Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern 1, W. Germany

Artificial
Intelligence
Laboratories

SEKI-REPORT

Term Orderings With Status

Joachim Steinbach
SEKI Report SR-88-12

TERM ORDERINGS WITH STATUS

Joachim Steinbach

Universität Kaiserslautern
Fachbereich Informatik
Postfach 3049
D-6750 Kaiserslautern (FRG)

This research was supported by the Deutsche Forschungsgemeinschaft , SFB 314 (D4)

Abstract

The effective calculation with term rewriting systems presumes termination. Orderings on terms are able to guarantee termination. This report deals with some of those term orderings : Several path and decomposition orderings and the Knuth-Bendix ordering. We pursue three aims : Firstly, known orderings will get new definitions. In the second place, new ordering methods will be introduced : We will extend existing orderings by adding the principle of status ([KL80]). Thirdly, the comparison of the power as well as the time behaviour of all orderings will be presented.

More precisely, after some preliminary remarks to termination of rewrite systems we present the ordering methods. All orderings are connected by an essential characteristic : Each operator has a status that determines the order according to which the subterms are compared. We will present the following well-known orderings : The recursive path ordering with status ([KL80]), the path of subterms ordering ([Ru87]) and another path ordering with status ([KNS85]). A new recursive decomposition ordering with status will lead the catalogue of orderings introduced here. It is different from that of [Le84]. Moreover, we give a new definition based on decompositions of the path of subterms ordering (see [St88a]). An extension by incorporating status to this ordering as well as to the improved recursive decomposition ordering (cf. [Ru87]) will be a part of the paper. All orderings based on decompositions will be presented in a new and simple style : The decomposition of a term consists of terms only. The original definitions take tuples composed of three (or even four) components. Additionally to path and decomposition orderings, we deal with the weight oriented ordering ([KB70]) and incorporate status. Finally, important properties (simplification ordering, stability w.r.t. substitutions, etc.) of the newly introduced orderings will be listed.
Besides the introduction of new orderings, another main point of this report 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. It turned out that the new version with status of the improved recursive decomposition ordering (equivalent to the path ordering with status of [KNS85]) is the most powerful ordering of the class of path and decomposition orderings presented. This ordering and the Knuth-Bendix ordering with status overlap.
The orderings are implemented in our algebraic specification laboratory TRSPEC and the completion system COMTES. A series of experiments has been conducted to study the time behaviour of the orderings. An evaluation of these chronometries concludes the paper.

Keywords

Homeomorphic embedding, Incrementality, Knuth-Bendix ordering, Lexicographical ordering, Multiset ordering, Path of subterms ordering, Recursive decomposition ordering, Recursive path ordering, Simplification ordering, Status, Termination, Term rewriting system, Well-founded ordering

1 Introduction and Notations

Term rewriting systems gain more and more importance because they are a useful model for non-deterministic computations (since they are based on directed equations with no explicit control), with various applications in many areas of computer science and mathematics. Automatic theorem proving and program verification, abstract data type specifications and algebraic simplification, to name a few, have centred on this concept.

In order to emphasize definitions we will use italic type. A *term rewriting system* (*TRS*, for short) $\Re$ over a set of terms Γ is a finite or countably infinite set of rules, each of the form $l \rightarrow_{\Re} r$, where l and r are terms in Γ, such that every variable that occurs in r also occurs in l. The *set Γ of all terms* is constructed from elements of a set $\mathcal{F}$ of *operators* (or *function symbols*) and some denumerably infinite set $\mathcal{X}$ of *variables*. The set of *ground terms* (terms without variables) is denoted by Γ_G. The leading function symbol and the tuple of the (direct) arguments of a term t are referred to by *top*(t) and *args*(t), respectively.

A TRS $\Re$ generates a binary relation $\Rightarrow_{\Re}$ on Γ as follows : $s \Rightarrow_{\Re} t$ (term s *rewrites to* term t) if and only if s contains an instance $\sigma(l)$ of the left hand side of a rule $l \rightarrow_{\Re} r$ and t is derived from s by replacing the subterm $\sigma(l)$ by $\sigma(r)$. A *substitution* σ is defined as an endomorphism on Γ with the finite domain $\{x \mid \sigma(x) \neq x\}$, i.e. σ simultaneously replaces all variables of a term by terms. The structure of a term is partially altered by rule application. Consequently, it is advantegeous to have a precise scheme for specifying how and what particular part of it is to be changed. For this, we use the formalism of labelling terms 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 $O(t)$. $Ot(t)$ denotes the set of all *terminal occurrences* (occurrences of the leaves) of the term t. We write $t[u \leftarrow s]$ to denote the term that results from t by replacing t/u (the subterm of t at occurrence u) by s at the occurrence $u \in O(t)$. Furthermore, we write $s[t]$ to indicate that s contains the term t as a subterm. Analogous with $t[u \leftarrow s]$, $t[r \leftarrow s]$ stands for the term that results from t by replacing its subterm r at a fixed position by the term s.

A *derivation in $\Re$* is a sequence $t_0 \Rightarrow_{\Re} t_1 \Rightarrow_{\Re} t_2 \Rightarrow_{\Re} \ldots$. The simplicity of the semantic of a TRS is guaranteed whenever the result of such a computation does not depend on the choice of the rules to be applied. This property is called *confluence* and is related with the so-called *Church-Rosser* property that justifies the possible solution of the word problem by checking the equality of *normal*

forms (irreducible terms). If a TRS is not confluent, it can sometimes be transformed into a confluent one using the *Knuth-Bendix completion* procedure (cf. [KB70]) which adds new rules (non-convergent *critical pairs* that are derived from the *overlappings* of two left members of rules) to the initial rewrite system. Unfortunately, the successful use of this process crucially depends on the ability of proving the termination of a TRS.

A TRS $\mathfrak{R}$ over a set of terms Γ *is (finitely) terminating* or *noetherian* if there exists no infinite derivation in $\mathfrak{R}$. A trivial example of a terminating TRS is

$$\neg\neg x \rightarrow_{\mathfrak{R}} x$$
$$x \wedge x \rightarrow_{\mathfrak{R}} x$$

since the number of symbols is reduced by each application of a rule. But in general, the termination of an arbitrary TRS is an *undecidable* property, even if the number of rules is bounded by 2 ([HL78] , [De85]). Thus, the best we can hope for are different strategies which are together able to cope with many rewrite rule systems occurring in practice. These methods are based on verifying that the rewrite relation $\Rightarrow_{\mathfrak{R}}$ is included in an ordering on terms. Such an ordering must be well-founded to forbid infinite derivations of terms. A *(partial) ordering* on Γ_G is a transitive and irreflexive binary relation $>$ and it is called *well-founded* if there are no infinite descending chains. To check the inclusion '$\Rightarrow_{\mathfrak{R}} \subseteq >$' all (infinitely many) possible derivations must be tested. The key idea is to restrict this infinite test to a finite one. For that purpose we have to require a reduction ordering $>$.

A *reduction ordering* $>$ is a well-founded partial ordering and *compatible* (or sometimes called *monotonous*) with the structure of terms (the so-called *replacement property*), i.e. $t_1 > t_2$ implies $t[u \leftarrow t_1] > t[u \leftarrow t_2]$ for any $t, t_1, t_2 \in \Gamma$ and $u \in O(t)$. In other words, decreasing a subterm decreases any superterm containing it. An example of a reduction ordering is the ordering on the size of terms : $s \triangleright t$ if and only if $|s| > |t|$ where $|t|$ is the *size* of the term t (i.e. the number of function symbols and variables appearing in t) and $>$ is the natural ordering on integers. The well-foundedness of this ordering is a consequence of the ordering on integers being well-founded. Its compatibility is obvious.

The notion of reduction orderings suggests the following meaning of proving termination of rewrite systems :

> Theorem ([La77]) : A rewrite system $\mathfrak{R}$ *over* Γ *terminates* if and only if there exists a reduction ordering $>$ on Γ_G such that $\sigma(l) > \sigma(r)$ for each rule $l \rightarrow_{\mathfrak{R}} r$ and for any substitution σ of terms in Γ for the variables appearing in l.

The demand for a reduction ordering for the theorem is necessary since a reduction step $t[\sigma(l)] \Rightarrow_{\Re} t[\sigma(l) \leftarrow \sigma(r)]$ with an application of $l \rightarrow_{\Re} r$ must be in decreasing order, i.e. $t[\sigma(l)] > t[\sigma(l) \leftarrow \sigma(r)]$, and we only have required $l > r$. This representation reveals another dilemma which is universal quantification on substitutions or the so-called *stability w.r.t. substitutions* (cf. theorem) :

$$s > t \quad \text{implies} \quad \sigma(s) > \sigma(t).$$

Summarizing is to remark that a termination proof of a TRS requires a reduction ordering stabilized w.r.t. substitutions. Guaranteeing these properties is very difficult. This fact leads to the basic idea of characterizing classes of orderings for which there is no need to prove the conditions. One possible solution is represented by the class of simplification orderings ([De82]) which are at least reduction orderings. A partial ordering is a *simplification ordering* if it has two characteristics : The replacement property and the *subterm property* (any term is greater than any of its proper subterms). The requirement of the subterm property is evident since it is closely connected with the non-termination of rewrite systems.

It is obvious that a TRS $\Re$ is not terminating if the same term repeatedly appears in a derivation : $\ldots \Rightarrow_{\Re} s \Rightarrow_{\Re} \ldots \Rightarrow_{\Re} t \Rightarrow_{\Re} \ldots$ and $s = t$. Generally speaking, a TRS $\Re$ is non-terminating if a derivation contains two terms s and t where s is a subterm of t and t is derived from s. This property, called *looping*, provides a sufficient but not necessary condition for $\Re$ to be non-terminating. Consider the system consisting of the single rule $x*y \rightarrow_{\Re} (0+x)*y$ which produces the infinite derivation $x*y \Rightarrow_{\Re} (0+x)*y \Rightarrow_{\Re} (0+(0+x))*y \Rightarrow_{\Re} (0+(0+(0+x)))*y \Rightarrow_{\Re} \ldots$. Clearly, this TRS neither terminates nor loops. To detect the non-termination of this kind of rewrite system a weaker condition is needed. It is called *homeomorphic embedding* or plunging ordering. This is a transitive and reflexive binary relation $\trianglelefteq$ on terms. Embedding can be seen as a way to map injectively the set of symbols of a term s into the set of symbols of a term t, having regard to the topology (i.e. the structure) of s. But the mapping is not surjective in general : If it is surjective then s and t are identical. Therefore, we write $s \trianglelefteq t$ if s can be obtained from t by deletion of selected symbols :

We shall say that a derivation $t_1 \Rightarrow_{\Re} t_2 \Rightarrow_{\Re} \ldots$ is *self-embedding* if $t_i \trianglelefteq t_k$ for some $i < k$ (cf. [Pl85]). A rewrite system is self-embedding if it allows a self-embedding derivation. Note that non-termination allows a self-embedding derivation ([De82]). Moreover, self-embedding does not imply non-termination : The rewrite system $(x^2)^2 \rightarrow_{\Re} (-(x^2))^2$ is self-embedding and, nevertheless,

terminates. But we can use the homeomorphic embedding to specify a sufficient condition for the termination of rewrite systems : Simplification orderings. The close relationship of a simplification ordering $>$ to $\trianglelefteq$ is attested by the embedding lemma of Dershowitz :

Lemma ([De82]) : If $s \trianglelefteq t$, then $s \leq t$ in any simplification ordering $>$.

As usual, $\geq$ is a quasi-ordering on Γ_G. A *quasi-ordered set* $(\Gamma_G, \geq)$ consists of the set Γ_G and a transitive and reflexive binary relation $\geq$ defined on elements of Γ_G. A quasi-ordering defines an equivalence relation $=$ as both $\geq$ and $\leq$, and a partial ordering $>$ as $\geq$ but not $\leq$.

A great number of simplification orderings have been defined. Most of them are *precedence orderings* using a special ordering on operators. More precisely, a *precedence* is a partially ordered set $(\mathcal{F}, \triangleright)$ consisting of the set $\mathcal{F}$ of operators and an irreflexive and transitive binary relation $\triangleright$ defined on elements of $\mathcal{F}$. Therefore, we consider an ordering $>$ with p (the precedence) as a parameter, written as $>(p)$. If there is no ambiguity, we will use the notation $>$ instead of $>(p)$.

A partial ordering $>$ on a set M may be extended : An ordering $\blacktriangleright$ on M is an *extension* of $>$ if and only if $s > t$ implies $s \blacktriangleright t$ for all $s,t \in M$. We also say $>$ is *included* (or *contained*) in $\blacktriangleright$ and write $> \subseteq \blacktriangleright$. 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. If two elements s and t of M are incomparable, we will write $s \# t$. Let p, q be precedences. An ordering $>$ is said to be *monotonous w.r.t. the precedence* if $p \subseteq q$ implies $>(p) \subseteq >(q)$. In other words, if the precedence is increased the ordering becomes stronger.
An extension of a precedence can often be used to enable the comparison of two terms which cannot be compared with the original precedence. Thanks to the monotony (w.r.t. the precedence), the comparisons without the extension are still valid. This process of 'correctly' increasing a precedence is called *incrementality*. All the orderings described in this paper will be simplification orderings with this property.

Note that a partial ordering $>$ is used to compare 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 :

$(m_1,m_2,\ldots,m_p) >^{lex} (n_1,n_2,\ldots,n_q)$
if either $p > 0 \quad \wedge \quad q = 0$
or $m_1 > n_1$
or $m_1 = n_1 \quad \wedge \quad (m_2,\ldots,m_p) >^{lex} (n_2,\ldots,n_q)$.

If there is no order among the elements of such tuples then the structures over M^n are called multisets. *Multisets* 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

$$M_1 \gg M_2$$
$$\text{iff} \quad \text{i) } M_1 \neq M_2 \quad \wedge$$
$$\text{ii) } (\forall y \in M_2 \backslash M_1)\ (\exists x \in M_1 \backslash M_2)\ x > y$$

i.e. $M_1 \gg M_2$ 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 and, in particular, a more formal description of multisets and multiset orderings, see [DM79], [St86], [JL82], [MS86] and [Fe88].

The main components of the next two chapters are the definitions of well-known and new simplification orderings. All orderings are connected by an essential characteristic : Each operator $f \in \mathcal{F}$ has a *status* $\tau(f)$ that determines the order according to which the subterms of f are compared ([KL80]). Formally, status is a function which maps the set of operators into the set {mult , left , right} :

$$\tau : \mathcal{F} \rightarrow \{\text{mult , left , right}\}.$$

Therefore, a function symbol can have one of the following three statuses : *Mult* (the arguments will be compared as multisets), *left* (lexicographical comparison from left to right) and *right* (the arguments will lexicographically be compared from right to left). The result of an application of the function args to a term $t = f(t_1,...,t_n)$ depends on the status of f : If $\tau(f) = \text{mult}$, then args(t) is the multiset $\{t_1,...,t_n\}$ and otherwise, args(t) delivers the tuple $(t_1,...,t_n)$. Of course, it is possible and correct to generalize the lexicographical status by fixing any order among the immediate subterms. For simplicity only, we are satisfied with left, right and mult status. Obviously, if the precedence is a quasi-ordering, two equivalent symbols w.r.t. the precedence are supposed to have the same status. With this requirement ambiguities will be avoided.

Each definition of the orderings in chapter 2 (resp. 3) will be preceded by an abstract verbal description of its operational method (the technique of comparing terms). Furthermore, some helpful and interesting remarks are given. The list of orderings consists of the following well-known ones : The recursive path ordering with status (*RPOS*) of Kamin/Lévy and Dershowitz, the path of subterms ordering (*PSO*) of Rusinowitch and Plaisted, the path ordering with status (*KNSS*) of Kapur, Narendran and Sivakumar. A new recursive decomposition ordering with status (*RDOS*) will lead the catalogue of orderings introduced in this paper. It is different from that of [Le84].

[St88] contains an ordering on decompositions (called PSD) equivalent to the PSO. An extension by incorporating status (*PSDS*) will be presented here. Finally, we have also added the principle of status to the improved recursive decomposition ordering of Rusinowitch ([Ru87]) denoted by *IRDS*. Concluding, important properties (simplification ordering, stability w.r.t. substitutions, etc.) of the newly introduced orderings will be listed. The orderings based on decompositions (RDOS, PSDS, IRDS) will be presented in a new and simple style : The decomposition of a term consists of terms only. The original definitions take tuples composed of three (or even four) components.

In chapter 3, we deal with the weight oriented ordering of Knuth and Bendix and incorporate status (*KBOS*).

Besides the introduction of new orderings, 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 w.r.t. an underlying fixed total precedence and irrespective of the precedence.

An implementation of the orderings (except for the RDOS) is integrated into our rewrite rule laboratories *TRSPEC* (a term rewriting based system for algebraic specifications) and *COMTES* (completion of term rewriting systems). The TRSPEC-system (see [AGGMS87] , [ABGM86]) is implemented in Common Lisp and is currently running on Apollo Domain systems. It provides tools for specifying functions by term rewriting systems and for proving equational properties of these functions in the initial algebra. It allows to compile function specifications into executable lisp functions for computing normal forms of terms. The COMTES-system (see [Wa86] , [WS84] , [St86]) is part of the TRSPEC. It is a parametric completion system that is especially suited for efficiency experiments. Various reduction strategies and a couple of different term ordering methods can be used as well as different techniques for avoiding a failure of the completion process.

In the fifth chapter we enumerate a few aspects of the lisp code of the orderings. Only deviations from the original definitions will be described. A series of experiments has been conducted to study the time behaviour of the orderings. An evaluation of these results concludes the chapter.

The last chapter only contains the proofs of the previous lemmata.

2 Path orderings

All orderings described in this and the next section are recursively defined simplification orderings. Most of them are well-known : A version with status exists for the recursive path ordering (RPO), the recursive decomposition ordering (RDO, a new and more powerful version will be presented) and the path ordering of Kapur, Narendran and Sivakumar (KNS). We have added the principle of status to the others. The main point of the next two chapters is the description of all these orderings with status. For a better understanding, these methods of comparing terms will be demonstrated by an example at the end of the chapter. The orderings described satisfy properties that qualify them for proving termination of term rewriting systems : Well-foundedness, stability (w.r.t. substitutions) and monotony (w.r.t. the precedence). Due to lack of space, we only give proofs of properties which have not yet been proved.

To define the orderings, we need some kind of formalism. A *path* of a term is a sequence of terms starting with the whole term followed by a path of one of its arguments :

- $path_\varepsilon(\Delta) = \Delta$ if Δ is a constant symbol or a variable,
- $path_{i.u}(f(t_1,...,t_n)) = f(t_1,...,t_n) ; path_u(t_i)$ if $u \in Ot(t_i)$.

Moreover, $path(\{t_1,...,t_n\}) = \{ path_u(t_i) \mid i \in [1,n] , u \in Ot(t_i) \}$ is the multiset of all paths of the specified terms $t_1,...,t_n$. A path will be enclosed in square brackets. For a path $p = [t_1;t_2;...;t_n]$ we denote by $set(p)$ the set $\{t_1,...,t_n\}$ of all terms in p. This set will also be called path-decomposition and its abbreviation is $dec_u(t)$ (and is equal to $set(path_u(t))$). An element (i.e. a term) of a path-decomposition is called an *elementary decomposition*. Analogous with paths, the *decomposition* $dec(\{t_1,...,t_n\}) = \{ dec_u(t_i) \mid i \in [1,n] , u \in Ot(t_i) \}$ is the multiset of all path-decompositions of the terms $t_1,...,t_n$. There are two operations on a path-decomposition $P \subseteq \Gamma$ to describe. The set of subterms and the set of superterms of P relative to a term t are defined as

- $sub(P , t) = \{s \in P \mid (\exists u \neq \varepsilon)\ t/u = s\}$ and
- $sup(P , t) = \{s \in P \mid (\exists u \neq \varepsilon)\ s/u = t\}$.

Analogous with decompositions we use sub and sup to denote subsequences of paths. Suppose $t = (x * y) + (x * z)$, then $path_{21}(t) = [t ; path_1(x * z)] = [t ; x * z ; path_\varepsilon(x)] = [t ; x * z ; x]$, $path(\{t\}) = \{path_{11}(t) , path_{12}(t) , path_{21}(t) , path_{22}(t)\}$, $set(path_{21}(t)) = \{t , x*z , x\} = dec_{21}(t)$, $dec(\{t\}) = \{ \{t , x*y , x\} , \{t , x*y , y\} , \{t , x*z , x\} , \{t , x*z , z\} \}$, $sub(dec_{21}(t) , x) = \emptyset$, $sup(dec_{21}(t) , x*z) = \{t\}$ and $sup(path_{21}(t) , x) = [t ; x*z]$.

In the rest of this chapter, 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. Moreover, we synonymously use $>_{ord}$ with ord to denote an ordering. The index $\tau(f)$ of $>_{ord,\tau(f)}$ marks the extension of $>_{ord}$ w.r.t. the status of the operator f :

$$
\begin{array}{lllll}
(s_1,\ldots,s_m) >_{ord,\tau(f)} (t_1,\ldots,t_n) & & & & \\
\text{iff} \quad \tau(f) = \text{mult} & \wedge & \{s_1,\ldots,s_m\} & \gg_{ord} & \{t_1,\ldots,t_n\} \\
\text{or} \quad \tau(f) = \text{left} & \wedge & (s_1,\ldots,s_m) & >_{ord}^{lex} & (t_1,\ldots,t_n) \\
\text{or} \quad \tau(f) = \text{right} & \wedge & (s_m,\ldots,s_1) & >_{ord}^{lex} & (t_n,\ldots,t_1)
\end{array}
$$

Permitting variables, we have to consider each and every one of them as an additional constant symbol uncomparable (w.r.t. $\triangleright$) to all the other operators in $\mathcal{F}$.

All of the following orderings uniquely define a congruence $\sim$ dependent on $\mathcal{F}$ and τ via : $f(s_1,\ldots,s_m) \sim g(t_1,\ldots,t_n)$ iff $f = g$ and $m = n$ and i) $\tau(f) = \text{mult}$ and there is a permutation π of the set $\{1,\ldots,n\}$ such that $s_i \sim t_{\pi(i)}$, for all $i \in [1,n]$ or ii) $\tau(f) \neq \text{mult} \wedge s_i \sim t_i$, for all $i \in [1,n]$.

Most of the orderings are based on the principle of root orderings, i.e. two terms are compared depending on their leading function symbols. This or other kinds of case distinctions will be represented as the union of conditions that will be marked by Roman numerals i), ii), and so on. The lexicographical performance of conditions will be indicated by hyphens, i.e.

$$
\begin{array}{l}
s > t \\
\text{iff} \quad - \; s >_1 t \\
\qquad - \; s >_2 t
\end{array}
$$

stands for that $s > t$ iff $s >_1 t$ or $(s =_1 t \wedge s >_2 t)$. Here, the equality sign $=_1$ is the congruence relation induced by the quasi-ordering $\geq_1$.

The comparison with respect to the recursive path ordering with status (RPOS, for short) is based on the following idea : A term is decreased by replacing a subterm with any number of smaller terms which are connected by any structure of operators smaller (w.r.t. $\triangleright$) than the leading function symbol of the replaced subterm. The method of comparing two terms depends on the leading function symbols. The relationship between these operators w.r.t. $\triangleright$ and the status τ is responsible for decreasing one of the (or both) terms in the recursive definition of the RPOS. If one of the terms is 'empty' (i.e. totally decreased) then the other one is greater.

2.1 Definition ([KL80] , [De82])
recursive path ordering with status : **RPOS**

$s >_{RPOS} t$
iff i) $top(s) \triangleright top(t) \wedge \{s\} \gg_{RPOS} args(t)$
ii) $top(s) = top(t) \wedge \tau(top(s)) = mult \wedge args(s) \gg_{RPOS} args(t)$
iii) $top(s) = top(t) \wedge \tau(top(s)) \neq mult \wedge args(s) >_{RPOS,\tau(top(s))} args(t)$
$\wedge \{s\} \gg_{RPOS} args(t)$
iv) $args(s) \geq_{RPOS} \{t\}$

2.2 Remarks

- The relation $s \geq_{RPOS} t$ is valid if $s >_{RPOS} t$ or $s \sim t$ (see definition on page 8).
- The definition of the RPOS is non-deterministic because condition iv) could be used whenever either i) or ii) or iii) is satisfied.
- The multiset ordering and the simple path ordering ([Pl78b]) are special cases of the RPO, in which the multiset constructor { ... } is greater than any of the other operators ([De85] , [De82]).
- One way of extending the RPO is to allow some component of a term $f(t_1,...,t_n)$ to serve as the operator f. For example, we can consider t_k to be the operator and compare two terms by first recursively comparing their k-th operands : To prove that $s = if(if(x,y,z),u,v) \rightarrow if(x,if(y,u,v),if(z,u,v)) = t$ terminates we consider the condition (the term if(x,y,z)) to be the function symbol. The term if(x,y,z) of s is greater than the condition x of t. By definition of the RPO, we need to show that $\{s\} \gg_{RPO} \{if(y,u,v) , if(z,u,v)\}$. Again, if(x,y,z) is greater than y and z , and thus $\{s\} \gg_{RPO} \{u , v\}$ holds.
This kind of extension of the RPO represents a simplification ordering for the same reasons that the original definition does ([De85] , [De82]).
- A more general technique than the previous one of extending the RPO changes terms by replacing their operators with the whole term itself ordered by some <u>other</u> well-founded ordering (instead of the RPO itself). This method is called *semantic path ordering* and was developed by Kamin and Lévy ([KL80] , [De85] , [De83]).
- Another extension of this ordering (from Forgaard, [Fo84]) takes the monotony w.r.t the precedence of the RPO into consideration. A term s will only be greater than a term t in this 'lifted' ordering if s is greater (relative to the RPO) than t with respect to all total extensions of the given precedence. The so-called *closure ordering* in [Le87] is a generalization of this method because it lifts <u>any</u> term ordering. Note that the lifted ordering and its corresponding ordering will be equivalent if the underlying precedence $\triangleright$ is total. This statement holds since there is no other extension of a total $\triangleright$ than $\triangleright$ itself.

- An algorithm that automatically proves the termination of rewrite rules is described in [DF85]. The incrementality procedure is implemented in REVE 2, a rewrite rule based theorem prover. The strategy is based on the notion of 'minimal extenders', i.e. if the RPO cannot order its arguments, it will return sets of minimal suggestions that cause them to be comparable (see [Fo84] , [DF85]).
- Given the equation $s = t$, the problem whether there exists a partial order $\triangleright$ on the operators that occur in s and t such that the equation can be oriented by the RPO is NP-complete ([KN85]).
- The RPO has also been adapted to handling associative-commutative operators by flattening and transforming terms (distributing large operators over small ones) before comparing them (see [De85] , [De83]).
- For more details about the RPOS (respectively the RPO), see [Ai85] , [De85] , [De83] , [De80] , [DF85] , [KNS85] , [KN85] , [Le86] , [Le81a] , [Ok86] , [Pe81] , [Ru87] , [St88] , [St86].

Plaisted's path of subterms ordering (PSO, for short) is a predecessor of the RPO and compares two terms by comparing all their paths. A slightly modified version (equivalent to the original) of Rusinowitch is given next.

2.3 Definition ([Ru87] , [Pl78a] , [St88])
path of subterms ordering : **PSO**

$s >_{PSO} t$
iff $path(\{s\}) \gg_{PO} path(\{t\})$
with $p >_{PO} q$
iff $set(p) \gg_{T} set(q)$
with $s >_{T} t$
iff $- top(s) \triangleright top(t)$
$- path(args(s)) \gg_{PO} path(args(t))$

2.4 Remarks

- Terms are equivalent w.r.t the PSO if they are contained in the relation $\sim$ (see definition on page 8). This PSO-equivalence is a special case of $\sim$ since the PSO does not compare terms with lexicographical status. Therefore, s and t are equivalent if they are permutatively congruent which means that s and t are syntactically equal if the permutations of permutatively congruent subterms are ignored. Two zero-ary symbols (constants or variables) are permutatively congruent if they are syntactically equal.

- More information about the PSO is available in [Pl78a] , [Pl78b] , [Ru87] , [St88] , [St86].

Like the PSO, the KNSS is an ordering which compares terms using their paths. 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 KNSS is a consequence of these experiments and it extends the RPOS.

2.5 Definition ([KNS85])
path ordering with status of Kapur, Narendran and Sivakumar : **KNSS**

$s >_{KNSS} t$
iff $path(\{s\}) \gg_{LK} path(\{t\})$
with $p >_{LK} q$
iff $(\forall t' \in q)\ (\exists s' \in p)\ \ s' >_{LT} t'$
with $p \ni s >_{LT} t \in q$
iff i) $top(s) \triangleright top(t)$
ii) $top(s) = top(t) \quad \wedge \quad \tau(top(s)) = mult \quad \wedge$
- $sub(p , s) >_{LK} sub(q , t)$
- $path(args(s)) \gg_{LK} path(args(t))$
- $sup(p , s) >_{LK} sup(q , t)$

iii) $top(s) = top(t) \quad \wedge \quad \tau(top(s)) \neq mult \quad \wedge$
- $args(s) >_{KNSS,\tau(top(s))} args(t) \quad \wedge$
$\{s\} \gg_{KNSS} args(t)$
- $sup(p , s) >_{LK} sup(q , t)$

2.6 Remarks

- Like the previous orderings, the equivalence w.r.t the KNSS is included in $\sim$ (cf. definition on page 8).
- The third part of $>_{LT}$ consists of a lexicographical test divided into two conditions. We compromised in favour of the clearness and suffer the ambiguity of the definition. Therefore, note that the second condition '$sup(p , s) >_{LK} sup(q , t)$' will be tested if only $args(s) =_{KNSS} args(t)$ holds (the other test '$\{s\} =_{KNSS} args(t)$' is redundant).
- This path ordering can possibly be extended by way of allowing terms from different paths to take care of terms along a path. Thus we could remove the restriction that a path must be taken care of by a sole path. Let s and t be the following terms : $s = 2 * 8$ and $t = 4^2$

with $2 \triangleright 4, 8 \triangleright 2$. The term 8 of the path [s ; 8] takes care of the term t of [t ; 4] and $2 \in [s ; 2]$ is greater than $4 \in [t ; 4]$. Note that this new scheme has not been clearly defined yet ([KNS85]).

- More information about the KNS(S) are given in [KNS85] , [Ru87] and [St86].

Like the KNSS, the recursive decomposition ordering with status (RDOS in short) has been developed from the RPO. One of the important differences to the RPO is the fact that the RDOS stops a comparison as soon as it has to compare incomparable operators. The RDOS defined in 2.7 is different from that contained in [Le84] : Both orderings are based on [RJ81] but the incorporations of status are different. Moreover, our various decomposition orderings (see 2.7, 2.9 and 2.11) are founded on another decomposition : We use terms (cf. the definitions on page 7) instead of triples or even quadruplets (see 2.8, at the bottom of this page).
A term s is greater than a term t (w.r.t. RDOS) if the decomposition of s is greater than the decomposition of t. The ordering on these multisets ($\gg\gg_{LD}$) is an extension of the basic ordering on terms ($>_{LD}$) to multisets of multisets.

2.7 Definition (based on [RJ81] , [JLR82] , [Ru87])
recursive decomposition ordering with status : **RDOS**

$s >_{RDOS} t$
iff $dec(\{s\}) \gg\gg_{LD} dec(\{t\})$
with $dec_u(s) \ni s' >_{LD} t' \in dec_v(t)$
iff i) $top(s') \triangleright top(t')$
ii) $top(s') = top(t') \quad \wedge \quad \tau(top(s')) = mult \quad \wedge$
- $sub(dec_u(s) , s') \gg_{LD} sub(dec_v(t) , t')$
- $args(s') \gg_{RDOS} args(t')$

iii) $top(s') = top(t') \quad \wedge \quad \tau(top(s')) \neq mult \quad \wedge$
$args(s') >_{RDOS,\tau(top(s'))} args(t') \quad \wedge \quad \{s'\} \gg_{RDOS} args(t')$

2.8 Remarks

- Terms are equal ($s =_{RDOS} t$) w.r.t. the RDOS if they are equivalent w.r.t. $>_{RDOS}$ ($s \sim t$; $\sim$ is defined on page 8).
- The original decomposition ordering works on triples (so-called *elementary decompositions*) instead of terms. An elementary decomposition divides a term into three parts : The leading function symbol, any selected immediate subterm, the rest of the immediate subterms. Obviously, an elementary decomposition can easily be constructed if the appropriate term is

given. Therefore, the information of an elementary decomposition can be put forward during the definition of the ordering.

- An essential advantage of the RDOS is the possibility of strengthening the precedence during the process of comparing. This method (without regard to status) is implemented in Lisp on the Multics of INRIA and in Clu inside the REVE-system ([Ch84] , [JLR82]).
- A more efficient RDO without a loss of power is presented in [JLR82] and [RJ81]. Originally, two terms s and t are compared w.r.t. the RDO by means of constructing the entire dec({s}) and the entire dec({t}). It can be proved that selected subsets of path-decompositions are sufficient. Moreover, a path-decomposition can be restricted to a part of itself (cf. [St86] for an example).
- Additional remarks are given in [Ch84] , [De85] , [De83] , [JLR82] , [KNS85] , [Le87] , [Le84] , [Le83a] , [Le83b] , [Le82] , [Le81b] , [Re81] , [St88] , [St86].

Another ordering based on decompositions results from the PSO. It is remarkable that the PSO is an extremely recursive ordering which takes three suborderings ($>_{PO}$, $>_T$ and $\triangleright$) into account. We have succeeded in redefining this path ordering in such a way that the result, called PSD, provides a much simpler method of using decompositions (cf. [St88]). The PSD has another advantage over the PSO : The combination with the concept of status is much easier. The PSD with status (PSDS, for short) as well as the PSD depends on the fact that a path is an ordered path-decomposition.

2.9 Definition (based on [St88] , [Ru87] , [Pl78a])
path of subterms ordering on decompositions and with status : **PSDS**

$s >_{PSDS} t$
iff $dec(\{s\}) \gg\gg_{LP} dec(\{t\})$
with $s >_{LP} t$
iff i) $top(s) \triangleright top(t)$
ii) $top(s) = top(t) \wedge \tau(top(s)) = mult \wedge dec(args(s)) \gg\gg_{LP} dec(args(t))$
iii) $top(s) = top(t) \wedge \tau(top(s)) \neq mult \wedge$
$args(s) >_{PSDS,\tau(top(s))} args(t) \wedge \{s\} \gg_{PSDS} args(t)$

2.10 Remarks

- $s =_{PSDS} t$ if and only if $s \sim t$ (cf. definition of $\sim$ on page 8).
- This relatively simple definition of the complicated PSO has two further advantages. In the first place, we can compare the implementations according to their efficiency (see chapter 5, on page

29). Secondly, it is easy to compare the PSO with the decomposition orderings, e.g. with the improved recursive decomposition ordering of Rusinowitch (so-called IRD, see 2.11). The essential difference between the PSO (= PSD) and the IRD concerns the way by which a comparison is processed. While the PSD works according to the principle of 'breadth-first' the IRD reveals the use of the principle of 'depth-first' : If the leading function symbols of the terms to compare are identical, the IRD chooses only one subterm. On the other hand, the PSD proceeds by simultaneously considering the decomposition multiset of all subterms.

In addition to the definition of Rusinowitch, we have incorporated status to the IRD (IRDS in short), so that it is equivalent to the path ordering of Kapur, Narendran and Sivakumar (IRDS = KNSS, see lemma 4.6 on page 25).

2.11 Definition (based on [Ru87])
improved recursive decomposition ordering with status : **IRDS**

$s >_{IRDS} t$
iff $dec(\{s\}) \gg\gg_{EL} dec(\{t\})$
with $dec_u(s) \ni s' >_{EL} t' \in dec_v(t)$
iff i) $top(s') \triangleright top(t')$
ii) $top(s') = top(t') \wedge \tau(top(s')) = mult \wedge$
- $sub(dec_u(s), s') \gg_{EL} sub(dec_v(t), t')$
- $dec(args(s')) \gg\gg_{EL} dec(args(t'))$

iii) $top(s') = top(t') \wedge \tau(top(s')) \neq mult \wedge$
$args(s') >_{IRDS,\tau(top(s'))} args(t') \wedge \{s'\} \gg_{IRDS} args(t')$

2.12 Remarks

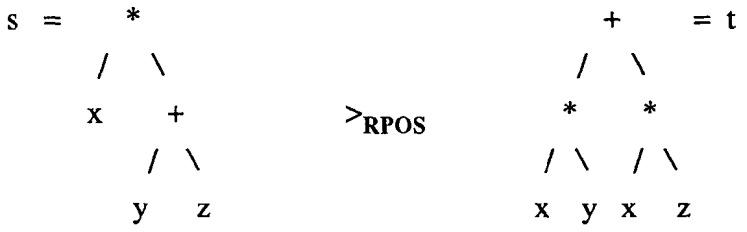
- The relation $s \geq_{IRDS} t$ is true if and only if $s >_{IRDS} t$ or $s \sim t$ ($\sim$ is defined on page 8).
- The IRDS is a proper extension of the RDOS, due to a slight change of the second part (ii) of its definition : RDOS compares the arguments of two terms w.r.t. $>_{RDOS}$ while IRDS compares the multiset sums of the decompositions of the arguments. For example, $and(not(not(x)),y,not(z)) >_{IRDS} and(y,nand(x,z),x)$ with $not \triangleright nand$, but the terms are not comparable with the RDOS (see [St88]).

We will illustrate the definitions of the orderings by an example.

2.13 Example

We want to prove that the distributive law $x * (y+z) \rightarrow (x*y) + (x*z)$ terminates. We use the total precedence $* \triangleright +$ and the statuses $\tau(+) = \tau(*) = \text{left}$.

2.13.1



Since $* \triangleright +$, we must show that $\{s\} \gg_{RPOS} \text{args}(t)$

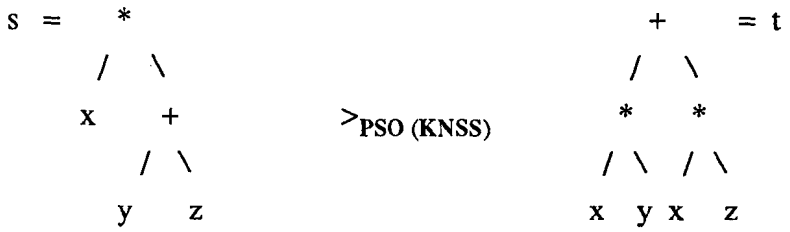
.

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

$$(x, y+z) >_{RPOS,left} (x, y) \quad \wedge \quad \{s\} \gg_{RPOS} \{x, y\}$$

because $y + z >_{RPOS} y$ (by using the fourth condition of 2.1) and $s >_{RPOS} x$, $s >_{RPOS} y$. $s >_{RPOS} x * z$ is proved in the same way.

2.13.2



We have to show that $\text{path}(\{s\}) \gg_{PO\,(LK)} \text{path}(\{t\})$:

$\text{path}(\{s\}) = \{ [s ; x] , [s ; y+z ; y] , [s ; y+z ; z] \}$ and

$\text{path}(\{t\}) = \{ [t ; x*y ; x] , [t ; x*y ; y] , [t ; x*z ; x] , [t ; x*z ; z] \}$.

In accordance with the extension of $>_{PO\,(LK)}$ to multisets the following has to be proved : For every path in t we can find a path in s which is greater w.r.t. $>_{PO\,(LK)}$.

- i) $[s\,;x] >_{PO} [t\,;x*y\,;x]$
  iff $\{s\,,x\} \gg_T \{t\,,x*y\,,x\}$ iff $\{s\} \gg_T \{t\,,x*y\}$
  (the last equivalence is valid since the used multiset ordering is closed under differences, i.e. $M \gg N$ iff $M \backslash N \gg N \backslash M$)

  - $s >_T t$
    because $top(s) = * \triangleright + = top(t)$

  - $s >_T x*y$
    iff $path(\{x\,,y+z\}) \gg_{PO} path(\{x\,,y\})$ iff $path(\{y+z\}) \gg_{PO} path(\{y\})$:
    This can be verified because y is a proper subterm of y+z.

  ii) $[s\,;x] >_{PO} [t\,;x*z\,;x]$,
  $[s\,;y+z\,;y] >_{PO} [t\,;x*y\,;y]$,
  $[s\,;y+z\,;z] >_{PO} [t\,;x*z\,;z]$:
  The proofs of these three statements can easily be done with the considerations of i).

- i) $[s\,;x] >_{LK} [t\,;x*y\,;x]$:

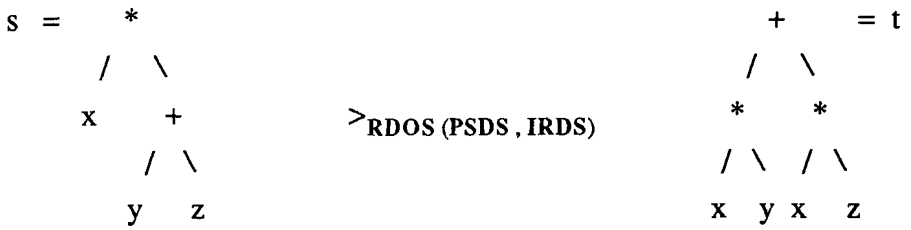
  - $s >_{LT} t$
    because $top(s) = * \triangleright + = top(t)$

  - $s >_{LT} x*y$
    iff $(x\,,y+z) >_{KNSS,left} (x\,,y) \wedge \{s\} \gg_{KNSS} \{x\,,y\}$:
    This can be verified because y is a proper subterm of y+z, x and y are proper subterms of s.

  - $[s\,;x] \ni x >_{LT} x \in [t\,;x*y\,;x]$
    ➞ $sub([s\,;x]\,,x) =[\,]= sub([t\,;x*y\,;x]\,,x) \wedge path(args(x \in s)) = \emptyset = path(args(x \in t))$
    ➞ We have to show that $sup([s\,;x]\,,x) = [s] >_{LK} [t\,;x*y] = sup([t\,;x*y\,;x]\,,x)$:
    As $s >_{LT} t$ and $s >_{LT} x*y$, the requirement holds.

  ii) $[s\,;x] >_{LK} [t\,;x*z\,;x]$,
  $[s\,;y+z\,;y] >_{LK} [t\,;x*y\,;y]$,
  $[s\,;y+z\,;z] >_{LK} [t\,;x*z\,;z]$:
  This propositions are valid if we use the steps of the previous case.

2.13.3



We have to prove $dec(\{s\}) \gg_{LD\,(LP\,,\,EL)} dec(\{t\})$:

$dec(\{s\}) = \{dec_1(s)\,,\, dec_{21}(s)\,,\, dec_{22}(s)\}$,

$dec(\{t\}) = \{dec_{11}(t)\,,\, dec_{12}(t)\,,\, dec_{21}(t)\,,\, dec_{22}(t)\}$.

In accordance with the definition of the RDOS (PSDS , IRDS), for every $dec_v(t)$ we have to find a $dec_u(s)$ which is greater than $dec_v(t)$ with respect to $\gg_{LD\,(LP\,,\,EL)}$. Because of the demand for stability w.r.t. substitutions our search for $dec_u(s)$ is restricted to the leaves s/u and t/v, respectively (it is important to observe this rule if t/v is a variable, i.e. s/u must be the same variable).
We can verify

i) $dec_1(s) = \{s, x\} \gg_{LD\,(LP\,,\,EL)} \{t\,,\, x*y\,,\, x\} = dec_{11}(t)$ iff $\{s\} \gg_{LD\,(LP\,,\,EL)} \{t\,,\, x*y\}$:

- $s >_{LD\,(LP\,,\,EL)} t$
  because $top(s) = * \triangleright + = top(t)$

- $s >_{LD\,(LP\,,\,EL)} x*y$
  iff $(x\,,\, y+z) >_{RDOS\,(PSDS\,,\,IRDS),left} (x\,,\, y)$ $\wedge$ $\{s\} \gg_{RDOS\,(PSDS\,,\,IRDS)} \{x, y\}$:
  This can be verified since y is a proper subterm of y+z, x and y are proper subterms of s.

ii) $dec_1(s) = \{s\,,\, x\} \gg_{LD\,(LP\,,\,EL)} \{t\,,\, x*z\,,\, x\} = dec_{21}(t)$,
$dec_{21}(s) = \{s\,,\, y+z\,,\, y\} \gg_{LD\,(LP\,,\,EL)} \{t\,,\, x*y\,,\, y\} = dec_{12}(t)$,
$dec_{22}(s) = \{s\,,\, y+z\,,\, z\} \gg_{LD\,(LP\,,\,EL)} \{t\,,\, x*z\,,\, z\} = dec_{22}(t)$:
It is easy to show these statements with the considerations of i).

For more examples, see [Ch84] (RDO), [De85] (RPO), [De83] (RPO, RDO), [De82] (RPO), [JLR82] (RDO), [KL80] (RPOS), [KNS85] (KNS), [Le87] (lexicographic RDO), [Le84] (RDO), [Le83a] (RDO), [Re81] (RDO), [RJ81] (RDO), [Ru87] (PSO, RDO, IRD), [St88] (RPO, RDO, IRD), [St86] (RPO, lexicographical RPO, RDO, IRD, KNS, PSO).

We continue with the enumeration of important properties of the IRDS (see chapter 6 for their proofs). To guarantee termination of rewrite systems, the IRDS must be a simplification ordering (irreflexivity, transitivity, subterm property and replacement property) and stable w.r.t. substitutions.

2.14 Lemma IRDS is a simplification ordering on Γ_G.

Proof : see 6.1 - 6.4 (pp. 33 - 35)

2.15 Lemma IRDS is stable w.r.t. substitutions.

Proof : see 6.5 (pp. 35 - 36)

Dershowitz gives a sufficient criterion for proving that a TRS terminates for every input ([De82]) : A TRS $\Re = \{l_i \rightarrow_{\Re} r_i \mid i \in [1,n]\}$ over a set of terms Γ terminates if there exists a simplification ordering $>$ over Γ such that $\sigma(l_i) > \sigma(r_i)$ $(i \in [1,n])$ for any substitution σ of terms of Γ for the variables of l_i. The test whether $\sigma(l_i) > \sigma(r_i)$ is true for all substitutions does not stop. Therefore, if we are able to prove the stability w.r.t. substitutions of the prescribed simplification ordering $>$, then showing $l_i > r_i$ instead of $\sigma(l_i) > \sigma(r_i)$ will be enough. Since the improved decomposition ordering with status is a simplification ordering (lemma 2.14) and closed under instanciation (lemma 2.15), it may be used in conjunction with Dershowitz's termination theorem to prove the termination of a TRS in a relatively simple manner.

The main importance of simplification orderings is their well-foundedness. As well as the IRDS the other orderings described here are well-founded, too. Instead of proving that they are simplification orderings it is sufficient to show that they are partial orderings and monotonic w.r.t. the precedence. This reasoning will be justified by the following lemma and the fact that the IRDS is stronger than all other orderings w.r.t. total precedences (see chapter 4).

2.16 Lemma A partial and monotonous (w.r.t. the precedence) ordering $>$ is well-founded if there exists a well-founded ordering $\triangleright$ which contains $>$ w.r.t. total precedences.

Proof : see 6.6 (on page 36)

To use this lemma we take the IRDS as $\triangleright$ and show (in chapter 4) the inclusion of $>$ in $>_{IRDS}$. Consequently, there only remains to prove that $>$ is a partial ordering and monotonous w.r.t. the precedence. The proof of the irreflexivity and the transitivity of the RPOS is formulated in [KL80].

In [Pl78b], Plaisted shows that the PSO is a partial ordering. Analogous with the proof of the IRDS, this property can be certified to the remaining orderings. Thus, all we need is the guarantee of the monotony w.r.t. the precedence.

2.17 Lemma RPOS is monotonous w.r.t. the precedence.

Proof : see 6.7 (pp. 36 - 37)

2.18 Lemma RDOS is monotonous w.r.t. the precedence.

Proof : see 6.8 (on page 37)

2.19 Lemma PSO, PSDS and IRDS (KNSS) are monotonous w.r.t. the precedence.

Proof : The proof of the monotony of the PSO is given in [St86]. The proofs of the remaining orderings PSDS and IRDS (= KNSS) are very similar to that of 2.18 . ❑

The applicability of the orderings for termination proofs of term rewriting systems is ensured only if, additionally to the properties already proved, the stability w.r.t. substitutions is fulfilled. The following affirmations serve this purpose.

2.20 Lemma RPOS is stable w.r.t. substitutions.

Proof : see 6.9 (on page 38)

2.21 Lemma RDO, RDOS, PSD and PSDS are stable w.r.t. substitutions.

Proof : The proofs of the stability w.r.t. substitutions of the decomposition orderings (RDO, RDOS, PSD and PSDS) are very similar to that of the IRDS (see 2.15). Therefore, and due to lack of space, we renounce to show the property for these orderings. ❑

Since additionally the PSO is equivalent to the PSD, and the IRDS and the KNSS coincide (cf. chapter 4), all orderings presented are stable w.r.t. substitutions.
Collecting all the previous lemmata, we may conclude this chapter with the remarkable result: All orderings described in this chapter are well-founded and stable w.r.t. substitutions and therefore, they are methods usable in practice for proving the termination of an arbitrary TRS.

3 Knuth-Bendix ordering with status

To prove termination of term rewriting systems we can use the notion of a *well-founded set* $(S, >_S)$ which is a set S and a partial ordering $>_S$ on S such that any decreasing sequence $e_1 >_S e_2 >_S \ldots$ of elements of S only consists of a finite number of elements. To construct an ordering we choose a well-founded set $(S, >_S)$ and a so-called *termination function* which maps the term algebra into S. S can be the term algebra itself : The path and decomposition orderings of the previous chapter are based on this notion. The ordering of Knuth and Bendix (KBO, for short) takes $(\mathbb{N}, >)$ as the underlying well-founded set, i.e. it assigns 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 lexicographically comparing the subterms. Analogous with the path and decomposition orderings, we succeeded in adding the idea of status and therefore, in extending the method of comparing the arguments of two equivalent function symbols. To describe this strategy, called Knuth-Bendix ordering with status (KBOS, for short), we need some prerequisites and helpful definitions.

If Δ is a function symbol or a variable and t is a term we denote the number of occurrences of Δ in t by $\#_\Delta(t)$. 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

$$\varphi(c) \geq \varphi_0 \qquad \text{if c is a constant and}$$
$$\varphi(f) > 0 \qquad \text{if f has one argument.}$$

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

$$\varphi(t) = \varphi(f) + \sum \varphi(t_i).$$

3.1 Definition (based on [Ma87] , [KB70])
Knuth-Bendix ordering with status : **KBOS**

$s >_{KBOS} t$
iff $(\forall x \in \mathcal{X})\ \#_x(s) \geq \#_x(t) \quad \wedge$
- $\varphi(s) > \varphi(t)$
- $top(s) \triangleright top(t)$
- $args(s) >_{KBOS,\tau(top(s))} args(t)$

3.2 Remarks

- The definition is slightly different from that given in [KB70]. Knuth and Bendix require that the precedence be total and that $\#_x(s) = \#_x(t)$ if $\varphi(s) = \varphi(t)$. The version given here is from [Ma87] and is an extension of the original definition. This claim can be substantiated with the help of the rule $(y \sqsupset x) \vee x \rightarrow (y \vee 0) \sqsupset x$ which can only be oriented with the new definition $(\#_x(s) \geq \#_x(t))$ if $\vee \triangleright \sqsupset$ is presumed.
- The variable condition $(\#_x(s) \geq \#_x(t))$ guaranteeing the stability certainly is a very strong restriction. Note that, for example, the distributive law cannot be oriented in the usual direction.
- There exists a slight improvement of the ordering that allows at most one unary operator f with weight zero ([KB70]). To conserve the well-foundedness all other operators in $\mathcal{F}$ have to be smaller than f (with respect to the precedence).
- Using a quasi-ordering on the function symbols instead of a partial ordering, we may admit more than one unary operator with weight zero. On the premise that all these operators are equal w.r.t. the precedence, the induced KBOS also is a simplification ordering stabilized w.r.t. substitutions. For example, the termination of $(x*y)^2 \rightarrow x^2*(-y)^2$ can be shown with the improved version but cannot be shown with the partial version of the KBOS.
- [Ma87] contains a simple and practical decision procedure for determining whether or not a set of rules can be ordered by a KBO. The basic idea of this algorithm is to transform the desired rules to linear inequalities which the weights must satisfy. The solutions to these inequalities are determined by using the simplex method ([Ma87]).
- Note that the terms of the following example are not comparable with the KBO (without status).

3.3 Example

Consider the terms $s = x * ((-y) * y)$ and $t = (- y * y) * x$ and the following functions on the operators :

symbol	x,y	-	*
φ	1	1	0

$\tau(*) = \text{mult}$

$* \triangleright -$

We want to prove that $s >_{KBOS} t$. Since $\varphi(s) = \varphi(*) + \varphi(x) + \varphi((-y)*y) = 0 + 1 + \varphi(*) + \varphi(-y) + \varphi(y) = 0 + 1 + 0 + \varphi(-) + \varphi(y) + 1 = 4$ and $\varphi(t)$ are equal and $top(s) = top(t) = *$ we have to apply the KBOS recursively on the multisets of arguments and have to verify $\{x , (-y)*y\} \gg_{KBOS} \{-y*y , x\}$. This is true because $s' = (-y)*y >_{KBOS} -y*y = t'$, since $\varphi(s') = \varphi(t')$ and $top(s') = * \triangleright - = top(t')$.

We want to use this new version of the well-known KBO as an ordering to prove the termination of rewrite rules. Therefore, we conclude this section with the reference to some important properties. We show that the KBOS is a simplification ordering (a partial ordering with the subterm and the replacement property) and that it is stable w.r.t. substitutions.

3.4 Lemma KBOS is a simplification ordering on T_G.

Proof : see 6.10 - 6.13 (pp. 39 - 41)

3.5 Lemma KBOS is stable w.r.t. substitutions.

Proof : see 6.14 - 6.15 (pp. 41 - 42)

4 Comparison

In this chapter we compare the power of the presented orderings. For completeness, we are additionally including the basic orderings restricted to multiset status. The power of an ordering is represented by 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 can be *equivalent* ($>=\triangleright$), one ordering can be *properly included* in the other ($>\subset\triangleright$) or they *overlap* ($>\#\triangleright$). The orderings $>$ and $\triangleright$ overlap if there exist some terms such that $s_1 > t_1 \wedge s_1 \not\triangleright t_1$ and $s_2 \triangleright t_2 \wedge s_2 \not> t_2$. Consequently, the proof of such an overlapping is composed by specifying two counter-examples. At the end of this chapter a synopsis of the previous lemmata will be listed in the form of a diagram.

Note that the orderings described in this report depend on a parameter : The precedence. This parameter may be either partial or total. For the following reason we are only interested in the latter : Our results would be more general if we could give some information about the comparisons of orderings separated from the precedence. The following proposition summarizes these reflections.

4.1 Proposition Let $>$ and $\triangleright$ be orderings which are monotonous w.r.t. the precedence. Furthermore, let $>$ be included in $\triangleright$ w.r.t. total precedences. Then, if $s >(p) t$ holds for some precedence p, there exists a precedence q (possibly different from p) so that $s \triangleright(q) t$.

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

This statement will be of practical importance if we consider it together with the relations between orderings with an underlying total precedence (see figure 4.12 on page 27) : Only two (either IRDS (KNSS) or KBOS) of the thirteen orderings collected in the diagram are needed to cover the union of comparable terms of all the orderings presented here. In other words, if terms can be oriented with any ordering (of the figure) there exists a precedence such that the terms are also comparable with either the IRDS (KNSS) or the KBOS. Consequently, if you are implementing a system where the termination of a rewriting system must be guaranteed, only two of the thirteen orderings will have to be made available for the user. The cause of it is that the IRDS (resp. the KNSS) is stronger than all other path orderings irrespective of the precedence.

The relations $\subset$, = and # w.r.t. a total precedence have the following meanings : Let p (resp. p' and p") be a total precedence, τ (resp. τ' and τ") a status, s and t terms.

i) $>\subset\triangleright$ iff $s >(p,\tau)\ t \rightarrow s \triangleright(p,\tau)\ t \wedge (\exists p',\tau')(\nexists p'',\tau'')\ s \triangleright(p',\tau')\ t \wedge s >(p'',\tau'')\ t$

ii) $>=\triangleright$ iff $s >(p,\tau)\ t \leftrightarrow s \triangleright(p,\tau)\ t$

iii) $>\#\triangleright$ iff $(\exists p',\tau')(\nexists p'',\tau'')\ s >(p',\tau')\ t \wedge s \triangleright(p'',\tau'')\ t \wedge (\exists p',\tau')(\nexists p'',\tau'')\ s \triangleright(p',\tau')\ t \wedge s >(p'',\tau'')\ t$

4.2 Lemma Let $\triangleright$ be total :

i) RPO $\subset$ RPOS
ii) PSD $\subset$ PSDS
iii) RDO $\subset$ RDOS
iv) IRD $\subset$ IRDS
v) KBO $\subset$ KBOS

Proof : see 6.16 (pp. 42 - 43)

4.3 Lemma Let $\triangleright$ be total :

i) RPO $\subset$ PSO
ii) PSO = PSD
iii) PSD $\subset$ IRD
iv) IRD = KNS
v) IRD $\supset$ RDO
vi) RDO $\supset$ RPO

Proof : see 6.17 (on page 43)

4.4 Lemma Let $\triangleright$ be total : RPOS $\subset$ RDOS.

Proof : see 6.18 (pp. 43 - 45)

4.5 Lemma Let $\triangleright$ be total : RDOS $\subset$ IRDS.

Proof : see 6.19 (pp. 45 - 46)

4.6 Lemma IRDS = KNSS.

Proof: see 6.20 - 6.21 (pp. 46 - 48)

4.7 Lemma Let $\triangleright$ be total: IRDS $\supset$ PSDS.

Proof: see 6.22 - 6.24 (pp. 48 - 51)

4.8 Lemma Let $\triangleright$ be total: PSDS $\supset$ RPOS.

Proof: see 6.25 (on page 51)

4.9 Lemma Let $\triangleright$ be total:

i) RDO , PSD , IRD # RPOS
ii) PSD , PSDS # RDO , RDOS
iii) IRD # RDOS , PSDS
iv) KBOS , KBO # RPO , RPOS , PSD , PSDS , RDO , RDOS , IRD , IRDS

Proof: see 6.26 (on page 52)

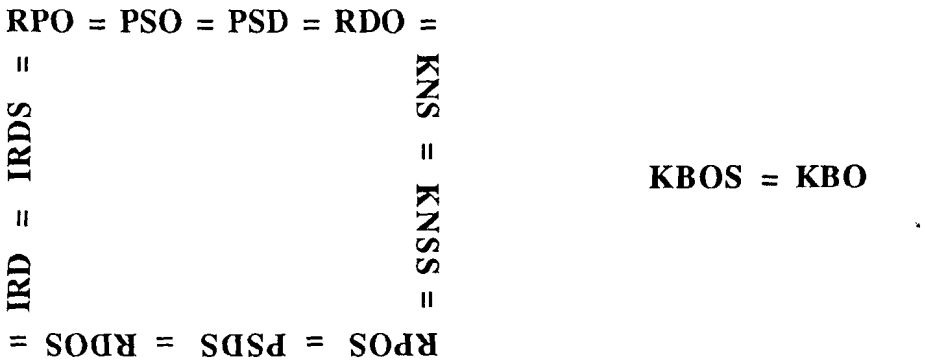
In order to retain these relations and to find one of them easily we use a kind of *Hasse diagrams*. If $> \subset \triangleright$ then we arrange $\triangleright$ above $>$ joining them with a thick arrow:

The diagram of 4.12 on page 27 summarizes the previous lemmata. Arbitrary terms provided, we have compared the orderings. It is a question now whether the relations among the orderings will endure if this precondition is restricted. We have investigated the restrictions to ground terms and to monadic terms.

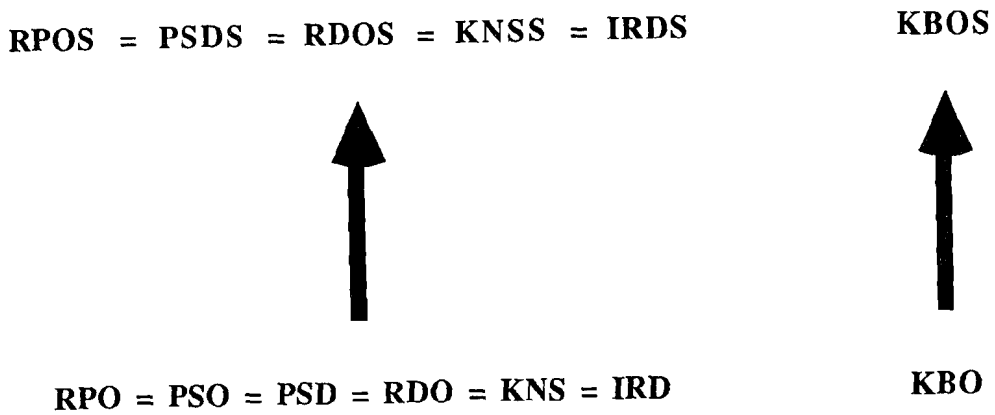
If we consider T_G (the set of ground terms) instead of T and take a total precedence $\triangleright$, then the path and decomposition orderings are total on $T_G/\sim$. The orderings with multiset status are equivalent and included in the orderings on arbitrary status which themselves are equivalent. The Knuth-Bendix ordering is also total on $T_G/\sim$ (KBO $\subset$ KBOS) and overlaps with the others. $f(g(a)) >_{KBO} g(f(f(a)))$ and $g(f(f(a))) \blacktriangleright f(g(a))$ hold under the following presumptions: $\varphi(f) = 0$, the total precedence $f \triangleright g \triangleright a$ and $\blacktriangleright$ is any of the remaining orderings. The graph of the results on ground terms can be found on the following page.

A *monadic term* only contains unary function symbols and either a constant or a variable. The subset of the monadic terms without constants can unequivocally be transformed into strings and vice versa. Therefore, the subsequent result refers to the corresponding orderings on string rewriting systems. Note that, on monadic terms, an ordering with status and the version without status coincide. On condition of a total precedence, all orderings with the exception of the KBO(S) are the same. The counter-example taken from ground terms is responsible for the overlapping of the KBO(S) with the others. Due to lack of space, the proofs cannot explicitly be given here but may be found in [St86]. Analogous with arbitrary terms, a synopsis in the form of a diagram is presented below.
It is mentionable that the class of path and decomposition orderings on monadic terms is equivalent to the *syllabled* or collected ordering $>_{CO}$ (see [Si87], [Wi88] or [Ba81]) on the reverse words: $u >_{RPO,...} v$ iff $reverse(u) >_{CO} reverse(v)$.

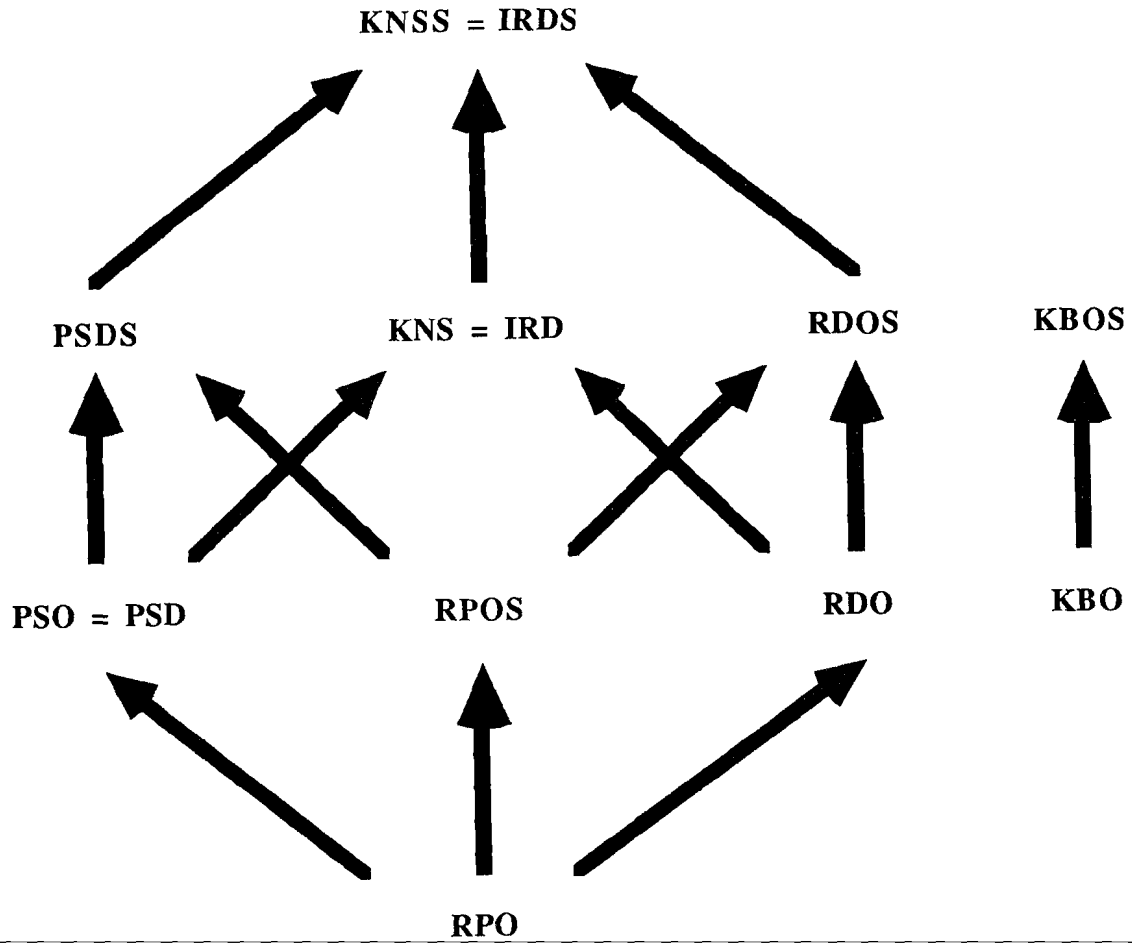
4.10 Total precedence and monadic terms



4.11 Total precedence and ground terms



4.12 Total precedence and arbitrary terms



5 Implementation and Conclusion

The TRSPEC is a system for algebraic specifications based on term rewriting techniques and developed at the university of Kaiserslautern from the research group PROGRESS (**Pro**jektgruppe **Re**duktionssysteme). It is implemented in Common Lisp and currently running on Apollo Domain Systems as well as on SUN workstations. It consists of approximately 8000 lines of source code (without documentation) corresponding to 400 KB of compiled code.
The kernel of the TRSPEC-system is COMTES, an extended Knuth-Bendix completion procedure. Furthermore, the TRSPEC contains the following tools : A parser for transforming hierarchical structured specifications into an internal representation and for checking the syntax of the specification. A checker for testing the completeness and uniqueness of function definitions. A compiler for transforming function definitions into executable lisp code. A prover for proving inductive properties of the defined functions.
COMTES can be viewed as a parametric system that is particularly suited for efficiency experiments. Beside different reduction strategies, the parameters also include various termination methods.

The three main topics of this chapter are the following :

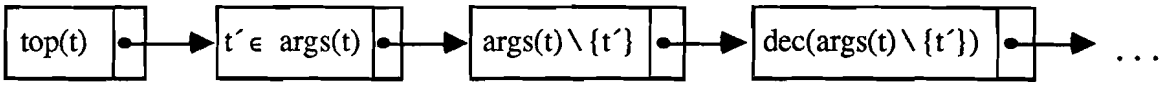
i) The comparison between the definition and the implementation of the orderings,
ii) The study of the time complexity of the orderings guided by a series of experiments and
iii) Concluding remarks.

5.1 Differences between the definitions and their implementations

The path orderings RPOS, PSO, KNSS, IRDS, PSDS and the Knuth-Bendix ordering are implemented. Each technique only checks whether a term s is greater (w.r.t. the technique) than a term t. Consequently, if the test $s > t$ stops with negative result, we have to repeat it in reverse order : $t > s$?

Before comparing the methods and their implementations let us note that we made a general multiset ordering available. It depends on the following parameters : An equality relation (for example the term congruence $\sim$), an ordering, a precedence, a status and a weight function (the latter is empty if the second parameter is a path ordering).

- The implementation and the definition of the RPOS are essentially equal. Additionally, a process will previously be performed which checks the subterm property. The non-determinism in the definition will be evaded by testing the fourth condition only if the three previous cases failed.
- The PSO compares two terms by comparing all their paths. Like the definition on page 7, a path as well as the multiset of all paths of a term will be explicitly produced and composed of terms. The implementation of the PSO is also identical to its definition.
- We utilized some ideas of the KNSS given in [KNS85]. These notions are based on the RPOS since KNSS originates from the RPOS. Before constructing the paths of two terms s and t, case distinctions relative to the leading function symbols will be performed : If top(s) ▷ top(t), then $\{s\} \gg_{KNSS} args(t)$ must hold (therefore, we can renounce with the construction of the paths). Only if top(s) = top(t) we use the demand $(path(\{s\}) \gg_{LK} path(\{t\}))$ of the definition 2.5. Two additional checks are installed to enhance the efficiency : i) The test of the subterm property (cf. RPOS). ii) Before comparing two paths their ends will be checked (i.e. if the seemingly smaller path ends with a variable the other path must have the same end).
- The definition and the implementation of the IRDS coincide except for the data structure of an elementary decomposition. A component of a path-decomposition used in the definition is a term. Such an elementary decomposition is divided and implemented as a list. It has been attuned to the definition of the IRDS. An elementary decomposition (a term) t consists of the following elements (only important ones) :



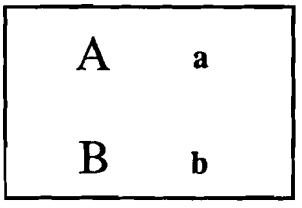
Note that this is an extremely recursive structure (cf. the fourth cell). The direct access to any component used in the definition of the IRDS is its advantage.

- The implementations of the PSDS and the IRDS only differ in the principle of comparing the subterms of terms with equal top symbols : The PSDS works breadth-first while the IRDS works depth-first.
- The definition and the implementation of the KBOS coincide. As an additional trick, the subterm property will previously be tested.

5.2 Time behaviour

We chose 18 pairs of terms. Each pair is comparable w.r.t. all orderings implemented. The chronometries are entered in the table on page 31. The column at the left margin contains the sizes of the terms s and t of a pair. More precisely, the first value denotes the addition of both sizes (|s| + |t|)

while the second number (enclosed in parenthesis) stands for the size of the greater (w.r.t. the ordering) term (|s|). The values of the other sections of the table have the following meanings :



A is the mean (or average) of a series of four experiments (with the same pair) and **a** is the corresponding standard deviation. The values refer to a positive test, i.e. we conjectured that $s > t$, and really $s > t$ holds. B and **b** are responsible for the negative test (the hypothesis was incorrect). The values A, **a**, B and **b** represent internal run time measured in milliseconds.

The table is somewhat difficult to survey. But we would not renounce it since there are many details which can explain the time complexity of an ordering. Nevertheless, we now want to summarize the informations of the table into a lucid one :

	RPOS	PSO	KNSS	IRDS	PSDS	KBOS
average standard deviation (in %)	11,8	2,6	6,5	4,8	3,2	5,9
average time for the positive test divided by the time for the negative test	0,6	2,2	1,6	1,2	1,6	0,9
total factors	1,0	19,0	4,1	8,1	16,0	1,7

The total factors are the average factors of all (positive and negative) tests related to the RPOS.

An amazing fact is that, in contrast to the other orderings, the RPOS needs more time for the negative test (the cause of it could be the non-determinism of condition iv). Compared to the other orderings, the RPOS and the KBOS come off well (see total factors) : The KBOS indeed is slower than the RPOS by factor 1,7 but all other orderings are slower by minimum factor 4. The probable reason is the time for constructing the paths and the decompositions, respectively. In general, it is more expensive to build up a path-decomposition than the corresponding path (since the data structure of a path-decomposition is more complex than that of a path, cf. 5.1). The time difference between the KNSS and the IRDS confirms our observation. In contrast to that, the total factors of the PSDS and the PSO do not agree with the previous evaluation (that the time for the construction of a path is lower than that of the corresponding path-decomposition). This originates in the fact that the average values distort the measurements : In 75% of the cases the PSDS is slower

size \ order	RPOS		PSO		KNSS		IRDS		PSDS		KBOS	
3 (2)	102	10	2101	360	167	99	95	13	91	1	110	11
	72	0	978	139	123	11	2475	573	2694	168	113	1
5 (2)	643	10	2270	505	2538	587	4466	14	4463	97	2329	3
	975	159	3305	294	2721	420	4946	687	4624	387	3365	558
6 (5)	163	9	2850	78	172	11	156	11	166	23	170	11
	80	16	2282	714	118	9	4699	448	4199	402	113	1
6 (3)	1396	123	8510	78	7308	114	10056	176	9437	439	2547	12
	3310	1320	7722	648	6853	542	10135	44	10276	83	2635	131
6 (2)	695	27	5936	122	2135	481	8170	104	7550	662	2603	3
	947	2	4699	119	3463	20	7939	567	7554	494	2539	8
7 (4)	1785	637	16839	602	12517	98	17338	1163	17593	273	2656	87
	2431	580	11414	52	7886	522	10475	897	12915	570	2603	29
7 (3)	2285	292	5845	645	6051	888	9544	811	9459	847	3252	550
	2768	252	6875	875	2598	287	8944	919	9037	1002	3828	463
8 (4)	1073	17	18431	1706	12465	1652	17954	1903	20034	2143	4553	753
	2921	57	12752	148	3222	523	12634	90	15648	476	3846	136
8 (4)	1393	1	5889	42	4400	496	9354	106	9093	568	4126	635
	1571	238	5682	454	3479	505	10144	1489	8612	1406	3737	3
8 (4)	867	153	5708	557	2618	511	9243	1084	9760	900	3786	2
	1199	525	5819	874	5314	1059	9222	802	8705	653	4921	111
8 (4)	887	12	6644	545	5078	638	14075	184	14067	52	3802	13
	687	9	5345	540	4936	784	8195	83	8479	617	3841	101
8 (4)	1155	144	7674	605	4947	536	14414	1574	13762	1783	4375	613
	958	9	6934	117	5486	149	9343	699	8960	305	3788	1
8 (4)	1994	465	17284	121	12722	1121	23344	559	26882	180	4628	491
	3553	713	17131	89	11431	264	19961	426	19826	1923	3787	2
10 (6)	2372	23	66782	903	21450	1165	25992	185	45005	460	4813	653
	2105	503	26911	266	13616	108	18488	152	24283	189	5157	471
11 (6)	2419	442	32300	2499	14274	1321	24581	2264	29806	2671	4350	82
	3641	154	22359	364	6317	508	18441	434	20362	481	5377	87
12 (6)	3992	339	60837	4045	17560	402	38248	534	76899	5223	6028	585
	5792	513	39555	3066	6395	376	34471	2326	51600	2819	5855	590
12 (6)	5203	698	196710	9998	66610	3768	61486	4812	168641	9808	15115	454
	9114	525	106686	4934	26425	2601	39529	3696	74628	337	14674	248
17 (5)	3420	504	691861	3498	32844	113	119854	534	422582	638	7306	678
	14959	991	245187	3888	26548	1023	98380	5351	251430	6041	7234	500

than the PSO. Generally, the PSDS and the IRDS have approximately the same time behaviour. The reason why the IRDS is much better than the PSDS in the last three experiments could be the following : The selection of a direct subterm (second condition of 2.11) is directly successful.
Considering the remarks of this section we come to the conclusion that the implementation of the RPOS, the KNSS and the KBOS is the most favourable solution. The RPOS serves as a pre-processor for the KNSS since the RPOS is faster than the KNSS.

5.3 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. In this paper, a collection of simplification orderings has been pointed out, including the well-known recursive path and decomposition ordering, the improved decomposition ordering of Rusinowitch, the path of subterms ordering (and an equivalent version on decompositions), the path ordering of Kapur/Narendran/Sivakumar and the Knuth-Bendix ordering. Most of the original definitions of those orderings cannot orient for example the associative laws. However, this is possible by using the principle of status (cf. [KL80]) in conjunction with the orderings. A variant with status exists as RPOS ([KL80]), RDOS (two versions : [Le84] and [Ru87]) and KNSS ([KNS85]). A part of our work consisted of adding the principle of status to existing orderings and therefore, of creating the following new orderings : A new RDOS , PSDS (PSO with status) , IRDS (IRD of Rusinowitch with status) and KBOS. It turned out that the IRDS (= KNSS) is the most powerful ordering of the class of path and decomposition orderings presented. Since the KBOS and the IRDS overlap only two (resp. three) of the thirteen orderings must be exposed : **IRDS = KNSS** and **KBOS**. The definition of the KNSS is more suitable for an implementation than the IRDS. On the contrary, we prefer the IRDS to twig the technique of comparing terms. The cause of it is that the definitions of the various orderings based on decompositions have a new outward : The used decompositions only contain terms instead of vectors with three (or even four) components.
This report extends the considerable set of different decomposition orderings. To preserve the survey we finish with the comparison between the well-known and new decomposition orderings. Due to lack of time, we can only conjecture the following dependencies :

- The RDOS of [Le84] is included in our RDOS.
- The RDS of [Ru87] is a technique which connects the notions of the PSO (the principle of breadth-first), the IRD (the principle of depth-first) and lexicographical status. We suppose that the RDS and the IRDS overlap.
- The PSDS is included in the RDS of [Ru87].

A closer examination of these conjectures as well as the exact time complexities of the orderings will be part of the future plans.

6 Proofs

At the beginning we want to point to a general fact : The proof of a lemma based on an ordering with status contains the proof of the same lemma relative to the corresponding ordering without status.
In this chapter, when writing a numeral in a circle (e.g. ①, ②, ...) we will refer to a counter-example. All examples are given on page 52.

6.1 Lemma IRDS is irreflexive.

Proof : We must show that $t \not>_{IRDS} t$. It obviously follows from the definition. ❑

6.2 Lemma IRDS is transitive.

Proof : It is clear that if an ordering is transitive then its extension to multisets preserves this property (*) (cf. [DM79]). Thus it is sufficient to show that $>_{EL}$ is transitive :
$r >_{EL} s >_{EL} t \Rightarrow r >_{EL} t$. We prove it by induction on $|r| + |s| + |t|$.
Let be $r = f(r_1,...,r_k)$, $s = g(s_1,...,s_m)$ and $t = h(t_1,...,t_n)$ with $r \in dec_u(r')$, $s \in dec_v(s')$ and $t \in dec_w(t')$. We have to consider four cases :

i) $f \triangleright g \vee g \triangleright h$
- $\Rightarrow f \triangleright h$ because $\triangleright$ is a partial ordering and $r >_{EL} s >_{EL} t$
- $\Rightarrow r >_{EL} t$ by definition of $>_{EL}$

ii) $f = g = h \wedge \tau(f) = mult \wedge$
$[sub(dec_u(r'), r) \gg_{EL} sub(dec_v(s'), s) \vee sub(dec_v(s'), s) \gg_{EL} sub(dec_w(t'), t)]$
- $\Rightarrow sub(dec_u(r'), r) \gg_{EL} sub(dec_w(t'), t)$
  by induction hypothesis (since $(\forall \gamma \in sub(dec_\psi(\Delta'), \Delta))\ |\gamma| < |\Delta|$) and (*)
- $\Rightarrow r >_{EL} t$ by definition of $>_{EL}$

iii) $f = g = h \wedge \tau(f) = mult \wedge sub(dec_u(r'), r) = sub(dec_v(s'), s) = sub(dec_w(t'), t) \wedge$
$dec(args(r)) \gg\gg_{EL} dec(args(s)) \gg\gg_{EL} dec(args(t))$
- $\Rightarrow$ - $sub(dec_u(r'), r) = sub(dec_w(t'), t)$
  - $dec(args(r)) \gg\gg_{EL} dec(args(t))$
  by induction hypothesis (since $(\forall \gamma \in dec(args(\Delta)))\ |\gamma| < |\Delta|$) and (*)
- $\Rightarrow r >_{EL} t$ by definition of $>_{EL}$

iv) $f = g = h \quad \wedge \quad \tau(f) \neq mult \quad \wedge \quad args(r) >_{IRDS,\tau(f)} args(s) >_{IRDS,\tau(f)} args(t) \quad \wedge$
$\{r\} \gg_{IRDS} args(s) \quad \wedge \quad \{s\} \gg_{IRDS} args(t)$
$\Rightarrow$ - $args(r) >_{IRDS,\tau(f)} args(t)$
because $>_{IRDS}$ is equivalent with $\gg\gg_{EL}$ on decompositions, the lexicographical extension of an ordering is transitive if the ordering is, and by induction hypothesis since $(\forall \gamma \in dec(args(\Delta)))\ |\gamma| < |\Delta|$
- $\{r\} \gg_{IRDS} args(t)$
because $r >_{IRDS} s$ ($\Rightarrow \{r\} \gg_{IRDS} \{s\} \gg_{IRDS} args(t)$), by induction hypothesis and (*). It remains to be shown that $r >_{IRDS} s$. This is equivalent to $dec(\{r\}) \gg\gg_{EL} dec(\{s\})$: $dec(\{r\}) = \{d \cup \{r\} \mid d \in dec(args(r))\}$ and $dec(\{s\}) = \{d \cup \{s\} \mid d \in dec(args(s))\}$. Since $dec(\{r\}) \gg\gg_{EL} dec(args(s))$ (precondition), it remains to be shown that $r >_{EL} s$ which is also a precondition. ❑

6.3 Lemma IRDS has the subterm property.

Proof: We have to prove that $\varepsilon \neq u \in O(t)$ implies $t >_{IRDS} t/u$. Let $u \in O(t)$ and $u \neq \varepsilon$. Then, $t >_{RPOS} t/u$ (see [KL80]). This fact implies that $t >_{IRDS} t/u$ because $RPOS \subset IRDS$ (see 6.18 and 6.19). ❑

6.4 Lemma IRDS has the replacement property.

Proof: The following has to be shown:
$(\forall i \in [1,n])\ s >_{IRDS} s' \Rightarrow t := f(t_1,...,t_i[\varepsilon \leftarrow s],...,t_n) >_{IRDS} f(t_1,...,t_i[\varepsilon \leftarrow s'],...,t_n) =: t'$.
Note that $t >_{IRDS} t'$ iff $dec(\{t\}) \gg\gg_{EL} dec(\{t'\})$.
$dec(\{t\}) = \{d \cup \{t\} \mid d \in dec(args(t_j)), j \in [1,n], j \neq i\} \cup \{d \cup \{t\} \mid d \in dec(\{s\})\}$ and
$dec(\{t'\}) = \{d \cup \{t'\} \mid d \in dec(args(t_j)), j \in [1,n], j \neq i\} \cup \{d \cup \{t'\} \mid d \in dec(\{s'\})\}$.

Therefore, we have to prove
i) $dec_u(t) \ni t >_{EL} t' \in dec_u(t')$, $\forall u \in Ot(t) \setminus \{i.v \mid v \in Ot(s)\}$
If this statement is valid the path-decompositions of $\{d \cup \{t\} \mid d \in dec(args(t_j)), j \in [1,n], j \neq i\}$ are greater than those of $\{d \cup \{t'\} \mid d \in dec(args(t_j)), j \in [1,n], j \neq i\}$

ii) $dec(\{s\}) \gg\gg_{EL} dec(\{s'\})$:
This requirement directly follows from the precondition that $s >_{IRDS} s'$.

The proof of i) is divided into two parts depending on the status of the leading function symbol f of t and t':

- $\tau(f) = mult$:
  Obviously, $sub(dec_u(t), t) = sub(dec_u(t'), t')$ if u is restricted to the set of the precondition. Therefore, the question is whether $dec(args(t)) \gg_{EL} dec(args(t'))$ is true which is equivalent to the statement that $dec(\{s\}) \gg_{EL} dec(\{s'\})$ (which is proved in ii) since the remaining arguments of t and t' are identical.

- $\tau(f) = left$
  ➟ We must show that $\alpha)$ $args(t) >_{IRDS,\tau(f)} args(t')$ and $\beta)$ $\{t\} \gg_{IRDS} args(t')$:
  $\alpha)$ is true because $args(t) \ni t_j = t_j \in args(t')$, for all $j < i$ and $args(t) \ni t_i = s >_{IRDS} s' = t_i \in args(t')$ which is the precondition.
  $\beta)$ is valid since $args(t) \gg_{IRDS} args(t')$ (cf. previous case), $\{t\} \gg_{IRDS} args(t)$ (subterm property of the IRDS, see 6.3), and the IRDS is transitive (6.2).

- $\tau(f) = right$:
  analogous with the previous case ❑

6.5 Lemma IRDS is stable w.r.t. substitutions.

Proof (cf. [RJ81]) : We have to show that $s >_{IRDS} t$ implies $\sigma(s) >_{IRDS} \sigma(t)$, for any substitution σ. Strictly speaking, we must prove : $(\forall v \in Ot(t))\ (\exists u \in Ot(s))\ dec_u(s) \gg_{EL} dec_v(t)$ ➟ $(\forall v' \in Ot(\sigma(t)))\ (\exists u' \in Ot(\sigma(s)))\ dec_{u'}(\sigma(s)) \gg_{EL} dec_{v'}(\sigma(t))$.
Let $v' \in Ot(\sigma(t))$, then $(\exists i, v \in \mathbb{N}^*)\ v' = v.i \ \wedge \ v \in Ot(t)$
Since $s >_{IRDS} t$ there exists $u \in Ot(s)$ such that $dec_u(s) \gg_{EL} dec_v(t)$. To prove that $(\exists \psi \in Ot(\sigma(s)))\ dec_\psi(\sigma(s)) \gg_{EL} dec_{v'}(\sigma(t))$ we have to distinguish two cases :

i) $t/v \notin \mathcal{X}$ ➟ $v' = v$

- $s/u \notin \mathcal{X}$
  ➟ $u \in Ot(\sigma(s))$
  ➟ $dec_u(\sigma(s)) \gg_{EL} dec_v(\sigma(t))$
  since $dec_u(s) \gg_{EL} dec_v(t)$
  ➟ $\psi = u$

- $s/u = x \in \mathcal{X}$
  ➟ $(\forall w \in Ot(\sigma(x)))\ dec_{u.w}(\sigma(s)) \gg_{EL} dec_v(\sigma(t))$
  because $dec_u(s) \gg_{EL} dec_v(t)$
  ➟ $\psi = u.w'$ with $w' \in Ot(\sigma(x))$

ii) $t/v = x \in \mathcal{X}$ ➞ $s/u = x$ (otherwise $dec_u(s) \not\gg_{EL} dec_v(t)$)
➞ $v' = v.i$
➞ $dec_{u.i}(\sigma(s)) \gg_{EL} dec_{v.i}(\sigma(t))$
because $dec_u(s) \gg_{EL} dec_v(t)$ and $\sigma(s)/u = \sigma(t)/v = \sigma(x)$
➞ $\psi = u.i$ ❑

6.6 Lemma A partial and monotonous (w.r.t. the precedence) ordering $>$ is well-founded if there exists a well-founded ordering $\triangleright$ which contains $>$ w.r.t. total precedences.

Proof : Assume that p is a partial precedence and $>$ is not well-founded
➞ $\exists\ t_1 >(p)\ t_2 >(p) \ldots$
➞ $\exists\ t_1 >(q)\ t_2 >(q) \ldots$ with $p \subseteq q$ and q is total
because $>$ is monotonous w.r.t. the precedence
➞ $\exists\ t_1 \triangleright(q)\ t_2 \triangleright(q) \ldots$
because $> \subseteq \triangleright$ w.r.t. total precedences
↯ to $\triangleright$ is well-founded ❑

6.7 Lemma RPOS is monotonous w.r.t. the precedence.

Proof : We show by induction on $|s| + |t|$ that $s >_{RPOS}(q) t$ if $s >_{RPOS}(p) t$ and $p \subseteq q$. The requirements i), ii) and iii) of the definition of the RPOS (on page 9) are fulfilled by the induction hypothesis and the fact that $f \triangleright g$ (resp. $f = g$) $\in p$ implies $f \triangleright g$ (resp. $f = g$) $\in q$. The only crucial case is iv) :

$args(s) \geq_{RPOS}(p) \{t\}$
➞ - $args(s) \gg_{RPOS}(p) \{t\}$
➞ $args(s) \gg_{RPOS}(q) \{t\}$
by induction hypothesis
or
- $args(s) = \{s'\}$ ∧ $s' \sim(p) t$
➞ $s' \sim(q) t$
because $f = g \in p$ implies $f = g \in q$
➞ $args(s) \geq_{RPOS}(q) \{t\}$ (*)

Let s, t be $s = f(s_1,...,s_m)$ and $t = g(t_1,...,t_n)$ with $f \# g \in p$ and $f \triangleright g \in q$ (the other cases $f \triangleright g$, $g \triangleright f \in p$ are very easy to prove). The definition of the RPOS must be unequivocal and therefore, we have additionally to show that $\{s\} \gg_{RPOS}(q) args(t)$.

Assume that there exists a t_i such that $s \not>_{RPOS}(q) t_i$.
$\Rightarrow$ $s \not>_{RPOS}(q) t$
because the RPOS is transitive and possesses the subterm property ([KL80])
$\Rightarrow$ $args(s) \not\geq_{RPOS}(q) \{t\}$
$\lightning$ to (*) ❑

6.8 Lemma RDOS is monotonous w.r.t. the precedence.

Proof : Since $>_{RDOS}$ is equivalent to $\gg_{LD}$ on elementary decompositions and $\gg_{LD}$ is the extension of $>_{LD}$ to multisets of multisets, it is sufficient to show that $>_{LD}$ is monotonous w.r.t. the precedence : $s >_{LD}(p) t \Rightarrow s >_{LD}(q) t$ if $p \subseteq q$. The proof consists of the induction on $|s|+|t|$ and of the case distinctions on the leading function symbols of s and t. Let $s = f(s_1,...,s_m)$, $t = g(t_1,...,t_n)$ and $s \in dec_u(s')$, $t \in dec_v(t')$.

i) $f \triangleright g \in p$
$\Rightarrow$ $f \triangleright g \in q$
because $p \subseteq q$
$\Rightarrow$ $s >_{LD}(q) t$

ii) $f = g \in p \wedge \tau(f) = mult \wedge sub(dec_u(s'), s) \gg_{LD}(p) sub(dec_v(t'), t)$
$\Rightarrow$ $f = g \in q \wedge \tau(f) = mult$
because $p \subseteq q$
$\Rightarrow$ We have to show $sub(dec_u(s'), s) \gg_{LD}(q) sub(dec_v(t'), t)$: This is valid because of the induction hypothesis (since $(\forall \gamma \in sub(dec_\psi(\Delta'), \Delta)) |\gamma| < |\Delta|$) and because the extension of $>_{LD}$ to multisets preserves the required condition.

iii) $f = g \in p \wedge \tau(f) = mult \wedge sub(dec_u(s'), s) =(p) sub(dec_v(t'), t) \wedge$ $args(s) \gg_{RDOS}(p) args(t)$
$\Rightarrow$ - $f = g \in q \wedge \tau(f) = mult \wedge sub(dec_u(s'), s) =(q) sub(dec_v(t'), t)$
because $p \subseteq q$ and $sub(dec_u(s'), s) = sub(dec_v(t'), t)$
- $args(s) \gg_{RDOS}(q) args(t)$
because $\gg_{RDOS}$ is the extension of $>_{RDOS}$ to multisets and $>_{RDOS}$ is equivalent to $\gg_{LD}$ and because of the induction hypothesis (since $(\forall \gamma \in args(\Delta)) |\gamma| < |\Delta|$)

iv) $f = g \in p \wedge \tau(f) \neq mult \wedge args(s) >_{RDOS,\tau(f)}(p) args(t) \wedge \{s\} \gg_{RDOS}(p) args(t)$
$\Rightarrow$ $args(s) >_{RDOS,\tau(f)}(q) args(t) \wedge \{s\} \gg_{RDOS}(q) args(t)$
with the help of the considerations of the previous cases ❑

6.9 Lemma RPOS is stable w.r.t. substitutions.

Proof : We have to prove that, for arbitrary terms s and t, $s >_{RPOS} t$ implies $\sigma(s) >_{RPOS} \sigma(t)$ for any substitution σ. We show this statement inductively on $|s| + |t|$. Let $s = f(s_1,...,s_m)$ and $t = g(t_1,...,t_n)$.

i) $f \triangleright g \quad \wedge \quad \{s\} \gg_{RPOS} args(t)$
- $\Rightarrow \{\sigma(s)\} \gg_{RPOS} args(\sigma(t))$
  by induction hypothesis
- $\Rightarrow \sigma(s) >_{RPOS} \sigma(t)$
  because $top(\sigma(s)) = f \triangleright g = top(\sigma(t))$

ii) $f = g \quad \wedge \quad \tau(f) = mult \quad \wedge \quad args(s) \gg_{RPOS} args(t)$
- $\Rightarrow args(\sigma(s)) \gg_{RPOS} args(\sigma(t))$
  by induction hypothesis
- $\Rightarrow \sigma(s) >_{RPOS} \sigma(t)$
  because $top(\sigma(s)) = top(\sigma(t)) = f \quad \wedge \quad \tau(f) = mult$

iii) $f = g \quad \wedge \quad \tau(f) \neq mult \quad \wedge \quad args(s) >_{RPOS,\tau(f)} args(t) \quad \wedge \quad \{s\} \gg_{RPOS} args(t)$
- $\Rightarrow$ - $args(\sigma(s)) >_{RPOS,\tau(f)} args(\sigma(t))$
  by induction hypothesis and since $s \sim t$ implies $\sigma(s) \sim \sigma(t)$
  - $\{\sigma(s)\} \gg_{RPOS} args(\sigma(t))$
  by induction hypothesis
- $\Rightarrow \sigma(s) >_{RPOS} \sigma(t)$
  because $top(\sigma(s)) = top(\sigma(t)) = f \quad \wedge \quad \tau(f) \neq mult$

iv) $args(s) \geq_{RPOS} \{t\}$:

- $(\exists i \in [1,m])\ s_i >_{RPOS} t$
  - $\Rightarrow (\exists i \in [1,m])\ \sigma(s_i) >_{RPOS} \sigma(t)$
    by induction hypothesis
  - $\Rightarrow \sigma(s) >_{RPOS} \sigma(t)$
    by definition of the RPOS

- $(\exists i \in [1,m])\ s_i =_{RPOS} t$
  - $\Rightarrow (\exists i \in [1,m])\ \sigma(s_i) =_{RPOS} \sigma(t)$
    because $s =_{RPOS} t$ iff $s \sim t$ and $s \sim t$ implies $\sigma(s) \sim \sigma(t)$
  - $\Rightarrow \sigma(s) >_{RPOS} \sigma(t)$
    by definition of the RPOS

❑

6.10 Lemma KBOS is irreflexive.

Proof: We must show that $t \not>_{KBOS} t$. This is easily proved by induction on the size of t. ❑

6.11 Lemma KBOS is transitive.

Proof: We have to prove that $r >_{KBOS} s >_{KBOS} t \rightarrow r >_{KBOS} t$. It is clear that $(\forall x \in \mathcal{X})\ \#_x(r) \geq \#_x(t)$ if $(\forall x \in \mathcal{X})\ \#_x(r) \geq \#_x(s) \geq \#_x(t)$ because $>$ on $\mathbb{N}$ is a partial ordering. We have to consider three cases which will be proved by induction on $|r| + |s| + |t|$.

i) $\varphi(r) > \varphi(s) \quad \vee \quad \varphi(s) > \varphi(t)$
   $\rightarrow \varphi(r) > \varphi(t)$
   because $>$ is a partial ordering
   $\rightarrow r >_{KBOS} t$

ii) $\varphi(r) = \varphi(s) = \varphi(t) \quad \wedge \quad [top(r) \triangleright top(s) \quad \vee \quad top(s) \triangleright top(t)]$
   $\rightarrow \varphi(r) = \varphi(t) \quad \wedge \quad top(r) \triangleright top(t)$
   because $\triangleright$ is a partial ordering
   $\rightarrow r >_{KBOS} t$

iii) $\varphi(r) = \varphi(s) = \varphi(t) \quad \wedge \quad top(r) = top(s) = top(t)$:
   Let be $r = f(r_1,...,r_n)$, $s = f(s_1,...,s_n)$ and $t = f(t_1,...,t_n)$.
   - $\tau(f) = left$
     $\rightarrow (\exists i, j)\ (\forall k < i)\ (\forall l < j)\ r_k \sim s_k \quad \wedge \quad s_l \sim t_l \quad \wedge \quad r_i >_{KBOS} s_i \quad \wedge \quad s_j >_{KBOS} t_j$

     α) $i \neq j$
        $\rightarrow r_i >_{KBOS} s_i =_{KBOS} t_i \quad \vee \quad r_j =_{KBOS} s_j >_{KBOS} t_j$
        $\rightarrow r_i >_{KBOS} t_i \quad \vee \quad r_j >_{KBOS} t_j$
        because $s_i \sim t_i$ (resp. $r_j \sim s_j$)
        $\rightarrow r >_{KBOS} t$
        by definition of the KBOS

     β) $i = j$
        $\rightarrow r_i >_{KBOS} s_i >_{KBOS} t_i$
        $\rightarrow r_i >_{KBOS} t_i$
        by induction hypothesis
        $\rightarrow r >_{KBOS} t$
        by definition of the KBOS

- $\tau(f) = right$:
  analogous with the case $\tau(f) = left$

- $\tau(f) = mult$
  - ➡ $\{r_1,...,r_n\} \gg_{KBOS} \{s_1,...,s_n\} \gg_{KBOS} \{t_1,...,t_n\}$
  - ➡ $\{r_1,...,r_n\} \gg_{KBOS} \{t_1,...,t_n\}$
    by induction hypothesis and since the extension of an ordering to multisets is transitive (see [DM79])
  - ➡ $r >_{KBOS} t$
    by definition of the KBOS ❑

6.12 Lemma KBOS has the subterm property.

Proof : We have to show that $\varepsilon \neq u \in O(t)$ ➡ $t >_{KBOS} t/u$. Let us consider a term t and an occurrence $u \in O(t)$, $top(t) = f$ and $top(t/u) = g$. It is clear that $(\forall x \in \mathcal{X})\ \#_x(t) \geq \#_x(t/u)$.
We must distinguish two cases which will be proved by induction on $|t|$:
i) $\varphi(t) > \varphi(t/u)$
  ➡ $t >_{KBOS} t/u$

ii) $\varphi(t) = \varphi(t/u)$
  ➡ $\varphi(f) = 0 \quad \wedge \quad f$ is a unary function symbol
  - $f \neq g$
    - ➡ $f \triangleright g$
      (see 3.2 on page 21)
    - ➡ $t >_{KBOS} t/u$
      by definition of the KBOS
  - $f = g$:
    Let $t = f(t')$, $t/u = f(t'')$
    - ➡ $t' >_{KBOS} t''$
      by induction hypothesis since t' (resp t'') is a proper subterm of t (resp. t')
    - ➡ $t >_{KBOS} t/u$
      by definition of the KBOS ❑

6.13 Lemma KBOS has the replacement property.

Proof : It is to show $(\forall i \in [1,n])\ s >_{KBOS} s'$ ➡ $t := f(t_1,...,t_i[\varepsilon \leftarrow s],...,t_n) >_{KBOS} f(t_1,...,t_i[\varepsilon \leftarrow s'],...,t_n) =: t'$.

Clearly, $(\forall x \in \mathcal{X})\ \#_x(s) \geq \#_x(s') \rightarrow (\forall x \in \mathcal{X})\ \#_x(t) \geq \#_x(t')$.
We have to examine two cases :

i) $\varphi(s) > \varphi(s')$
   $\rightarrow \varphi(t) > \varphi(t')$
   $\rightarrow t >_{KBOS} t'$

ii) $\varphi(s) = \varphi(s')$
   $\rightarrow \varphi(t) = \varphi(t')$
   $\rightarrow$ We have to show that $args(t) >_{KBOS,\tau(f)} args(t')$ (because $top(t) = top(t')$) :
   Since the arguments of t and t' are equal except the t_i's, the requirement above is fulfilled irrespective of the status of f. ❑

With the help of the following lemma we will prove that the KBOS is stable w.r.t. substitutions.

6.14 Lemma Let be $s, t \in \Gamma$ and $(\forall x \in \mathcal{X})\ \#_x(s) \geq \#_x(t)$.
Then, $(\forall \sigma)$ i) $(\forall x \in \mathcal{X})\ \#_x(\sigma(s)) \geq \#_x(\sigma(t))$
ii) $\varphi(s) > \varphi(t) \rightarrow \varphi(\sigma(s)) > \varphi(\sigma(t))$
iii) $\varphi(s) = \varphi(t) \rightarrow \varphi(\sigma(s)) \geq \varphi(\sigma(t))$

Proof :
i) Let $x \in \mathcal{X}$, we have to show that $\#_x(\sigma(s)) - \#_x(\sigma(t)) \geq 0$.
Note that $\#_x(\sigma(s)) = \sum_{y \in \mathcal{X}} \#_y(s) * \#_x(\sigma(y))$

$$\rightarrow \#_x(\sigma(s)) - \#_x(\sigma(t)) = \sum_{y \in \mathcal{X}} \#_y(s) * \#_x(\sigma(y)) - \sum_{y \in \mathcal{X}} \#_y(t) * \#_x(\sigma(y)) =$$

$$= \sum_{y \in \mathcal{X}} [\#_y(s) - \#_y(t)] * \#_x(\sigma(y)) \geq 0$$

which follows from the precondition $(\forall x \in \mathcal{X})\ \#_x(s) \geq \#_x(t)$

ii) We must prove that $\varphi(\sigma(s)) - \varphi(\sigma(t)) > 0$:
Note that $\varphi(\sigma(s)) = \varphi(s) + \sum_{x \in \mathcal{X}} \#_x(s) * [\varphi(\sigma(x)) - \varphi_0]$.

$$\rightarrow \varphi(\sigma(s)) - \varphi(\sigma(t)) = \varphi(s) + \sum_{x \in \mathcal{X}} \#_x(s) * [\varphi(\sigma(x)) - \varphi_0] - \varphi(t) - \sum_{x \in \mathcal{X}} \#_x(t) * [\varphi(\sigma(x)) - \varphi_0] =$$

$$= \varphi(s) - \varphi(t) + \sum_{x \in \mathcal{X}} [\varphi(\sigma(x)) - \varphi_0] * [\#_x(s) - \#_x(t)]$$

This expression is greater than zero because $\varphi(s) - \varphi(t) > 0$ (precondition), $\varphi(\sigma(x)) - \varphi_0 \geq 0$ since $\varphi(x) = \varphi_0$, $\#_x(s) - \#_x(t) \geq 0$ (precondition).

iii) This fact is proved the same way as before. ❑

6.15 Lemma KBOS is stable w.r.t. substitutions.

Proof : We have to show that $(\forall\sigma)\ s >_{KBOS} t \rightarrow \sigma(s) >_{KBOS} \sigma(t)$. This will be accomplished by induction on $|s| + |t|$. Let be $s = f(s_1,...,s_m)$ and $t = g(t_1,...,t_n)$. Note that the variable condition is fulfilled : $\#_x(\sigma(s)) \geq \#_x(\sigma(t))$ (see 6.14 i).

i) $\varphi(s) > \varphi(t)$
- $\rightarrow \varphi(\sigma(s)) > \varphi(\sigma(t))$ (cf. 6.14 ii)
- $\rightarrow \sigma(s) >_{KBOS} \sigma(t)$

ii) $\varphi(s) = \varphi(t) \quad \wedge \quad f \triangleright g$
- $\rightarrow \varphi(\sigma(s)) \geq \varphi(\sigma(t)) \quad \wedge \quad top(\sigma(s)) = f \triangleright g = top(\sigma(t))$ (cf. 6.14 iii)
- $\rightarrow \sigma(s) >_{KBOS} \sigma(t)$

iii) $\varphi(s) = \varphi(t) \quad \wedge \quad f = g \quad \wedge \quad \tau(f) = mult$
- $\rightarrow \{s_1,...,s_m\} \gg_{KBOS} \{t_1,...,t_n\}$
- $\rightarrow \{\sigma(s_1),...,\sigma(s_m)\} \gg_{KBOS} \{\sigma(t_1),...,\sigma(t_n)\}$
  by induction hypothesis and the fact that $s =_{KBOS} t \rightarrow \sigma(s) =_{KBOS} \sigma(t)$
- $\rightarrow \sigma(s) >_{KBOS} \sigma(t)$

iv) $\varphi(s) = \varphi(t) \quad \wedge \quad f = g \quad \wedge \quad \tau(f) = left$
- $\rightarrow (\exists i)\,(\forall j < i)\ s_j =_{KBOS} t_j \quad \wedge \quad s_i >_{KBOS} t_i$
- $\rightarrow (\exists i)\,(\forall j < i)\ \sigma(s_j) =_{KBOS} \sigma(t_j) \quad \wedge \quad \sigma(s_i) >_{KBOS} \sigma(t_i)$
  by induction hypothesis and the fact that $s =_{KBOS} t \rightarrow \sigma(s) =_{KBOS} \sigma(t)$
- $\rightarrow \sigma(s) >_{KBOS} \sigma(t)$

v) $\varphi(s) = \varphi(t) \quad \wedge \quad f = g \quad \wedge \quad \tau(f) = right$:
This can be proved with the help of the considerations of the previous case. ❑

6.16 Lemma Let $\triangleright$ be total :

i) RPO $\subset$ RPOS
ii) PSD $\subset$ PSDS
iii) RDO $\subset$ RDOS
iv) IRD $\subset$ IRDS
v) KBO $\subset$ KBOS

Proof : - $(\forall$<ord>$\in \{RPO, PSD, RDO, IRD, KBO\})$ <ord> $\subseteq$ <ord>.S :
trivial, according to the definition of <ord>.S relative to <ord>

- $(\forall$<ord>$\in \{RPO, PSD, RDO, IRD\})$ <ord> $\neq$ <ord>.S : ③

- KBO $\neq$ KBOS : ②

❑

6.17 Lemma Let $\triangleright$ be total :

i) RPO $\subset$ PSO
ii) PSO = PSD
iii) PSD $\subset$ IRD
iv) IRD = KNS
v) IRD $\supset$ RDO
vi) RDO $\supset$ RPO

Proof : i) RPO $\subseteq$ PSO : [Ru87]
RPO $\neq$ PSO : ④
ii) PSO = PSD : [St88]
iii) PSD $\subseteq$ IRD : [St88]
PSD $\neq$ IRD : ①
iv) IRD = KNS : [Ru87]
v) IRD $\supseteq$ RDO : trivial, according to the definition of the IRD (cf. 2.12 on page 14)
IRD $\neq$ RDO : ⑥
vi) RDO $\supseteq$ RPO : [RJ81]
RDO $\neq$ RPO : ①

❑

6.18 Lemma Let $\triangleright$ be total : RPOS $\subset$ RDOS.

Proof : We have to show that $s >_{RPOS} t \Rightarrow s >_{RDOS} t$. The proof is performed by using induction on $|s| + |t|$. Let s, t be $s = f(s_1,...,s_m)$ and $t = g(t_1,...,t_n)$.

i) $f \triangleright g \wedge \{s\} \gg_{RPOS} \{t_1,...,t_n\}$
$\Rightarrow$ $s >_{RDOS} t_i$, for all $i \in [1,n]$
by induction hypothesis
$\Rightarrow$ $dec(\{s\}) \gg\gg_{LD} dec(\{t_i\})$, for all $i \in [1,n]$
by definition of the RDOS

Note that $dec(\{t\}) = \{d \cup \{t\} \mid d \in dec(\{t_i\}), i \in [1,n]\}$.

➡ $dec(\{s\}) \gg_{LD} dec(\{t\})$
because $top(s) = f \triangleright g = top(t)$ and there does not exist a subterm of t equivalent to s (otherwise s would not be greater than all the t_i's).

➡ $s >_{RDOS} t$
by definition of the RDOS

ii) $g \triangleright f \quad \wedge \quad \{s_1,...,s_m\} \geq_{RPOS} \{t\}$

➡ $(\exists i \in [1,m])\ s_i \geq_{RDOS} t$
by induction hypothesis

➡ $(\exists i \in [1,m])\ dec(\{s_i\}) \underline{\gg}_{LD} dec(\{t\})$
by definition of the RDOS

➡ $dec(\{s\}) \gg_{LD} dec(\{t\})$
because $(\forall u \in Ot(s_i))\ dec_u(s_i) \subset dec_{i.u}(s)$

➡ $s >_{RDOS} t$
by definition of the RDOS

iii) $f = g \quad \wedge \quad \tau(f) = mult \quad \wedge \quad \{s_1,...,s_n\} \gg_{RPOS} \{t_1,...,t_n\}$

➡ $\{s_1,...,s_n\} \gg_{RDOS} \{t_1,...,t_n\}$
by induction hypothesis

➡ $(\forall t_j)(\exists s_i)\ s_i \geq_{RDOS} t_j \quad \wedge \quad \{s_1,...,s_n\} \neq \{t_1,...,t_n\}$
by definition of multiset orderings

➡ $(\forall t_j)(\exists s_i)\ dec(\{s_i\}) \underline{\gg}_{LD} dec(\{t_j\})$ (*)
by definition of the RDOS

Note that $dec(\{s\}) = \{d \cup \{s\} \mid d \in dec(\{s_i\}),\ i \in [1,n]\}$ and
$dec(\{t\}) = \{d \cup \{t\} \mid d \in dec(\{t_i\}),\ i \in [1,n]\}$.

➡ We have to show that $(\forall v \in Ot(t))\ (\exists u \in Ot(s))\ dec_u(s) \ni s >_{LD} t \in dec_v(t)$
since (*) holds

➡ We have to prove that
$sub(dec_u(s), s) \gg_{LD} sub(dec_v(t), t) \quad \vee \quad [sub(dec_u(s), s) =_{LD} sub(dec_v(t), t) \quad \wedge$
$args(s) \gg_{RDOS} args(t)]$ (by definition of $>_{LD}$):
This can easily be shown with (*) and the fact that
$args(s) = \{s_1,...,s_n\} \gg_{RDOS} \{t_1,...,t_n\} = args(t)$.

iv) $f = g \quad \wedge \quad \tau(f) \neq mult \quad \wedge \quad args(s) >_{RPOS,\tau(f)} args(t) \quad \wedge \quad \{s\} \gg_{RPOS} args(t)$

➡ - $args(s) >_{RDOS,\tau(f)} args(t)$
by induction hypothesis and $s =_{RPOS} t$ iff $s \sim t$ iff $s =_{RDOS} t$

- $\{s\} \gg_{RDOS} args(t)$
by induction hypothesis

$\rightarrow$ $s >_{RDOS} t$
by definition of the RDOS

The RDOS and the RPOS are not equivalent : The termination of the rule ① of the counter-examples 6.27 (on page 52) can be proved with the RDOS but cannot be proved with the RPOS. ❑

6.19 Lemma Let $\triangleright$ be total : RDOS $\subset$ IRDS.

Proof : It is sufficient to show that $s >_{LD} t$ implies $s >_{EL} t$ because $>_{RDOS}$ (resp. $>_{IRDS}$) is equivalent to $\gg_{LD}$ (resp. $\gg_{EL}$) on elementary decompositions. Therefore, let s and t be $s = f(s_1,...,s_m)$ and $t = g(t_1,...,t_n)$. Furthermore, $s \in dec_u(s')$ and $t \in dec_v(t')$. The proof will be performed by using induction on $|s| + |t|$.

i) $f \triangleright g$
$\rightarrow$ $s >_{EL} t$
by definition of $>_{EL}$

ii) $f = g \wedge \tau(f) = mult \wedge sub(dec_u(s'), s) \gg_{LD} sub(dec_v(t'), t)$
$\rightarrow$ $sub(dec_u(s'), s) \gg_{EL} sub(dec_v(t'), t)$
by induction hypothesis since $(\forall \gamma \in sub(dec_\psi(\Delta'), \Delta)) |\gamma| < |\Delta|$
$\rightarrow$ $s >_{EL} t$
by definition of $>_{EL}$

iii) $f = g \wedge \tau(f) = mult \wedge sub(dec_u(s'), s) =_{LD} sub(dec_v(t'), t) \wedge args(s) \gg_{RDOS} args(t)$
$\rightarrow$ - $sub(dec_u(s'), s) =_{EL} sub(dec_v(t'), t)$
because $s =_{LD} t$ iff $s \sim t$ iff $s =_{EL} t$

- $args(s) \gg_{IRDS} args(t)$
by induction hypothesis and the fact that $>_{IRDS}$ is equivalent to $\gg_{EL}$ on elementary decompositions
$\rightarrow$ $(\forall j \in [1,n])\ (\exists i \in [1,m])\ s_i \geq_{IRDS} t_j \wedge args(s) \neq args(t)$
by definition of multiset orderings
$\rightarrow$ $(\forall j \in [1,n])\ (\exists i \in [1,m])\ dec(\{s_i\}) \underline{\gg}_{EL} dec(\{t_j\})$
by definition of the IRDS
$\rightarrow$ $dec(args(s)) \gg_{EL} dec(args(t))$

$\rightarrow$ $s >_{EL} t$
by definition of $>_{EL}$

iv) $f = g \quad \wedge \quad \tau(f) = \text{left} \quad \wedge \quad args(s) >_{RDOS,left} args(t) \quad \wedge \quad \{s\} \gg_{RDOS} args(t)$

$\Rightarrow$ - $(\exists i \in [1,n])\ (\forall j < i)\ s_j =_{RDOS} t_j \quad \wedge \quad dec(\{s_i\}) \gg\gg_{LD} dec(\{t_i\})$

$\Rightarrow$ $(\exists i \in [1,n])\ (\forall j < i)\ s_j =_{IRDS} t_j \quad \wedge \quad dec(\{s_i\}) \gg\gg_{EL} dec(\{t_i\})$
because $s =_{RDOS} t$ iff $s \sim t$ iff $s =_{IRDS} t$ and by induction hypothesis

$\Rightarrow$ $(\exists i \in [1,n])\ (\forall j < i)\ s_j =_{IRDS} t_j \quad \wedge \quad s_i >_{IRDS} t_i$
by definition of the IRDS

$\Rightarrow$ $args(s) >_{IRDS,left} args(t)$

- $\{s\} \gg_{IRDS} args(t)$
since $>_{IRDS}$ (resp. $>_{RDOS}$) is equivalent to $\gg\gg_{EL}$ (resp. $\gg\gg_{LD}$) on elementary decompositions and by induction hypothesis

$\Rightarrow$ $s >_{EL} t$
by definition of $>_{EL}$

v) $f = g \quad \wedge \quad \tau(f) = \text{right} \quad \wedge \quad args(s) >_{RDOS,right} args(t) \quad \wedge \quad \{s\} \gg_{RDOS} args(t)$:
analogous with iv)

IRDS $\neq$ RDOS : The termination of the rule ⑥ in 6.27 (on page 52) can be proved with the IRDS but not with the RDOS. ❑

With the help of the following lemma (the idea originates from [KNS85]) we will prove that the IRDS and the KNSS are equivalent.

6.20 Lemma Let be p, q and r (parts of) paths.
Then, $p >_{LK} q$ iff $p.r >_{LK} q.r$.

Proof : Let s be a term.

- $p >_{LK} q \Rightarrow p.r >_{LK} q.r$:
We show that $p >_{LK} q \Rightarrow p.[s] >_{LK} q.[s]$. This is true because $p.[s] \ni s >_{LT} s \in q.[s]$:

i) $\tau(top(s)) = \text{mult}$

$\Rightarrow$ $sub(p.[s] , s) = sub(q.[s] , s) = [\,] \quad \wedge$
$path(args(s \in p.[s])) = path(args(s \in q.[s])) \quad \wedge$
$p' := sup(p.[s] , s) >_{LK} sup(q.[s] , s) =: q'$
because $p' = p$, $q' = q$ and $p >_{LK} q$

ii) $\tau(top(s)) \neq \text{mult}$

$\Rightarrow$ $args(s \in p.[s]) = args(s \in q.[s]) \quad \wedge$
$p' := sup(p.[s] , s) >_{LK} sup(q.[s] , s) =: q'$
because $p' = p$, $q' = q$ and $p >_{LK} q$

- $p.r >_{LK} q.r \Rightarrow p >_{LK} q$:

Analogous with the if-part we prove that $p.[s] >_{LK} q.[s]$ implies $p >_{LK} q$, i.e. $(\forall t' \in q)\ (\exists s' \in p)\ s' >_{LT} t'$. Proving this statement by contradiction we assume that $(\exists t' \in q)\ (\nexists s' \in p)\ s' >_{LT} t'$.

$\Rightarrow$ $p.[s] \ni s >_{LT} t' \in q$
because $p.[s] >_{LK} q.[s]$ and therefore, $p.[s] >_{LK} q$

$\Rightarrow$ $p.[s] \ni s >_{LT} s \in q.[s]$ (*)
because $(\nexists s' \in p)\ s' >_{LT} s \in q.[s]$, otherwise $s' >_{LT} t' \in q$ since $s' >_{LT} s >_{LT} t'$ and $>_{LT}$ is transitive ($>_{LT}$ is transitive since $\triangleright$, $>_{LK}$ and $>_{KNSS}$ are transitive)

i) $\tau(top(s)) = mult$

$\Rightarrow$ $sub(p.[s] , s) = sub(q.[s] , s) = [\,] \quad \wedge$
$path(args(s \in p.[s])) = path(args(s \in q.[s])) \quad \wedge$
$sup(p.[s] , s) >_{LK} sup(q.[s] , s)$
because (*) holds

$\Rightarrow$ $p >_{LK} q$
because $sup(p.[s] , s) = p$ and $sup(q.[s] , s) = q$

$\lightning$ to the assumption

ii) $\tau(top(s)) \neq mult$

$\Rightarrow$ $args(s \in p.[s]) = args(s \in q.[s]) \quad \wedge$
$sup(p.[s] , s) >_{LK} sup(q.[s] , s)$
because (*) holds

$\Rightarrow$ $p >_{LK} q$
analogous with i) ❑

6.21 Lemma IRDS = KNSS.

Proof : The proof is divided into two parts. In the first one, we will give an alternative definition of the KNSS denoted by KNSS*, using the previous lemma. Then, we will show the equivalence of the IRDS and the KNSS*. Like Rusinowitch (cf. [Ru87]), we are able to make the check '$sup(p , s) >_{LK} sup(q , t)$' in ii) and iii) of definition 2.5 (on page 11) redundant by requiring that p and q have no common suffix. Thus, we need the denotation Θ of removing equivalent suffixes. Θ is recursively defined as follows : $p.[s] \Theta q.[t] = p \Theta q$ if $s \sim t$, and $p.[s] \Theta q.[t] = p.[s]$ otherwise. The lemma 6.20 now leads to another (but equivalent to the original) definition of the KNSS :

$s >_{KNSS*} t$
iff $path(\{s\}) \gg_{LQ} path(\{t\})$
with $p >_{LQ} q$
iff $(\forall t' \in q \Theta p)(\exists s' \in p \Theta q)\ s' >_{LU} t'$
with $p \Theta q \ni s >_{LU} t \in q \Theta p$
iff i) $top(s) \triangleright top(t)$
ii) $top(s) = top(t) \wedge \tau(top(s)) = mult \wedge$
- $sub(p \Theta q, s) >_{LQ} sub(q \Theta p, t)$
- $path(args(s)) \gg_{LQ} path(args(t))$

iii) $top(s) = top(t) \wedge \tau(top(s)) \neq mult \wedge$
$args(s) >_{KNSS*,\tau(top(s))} args(t) \wedge$
$\{s\} \gg_{KNSS*} args(t)$

The proof of the equivalence of both definitions (KNSS and KNSS*) follows directly from 6.20. The equivalence of the KNSS* and the IRDS is based on the following two statements :

i) $(\forall t' \in q \Theta p)(\exists s' \in p \Theta q)\ s' >_{LU} t'$ iff $set(p) \gg_{LU} set(q)$
ii) $dec_u(t) = set(path_u(t))$ (cf. definition of dec on page 7)

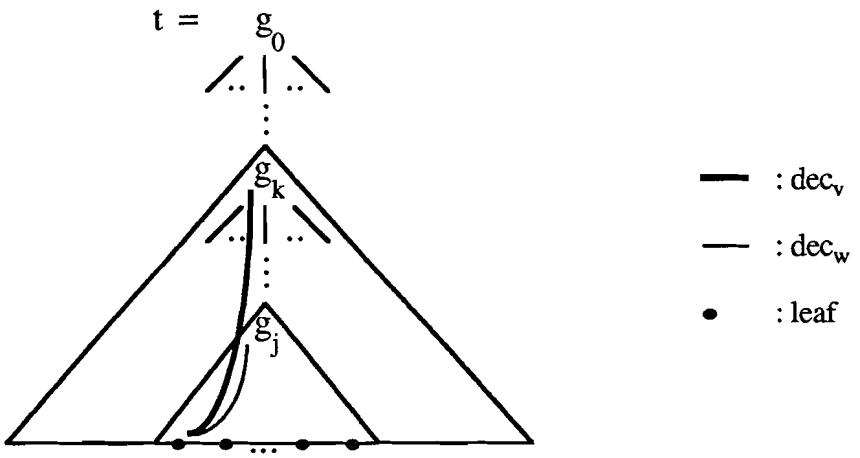
The proof of i) is easy because
$set(p) \gg_{LU} set(q)$ iff $(\forall t' \in set(q) \backslash set(p))(\exists s' \in set(p) \backslash set(q))\ s' >_{LU} t' \wedge set(p) \neq set(q)$.
Furthermore, $s_i \sim t_j$ implies $[s_1;...;s_m] \Theta [t_1;...;t_n] = [s_1;...;s_{i-1}]$ and $[t_1;...;t_n] \Theta [s_1;...;s_m] = [t_1;...;t_{j-1}]$. ❑

With the help of the following two lemmata we will show that the PSDS is included in the IRDS.

6.22 Lemma Let $\triangleright$ be total, $dec_u(t) = \{t_i \mid i \in [0,n]\}$, $g_i = top(t_i)$, t_{i+1} a direct subterm of t_i and $k = \min \{i \mid g_i \trianglerighteq g_j, j \in [0,n]\}$. Then, $(\forall i \in [0,n])\ i \neq k \rightarrow t_k >_{LP} t_i$ if $\tau(g_k) = mult$.

Proof : i) $(\nexists j \in [0,n])\ g_k = g_j$ with $j \neq k$
$\rightarrow (\forall i \in [0,n])\ i \neq k \rightarrow t_k >_{LP} t_i$
because $(\forall i \neq k)\ g_k \triangleright g_i$ and by definition of $>_{LP}$

ii) $(\exists j \in [0,n])\ g_k = g_j$ with $j \neq k$
$\rightarrow$ We have to prove that $dec(args(t_k)) \gg_{LP} dec(args(t_j))$ (because $\tau(g_k) = mult$) :
This is fulfilled since t_j is a subterm of t_k (because $k = \min...$) and therefore, for each path-decomposition dec_w of $dec(args(t_j))$ there exists a path-decomposition dec_v (w is a suffix of v, i.e. w and v mark the same leaf) of $dec(args(t_k))$ which is greater because $dec_w \subset dec_v$:



Remark : If g_n is a variable, the lemma does not hold. However, this does not bring discredit since the variable case is irrelevant and will not be considered (see 6.24 ii). The following lemma must also be restricted in this way. ❑

6.23 Lemma Let $\triangleright$ be total, $dec_u(t) = \{t_i \mid i \in [0,n]\}$, $g_i = top(t_i)$, t_{i+1} a direct subterm of t_i and $k = \min \{i \mid g_i \trianglerighteq g_j, j \in [0,n]\}$. Then, $(\forall i \in [0,n])\ i \neq k \rightarrow t_k >_{EL} t_i$ if $\tau(g_k) = mult$.

Proof : i) $(\exists j \in [0,n])\ g_k = g_j$ with $j \neq k$
$\rightarrow$ $(\forall i \in [0,n])\ i \neq k \rightarrow t_k >_{EL} t_i$
because $(\forall i \neq k)\ g_k \triangleright g_i$ and by definition of $>_{EL}$

ii) $(\exists j \in [0,n])\ g_k = g_j$ with $j \neq k$
$\rightarrow$ We have to prove that $sub(dec_u(t), t_k) \gg_{EL} sub(dec_u(t), t_j)$ (because $\tau(g_k) = mult$) : This is fulfilled since t_j is a subterm of t_k and therefore, $sub(dec_u(t), t_k) \supset sub(dec_u(t), t_j)$. (cf. proof of 6.22) ❑

6.24 Lemma Let $\triangleright$ be total : IRDS $\supset$ PSDS.

Proof : We have to prove that $s >_{PSDS} t$ implies $s >_{IRDS} t$ which is equivalent by definition of PSDS and IRDS to $dec(\{s\}) \gg\gg_{LP} dec(\{t\}) \rightarrow dec(\{s\}) \gg\gg_{EL} dec(\{t\})$. W.l.o.g. let $dec(\{s\}) \cap dec(\{t\}) = \emptyset$. By definition of multiset orderings, $dec(\{s\}) \gg\gg_{LP} dec(\{t\})$ if and only if $(\forall v \in Ot(t))\ (\exists u \in Ot(s))\ dec_u(s) \gg_{LP} dec_v(t)$. We show that $dec_u(s) \gg_{LP} dec_v(t) \rightarrow (\exists \psi \in Ot(s))\ dec_\psi(s) \gg_{EL} dec_v(t)$ by induction on $\max \{|s'| \mid s' \in dec_u(s)\} + \max \{|t'| \mid t' \in dec_v(t)\} = |s| + |t|$ which implies the inclusion IRDS $\supset$ PSDS.
Let be $dec_u(s) = \{s_i \mid i \in [0,m]\}$ with s_{i+1} is a direct subterm of s_i and $top(s_i) = f_i$. Furthermore, let be $dec_v(t) = \{t_i \mid i \in [0,n]\}$ with t_{i+1} is a direct subterm of t_i and $top(t_i) = g_i$.

Note that
- $(\exists k \in [0,m])\ (\forall i \in [0,m])\ f_k \trianglerighteq f_i \quad \wedge$
  $(\exists l \in [0,n])\ (\forall i \in [0,n])\ g_l \trianglerighteq g_i$
  because $\triangleright$ is total
- $f_k \trianglerighteq g_l$
  otherwise $dec_u(s) \not>_{LP} dec_v(t)$

i) $f_k \triangleright g_l$
➡ $dec_\psi(s) \gg_{EL} dec_v(t) \quad$ if $\psi = u$
by definition of $>_{EL}$

ii) $f_k = g_l \quad \wedge \quad \tau(f_k) = mult$:
Let l^*, k^* be $k^* = \min\{i \mid f_k = f_i, i \in [0,m]\}$ and $l^* = \min\{i \mid g_l = g_i, i \in [0,n]\}$.
➡ $s_{k^*} >_{LP} t_{l^*}$
because $>_{LP}$ is transitive and lemma 6.22
➡ $dec(args(s_{k^*})) \gg_{LP} dec(args(t_{l^*}))$
because $top(s_{k^*}) = top(t_{l^*}) = f_k$ and by definition of $>_{LP}$
➡ $(\forall w' \in Ot(t))\ (\exists w \in Ot(s))\ sub(dec_w(s), s_{k^*}) \geq_{LP} sub(dec_{w'}(t), t_{l^*})$
because $dec(args(s_{k^*})) = \{sub(dec_w(s), s_{k^*}) \mid w \in Ot(s)\}$ and
$dec(args(t_{l^*})) = \{sub(dec_w(t), t_{l^*}) \mid w \in Ot(t)\}$
➡ $(\exists w \in Ot(s))\ sub(dec_w(s), s_{k^*}) \geq_{LP} sub(dec_v(t), t_{l^*})$
because $v \in Ot(t)$
- $sub(dec_w(s), s_{k^*}) \gg_{LP} sub(dec_v(t), t_{l^*})$
  ➡ $sub(dec_w(s), s_{k^*}) \gg_{EL} sub(dec_v(t), t_{l^*})$
  by induction hypothesis
  ➡ $dec_w(s) \ni s_{k^*} >_{EL} t_{l^*} \in dec_v(t) \quad$ with $\psi = w$
  by definition of $>_{EL}$
  ➡ $dec_w(s) \gg_{EL} dec_v(t)$
  because $>_{EL}$ is transitive and lemma 6.23
- $sub(dec_w(s), s_{k^*}) =_{LP} sub(dec_v(t), t_{l^*})$
  ➡ $sub(dec_w(s), s_{k^*}) =_{EL} sub(dec_v(t), t_{l^*})$
  because the argument of s_{k^*} which belongs to the path with the terminal occurrence w is equivalent to t_{l^*+1}
  ➡ We have to show (cf. definition of $>_{EL}$) that $dec(args(s_{k^*})) \gg_{EL} dec(args(t_{l^*}))$:
  This is valid because $dec(args(s_{k^*})) \gg_{LP} dec(args(t_{l^*}))$ and by the induction hypothesis and the definition of multiset orderings.
  ➡ $dec_w(s) \ni s_{k^*} >_{EL} t_{l^*} \in dec_v(t) \quad$ with $\psi = w$
  by definition of $>_{EL}$
  ➡ $dec_w(s) \gg_{EL} dec_v(t)$
  because $>_{EL}$ is transitive and lemma 6.23

iii) $f_k = g_l \quad \wedge \quad \tau(f_k) = \text{left}$

$\rightarrow$ $(\exists k' \in [0,m])\ args(s_{k'}) >_{PSDS,left} args(t_l) \quad \wedge \quad \{s_{k'}\} \gg_{PSDS} args(t_l)$

because $dec_u(s) \gg_{LP} dec_v(t)$ and by definition of $>_{LP}$

W.l.o.g let be $k' = k$. We have to show that there is an $\psi \in Ot(s)$ with $dec_\psi(s) \gg_{EL} dec_v(t)$. We prove : $\psi = u$. If we could show that $args(s_k) >_{IRDS,left} args(t_l)$ and $\{s_k\} \gg_{IRDS} args(t_l)$, this would imply the desired aim since the other t_i's are smaller (w.r.t. $>_{EL}$) than s_k (because $top(s_k) = f_k \triangleright f_i = top(t_i)$).

Let s_k, t_l be $s_k = g_l(s_1',...,s_p')$ and $t_l = g_l(t_1',...,t_p')$. Then,

- $(\exists i \in [1,p])\ (\forall j < i)\ s_j' =_{PSDS} t_j' \quad \wedge \quad s_i' >_{PSDS} t_i'$

  because $args(s_k) >_{PSDS,left} args(t_l)$

  $\rightarrow$ $(\exists i \in [1,p])\ (\forall j < i)\ s_j' \sim t_j' \quad \wedge \quad dec(\{s_i'\}) \gg\gg_{LP} dec(\{t_i'\})$

  since $=_{PSDS}$ is equivalent to $\sim$ and by definition of the PSDS

  $\rightarrow$ $(\exists i \in [1,p])\ (\forall j < i)\ s_j' =_{IRDS} t_j' \quad \wedge \quad dec(\{s_i'\}) \gg\gg_{EL} dec(\{t_i'\})$

  since $\sim$ is equivalent to $=_{IRDS}$ and by the induction hypothesis (s_i' is a proper subterm of s)

  $\rightarrow$ $args(s_k) >_{IRDS,left} args(t_l)$

  by definition of the lexicographical extension of the IRDS

- $(\forall i \in [1,p])\ s_k >_{PSDS} t_i'$

  because $\{s_k\} \gg_{PSDS} args(t_l)$ and by definition of multiset orderings

  $\rightarrow$ $(\forall i \in [1,p])\ dec(\{s_k\}) \gg\gg_{LP} dec(\{t_i'\})$

  by definition of the PSDS

  $\rightarrow$ $(\forall i \in [1,p])\ dec(\{s_k\}) \gg\gg_{EL} dec(\{t_i'\})$

  by induction hypothesis

  $\rightarrow$ $\{s_k\} \gg_{IRDS} args(t_l)$

  by definition of the IRDS and its extension to multisets

iv) $f_k = g_l \quad \wedge \quad \tau(f_k) = \text{right}$:

analogous with iii)

The IRDS and the PSDS are not equivalent : The termination of the rule ① of the counter-examples (on page 52) can be proved with the IRDS but cannot be proved with the PSDS. ❑

6.25 Lemma Let $\triangleright$ be total : PSDS $\supset$ RPOS.

Proof : The proof of this statement is the same as that of lemma 6.18 except for exchanging $>_{RDOS}$ (resp. $>_{LD}$) for $>_{PSDS}$ (resp. $>_{LP}$). The inclusion is proper with the counter-example ④ (on the next page). ❑

6.26 Lemma Let $\triangleright$ be total :

i) RDO , PSD , IRD # RPOS
ii) PSD , PSDS # RDO , RDOS
iii) IRD # RDOS , PSDS
iv) KBOS , KBO # RPO , RPOS , PSD , PSDS , RDO , RDOS , IRD , IRDS

Proof : i) RDO , PSD , IRD $\not\in$ RPOS : ④
RPOS $\not\in$ RDO , PSD , IRD : ③

ii) PSD , PSDS $\not\in$ RDO , RDOS : ⑥
RDO , RDOS $\not\in$ PSD , PSDS : ①

iii) IRD $\not\in$ RDOS : ⑥
IRD $\not\in$ PSDS : ①
RDOS , PSDS $\not\in$ IRD : ③

iv) KBOS , KBO $\not\in$ RPO , RPOS , PSD , PSDS , RDO , RDOS , IRD , IRDS : ⑤
RPO , RPOS , PSD , PSDS , RDO , RDOS , IRD , IRDS $\not\in$ KBOS , KBO :
$x^2 \rightarrow x * x$, with $^2 \triangleright *$ ❑

6.27 Counter-examples

① $(-x - (-x)) - (-y - (-y)) \rightarrow (x - y) - (x - y)$
with $\tau(-) = \text{mult}$

② $x * ((-y) * y) \rightarrow (- y * y) * x$
with $* \triangleright -$ and $\tau(*) = \text{mult}$

③ $(x \wedge y) \wedge z \rightarrow x \wedge (y \wedge z)$
with $\tau(\wedge) = \text{left}$

④ $\neg x \supset (y \supset z) \rightarrow y \supset (x \vee z)$
with $\neg \triangleright \supset \triangleright \vee$

⑤ $(\neg x \supset y) \vee z \rightarrow (y \vee z) \vee x$
with $\varphi(\supset) > \varphi(\vee)$

⑥ and(not(not(x)),y,not(z)) $\rightarrow$ and(y,nand(x,z),x)
with not $\triangleright$ nand and $\tau(\text{and}) = \text{mult}$

Acknowledgement

There remains the pleasant duty to thank those who somehow co-operated in forming this paper: Jürgen Avenhaus , Werner Engeln, Roland Fettig, Manuela Gaß, Bernhard Gramlich, Klaus Madlener and Inger Sonntag.

Bibliography

[ABGM86] J. Avenhaus, B. Benninghofen, R. Göbel, K. Madlener : *TRSPEC: A Term Rewriting Based System for Algebraic Specifications*
Proceedings of the 8th International Conference on Automated Deduction, LNCS 230, Oxford, England, July 1986, pp. 665 - 667
A brief edition of [AGGMS87].

[AGGMS87] J. Avenhaus, R. Göbel, B. Gramlich, K. Madlener, J. Steinbach : *TRSPEC: A Term Rewriting Based System for Algebraic Specifications*
Proceedings of the 1st International Workshop on Conditional Term Rewriting Systems, LNCS 308, Orsay (Paris), France, July 8 - 10, 1987, pp. 245 - 248
The essential features of the TRSPEC-system together with an overview of future extensions are presented.

[Ai85] H. Ait-Kaci : *An algorithm for finding a minimal recursive path ordering*
R.A.I.R.O. Theoretical Informatics, Vol. 19, No. 4, 1985, pp. 359 - 382
Proposes an automatic inference method to compute a minimal precedence which induces a recursive path ordering on a given set of rules.

[Ba81] G. Bauer : *Zur Darstellung von Monoiden durch konfluente Regelsysteme*
Dissertation, Fachbereich Informatik, Universität Kaiserslautern, W. Germany, Februar 1981
On the representation of monoids with confluent rewrite systems. It contains an early version of the syllabled ordering on an alphabet with two letters.

[Bu85] B. Buchberger : *Basic features and development of the critical-pair/completion procedure*
Proceedings of the 1st International Conference on Rewriting Techniques and Applications, LNCS 202, Dijon, France, May 1985, pp. 1 - 45
Presents the history and basic features of the completion method for various applications.

[Ch84] G. Choque : *How to compute a complete set of minimal incrementations with the recursive decomposition ordering?*
Internal Report 84-R-056, Centre de recherche en informatique de Nancy, France, 1984
Shows how to construct a precedence during the comparison of two terms according to the RDO.

[De85] N. Dershowitz : *Termination*
Proceedings of the 1st International Conference on Rewriting Techniques and Applications, LNCS 202, Dijon, France, May 1985, pp. 180 - 224
Surveys methods for proving termination of term rewriting systems. Illustrations of the use of path orderings and other simplification orderings are given.

[De83] N. Dershowitz : *Well-founded orderings*
Technical Report ATR-83(8478)-3, Information Sciences Research Office, The Aerospace Corporation, El Segundo, California, May 1983
Makes a set of rules available for building well-founded orderings. Gives some remarks on the RPO, the RDO and polynomial orderings.

[De82] N. Dershowitz : *Orderings for term rewriting systems*
Journal of Theoretical Computer Science, Vol. 17, No. 3, March 1982, pp. 279 - 301
Contains the definition of simplification orderings and introduces the recursive path orderings.

[De80] N. Dershowitz : *On representing ordinals up to Γ_0*
Unpublished note, Department of Computer Science, University of Illinois, Urbana, Illinois, 1980
Extends the RPO to function symbols that are themselves terms and discusses the relation between the ordering and ordinals.

[De79] N. Dershowitz : *A note on simplification orderings*
Information Processing Letters, Vol. 9, No. 5, November 1979, pp. 212 - 215
The original definition of simplification orderings.

[DF85] D. Detlefs, R. Forgaard : *A procedure for automatically proving the termination of a set of rewrite rules*
Proceedings of the 1st International Conference on Rewriting Techniques and Applications, LNCS 202, Dijon, France, May 1985, pp. 255 - 270
Presents an algorithm that can automatically prove the termination of sets of rewrite rules by constructing the precedence over an RPO.

[DM79] N. Dershowitz, Z. Manna : *Proving termination with multiset orderings*
Communications of the Association for Computing Machinery, Vol. 22, No. 8, August 1979, pp. 465 - 476
Defines and elucidates multiset orderings.

[Fe88] R. Fettig : *Dynamische Multisetordnungen für Grundterme*
Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, W. Germany, Mai 1988
Incorporates the new multiset orderings of [St86] and [MS86] to term orderings on ground terms.

[Fo84] R. Forgaard : *A program for generating and analyzing term rewriting systems*
Master's Thesis, Massachusetts Institute of Technology, Laboratory for Computer Science, MIT/LCS/TR-343, September 1984
The design and implementation of REVE2 is described. It contains two important features : Automatic orderings (RPOS and closure ordering) and failure-resistant Knuth-Bendix.

[HL78] G. Huet, D. S. Lankford : *On the uniform halting problem for term rewriting systems*
Rapport Laboria 283, IRIA, Paris, INRIA Rocquencourt, France, March 1978
Shows that the uniform halting problem for term rewriting systems is undecidable in general.

[HO80] 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
Surveys applications of rewrite rules to equational reasoning and contains a section on termination.

[JL87] J.-P. Jouannaud, P. Lescanne : *Rewriting systems*
Technology and Science of Informatics, Vol. 6, No. 3, 1987, pp. 181 - 199
Describes the main achievement and most active areas of research in rewriting systems.

[JL82] J.-P. Jouannaud, P. Lescanne : *On multiset orderings*
Information Processing letters 15(2), September 1982, pp. 57 - 63
Proposes two well-founded orderings on multisets that extend the Dershowitz-Manna ordering (cf. [DM79]). These orderings do not have the property of the monotony.

[JLR82] J.-P. Jouannaud, P. Lescanne, F. Reinig : *Recursive decomposition ordering*
I.F.I.P. Working Conference on Formal Description of Programming Concepts II (D. Bjørner, ed.), North Holland, Garmisch Partenkirchen, W. Germany, 1982, pp. 331 - 348
Contains the definition of the RDO and proves that it is stronger than the RPO.

[KB70] D. E. Knuth, P. B. Bendix : *Simple word problems in universal algebras*
Computational problems in abstract algebra (J. Leech, ed.), Pergamon Press, 1970, pp. 263 - 297
Contains the completion procedure and the definition of the KBO.

[KL80] S. Kamin, J.-J. Lévy : *Attempts for generalizing the recursive path orderings*
Unpublished manuscript, Department of Computer Science, University of Illinois, Urbana, Illinois, February 1980
Defines the RPO with status and the semantic path orderings.

[KN85] M. S. Krishnamoorthy, P. Narendran : *Note on recursive path ordering*
Theoretical Computer Science 40, 1985, pp. 323 - 328
Proves that the problem of deciding whether a given pair of terms can be made RPO-comparable, by choosing a partial ordering on their function symbols, is NP-complete.

[KNS85] D. Kapur, P. Narendran, G. Sivakumar : *A path ordering for proving termination of term rewriting systems*
Proceedings of the 10th Colloquium on Trees in Algebra and Programming, LNCS 185, 1985, pp. 173 - 187
Presents a new ordering scheme based on the RPO and shows that it strictly contains the RPO.

[La77] D. S. Lankford : *Some approaches to equality for computational logic : A survey and assessment*
Report ATP-36, Automatic Theorem Proving Project, Departments of Mathematics and Computer Science, University of Texas, Austin, Texas 78712, Spring 1977
Reviews recent approaches to equality in computational logic. Particularly, it contains a section on rewrite rules and general aspects of termination.

[Le87] P. Lescanne : *On the recursive decomposition ordering with lexicographical status and other related orderings*
preprint, December 1987
Defines the closure ordering and compares it with the RPO and RDO with lexicographical status.

[Le86] P. Lescanne : *Divergence of the Knuth-Bendix completion procedure and termination orderings*
Bulletin of the European Association for Theoretical Computer Science, No. 30, 1986, pp. 80 - 83
Illustrates the influence of a termination ordering (e.g. RPO) on the behaviour of the Knuth-Bendix completion procedure.

[Le84] P. Lescanne : *Uniform termination of term rewriting systems - the recursive decomposition ordering with status*
Proceedings of the 9th Colloquium on Trees in Algebra and Programming (B. Courcelle, ed.), Cambridge University Press, Bordeaux, France, 1984, pp. 182 - 194
Presents a recursive decomposition ordering with status which is different from our RDOS.

[Le83a] P. Lescanne : *How to prove termination? An approach to the implementation of a new recursive decomposition ordering*
Proceedings of an NSF Workshop on the Rewrite Rule Laboratory (Guttag, Kapur, Musser, eds.), General Electric Research and Development Center Report 84GEN008, September 1983, pp. 109 - 121
It is an earlier version of [Le84].

[Le83b] P. Lescanne : *Computer experiments with the REVE term rewriting system generator*
Internal Report 83-R-037, Centre de recherche en informatique de Nancy, France, 1983
Describes REVE, a term rewriting system generator. An incremental method for proving the termination of rewrite systems is presented. It is based on the recursive decomposition ordering.

[Le82] P. Lescanne : *Some properties of decomposition ordering, a simplification ordering to prove termination of rewriting systems*
R.A.I.R.O. Theoretical Informatics, Vol. 16, No. 4, 1982, pp. 331 - 347
An earlier version of the RDO (based on the concepts of [Le81a] and [Le81b]) is presented and proved to be equivalent to the RPO.

[Le81a] P. Lescanne : *Two implementations of the recursive path ordering on monadic terms*
Proceedings of the 19th Allerton Conference on Communication, Control and Computing, Allerton House, Monticello, Illinois, September 1981, pp. 634 - 643
Discusses an efficient implementation of the RPO on monadic terms. Contains the definition of the RDO on monadic terms.

[Le81b] P. Lescanne : *Decomposition ordering as a tool to prove the termination of rewriting systems*
Proceedings of the 7th International Joint Conference on Artificial Intelligence, Vancouver, Canada, August 1981, pp. 548 - 550
Describes an earlier version of the RDO on left-weighted terms in case of total precedences.

[Ma87] U. Martin : *How to choose the weights in the Knuth-Bendix ordering*
Proceedings of the 2nd International Conference on Rewriting Techniques and Applications, LNCS 256, Bordeaux, France, May 25 - 27, 1987, pp. 42 - 53
Presents an algorithm based on the simplex algorithm, for determining whether or not a set of rules can be ordered by a KBO.

[MS86] J. Müller, J. Steinbach : *Topologische Multisetordnungen*
Proceedings of the 10th German Workshop and 2nd Austrian Conference on Artificial Intelligence, Ottenstein, Austria, Informatik Fachberichte 124, September 1986, pp. 254 - 264
This paper introduces multiset orderings that are based on the ideas of [JL82].

[Ok86] M. Okada : *Ackermann's ordering and its relationship with ordering systems in term rewriting theory*
Gives an example of a link between a proof theoretic ordering in logic and term rewriting orderings. It considers relationships with the RPO and the semantic path orderings.

[Pe81] A. Pettorossi : *Comparing and putting together recursive path ordering, simplification orderings and non-ascending property for termination proofs of term rewriting systems*
Proceedings of the 8th EATCS International Colloquium on Automata, Languages and Programming, LNCS 115, Acre, Israel, July 1981, pp. 432 - 447
Defines a sufficient condition for proving termination of rewrite rules and compares it with other known methods (e.g. the RPO).

[Pl85] D. A. Plaisted : *The undecidability of self-embedding for term rewriting systems*
Information Processing Letters 20, February 1985, pp. 61 - 64
Proves that it is undecidable whether a term rewriting system is non-self-embedding.

[Pl78a] 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
Defines the path of subterms ordering and proves that it is well-founded.

[Pl78b] D. A. Plaisted : *Well-founded orderings for proving termination of systems of rewrite rules*
Report UIUCDCS-R-78-932, Department of Computer Science, University of Illinois, Urbana, IL, July 1978
Defines the simple path ordering (an earlier version of [Pl78a]).

[Re81] F. Reinig : *Les ordres de décomposition : un outil incrémental pour prouver la terminaison finie de systèmes de réécriture de termes*
Thèse présentée pour l'obtention du grade de docteur de 3ème cycle en informatique, Université de Nancy I & Centre de recherche en informatique de Nancy, France, Octobre 1981
Describes the RDO (generalized version of the 'RDO' in [Le81a] and [Le81b]) and proves its well-foundedness. Moreover, it shows that it is stronger than the RPO (in French).

[RJ81] F. Reinig, J.-P. Jouannaud : *Decomposition orderings , a new family of recursive simplification orderings*
Report CRIN 81-R-040, Centre de recherche en informatique de Nancy, France, June 1981
Contains the same results as [Re81] in English and is an earlier version of [JLR82].

[Ru87] M. Rusinowitch : *Path of subterms ordering and recursive decomposition ordering revisited*
Journal of Symbolic Computation 3 (1 & 2), 1987, pp. 117 - 131
Defines the IRD and compares it with the RPO, RDO and KNS relative to an underlying partial precedence.

[Si87] C. C. Sims : *Verifying nilpotence*
Journal of Symbolic Computation 3, 1987, pp. 231 - 247
Describes a new procedure based on string rewriting rules for verifying that a finitely presented group is nilpotent. Contains the definition of the collected (or syllabled) ordering.

[St88] J. Steinbach : *Comparison of simplification orderings*
SEKI REPORT SR-88-02, Artificial Intelligence Laboratories, Department of Computer Science, University of Kaiserslautern, W. Germany, February 1988
Defines the PSD and extends the comparison of [Ru87] to total precedences and to monadic terms.

[St86] J. Steinbach : *Ordnungen für Term-Ersetzungssysteme*
Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, W. Germany, Juni 1986
Presents several new multiset orderings. Moreover, it illustrates some of the most popular simplification orderings. Finally, the paper extends the comparison of [Ru87].

[Wa86] E. Wagner : *Strategien für den Knuth-Bendix Algorithmus*
Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, W. Germany, Juni 1986
Contains the description of the COMTES-system.

[Wi88] D. Wissmann : *Applying rewriting techniques to groups with power-commutation-presentations*
to appear in Proceedings of the 1988 International Symposium on Symbolic and Algebraic Computation, Rome, July 4 - 8, 1988
A modified version of the Knuth-Bendix completion procedure is given which transforms string rewriting systems related to special groups into equivalent canonical systems of the same type. Contains the definition of the collected (or syllabled) ordering.

[WS84] E. Wagner, J. Steinbach : *Implementierung einer Grundversion des Knuth-Bendix Algorithmus*
Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, W. Germany, Sommer 1984
Describes the theoretical aspects and the implementation of a straightforward completion algorithm (a preliminary version of COMTES) for term rewriting systems.