Rewrite Systems

K. Arkoudas

May 1994

Figure 1: Terms as trees

Introduction

Terms

Rewrite rules are applied to **terms**, which are syntactic objects built from certain constants, variables, and function symbols. In particular, let F be a given set of function symbols. For each function symbol $f \in F$ there is an associated integer $n \geq 0$ denoting the -*arity* of f (the number of its arguments). Function symbols of zero -arity (of no arguments) are treated as *constants*. Now for $n \geq 0$ let $F_n = \{f \in F | f \text{ has } -\text{arity } n\}$ (so that $F = \bigcup_{n \geq 0} F_n$) and let $V = \{u, v, w, x, y, z, u_1, v_1, \ldots\}$ be a countably infinite supply of variable symbols. We write $T(F, V)$ to denote the set of **terms over** F **and** V. This set is recursively defined as follows:
(1) All symbols in F_0 (i.e all constants) and all variables in V are in $T(F, V)$.
(2) If $f \in F_n$ and $t_1, \ldots, t_n$ are in $T(F, V)$, then $f(t_1, \ldots, t_n) \in T(F, V)$.

For example, if F is the set $AR = \{0, s, +, *\}$ (where 0 is a constant, s is unary, and $+, *$ are binary), then the following are terms over AR and V:
(a) 0
(b) $s(0)$
(c) $*(s(0), +(s(s(0)), s(0)))$
(d) $+(x, s(s(+(y, 0))))$
(e) $s(*(z, s(+(x, s(0)))))$
Using the more familiar *infix notation* these terms can be written as:
(a) 0
(b) $s(0)$
(c) $s(0) * (s(s(0)) + s(0))$
(d) $x + s(s(y + 0))$
(e) $s(z * s(x + s(0)))$

It is clear from their definition that terms are ordered tree structures, and we often depict them as trees. E.g the figure above displays the terms (b), (c), and (d) that were given in the previous page.

We use the letters $s, t, ...$ to refer to terms in $T(F, V)$. If a term t contains no variable symbols, we say it is a **ground term**. Terms (a), (b), and (c) above are ground, whereas (d) and (e) are not. We write $T(F)$ to denote the

set of ground terms over F (so that $T(F) \subset T(F,V)$).

A **subterm** of a term $t = f(t_1, \ldots, t_n)$ (for some $f \in F_n, n \geq 0$) is either (i) t itself, or (ii) a subterm of a t_i, for some $i \in \{1, \ldots, n\}$. In the tree representation of terms, the subterms of t are its subtrees. If s is a subterm of t and $s \neq t$, we say that s is a **proper subterm** of t. The **size** of a term t is the number

$$\sum_{f \in F} OC(f,t) + \sum_{v \in V} OC(v,t)$$

where $OC(f,t)$ (or $OC(v,t)$) is the number of times f (or v) occurs in t. More simply, the size of t is the number of its nodes (viewing t as a tree), or equivalently, the number of its subtrees (i.e the number of its subterms).

A **substitution** is a function σ from a finite subset of V to $T(F,V)$. We often denote a substitution σ by an expression of the form $\{v_1 \mapsto t_1, \ldots, v_k \mapsto t_k\}$, where $\{v_1, \ldots, v_k\}$ is Dom_σ (the domain of σ) and $\{t_1, \ldots, t_k\}$ is Ran_σ (the range of σ), the obvious interpretation being that σ maps v_1 to t_1 ("replaces" v_1 by t_1), v_2 to t_2, etc. If Ran_σ comprises ground terms only, we say that σ is a **ground substitution**. We can recursively extend a substitution σ to $T(F,V)$. In particular, givan a term t and a substitution σ, we define the **application of σ to** t, written as $t\sigma$, in the following way:
Case 1: t is a variable. Then if $t \in Dom_\sigma$ $t\sigma = \sigma(t)$; otherwise, $t\sigma = t$.
Case 2: t is of the form $f(t_1, \ldots, t_n)$ for some $f \in F_n (n \geq 0)$. Then $t\sigma = f(t_1\sigma, \ldots, t_n\sigma)$. In other words, $t\sigma$ is obtained by replacing all occurences of variable symbols in t by the terms specified by σ. For example, if t is $x+s(x+s(0*y))$ and σ is $\{x \mapsto s(0), y \mapsto z\}$, the $t\sigma = s(0)+s(s(0)+s(0*z))$.

Two terms s and t are said to be **unifiable** if there is a substitution σ which **unifies** s and t, i.e such that $s\sigma = t\sigma$; σ is called the **unifier** of s and t. For instance, the terms $(x+u)*s(s(u))$ and $(s(y+z)+0)*w$ are unified by the substitution $\{x \mapsto s(y+z), w \mapsto s(s(0)), u \mapsto 0\}$. The **composition** of two substitutions σ and τ, denoted by $\sigma\tau$, is the substitution obtained by applying τ to the range of σ and then adding to σ all those mappings $v_i \mapsto t_i$ in τ for which $v_i \notin Dom_\sigma$. For example, if $\sigma = \{x \mapsto y+z\}$ and $\tau = \{y \mapsto 0, z \mapsto s(0), w \mapsto 0*u\}$, then $\sigma\tau = \{x \mapsto 0+s(0), y \mapsto 0, z \mapsto s(0), w \mapsto 0*u\}$. A unifier σ of two terms s and t is called the **most general unifier** of s and t iff for every unifier τ of s and t there is a substitution τ' such that $\tau = \sigma\tau'$. If two terms are unifiable at all, then they have a most general unifier which is unique and easy to compute.

Rewriting

Given a set of function symbols F, a **rewrite system** R on F is a finite set of ordered pairs $< l_i, r_i >, i = 1, \ldots, p$, called the **rewrite rules** of R, where l_i, r_i are terms over F and V and $Var(r_i) \subseteq Var(l_i)$.[1] A rewrite rule $< l_i, r_i >$ is usually written as $l_i \longrightarrow r_i$. An example of a rewrite system on $F = \{\wedge, \vee, \neg, 0, 1\}$ is the following:

$$\begin{array}{l} \neg\neg x \longrightarrow x \\ \neg(x \vee y) \longrightarrow (\neg x \wedge \neg y) \\ \neg(x \wedge y) \longrightarrow (\neg x \vee \neg y) \\ x \wedge (y \vee z) \longrightarrow (x \wedge y) \vee (x \wedge z) \\ (y \vee z) \wedge x \longrightarrow (y \wedge x) \vee (z \wedge x) \end{array} \qquad \textbf{(I)}$$

This system models the transformation of propositional logic formulae (built from $\vee, \wedge$, and $\neg$) into disjunctive normal form.

Given a rewrite system R, we define the binary **rewrite relation** $\Longrightarrow_R$ on $T(F, V)$ as follows:
$s \Longrightarrow_R t$ iff there is a subterm u of s and a substitution σ such that $u = l_i\sigma$ for some rule $l_i \longrightarrow r_i$ in R, and t is s with the subterm u replaced with $r_i\sigma$. We will drop the subscript $_R$ whenever it is immaterial or can be easily inferred. If $s \Longrightarrow_R t$ we say that s **rewrites to** t or that t **is directly deriveable** from s (or deriveable in one step). In system (I) above, for example, we have $s = \neg\neg(0 \wedge 1) \Longrightarrow (0 \wedge 1) = t$ since the substitution $\{x \mapsto (0 \wedge 1)\}$ makes the left-hand side of the first rule match s; t is obtained by applying the same substitution to the right-hand side of the same rule. Note that the same term s also rewrites to $t' = \neg(\neg 0 \vee \neg 1)$, since s's subterm $\neg(0 \wedge 1)$ matches the left-hand side of the third rule. Also note that t' further rewrites to $t'' = (\neg\neg 0 \wedge \neg\neg 1)$ (by the second rule) and that with two more applications of the first rule, t'' is reduced to $t = (0 \wedge 1)$.

A sequence of terms $t_1 \Longrightarrow_R t_2 \Longrightarrow_R t_3 \Longrightarrow_R \ldots$ (each of which rewrites to the next) is called a **derivation**. We write $s \Longrightarrow_R^+ t$ to indicate that t is deriveable from s in one or more steps (i.e $\Longrightarrow_R^+$ is the irreflexive transitive closure of $\Longrightarrow_R$). We write $s \Longrightarrow_R^* t$ to indicate that either $s = t$ or $s \Longrightarrow_R^+ t$ (i.e $\Longrightarrow_R^*$ is the reflexive transitive closure of $\Longrightarrow_R$). We use the symbol $\Longleftarrow_R$ to denote the inverse of $\Longrightarrow_R$, so that $t \Longleftarrow_R s$ iff $s \Longrightarrow_R t$. We write $\Longleftrightarrow_R$ for the union of $\Longrightarrow_R$ and $\Longleftarrow_R$ (i.e $s \Longleftrightarrow t$ iff $s \Longrightarrow_R t$ or $s \Longleftarrow_R t$). We write $\Longleftarrow_R^+$, $\Longleftarrow_R^*$, $\Longleftrightarrow_R^+$, and $\Longleftrightarrow_R^*$ for the irreflexive and reflexive transitive

[1] When t is a term, $Var(t)$ is the set of variable symbols that occur in t.

closures of $\Longleftarrow_R$ and $\Longleftrightarrow_R$, respectively.

If a term s does not rewrite to any terms (i.e if s is a $\Longrightarrow_R$—minimal element in $T(F,V)$), then we call s **irreducible**. If $s \Longrightarrow^* t$ and t is irreducible, we say that t is a **normal form** of s. In the above example, the term $(0 \wedge 1)$, being irreducible, is a normal form of the term $\neg\neg(0 \wedge 1)$.

Finally, a rewrite system R is called **terminating** iff there are no infinite derivations $t_1 \Longrightarrow_R t_2 \Longrightarrow_R t_3 \ldots$. Termination will be discussed in the second section of this paper.

Equational theories

An **equational system** E is a finite set of equations $\{s_1 = t_1, \ldots, s_n = t_n\}$ where the s_is and the t_is are terms over F and V (for some set of function symbols F). The semantics of equational systems can be developed in a manner similar to the semantics of first-order theories. In particular, we define an E–**interpretation** to be a pair $< D, \alpha >$, where D is a non-empty domain and α is a mapping which assigns (1) a function $f^I : D^n \rightarrow D$ to each function symbol $f \in F_n$ (to constants $f \in F_0$ α assigns individual elements of D), and (2) an element of D to each variable v. The **interpretation** $I(t)$ **of a term** $t \in T(F,V)$ is defined as follows:
(i) If t is a variable, then $I(t) = \alpha(t)$.
(ii) If $t = f(t_1, \ldots, t_n)$, then $I(t) = f^I(I(t_1), \ldots, I(t_n))$.
We say that I **satisfies** an equation $s = t$, or that $s = t$ is **true** in I, written as $I \models s = t$, iff $I(s) = I(t)$, i.e iff I makes s and t denote the same element in D. Consider, for example, the following equational system on $AR = \{0, s, +, *\}$:
$P = \{0 + x = x, s(x) + y = s(x + y), 0 * x = 0, s(x) * y = y + (x * y)\}$
If we let $I = (N, \alpha)$ be the customary interpretation of this system (where N is the set of the natural numbers and α assigns zero to 0, the successor function to s, etc.), then we have $I \models 0 + s(0) = s(0) * s(0)$, $I \models 0 + (0 * z) = 0$, $I \models s(0) * s(s(0)) = s(0) + s(0), I \models x + y = y + x$, etc. If E is a set of equations and I is an E– interpretation, we write $I \models E$ to indicate that I satisfies all of E's equations, in which case we say that I is a **model** of E. Given two sets of equations E_1 and E_2, we write $E_1 \models E_2$ to mean that every interpretation which is a model of E_1 is also a model of E_2. If $E_1 \models E_2$ we say that the equations in E_2 **follow logically from** or are **logical consequences** of E_1.

Given an equational system E, the fundamental problem is to determine whether a certain equation $s = t$ is a logical consequence of E, i.e whether $E \models \{s = t\}$. In the system P for example, we might want to know whether or not $P \models x+y = y+x$, i.e whether commutativity is a logical consequence of the way we have defined addition in P.[2] Usually it is not possible to prove the $E \models \{s = t\}$ directly, as we generally have to consider infinitely many possible interpretations. Just as in predicate calculus, what we need here is a syntactic notion $\vdash$ which will (hopefully) coincide with with $\models$ and which we can thus use to produce finite arguments (*proofs*) showing that $E \vdash s = t$ and, therefore (since $\vdash$ and $\models$ coincide), that $E \models s = t$ for various E, s,and t. The following **rules of inference** are used to define such a notion:

$$\textit{From } s = t \textit{ infer } s\sigma = t\sigma$$
$$\textit{From } s = t \textit{ infer } t = s$$
$$\textit{From } s = t \textit{ infer } f(\cdots s \cdots) = f(\cdots t \cdots)$$
$$\textit{From } s = t \textit{ and } t = u \textit{ infer } s = u$$
$$\textit{From True infer } s = s$$

We write $E \vdash s = t$ iff $s = t$ is deriveable from E using these rules, i.e iff there is a sequence of equations $e_1, e_2, \ldots, e_n$ where each e_i is either an equation in E or else it is obtained from earlier equations in the list by a single application of one of the above rules. Equivalently, $E \vdash s = t$ iff there is a sequence of terms $s = r_1, r_2, \ldots, r_n = t$, where each r_{i+1} is obtained from r_i by replacing a subterm u of r_i by another term v such that either $u = v$ or $v = u$ is an instance[3] of an equation in E. For example, the following establishes that $P \vdash (s(0) * s(0)) = s(0)$:
$s(0) * s(0) = s(0) + (0 * s(0)) = s(0) + 0 = s(0 + 0) = s(0)$.
The following classic theorem of Birchoff assures us that $\vdash$ and $\models$ are equivalent:

THEOREM 1.1.1 *For all equational systems E, $E \models s = t$ iff $E \vdash s = t$.*

This is similar to Godel's theorem of the completeness of the predicate calculus.

By utilizing some of the techniques of rewrite systems, one can, for some equational systems E, develop an algorithm which will determine, for arbitrary terms s and t, whether or not $E \vdash s = t$ and hence, by Birchoff's theorem, whether or not $s = t$ is a logical consequence of E. The following

[2] Incidentally, it is *not*.

[3] An **instance** of an equation $s = t$ is an equation $s\sigma = t\sigma$, where σ is any substitution.

sections explain the details.

Equational theories of rewrite systems

Let R be a rewrite system $\{l_1 \longrightarrow r_1, \ldots, l_p \longrightarrow r_p\}$ on a set of function symbols F. The **equational theory** of R is the equational system $E_R = \{l_1 = r_1, \ldots, l_p = r_p\}$. We observe the following:
LEMMA 1.1.1 *For all terms s and t, $E_R \vdash s = t$ iff $s \Longleftrightarrow_R^* t$.*
The verification of this is straight-forward: In the left–to–right direction, if $E_R \vdash s = t$ then there is a sequence of terms $s = r_1, r_2, \ldots, r_n = t$ where each r_{i+1} is obtained by replacing a subterm u of r_i with a term v such that either (i) the equation $u = v$ is an instance of an equation in E_R, or (ii) the equation $v = u$ is an instance of an equation in E_R. If (i) is the case, then by definition of E_R, we have $u \Longrightarrow_R v$, and hence $r_i \Longrightarrow_R r_{i+1}$. If (ii) is the case, we have $v \Longrightarrow_R u$, and hence $r_{i+1} \Longrightarrow_R r_i$, i.e $r_i \Longleftarrow_R r_{i+1}$.Hence, for all $i = 1, \ldots, n-1$ we have $r_i \Longleftrightarrow_R r_{i+1}$, so that $s = r_1 \Longleftrightarrow_R^* r_n = t$. The same type of reasoning can establish the converse direction.

In the next section we show that for certain rewrite systems R, namely those that are *Church–Rosser* and *terminating*, the $\Longleftrightarrow_R^*$ relation is recursive. For such R, lemma 1.5.1 in combination with Birchoff's theorem implies that the equational theory E_R has a decideable validity problem, i.e that there is an algorithm for deciding whether or not $E_R \models s = t$. In practice, given an equational system E we try to orient the equations of E into rewrite rules so as to obtain a rewrite system R which is Church-Rosser and terminating. If we succeed in doing this, then by the above remarks the validity problem of E_R (i.e of E) becomes mechanically decideable.

The Church-Rosser property, termination, and normal forms

Let a rewrite system R on F be given. We say that two terms s and t are **joinable**, written as $s \downarrow t$, iff there is a term u such that $s \Longrightarrow_R^* u$ and $t \Longrightarrow_R^* u$ (fig.1.2). For example, in the rewrite system given in pg. X., the terms $\neg\neg(0 \vee 1)$ and $\neg(\neg 0 \wedge \neg 1)$ are joinable (take u to be $(0 \vee 1)$).

A rewrite system R has the **Church-Rosser property**, abbreviated as "the CR property", iff for all s and t, $s \Longleftrightarrow_R^* t$ iff $s \downarrow t$. The CR property

is depicted in Fig. X. Systems which have the CR property are called CR systems.

Hence, for CR rewrite systems we can check whether $s \Longleftrightarrow^* t$ by searching the rewrite paths beginning with s and those beginning with t for a common term. Of course this is quite inefficient, but at least the bidirectionality of $\Longleftrightarrow^*$ has been eliminated. From now on we will only need to consider rewriting in one direction, from left to right.

We note the following important fact about rewrite systems:

LEMMA 1.1.2 *In a CR rewrite system every term has at most one normal form.*

For suppose that $s \Longrightarrow^* s_1$ and $s \Longrightarrow^* s_2$ where s_1 and s_2 are two distinct irreducible terms. Then, by definition of $\Longleftrightarrow$, we have $s_1 \Longleftrightarrow^* s_2$. But then the CR property implies that s_1 and s_2 are joinable, which means that either they are identical or else there is a term t such that $s_1 \Longrightarrow^+ t$ and $s_2 \Longrightarrow^+ t$; either case contradicts our assumption that s_1, s_2 are distinct and irreducible.

It could be, however, that a term s has *no* normal form in a CR system. This can obviously happen in a non-terminating system in which all derivations beginning with s are infinite. On the other hand, in a terminating rewrite system *every* s has *at least one* normal form. For either (1) there exist one or more derivations beginning with s or (2) not (i.e no rewrite rules apply to s). If there is a derivation, then by termination, it must have a finite length, i.e it must eventually stop at some irreducible term t, which is thus a normal form of s. And if there are no derivations beginning with s then s is irreducible; and since $s \Longrightarrow^* s$, s is a normal form of itself. We conclude the following:

LEMMA 1.1.3 *In a terminating rewrite system every term has at least one normal form.*

Of course in a terminating system a term might have *more than one* normal form. If, however, the system is *both* CR *and* terminating, then the two above lemmas imply that every term has *exactly one* normal form:

LEMMA 1.1.4 *In a terminating CR rewrite system every term has a unique normal form.*

We can use this lemma to prove the following:

LEMMA 1.1.5 *In a terminating CR rewrite system two terms s and t are joinable iff they have identical normal forms.*

The proof is not difficult. One direction (right–to–left) is obvious. In the

converse direction, suppose s and t are joinable, i.e there is a u such that $s \Longrightarrow^* u$ (1) and $t \Longrightarrow^* u$ (2). By lemma 1.1.4, u has a unique normal form u', i.e $u \Longrightarrow^* u'$ (3) and u' is irreducible. Now (1), (2), and (3) imply that $s \Longrightarrow^* u'$ and $t \Longrightarrow^* u'$, and since u' is irreducible, it is the unique (by the CR property) normal form of s and t.

Therefore, if a rewrite system is terminating and CR, there is a simple algorithm for determining whether or not $s \Longleftrightarrow^* t$: Start a derivation (any derivation) from s. In a finite number of steps (zero or more) this derivation *must* eventually lead us to s', the unique normal form of s. Then start a derivation from t to eventually obtain t', the unique normal form of t. Then simply check to see if $s' = t'$. If the equality holds then s and t are joinable so that, by the CR property, $s \Longleftrightarrow^* t$. On the other hand, if $s' \neq t'$ then the last lemma implies that s and t are not joinable, and the CR property entails that $s \not\Longleftrightarrow^* t$.

By lemma 1.1.1, the above is essentially an algorithm for deciding whether $E_R \models s = t$. Obviously, however, to be able to use this algorithm one must be certain that the rewrite system at hand is both CR and terminating. In the following sections we discuss methods for proving these two properties. It turns out that termination is generally undecidable; the CR property, on the other hand, *is* decidable, but only when the system is known to be terminating.

Confluence and local confluence

Let us write $s \uparrow t$ to indicate that both s and t are deriveable from another term r, i.e that there is a term r such that $r \Longrightarrow^* s$ and $r \Longrightarrow^* t$ (see fig. X). We say that a rewrite system is **confluent** iff for all terms s and t, $s \uparrow t$ implies $s \downarrow t$.

Further, let us define a family of binary prediacates $JOIN^k$ on terms as follows: $JOIN^1(s,t) \Leftrightarrow s \downarrow t$, and $JOIN^{k+1}(s,t) \Leftrightarrow$ *there is a term u such that $s \downarrow u$ and $JOIN^k(u,t)$*. Figures X and Y depict the relation between s and t when $JOIN^2(s,t)$ and $JOIN^3(s,t)$ hold, respectively. In general, $JOIN^k(s,t)$ holds whenever there are k consequtive *valleys* (and thus $k-1$ consequtive *peaks*) between s and t. We can now prove the following:

LEMMA 1.1.6 *If R is a confluent rewrite system, then for all terms s, t and integers $k > 0$, $JOIN^k(s,t)$ implies that s and t are joinable, i.e that $s \downarrow t$.*

Proof. By induction on k. For $k = 1$ the lemma holds by definition of

$JOIN^k$. Assume now that $JOIN^{k+1}(s,t)$, so that there is a term u such that $s \downarrow u$ (1) and $JOIN^k(u,t)$. From the induction hypothesis we have $u \downarrow t$ (2). Now (1) and (2) imply that there are terms v_1 and v_2 such that $s \Longrightarrow_R^* v_1$ (3), $u \Longrightarrow_R^* v_1$, and $u \Longrightarrow_R^* v_2$, $t \Longrightarrow_R^* v_2$ (4). Hence, since $u \Longrightarrow_R^* v_1$, $u \Longrightarrow_R^* v_2$ and R is confluent, v_1 and v_2 are joinable, i.e there is a term v_3 such that $v_1 \Longrightarrow_R^* v_3$ (5) and $v_2 \Longrightarrow_R^* v_3$ (6). Finally, $\{(3),(5)\}$ and $\{(4),(6)\}$ respectively entail that $s \Longrightarrow_R^* v_3$ and $t \Longrightarrow_R^* v_3$, i.e that s and t are joinable. $\Box$

Finally, we can prove that confluence and the CR property are equivalent:

THEOREM 1.1.2 *A term rewrite system R has the CR property iff it is confluent.*

Proof. The left–to–right direction is trivial. In the other direction, assume that R is confluent. We must show that for all s and t, $s \Longleftrightarrow_R^* t$ iff $s \downarrow t$. The if part of this is obvious. For the only if part, assume that $s \Longleftrightarrow_R^* t$. Then either (1) $s = t$, or else (2) there are terms $r_0, r_1, \ldots, r_n$, $n \geq 0$ such that $r_0 = s \Longleftrightarrow r_1 \Longleftrightarrow \cdots \Longleftrightarrow r_{n+1} = t$. In the first case, s and t are trivially joinable. In the second case note that for any terms u and v, $u \Longleftrightarrow v$ implies $u \downarrow v$, since either $u \Longrightarrow^* v$ (and $v \Longrightarrow^* v$) or $v \Longrightarrow^* u$ (and $u \Longrightarrow^* u$). Hence we have $r_i \downarrow r_{i+1}$ for all $i = 0, \ldots, n$ in the above chain, so that $JOIN^{n+1}$ holds. But then confluence and the preceding lemma imply that s and t are joinable. $\Box$.

Therefore, instead of checking directly for the CR property (which involves the inefficient bi-directionality of $\Longleftrightarrow^*$ chains), we can check for confluence. Detecting confluence is greatly simplified when the system is terminating. For such systems we need only be concerned with a simpler, "localized" type of confluence. In particular, let us call a rewrite system **locally confluent** iff for all r, s, and t, if $r \Longrightarrow s$ and $r \Longrightarrow t$, then s and t are joinable. The crucial difference with confluence is that *only one* step is taken from r rather than arbitrarily many. The following theorem assures us that when the system is terminating we need only check for local confluence :

THEOREM 1.1.3 *If a rewrite system R is locally confluent and terminating, then it is confluent*

Proof. First note that if R is terminating the $\Longrightarrow_R^+$ relation (which is a partial order) is well-founded. In what follows we extend the partial order on terms $\Longrightarrow_R^+$ to a relation $>>$ on multisets of terms (in the manner explained in the Appendix). From the theorem proved in the Appendix and the above remark it follows that if R is terminating, the $>>$ extension of $\Longrightarrow_R^+$ is a

well-founded partial order on multisets of terms.

We now proceed with the proof. We show that on the assumption of termination and local confluence,, R is CR (and thus, by the previous theorem, confluent). Assume then that $s \Longleftrightarrow^* t$ (we drop the subscript $_R$ for notational simplicity). To prove the CR property we must show that s and t are joinable.[4] Now the fact that $s \Longleftrightarrow^* t$ means that there are $n > 0$ terms $r_1, \ldots, r_n$ (not necessarily all distinct) such that $s = r_1 \Longleftrightarrow r_2 \Longleftrightarrow \ldots \Longleftrightarrow r_n = t$. Call this chain C. Let T be the multiset of terms appearing in C (i.e $T = \{r_1, \ldots, r_n\}$). Now in the chain C call a triple of adjacent terms $< r_{i-1}, r_i, r_{i+1} >$ a *peak* if $r_{i-1} \Longleftarrow r_i \Longrightarrow r_{i+1}$. Clearly, if there are no peaks in C then s and t are joinable, so let $< r_{i-1}, r_i, r_{i+1} >$ be such a peak. Since the system is locally confluent, $r_{i-1} \downarrow r_{i+1}$, i.e there are chains $c' : r_{i-1} \Longrightarrow^* u$ and $c'' : r_{i+1} \Longrightarrow^* u$ for some term u. Now consider the chain

$$C_1 : s = r_1, \ldots, \overbrace{r_{i-1}, \ldots, u}^{c'}, \overbrace{\ldots, r_{i+1}}^{c''}, \ldots, r_n = t$$

obtained from C by replacing the $r_{i-1} \Longleftarrow r_i$ and $r_i \Longrightarrow r_{i+1}$ steps with c' and c'' respectively (fig. X). C_1 is still a $\Longleftrightarrow^*$ chain connecting s and t but with the $< r_{i-1}, r_i, r_{i+1} >$ peak removed (of course this removal might have introduced additional peaks). Now let T_1 be the set of terms appearing in C_1. Since the removed term r_i is $\Longrightarrow^+$– greater than all of its replacements, namely all the terms appearing in c' and c'', we have $T >> T_1$.

Now we can repeat the same process with the C_1 chain, i.e we can remove a peak from it to obtain a new chain C_2, where T_2, the multiset of terms in C_2, is $>>$– less than T_1. In general, as long as there are peaks in C_i we can get a new chain C_{i+1} such that $T_i >> T_{i+1}$. But since $>>$ is well-founded, the descending sequence $T >> T_1 >> T_2 >> \cdots$ must stop at some point. At that point there are no peaks left in the chain connecting s and t, which means that $s \downarrow t$. $\Box$

Checking for local confluence: Critical pairs

...

[4] The converse is obvious.

Termination

Following convention, in this section we will only be concerned with the restriction of $\Longrightarrow_R$ to *ground* terms. Therefore, whenever we write $s \Longrightarrow_R t$, it should be inferred that s and t are ground terms (unless we explicitly say otherwise).

There are several techniques for proving that a rewrite system R is terminating. The main idea behind many of them is to try to associate with each term t some sort of numerical quantity $Q(t)$ in such a way that $s \Longrightarrow_R t$ will imply that $Q(s) > Q(t)$ (1). If the "quantity" Q is always non-negative then condition (1) guarantees that R is terminating. For suppose that there was an infinite derivation $t_1 \Longrightarrow_R t_2 \Longrightarrow_R t_3 \Longrightarrow_R \cdots$ (2). Then, by condition (1), $Q(t_1) > Q(t_2) > Q(t_3) > \cdots$ (3). But since the values of Q are non-negative integers (natural numbers), infinite chains such as (3) cannot exist because a sequence of natural numbers cannot decrease *ad infinitum*– the "lowest" it can go is zero. This contradiction enables us to conclude that infinite derivations like (2) are not possible, which means that R is terminating.

Q could be anything as long as it has non-negative values. In some simple cases $Q(t)$ might just be the size of t (in which case it is obviously non-negative). Suppose, for example, that we had a system on $\{\neg, \vee, 0, 1\}$ which consisted of only two rules:

$$\neg\neg x \longrightarrow x$$
$$x \vee 0 \longrightarrow x$$

It is easy to see that each rule reduces the size of whatever term it is applied to. Therefore, if we let $Q(t)$ be the size of t, we ensure that $s \Longrightarrow t$ implies $Q(s) > Q(t)$ for all ground terms s and t. It follows that infinite derivations are not possible in this system.

Of course if we decide to let $Q(t)$ be the size of t we cannot conclude that (1) is satisfied simply by looking at each rule $l_i \longrightarrow r_i$ and verifying that the size of r_i is smaller than the size of l_i. What we must verify is that each rule decreases the size of every *ground* term it applies to. For instance, let $F = \{0, 1, f, g\}$ and let R comprise only the following rule:

$$f(x, y, z) \longrightarrow g(x, x)$$

Although it looks as if it does, this rule does *not* in fact decrease the size of all ground terms to which we can apply it. The reason is that the occurences of x double on the right-hand side, so if x happens to stand for a large

ground term, applying the rule to a superterm t of it will increase rather than decrease t's size (e.g take t to be $f(g(0,0),1,1)$). So taking Q to be term size will not do the trick in this case. Indeed, most rewrite systems have both size-increasing and size-decreasing rules, so, as a numerical measure, term size is not widely applicable.

The idea of proving termination through the use of non-negative quantities that decrease at each step of the computation can be applied to all sorts of discrete computational processes, not just rewrite systems. To take an example, suppose we have a yard Y full of trains, where each train consists of one or more linked cars, and we want to empty Y (remove all the trains). Here is a very simple algorithm to do this:

```
WHILE (Y is not empty) DO {
   * Pick a train T.
   * If T has only one car, remove it from the yard.
   * Otherwise split T into two shorter trains.
}
```

Note that this algorithm is non-deterministic: it does not tell us *which* train to pick or *how* to split a train.

Intuitively it is obvious that the algorithm halts for all "input" yards Y. We can prove that formally by assigning to each yard Y the quantity $Q(Y) = 2c(Y) - t(Y)$, where $c(Y)$ and $t(Y)$ are, respectively, the numbers of cars and trains in Y. Since for all Y we have $t(Y) \leq c(Y)$ (we cannot have more trains than cars), it follows that $c(Y) - t(Y) \geq 0$, and since $c(Y) \geq 0$ we have $2c(Y) - t(Y) = Q(Y) \geq 0$, i.e Q is always non-negative (it is zero iff Y is empty). All we now have to do is show that Q decreases with each iteration of the WHILE loop. This is true because on each iteration eaxctly one of two things happens: either (a) an 1–car train is removed from Y, in which case $2c(Y)$ is decreased by 2 and $-t(Y)$ is increased by one, so that Q decreases by one, or (b) a train is split into two trains, in which case $2c(Y)$ stays the same while $t(Y)$ increases by one, hence Q again decreases by one.

As another example consider the "Dutch flag" problem. Suppose we are given as input a sequence S of adjacent marbles, where each marble is either red or white or blue. Computation proceeds by re-arranging pairs of adjacent marbles in accordance with the following rules:

(1) White, Red $\longrightarrow$ *Red, White*
(2) Blue, Red $\longrightarrow$ *Red, Blue*
(3) Blue, White $\longrightarrow$ *White, Blue*

The meaning of the first rule is that if anywhere in S there is a white marble immediately followed by a red marble, we can exchange the two by placing the red marble on the left and the white one on the right. The other two rules have similar interpretations. The object is to arrange the marbles of S in a Red—White—Blue pattern, i.e in such a way that all the red marbles come first (leftmost), followed by all the white marbles, followed by all and only the blue marbles. This problem is trivially reducible to numerical sorting if we only let red marbles denote, say the number one, white marbles the number two, and white marbles the number three (or any other greater number).

Note that as long as the marbles are not in the desired order at least one of rules (1) – (3) will be applicable. For, the marbles are in the correct order iff

(i) No blue marble has any red or white marbles to its right, and

(ii) No white marble has any red marbles to its right.

If either (i) or (ii) does not hold, one of rules (1) – (3) applies (and vice-versa). Therefore, if no rule applies then both (i) and (ii) are satisfied and the sequence is properly sorted. Thus we know that *if* the process of re-arranging the marbles of S according to the three rules ever comes to a halt, *then* the marbles of S will be in correct order. But *is* computation guaranteed to stop for all input sequences? Intuitively we can see that this is indeed the case, but we can also give a formal proof by capitalizing on the resemblance of the problem with numerical sorting and introducing a total order $<$ on the colors as follows: *Red* $<$ *White* $<$ *Blue*. We can then assign to each sequence $S = [m_1, \ldots, m_k]$ the non-negative quantity $Q(S) =$ *the number of* **inversions** *in* S, an inversion being a pair of marbles m_i, m_j where $i < j$ $(1 \leq i, j \leq k)$, such that $COLOR(m_i) < COLOR(m_j)$[5]. $Q(S)$ can be easily computed by summing up for each $i = 1, 2, \ldots, k$ the number of marbles to the left of m_i having a color greater than that of m_i. Clearly, $Q(s) = 0$ iff S is in the correct order. The key step now is simply to observe that an application of any one of the three rules removes exactly one inversion and does not engender any new ones. In other words $Q(S)$ is decreased at each step of the computation; thus termination is guaranteed.

Generally speaking, Q does not have to be a numerical quantity. Its values could be anything as long as they comprise a well-founded set. If we

[5]This concept is also borrowed from numerical sorting.

view Q as a function whose domain consists of the objects manipulated by the computation at hand, then all we ask of it is a well-founded range. For example, instead of assigning to each sequence $S = [m_1, \ldots, m_k]$ the number of its inversions, we could assign to it the *sequence of colors* $[c_1, \ldots, c_k]$, where c_i is the color of m_i. Since the set of colors $C = \{$ *Red, White, Blue* $\}$ is well-founded under the ordering *Red* $<$ *White* $<$ *Blue* , the set $SEQ(C)$ of all finite sequences $[c_1, \ldots, c_k]$ of these colors ($k > 0$) is well-founded under $<_{lex}$, the lexicographic extension of $<$. Thus we can let Q map sequences of marbles $S = [m_1, \ldots, m_k]$ to elements of $SEQ(C)$ (namely $Q([m_1, \ldots, m_k]) = [c_1, \ldots, c_k]$, where $c_i = COLOR(m_i)$). If we apply any one of the three rules to a sequence S the value $Q(S)$ will decrease under the $<_{lex}$ ordering (because the sequence S' we obtain from S by flipping two adjacent marbles m_i, m_{i+1} has identical elements $m_1, \ldots, m_{i-1}$ with S, while the c_i of S' is "less than" the c_i of S). Termination is again established owing to the fact that we cannot have an infinite $>_{lex}-$ descending chain of elements of $SEQ(C)$.

Importing these ideas into the rewrite system platform, we conclude the following:

THEOREM 1.1.1 *A rewrite system R is terminating if there is a total function Q from the ground terms to a well-founded set $(W, \succ)$ such that for all s and t,*

$$s \Longrightarrow_R t \ \ implies \ \ Q(s) \succ Q(t)$$

From now on we will frequently refer to Q as the "termination function". An alternative but equivalent phrasing of the above theorem is the following:

THEOREM 1.1.2 *A rewrite system R is terminating if there is a well-founded partial order $\Re$ on the ground terms which includes the $\Longrightarrow_R$ relation, i.e such that*

$$s \Longrightarrow_R t \ \ implies \ \ s \Re t$$

The antecedents of these two theorems are equivalent: if a function Q of the kind described by the first theorem exists, then the $\Re$ of theorem 2 exists and can be defined as: $s \Re t \ \ iff \ \ Q(s) \succ Q(t)$. And if the $\Re$ of theorem 2 exists, then we can simply let the W of theorem 1 be the set of ground terms and Q be the identity function. Hence neither antecedent is true if the other is not, and if either one *is* true, then we can infer the common consequent of the two theorems: that R is terminating.

As has already been remarked, the range of Q is often N, the natural numbers. Sometimes the number $Q(t)$ is the value of a polynomial function we have associated with t. This is representative of a class of methods known as **polynomial interpretations**. Such interpretations assign an integer polynomial function P_f of degree n to each function symbol $f \in F_n$ (so if f is a constant, P_f is a natural number). Then, given a ground term $t = f(t_1, \ldots, t_n)$ $(n \geq 0)$, the value $Q(t)$ is recursively calculated as follows:

$$Q(t) = P_f(Q(t_1), \ldots, Q(t_n))$$

The recursion stops at the "base level" of constants.

The polynomials P_f must be chosen in a way that will satisfy two conditions, one local and the other global:
(1) Local condition: For every rule $l_i \longrightarrow r_i$ and every substitution σ that replaces the variables of l_i and r_i with ground terms, $Q(l_i\sigma) > Q(r_i\sigma)$.
(2) Global condition: For all ground terms s and t and function symbols f, if $Q(s) > Q(t)$ then $Q(f(\cdots s \cdots)) > Q(f(\cdots t \cdots))$.

It is easy to see that if both of these conditions are satisfied then the condition "$s \Longrightarrow_R t$ implies $Q(s) > Q(t)$" is also satisfied, for all ground terms s and t. For, if $s \Longrightarrow_R t$, there is a subterm u of s, a rule $l_i \longrightarrow r_i$ in R and a ground substitution σ such that $u = l_i\sigma$ and t is s with u replaced by $v = r_i\sigma$. From the local condition it follows that $Q(u) > Q(v)$; and from the global condition it follows that $Q(s = f(\cdots u \cdots)) > Q(t = f(\cdots v \cdots))$, where f is the outermost (root) function symbol of s. Furthermore, note that the global condition can be ensured by choosing all polynomials P_f to be *monotonic*, i.e such that for all numbers x and x', $x' > x$ implies $P_f(\cdots x' \cdots) > P_f(\cdots x \cdots)$. To see this, suppose that all P_f are monotonic and that for some s and t we have $Q(s) > Q(t)$. Then for any function symbol f, we have

$$Q(f(\cdots s \cdots)) = P_f(\cdots Q(s) \cdots) > Q(f(\cdots t \cdots)) = P_f(\cdots Q(t) \cdots)$$

i.e condition (2) is satisfied in virtue of the monotonicity of P_f. A simple way to ensure that P_f is monotonic is to choose all of its coefficients to be positive numbers. We now present some examples of polynomial interpretations.
EXAMPLE 1 Consider the following one-rule system on $\{\vee, 0, 1\}$:

$$(x \vee y) \vee z \longrightarrow x \vee (y \vee z)$$

Here is a simple polynomial interpretation which proves that this system terminates : Let $P_0 = P_1 = 1$ and let $P_\vee(n,m) = 2n + m$ (for all natural numbers n and m). To see that the local condition is satisfied, let σ be a ground substitution such that $s = l_1\sigma$ and $t = r_1\sigma$ for some ground terms s and t. Thus s must be of the form $(s_1 \vee s_2) \vee s_3$ for some ground terms s_1, s_2, s_3 and t must be $s_1 \vee (s_2 \vee s_3)$. Then

$$\begin{aligned} Q(s) &= P_\vee(Q(s_1 \vee s_2), Q(s_3)) \\ &= 2Q(s_1 \vee s_2) + Q(s_3) \\ &= 2[P_\vee(Q(s_1), Q(s_2))] + Q(s_3) \\ &= 2[2Q(s_1) + Q(s_2)] + Q(s_3) \\ &= 4Q(s_1) + 2Q(s_2) + Q(s_3) \qquad (1) \end{aligned}$$

For t we have

$$\begin{aligned} Q(t) &= P_\vee(Q(s_1), Q(s_2 \vee s_3)) \\ &= 2Q(s_1) + Q(s_2 \vee s_3) \\ &= 2Q(s_1) + 2Q(s_2) + Q(s_3) \qquad (2) \end{aligned}$$

By the way we have chosen P_0, P_1, and $P_\vee$, the value $Q(r)$ of *any* ground term r must be at least one. Hence $4Q(s_1) > 2Q(s_1)$ and a term–by–term comparison of lines (0.1) and (0.2) above shows that $Q(s) > Q(t)$. Moreover, the global condition is satisfied because P_f is monotonic. Therefore, for *all* ground terms s and t, $s \Longrightarrow t$ implies $Q(s) > Q(t)$ and the system is terminating. □

EXAMPLE 2 Consider the following system on the ground terms built from the binary operators $\cdot$, $+$ and the constants a, b, c, d, and e:

$$\begin{aligned} &x \cdot (y + z) \longrightarrow (x \cdot y) + (x \cdot z) \\ &(y + z) \cdot x \longrightarrow (y \cdot x) + (z \cdot x) \\ &(x + y) + z \longrightarrow x + (y + z) \end{aligned}$$

Our interpretation for this system is similar to that of the previous example. For the constants we let $P_a = P_b = P_c = P_d = P_e = 2$ and to the two binary operators $+$ and $\cdot$ we assign the monotonic polynomials $P_+(n,m) = 2n + m + 1$ and $P_\cdot(n,m) = nm$. Let s and t be any two ground terms of the form $s_1 \cdot (s_2 + s_3)$ and $(s_1 \cdot s_2) + (s_1 \cdot s_3)$, respectively. Then

$$Q(s) = Q(s_1)Q(s_2 + s_3)$$

$$
\begin{aligned}
&= Q(s_1)[2Q(s_2) + Q(s_3) + 1] \\
&= 2Q(s_1)Q(s_2) + Q(s_1)Q(s_3) + Q(s_1)
\end{aligned}
$$

while

$$
\begin{aligned}
Q(t) &= 2Q(s_1 \cdot s_2) + Q(s_1 \cdot s_3) + 1 \\
&= 2Q(s_1)Q(s_2) + Q(s_1)Q(s_3) + 1
\end{aligned}
$$

Since all ground terms are greater that 1, $Q(s) > Q(t)$. This shows that the first rule satisfies the local condition. Equally simple calculations show the same for the other two rules. Finally, since all of the chosen polynomials are monotonic, the global condition is satisfied as well. Thus the system is terminating. $\Box$

Polynomial interpretations are special cases of a broader class of methods known as **simplification orderings**. Let $T(F)$ be a set of ground terms over a set of function symbols F and let $\succ$ be any partial order on $T(F)$. We say that $\succ$ is a *simplification ordering* iff it has the following two properties:
(1) The subterm property: Every term is greater than each of its subterms, i.e

$$\text{for all } t \text{ and } f,\ f(\cdots t \cdots) \succ t.$$

(2) The replacement property: For all s, t, and f,

$$s \succ t \text{ implies } f(\cdots s \cdots) \succ f(\cdots t \cdots).$$

If $\succ$ has the replacement property, we call it **monotonic**.

It is easy to see that polynomial interpretations "are" simpification orderings. The partial order in such cases is the relation $\succ_Q$ that is induced on $T(F)$ by a polynomial–defined termination function Q as follows : $s \succ_Q t$ *iff* $Q(s) > Q(t)$. Clearly, if the polynomials which are used to define Q are monotonic and have positive coefficients, $\succ_Q$ is a simplification ordering.

A useful property of simplification orderings $\succ$ is that they are always in congruity with syntactic simplicity, meaning that if a term s is syntactically "simpler" (less complex) than another term t, then $t \succeq s$ (where $\succeq$ is the reflexive closure of $>$). This can be made more precise if we give a formal definition of the "syntactically simpler" concept. We do that as follows. We say that a ground term $s = f(s_1, \ldots, s_m)$ is "simpler" than a ground term $t = g(t_1, \ldots, t_n)$, written as $s \sqsubseteq t$, iff:

- either $s \sqsubseteq t_i$ for some $i \in \{1, \ldots, n\}$, or

- $f = g$ and $s_i \sqsubseteq t_i$ for all $i = 1, 2, \ldots, m = n$.

This relation is also known as the "homeomorphic embedding" relation.

To take an example, $0 \vee 1 \sqsubseteq \neg 0 \vee \neg 1$ since 0 is embedded in $\neg 0$ and 1 is embedded in $\neg 1$, so that the second clause of the definition entails that the disjunction of 0 and 1 is "simpler" than the disjunction of $\neg 0$ and $\neg 1$. Note that $c \sqsubseteq c$ for all constants c by clause 2, because the second condition specified by that clause is vacuously true by virtue of c having no proper subterms. As a result, $t \sqsubseteq t$ for *all* ground terms t, which can be shown by repeatedly applying the second clause until we reach the "leaves" of t (its constants). From this we can further infer that a term is simpler than any of its superterms (by clause 1). In general, s is simpler than t iff the latter can be obtained from s by inserting function symbols at appropriately selected places (viewing terms as trees).

We can now make our aforementioned claim precise in the following manner:

THEOREM 1.2.3 *If $\succ$ is a simplification ordering on a set of ground terms $T(F)$, then $\preceq$ [6] contains the homeomorphic embedding relation, i.e*

for all s and t in $T(F)$, $s \sqsubseteq t$ implies $s \preceq t$.

Proof. By strong induction on the size of t. When t has size one it is a constant, so if $s \sqsubseteq t$ then $s = t$ and $s \preceq t$. Now assume that for all terms of size $\leq k$ (for some $k > 0$) and all terms s we have $s \sqsubseteq t \Rightarrow s \preceq t$ (IND). Let t be any term of size $k + 1$ and let s be any term such that $s = f(s_1, \ldots, s_m) \sqsubseteq t = g(t_1, \ldots, t_n)$. By definition of $\sqsubseteq$, there are two cases:

- $s \sqsubseteq t_j$ for some $j \in \{1, \ldots, n\}$. In that case (IND) implies that $s \preceq t_j$ and since $t_j \preceq g(t_1 \cdots t_j \cdots t_n) = t$, the transitivity of $\preceq$ entails that $s \preceq t$.

- $f = g$ and $s_i \sqsubseteq t_i$ for $i = 1, \ldots, n = m$. By (IND), $s_i \preceq t_i$ for all $i \in \{1, \ldots, n\}$. Thus $s_1 \preceq t_1$ and, by the subterm property,
$$f(s_1, t_2, \ldots, t_n) \preceq f(t_1, t_2, \ldots, t_n) \qquad (1)$$
Now $s_2 \preceq t_2$, so, by the same property,
$$f(s_1, s_2, t_3, \ldots, t_n) \preceq f(s_1, t_2, t_3, \ldots, t_n) \qquad (2)$$
Proceeding in the same fashion, we deduce $n - 2$ more inequations of this form, the last one being

[6] $x \prec y$ simply means $y \succ x$. And $x \preceq y$ means $x \prec y$ *or* $x = y$.

$$f(s_1, \ldots, s_n) \preceq f(s_1, \ldots, s_{n-1}, t_n) \quad (n)$$

Then from inequations (1) – (n) and transitivity we conclude that $s = f(s_1, \ldots, s_n) \preceq f(t_1, \ldots, t_n) = t$.

By induction, the result holds for all t. $\square$

Furthermore, it turns out that in any infinite sequence of terms built from a finite set of function symbols there must be two terms s and t, occupying different places in the sequence, such that $s \sqsubseteq t$ (a result proved by Kruskal). It follows that homeomorphic embedding is a necessary condition for non-termination. For if R does not terminate there is an infinite sequence of terms $u_1 \Longrightarrow_R u_2 \Longrightarrow_R u_3 \Longrightarrow_R \ldots$ Since the function symbols that appear in this derivation are finitely many (they form a subset of the function symbols of u_1 and the function symbols in the rules $l_i \longrightarrow r_i$), Kruskal's result applies and entails that there are terms t_i and t_j, $i < j$, such that $t_i \sqsubseteq t_j$. We call the above derivation "self-embedding" on account of that. We conclude:

THEOREM 1.2.4. *If a rewrite system R is not terminating, then it has an infinite self-embedding derivation.*

Now suppose that, given a rewrite system R on some F, we manage to construct a simplification ordering $\succ$ on $T(F)$ that contains the $\Longrightarrow_R$ relation, i.e such that $s \Longrightarrow_R t$ implies $s \succ t$. That would be tantamount to proving that R is terminating. For, by way of contradiction, assume that there existed an infinite derivation $t_1 \Longrightarrow_R t_2 \Longrightarrow_R \ldots$ Then, by the discussion in the previous paragraph, this derivation contains two terms t_i and t_j, $i < j$, such that $t_i \sqsubseteq t_j$. Now theorem 1.2.3 implies that $t_j \succeq t_i$ (A). And since $\succ$ contains $\Longrightarrow_R$, we must also have $t_1 \succ t_2 \succ t_3 \succ \ldots$, and by transitivity, $t_i \succ t_j$ (b). But (A) and (B) are inconsistent with $\succ$ being a partial order, so our assumption that R is not terminating must be incorrect.

When constructing a simplification ordering for the purpose of showing that R is terminating one does not need to check that $s \succ t$ for *all* s and t such that $s \Longrightarrow_R t$. Rather, it is sufficient to check that $l_i\sigma \succ r_i\sigma$ for the rules $l_i \longrightarrow r_i$ (where σ is any substitution that replaces the variables of l_i and r_i by ground terms). The replacement property of $\succ$ will do the rest. In particular, if $s \Longrightarrow_R t$ then for some subterms u of s and v of t we have $u = l_i\sigma$ and $v = r_i\sigma$, for some ground substitution σ and rule $l_i \longrightarrow r_i$. If we have made sure that $u = l_i\sigma \succ r_i\sigma = v$, then because $\succ$ is monotonic and t is just s with u replaced by v, we must also have $s \succ t$. In summary we conclude the following:

THEOREM 1.2.5. *A rewrite system R on F is terminating if there is a simplification ordering $\succ$ on $T(F)$ such that $l_i\sigma \succ r_i\sigma$ for every rule $l_i \longrightarrow r_i$ in R and every substitution σ that replaces the variables of l_i and r_i with ground terms.*

As was already pointed out, polynomial interpretations give rise to simplification orderings. Polynomial interpretations, however, are inadequate for proving the termination of various systems which are in fact terminating. A more powerful class of simplification orderings are the **recursive path orderings**, abbreviated as rpos. An rpo is based on a partial ordering $\succ$ of the function symbols in F. Given such an ordering $\succ$, we define the *recursive path ordering* $\succ^*$ on $T(F)$ as follows:

$$s = f(s_1, \ldots, s_m) \succ^* t = g(t_1, \ldots, t_n) \text{ iff}$$

either (1) $f = g$ and $\{s_1, \ldots, s_m\} \succ\succ^* \{t_1, \ldots, t_{n=m}\}$,
or (2) $f \succ g$ and $s \succ^* t_i$ for $i = 1, 2, \ldots, n$,
or (3) $f \not\succ g$ and $f \neq g$ and $s_i \succ^* t$ for $i = 1, \ldots, m$
where $\succ\succ^*$ is the extension of $\succ^*$ to multisets over $T(F)$.

Therefore, to determine whether or not $s \succ^* t$ for some given s and t, the first thing we must do is compare the outermost function symbols of s and t. Exactly one of conditions (1), (2), (3) above is bound to hold. Depending on what the case actually is, we continue the test recursively with the proper subterms of s and t and in accordance with the rules provided in the three clauses of the definition. The following can be proved:

THEOREM 1.2.6. *All recursive path orderings are simplification orderings.*

As was already said, rpos can handle several systems which cannot be shown to be terminating with polynomial interpretations. An example of such a system is the one that was presented on page X., on the operators $\{\wedge, \vee, \neg, 0, 1\}$. We repeat the rules here for convenience:

$$\begin{aligned}
&\neg\neg x \longrightarrow x \\
&\neg(x \vee y) \longrightarrow (\neg x \wedge \neg y) \\
&\neg(x \wedge y) \longrightarrow (\neg x \vee \neg y) \\
&x \wedge (y \vee z) \longrightarrow (x \wedge y) \vee (x \wedge z) \\
&(y \vee z) \wedge x \longrightarrow (y \wedge x) \vee (z \wedge x)
\end{aligned}$$

Now consider the rpo $\succ^*$ arising from ordering the non-constant function symbols as follows: $\neg \succ \wedge \succ \vee$. By the above theorem, $\succ^*$ is a simplification ordering. Consequently, all we have to do is verify that $l_i\sigma \succ^* r_i\sigma$ for all σ that make l_i and r_i ground. The first rule is trivial because for all $x \in T(F)$,

$\neg\neg x$ is a superterm of x, hence, by the subterm condition for simplification orderings, $\neg\neg x \succ^* x$ for all ground terms x. For the second rule, we must prove that for all ground terms s_1 and s_2 we have $\neg(s_1 \vee s_2) \succ^* (\neg s_1 \wedge \neg s_2)$. Since $\neg \succ \wedge$, the above will hold iff $\neg(s_1 \vee s_2) \succ^* \neg s_1$ (i) and $\neg(s_1 \vee s_2) \succ^* \neg s_2$ (ii) – by definition of $\succ^*$. Both (i) and (ii) hold because $\neg s_1$ and $\neg s_2$ are homeomorphically embedded in $\neg(s_1 \vee s_2)$ and $\succ^*$, being a simplification ordering, contains the $\sqsubseteq$ relation. The same reasoning establishes that the third rule is a reduction. For the fourth rule, since $\wedge \succ \vee$, we must have $s_1 \wedge (s_2 \vee s_3) \succ^* s_1 \wedge s_2$ and $s_1 \wedge (s_2 \vee s_3) \succ^* s_1 \wedge s_3$ for all ground terms s_1, s_2, and s_3. Once again that holds because $s_1 \wedge s_2$ and $s_1 \wedge s_3$ are syntactically simpler than $s_1 \wedge (s_2 \vee s_3)$ and $\succ^*$ is a simplification ordering. And similar reasoning can be employed to derive the same conclusion with regard to the fifth and last rule. Termination is thus established by theorem 1.2.5.

Rpos are natural choices for rewrite systems defining primitive recursive functions. Primitive recursive definitions are incremental in a sense, meaning that we begin with some "initial" functions, then we define a function f_1 in terms of the initial functions, then a function f_2 in terms of f_1, and so forth. This suggests the following natural ordering $\prec$:

$$\textit{Initial functions} \quad \prec \quad f_1 \prec \quad f_2 \quad \prec \cdots$$

We can then define an rpo $\succ^*$ on top of this ordering $\prec$. For example, consider the following system defining the factorial function:

$$\begin{aligned}
&s(x)! \longrightarrow s(x) * x! \\
&0! \longrightarrow s(0) \\
&0 * x \longrightarrow 0 \\
&s(x) * y \longrightarrow y + (x * y) \\
&0 + x \longrightarrow x \\
&s(x) + y \longrightarrow s(x + y)
\end{aligned}$$

Let the precedence of the function symbols be $! \succ * \succ + \succ s \succ 0$. Then it is easy to see that under the rpo $\succ^*$ arising from $\succ$, every rule is a reduction. This is so because the rules, being primitive recursive, co-operate very well with clause (2) of the definition of rpos. For the first rule, for example, we clearly have $s(t)! \succ^* s(t) * t!$ for any ground term t because $! \succ *$ and $s(t)! \succ^* s(t),\ s(t)! \succ^* t!$. Similar considerations establish similar conclusions for the rest of the rules.

Despite their power and flexibility, rpos cannot handle all terminating systems. To take a simple example, suppose we had a terminating system with an associativity rule $f(f(x, y), z) \longrightarrow f(x, f(y, z))$. This rule cannot be

shown to be a reduction by any rpo, because, since $f = f$, for the left side to be rpo–greater than the right side we would have to have $\{f(x, y), z\} \succ\succ^* \{x, f(y, z)\}$ for all ground values of x, y, z — which is not the case (take $x = y = z$ for instance). A different class of orderings which can handle associativity, as well as some other rules for which rpos are inadequate, can be found in the so-called **lexicographic path orderings**.

.
.
.

Appendix A

Multisets

Informally, a **multiset** M **over a set** D is a set containing elements of D, the difference with conventional sets being that an element might occur more than once in M. D is called the **underlying domain** of M. Two examples of multisets over N (the set of natural numbers) are $M_1 = \{0, 0, 1, 3, 4, 4\}$ and $M_2 = \{5, 6\}$. M_1 contains two occurences of 0, one of 1, none of 2, one of 3, two of 4, and zero of all numbers $n > 4$. As M_2 shows, a traditional set is a multiset all of whose elements occur exactly once.

Formally, a multiset can be viewed as an ordered pair $< D, m >$ where D is the underlying domain and m is a total function from D to the natural numbers; m is called the **multiplicity function** of M, meaning that for any $x \in D$, $m(x)$ denotes the number of times x occurs in M (thus $m(x) = 0$ if x does not occur in M, $m(x) = 3$ if x occurs three times in M, etc.) We say that an element $x \in D$ **belongs to** M, or "is in M", iff $m(x) > 0$. By an "element of M" we will mean a *single occurence* of some $x \in D$. Instead of writing $m(x)$ we will write directly $M(x)$ (i.e we informally identify M and m). From what was said above, conventional sets S over a universe of discourse U can be seen as multisets M over U such that for all $x \in U$ either $M(x) = 0$ or $M(x) = 1$. A multiset M over D is **finite** iff the set $\{x \in D | M(x) > 0\}$ is finite. The empty multiset $\emptyset$ over D is (uniquely) defined as having zero occurences of all $x \in D$. Given a set D, we write $MUL(D)$ to denote the class of all finite multisets over D.

Two multisets M_1 and M_2 (over D) are equal iff $(\forall x \in D) M_1(x) = M_2(x)$. Union is defined as follows: $(\forall x \in D)(M_1 \cup M_2)(x) = M_1(x) + M_2(x)$. E.g $\{2, 2, 1\} \cup \{0, 2, 1\} = \{0, 1, 1, 2, 2, 2\}$. The inclusion relation $\subseteq$ is defined as

follows: $M_1 \subseteq M_2$ iff $(\forall x \in D)(M_1(x) \leq M_2(x))$. The difference $M_1 - M_2$ is defined by $(M_1 - M_2)(x) = M_1(x) \dot{-} M_2(x)$, where $\dot{-}: N \times N \to N$ is the partial subtraction function.

Let $>$ be a partial order on the underlying domain D. Then we can use $>$ to define a binary relation $>>$ on $MUL(D)$ as follows: $M_1 >> M_2$ iff there are multisets $M_1' \neq \emptyset, M_2'$ and M such that $M_1 = M_1' \cup M, M_2 = M_2' \cup M$ and for all y in M_2' there is an x in M_1' such that $x > y$. In this definition M is "the common part" of M_1 and M_2, so what is really said is: M_1 is "greater" than M_2 iff M_2 can be obtained from M_1 by removing *one or more* elements x from M (namely those in M_1') and replacing each such x with a finite number (maybe zero) of elements y (in M_2') each of which is smaller (in the $>$ ordering) than x. For example, $M_1 = \{3,3,4,5,5\} >> \{3,1,1,2,4,5,5\} = M_2$ because an occurence of 3 in M_1 has been replaced by two occurences of 1 and one of 2 in M_2. Here $M = \{3,4,5,5\}, M_1' = \{3\}$ and $M_2' = \{1,1,2\}$; since the replacements (1 and 2) are both less than the removed element (3), we have $M_1 >> M_2$. Also, $\{2,2,5,8\} >> \{2,8\}$ because in this case two elements from the first multiset have been *deleted*, i.e removed and replaced by nothing; here $M_2' = \emptyset$, so the universally quantified condition in the definition of $>>$ is vacuously true.

A basic result about multisets is that the relation $>>$ is well-founded iff $>$ is well-founded. Of course first we must prove that $>>$ is a partial order:

LEMMA 1 $(MUL(D), >>)$ *is a partial order.*

Proof. To show irreflexivity, suppose that $M >> M$ for some multiset M, so that there are multisets $A \neq \emptyset, B, C$ such that $M = A \cup C$ (1), $M = B \cup C$ (2), and for all $y \in B$ there is an $x \in A$ such that $x > y$(3). It follows from (1) and (2) that $A = B$, so that condition (3) becomes: For all $y \in B$ there is an $x \in B$ such that $x > y$ and $x \neq y$ (because $>$ is irreflexive) (3′). But (3′) entails that B is infinite, which is inconsistent with our assumptions.

To show transitivity, define a restriction $>>^1$ of the $>>$ relation on $MUL(D)$ as follows: $M_1 >>^1 M_2$ iff there are multisets M_2', M, and an *one-element multiset* M_1' such that $M_1 = M_1' \cup M, M_2 = M_2' \cup M$ and for all $y \in M_2', x > y$, where x is the sole element of M_1. In other words, $M_1 >>^1 M_2$ iff M_2 can be gotten from M_1 by replacing exactly one element of M_1 with a finite number of $>$ – smaller elements. Since $>>$ is the irreflexive transitive closure of $>>^1$, it must be transitive. Irreflexivity and transitivity together imply anti-symmetry. □

We can now prove the following :

THEOREM 1 *The partial order* $(MUL(D), >>)$ *is well-founded iff* $(D, >)$ *is.*

Proof. The "only if" part is easy: Assume that $>>$ is well-founded and, by way of contradiction, assume that $>$ is not, so that there is an infinite chain $x_1 > x_2 > x_3 > \ldots$. But then $\{x_1\} >> \{x_2\} >> \ldots$ would also be an infinite chain, contradicting our assumption that $>>$ is well-founded.

In the other direction, assume that $>$ is well-founded while $>>$ is not, so that there is an infinite chain of multisets $M_1 >> M_2 >> M_3 >> \ldots$. Let D' be the set obtained by enlarging D with two new elements , a "top element" $\top$ and a "bottom element" $\perp$, and let $>'$ be the partial order defined on D' by adding to $>$ the pairs $< \top, x >, < x, \perp >$ (for all $x \in D$), and $< \top, \perp >$. It is clear that $(D', >')$ is well-founded, since we have assumed that $(D, >)$ is.

We now use the above chain $M_1 >> M_2 >> \ldots$ to construct a tree T whose nodes will be elements of D'. The construction will proceed gradually in stages $i = 0, 1, 2, \ldots$. Let T_i be the tree constructed after stage i and before stage $i+1$, and let L_i be the set of T_i's leaves. For all i, L_i will be a multiset over D'. Now, at stage 0 let T_0 be a single $\top$ node, which will serve as the root of T. At stage 1 make the elements of M_1 children of the root, so that $L_1 = M_1$. Now since $M_1 >> M_2$ there are multisets $M_1' \neq \emptyset, M_2'$, and M such that $M_1 = M_1' \cup M, M_2 = M_2' \cup M$ and for all $y \in M_2'$ there is an $x \in M_1'$ such that $x > y$. Thus at stage 2 do two things:
(I) Add a child node $\perp$ to each x in M_1', and
(II) For each y in M_2' pick one (any one) x in M_1' such that $x > y$ and grow a child node y from it. Thus $L_2 - \{\perp\} = M_2$.
Note two things: (a) M_2' might be empty (if all x in M_1' are deleted rather than replaced). In that case step (II) will not add any new nodes to the tree. But since $M_1' \neq \emptyset$ we *must* add *at least one* new $\perp$ node to the tree at step (I). Since these two steps are repeated at every stage $i > 1$, step (I) ensures that the final tree

$$T = \lim_{i \to \infty} T_i$$

is infinite. (b) Even if M_2' is non-empty, it is finite, so at step (II) we will only add a finite number of new nodes. Hence, for each i, L_i is finite. In particular, $L_i - \{\perp\} = M_i$ for all $i > 0$.

At stage 3 we proceed in the same way. Each element x of M_2 that is removed in going from M_2 to M_3 (and there is at least one such x) is a leaf

of T_2; add to it a child $\perp$. Then for each replacement y in M_3 (if there are any) choose a removed element x that is greater than it and add it to x as a child. Now M_3 is L_3 (minus $\perp$ of course), so we can repeat the same process.

By remark (a), T is infinite and, by remark (b), each node in it has a finite number of children. Hence Koning's lemma applies and entails that T has an infinite path. But, by construction, all paths in T are $>'$– descending, hence an infinite path in it would be an infinite $>'$– descending sequence in D', implying that $(D', >')$ and thus $(D, >)$ is not well-founded– contradicting our assumption. Therefore, no infinite chain $M_1 >> M_2 >> \ldots$ can exist in $MUL(D)$ if $(D, >)$ is well-founded. $\square$