Fachbereich Informatik
Universität Kaiserslautern
D-67663 Kaiserslautern

SEKI - REPORT

On the Complexity of Simplification Orderings

Joachim Steinbach

SEKI Report SR-93-18 (SFB)

On the Complexity of Simplification Orderings*

Joachim STEINBACH
Universität Kaiserslautern
FB Informatik
Postfach 3049
67653 Kaiserslautern
Germany
e-mail: steinba@informatik.uni-kl.de

December 17, 1993

Abstract

Various methods for proving the termination of term rewriting systems have been suggested. Most of them are based on the notion of simplification ordering. In this paper, the theoretical time complexities (of the worst cases) of a collection of well-known simplification orderings will be presented.

1 Introduction and Summary

Term rewriting systems (TRSs, for short) provide a powerful tool for expressing non-deterministic computations and as a result they have been widely used as, for example, in theorem provers. Moreover, they can usefully be applied in many other areas of computer science and mathematics such as abstract data type specifications and program verification. A main requirement of TRSs is expressed by the termination property.

There exist various methods of proving the termination of TRSs. Most of these are based on *reduction orderings* which are *well-founded*, *compatible with the structure of terms*[1] and *stable with respect to* (w.r.t., for short) *substitutions*. The notion of reduction ordering leads to the following description of termination of rewriting systems: A TRS $\mathcal{R}$ terminates if, and only if, there exists a reduction ordering $\succ$ such that $l \succ r$ for each rule $l \rightarrow r$ of $\mathcal{R}$. With *simplification orderings* we refer to a special class of reduction orderings that require the so-called *subterm property* (see, for example, [Dershowitz, 1987]).

*This research was supported by the Deutsche Forschungsgemeinschaft, SFB 314 (D4-Projekt).
[1] also called *monotonic*

After briefly recapitulating the most essential notions used in connection with TRSs and termination, we will give (in section 3) the definitions of well-known simplification orderings including the recursive path ordering with status (RPOS), the path of subterms ordering (PSO), the path ordering with status of Kapur & Narendran & Sivakumar (KNSS), the improved recursive decomposition ordering with status (IRDS), the path of subterms ordering with status on decompositions (PSDS), the Knuth-Bendix ordering with status (KBOS) and the polynomial ordering (POL). In section 4, the time complexities of these orderings will be studied. More formally, the time for comparing two terms w.r.t. a *given* ordering[2] will be presented. The following (worst case) complexities will be proved[3] (s and t are the terms to be compared while $|.|$ represents the number of symbols occurring in a term):

IRDS	KBOS	KNSS	PSDS	PSO	RPO
$O(\|s\|^4 \cdot \|t\|^4)$	$O(\|s\| \cdot \|t\|)$	$O(\|s\|^3 \cdot \|t\|^3)$	$O(\|s\|^4 \cdot \|t\|^4)$	$O(\|s\|^3 \cdot \|t\|^3)$	$O(\|s\| \cdot \|t\|)$

2 Notations

We assume familiarity with the standard definitions[4] of the set of *function symbols* (or operators) $\mathcal{F}$ and their *arities* $\mathcal{A}r$, the set of *variables* $\mathcal{X}$, the set of *terms* $\mathcal{T}(\mathcal{F},\mathcal{X})$ and the set of (*terminal*) *occurrences* $\mathcal{P}os(t)$ ($\mathcal{P}os^*(t)$) of a term t. Furthermore, the *leading function symbol* and the *tuple of the arguments* of a term t are referred to by $\mathcal{H}ead(t)$ and $\mathcal{A}rgs(t)$, respectively. The *size*[5] of a term t (the cardinality of a set $\mathcal{M}$) is denoted by $|t|$ ($|\mathcal{M}|$). $||t||$ stands for the *depth* of t, i.e. (i) $||\Delta|| = 1$ if $\Delta \in \mathcal{X}$ or $\Delta \in \mathcal{F} \wedge \mathcal{A}r(\Delta) = 0$ and (ii) $||f(t_1,\ldots,t_n)|| = 1 + \max\{||t_i|| \mid i = 1,\ldots,n\}$.

The *lexicographic* (*multiset*) *extension* of an ordering $\succ$ to tuples (multisets) of elements is denoted by $\succ^{lex}$ ($\succ^{mul}$).[6] In order to combine $\succ^{lex}$ and $\succ^{mul}$, each operator f has a so-called *status* τ that determines the order according to which the arguments of f are compared ([Kamin and Lévy, 1980]). Formally speaking, τ maps $\mathcal{F}$ into the set {*mul,left,right*}.[7] The orderings of this paper (except POL) use a congruence $\sim$ depending on $\mathcal{F}$ and τ via $f(s_1,\ldots,s_m) \sim g(t_1,\ldots,t_n)$ if, and only if, $f = g$ and $m = n$ and (i) $\tau(f) = mul$ and there exists 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) $\tau(f) \neq mul \wedge s_i \sim t_i$, for all $i \in [1,n]$.

For the sake of compactness, the representations of figure 1 will be used for formal descriptions of orderings. Case (i) states that $s \succ t$ if, and only if, at least one of the

[2]i.e. the ordering is completely defined by its parameters (a precedence, a weight function, an interpretation and a status function)

[3]The complexity of RPO is proved in [Krishnamoorthy and Narendran, 1985]. In [Kapur *et al.*, 1985], it is shown that $O(|s|^5 \cdot |t|^5)$ is an upper bound for KNSS.

[4]see, for example, [Huet and Oppen, 1979] and [Dershowitz, 1987]

[5]i.e. the number of symbols

[6]$(m_1,m_2,\ldots,m_p) \succ^{lex} (n_1,n_2,\ldots,n_q)$ iff (i) $p > 0 \wedge q = 0$ or (ii) $m_1 \succ n_1$ or (iii) $m_1 \sim n_1 \wedge (m_2,\ldots,n_p) \succ^{lex} (n_2,\ldots,n_q)$. $M_1 \succ^{mul} M_2$ iff $M_1 \neq M_2 \wedge (\forall y \in M_2 \backslash M_1)(\exists x \in M_1 \backslash M_2)\ x \succ y$.

[7]The arguments of f will be compared as multisets (lexicographically from left to right or vice versa) if $\tau(f) =$ mul ($\tau(f) =$ left or $\tau(f) =$ right).

conditions $cond_i$ is satisfied. Case (ii) stands for the *lexicographic* evaluation, i.e. $s \succ t$ if, and only if, $s \succ_1 t$ or $(s \sim_1 t \wedge s \succ_2 t)$, and so on.

(i)	$s \succ t$ iff	$1)$	$cond_1$	(ii)	$s \succ t$ iff	—	$s \succ_1 t$
			$\vdots$				$\vdots$
		$n)$	$cond_n$			—	$s \succ_n t$

Figure 1

Most of the orderings presented in this paper are based on an ordering $\succ$ on the operators, called *precedence.* We use $\succ_{ord}$ for denoting the ordering *ord* using the precedence $\succ$.

3 Definitions of the Orderings

This section gives a description of the orderings to be examined. The formal definitions can be found in figures 2 and 3.

3.1 Path and Decomposition Orderings

The method of comparing two terms w.r.t. the *recursive path ordering with status* (RPOS, for short) depends on their leading function symbols ([Dershowitz, 1982], [Kamin and Lévy, 1980], see figure 2). The relationship between these operators w.r.t. the precedence is responsible for decreasing one (or both) of the terms in the recursive definition of the RPOS. If one of the terms is *'empty'* (i.e. totally decreased), then the other one must be greater.

In order to define other path 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:

- $\mathcal{P}ath_\Lambda(\Delta) = \Delta$ if Δ is a constant or a variable
- $\mathcal{P}ath_{i.u}(f(t_1,\ldots,t_n)) = f(t_1,\ldots,t_n); \mathcal{P}ath_u(t_i)$ if $i \in [1,n]$ and $u \in \mathcal{P}os^*(t_i)$

Moreover,

$$\bullet\ \mathcal{P}ath(\{t_1,\ldots,t_n\}) = \{\mathcal{P}ath_u(t_i) \mid i \in [1,n], u \in \mathcal{P}os^*(t_i)\}$$

is the multiset of all paths of the specified terms $t_1,\ldots,t_n$. A path is enclosed in square brackets. The set $\{t_1,\ldots,t_n\}$ of all the terms occurring in a path $P = [t_1;\ldots;t_n]$ is denoted by $\mathcal{S}et(P)$. The *path of subterms* and the *path of superterms* of a path relative to a term t_i are defined as follows:

- $\mathcal{S}ub([t_1;\ldots;t_i;\ldots;t_n],t_i) = [t_{i+1};\ldots;t_n]$
- $\mathcal{S}up([t_1;\ldots;t_i;\ldots;t_n],t_i) = [t_1;\ldots;t_{i-1}]$

$s \succ_{\text{RPOS}} t$

iff
1) $\mathcal{H}ead(s) \succ \mathcal{H}ead(t) \;\wedge\; \{s\} \succ^{mul}_{\text{RPOS}} \mathcal{A}rgs(t)$
2) $\mathcal{H}ead(s) = \mathcal{H}ead(t) \;\wedge\; \tau(\mathcal{H}ead(s)) = mul \;\wedge\; \mathcal{A}rgs(s) \succ^{mul}_{\text{RPOS}} \mathcal{A}rgs(t)$
3) $\mathcal{H}ead(s) = \mathcal{H}ead(t) \;\wedge\; \tau(\mathcal{H}ead(s)) \neq mul \;\wedge\; \mathcal{A}rgs(s) \succ^{\tau(\mathcal{H}ead(s))}_{\text{RPOS}} \mathcal{A}rgs(t)$
$\wedge\; \{s\} \succ^{mul}_{\text{RPOS}} \mathcal{A}rgs(t)$
4) $\mathcal{A}rgs(s) \succsim^{mul}_{\text{RPOS}} \{t\}$

$s \succ_{\text{PSO}} t$ iff $\mathcal{P}ath(\{s\}) \succ^{mul}_{PO} \mathcal{P}ath(\{t\})$

with $p \succ_{PO} q$ iff $\mathcal{S}et(p) \succ^{mul}_{T} \mathcal{S}et(q)$

with $u \succ_T v$ iff
- $\mathcal{H}ead(u) \succ \mathcal{H}ead(v)$
- $\mathcal{P}ath(\mathcal{A}rgs(u)) \succ^{mul}_{PO} \mathcal{P}ath(\mathcal{A}rgs(v))$

$s \succ_{\text{KNSS}} t$ iff $\mathcal{P}ath(\{s\}) \succ^{mul}_{LK} \mathcal{P}ath(\{t\})$

with $p \succ_{LK} q$ iff $(\forall t' \in q)(\exists s' \in p)\; s' \succ_{LT} t'$

with $p \ni u \succ_{LT} v \in q$

iff
1) $\mathcal{H}ead(u) \succ \mathcal{H}ead(v)$
2) $\mathcal{H}ead(u) = \mathcal{H}ead(v) \;\wedge\; \tau(\mathcal{H}ead(u)) = mul \;\wedge$
   - $\mathcal{S}ub(p, u) \succ_{LK} \mathcal{S}ub(q, v)$
   - $\mathcal{P}ath(\mathcal{A}rgs(u)) \succ^{mul}_{LK} \mathcal{P}ath(\mathcal{A}rgs(v))$
   - $\mathcal{S}up(p, u) \succ_{LK} \mathcal{S}up(q, v)$
3) $\mathcal{H}ead(u) = \mathcal{H}ead(v) \;\wedge\; \tau(\mathcal{H}ead(u)) \neq mul \;\wedge$
   - $\mathcal{A}rgs(u) \succ^{\tau(\mathcal{H}ead(u))}_{\text{KNSS}} \mathcal{A}rgs(v)$
   - $\mathcal{S}up(p, u) \succ_{LK} \mathcal{S}up(q, v)$

$s \succ_{\text{IRDS}} t$ iff $\mathcal{D}ec(\{s\}) \, (\succ^{mul}_{EL})^{mul} \, \mathcal{D}ec(\{t\})$

with $\mathcal{D}ec_p(s') \ni u \succ_{EL} v \in \mathcal{D}ec_q(t')$

iff
1) $\mathcal{H}ead(u) \succ \mathcal{H}ead(v)$
2) $\mathcal{H}ead(u) = \mathcal{H}ead(v) \;\wedge\; \tau(\mathcal{H}ead(u)) = mul \;\wedge$
   - $\mathcal{S}ub(\mathcal{D}ec_p(s'), u) \succ^{mul}_{EL} \mathcal{S}ub(\mathcal{D}ec_q(t'), v)$
   - $\mathcal{D}ec(\mathcal{A}rgs(u)) \, (\succ^{mul}_{EL})^{mul} \, \mathcal{D}ec(\mathcal{A}rgs(v))$
3) $\mathcal{H}ead(u) = \mathcal{H}ead(v) \;\wedge\; \tau(\mathcal{H}ead(u)) \neq mul \;\wedge$
   $\mathcal{A}rgs(u) \succ^{\tau(\mathcal{H}ead(u))}_{\text{IRDS}} \mathcal{A}rgs(v)$

$s \succ_{\text{PSDS}} t$ iff $\mathcal{D}ec(\{s\}) \, (\succ^{mul}_{LP})^{mul} \, \mathcal{D}ec(\{t\})$

with $u \succ_{LP} v$ iff
1) $\mathcal{H}ead(u) \succ \mathcal{H}ead(v)$
2) $\mathcal{H}ead(u) = \mathcal{H}ead(v) \;\wedge\; \tau(\mathcal{H}ead(u)) = mul \;\wedge$
   $\mathcal{D}ec(\mathcal{A}rgs(u)) \, (\succ^{mul}_{LP})^{mul} \, \mathcal{D}ec(\mathcal{A}rgs(v))$
3) $\mathcal{H}ead(u) = \mathcal{H}ead(v) \;\wedge\; \tau(\mathcal{H}ead(u)) \neq mul \;\wedge$
   $\mathcal{A}rgs(u) \succ^{\tau(\mathcal{H}ead(u))}_{\text{PSDS}} \mathcal{A}rgs(v)$

Figure 2: Path and Decomposition Orderings

Consider the following example: $t = (x+y)*z$ implies $\mathcal{P}ath_{12}(t) = [t; x+y; y], \mathcal{P}ath(\{t\}) = \{\mathcal{P}ath_{11}(t), \mathcal{P}ath_{12}(t), \mathcal{P}ath_2(t)\}, \mathcal{S}ub(\mathcal{P}ath_2(t), z) = []$ and $\mathcal{S}up(\mathcal{P}ath_{11}(t), x+y) = [t]$.

Plaisted's *path of subterms ordering* (PSO, for short) is a predecessor of the recursive path ordering (RPO) of Dershowitz and compares two terms by comparing all their paths ([Plaisted, 1978]). A slightly modified version (which is equivalent to the original one) of Rusinowitch ([Rusinowitch, 1987], [Steinbach, 1989]) is given in figure 2.

A further ordering based on paths has been devised by *Kapur*, *Narendran* and *Sivakumar* ([Kapur *et al.*, 1985]). It is called KNSS (KNS with status, see figure 2) and extends RPOS. In [Kapur *et al.*, 1985], it has been stated that the ordering relation $p \succ_{LK} q$ between two paths implies the ordering relation of two paths $p.p' \succ_{LK} q.p'$, where p and q have been extended on the right-hand side by a path p'.

The specific definitions of the decomposition orderings require some additional notations. The set $\mathcal{S}et(P)$ of a path P is called *path-decomposition* and its abbreviation is

- $\mathcal{D}ec_u(t) = \mathcal{S}et(\mathcal{P}ath_u(t))$.

An element (i.e. a term) of a path-decomposition is called an *elementary decomposition*. Analogous to paths, the *decomposition*

- $\mathcal{D}ec(\{t_1, \ldots, t_n\}) = \{\mathcal{D}ec_u(t_i) \mid i \in [1,n], u \in \mathcal{P}os^*(t_i)\}$

represents the multiset of all path-decompositions[8] of the terms $t_1, \ldots, t_n$ as well as $\mathcal{S}ub$ and $\mathcal{S}up$ denote subsets of path-decompositions.[9]

As in the case of KNSS, the first recursive decomposition ordering has been developed from RPO.[10] One of the important differences to RPO is the fact that a comparison is stopped as soon as incomparable operators are to be compared. We will present an extension (called IRD) of the original decomposition ordering, developed by *Rusinowitch* ([Rusinowitch, 1987]). We have incorporated status ([Steinbach, 1989]) to IRD (IRDS, for short), so that it is equivalent to KNSS (see figure 2). Moreover, our decomposition orderings employ a different concept of decomposition: We use terms instead of triples (see [Steinbach, 1989]). A term s is greater than a term t (w.r.t. IRDS) if the decomposition of s is greater than the decomposition of t. The ordering $(\succ_{EL}^{mul})^{mul}$ on these multisets is an extension of the basic ordering on terms ($\succ_{EL}$) to multisets of multisets.

A further ordering based on decompositions results from PSO. We have succeeded in redefining this path ordering such that the resulting ordering, called PSD, provides a

[8]Note that a path-decomposition is a *set* of terms, whereas a decomposition is a *multiset* of path-decompositions.

[9]For example, let $t = (x+y)*z$. Then $\mathcal{D}ec_{11}(t) = \{t, x+y, x\}$, $\mathcal{D}ec(\{t\}) = \{\{t, x+y, x\}, \{t, x+y, y\}, \{t, z\}\}$, $\mathcal{S}ub(\mathcal{D}ec_2(t), z) = \emptyset$ and $\mathcal{S}up(\mathcal{D}ec_{11}(t), x+y) = \{t\}$.

[10]The idea of decomposition orderings goes back to Lescanne, Jouannaud and Reinig ([Jouannaud *et al.*, 1982]).

much simpler method of using decompositions (see [Steinbach, 1988]). PSD has another advantage over PSO: The combination with the concept of status (PSDS, for short) is much easier ([Steinbach, 1989], see figure 2). The essential difference between PSDS and IRDS lies in the method by which a comparison is processed: If the leading function symbols of the terms to be compared are identical, IRDS chooses only *one* subterm while PSDS proceeds by simultaneously considering the multiset of the decompositions of *all* subterms.

3.2 Knuth-Bendix Orderings with Status

The *ordering of Knuth and Bendix* (KBO, for short) assigns natural (or possibly real) numbers to function symbols. The value or weight of a term is obtained by adding the numbers of the operators it contains. Two terms are compared by comparing their weights, and, if the weights are equal, by lexicographically comparing their subterms (see [Knuth and Bendix, 1967]). In [Lankford, 1979], a generalization of this ordering is described: The comparison of terms depends on polynomials instead of weights (see subsection 3.3).

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

- $\varphi(c) \geq \varphi_0$ if c is a constant and
- $\varphi(f) = 0$ for one unary operator f, at most: f has to be maximal w.r.t. $\succ$.

Now we extend the weight function to terms. For any term $t = g(t_1, \ldots, t_n)$ let $\varphi(t) = \varphi(g) + \sum \varphi(t_i)$.

KBOS ([Steinbach, 1989], see figure 3) is an extended version of the original KBO which does not consider a status function.

3.3 Polynomial Orderings

Polynomial orderings (POL, for short) have been developed by Manna & Ness ([Manna and Ness, 1970]) and Lankford ([Lankford, 1975], [Lankford, 1979]). Terms are compared w.r.t. POL (i) by mapping them into polynomials over $\mathbb{N}$ and (ii) by comparing the polynomials w.r.t. $>_{\mathbb{N}}$ where natural numbers are substituted for the variables (see figure 3).

The set of all *polynomials* over a set $\{x_1, \ldots, x_n\}$ of n distinct *variables* and with *coefficients* in $\mathbb{N}$ is denoted by $\mathbb{N}[x_1, \ldots, x_n]$. A polynomial is composed of a sum of *monomials*[11] of the form $\alpha_{r_1 \ldots r_n} \cdot x_1^{r_1} \cdot \ldots \cdot x_n^{r_n}$. A polynomial $\sum \alpha_{r_1 \ldots r_n} \cdot x_1^{r_1} \cdot \ldots \cdot x_n^{r_n}$ based on n distinct variables is represented by $p(x_1, \ldots, x_n)$. Since every ground polynomial is equal to a natural number, we identify the set of ground polynomials with $\mathbb{N}$. A polynomial $p = \sum_{r_1 \ldots r_n} \alpha_{r_1 \ldots r_n} x_1^{r_1} \cdot \ldots \cdot x_n^{r_n} \in \mathbb{N}[x_1, \ldots, x_n]$ possesses a *strict arity*

[11] We use $\alpha_{r_1 \ldots r_n}$ for referring to the exponents of the variables (e.g., $\alpha_{210}x^2y + \alpha_{101}xz$).

$$s \succ_{\text{KBOS}} t \text{ iff } (\forall x \in \mathcal{X})\, \#_x(s) \geq \#_x(t) \wedge \begin{array}{ll} 1) & s = f(t) \\ 2) & -\varphi(s) > \varphi(t) \\ & -\mathcal{H}ead(s) \succ \mathcal{H}ead(t) \\ & -\mathcal{A}rgs(s) \succ_{\text{KBOS}}^{\tau(\mathcal{H}ead(s))} \mathcal{A}rgs(t) \end{array}$$

$$s \succ_{\text{POL}} t \quad \text{iff} \quad [s] \sqsupset [t]$$

$$\text{with} \quad p \sqsupset q \text{ iff } (\forall X_i \geq \mu)\, p(X_1, \ldots, X_n) > q(X_1, \ldots, X_n)$$

$$\text{where} \quad \mu = min\{[c]() \mid c \in \mathcal{F} \wedge \mathcal{A}r(c) = 0\}^{12}$$

Figure 3: Knuth-Bendix and Polynomial Orderings

m ($\leq n$) if there occur m variables in p that differ by pairs, i.e. for every x_i there is a monomial in p containing x_i with a non-zero coefficient.

A *polynomial interpretation* $[.]$: $\mathcal{F} \cup \mathcal{X} \mapsto \mathbb{N}[\mathcal{V}]$ over $\mathbb{N}$ assigns a polynomial $p \in \mathbb{N}[x_1, \ldots, x_n]$ of strict arity n to each n-ary function symbol and a variable X over $\mathbb{N}$ to each variable $x \in \mathcal{X}$ where $\mathcal{V}$ is a finite set of (polynomial) variables over $\mathbb{N}$. This mapping can be extended to $[.]$: $\mathcal{T}(\mathcal{F}, \mathcal{X}) \mapsto \mathbb{N}[\mathcal{V}]$ by defining $[f(t_1, \ldots, t_n)] = [f]([t_1], \ldots, [t_n])$.

4 Complexity of the Orderings

In this section, we study time complexities of the orderings presented in the former section. The following lemmas consider upper bounds which are not necessary be strict. Obviously, these time complexities depend on the formal definitions of the orderings. Thus, two equivalent (w.r.t. the power) orderings can possess different complexities. Note that the powers of KNSS and IRDS are equivalent. However, their time complexities differ (see lemmas 4.1 and 4.3). Analogously, PSO and PSDS are equivalent if lexicographic status is excluded (see lemmas 4.4 and 4.5). One of the most interesting results is the fact that decomposition orderings are more time-consuming than path orderings. This has also been confirmed by various examples of sets of equations oriented with the help of our completion environment COMTES ([Avenhaus *et al.*, 1989]).

To prove any assertion on the time complexity of polynomial orderings is somewhat difficult. It depends (i) on the method for proving the positiveness of polynomials and (ii) on the interpretations of the operators. Thus, for example, the time complexity of the approach of [BenCherifa and Lescanne, 1987] cannot be determined since it is no

[12] If $\mathcal{F}$ does not contain any constant symbol, μ can be arbitrarily chosen.

decision procedure. However, the polynomial ordering's *empirical* time complexity is higher than that of the other orderings.

The technique of proving (most of) the lemmas is based on a method similar to dynamic programming (introduced in [Kapur *et al.*, 1985] for the first time). All orderings except POL are recursively defined. Therefore, we assume that substructures of elements are already compared by simultaneously storing the results in an array that can easily be accessed. Then it remains to compute the additional time required to compare two elements under these assumptions.

Lemma 4.1 *Given two terms s, t, a precedence $\succ$ and a status function τ, $s \succ_{\text{IRDS}} t$ can be determined in time $O(|s|^4 \cdot |t|^4)$.*

Proof: This upper bound will be proved by a method similar to dynamic programming. In order to compare s and t w.r.t. IRDS, all path-decompositions must be compared w.r.t. $\succ_{EL}^{mul}$. Thus, the time for comparing s and t w.r.t. IRDS is not greater than

$$O(|\mathcal{P}os^*(s)| \cdot ||s|| \cdot |\mathcal{P}os^*(t)| \cdot ||t|| \cdot \text{TEL}(s,t))$$

where TEL provides the time for comparing two terms w.r.t. $\succ_{EL}$. We will prove that $\text{TEL}(u,v) = O(|\mathcal{P}os^*(u)| \cdot ||u|| \cdot |\mathcal{P}os^*(v)| \cdot ||v||)$ holds. This implies the time for comparing s and t w.r.t. IRDS to be

$$O(|\mathcal{P}os^*(s)| \cdot ||s|| \cdot |\mathcal{P}os^*(t)| \cdot ||t|| \cdot |\mathcal{P}os^*(s)| \cdot ||s|| \cdot |\mathcal{P}os^*(t)| \cdot ||t||)$$

which concludes the proof since $|\mathcal{P}os^*(u)| \leq |u|$ and $||u|| \leq |u|$ for all terms u.

It remains to be proved that two terms u and v can be compared w.r.t. $\succ_{EL}$ in time $\text{TEL}(u,v) = O(|\mathcal{P}os^*(u)| \cdot ||u|| \cdot |\mathcal{P}os^*(v)| \cdot ||v||)$. Assume that all proper subterms of u have already been compared w.r.t. $\succ_{EL}$ with all proper subterms of v. Assume further that the results are stored in a 2-dimensional array $\mathcal{A}$ that can be accessed easily: $\mathcal{A}(p,q)$ provides the result of comparing $u|_p$ and $v|_q$ ($p \neq \Lambda \neq q$). Now, we have to determine the additional time required to compare u and v. We consider the worst case (i.e. $\mathcal{H}ead(u) = \mathcal{H}ead(v) \wedge \tau(\mathcal{H}ead(u)) = mul$): In order to compare

- $\mathcal{S}ub(\mathcal{D}ec_p(s'), u)$ and $\mathcal{S}ub(\mathcal{D}ec_q(t'), v)$, $||u|| \cdot ||v||$ comparisons w.r.t. $\succ_{EL}$ are needed
- $\mathcal{D}ec(\mathcal{A}rgs(u))$ and $\mathcal{D}ec(\mathcal{A}rgs(v))$, $|\mathcal{P}os^*(u)| \cdot |\mathcal{P}os^*(v)| \cdot ||u|| \cdot ||v||$ comparisons are needed since $|\mathcal{P}os^*(\mathcal{A}rgs(t))| = |\mathcal{P}os^*(t)|$ and each path-decomposition of $\mathcal{D}ec(\mathcal{A}rgs(t))$ contains at most $||t||$ elements.

These observations, together with $|\mathcal{P}os^*(t)| \leq |t|$ and $||t|| \leq |t|$ (for all terms t), imply the complexity of $\succ_{EL}$ as it is stated above. □

Lemma 4.2 *Given two terms s, t, a precedence $\succ$, a weight function φ and a status function τ, $s \succ_{\text{KBOS}} t$ can be determined in time $O(|s| \cdot |t|)$.*

Proof: We use induction on the size of the terms.

Basis step: $|s| = |t| = 1$. obvious

Induction step: Let $s = f(s_1, \ldots, s_m)$ and $t = g(t_1, \ldots, t_n)$.

1) $\varphi(s) > \varphi(t)$: We need time $O(|s| + |t|)$ for computing the weights of the terms.

2) $\varphi(s) = \varphi(t) \wedge f \succ g$: analogous to 1)

3) $\varphi(s) = \varphi(t) \wedge f = g \wedge \tau(f) = mul \wedge \{s_1, \ldots, s_m\} \succ_{\text{KBOS}}^{mul} \{t_1, \ldots, t_n\}$: While comparing the two multisets, each s_i, $1 \le i \le m$, will have to be compared with each t_j, $1 \le j \le n$, in the worst case. This will take at most

$$\sum_{i=1}^{m}(\sum_{j=1}^{n}(|s_i| \cdot |t_j|)) < \sum_{i=1}^{m}(|s_i| + 1) \cdot \sum_{j=1}^{n}(|t_j| + 1) = (|s| + m - 1) \cdot (|t| + n - 1)$$

comparisons.

4) $\varphi(s) = \varphi(t) \wedge f = g \wedge \tau(f) \neq mul \wedge (s_1, \ldots, s_m) \succ_{\text{KBOS}}^{\tau(f)} (t_1, \ldots, t_n)$: Then at most

$$\sum_{i=1}^{min\{m,n\}} (|s_i| \cdot |t_i|) \le \sum_{i=1}^{min\{m,n\}} (|s_i| + 1) \cdot \sum_{i=1}^{min\{m,n\}} (|t_i| + 1) = (|s| + \min\{m,n\} - 1) \cdot (|t| + \min\{m,n\} - 1)$$

comparisons have to be done. □

Lemma 4.3 *Given two terms s, t, a precedence $\succ$ and a status function τ, $s \succ_{\text{KNSS}} t$ can be determined in time $O(|s|^3 \cdot |t|^3)$.*

Proof: This upper bound will be proved similarly to that of lemma 4.1. In order to compare s and t w.r.t. KNSS, all paths of s and t must be compared w.r.t. $\succ_{LK}$. Thus, the time for comparing s and t w.r.t. KNSS is not greater than

$$O(|\mathcal{P}os^*(s)| \cdot |\mathcal{P}os^*(t)| \cdot \text{TLK}(P, Q))$$

where P (Q) is the maximal path of s (t) and TLK provides the time for comparing two paths w.r.t. $\succ_{LK}$. We will prove that $\text{TLK}([u_1; \ldots; u_m], [v_1; \ldots; v_n]) = O(m \cdot n \cdot |\mathcal{P}os^*(u_1)| \cdot |\mathcal{P}os^*(v_1)|)$ holds, which implies the time for comparing s and t w.r.t. KNSS to be

$$O(|\mathcal{P}os^*(s)| \cdot |\mathcal{P}os^*(t)| \cdot ||s|| \cdot ||t|| \cdot |\mathcal{P}os^*(s)| \cdot |\mathcal{P}os^*(t)|)$$

which concludes the proof since $||u|| \le |u|$ and $|\mathcal{P}os^*(u)| \le |u|$ for all terms u.

It remains to be proved that two paths $P = [u_1; \ldots; u_m]$ and $Q = [v_1; \ldots; v_n]$ can be compared in time $\text{TLK}(P, Q) = O(m \cdot n \cdot |\mathcal{P}os^*(u_1)| \cdot |\mathcal{P}os^*(v_1)|)$. Assume that all paths of the proper subterms of u_1 have been compared with all paths of the proper subterms of v_1. Let $\mathcal{A}$ be a 4-dimensional array in which the results are stored: $\mathcal{A}(p, i, q, k)$

provides the result of comparing $\mathcal{P}ath_i(u|_p)$ with $\mathcal{P}ath_k(v|_q)$ such that $p \neq \Lambda \neq q$. $PCOMP(P,Q)$ denotes the additional time required to compare P and Q under these assumptions.

For any term v in Q, we can determine in time $O(|\mathcal{P}os^*(u)| \cdot |\mathcal{P}os^*(v)|)$ whether there exists a term u in P that covers it:

> The equivalence of the leading operators of u and v of which the status is of type multiset presents the worst case. Then the following observations lead to the complexity stated above:
>
> - The comparison of $\mathcal{S}ub(P,u)$ and $\mathcal{S}ub(Q,v)$ w.r.t. $\succ_{LK}$ needs a time of $O(1)$ by using the array $\mathcal{A}$.
> - In order to compare the multisets $\mathcal{P}ath(\mathcal{A}rgs(u))$ and $\mathcal{P}ath(\mathcal{A}rgs(v))$ w.r.t. $\succ_{LK}^{mul}$, $|\mathcal{P}os^*(u)| \cdot |\mathcal{P}os^*(v)|$ comparisons must be done (by using the array $\mathcal{A}$) since $|\mathcal{P}ath(\mathcal{A}rgs(t))| = |\mathcal{P}ath(\{t\})| = |\mathcal{P}os^*(t)|$ holds for all terms t.
> - The comparison of $\mathcal{S}up(P,u)$ and $\mathcal{S}up(Q,v)$ is redundant if the common suffix of the paths (P and Q) has been removed.

Therefore, $PCOMP(P,Q) = O(|\mathcal{P}os^*(u_1)| \cdot |\mathcal{P}os^*(v_1)|)$ because u_1 (v_1) is the greatest term w.r.t. $|.|$ of P (Q). Note that P (Q) contains m (n) terms. Thus, $m \cdot n$ comparisons are necessary in the worst case, and each of these is bounded by $PCOMP(P,Q)$. This concludes the proof: $\text{TLK}(P,Q) = O(m \cdot n \cdot |\mathcal{P}os^*(u_1)| \cdot |\mathcal{P}os^*(v_1)|)$. □

Lemma 4.4 *Given two terms s, t, a precedence $\succ$ and a status function τ, $s \succ_{\text{PSDS}} t$ can be determined in time $O(|s|^4 \cdot |t|^4)$.*

Proof: This upper bound will be proved similarly to lemma 4.1. In order to compare s and t w.r.t. PSDS, all path-decompositions must be compared w.r.t. $\succ_{LP}^{mul}$. Thus, the time for comparing s and t w.r.t. PSDS is not greater than

$$O(|\mathcal{P}os^*(s)| \cdot ||s|| \cdot |\mathcal{P}os^*(t)| \cdot ||t|| \cdot \text{TLP}(s,t))$$

where TLP provides the time for comparing two terms w.r.t. $\succ_{LP}$. We will prove that $\text{TLP}(u,v) = O(|\mathcal{P}os^*(u)| \cdot ||u|| \cdot |\mathcal{P}os^*(v)| \cdot ||v||)$ holds, which implies the time for comparing s and t w.r.t. PSDS to be

$$O(|\mathcal{P}os^*(s)|^2 \cdot ||s||^2 \cdot |\mathcal{P}os^*(t)|^2 \cdot ||t||^2)$$

which concludes the proof since $||u|| \leq |u|$ and $|\mathcal{P}os^*(u)| \leq |u|$ for all terms u.

It remains to be proved that two terms u and v can be compared w.r.t. $\succ_{LP}$ in time $\text{TLP}(u,v) = O(|\mathcal{P}os^*(u)| \cdot ||u|| \cdot |\mathcal{P}os^*(v)| \cdot ||v||)$. Assume that all proper subterms of u have already been compared w.r.t. $\succ_{LP}$ with all proper subterms of v. Assume in addition that the results are stored in a 2-dimensional array $\mathcal{A}$ that can be accessed easily: $\mathcal{A}(p,q)$ contains the result of the comparison of $u|_p$ and $v|_q$ (with $p \neq \Lambda \neq q$) w.r.t. $\succ_{LP}$. $TCOMP(u,v)$ denotes the additional time required to compare u and v under these assumptions. A straightforward 2-pass algorithm can decide in time $O(|\mathcal{P}os^*(u)|\cdot||u||\cdot|\mathcal{P}os^*(v)|\cdot||v||)$ whether $u \succ_{LP} v$: In the 1^{st} pass, it is checked whether $\mathcal{H}ead(u) \succ \mathcal{H}ead(v)$. The 2^{nd} pass uses the array $\mathcal{A}$ for comparing the decompositions of the arguments of u and v (while $\mathcal{H}ead(u) = \mathcal{H}ead(v)$). If $\tau(\mathcal{H}ead(u)) = mul$, then at most $|\mathcal{P}os^*(u)|\cdot|\mathcal{P}os^*(v)|$ path-decompositions must be compared. Since each path-decomposition of a subterm of u (v) contains at most $||u||$ ($||v||$) terms, the upper bound stated above is correct. □

Lemma 4.5 *Given two terms s,t and a precedence $\succ$, $s \succ_{\text{PSO}} t$ can be determined in time $O(|s|^3 \cdot |t|^3)$.*

Proof: This upper bound will be proved similarly to lemma 4.3. In order to compare s and t w.r.t. PSO, all paths of s and t must be compared w.r.t. $\succ_{PO}$. Thus, the time for comparing s and t w.r.t. PSO is not greater than

$$O(|\mathcal{P}os^*(s)| \cdot |\mathcal{P}os^*(t)| \cdot \text{TPO}(P,Q))$$

where P (Q) is the maximal path of s (t) and $\text{TPO}([u_1;\ldots;u_m],[v_1;\ldots;v_n]) = O(m \cdot n \cdot |\mathcal{P}os^*(u_1)| \cdot |\mathcal{P}os^*(v_1)|)$ provides the time for comparing two paths w.r.t. $\succ_{PO}$. This concludes the proof (see the 1^{st} part of the proof of lemma 4.3).

It remains to be proved that two paths $P = [u_1;\ldots;u_m]$ and $Q = [v_1;\ldots;v_n]$ can be compared in time $\text{TPO}(P,Q) = O(m\cdot n\cdot|\mathcal{P}os^*(u_1)|\cdot|\mathcal{P}os^*(v_1)|)$. Assume that all paths of the proper subterms of u_1 have been compared with all paths of the proper subterms of v_1. Let $\mathcal{A}$ be a 4-dimensional array in which the results are stored: $\mathcal{A}(p,i,q,k)$ provides the result of comparing $\mathcal{P}ath_i(u|_p)$ with $\mathcal{P}ath_k(v|_q)$ such that $p \neq \Lambda \neq q$. $PCOMP(P,Q)$ denotes the additional time required to compare P and Q under these assumptions.

For any term v in Q, we can find out whether there exists a term u in P that covers it in time $O(|\mathcal{P}os^*(u)| \cdot |\mathcal{P}os^*(v)|)$ by a straightforward 2-pass algorithm: In the 1^{st} pass, it is checked whether there is a term u in P such that $\mathcal{H}ead(u) \succ \mathcal{H}ead(v)$; if none exists, then in the 2^{nd} pass, we check whether $\mathcal{H}ead(u) = \mathcal{H}ead(v)$ in which case the array $\mathcal{A}$ is used to compare the paths of the arguments of u and v. It is obvious that $|\mathcal{P}ath(\mathcal{A}rgs(t))| = |\mathcal{P}ath(\{t\})| = |\mathcal{P}os^*(t)|$ holds for all terms t. Therefore, $PCOMP(P,Q) = O(|\mathcal{P}os^*(u_1)| \cdot |\mathcal{P}os^*(v_1)|)$ since u_1 (v_1) is the greatest term w.r.t. $|.|$ of P (Q). Note that P (Q) contains m (n) terms. Thus, $m \cdot n$ comparisons are necessary in the worst case, and each of these is bounded by $PCOMP(P,Q)$. This concludes the proof: $\text{TPO}(P,Q) = O(m \cdot n \cdot |\mathcal{P}os^*(u_1)| \cdot |\mathcal{P}os^*(v_1)|)$. □

Lemma 4.6 ([Krishnamoorthy and Narendran, 1985]) *Given two terms s,t and a precedence $\succ$, $s \succ_{\text{RPO}} t$ can be determined in time $O(|s| \cdot |t|)$.*

In [Snyder, 1993], it has been shown that although the straightforward implementation of the recursive definition of the RPOS can result in exponential behaviour in the general case, it is possible to compare two *ground* terms w.r.t. the RPOS with *total* precedences in $O(n \cdot lg\, n)$, where n is the combined size of the two terms to be compared. The algorithm is based on the following concepts: (i) sorting the set of all subterms of the two terms to be compared, (ii) using a priority queue to order the subterms (i.e. doing a Heap sort) and (iii) proceeding bottom-up to insert the subterms into the queue. Unfortunately, this approach probably cannot be generalized for comparing *non-ground* terms with the RPOS based on *non-total* precedences.

References

[Avenhaus *et al.*, 1989] Jürgen Avenhaus, Klaus E. Madlener, and Joachim Steinbach. COMTES - An experimental environment for the completion of term rewriting systems. In N. Dershowitz, editor, *Proc. 3rd RTA (LNCS 355)*, pages 542–546, Chapel Hill (North Carolina), April 1989.

[BenCherifa and Lescanne, 1987] Ahlem BenCherifa and Pierre Lescanne. Termination of rewriting systems by polynomial interpretations and its implementation. *SCP*, 9(2):137–160, October 1987.

[Dershowitz, 1982] Nachum Dershowitz. Orderings for term rewriting systems. *JTCS*, 17(3):279–301, March 1982.

[Dershowitz, 1987] Nachum Dershowitz. Termination of rewriting. *JSC*, 3:69–116, February/April 1987. see also Corrigendum - Termination of rewriting (*JSC* 4:409-410, 1987) and 1st RTA, volume 202 of LNCS, pages 180-224, Dijon (France), May 1985.

[Huet and Oppen, 1979] Gérard Huet and Derek C. Oppen. Equations and rewrite rules: A survey. In R. Book, editor, *Symposium on Formal Language Theory*, pages 349–405, Santa Barbara (California), December 1979. Academic Press.

[Jouannaud *et al.*, 1982] Jean-Pierre Jouannaud, Pierre Lescanne, and Fernand Reinig. Recursive decomposition ordering. In D. Bjørner, editor, *Working Conference on Formal Description of Programming Concepts II (IFIP)*, pages 331–348, Garmisch-Partenkirchen (Germany), 1982.

[Kamin and Lévy, 1980] Sam Kamin and Jean-Jacques Lévy. Attempts for generalizing the recursive path orderings. Urbana (Illinois), February 1980.

[Kapur *et al.*, 1985] Deepak Kapur, Paliath Narendran, and G. Sivakumar. A path ordering for proving termination of term rewriting systems. In H. Ehrig, editor, *10th CAAP*, volume 185 of *LNCS*, pages 173–187, Berlin (Germany), March 1985.

[Knuth and Bendix, 1967] Donald E. Knuth and Peter B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Conference on Computational Problems in Abstract Algebra*, pages 263–297, Oxford (England), August/September 1967. Pergamon Press.

[Krishnamoorthy and Narendran, 1985] M.S. Krishnamoorthy and Paliath Narendran. Note on recursive path ordering. *JTCS*, 40:323–328, 1985.

[Lankford, 1975] Dallas S. Lankford. Canonical algebraic simplification in computational logic. Memo ATP-25, U. of Texas, Austin (Texas), May 1975.

[Lankford, 1979] Dallas S. Lankford. On proving term rewriting systems are noetherian. Memo MTP-3, Louisiana Technical U., Dept. of Mathematics, Ruston (Louisiana), May 1979.

[Manna and Ness, 1970] Zohar Manna and Stephen Ness. On the termination of Markov algorithms. In *3rd International Conference on System Science*, pages 789–792, Honolulu (Hawaii), 1970.

[Plaisted, 1978] David Alan Plaisted. A recursively defined ordering for proving termination of term rewriting systems. Report UIUCDCS-R-78-943, Dept. of Computer Science, U. of Illinois at Urbana-Champaign, Urbana (Illinois), September 1978.

[Rusinowitch, 1987] Michaël Rusinowitch. Path of subterms ordering and recursive decomposition ordering revisited. *JSC*, 3:117–131, February/April 1987. also appeared in 1st RTA, Dijon (France), pp. 225-240, May 1985.

[Snyder, 1993] Wayne Snyder. On the complexity of recursive path orderings. *IPL*, 46:257–262, July 1993.

[Steinbach, 1988] Joachim Steinbach. Term orderings with status. SEKI-Report SR-88-12, Dept. of Computer Science, U. of Kaiserslautern, Kaiserslautern (Germany), 1988.

[Steinbach, 1989] Joachim Steinbach. Extensions and comparison of simplification orderings. In N. Dershowitz, editor, *3rd RTA*, volume 355 of *LNCS*, pages 434–448, Chapel Hill (North Carolina), April 1989.