A total AC-compatible ordering based on RPO*

To appear in Theoretical Computer Science, spring 1995 (no. 1910)

Albert Rubio and Robert Nieuwenhuis
Technical University of Catalonia
Pau Gargallo 5, 08028 Barcelona, Spain.
E-mail: {`rubio,roberto`}`@lsi.upc.es`

April 4, 1995

Abstract

We define a simplification ordering on terms which is AC-compatible and total on non-AC-equivalent ground terms, without any restrictions on the signature like the number of AC-symbols or free symbols.

Unlike previous work by Narendran and Rusinowitch [NR91], our AC-RPO ordering is not based on polynomial interpretations, but on a simple extension of the well-known RPO ordering (with a total (arbitrary) precedence on the function symbols). This solves an open question posed e.g. by Bachmair [Bac92].

A second difference is that this ordering is also defined on terms with variables, which makes it applicable in practice for complete theorem proving strategies with built-in AC-unification and for orienting non-ground rewrite systems.

The ordering is defined in a simple way by means of rewrite rules, and can be easily implemented, since its main component is RPO.

1 Introduction

Automated termination proofs are well-known to be crucial for using rewriting-like methods in theorem proving and programming. In such methods, sometimes particular treatments are preferable (e.g. for efficiency reasons) for equational theories including "permutative" axioms like commutativity (which cannot be oriented into rewrite rules by means of well-founded orderings). Especially interesting because of its many practical applications is the case when the equational theory is associativity and commutativity (AC) for some of the operators. Therefore a lot of work has been done on the development of suitable *AC-compatible* reduction or simplification orderings, like [DHJP83, BP85, GL86, BCL87, KSZ90, Bac92].

An essential additional property of the ordering that is needed in order to preserve the completeness of most rewrite-based theorem proving techniques for first-order clauses (modulo AC) is its totality on (AC-different) ground terms. Such a total ordering was finally found by Narendran and Rusinowitch (cf. [NR91], where it is applied to prove that any AC-ground theory admits a finite convergent rewrite system, see also [Mar91]). Here we also define a simplification ordering

---

*This paper is an extended and revised version of [RN93], where a different formulation of this ordering was given. Here we include a simpler more elegant formulation which moreover works uniformly for both ground terms and terms with variables. We would like to thank Pierre Lescanne for a very useful suggestion in this direction made at RTA-93. This work is partially supported by the Esprit basic research working group 6028, CCL.

which is AC-compatible and total on non-AC-equivalent ground terms, without any restrictions on the signature like the number of AC-symbols or free symbols. However, an important difference with [NR91] is that our ordering is not based on polynomial interpretations, but, in a simple way, on a total (arbitrary) precedence on the function symbols, like in LPO or RPO.

The fact that our ordering is also defined on terms with variables makes it applicable in practice for orienting non-ground rewrite systems. However, our main motivation for this work was that an ordering like this one is essential to obtain complete theorem proving strategies with built-in AC-unification, and in particular for the extension to the AC-case [NR94] (see also [Vig94]) of our completeness proofs for deduction with ordering constrained clauses [NR92b] and *basic superposition* strategies [NR92a, BGLS92], which have turned out quite useful in the AC case, as it was already pointed out in [KKR90].

Furthermore, we believe that a simple ordering like the one defined here can also be a first step towards a decision procedure for the satisfiability of AC-ordering constraints, like done in [Com90] and [JO91] for LPO and RPO.

Let us now first give some intuition about the definition of the ordering. Let $\succ_{rpo}$ be a recursive path ordering (with status) generated by a total precedence $\succ_{\mathcal{F}}$, in which AC-symbols, denoted by $\mathcal{F}_{AC}$, have multiset status and where all other symbols have lexicographical left-to-right status. The ordering $\succ_{rpo}$ is a simplification ordering on $\mathcal{T}(\mathcal{F},\mathcal{X})$, which is total on $\mathcal{T}(\mathcal{F})$ (considering $=$ as equality up to permutation of arguments of multiset symbols). Moreover, $\succ_{rpo}$ is compatible wrt. $=$, i.e. $s' = s \succ_{rpo} t = t'$ implies $s' \succ_{rpo} t'$, for all terms s, s', t and t' in $\mathcal{T}(\mathcal{F},\mathcal{X})$.

This ordering $\succ_{rpo}$ is total on the AC-congruence classes of $\mathcal{T}(\mathcal{F})$ (i.e. all AC-different terms are comparable). However, it is not *AC-compatible*: not all terms of one class compare in the same way with each other term. *Flattening* terms (removing nested AC-symbols) before comparing them under $\succ_{rpo}$ solves this problem, but then monotonicity is lost:

Example 1.1 If $f \succ g$ where $f \in \mathcal{F}_{AC}$, then $f(a,a) \succ_{rpo} g(a)$, but for the terms $f(a, f(a,a))$ and $f(a, g(a))$, after flattening we get $f(a,a,a) \not\succ_{rpo} f(a, g(a))$.

Here the problem is caused by the symbol g immediately below the AC-symbol f. Since $f \succ g$, we have $f(a,a) \succ_{rpo} g(a)$, but after adding the context $f(a,[])$ and flattening, the innermost symbol f disappears and the g remains, reversing the orientation. These situations are avoided by the ordering defined here, which, roughly speaking, compares terms only after removing such symbols g:

Flattening of AC-terms consists of rewriting them to normal form wrt. the set R_F of *flattening* rules of the form:

$$f(\ldots, f(x_1,\ldots,x_n),\ldots) \rightarrow f(\ldots,x_1,\ldots,x_n,\ldots)$$

for all AC-function symbols f. We also define a set of *interpretation* rules R_I of the form

$$f(\ldots, g(x_1,\ldots,x_n),\ldots) \rightarrow f(\ldots,x_i,\ldots)$$

for all AC-symbols f with $f \succ g$ and all $i \in 1\ldots n$.

Now let $snf_R(s)$ and $snf_R(t)$ denote the sets of normal forms (by rewriting with these rules R) of s and t respectively. Then, roughly, in the AC-RPO ordering $\succ$ we define $s \succ t$ if for all terms in $snf_R(t)$ there is a bigger one (wrt. $\succ_{rpo}$) in $snf_R(s)$.

On the other hand, if for all terms in $snf_R(t)$ there is only a bigger or equal one in $snf_R(s)$ (note that then s and t have the same head symbol f) then a comparison is required between the top level subterms of s and t (after rewriting s and t with R_F only at the topmost position).

This comparison is done with the multiset extension of $\succ$ if f is an AC symbol and with the lexicographic extension otherwise (see section 3 for the formal definition).

Example 1.2 Suppose $a \succ f \succ g$ where $f \in \mathcal{F}_{AC}$. Then for the example above monotonicity is preserved: we have $f(a,a) \succ g(a)$ because $snf_R(f(a,a)) = \{f(a,a)\}$ and $snf_R(g(a)) = \{g(a)\}$ and $f(a,a) \succ_{rpo} g(a)$. Also, $f(a,f(a,a)) \succ f(a,g(a))$, since $snf_R(f(a,f(a,a))) = \{f(a,a,a)\}$ and $snf_R(f(a,g(a))) = \{f(a,a)\}$ (note that $f \succ g$ and hence g must be removed) and $f(a,a,a) \succ_{rpo} f(a,a)$.

It is worth emphasizing that only for simplicity reasons the ordering is defined in this paper in terms of the *set of normal forms*. In section 6 we will provide an equivalent more efficient definition, which is based on the fact that for any ground term s the unique maximal (wrt. $\succ_{rpo}$) term in $snf_R(s)$ can be obtained by rewriting innermost with R and keeping the maximal (wrt. $\succ_{rpo}$) candidate when n rules of the form $f(\ldots, g(x_1,\ldots,x_n),\ldots) \rightarrow f(\ldots,x_i,\ldots)$ can be applied.

To obtain a family of different orderings we introduce (in the formal definition given in the next section) the possibility of having rules in R_I not only for AC-symbols but also for some non-AC-symbols. This is done by means of splitting $\mathcal{F}$ in two subsets: the set of interpreted symbols (which must include $\mathcal{F}_{AC}$) and the set of non-interpreted symbols.

Example 1.3 Let $\mathcal{F}$ be $\{f,g,a\}$, where f is an AC-symbol and g is a non-AC-symbol. If only the AC-symbols are interpreted then $g(f(a,a)) \succ f(g(a),a)$ for any precedence between f, g and a. But if we allow interpreting g then, with any precedence satisfying $g \succ_{\mathcal{F}} f$, we have $f(g(a),a) \succ g(f(a,a))$, since the set of normal forms (wrt. the new R) of $f(g(a),a)$ and $g(f(a,a))$ are respectively $\{f(g(a),a)\}$ and $\{g(a)\}$.

We also want to make the observation that our ordering can be redefined to allow also part of the non-AC symbols to have a multiset status in the underlying RPO ordering. Here we do not give any specific proofs for this fact, as it is straightforward to extend all the existing proofs to this case. This allows us in some cases to obtain different orientations, although of course totality on the AC-congruence classes of $\mathcal{T}(\mathcal{F})$ is then reduced to totality up to permutation of arguments of such multiset operators.

This paper is structured in the following way. Section 2 provides the necessary technical background. In sections 3 and 4 we give the main definitions and some basic properties of the rewrite system R. In section 5 we prove the main properties of the ordering for the ground case and section 6 is on the implementation of the ordering. Section 7 proves the applicability of the ordering to terms with variables, and the last three sections are devoted to examples, extensions and comparisons with other related work and future work.

2 Definitions and basic properties

In the following we consider that $\mathcal{F}$ is a finite set of function symbols that is totally ordered by a precedence $\succ_{\mathcal{F}}$, where $\mathcal{F}_{AC}$ is the subset containing all AC-symbols of $\mathcal{F}$. We assume that $\bot$ is the smallest constant symbol of $\mathcal{F}$.

The arity of a function symbol f, denoted by $\alpha(f)$, is a set of natural numbers that indicates the number of arguments that f may take. If $f \in \mathcal{F}_{AC}$ then $\alpha(f)$ contains all natural numbers greater than 1; otherwise, $\alpha(f)$ contains one natural number. $\mathcal{T}(\mathcal{F})$ and $\mathcal{T}(\mathcal{F},\mathcal{X})$ are defined as usual according to these arities, if $\mathcal{X}$ is a set of variables, whose elements will be denoted by $x, y, z, \ldots$, possibly with subscripts.

We denote by $=_{AC}$ the congruence generated on $\mathcal{T}(\mathcal{F})$ by the associativity and commutativity axioms for the symbols in $\mathcal{F}_{AC}$. Let s, t, s' and t' be arbitrary terms in $\mathcal{T}(\mathcal{F})$, and let u be a non-empty context in $\mathcal{T}(\mathcal{F})$. Then an ordering on $\mathcal{T}(\mathcal{F})$ (a transitive irreflexive relation) $\succ$ fulfills the *subterm property* if $u[t] \succ t$. It fulfills the *deletion property* if $f(\ldots,t,\ldots) \succ f(\ldots\ldots)$ for all f in $\mathcal{F}_{AC}$. Furthermore, it is *monotonic* if $s \succ t$ implies $u[s] \succ u[t]$. A monotonic ordering that fulfills the subterm and deletion properties is called a *simplification ordering* and is *well-founded* ([Der87]): there are no infinite sequences $t_1 \succ t_2 \succ \ldots$ Finally, an ordering $\succ$ is *AC-compatible* if $s' =_{AC} s \succ t =_{AC} t'$ implies $s' \succ t'$.

Let s be a ground term $f(s_1 \ldots s_n)$ with $n \geq 0$. Then the size of s, denoted by $|s|$, is defined as $|s_1| + \ldots + |s_n| + 1$. Furthermore, f is the head symbol of s, denoted by $\mathcal{H}ead(s)$.

The extension of the $=_{AC}$ relation to multisets $==_{AC}$ is defined by:
$\{s_1, \ldots, s_m\} ==_{AC} \{t_1, \ldots, t_n\}$ iff $m = n$ and if $n > 0$ then there exists some t_j s.t. $s_1 =_{AC} t_j$ and $\{s_1, \ldots, s_m\} \setminus \{s_1\} ==_{AC} \{t_1, \ldots, t_n\} \setminus \{t_j\}$.

Given a relation $\succ$, the AC-multiset extension of $\succ$ on finite multisets, denoted by $\succ\!\!\succ$, is defined by:
$M = \{s_1, \ldots, s_m\} \succ\!\!\succ \{t_1, \ldots, t_n\} = N$ if

- $M \neq \emptyset$ and $N = \emptyset$ or
- $s_i =_{AC} t_j$ and $M \setminus \{s_i\} \succ\!\!\succ N \setminus \{t_j\}$, for some i in $1 \ldots m$ and j in $1 \ldots n$ or
- $s_i \succ t_{j_1} \wedge \ldots \wedge s_i \succ t_{j_k}$ and
  $(M \setminus \{s_i\} \succ\!\!\succ N \setminus \{t_{j_1}, \ldots, t_{j_k}\}$ or $M \setminus \{s_i\} ==_{AC} N \setminus \{t_{j_1}, \ldots, t_{j_k}\})$
  for some i in $1 \ldots m$ and $1 \leq j_1 < \ldots < j_k \leq n$ $(k \geq 0)$

i.e. after eliminating each occurrence at both sides of an AC-equivalent element, M is non-empty and for every element in N there is a bigger one wrt. $\succ$ in M.

Let each symbol f in the set of function symbols $\mathcal{F}$ have a status, denoted by $Stat(f)$, which can be lexicographic (lex) or multiset (mul). In the following we consider $=$ as equality of terms up to permutation of arguments for such function symbols with multiset status.

The *recursive path ordering with status* on $\mathcal{T}(\mathcal{F}, \mathcal{X})$ is based on a (*precedence*) ordering $\succ_{\mathcal{F}}$ on the set of function symbols $\mathcal{F}$: $s = f(s_1 \ldots s_m) \succ_{rpo} t$ iff

1. $\exists i \in 1 \ldots m$ with $s_i \succ_{rpo} t$ or $s_i = t$, or
2. $t = g(t_1 \ldots t_n)$ with $f \succ_{\mathcal{F}} g$ and $s \succ_{rpo} t_j$ for all $j \in 1 \ldots n$, or
3. $t = f(t_1 \ldots t_m)$, $Stat(f) = lex$, and $\langle s_1, \ldots, s_m \rangle \succ_{rpo}^{lex} \langle t_1, \ldots, t_n \rangle$, and $s \succ_{rpo} t_k$ for all $k \in 1 \ldots m$, or
4. $t = f(t_1 \ldots t_n)$, $Stat(f) = mul$, and $\{s_1, \ldots, s_m\} \succ\!\!\succ_{rpo} \{t_1, \ldots, t_n\}$.

where $\succ\!\!\succ_{rpo}$ and $\succ_{rpo}^{lex}$ are respectively the multiset and lexicographic extension of $\succ_{rpo}$.

In the following, by $s \succeq_{rpo} t$, we mean $s \succ_{rpo} t$ or $s = t$. By $max_{rpo}(S)$ we denote a maximal element wrt. $\succ_{rpo}$ of the set of ground terms S (note that all maximal ground terms in S are equal up to permutation of arguments, i.e. $max_{rpo}(S)$ is an arbitrarily chosen maximal term in S).

Lemma 2.1 The ordering $\succ_{rpo}$ is a simplification ordering on $\mathcal{T}(\mathcal{F}, \mathcal{X})$, where $s \succ_{rpo} t$ or $t \succ_{rpo} s$ for all terms s and t in $\mathcal{T}(\mathcal{F})$ with $s \neq t$. Furthermore, $s' = s \succ_{rpo} t = t'$ implies $s' \succ_{rpo} t'$, for all terms s, s', t and t' in $\mathcal{T}(\mathcal{F}, \mathcal{X})$.

Given a term t and a (terminating) rewrite system R, the set of normal forms of t wrt. R will be denoted by $snf_R(t)$.

3 The ordering

In this section the rewrite system R and the AC-RPO ordering (denoted by $\succ$) are formally defined. Note that the sets of rules are infinite due to the fact that AC-symbols can have arbitrary large arities (greater than or equal to 2).

As said, we consider that the set of function symbols $\mathcal{F}$ is totally ordered by a precedence $\succ_{\mathcal{F}}$, where $\mathcal{F}_{AC}$ is the subset containing all AC-symbols of $\mathcal{F}$. Furthermore, it is supposed that some of the function symbols in $\mathcal{F}$ (at least the AC-symbols) are chosen to be *interpreted function symbols.* We denote this subset by $\mathcal{F}_I$, and we have that $\mathcal{F}_{AC} \subseteq \mathcal{F}_I$.

Definition 3.1 By $\succ_{rpo}$ we denote the recursive path ordering with status where $Stat(f) = mul$ if $f \in \mathcal{F}_{AC}$ and $Stat(f) = lex$ otherwise.

Definition 3.2 The rewrite system R_F contains the following rules (also called of type 1):

$$f(x_1 \dots x_m, f(y_1 \dots y_r), z_1 \dots z_n) \to f(x_1 \dots x_m, y_1 \dots y_r, z_1 \dots z_n)$$

for all $f \in \mathcal{F}_{AC}$ and $m + n \geq 1$ and $r \geq 2$.

Definition 3.3 The rewrite system R_I consists of the rules (of type 2):

$$f(x_1 \dots x_m, g(y_1 \dots y_r), z_1 \dots z_n) \to f(x_1 \dots x_m, y_j, z_1 \dots z_n)$$

for all $f \in \mathcal{F}_I$, non-constant symbols g s.t. $f \succ_{\mathcal{F}} g$, $m + n + 1 \in \alpha(f)$ and $r \in \alpha(g)$ and for all $j = 1 \dots r$, and the rules (of type 3):

$$f(x_1 \dots x_m, c, y_1 \dots y_n) \to f(x_1 \dots x_m, \perp, y_1 \dots y_n)$$

for all $f \in \mathcal{F}_I$, constant symbols c s.t. $f \succ_{\mathcal{F}} c$ and $c \neq \perp$, and $m + n + 1 \in \alpha(f)$.

Let R denote the rewrite system $R_F \cup R_I$. If $f(s_1, \dots, s_m)$ is the normal form of a ground term s wrt. rewriting by R_F only at the topmost position, then the *top-flattening of* s, denoted by $tf(s)$, is $\langle s_1, \dots s_m \rangle$.

Definition 3.4 Let s and t be terms in $\mathcal{T}(\mathcal{F})$, s.t. $tf(s) = \langle s_1, \dots, s_m \rangle$ and $tf(t) = \langle t_1, \dots, t_n \rangle$. Then $s \succ t$ if and only if:

- $\forall t' \in snf_R(t) \exists s' \in snf_R(s)$ s.t. $s' \succ_{rpo} t'$ or
- $\forall t' \in snf_R(t) \exists s' \in snf_R(s)$ s.t. $s' \succeq_{rpo} t'$ and
  1. $\mathcal{H}ead(s) \in \mathcal{F}_{AC}$ and $\{s_1, \dots, s_m\} \succ\!\!\succ \{t_1, \dots, t_n\}$ or
  2. $\mathcal{H}ead(s) \notin \mathcal{F}_{AC}$ and $\langle s_1, \dots, s_m \rangle \succ^{lex} \langle t_1, \dots, t_n \rangle$.

4 Properties of the rewrite system R

In this section we will give some properties on R. These properties also lead to different ways of computing the set of normal forms efficiently.

Lemma 4.1 Let s and t be terms in $\mathcal{T}(\mathcal{F})$. If $s =_{AC} t$ then $tf(s) ==_{AC} tf(t)$.

Lemma 4.2 The relation $\rightarrow_R$ is included in $\succ_{rpo}$, hence terminating.

Lemma 4.3 Let s and t be ground terms. The following properties are satisfied:

1. $\mathcal{H}ead(s') = \mathcal{H}ead(s)$ for all s' in $snf_R(s)$
2. If $s = f(s_1 \dots s_m)$ and $f(s'_1 \dots s'_n) \in snf_R(s)$ then $n \geq m$.
3. $s \succeq_{rpo} s'$ for all s' in $snf_R(s)$.
4. If $f \notin \mathcal{F}_I$ then $snf_R(f(s_1 \dots s_n)) = \{f(s'_1 \dots s'_n) \mid s'_i \in snf_R(s_i)\}$.

The rewrite relation $\rightarrow_R$ is not confluent, because of the rules of type 2 that can rewrite e.g. $f(a, g(a, b))$ into both $f(a, a)$ and into $f(a, b)$ if $f \in \mathcal{F}_I$ and e.g. $a \succ_{\mathcal{F}} b \succ_{\mathcal{F}} f \succ_{\mathcal{F}} g$. However, the following property similar to confluence holds when computing sets of normal forms wrt. $\rightarrow_R$. Roughly speaking, it states that confluence for computing the set of normal forms of a term t is preserved if all terms created by a rule of type 2 are kept:

Lemma 4.4 (confluence for computing snf_R)

$$snf_R(t[l\sigma]) = snf_R(t[r_1\sigma]) \cup \ldots \cup snf_R(t[r_n\sigma])$$

where σ is a substitution and $\{r_1, \ldots, r_n\}$ is the set of right hand sides of all rules in R with left hand side l. Note that $n > 1$ only for a rule of type 2, i.e. when l is of the form $f(\ldots, g(x_1, \ldots, x_n), \ldots)$.

Proof We first prove the termination and confluence of the following rewrite relation $\rightarrow_{SR}$ on sets of terms:

$$S \cup \{\ t[l\sigma]\ \} \rightarrow_{SR} S \cup \{\ t[r_1\sigma]\ \} \cup \ldots \cup \{\ t[r_n\sigma]\ \}$$

with $\{r_1, \ldots, r_n\}$ as above. Termination follows from the termination of $\rightarrow_R$. The proof of (local) confluence is straightforward. It is based on an analysis of all possible critical overlaps. Here we only analyze one case. All other cases are equally simple:

The set $X \cup \{t[f(\ldots g(t_1 \ldots g(s_1 \ldots s_m) \ldots t_n) \ldots)]\}$, where $f \in \mathcal{F}_I$, $g \in \mathcal{F}_{AC}$ and $f \succ_{\mathcal{F}} g$, can be rewritten into: $X \cup \{t[f(\ldots g(t_1 \ldots s_1 \ldots s_m \ldots t_n) \ldots)]\}$, with a rule of type 1, and into $X \cup \{t[f(\ldots t_j \ldots)] \mid j = 1 \ldots n\} \cup \{t[f(\ldots g(s_1 \ldots s_m) \ldots)]\}$, with a rule of type 2. But then using two rules of type 2, both sets can be reduced in one step to $X \cup \{t[f(\ldots t_j \ldots)] \mid j = 1 \ldots n\} \cup \{t[f(\ldots s_i \ldots)] \mid i = 1 \ldots m\}$.

Now it only remains to be proved that, given a ground term t, a term s is a normal form wrt. R of t iff s belongs to the normal form of $\{t\}$ wrt. $\rightarrow_{SR}$. The right-to-left implication is trivial. For the reverse implication we proceed by induction on the size of t wrt. $\succ_{rpo}$. If t is in normal form wrt. R then the result holds. Otherwise, we have $t \rightarrow_R t' \rightarrow^*_R s$ and then also $\{t\} \rightarrow_{SR} \{t'\} \cup S \rightarrow^*_{SR} S' \cup S''$, where S' and S'' are the normal forms of $\{t'\}$ and S respectively. Then, by induction hypothesis, since $t \succ_{rpo} t'$, we have $s \in S'$. $\square$

The confluence property for computing the set of normal forms of a term of the previous lemma will be used very often in what follows. It also provides already a more efficient method for computing the set of normal forms. Later on we will go more into detail about this.

The following lemma states that AC-equal terms have the same sets of normal forms (as always, up to permutation of arguments of multiset operators):

Lemma 4.5 Let s and t be two ground terms. If $s =_{AC} t$ then $snf_R(s) = snf_R(t)$.

Proof By induction on $|s| + |t|$. Assume $f(s_1 \ldots s_m)$ and $f(t_1 \ldots t_m)$ are the terms obtained by applying the flattening rules of R_F at the topmost position of s and t respectively, and note that $snf_R(s) = snf_R(f(s_1 \ldots s_m))$ and $snf_R(t) = snf_R(f(t_1 \ldots t_m))$ by the confluence lemma 4.4.

Now for all s_i there exists t_j with $s_i =_{AC} t_j$ (in fact, if $f \notin \mathcal{F}_{AC}$ then $i = j$), and by inductive hypothesis $snf_R(s_i) = snf_R(t_j)$. Then by lemma 4.4, $snf_R(s) = \{s' \mid s' \in snf_R(f(s'_1 \ldots s'_m)) \wedge s'_i \in snf_R(s_i)\}$ and $snf_R(t) = \{t' \mid t' \in snf_R(f(t'_1 \ldots t'_m)) \wedge t'_i \in snf_R(t_i)\}$, and these sets are equal (up to permutation of arguments of multiset operators). □

5 A total AC-compatible simplification ordering on $\mathcal{T}(\mathcal{F})$

This section is devoted to the ground case. There are subsections for each one of the properties needed, which together amount to the following main result:

Theorem 5.1 The ordering $\succ$ is an AC-compatible simplification ordering on $\mathcal{T}(\mathcal{F})$ that is total on $\mathcal{T}(\mathcal{F})$ up to $=_{AC}$.

In what follows in this section, and as we are dealing with ground terms only and $\succ_{rpo}$ is total on ground terms (up to permutation of arguments of multiset operators), to abbreviate the notation we will use $mnf(t)$ for $\max_{rpo}\{snf_R(t)\}$, which is a unique ground term (up to permutation of arguments of multiset operators). Now we can also write $mnf(s) \succ_{rpo} mnf(t)$ instead of "for all $t' \in snf_R(t)$ there exists $s' \in snf_R(s)$ with $s' \succ_{rpo} t'$", (and similarly for $\succeq_{rpo}$).

5.1 AC-compatibility

Lemma 5.2 The relation $\succ$ is AC-compatible on $\mathcal{T}(\mathcal{F})$.

Proof Let s, s', t and t' be terms in $\mathcal{T}(\mathcal{F})$ with $s' =_{AC} s \succ t =_{AC} t'$. We prove by induction on $|s| + |t|$ that $s' \succ t'$. There are two possibilities:

1. $mnf(s) \succ_{rpo} mnf(t)$. Then by lemma 4.5 $mnf(s') \succ_{rpo} mnf(t')$.
2. $mnf(s) = mnf(t)$. Then by lemma 4.5 $mnf(s') = mnf(t')$. Now we suppose $tf(s) = \langle s_1, \ldots, s_m \rangle$, $tf(s') = \langle s'_1, \ldots, s'_m \rangle$, $tf(t) = \langle t_1, \ldots, t_n \rangle$ and $tf(t') = \langle t'_1, \ldots, t'_n \rangle$, with $tf(s) ==_{AC} tf(s')$ and $tf(t) ==_{AC} tf(t')$. We distinguish two cases:
   (a) $\mathcal{H}ead(s) \in \mathcal{F}_{AC}$. By induction hypothesis for all pairs (s_i, t_j) s.t. $s_i \succ t_j$ there is a pair $(s'_{i'}, t'_{j'})$ s.t. $s_i =_{AC} s'_{i'}$, $t_j =_{AC} t'_{j'}$ and $s'_{i'} \succ t'_{j'}$, and for all pairs (s_i, t_j) s.t. $s_i =_{AC} t_j$ there is a pair $(s'_{i'}, t'_{j'})$ s.t. $s'_{i'} =_{AC} t'_{j'}$. Therefore, since $\{s_1, \ldots, s_m\} \succ\!\!\succ \{t_1, \ldots, t_n\}$, we have $\{s'_1, \ldots, s'_m\} \succ\!\!\succ \{t'_1, \ldots, t'_n\}$ and $s' \succ t'$.
   (b) $\mathcal{H}ead(s) \notin \mathcal{F}_{AC}$. Then $s_1 =_{AC} t_1 \quad \wedge \ldots \wedge \quad s_{i-1} =_{AC} t_{i-1} \quad \wedge \quad s_i \succ t_i$ for some i in $1 \ldots m$. Moreover $s'_j =_{AC} s_j =_{AC} t_j =_{AC} t'_j$ for all j in $1 \ldots i{-}1$. Now by induction hypothesis $s'_i =_{AC} s_i \succ t_i =_{AC} t'_i$ implies $s'_i \succ t'_i$. Therefore $s' \succ t'$. □

5.2 Ordering relation

Lemma 5.3 The relation $\succ$ is an ordering relation on $\mathcal{T}(\mathcal{F})$, i.e. it is irreflexive and transitive.

Proof Transitivity: For all $s, t, u \in \mathcal{T}(\mathcal{F})$ we prove by induction on $|s| + |t| + |u|$ that $s \succ t$ and $t \succ u$ implies $s \succ u$. If $s \succ t$ and $t \succ u$ then $mnf(s) \succeq_{rpo} mnf(t) \succeq_{rpo} mnf(u)$. We distinguish two cases:

1. $mnf(s) \succ_{rpo} mnf(t)$ or $mnf(t) \succ_{rpo} mnf(u)$. Then $mnf(s) \succ_{rpo} mnf(u)$ follows from transitivity of $\succ_{rpo}$ and from the compatibility of $\succ_{rpo}$ with $=$.
2. Otherwise $mnf(s) = mnf(t) = mnf(u)$, and therefore $\mathcal{H}ead(s) = \mathcal{H}ead(t) = \mathcal{H}ead(u)$. Suppose $tf(s) = \langle s_1, \ldots, s_m \rangle$, $tf(t) = \langle t_1, \ldots, t_n \rangle$ and $tf(u) = \langle u_1, \ldots, u_r \rangle$. If $\mathcal{H}ead(s) \in \mathcal{F}_{AC}$ then $\{s_1, \ldots, s_m\} \succ\!\!\succ \{t_1, \ldots, t_n\}$ and $\{t_1, \ldots, t_n\} \succ\!\!\succ \{u_1, \ldots, u_r\}$. By induction hypothesis, $s_i \succ t_j$ and $t_j \succ u_k$ implies $s_i \succ u_k$, and by AC-compatibility $s_i =_{AC} t_j$ and $t_j \succ u_k$ implies $s_i \succ u_k$ and $s_i \succ t_j$ and $t_j =_{AC} u_k$ implies $s_i \succ u_k$ for all i in $1 \ldots m$, j in $1 \ldots n$ and k in $1 \ldots r$. Now, from the definition of $\succ\!\!\succ$ it follows that $\{s_1, \ldots, s_m\} \succ\!\!\succ \{u_1, \ldots, u_r\}$. If $\mathcal{H}ead(s) \notin \mathcal{F}_{AC}$ then $s_1 =_{AC} t_1 \quad \wedge \ldots \wedge \quad s_{p-1} =_{AC} t_{p-1} \quad \wedge \quad s_p \succ t_p$, for some p in $1 \ldots m$, and $t_1 =_{AC} u_1 \quad \wedge \ldots \wedge \quad t_{q-1} =_{AC} u_{q-1} \quad \wedge \quad t_q \succ u_q$, for some q in $1 \ldots m$. But then by induction hypothesis and AC-compatibility, for $k = min(p, q)$ we have $s_1 =_{AC} u_1 \quad \wedge \ldots \wedge \quad s_{k-1} =_{AC} u_{k-1} \quad \wedge \quad t_k \succ u_k$.

Irreflexivity: We prove $s \not\succ s$ by induction on $|s|$. By irreflexivity of $\succ_{rpo}$ we have $mnf(s) \not\succ_{rpo} mnf(s)$, so we only need to consider the second case of the definition of $\succ$. If $tf(s)$ is $\langle s_1, \ldots, s_n \rangle$, then by inductive hypothesis, the relation $\succ$ is irreflexive for all s_i with i in $1 \ldots n$. If $\mathcal{H}ead(s) \in \mathcal{F}_{AC}$ then, since $\succ$ is irreflexive, transitive and AC-compatible for all s_i, from the definition of $\succ\!\!\succ$ it follows that $\{s_1, \ldots, s_n\} \not\succ\!\!\succ \{s_1, \ldots, s_n\}$. On the other hand if $\mathcal{H}ead(s) \notin \mathcal{F}_{AC}$ then $s \not\succ s$, since $s_i \not\succ s_i$ for all i in $1 \ldots n$. $\Box$

5.3 Subterm and deletion properties

Lemma 5.4 The ordering $\succ$ fulfills the subterm and deletion properties.

Proof Subterm property: We obtain $f(\ldots s \ldots) \succ s$ for all $s \in \mathcal{T}(\mathcal{F})$ by proving directly $mnf(f(\ldots s \ldots)) \succ_{rpo} mnf(s)$.

From the confluence lemma 4.4 it follows that $snf_R(f(\ldots s \ldots)) =$ $\{snf_R(f(u_1 \ldots u_p, s', v_1 \ldots v_q)) \mid s' \in snf_R(s) \wedge f(u_1 \ldots u_p, \perp, v_1 \ldots v_q) \in snf_R(f(\ldots \perp \ldots))\}$. Note that the reducibility of a context $f(u_1 \ldots u_p, s, v_1 \ldots v_q)$ of a term s by our particular set of rules R does not depend on s, and therefore all remaining reduction steps of $f(u_1 \ldots u_p, s', v_1 \ldots v_q)$ must involve the subterm s' and (since s' itself is also in normal form) be applied at the topmost position of $f(u_1 \ldots u_p, s', v_1 \ldots v_q)$.

Now by the properties of $\succ_{rpo}$ it follows that in fact $mnf(f(\ldots s \ldots)) \in \{s'' \mid s'' \in snf_R(f(u_1 \ldots u_p, s', v_1 \ldots v_q)) \wedge s' \in snf_R(s)\}$ where $f(u_1 \ldots u_p, \perp, v_1 \ldots v_q) = mnf(f(\ldots, \perp, \ldots))$. This allows us to suppose here w.l.o.g. that contexts of the form $f(\ldots s \ldots)$ of the term s are already normalized by R (this will also be done in many subsequent lemmas, like the monotonicity lemmas, when dealing with such contexts).

So here it suffices to prove that for all terms $f(u_1 \ldots u_p, s', v_1 \ldots v_q)$ that are in normal form except for the rules applied at the topmost position and that involve s', we have $mnf(f(u_1 \ldots u_p, s', v_1 \ldots v_q)) \succ_{rpo} s'$. We proceed by induction on $|s|'$.

Assume $s' = g(s_1 \ldots s_m)$, and, since s' is in normal form, each s_i is in normal form wrt. R too. We consider the following four cases:

1. $f \notin \mathcal{F}_I$ or $g \succ_{\mathcal{F}} f$. Then
   $snf_R(f(u_1 \ldots u_p, g(s_1 \ldots s_m), v_1 \ldots v_q)) = \{f(u_1 \ldots u_p, g(s_1 \ldots s_m), v_1 \ldots v_q)\}$,
   which (by the subterm property of $\succ_{rpo}$) is greater wrt. $\succ_{rpo}$ than $g(s_1 \ldots s_m)$.
2. $f = g$. Then if $f \notin \mathcal{F}_{AC}$ as before it holds; otherwise
   $snf_R(f(u_1 \ldots u_p, f(s_1 \ldots s_m), v_1 \ldots v_q)) = \{f(u_1 \ldots u_p, s_1 \ldots s_m, v_1 \ldots v_q)\}$,
   which is greater wrt. $\succ_{rpo}$ than $f(s_1 \ldots s_m)$ since $p + q > 0$.
3. $f \in \mathcal{F}_I$, $f \succ_{\mathcal{F}} g$ and $m > 0$. Then
   $snf_R(f(u_1 \ldots u_p, g(s_1 \ldots s_m), v_1 \ldots v_q)) = \cup_{j=1}^{m} snf_R(f(u_1 \ldots u_p, s_j, v_1 \ldots v_q))$.
   Suppose $s_k = max_{rpo}\{s_1, \ldots, s_m\}$. Then by induction hypothesis there is a $t' \in snf_R(f(u_1 \ldots u_p, s_k, v_1 \ldots v_q))$ with $t' \succ_{rpo} s_k$, and since $f \succ_{\mathcal{F}} g$ and $\mathcal{H}ead(t') = f$, by definition of $\succ_{rpo}$ it follows that $t' \succ_{rpo} g(s_1 \ldots s_m)$.
4. $f \in \mathcal{F}_I$, $f \succ_{\mathcal{F}} g$ and $m = 0$. Then we know that $snf_R(f(u_1 \ldots u_p, g, v_1 \ldots v_q)) = \{f(u_1 \ldots u_p, \bot, v_1 \ldots v_q)\}$, which is greater wrt. $\succ_{rpo}$ than g.

Deletion property: We obtain $f(\ldots s \ldots) \succ f(\ldots\ldots)$ for all $f \in \mathcal{F}_{AC}$ by proving directly $mnf(f(\ldots s \ldots)) \succ mnf(f(\ldots\ldots))$.

By the forms our rules have and by confluence (lemma 4.4), we know that for all $s'' = f(u_1 \ldots u_p, v_1 \ldots v_q) \in snf_R(f(\ldots\ldots))$ there exists at least one $s' = f(u_1 \ldots u_p, t, v_1 \ldots v_q) \in snf_R(f(\ldots s \ldots)$, and (by the deletion property of $\succ_{rpo}$) $s' \succ_{rpo} s''$. $\square$

5.4 Monotonicity

The following lemma will be used to prove the monotonicity of $\succ$ and also in following sections for different purposes. It describes when the ordering relation $\succ_{rpo}$ between terms is preserved under addition of a context and interpretation:

Lemma 5.5 For all ground $f(\ldots s \ldots)$ and $f(\ldots t \ldots)$,

1. $mnf(s) \succeq_{rpo} mnf(t)$ implies $mnf(f(\ldots s \ldots)) \succeq_{rpo} mnf(f(\ldots t \ldots))$
2. $mnf(s) \succ_{rpo} mnf(t)$ implies $mnf(f(\ldots s \ldots)) \succ_{rpo} mnf(f(\ldots t \ldots))$ if $\mathcal{H}ead(s) \succeq_{\mathcal{F}} f$.

Proof We first prove by induction on $|s| + |t|$ the result for the case when $mnf(s) = s$ and $mnf(t) = t$, i.e. when s and t are normal forms wrt. R. If $s = t$ then this holds trivially, so we show $s \succ_{rpo} t$ implies $mnf(cs) \succeq_{rpo} mnf(ct)$ for $cs = f(u_1 \ldots u_p, s, v_1 \ldots v_q)$ and $ct = f(u_1 \ldots u_p, t, v_1 \ldots v_q)$ for all $u_1 \ldots u_p, v_1 \ldots v_q$ in $\mathcal{T}(\mathcal{F})$ s.t. $p + q + 1 \in \alpha(f)$. As in lemma 5.2, we can suppose w.l.o.g. that the context $f(u_1 \ldots u_p, v_1 \ldots v_q)$ is in normal form wrt. R. Let s be $g(s_1 \ldots s_m)$ and t be $h(t_1 \ldots t_n)$.

If $f \notin \mathcal{F}_I$ or $g \succ_{\mathcal{F}} f$ then by monotonicity of $\succ_{rpo}$ and since $\rightarrow_R \subseteq \succ_{rpo}$, it follows that $mnf(cs) = f(\ldots s \ldots) \succ_{rpo} f(\ldots t \ldots) \succeq_{rpo} mnf(f(\ldots t \ldots)) = mnf(ct)$.

Otherwise, if $f \succ_{\mathcal{F}} h$ and t is not a constant then by lemma 4.4 $snf_R(ct) = snf_R(f(\ldots t_1 \ldots)) \cup \ldots \cup snf_R(f(\ldots t_n \ldots))$. Since $s \succ_{rpo} t$, we have $s \succ_{rpo} t_i$ for each t_i, and by induction hypothesis, $mnf(cs) \succeq_{rpo} mnf(f(\ldots t_i \ldots))$ for each t_i which implies $mnf(cs) \succeq_{rpo} mnf(ct)$. If $\mathcal{H}ead(s) = f$ then $mnf(cs) \succ_{rpo} mnf(f(\ldots t_i \ldots))$ by induction hypothesis and then $mnf(cs) \succ_{rpo} mnf(ct)$. If ($f \succ_{\mathcal{F}} h$ and) t is a constant ($\bot$ or not) then $mnf(cs) \succeq_{rpo} mnf(ct) = f(u_1 \ldots u_p, \bot, v_1 \ldots v_q)$. When $f \in \mathcal{F}_{AC}$, the result follows similarly (using the fact that if $\mathcal{H}ead(s) = f$ then $m > 1$). If $f \notin \mathcal{F}_{AC}$ and $\mathcal{H}ead(s) = f$, then, since $s \succ_{rpo} \bot$, $mnf(cs) = f(u_1 \ldots u_p, s, v_1 \ldots v_q) \succ_{rpo} f(u_1 \ldots u_p, \bot, v_1 \ldots v_q)$ by monotonicity of $\succ_{rpo}$.

For the remaining cases we have $f \in \mathcal{F}_I$ and $h \succeq_{\mathcal{F}} f \succeq_{\mathcal{F}} g$:

1. $h \succeq_{\mathcal{F}} f \succ_{\mathcal{F}} g$. Then we have $snf_R(cs) = snf_R(f(\ldots s_1 \ldots)) \cup \ldots \cup snf_R(f(\ldots s_m \ldots))$. Also, since $s \succ_{rpo} t$ and also $h \succ_{\mathcal{F}} g$, we have $s_i \succeq_{rpo} t$ for some s_i, and by induction hypothesis $mnf(f(\ldots s_i \ldots)) \succ_{rpo} mnf(ct)$. But as $snf_R(cs) \supseteq snf_R(f(\ldots s_i \ldots))$, we get $mnf(cs) \succeq_{rpo} mnf(ct)$.

2. $h \succ_{\mathcal{F}} f = g$. If $f \in \mathcal{F}_{AC}$ then $mnf(cs) = f(\ldots s_1 \ldots s_m \ldots)$ and since $s \succ_{rpo} t$ with $h \succ_{\mathcal{F}} g$, we have $s_i \succeq_{rpo} t$ for some s_i. Then by the deletion property and monotonicity of $\succ_{rpo}$ and since $\rightarrow_R \subseteq \succ_{rpo}$ and $m > 1$ we have $mnf(cs) = f(\ldots s_1 \ldots s_m \ldots) \succ_{rpo} f(\ldots s_i \ldots) \succeq_{rpo} f(\ldots t \ldots) \succeq_{rpo} mnf(f(\ldots t \ldots)) = mnf(ct)$. Otherwise, if $f \notin \mathcal{F}_{AC}$ then $mnf(cs) = f(\ldots s \ldots) \succ_{rpo} f(\ldots t \ldots) \succeq_{rpo} mnf(ct)$.

3. $h = f = g$. If $f \in \mathcal{F}_{AC}$ then since $s \succ_{rpo} t$ we have $\{s_1 \ldots s_m\} \succ\!\!\succ_{rpo} \{t_1 \ldots t_n\}$, and $mnf(cs) = f(\ldots s_1 \ldots s_m \ldots) \succ_{rpo} f(\ldots t_1 \ldots t_n \ldots) = mnf(ct)$. If $f \notin \mathcal{F}_{AC}$ then, as in the previous case, $mnf(cs) \succ_{rpo} mnf(ct)$.

Now we consider the general case, i.e. when s and t are not in normal form. We know by the confluence lemma 4.4 that $snf_R(cs) = snf_R(f(\ldots s_1 \ldots)) \cup \ldots \cup snf_R(f(\ldots s_m \ldots))$, where $snf_R(s) = \{s_1 \ldots s_m\}$, and $snf_R(ct) = snf_R(f(\ldots t_1 \ldots)) \cup \ldots \cup snf_R(f(\ldots t_n \ldots))$ if $snf_R(t) = \{t_1 \ldots t_n\}$. Now since $mnf(s) \succeq_{rpo} mnf(t)$ we know that for each such t_i there is an s_j with $s_j \succeq_{rpo} t_i$. But then by the first part of this proof, $mnf(f(\ldots s_j \ldots)) \succeq_{rpo} mnf(f(\ldots t_i \ldots))$, and therefore $mnf(f(\ldots s \ldots)) \succeq_{rpo} mnf(f(\ldots t \ldots))$ (and similarly we get $\succ_{rpo}$ instead of $\succeq_{rpo}$ when $g \succeq_{\mathcal{F}} f$). □

Lemma 5.6 The ordering $\succ$ is monotonic, i.e. $s \succ t$ implies $f(\ldots s \ldots) \succ f(\ldots t \ldots)$ for all $f(\ldots s \ldots)$ and $f(\ldots t \ldots)$ in $\mathcal{T}(\mathcal{F})$.

Proof We will show that $cs = f(u_1 \ldots u_p, s, v_1 \ldots v_q) \succ f(u_1 \ldots u_p, t, v_1 \ldots v_q) = ct$ for all $u_1 \ldots u_p, v_1 \ldots v_q$ in $\mathcal{T}(\mathcal{F})$ s.t. $p + q + 1 \in \alpha(f)$.

If $s \succ t$ then $mnf(s) \succeq_{rpo} mnf(t)$, and by lemma 5.5 $mnf(cs) \succeq_{rpo} mnf(ct)$. If $mnf(cs) \succ_{rpo} mnf(ct)$ then $cs \succ ct$, so from now on we suppose $mnf(cs) = mnf(ct)$. We distinguish three cases:

1. $f \notin \mathcal{F}_{AC}$. Then $cs \succ ct$, since
   $tf(cs) = \langle u_1 \ldots u_p, s, v_1 \ldots v_q \rangle \succ^{lex} \langle u_1 \ldots u_p, t, v_1 \ldots v_q \rangle = tf(ct)$.

2. $f \in \mathcal{F}_{AC}$ and $mnf(s) \succ_{rpo} mnf(t)$. Then $\mathcal{H}ead(s) \neq f$ since otherwise, by lemma 5.5, $mnf(cs) \succ_{rpo} mnf(ct)$ and also $tf(f(u_1 \ldots u_p, s, v_1 \ldots v_q)) = \{u'_1 \ldots u'_{p'}, s, v'_1 \ldots v'_{q'}\}$. Furthermore, depending on whether $\mathcal{H}ead(t) = f$ or not, $tf(f(u_1 \ldots u_p, t, v_1 \ldots v_q))$ can be either $\{u'_1 \ldots u'_{p'}, t, v'_1 \ldots v'_{q'}\}$ or of the form $\{u'_1 \ldots u'_{p'}, t_1 \ldots t_n, v'_1 \ldots v'_{q'}\}$, where each t_i is a proper subterm of t. In both cases $s \succ t$ implies, by the subterm property and transitivity of $\succ$, that $s \succ t_i$, and then $tf(f(u_1 \ldots u_p, s, v_1 \ldots v_q)) \succ\!\!\succ tf(f(u_1 \ldots u_p, t, v_1 \ldots v_q))$.

3. $f \in \mathcal{F}_{AC}$ and $mnf(s) = mnf(t)$, and therefore $\mathcal{H}ead(s) = \mathcal{H}ead(t)$. Now, if $f \neq \mathcal{H}ead(s)$ then $tf(f(u_1 \ldots u_p, s, v_1 \ldots v_q)) = \{u'_1 \ldots u'_{p'}, s, v'_1 \ldots v'_{q'}\} \succ\!\!\succ \{u'_1 \ldots u'_{p'}, t, v'_1 \ldots v'_{q'}\} = tf(f(u_1 \ldots u_p, t, v_1 \ldots v_q))$. If $f = \mathcal{H}ead(s)$ then $tf(f(u_1 \ldots u_p, s, v_1 \ldots v_q)) = \{u'_1 \ldots u'_{p'}, s_1 \ldots s_m, v'_1 \ldots v'_{q'}\} \succ\!\!\succ \{u'_1 \ldots u'_{p'}, t_1 \ldots t_n, v'_1 \ldots v'_{q'}\} = tf(f(u_1 \ldots u_p, t, v_1 \ldots v_q))$, since $tf(s) = \{s_1, \ldots, s_m\} \succ\!\!\succ \{t_1, \ldots, t_n\} = tf(t)$. □

5.5 Totality

Lemma 5.7 The ordering $\succ$ is total on the AC-congruence classes of $\mathcal{T}(\mathcal{F})$, i.e. for all s, t in $\mathcal{T}(\mathcal{F})$ either $s =_{AC} t$ or $s \succ t$ or $t \succ s$.

Proof By induction on $|s|+|t|$. Suppose $s \neq_{AC} t$. If $mnf(s) \succ_{rpo} mnf(t)$ or $mnf(t) \succ_{rpo} mnf(s)$ then s and t are comparable. Otherwise, $mnf(s) = mnf(t)$ and $\mathcal{H}ead(s) = \mathcal{H}ead(t)$. Suppose $tf(s) = \langle s_1, \ldots, s_m \rangle$ and $tf(t) = \langle t_1, \ldots, t_n \rangle$. There are two possibilities:

1. $\mathcal{H}ead(s) \in \mathcal{F}_{AC}$. Then by induction hypothesis all terms s_i and t_j are comparable by $\succ$, i.e. $u =_{AC} v$ or $u \succ v$ or $v \succ u$ for all $u, v \in \{s_1, \ldots, s_m, t_1, \ldots, t_n\}$. It cannot be the case that $\{s_1, \ldots, s_m\} ==_{AC} \{t_1, \ldots, t_n\}$ because $s \neq_{AC} t$, and therefore $m+n \neq 0$. We now prove that such sets $\{s_1, \ldots, s_m\}$ and $\{t_1, \ldots, t_n\}$ are comparable wrt. $\succ\!\!\succ$ by induction on $m+n$. If $m=0$ or $n=0$ then this is the case. Otherwise, since $\succ$ is total on $\{s_1, \ldots, s_m\}$ and on $\{t_1, \ldots, t_n\}$, there are terms u and v that are maximal wrt. $\succ$ in $\{s_1, \ldots, s_m\}$ and in $\{t_1, \ldots, t_n\}$ respectively. If $u =_{AC} v$ then by induction hypothesis $\{s_1, \ldots, s_m\} \setminus \{u\}$ is comparable (wrt. $\succ\!\!\succ$) with $\{t_1, \ldots, t_n\} \setminus \{v\}$ (since they are not equivalent wrt. $==_{AC}$) and therefore $\{s_1, \ldots, s_m\}$ and $\{t_1, \ldots, t_n\}$ are comparable wrt. $\succ\!\!\succ$. If $u \neq_{AC} v$ then suppose wlog. that $u \succ v$. By transitivity and AC-compatibility $u \succ t_j$ for all j in $1 \ldots n$ and therefore $\{s_1, \ldots, s_m\} \succ\!\!\succ \{t_1, \ldots, t_n\}$.
2. $\mathcal{H}ead(s) \notin \mathcal{F}_{AC}$. Then if i is the minimum in $1 \ldots m$ s.t. $s_i \neq_{AC} t_i$, by induction hypothesis $s_i \succ t_i$ or $t_i \succ s_i$ and therefore $s \succ t$ or $t \succ s$. $\square$

6 Implementing the ordering

From the two following lemmas we will obtain an algorithm to efficiently compare terms in our ordering.

Lemma 6.1 $mnf(f(s_1 \ldots s_n)) = mnf(f(mnf(s_1) \ldots mnf(s_n)))$ for every $f(s_1 \ldots s_n)$ in $\mathcal{T}(\mathcal{F})$.

Proof By lemma 4.4 we have $mnf(f(s_1 \ldots s_n)) = max_{rpo}(snf_R(f(s_1 \ldots s_n))) = max_{rpo}\{t \mid t \in snf_R(f(s'_1 \ldots s'_n)) \wedge s'_i \in snf_R(s_i)\} = max_{rpo}\{mnf(f(s'_1 \ldots s'_n)) \mid s'_i \in snf_R(s_i)\}$, which is equal to $mnf(f(mnf(s_1) \ldots mnf(s_n)))$, since applying lemma 5.5 n times:
$mnf(f(mnf(s_1), mnf(s_2) \ldots mnf(s_n))) \succeq_{rpo} mnf(f(s'_1, mnf(s_2), \ldots mnf(s_n))) \succeq_{rpo} \ldots\ldots \succeq_{rpo} mnf(f(s'_1, s'_2 \ldots s'_{n-1}, mnf(s_n))) \succeq_{rpo} mnf(f(s'_1, s'_2, \ldots s'_{n-1}, s'_n))$. Then, by transitivity of $\succ_{rpo}$ it follows that: $mnf(f(mnf(s_1) \ldots mnf(s_n))) \succeq_{rpo} mnf(f(s'_1 \ldots s'_n))$. $\square$

Lemma 6.2 If $t_1, \ldots, t_n$ are ground terms in normal form wrt. R, $f \in \mathcal{F}_I$ and $f \succ_{\mathcal{F}} g$ then $mnf(f(\ldots g(t_1 \ldots t_n) \ldots)) = mnf(f(\ldots t \ldots))$, where t is the maximal term wrt. $\succ_{rpo}$ in $\{t_1, \ldots, t_n\}$.

Proof By lemma 4.4 it holds that $mnf(f(\ldots g(t_1 \ldots t_n) \ldots)) =$
$max_{rpo}(snf_R(f(\ldots g(t_1 \ldots t_n) \ldots))) = max_{rpo}(snf_R(f(\ldots t_1 \ldots)) \cup \ldots \cup snf_R(f(\ldots t_n \ldots)))$
$= max_{rpo}\{mnf(f(\ldots t_1 \ldots)), \ldots, mnf(f(\ldots t_n \ldots))\}$ which is equal to $mnf(f(\ldots t \ldots))$, since $t = mnf(t) \succeq_{rpo} mnf(t_i) = t_i$ (note that each t_i is in normal form), and then by lemma 5.5 we have $mnf(f(\ldots t \ldots)) \succeq_{rpo} mnf(f(\ldots t_i \ldots))$. $\square$

From this lemma and lemma 6.1 we obtain an algorithm to efficiently compute the $mnf(t)$ for some ground term t by rewriting innermost and instead of applying m rules of type 2 of the form $f(\ldots g(u_1 \ldots u_m) \ldots) \rightarrow f(\ldots u_j \ldots)$, just using one of them: $f(\ldots g(u_1 \ldots u_m) \ldots) \rightarrow f(\ldots u \ldots)$ where u is maximal wrt. $\succ_{rpo}$ in $\{u_1, \ldots, u_m\}$ (this way of rewriting innermost with R and applying rules of type 2 keeping only the maximal subterm was the original definition of the interpretation in our preliminary paper [RN93]). In this way we also obtain, as a side effect, the $mnf(t')$ for all subterms t' of t, which may be needed when checking whether $t \succ s$ (for some other ground term s) using the second branch of the definition of $\succ$.

Note that non-innermost versions of this strategy do not work, because then sometimes for a term t a non-maximal normal form is obtained: with $f \succ g \succ h \succ i \succ a \succ b$ and $h, f \in \mathcal{F}_{AC} = \mathcal{F}_I$, for the term $t = h(\ b,\ i(\ f(a,b), g(f(b,b))\ )\ )$ we have $mnf(t) = h(b, g(f(b,b)))$, but by non-innermost rewriting we would obtain $h(b, f(b,b))$.

This method provides a simple way to implement our ordering in any existing rewrite laboratory in which a rewriting mechanism and the RPO ordering are available.

In this way, to get an even easier definition of the rewrite rules and avoid fixing a strategy while keeping the uniqueness of the normal form in the ground case, we can write such rules of type 2 as:

$$f(\ldots g(u_1 \ldots u_m) \ldots) \to f(\ldots u \ldots) \qquad \text{if } u \text{ is maximal wrt. } \succ \text{ in } \{u_1 \ldots u_m\}.$$

Note that we have used $\succ$ instead of $\succ_{rpo}$. This is allowed since $s \succ t$ implies $mnf(s) \succeq_{rpo} mnf(t)$, for all ground terms s and t.

7 Terms with variables

In theorem 7.2 we will prove that $\succ$ is stable under ground substitutions: for all s and t in $\mathcal{T}(\mathcal{F}, \mathcal{X})$ and for all ground substitutions σ, if $s \succ t$ then $s\sigma \succ t\sigma$. This allows us to use the ordering for all practical purposes like orienting non-ground rewrite systems or —which was our main motivation for this work— to obtain complete theorem proving strategies with built-in AC-unification, and to the extension to the AC-case of the deduction methods for constrained clauses. First we need the following lemma:

Lemma 7.1 Let s and t be terms in $\mathcal{T}(\mathcal{F}, \mathcal{X})$ in normal form wrt. R. Then $s \succ_{rpo} t$ implies $mnf(s\sigma) \succ_{rpo} mnf(t\sigma)$ for all ground substitutions σ.

Proof By induction on $|s| + |t|$. If t is some variable x, then, since $s \succ_{rpo} t$, x is a proper subterm of s and $t\sigma$ is a proper subterm of $s\sigma$. Then by (the proof of) lemma 5.4, $mnf(s\sigma) \succ_{rpo} mnf(t\sigma)$.

If t is not a variable, then it is of the form $g(t_1 \ldots t_n)$ with $n \geq 0$. Since $s \succ_{rpo} t$, the term s cannot be a variable, so let s be of the form $f(s_1 \ldots s_m)$ with $m \geq 0$. Then we have the following cases:

1. $g \succ_{\mathcal{F}} f$. Then there is an s_i s.t. $s_i \succeq_{rpo} t$. If $s_i = t$ then, by lemma 5.4, $mnf(s\sigma) \succ_{rpo} mnf(s_i\sigma) = mnf(t\sigma)$. Similarly, if $s_i \succ_{rpo} t'$ then, by the induction hypothesis and by lemma 5.4, $mnf(s\sigma) \succ_{rpo} mnf(s_i\sigma) \succ_{rpo} mnf(t\sigma)$.

2. $f \succ_{\mathcal{F}} g$. Then $s \succ_{rpo} t_i$ and, by induction hypothesis, $mnf(s\sigma) \succ_{rpo} mnf(t_i\sigma)$ for all i in $1 \ldots n$. Now, since $\mathcal{H}ead(s\sigma) = f$, $mnf(s\sigma) \succ_{rpo} g(mnf(t_1\sigma), \ldots, mnf(t_n\sigma))$ and therefore $mnf(s\sigma) \succ_{rpo} t'$ for all $t' \in \{g(t'_1, \ldots t'_n) \mid t'_i \in snf_R(t_i\sigma)\}$. But since $t' \succeq_{rpo} mnf(t')$ because $\to_R \subseteq \succ_{rpo}$, and $mnf(t') = mnf(t\sigma)$ for one such t', we obtain $mnf(s\sigma) \succ_{rpo} mnf(t\sigma)$.

3. $f = g$ and $f \notin \mathcal{F}_{AC}$ (but maybe $f \in \mathcal{F}_I$). Then $n = m$ and $s_1 = t_1 \wedge \ldots \wedge s_{i-1} = t_{i-1} \wedge s_i \succ_{rpo} t_i$ for some i in $1 \ldots m$, where s_i is not variable, and $s \succ_{rpo} t_k$ for all k in $i+1 \ldots n$. Moreover, by lemma 4.4 and the properties of $\succ_{rpo}$, it holds that
   $mnf(s\sigma) = f(s'_1, \ldots, s'_n)$, where $mnf(f(\perp, s_j)) = f(\perp, s'_j)$ for all j in $1 \ldots n$, and
   $mnf(t\sigma) = f(t'_1, \ldots, t'_n)$, where $mnf(f(\perp, t_j)) = f(\perp, t'_j)$ for all j in $1 \ldots n$
   (note that we have no flattening rules for f). But $s'_j = t'_j$ for j in $1 \ldots i-1$, and, by the fact that s_i is not a variable and by induction hypothesis, we have $s'_i = mnf(s_i\sigma) \succ_{rpo} mnf(t_i\sigma) \succeq_{rpo} t'_i$ and $mnf(s\sigma) \succ_{rpo} mnf(t_k\sigma) \succeq_{rpo} t'_k$ for all k in $i+1 \ldots n$. Therefore, $mnf(s\sigma) \succ_{rpo} mnf(t\sigma)$.

4. $f = g$ and $f \in \mathcal{F}_{AC}$. Then $\{s_1 \dots s_m\} \succ\!\!\succ_{rpo} \{t_1 \dots t_n\}$. Now let $\{s_1 \dots s_p\}$ and $\{t_1 \dots t_q\}$ be the sets obtained by deleting all common terms $\{s_{p+1} \dots s_m\}$ and $\{t_{q+1} \dots t_n\}$ between $\{s_1 \dots s_m\}$ and $\{t_1 \dots t_n\}$.

   Then still $\{s_1 \dots s_p\} \succ\!\!\succ_{rpo} \{t_1 \dots t_q\}$ and for all t_j there exists an s_i s.t. $s_i \succ_{rpo} t_j$ and by induction hypothesis, $mnf(s_i\sigma) \succ_{rpo} mnf(t_j\sigma)$. Note that for all these s_i, we have $\mathcal{H}ead(s_i) \succ_{\mathcal{F}} f$ since s is in normal form wrt. R and $s_i \neq \bot$. Let $mnf(s_1\sigma)$ be maximal in $\{mnf(s_i\sigma) \mid i = 1 \dots p \wedge \mathcal{H}ead(s_i) \succ_{\mathcal{F}} f\}$. Then $mnf(s_1\sigma) \succ_{rpo} mnf(t_j\sigma)$ for all j in $1 \dots q$.

   Now note that $mnf(s\sigma) = mnf(f(s_1\sigma \dots s_m\sigma))$ which by lemma 6.1 is equivalent to $mnf(f(mnf(s_1\sigma) \dots mnf(s_m\sigma)))$ which will be of the form $f(mnf(s_1\sigma), u_{21} \dots u_{2k_2} \; \dots \; u_{m1} \dots u_{mk_m})$, since $\mathcal{H}ead(s_1) \succ_{\mathcal{F}} f$.

   Similarly, $mnf(t\sigma)$ is equivalent to $mnf(f(mnf(t_1\sigma) \dots mnf(t_n\sigma)))$ which will be of the form $f(u'_{11} \dots u'_{1k'_1} \; \dots \; u'_{n1} \dots u'_{nk'_n})$.

   Now to prove that $mnf(s\sigma) \succ_{rpo} mnf(t\sigma)$ we need $\{mnf(s_1\sigma), u_{21} \dots u_{2k_2} \dots u_{m1} \dots u_{mk_m}\} \succ\!\!\succ_{rpo} \{u'_{11} \dots u'_{1k'_1} \dots u'_{n1} \dots u'_{nk'_n}\}$. This is indeed the case, as (i) the terms u and u' coming from the common terms $\{s_{p+1} \dots s_m\}$ and $\{t_{q+1} \dots t_n\}$ are equal and can again be removed, and (ii) for the remaining terms u' we have $mnf(s_1\sigma) \succ_{rpo} u'$. For part (ii) note the following: Each $mnf(t_j\sigma)$ produces terms $u'_{j1} \dots u'_{jk'_j}$ in $mnf(t\sigma)$. But by the rules of R in fact these $u'_{j1} \dots u'_{jk'_j}$ also fulfil that $f(\bot, u'_{j1} \dots u'_{jk'_j}) = mnf(f(\bot, mnf(t_j\sigma)))$. But then by lemma 5.5 $mnf(s_1\sigma) \succ_{rpo} mnf(t_j\sigma)$ implies $mnf(f(\bot, mnf(s_1\sigma))) \succ_{rpo} mnf(f(\bot, mnf(t_j\sigma)))$ since $\mathcal{H}ead(s_1) \succ_{\mathcal{F}} f$, that is, $f(\bot, mnf(s_1\sigma)) \succ_{rpo} f(\bot, u'_{j1} \dots u'_{jk'_j}))$, and therefore $mnf(s_1\sigma) \succ_{rpo} u'_{j1} \dots u'_{jk'_j}$. □

Theorem 7.2 For all s and t in $\mathcal{T}(\mathcal{F}, \mathcal{X})$ and for all ground substitutions σ, if $s \succ t$ then $s\sigma \succ t\sigma$.

Proof By induction on $|s| + |t|$. Suppose $snf_R(s) = \{s'_1, \dots, s'_p\}$ and $snf_R(t) = \{t'_1, \dots, t'_q\}$. Then by the confluence lemma 4.4 $snf_R(s\sigma) = snf_R(s'_1\sigma) \cup \dots \cup snf_R(s'_p\sigma)$ and $snf_R(t\sigma) = snf_R(t'_1\sigma) \cup \dots \cup snf_R(t'_q\sigma)$.

If $s \succ t$ by the first branch of the ordering, i.e. for all $t' \in \{t'_1, \dots, t'_q\}$ there exists $s' \in \{s'_1, \dots, s'_p\}$, s.t. $s' \succ_{rpo} t'$, then by lemma 7.1 for all t'_j there is an s'_i s.t. $mnf(s'_i\sigma) \succ_{rpo} mnf(t'_j\sigma)$, i.e. $mnf(s\sigma) \succ_{rpo} mnf(t\sigma)$ and $s\sigma \succ t\sigma$.

If $s \succ t$ by the second branch of the ordering, i.e. for all $t' \in \{t'_1, \dots, t'_q\}$ there exists $s' \in \{s'_1, \dots, s'_p\}$, s.t. $s' \succeq_{rpo} t'$, then by lemma 7.1 for all t'_j there is an s'_i s.t. $mnf(s'_i\sigma) \succeq_{rpo} mnf(t'_j\sigma)$, i.e. $\mathcal{H}ead(s) = \mathcal{H}ead(t)$ and $mnf(s\sigma) \succeq_{rpo} mnf(t\sigma)$ and we can apply the second part of the definition of $\succ$ to prove $s\sigma \succ t\sigma$.

Suppose $tf(s) = \{s_1, \dots, s_m\}$ and $tf(t) = \{t_1, \dots, t_n\}$. If $\mathcal{H}ead(s) \notin \mathcal{F}_{AC}$ then $s_1 =_{AC} t_1 \wedge \dots \wedge s_{i-1} =_{AC} t_{i-1} \wedge s_i \succ t_i$ implies by induction hypothesis $s_1\sigma =_{AC} t_1\sigma \wedge \dots \wedge s_{i-1}\sigma =_{AC} t_{i-1}\sigma$ and $s_i\sigma \succ t_i\sigma$, which implies $s\sigma \succ t\sigma$.

If $\mathcal{H}ead(s) \in \mathcal{F}_{AC}$ then $\{s_1, \dots, s_m\} \succ\!\!\succ \{t_1, \dots, t_n\}$, and by induction hypothesis $s_i \succ t_j$ implies $s_i\sigma \succ t_j\sigma$, which implies $\{s_1\sigma, \dots, s_m\sigma\} \succ\!\!\succ \{t_1\sigma, \dots, t_n\sigma\}$.

However, $tf(s\sigma)$ is not necessarily $\{s_1\sigma, \dots, s_m\sigma\}$, since some s_i may be a variable x with $\mathcal{H}ead(x\sigma) = f$. But note that the (occurrences of) common terms in both $tf(s)$ and $tf(t)$ need not be considered and that $\{s_1, \dots, s_m\} \succ\!\!\succ \{t_1, \dots, t_n\}$ implies that for each t_j with no corresponding occurrence in $\{s_1, \dots, s_m\}$ there must be a non-variable s_i with $s_i \succ t_j$. Therefore, we have $tf(s\sigma) \succ\!\!\succ tf(t\sigma)$ and $s\sigma \succ t\sigma$. □

As a consequence of the lemma 7.1, the implementation technique described in the previous section can also be applied when comparing terms with variables, with the modification that in the application of rules of type 2 there may be more than one maximal term, and all rules $f(\dots g(u_1 \dots u_m) \dots) \rightarrow f(\dots u \dots)$ where u is maximal wrt. $\succ_{rpo}$ in $\{u_1, \dots, u_m\}$ have to be applied. This allows one to obtain smaller sets of normal forms.

8 Examples

Due to its totality property (which distinguishes it from all other AC-compatible path orderings) the main field of application of AC-RPO is the one of complete strategies for theorem proving and unfailing Knuth-Bendix completion modulo AC. However, it can also be used to orient AC-rewrite systems. Below we show some examples of proving termination of such rewrite systems.

In the following example, note that our ordering orients the distributivity axiom in the opposite way wrt. the usual orientation:

Example 8.1 Suppose $*, + \in \mathcal{F}_{AC}$ and $* \succ_{\mathcal{F}} +$. Then $x * y + x * z \succ x * (y + z)$, since $snf_R(x * y + x * z) = \{x * y + x * z\}$, $snf_R(x * (y + z)) = \{x * y, x * z\}$ and $x * y + x * z \succ_{rpo} x * y$ and $x * y + x * z \succ_{rpo} x * z$.

Example 8.2 *Rings.* With $+ \in \mathcal{F}_{AC}$ and $* \succ_{\mathcal{F}} I \succ_{\mathcal{F}} + \succ_{\mathcal{F}} 0$, we have the following canonical term rewrite system:

$$\begin{array}{lcl}
0 + x & \rightarrow & x \\
I(x) + x & \rightarrow & 0 \\
I(0) & \rightarrow & 0 \\
I(I(x)) & \rightarrow & x \\
I(x + y) & \rightarrow & I(x) + I(y) \\
x * (y + z) & \rightarrow & (x * y) + (x * z) \\
(x + y) * z & \rightarrow & (x * z) + (y * z) \\
0 * x & \rightarrow & 0 \\
x * 0 & \rightarrow & 0 \\
x * I(y) & \rightarrow & I(x * y) \\
I(x) * y & \rightarrow & I(x * y)
\end{array}$$

Note that here the distributivity axiom can be oriented in the usual way since $*$ is not an AC-symbol.

Example 8.3 *Milners's nondeterministic machines.* With $+ \in \mathcal{F}_{AC}$ and $L \succ_{\mathcal{F}} T \succ_{\mathcal{F}} + \succ_{\mathcal{F}} 0$ we have the following canonical term rewrite system:

$$\begin{array}{lcl}
0 + x & \rightarrow & x \\
x + x & \rightarrow & x \\
T(T(x)) & \rightarrow & T(x) \\
L(T(x)) & \rightarrow & L(x) \\
T(x) + x & \rightarrow & T(x) \\
T(x + y) + x & \rightarrow & T(x + y) \\
T(T(y) + x) & \rightarrow & T(x + y) + T(y) \\
L(T(y) + x) & \rightarrow & L(x + y) + L(y)
\end{array}$$

9 Comparison with other related work and extensions

First let us mention some aspects related to the application of the ordering to the orientation of AC-rewrite systems. As shown in example 8.1, in many cases the orientations obtained by our ordering are different from the usual ones.

An AC-compatible path ordering that does orient the distributivity axiom in the "appropriate" way is the one given in [KSZ90] (this is also the case using polynomial interpretations [BCL87]). But this ordering is not total on ground terms because of its way of comparing terms with the same AC-top-symbol, since sometimes the subterms can be used only once in a comparison. For instance, if $a \succ_{\mathcal{F}} b \succ_{\mathcal{F}} c$ then the terms $f(a,c)$ and $f(b,b)$ are incomparable, and it seems that this problem is inherent to the ordering and difficult to overcome. This problem does not occur in our ordering, since we compare multisets in the usual way (as RPO, for instance) which preserves the totality property.

In the *associative path ordering* (APO) ([BP85]) for many signatures no precedence satisfying the rather severe restrictions exists. In its recent extension EAPO ([DP93]) this problem is solved and for all signatures always some admissible precedence exists (although not all precedences can be used like in our case). APO roughly consists of interpreting the terms by rewriting them with a set of distributivity axioms and then comparing them by RPO. Therefore these orderings are non-total in the sense that terms equivalent under the distributivity laws will be incomparable. A well-known trick to attack this problem is to use a lexicographic combination where the first component is APO and the second component is some other ad-hoc ordering to deal with cases of distributivity-equivalent terms. This can be done since the incomparability relation under APO is a congruence. In fact it is the ACD-congruence relation (where D stands for distributivity). But surprisingly, this is not the case for EAPO, since there may be ACD-equivalent terms with different transformations and therefore comparable under RPO, which makes them comparable under EAPO. Let us give an example:

Example 9.1 Let h, g and f be AC-symbols, with $h \succ_{\mathcal{F}} g \succ_{\mathcal{F}} f$ (note that this precedence is not allowed in APO). Then the rules applied in the transformation are:

$$\begin{array}{rcl} h(f(x,y),z) & \rightarrow & f(h(x,z),h(y,z)) \\ h(g(x,y),z) & \rightarrow & g(h(x,z),h(y,z)) \\ g(f(x,y),z) & \rightarrow & f(g(x,z),g(y,z)) \end{array}$$

Now the following two terms:
$s = f(f(g(h(a,a),h(a,a)),g(h(a,a),h(a,a))),f(g(h(a,a),h(a,a)),g(h(a,a),h(a,a))))$ and
$t = f(g(h(a,a),h(a,a)),g(h(a,a),h(a,a)))$ are ACD-equivalent (both can be obtained from the term $h(g(a,a),f(a,a)))$ and are in normal form wrt. the set of rules). Finally if we compare s and t after flattening by RPO, we have s greater than t. Note that $s \succ_{EAPO} t$ is in fact needed, since t is a proper subterm of s and EAPO is a simplification ordering.

This means that EAPO could only be used as the first component in such a lexicographic combination if the EAPO incomparability relation is indeed a congruence, which, to our knowledge, is an open question.

By a similar method, one can obtain from our ordering an AC-compatible simplification ordering total on ground terms that does orient distributivity in the "right" way: take as first components any desired orderings (like APO) for which the incomparability relation is a congruence, and use our ordering as last component to make the combination total on ground terms. This has been pointed out to us by Claude Marché, who describes and uses this technique in his PhD Thesis [Mar93], where this method is also used in practice for efficiency reasons.

10 Future work

As mentioned in the introduction, a very interesting open question is whether it is possible to decide the satisfiability of AC-ordering constraints, i.e. given $s > t$ with s and t in $\mathcal{T}(\mathcal{F},\mathcal{X})$, (or a quantifier-free boolean formula built over such inequations) to decide whether there exists a ground σ such that $s\sigma \succ t\sigma$ for our ordering. We believe that the ordering defined here should be simple enough to lead to a procedure for deciding such problems, which, as we have shown for the case modulo the empty theory, has interesting applications in automated theorem proving.

References

[Bac92] Leo Bachmair. Associative-commutative reduction orderings. *Information Processing Letters*, 43:21–27, 1992.

[BCL87] Ahlem Ben-Cherifa and Pierre Lescanne. Termination of rewriting systems by polynomial interpretations and its implementation. *Science of Computer Programming*, 9:137–160, 1987.

[BGLS92] Leo Bachmair, Harald Ganzinger, Christopher Lynch, and Wayne Snyder. Basic paramodulation and superposition. In Deepak Kapur, editor, *11th International Conference on Automated Deduction*, LNAI 607, pages 462–476, Saratoga Springs, New York, USA, June 15–18, 1992. Springer-Verlag.

[BP85] L. Bachmair and D. A. Plaisted. Termination orderings for associative-commutative rewriting systems. *Journal of Symbolic Computation*, 1:329–349, 1985.

[Com90] Hubert Comon. Solving symbolic ordering constraints. *International Journal of Foundations of Computer Science*, 1(4):387–411, 1990.

[Der87] Nachum Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3:69–116, 1987.

[DHJP83] N. Dershowitz, J. Hsiang, A. Josephson, and D. Plaisted. Associative-commutative rewriting. In *Int. Joint Conference on Artificial Inteligence*, pages 940–944, 1983.

[DP93] Catherine Delor and Laurence Puel. Extension of the associative path ordering to a chain of associative commutative symbols. In C. Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications*, LNCS 690, pages 389–404, Montreal, Canada, June 16–18, 1993. Springer-Verlag.

[GL86] Isabelle Gnaedig and Pierre Lescanne. Proving termination of associative-commutative rewrite systems by rewriting. In Jörg H. Siekmann, editor, *Proc. 8th International Conference on Automated Deduction*, LNCS 230, pages 52–61, Oxford, England, July 27–August 1, 1986. Springer-Verlag.

[JO91] J-P. Jouannaud and M. Okada. Satisfiability of systems of ordinal notations with the subterm property is decidable. In *Automata, Languages and Programming, 18th International Colloquium*, LNCS 510, Madrid, Spain, July 16–20 1991. Springer-Verlag.

[KKR90] Claude Kirchner, Hélène Kirchner, and Michaël Rusinowitch. Deduction with symbolic constraints. *Revue Française d'Intelligence Artificielle*, 4(3):9–52, 1990.

[KSZ90] D. Kapur, G. Sivakumar, and H. Zhang. A new method for proving termination of ac-rewrite systems. In *Conf. Found. of Software Technology and Theoretical Computer Science*, LNCS 472, pages 134–148, New Delhi, India, December 1990. Springer-Verlag.

[Mar91] Claude Marché. On ground AC-completion. In Ronald V. Book, editor, *Rewriting Techniques and Applications, 4th International Conference*, LNCS 488, pages 411–422, Como, Italy, April 10–12, 1991. Springer-Verlag.

[Mar93] Claude Marché. *Réécriture modulo une théorie présentée par un systéme convergent et décidabilité du problème du mot dans certaines classes de théories équationnelles.* PhD thesis, Univ. de Paris-Sud, Centre d'Orsay, 1993.

[NR91] Paliath Narendran and Michael Rusinowitch. Any ground associative commutative theory has a finite canonical system. In *Fourth int. conf. on Rewriting Techniques and Applications*, LNCS 488, pages 423–434, Como, Italy, April 1991. Springer-Verlag.

[NR92a] Robert Nieuwenhuis and Albert Rubio. Basic superposition is complete. In B. Krieg-Brückner, editor, *European Symposium on Programming*, LNCS 582, pages 371–390, Rennes, France, February 26–28, 1992. Springer-Verlag.

[NR92b] Robert Nieuwenhuis and Albert Rubio. Theorem proving with ordering constrained clauses. In Deepak Kapur, editor, *11th International Conference on Automated Deduction*, LNAI 607, pages 477–491, Saratoga Springs, New York, USA, June 15–18, 1992. Springer-Verlag.

[NR94] Robert Nieuwenhuis and Albert Rubio. AC-Superposition with constraints: No AC-unifiers needed. In Allan Bundy, editor, *12th International Conference on Automated Deduction*, LNAI, Nancy, France, June 1994. Springer-Verlag.

[RN93] Albert Rubio and Robert Nieuwenhuis. A precedence-based total AC-compatible ordering. In C. Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications*, LNCS 690, pages 374–388, Montreal, Canada, June 16–18, 1993. Springer-Verlag.

[Vig94] Laurent Vigneron. Associative Commutative Deduction with constraints. In Allan Bundy, editor, *12th International Conference on Automated Deduction*, LNAI, Nancy, France, June 1994. Springer-Verlag.