrow_R^* u_1 \rightarrow_K t_2 \rightarrow_R^* u_2 \rightarrow_K \dots$$

using well-founded orderings.

We observe that the requirements for the orderings used here are less restrictive than for reduction orderings:

K-rules are only used at the top, so we need stability under substitutions, but compatibility with contexts is unnecessary.

While $\rightarrow_K$-steps should be decreasing, for $\rightarrow_R$-steps it would be sufficient to show that they are not increasing.

This motivates the following definitions:

Rewrite quasi-ordering $\succsim$:

reflexive and transitive binary relation, stable under substitutions, compatible with contexts.

Reduction pair $(\succsim, \succ)$:

$\succsim$ is a rewrite quasi-ordering.

$\succ$ is a well-founded ordering that is stable under substitutions.

$\succsim$ and $\succ$ are compatible: $\succsim \circ \succ \subseteq \succ$ or $\succ \circ \succsim \subseteq \succ$.

(In practice, $\succ$ is almost always the strict part of the quasi-ordering $\succsim$.)

Clearly, for any reduction ordering $\succ$, $(\succeq, \succ)$ is a reduction pair. More general reduction pairs can be obtained using argument filterings:

Argument filtering π:

$\pi : \Omega \cup \Omega^\sharp \rightarrow \mathbb{N} \cup \text{list of } \mathbb{N}$

$$\pi(f) = \begin{cases} i \in \{1, \dots, \text{arity}(f)\}, \text{ or} \\ [i_1, \dots, i_k], \text{ where } 1 \le i_1 < \dots < i_k \le \text{arity}(f),\ 0 \le k \le \text{arity}(f) \end{cases}$$

Extension to terms:

$\pi(x) = x$

$\pi(f(t_1, \dots, t_n)) = \pi(t_i)$, if $\pi(f) = i$

$\pi(f(t_1, \dots, t_n)) = f'(\pi(t_{i_1}), \dots, \pi(t_{i_k}))$, if $\pi(f) = [i_1, \dots, i_k]$,
where f'/k is a new function symbol.

Let $\succ$ be a reduction ordering, let π be an argument filtering. Define $s \succ_\pi t$ iff $\pi(s) \succ \pi(t)$ and $s \succsim_\pi t$ iff $\pi(s) \succeq \pi(t)$.

Lemma 5.2 *$(\succsim_\pi, \succ_\pi)$ is a reduction pair.*

Proof. Follows from the following two properties:

$\pi(s\sigma) = \pi(s)\sigma_\pi$, where σ_π is the substitution that maps x to $\pi(\sigma(x))$.

$$\pi(s[u]_p) = \begin{cases} \pi(s), & \text{if } p \text{ does not correspond to any position in } \pi(s) \\ \pi(s)[\pi(u)]_q, & \text{if } p \text{ corresponds to } q \text{ in } \pi(s) \end{cases}$$ □

For interpretation-based orderings (such as polynomial orderings) the idea of "cutting out" certain subterms can be included directly in the definition of the ordering:

Reduction pairs by interpretation:

Let $\mathcal{A}$ be a Σ-algebra; let $\succ$ be a well-founded strict partial ordering on its universe.

Assume that all interpretations $f_\mathcal{A}$ of function symbols are *weakly monotone*, i. e., $a_i \succeq b_i$ implies $f(a_1, \dots, , a_n) \succeq f(b_1, \dots, b_n)$ for all $a_i, b_i \in U_\mathcal{A}$.

Define $s \succsim_\mathcal{A} t$ iff $\mathcal{A}(\beta)(s) \succeq \mathcal{A}(\beta)(t)$ for all assignments $\beta : X \to U_\mathcal{A}$; define $s \succ_\mathcal{A} t$ iff $\mathcal{A}(\beta)(s) \succ \mathcal{A}(\beta)(t)$ for all assignments $\beta : X \to U_\mathcal{A}$.

Then $(\succsim_\mathcal{A}, \succ_\mathcal{A})$ is a reduction pair.

For polynomial orderings, this definition permits interpretations of function symbols where some variable does not occur at all (e. g., $P_f(X_1, X_2) = 2X_1 + 1$ for a *binary* function symbol). It is no longer required that *every* variable must occur with some positive coefficient.

Theorem 5.3 (Arts and Giesl) *Let K be a cycle in the dependency graph of the TRS R. If there is a reduction pair $(\succsim, \succ)$ such that*

- *$l \succsim r$ for all $l \to r \in R$,*
- *$l \succsim r$ or $l \succ r$ for all $l \to r \in K$,*
- *$l \succ r$ for at least one $l \to r \in K$,*

then there is no K-minimal infinite sequence.

Proof. Assume that

$$t_1 \to_R^* u_1 \to_K t_2 \to_R^* u_2 \to_K \dots$$

is a K-minimal infinite rewrite sequence.

As $l \succsim r$ for all $l \to r \in R$, we obtain $t_i \succsim u_i$ by stability under substitutions, compatibility with contexts, reflexivity and transitivity.

As $l \succsim r$ or $l \succ r$ for all $l \to r \in K$, we obtain u_i $(\succsim \cup \succ)$ t_{i+1} by stability under substitutions.

So we get an infinite $(\succsim \cup \succ)$-sequence containing infinitely many $\succ$-steps (since every DP in K, in particular the one for which $l \succ r$ holds, is used infinitely often).

By compatibility of $\succsim$ and $\succ$, we can transform this into an infinite $\succ$-sequence, contradicting well-foundedness. $\square$

The idea can be extended to SCCs in the same way as for the subterm criterion:

Search for a reduction pair $(\succsim, \succ)$ such that $l \succsim r$ for all $l \to r \in R$ and $l \succsim r$ or $l \succ r$ for all DPs $l \to r$ in the SCC. Delete all DPs in the SCC for which $l \succ r$. Then re-compute SCCs for the remaining graph and re-start.

Example: Consider the following TRS R from [Arts and Giesl]:

$$\mathit{minus}(x, 0) \to x \tag{1}$$

$$\mathit{minus}(s(x), s(y)) \to \mathit{minus}(x, y) \tag{2}$$

$$\mathit{quot}(0, s(y)) \to 0 \tag{3}$$

$$\mathit{quot}(s(x), s(y)) \to s(\mathit{quot}(\mathit{minus}(x, y), s(y))) \tag{4}$$

(R is not contained in any simplification ordering, since the left-hand side of rule (4) is embedded in the right-hand side after instantiating y by $s(x)$.)

R has three dependency pairs:

$$\mathit{minus}^\sharp(s(x), s(y)) \to \mathit{minus}^\sharp(x, y) \tag{5}$$

$$\mathit{quot}^\sharp(s(x), s(y)) \to \mathit{quot}^\sharp(\mathit{minus}(x, y), s(y)) \tag{6}$$

$$\mathit{quot}^\sharp(s(x), s(y)) \to \mathit{minus}^\sharp(x, y) \tag{7}$$

The dependency graph of R is

$$\begin{tikzcd} (5) \arrow[loop below] & (7) \arrow[l] & (6) \arrow[l] \arrow[loop below] \end{tikzcd}$$

There are exactly two SCCs (and also two cycles). The cycle at (5) can be handled using the subterm criterion with $\pi(minus^\sharp) = 1$. For the cycle at (6) we can use an argument filtering π that maps *minus* to 1 and leaves all other function symbols unchanged (that is, $\pi(g) = [1, \dots, \text{arity}(g)]$ for every g different from *minus*.) After applying the argument filtering, we compare left and right-hand sides using an LPO with precedence $quot > s$ (the precedence of other symbols is irrelevant). We obtain $l \succ r$ for (6) and $l \succsim r$ for (1), (2), (3), (4), so the previous theorem can be applied.

DP Processors

The methods described so far are particular cases of *DP processors:*

A DP processor

$$\frac{(G, R)}{(G_1, R_1),\ \dots,\ (G_n, R_n)}$$

takes a graph G and a TRS R as input and produces a set of pairs consisting of a graph and a TRS.

It is sound and complete if there are K-minimal infinite sequences for G and R if and only if there are K-minimal infinite sequences for at least one of the pairs (G_i, R_i).

Examples:

$$\frac{(G, R)}{(SCC_1, R),\ \dots,\ (SCC_n, R)}$$

where $SCC_1, \dots, SCC_n$ are the strongly connected components of G.

$$\frac{(G, R)}{(G \setminus N, R)}$$

if there is an SCC of G and a simple projection π such that $\pi(l) \trianglerighteq \pi(r)$ for all DPs $l \to r$ in the SCC, and N is the set of DPs of the SCC for which $\pi(l) \vartriangleright \pi(r)$.

(and analogously for reduction pairs)

Innermost Termination

The dependency method can also be used for proving termination of *innermost rewriting:* $s \xrightarrow{\text{i}}_R t$ if $s \to_R t$ at position p and no rule of R can be applied at a position strictly below p. (DP processors for innermost termination are more powerful than for ordinary termination, and for program analysis, innermost termination is usually sufficient.)

6 Implementing Saturation Procedures

Problem:

Refutational completeness is nice in theory, but . . .

. . . it guarantees only that proofs will be found eventually, not that they will be found quickly.

Even though orderings and selection functions reduce the number of possible inferences, the search space problem is enormous.

First-order provers "look for a needle in a haystack": It may be necessary to make some millions of inferences to find a proof that is only a few dozens of steps long.

Coping with Large Sets of Formulas

Consequently:

- We must deal with large sets of formulas.
- We must use efficient techniques to find formulas that can be used as partners in an inference.
- We must simplify/eliminate as many formulas as possible.
- We must use efficient techniques to check whether a formula can be simplified/eliminated.

Note:

Often there are several competing implementation techniques.

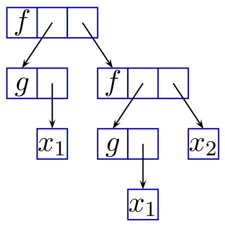
Design decisions are not independent of each other.

Design decisions are not independent of the particular class of problems we want to solve. (FOL without equality/FOL with equality/unit equations, size of the signature, special algebraic properties like AC, etc.)

6.1 Term Representations

The obvious data structure for terms: Trees

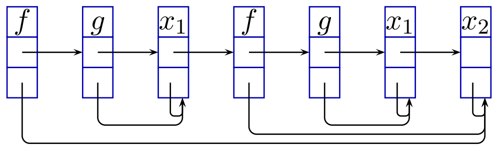
$f(g(x_1), f(g(x_1), x_2))$



optionally: (full) sharing

An alternative: Flatterms

$f(g(x_1), f(g(x_1), x_2))$



need more memory;
but: better suited for preorder term traversal and easier memory management.

6.2 Index Data Structures

Problem:

For a term t, we want to find all terms s such that

- s is an instance of t,
- s is a generalization of t (i. e., t is an instance of s),
- s and t are unifiable,
- s is a generalization of some subterm of t,
- ...

Requirements:

fast insertion,

fast deletion,

fast retrieval,

small memory consumption.

Note: In applications like functional or logic programming, the requirements are different (insertion and deletion are much less important).

Many different approaches:

- Path indexing
- Discrimination trees
- Substitution trees
- Context trees
- Feature vector indexing
- ...

Perfect filtering:

The indexing technique returns exactly those terms satisfying the query.

Imperfect filtering:

The indexing technique returns some superset of the set of all terms satisfying the query.

Retrieval operations must be followed by an additional check, but the index can often be implemented more efficiently.

Frequently: All occurrences of variables are treated as different variables.