Path Orderings, Quasi-orderings and Termination of Term Rewriting Systems

Cristina Borralleras and Albert Rubio

September, 1997

Abstract

In this paper we present some original variations of the recursive path ordering. Additionally we define a restricted *semantic path ordering* which, in general, does not include the subterm relation, but is shown to be monotonic. By combining both kind of orderings we can prove (automatically) the termination of several (non-simply terminating) examples.

1 Introduction

Rewriting systems are sets of rules (directed equations) used to compute by repeatedly replacing parts of a given formula with equal ones until the simplest possible form is obtained. Depending on the kind of objects that are going to be rewritten we have different kinds of rewriting system. For instance we may have *string rewriting system* (Thue or semi-Thue systems) or *term rewriting systems*.

Rewriting systems are useful for many applications. A program can be expressed in the form of term rewriting system. Such programs are easy to understand and have a simple, elegant syntax and semantics. (we can see a rewrite system as a nondeterministic functional program). To verify the correctness of such a program, among other things, one must show that it always terminates, i.e. given any input term, execution will always reach a stage for which there is no way to continue applying rules.

Rewrite systems have also been used as a decision procedure for equational theories, as a simplification method in automated theorem proving or as a formalism to express transformations. For more applications see [DJ90, Klo92].

Termination is a fundamental property for most of applications of rewrite systems. Termination of a rewrite system is, in general, an undecidable property, even if it includes a single rule. Several methods for proving termination of term (and string) rewriting systems, more or less suitable for automation, have been developed: e.g. semantic interpretations ([Itu67, Lan79, BCL87, Zan92, FZ93b, Zan95]), path orderings ([Der82, KL80, Les82, KNS85]), semantic orderings ([KL80, Ges92, DH93]), transformation orderings ([BD86, BL90, FZ93a]). A survey of such methods for proving termination is given in [Der87] (see also [DJ90]).

Orderings on terms are frequently used in termination proofs of term rewriting systems (TRS), i.e. showing that the rewrite relation defined by the system is included in some *well-founded* ordering on terms. Due to its simplicity to be used and automated, the class of path orderings, and in particular the *recursive path ordering* ([Der82]), has received a special attention. This orderings are (in general) *simplification orderings*, i.e. monotonic well-founded orderings including the subterm relation. Unfortunately, this property, that makes them simpler, makes them weaker too, as there are many TRS that are terminating but they cannot be proved terminating by any simplification ordering, i.e. they are not *simply terminating*.

In this report we present a new path ordering called the *argument recursive path ordering* (arpo), which, unlike the recursive path ordering (rpo), compares first the arguments and in case of equality compares then the root symbols.

Although arpo can be used to prove termination directly, it turns out to be specially useful when combined with term interpretations. In order to assure that this interpretations can be used to prove the termination of a TRS, we define a restricted version of the *semantic path ordering* ([KL80]), which is monotonic and does not include, in general, the subterm relation, and therefore it is not a simplification ordering.

We use arpo on interpreted terms as underlying ordering in the restricted semantic path ordering. As this ordering is not a simplification ordering we can deal with several non-simply terminating, but terminating, term rewriting systems. Moreover we present some examples which cannot be proved terminating if we use rpo instead of arpo as underlying ordering.

Finally we combine arpo with rpo, obtaining what we call the *head-argument recursive path ordering* (harpo), which, depending on the pair of root symbols of the terms to be compared, compares first the head symbols and then the arguments recursively (as rpo does) or compares first the arguments and then the head symbols (as arpo does). With this ordering we can get the full power of both kind of orderings.

Formal definitions and basic tools are introduced in section 2. In section 3, the argument recursive path ordering (arpo) is defined and it is proved to be a simplification ordering. In section 4 we present the restricted semantic path ordering which is based on a quasi-monotonic quasi-ordering defined by an interpretation. This ordering is proved to be a reduction ordering (i.e. monotonic and well-founded). Some examples of termination proofs by using this ordering are given in section 5. The ordering harpo is defined and studied in section 6. Some applications of this ordering are presented in section 7. In section 8 we analyze the totality property of the presented orderings. Finally some future work is discussed in section 9.

2 Preliminaries

We present some standard notations and definitions used in this report. Most of the definitions are consistent with [DJ90].

2.1 General definitions.

Given a (finite) set of function symbols $\mathcal{F}$, usually called a signature, and a set of variables symbols $\mathcal{X}$, the set of (first-order) terms $\mathcal{T}(\mathcal{F},\mathcal{X})$ over $\mathcal{F}$ and $\mathcal{X}$ is the smallest set containing $\mathcal{X}$ such that $f(t_1,\ldots,t_n) \in \mathcal{T}(\mathcal{F},\mathcal{X})$ whenever f is an n-ary function symbol in $\mathcal{F}$ and $t_1,\ldots,t_n \in \mathcal{T}(\mathcal{F},\mathcal{X})$.

If function symbols have variable arity, if $f \in \mathcal{F}$ and $t_1,\ldots,t_n \in \mathcal{T}(\mathcal{F},\mathcal{X})$ with $n \geq 0$ then $f(t_1,\ldots,t_n) \in \mathcal{T}(\mathcal{F},\mathcal{X})$.

The set of variable-free terms is denoted by $\mathcal{T}(\mathcal{F})$.

A *rewrite rule* is an ordered pair of terms $l \rightarrow r$.

A (finite or infinite) set of rewrite rules R over $\mathcal{T}(\mathcal{F},\mathcal{X})$ is called a *term rewriting system*, or simply a rewrite system.

For given rewrite system $\mathcal{R}(\mathcal{F},\mathcal{X},R)$, a term $s \in \mathcal{T}(\mathcal{F})$ *rewrites to* a term $t \in \mathcal{T}(\mathcal{F})$, written $s \rightarrow_{\mathcal{R}} t$, if $s|_p = l\sigma$ and $t = s[r\sigma]_p$, for some rule $l \rightarrow r$ in $\mathcal{R}$, position p in s, and substitution σ (from variables in l to terms in $\mathcal{T}(\mathcal{F})$). This is the same as saying that $s = u[l\sigma]_p$ and $t = u[r\sigma]_p$ for some context u and position p in u.

In other words, a rule $l \rightarrow_{\mathcal{R}} r$ applies to a term $t \in \mathcal{T}(\mathcal{F})$ if a subterm s of t matches the left-hand side l with some substitution σ of terms in $\mathcal{T}(\mathcal{F})$ for variables appearing in l (i.e. $s = l\sigma$). The rule is applied by replacing the subterm s in t with the corresponding right-hand side $r\sigma$ of the rule, with the same substitution σ of terms for variables has been made.

In general, the choice of which subterm to apply a rule, as well as the choice of which rule to apply from amongst the applicable rules, is made nondeterministically.

A binary relation $\rightarrow_{\mathcal{R}}$ over a set of terms $\mathcal{T}(\mathcal{F},\mathcal{X})$ is a *rewrite relation* if it is both monotonic and stable under substitutions. That is to say, $\rightarrow_R$ is a rewrite relation if $s \rightarrow_{\mathcal{R}} t$ implies $u[s\sigma]_p \rightarrow u[t\sigma]_p$ for all terms s and t in $\mathcal{T}(\mathcal{F},\mathcal{X})$, contexts u, positions p, and substitutions σ. The inverse relation $\leftarrow$, symmetric closure $\leftrightarrow$, reflexive closure $\xrightarrow{0}_{\mathcal{R}}$, reflexive-transitive closure $\xrightarrow{*}_{\mathcal{R}}$ and transitive closure $\xrightarrow{+}_{\mathcal{R}}$ of any rewrite relation are also rewrite relations.

We say that a term s is *irreducible*, or in normal form wrt. a rewrite system $\mathcal{R}$ if there is no term in $\mathcal{T}(\mathcal{F},\mathcal{X})$ such that $s \rightarrow_{\mathcal{R}} t$.

The term t is a *normal form* of s wrt. $\mathcal{R}$ if $s \xrightarrow{*}_{\mathcal{R}} t$ and t is irreducible.

A rewrite system is convergent if every term s in $\mathcal{T}(\mathcal{F})$ has a unique normal form. When a system is terminating every term has at least one normal form. Following, termination of rewrite systems is defined.

2.2 Termination of rewrite systems.

A rewrite system $\mathcal{R}(\mathcal{F},\mathcal{X},R)$ is terminating for a set of terms $\mathcal{T}(\mathcal{F})$ if there exists no infinite sequence (derivation) of terms t_i in $\mathcal{T}(\mathcal{F})$ such that $t_1 \rightarrow_{\mathcal{R}} t_2 \rightarrow_{\mathcal{R}} \ldots$

A (partial) ordering $\succ$ of a set $\mathcal{T}(\mathcal{F})$ is well-founded if there exists no infinite descending chain $t_1 \succ t_2 \succ \ldots$ of elements of $\mathcal{T}(\mathcal{F})$.

Thus, if $\rightarrow_{\mathcal{R}}$ is contained in some well-founded partial ordering $\succ$ on $\mathcal{T}(\mathcal{F},\mathcal{X})$ then $\mathcal{R}$ is obviously terminating.

Definition 2.1. A (strict partial) ordering $\succ$ is a transitive and irreflexive binary relation.

Definition 2.2. An ordering $\succ$ is monotonic if $s \succ t$ implies $f(\ldots,s,\ldots) \succ f(\ldots,t,\ldots)$, for all terms s and t and function symbols f.

Definition 2.3. An ordering $\succ$ is said to be stable under substitutions if $s \succ t$ implies $s\sigma \succ t\sigma$, for all terms s and t and substitutions σ.

Definition 2.4. An ordering on terms is called *rewrite ordering* if it is monotonic and stable under substitutions.

Thus, a rewrite relation is called a rewrite ordering if it is transitive and irreflexive; and the proof of the termination of a rewrite system consists in proving its transitive closure $\xrightarrow{+}_{\mathcal{R}}$ is a well-founded ordering.

Definition 2.5. A *reduction ordering* is a well-founded rewrite ordering on terms.

Theorem 2.1. *A rewrite system $\mathcal{R}$ is terminating if all rules are contained in a reduction ordering $\succ$, i.e. $l \succ r$ for every $l \rightarrow r \in R$.*

(Conversely, if $\mathcal{R}$ is terminating, the relation $\xrightarrow{+}_{\mathcal{R}}$ is itself a reduction ordering).

Simplification orderings are another class of well-founded orderings. In general, a reduction ordering may be not a simplification ordering, but actually almost all reduction orderings used to show termination of rewrite systems are simplification orderings.

Definition 2.6. An ordering $\succ$ possesses the subterm property if $f(\ldots s \ldots) \succ s$ for all terms s and function symbols f in $\mathcal{F}$.

Definition 2.7. An ordering $\succ$ possesses the deletion property if $f(\ldots s \ldots) \succ f(\ldots\ldots)$ for all terms s and variable arity function symbols f in $\mathcal{F}$.

Definition 2.8. A rewrite ordering is a *simplification ordering* if it enjoys the subterm property and the deletion property (the last one only for variable arity function symbols)

When $\mathcal{F}$ is finite, is proved [Der82] (using Kruskal's theorem [Kru60]) that any simplification ordering on $\mathcal{T}(\mathcal{F},\mathcal{X})$ is well-founded (and hence, it is reduction ordering).

Thus, when $\mathcal{F}$ is finite, simplification orderings can also be used for proving termination of rewrite systems.

Theorem 2.2. *Let $\mathcal{F}$ be finite. A rewrite system $\mathcal{R}(\mathcal{F},\mathcal{X},R)$ is terminating if all rules are contained in a simplification ordering $\succ$, i.e. $l \succ r$ for every $l \rightarrow r \in R$.*

We will make extensive use of multisets. For this reason we will need to define the extension of an equivalence and an ordering on elements to multisets of elements.

Definition 2.9. The extension of an equivalence $=$ to multisets, denoted by $==$, is defined by: $\{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\}$ iff $m = n$ and if $n > 0$ then there exists some t_j s.t. $s_1 = t_j$ and $\{s_1, \ldots, s_m\} \backslash \{s_1\} == \{t_1, \ldots, t_n\} \backslash \{t_j\}$.

Definition 2.10. The extension, wrt. an equivalence $=$, of an ordering $\succ$ to finite multisets, denoted by $\succ\!\succ$, is defined by:

Let $V = \{s_1, \ldots, s_m\}$ and $W = \{t_1, \ldots, t_n\}$, $V = M \cup X \succ\!\succ N \cup Y = W$ iff $X \neq \emptyset$, M==N and for all $y \in Y$ there exists some $x \in X$ s.t. $x \succ y$.

From this definition it follows that for comparing two multisets, once equal elements have been deleted, it suffices to check whether for every element in the second multiset Y, there is a greater element in the first multiset X.

A very common sort of orderings for proving termination of rewrite systems (such that will exploit paths in the tree structure of the terms) are the path orderings. A frequently used path ordering construct directly from a well-founded ordering on the signature, the *precedence*, which is a simplification ordering is the recursive path ordering.

Definition 2.11. The *recursive path ordering* (rpo)

Let $\succ_{\mathcal{F}}$ be a partial ordering on a finite set of function symbols $\mathcal{F}$. The *recursive path ordering* **(rpo)** on the set $\mathcal{T}(\mathcal{F})$ of terms over $\mathcal{F}$, denoted by $\succ_{rpo}$, is defined recursively as follows:

$s = f(s_1, \ldots, s_m) \succ_{rpo} g(t_1, \ldots, t_n) = t$ iff

(i) $s_i \succeq_{arpo} t$, for some $i = 1, \ldots, m$, or

(ii) $f \succ_{\mathcal{F}} g$ and $s \succ_{rpo} t_i$ for all $i = 1 \ldots n$

(iii) f=g and $\{s_1, \ldots, s_m\} \succ\!\succ_{rpo} \{t_1, \ldots, t_n\}$.

where $\succ\!\succ_{rpo}$ is the extension of $\succ_{rpo}$ to multisets.

Since **rpo** does not take into account the position of the subterms in the term, all terms that are equal up to permutation of arguments, e.g. f(a,g(b,a)) and f(g(a,b),a), are incomparable under rpo. For that very reason, $=$ is sometimes interpreted in rpo as equality up to permutation of arguments instead of syntactic equality, and hence $s \succeq_{rpo} t$ means either $s \succ_{rpo} t$ or s and t are equal up to permutation of arguments.

3 The Argument Recursive Path Ordering

Definition 3.1. Let $\succ_{\mathcal{F}}$ be a (partial) ordering on a finite set of function symbols $\mathcal{F}$. The *argument recursive path ordering* **(arpo)**, denoted by $\succ_{arpo}$, on the set $\mathcal{T}(\mathcal{F})$ of terms over $\mathcal{F}$, is defined recursively as follows:

$s = f(s_1, \ldots, s_m) \succ_{arpo} g(t_1, \ldots, t_n) = t$ iff

(i) $s_i \succeq_{arpo} t$, for some $i = 1, \ldots, m$ or

(ii) $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{t_1, \ldots, t_n\}$ or

(iii) $\{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\}$ and $f \succ_{\mathcal{F}} g$,

where $\succ\succ_{arpo}$ is the multiset extension of $\succ_{arpo}$, and $=$ is interpreted as equality up to permutation of arguments. Hence, two terms are considered equal if they are the same except for permutation among subterms.

Lemma 3.1. *The ordering $\succ_{arpo}$ is compatible with the equality up to permutation of arguments, i.e. $s' = s \succ_{arpo} t = t'$ implies $s' \succ_{arpo} t'$ for all $s', s, t', t \in \mathcal{T}(\mathcal{F})$.*

Proof. By induction on the size of the four terms. Let $s = f(s_1, \ldots, s_m)$, $s' = f(s'_1, \ldots, s'_m)$, $t = g(t_1, \ldots, t_n)$ i $t = g(t'_1, \ldots, t'_n)$. We will make a case analysis on $s \succ_{arpo} t$:

- $s \succ_{arpo} t$ by (i): $s_i \succeq_{arpo} t$ for some $i = 1, \ldots, m$. As s'=s, for all s_i there exists s'_j s.t. $s_i = s'_j$. Hence, we have $s'_j = s_i \succeq_{arpo} t = t'$ and by induction hypothesis $s'_j \succeq_{arpo} t'$ that, by (i), gives $s' \succ_{arpo} t'$.
- $s \succ_{arpo} t$ by (ii):
  $\{s'_1, \ldots, s'_m\} == \{s_1, \ldots, s_m\} \succ\succ_{arpo} \{t_1, \ldots, t_n\} == \{t'_1, \ldots, t'_n\}$. By induction hypothesis, the ordering $\succ_{arpo}$ and equality up to permutation of arguments are compatible on the set of subterms. As the multiset extension of a compatible (with equality up to permutation of arguments) relation is also compatible, it follows that $\{s'_1, \ldots, s'_m\} \succ\succ_{arpo} \{t'_1, \ldots, t'_n\}$, and by (ii), $s' \succ_{arpo} t'$.
- $s \succ_{arpo} t$ by (iii):
  $\{s'_1, \ldots, s'_m\} == \{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\} == \{t'_1, \ldots, t'_n\}$ and $f \succ_{\mathcal{F}} g$. By transitivity of $==$, $\{s'_1, \ldots, s'_m\} == \{t'_1, \ldots, t'_n\}$, and with $f \succ_{\mathcal{F}} g$, by (iii), we conclude $s' \succ_{arpo} t'$.

□

Lemma 3.2. *The ordering $\succ_{arpo}$ enjoys the subterm property.*

Proof. $s = f(\ldots, t, \ldots) \succ_{arpo} t$ for all $f \in \mathcal{F}$ and for all $t \in \mathcal{T}(\mathcal{F})$. It is directly proved by definition of arpo. There exists a subterm of s equal to t, so by (i), $s \succ_{arpo} t$, i.e. for all $f \in \mathcal{F}$ and for all $t \in \mathcal{T}(\mathcal{F})$, $f(\ldots, t, \ldots) \succ_{arpo} f(\ldots, \ldots)$. □

Lemma 3.3. *The ordering $\succ_{arpo}$ satisfies the deletion property.*

Proof. By definition of $\succ\succ_{arpo}$, $\{\ldots, t, \ldots\} \succ\succ_{arpo} \{\ldots, \ldots\}$, thus, by (ii), we have $f(\ldots, t, \ldots) \succ_{arpo} f(\ldots, \ldots)$. □

Lemma 3.4. *$\succ_{arpo}$ is transitive.*

Proof. $s \succ_{arpo} t$ and $t \succ_{arpo} u$ implies $s \succ_{arpo} u$ for all $s, t, u \in \mathcal{T}(\mathcal{F})$. By induction on the size of s, t and u. Given $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$ and $u = h(u_1, \ldots, u_r)$.

1. $s \succ_{arpo} t$ by (i): $s_i \succ_{arpo} t$ or $s_i = t$ for some $i = 1, \ldots, m$
   If $s_i = t$, since $t \succ_{arpo} u$, by compatibility of $=$ and $\succ_{arpo}$, $s_i \succ_{arpo} u$ and by (i) $s \succ_{arpo} u$. Otherwise, if $s_i \succ_{arpo} t$, as $t \succ_{arpo} u$, by induction hypothesis $s_i \succ_{arpo} u$ and by (i) $s \succ_{arpo} u$.

2. $s \succ_{arpo} t$ by (ii) or by (iii) and $t \succ_{arpo} u$ by (i). Hence, $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{t_1, \ldots, t_n\}$ or $\{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\}$, which, by definition of the multiset extension, implies that for all $j \in \{1, \ldots, n\}$ there exists $i \in \{1, \ldots, m\}$ s.t. $s_i \succeq_{arpo} t_j$. Besides, $t \succ_{arpo} u$ by (i) implies that there exists $j \in \{1, \ldots, n\}$ s.t. $t_j \succeq_{arpo} u$. Therefore, $s_i \succeq_{arpo} t_j \succeq_{arpo} u$ and by induction hypothesis and by compatibility of $\succ_{arpo}$ and $=$ we obtain $s_i \succeq_{arpo} u$ for some $i \in \{1, \ldots, m\}$. So, by (i), we have $s \succ_{arpo} u$.

3. $s \succ_{arpo} t$ and $t \succ_{arpo} u$ by (ii) or by (iii). We consider the four possible cases:

   - $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{t_1, \ldots, t_n\} \succ\succ_{arpo} \{u_1, \ldots, u_r\}$.
     By induction hypothesis, the relation $\succ_{arpo}$ on the set $\mathcal{S}$ of subterms s_i, t_j and u_k is transitive, i.e. for all $v_i, v_j, v_k \in S$ $v_i \succ_{arpo} v_j \succ_{arpo} v_k \Rightarrow v_i \succ_{arpo} v_k$ (the size of v_i, v_j and v_k is smaller than the size of s,t and u). Since the extension of a transitive relation to multiset is also transitive, if follows that $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{u_1, \ldots, u_r\}$, and by (ii), we conclude $s \succ_{arpo} u$.
   - $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{t_1, \ldots, t_n\} == \{u_1, \ldots, u_r\}$ and $g \succ_{\mathcal{F}} h$. The compatibility of $\succ_{arpo}$ and $=$ on the set of the subterms implies $\succ\succ_{arpo}$ and $==$ are compatibles. So, we have $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{u_1, \ldots, u_r\}$, and using (ii) we get $s \succ_{arpo} u$.
   - $\{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\} \succ\succ_{arpo} \{u_1, \ldots, u_r\}$ and $f \succ_{\mathcal{F}} g$. Is proved by the same argument of the previous case.
   - $\{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\} == \{u_1, \ldots, u_r\}$ and $f \succ_{\mathcal{F}} g \succ_{\mathcal{F}} h$. By transitivity of both, $==$ and $\succ_{\mathcal{F}}$, we have $\{s_1, \ldots, s_m\} == \{u_1, \ldots, u_r\}$ and $f \succ_{\mathcal{F}} h$, and using (iii) we get $s \succ_{arpo} u$.

□

Lemma 3.5. *$\succ_{arpo}$ is irreflexive.*

Proof. $t \not\succ_{arpo} t$ for all term t in $\mathcal{T}(\mathcal{F})$. By induction on the size of $t = f(t_1, \ldots, t_n)$. We show that it is not possible, in any case of the definition of $\succ_{arpo}$, to have $t \succ_{arpo} t$.

- Suppose $t \succ_{arpo} t$ by (i): then there is some i in $\{1, \ldots, n\}$ with $t_i \succ_{arpo} f(\ldots, t_i, \ldots)$. By the subterm property we have $f(\ldots, t_i, \ldots) \succ_{arpo} t_i$ so, by transitivity, $t_i \succ_{arpo} t_i$ which contradicts the induction hypothesis.

- Suppose $t \succ_{arpo} t$ by (ii): then we have $\{t_1, \ldots, t_n\} \succ\succ_{arpo} \{t_1, \ldots, t_n\}$. By induction hypothesis, $\succ_{arpo}$ is irreflexive for its subterms, and, since the multiset extension of an irreflexive relation is also irreflexive, $\{t_1, \ldots, t_n\} \not\succ\succ_{arpo} \{t_1, \ldots, t_n\}$, which leads to a contradiction.
- Suppose $t \succ_{arpo} t$ by (ii): then we have $f \succ_{\mathcal{F}} f$, which contradicts the irreflexivity of $\succ_{\mathcal{F}}$.

□

Lemma 3.6. *$\succ_{arpo}$ is monotonic.*

Proof. $s \succ_{arpo} t$ implies $f(\ldots, s, \ldots) \succ_{arpo} f(\ldots, t, \ldots)$ for all $s, t \in \mathcal{T}(\mathcal{F})$ and for all $f \in \mathcal{F}$.

If $s \succ_{arpo} t$, by definition of $\succ\succ_{arpo}$, $\{\ldots, s, \ldots\} \succ\succ_{arpo} \{\ldots, t, \ldots\}$, and by (ii) $f(\ldots, s, \ldots) \succ_{arpo} f(\ldots, t, \ldots)$. □

Lemma 3.7. *$\succ_{arpo}$ is stable under substitutions.*

Proof. Let $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, then $s \succ_{arpo} t \Rightarrow s\sigma \succ_{arpo} t\sigma$ for all substitution σ (of variables in $\mathcal{X}$ by terms in $\mathcal{T}(\mathcal{F}, \mathcal{X})$). We will prove it by induction on the size of s and t.

Since every variable is incomparable with any other symbol with $\succ_{\mathcal{F}}$ not in $\mathcal{F}$. Variables only can be compared by $=$. Hence, if $s \succ_{arpo} t$ then s cannot be a variable. Assume $s = f(s_1, \ldots, s_n)$. We will make a case analysis on the proof of $s \succ_{arpo} t$.

1. $s \succ_{arpo} t$ by (i), i.e. $s_i \succeq_{arpo} t$ for some $i \in \{1, \ldots, n\}$. By induction hypothesis, $s_i\sigma \succeq_{arpo} t\sigma$. Since $s\sigma = f(s_1, \ldots, s_n)\sigma = f(s_1\sigma, \ldots, s_n\sigma)$, by (i) we have $s\sigma \succ t\sigma$.
2. $s = f(s_1, \ldots, s_m) \succ_{arpo} g(t_1, \ldots, t_n) = t$ by (ii), i.e. $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{t_1, \ldots, t_n\}$. By induction hypothesis $\succ_{arpo}$ is stable under substitutions on the set of subterms and as the multiset extension of an stable under substitutions relation is also stable, $\{s_1, \ldots, s_m\} \succ\succ_{arpo} \{t_1, \ldots, t_n\}$ implies $\{s_1, \ldots, s_m\}\sigma \succ\succ_{arpo} \{t_1, \ldots, t_n\}\sigma$, i.e. $\{s_1\sigma, \ldots, s_m\sigma\} \succ\succ_{arpo} \{t_1\sigma, \ldots, t_m\sigma\}$, that by (ii) gives $f(s_1\sigma, \ldots, s_m\sigma) \succ_{arpo} g(t_1\sigma, \ldots, t_n\sigma)$, i.e. $f(s_1, \ldots, s_m)\sigma \succ_{arpo} g(t_1, \ldots, t_n)\sigma$. So, $s\sigma \succ_{arpo} t\sigma$ holds.
3. $s = f(s_1, \ldots, s_m) \succ_{arpo} g(t_1, \ldots, t_n) = t$ by (iii), i.e. $\{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\}$ and $f \succ_{\mathcal{F}} g$. Thus, $\{s_1\sigma, \ldots, s_m\sigma\} == \{t_1\sigma, \ldots, t_m\sigma\}$, which, by (iii), implies $f(s_1\sigma, \ldots, s_m\sigma) \succ_{arpo} g(t_1\sigma, \ldots, t_m\sigma)$, and hence $s\sigma \succ_{arpo} t\sigma$.

□

Theorem 3.8. *The argument recursive path ordering $\succ_{arpo}$, is a simplification ordering.*

Proof. All the properties have been proved in lemmas 3.2 to 3.7. □

4 Quasi-orderings and Termination

We begin giving some definitions about quasi-orderings that will be used in the following sections.

Definition 4.1. A quasi-ordering $\succeq$ is a transitive and reflexive binary relation.

Definition 4.2. The equivalence relation $\sim$ induced by the quasi-ordering $\succeq$ is defined as both $\succeq$ and $\preceq$.

Definition 4.3. The strict part $\succ$ of a quasi-ordering $\succeq$ is defined as $\succeq$ but not $\preceq$.

Definition 4.4. A quasi-ordering $\succeq$ is quasi-monotonic if

$$s \succeq t \quad implies \quad f(\ldots, s, \ldots) \succeq f(\ldots, t, \ldots)$$

for all terms s and t and function symbols f.

We remark that $s \sim t$ implies $f(\ldots, s, \ldots) \sim f(\ldots, t, \ldots)$ but $s \succ t$ only implies $f(\ldots, s, \ldots) \succeq f(\ldots, t, \ldots)$ instead of $f(\ldots, s, \ldots) \succ f(\ldots, t, \ldots)$, i.e. $\sim$ is a congruence, but $\succ$ may be non-monotonic. We say that a quasi-ordering $\succeq$ is monotonic if its strict part is monotonic.

Definition 4.5. A quasi-ordering $\succeq$ is said to be stable under substitutions if both the strict part $\succ$ and the equivalence relation $\sim$ induced by $\succeq$ are stable under substitutions.

Definition 4.6. A quasi-ordering is well-founded if its strict part is a well-founded ordering.

We will consider the interpretation functions $I : \mathcal{T}(\mathcal{F}, \mathcal{X}) \to \mathcal{T}(\mathcal{F}', \mathcal{X}')$ defined as:

- For variables, I is a bijection.
- For any term $f(t_1, \ldots, t_n)$, I can be
  - $I(t_i)$ for some $i \in \{1, \ldots, n\}$, or
  - $f_I(I(t_{i_1}), \ldots, I(t_{i_k}))$, for some $k \geq 0$ and $i_1, \ldots, i_k \in \{1, \ldots, n\}$.

  where i and $i_1, \ldots, i_k$ are fixed for any given symbol $f \in \mathcal{F}$. This means that I only depends on the root symbol.

This means that the interpretation either eliminates and/or repeats one or more arguments, or selects an argument. For instance, $I(f(x, y, z, t)) = f_I(y, y, t)$, $I(g(x, y)) = g_I$ or $I(h(x, y, z)) = x$.

Definition 4.7. $\succeq_I$ is defined as:

$$s \succeq_I t \quad \Leftrightarrow \quad I(s) \succeq I(t)$$

It is well-known that this definition implies:

$$s \succ_I t \quad \Leftrightarrow \quad I(s) \succ I(t) \quad and$$

$$s \sim_I t \quad \Leftrightarrow \quad I(s) \sim I(t)$$

The following two properties are well-known to be true for any interpretation.

Proposition 4.1. *If $\succeq$ is a quasi-ordering then $\succeq_I$ is a quasi-ordering.*

Proposition 4.2. *If $\succeq$ is well-founded then $\succeq_I$ is well-founded.*

For the following proposition we will use the restrictions on the interpretations.

Proposition 4.3. *If $\succeq$ is quasi-monotonic then $\succeq_I$ is quasi-monotonic.*

Proof. We have to prove that $s \succeq_I t$ implies $f(\ldots s \ldots) \succeq_I f(\ldots t \ldots)$ for all $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and for all $f \in \mathcal{F}$. By definition of $\succeq_I$, this is equivalent to prove $I(s) \succeq I(t)$ implies $I(f(\ldots, s, \ldots)) \succeq I(f(\ldots, t, \ldots))$ for all $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and for all $f \in \mathcal{F}$.

We proceed by analyzing the two possible kind of interpretation we may have for I (note that the interpretation depends only on the root symbol.

- $I(f(u_1, \ldots, u_n)) = I(u_i)$ for some $i \in \{1, \ldots, n\}$
  1. Let s be the subterm in the position i. Then $I(f(\ldots, s, \ldots)) = I(s)$ and $I(f(\ldots, t, \ldots)) = I(t)$.
     Thus, $I(s) \succeq I(t)$ implies $I(f(\ldots, s, \ldots)) \succeq I(f(\ldots, t, \ldots))$.
  2. Let s be the subterm in a position other than i, then $I(f(\ldots, s, \ldots))$ and $I(f(\ldots, t, \ldots))$ are equal to $I(v_i)$ for some v_i. Thus, by reflexivity of $\succeq$, $I(f(\ldots, s, \ldots)) \succeq I(f(\ldots, t, \ldots))$.
- $I(f(u_1, \ldots, u_n)) = f_I(I(u_{i_1}), \ldots, I(u_{i_k}))$ for some $k \geq 0$ and $i_1, \ldots, i_k \in \{1, \ldots, n\}$.
  1. Let p be the position of the subterm s in $f(\ldots, s, \ldots)$. If some of the $i_1, \ldots, i_k$ are p then $I(f(\ldots, s, \ldots)) = f_I(\ldots, I(s), \ldots, I(s), \ldots)$ and $I(f(\ldots, t, \ldots)) = f_I(\ldots, I(t), \ldots, I(t), \ldots)$. Note that, for simplicity, we have written it $I(s)$ and $I(t)$ twice, although they can occur one n times.

     If $I(s) \succeq I(t)$ then by the quasi-monotonicity of $\succ$ (applied n times), we have $f_I(\ldots, I(s), \ldots, I(s), \ldots) \succeq f_I(\ldots, I(t), \ldots, I(s), \ldots) \succeq \ldots \succeq f_I(\ldots, I(t), \ldots, I(t), \ldots)$ and, by the transitivity of $\succeq$, we obtain $f_I(\ldots, I(s), \ldots, I(s), \ldots) \succeq f_I(\ldots, I(t), \ldots, I(t), \ldots)$, and hence $I(f(\ldots, s, \ldots)) \succeq I(f(\ldots, t, \ldots))$.
  2. If the position of s in $f(\ldots, s, \ldots)$ is none of the i_j then, as in case 2 above, we have the same interpretation for both terms and hence it holds.

□

Definition 4.8. The interpretation of a substitution σ from variables of $\mathcal{X}$ to terms of $\mathcal{T}(\mathcal{F},\mathcal{X})$, denoted by $I(\sigma)$, is defined as $I(\sigma) = \{I(x_i) \mapsto I(x_i\sigma) \mid x_i \in Dom(\sigma)\}$.

Note that we need I to be a bijection on variables, since otherwise $I(\sigma))$ may not be a correct substitution.

Proposition 4.4. *$I(s\sigma) = I(s)I(\sigma)$ for all term $s \in \mathcal{T}(\mathcal{F},\mathcal{X})$ and for every substitution σ of terms in $\mathcal{T}(\mathcal{F},\mathcal{X})$ for the variables of s.*

Proof. By induction on the size of s. We distinguish two cases depending on whether s is a variable or a term $f(s_1,\ldots,s_n)$:

- s=x for some $x \in \mathcal{X}$. By definition of $I(\sigma)$, $I(x)I(\sigma) = I(x)\{I(x_i) \mapsto I(x_i\sigma) \mid x_i \in Dom(\sigma)\}$. Then if $x \in Dom(\sigma)$ we have $I(x)\{I(x) \mapsto I(x\sigma)\} = I(x\sigma)$, and otherwise, if $x \notin Dom(\sigma)$ then $I(x)I(\sigma) = I(x) = I(x\sigma)$.
- $s = f(s_1,\ldots,s_n) \in \mathcal{T}(\mathcal{F},\mathcal{X})$.
  We have $I(f(s_1,\ldots,s_n)\sigma) = I(f(s_1\sigma,\ldots,s_n\sigma))$.
  We proceed by analyzing the two possible kind of interpretations:
  - $I(f(s_1,\ldots,s_n)\sigma) = f_I(I(s_{i_1}\sigma),\ldots,I(s_{i_k}\sigma))$ where $s_{i_j} \in \{s_1\ldots s_n\}$. By induction hypothesis, it is equal to $f_I(I(s_{i_1})I(\sigma),\ldots,I(s_{i_k})I(\sigma))$ $= f_I(I(s_{i_1}),\ldots,I(s_{i_k}))I(\sigma)$, which is equal by definition to $I(f(s_1,\ldots,s_n))I(\sigma)$.
  - $I(f(s_1,\ldots,s_n)\sigma) = I(s_i\sigma)$. By induction hypothesis is equal to $I(s_i)I(\sigma)$ and by definition it is equal to $I(f(s_1,\ldots,s_n))I(\sigma)$.

□

Proposition 4.5. *If $\succ$ is stable under substitutions then $\succ_I$ is stable under substitutions.*

Proof. Suppose that $s \succ_I t$ with $s,t \in \mathcal{T}(\mathcal{F},\mathcal{X})$, then we have to show that $s\sigma \succ_I t\sigma$ for every substitution σ from variables of s,t to terms of $\mathcal{T}(\mathcal{F},\mathcal{X})$.

By definition of $\succ_I$, $s\sigma \succ_I t\sigma$ iff $I(s\sigma) \succ I(t\sigma)$ that by proposition 4.4 it is $I(s)I(\sigma) \succ I(t)I(\sigma)$. Now as $s \succ_I t$ iff $I(s) \succ I(t)$, by stability under substitutions of $\succ$ we can conclude $I(s)I(\sigma) \succ I(t)I(\sigma)$. □

Proposition 4.6. *If $\sim$ is stable under substitutions then $\sim_I$ is stable under substitutions.*

Proof. Suppose that $s \sim_I t$ with $s,t \in \mathcal{T}(\mathcal{F},\mathcal{X})$, then we have to show that $s\sigma \sim_I t\sigma$ for every substitution σ from variables of s,t to terms of $\mathcal{T}(\mathcal{F},\mathcal{X})$.

By definition of $\sim_I$, $s\sigma \sim_I t\sigma$ iff $I(s\sigma) \sim I(t\sigma)$ that by proposition 4.4 it is $I(s)I(\sigma) \sim I(t)I(\sigma)$. Now as $s \sim_I t$ iff $I(s) \sim I(t)$, by stability under substitutions of $\sim$ we can conclude $I(s)I(\sigma) \sim I(t)I(\sigma)$. □

Using the quasi-ordering $\succeq_I$ we will define now a restricted version of the *semantic path ordering* ([KL80]), denoted as $\succ_e$:

Definition 4.9. The semantic path ordering $\succ_e$, is defined as:
$s = f(s_1, \ldots, s_m) \succ_e g(t_1, \ldots, t_n) = t$ iff

1. $s_i \succeq_e t$, for some $i = 1, \ldots, m$,
2. $s \succ_I t$ *and* $s \succ_e t_i$ for all $i = 1, \ldots, n$,
3. $s \sim_I t$ and $\{s_1, \ldots, s_m\} \succ\!\succ_e \{t_1, \ldots, t_n\}$,

where $\succ\!\succ_e$ is the multiset extension induced by $\succ_e$.

The following two propositions are well-known to be true.

Proposition 4.7. *If $\succeq_I$ is well-founded then $\succ_e$ is well-founded.*

Proposition 4.8. *If $\succeq_I$ is stable under substitutions then $\succ_e$ is stable under substitutions.*

But, the semantic path ordering, is in general not monotonic, even when $\succeq_I$ is monotonic. For instance, let $f, g \in \mathcal{F}$ and consider the following interpretation:

$$I(f(x)) = f(x) \quad and \quad I(g(x)) = g$$

and any monotonic ordering fulfilling $f(a) \succ g$. Given the two terms $g(f(a))$ and $f(a)$, we know by (1) that $g(f(a)) \succ_e f(a)$. However if we add the context $f([])$ onto s and t, we have $f(g(f(a))) \not\succ_e f(f(a))$ (in fact, we can prove $f(f(a)) \succ_e f(g(f(a)))$, since both $f(f(a)) \succ_I f(g(f(a)))$ and $f(f(a)) \succ_e g(f(a))$ are satisfied).

Analyzing this example we can observe that the non-monotonicity problem appears when the terms are compared in the first step by (1). More precisely, we can prove that if the terms are compared in the first step by (2) or by(3), the quasi-monotonicity of $\succeq_I$ implies the monotonicity of $\succ_e$.
In order to do this, we define $\succ_r$ as the restricted part of $\succ_e$ which always compares the terms in the first step by (2) or by (3).

Definition 4.10. $\succ_r$ is defined as:
$s = f(s_1, \ldots, s_m) \succ_r g(t_1, \ldots, t_n) = t$ iff

1. $s \succ_I t$ *and* $s \succ_e t_i$ for all $i = 1, \ldots, n$.
2. $s \sim_I t$ and $\{s_1, \ldots, s_m\} \succ\!\succ_e \{t_1, \ldots, t_n\}$

The following property trivially holds from the definition of $\succ_r$.

Proposition 4.9. $\succ_r \subseteq \succ_e$.

And this lemma trivially follows from the previous property.

Lemma 4.10. *If $\succ_e$ is well-founded then $\succ_r$ is well-founded.*

Lemma 4.11. *If $\succeq_I$ is quasi-monotonic then $\succ_r$ is monotonic.*

Proof. Suppose that $s \succ_r t$ with $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, then we have to show that $f(\ldots, s, \ldots) \succ_r f(\ldots, t, \ldots)$ for all $f \in \mathcal{F}$.

If $s \succ_r t$, by definition of $\succ_r$ we have $s \succeq_I t$, and, by the quasi-monotonicity of $\succeq_I$, $f(u_1, \ldots, s, \ldots, u_k) \succeq_I f(u_1, \ldots, t, \ldots, u_k)$.

If $f(u_1 \ldots, s, \ldots u_k) \succ_I f(u_1 \ldots, t, \ldots u_k)$, since $f(u_1 \ldots, s, \ldots u_k) \succ_I u_i$ and $f(u_1 \ldots, s, \ldots u_k) \succ_I t$ by case (1) of the definition of $\succ_e$ (note that $\succ_r \subseteq \succ_e$), we have $f(u_1 \ldots, s, \ldots u_k) \succ_e f(u_1 \ldots, t, \ldots u_k)$.

If $f(u_1 \ldots, s, \ldots u_k) \sim_I f(u_1 \ldots, t, \ldots u_k)$, since $\succ_r \subseteq \succ_e$, by definition of $\succ\!\succ_e$ $\{u_1 \ldots s \ldots u_k\} \succ\!\succ_e \{u_1 \ldots t \ldots u_k\}$. □

Lemma 4.12. *If $\succeq_I$ is stable under substitutions then $\succ_r$ is stable under substitutions.*

Proof. We have to prove that for all terms s and t, if $s \succ_r t$ then $s\sigma \succ_r t\sigma$ for every substitution σ.

By proposition 4.8 $\succ_e$ is stable under substitutions as $\succeq_I$ is. Therefore, $s \succ_I t$ and $s \succ_e t_i$ implies $s\sigma \succ_I t\sigma$ and $s\sigma \succ_e t_i\sigma$. Furthermore, the stability under substitutions of $\sim_I$ gives that $s \sim_I t$ implies $s\sigma \sim_I t\sigma$ and, since the multiset extension of a stable under substitutions relation is also stable under substitutions, $\{s_1, \ldots, s_m\} \succ\!\succ_e \{t_1, \ldots, t_n\}$ implies $\{s_1, \ldots, s_m\}\sigma \succ\!\succ_e \{t_1, \ldots, t_n\}\sigma$.

Hence, by definition of $\succ_r$, we have $s\sigma \succ_r t\sigma$. □

Theorem 4.13. *The ordering $\succ_r$ is a reduction ordering.*

The definition of $\succ_r$ is parameterized by $\succeq_I$. Hence to obtain an actual $\succ_r$ we have to provide an actual $\succeq_I$, which will be given by a term interpretation I (fulfilling the required properties) and a well-founded quasi-monotonic and stable under substitutions quasi-ordering $\succeq$. In what follows we will treat several examples, providing the interpretation and using $\succeq_{arpo}$, defined as $\succ_{arpo} \cup =$, as quasi-ordering.

Note that, since $\succ$ is compatible with $=$, $\succeq$ is a quasi-ordering whose strict part and induced equivalence coincide with $\succ$ and $=$ respectively.

Example 4.14. *The following TRS, originally from Geerling [Gee91],*

$$f(s(x), y, y) \to g(y, x, s(x))$$

cannot be proved terminating by using neither the recursive path ordering nor the argument recursive path ordering. It is easily proved terminating by using $\succ_r$ defined from $\succ_{arpo}$ ($\succ_{rpo}$ is also possible), and the following interpretation function I: $I(f(x, y, z)) = f(I(x), I(y))$ and $I(g(x, y, z)) = g(I(x), I(y))$.

We need to show $f(s(x), y, y) \succ_r g(y, x, s(x))$, that by (1) is the same as:

- *I(f(s(x),y,y)) $\succ_{arpo}$ I(g(y,x,s(x))): applying the definition of I, is the same as f(s(x),y) $\succ_{arpo}$ g(y,x), that is true by (ii).*

- $f(s(x),y,y) \succ_e y$: *by (1)*
- $f(s(x),y,y) \succ_e x$: *by (1)*
- $f(s(x),y,y) \succ_e s(x)$: *by (1)*

From now on, for simplicity reasons, when defining the interpretation we are going to use, we will write $I(f(x,y,z)) = f(x,y)$ instead of $I(f(x,y,z)) = f(I(x), I(y))$, as we assume that the interpretation is of a term is computed innermost.

5 Applications of arpo with interpretations

In this section we will prove the termination of two rewrite systems by using $\succ_r$ defined from $\succeq_{arpo}$. In both cases we will able to see that the termination cannot be proved by defining $\succ_r$ from $\succeq_{rpo}$ instead of $\succeq_{arpo}$.

Actually, we will use the case (1) of the definition of $\succ_r$, so for every rule $l \to r$ in R we will prove $l \succ_I r \quad and \quad l \succ_e r_i$ for all r_i subterm of r.

Example 5.1. *This TRS for remainder, taken from [AG97], is not simply terminating.*

$$le(0,y) \to true$$

$$le(s(x),0) \to false$$

$$le(s(x),s(y)) \to le(x,y)$$

$$minus(x,0) \to x$$

$$minus(s(x),s(y)) \to minus(x,y)$$

$$mod(0,y) \to 0$$

$$mod(s(x),0) \to 0$$

$$mod(s(x),s(y)) \to if(le(y,x),s(x),s(y))$$

$$if(true,s(x),s(y)) \to mod(minus(x,y),s(y))$$

$$if(false,s(x),s(y)) \to s(x)$$

Termination of this TRS can be proved with $\succ_r$ taking the following term interpretation function:

$$I(minus(x,y)) = minus(x)$$

$$I(if(x,y,z)) = if(y,z),$$

and for the rest of the symbols (le, mod, s, true, false, 0), the identity function.

We take as ordering on the signature, $\succ_{\mathcal{F}}$ s.t. $mod \succ_{\mathcal{F}} if$ and $s \succ_{\mathcal{F}} minus$.

- *$l \succ_I r$ for every rule $l \to r$. Thus, we show that $I(l) \succ_{arpo} I(r)$:*

- $le(0,y) \succ_{arpo} true$: by (ii)
- $le(s(x),0) \succ_{arpo} false$: by (ii)
- $le(s(x),s(y)) \succ_{arpo} le(x,y)$: by (ii)
- $minus(x) \succ_{arpo} x$: by (i)
- $minus(s(x)) \succ_{arpo} minus(x)$: by (ii)
- $mod(0,y) \succ_{arpo} 0$: by (i)
- $mod(s(x),0) \succ_{arpo} 0$: by (i)
- $mod(s(x),s(y)) \succ_{arpo} if(s(x),s(y))$: by (iii) since $mod \succ_{\mathcal{F}} if$
- $if(s(x),s(y)) \succ_{arpo} mod(m(x),s(y))$: by (ii) since $s \succ_{\mathcal{F}} m$
- $if(s(x),s(y)) \succ_{arpo} s(x)$: by (i)

(note that this TRS cannot be proved terminating by using rpo instead of arpo, since both, $if \succ_{\mathcal{F}} mod$ and $mod \succ_{\mathcal{F}} if$, should be true).

- *and for every rule $l \to r$, $l \succ_e r_i$ for all r_i subterm of r.*
  - $le(s(x),s(y)) \succ_e x$: by (1)
  - $le(s(x),s(y)) \succ_e y$: by (1)
  - $minus(s(x),s(y)) \succ_e x$: by (1)
  - $minus(s(x),s(y)) \succ_e y$: by (1)
  - $mod(s(x),s(y)) \succ_e le(y,x)$: applying (2), $mod(s(x),s(y)) \succ_I le(y,x)$ that is proved by (ii), and $mod(s(x),s(y)) \succ_e x$ and $mod(s(x),s(y)) \succ_e y$, both of them satisfied by (1).
  - $mod(s(x),s(y)) \succ_e s(x)$: by (1)
  - $mod(s(x),s(y)) \succ_e s(y)$: by (1)
  - $if(true,s(x),s(y)) \succ_e mod(minus(x,y),s(y))$: applying (2),
    $if(s(x),s(y)) \succ_I mod(minus(x),s(y))$
    satisfied by (ii) since $s \succ_{\mathcal{F}} minus$
    and $if(true,s(x),s(y)) \succ_e minus(x,y)$ easily proved by (2)
    and $if(true,s(x),s(y)) \succ_e s(y)$ by (1).
  - $if(false,s(x),s(y)) \succ_e x$: by (1)

Example 5.2. *The following TRS eliminates even numbers from a list. This can be proved terminating by definition of $\succ_r$ from $\succ_{arpo}$ and the interpretation function I, $I(if(x,y) = if(y))$, (identity function for the others)*

$$ev(0) \to true$$

$$ev(s(0)) \to false$$

$$ev(s(s(x))) \to ev(x)$$
$$rmev(nil) \to nil$$
$$rmev(add(x,l)) \to if(ev(x), add(x,l))$$
$$if(true, add(x,l)) \to rmev(l)$$
$$if(false, add(x,l)) \to add(x, rmev(l))$$

Is proved taking $rmev \succ_{\mathcal{F}} if$.

Example 5.3. *The following TRS for the computation of the gcd is not simply terminating.*

$$le(0,y) \to true$$
$$le(s(x),0) \to false$$
$$le(s(x),s(y)) \to le(x,y)$$
$$minus(x,0) \to x$$
$$minus(s(x),s(y)) \to minus(x,y)$$
$$gcd(0,y) \to 0$$
$$gcd(s(x),0) \to 0$$
$$gcd(s(x),s(y)) \to if(le(y,x), s(x), s(y))$$
$$if(true, s(x), s(y)) \to gcd(minus(x,y), s(y))$$
$$if(false, s(x), s(y)) \to gcd(minus(y,x), s(y))$$

We can prove it is terminating by using $\succ_r$ *defined from* $\succ_{arpo}$*, the following interpretation function*

$$I(minus(x,y)) = minus(x)$$
$$I(if(x,y,z)) = if(y,z)$$
$$I(gcd(x,y)) = gcd(x,x,y)$$

(the identity function for the rest of the symbols)
and an ordering on the signature satisfying $s \succ_{\mathcal{F}} minus$.

Note that another possibility could be the following interpretation

$$I(minus(x,y)) = minus(x)$$
$$I(if(x,y,z)) = if(y,z)$$

but in this case we would need $s \succ_{\mathcal{F}} minus$ *and* $gcd \succ_{\mathcal{F}} if$.

6 Head-Argument Recursive Path Ordering

The ordering $\succ_{harpo}$ is defined by combining the recursive path ordering, $\succ_{rpo}$, and the argument recursive path ordering, $\succ_{arpo}$, in a way that for every pair of symbols in the signature, it should be fixed whether terms headed by these symbols are first compared by the precedence as rpo does, or by the arguments as arpo does.

To this end we will use a function *"ha"* which for every pair of symbols associates an H if we will compare by head first or A otherwise. In order to prove the transitivity of the ordering we need to impose some restrictions on *"ha"*.

Definition 6.1. A function ha defined as $ha : \mathcal{F} \times \mathcal{F} \to \{A, H\}$ is *compatible with the ordering* $\succ_{\mathcal{F}}$ on $\mathcal{F}$ if

$$\text{for all } f, g \in \mathcal{F},\ ha(f,g) = ha(g,f) = A \text{ or } ha(f,g) = ha(g,f) = H$$

and satisfies the following restrictions:
Let $f, g, h \in \mathcal{F}$, $f \neq g$ and $g \neq h$ and $f \neq h$:

- (r1): if $ha(f,g) = A$ and $ha(g,h) = A$ then $ha(f,h) = A$.
  Note, by commutativity of ha, it follows that if $ha(f,g) = A$ and $ha(g,h) = H$ then $ha(f,h) = H$.
- (r2): Given $ha(f,g) = A$, $ha(f,h) = H$ and $ha(g,h) = H$,
  - $f \succ_{\mathcal{F}} h \Rightarrow g \succ_{\mathcal{F}} h$
  - $h \succ_{\mathcal{F}} f \Rightarrow h \succ_{\mathcal{F}} g$

Definition 6.2. Let $\succ_{\mathcal{F}}$ be a partial ordering on the finite set of function symbols $\mathcal{F}$, the *head-argument recursive path ordering* **(harpo)** on the set $\mathcal{T}(\mathcal{F})$ of terms over $\mathcal{F}$, denoted by $\succ_{harpo}$, is defined as follows:
$s = f(s_1, \ldots, s_m) \succ_{harpo} g(t_1, \ldots, t_n) = t$ iff

(i) $s_i \succeq_{harpo} t$ for some $i = 1, \ldots, m$,

(ii) If $ha(f,g) = A$:

(a) $\{s_1, \ldots, s_m\} \succ\succ_{harpo} \{t_1, \ldots, t_n\}$ or

(b) $\{s_1, \ldots, s_m\} == \{t_1, \ldots, t_n\}$ and $f \succ_{\mathcal{F}} g$.

(iii) If $ha(f,g) = H$:

(a) $f \succ_{\mathcal{F}} g$ and $s \succ_{harpo} t_i$ for all $i = 1 \ldots n$ or

(b) f=g and $\{s_1, \ldots, s_m\} \succ\succ_{harpo} \{t_1, \ldots, t_n\}$.

where $ha : \mathcal{F} \times \mathcal{F} \to \{A, H\}$ is compatible with $\succ_{\mathcal{F}}$, $=$ is interpreted as equality up to permutation of arguments, and $\succeq_{harpo}$ is $\succ_{harpo} \cup =$.

Lemma 6.1. *The ordering $\succ_{harpo}$ is compatible with equality up to permutation of arguments. That is, for all $s', s, t', t \in \mathcal{T}(\mathcal{F})$, $s' = s \succ_{harpo} t = t' \Rightarrow s' \succ_{harpo} t'$.*

Proof. By induction on the size of the four terms. Let $s = f(s_1, \dots, s_m)$, $s' = f(s'_1, \dots, s'_m)$, $t = g(t_1, \dots, t_n)$ and $t' = g(t'_1, \dots, t'_n)$. We will make a case analysis on $s \succ_{harpo} t$:

- $s \succ_{harpo} t$ by (i): $s_i \succeq_{harpo} t$ for some $i = 1, \dots, m$. As s'=s, for all s_i there exists s'_j s.t. $s_i = s'_j$. So, we have $s'_j = s_i \succeq_{harpo} t = t'$ and by induction hypothesis $s'_j \succeq_{harpo} t'$ that, by (i), gives $s' \succ_{harpo} t'$.
- If $ha(f, g) = A$ (case ii): is proved as done by arpo in the lemma 3.1.
- If $ha(f, g) = H$ (case iii):
  - $s \succ_{harpo} t$ by (iii-a): $f \succ_{\mathcal{F}} g$ and $s \succ_{harpo} t_i$ for all $i = 1, \dots, n$. Moreover, as $t = t'$, for every t_i there exists t'_j s.t. $s' = s \succ_{harpo} t_i = t'_j$. Thus, by induction hypothesis, $s' \succ_{harpo} t'_j$ for all $j = 1, \dots, n$, and with $f \succ_{\mathcal{F}} g$, by (iii-a), implies $s' \succ_{harpo} t'$.
  - $s \succ_{harpo} t$ by (iii-b):
    $\{s'_1, \dots, s'_m\} == \{s_1, \dots, s_m\} \succ\succ_{harpo} \{t_1, \dots, t_n\} == \{t'_1, \dots, t'_n\}$
    By induction hypothesis, the ordering $\succ_{harpo}$ and equality up to permutation of arguments are compatible on the set of the subterms. As the multiset extension of a compatible (with equality up to permutation of arguments) relation is also compatible, it follows that $\{s'_1, \dots, s'_m\} \succ\succ_{harpo} \{t'_1, \dots, t'_n\}$, and by (iii-b), $s' \succ_{harpo} t'$.

□

Lemma 6.2. *If $s \succeq_{harpo} t = g(t_1, \dots, t_n)$ then $s \succ_{harpo} t_i$ for all $i = 1, \dots, n$.*

Proof. By induction on the size of $s = f(s_1, \dots, s_m)$ and $t = g(t_1, \dots, t_n)$.

1. $s = t$ then, as $=$ is equality up to permutation of arguments, we have that for $i = 1, \dots, n$ there is some $j = 1, \dots, m$, s.t. $s_j = t_i$, and hence by (i) $s \succ_{harpo} t_i$.
2. $s \succ_{harpo} t$ by (i), i.e. $s_j \succeq_{harpo} t$ for some $j = 1, \dots, m$. Then, by induction hypothesis, $s_j \succ_{harpo} t_i$ for all $i = 1, \dots, n$, and by (i) $s \succ_{harpo} t_i$.
3. $s \succ_{harpo} t$ by (ii). In this case we have $\{s_1, \dots, s_m\} \succ\succ_{harpo} \{t_1, \dots, t_n\}$, or $\{s_1, \dots, s_m\} == \{t_1, \dots, t_n\}$, so we can say that for all t_i there exists s_j s.t. $s_j \succeq_{harpo} t_i$, and by (i) $s \succ_{harpo} t_i$.
4. $s \succ_{harpo} t$ by (iii). Case (iii-a) directly gives $s \succ_{harpo} t_i$ for all $i = 1, \dots, n$, and case (iii-b) gives $\{s_1, \dots, s_m\} \succ\succ_{harpo} \{t_1, \dots, t_n\}$ and so, $s \succ_{harpo} t_i$ can be proved by using the same arguments that in case (ii).

$\square$

Lemma 6.3. *The ordering $\succ_{harpo}$ is transitive.*

Proof. Given $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$ and $u = h(u_1, \ldots, u_r)$, we must show that both $s \succ_{harpo} t$ and $t \succ_{harpo} u$ implies $s \succ_{harpo} u$. By induction on the size of s,t and u. We make a case analysis on the case (i, ii or iii) which makes a term $\succ_{harpo}$ than another. For short we will use $\{\vec{s}\}$ for $\{s_1, \ldots, s_m\}$, $\{\vec{t}\}$ for $\{t_1, \ldots, t_n\}$ and $\{\vec{u}\}$ for $\{u_1, \ldots, u_r\}$.

1. $s \succ_{harpo} t$ by (i), i.e. $s_i \succeq_{harpo} t$ for some $i = 1, \ldots, m$. Thus, we have $s_i \succeq_{harpo} t \succ_{harpo} u$ for some $i = 1, \ldots, m$, and by induction hypothesis and compatibility of $\succ_{harpo}$ wrt. $=$, we obtain $s_i \succ_{harpo} u$ for some i. Hence, by (i), $s \succ_{harpo} u$.

2. $t \succ_{harpo} u$ by (i), i.e. there exists t_i s.t. $t_i \succeq_{harpo} u$. As by lemma 6.2 $s \succ_{harpo} t_i$, by induction hypothesis and compatibility of $\succ_{harpo}$ wrt. $=$, we have $s \succ_{harpo} u$.

3. $s \succ_{harpo} t$ and $t \succ_{harpo} u$ by (ii) or by (iii).

   (a) f=g : We have $f(s_1 \ldots s_m) \succ_{harpo} f(t_1 \ldots t_n)$, which implies $\{\vec{s}\} \succ\succ_{harpo} \{\vec{t}\}$.
   We distinguish by cases of $f(t_1 \ldots t_n) \succ_{harpo} h(u_1 \ldots u_r)$:

   - If $f(t_1 \ldots t_n) \succ_{harpo} h(u_1 \ldots u_r)$ by (ii-a) or by (iii-b) then $\{\vec{t}\} \succ\succ_{harpo} \{\vec{u}\}$. By induction hypothesis on the subterms and by transitivity of the multiset extension of the ordering on the set of subterms, we conclude $\{\vec{s}\} \succ\succ_{harpo} \{\vec{u}\}$ that in both cases, (ii-a) and (iii-b), implies $s \succ_{harpo} u$.
   - If $f(t_1 \ldots t_n) \succ_{harpo} h(u_1 \ldots u_r)$ by (ii-b) then $\{\vec{t}\} == \{\vec{u}\}$. Thus we are given $\{\vec{s}\} \succ\succ_{harpo} \{\vec{t}\} == \{\vec{u}\}$ and since the compatibility of $\succ_{harpo}$ wrt. $=$ assures the compatibility of $\succ\succ_{harpo}$ wrt. $==$, we have $\{\vec{s}\} \succ\succ_{harpo} \{\vec{u}\}$. Thus, as we are in a case such that $ha(f, h) = A$, by (ii-a) we can conclude $s \succ_{arpo} u$.
   - If $f(t_1 \ldots t_n) \succ_{harpo} h(u_1 \ldots u_r)$ by (iii-a) then $f \succ_{\mathcal{F}} h$ and $t \succ_{harpo} u_i$ for all $i = 1 \ldots r$. By induction hypothesis we obtain $s \succ_{harpo} u_i$, and by (iii-a) we conclude $s \succ u$.

   (b) $f \neq g$ and g=h: If $ha(f, g) = H$, as $f \neq g$, $s \succ_{harpo} t$ by (iii-a). Since $g = h$, $f \succ_{\mathcal{F}} g$ implies $f \succ_{\mathcal{F}} h$. Moreover, as it has been proved in the previous case, $s \succ_{harpo} u_i$. Hence, by (iii-a) we conclude $s \succ_{harpo} u$. Otherwise, if $ha(f, g) = A$ then $\{\vec{s}\} \succ\succ_{harpo} \{\vec{t}\} \succ\succ_{harpo} \{\vec{u}\}$ or $\{\vec{s}\} == \{\vec{t}\} \succ\succ_{harpo} \{\vec{u}\}$. Now, by compatibility of $\succ_{harpo}$ wrt. $=$, and induction hypothesis (as the multiset extension of a transitive realtion is transitive), in both cases we have $\{\vec{s}\} \succ\succ_{harpo} \{\vec{u}\}$, and thus, by (ii-a), we have $s \succ_{harpo} u$.

(c) $f \neq g$, $g \neq h$ and $f = h$. Since $f = h$, it is not important whether $ha(f,h) = A$ or $ha(f,h) = H$ because, by definition of harpo, the comparations needed to assure $s \succ_{harpo} u$ are, in both cases, the same. By symmetry of ha, as $f = h$, we know either $ha(f,g) = ha(g,h) = A$ or $ha(f,g) = ha(g,h) = H$.

Supose $ha(f,g) = ha(g,h) = H$. Then, since $f \neq g$ and $h \neq g$, both $s \succ_{harpo} t$ and $t \succ_{harpo} u$ are satisfied by (iii-a) and hence $f \succ_{\mathcal{F}} g$ and $g \succ_{\mathcal{F}} h$, but as $f = h$ this contradicts the irreflexivity of $\succ_{\mathcal{F}}$. Therefore $ha(f,g) = ha(g,h) = A$.

In such a case, we have either $\{\vec{s}\} \succ\!\succ_{harpo} \{\vec{t}\}$ or $\{\vec{s}\} == \{\vec{t}\}$. Since $\{\vec{t}\} \succ\!\succ_{harpo} \{\vec{u}\}$, as proved before, $\{\vec{s}\} \succ\!\succ_{harpo} \{\vec{u}\}$, which by (ii-a) gives $s \succ_{harpo} u$.

(d) $f \neq g$, $g \neq h$ and $f \neq h$. There are four cases depending on how the root symbols are compared:

- $ha(f,g) = A$ and $ha(g,h) = A$: According to the definition of $\succ_{harpo}$, ha is compatible with $\succ_{\mathcal{F}}$ and by the constraint (r1) we know that $ha(f,h) = A$. This case is proved as it has been done for arpo.
- $ha(f,g) = H$ and $ha(g,h) = H$: Since $f \neq g$ and $g \neq h$, by (iii-a), $f \succ_{\mathcal{F}} g$ and $g \succ_{\mathcal{F}} h$ and by transitivity of $\succ_{\mathcal{F}}$ we also have $f \succ_{\mathcal{F}} h$. We distinguish whether $ha(f,h) = A$ or $ha(f,h) = H$: Suppose $ha(f,h) = A$ then as, by symmetry of ha, $ha(h,g) = H$, (r2) assures that $f \succ_{\mathcal{F}} g \Rightarrow h \succ_{\mathcal{F}} g$. This, by irreflexivity of $\succ_{\mathcal{F}}$, contradicts $f \succ_{\mathcal{F}} g$ and $g \succ_{\mathcal{F}} h$, so $ha(f,h) = A$ is not possible and thus $ha(f,h) = H$. Since it has been proved before, we have $s \succ_{harpo} u_i$ for all $i = 1, \ldots, r$, and as we know $f \succ_{\mathcal{F}} h$, by (iii-a), $s \succ_{harpo} u$.
- $ha(f,g) = A$ and $ha(g,h) = H$: By constraint (r1) we know that $ha(f,h) = H$. Futhermore $t \succ_{harpo} u$, $g \neq h$ and $ha(g,h) = H$, by definition of harpo (case (iii-a)), implies $g \succ_{\mathcal{F}} h$, and since by symmetry of ha $ha(g,f) = A$, by the constraint (r2) we have $f \succ_{\mathcal{F}} h$. We have $s \succ_{harpo} u_i$ (as proved before), and as we know $f \succ_{\mathcal{F}} h$, by (iii-a) we conclude $s \succ_{harpo} u$.
- $ha(f,g) = H$ and $ha(g,h) = A$: By constraint (r1) we know that $ha(f,h) = H$. $s \succ_{harpo} t$, $f \neq g$ and $ha(f,g) = H$, by (iii-a), implies $f \succ_{\mathcal{F}} g$ and, since $ha(g,f) = H$ and by symmetry of ha $ha(h,f) = H$, as $f \succ_{\mathcal{F}} g$ the constraint (r2) assures $f \succ_{\mathcal{F}} h$ and since, as proved before, $s \succ_{harpo} u_i$, by (iii-a) we can conclude $s \succ_{harpo} u$.

□

Lemma 6.4. $\succ_{harpo}$ *is an ordering, i.e. irreflexive and transitive.*

Lemma 6.5. $\succ_{harpo}$ *enjoys the subterm property.*

Lemma 6.6. $\succ_{harpo}$ *enjoys the deletion property.*

Lemma 6.7. $\succ_{harpo}$ *is monotonic.*

Lemma 6.8. $\succ_{harpo}$ *is stable under substitutions.*

As the ordering behaves like arpo (and also as rpo) when comparing to terms with the same root symbol, the proof of the irreflexivity, monotonicity and deletion properties is the same as for arpo (and in fact as in rpo). The subterm property is included in (i) as happens in both arpo and rpo. The transitivity of $\succ_{harpo}$ has been proved in the lemma 6.3. The stability under substitutions of $\succ_{harpo}$ can be proved by making a case analysis on the value of *ha* on the root symbols and combining the proofs of the stability under substitutions of both arpo and rpo. (In a similar way as done in the proof of lemma 6.1).

Theorem 6.9. *The head-argument recursive path ordering, $\succ_{harpo}$, is a simplification ordering.*

According to the definition of harpo the function *ha* has to be compatible with the precedence. This means that given *ha* and the precedence we have to check that the restrictions (r1) and (r2) are fulfilled. The checking of such restrictions implies lots of comparisons between the symbols of the signature. In order to simplify this checking we will provide a necessary and sufficient condition based on partitioning the signature in disjoint subsets, which is the one we are going to use in the examples when applying harpo.

The restrictions that *ha* has to satisfy in order to be compatible with $\succ_{\mathcal{F}}$, divides $\mathcal{F}$ in a partition $\mathcal{F}_1, \mathcal{F}_2, \ldots$, such that the symbols occurring in the same $\mathcal{F}_i$ are all compared by H or all compared by A. The symbols in $\mathcal{F}_i$ and $\mathcal{F}_j$, being $i \neq j$, are compared by H. Moreover, $\succ_{\mathcal{F}}$ is compatible with this partition, i.e. all symbols in some $\mathcal{F}_i$ compares in the same way with all symbols in another $\mathcal{F}_j$.

if any symbol of a group is comparable by $\succ_{\mathcal{F}}$ with any symbol of another group, then all the symbols of the group enjoy the same constraint.

Definition 6.3. Let $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_k$ be disjoint subsets of a set of function symbols $\mathcal{F}$ s.t $\mathcal{F} = \mathcal{F}_1 \biguplus \mathcal{F}_2 \biguplus \ldots \biguplus \mathcal{F}_k$

1. $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_k$ is a partition wrt. *ha* if it fulfils:
   - (α) for all $\mathcal{F}_i$,

     for all $f, g \in \mathcal{F}_i$, $ha(f, g) = A$, or
     for all $f, g \in \mathcal{F}_i$, $ha(f, g) = H$
   - (β) for all f, g, s.t. $f \in \mathcal{F}_i$, $g \in \mathcal{F}_j$ and $i \neq j$, $ha(f, g) = H$
2. $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_k$ is *compatible with the precedence* if it fulfils:

- (γ) for all f_1, f_2, g_1, g_2 s.t. $f_1, f_2 \in \mathcal{F}_i$, $g_1, g_2 \in \mathcal{F}_j$, $i \neq j$

$$f_1 \succ g_1 \Rightarrow f_2 \succ g_2$$

Proposition 6.10. *A function $ha : \mathcal{F} \times \mathcal{F} \to \{A, H\}$ is* compatible with the precedence *iff there exists a partition of the signature wrt. ha compatible with the precedence.*

Proof. To prove the right-to-left implication, we will show that if such partition exists then all the restrictions in the definition of ha are satisfied.

Note that condition (β) implies that : for all f and g,

$$ha(f, g) = A \quad \Rightarrow \quad f, g \in \mathcal{F}_i \quad \text{for some i. } (\delta)$$

- (r1) If $ha(f, g) = A$ and $ha(g, h) = A$, by (δ), $f, g, h \in \mathcal{F}_i$ for some i, and by (α) $ha(f, h) = A$.
- (r2) Given $ha(f, g) = A$, $ha(g, h) = H$ and $ha(f, h) = H$ then condition (δ) assures that $f, g \in \mathcal{F}_i$ and condition (α) that $h \in \mathcal{F}_j$ for some $i \neq j$. Therefore, the restriction (γ) gives both $f \succ_{\mathcal{F}} h \Rightarrow g \succ_{\mathcal{F}} h$ and $h \succ_{\mathcal{F}} f \Rightarrow h \succ_{\mathcal{F}} g$.

To prove the left-to-right implication we take the following partition $\mathcal{F} = \mathcal{F}_1 \biguplus \mathcal{F}_2 \biguplus \ldots \biguplus \mathcal{F}_n$ with a maximal n fulfilling

$$ha(f, g) = A \quad \Rightarrow \quad f, g \in \mathcal{F}_i \quad \text{for some i}$$

We will prove that

1. $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n$ is a partition wrt. ha.
    - As we have defined this partition, for all $\mathcal{F}_i$ it is true that either the cardinality is 1 or the symbols in $\mathcal{F}_i$ are compared with each other by A. So (α) holds.
    - The partition we have built fulfils that

      $$ha(f, g) = A \quad \Rightarrow \quad f, g \in \mathcal{F}_i \quad \text{for some i}$$

      So, we can assure that (β) is satisfied, i.e. for all f, g, s.t. $f \in \mathcal{F}_i$, $g \in \mathcal{F}_j$ and $i \neq j$, $ha(f, g) = H$.
2. $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n$ is compatible with the precedence.

   (γ) Given $f_1, f_2 \in \mathcal{F}_i$ and $g_1, g_2 \in \mathcal{F}_j$, $i \neq j$ we have to prove that

   $$f_1 \succ g_1 \Rightarrow f_2 \succ g_2$$

    - if $g_1 = g_2$ then
        - if $f_1 = f_2$ is trivial.

  - if $f_1 \neq f_2$ then by the definition of the partition we know that $ha(f_1, f_2) = A$, $ha(f_1, g_1) = H$ and $ha(f_2, g_1) = H$, and by the first part of (r2) we have $f_1 \succ g_1 \Rightarrow f_2 \succ g_1$. So, as $g_1 = g_2$, $f_1 \succ g_1 \Rightarrow f_2 \succ g_2$ holds.

- if $g_1 \neq g_2$ then by the definition of the partition we know that $ha(g_1, g_2) = A$, $ha(g_1, f_1) = H$ and $ha(g_2, f_1) = H$, and the second part of (r2) gives $f_1 \succ g_1 \Rightarrow f_1 \succ g_2$.
  - if $f_1 = f_2$ then we have $f_1 \succ g_1 \Rightarrow f_2 \succ g_2$.
  - if $f_1 \neq f_2$ then by the definition of the partition we know that $ha(f_1, f_2) = A$, $ha(f_1, g_2) = H$ and $ha(f_2, g_2) = H$, and applying the first part of (r2) we obtain $f_1 \succ g_2 \Rightarrow f_2 \succ g_2$. So $f_1 \succ g_1 \Rightarrow f_2 \succ g_2$ holds.

$\square$

7 Applications of harpo

For short, in the following examples, to represent the fact that

$$\text{for all } f \in \mathcal{F}_i \text{ and for all } g \in \mathcal{F}_j,\ f \succ_{\mathcal{F}} g$$

we will use the notation $\mathcal{F}_i \succ \mathcal{F}_j$.

The examples 7.1 and 7.2 show how we can prove that a TRS is simply terminating by using the simplification ordering harpo. The rest are examples to prove termination of TRSs by using the reduction ordering $\succ_r$ defined from $\succ_{harpo}$. (In all the examples we apply the case (1) of the definition of $\succ_r$).

Example 7.1. *The following TRS computes the dual logarithm [AG97]. Using neither the recursive path ordering (rpo) nor the argument recursive path ordering (arpo) we are able to prove this TRS is simply terminating. We will prove it by using the ordering $\succ_{harpo}$.*

$$half(0) \to 0$$

$$half(s(s(x))) \to s(half(x))$$

$$log(0) \to 0$$

$$log(s(s(x))) \to s(log(s(half(x))))$$

Let $\succ_{\mathcal{F}}$ be the partial ordering over the signature $\mathcal{F}$, s.t. $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3$ where $\mathcal{F}_1 = \{log\}$, $\mathcal{F}_2 = \{s \succ_{\mathcal{F}} half\}$, $\mathcal{F}_3 = \{0\}$ and symbols of $\mathcal{F}_2$ are compared by A (i.e. for all $f, g \in \mathcal{F}_2$, ha(f,g)=A), and $\mathcal{F}_1 \succ \mathcal{F}_2$. (remember that symbols in different parts are compared by H).

We will show that the rewrite relation defined by this TRS is included in the simplification ordering $\succ_{harpo}$:

- *half(0) $\succ_{harpo}$ 0 : by (i)*

- *half(s(s(x)))* $\succ_{harpo}$ *s(half(x))* : *since* $ha(half, s) = A$, *is proved by applying twice (ii-a) on the arguments and once (i).*
- *log(0)* $\succ_{harpo}$ *0* : *by (i)*
- *log(s(s(x)))* $\succ_{harpo}$ *s(log(s(half(x))))* : *since* $ha(log, s) = H$, *we need both*
  - *log* $\succ_{\mathcal{F}} s$, *that it is true because* $\mathcal{F}_1 \succ_{\mathcal{F}} \mathcal{F}_2$, *and*
  - *log(s(s(x)))* $\succ_{harpo}$ *log(s(half(x)))* : *by applying (ii-a) or (iii-b) we need*

    s(s(x))) $\succ_{harpo}$ *s(half(x)), and by the same argument,*
    s(x) $\succ_{harpo}$ *half(x)* : *that it is proved by (ii-b) since* $s \succ_{\mathcal{F}} half$.

Note that is not possible to prove the termination of this TRS neither by using $\succ_{rpo}$ *because both,* $half \succ_{\mathcal{F}} s$ *and* $s \succ_{\mathcal{F}} half$, *should be true, nor by using* $\succ_{arpo}$ *since the ordering cannot be satisfied for the last rule.*

Example 7.2. *This TRS to eliminate even numbers from a list is almost equal to the TRS of the example 5.2. In this case we will prove it is terminating by showing that the rewrite relation is included in* $\succ_{harpo}$.

$$ev(0) \to true$$

$$ev(s(0)) \to false$$

$$ev(s(s(x))) \to ev(x)$$

$$rmev(nil) \to nil$$

$$rmev(add(x, l)) \to if(ev(x), x, l)$$

$$if(true, x, l) \to rmev(l)$$

$$if(false, x, l) \to add(x, rmev(l))$$

It can be proved with $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3$ *where* $\mathcal{F}_1 = \{rmev, if\}$, $\mathcal{F}_2 = \{add, ev, true, false\}$, $\mathcal{F}_3 = \{s, nil, 0\}$, *such that* $\mathcal{F}_1 \succ \mathcal{F}_2$, *and symbols in both* $\mathcal{F}_1$ *and* $\mathcal{F}_2$ *are compared by* A *(for* $\mathcal{F}_3$ *both,* H *and* A*, are possible).*

- *ev(0)* $\succ_{harpo}$ *true* : *by (ii-a)*
- *ev(s(0))* $\succ_{harpo}$ *false* : *by (ii-a)*
- *ev(s(s(x)))* $\succ_{harpo}$ *ev(x)* : *by (ii-a)*
- *rmev(nil)* $\succ_{harpo}$ *nil* : *by (i)*
- *rmev(add(x,l))* $\succ_{harpo}$ *if(ev(x),x,l)* :
  by (ii-a) since $ha(rmev, if) = A$ *and* $ha(add, ev) = A$.

- *if(true,x,l)* $\succ_{harpo}$ *rmev(l)* : *by (ii-a)*
- *if(false,x,l)* $\succ_{harpo}$ *add(x,rmev(l))* :
  by (iii-a) since $ha(if, add) = H$, $if \succ_{\mathcal{F}} add$ *and* $ha(if, rmev) = A$.

The termination of the rest of TRSs presented in this section will be proved by using the reduction ordering $\succ_r$ defined from $\succ_{harpo}$. We will prove that $l \succ_r r$ for every rule $l \to r \in R$, i.e. $I(l) \succ_{harpo} I(r)$ and $l \succ_e r_i$ for all r_i subterm of r.

Example 7.3. *This TRS eliminates duplicates from a list.*

$$eq(0,0) \to true$$

$$eq(0,s(x)) \to false$$

$$eq(s(x),0) \to false$$

$$eq(s(x),s(y)) \to eq(x,y)$$

$$rm(x,nil) \to nil$$

$$rm(x,add(y,l)) \to if(eq(x,y),x,y,l)$$

$$if(true,x,y,l) \to rm(x,l)$$

$$if(false,x,y,l) \to add(y,rm(x,l))$$

$$purge(nil) \to nil$$

$$purge(add(x,l)) \to add(x,rm(x,purge(l)))$$

Given the following interpretation function: $I(if(x,y,z,w)) = if(y,z,w)$, *and the identity function for the rest of the symbols in the signature,*
a suitable partition of $\mathcal{F}$ *could be (there are some other possibilities)* $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3 \cup \mathcal{F}_4$ *s.t.* $\mathcal{F}_1 \succ \mathcal{F}_2 \succ \mathcal{F}_3$ *and*
$\mathcal{F}_1 = \{purge\}$,
$\mathcal{F}_2 = \{rm, if, eq, true, false\}$ *with the symbols compared by* A,
$\mathcal{F}_3 = \{add\}$, *and*
$\mathcal{F}_4 = \{nil, s\}$.

- *for every rule* $l \to r$, $I(l) \succ_{harpo} I(r)$ *is satisfied:*
  - *eq(0,0)* $\succ_{harpo}$ *true* : *by (ii-a)*
  - *eq(0,s(x))* $\succ_{harpo}$ *false* : *by (ii-a)*
  - *eq(s(x),0)* $\succ_{harpo}$ *false* : *by (ii-a)*
  - *eq(s(x),s(y))* $\succ_{harpo}$ *eq(x,y)* : *by (ii-a)*
  - *rm(x,nil)* $\succ_{harpo}$ *nil* : *by (i)*
  - *rm(x,add(y,l))* $\succ_{harpo}$ *if(x,y,l)* : *by (ii-a)*

- $if(x,y,l) \succ_{harpo} rm(x,l)$: by (ii-a)
- $if(x,y,l) \succ_{harpo} add(y,rm(x,l))$: by (iii-a) since $ha(if, add) = H$, $if \succ_{\mathcal{F}} add$ and $ha(if, rm) = A$.
- $purge(nil) \succ_{harpo} nil$: by (i)
- $purge(add(x,l)) \succ_{harpo} add(x,rm(x,purge(l)))$:
  by (iii-a) since $ha(purge, add) = H$, $purge \succ_{\mathcal{F}} add$, and both $purge(add(x,l)) \succ_{harpo} x$ and $purge(add(x,l)) \succ_{harpo} rm(x,purge(l))$ are satisfied. The first one is true by (i) and the second one because $ha(purge, rm) = H$, $purge \succ_{\mathcal{F}} rm$ and $purge(add(x,l)) \succ_{harpo} purge(l)$.

- *and for every rule $l \rightarrow r$, $l \succ_e r_i$ for each r_i subterm of r:*
  - $eq(s(x),s(y)) \succ_e x$: by (1)
  - $eq(s(x),s(y)) \succ_e y$: by (1)
  - $rm(x,add(y,l)) \succ_e eq(x,y)$: by (2)
  - $rm(x,add(y,l)) \succ_e x$: by (1)
  - $rm(x,add(y,l)) \succ_e y$: by (1)
  - $rm(x,add(y,l)) \succ_e l$: by (1)
  - $if(true,x,y,l) \succ_e x$: by (1)
  - $if(true,x,y,l) \succ_e l$: by (1)
  - $if(false,x,y,l) \succ_e y$: by (1)
  - $if(false,x,y,l) \succ_e rm(x,l)$: by (2)
  - $purge(add(x,l)) \succ_e x$: by (1)
  - $purge(add(x,l)) \succ_e rm(x,purge(l))$: by (2)

Example 7.4. *This TRS [AG97] is equivalent to the TRS of the above example:*

$$eq(0, 0) \rightarrow true$$

$$eq(0, s(x)) \rightarrow false$$

$$eq(s(x), 0) \rightarrow false$$

$$eq(s(x), s(y)) \rightarrow eq(x, y)$$

$$rm(x, nil) \rightarrow nil$$

$$rm(x, add(y, l)) \rightarrow if(eq(x, y), x, add(y, l))$$

$$if(true, x, add(y, l)) \rightarrow rm(x, l)$$

$$if(false, x, add(y, l)) \rightarrow add(y, rm(x, l))$$

$$purge(nil) \rightarrow nil$$

$$purge(add(x,l)) \rightarrow add(x, purge(rm(x,l)))$$

An interpretation function to prove termination of this TRS is

$$I(rm(x,y)) = rm(y)$$

$$I(if(x,y,z)) = if(z)$$

$$I(add(x,y)) = add(y)$$

and the identity function for the rest of the function symbols (eq, s, true, false and nil).

A suitable partition of $\mathcal{F}$ *could be* $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3$ *s.t.* $\mathcal{F}_1 \succ \mathcal{F}_2$ *where* $\mathcal{F}_1 = \{purge\}, \mathcal{F}_2 = \{add \succ_{\mathcal{F}} rm \succ_{\mathcal{F}} if, eq, true, false\}$ *and* $\mathcal{F}_3 = \{0, nil, s\}$ *, and the symbols in both,* $\mathcal{F}_2$ *and* $\mathcal{F}_3$*, are compared by* A*.*

Example 7.5. *The following TRS for division is not simply terminating.*

$$le(0,y) \rightarrow true$$

$$le(s(x),0) \rightarrow false$$

$$le(s(x),s(y)) \rightarrow le(x,y)$$

$$minus(0,y) \rightarrow 0$$

$$minus(s(x),y) \rightarrow if(le(s(x),y),s(x),y)$$

$$if(true,s(x),y) \rightarrow 0$$

$$if(false,s(x),y) \rightarrow s(minus(x,y))$$

$$quot(0,s(y)) \rightarrow 0$$

$$quot(s(x),s(y)) \rightarrow s(quot(minus(x,y),s(y)))$$

We can prove it is terminating by using $\succ_r$ *defined from* $\succ_{harpo}$*, the following interpretation function*

$$I(le(x,y)) = le(x)$$

$$I(minus(x,y)) = minus(x)$$

$$I(if(x,y,z)) = if(y)$$

$$I(quot(x,y)) = quot(x)$$

(the identity function for the rest of the symbols)

and this partition of $\mathcal{F}$*:* $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ *s.t.* $\mathcal{F}_1 \succ \mathcal{F}_2$ *where*

$\mathcal{F}_1 = \{quot\}$,

$\mathcal{F}_2 = \{s, minus, if, le, true, false, 0\}$ *s.t. the symbols are compared by* A *and* $s \succ_{\mathcal{F}} minus \succ_{\mathcal{F}} if$ *and* $minus \succ_{\mathcal{F}} le$*.*

Example 7.6. *This TRS is used to sort a list by the well-known quicksort-algorithm [AG97].*

$$le(0, y) \to true$$

$$le(s(x), 0) \to false$$

$$le(s(x), s(y)) \to le(x, y)$$

$$app(nil, y) \to y$$

$$app(add(n, x), y) \to add(n, app(x, y))$$

$$low(n, nil) \to nil$$

$$low(n, add(m, x)) \to if_{low}(le(m, n), n, add(m, x))$$

$$if_{low}(true, n, add(m, x)) \to add(m, low(n, x))$$

$$if_{low}(false, n, add(m, x)) \to low(n, x)$$

$$high(n, nil) \to nil$$

$$high(n, add(m, x)) \to if_{high}(le(m, n), n, add(m, x))$$

$$if_{high}(true, n, add(m, x)) \to high(n, x)$$

$$if_{high}(false, n, add(m, x)) \to add(m, high(n, x))$$

$$quick(nil) \to nil$$

$$quick(add(n, x)) \to app(quick(low(n, x)), add(n, quick(high(n, x))))$$

The termination of this TRS can be proved by using $\succ_r$ defined from $\succ_{harpo}$, the following interpretation function:

$$I(le(x, y)) = le(x)$$

$$I(low(x, y)) = low(y)$$

$$I(if_{low}(x, y, z)) = if_{low}(z)$$

$$I(high(x, y)) = high(y)$$

$$I(if_{high}(x, y, z)) = if_{high}(z)$$

(the identity function for the rest of the symbols).

A good partition of $\mathcal{F}$ could be $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ s.t. $\mathcal{F}_1 \succ \mathcal{F}_2$ where

$\mathcal{F}_1 = \{quick \succ_{\mathcal{F}} app\}$, the symbols are compared by H.

$\mathcal{F}_2 = \{add, low, if_{low}, high, if_{high}, le, s, nil, 0, true, false\}$, the symbols are compared by A, and $low \succ_{\mathcal{F}} if_{low}$ and $high \succ_{\mathcal{F}} if_{high}$.

Example 7.7. *This TRS replaces all the instances of an element from a list by another and, in particular, by the minimum element of the list.*

$$eq(0,0) \to true$$

$$eq(0,s(x)) \to false$$
$$eq(s(x),0) \to false$$
$$eq(s(x),s(y)) \to eq(x,y)$$
$$le(0,y) \to true$$
$$le(s(x),0) \to false$$
$$le(s(x),s(y)) \to le(x,y)$$
$$min(add(x,nil)) \to x$$
$$min(add(x,add(y,l))) \to min2(le(x,y),add(x,add(y,l)))$$
$$min2(true,add(x,add(y,l))) \to min(add(x,l))$$
$$min2(false,add(x,add(y,l))) \to min(add(y,l))$$

$$repl(x,y,nil) \to nil$$

$$repl(x,y,add(z,l)) \to if(eq(x,z),x,y,add(z,l))$$
$$if(true,x,y,add(z,l)) \to add(y,repl(x,y,l))$$
$$if(false,x,y,add(z,l)) \to add(z,repl(x,y,l))$$

$$repmin(x,nil) \to nil$$

$$repmin(x,add(y,l)) \to subs(eq(x,min(add(y,l))),x,add(y,l))$$

$$subs(true,x,l) \to l$$

$$subs(false,x,l) \to repl(x,min(l),l)$$

It is proved by using the interpretation function

$$I(min2(x,y)) = min2(y)$$

$$I(if(x,y,z,w)) = if(y,z,w)$$
$$I(subs(x,y,z)) = subs(y,z),$$

(the identity function for the others),

and the following partition of $\mathcal{F}$: $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3 \cup \mathcal{F}_4$ *s.t.* $\mathcal{F}_1 \succ \mathcal{F}_2 \succ \mathcal{F}_3$ *where* $\mathcal{F}_1 = \{repmin, subs, min \succ_{\mathcal{F}} min2, le\}$ *symbols compared by* A, $\mathcal{F}_2 = \{repl \succ_{\mathcal{F}} if, eq, true, false\}$ *symbols compared by* A, $\mathcal{F}_3 = \{add\}$ *and* $\mathcal{F}_4 = \{nil, s, 0\}$.

8 Total orderings

A useful property of an ordering, for instance when applied in automated deduction, is the totality: for all pair of terms s and t, either s and t are equal (in some sense) or they must be comparable.

Definition 8.1. An ordering $\succ$ is said to be total on a set $\mathcal{A}$ up to an equivalence relation $=$ if for all terms s and t in $\mathcal{A}$, s=t or $s \succ t$ or $t \succ s$.

Proposition 8.1. *Let $\succ_{\mathcal{F}}$ be a total ordering on the set of function symbols $\mathcal{F}$. Then the argument recursive path ordering $\succ_{arpo}$, is a total ordering on $\mathcal{T}(\mathcal{F})$ (up to equality under permutation of arguments).*

Note that, as it happens in rpo, arpo is not total on $\mathcal{T}(\mathcal{F})$ if syntactic equality is considered. For instance, the terms f(a,g(a,b)) and f(g(b,a),a), which are not syntactically equals, are incomparable under arpo.

Proof. We must prove that for all $s, t \in \mathcal{T}(\mathcal{F})$, s=t or $s \succ_{arpo} t$ or $t \succ_{arpo} s$, when equality up to permutation of arguments is considered and $\succ_{\mathcal{F}}$ is total.

By induction on the size of the two terms s and t, where $s = f(s_1, \ldots, s_n)$ and $t = g(t_1, \ldots, t_m)$.

Let $\succ$ be a total ordering on a set of terms $\mathcal{A}$. Is known that the extension (wrt. equality under permutation of arguments) of $\succ$ to finite multisets is also total.

By induction hypothesis, $\succ_{arpo}$ is total on the set of subterms s_i, t_j, so the multiset extension of $\succ_{arpo}$ wrt. equality under permutation of arguments is total. According to this, one of the next three cases is true:

- $\{s_1, \ldots, s_n\} \succ\succ \{t_1, \ldots, t_m\}$ and by (ii) $s \succ_{arpo} t$, or
- $\{t_1, \ldots, t_m\} \succ\succ \{s_1, \ldots, s_n\}$ and by (ii) $t \succ_{arpo} s$, or
- $\{s_1, \ldots, s_n\} == \{t_1, \ldots, t_m\}$ and since $\succ_{\mathcal{F}}$ is total, $f \succ_{\mathcal{F}} g$ or $g \succ_{\mathcal{F}} f$ or else the two terms s and t have the same root symbol. If $f \succ_{\mathcal{F}} g$ then by (iii) $s \succ_{arpo} t$, otherwise if $g \succ_{\mathcal{F}} f$ then by (iii) $t \succ_{arpo} s$, or else the terms s and t are equal up to permutation of arguments and so s=t.

□

Proposition 8.2. *Let $\succ_{\mathcal{F}}$ be a total ordering on the set of function symbols $\mathcal{F}$. Then the head-argument recursive path ordering $\succ_{harpo}$, is a total ordering on $\mathcal{T}(\mathcal{F})$ (up to equality under permutation of arguments).*

Proof. We must prove that for all $s, t \in \mathcal{T}(\mathcal{F})$, s=t or $s \succ_{harpo} t$ or $t \succ_{harpo} s$, when equality under permutation of arguments is considered and $\succ_{\mathcal{F}}$ is total. By induction on the size of the two terms.

- If the terms s and t have the same root symbol, the comparison of the two terms is the same whether it is done either by A or by H. Therefore, we can prove it exactly as done in the case of arpo.

- Let $s = f(s_1, \ldots, s_n)$ *and* $t = g(t_1, \ldots, t_m)$, $f \neq g$. As we know, for all pair of function symbols f and g, $ha(f,g) = A$ or $ha(f,g) = H$, hence we consider this two possibilities:
  - $ha(f,g) = A$: is proved as done in arpo.
  - $ha(f,g) = H$: Since $\mathcal{F}$ is total and $f \neq g$ then $f \succ_{\mathcal{F}} g$ or $g \succ_{\mathcal{F}} f$. If $f \succ_{\mathcal{F}} g$
    * If $s \succ_{harpo} t_j$ for all $j = 1, \ldots, m$ then, by (iii-a), we have $s \succ_{harpo} t$, otherwise
    * there exists some $j = 1, \ldots, m$ s.t. $s \not\succ_{harpo} t_j$. Applying induction hypothesis it follows either $t_j \succ_{harpo} s$ or $t_j = s$, and so, by (i), we conclude $t \succ_{harpo} s$.

  If $g \succ_{\mathcal{F}} f$ we would proceed as done in the previous case.

□

9 Future work

To increase the power of this technique it would be interesting to extend this orderings in two senses. First of all, by adding the possibility of giving a *Status* to the function symbols, allowing to compare their arguments not only as multisets but also as strings (lexicographically) as in the *recursive path ordering with status*. In this case we would need the function symbols to have a fixed arity (or at least bounded arity), otherwise we could not ensure the well-foundedness of the ordering.

On the other hand we would like to add the possibility of dealing with AC-symbols, i.e. function symbols enjoying the associativity and commutativity (AC) properties. This is not a trivial extension at all; allowing this kind of symbols implies that the ordering must be compatible with the equivalence classes defined by the AC axioms, i.e. it should compare in the same way all terms in two different classes. Unfortunately AC-compatibility and monotonicity cannot be both kept easily.

References

[AG97] T. Arts and J. Giesl. Proving innermost normalisation automatically. In H. Comon, editor, *8th International Conference on Rewriting Techniques and Applications*, LNCS 1232, pages 157–171, Sitges, Spain, 1997. Springer-Verlag.

[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.

[BD86] Leo Bachmair and Nachum Dershowitz. Commutation, transformation, and termination. In Jörg H. Siekmann, editor, *8th International Conference on Automated Deduction*, LNCS 230, pages 5–20, Oxford, England, July 27–August 1, 1986. Springer-Verlag.

[BL90] Françoise Bellegarde and Pierre Lescanne. Termination by completion, 1990.

[Der82] Nachum Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982.

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

[DH93] Nachum Dershowitz and Charles Hoot. Topics in termination. In C. Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications*, LNCS 690, pages 198–212, Montreal, Canada, 1993.

[DJ90] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 6, pages 244–320. Elsevier Science Publishers B.V., Amsterdam, New York, Oxford, Tokyo, 1990.

[FZ93a] Maria Ferreira and Hans Zantema. Dummy elimination: making termination easier. In H. Reichel, editor, *10th Int. Conf. on Fundamentals of Computation Theory*, LNCS 965, pages 243–252. Springer-Verlag, June 16–18, 1993.

[FZ93b] Maria Ferreira and Hans Zantema. Total termination of term rewriting. In C. Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications*, LNCS 690, pages 213–227, Montreal, Canada, 1993.

[Gee91] M. Geerling. Termination of term rewriting systems. Master's thesis, Utrecht University, PO Box 80.089, 3508 TB Utrecht, The Netherlands, 1991.

[Ges92] Alfons Geser. On a monotonic semantic path ordering. Technical Report 92-13, Universität Ulm, Fakultät für Informatik, 1992.

[Itu67] R. Iturriaga. *Contributions to mechanical mathematics.* PhD thesis, Dept. of Computer Science, Carnegie-Melon Univ., Pittsburgh, PA, 1967.

[KL80] S. Kamin and J.-J. Levy. Two generalizations of the recursive path ordering. Unpublished note, Dept. of Computer Science, Univ. of Illinois, Urbana, IL, 1980.

[Klo92] J.W. Klop. Term rewriting systems. In S. Abramsky, D.M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, pages 1–116. Oxford University Press, 1992.

[KNS85] Deepak Kapur, Paliath Narendran, and G. Sivakumar. A path ordering for proving termination for term rewriting systems. In *Proc. of 10th Colloquium on Trees in Algebra and Programming*, LNCS 185, pages 173–185, Germany, 1985. Springer-Verlag.

[Kru60] Joseph B. Kruskal. Well-quasi-ordering, the Tree Theorem, and Vazsonyi's conjecture. *Transactions of the American Mathematical Society*, 95:210–225, May 1960.

[Lan79] D. S. Lankford. On proving term rewriting systems are noetherian. Technical Report Memo MTP-3, Dept. of Mathematics, Louisiana Tech. Univ., Ruston, LA, 1979.

[Les82] P. Lescanne. Some properties of descomposition ordering, a simplification ordering to prove termination of rewriting systems. *RAIRO Informatique théorique*, 16:331–347, 1982.

[Zan92] H. Zantema. Type removal in term rewriting. In M. Rusinowitch and J.L. Rémy, editors, *Proc. 3rd International Workshop on Conditional Term Rewriting Systems*, LNCS 656, pages 155–167, Pont-à-Mousson, France, 1992. Springer-Verlag.

[Zan95] H. Zantema. Termination of term rewriting by semantic labelling. *Fundamenta Informaticae*, 24:89–105, 1995.