Unique Normal Forms and Confluence of Rewrite Systems: Persistence

Rakesh M. Verma*
Department of Computer Science
University of Houston
Houston, TX 77204, U. S. A.
E-mail: rmverma@cs.uh.edu

Abstract

Programming language interpreters, proving theorems of the form $A = 2?$, abstract data types, and program optimization can all be represented by a finite set of rules called a rewrite system. In this paper, we study two fundamental concepts, uniqueness of normal forms and confluence, for nonlinear systems in the absence of termination. This is a difficult topic with only a few results so far. Through a novel approach, we show that every persistent system has unique normal forms. This result is tight and a substantial generalization of previous work. In the process we derive a necessary and sufficient condition for persistence for the first time and give new classes of persistent systems. We also prove the confluence of the union (function symbols can be shared) of a nonlinear system with a left-linear system under fairly general conditions. Again persistence plays a key role in this proof. We are not aware of any confluence result that allows the same level of function symbol sharing.

1 Introduction

Two of the most challenging and important problems in rewriting are proving the Unique-Normal-Form and Church-Rosser (also called confluence) properties for non-left-linear (nonlinear, for short) systems, particularly in the absence of termination. There is considerable progress on proving Church-Rosser theorems for left-linear systems (systems in which the left-hand sides (lhs's) of the rules contain at most one occurrence of any variable) [Church and Rosser, 1936; Rosen, 1973; O'Donnell, 1977; Huet, 1980]. In contrast, for nonlinear systems there are only a handful of general results and almost all of them require termination [Newman, 1942; Knuth and Bendix, 1970: Huet, 1980; Middeldorp and Toyama, 1991; Rao, 1993J. We know of only four works that do not require termination [Klop, 1980; Chew, 1981; Toyama, 1987; Oyamaguchi and Ohta, 1992].

In 1980, Klop proved the Church-Rosser property for the disjoint sum of an orthogonal (i.e., left-linear and nonoverlapping; see next section for precise definitions) combinatory reduction system and a single nonlinear rule of various specific forms (e.g., $D(x,x) \rightarrow x$ and $D(x,x)$ -4 $E(x)$). In 1987, Toyama proved that the disjoint-sum of two Church-Rosser rewrite systems is Church-Rosser. In 1992, Oyamaguchi and Ohta showed the Church-Rosser property for non-E-overlapping right-ground (i.e., right-hand sides contain no variables) rewrite systems. A weaker result than Church-Rosser, viz., uniqueness of normal forms for strongly nonoverlapping,[1] compatible systems was shown by Chew in the 1981 STOC [1981] (see also [Klop and de Vrijer, 1989] for some unique-normal-form results for A-calculus + specific rules). A strongly nonoverlapping system is one that remains nonoverlapping even when the variables in the lhs's are renamed to make the rules left-linear. The non-E-overlapping requirement is stronger than Chew's strongly nonoverlapping requirement and is in a sense the strongest version of nonoverlapping requirement possible.

In this paper, we attack these two fundamental problems and prove the following results:

• Every persistent system has the unique normal form property. Roughly speaking, persistence means that no rule can be applied inside the template (non-variable part) of an lhs. Persistence is a substantially weaker requirement than the strong-nonoverlap requirement, hence this result is a substantial generalization of Chew's result [Chew, 1981]. (To keep the technical details understandable we do not permit root overlaps. This generalization will be discussed in detail in the full version.) The approach used in proving this result is also novel and should be outlined.

We introduce the idea of constraints and their satisfiability in a rewrite system. We then characterize nonoverlaps and persistence as certain kinds of unsatisfiable constraints. We then prove that these kinds of constraints remain unsatisfiable even when certain kinds of rules are added to a persistent system and exploit this fact to first prove a slightly weaker uniquely normalization property and then the unique normal form property (UN). This stepwise approach makes for easier understandability. Our approach also yields a *neces-*

*Research supported in part by NSF grant CCR-9303011.

[1] Chew allows root overlaps provided they are compatible, e.g., $x + 0 \rightarrow x$ and $0 + x \rightarrow x$ root overlap in $0 + 0$.

sary and sufficient condition for persistence for the first time, which we then use to give several syntactically-checkable sufficient conditions for persistence. Some of these classes are new, not known previously to be persistent. These results are significant generalizations of some of the results in [Verma, 1991], where a sufficient condition for persistence, based on proving unique normalization (plus other technical conditions), was given. Our results are interesting from another viewpoint also. Persistence was introduced by us in a different context [Verma, 1991), viz., for generalizing to nonlinear systems the congruence-closure normalization algorithm of Chew [Chew, 1980], which in turn generalizes the congruence-closure algorithm of [Downey *et a*/., 1980; Kozen, 1977; Nelson and Oppen, 1980] to rules containing variables. Recently, we showed that persistence also plays a central role in transforming certain kinds of rewrite systems into constructor-based rewrite systems [Thatte, 1988; Verma, 1995]. Therefore, our results in this paper provide substantial new evidence for the fundamental role of persistence in rewriting.

- We prove that the union (generalization of disjoint sum, function symbols can be shared) of a system *R2* with a left-linear system *R* is confluent provided that the union is semi-terminating (no sequence containing infinite *R2* reductions), persistent and rhs's of rules in *R1* do not share any function symbols with lhs's of rules in *R2.* We are not aware of any confluence result which allows this much function sharing. The closest result is that of Klop's on CRS's. However, Klop's proof cannot be used directly since it uses postponement of certain kinds of reductions, which does not hold for us. Moreover, Klop gets persistence for free because of the specificity of rules in *R2.* Note that Toyama's proof technique cannot be used since it uses the non-increasing nature of ranks of terms, which does not hold for non-disjoint sums. Recently, Rao [Rao, 1993] generalizing a result of [Middeldorp and Toyama, 1991] proved a confluence result for terminating systems that allows some sharing provided that the union is a hierarchical combination and constructor-based. In particular, no sharing of defined symbols is allowed in the lhs's and only constructors can be shared between lhs of the higher system with rhs's of the lower. We note that Rao's proof is somewhat easier since his conditions ensure that the union is also terminating and persistent. The full proof of our result is fairly long. We sketch the important details here and leave the rest to the full version. We then give several sufficient conditions that can be checked syntactically, which ensure that the union has the properties we need.

- Finally, we consider the confluence of one-rule systems. The motivation for studying properties of one rule systems is Dauchet's interesting result: every (deterministic) turing machine can be simulated by a single left-linear, nonoverlapping rule [Dauchet, 1992]. We show that there is a single nonoverlapping rule that is not persistent and not confluent. The smallest previous example known to us contains 3 rules. We also state a confluence theorem for single rule persistent systems (proof omitted for space).

2 Preliminaries

To save space, we assume familiarity with basic notions of rewriting (see [Dershowitz and Jouannaud, 1990; Klop, 1992] for excellent surveys). Let V be a countable set of elements called *variables* and Σ be a countable set of function symbols with $\Sigma \cap V = \emptyset$. $\mathcal{T}$ is the set of all terms of a first-order language constructed from V and Σ. It is convenient to think of terms as ordered rooted trees. $\mathcal{T}(S)$ denotes that the terms are constructed from function symbols in S (the set V of variables is implicit). The root symbol of a term is: $root(t) = f$ if $t = f(t_1, \ldots, t_n)$, and $root(t) = t$ if $t \in V$. Consider an extra constant $\Box$ called a hole and the set $\mathcal{T}' = \mathcal{T}(\Sigma \cup \{\Box\})$. Then $C \in \mathcal{T}'$ is called a *context* on Σ. We use the notation $C[,\ldots,]$ for the context contatining n holes ($n \geq 0$). A is a *subterm* of B if $B = C[A]$ for some context C.

The notion of a *path* or *occurrence* is used to refer to subterms in a term as follows. A path is either the empty string λ that reaches the root or $o.i$ (o is a path and i an integer) which reaches the ith argument of the root of the subterm reached by o. t/o refers to the subterm of t reached by o and $t[o \leftarrow s]$ denotes the term obtained by replacing the subterm t/o by s. $o \leq q$ whenever $\exists p\; o.p = q$; if $p \neq \lambda$ also, then $o < q$. For any term t its set of occurrences is denoted $O(t)$.

A *substitution* maps variables to terms. An *instance* (also called a *redex*) $\sigma(s)$ of a term s is obtained by substituting $\sigma(x)$ for every variable x in s. A *rule* is a pair of terms $l \to r$, such that $l \notin V$ and every variable occurring in r also appears in l (the variables in a rule are implicitly universally quantified). A *system* R is a finite set of rules. By substituting different terms for the variables, we can produce many different rule instances from the same rule. We say that a rule $A \to B$ is a *rule instance* of the rule $l \to r$ if we can substitute terms for the variables in $l \to r$ to get $A \to B$. As the variables of each rule are universally quantified we shall assume hereafter that any two distinct rules do not share any variable. Terms s and t are *unifiable* if and only if there exists a ground term C which is an instance of both s and t. We say s *overlaps* t if and only if a non-variable subterm u (proper subterm, if $s = t$) of one of the two terms unifies with the other term. (When checking for overlaps it is best to relabel the variables in s and t so that they do not share any variables.) A set $S \subseteq \mathcal{T}$ is *nonoverlapping* if and only if for all $s, t \in S$, *not*(s overlaps t). (Since s and t could be equal, the definition of nonoverlapping does not allow self-overlapping rules like associativity.) We say that s and t *root overlap* if and only if they are left-hand sides of two distinct rules in the system and they can be instantiated to the same term. A system R is *nonoverlapping* if and only if the set of lhs's is nonoverlapping and there are no root overlaps.

Definition 1 (rewrite relation) *We use $\to$ to denote rewrite relations, where $s \to_R t$ (read s reduces to t) if and only if $\exists$ a rule $l \to r$, an occurrence $o \in O(s)$, and a substitution σ such that the subterm $s/o = \sigma(l)$ and $t = s[o \leftarrow \sigma(r)]$.*

Notation. The letters a, b, c, etc., denote constants,

v, x, y, z denote variables, and f, g, h, etc., denote function symbols of nonzero arity. We use $=_R$ to represent the least equivalence relation containing $\rightarrow_R$. When the set of rules R is clear from the context, we drop the subscripts from $\rightarrow$. The reflexive-transitive closure of $\rightarrow$ is denoted by $\xrightarrow{*}$ (i.e., a sequence of zero or more reductions). We use $p : A \xrightarrow{*} B$ to give the name p to the sequence of reductions from A to B ($|p|$ denotes its length). The transitive closure of $\rightarrow$ is denoted by $\xrightarrow{+}$ (i.e., a sequence of one or more reductions). We use $\xrightarrow{r}$ to indicate root reduction (i.e., reduction of the entire term) and $\xrightarrow{nr}$ to indicate nonroot reduction (i.e., reduction at a proper subterm). Similarly, $\xrightarrow{nr*}$ represents a sequence of zero or more nonroot reductions, etc. For every natural number n, $[n]$ denotes $\{1, 2, \ldots, n\}$, $[0] = \emptyset$.

Lemma 2 ([Newman, 1942]) *A noetherian relation is locally confluent (LCR) if and only if it is confluent. (We say that relation $\rightarrow$ is* locally confluent *if and only if $\forall A, B, C$ $A \rightarrow B$ and $A \rightarrow C$ implies $\exists$ D such that $B \xrightarrow{*} D$ and $C \xrightarrow{*} D$. We say that relation $\rightarrow$ is* confluent *(CR) if and only if $\forall A, B, C$ $A \xrightarrow{*} B$ and $A \xrightarrow{*} C$ implies $\exists$ D such that $B \xrightarrow{*} D$ and $C \xrightarrow{*} D$.)*

Definition 3 *Let R be a rewrite system. A term t is a* normal form *if there is no term u such that $t \rightarrow u$. A term t* has a normal form *if there is a normal form u such that $t \xrightarrow{*} u$. R (or $\rightarrow_R$) is uniquely normalizing (is $UN^{\rightarrow}$) if for all terms A, B, C such that $A \xrightarrow{*} B$ and $A \xrightarrow{*} C$ and B, C are normal forms we have $B = C$. R (or $\rightarrow_R$) has unique normal forms (is UN) if for all normal forms A, B with $A =_R B$ we have $A = B$. R is terminating if there are no infinite reduction sequences $t_0 \rightarrow t_1 \rightarrow \ldots$.*

The relation between the various properties CR, UN, and $UN^{\rightarrow}$ is given in the next lemma.

Lemma 4 *The following implications hold for every system R: $CR \Rightarrow UN \Rightarrow UN^{\rightarrow}$. The reverse implications generally do not hold.*

Proof: The proofs of both statements are standard (see, for example [Klop, 1980]). We include examples to show that the reverse implications do not hold always since they are useful in grasping some of the implications of our results.

$UN^{\rightarrow} \not\Rightarrow UN$. Let $R = \{a \rightarrow b, a \rightarrow c, c \rightarrow c, d \rightarrow c, d \rightarrow e\}$. R is $UN^{\rightarrow}$, but not UN since $b =_R e$ and b, e are distinct normal forms.

$UN \not\Rightarrow CR$. Let $R = \{a \rightarrow b, b \rightarrow b, a \rightarrow c, c \rightarrow c\}$. R is UN (a, b and c are not normal forms) but R is not CR since b and c do not have a common reduct. □

Two rewrite systems R_1 and R_2 (see [Toyama, 1987] for formal definitions) are disjoint if no function symbol appears in a rule from R_1 and a rule from R_2. The disjoint sum is obtained by taking the union of two disjoint systems. Note that this disjoint sum is different from Klop's [Klop, 1980]. The disjoint sum of combinatory reduction systems (in which terms are written in "combinator notation") is defined as the union of two systems with disjoint constant symbols, but with the same application function symbol. Klop pointed out that this sum does not preserve confluence.

3 Persistence

In this section we first define persistence and then give a necessary and sufficient condition for it. From the necessary and sufficient condition, we derive several syntactically-checked sufficient conditions for it. Finally, we use our characterization of persistence to prove that every persistent system is UN. Intuitively, persistence requires that the template of the lhs in a redex is untouched by nonroot reductions, and any root reduction after a sequence of nonroot reductions starting from a redex of l ($l \rightarrow r \in R$) can only be applied with the same rule. Hence all the terms in the sequence of nonroot reductions are instances of the template of l. (This property is the inspiration for the term *persistence*.)

Definition 5 *R (or $\rightarrow_R$) is persistent if for every term A such that (i) $A \xrightarrow{r}_R B$ via $l \rightarrow r \in R$, (ii) $A \xrightarrow{nr*}_R A'$, and (iii) $A' \rightarrow_R B'$ via $l' \rightarrow r' \in R$ applied at $o \in O(A')$, either (1) $A' \xrightarrow{r}_R B'$ and $l' \rightarrow r' = l \rightarrow r$, or (2) there is a $u \in O(l), u \leq o$, and l/u is a variable.*

Note that persistence implies that the system must be nonoverlapping, but the converse is false, e.g., let $R = \{f(x, x) \rightarrow a, f(x, g(x)) \rightarrow a, b \rightarrow g(b)\}$. The above definition of persistence is a slight modification of the definition in [Verma, 1991] to avoid root overlaps.

3.1 Necessary and Sufficient Conditions

We begin with some definitions.

Definition 6 *An equation is a pair of terms (s, t) with implicitly universally quantified variables. We say that an equation (s, t) is provable in a system R if s and t are joinable (reduce to a common term) in R.*

A constraint is a pair of terms (s, t) with implicitly existentially quantified variables. We say that a constraint (s, t) is satisfiable in a system R, if there is a substitution σ such that $(\sigma(s), \sigma(t))$ is provable in R. Substitution σ is then called a satisfying substitution. A constraint (s, t) is left-reducible if there is a substitution σ such that $\sigma(s) \xrightarrow{*} \sigma(t)$. We say that $\sigma(s)$ is a solution in this case.

We extend these definitions to sets of equations and sets of constraints. In general, each constraint/equation may be separately existentially/universally quantified, but here we only need sets of constraints that are quantified at the top level (i.e., a set of variable sharing constraints, so the substitution must be consistent for different occurrences of the same variable).

Let R be a rewrite system and let $l \rightarrow r$, $l' \rightarrow r'$ be two rules (not necessarily distinct) in R. Assume, for simplicity, that distinct rules do not share any variables. Let $Unif$ denote the (ordinary) unification algorithm and $Unif^{\infty}$ denote the unification algorithm "without the occurs-check" (for concreteness, let these algorithms be the ones in [Dwork *et al.*, 1984]). We say that a nonoverlap of l with a non-variable subterm u of l' (proper subterm, if the two rules are the same) is due to *occurs-check* (O-nonoverlap) if l and u unify with $Unif^{\infty}$ (but not with

Unif). We say that a nonoverlap is due to *inhomogeneity* (I-nonoverlap), if l and u do not unify even with $Unif^\infty$. For an O-nonoverlap of l and u, $Unif(l,u)$ generates a set of $n > 0$ constraints, where each constraint e_i is of the form $(z_i, C_i[z_i, \ldots, z_i])$ (if one of the terms in a constraint is a variable, in this paper we adopt the convention of listing it first) with some nonempty context C_i not containing z_i and whose template is (properly) included in the template of one of the terms (the z_i's are not necessarily distinct). We say that such a set of $n > 0$ constraints is *left-unreducible* in R iff there is no substitution σ such that $\sigma(z_i) \xrightarrow{*}_R \sigma(C_i[z_i, \ldots, z_i])$ for all $i \in [n]$. In other words, the set of constraints is not left-reducible simultaneously.

Example. The nonoverlap of $f(x, g(x))$ and $f(x,x)$ at the root is due to the occurs-check, whereas all the nonoverlaps of $h(z, g(z), a)$ and $h(x, x, b)$ are due to inhomogeneity.

Theorem 7 *A rewrite system R is persistent if and only if the following conditions hold: (i) R is nonoverlapping, and (ii) For each O-nonoverlap in R, the corresponding set of constraints is left-unreducible in R.*

Proof (sketch): Necessity is obvious. For sufficiency, a direct proof that the stated conditions imply persistence is quite difficult. We prove the following stronger claim that implies persistence. Let $L(R)$ denote the set of distinct (distinct means distinct even after variable renaming) lhs's of rules in R and let $SL(R)$ denote the set of all distinct *non-variable subterms* of the terms in $L(R)$. (Note that since R is nonoverlapping the lhs's of two different rules must be distinct, but proper subterms of lhs's may not be.)

Claim 8 *There is no term A such that $p : A \xrightarrow{nr*} B = \sigma(l)$ and $q : A \xrightarrow{nr*} C = \sigma'(l')$ for distinct l, l' in $SL(R)$, unless l and l' overlap.*

The proof is omitted to save space. Obviously persistence follows from the claim, since the set of lhs's contains neither root nor nonroot overlaps. □

Remark. In general, checking the second condition of Theorem 7 is easily seen to be undecidable even for a certain fixed rewrite system. However, below are some easily-checked *syntactic* conditions that imply the above condition.

Corollary 9 *1. Any nonoverlapping rewrite system in which all nonoverlaps are due to inhomogeneity is persistent.*

2. Any strongly nonoverlapping system (see [Chew, 1981]) is persistent.

3. Any nonoverlapping rewrite system in which no function symbol appearing in the lhs of any nonlinear rule appears on the rhs of any *rule is persistent.*

4. Any nonoverlapping quasi-terminating (see [Dershowitz, 1987] for definition) is persistent.

5. Any left-linear nonoverlapping system is persistent.

Proof: Straightforward. □

Remarks. Note that except for quasi-termination all the other conditions can be easily checked syntactically given a rewrite system. The classes (1) and (4) of the corollary are new classes for which persistence was not known earlier. Note further that these applications are not meant to be exhaustive. They are included because of their utility and because some of them have appeared in the earlier literature in various contexts.

Examples. We give some simple examples to illustrate these classes.

For 1. Let $R = \{f(x,x) \to a, f(a,b) \to b\}$. R has only I-nonoverlaps and R is not strongly nonoverlapping.

For 2. Let $R = \{f(x,x) \to e, g(x) \to f(x, g(x)), c \to g(c)\}$ ([Klop, 1980]). Note that R is strongly nonoverlapping and persistent (trivially) since every proper subterm (if it exists) of every lhs is a variable and the root symbols of all lhs's are different. Note also that R is not confluent since $g(c) \xrightarrow{*} g(e)$ and $g(c) \xrightarrow{*} e$ but $g(e)$ and e are not joinable. Note however that R is UN.

For 3. Let $R = \{f(x,x) \to a, f(x, g(x)) \to b, c \to h(c)\}$. Note that if we change the third rule to $c \to g(c)$ the resulting system is not persistent and not confluent also.

For 4. Let $R = \{f(x,x) \to f(a,a), f(x, g(x)) \to b\}$.

For 5. Let $R = \{f(x,y) \to e, g(x) \to f(x, g(x)), c \to g(c)\}$. Note that this is the left-linear version of the Example for 2. R is persistent and, of course, confluent.

We now prove the following important consequence of persistence: every persistent system R has the unique normal form property.

3.2 Unique Normal Form Property

Lemma 10 *R is persistent iff $R' = R \cup \{h(x,x) \to a, h(N_1, N_2) \to b\}$ is persistent, where N_1, N_2 are two distinct normal forms with respect to R, and h, a, b are new function symbols not in the signature of R and not in N_1 or N_2.*

Proof: Clearly, R' persistent implies R persistent. Now suppose R is persistent. We note that R' is also nonoverlapping and that the new rules only have I-nonoverlaps with R and with each other (N_1 and N_2 are distinct terms and both contain no h's). So it suffices to show that every set of constraints corresponding to an O-nonoverlap that is left-unreducible in R is also left-unreducible in R'. Suppose that there is a set of constraints that is left-reducible in R' but left-unreducible in R. Consider the case of one constraint $z = C[z, \ldots, z]$. Note that C is some nonempty context and all the function symbols of C appear in the lhs's of some rules in R. In particular, h, a and b do not appear in C. Let σ be the substitution such that $p : \sigma(z) = A \xrightarrow{*}_{R'} C[A, \ldots, A]$. If A contains no h's, then the constraint is also left-reducible in R since the rhs's of rules in R cannot introduce any h in the term being reduced and hence the two extra rules in R' can never be applied in p. So suppose that A contains some h-preredexes (terms $h(\ldots)$).

Therefore, let $A = C''[A_1, \ldots, A_n]$ where the context C'' does not contain any symbols in $\{h, a, b\}$ and $root(A_i) \in \{h, a, b\}$ for $i \in [n]$. Clearly, C' cannot be empty. Also, note that the C' can be modified, but it cannot be completely erased. Formally, every term t in the reduction sequence p must be of the form $C_t[\ldots]$ for a nonempty context C_t not containing any symbols in $\{h, a, b\}$. We call a reduction in p an *external* reduction if it occurs outside every subterm of the form $h(\ldots)$, and *internal* otherwise. Define a transformation T on terms in $\mathcal{T}(F \cup \{h, a, b\}, V)$ to $\mathcal{T}(F, V)$ as follows: if $t = C[t_0, \ldots, t_n]$ $(n \geq 0)$ where C is a (possibly empty) context not containing any symbols in $\{h, a, b\}$ and $root(t_i) \in \{h, a, b\}$ for $i \in [n]$, then $T(t) = C[u, \ldots, u]$, where $u \in V$ is a fixed variable.

Claim: $A \to_{R'} B$ implies $T(A) \xrightarrow{*}_R T(B)$.

Proof of claim. If the reduction from A to B is an internal step, then by the construction of R', $T(A) = T(B)$, which is very important in the sequel. If the reduction step is external, then $A = C[A_1, \ldots A_n]$, where C is a nonempty context not containing $\{h, a, b\}$ and there is an occurrence $o \in O(C)$ such that $C/o \neq \Box$, $A/o = \sigma(l)$ for some substitution σ and rule $l \to r$ in R (note R and not R'). Also, $B = A[o \leftarrow \sigma(r)] = C'[A_{i_1}, \ldots, A_{i_m}]$, where $i_j \in [n]$ and C' is a possibly empty context not containing symbols in $\{h, a, b\}$. Now the template of l must be contained in C since rules in R do not contain any function symbols from $\{h, a, b\}$, and variables in l cover subterms of the form A_i (if any are dominated by occurrence o). Now $T(A) = C[u, \ldots, u]$. Clearly, $T(A)/o = \sigma'(l)$, where $\sigma'(x) = T(\sigma(x))$ and $T(A)[o \leftarrow \sigma'(r)] = C'[u, \ldots, u] = T(B)$ and we are done. $\Box$

By the above claim and a simple induction we have the following claim: $T(A) \xrightarrow{*}_R C[T(A), \ldots, T(A)]$ via the external reductions in p. But, this contradicts the assumption that the constraint was left-unreducible in R and the proof is complete. $\Box$

Theorem 11 *Every persistent system R is uniquely normalizing $UN^{\to}$.*

Proof: Suppose that R is not uniquely normalizing. Then, $\exists$ A such that $A \xrightarrow{*} B$ and $A \xrightarrow{*} C$, where B, C are distinct normal forms. Let $R' = R \cup \{h(x, x) \to a, h(B, C) \to b\}$, where h, a, b are new function symbols not in the signature of R and not in B, C. Now $h(A, A) \xrightarrow{nr*}_{R'} h(B, C)$ so R' is not persistent. Therefore, by Lemma 10, R is not persistent, which contradicts the assumption. $\Box$

Theorem 12 *Every persistent system R has the unique normal form property (UN).*

Proof: Suppose that R is persistent but does not have the unique normal form property. Then, there are distinct R-normal forms A and B such that $A =_R B$. Consider a shortest proof of $A =_R B$. It is of the form $q : A_1 = A =_R A_2 =_R \ldots =_R B = A_n$ for some $n > 1$, where each $=_R$ step is either $\to_R$ or $\leftarrow_R$. Since A and B are R-normal forms the first step in q is $\leftarrow$ and the last step in q is $\to$. Call a term A_i in q a *peak* if $A_{i-1} \leftarrow A_i \to A_{i+1}$ in q. Call a term A_i in q a *valley* if $A_{i-1} \to A_i \leftarrow A_{i+1}$. Because the arrows at the ends are in opposite directions and towards A and B, there is at least one peak in q. Let $A_{i_1}, \ldots, A_{i_m}$ be all the peaks in q; and let $A_1 = A_{j_1}, \ldots, A_{j_{m+1}} = A_n$ be all the valleys in q. Here $m > 0$ and $1 = j_1 < i_1 < j_2 \ldots < i_m < j_{m+1} = n$. Note that the terms at both ends are also considered valleys. Therefore, we can write q as:

$$A = A_{j_1} \xleftarrow{*} A_{i_1} \xrightarrow{*} A_{j_2} \xleftarrow{*} A_{i_2} \ldots \xleftarrow{*} A_{i_m} \xrightarrow{*} A_{j_{m+1}} = B.$$

Let h_{2m} be a new function symbol of arity $2m$ and a, b be two new constants not in the signature of R and not in A or B. Consider the system

$$R' = R \cup \{h_{2m}(A, x_1, x_1, \ldots, x_{m-1}, x_{m-1}, B) \to a\}$$
$$\cup \{h_{2m}(x_1, x_1, \ldots, x_m, x_m) \to b\}.$$

By a similar argument as in the proof of Lemma 10, we have that R' is also persistent since R is persistent. But, consider the term $H = h_{2m}(A_{i_1}, A_{i_1}, \ldots, A_{i_m}, A_{i_m})$. Clearly, H is an instance of the second new rule. Now, by the reductions in q, we have $H \xrightarrow{nr*}_R H' = h_{2m}(A, A_{j_2}, A_{j_2}, \ldots, A_{j_m}, A_{j_m}, B)$. But, since H' is an instance of the first new rule in R', we have that R' is not persistent. This contradicts our assumption that R is persistent and the proof is complete. $\Box$

As a consequence of Theorem 12 we have that all the classes of Corollary 9 are UN.

4 The Church-Rosser property of the Union

Let R_1 be any left-linear system. Let R_2 be any system such that every function symbol appearing in the lhs of any rule in R_2 does not appear on the rhs of any rule in R_1 (we call this *lr*-disjoint; similarly one can define *ll*-disjoint, etc). Further, assume that $R = R_1 \cup R_2$ is persistent and satisfies the following finiteness condition called semi-termination.

(F) There is no sequence of R-reductions from any term t containing an infinite number of R_2 reduction steps.

Note that (F) immediately implies that R_2 is terminating, but termination of R_2 is not sufficient for semi-termination of R as shown below.

Example. Let $R_1 = \{a \to b\}$, $R_2 = \{h(x, x) \to h(a, b)\}$. Now R_1 and R_2 are both terminating (in fact, simply terminating, i.e., their termination can be established by simplification orderings; see [Dershowitz, 1987] for a definition), but R has the following cyclic derivation: $h(a, b) \to_R h(b, b) \to h(a, b)$. The example can be easily modified so that R is not even quasiterminating. Note that all conditions except (F) are satisfied.

We now prove that R is confluent and then give sufficient conditions that ensure persistence and finiteness of R. Observe that Toyama's [Toyama, 1987] technique cannot be used since it depends on the non-increasing property of ranks w.r.t. reductions, which does not hold for us. We need the following lemmas for the proof of confluence.

Lemma 13 *For all A, B, C such that $A \xrightarrow{*}_{R_1} B$ and $A \xrightarrow{*}_R C$, there exists D, E such that $C \xrightarrow{*}_{R_1} D$ and $B \xrightarrow{*}_R D$.*

Proof (sketch): An easy argument using the confluence of R_1 shows that it is sufficient to prove the following: for all A, B, C such that $A \xrightarrow{*}_{R_1} B$ and $A \to_{R_2} C$, there exists D such that $C \xrightarrow{*}_{R_1} D$ and $B \xrightarrow{*}_R D$ (actually we show something stronger, viz., $B \xrightarrow{*}_{R_1} \cdot \xrightarrow{*}_{R_2} D$). $\Box$

We classify the set of function symbols that appear in R into *linear* and *nonlinear* as follows: if a function symbol f appears in the lhs of *any* rule in R_2, then f is nonlinear and linear otherwise. For the next lemma, we need the following definition of the *norm* of a term.

Definition 14 ([Klop, 1980]) *The* nonlinear height *of a term t (notation $|t|_n$) is the maximum number of nonlinear symbols on any path from the root of t to a leaf. The norm of t, denoted $\| t \| = max\{|u|_n \mid t \xrightarrow{*}_R u\}$.*

Lemma 15 *(1) $\| t \|$ is finite for every t. (2) $t \xrightarrow{*} u$ implies $\| t \| \geq \| u \|$. (3) If u is a subterm of t, then $\| u \| \leq \| t \|$. In particular, if $root(t)$ is nonlinear and u is a proper subterm, then $\| (\|u) < \| (\|t)$.*

Proof: Part (1) Follows from the lr-disjoint and finiteness (F) conditions. Part (2) follows from the definition of norm and transitivity of $\xrightarrow{*}$. Part (3) is straightforward. □

Lemma 16 *If $A \rightarrow_{R_2} B$, $A \xrightarrow{*}_{R_1} C'$ and $C' \rightarrow_{R_2} C$, then there is a D such that $B \xrightarrow{*}_R D$ and $C \xrightarrow{*}_R D$.*

Proof (sketch): By induction on the norm. By Lemma 13 we have D' and E such that $C' \xrightarrow{*}_{R_1} D'$, $D' \xrightarrow{*}_{R_2} E$ and $B \xrightarrow{*}_{R_1} E$. Also, all the reduction steps from C' to D' are covered by the steps from D' to E. We consider three cases: (i) the reduction step Rs from C' to C is independent of all the steps from D' to E, (ii) Rs is covered by a step from D' to E, and (iii) Rs covers some (possibly all) of the reductions steps from D' to E. Case (i) is easy. For cases (ii) and (iii) we use persistence and the induction hypothesis to find the desired term D. □

We now define the nonlinear derivation height (notation DH_n) of a term.

Definition 17 $DH_n(t) = max\{n \mid \exists u, t \xrightarrow{*}_R u$ *with n R_2-reductions* $\}$.

Lemma 18 *(1) $DH_n(t)$ is finite for every t. (2) If $t \rightarrow_R u$, then $DH_n(t) \geq DH_n(u)$. If $t \rightarrow_{R_2} u$, then $DH_n(t) > DH_n(u)$.*

Proof: Use the definition of DH_n, transitivity of $\xrightarrow{*}$, and the lr-disjoint and finiteness (F) conditions. □

Theorem 19 *If R_1 is left-linear, $R = R_1 \cup R_2$ is lr-disjoint, persistent and semi-terminating, then R is Church-Rosser.*

Proof: We prove that CR(A) by induction on $DH_n(A)$. The base case is $DH_n(A) = 0$. In this case, the only derivations possible from A consist solely of R_1-reductions. Since R is persistent so is R_1 and since every left-linear persistent system is confluent [O'Donnell, 1977; Rosen, 1973; Huet, 1980], the claim holds for the base case. Assume $CR(A)$ for $DH_n(A) < m$ $(m > 0)$. We show the following claim:

Claim. $A \rightarrow_{R_2} B$, $A \xrightarrow{*}_{R_1} C'$, and $C' \xrightarrow{*}_{R_2} C$ implies there is a D such that $B \xrightarrow{*}_R D$ and $C \xrightarrow{*}_R D$ for the case $DH_n(A) \leq m$.

Proof of claim: If there are zero reductions from C' to C, then we use Lemma 13 to get the desired term D. Otherwise, let $C' \rightarrow_{R_2} C''$ be the first R_2-reduction step. By Lemma 16 we have a D' such that $B \xrightarrow{*} D'$ and $C'' \xrightarrow{*} D'$. Now, $DH_n(C'') < DH_n(C') \leq DH_n(A)$ by Lemma 18, therefore by the induction hypothesis for the theorem, i.e., $CR(C'')$, we have a D such that $D' \xrightarrow{*} D$ and $C \xrightarrow{*} D$. □

Induction Step: We now prove $CR(A)$ when $DH_n(A) = m$. So suppose that $A \xrightarrow{*}_R B$ and $A \xrightarrow{*}_R C$. If all the reductions in either $A \xrightarrow{*}_R B$ or $A \xrightarrow{*}_R C$ are R_1-reductions, then we are done by Lemma 13. Otherwise we have the following situation: $A \xrightarrow{*}_{R_1} C' \rightarrow_{R_2} C'' \xrightarrow{*} C$, and $A \xrightarrow{*}_{R_1} B'$, $B' \rightarrow_{R_2} B''$ and $B'' \xrightarrow{*} B$, where $C' \rightarrow C''$ and $B' \rightarrow B''$ are the first reductions from R_2 on the respective derivations. Now, we have the following derivations. By confluence of R_1 (see base case) we have a D' such that $B' \xrightarrow{*}_{R_1} D'$ and $C' \xrightarrow{*}_{R_1} D'$. By Lemma 13 we have a D'' such that $D' \xrightarrow{*}_{R_1} \cdot \xrightarrow{*}_{R_2} D''$ and $C'' \xrightarrow{*}_{R_1} D''$. By the above claim, we have an E such that $B'' \xrightarrow{*}_R E$ and $D'' \xrightarrow{*}_R E$. By Lemma 18, $DH_n(B'') < DH_n(B') \leq DH_n(A)$ and $DH_n(C'') < DH_n(C') \leq DH_n(A)$. Therefore, by the induction hypothesis of the theorem, i.e., $CR(B'')$ and $CR(C'')$, we have a D such that $B \xrightarrow{*}_R D$ and $C \xrightarrow{*}_R D$ and the proof is complete. □

Remarks. The above result can be generalized in several different ways, we omit proofs of the generalizations for lack of space. First, the definition of nonlinear symbols can be narrowed to exactly the symbols in lhs's of nonlinear rules of R_2. Second, we can drop the finiteness requirement and prove CR(A) for only those terms A for which $DH_n(A)$ and $\| A \|$ are both finite. Third, we do not really need full persistence of R, a slightly weaker form is sufficient. This is important because it permits some kinds of harmless root and nonroot overlaps in R. Finally, note that this proof shows some similarities to Klop's proof. However, as noted earlier Klop's proof cannot be used since it uses postponement of nonlinear reductions, which does not hold for us and also persistence is immediate there.

We now give sufficient conditions that ensure persistence and semitermination of the union. First, we note that nonoverlapping and semitermination imply persistence.

Lemma 20 If *the lr-disjoint union R of a left-linear system R_1 and any system R_2 is nonoverlapping and semi-terminating, then R is persistent.*

Proof: A left-linear rule can have only l-nonoverlap with another (not necessarily left-linear) rule. Therefore, the O-nonoverlaps of R are exactly those of R_2. We show that the satisfiability of any constraint contradicts semitermination of R. Since the O-nonoverlaps are between rules of R_2 each constraint is of the form $(z, C[z, \ldots, z])$ for a nonempty context C not containing z. Futher, C consists of function symbols appearing in lhs's of R_2. Let t be any solution of a constraint. Then, $p : \sigma(z) = t \xrightarrow{*}_R C[t, \ldots, t]$. All the reductions in p cannot be from R_1 since lhs's of rules from R_2 do not share any function symbols with rhs's of rules from R_1. Hence, there must be at least one R_2 reduction in p. But then we can construct a sequence containing infinitely many R_2 steps from t, contradicting the semi-termination of R. □

Theorem 21 *The following conditions are sufficient for the persistence and semi-termination of a nonoverlapping system R. R_1 is linear (i.e., left-linear & right-linear), and (1) No function symbol that appears in the lhs of any rule in R_2 appears in the rhs of any rule, or (2) All R_2 rules are height decreasing, i.e. $ht(l) > ht(r)$ for all $l \to r \in R_2$, or (3) All R_2 rules are nonlinear-height decreasing, i.e, $|l|_n > |r|_n$ for every rule $l \to r \in R_2$.*

5 Confluence of a Single Rule

We give an example to show that a single nonoverlapping rule need not be confluent nor persistent. Consider the nonoverlapping rule, $f(x, f(z, y, g(z)), x) \to g(f(a, f(a, a, g(a)), a))$ and let T be the term $f(a, f(a, a, g(a)), a)$. Clearly, $T \to g(T)$. Now consider the term $U = f(b, f(T, f(a, a, g(a)), g(T)), b)$. Clearly, $U \overset{r}{\to} g(T)$ and $U \overset{nr}{\to} f(b, f(g(T), f(a, a, g(a)), g(T)), b)$ $\overset{nr}{\to} f(b, g(T), b)$. Now it is easily seen that $f(b, g(T), b)$ and $g(T)$ do not have a common reduct, since the rule can never be applied to $g(T)$ at the root level. For the same reason the rule cannot be applied at the root level to $f(b, g(T), b)$ also since the inner term $g(T)$ can never be reduced to a term of the form $f(...)$. The smallest, previously known to us, non-confluent nonoverlapping rewrite systems have three rules [Huet, 1980; Klop, 1980]. However, we have the following result:

Theorem 22 *Every single persistent rule is confluent.*

6 Conclusions

In this paper we have studied two fundamental concepts, uniqueness of normal forms and confluence, for nonlinear systems in the absence of termination. This is a difficult topic with only a few results so far. Through a novel approach, we have shown that every persistent system has unique normal forms. This result is tight and a substantial generalization of previous work. In the process we derived a necessary and sufficient condition for persistence and gave several new classes of persistent systems. We also proved the confluence of the union of a nonlinear system with a left-linear system under fairly general conditions. Again persistence plays a key role in this proof. There are several promising directions for future work. First, we note that the finiteness requirement can be weakened somewhat although it cannot be dropped completely. The proof of this is likely to be difficult but fruitful since it might lead to new techniques for dealing with unions (or decompositions) rather than disjoint sums. Second, our work here suggests some natural generalizations to deal with non-persistent systems. Any progress along these two lines will obviously be of considerable importance to rewriting and its applications.

References

[Chew, 1980] P. Chew. An improved algorithm for computing with equations. In *IEEE FOCS,* pp. 108-117, 1980.

[Chew, 1981] P. Chew. Unique normal forms in term rewriting systems with repeated variables. In *Proc. 13th ACM STOC,* pages 7-18, 1981.

[Church and Rosser, 1936] A. Church and J.B. Rosser. Some properties of conversion. *Trans. AMS,* 39:472-482,1936.

[Dauchet, 1992] M. Dauchet. Simulation of turing machines a regular rewrite rule. *TCS,* 103, 1992. Also in RTA 1989.

[Dershowitz and Jouannaud, 1990] N. Dershowitz and J.P. Jouannaud. Rewrite systems. In *Handbook of Theoretical Computer Science,* vol. 2, ch. 6. North-Holland, 1990.

[Dershowitz, 1987] N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation,* 3:69-116,1987.

[Downey *et al.,* 1980] P.J. Downey, R. Sethi, and R.E. Tarjan. Variations on the common subexpression problem. *JACM,* 27(4):758-771, 1980.

[Dwork *et al,* 1984] C. Dwork, P. Kanellakis, and J.C. Mitchell. On the sequential nature of unification. *Journal of Logic Programming,* 1:35-50, 1984.

[Huet, 1980] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *JACM,* 27(4):797-821, 1980.

[Klop and de Vrijer, 1989] J.W. Klop and R. de Vrijer. Unique normal forms for lambda calculus with surjective pairing. *Information and Control,* 80:97-113, 1989.

[Klop, 1980] J.W. Klop. *Combinatory Reduction Systems.* PhD thesis, Mathematisch Centrum, Amsterdam, 1980.

[Klop, 1992] J.W. Klop. Rewrite systems. In *Handbook of Logic in Computer Science.* Oxford, 1992.

[Knuth and Bendix, 1970] D.E. Knuth and P. Bendix. Simple word problems in universal algebra. In J. Leech, editor, *Computational Problems in Abstract Algebra,* pages 263-297. Oxford, Pergammon Press, 1970.

[Kozen, 1977] D. Kozen. Complexity of finitely presented algebras. In *Proc. ACM STOC,* pages 164-177, 1977.

[Middeldorp and Toyama, 1991] A. Middeldorp and Y. Toyama. Completeness of combinations of constructor systems. In *RTA,* pp. 188-199, 1991.

[Nelson and Oppen, 1980] G. Nelson and D.C. Oppen. Fast decision algorithms based on congruence closure. *JACM,* 27:356-364, 1980. Also in 18th IEEE FOCS Symp., 1977.

[Newman, 1942] M.H.A. Newman. On theories with a combinatorial definition of equivalence. *Ann. Math.,* 43, 1942.

[O'Donnell, 1977] M.J. O'Donnell. *Computing in Systems Described by Eauations,* vol. 58 of *LNCS.* Spr.-Ver., 1977.

[Oyamaguchi and Ohta, 1992] M. Oyamaguchi and Y. Ohta. Church rosser property of right-ground systems. *Trans. IEICE,* J76-D-I, 1992.

[Rao, 1993] M.R.K Rao. Completeness of hierarchical combinations of term rewriting systems. In *FST-TCS,* 1993.

[Rosen, 1973] B.K. Rosen. Tree manipulating systems and church-rosser theorems. *JACM,* 20:160-187, 1973.

[Thatte, 1988] S. Thatte. Implementing first-order rewriting with constructor systems. *TCS,* 61:83-92, 1988.

[Toyama, 1987] Y. Toyama. On the church-rosser property for the direct sum of term rewriting systems. *JACM,* 34(1):128-143,1987.

[Verma, 1991] Rakesh M. Verma. A theory of using history for equational systems with applications. In *82nd IEEE FOCS Symp.,* pages 348-357, 1991. (to appear in JACM)

[Verma, 1995] R.M. Verma. Transformations and confluence for rewrite systems. *Theoretical Computer Science,* 1995.