Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern

SEKI - REPORT

Comparison of Simplification Orderings

Joachim Steinbach
SEKI Report SR-88-02

I Introduction

Term rewriting systems, also called sets of rewrite rules, gain more and more in importance, because they are a useful model for non-deterministic computations, with various applications in many areas of computer science and mathematics, including automatic theorem proving and program verification, data type specifications and algebraic simplification. Many of these fields require a terminating term rewriting system, i. e. no infinite derivation of terms should be possible. Termination of rewrite rules in general is an undecidable property, even if the number of rules is bounded by 2 ([HULA78], [DE85]). So the best we can hope for is, that we have different strategies at hand which together cope with many rewrite rule systems occuring in practise. A great many methods for proving termination of rewriting systems have been developed. All of them are based on the fact, that a rewrite system is terminating if and only if there exists a well-founded ordering > (a partial ordering without infinite descending sequences of terms) such that the left hand side is greater than (relative to >) the corresponding right hand side for each rule and for any substitution of the variables of the rule with terms ([DE82]). To illustrate this, let us consider a simple example: the system

$$\neg\neg x \longrightarrow x$$

$$x \wedge x \longrightarrow x$$

is terminating, since the number of symbols is reduced by each application of a rule, and the ordering on the size of terms is well-founded: s > t (s and t are terms) if and only if the number of symbols in s is greater than the number of symbols in t.

There are two desirable characteristics in this ordering, as well as in many others:

—the subterm property (any term is greater than any of its proper subterms)

—the replacement property (decreasing a subterm decreases any superterm containing it).

Orderings with these properties are called simplification orderings.

In this paper, we first draw parallels between the operational methods (methods of comparing terms) of some of the most popular simplification orderings (in section III). Among them there will be the recursive path ordering of Dershowitz (1982) and one of its extensions proposed by Forgaard (1984), the Knuth-Bendix ordering (1970), the recursive decomposition ordering of Jouannaud, Lescanne and Reinig (1982) and Rusinowitch (1985), the path of subterms ordering of Plaisted (1978) and Rusinowitch (1985) and the path ordering proposed by Kapur, Narendran and Sivakumar (1985). As Rusinowitch has discovered, the path of subterms ordering and the decomposition ordering have several similarities. This observation leads to the redefinition of the former using decompositions.

The second main point of this paper is the comparison of the power of these orderings, i. e. we will compare the sets of comparable terms for each combination of two orderings. This will be done under several prerequisites, including ground terms ([RU85]), (universal) terms ([RU85]), (universal) terms with an underlying total precedence and irrespective of a precedence, and monadic terms.

Thus, the report in hand is an extension of [RU85].

II Notations

We suppose the reader to be familiar with the features of term rewriting systems ([HUOP80]). Nevertheless we will briefly repeat some fundamental definitions on terms and on orderings.

Terms

We consider the set T of all *terms* constructed from elements of a set $\mathcal{F}$ of operator (or function) symbols and some denumerably infinite set X of variables. The leading function symbol and the tuple of the direct arguments of a term t is characterized as *Top(t)* and *Args(t)*, respectively. Terms may be viewed as trees with positions which are sequences of non negative integers. The set of all positions of a term t is called the set of *occurrences* and its abbreviation is $O(t)$. We write $t[u \leftarrow s]$ to denote the term that results from t by replacing t/u (the subterm according to u) by s at occurrence u.

A *substitution* σ is considered as a multiple replacement, simultaneously replacing all variables of a term by terms.

A *term rewriting system* (TRS, for short) $\mathfrak{R}$ over T is a finite or countably infinite set of rules, each of the form $l \rightarrow_{\mathfrak{R}} r$, where l and r are terms in T, such that every variable that occurs in r also occurs in l. Like this syntax, the semantic of rewrite systems is also very simple. A TRS $\mathfrak{R}$ generates a binary relation $\Rightarrow_{\mathfrak{R}}$ on T as follows: $s \Rightarrow_{\mathfrak{R}} t$ (term s *rewrites to* term t) if and only if s contains an instance $\sigma(l)$ of the left hand side of a rule $l \rightarrow_{\mathfrak{R}} r \in \mathfrak{R}$ and t originates from s by replacing the subterm $\sigma(l)$ by $\sigma(r)$.

A *derivation* in $\mathfrak{R}$ is a sequence $t_0 \Rightarrow_{\mathfrak{R}} t_1 \Rightarrow_{\mathfrak{R}} t_2 \Rightarrow_{\mathfrak{R}} \ldots$. For the sake of readability, we will use the symbols $\rightarrow$ and $\Rightarrow$ to denote the relations $\rightarrow_{\mathfrak{R}}$ and $\Rightarrow_{\mathfrak{R}}$, respectively if there is no ambiguity.

Termination

A TRS $\Re$ *terminates* if there is no infinite derivation in $\Re$. It is obvious that a rewrite system is not terminating if a derivation repeats a term: $...\Rightarrow t_i \Rightarrow ... \Rightarrow t_k \Rightarrow ...$ and $t_i = t_k$. A less special condition for nontermination is given if t_i is a subterm of t_k ($i < k$). This property is called *looping*. But $\Re$ needs not to be looping to be non-terminating. Consider the system consisting of the single rule $x * y \rightarrow (0 + x) * y$, which produces the infinite derivation $x * y \Rightarrow (0 + x) * y \Rightarrow (0 + (0 + x)) * y \Rightarrow (0 + (0 + (0 + x))) * y \Rightarrow ...$. Certainly, this TRS does not terminate and does not loop. A more general property than looping which detects the nontermination of the last example is called *homeomorphic embedding*. This is a binary relation $\trianglelefteq$ on terms. Let's write $s \trianglelefteq t$ if s may be obtained from t by deletion of selected symbols. We shall say that a derivation $t_1 \Rightarrow t_2 \Rightarrow ...$ is *self-embedding* if $t_i \trianglelefteq t_k$ for some $i < k$. Note that nontermination allows a self-embedding derivation ([DE82]). Moreover, self-embeddingness does not imply nontermination: the rule $(x^2)^2 \rightarrow (-(x^2))^2$ is self-embedding and nevertheless, terminates. But we can use homeomorphic embeddingness to specify a sufficient condition for the termination of rewrite systems.

To express proofs of termination we use the straightforward method ([DE85]) : a TRS $\Re$ over Γ is terminating if and only if there exists a well-founded ordering $>$ over Γ such that $s \Rightarrow_{\Re} t$ implies $s > t$ for all terms s and t in Γ. A *(partial) ordering* is a transitive and irreflexive binary relation $>$ and it is said to be *well-founded* if there are no infinite descending sequences of elements. If we take advantage of the structure of terms the following equivalent formulation holds : a rewrite system $\Re$ over Γ terminates if and only if there exists a well-founded ordering $>$ over Γ such that $\sigma(l) > \sigma(r)$ for each rule $l \rightarrow_{\Re} r$ and for any substitution σ of terms in Γ for the variables appearing in l, and that furthermore $>$ possesses the replacement property

([LA77]). An ordering $>$ has the *replacement property* if $t_1 > t_2$ implies $t[u \leftarrow t_1] > t[u \leftarrow t_2]$ for any $t, t_1, t_2 \in \Gamma$ and $u \in O(t)$. If we collect the properties of $>$ mentioned above (partial ordering and replacement property), adding the *subterm property* ($u \neq \varepsilon \Rightarrow t > t/u$), a useful class of well-founded orderings is defined, called *simplification orderings* (SO, for short) ([DE82]). According to the termination theorem of Lankford we can use any SO to prove termination. The relationship of a SO $>$ to the homeomorphic embeddingness $\trianglelefteq$ is implied by the embedding lemma of Dershowitz: $\trianglelefteq$ is contained in $\leq$ ([DE82]).

Since terms are concatenations of function symbols it is near at hand to use a special ordering on function symbols (called precedence) to define a (simplification) ordering. A *precedence* is a partially ordered set $(\mathcal{F}, \triangleright)$ consisting of the set of operators $\mathcal{F}$ and an irreflexive and transitive binary relation $\triangleright$ defined on elements of $\mathcal{F}$. Therefore we consider a SO $>$ as parametrized with one argument p (the precedence), written as $>(p)$. If there is no ambiguity, we will use the notation $>$ instead of $>(p)$.

Originally, a partial ordering $>$ works on elements of any set M. Since operators have terms as arguments we define an extension of $>$, called *lexicographically greater* ($>^{lex}$), on tuples of elements as follows: $(m_1,m_2,...,m_p) >^{lex} (n_1,n_2,...,n_q)$ if either $p>0 \,\&\, q=0$ or $m_1 > n_1$ or $m_1 = n_1 \,\&\, (m_2,...,m_p) >^{lex} (n_2,...,n_q)$. If there is no order among the elements of such tuples then the structures over M^n are called *multisets*, which are like sets, but allow multiple occurrences of identical elements. The extension of $>$ on multisets of elements is defined as follows: a multiset M_1 is greater than a multiset M_2 over M , denoted by $M_1 \gg M_2$ if M_2 can be obtained from M_1 by replacing one or more elements in M_1 by any finite number of elements, each of which is smaller (with respect to $>$ on M) than one of the replaced elements. For more details, the formal description in particular, see [DEMA79] or [ST86].

A partial ordering $>$ on any set M may also be extended in another way. An ordering $>'$ on M is an *extension* of $>$ if and only if $s > t$ implies $s >' t$ for all $s,t \in M$. Given this fact, we say $>$ is *included* in $>'$ and denote it by the terminology of sets : $> \subseteq >'$. A partial ordering $>'$ is said to be *total* if for any two distinct elements s, t (of M), either $s >' t$ or $t >' s$ holds. Given an ordering $>$ with the property that for any precedence p and for any terms with $s >(p) t$ it also holds that $s >(q) t$ for any extension q of p then $>$ is said to be *monotonic w.r.t. the precedence*: $p \subseteq q \rightarrow >(p) \subseteq >(q)$.

III Simplification orderings

All orderings described in this paper are recursively defined simplification orderings. Comparing two terms means to compare the whole terms and then the multisets (or tuples) consisting of the direct arguments. The main constituent of this section is the description of some simplification orderings. For a better understanding, these methods of comparing terms will be demonstrated by examples.

The following orderings satisfy properties that qualify them for proving termination of term rewriting systems:

- well-foundedness
- stability (with respect to substitutions)
- monotonicity w.r.t. the precedence.

We don't prove them here due to lack of space. However, the majority of the proofs are given in the respective references.

In the rest of this paragraph, when writing s,t and $\triangleright$ we will always assume that s and t are terms over Γ and $\triangleright$ is a precedence on the set $\mathcal{F}$ of operators. Permitting variables, we have to consider each and every one of them as an extra constant symbol uncomparable with all other operators in $\mathcal{F}$.

We make the point that the notation of the extensions of an ordering (at the end of section II) will be applied here. To define the orderings, we need some more designations which will be introduced at the beginning of the corresponding ordering.

Recursive path ordering and extended recursive path ordering

The idea of the comparison with respect to the recursive path ordering (rpo, for short) is that a term is decreased by replacing a subterm with any number of smaller terms which are connected by any structure of function symbols smaller (with respect to $\triangleright$) than the leading function symbol of the replaced subterm. The method of comparing two terms depends on the leading function symbols. The relation of these operators w.r.t. $\triangleright$ is responsible for decreasing one of the (or both) terms in the recursive definition of the rpo. If one of the terms is empty then the other is greater:

Definition ([DE82]): recursive path ordering

$s >_{rpo} t$

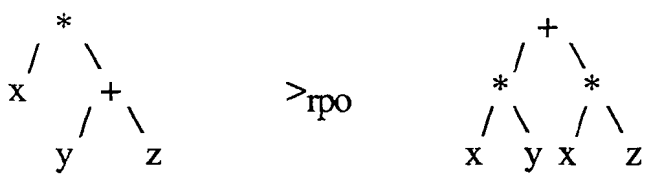
if $Args(s) \geq\geq_{rpo} \{t\}$

or $Top(s) \triangleright Top(t)$ & $\{s\} \gg_{rpo} Args(t)$

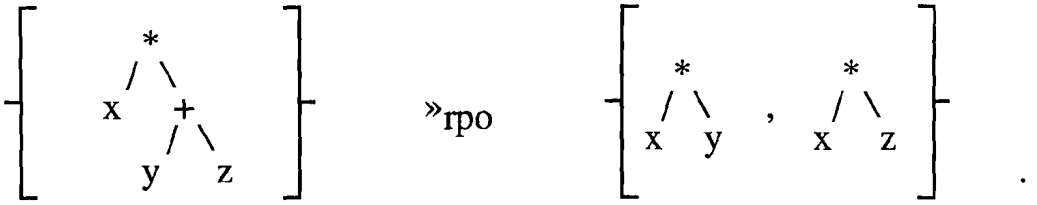
or $Top(s) = Top(t)$ & $Args(s) \gg_{rpo} Args(t)$

The relation $s \geq_{rpo} t$ is valid if $s >_{rpo} t$ or s and t are *permutatively congruent* ($s \approx t$) which means that s and t are equal syntactically when leaving out permutations of permutatively congruent subterms. Two zero-ary symbols (constants or variables) are permutatively congruent if they are syntactically equal.

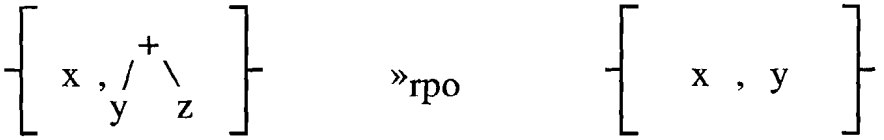
Example: Let $\mathcal{F} = \{*, +\}$. We wish to prove that the distributive law $x*(y+z) \rightarrow (x*y)+(x*z)$ terminates. We use the rpo with the operators total ordered by $* \triangleright +$.



Since * ▷ +, we must show



The single term on the left has to be greater than both terms on the right: it is greater than x*y, because we have to remove the leading function symbols and can show that



After removing the variable x in both multisets, the immediate subterm y (of y+z) is equal to the variable y of the right multiset: $Args(y+z) \geq\geq_{rpo} \{y\}$. $x*(y+z) >_{rpo} x*z$ is proved the same way.

An extension of this ordering (from Forgaard; we call it tpo) is very simple and bases on the fact that the rpo is monotonic w.r.t. the precedence. Only if a term s is greater than (with respect to the rpo) another term t for all total extensions of the given precedence, s is greater than t in this lifted ordering:

Definition ([DE85]): lifted recursive path ordering

$s >_{tpo(\triangleright)} t$

if $(\forall \triangleright'$ total$)$ $\triangleright \subseteq \triangleright' \Rightarrow s >_{rpo(\triangleright')} t$

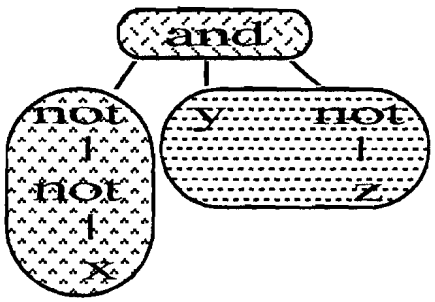
Realize that the distributive law treated in the example will be oriented in the same direction with the tpo, as we have to compare the terms with all total extensions of the precedence * ▷ + and, clearly, there is no other extension than itself (since * ▷ + is already total).

Recursive decomposition ordering and an extension

Like tpo, the recursive decomposition ordering (rdo in short) has arisen from the rpo. One of the important differences to the rpo is the fact that the rdo stops the comparison when it has to compare incomparable operators. The rdo is based on a preliminary analysis of terms, called decomposition, on which comparisons are recursively performed. The decomposition divides a term into three parts:

- the leading function symbol
- any selected immediate subterm
- the rest of the immediate subterms.

For instance



Formally, it is defined as follows: the *sub-decomposition* $D_u(t,p)$ of a term t at the occurrence p according to the terminal occurrence u, with $u \geq_o p$, is the tripel

$$D_u(t,p) = < Top(t/p) ; t' ; T >$$

where $t' = t/suc(p,u)$ and $T = Args(t/p) \setminus \{t'\}$.

A *terminal occurrence* is an occurrence of a leaf. We use the classical *prefix ordering* $>_o$ to compare occurrences: $u >_o v$ if and only if $u=v.w$ with $w \neq \varepsilon$. For $u \geq_o v$ we define $suc(v,u) = v.i$ if $u=v.i.w$ with $i \in N_+$, and $suc(v,v)=\infty$. By convention, t/∞ is the empty term. Furthermore, if $u=v.w$ we denote by $dif(u,v) = w$ the right quotient of u by v.

Example: Let $\mathcal{F}$ = {not , and} and t = and(not(not(x)),y,not(z)) the term shown as a tree on the preceding page. The sub-decomposition $D_{111}(t,\varepsilon)$ has the form <and ; not(not(x)) ; {y , not(z)}> . $D_2(t,\varepsilon)$ = <and ; y ; {not(not(x)) , not(z)}> and $D_{31}(t,31)$ = <z ; ; {}>.

Now we extend this definition to a set of prefixes of u (the *path-decomposition* $D_u(t)$) and then to the set of all terminal occurrences (the *decomposition* D(t)):

$$D_u(t) = \{\ D_u(t,p) \mid D_u(t,p) \text{ is a decomposition, } u \geq_o p\ \}$$

$$D(t) = \{\ D_u(t) \mid u \text{ is a terminal occurrence of } t\ \}.$$

Now a term s is greater than t (with respect to rdo) if the decomposition of s is greater than the decomposition of t. The ordering on these multisets ($\gg\gg_D$) is an extension of the basic ordering on sub-decompositions ($>_D$) to multisets of multisets:

Definition ([RU85]): recursive decomposition ordering

$s >_{\mathbf{rdo}} t$

if $D(s) \gg\gg_D D(t)$

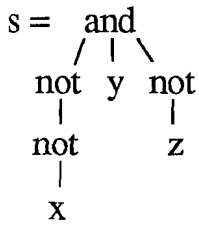
with $D_u(s,p) = \langle f ; s' ; S\rangle >_D \langle g ; t' ; T\rangle = D_v(t,q)$

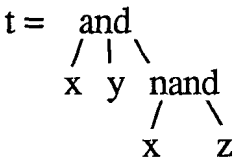
if lexicographically

- $f \triangleright g$
- $D_{dif(u,suc(p,u))}(s') \gg_D D_{dif(v,suc(q,v))}(t')$
- $S \gg_{rdo} T$

Terms are equal with respect to the rdo if they are permutatively congruent.

We will illustrate this slightly complicated definition by an example. Let s=and(not(not(x)),y,not(z)) and t=and(x,y,nand(x,z)) be terms constructed by the boolean operators not, and, nand. Moreover, suppose the precedence asserts not $\triangleright$ nand. To compare s and t, we first have to generate their decompositions :





$D(s) = \{ D_{111}(s) , D_2(s) , D_{31}(s) \}$

$D_{111}(s) = \{ D_{111}(s,\varepsilon) , D_{111}(s,1) , D_{111}(s,11) , D_{111}(s,111) \}$

$D_2(s) = \{ D_2(s,\varepsilon) , D_2(s,2) \}$

$D_{31}(s) = \{ D_{31}(s,\varepsilon) , D_{31}(s,3) , D_{31}(s,31) \}$

$D(t) = \{ D_1(t) , D_2(t) , D_{31}(t) , D_{32}(t) \}$

$D_1(t) = \{ D_1(t,\varepsilon) , D_1(t,1) \}$

$D_2(t) = \{ D_2(t,\varepsilon) , D_2(t,2) \}$

$D_{31}(t) = \{ D_{31}(t,\varepsilon) , D_{31}(t,3) , D_{31}(t,31) \}$

$D_{32}(t) = \{ D_{32}(t,\varepsilon) , D_{32}(t,3) , D_{32}(t,32) \}$

We want to prove that s is greater than t. Consequently, in accordance with the definition of the rdo, for every $D_v(t)$ we have to find a $D_u(s)$ which is greater with respect to $\gg_D$. By reason of stability for substitutions our search for $D_u(s)$ is restricted by the leaves s/u and t/v, respectively. We can verify

- $D_{111}(s) \gg_D D_1(t)$:

(i) $D_{111}(s,111) = D_1(t,1) = \langle x ; \ ; \{\} \rangle$

(ii) $D_{111}(s,\varepsilon) = \langle and ; not(not(x)) ; \{y , not(z)\} \rangle \ >_D \ \langle and ; x ; \{y , nand(x,z)\} \rangle = D_1(t,\varepsilon)$

because $D_{11}(not(not(x))) \gg_D D_\varepsilon(x)$

since $D_\varepsilon(x) \subseteq D_{11}(not(not(x)))$

- $D_{111}(s) \gg_D D_{31}(t)$:

(i) $D_{111}(s,111) = D_{31}(t,31) = \langle x\,;\ ;\ \{\}\rangle$

(ii) $D_{111}(s,11) = \langle not\,;\,x\,;\,\{\}\rangle >_D \langle nand\,;\,x\,;\,\{z\}\rangle = D_{31}(t,3)$
because $not \triangleright nand$

(iii) $D_{111}(s,\varepsilon) = \langle and\,;\,not(not(x))\,;\,\{y\,,\,not(z)\}\rangle >_D$
$\langle and\,;\,nand(x,z)\,;\,\{x\,,\,y\}\rangle = D_{31}(t,\varepsilon)$
because $D_{11}(not(not(x))) \gg_D D_1(nand(x,z))$

- $D_{31}(s) \gg_D D_{32}(t)$:

It is straightforward to prove this statement with the considerations of the latter.

- What about $D_2(t)$?

The only decomposition applicable is $D_2(s)$, since s/2 is the only leaf consisting of the variable y.

(i) $D_2(s,2) = D_2(t,2) = \langle y\,;\ ;\ \{\}\rangle$

(ii) $D_2(s,\varepsilon) = \langle and\,;\,y\,;\,\{not(not(x))\,,\,not(z)\}\rangle \not>_D$
$\langle and\,;\,y\,;\,\{x\,,\,nand(x,z)\}\rangle = D_2(t,\varepsilon)$
because neither not(not(x)) nor not(z) can bound nand(x,z) and they cannot help each other to do that. Instead of comparing the multisets of subterms it would be possible to compare the multiset sums of decompositions. Let s_1=not(not(x)), s_2=not(z) and t_1=x, t_2=nand(x,z). Consequently, $S=\{s_1,s_2\}$ and $T=\{t_1,t_2\}$.

Now we want to compare the multisets

$$D(s_1) \cup D(s_2) =$$

$$\{ \{D_{11}(s_1,\varepsilon), D_{11}(s_1,1), D_{11}(s_1,11)\}, \{D_1(s_2,\varepsilon), D_1(s_2,1)\} \}$$

and

$$\{ \{D_\varepsilon(t_1,\varepsilon)\}, \{D_1(t_2,\varepsilon), D_1(t_2,1)\}, \{D_2(t_2,\varepsilon), D_2(t_2,2)\} \}$$

$$= D(t_1) \cup D(t_2)$$

We can prove that $D(s_1) \cup D(s_2) \gg_D D(t_1) \cup D(t_2)$. It follows from the other comparisons already considered.

Summing up, we see that the comparison of the third part of a sub-decomposition ($S \gg_{rdo} T$), the multisets of subterms, is not ideal. A more successful test is to compare the multiset sums of the decompositions of these subterms. With this slight change we have a real extension of the rdo (ird in short). Concluding the example above, the following relations hold : $s \not>_{rdo} t$ but $s >_{ird} t$.

Definition ([RU85]): improved recursive decomposition ordering

$s >_{\mathbf{ird}} t$

if $D(s) \gg_{ED} D(t)$

with $D_u(s,p) = \langle f ; s' ; S \rangle >_{ED} \langle g ; t' ; T \rangle = D_v(t,q)$

if in a lexicographical way

- $f \triangleright g$
- $D_{dif(u,suc(p,u))}(s') \gg_{ED} D_{dif(v,suc(q,v))}(t')$
- $\bigcup_{s'' \in S} D(s'') \gg_{ED} \bigcup_{t'' \in T} D(t'')$

Remark: The original decomposition ordering works on decomposition-quadruplets instead of triples. An extra operator $\square$ is used to mark the place where a sub-decomposition is applied: $d_u(t,p) = D_u(t,p) \cdot \langle t [p \leftarrow \square] \rangle$. This additional fourth component is called *context*. Rusinowitch proved that it is redundant. From this fact results the definition of the rdo presented here.

Two path orderings

The decomposition orderings and the ordering defined next, Plaisted´s path of subterms ordering (pso, for short), have much in common. The pso is a predecessor of the rpo and its original definition depends on paths of subterms. Such a path of a term is a sequence starting with the whole term followed by a path from some argument : all paths of the term -(x*2) are [-(x*2) ; x*2 ; x] and [-(x*2) ; x*2 ; 2]:

The pso compares two terms by comparing all their paths. Its slightly modified version (equivalent to the original) of Rusinowitch is given next.

Definition ([PL78] , [RU85]): path of subterms ordering

$s >_{\mathbf{pso}} t$

if $SPATHS(s) \gg_{PO} SPATHS(t)$

with $p >_{PO} q$

if $MOS(p) \gg_{T} MOS(q)$

with $s' >_{T} t'$

if lexicographically

- $Top(s') \triangleright Top(t')$
- $\bigcup_{s_i \in Args(s')} SPATHS(s_i) \gg_{PO} \bigcup_{t_i \in Args(t')} SPATHS(t_i)$

SPATHS(t) denotes the multiset of paths of subterms of t : $SPATHS(t)=\{[t;t_1;...;t_n] \mid t_n$ is a leaf of t$\}$ and $[t;t_1;...;t_n]$ is a path of t. MOS(p) is the multiset of subterms occuring in the path p : $MOS([t_0;t_1;...;t_n]) = \{t_0 , t_1 , ... , t_n\}$.

It is conspicuous that the pso is an extremely recursive ordering based on three suborderings ($>_{PO}$, $>_T$ and $\triangleright$). We succeeded in redefining this path ordering in a simpler manner using decompositions. The new method is called psd and is based on the fact that a path is a kind of an ordered path-decomposition:

Definition: path of subterms ordering on decompositions

$s >_{psd} t$

if $D(s) \gg_P D(t)$

with $\langle f ; s' ; S \rangle >_P \langle g ; t' ; T \rangle$

if lexicographically

- $f \triangleright g$
- $\bigcup_{s'' \in \{s'\} \cup S} D(s'') \gg_P \bigcup_{t'' \in \{t'\} \cup T} D(t'')$

The proof of the equivalence between the pso and the psd is given in the next chapter. More information about the pso is available in [PL78], [RU85], [ST86].

This relatively simple definition of the complicated pso has two obvious advantages. First, we could compare the implementations relative to their efficiency. A closer look at this work will be done soon.

Secondly, it is easy to compare the pso with the decomposition orderings. The essential difference is that the pso works breadth first and the rdo works depth first ([RU85]) . The new definition of the pso makes it clear :

$\langle f ; s' ; S \rangle >_D \langle f ; t' ; T \rangle$ if $D_{u'}(s') \gg_D D_{v'}(t')$	$\langle f ; s' ; S \rangle >_P \langle f ; t' ; T \rangle$ if $\bigcup_{s'' \in \{s'\} \cup S} D(s'') \gg_P \bigcup_{t'' \in \{t'\} \cup T} D(t'')$

If the leading function symbols of the terms to compare are identical, rdo chooses only one subterm ($D_{u'}$ and $D_{v'}$, respectively). On the other hand, psd has regard to the decomposition multisets of all subterms simultaneously.

Like pso, **kns** is an ordering which compares terms using the paths in them. It has been devised by Kapur, Narendran and Sivakumar. They have implemented the rpo within their rewrite rule laboratory and have found it weak in handling terms which should intuitively be comparable. The kns was a consequence of these experiments. Rusinowitch proved that kns and ird are equivalent ([RU85]) and therefore we renounce to give a description of the former. For the sake of completeness only, we have mentioned this path ordering. The gentle reader is refered to [KANASI85], [RU85], [ST86].

Knuth-Bendix ordering

A quite different type of technique for ordering terms is based upon assigning natural (or possibly real) numbers to the function symbols and then to terms by adding the numbers of the operators (called weight of a term) they contain. Two terms are compared by comparing their weights, and if the weights are equal, by comparing the subterms lexicographically. To describe this strategy, called Knuth-Bendix ordering (kbo, for short), we need some assumptions and helpful definitions.

If Δ is a function symbol or a variable and t is a term we denote by $\#_\Delta(t)$ the *number of occurrences* of Δ in t. We assign a non negative integer $\varphi(f)$ (the *weight* of f) to each operator in $\mathcal{F}$ and a positive integer φ_0 to each variable , such that

- $\varphi(c) \geq \varphi_0$ if c is a constant
- $\varphi(f) > 0$ if f has one argument.

Now we extend the weight function to terms. For any term $t = f(t_1,...,t_n)$ let

$$\varphi(t) = \varphi(f) + \sum_{i=1}^{n} \varphi(t_i).$$

For example, let $\mathcal{F}$ = {nil , car , cdr , cons}. The weight function

symbol	x	nil	car	cdr	cons
φ	2	2	1	1	0

is correctly defined, whereas φ_0=2, φ(nil)=1, ... isn´t allowed. The weight of the term cons(car(cdr(x)),nil) is 0+1+1+2+2 = 6.

Definition ([KNBE70] , [MA87]): Knuth-Bendix ordering

$s >_{kbo} t$

if $(\forall x \in \mathcal{X})\ \#_x(s) \geq \#_x(t)$

and lexicographically

- $\varphi(s) > \varphi(t)$
- $Top(s) \triangleright Top(t)$
- $Args(s) >_{kbo}^{lex} Args(t)$

The variable condition guaranteeing the stability certainly is a very strong restriction. Note that, for example, the distributive law cannot be oriented in the usual direction.

We also want to remark that there exists a slight improvement of the ordering, allowing at most one unary operator f with weight zero ([KNBE70]). To conserve the well-foundedness all other operators from $\mathcal{F}$ have to be smaller than f (with respect to the precedence).

We conclude this section with an example : let $\mathcal{F} = \{ -, * \}$ and $\mathcal{X} = \{ x \}$.

symbol	x	-	*
φ	1	1	0

$\triangleright$ is the empty precedence.

Consider the terms $s = (-x)*x$ and $t = x*(-x)$. Since $\varphi(s) = \varphi(t) = 3$ and $Top(s) = Top(t) = *$ we have to apply the kbo recursively on the first arguments and have to verify : $-x >_{kbo} x$ .This is true because $\varphi(-x) = 2 > 1 = \varphi(x)$ and therefore $s >_{kbo} t$.

IV Comparison

In this chapter we compare the power of the orderings presented. The power of an ordering is the set of comparable terms. We do not compare the size of these sets but examine the relation between two sets. There are three possible relations. Two orderings could be *equivalent* ($> = >'$), one ordering may be *properly included* in the other ($> \subset >'$) or they *overlap* ($> \# >'$). The orderings $>$ and $>'$ overlap if there exist some terms such that $s_1 > t_1 \,\&\, \neg(s_1 >' t_1)$ and $s_2 >' t_2 \,\&\, \neg(s_2 > t_2)$. Consequently, the proof of such an overlapping is composed by specifying two counter-examples. These kinds of proofs together with a synopsis of the previous lemmata will be listed as diagrams at the end of this chapter.

Note that the orderings described in this report depend on the parameter of the precedence. This parameter may be either partial or total. For that reason we have diagrams with an underlying partial precedence and with an underlying total precedence. Furthermore, the dependences of the orderings will be checked on additional premises where the set of terms is restricted to ground terms (terms without variables) and monadic terms (terms built up with zero-ary and unary symbols only), respectively.

Ground terms

If Γ is the set of all ground terms and $\triangleright$ is total, then the path orderings (pso and kns), the decomposition orderings (psd, rdo and ird) and the recursive path ordering are total on $\Gamma/\approx$ and therefore equivalent ([RU85] and the results on the following pages). The Knuth-Bendix ordering is total on Γ ([KNBE70]) and overlaps with the others, since $1 >_{kbo} not(0)$ if $\varphi(1) > \varphi(not) + \varphi(0)$ and $not(0) >' 1$ for any of the remaining orderings $>'$ under the total precedence $not \triangleright 0 \triangleright 1$.

Terms with an underlying partial precedence

A (universal) term is an element of the set Γ of all terms. Most of the considerations significant for this section are made by Rusinowitch ([RU85]). An overview of his results will be given and some new outcomes will be proved explicitly.

For the rest of this section, we assume that Γ is the set of all terms and $\triangleright$ is a partial precedence. Then the recursive path ordering is properly included in the recursive decomposition ordering ([REJO81]) and the path of subterms ordering. Furthermore, it is less powerful than its lifted version from Forgaard:

Lemma 1 $>_{rpo} \subset >_{tpo}$.

Proof : Let $s >_{rpo(\triangleright)} t$. Since the rpo is monotonic w.r.t. the precedence, we can conclude that

$$s >_{rpo(\triangleright')} t \qquad \text{if } \triangleright \subseteq \triangleright'.$$

This fact is valid particularly if $\triangleright'$ is total and therefore $s >_{tpo} t$ (according to the definition of the tpo).

❑

The recursive decomposition ordering is properly included in its improved variant which is equivalent to the path ordering of Kapur, Narendran and Sivakumar. The other path ordering (pso) is the connection link to the decomposition orderings, since it is equivalent to the psd:

Lemma 2 $>_{pso} = >_{psd}$.

Proof : Proving the equivalence of the two different definitions, we change the original pso step by step to obtain the form built up by decompositions.

Let $MOS^*(t) = \{MOS(p) \mid p \in SPATHS(t)\}$ be the multiset of all multisets of subterms, then

$s >_{pso} t$

if $MOS^*(s) \gg_{T_1} MOS^*(t)$

with $s' >_{T_1} t'$

if in a lexicographical way

- $Top(s') \triangleright Top(t')$
- $\bigcup_{s_i \in Args(s')} MOS^*(s_i) \gg_{T_1} \bigcup_{t_i \in Args(t')} MOS^*(t_i)$

The multiset $MOS(p) = \{t_0,t_1,...,t_n\}$ of subterms occuring in a path p only contains terms. Guiding to decompositions, we modify these terms in such a way that they are vectors consisting of the leading function symbol and the list of arguments : $t_i \dashrightarrow \langle Top(t_i) ; Args(t_i)\rangle$. This transformation of MOS(p) changes the definition of the pso in the following way :

$s >_{pso} t$

if $MOS^*(s) \gg_{T_2} MOS^*(t)$

with $\langle f ; S\rangle >_{T_2} \langle g ; T\rangle$

if lexicographically

- $f \triangleright g$
- $\bigcup_{s_i \in S} MOS^*(s_i) \gg_{T_2} \bigcup_{t_i \in T} MOS^*(t_i)$

Splitting up the second component of an element of MOS(p) leads us to a sub-decomposition: let $MOS(p) = \{ \langle Top(t_0) ; T_0\rangle , \langle Top(t_1) ; T_1\rangle , \dots , \langle Top(t_{n-1}) ; T_{n-1}\rangle , \langle Top(t_n) ; \{\}\rangle \}$. Moreover, let $t_0=t$ and $t/u=t_n$, then

$D_u(t) = \{ \langle Top(t_0) ; t_1 ; T_0'\rangle , \langle Top(t_1) ; t_2 ; T_1'\rangle , \dots , \langle Top(t_{n-1}) ; t_n ; \{\}\rangle , \langle Top(t_n) ; ; \{\}\rangle \}$

$\rightarrow D_u(t) = \{\langle Top(t_i) ; t_{i+1} ; T_i'\rangle \mid \langle Top(t_i) ; \{t_{i+1}\} \cup T_i'\rangle \in MOS(p)\}$

$\Rightarrow$ $s >_{pso} t$

if $D(s) \gg_{T3} D(t)$

with $\langle f ; s' ; S \rangle >_{T3} \langle g ; t' ; T \rangle$

if in a lexicographical way

- $f \triangleright g$
- $\bigcup_{s'' \in \{s'\} \cup S} D(s'') \gg_{T3} \bigcup_{t'' \in \{t'\} \cup T} D(t'')$

This definition is identical with that on page 16, leaving out the denotation of $>_{T3}$ and $>_P$, which concludes the proof.

❑

All other combinations of two orderings are overlappings.

Terms with an underlying total precedence

If we assume an underlying total precedence, one of these overlappings turns into a proper inclusion:

Lemma 3 $>_{psd} \subset >_{ird}$.

Proof : We have to show that $s >_{psd} t \Rightarrow s >_{ird} t$

$\leftrightarrow D(s) \gg_P D(t) \Rightarrow D(s) \gg_{ED} D(t)$ (definition of $>_{psd}$ and $>_{ird}$)

$D(s) \gg_P D(t) \leftrightarrow (\forall v \in Ot(t))(\exists u \in Ot(s)) D_u(s) \gg_P D_v(t)$

with $Ot(t)$ denotes the set of all terminal occurrences of the term t.

We have to show: $D_u(s) \gg_P D_v(t)$ (*)

$\Rightarrow (\exists u' \in Ot(s)) D_{u'}(s) \gg_{ED} D_v(t)$

Let $D_u(s) = \{ \langle f_i ; s_{i+1} ; S_i \rangle \mid i \in [0,m] \}$ with $s_{i+1} \in Args(s_i)$ & $s/u_i = s_i$

$D_v(t) = \{ \langle g_i ; t_{i+1} ; T_i \rangle \mid i \in [0,n] \}$ with $t_{i+1} \in Args(t_i)$ & $t/v_i = t_i$

$\rightarrow (\forall i)(\exists j)\ f_j \triangleright g_i \vee f_j = g_i$, since (*) and $\triangleright$ total

case (i) : $(\forall i)(\exists j)\ f_j \triangleright g_i \rightarrow u' = u$ ✔

case (ii): Let $g_p = f_q \rightarrow \bigcup_{s'' \in \{s_{q+1}\} \cup S_q} D(s'') \ \gg_P \bigcup_{t'' \in \{t_{p+1}\} \cup T_p} D(t'')$, since (*)

$\rightarrow (\exists s' \in \{s_{q+1}\} \cup S_q)(\exists u'' \in O(s))\ D_{dif(u'',suc(u_q,u''))}(s') = D_{s'}$

$\geq\geq_P\ D_{t'} = D_{dif(v,suc(v_p,v))}(t_{p+1})$

(ii.1) $D_{s'} \gg_P D_{t'} \rightarrow u' = u''$ ✔

(ii.2) $D_{s'} = D_{t'}$

$\rightarrow$ we have to show: $\bigcup_{s'' \in S_q} D(s'') \ \gg\gg_{ED} \bigcup_{t'' \in T_p} D(t'')$

this is valid by induction if we choose $u' = u''$, since

$\bigcup_{s'' \in S_q} D(s'') \ \gg\gg_P \bigcup_{t'' \in T_p} D(t'')$

❑

The superiority of the tpo in relation to the rpo ($>_{rpo} \subset >_{tpo}$) expires:

Lemma 4 $>_{rpo} = >_{tpo}$.

Proof : Since $\triangleright$ is total, there is no other extension than itself. This fact leads to the assertion of the lemma.

❑

The relations being left are the same as under a partial precedence.

Terms irrespective of a precedence

Our results would be stronger and more general if we could give some declaration about the comparisons of orderings separated from the precedence. The following proposition summarizes these reflections.

Proposition Let $>$ and $>'$ be some orderings monotonic w.r.t. the precedence. Furthermore, let $>$ be included in $>'$ over any total precedence. Then, having $s >(p) t$ for some precedence p, there exists a precedence q so that $s >'(q) t$.

Proof: We prove this proposition by specifying q. Let $s >(p) t$, then $s >(p'') t$ for each extension p'' of p, since $>$ is monotonic w.r.t. the precedence. Let q be one of the total extensions: $s >(q) t$. With the additional premise $> \subseteq >'$ over any total precedence, the terms s and t are ordered in the same direction under $>'$.

❑

This statement will get practical importance if we consider it together with the relations between orderings with an underlying total precedence (see figure 2 on page 28): only two (ird and kbo or kns and kbo) of the eight orderings collected in the diagram are sufficient to cover the union of comparable terms of all orderings presented here. In other words, if terms can be oriented with any ordering (of figure 2) there exists a precedence so that the terms are also comparable with the ird (kns) or the kbo. Consequently, if you are implementing a system where termination of rewriting must be guaranteed, then you only have to make available to the user two of the eight orderings.

Monadic terms

A *monadic term* only contains unary function symbols and either a constant or a variable. The subset of the monadic terms without constants can be unequivocally transformed into strings and vice versa. For this reason the subsequent results refer to the corresponding orderings on string rewriting systems. The proofs cannot be explicitly given here due to lack of space, but may be found in [ST86]. We assume that $\triangleright$ is a partial precedence.

Lemma 5 - $>_{rdo} = >_{ird}$

- $>_{pso} = >_{kns}$
- $>_{pso} \subset >_{tpo}$.

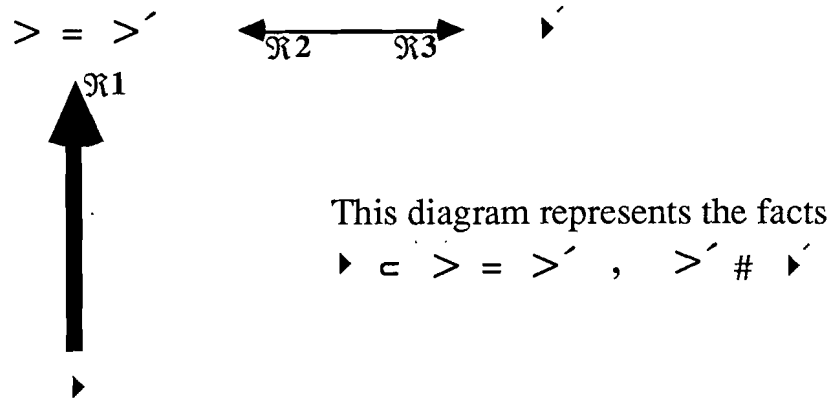
Proof : [ST86]

❑

These three lemmata and the relations between orderings on terms with an underlying partial precedence yield the dependences as follows: the path orderings and the decomposition orderings coincide. They are more powerful than the rpo and properly included in the tpo.

Having a total precedence, all orderings with the exception of the kbo are the same. The counter-example on page 20 is responsible for the overlapping of the kbo with the others.

In order to retain these relations and to find one of them easily we use slightly modified Hasse diagrams:



We have two different kinds of arrows: if $\succ \subset \rightarrow$ then we arrange $\rightarrow$ above $\succ$ joining them with a thick arrow; if two orderings overlap, we use a thin arrow. If an ordering $\rightarrow$ is not included in another ordering $\succ$ then there must be a rule that can be oriented by $\rightarrow$ but not by $\succ$. Such a rule is named by $\Re i$ and is given explicitly on page 29. For instance, the rule $\Re 1$ is orientable by $>$ and not by $\succ$. The rules $\Re 2$ and $\Re 3$ are interpreted analogously.

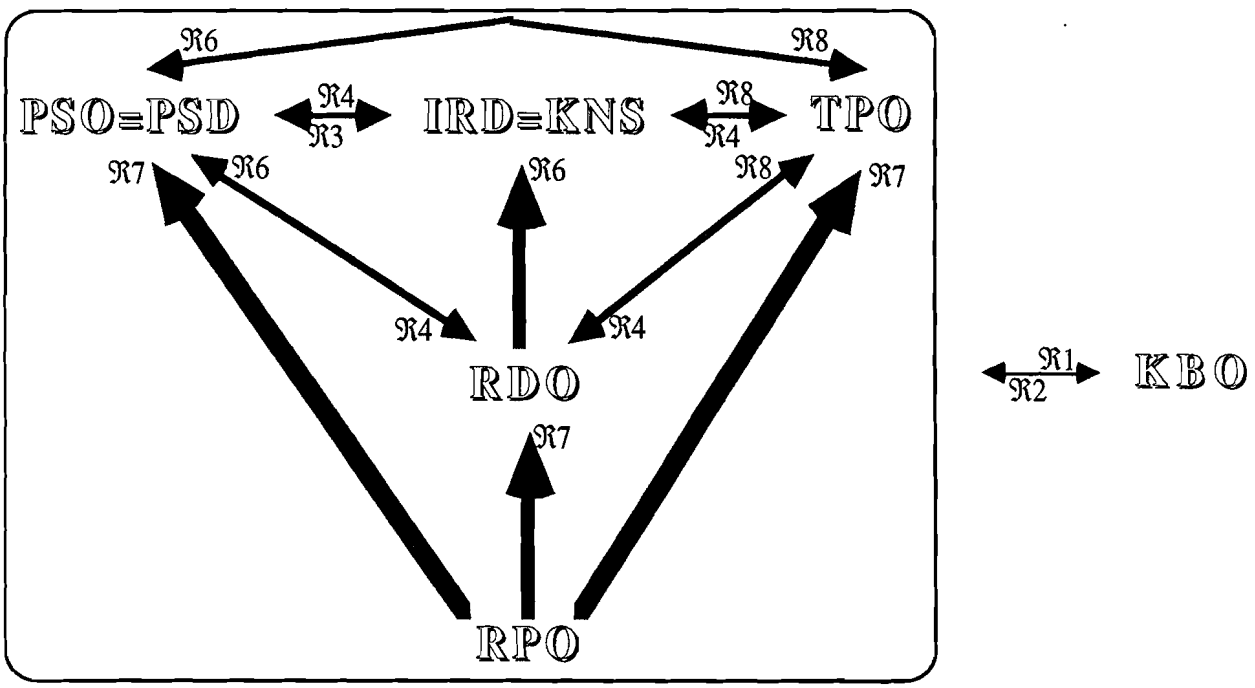


Figure 1: Universal terms and partial precedence

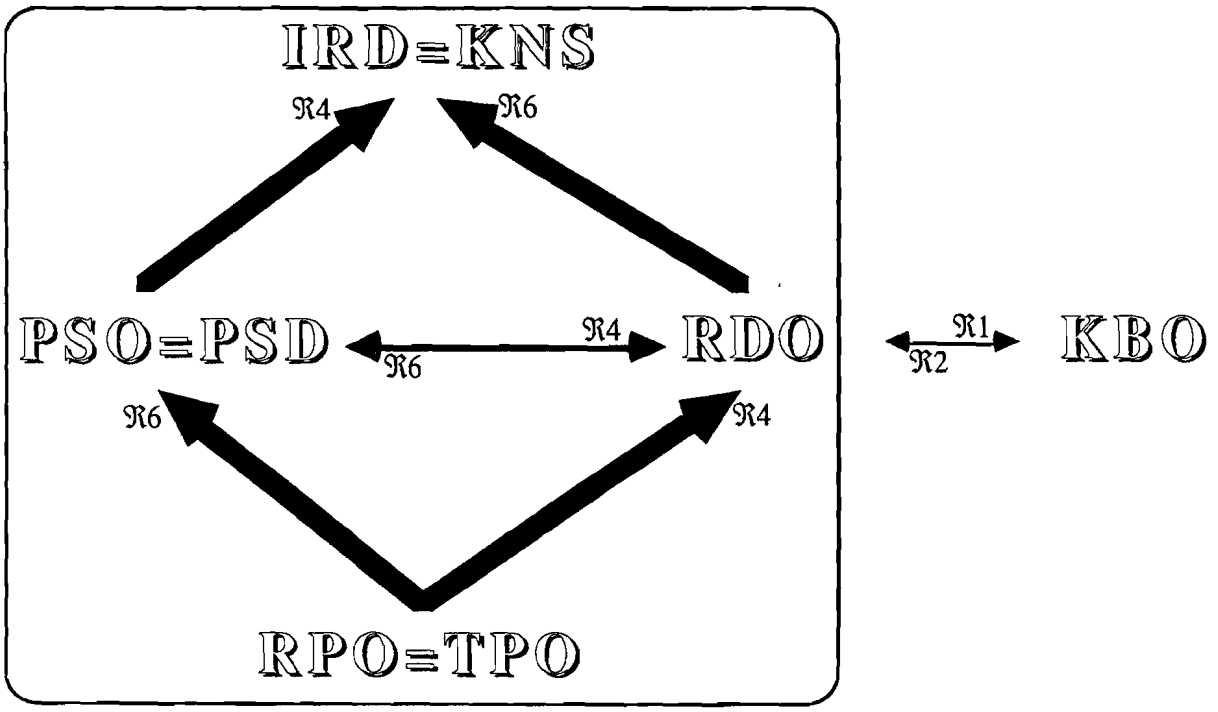


Figure 2 : Universal terms and total precedence

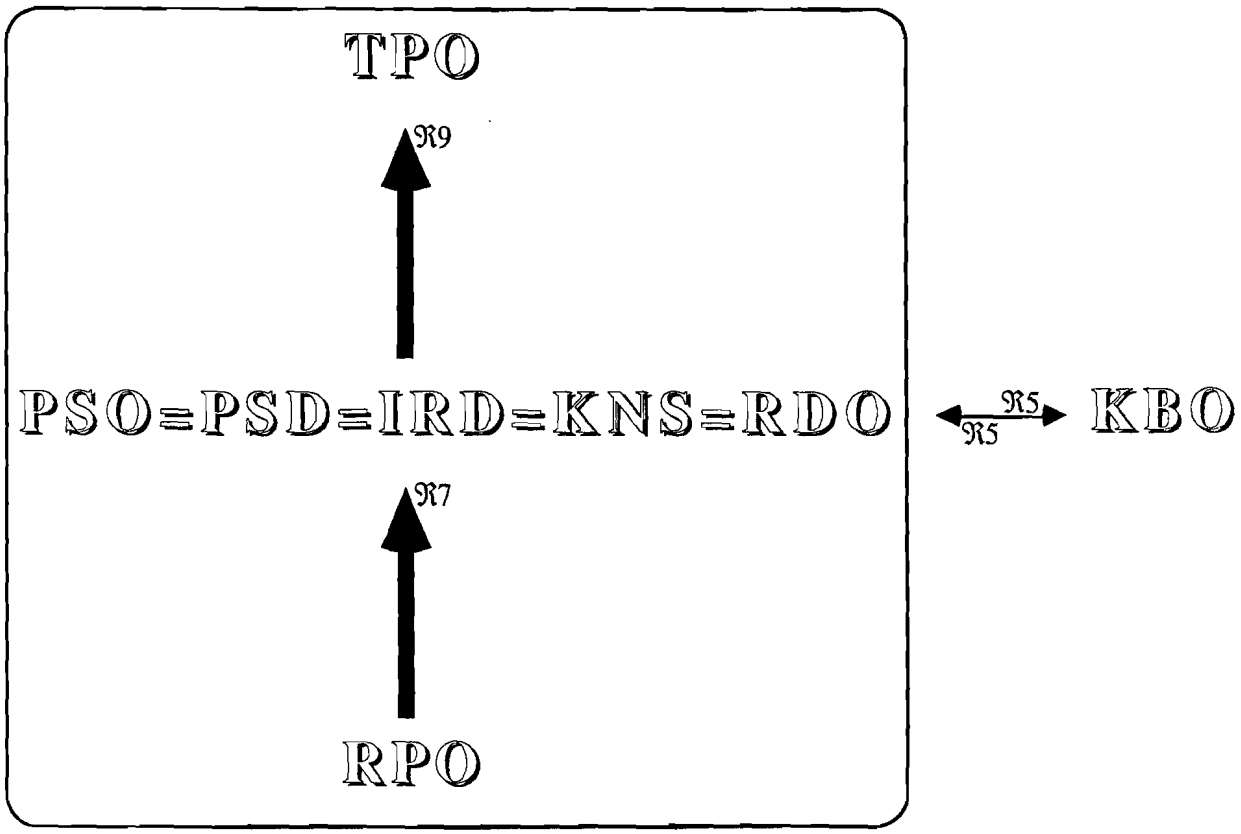


Figure 3 : Monadic terms and partial precedence

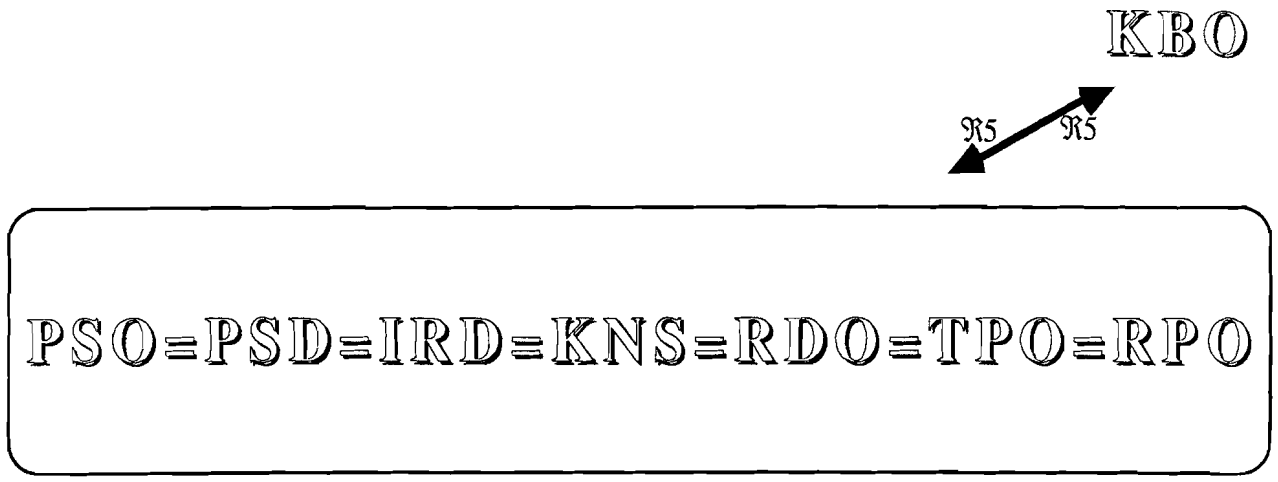


Figure 4 : Ground terms and total precedence

Counter-examples

The following pairs of terms are witness for the overlappings of the orderings.

ℜ1 $(-x) * x \rightarrow x * (-x)$

ℜ2 $x * (y + z) \rightarrow (x * y) + (x * z)$
with $* \triangleright +$

ℜ3 $0 + -1 \rightarrow -(0 + 1)$
with $+ \triangleright 0$, $-1 \triangleright - \triangleright 1$

ℜ4 $((x \wedge x) \wedge (x \wedge x)) \wedge ((y \wedge y) \wedge (y \wedge y)) \rightarrow (x \wedge y) \wedge (x \wedge y)$

ℜ5 $not(0) \leftrightarrow 1$
with $\varphi(1)=3, \varphi(0)=\varphi(not)=1$ and $not \triangleright 0 \triangleright 1$

ℜ6 $and(not(not(x)),y,not(z)) \rightarrow and(x,y,nand(x,z))$
with $not \triangleright nand$

ℜ7 $first(rest(x)) \rightarrow car(cdr(x))$
with $first \triangleright cdr$, $rest \triangleright car$

ℜ8 $cons(car(x),cdr(x)) \rightarrow first(list(x))$
with $car \triangleright first$, $cdr \triangleright list$

ℜ9 $sqrt(sqr(sqr(x))) \rightarrow sqr(-(x))$
with $sqrt \triangleright -$

V Concluding remarks

Various methods for proving the termination of term rewriting systems have been suggested. Most of them are based on the following notion of a simplification ordering : any term that is syntactically simpler than another is smaller than the other.

A collection of simplification orderings has been pointed out, including recursive path ordering and recursive decomposition ordering. Four of the orderings presented are part of an implementation of the Knuth-Bendix completion procedure, called COMTES. A series of experiments have been conducted to study the time complexity of the orderings kbo, rpo, kns and pso (the last two orderings are implemented over paths). The implementation of the ird is in progress just now. Summarizing, we obtained the average time factors

kbo: 1 rpo: 5 kns: 25 pso: 165.

Orderings might also be compared in another way. The chief ingredient of this paper was the confrontation of their powers. The quintessence of this comparison is the fact that the extended recursive decomposition ordering proposed by Rusinowitch is one of the most powerful simplification orderings on terms.

Finally, let us remark a supplementary aspect not yet noted. A simple extension to some of the orderings has been developed: each operator f has a status which fixes the order of comparing the subterms of f ([KALE80] , [LE84] , [LE87]). This concept is useful for instance in proving termination of the associative law.

Bibliography

[DE85] N. Dershowitz : *Termination* , Proceedings of the first international conference on rewriting techniques and applications, Dijon, France, May 1985, pp. 180-224

[DE82] N. Dershowitz : *Orderings for term rewriting systems* , Journal of theoretical computer science, Vol. 17, No. 3 (March 1982), pp. 279-301

[DE79] N. Dershowitz : *A note on simplification orderings* , Information processing letters, Vol. 9, No. 5 (November 1979), pp. 212-215

[DEMA79] N. Dershowitz, Z. Manna : *Proving termination with multiset orderings* , Communications of the ACM, Vol. 22, No. 8 (August 1979), pp. 465-476

[HULA78] G. Huet, D. S. Lankford : *On the uniform halting problem for term rewriting systems* , Rapport Laboria 283, IRIA, Paris, Mars 1978, INRIA Rocquencourt, France

[HUOP80] G. Huet, D. Oppen : *Equations and rewrite rules : a survey* , Formal languages : perspectives and open problems (R. Book, ed.), Academic Press, New York, 1980, pp. 349-405

[KALE80] S. Kamin, J. J. Levy : *Attempts for generalizing the recursive path orderings* , Unpublished manuscript, 1980, Dept. of Computer Science, Univ. of Illinois, Urbana, Illinois , IL (1980)

[KANASI85] D. Kapur, P. Narendran, G. Sivakumar : *A path ordering for proving termination of term rewriting systems* , Proceedings of the tenth colloquium on trees in algebra and programming, 1985, pp. 173-187

[KNBE70] D. E. Knuth, P. B. Bendix : *Simple word problems in universal algebras* , In: Computational problems in abstract algebra (J. Leech, ed.), Pergamon Press (1970), pp. 263-297

[LA77] D. S. Lankford : *Some approaches to equality for computational logic : a survey and assessment* , Memo ATP-36, Automatic theorem proving project, University of Texas, TX, Spring 1977

[LE84] P. Lescanne : *Uniform termination of term rewriting systems - the recursive decomposition ordering with status* , Proceedings of the ninth Colloquium on trees in algebra and programming, B. Courcelle (ed.), Cambridge University Press, Bordeaux (France), 1984, pp. 182-194

[LE87] P. Lescanne : *On the recursive decomposition ordering with lexicographical status and other related orderings* , preprint, December 1987

[MA87] U. Martin : *How to choose the weights in the Knuth-Bendix ordering* , Proceedings of the second international conference on rewriting techniques and applications, Bordeaux, France, May 25 - 27, 1987, pp. 42-53

[PL78] D. A. Plaisted : *A recursively defined ordering for proving termination of term rewriting systems* , Report UIUCDCS-R-78-943, Department of computer science, University of Illinois, Urbana, IL, September 1978

[REJO81] F. Reinig, J. P. Jouannaud : *Decomposition orderings : a new family of recursive simplification orderings* , Centre de recherche en informatique de Nancy, France, June 1981, CRIN 81-R-040

[**RU85**] M. Rusinowitch : *Path of subterms ordering and recursive decomposition ordering revisited* , Proceedings of the first international conference on rewriting techniques and applications, Dijon, France, May 1985, pp. 225-240, also Journal of Symbolic Computation 3 (1 & 2), 1987, pp. 117-132

[**ST86**] J. Steinbach : *Ordnungen für Term-Ersetzungssysteme* , Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, Juni 1986

Summary

Termination is an important property of term rewriting systems. Simplification orderings are often used methods which guarantee termination. We describe the basic ideas of comparing terms and present the formal definitions and some examples of the most popular simplification orderings. A new definition of one of them is given which is simpler than the original and therefore better to handle. Furthermore, we complete the comparison found in the literature, i. e. we mark off the power (the sets of comparable terms) of the orderings.

Keywords

Termination, Simplification orderings, Recursive path ordering, Recursive decomposition ordering, Path of subterms ordering, Knuth-Bendix ordering