Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern

SEKI - REPORT

Path and Decomposition Orderings for Proving AC-Termination

J. Steinbach

SEKI Report SR-89-18

Path and Decomposition Orderings for Proving AC-Termination

J. Steinbach

SEKI Report SR-89-18

Path and decomposition orderings for proving AC-termination

Joachim Steinbach

Universität Kaiserslautern
Fachbereich Informatik
Postfach 3049
D-6750 Kaiserslautern
F.R.G.
e-mail: steinba@uklirb.uucp

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

Abstract

Term rewriting provides a simple mechanism that can be applied to reasoning in structures defined by equations. An equation is converted into a directed rewrite rule by comparing both sides w.r.t. an ordering. However, there exist equations which are incomparable. The handling of such equations includes, for example, partitioning the given equational theory into a set R of rules and a set E of equations. An appropriate reduction relation allows reductions modulo the equations in E. The effective computation with this relation presumes E-termination. We will give an overview and a classification of the well-known methods guaranteeing AC-termination based on the recursive path ordering. Furthermore, we will show that these techniques (called associative path orderings) cannot use quasi-orderings on operators. Above all, this report will deal with some new orderings applicable to AC-theories. We apply a slight extension (the embedding of status) of the concept of the associative path ordering of Gnaedig and Lescanne to several path and decomposition orderings. Since these orderings are stronger than the recursive path ordering, the corresponding orderings restricted to AC-theories are more powerful than the associative path ordering. From a practical point of view these new AC-orderings are more interesting than the associative path ordering because the automatic detection of an admissible precedence for orienting the rules of a given system is much easier.

Keywords

Associative and commutative operators, Associative path ordering, E-commuting, E-compatible, Equational theories, Flattening, Lexicographical ordering, Multiset ordering, Path of subterms ordering, Recursive decomposition ordering, Recursive path ordering, Simplification ordering, Status, Termination, Term rewriting system, Well-founded ordering

Contents

1 Motivation

Term rewriting systems gain more and more in importance because they are a useful model for non-deterministic computations: They are based on directed equations with no explicit control. Various applications in many areas of computer science and mathematics including automatic theorem proving and program verification, abstract data type specifications and algebraic simplification have been developed.

The basic concept of term rewriting systems [see for example [AM89], [HO80]] is that of reducing a given term to an easier one. An equation is converted into a directed rewrite rule in such a way that the right-hand side of the rule is easier than the left-hand side. In order to exclude infinite derivations of terms a rewrite system must terminate. Orderings on terms are able to guarantee termination. A survey of the most important ones is given in [De87].

The basic idea of using an ordering $>$ is to verify that the rewrite relation $\Longrightarrow_{\mathfrak{R}}$ [induced by the rule system $\mathfrak{R}$] is included in $>$. Such an ordering must be well-founded to avoid infinite derivations of terms. To check the inclusion $'\Longrightarrow_{\mathfrak{R}} \subseteq >'$ all [infinitely many] possible derivations must be tested. In order to restrict this infinite test to a finite one a reduction ordering is required. A reduction ordering is a well-founded ordering which has the replacement property [also called compatibility with the structure of terms] which ensures that a value of a term will be decreased if any one of its subterms is decreased. The notion of reduction orderings leads to the following description of termination of rewrite systems [see [La77]]:

> *A rewrite system $\mathfrak{R}$ terminates if and only if there exists a reduction ordering $>$ such that $\sigma[l] > \sigma[r]$ for each rule $l \longrightarrow_{\mathfrak{R}} r$ and for any ground substitution σ.*

The theorem above reveals another dilemma which is known as the universal quantification on substitutions or the so-called stability w.r.t. substitutions: $s > t$ implies $\sigma[s] > \sigma[t]$, for all σ.

Summarizing, it is to remark that a termination proof of a term rewriting system requires a reduction ordering stabilized w.r.t. substitutions. In general, it is very difficult to guarantee the well-foundedness of a reduction ordering. This fact

leads to the basic concept of characterizing classes of orderings for which there is no need to prove this condition. One possible solution is represented by the class of simplification orderings which are at least reduction orderings:

An ordering is a simplification ordering if and only if it has
- *the replacement property,*
- *the subterm property [any term is greater than any of its proper subterms] and*
- *the deletion property [deleting subterms reduces the term].*

Simplification orderings are discussed in detail in [De87]. Well-known simplification orderings are the recursive path orderings and the Knuth-Bendix orderings. Unfortunately, the termination of an arbitrary term rewriting system is an undecidable property, even in the 'one-rule case' [[Da88]].

An additional negative fact derives from the existence of equations of which the left-hand side and the right-hand side are incomparable in any case. For example, a rewriting system containing the commutativity axiom $x+y = y+x$ as a rule is non-terminating. However, if the termination property is not satisfied, the set of axioms can be split into two parts: Those axioms causing non-termination are used as equations E while the others are used as rewrite rules $\mathfrak{R}$. The appropriate reduction relation allows reductions modulo the equations in E. The effective computation with this relation presumes

- *a complete unification algorithm for the equational theory E,*
- *E-termination, i.e. there is no infinite sequence of terms of the form* $t_1 =_E t_1' \Longrightarrow_{\mathfrak{R}} t_2 =_E t_2' \Longrightarrow_{\mathfrak{R}} \cdots$.

The classical Knuth-Bendix completion procedure has first been extended to rewriting modulo equations by Lankford and Ballantyne [see [LB77a]], for example for the case of equations that consist of associativity and commutativity. A different approach by Huet [[Hu80a]] deals with left-linear rewriting systems. The method by Peterson and Stickel [[PS81]] may be applied to linear theories with a finite and complete unification algorithm. These methods have been unified in a more general framework by Jouannaud and Kirchner [[JK86]].

We now adapt the general results on termination from the previous page to the case of equational term rewriting systems.

An equational term rewriting system terminates if there is an ordering $>$ which contains the rewrite relation $\Longrightarrow_{\mathfrak{R}/E} = =_E \circ \Longrightarrow_{\mathfrak{R}} \circ =_E$ [where $\circ$ denotes composition of relations]. Testing this inclusion requires all derivations of the form $s \Longrightarrow_{\mathfrak{R}/E} t$

to be checked. This requirement can be refined: If > is E-compatible [PS81], then > contains $\Longrightarrow_{\mathfrak{R}/E}$ if and only if it contains $\stackrel{+}{\Longrightarrow}_{\mathfrak{R}}$ [cf. BP85]. An ordering > is E-compatible if and only if

$$\begin{array}{lll} s & =_E & s' \\ > & & \\ t & =_E & t' \end{array} \quad \text{implies} \quad \begin{array}{l} s' \\ > \\ t' \end{array}$$

If a reduction ordering > is E-compatible and $\sigma(l) > \sigma(r)$, for every rule $l \longrightarrow_{\mathfrak{R}} r$ and every ground substitution σ, then the equational term rewriting system $\mathfrak{R}/E$ terminates.

Jouannaud and Muñoz succeeded in weakening the E-compatibility for this statement (see [JM84]). They introduced a property called E-commutation:

$$\begin{array}{lll} s & =_E & s' \\ > & & \\ t & & \end{array} \quad \text{implies} \quad \begin{array}{lll} \exists t' & & s' \\ & & > \\ t & =_E & t' \end{array}$$

Jouannaud and Muñoz reduce the E-termination problem to the ordinary termination problem, in order to apply well known techniques based on simplification orderings. E-termination and termination of a rewriting relation are the same if the used rewriting relation is E-commuting. If this property does not hold, it can be achieved by systematically adding rules which are generated by orienting critical pairs between rules and equations.

The following theorem [St89b] points out the main importance of E-commutation for the E-termination problem. The theorem is a modification of a result contained in [JM84].

> *A rewrite system $\mathfrak{R}$ is E-terminating if there exists an E-commuting simplification ordering > such that $\sigma(l) > \sigma(r)$ for each rule $l \longrightarrow_{\mathfrak{R}} r$ and for any ground substitution σ.*

Obviously, E-termination strongly depends on the given underlying theory E. For example, E must satisfy the following two conditions in order to prevent infinite derivations (see [JM84]):

- If $s = t \in E$, then the set of variables of both terms must be identical. Otherwise there will be loops since we may instantiate the additional variable by an instance of a left-hand side of a rule of $\mathfrak{R}$, then rewrite the term using the rule and thus arrive at the 'starting term' by applying the 'starting equation' twice.

For example,

$$\mathfrak{R}: \quad x * 1 \rightarrow x$$
$$E: \quad x * 0 = 0$$
$$0 =_E (x * 1) * 0 \rightarrow_{\mathfrak{R}} x * 0 =_E 0$$

- Furthermore, E-termination cannot be satisfied if there is an equation of the form $t =_E x$ such that x has more than one occurrence in t. In this case a left-hand side l of a rule of $\mathfrak{R}$ is E-equal to a term with several occurrences of l. Therefore, we can rewrite one of these and start the process again.
  For example,

$$\mathfrak{R}: \quad \neg\neg x \rightarrow x$$
$$E: \quad x \wedge x = x$$
$$\neg\neg x =_E \neg\neg x \wedge \neg\neg x \rightarrow_{\mathfrak{R}} \quad x \wedge \neg\neg x =_E x \wedge (\neg\neg x \wedge \neg\neg x) \rightarrow_{\mathfrak{R}}$$
$$x \wedge (x \wedge \neg\neg x) =_E \ \ldots$$

In this report we are going to deal with a special theory E: associative-commutative axioms. An equational theory E is called an associative-commutative theory if every equation in E is either an associative or commutative axiom:

- $f[x,f[y,z]] = f[f[x,y],z] : f \in \mathfrak{F}_A$ and
- $f[x,y] = f[y,x] : f \in \mathfrak{F}_C$.

In order to describe the fact that f is both associative and commutative we use '$f \in \mathfrak{F}_{AC}$'. An equational term rewriting system $(\mathfrak{R},E)$ will be an associative-commutative term rewriting system if E is an associative-commutative theory.

There only exist a few orderings for this kind of rewriting systems including the associative path orderings [[Gn88], [GL86], [BP85], [BP85a], [DHJP83]], the orderings on special polynomial interpretations [[BL87a], [BL86], [La79], [SZ89]] and the associative-commutative Knuth-Bendix orderings [[St89b]].

Associative path orderings extend the recursive path orderings to AC-congruence classes. They are based on flattening and transforming the terms by a rewriting system with rules similar to the distributivity (or the endomorphism) axioms. Furthermore, the precedence on the operators has to satisfy a special property.

The polynomial interpretation I for an associative (and commutative) operator must be of the form $I[f][x,y] = axy + b[x+y] + c$ such that $ac+b-b^2 = 0$. The fundamental disadvantage of polynomial orderings is the difficulty of choosing interpretations for operators such that a given rewrite system terminates.

The associative-commutative Knuth-Bendix ordering [ACK, [St89b]] is a modification of the well-known Knuth-Bendix ordering of [KB70]. The transformation of terms required by the associative path orderings, is reduced to a minimum. Moreover, the algorithm of [Ma87] for finding an adequate weight function to prove the termination of a given rewriting system w.r.t. the Knuth-Bendix ordering, can be applied here. Unfortunately, the applicability of the ACK is bounded by that of the Knuth-Bendix ordering.

Here, we supply a concept which extends the associative path ordering in two ways. First of all, a proof will be given to justify the use of status in the definition of the associative path orderings. Furthermore, we have succeeded in applying the concepts of the associative path orderings to other stronger orderings including the path of subterms ordering, the path ordering of Kapur, Narendran and Sivakumar and several decomposition orderings. Thus, this report is the extension of [St89a] ([St88a]) to AC-theories. Another advantage of, for example, the AC-decomposition orderings is that the precedence can often be derived from the structure of the reduction rules in an easier way than with the associative path ordering.

After giving some indispensable definitions in the next chapter, the classical path and decomposition orderings (see [St89a] or [St88a]) will briefly be presented. In chapter 4, we recapitulate, classify and slightly extend (by incorporating status) the concept of the restriction to AC-theories contained in [BP85] and [GL86]. Moreover, it will be shown that these techniques cannot use quasi-orderings on the set of function symbols. Subsequently, the application of this approach to the definitions of the path and decomposition orderings of chapter 3 as well as the appropriate lemmas and some examples will be given.

2 Notation

A term rewriting system $\mathfrak{R}$ over a set of terms Γ is a finite or countably infinite set of rules, each of the form $l \rightarrow_{\mathfrak{R}} r$, where l and r are terms in 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 $\mathfrak{F}$ of operators (or function symbols) and some denumerably infinite set $\mathfrak{V}$ 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. The size |t| of a term t is the number of operators and variables occurring in t.

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. We use the formalism of positions of terms 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) is the set of all terminal occurrences (occurrences of leaves) of 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 $u \in O(t)$) by s.

A (partial) ordering on Γ_G is a transitive and irreflexive binary relation >. It is called well-founded if there are no infinite descending chains. Most of the orderings on terms are precedence orderings using a special ordering on operators. More precisely, a precedence is a partially ordered set $(\mathfrak{F}, \triangleright)$ consisting of the set $\mathfrak{F}$ of operators and an irreflexive and transitive binary relation $\triangleright$ defined on elements of $\mathfrak{F}$. Obviously, a precedence can also be a quasi-ordering. As usual, a quasi-ordered set $(\mathfrak{F}, \trianglerighteq)$ consists of the set $\mathfrak{F}$ and a transitive and reflexive binary relation $\trianglerighteq$ defined on elements of $\mathfrak{F}$. A quasi-ordering defines an equivalence relation = as both $\trianglerighteq$ and $\trianglelefteq$ $(\trianglerighteq \cap \trianglelefteq)$, and a partial ordering $\triangleright$ as $\trianglerighteq$ but not $\trianglelefteq$ $(\trianglerighteq \setminus \trianglelefteq)$.

Note that a term ordering > is used to compare terms. Since operators have terms as arguments we define an extension of >, called lexicographically greater $(>^{lex})$, on tuples of terms as follows:

$$
\begin{array}{lllll}
(s_1, s_2, \ldots, s_m) & >^{lex} & (t_1, t_2, \ldots, t_n) & & \\
\text{if either} \quad m > 0 & \wedge & n = 0 & & \\
\quad \text{or} \quad s_1 > t_1 & & & & \\
\quad \text{or} \quad s_1 = t_1 & \wedge & (s_2, \ldots, s_m) & >^{lex} & (t_2, \ldots, t_n).
\end{array}
$$

If there is no order of succession among the terms of such tuples, these structures are called multisets. Multisets differ from sets by allowing multiple occurrences of identical elements. The multiset difference is represented by \. The extension of $\succ$ on multisets of terms is defined as follows. A multiset S is greater than a multiset T, denoted by

$$
\begin{aligned}
& S \succ\succ T \\
& \text{iff} \quad \cdot \; S \neq T \quad \wedge \\
& \qquad \cdot \; [\forall t \in T \backslash S][\exists s \in S \backslash T] \; s \succ t
\end{aligned}
$$

i.e., $S \succ\succ T$ if T can be obtained from S by replacing one or more terms in S by any finite number of terms, each of which is smaller (w.r.t. $\succ$) than one of the replaced terms.

To combine these two concepts of tuples and multisets, we assign a status $\tau(f)$ to each operator $f \in \mathfrak{F}$ that determines the order according to which the subterms of f are compared. Formally, a status is a function which maps the set of operators into the set {mult , left , right}. Thus, 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 $t = f(t_1, \ldots, t_n)$ depends on the status of f : If $\tau(f) =$ mult, then args(t) is the multiset $\{t_1, \ldots, t_n\}$ and otherwise, args(t) leads to the tuple $(t_1, \ldots, t_n)$. 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.

In the remaining parts of this report, by writing s, t and $\triangleright$ we will always assume that s and t are terms over Γ and $\triangleright$ is a (partial or quasi-) precedence on the set $\mathfrak{F}$ 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:

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

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

All of the following orderings uniquely define a congruence $\sim$ dependent on $\mathfrak{F}$ and τ via:

$$
\begin{array}{l}
f(s_1,\ldots,s_m) \sim g(t_1,\ldots,t_n) \\
\text{iff} \\
f = g \wedge m = n \wedge \quad \text{i)}\ \tau(f) = \text{mult} \wedge (\exists\pi)(\forall i)\ s_i \sim t_{\pi(i)} \\
\qquad\qquad\qquad\qquad\ \text{or} \quad \text{ii)}\ \tau(f) \neq \text{mult} \wedge (\forall i)\ s_i \sim t_i
\end{array}
$$

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.

$$
\begin{array}{lll}
s > t & \text{iff} & - \ s >_1 t \\
 & & - \ s >_2 t
\end{array}
$$

stands for : $s > t$ 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 $\geq_1$.

3 Path and decomposition orderings

All orderings described in this chapter are recursively defined simplification orderings and contained in [St89a] :

- **Recursive path ordering with status** (RPOS)
  This is an extension of Dershowitz' RPO introduced by Kamin and Lévy ([KL80], [De82]).

- **Recursive decomposition ordering with status** (RDOS)
  The original recursive decomposition ordering has been first defined by Lescanne. We present a version ([St89a]) which is different in regard to the status. It is an extension of Lescanne's ordering.

- **Path of subterms ordering with status on decompositions** (PSDS)
  The PSDS, based on decompositions, results from the PSO (path of subterms ordering, [Pl78a]). The corresponding ordering without status (called PSD in [St89a]) is equivalent to the path of subterms ordering of Plaisted. In contrast to the PSO, the PSD has the advantage that it can easily be extended by the principle of status (see [St89a]).

- **Improved recursive decomposition ordering with status** (IRDS)
  The improved recursive decomposition ordering has been developed by Rusinowitch ([Ru87]). He has also incorporated status in it. However, we present another (simpler) version (of [St89a]) which is similar to it. The power of these two orderings overlap. Moreover, the path ordering of Kapur, Narendran and Sivakumar ([KNS85]) is equivalent to the IRDS ([St89a]).

The main point of this chapter is a brief description of all these orderings with status. For a better understanding, the important ideas of these methods of comparing terms will be demonstrated by an example. Note that it is superfluous to present the path of subterms ordering of Plaisted and the path ordering of Kapur, Narendran and Sivakumar since they are equal to the PSD(S) and the IRDS, respectively (see 3.3).

The orderings described satisfy properties that qualify them for proving termination of term rewriting systems (the proofs can be found in [St88a]):

- Simplification ordering:

  - $s > t$ implies $f[...,s,...] > f[...,t,...]$, for all $f \in \mathfrak{F}$
    [replacement property]

  - $f[...,t,...] > t$, for all $f \in \mathfrak{F}$
    [subterm property]

  - $f[...,t,...] > f[...,...]$, for all $f \in \mathfrak{F}$ with $\tau[f]$=mult
    [deletion property]

- Well-foundedness:

  $\nexists\ t_1 > t_2 > ...$
  [follows directly from the previous property]

- Stability w.r.t. substitutions:

  $s > t$ implies $\sigma[s] > \sigma[t]$, for all ground substitutions σ

- Monotony w.r.t. the precedence $\triangleright$:

  $\triangleright_1 \subseteq \triangleright_2$ implies $>[\triangleright_1] \subseteq >[\triangleright_2]$

  All orderings of this report are precedence orderings denoted by >[p], where the parameter p stands for the precedence. An ordering is called monotonous w.r.t $\triangleright$ if it is strengthened by increasing the precedence.

3.1 Recursive path ordering

The comparison w.r.t. 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.

Definition 3.1.1 ([KL80] , [De82])

$s >_{RPOS} t$

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

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

■

Lemma 3.1.2 ([KL80], [St88a])

The RPOS is a simplification ordering, monotonous w.r.t. the precedence and stable w.r.t. substitutions.

■

Example 3.1.3

We want to prove that the distributive law $s = x*(y+z) \rightarrow (x*y)+(x*z) = t$ terminates. We use the total precedence $* \triangleright +$ and the statuses $\tau(+)=\tau(*)=$left. Since $* \triangleright +$ we must show that $\{s\} \gg_{RPOS} args(t) = \{ x*y , x*z \}$. 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 $(x , y+z) >_{RPOS,left} (x , y) \wedge \{s\} \gg_{RPOS} \{x,y\}$ because $y+z >_{RPOS} y$ (by using the subterm property of the RPOS, 3.1.1 iv) and $s >_{RPOS} x$, $s >_{RPOS} y$. $s >_{RPOS} x*z$ is proved the same way.

■

Remark 3.1.4

We would like to point out that there exist two different versions of the RPOS in the publications:

- a somehow non-deterministic one (see definition 3.1.1):

$$\begin{array}{lll} & top(s) \rhd top(t) & \wedge \quad \ldots \\ \text{or} & top(s) = top(t) & \wedge \quad \ldots \\ \text{or} & args(s) \gg_{RPOS} \{t\} & \end{array}$$

  The third alternative can be tested irrespective of the leading function symbols. For example, if $top(s) \rhd top(t)$ the first *or* the third condition could be checked.

- a deterministic version:

$$\begin{array}{lll} & top(s) \rhd top(t) & \wedge \quad \ldots \\ \text{or} & top(s) = top(t) & \wedge \quad \ldots \\ \text{or} & \neg(top(s) \trianglerighteq top(t)) & \wedge \quad args(s) \gg_{RPOS} \{t\} \end{array}$$

  Note that the three alternatives are disjoint.

In contrast to the RPO without status where both versions are equivalent, the powers of these approaches w.r.t. the RPOS differ: The non-deterministic RPOS is a proper extension of the deterministic one. Moreover,

- the non-deterministic RPOS is a simplification ordering, whereas

- the deterministic RPOS does not have the subterm property since $(x*y)*z \not>_{RPOS} x*y$ if $\tau(*)$=right.

■

3.2 Decomposition orderings

To define the decomposition orderings, we need some kind of formalism.

Definition 3.2.1 [Decomposition]

- Path-decomposition of a term:
  $dec_{i.u}[f[t_1,...,t_n]] = \{f[t_1,...,t_n]\} \cup dec_u[t_i]$ with $dec_\varepsilon[t] = \{t\}$
- Decomposition of a set of terms:
  $dec[\{t_1,...,t_n\}] = \{ dec_u[t_i] \mid i \in [1,n],\ u \in Ot[t_i] \}$
- Set of proper subterms of a path-decomposition P w.r.t. a term t:
  $sub[P , t] = \{ s \in P \mid [\exists u \neq \varepsilon]\ t/u = s \}$

■

Example 3.2.2

Suppose $t = [x*y]+[x*z]$, then $dec_{11}[t] = \{ t , x*y , x \}$ and $dec[\{t\}] = \{ dec_{11}[t], dec_{12}[t] , dec_{21}[t] , dec_{22}[t] \}$. Moreover, $sub[dec_{11}[t] , x*y] = \{x\}$.

■

The recursive decomposition ordering with status [RDOS, for 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. A term s is greater than a term t [w.r.t. the RDOS] if the decomposition of s is greater than the decomposition of t. The ordering on these multisets [$»»_{LD}$] is an extension of the basic ordering on terms [$>_{LD}$] to multisets of multisets.

Definition 3.2.3 [[St89a]]

$s >_{RDOS} t$ iff $dec[\{s\}] » »_{LD} dec[\{t\}]$

with $dec_u[s'] \ni s >_{LD} t \in dec_v[t']$

iff i) $top[s] \triangleright top[t]$

ii) $top[s] = top[t] \wedge \tau[top[s]] = mult \wedge$
- $sub[dec_u[s'],s] »_{LD} sub[dec_v[t'],t]$
- $args[s] »_{RDOS} args[t]$

iii) $top[s] = top[t] \wedge \tau[top[s]] \neq mult \wedge$
$\{s\} »_{RDOS} args[t] \wedge args[s] >_{RDOS,\tau[top[s]]} args[t]$

■

Another ordering based on decompositions results from the path of subterms ordering. It is remarkable that the PSO is an extremely recursive ordering which takes three suborderings into account. The next definition (without status) is equivalent to the PSO providing a much simpler method of using decompositions.

Definition 3.2.4 [St89a]

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

■

The essential difference between the PSDS and the improved recursive decomposition ordering with status (IRDS) concerns the way by which a comparison is processed. While the PSDS works according to the principle 'breadth-first' the IRDS reveals the use of the principle 'depth-first': If the leading function symbols of the terms to be compared are identical, the IRDS chooses only *one* subterm. On the other hand, the PSDS proceeds by simultaneously considering the decomposition multiset of *all* subterms. Furthermore, the IRDS is a proper extension of the RDOS, due to a slight change of the second part (ii) of its definition.

Definition 3.2.5 [St89a]

$s >_{IRDS} t$ 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) $top(s) = top(t) \wedge \tau(top(s)) = mult \wedge$
- $sub(dec_u(s'),s) \gg \gg_{EL} sub(dec_v(t'),t)$
- $dec(args(s)) \gg \gg_{EL} dec(args(t))$

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

■

Concluding, we point out that all orderings presented can be used to prove the termination of arbitrary term rewriting systems.

Lemma 3.2.6 [St88a]

The RDOS, PSDS and the IRDS are simplification orderings, monotonous w.r.t. to the precedence and stable w.r.t to substitutions.

■

Vicariously, we will illustrate the definition of the IRDS by an example.

Example 3.2.7 (see example 3.1.3)

We will prove the termination of the distributive law $s = x*(y+z) \rightarrow (x*y)+(x*z) = t$. We use the total precedence $* \triangleright +$ and the statuses $\tau(+) = \tau(*) =$ = left.
We have to prove that $dec(\{s\}) \gg \gg_{EL} dec(\{t\})$:

$dec(\{s\}) = \{ dec_1(s) , dec_{21}(s) , dec_{22}(s) \}$ and
$dec(\{t\}) = \{ dec_{11}(t) , dec_{12}(t) , dec_{21}(t) , dec_{22}(t) \}$.

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

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

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

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

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

■

3.3 Comparison

In this section we compare the power of the presented orderings in addition to some well-known path orderings. For the sake of completeness, we include the basic orderings restricted to multiset status. The power of an ordering is represented by the set of comparable terms. We examine the relation between two sets. There are three possible relations:

- Two orderings can be equivalent [$> = \succ$],
- one ordering can be properly included in the other [$> \subset \succ$] or
- they overlap [$> \between \succ$].

The orderings $>$ and $\succ$ overlap if there exist some terms such that $s > t \wedge s \not\succ t$ and $s' \succ t' \wedge s' \not> t'$.

Note that the orderings described relate to a parameter: the precedence [$>(p)$ denotes the ordering $>$ with the precedence p as a parameter]. This parameter may be either partial or total. Obviously, the conventional notion of the comparison of two orderings requires the orderings to be compared w.r.t. all possible (partial) precedences. This is a very strong condition. From a practical point of view, it is sufficient to consider the comparison of two orderings w.r.t. total precedences, only. The following proposition substantiates these reflections:

Proposition 3.3.1 [St89a], [St88a]

Let $>, \succ$ be orderings which are monotonous w.r.t. $\triangleright$ and $> \subseteq \succ$ w.r.t. total precedences. Then,

$$s >(p)\ t \quad \leadsto \quad (\exists q)\ s \succ(q)\ t.$$

■

Note that we cannot deduce assertions w.r.t. partial precedences from the corresponding relations w.r.t. total precedences. But, this is not what we want. We are only interested in generalizing the comparison of two orderings in the following manner: Is there any ordering $\succ$ using *any* precedence q such that $s \succ(q)\ t$ if $s >(p)\ t$ where $>$ is another (given) ordering based on a precedence p ? The traditional request would contain the search for a stronger ordering $\succ$ based on the same precedence (p) as $>$. Obviously, the above proposition generalizes this requirement since we could give some information about the comparison of orderings independent of the precedence. In order to use the proposition we have to compare the orderings w.r.t. total precedences, only. The relations $\subset$, $=$ and $\between$ w.r.t. total precedences 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.

- $> \subset \rhd$ iff $s >_{(p,\tau)} t \rightsquigarrow s \rhd_{(p,\tau)} t \wedge (\exists p',\tau')(\nexists p'',\tau'')\ s \rhd_{(p',\tau')} t \wedge s >_{(p'',\tau'')} t$

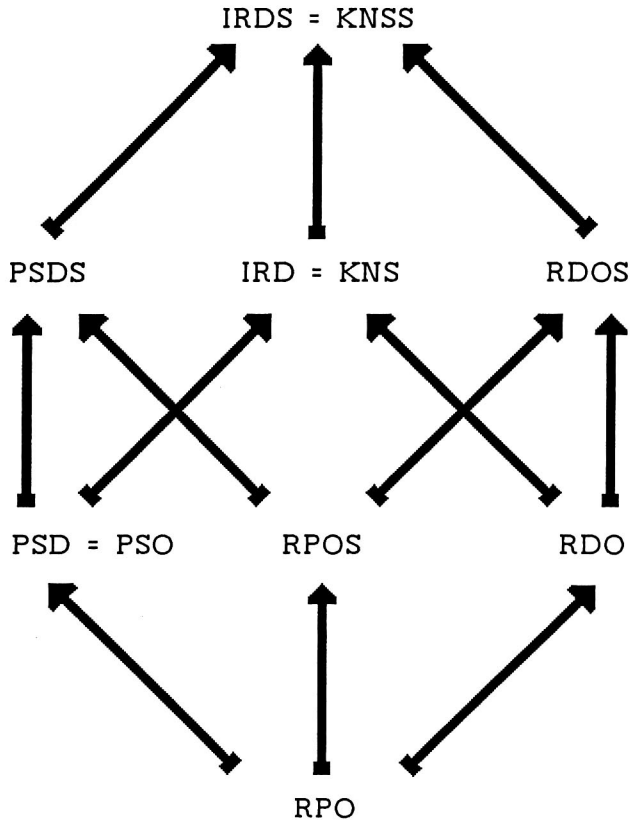
- $> = \rhd$ iff $s >_{(p,\tau)} t \leftrightsquigarrow s \rhd_{(p,\tau)} t$

- $> \# \rhd$ iff $(\exists p',\tau')(\nexists p'',\tau'')\ s >_{(p',\tau')} t \wedge s \rhd_{(p'',\tau'')} t \wedge (\exists p',\tau')(\nexists p'',\tau'')\ s \rhd_{(p',\tau')} t \wedge s >_{(p'',\tau'')} t$

The following lemma reflects the comparison w.r.t. to total precedences. The proof of the lemma can be found in [St88a]. To present the relations in a simple way, we use a kind of Hasse diagrams. If $> \subset \rhd$ then we arrange $\rhd$ above $>$ joining them with an arrow.

Lemma 3.3.2 ([St89a], [St88a])

Assuming arbitrary terms and total precedences. Then, the following relations are valid:



with **IRD** = Improved recursive decomposition ordering [Ru87]
IRDS = Improved recursive decomposition ordering with status [St89a]
KNS = Path ordering of Kapur, Narendran and Sivakumar [KNS85]
KNSS = Path ordering with status of Kapur, Narendran and Sivakumar [KNS85]

PSD = Path of subterms ordering on decompositions [[St89a]]
PSDS = Path of subterms ordering with status on decompositions [[St89a]]
PSO = Path of subterms ordering [[Pl78a]]
RDO = Recursive decomposition ordering [[JLR82]]
RDOS = Recursive decomposition ordering with status [[St89a]]
RPO = Recursive path ordering [[De82]]
RPOS = Recursive path ordering with status [[KL80]]

■

This lemma will be of practical importance if we consider it together with the proposition 3.3.1: Only one (either the IRDS or the KNSS) of the eleven orderings collected in the diagram is needed to cover the union of comparable terms of all the orderings referenced in lemma 3.3.2. 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 in the same way with the IRDS (= KNSS). Consequently, if you are implementing a system where the termination of a rewriting system must be guaranteed, only one of the eleven orderings will have to made available for the user. The cause of it is that the IRDS (the KNSS, respectively) is stronger than all other path and decomposition orderings irrespective of the precedence.

4 An improved concept of the APO

Termination orderings based on transformation techniques have been first proposed in [DHJP83]. This method is rather complex and it was not possible to lift the corresponding ordering to terms containing variables.

The associative path ordering [APO, for short] of [BP85] is similar to this ordering. However, the APO is conceptually simpler since any term is transformed into a single term whereas in [DHJP83] the transformation of a term leads to a multiset of terms. The stability w.r.t. substitutions can be guaranteed by checking a finite number of the potentially [infinitely many] ground substitutions.

In [GL86], Gnaedig and Lescanne simplify the APO to the so-called NFLO. The NFLO generalizes the recursive path ordering and the transformation can be described by a reduction relation. In contrast to the APO, this ordering makes a strong difference between the flattening and the distributing [applying distributivity axioms] processes. First of all, distributivity rules map the terms to compare in the same class modulo distributivity, in order to insure the replacement property. Then, for satisfying AC-commutation, these terms will be flattened along its AC-operators [to represent the associativity axioms] and compared w.r.t. the recursive path ordering without status [to represent the commutativity axioms]. Besides the simplicity, Gnaedig and Lescanne gave a proof for the universal quantification on substitutions based on simple observations on the substitution mechanism. Obviously, from a practical point of view, this method is more interesting than that of Bachmair and Plaisted. However, the root of the NFLO is the APO, and therefore, in the remaining parts of this paper, we will refer to this ordering by associative path ordering.

The associative path ordering is simple enough to allow implementations, yet it is applicable to a variety of theories. It may be used to prove termination of a set of term rewriting rules containing associative, commutative and associative-commutative operators. Furthermore, the APO could be part of a Knuth-Bendix completion algorithm for associative-commutative operators. Note that it is impossible to implement a full Knuth-Bendix procedure for associative but not commutative operators since there exists no finite unification algorithm for associative operators [see [Pl83]].

The basic concept of both methods is that simplification orderings have to be E-commuting to provide termination proofs for E-rewriting [[JM84], [St89b]]:

Let > be a simplification ordering which is E-commuting and stable w.r.t. substitutions:

$\mathfrak{R}$ is E-terminating
if
$l > r$, for all $l \rightarrow r \in \mathfrak{R}$

Instead of E-commutation, the approaches described above [APO, NFLO] use the stronger restriction of E-compatibility.

In this chapter we will try to give a motivation for the use of Gnaedig's and Lescanne's approach. They have presented their method for AC-theories, only. First of all, we will split AC into the concepts of C- and A-theories. In practice, this will be an enhancement of the efficiency if either C-theories or A-theories are used. The reason is that the conditions required of the RPO are not as strict as those for the combined theories.

Subsequently, the power of the various methods will be compared.

Another main point of this chapter is the introduction of an improvement of the APO. The extension we will deal with consists of the incorporation of status to the RPO. It will be shown that the concepts of the APO [based on the RPO] can be transferred to the RPOS by guaranteeing multiset status of associative and commutative operators.

Last but not least, we will show that partial [instead of quasi-] precedences are necessary to guarantee the stability w.r.t. substitutions of the techniques presented in this chapter.

4.1 C-theories

Assume that E contains only commutative axioms such as f[x,y] = f[y,x]. Therefore, an ordering > is needed which is C-compatible:

$$\begin{array}{lll} s & =_C & s' \\ > & & \\ t & =_C & t' \end{array} \quad \text{implies} \quad \begin{array}{l} s' \\ > \\ t' \end{array}$$

It is not very difficult to detect that the RPOS is C-compatible if each commutative function symbol has multiset status:

Definition 4.1.1

Let $\triangleright$ be a precedence and τ a status function requiring multiset status of each C-operator:

$$s >_C t \quad \text{iff} \quad s >_{RPOS} t$$

■

Lemma 4.1.2

$>_C$ is a simplification ordering, C-compatible and stable w.r.t. substitutions.

■

Obviously, the combination of the C-compatibility with the other important properties of the RPOS - simplification ordering and stability w.r.t. substitutions [lemma 3.1.2] - ensures the applicability in practice.

Example 4.1.3 [Dershowitz]

$$\begin{array}{lll} \mathfrak{R}: & x + 0 & \rightarrow x \\ & x * 0 & \rightarrow 0 \\ & x * (y + 1) & \rightarrow (x * y) + x \\ E: & x + y = y + x & \\ & x * y = y * x & \end{array}$$

The RPOS guarantees the termination if $\tau(*) = \tau(+) = \text{mult}$ and $* \triangleright +$.

■

4.2 A-theories

In order to restrict the RPOS so that it can prove the A-termination of term rewriting systems we will pursue the following strategy:

- Establishing exactly one representative of each A-equivalence class
- Reducing each term to the representative of its A-equivalence class
- Comparing the representatives w.r.t. the RPOS

An ordering constructed by these rules is A-compatible since $rep(t') = rep(t) >_{RPOS} rep(s) = rep(s')$ implies $rep(t') >_{RPOS} rep(s')$, where $t' =_A t \wedge s =_A s'$ and rep denotes the representative of a term.

Obviously, the main problem of the above method is the definition of the representatives. Usually, terms with A-operators are described by flattened terms having no nested occurrences of identical associative operators, e.g. $+(+(1,+(2,+(1,3))),2) = +(1,2,1,3,2)$. This representation requires the operators to have variable arity, i.e. associative function symbols may possess any positive number (> 1) of arguments, whereas non-associative operators have a fixed arity.

Definition 4.2.1 (Varyadic terms)

Let $\Gamma(\mathfrak{F},\mathfrak{V})$ be an algebra of terms and α' the arity function on $\mathfrak{F}$. The varyadic term algebra $\Gamma^*(\mathfrak{F},\mathfrak{V})$ is the algebra of terms, where the A-operators have a variable arity $\alpha: \mathfrak{F} \rightarrow 2^{\mathbb{N}}$ (the set of all subsets of $\mathbb{N}$) such that

$$\alpha(f) = \{\alpha'(f)\} \qquad \text{if } f \notin \mathfrak{F}_A$$
$$\alpha(f) = \mathbb{N} \setminus \{0,1\} \qquad \text{otherwise}$$

■

Based on this background, the flattening operation $\overline{\ }$ is defined as follows:

Definition 4.2.2 ([BP85a])

Let be $t = f(t_1,...,t_n)$ a term. Then

$$\overline{t} = \begin{cases} t & \text{if } t \text{ is a constant or a variable} \\ f(\overline{t_1},...,\overline{t_n}) & \text{if } f \notin \mathfrak{F}_A \\ t' & \text{otherwise} \end{cases}$$

with t' results from t by replacing t_i by $\overline{t_i}$ if $top(t_i) \neq f$, and replacing t_i by $s_1,...,s_m$ if $\overline{t_i} = f(s_1,...,s_m)$.

■

Note that the flattened form of a term exists and is unique. Some helpful assertions about the flattening operator will be enumerated in the following lemma:

Lemma 4.2.3 [Ze89]

Let be $s, t \in \Gamma^*(\mathfrak{F},\mathfrak{V})$, σ a substitution and $>$ a simplification ordering:

- $top(t) = top(\overline{t})$
- $\overline{\overline{t}} = \overline{t}$
- $\overline{\sigma(\overline{t})} = \overline{\sigma(t)}$
- $s =_A t \iff \overline{s} = \overline{t}$
- $t \geq \overline{t}$
- $\{\overline{t}\} \gg \{\overline{t_1},...,\overline{t_n}\}$ if $t = f(t_1,...,t_n)$

■

In order to compare terms (w.r.t. the RPOS) with operators having variable arity, we do not need to change the definition of the RPOS. However, the deletion property ($f(...,t,...) > f(...,...)$) must be satisfied to preserve the property of being a simplification ordering. The recursive path ordering without status does have the deletion property (see [De82]). Unfortunately, the recursive path ordering with status does not preserve this characteristic. Consider a simple example: Let be $\tau(f)$ = left, $s = f(x,x,y)$ and $t = f(x,y)$. Due to the deletion property, s must be greater than t, but the comparison of the tuples (x,x,y) and (x,y) requires the orientation of $x = y$ which is impossible.

Thus, in order to guarantee the property of a simplification ordering, the status function of the RPOS must satisfy the multiset status of each varyadic function symbol. Consequently, each A-operator has to have multiset status.

Now, an A-compatible ordering could be defined as $s >_A t$ iff $\overline{s} >_{RPOS} \overline{t}$. Unfortunately, $>_A$ does not have the replacement property: Let be $f \in \mathfrak{F}_A$ and $f \triangleright g$. Then, $\overline{f(a,b)} >_{RPOS} \overline{g(a,b)}$ but $\overline{f(f(a,b),c)} = f(a,b,c) <_{RPOS} \overline{f(g(a,b),c)}$. Requiring the A-operators to be minimal w.r.t. the precedence - $(\forall f \in \mathfrak{F}_A)(\exists g \in \mathfrak{F})\ f \triangleright g$ - the ordering $>_A$ can be used for proving the A-termination.

Definition 4.2.4

Let $\triangleright$ be a precedence such that each A-operator is minimal. Furthermore, τ is a status function requiring multiset status of each A-operator:

$$s >_A t$$
iff
$$\overline{s} >_{RPOS} \overline{t}$$

■

Lemma 4.2.5

$>_A$ is a simplification ordering which is A-compatible and stable w.r.t. substitutions.

■

The following lemma helps to prove the above one:

Lemma 4.2.6

Let $\triangleright$ be a precedence such that each A-operator is minimal and τ is a status function requiring multiset status of each A-operator:

$$s >_{RPOS} t \quad \rightsquigarrow \quad \overline{s} >_{RPOS} \overline{t}$$

■

Due to lemma 4.2.5 and the theorem on page 20, $>_A$ can be used to prove the A-termination of term rewriting systems. However, the restriction of the precedence (each A-operator must be minimal) is very strong:

Example 4.2.7

Let be $\mathfrak{R}$: $(x + y) * z \rightarrow (x * z) + (y * z)$
$f(x) * f(y) \rightarrow f(x * y)$
E: $(x + y) + z = x + (y + z)$
$(x * y) * z = x * (y * z)$

The rule system requires $* \triangleright +$ (first rule) and $* \triangleright f$ (second rule), i.e. the A-operator $*$ is only sub-minimal. Therefore, we are not able to prove the E-termination of $\mathfrak{R}$ with the help of $>_A$.

■

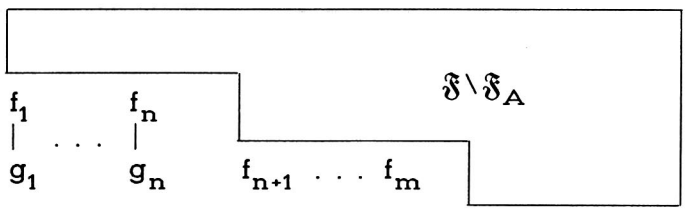
4.3 Associative distributivity

In order to permit sub-minimal (w.r.t. $\triangleright$) A-operators, the precedence must satisfy the following condition:

Definition 4.3.1 (Associative pair condition, [BP85])

A precedence $\triangleright$ has the associative pair condition if and only if $(\forall f \in \mathfrak{F}_A)$
- f is minimal or
- $(\exists g \in \mathfrak{F}_A)$ f is minimal w.r.t. $\mathfrak{F}\setminus\{g\}$.

Graphical explanation:



■

This condition requires that each A-operator must be either minimal or sub-minimal w.r.t. the precedence. The smaller element of a sub-minimal A-operator must also be an A-operator. However, as we know (example on page 23), the replacement property of $>_A$ is not fulfilled: If $* \triangleright +$, $x*(y+z) >_A (x*y)+(x*z)$ but $u*(x*(y+z)) <_A u*((x*y)+(x*z))$. Note that the term $u*(x*(y+z))$ (which should be greater) has to be flattened causing a reduction w.r.t. the recursive path ordering. On the other side, $u*((x*y)+(x*z))$ (the smaller term) is left unchanged. The trick consists of reducing $u*((x*y)+(x*z))$, too. I.e., we will force the replacement property using the intuitive idea that the smaller term has only to be reduced when the bigger one is reduced by flattening.

Definition 4.3.2 (Distributivity operation, [GL86])

Let $\triangleright$ be a precedence satisfying $f \triangleright g$ where f and g are associative operators. The distributivity operation $\delta : \Gamma \rightarrow \Gamma$ rewrites a term to an irreducible form with the system described by the two rules

D: $f(x,g(y,z)) \rightarrow g(f(x,y),f(x,z))$
$f(g(x,y),z) \rightarrow g(f(x,z),f(y,z))$

■

Obviously, for each pair f, g of AC-operators with $f \triangleright g$ such a rule system D can be found. Note that for each term there exists an irreducible form since this system terminates using the recursive path ordering with $f \triangleright g$ (due to the associative pair condition). But this normalform is not unique because the system is not confluent at all. Though, two normalforms of a term are AC-equivalent ([GL86]). This property is sufficient because the flattening operation maps two A-equivalent terms to terms being syntactically equal, i.e. AC-equivalent terms will result in C-equivalent terms. Thus, the flattened δ-normalforms of two terms are C-equivalent. Due to the C-compatibility of the RPOS it is possible to define an A-compatible ordering:

Definition 4.3.3

Let $\triangleright$ be a precedence satisfying the associative pair condition and τ requires multiset status of each A-operator:

$$s >_{AD} t \quad \text{iff} \quad - \ \overline{\delta(s)} >_{RPOS} \overline{\delta(t)} \qquad - \ s \overset{+}{\Longrightarrow}_{D/A} t$$

■

The comparison of two terms w.r.t. $>_{AD}$ requires both terms to be reduced to normalforms (applying the rules of D), first and then a comparison of the flattened normalforms w.r.t. the recursive path ordering with status.
The problem of proving the termination of a rule consisting of D-equivalent terms (their normalforms are syntactically equal) is solved by checking whether equivalent terms are D-equivalent ($s \overset{+}{\Longrightarrow}_{D/A} t$ with $\Longrightarrow_{D/A} = =_A \circ \Longrightarrow_D \circ =_A$).

Note that the associative pair condition guarantees that the transformation of distributing and flattening is well-defined and that $>_{AD}$ is in fact a simplification ordering.

Lemma 4.3.4

$>_{AD}$ is a simplification ordering which is A-compatible and stable w.r.t. substitutions.

■

Example 4.3.5 (Example 4.2.7 continued)

Let be $\mathfrak{R}$: $(x + y) * z \rightarrow (x * z) + (y * z)$
E: $(x + y) + z = x + (y + z)$
$(x * y) * z = x * (y * z)$

With the help of $>_{AD}$ the E-termination of $\mathfrak{R}$ can be proved since $\overline{\delta([x+y]*z]}$ $= [x*z]+[y*z] = \overline{\delta[[x*z]+[y*z]]}$ and $[x+y]*z \overset{+}{\Longrightarrow}_{D/A} [x*z]+[y*z]$.

■

Remark 4.3.6 [BP85]

In [BP85] a more general method than $>_{AD}$ is presented. If two terms have the same flattened δ-normalforms (i.e. $\overline{\delta[s]} =_{RPOS} \overline{\delta[t]}$) they could be still compared using some reduction ordering $\succ$ that is A-compatible and well-founded on every set $[t] = \{s \mid \overline{\delta[s]} = \overline{\delta[t]}\}$. Such an ordering is called admissible for the transformation $\bar{\delta}$. For example, comparing terms by the inverse of their sizes is an admissible ordering for the transformation used here. More precisely, $\succ$ can be defined as $s \succ t$ iff $|s| < |t|$ and no variable appears more often in s than in t.

■

4.4 Associative endomorphism

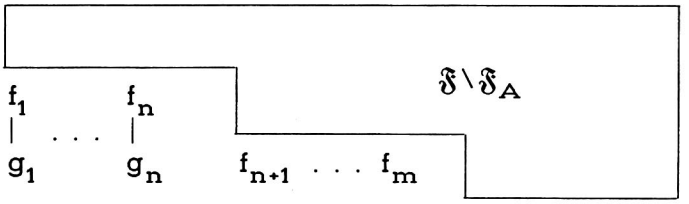
The generalization $>_{AD}$ of $>_A$ presented in section 4.3 admits precedences where an A-operator is greater than another A-operator. However, the rule f[x]*f[y] → f[x*y] [see example 4.2.7 on page 24] cannot be oriented with the help of $>_{AD}$ because the A-operator * must be greater than f which is not associative.

A solution for this problem consists of applying the basic concept of the former section's approach: Instead of using distributivity axioms we take rules similar to the endomorphism for pre-reducing terms. In order to exactly define this method, the set of admissible precedences must be established first.

Definition 4.4.1 [Simple pair condition, [Ze89] and [GL86]]

A precedence ▷ has the simple pair condition if and only if [∀f ∈ $\mathfrak{F}_A$]
- f is minimal or
- [∃ unary g ∈ $\mathfrak{F}$] f is minimal w.r.t. $\mathfrak{F}\backslash\{g\}$.

Graphical explanation:



■

This condition requires that each A-operator must be either minimal or sub-minimal w.r.t. the precedence. In contrast with the associative pair condition [definition 4.3.1 on page 25] the smaller element of a sub-minimal A-operator must be a unary function symbol. To guarantee the replacement property, the terms to be compared must be reduced in the following way.

Definition 4.4.2 [Endomorphism operation, [GL86]]

Let ▷ be a precedence satisfying f ▷ g where f is an associative operator and g is a unary function symbol. The endomorphism operation ε : Γ → Γ rewrites a term to an irreducible form with the system described by the two rules

E: f[g[x],y] → g[f[x,y]]
f[x,g[y]] → g[f[x,y]]

■

As for the approach of 4.3, for each pair f, g with $f \triangleright g$ there is such a rule system E. Note that this rule system (E) is confluent and terminating using the recursive path ordering with $f \triangleright g$. The normalform of a term t is denoted by $\varepsilon(t)$. It can be proved that two terms are AE-equivalent if and only if their flattened ε-normalforms are syntactically equal: $s =_{AE} t$ iff $\overline{\varepsilon(s)} = \overline{\varepsilon(t)}$. Based on this fact, another A-compatible ordering (see [GL86]) can be defined.
Analogous with $>_{AD}$, the comparison of two terms w.r.t. $>_{AE}$ requires both terms to be reduced to normalforms (applying the rules of E) and the flattened normalforms to be compared w.r.t. the recursive path ordering with status. If the normalforms are equivalent w.r.t. the RPOS, the test whether one term can be derived from the other one (w.r.t. $\overset{+}{\Longrightarrow}_{E/A}$) will be performed.

Definition 4.4.3

Let $\triangleright$ be a precedence satisfying the simple pair condition, τ requires multiset status of each A-operator:

$s >_{AE} t$ iff
- $\overline{\varepsilon(s)} >_{RPOS} \overline{\varepsilon(t)}$
- $s \overset{+}{\Longrightarrow}_{E/A} t$

■

Lemma 4.4.4

$>_{AE}$ is a simplification ordering which is A-compatible and stable w.r.t. substitutions.

■

Example 4.4.5 (Example 4.2.7 continued)

Let be $\mathfrak{R}$: $f(x) * f(y) \rightarrow f(x * y)$
E: $(x * y) * z = x * (y * z)$

With the help of $>_{AE}$ the termination of $\mathfrak{R}$ can be proved because $\overline{\varepsilon(f(x)*f(y))} = f(f(x*y)) >_{RPOS} f(x*y) = \overline{\varepsilon(f(x*y))}$ by using the subterm property of the RPOS.

■

4.5 AC-theories

This section deals with the extensions [to C-theories] of the orderings presented in sections 4.2, 4.3 and 4.4. Thus, we will now demonstrate the contents of [GL86] extended by the incorporation of status to the recursive path ordering.

Definition 4.5.1

Let τ be a status function such that each C-operator and each A-operator has multiset status.

a] Let $\triangleright$ be a precedence such that each A-operator is minimal:

$s >_{AC} t$
iff
$\overline{s} >_{RPOS} \overline{t}$

b] Let $\triangleright$ be a precedence satisfying the associative pair condition:

$s >_{ACD} t$
iff
- $\overline{\delta[s]} >_{RPOS} \overline{\delta[t]}$
- $s \overset{+}{\Longrightarrow}_{D/AC} t$

c] Let $\triangleright$ be a precedence satisfying the simple pair condition:

$s >_{ACE} t$
iff
- $\overline{\varepsilon[s]} >_{RPOS} \overline{\varepsilon[t]}$
- $s \overset{+}{\Longrightarrow}_{E/AC} t$

■

Guaranteeing the C-compatibility of the A-compatible orderings presented in the former sections, we only have to require multiset status of commutative operators. Thus, the union of the approach for C-theories with each method for A-theories must be considered. Note that the orderings $>_{ACD}$ and $>_{ACE}$ can be used without the second alternative [$s \overset{+}{\Longrightarrow} t$], but in this case, it is not possible to orient the D- or the E-rules.

Lemma 4.5.2

$>_{AC}$, $>_{ACD}$ and $>_{ACE}$ are simplification orderings which are AC-compatible and stable w.r.t. substitutions.

■

In addition to the assertions of [GL86], we can prove that $>_{ACD}$ ($>_{ACE}$) without the second condition ($s \overset{+}{\Longrightarrow} t$) is ACD-compatible (ACE-compatible).

Lemma 4.5.3

- The ordering $s > t$ iff $\overline{\delta(s)} >_{RPOS} \overline{\delta(t)}$ is ACD-compatible.
- The ordering $s > t$ iff $\overline{\varepsilon(s)} >_{RPOS} \overline{\varepsilon(t)}$ is ACE-compatible.

■

Example 4.5.4 Associative-commutative rings with unit ([BP85])

$\mathfrak{R}$: $x + 0 \rightarrow x$
$i(x) \rightarrow c * x$
$x + (c * x) \rightarrow 0$
$1 + c \rightarrow 0$
$c * c \rightarrow 1$
$x * (y + z) \rightarrow (x * y) + (x * z)$
$x * 0 \rightarrow 0$
$x * 1 \rightarrow x$

E: $(x + y) + z = x + (y + z)$
$x + y = y + x$
$(x * y) * z = x * (y * z)$
$x * y = y * x$

The classical convergent equational rewriting system for associative-commutative rings with unit contains the rules $x*i(y) \rightarrow i(x*y)$ and $i(x+y) \rightarrow i(x)+i(y)$. Therefore, the termination cannot be proved with the help of an ordering presented in this section since $*$ must be greater (w.r.t. $\triangleright$) than i and i greater than $+$: $* \triangleright i \triangleright +$. Bachmair and Plaisted introduce a different system $\mathfrak{G}$ for the same structure. The idea is to use a new constant c which represents i(1). Applying AC-completion to $\mathfrak{G}$ the rule system above will be obtained. The termination of this equational rewriting system can be proved with the help of $>_{ACD}$ corresponding to the precedence $i \triangleright * \triangleright +$, $i \triangleright c \triangleright 0$, $c \triangleright 1$ and the status function $\tau(+) = \tau(*) = mult$.

■

Remark 4.5.5 [[Gn88]]

In [Gn88] the conditions under which the orderings of this section can be total are investigated.
Assuming total precedence, $>_{ACD}$ [$>_{ACE}$] is total on $\Gamma[\mathfrak{F}]/ACD$ [$\Gamma[\mathfrak{F}]/ACE$] where $\Gamma[\mathfrak{F}]/ACD$ [$\Gamma[\mathfrak{F}]/ACE$] is the algebra $\Gamma[\mathfrak{F}]$ quotiented by AC-axioms and the distributivity [endomorphism] axiom.
Furthermore, Gnaedig found out that there exists no AC-commuting total ordering on $\Gamma[\mathfrak{F}]$. However, the problem is not yet solved for a total ordering on $\Gamma[\mathfrak{F}]/AC$.

■

All orderings presented in this chapter are based on a precedence $\triangleright$. This ordering on the operators is not clearly defined anywhere. Note that it could be a partial ordering as well as a quasi-ordering. Assuming quasi-precedences we will consider the following example.

Example 4.5.6

$\mathfrak{R}$: $\quad false \vee x \rightarrow true \wedge x$

E: $\quad (x \vee y) \vee z = x \vee (y \vee z)$
$\quad x \vee y = y \vee x$
$\quad (x \wedge y) \wedge z = x \wedge (y \wedge z)$
$\quad x \wedge y = y \wedge x$

$\triangleright$: $\quad \vee \simeq \wedge$
$\quad false \triangleright true$

It is obvious that the rule of $\mathfrak{R}$ can be oriented in the desired direction. However, this orientation is not stable w.r.t. substitutions: Let be $\sigma = \{x \leftarrow false \vee false\}$. Then, $\sigma(false \vee x) = false \vee (false \vee false) \neq true \wedge (false \vee false) = \sigma(true \wedge x)$. $>$ is any of the orderings of lemma 4.5.7.

■

This example induces that quasi-precedences on AC-operators are prohibited in order to guarantee the stability w.r.t. substitutions.

Lemma 4.5.7

$>_A$, $>_{AD}$, $>_{AE}$, $>_{AC}$, $>_{ACD}$ and $>_{ACE}$ are not stable w.r.t. substitutions if the precedence on the A[C]-operators is a quasi-ordering.

■

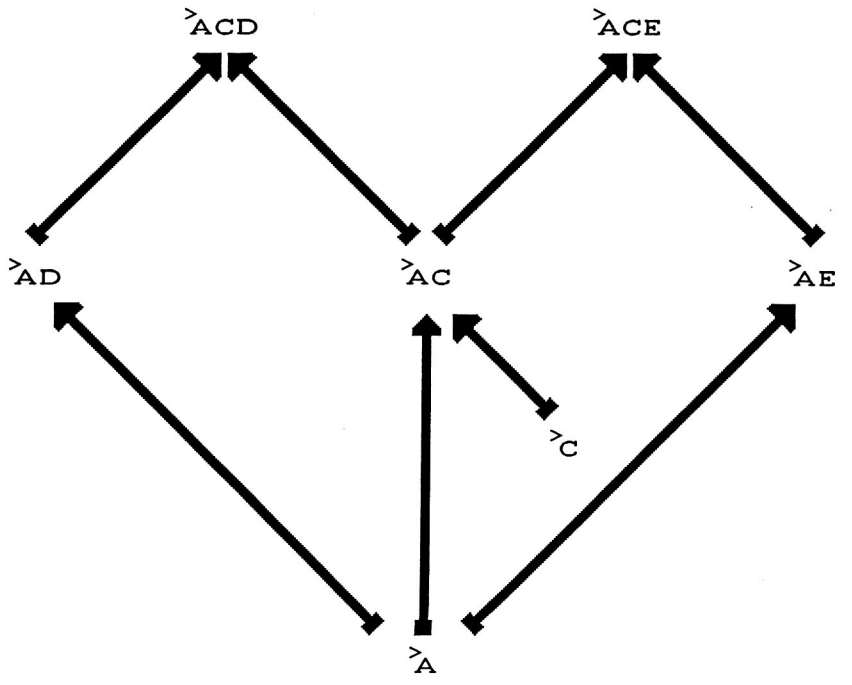
It is obvious that quasi-orderings on non AC-operators can still be used.

4.6 Comparison

Like in section 3.3 we will now compare the power of the orderings presented in this chapter. Note that we will only examine the relation between (not the cardinality of) two sets of comparable terms. The following lemma describes the comparison w.r.t. total precedences (we obtain the same results if partial orderings are used).

Lemma 4.6.1

Assuming arbitrary terms and total precedences (that are admissible related to the corresponding orderings), the following relations (cf. page 17) are valid:



■

A difference between the orderings presented here and those in other papers published is the additional status. It is obvious that the orderings without status are included in the corresponding orderings with status. The following example hints to this fact (see also example 6.18 on page 45):

Example 4.6.2 (Padawitz)

$\mathfrak{R}$: $x * 0 \to 0$
$x * s(y) \to (x * y) + x$
$1 \to s(0)$
$fac(0) \to 1$
$fac(s(x)) \to s(x) * fac(x)$
$floop(0,y) \to y$
$floop(s(x),y) \to floop(x,s(x) * y)$

E: $(x + y) + z = x + (y + z)$
$x + y = y + x$
$(x * y) * z = x * (y * z)$
$x * y = y * x$

$\triangleright$: $fac \triangleright * \triangleright +$
$fac \triangleright 1 \triangleright s \triangleright 0$
$floop \triangleright *$
$\tau(floop) = left$

floop is an iterative program for the factorial function. Its first argument is the argument of fac, while the second argument serves as an accumulator for the result of fac: $floop(x,y) = y * fac(x)$.

■

5 Path and decomposition orderings modulo AC

This chapter deals with the application of the concepts used in chapter 4 to the decomposition orderings of chapter 3. We will briefly recall those approaches:

- $>_C$ represents the RPOS with multiset status of C-operators
- $>_A$ requires flattening the terms and A-operators to be minimal w.r.t. $\triangleright$
- $>_{AD}$ transforms terms into their flattened D-normalforms by satisfying the associative pair condition
- $>_{AE}$ is similiar to $>_{AD}$ except for using E-normalforms instead of D-normalforms and the simple instead of the associative pair condition
- $>_{AC}$ is the union of $>_A$ and $>_C$
- $>_{ACD}$ incorporates multiset status of C-operators to $>_{AD}$
- $>_{ACE}$ extends $>_{AE}$ by requiring multiset status of C-operators

Note that an essential precondition must be guaranteed to use any of the orderings above: each A-operator (C-operator) must have multiset status. Another fact the presented orderings have in common is that they are based on the recursive path ordering with status. Our goal consists of extending each strategy by making the basic ordering stronger. More precisely, the recursive path ordering with status will be replaced by a more powerful ordering. In chapter 3, the hierarchy w.r.t. the potency of some well-known path and decomposition orderings is given. The three decomposition orderings RDOS, PSDS and IRDS cover the biggest part of the diagram. Therefore, we checked whether they are applicable to the methods of the former chapter. It turned out that there are no more restrictions (except those for the RPOS) needed. We once more would like to point out that quasi-precedences on AC-operators are forbidden.

For a clear-cut representation of the orderings we choose their parameters in the following way:

Definition 5.1

Let be $> \in \{>_A, >_C, >_{AC}, >_{AD}, >_{AE}, >_{ACD}, >_{ACE}\}$ and
$\triangleright \in \{>_{RDOS}, >_{PSDS}, >_{IRDS}\}$
two kinds of orderings.
Then, $>[\triangleright]$ denotes the method of $>$ where $>_{RPOS}$ is replaced by $\triangleright$.

■

For example, $s >_{AC}[>_{IRDS}] t$ iff $\overline{s} >_{IRDS} \overline{t}$ by guaranteeing that each A-operator is minimal w.r.t. $\triangleright$ and each A-/C-operator has multiset status.

The following theorem reflects the correctness of the promising manipulation of the AC-compatible orderings based on the recursive path ordering with status. The proof can be found in the appendix.

Theorem 5.2

Let be $\triangleright \in \{>_{RDOS}, >_{PSDS}, >_{IRDS}\}$.

- $>_C[\triangleright]$ are simplification orderings, C-compatible and stable w.r.t. substitutions.
- $>_A[\triangleright]$, $>_{AD}[\triangleright]$ and $>_{AE}[\triangleright]$ are simplification orderings, A-compatible and stable w.r.t. substitutions.
- $>_{AC}[\triangleright]$, $>_{ACD}[\triangleright]$ and $>_{ACE}[\triangleright]$ are simplification orderings, AC-compatible and stable w.r.t. substitutions.

■

We will now demonstrate the practical applicability of the presented orderings by vicariously proving the termination of an AC-rewriting system with $>_{AC}[>_{IRDS}]$ (that cannot be done with the help of the APO).

Example 5.3

Assuming we have the two boolean constructors ff (false) and $\neg$ (not). Furthermore, there is a complete definition of the boolean operator $\wedge$ (and) in addition to the rule $\neg\neg x \rightarrow x$. Then, the following system is a complete definition of the boolean implication ($\supset$).

$$\begin{aligned} ff \supset y &\rightarrow \neg ff \\ x \supset ff &\rightarrow \neg x \\ \neg x \supset \neg y &\rightarrow y \supset (x \wedge y) \end{aligned}$$

Consider $\wedge$ to be associative and commutative. Then, the AC-ordering on the improved recursive decomposition ordering $>_{AC}[>_{IRDS}]$ is able to prove the termination using the precedence $\supset \triangleright \neg \triangleright \wedge$.
Note that the last rule cannot be oriented in this way by the associative path ordering.

■

The theorem 5.2 guarantees the correctness of merging the concepts of the various versions of the associative path orderings with the recursive decomposition orderings RDOS, PSDS and IRDS. In order to complete this study we would like to point out that it is possible to apply this method to all other path and decomposition orderings of chapter 3. This results from the KNSS being equivalent to the IRDS, and the remaining orderings (not treated) being simple versions (without status) of the RDOS, PSDS and IRDS, respectively.

Summarizing, it is to remark that the strategy of the APO as well as the specialized (to A- or C-theories, only) and improved one (to the RPO with status) can be applied to the path and decomposition orderings presented in chapter 3. The power of the A-, C- and AC-orderings is induced by the power of the classical path and decomposition orderings (see lemma 3.3.2 on page 17) and by the power of the corresponding E-termination methods (see lemma 4.6.1 on page 33). Thus, $>_{ACD}[>_{IRDS}]$ and $>_{ACE}[>_{IRDS}]$ are the most powerful techniques presented in this report. Note that it is easier to construct a precedence for a given rule system with the help of a decomposition ordering than with the recursive path ordering (see [Ch84]).

6 A digest of examples

This chapter deals with the power as well as the limits of the practical applicability of the orderings presented. We collected some examples of rewrite systems ($\mathfrak{R}$) together with underlying theories (E). If there exists an associative path ordering (with status) the appropriate precedence is given ($\triangleright$). The version ($>_A$, $>_C$, $>_{AC}$, $>_{AD}$, $>_{AE}$, $>_{ACD}$ or $>_{ACE}$) of the APO which should be used is determined by the precedence and the equational theory.

For representing the rewriting systems we use x, y, z and u as variables. Function symbols will be denoted by h,k (binary function symbols), f, i, j, L, T (unary operators) and e (constant symbol). We also employ special operators to point to certain models. These symbols together with their meanings will be listed below:

*	multiplication on natural numbers ($\mathbb{N}$)
+	addition on $\mathbb{N}$
-	unary subtraction on $\mathbb{N}$
/	division on $\mathbb{N}$
sq	square function on $\mathbb{N}$
exp	exponential function : $\exp(x) = e^x$
b	binomial function
0,1	the natural numbers 0 and 1, respectively
s	successor : $s(x) = x + 1$
$\vee$	boolean or
$\wedge$	boolean and
$\oplus$	exclusive or
$\supset$	boolean implication
$\equiv$	boolean equivalence
$\neg$	boolean not
false	boolean false
true	boolean true

To enhance readability we will often use infix notation.

Example 6.1 [[Hu80a]]

$\mathfrak{R}$:
$$x + 0 \rightarrow x$$
$$x * 1 \rightarrow x$$
$$f(0) \rightarrow 1$$
$$f(x + y) \rightarrow f(x) * f(y)$$
$$(x + y) * z \rightarrow (x * z) + (y * z)$$

E:
$$(x + y) + z = x + (y + z)$$
$$x + y = y + x$$
$$(x * y) * z = x * (y * z)$$
$$x * y = y * x$$

$\triangleright$: $f \triangleright * \triangleright +$, $0 \triangleright 1$

■

Example 6.2 Abelian group theory

$\mathfrak{R}$:
$$x + 0 \rightarrow x$$
$$x + i(x) \rightarrow 0$$
$$i(0) \rightarrow 0$$
$$i(i(x)) \rightarrow x$$
$$i(x + y) \rightarrow i(x) + i(y)$$

E:
$$(x + y) + z = x + (y + z)$$
$$x + y = y + x$$

$\triangleright$: $i \triangleright 0 \triangleright +$

■

Example 6.3 [[PF86]]

$\mathfrak{R}$:
$$s(x) + y \rightarrow x + s(y)$$
$$s(x) + y \rightarrow s(x + y)$$
$$0 + y \rightarrow 0$$

E:
$$x + y = y + x$$

The first rule cannot be oriented since it is not E-terminating: $s(0)+0 \Longrightarrow_{\mathfrak{R}} 0+s(0) =_E s(0)+0$. For a complete definition of $+$, it is sufficient to use the second and the last rule. A termination proof of these two rules is based on the precedence $+ \triangleright s$ and the orderings including the transformation with endomorphism $[>_{ACE}]$.

■

Example 6.4 Exponential function [JM84]

$\mathfrak{R}$:
$$
\begin{aligned}
x + 0 &\to x \\
x * 1 &\to x \\
(x + y) * z &\to (x * z) + (y * z) \\
x * (y + z) &\to (x * y) + (x * z) \\
\exp(0) &\to 1 \\
\exp(x + y) &\to \exp(x) * \exp(y)
\end{aligned}
$$

E:
$$
\begin{aligned}
(x + y) + z &= x + (y + z) \\
x + y &= y + x \\
(x * y) * z &= x * (y * z) \\
x * y &= y * x
\end{aligned}
$$

$\triangleright$:
$$
\begin{aligned}
&\exp \triangleright * \triangleright + \\
&0 \triangleright 1
\end{aligned}
$$

■

Example 6.5 Boolean algebra

$\mathfrak{R}$:
$$
\begin{aligned}
x \oplus \text{false} &\to x \\
x \oplus x &\to \text{false} \\
x \wedge \text{false} &\to \text{false} \\
x \wedge \text{true} &\to x \\
x \wedge x &\to x \\
(x \oplus y) \wedge z &\to (x \wedge z) \oplus (y \wedge z)
\end{aligned}
$$

E:
$$
\begin{aligned}
(x \wedge y) \wedge z &= x \wedge (y \wedge z) \\
x \wedge y &= y \wedge x \\
(x \oplus y) \oplus z &= x \oplus (y \oplus z) \\
x \oplus y &= y \oplus x
\end{aligned}
$$

The E-termination of the rule system $\mathfrak{R}$ cannot be proved with the help of any ordering presented since the second rule requires $\oplus \triangleright \text{false}$ which does not satisfy the associative [or the simple] pair condition.

■

Example 6.6 Disjunctive normalform [De82]

$\mathfrak{R}$:
$$
\begin{aligned}
\neg\neg x &\to x \\
\neg(x \vee y) &\to \neg\neg\neg x \wedge \neg\neg\neg y \\
\neg(x \wedge y) &\to \neg\neg\neg x \vee \neg\neg\neg y \\
x \wedge x &\to x \\
x \vee x &\to x
\end{aligned}
$$

E: $(x \wedge y) \wedge z = x \wedge (y \wedge z)$
$x \wedge y = y \wedge x$
$(x \vee y) \vee z = x \vee (y \vee z)$
$x \vee y = y \vee x$

Note that there does not exist an AC-path or decomposition ordering which guarantees the termination of $\mathfrak{R}$ modulo E since even the classical termination of $\mathfrak{R}$ cannot be shown with any path or decomposition ordering. However, the associative-commutative Knuth-Bendix ordering *is able to prove the E-termination (see [St89b]).*

Obviously, the system containing the rules

$\neg\neg x \rightarrow x$
$\neg(x \wedge y) \rightarrow \neg x \vee \neg y$
$\neg(x \vee y) \rightarrow \neg x \wedge \neg y$
$x \wedge (y \vee z) \rightarrow (x \wedge y) \vee (x \wedge z)$
$(x \vee y) \wedge z \rightarrow (x \wedge z) \vee (y \wedge z)$

is terminating modulo the theory where $\wedge$ and $\vee$ are associative-commutative (if $\neg \triangleright \wedge \triangleright \vee$).

■

Example 6.7 Unary integer addition (Dershowitz)

$\mathfrak{R}$: $x + 0 \rightarrow x$
$0 + y \rightarrow y$
$-0 \rightarrow 0$
$-(-x) + y \rightarrow x + (-y)$
$--x \rightarrow x$
$(-1) + 1 \rightarrow 0$
$-(x + 1) + 1 \rightarrow -x$

E: $(x + y) + z = x + (y + z)$

$\triangleright$: $- \triangleright +$
$1 \triangleright 0$

■

Example 6.8 Group theory

$\mathfrak{R}$: $(x/x)/((y/y)/y) \rightarrow y$
$(x/y)/(z/y) \rightarrow x/z$
$x/x \rightarrow 1$
$1/x \rightarrow i(x)$
$x/i(y) \rightarrow x * y$

E: $(x * y) * z = x * (y * z)$
$x * y = y * x$

▷: $/ \triangleright 1$
$/ \triangleright i$
$/ \triangleright *$

■

Example 6.9 Group theory [KB70]

ℜ: $1 * y \rightarrow y$
$e * y \rightarrow y$
$i(x) * x \rightarrow 1$
$j(x) * x \rightarrow e$

E: $(x * y) * z = x * (y * z)$

▷: $i \triangleright 1$
$j \triangleright e$

■

Example 6.10 Square function

ℜ: $sq(0) \rightarrow 0$
$sq(s(x)) \rightarrow s((sq(x) + x) + x)$
$x + 0 \rightarrow x$
$x + s(y) \rightarrow s(x + y)$

E: $(x + y) + z = x + (y + z)$
$x + y = y + x$

▷: $sq \triangleright + \triangleright s$

■

Example 6.11

ℜ: $x * (y + z) \rightarrow (x * y) + (x * z)$
$(u + (x * y)) + (x * z) \rightarrow u + (x * (y + z))$

E: $(x + y) + z = x + (y + z)$

The second rule of $\mathfrak{R}$ cannot be oriented with the help of any path or decomposition ordering (with $\tau(+)$=mult). However, the ordering on polynomial interpretations restricted to A-theories suffices using the following interpretations: $I(*)[x,y] = xy$ and $I(+)[x,y] = x + y + 1$.

■

Example 6.12 Milner's theory of nondeterministic machines ([Hu80b])

$\mathfrak{R}$:
$$\begin{aligned}
x + x &\to x\\
x + 0 &\to x\\
T[x] + x &\to T[x]\\
T[x + y] + T[y] &\to T[x + T[y]]\\
L[x + T[y]] &\to L[x + y] + L[y]\\
T[x + y] + x &\to T[x + y]\\
T[T[x]] &\to T[x]\\
L[T[x]] &\to L[x]
\end{aligned}$$

E:
$$\begin{aligned}
[x + y] + z &= x + [y + z]\\
x + y &= y + x
\end{aligned}$$

$\triangleright$:
$$L \triangleright + \triangleright T$$

■

Example 6.13 Arithmetic theories ([Hu80b])

$\mathfrak{R}$:
$$\begin{aligned}
0 + y &\to y\\
0 * y &\to 0\\
s[x] + y &\to s[x + y]\\
s[x] * y &\to [x * y] + y\\
x * [y + z] &\to [x * y] + [x * z]
\end{aligned}$$

E:
$$\begin{aligned}
[x + y] + z &= x + [y + z]\\
x + y &= y + x\\
[x * y] * z &= x * [y * z]\\
x * y &= y * x
\end{aligned}$$

It is impossible to guarantee the E-termination of $\mathfrak{R}$ with one of the presented methods since we have to require that $+ \triangleright s$ (third rule) is valid together with $* \triangleright +$ (last rule). A combination of the associative and the simple pair condition (and therefore a fusion of ε and δ) could help.

■

Example 6.14 Taussky group

$\mathfrak{R}$: $1 * 1 \rightarrow 1$
$x * i(x) \rightarrow 1$
$h(x * y , y) \rightarrow k(x * y , x)$
$k(1,y) \rightarrow y$

E: $(x * y) * z = x * (y * z)$

$\triangleright$: $i \triangleright 1$
$h \triangleright k$

■

Example 6.15 Boolean ring

$\mathfrak{R}$: $\neg x \rightarrow x \oplus true$
$x \supset y \rightarrow (x \wedge y) \oplus (x \oplus true)$
$x \vee y \rightarrow (x \wedge y) \oplus (x \oplus y)$
$x \equiv y \rightarrow x \oplus (y \oplus true)$

E: $(x \wedge y) \wedge z = x \wedge (y \wedge z)$
$x \wedge y = y \wedge x$

$\triangleright$: $\neg \triangleright \oplus$
$\neg \triangleright true$
$\supset \triangleright \oplus$
$\supset \triangleright \wedge$
$\supset \triangleright true$
$\vee \triangleright \oplus$
$\vee \triangleright \wedge$
$\equiv \triangleright \oplus$
$\equiv \triangleright true$

■

Example 6.16 Addition on integers modulo 2

$\mathfrak{R}$: $s(s(0)) \rightarrow 0$
$x + 0 \rightarrow x$
$x + x \rightarrow 0$

E: $(x + y) + z = x + (y + z)$
$x + y = y + x$

With the help of the interpretations $I[+][x,y] = x + y + 2$, $I[0][] = 1$ and $I[s][x] = x + 1$, the system $\mathfrak{R}$ is E-terminating. However, there is no path or decomposition ordering satisfying the desired property since the associative-commutative + must be greater than the constant symbol 0.

■

Example 6.17 Binomial coefficients

$\mathfrak{R}$: $x + 0 \rightarrow x$
$x + s[y] \rightarrow s[x + y]$
$b[0,s[y]] \rightarrow 0$
$b[x,0] \rightarrow s[0]$
$b[s[x],s[y]] \rightarrow b[x,s[y]] + b[x,y]$

E: $[x + y] + z = x + [y + z]$
$x + y = y + x$

$\triangleright$: $b \triangleright + \triangleright s$

■

Example 6.18 [Jantke & Thomas]

$\mathfrak{R}$: $x + 0 \rightarrow x$
$x + s[y] \rightarrow s[x + y]$
$f[g[f[x]]] \rightarrow f[h[s[0],x]]$
$f[g[h[x,y],y]] \rightarrow f[h[s[x],y]]$
$f[h[x,h[y,z]]] \rightarrow f[h[x + y,z]]$

E: $[x + y] + z = x + [y + z]$
$x + y = y + x$

$\triangleright$: $g \triangleright h \triangleright + \triangleright s$
$g \triangleright 0$
$\tau[h] = right$

■

Example 6.19 Vector spaces [Pl83]

In [Pl83] a solution of the problem of handling more than two related associative-commutative operators is given. For example, vector spaces have two addition and two multiplication operations: scalar addition [s-add], vector addition [v-add], scalar-scalar multiplication [s-mult] and

scalar-vector multiplication (v-mult). It is obvious that rings satisfy the associative pair condition (mult ▷ add). But, using vector spaces, both s-add and v-add must be considered less than v-mult since for example (2 + 3)X → 2X + 3X and 2(X + Y) → 2X + 2Y (where X and Y are vectors). Plaisted identifies s-add and v-add which causes the associative pair condition to be satisfied. Note that this technique is not stable w.r.t. substitutions, in general (lemma 4.5.7 on page 32).

However, it could be possible to extend the associative pair condition by allowing linear sequences (w.r.t. ▷) of more than two associative-commutative operators. In that case a hierarchy (induced by ▷) of distributivity axioms must probably be applied to the terms to be compared.

■

7 Conclusion

This paper introduces several classes of termination orderings for associative and (or) commutative term rewriting systems. These orderings extend the well-known associative path orderings (see for example [BP85], [GL86]) by using a stronger (than the RPO) underlying ordering. More precisely, we apply the basic features of the associative path ordering (APO) to a variety of path and decomposition orderings. The main ideas of the APO as well as those of the new orderings presented are the following ones:

- Applying distributivity (or endomorphism, resp.) axioms to the terms to be compared by requiring the associative (simple, resp.) pair condition and then
- Flattening the transformed terms and
- Comparing the flattened terms w.r.t. a classical term ordering.

In the case of the APO, the underlying term ordering is represented by the recursive path ordering. We succeeded in enhancing the APO by substituting several path and decomposition orderings for the RPO. It is possible to use

- the path of subterms ordering of Plaisted ([Pl78a]),
- the path ordering of Kapur, Narendran and Sivakumar ([KNS85]),
- the recursive decomposition ordering of Jouannaud, Lescanne and Reinig ([JLR82]) or
- the improved recursive decomposition ordering of Rusinowitch ([Ru87], [St89a])

instead of the recursive path ordering of Dershowitz ([De82]).

Additionally, two different kinds of extensions of the associative path ordering were presented:

- APO for either A-theories or C-theories, only. The restriction to only one of the two theories results in weakening the preconditions the precedence must satisfy. Also, the distributing operation is no longer responsible for the comparison. From a practical point of view, these restrictions could be a facilitation.

- Using the recursive path ordering with status instead of the RPO will increase the applicability in practice. The demand for multiset status of associative or commutative function symbols is not too restrictive.

It must be remarked that these two improvements demonstrated for the APO, can also be applied to all other path and decomposition orderings mentioned above.

Another supplement to the papers of Bachmair, Plaisted and Gnaedig, Lescanne is the fact that the APO including distribution of terms [applying endomorphism, respectively] is ACD-compatible [ACE-compatible, respectively]. Furthermore, we were able to show that quasi-orderings on AC-operators cannot be used since they injure the stability w.r.t. substitutions. Concluding, it is to remark that we have also succeeded in proving the subterm property of all the orderings presented [including the different versions of the APO].

There only exist a few other orderings proving AC-termination: the polynomial orderings on restricted interpretations of operators [BL87a] and the associative-commutative Knuth-Bendix orderings [St89b]. Comparing them, the following relations can be detected:

The AC-orderings based on path and decomposition orderings

- are more powerful than the APO [see example 5.3 on page 36],
- overlap with the polynomial orderings on restricted interpretations of operators:

  $[\neg x \supset y) \vee z \;>_{POL}\; \neg[\neg y \wedge \neg z] \vee x$ and
  $x \supset [y \vee ff] \;>_{path}\; [x \supset y) \vee x$
  with $\wedge, \vee \in \mathfrak{F}_{AC}$
- overlap with the associative-commutative Knuth-Bendix orderings [see example above]

Note that the APO overlaps with the POL and the ACK, and the POL overlaps with the ACK [see St89b].

The generalization of the presented approach for equational theories other than associative and commutative ones as well as a new method for AC-termination which lifts a total order on the AC-equivalence classes to general classes will be part of future plans.

Acknowledgement

There remains the pleasant duty to express my appreciation to Jürgen Avenhaus, Roland Fettig, Rita Kohl, Klaus Madlener, Inger Sonntag and Michael Zehnter for polishing the manuscript of this paper.

References

For a short-cut representation of the sources of the references we will use the following common abbreviations for conferences, journals and societies:

ACM	Association for Computing Machinery
CAAP	Colloquium on Trees in Algebra and Programming
CADE	Conference on Automated Deduction
CRIN	Centre de Recherche en Informatique de Nancy
ICALP	International Colloquium on Automata, Languages and Programming
IFIP	International Federation for Information Processing
IJCAI	International Joint Conference on Artificial Intelligence
INRIA	Institut National de Recherche en Informatique et en Automatique
IPL	Information Processing Letters
IRIA	Institut de Recherche d'Informatique et d'Automatique
J. SC	Journal of Symbolic Computation
J. TCS	Journal of Theoretical Computer Science
LNCS	Lecture Notes in Computer Science (Springer, Berlin)
MFCS	Mathematical Foundations of Computer Science
NSF	National Science Foundation
RAIRO	Revue francaise d'Automatique, d'Informatique et de Recherche Operationnelle / Informatique theorique
RTA	Rewriting Techniques and Applications
SCP	Science of Computer Programming, Elsevier Science Publishers B.V., North Holland
SIAM	Society for Industrial and Applied Mathematics
TAPSOFT	International Joint Conference on Theory and Practice of Software Development

[Ad87] Christian Adler
Vervollständigung von Termersetzungssystemen modulo einer gleichungsdefinierten Theorie
Master Thesis, Kaiserslautern, W. Germany, 1987

[AM89] Jürgen Avenhaus, Klaus E. Madlener
Term rewriting and equational reasoning
In: Formal Techniques in Artificial Intelligence - A source book, R.B. Banerji [ed.], Academic Press, 1989

[BD87] Leo Bachmair / Nachum Dershowitz
Completion for rewriting modulo a congruence
Proc. 2nd RTA, Bordeaux, France, 1987, LNCS 256, pp. 241-254

[BD86] Leo Bachmair / Nachum Dershowitz
Commutation, transformation and termination
Proc. 8th CADE, Oxford, U.K., 1986, LNCS 230, pp. 5-20

[Be86] Francoise Bellegarde
Rewriting systems on FP-expressions to reduce the number of sequences yielded
SCP 6, 1986, pp. 11-34

[BL88] Francoise Bellegarde / Pièrre Lescanne
Termination proofs based on transformation techniques
Internal Report, CRIN, Nancy, France, September 1988

[BL87] Francoise Bellegarde / Pièrre Lescanne
Transformation ordering
Proc. 12th CAAP [TAPSOFT], Pisa, Italy, 1987, LNCS 249, pp. 69-80

[BL87a] Ahlem Ben Cherifa / Pièrre Lescanne
Termination of rewriting systems by polynomial interpretations and its implementation
SCP 9 [2], October 1987, pp. 137-160

[BL86] Ahlem Ben Cherifa / Pièrre Lescanne
An actual implementation of a procedure that mechanically proves termination of rewriting systems based on inequalities between polynomial interpretations
Proc. 8th CADE, Oxford, U.K., 1986, LNCS 230, pp. 42-51

[BP85] Leo Bachmair / David A. Plaisted
Termination orderings for associative-commutative rewriting systems
J. SC 1, 1985, pp. 329-349

[BP85a] Leo Bachmair / David A. Plaisted
Associative path orderings
Proc. 1st RTA, Dijon, France, May 1985, LNCS 202, pp. 241-254

[Ch84] Ghislaine Choque
How to compute a complete set of minimal incrementations with the recursive decomposition ordering ?
Internal Report 84-R-056, CRIN, Nancy, France, 1984

[Da88] Max Dauchet
Termination of rewriting is undecidable in the one-rule case
Proc. 13th Symposium of MFCS, Carlsbad, CSSR, September 1988, LNCS 324, pp. 262-270

[De87] Nachum Dershowitz
Termination of rewriting
J. SC 3, 1987, pp. 69-116

[De87a] Nachum Dershowitz
Corrigendum - Termination of rewriting
J. SC 4, 1987, pp. 409-410

[De85] Nachum Dershowitz
Termination
Proc. 1st RTA, Dijon, France, May 1985, LNCS 202, pp. 180-224

[De82] Nachum Dershowitz
Orderings for term rewriting systems
J. TCS 17 (3), March 1982, pp. 279-301

[De81] Nachum Dershowitz
Termination of linear term rewriting systems
Proc. 8th ICALP, Acre (Akko), Israel, July 1981, LNCS 115, pp. 448-458

[De79] Nachum Dershowitz
A note on simplification orderings
IPL 9 (5), November 1979, pp. 212-215

[DHJP83] Nachum Dershowitz / Jieh Hsiang / N. Alan Josephson / David A. Plaisted
Associative-commutative rewriting
Proc. 8th IJCAI, Karlsruhe, W. Germany, August 1983, pp. 940-944

[DM79] Nachum Dershowitz / Zohar Manna
Proving termination with multiset orderings
Communications of the ACM 22, August 1979, pp. 465-476

[FH83] Francois Fages / Gérard Huet
Complete sets of unifiers and matchers in equational theories
Proc. 8th CAAP, L'Aquila, Italy, March 1983, LNCS 159, pp. 205-220

[GD88] Bernhard Gramlich / Jörg Denzinger
Efficient AC-matching using constraint propagation
SEKI-Report SR-88-15, Kaiserslautern, W. Germany, 1988

[GL86] Isabelle Gnaedig / Pièrre Lescanne
Proving termination of associative-commutative rewriting systems by rewriting
Proc. 8th CADE, Oxford, U.K., 1986, LNCS 230, pp. 52-60

[Gn88] Isabelle Gnaedig
Total orderings for equational theories
Working document, Nancy, France, 1988

[Gn87] Isabelle Gnaedig
Investigations on termination of equational rewriting
Report INRIA, Le Chesnay, 1987

[HL78] Gérard Huet / Dallas S. Lankford
On the uniform halting problem for term rewriting systems
Rapport Laboria 283, IRIA, Paris, INRIA Rocquencourt, France, March 1978

[HO80] Gérard Huet, Derek C. Oppen
Equations and rewrite rules - A survey
Formal Language Theory - Perspectives and open problems, R. Book [ed.], Academic Press, 1980, pp. 349-405

[Hu80a] Gérard Huet
Confluent reductions: Abstract properties and applications to term rewriting systems
J. ACM 27, 1980, pp. 797-821

[Hu80b] Jean-Marie Hullot
A catalogue of canonical term rewriting systems
Technical Report CSL-113, SRI International, Menlo Park, U.S.A., April 1980

[JK86] Jean-Pièrre Jouannaud / Hélène Kirchner
Completion of a set of rules modulo a set of equations
SIAM J. Computing 15 (4), 1986, pp. 1155-1194

[JKR83] Jean-Pièrre Jouannaud / Hélène Kirchner / Jean-Luc Rémy
Church-Rosser properties of weakly terminating equational term rewriting systems
Proc. 8th IJCAI, Karlsruhe, W. Germany, August 1983, pp. 909-915

[JLR82] Jean-Pièrre Jouannaud / Pièrre Lescanne / Fernand Reinig
Recursive decomposition ordering
IFIP Working Conference on Formal Description of Programming Concepts II (D. Bjørner, ed.), Garmisch-Partenkirchen, W. Germany, 1982, pp. 331-348

[JM84] Jean-Pièrre Jouannaud / Miguel Muñoz
Termination of a set of rules modulo a set of equations
Proc. 7th CADE, Napa, California, 1984, LNCS 170, pp. 175-193

[Jo83] Jean-Pièrre Jouannaud
Confluent and coherent equational term rewriting systems - Application to proofs in abstract data types
Proc. CAAP, L'Aquila, Italy, 1983, LNCS 159, pp. 269-283

[KB70] Donald E. Knuth / Peter B. Bendix
Simple word problems in universal algebras
Computational Problems in abstract algebra, Pergamon Press, 1970, pp. 263-297

[KL80] Sam Kamin / Jean-Jacques Lévy
Attempts for generalizing the recursive path orderings
Unpublished manuscript, Urbana, Illinois, 1980

[KN85] M. S. Krishnamoorthy / Paliath Narendran
Note on recursive path ordering
J. TCS 40, 1985, pp. 323-328

[KNS85] Deepak Kapur / Paliath Narendran / G. Sivakumar
A path ordering for proving termination of term rewriting systems
Proc. 10th CAAP, Berlin, W. Germany, March 1985, LNCS 185, pp. 173-187

[La79] Dallas S. Lankford
On proving term rewriting systems are noetherian
Memo MTP-3, Ruston, Louisiana, 1979

[La77] Dallas S. Lankford
Some approaches to equality for computational logic: A survey and assessment
Report ATP-36, University of Texas, Spring 1977

[LB77a] Dallas S. Lankford / A. M. Ballantyne
Decision procedures for simple equational theories with commutative-associative axioms: Complete sets of commutative-associative reductions
Technical Report ATP-39, University of Texas at Austin, Department of Mathematics and Computer Science, August 1977

[LB77b] Dallas S. Lankford / A. M. Ballantyne
Decision procedures for simple equational theories with commutative axioms: Complete sets of commutative reductions
Technical Report ATP-35, University of Texas at Austin, Department of Mathematics and Computer Science, March 1977

[Le87] Pièrre Lescanne
On the recursive decomposition ordering with lexicographical status and other related orderings
Preprint, December 1987

[Le84] Pièrre Lescanne
Uniform termination of term rewriting systems - The recursive decomposition ordering with status
Proc. 9th CAAP, Bordeaux, France, 1984, pp. 182-194

[Le83] Pièrre Lescanne
How to prove termination? An approach to the implementation of a new recursive decomposition ordering
Proc. NSF Workshop on the Rewrite Rule Laboratory, Guttag & Kapur & Musser (eds.), General Electric Research and Development Center Report 84 GEN008, September 1983, pp. 109-121

[Le82] Pièrre Lescanne
Some properties of decomposition ordering, a simplification ordering to prove termination of rewriting systems
RAIRO Theoretical Informatics 16 (4), 1982, pp. 331-347

[Le81] Pièrre Lescanne
Decomposition ordering as a tool to prove the termination of rewriting systems
Proc. 7th IJCAI, Vancouver, Canada, August 1981, pp. 548-550

[Ma87] Ursula Martin
How to choose the weights in the Knuth-Bendix ordering
Proc. 2nd RTA, Bordeaux, France, May 1987, LNCS 256, pp. 42-53

[PF86] Sara Porat / Nissim Francez
Full-commutation and fair-termination in equational (and combined) term-rewriting systems
Proc. 8th CADE, Oxford, U.K., July 1986, LNCS 230, pp. 21-41

[Pl83] David A. Plaisted
An associative path ordering
Proc. NFS Workshop on the Rewrite Rule Laboratory, Schenectady, U.S.A., November 1983, pp. 123-136

[Pl78a] David A. Plaisted
A recursively defined ordering for proving termination of term rewriting systems
Internal Report, University of Illinois, Urbana, U.S.A., September 1978

[Pl78b] David A. Plaisted
Well-founded orderings for proving termination of systems of rewrite rules
Internal Report, University of Illinois, Urbana, U.S.A., July 1978

[PS81] Gerald E. Peterson / Mark E. Stickel
Complete sets of reductions for some equational theories
Journal of the ACM 28 (2), April 1981, pp. 233-264

[Re81] Férnand Reinig
Les ordres de décomposition: Un outil incrémental pour prouver la términaison finie de systèmes de réécriture de termes
Thèse de 3ème cycle, Université de Nancy I & CRIN, France, Octobre 1981

[RJ81] Férnand Reinig / Jean-Pièrre Jouannaud
Decomposition orderings - A new family of recursive simplification orderings
Internal Report, CRIN, Nancy, France, June 1981

[Ru87] Michael Rusinowitch
Path of subterms ordering and recursive decomposition ordering revisited
J. SC 3, 1987, pp. 117-131

[Si89] Jörg H. Siekmann
Unification Theory
J. SC 7, 1989, pp. 207-274

[St89a] Joachim Steinbach
Extensions and comparison of simplification orderings
Proc. 3rd RTA, Chapel Hill, North Carolina, U.S.A., April 1989, pp. 434-448

[St89b] Joachim Steinbach
Proving termination of associative-commutative rewriting systems using the Knuth-Bendix ordering
SEKI-Report SR-89-13, Kaiserslautern, W. Germany, 1989

[St88a] Joachim Steinbach
Term orderings with status
SEKI-Report SR-88-12, Kaiserslautern, W. Germany, 1988

[St88b] Joachim Steinbach
Comparison of simplification orderings
SEKI-Report SR-88-02, Kaiserslautern, W. Germany, 1988

[St86] Joachim Steinbach
Ordnungen für Term-Ersetzungssysteme
Master Thesis, Kaiserslautern, W. Germany, June 1986

[SZ89] Joachim Steinbach / Michael Zehnter
Polynomial orderings - Known and new results
SEKI-Report, Kaiserslautern, W. Germany, 1989, forthcoming

[Ze89] Michael Zehnter
Theorievollständige Ordnungen - Eine Übersicht
Project report, Kaiserslautern, W. Germany, November 1989

Appendix: Proofs

Lemma 4.1.2 $>_C$ is C-compatible.

Proof: We have to show that $s =_C s' >_{RPOS} t' =_C t$ implies $s >_{RPOS} t$. It is sufficient to prove that $s =_C s' >_{RPOS} t'$ implies $s >_{RPOS} t'$ as well as $s' >_{RPOS} t' =_C t$ implies $s' >_{RPOS} t$. We will prove the first assertion by induction on $|s'| + |t'|$:

i) $top(s') \triangleright top(t') \wedge \{s'\} \gg_{RPOS} args(t')$

$\rightsquigarrow$ · $top(s) \triangleright top(t')$
since $top(s) = top(s')$

· $\{s\} \gg_{RPOS} args(t')$
by induction hypothesis

$\rightsquigarrow$ $s >_{RPOS} t'$
by definition of the RPOS

ii) $top(s') = top(t') \wedge \tau(top(s')) = mult \wedge args(s') \gg_{RPOS} args(t')$

$\rightsquigarrow$ $args(s) \gg_{RPOS} args(t')$
since $(\exists\pi)\ s'_i =_C s_{\pi(i)}$ and by induction hypothesis

$\rightsquigarrow$ $s >_{RPOS} t'$
since $top(s) = top(s')$ and by definition of the RPOS

iii) $top(s') = top(t') \wedge \tau(top(s')) = left \wedge \{s'\} \gg_{RPOS} args(t') \wedge args(s') >^{lex}_{RPOS} args(t')$

$\rightsquigarrow$ · $\{s\} \gg_{RPOS} args(t')$
since $s =_C s'$ and by induction hypothesis

· $args(s) >^{lex}_{RPOS} args(t')$
since $top(s) = top(s') \notin \mathfrak{F}_C$, $s'_i =_C s_i$ and by induction hypthesis

$\rightsquigarrow$ $s >_{RPOS} t'$
by definition of the RPOS

iv] $args[s'] \geq_{RPOS} \{t'\}$

$\rightsquigarrow$ $(\exists s_i')\ s_i' \geq_{RPOS} t'$

· $s_i' =_{RPOS} t'$
$\rightsquigarrow$ $(\exists s_j)\ s_j =_{RPOS} t'$
since $s_j =_C s_i$ and $=_C \subseteq =_{RPOS}$

$\rightsquigarrow$ $s >_{RPOS} t'$
since the RPOS has the subterm property

· $s_i' >_{RPOS} t'$
$\rightsquigarrow$ $(\exists s_j)\ s_j =_C s_i'$
since $s' =_C s$

$\rightsquigarrow$ $s_j >_{RPOS} t'$
by induction hypothesis

$\rightsquigarrow$ $s >_{RPOS} t'$
by definition of the RPOS

■

Lemma 4.2.5 $>_A$ is a simplification ordering which is A-compatible and stable w.r.t. substitutions.

Proof: i] $>_A$ is a partial ordering on $\Gamma[\mathfrak{F}]$ since the RPOS is a partial ordering on $\Gamma^*[\mathfrak{F}]$

ii] $>_A$ is A-compatible:

$s' =_A s >_A t =_A t' \quad \rightsquigarrow \quad s' >_A t'$
since $r =_A r' \quad \rightsquigarrow \quad \bar{r} = \bar{r}'$ (Lemma 4.2.3)

iii] $>_A$ has the replacement property:

$s >_A t$

$\rightsquigarrow$ $\bar{s} >_{RPOS} \bar{t}$
by definition of $>_A$

$\rightsquigarrow$ $f[\ldots, \bar{s}, \ldots] >_{RPOS} f[\ldots, \bar{t}, \ldots]$
since the RPOS has the replacement property

$\rightsquigarrow$ $\overline{f[\ldots, \bar{s}, \ldots]} >_{RPOS} \overline{f[\ldots, \bar{t}, \ldots]}$
with the help of lemma 4.2.6

$\rightsquigarrow$ $\overline{f[\ldots, s, \ldots]} >_{RPOS} \overline{f[\ldots, t, \ldots]}$
with the help of lemma 4.2.3 $[\bar{\bar{t}} = \bar{t}]$

$\rightsquigarrow$ $f[\ldots, s, \ldots] >_A f[\ldots, t, \ldots]$
by definition of $>_A$

iv] $>_A$ has the subterm property:

$f[\ldots, t, \ldots] >_{RPOS} t$
by the subterm property of the RPOS

$\rightsquigarrow$ $\overline{f[\ldots, t, \ldots]} >_{RPOS} \bar{t}$
with the help of lemma 4.2.6

$\rightsquigarrow$ $f[\ldots, t, \ldots] >_A t$
by definition of $>_A$

v] $>_A$ has the deletion property:
$f[\ldots, t, \ldots] >_A f[\ldots, \ldots]$ if f is a varyadic function symbol: This is valid since the RPO without status has the deletion property and each varyadic operator [associative function symbol] has multiset status

vi] $>_A$ is stable w.r.t substitutions:

$s >_A t$

$\rightsquigarrow$ $\bar{s} >_{RPOS} \bar{t}$
by definition of $>_A$

$\rightsquigarrow$ $[\forall\sigma]\ \sigma[\bar{s}] >_{RPOS} \sigma[\bar{t}]$
by the stability of the RPOS

$\rightsquigarrow$ $[\forall\sigma]\ \overline{\sigma[\bar{s}]} >_{RPOS} \overline{\sigma[\bar{t}]}$
with the help of lemma 4.2.6

$\rightsquigarrow$ $[\forall\sigma]\ \overline{\sigma[s]} >_{RPOS} \overline{\sigma[t]}$
with the help of lemma 4.2.3 $[\overline{\sigma[\bar{t}]} = \overline{\sigma[t]}]$

$\rightsquigarrow$ $[\forall\sigma]\ \sigma[s] >_A \sigma[t]$
by definition of $>_A$

■

Lemma 4.2.6 Let $\triangleright$ be a precedence such that each A-operator is minimal and τ is a status function requiring multiset status of each A-operator:

$$s >_{RPOS} t \quad \leadsto \quad \bar{s} >_{RPOS} \bar{t}$$

Proof: We will prove this statement by induction on $|s| + |t|$. Let be $s = f(s_1,...,s_m)$ and $t = g(t_1,...,t_n)$. We have to consider the following five cases:

i) $f \triangleright g \wedge g \notin \mathfrak{F}_A$

$\leadsto$ $\bar{s} = f(\overline{s_1},...,\overline{s_m})$, $\bar{t} = g(\overline{t_1},...,\overline{t_n})$
since $f, g \notin \mathfrak{F}_A$

$\leadsto$ $\{\bar{s}\} \gg_{RPOS} \{\overline{t_1},...,\overline{t_n}\}$
since $\{s\} \gg_{RPOS} \{t_1,...,t_n\}$ and by using the induction hypothesis

$\leadsto$ $\bar{s} >_{RPOS} \bar{t}$
by definition of the RPOS

ii) $f \triangleright g \wedge g \in \mathfrak{F}_A$

$\leadsto$ $\bar{s} = f(\overline{s_1},...,\overline{s_m})$, $\bar{t} = g(\overline{t'_1},...,\overline{t'_p})$
such that $\overline{t_i} = g(\overline{t'_{i_1}},...,\overline{t'_{i_\alpha}})$ or $\overline{t_i} = \overline{t'_j}$

$\leadsto$ $\{\bar{s}\} \gg_{RPOS} \{\overline{t'_1},...,\overline{t'_p}\}$
since $[\forall i \in [1,n]]\ s >_{RPOS} t_i$ $[t_i >_{RPOS} t'_{i_1},...,t'_{i_\alpha}]$ and by induction hypothesis

iii) $f = g \wedge f \notin \mathfrak{F}_A$

$\leadsto$ $\bar{s} = f(\overline{s_1},...,\overline{s_m})$, $\bar{t} = f(\overline{t_1},...,\overline{t_n})$
since $f \notin \mathfrak{F}_A$

$\leadsto$ $args(\bar{s}) >_{RPOS,\tau(f)} args(\bar{t})$ $[\wedge \{\bar{s}\} \gg_{RPOS} args(\bar{t})]$
since $args(s) >_{RPOS,\tau(f)} args(t)$ $[\wedge \{s\} \gg_{RPOS} args(t)]$ and by induction hypthesis

$\leadsto$ $\bar{s} >_{RPOS} \bar{t}$
by definition of the RPOS

iv] $f = g \quad \wedge \quad f \in \mathfrak{F}_A$

$\rightsquigarrow$ $\tau[f] = mult$
since $f \in \mathfrak{F}_A$

Let be $s_i >_{RPOS} t_j$.

- $top(s_i) \neq f \quad \wedge \quad top(t_j) \neq f$

  $\rightsquigarrow$ $\overline{s_i} >_{RPOS} \overline{t_j}$
  by induction hypothesis

- $top(s_i) \neq f \quad \wedge \quad \overline{t_j} = f(\overline{t'_1},...,\overline{t'_p})$

  $\rightsquigarrow$ $\{\overline{s_i}\} \gg_{RPOS} \{\overline{t'_1},...,\overline{t'_p}\}$
  because $\overline{s_i} >_{RPOS} \overline{t_j}$ [by induction hypothesis] and $\{\overline{t_j}\} \gg_{RPOS} \{\overline{t'_1},...,\overline{t'_p}\}$ [by the subterm property of the RPOS]

- $\overline{s_i} = f(\overline{s'_1},...,\overline{s'_p}) \quad \wedge \quad top(t_j) \neq f$

  $\rightsquigarrow$ $\overline{s_i} >_{RPOS} \overline{t_j}$
  by induction hypothesis

  $\rightsquigarrow$ $\{\overline{s'_1},...,\overline{s'_p}\} \gg_{RPOS} \{\overline{t_j}\}$
  since $\neg[f \trianglerighteq top(t_j)]$ and by definition of the RPOS

- $\overline{s_i} = f(\overline{s'_1},...,\overline{s'_p}) \quad \wedge \quad \overline{t_j} = f(\overline{t'_1},...,\overline{t'_q})$

  $\rightsquigarrow$ $\overline{s_i} >_{RPOS} \overline{t_j}$
  by induction hypothesis

  $\rightsquigarrow$ $\{\overline{s'_1},...,\overline{s'_p}\} \gg_{RPOS} \{\overline{t'_1},...,\overline{t'_q}\}$
  by definition of the RPOS

v] $\neg[f \trianglerighteq g]$

$\rightsquigarrow$ $[\exists i \in [1,m]] \ \overline{s_i} \geq_{RPOS} \overline{t}$
since $\{s_1,...,s_m\} \geq_{RPOS} \{t\}$ [by definition of the RPOS] and by induction hypothesis

- $top(s_i) \notin \mathfrak{F}_A$

  $\rightsquigarrow$ $\overline{s} >_{RPOS} \overline{t}$
  since $\overline{s} = f(..., \overline{s_i}, ...)$

- $top[s_i] \in \mathfrak{F}_A \quad \wedge \quad top[s_i] \neq f$:

  analogous with the previous case

- $\overline{s_i} = f[\overline{s'_1},...,\overline{s'_p}] \quad \wedge \quad f \in \mathfrak{F}_A$

  $\rightsquigarrow$ $\{\overline{s'_1},...,\overline{s'_p}\} \geq_{RPOS} \{\overline{t}\}$
  since $[\exists s'_j]\ s'_j >_{RPOS} t$ [by definition of the RPOS] and by induction hypothesis

  $\rightsquigarrow$ $\overline{s} >_{RPOS} \overline{t}$
  by definition of the RPOS

■

Lemma 4.3.4 $>_{AD}$ is a simplification ordering which is A-compatible and stable w.r.t substitutions.

Proof: i) $>_{AD}$ is a partial ordering on $\Gamma[\mathfrak{F}]$ since the RPOS and $\overset{+}{\Longrightarrow}_{D/A}$ are partial orderings.

ii) $>_{AD}$ is A-compatible:
We have to prove that $s' =_A s >_{AD} t =_A t'$ implies $s' >_{AD} t'$.

Note that

$s =_A t$

$\rightsquigarrow$ $\delta[s] =_A \delta[t]$
[see [BP85]]

$\rightsquigarrow$ $\overline{\delta[s]} = \overline{\delta[t]}$
with the help of lemma 4.2.3

$\rightsquigarrow$ $\overline{\delta[s']} = \overline{\delta[s]} >_{RPOS} \overline{\delta[t]} = \overline{\delta[t']}$
or $\overline{\delta[s]} = \overline{\delta[t]} \quad \wedge \quad \overline{\delta[s']} = \overline{\delta[s]} \overset{+}{\Longrightarrow}_{D/A} \overline{\delta[t]} = \overline{\delta[t']}$
by using the precondition and the definition of $>_{AD}$

$\rightsquigarrow$ $\overline{\delta[s']} >_{RPOS} \overline{\delta[t']}$
or $\overline{\delta[s']} = \overline{\delta[t']} \quad \wedge \quad \overline{\delta[s']} \overset{+}{\Longrightarrow}_{D/A} \overline{\delta[t']}$
since the RPOS is compatible with =

$\rightsquigarrow$ $s' >_{AD} t'$
by definition of $>_{AD}$

iii] $>_{AD}$ has the replacement property:
analogous with the proof of the same fact w.r.t. the RPO without status [see [GL86]] since any associative operator has multiset status

iv] $>_{AD}$ has the subterm property:

Note that
- $>_{AD}$ is stronger than $>_A$:
  $s >_A t \rightsquigarrow s >_{AD} t$
  This fact can easily be proved by an analysis of the definitions of $>_A$ and $>_{AD}$ [see lemma 4.6.1].
- $>_A$ has the subterm property:
  see lemma 4.2.5

Therefore, $f[..., t, ...] >_A t$ implies $f[..., t, ...] >_{AD} t$

v] $>_{AD}$ has the deletion property:
$f[..., t, ...] >_{AD} f[..., ...]$ if f is a varyadic operator: This is valid since the RPO without status has the deletion property and each varyadic [associative] operator has multiset status.

vi] $>_{AD}$ is stable w.r.t. substitutions:
analogous with the proof contained in [GL86]

■

Lemma 4.4.4 $>_{AE}$ is a simplification ordering which is A-compatible and stable w.r.t. substitutions.

Proof: i] $>_{AE}$ is a partial ordering on $\Gamma[\mathfrak{F}]$ since the RPOS and $\overset{+}{\Longrightarrow}_{E/A}$ are partial orderings.

ii] $>_{AE}$ is A-compatible:
We have to prove that $s' =_A s >_{AE} t =_A t'$ implies $s' >_{AE} t'$.

Note that

$s =_A t$
$\rightsquigarrow \overline{\varepsilon[s]} = \overline{\varepsilon[t]}$
[see [Ze89]]

$\rightsquigarrow \overline{\varepsilon[s']} = \overline{\varepsilon[s]} >_{RPOS} \overline{\varepsilon[t]} = \overline{\varepsilon[t']}$
or $\overline{\varepsilon[s]} = \overline{\varepsilon[t]} \wedge \overline{\varepsilon[s']} = \overline{\varepsilon[s]} \overset{+}{\Longrightarrow}_{E/A} \overline{\varepsilon[t]} = \overline{\varepsilon[t']}$
by using the precondition and the definition of $>_{AE}$

$\rightsquigarrow$ $\overline{\varepsilon[s']} >_{RPOS} \overline{\varepsilon[t']}$
or $\overline{\varepsilon[s']} = \overline{\varepsilon[t']} \wedge \overline{\varepsilon[s']} \overset{+}{\Longrightarrow}_{E/A} \overline{\varepsilon[t']}$
since the RPOS is compatible with =

$\rightsquigarrow$ $s' >_{AE} t'$
by definition of $>_{AE}$

iii] $>_{AE}$ has the replacement property:
analogous with the proof of the same fact w.r.t. the RPO without status [see [Gn88]] since any associative function symbol has multiset status

iv] $>_{AE}$ has the subterm property:

Note that
- $>_{AE}$ is stronger than $>_A$:
  $s >_A t \rightsquigarrow s >_{AE} t$
  This fact can easily be proved by an analysis of the definitions of $>_A$ and $>_{AE}$ [see lemma 4.6.1].
- $>_A$ has the subterm property:
  see lemma 4.2.5

Therefore, $f[..., t, ...] >_A t$ implies $f[..., t, ...] >_{AE} t$

v] $>_{AE}$ has the deletion property:
$f[..., t, ...] >_{AE} f[..., ...]$ if f is a varyadic function symbol: This is valid since the RPO without status has the deletion property and each varyadic [associative] operator has multiset status.

vi] $>_{AE}$ is stable w.r.t. substitutions:
analogous with the proof contained in [Gn88]

■

Lemma 4.5.2 $>_{AC}$, $>_{ACD}$ and $>_{ACE}$ are simplification orderings which are AC-compatible and stable w.r.t. substitutions.

Proof: The references to the proofs of these assertions based on the RPO without status are given in [GL86]. Together with the lemmata 4.2.5, 4.3.4 and 4.4.4 the corresponding properties about the RPOS are valid.

■

Lemma 4.5.3 · $>_{ACD^*}$ is ACD-compatible,
· $>_{ACE^*}$ is ACE-compatible.

Proof: i) $s' =_{ACD} s >_{ACD^*} t =_{ACD} t' \quad \rightsquigarrow \quad s' >_{ACD^*} t'$:

The two implications

$$s' =_{ACD} s >_{ACD^*} t \quad \rightsquigarrow \quad s' >_{ACD^*} t$$
$$\text{and} \quad s >_{ACD^*} t =_{ACD} t' \quad \rightsquigarrow \quad s >_{ACD^*} t'$$

suffice to prove the assertion. Since the second one is symmetrical to the first one (the proofs are very similar) we will only show the latter.

$s >_{ACD^*} t$

$\rightsquigarrow$ $\overline{\delta[s]} >_{RPOS} \overline{\delta[t]}$
by definition of $>_{ACD^*}$

$\rightsquigarrow$ $\overline{\delta[s']} >_{RPOS} \overline{\delta[t]}$
since $\overline{\delta[s']} =_C \overline{\delta[s]}$ (because $s =_D t \rightsquigarrow \delta[s] =_{AC} \delta[t]$ [GL86]) and $s =_A t$ iff $\bar{s} = \bar{t}$ (lemma 4.2.3)] and the RPOS is C-compatible

$\rightsquigarrow$ $s' >_{ACD^*} t$
by definition of $>_{ACD^*}$

ii) $s' =_{ACE} s >_{ACE^*} t =_{ACE} t' \quad \rightsquigarrow \quad s' >_{ACE^*} t'$:

The two implications

$$s' =_{ACE} s >_{ACE^*} t \quad \rightsquigarrow \quad s' >_{ACE^*} t$$
$$\text{and} \quad s >_{ACE^*} t =_{ACE} t' \quad \rightsquigarrow \quad s >_{ACE^*} t'$$

are sufficient for proving the assertion. Since the second one is symmetrical to the first one (the proofs are very similar) we will only show the first one.

$s >_{ACE^*} t$

$\rightsquigarrow$ $\overline{\varepsilon[s]} >_{RPOS} \overline{\varepsilon[t]}$
by definition of $>_{ACE^*}$

$\rightsquigarrow$ $\overline{\varepsilon[s']} >_{RPOS} \overline{\varepsilon[t]}$
since $\overline{\varepsilon[s']} =_C \overline{\varepsilon[s]}$ (because $s =_{ACE} t \rightsquigarrow \overline{\varepsilon[s]} =_C \overline{\varepsilon[t]}$, see [Gn88]) and the RPOS is C-compatible

$\rightsquigarrow$ $s' >_{ACE^*} t$
by definition of $>_{ACE^*}$

■

Lemma 4.5.7 $>_A, >_{AD}, >_{AE}, >_{AC}, >_{ACD}$ and $>_{ACE}$ are not stable w.r.t. substitutions if the precedence on the A(C)-operators is a quasi-ordering.

Proof: The proof of this lemma is performed by giving a counter-example: see example 4.5.6 on page 32.
Moreover, it is easy (by studying the corresponding proofs) to show that the restriction "no quasi-ordering on AC-operators" is sufficient for guaranteeing the stability w.r.t. substitutions. Note that the other (non AC-) function symbols can be quasi-ordered.

■

Lemma 4.6.1 Let $\triangleright$ be a partial (total) precedence. Then,

i) $>_C \subset >_{AC}$
ii) $>_A \subset >_{AC}$, $>_A \subset >_{AD}$, $>_A \subset >_{AE}$
iii) $>_{AC} \subset >_{ACD}$, $>_{AD} \subset >_{ACD}$
iv) $>_{AC} \subset >_{ACE}$, $>_{AE} \subset >_{ACE}$

Proof: i) It is obvious that $>_{AC} = >_C$ if there are no associative function symbols (flattening is redundant). The proper inclusion is guaranteed by example 6.1 (page 39).

ii) The definitions of $>_A$, $>_{AD}$ and $>_{AE}$ are equivalent if all associative operators are minimal w.r.t. the precedence. The proper inclusion is satisfied by the examples 4.3.5 (page 26) and 4.4.5 (page 29), respectively.

iii) Note that $>_{AC} = >_{ACD}$ if the associative-commutative operators are minimal w.r.t. $\triangleright$ (since distributing can be neglected). Furthermore, $>_{AD} = >_{ACD}$ if there are no commutative operators. The proper inclusion is guaranteed by example 4.4.5 (page 29).

iv) analogous with iii) by substituting E for D (and with the help of example 6.10 on page 42).

■

Theorem 5.2 Let be $>_\rhd \in \{>_{RDOS}, >_{PSDS}, >_{IRDS}\}$.

- $>_C[>_\rhd]$ are simplification orderings, C-compatible and stable w.r.t. substitutions.
- $>_A[>_\rhd]$, $>_{AD}[>_\rhd]$ and $>_{AE}[>_\rhd]$ are simplification orderings, A-compatible and stable w.r.t. substitutions.
- $>_{AC}[>_\rhd]$, $>_{ACD}[>_\rhd]$ and $>_{ACE}[>_\rhd]$ are simplification orderings, AC-compatible and stable w.r.t. substitutions.

Proof: The complexity of the assertions requires the theorem [and the proof] to be partitioned into the lemmata 5.2.1 - 5.2.5.

■

Lemma 5.2.1 Let be $>_\rhd \in \{>_{RDOS}, >_{PSDS}, >_{IRDS}\}$.
$>_C[>_\rhd]$ are simplification orderings, C-compatible and stable w.r.t. substitutions.

Proof: i) $>_C[>_\rhd]$ are partial orderings since the RDOS, the PSDS and the IRDS are partial orderings [St88a] and because of the definition of $>_C$.

ii) $>_C[>_\rhd]$ are C-compatible:

Note that the following fact is valid:

Let be $s =_C t$, $dec[\{s\}] = \{dec_{u_1}[s], \ldots, dec_{u_m}[s]\}$ and $dec[\{t\}] = \{dec_{v_1}[t], \ldots, dec_{v_n}[t]\}$.

$\rightsquigarrow$ $m = n$
since $s =_C t$ [especially, the sets of leaves of both terms are identical]

$\rightsquigarrow$ $[\exists\pi]\ [\forall i \in [1,n]]\ dec_{u_i}[s] = =_C dec_{v_{\pi[i]}}[t]$ (*)
with $\{s_1, \ldots, s_m\} = =_C \{t_1, \ldots, t_{u_n}\}$ iff $[\exists\pi']\ [\forall i \in [1,m]]$ $s_i =_C t_{\pi'[i]}$

This is obvious [by constructing π such that s/u_i and $t/v_{\pi[i]}$ are the same symbols [leaves].

$\rightsquigarrow$ $[\forall j]\ [\forall s_i \in dec_{u_j}[s]]\ [\exists k]\ [\exists t_l \in dec_{v_k}[t]]\ s_i =_C t_l$

We have to prove that

$$s' =_C s >_C[\rhd] t =_C t' \rightsquigarrow s' >_C[\rhd] t'$$

Let be $dec[\{s'\}] = \{S'_1, \ldots, S'_m\}$, $dec[\{s\}] = \{S_1, \ldots, S_m\}$, $dec[\{t\}] = \{T_1, \ldots, T_n\}$ and $dec[\{t'\}] = \{T'_1, \ldots, T'_n\}$.
With (*) the following holds (w.l.o.g. let be π the identity):
$[\forall i \in [1,m]]\ S'_i = =_C S_i$ and $[\forall i \in [1,n]]\ T'_i = =_C T_i$

$\rightsquigarrow$ $S'_i = =_* S_i$ and $T'_i = =_* T_i$
with $* \in \{LD, LP, EL\}$
because commutative operators have multiset status and by definition of $*$

$\rightsquigarrow$ $dec[\{s'\}] == ==_\Delta dec[\{s\}] \wedge dec[\{t\}] == ==_\Delta dec[\{t'\}]$
with $\Delta \in \{RDOS, PSDS, IRDS\}$
by definition of $dec[\{s'\}]$, $dec[\{s\}]$, $dec[\{t\}]$ and $dec[\{t'\}]$, respectively

$\rightsquigarrow$ $dec[\{s'\}] \gg \gg_\Delta dec[\{t'\}]$
since $dec[\{s\}] \gg \gg_\Delta dec[\{t\}]$ by precondition

$\rightsquigarrow$ $s' \rhd t'$
by definition of $\rhd$

$\rightsquigarrow$ $s' >_C[\rhd] t'$
since every commutative operator has multiset status

iii) $>_C[\rhd]$ have the replacement property:
This is valid since the RDOS, the PSDS and the IRDS have this characteristic [see [St88a]]

iv) $>_C[\rhd]$ have the subterm property:
analogous with iii)

v) $>_C[\rhd]$ have the deletion property:

Note that $\rhd$ have this property:

$f(\ldots, t, \ldots) >_{RPOS} f(\ldots, \ldots)$ if f is a varyadic operator
[see lemma 4.2.5 v)]

$\rightsquigarrow$ $f(\ldots, t, \ldots) \rhd f(\ldots, \ldots)$ if f is a varyadic operator
since $>_{RPOS} \subseteq \rhd$ [see lemma 3.3.2 and [St88a]]

This fact directly implies the assertion.

vi) $>_C[\rhd]$ are stable w.r.t. substitutions:
analogous with iii)

■

Lemma 5.2.2 Let be $\rhd \in \{>_{RDOS}, >_{PSDS}, >_{IRDS}\}$.
$>_A[\rhd]$ are simplification orderings, A-compatible and stable w.r.t. substitutions.

Proof: i) $>_A[\rhd]$ are partial orderings on $\Gamma[\mathfrak{F}]$ since $\rhd$ are partial orderings on $\Gamma^*[\mathfrak{F}]$ (see proof of lemma 5.2.1)

ii) $>_A[\rhd]$ are A-compatible:
$s' =_A s >_A[\rhd] t =_A t' \rightsquigarrow s' >_A[\rhd] t'$
since $r =_A r' \rightsquigarrow \bar{r} = \bar{r}'$ (lemma 4.2.3)

iii) $>_A[\rhd]$ have the replacement property:
We must prove that $s >_A[\rhd] t \rightsquigarrow f(\ldots,s,\ldots) >_A[\rhd] f(\ldots,t,\ldots)$.
This is equivalent to

$$\bar{s} \rhd \bar{t} \rightsquigarrow \overline{f(\ldots,s,\ldots)} \rhd \overline{f(\ldots,t,\ldots)}$$

Let be $dec(\{\bar{r}\}) = \{R_1,\ldots,R_m\}$.

$$\rightsquigarrow \quad dec(\{\overline{f(\ldots,r,\ldots)}\}) = \begin{cases} \bigcup V_i \cup \bigcup_{i=1}^{m} \{\overline{f(\ldots,r,\ldots)}\} \cup R_i \\ \text{if } top(r) \neq f \vee f \notin \mathfrak{F}_A \\ \bigcup V_i \cup \bigcup_{i=1}^{m} \{\overline{f(\ldots,r,\ldots)}\} \cup (R_i \backslash \{\bar{r}\}) \quad \text{otherwise} \end{cases}$$

by definition of dec

Note that $(\forall j)\,(\exists i)\; S_i \gg_* T_j$ for all $* \in \{LD, LP, EL\}$ (S_i and T_j are constructed from s and t like R_i from r). This is valid since $\bar{s} \rhd \bar{t}$.
We have to consider four cases:

- $top(\bar{s}) \neq f$, $top(\bar{t}) \neq f$:
$dec(\{\overline{f(\ldots,s,\ldots)}\}) \gg \gg_* dec(\{\overline{f(\ldots,t,\ldots)}\})$
by the above construction and since $\overline{f(\ldots,s,\ldots)} >_* \overline{f(\ldots,t,\ldots)}$
(because it is reduced to the comparison of the direct arguments)

- $top(s) \neq f$, $top(t) = f \in \mathfrak{F}_A$:
We have to show that

$$S_i \cup \{\overline{f(\ldots,s,\ldots)}\} \gg_* \{\overline{f(\ldots,t,\ldots)}\} \cup (T_j \backslash \{\bar{t}\})$$

This is valid since $T_j \backslash \{\bar{t}\} \subset T_j$ and $\overline{f(\ldots,s,\ldots)} >_* \overline{f(\ldots,t,\ldots)}$
(cf. previous case)

- $top[s] = f \in \mathfrak{F}_A$, $top[t] \neq f$:
  We have to prove that

$$\{\overline{f[\ldots,s,\ldots]}\} \cup [S_i \backslash \{\bar{s}\}] \gg_* \{\overline{f[\ldots,t,\ldots]}\} \cup T_j$$

  This is true because $\overline{f[\ldots,s,\ldots]} \gg_* \bar{s}$ (since $top[s] = f$), i.e. $\bar{s}$ can be replaced by $\overline{f[\ldots,s,\ldots]}$

- $top[s] = top[t] = f$:
  We must show that

$$\{\overline{f[\ldots,s,\ldots]}\} \cup [S_i \backslash \{\bar{s}\}] \gg_* \{\overline{f[\ldots,t,\ldots]}\} \cup [T_j \backslash \{\bar{t}\}]$$

  This can be proved with the help of the previous case.

iv] $>_A[\triangleright]$ have the subterm property:
$f[\ldots,t,\ldots] >_A t$ (see lemma 4.2.5)

$\rightsquigarrow$ $f[\ldots,t,\ldots] >_A[\triangleright] t$
since $>_A \subset >_A[\triangleright]$ (see lemma 3.3.2)

v] $>_A[\triangleright]$ have the deletion property:
$f[\ldots,t,\ldots] >_A f[\ldots,\ldots]$ if f is a varyadic function symbol (see lemma 4.2.5)

$\rightsquigarrow$ $f[\ldots,t,\ldots] >_A[\triangleright] f[\ldots,\ldots]$ if f is a varyadic operator
since $>_A \subset >_A[\triangleright]$ (see lemma 3.3.2)

vi] $>_A[\triangleright]$ are stable w.r.t. substitutions:

We must show that

$$s >_A[\triangleright] t \rightsquigarrow \sigma(s) >_A[\triangleright] \sigma(t) \text{ , for all } \sigma$$

This is equivalent to (by definition of $>_A$)

$$\bar{s} \rhd \bar{t} \rightsquigarrow \overline{\sigma(s)} \rhd \overline{\sigma(t)} \text{ , for all } \sigma.$$

Note that we have to compare path-decompositions of $\overline{\sigma(t)}$ with path-decompositions of $\overline{\sigma(s)}$.
Let be $dec_u(\bar{s}) = \{s'_1,\ldots,s'_m\}$ and $dec_v(\bar{t}) = \{t'_1,\ldots,t'_n\}$ such that $dec_u(\bar{s}) \gg_* dec_v(\bar{t})$ with $* \in \{LD, LP, EL\}$. Furthermore, let s'_i (t'_i) be an argument of s'_{i-1} (t'_{i-1}).
If s'_m and t'_n are constants then the assertion directly follows from the fact that $dec_u(\bar{s}) \gg_* dec_v(\bar{t})$. Moreover, it is easy to prove the assertion if $s'_m \in \mathfrak{V}$ and t'_n is a constant (since s'_m is not needed).

Therefore, the only crucial case is that of $s'_m = t'_n = x \in \mathfrak{V}$ and $s'_{m-1} = f[\ldots, x, \ldots] >_* g[\ldots, x, \ldots] = t'_{n-1}$ (the other cases - $s'_{m-i} >_* t'_{n-1}$ - can be reduced to this one). For reasons of simplicity, let be $s' = s'_{m-1}$ and $t' = t'_{n-1}$. We have to consider the following two cases:

α) $f = g$

$\rightsquigarrow \overline{\sigma[s]} \rhd \overline{\sigma[t]}$
since the variable x is replaced in s' and t' by the same structure [$\rightsquigarrow$ we have identical subpaths]

β) $f \rhd g$

$\rightsquigarrow f \notin \mathfrak{F}_A$
because associative operators are minimal w.r.t. $\rhd$

- $g \notin \mathfrak{F}_A$

  $\rightsquigarrow \overline{\sigma[s]} \rhd \overline{\sigma[t]}$
  with the help of the considerations of α)

- $g \in \mathfrak{F}_A$

  $\rightsquigarrow \overline{\sigma[s]} \rhd \overline{\sigma[t]}$
  since the paths of $\sigma[t']$ w.r.t. the substituted variable x are proper parts of the paths of $\overline{\sigma[s']}$ w.r.t. x [because s' cannot be flattened w.r.t. $\sigma[x]$]

■

Lemma 5.2.3 $>_{AD}[\rhd]$ are simplification orderings, A-compatible and stable w.r.t. substitutions.

Proof: i) $>_{AD}[\rhd]$ are partial orderings on $\Gamma[\mathfrak{F}]$ since $\rhd$ and $\overset{+}{\Longrightarrow}_{D/A}$ are partial orderings.

ii) $>_{AD}[\rhd]$ are A-compatible:

We have to prove that $s' =_A s >_{AD}[\rhd] t =_A t'$ implies $s' >_{AD}[\rhd] t'$.
Note that $s =_A t \rightsquigarrow \overline{\delta[s]} = \overline{\delta[t]}$ [see proof of lemma 4.3.4]

$\rightsquigarrow \overline{\delta[s']} = \overline{\delta[s]} \rhd \overline{\delta[t]} = \overline{\delta[t']}$
or $\overline{\delta[s]} = \overline{\delta[t]} \wedge \overline{\delta[s']} = \overline{\delta[s]} \overset{+}{\Longrightarrow}_{D/A} \overline{\delta[t]} = \overline{\delta[t']}$
by using the precondition and the definition of $>_{AD}[\rhd]$

$\rightsquigarrow$ $\overline{\delta[s']} \rhd \overline{\delta[t']}$
or $\overline{\delta[s']} = \overline{\delta[t']} \wedge \overline{\delta[s']} \overset{+}{\Longrightarrow}_{D/A} \overline{\delta[t']}$
since $\rhd$ are compatible with $=$

$\rightsquigarrow$ $s' >_{AD}[\rhd] t'$
by definition of $>_{AD}[\rhd]$

iii) $>_{AD}[\rhd]$ have the replacement property:

We have to prove that $s >_{AD}[\rhd] t \rightsquigarrow f[\ldots, s, \ldots] >_{AD}[\rhd] f[\ldots, t, \ldots]$. It is obvious that the relation $\overset{+}{\Longrightarrow}_{D/A}$ has the replacement property. Therefore, from now on, we consider only the case where $s >_{AD}[\rhd] t$ implies that $\overline{\delta[s]} \rhd \overline{\delta[t]}$. In this case, the above requirement is equivalent to

$$\overline{\delta[s]} \rhd \overline{\delta[t]} \rightsquigarrow \overline{\delta[f[\ldots, s, \ldots]]} \rhd \overline{\delta[f[\ldots, t, \ldots]]}.$$

We must consider two cases:

α) $f \notin \mathfrak{F}_A$

$\rightsquigarrow$ $\overline{\delta[f[\ldots, s, \ldots]]} = f[\ldots, \overline{\delta[s]}, \ldots]$ and
$\overline{\delta[f[\ldots, t, \ldots]]} = f[\ldots, \overline{\delta[t]}, \ldots]$

$\rightsquigarrow$ $\overline{\delta[f[\ldots, s, \ldots]]} \rhd \overline{\delta[f[\ldots, t, \ldots]]}$
"iff" [depending on $\rhd$]
$\overline{\delta[s]} \rhd \overline{\delta[t]}$
by definition of $\rhd$

$\rightsquigarrow$ assertion
since $\overline{\delta[s]} \rhd \overline{\delta[t]}$ [precondition]

β) $f \in \mathfrak{F}_A$:

W.l.o.g. let be $s' = f[s, r]$ and $t' = f[t, r]$ [note that $\tau[f] = \text{mult}$]. Furthermore, $f \triangleright g$ and $g \in \mathfrak{F}_A$. The case of s' and t' starting with g can be proved similar to the corresponding proof of $>_A$ and by using induction on $|s'| + |t'|$.
Note that r is not relevant for the proof since it occurs in s' as well as in t'. Therefore, independent of the leading function symbols of r, s and t, it occurs at the same level in the decompositions of s' and t'. For reasons of simplicity, we assume that r is an "empty" term [$\rightsquigarrow$ an associative operator can also have only one argument !]. It is easy to prove the following facts about decompositions:

if $top[\delta[s]] = f$

then $dec[\{\overline{\delta[f[s]]}\}] = dec[\{\overline{\delta[s]}\}]$ (*)

if $top[\delta[s]] = g \quad [\overline{\delta[s]} = g[s_1, \ldots, s_m]]$

then $dec[\{\overline{\delta[f[s]]}\}] =$

$$\bigcup_{u \in Ot[\overline{\delta[s]}]} \{s' \cup \{g[f[s_1], \ldots, f[s_m]]\} \mid s' \in dec_u[\overline{\delta[s]}]\}$$

We have to consider four subcases:

- $top[\overline{\delta[s]}] = f \quad , \quad top[\overline{\delta[t]}] = f$

  $\rightsquigarrow \quad \overline{\delta[f[s]]} \; \rhd \; \overline{\delta[f[t]]}$
  since (*) is valid

- $top[\overline{\delta[s]}] = f \quad , \quad top[\overline{\delta[t]}] = g$

  $\rightsquigarrow \quad \overline{\delta[f[s]]} \; \rhd \; \overline{\delta[f[t]]}$
  because $f \rhd g$

- $top[\overline{\delta[s]}] = g \quad , \quad top[\overline{\delta[t]}] = f$

  $\rightsquigarrow \quad \overline{\delta[f[s]]} \; \rhd \; \overline{\delta[f[t]]}$
  since $\overline{\delta[f[t]]} \subset \overline{\delta[f[s]]}$

- $top[\overline{\delta[s]}] = g = top[\overline{\delta[t]}]$

  $\rightsquigarrow \quad \overline{\delta[f[s]]} \; \rhd \; \overline{\delta[f[t]]}$
  because in each path-decomposition of $\overline{\delta[f[s]]}$ there exists a term with f as the leading function symbol
  [$\rightsquigarrow$ this term is greater than $g[f[t_1], \ldots, f[t_n]]$]

iv] $>_{AD}[\rhd]$ have the subterm property:
Note that $>_{AD}[\rhd]$ are stronger than $>_A[\rhd]$ and $>_A[\rhd]$ have the subterm property [see iv] of 4.3.4 and 5.2.2]. Therefore, $f[\ldots, t, \ldots] >_A[\rhd] t$ implies $f[\ldots, t, \ldots] >_{AD}[\rhd] t$.

v] $>_{AD}[\rhd]$ have the deletion property:
$f[\ldots, t, \ldots] >_{AD}[\rhd] f[\ldots, \ldots]$ if f is a varyadic operator: This is valid since $\rhd$ without status possess the deletion property and each varyadic [associative] operator has multiset status. Now, we will show that $\rhd$ without status have the deletion property:

Let be $dec[\{f[\ldots, \ldots]\}] = \{S_1, \ldots, S_m\}$ with $S_i := \{f[\ldots, \ldots]\} \cup S_i'$

$\rightsquigarrow$ $dec(\{f(\ldots,t,\ldots)\}) = \{T_1,\ldots,T_m, T_{m+1},\ldots,T_n\}$
with $T_i = \{f(\ldots,t,\ldots)\} \cup S'_i$ if $i \in [1,m]$ and $T_i = \{f(\ldots,t,\ldots)\} \cup T'_i$ otherwise
by definition of dec

$\rightsquigarrow$ It suffices to verify that

$$T_i \gg_* S_i$$

for all $i \in [1,m]$ and for all $* \in \{LD, LP, EL\}$

$\rightsquigarrow$ We have to prove that $f(\ldots,t,\ldots) >_* f(\ldots,\ldots)$
because of the definition of T_i and S_i, and since the multiset extension is closed under difference [see [St86]]

α] $t' = f(\ldots,t,\ldots) >_{LD} f(\ldots,\ldots) = s'$
iff - $sub(dec_u(t'),t') \gg_{LD} sub(dec_u(s'),s')$
- $args(t') \gg_{RDOS} args(s')$

Note that $sub(dec_u(t'),t') = sub(dec_u(s'),s') = S'_i$ [i depends on u] and $args(t') \gg_{RDOS} args(s')$ since $args(s') \subset args(t')$

β] $t' = f(\ldots,t,\ldots) >_{LP} f(\ldots,\ldots) = s'$
iff $dec(args(t')) \gg \gg_{LP} dec(args(s'))$

This is valid since

$$dec(args(t')) = \{S'_1,\ldots,S'_m,\{t\} \cup T'_{m+1},\ldots,\{t\} \cup T'_n\} \supset \{S'_1,\ldots,S'_m\} = dec(args(s'))$$

γ] $t' = f(\ldots,t,\ldots) >_{EL} f(\ldots,\ldots) = s'$
since
- $sub(dec_u(t'),t') = sub(dec_u(s'),s')$ [see α]]
- $dec(args(t')) \gg \gg_{EL} dec(args(s'))$ [see β]]
- and by definition of $>_{EL}$

vi] $>_{AD}[\triangleright]$ are stable w.r.t. substitutions:
We have to show that

$$s >_{AD}[\triangleright] t \rightsquigarrow \sigma(s) >_{AD}[\triangleright] \sigma(t) \text{ , for all } \sigma$$

This is equivalent to [by definition of $>_{AD}$]

$$\overline{\delta[s]} \rhd \overline{\delta[t]} \quad \leadsto \quad \overline{\delta[\sigma[s]]} \rhd \overline{\delta[\sigma[t]]} \quad , \text{ for all } \sigma.$$

Note that we have to compare path-decompositions of $\overline{\delta[\sigma[t]]}$ with path-decompositions of $\overline{\delta[\sigma[s]]}$.
Let be $dec_u[\overline{\delta[s]}] = \{s'_1, \ldots, s'_m\}$ and $dec_v[\overline{\delta[t]}] = \{t'_1, \ldots, t'_n\}$ such that $dec_u[\overline{\delta[s]}] \gg_* dec_v[\overline{\delta[t]}]$ with $* \in \{LD, LP, EL\}$. Furthermore, let s'_i $[t'_i]$ be an argument of s'_{i-1} $[t'_{i-1}]$.
If s'_m and t'_n are constants then the assertion directly follows from the fact that $dec_u[\overline{\delta[s]}] \gg_* dec_v[\overline{\delta[t]}]$. Moreover, it is easy to prove the assertion if $s'_m \in \mathfrak{V}$ and t'_n is a constant [since s'_m is not needed].
Therefore, the only crucial case is that of $s'_m = t'_n = x \in \mathfrak{V}$ and $s'_{m-1} = f[\ldots, x, \ldots] >_* g[\ldots, x, \ldots] = t'_{n-1}$ [the other cases - $s'_{m-i} >_* t'_{n-1}$ - can be reduced to this one]. For reasons of simplicity, let be $s' = s'_{m-1}$ and $t' = t'_{n-1}$. We have to consider the following two cases:

α] $f = g$

$\leadsto$ the substituted arguments of s' and t' w.r.t. x are the same

$\leadsto$ the comparison depends on the leading function symbols of s'_{m-2} and t'_{n-2}

$\leadsto$ $\overline{\delta[\sigma[s]]} \rhd \overline{\delta[\sigma[t]]}$
because the path-decompositions from s' and t' do not change

β] $f \triangleright g$:

- $f, g \notin \mathfrak{F}_A$

  $\leadsto$ $\overline{\delta[\sigma[s]]} \rhd \overline{\delta[\sigma[t]]}$
  because no flattening and distributing of s' and t' is needed

- $f \notin \mathfrak{F}_A$, $g \in \mathfrak{F}_A$

  $\leadsto$ $\overline{\delta[\sigma[s]]} \rhd \overline{\delta[\sigma[t]]}$
  since $f \triangleright g$

- $f, g \in \mathfrak{F}_A$:

  - $\sigma[x] = h[\ldots]$ with $f \neq h \neq g$:
    analogous with the first case of β]

  - $\sigma[x] = f[\ldots]$

    $\rightsquigarrow$ $\overline{\delta[\sigma[s]]} \rhd \overline{\delta[\sigma[t]]}$
    since the leading function symbol of $\sigma[x]$ can be replaced by $top[\delta[\sigma[s']]] = f$ which has more arguments than $\sigma[x]$

  - $\sigma[x] = g[r_1, \ldots, r_p]$

    $\rightsquigarrow$ $f[\ldots, x, \ldots]$ will be transformed into $g[f[\ldots, r_1, \ldots], \ldots, f[\ldots, r_p, \ldots]]$

    $\rightsquigarrow$ $\overline{\delta[\sigma[s]]} \rhd \overline{\delta[\sigma[t]]}$
    because the substituted arguments of $\delta[\sigma[t']]$ are contained in $g[f[\ldots, r_1, \ldots], \ldots, f[\ldots, r_p, \ldots]]$

■

Lemma 5.2.4 $>_{AE}[\rhd]$ are simplification orderings, A-compatible and stable w.r.t. substitutions.

Proof: i) $>_{AE}[\rhd]$ are partial orderings:
analogous with i) of lemma 5.2.3

ii) $>_{AE}[\rhd]$ are A-compatible:
analogous with ii) of lemma 4.4.4

iii) $>_{AE}[\rhd]$ have the replacement property:

We have to show that $s >_{AE}[\rhd] t \rightsquigarrow f[\ldots, s, \ldots] >_{AE}[\rhd] f[\ldots, t, \ldots]$. It is obvious that the relation $\overset{+}{\Longrightarrow}_{E/A}$ has the replacement property. Therefore, from now on, we consider only the case where $s >_{AE}[\rhd] t$ implies that $\overline{\varepsilon[s]} \rhd \overline{\varepsilon[t]}$. Then, the above requirement is equivalent to

$$\overline{\varepsilon[s]} \rhd \overline{\varepsilon[t]} \rightsquigarrow \overline{\varepsilon[f[\ldots, s, \ldots]} \rhd \overline{\varepsilon[f[\ldots, t, \ldots]]}.$$

We must consider two cases:

α] $f \notin \mathfrak{F}_A$

$\rightsquigarrow$ $\overline{\varepsilon[f[\ldots,s,\ldots]]} = f[\ldots,\overline{\varepsilon[s]},\ldots]$ and
$\overline{\varepsilon[f[\ldots,t,\ldots]]} = f[\ldots,\overline{\varepsilon[t]},\ldots]$

$\rightsquigarrow$ $\overline{\varepsilon[f[\ldots,s,\ldots]]} \rhd \overline{\varepsilon[f[\ldots,t,\ldots]]}$
"iff" [depending on $\rhd$]
$\overline{\varepsilon[s]} \rhd \overline{\varepsilon[t]}$
by definition of $\rhd$

$\rightsquigarrow$ assertion
since $\overline{\varepsilon[s]} \rhd \overline{\varepsilon[t]}$ [precondition]

β] $f \in \mathfrak{F}_A$:

W.l.o.g. let be $s' = f[s,r]$ and $t' = f[t,r]$ [note that $\tau[f] = \text{mult}$]. Furthermore, $f \triangleright g$ and g is a unary operator. The case that s' and t' begin with g can be proved similar to the corresponding proof of $>_A$ and by using induction on $|s'| + |t'|$.
Note that r is not relevant for the proof [see 5.2.3 iii] β]]. Therefore, we assume that r is an "empty" term which induces that an associative operator can also have one argument, only. It is easy to prove the following facts about decompositions [related to ε]:

if $top[\varepsilon[s]] = f$
then $dec[\{\overline{\varepsilon[f[s]]}\}] = dec[\{\overline{\varepsilon[s]}\}]$ [*]
if $top[\varepsilon[s]] = g \quad [\overline{\delta[s]} = g[h[s_1,\ldots,s_m]]]$
then if $h = f$
then $dec[\{\overline{\varepsilon[f[s]]}\}] = dec[\{\overline{\varepsilon[s]}\}]$ [**]
else $dec[\{\overline{\varepsilon[f[s]]}\}] = \bigcup_{u \in Ot[\overline{\varepsilon[s]}]} \{s' \cup \{g[f[h[s_1,\ldots,s_m]]], f[h[s_1,\ldots,s_m]]\} \mid s' \in dec_u(\overline{\varepsilon[s]}) \setminus \{\overline{\varepsilon[s]}\}\}$

We have to consider four subcases:

- $top[\overline{\varepsilon[s]}] = f = top[\overline{\varepsilon[t]}]$

  $\rightsquigarrow$ $\overline{\varepsilon[f[s]]} \rhd \overline{\varepsilon[f[t]]}$
  since [*] is valid

- $top[\overline{\varepsilon[s]}] = f \quad , \quad top[\overline{\varepsilon[t]}] = g$

  - $\overline{\varepsilon[t]} = g[f[t_1, \ldots, t_n]]$

    $\rightsquigarrow \quad \overline{\varepsilon[f[s]]} \succ \overline{\varepsilon[f[t]]}$
    because of $(**)$

  - $\overline{\varepsilon[t]} = g[h[t_1, \ldots, t_n]]$ with $h \neq f$

    $\rightsquigarrow \quad \overline{\varepsilon[f[s]]} \succ \overline{\varepsilon[f[t]]}$
    since $f \triangleright g$ and there is a term in s which is greater than $h[t_1, \ldots, t_n]$

- $top[\overline{\varepsilon[s]}] = g \quad , \quad top[\overline{\varepsilon[t]}] = f$

  - $\overline{\varepsilon[s]} = g[f[s_1, \ldots, s_m]]$

    $\rightsquigarrow \quad \overline{\varepsilon[f[s]]} \succ \overline{\varepsilon[f[t]]}$
    because of $(**)$

  - $\overline{\varepsilon[s]} = g[h[s_1, \ldots, s_m]]$ with $h \neq f$

    $\rightsquigarrow \quad \overline{\varepsilon[f[s]]} \succ \overline{\varepsilon[f[t]]}$
    since $\overline{\varepsilon[s]}$ can be replaced by $g[f[h[s_1, \ldots, s_m]]]$ which is a superterm

- $top[\overline{\varepsilon[s]}] = g = top[\overline{\varepsilon[t]}]$

  - $\overline{\varepsilon[s]} = g[f[s_1, \ldots, s_m]] \wedge \overline{\varepsilon[t]} = g[f[t_1, \ldots, t_n]]$

    $\rightsquigarrow \quad \overline{\varepsilon[f[s]]} \succ \overline{\varepsilon[f[t]]}$
    because of $(**)$

  - $\overline{\varepsilon[s]} = g[f[s_1, \ldots, s_m]] \wedge \overline{\varepsilon[t]} = g[h[t_1, \ldots, t_n]]$ with $h \neq f$

    $\rightsquigarrow \quad \overline{\varepsilon[f[s]]} \succ \overline{\varepsilon[f[t]]}$
    since $f \triangleright g$ and there exists a term in s which is greater than $h[t_1, \ldots, t_n]$

  - $\overline{\varepsilon[s]} = g[h[s_1, \ldots, s_m]] \wedge \overline{\varepsilon[t]} = g[f[t_1, \ldots, t_n]]$

    $\rightsquigarrow \quad \overline{\varepsilon[f[s]]} \succ \overline{\varepsilon[f[t]]}$
    because $\overline{\varepsilon[s]}$ can be replaced by $g[f[h[s_1, \ldots, s_m]]]$ which is a superterm

- $\overline{\varepsilon[s]} = g[h[s_1, \ldots, s_m]] \wedge \overline{\varepsilon[t]} = g[h[t_1, \ldots, t_n]]$

  $\rightsquigarrow \overline{\varepsilon[f[s]]} > \overline{\varepsilon[f[t]]}$
  since $\cdot\ dec_u(\varepsilon[s]) \cup \{g[f[h[s_1, \ldots, s_m]]]\} \setminus \{\overline{\varepsilon[s]}\} \gg_* dec_v(\overline{\varepsilon[t]}) \setminus \{\overline{\varepsilon[t]}\}$

  $\cdot\ f[h[s_1, \ldots, s_m]] >_* g[f[h[t_1, \ldots, t_n]]]$

  $\cdot\ f[h[s_1, \ldots, s_m]] >_* f[h[t_1, \ldots, t_n]]$

  with $* \in \{LD, LP, EL\}$

iv] $>_{AE}[\rhd]$ have the subterm property:
analogous with iv] of lemma 5.2.3

v] $>_{AE}[\rhd]$ have the deletion property:
analogous with v] of lemma 5.2.3

vi] $>_{AE}[\rhd]$ are stable w.r.t. substitutions:
analogous with the proof of 5.2.3 vi]

■

Lemma 5.2.5 $>_{AC}[\rhd]$, $>_{ACD}[\rhd]$ and $>_{ACE}[\rhd]$ are simplification orderings, AC-compatible and stable w.r.t. substitutions.

Proof: The proof of these facts follows directly from [i.e., can be done analogous with] the proofs of the previous [5.2.1 - 5.2.4] lemmata.

■