J. Symbolic Computation (1989) **7**, 319–341

Enumerating Outer Narrowing Derivations for Constructor-Based Term Rewriting Systems†

JIA-HUAI YOU

Department of Computing Science, University of Alberta,
Edmonton, Alberta, Canada T6G 2H1

(*Received* 18 *December* 1987)

Narrowing has been used as a mechanism for reasoning about equations and evaluating equational logic programs, where enumeration of all narrowing derivations is often necessary in order to generate complete sets of solutions. In this paper, a special type of narrowing derivations, called outer narrowing derivations, is examined for the class of constructor-based term rewriting systems. It is shown that every narrowing derivation in this class is subsumed by an outer narrowing derivation. This result is applied to a matching problem in equational theories, i.e., whether an arbitrary term is E-matchable to a term composed of constructors and disjoint variables. It is shown that outer narrowing derivations generate complete and minimal sets of E-matchers. An E-matching procedure is presented which enumerates all and only outer narrowing derivations for the E-matching problem considered in this paper.

1. Introduction

Narrowing is a method that combines term unification and rewriting. This method has been adopted in reasoning about equations encountered in theorem proving (Lankford 1975; Slagle 1974), universal unification and matching (Fay 1979; Hullot 1980; Jouannaud *et al.* 1983; Siekmann 1984), and logic programming with an equational flavor (Dershowitz 1985; Dershowitz & Plaisted 1985; Goguen & Meseguer 1984; Fribourg 1985; Reddy 1985; You & Subrahmanyam 1986a).

Given a term rewriting system R, a term t narrows to s at a nonvariable subterm t/u using the k-th rewrite rule $\alpha_k \rightarrow \beta_k$ in R, denoted by $t \sim>_{[u,k,\rho]} s$ (or $t \sim>_\rho s$ for abbreviation), if ρ is the most general unifier of t/u and α_k, and $s = \rho(t[u \leftarrow \beta_k])$, i.e., t/u is replaced by β_k and ρ is applied to the whole term. It is always assumed that the sets of variables in t and α_k are disjoint. This can be done by appropriate variable renaming. Narrowing gains its power by using term unification while reduction in contrast only uses one-sided unification, which is called *matching*.

Given a term, there might be several subterms at which narrowing is possible. It is important to know if some of the narrowing derivations can be pruned without losing completeness. The problem of special narrowing strategies appears to be a much harder problem than its counterpart concerning reduction, which enjoys several well-known results based on special

† Work supported by the Natural Sciences and Engineering Research Council of Canada.

0747–7171/89/030319+23 $03.00/0

strategies, such as the *innermost* and *outermost* strategies. Difficulties arise when these strategies are considered for narrowing. The following example shows that neither the innermost narrowing strategy which ignores narrowing steps at outer occurrences, nor the outermost narrowing strategy which ignores narrowing steps at inner occurrences can be complete even for some very simple systems. Consider the term rewriting system

$$R = \{ f(y, a) \to true,\ f(c, b) \to true,\ g(b) \to c\ \}.$$

We will use x, y, z, etc., to denote variables and others to denote function symbols. The system is obviously confluent. With the term $f(g(x), x)$, innermost narrowing leads to

$$f(g(x), x) \sim>_{\{x/b\}} f(c, b) \sim>_{\{\}} true,$$

while outermost narrowing yields

$$f(g(x), x) \sim>_{\{x/a\}} true.$$

The innermost and outermost derivations generate *uncompared* solutions; missing either of them would result in loss of a solution.

Fribourg (1985) showed the completeness of the innermost narrowing strategy under certain sufficient conditions. These conditions appear to be restrictive as illustrated above. As pointed out by Fribourg, the difficulty stems from the fact that functions, as defined by rewrite rules, are often *partial* functions over the domain of terms. Though restrictive, innermost narrowing can indeed eliminate many redundant derivations. Dincbas and Hentenryck (1987) investigated some practical aspects of several narrowing strategies, including innermost, outermost and a form of lazy strategy based on a *procedural* semantics of functional programming. To extend a functional language to the one that is capable of *solving*, Reddy (1985) defined a denotational semantics and outlined a lazy narrowing strategy. As pointed out by Reddy, however, lazy narrowing is not complete for the classic equality theory (and it was certainly not designed for that purpose). The problem of whether an approach along the same line as lazy narrowing can be adopted in the completely equational setting has not been explored. Other methods for reducing search space and for eliminating redundant solutions can be found in Hullot (1980), Réty *et al.* (1985) and Réty (1987) (also see Nutt *et al.* 1987). A more detailed account of these narrowing methods will be given in Subsection 3.3.

In this paper we consider a special type of narrowing derivations, called outer narrowing derivations, for solving equations in equational theories. In particular, we consider the application of outer narrowing to an E-matching problem in equational theories. Outer narrowing will be defined in this paper by means of an ordering on narrowing steps in a narrowing derivation. Informally, the main requirement for a narrowing derivation

$$A_0 \sim>_{[u_0, k_0, \rho_0]} \cdots \sim>_{[u_{n-1}, k_{n-1}, \rho_{n-1}]} A_n$$

to be outer is that no *later* narrowing step at an *outer* occurrence be able to be carried out *earlier* in the derivation, using the *same* rule and at the *same* occurrence. (This is by no means a precise definition. See Subsection 3.2) Intuitively, if an outer narrowing step can be "moved" to the early part of the sequence, then those steps before it may be wasted in that they do not contribute to the narrowability of an outer function. An outer narrowing derivation, however, may not be an outermost narrowing derivation. For example, the narrowing derivation given earlier

$$f(g(x), x) \sim>_{\{x/b\}} f(c, b) \sim>_{\{\}} true$$

is considered an outer narrowing derivation, in that the rule used in the second narrowing step is not applicable when tried to narrow the given term, due to the function symbol conflict between c and g. This derivation is obviously not an outermost narrowing derivation.

Outer narrowing can indeed prune redundant derivations. As a simple example, consider

$$R = \{ f(a, b, x) \to true,\ g(a) \to c\ \}.$$

The derivation

$$f(y, z, g(y)) \sim>_{\{y/a\}} f(a, z, c) \sim>_{\{z/b\}} \mathit{true}$$

is not outer since the second narrowing step can be performed right at the beginning, using the same rule and at the same occurrence. In doing so we get the same solution by one step of outer narrowing:

$$f(y, z, g(y)) \sim>_{\{y/a,\, z/b\}} \mathit{true}.$$

It was not very clear at first whether or not outer narrowing is complete when narrowing is. As a matter of fact, in the course of investigation we discovered a counterexample that shows that outer narrowing can lose potential solutions for left-linear and nonoverlapping term rewriting systems. We therefore restrict our attention to a subclass called *constructor-based* term rewriting systems. We will show in this paper that every narrowing derivation in this class is subsumed by an outer narrowing derivation.

A function symbol f is said to be *defined* if it appears as the leftmost functor in the left hand side of some rule $\alpha_k \to \beta_k$; in this case we say that the rule $\alpha_k \to \beta_k$ defines f. Otherwise it is called a *constructor*. A term rewriting system is *constructor-based* if it is left-linear, nonoverlapping, and has no defined function symbols appearing in the inner part of the left hand side of any rule. Some of the functional languages primarily based on recursive equations, such as HOPE (Burstall & Sannella 1980), SASL (Turner 1979), and ML (Milner 1984) fall into this category when restricted to defining first order functions. Although O'Donnell's language (O'Donnell 1985) is purely first order and in theory belongs to the class of nonoverlapping systems, it has mainly been used as a constructor-based language.

Our intention of studying special narrowing strategies was driven by a desire to discover *complete* and *minimal* E-unification or E-matching procedures for some nontrivial classes of equational theories, no matter whether they are used in programming or in theorem proving. Unfortunately, outer narrowing does not guarantee minimality in the general case. However, applying the result of outer narrowing we are able to obtain a complete and minimal procedure for a special case of the E-matching problem: given a constructor-based term rewriting system R and two terms t_1 and t_2 with disjoint sets of variables, where t_2 is composed of constructors and variables, the problem is to find a complete and minimal set Φ of E-matchers from t_1 to t_2 such that for every σ in Φ, $\sigma(t_1)$ is E-equivalent to t_2 under the equational theory described by R and no two E-matchers in Φ can compare under the same equational theory.

The minimality problem in universal unification and matching is known to be important. In general, unification and matching procedures are embedded into a larger reasoning system involving equality (Plotkin 1972), which performs reasoning by using E-unifiers or E-matchers provided by these special inference procedures. Without the minimality property the system may waste a lot of time performing redundant reasoning steps.

The minimality problem is also known to be difficult. Since the discovery of some equational theories for which complete and minimal sets of E-unifiers or E-matchers do not exist (Fages & Huet 1983), the questions as to which classes of equational theories complete and minimal sets of E-unifiers or E-matchers exist, and how to obtain them, have rarely been investigated. To our knowledge, all currently known results are based on the discovery of some complete and terminating algorithms for decidable equational theories. The minimal sets can then be obtained from complete sets by a filtering process. The class of equational theories and the matching problem considered in this paper are both semi-decidable; thus the filtering method no longer applies.

The next section provides notations used in this paper. It is shown in Section 3 that for constructor-based term rewriting systems, every narrowing derivation is subsumed by an outer narrowing derivation. Section 4 applies this result to a matching problem and shows that outer narrowing derivations yield complete and minimal sets of E-matchers. In Section 5, a pattern-driven (Subrahmanyam & You 1986) procedure is presented, which correctly enumerates all and only outer narrowing derivations for this matching problem. The final section contains a summary and some remarks.

2. Preliminary Definitions

2.1 DEFINITION. We denote by $T(F, V)$ the set of terms composed from a set of function symbols F and an enumerable, disjoint set of variables V. Terms are viewed as *labeled trees* in the following way: a term A is a partial function from the set of sequences of positive integers, denoted by I^*, to $F \cup V$ such that its domain satisfies:

(i) $\varepsilon \in D(A)$

(ii) $u \in D(t_i)$ iff $i.u \in D(f(t_1, \dots, t_i, \dots, t_n))$ $1 \le i \le n$.

$D(A)$ is called a set of *occurrences* of A; $O(A)$ denotes the nonvariable subset of $D(A)$. The set of occurrences is partially ordered by the prefix ordering: $u \le v$ iff $(\exists w)\ u.w = v$, and $u < v$ iff $u \le v$ & $u \ne v$. When $u < v$ we then say u is outer to v and v is inner to u. u and w are said to be *independent*, denoted by $u <> w$ iff $u \not\le w$ and $w \not\le u$. The quotient u/v of two occurrences u and v is defined as: $u/v = w$ iff $v.w = u$.

We use $V(A)$ to denote the set of variables occurring in an object A. We define $A[u \leftarrow B]$ as the term A, in which the subterm at occurrence u has been replaced by the term B. We denote by A/u the subterm of A at occurrence u.

2.2 DEFINITION. A *substitution* σ is a mapping from V to $T(F, V)$, extended to an endomorphism of $T(F, V)$. We denote by Ω the set of all substitutions. If $\sigma \in \Omega$ and $A \in T(F, V)$, we write σA for application of σ to A. The domain of a substitution σ, denoted by $D(\sigma)$, contains the variables that are not mapped to themselves. The set of variables introduced by a substitution σ is defined as: $I(\sigma) = \cup V(\sigma x)$ for all x in $D(\sigma)$. The composition of substitutions σ and θ is defined as: $(\sigma \bullet \theta)x = \sigma(\theta x)$. $\sigma_{|W}$ denotes the *restriction* of the substitution σ to the subset W of V.

2.3 DEFINITION. An *equational axiom* $A = B$ is a pair of terms separated by the symbol $=$. An *equational theory* is a set E of equational axioms. E-equality $=_E$ is defined as the smallest congruence containing E and closed under replacement and instantiation, i.e., $=_E$ is generated by E as the smallest congruence containing all pairs $\sigma A = \sigma B$ for $A = B$ in E and σ in Ω.

E-equality is extended to substitutions as follows: $\sigma =_E \theta$ iff $\forall x \in V \ \ \sigma x =_E \theta x$.

We will write, for a subset W of V: $\sigma =_E \theta\ [W]$ iff $\forall x \in W \ \ \sigma x =_E \theta x$.

In the same way, σ is more general than θ under the equational theory E over W:

$$\sigma \le_E \theta\ [W] \text{ iff } \exists \eta\ \ \eta \bullet \sigma =_E \theta\ [W].$$

Two terms A and B are said to be E-unifiable iff there exists σ in Ω, such that $\sigma A =_E \sigma B$. $U_E(A, B)$ denotes the set of all E-unifiers of A and B. Let W be a finite set of variables containing $V = V(A) \cup V(B)$. We say that a set of substitutions Σ is a complete set of unifiers of A and B away from W iff:

(i) $\forall \sigma \in \Sigma \ \ D(\sigma) \subseteq V \ \&\ I(\sigma) \cap W = \varnothing$ (protection of W)

(ii) $\Sigma \subseteq U_E(A, B)$ (correctness)

(iii) $\forall \sigma \in U_E(A, B)\ \exists \theta \in \Sigma \ \ \theta \le_E \sigma\ [V]$ (completeness)

In addition, Σ is said to be *minimal* if and only if it also satisfies the condition:

$\forall \sigma, \theta \in \Sigma \ \ \sigma \ne \theta \Rightarrow \sigma \not\le_E \theta\ [V]$ (minimality)

An E-unification procedure is complete if it generates a complete set of E-unifiers away from W for all E-unifiable input terms, and is complete and minimal if it generates a complete and minimal set of E-unifiers away from W for all E-unifiable input terms. We also use the terms completeness and minimality when it is understood that certain conditions have been imposed on the input terms.

Given two terms A and B such that $V(A) \cap V(B) = \emptyset$†, a substitution σ is said to be an E-matcher from A to B iff $\sigma(A) =_E B$. In the same way as for E-unification we compare E-matchers by $\leq_E [V]$ and define complete (and minimal) sets of E-matchers and complete (and minimal) E-matching procedures.

2.4 DEFINITION. A *term rewriting system* (or *rewrite system*) is a set of directed equations $R = \{\alpha_i \to \beta_i\}$ such that variables appearing in β_i must also appear in α_i. $\alpha_i \to \beta_i$ is called a *rewrite rule*.

The reduction (or rewriting) relation $\to^R$ associated with R is the finest relation over $T(F, V)$, containing R and closed by substitution and replacement. Equivalently, we say that a term A reduces to a term B at occurrence u and write $A \to^R B$ iff:

$$\exists \alpha_k \to \beta_k \in R \quad \exists \eta \quad \exists u \in O(A) \; A/u = \eta(\alpha_k) \;\&\; B = A[u \leftarrow \eta(\beta_k)].$$

From now on we will use $\to$ for $\to^R$. We denote by $\xrightarrow{*}$ the reflexive, transitive closure of $\to$. The symmetric closure of $\xrightarrow{*}$ is denoted by $\overset{*}{\leftrightarrow}$, which is the same as the E-equality $=_E$, when R is considered a set of equational axioms. If $R = \{\alpha_i \to \beta_i\}$ is a term rewriting system, we then say that R describes the equational theory $E = \{\alpha_i = \beta_i\}$.

A term is said to be in *normal form* if it is not reducible. A substitution is *normalized* if each substitute wherein is in normal form.

A term rewriting system R is said to be *canonical* iff $\xrightarrow{*}$ is noetherian, i.e., there does not exist any infinite derivation sequence: $A_0 \to A_1 \to \ldots$, and $\xrightarrow{*}$ is confluent, i.e.,

$$\forall A, B, C \;\; (A \xrightarrow{*} B \;\&\; A \xrightarrow{*} C) \Rightarrow \exists D \; (B \xrightarrow{*} D \;\&\; C \xrightarrow{*} D).$$

An equivalent characterization of the confluence property is the Church-Rosser property:

$$\forall A, B \quad A =_E B \Rightarrow \exists D \; (A \xrightarrow{*} D \;\&\; B \xrightarrow{*} D).$$

When a term A reduces to a term B at occurrence u using the rule $\alpha_k \to \beta_k$ in R, we also write $A \to_{[u,k]} B$ or $A \to_{[u, \alpha_k \to \beta_k]} B$ (or $A \to_u B$ for abbreviation) to denote a reduction step. In this case, u is called a *redex* of A. A sequence of reduction steps is called a *reduction derivation*. For notational convenience, we sometimes write $A_0 \xrightarrow{*}_{[U,K]} A_n$ (or $A_0 \xrightarrow{*}_U A_n$ for abbreviation) to denote $A_0 \to_{[u_0, k_0]} \cdots \to_{[u_{n-1}, k_{n-1}]} A_n$, where $[U, K]$ denotes the sequence of $[u_i, k_i]$.

REMARK. The difference between the reduction relation and reduction derivation should be noticed. In particular, for any relation $A_0 \xrightarrow{*} A_n$ there may be more than one reduction derivation leading A_0 to A_n.‡ In this paper we mainly deal with derivations which are notationally distinguished from the relation by attaching the occurrence and rule associated with the reduction step, such as $\to_{[u,k]}$. $[u, k]$ is omitted only if no confusion arises.

† The requirement that $V(A) \cap V(B) = \emptyset$ is not a limitation on practical applications of matching, and is to avoid some of the subtleties brought up by more general definitions. See Burckert *et al.* (1987) for details.

‡ In the case of left-linear, nonoverlapping systems (see the next definition), derivations were nicely explained by Huet (1986, 1987) in terms of a categorical structure.

2.5 DEFINITION. Given a term rewriting system R and a set of function symbols F (constants are treated as 0-nary functions), F is divided into *defined* function symbols, denoted by F_D, and *constructors* denoted by F_C. An n-ary function symbol f is in F_D iff there is a rule of the form $f(t_1, ..., t_n) \rightarrow \beta$ in R. Otherwise it is in F_C.

A term rewriting system R is said to be *left-linear* if no variable appears more than once at the left hand side of any rule.

A term rewriting system R is said to be *nonoverlapping* if for any two rules $\alpha_i \rightarrow \beta_i$ and $\alpha_j \rightarrow \beta_j$ in R (with variables properly renamed such that $V(\alpha_i) \cap V(\alpha_j) = \varnothing$), and for any $u \in O(\alpha_i)$, α_i/u and α_j have no common instance, except when $u = \varepsilon$ and $i = j$. This is also called *nonambiguous* or *superposition free* in the literature.

A term rewriting system R is said to be *constructor-based* if it is left-linear, nonoverlapping, and has no defined function symbols appearing in the inner part of the left hand side of any rule.

Finally, a narrowing step from a term A to B at occurrence $u \in O(A)$ using $\alpha_k \rightarrow \beta_k$ in R is denoted by $A \rightsquigarrow_{[u, k, \rho]} B$ or $A \rightsquigarrow_{[u, \alpha_k \rightarrow \beta_k, \rho]} B$ (or $A \rightsquigarrow_\rho B$ for abbreviation), where $V(A) \cap V(\alpha_k) = \varnothing$, ρ is the most general unifier of A/u and α_k, and $B = \rho(A[u \leftarrow \beta_k])$. A sequence of narrowing steps is called a *narrowing derivation*. We may denote a narrowing derivation

$$A_0 \rightsquigarrow_{[u_0, k_0, \rho_0]} \cdots \rightsquigarrow_{[u_{n-1}, k_{n-1}, \rho_{n-1}]} A_n$$

by $A_0 \overset{*}{\rightsquigarrow}_{[U, K, \sigma]} A_n$, where $[U, K, \sigma]$ is the sequence of $[u_i, k_i, \rho_i]$ and $\sigma = \rho_{n-1} \bullet \ldots \bullet \rho_0$.

3. Outer Reduction and Outer Narrowing

In this section we are going to establish the following result for constructor-based term rewriting systems: for any narrowing derivation

$$A_0 \rightsquigarrow_{[u_0, k_0, \rho_0]} \cdots \rightsquigarrow_{[u_{n-1}, k_{n-1}, \rho_{n-1}]} A_n,$$

there exists an outer narrowing derivation

$$A_0 = B_0 \rightsquigarrow_{[v_0, i_0, \gamma_0]} \cdots \rightsquigarrow_{[v_{m-1}, i_{m-1}, \gamma_{m-1}]} B_m,$$

such that there exists a substitution θ, $\theta(B_m) = A_n$ and $\theta \bullet \gamma_{m-1} \bullet \ldots \bullet \gamma_0 = \rho_{n-1} \bullet \ldots \bullet \rho_0$, when restricted to $V(A_0)$. This will be proven by establishing the following results: we show in Subsection 3.1 that for every reduction derivation there exists an outer reduction derivation, and in Subsection 3.2 that for every outer reduction derivation there exists an outer narrowing derivation. Since for every narrowing derivation there exists a corresponding reduction derivation, we conclude that every narrowing derivation is subsumed by an outer narrowing derivation. In Subsection 3.3, we will provide a detailed comparison of outer narrowing with Reddy's lazy narrowing, Fribourg's innermost narrowing and the works on basic narrowing by Hullot and Réty.

Since we are primarily interested in the bindings for the variables occurring in the given term, we will assume, for notational convenience, that the substitution σ generated by a narrowing derivation $C_0 \overset{*}{\rightsquigarrow}_{[U, K, \sigma]} C_m$ is already restricted to the variables in C_0 whenever it is referenced. Also, because constructor-based term rewriting systems are a subclass of *closed linear* term rewriting systems that were studied by O'Donnell (1977), it is convenient to use some of the properties possessed by closed linear systems. We hence introduce the class of closed linear term rewriting systems and outer reductions wherein.

3.1 OUTER REDUCTION IN CLOSED LINEAR TERM REWRITING SYSTEMS

In this section we essentially use the *closure* property to show that an arbitrary reduction derivation of a closed linear term rewriting system can be rearranged to yield an outer reduction derivation.

When a term A is rewritten to yield a term B, some of the subterms in A might reappear (perhaps more than once) in B. The *residue map* defined below is intended to capture this rearrangement process by mapping each occurrence v in a term A to a set of occurrences in B which are "images" of v under the rearrangement.

3.1 DEFINITION. The residue map r with respect to a left-linear term rewriting system R is a function defined as follows:

$$\begin{array}{ll} \text{for all } v \in D(A) & \\ r[A \rightarrow_{[u,k]} B]v & \\ = \{u.w.(v/v') \mid \exists v''\ \alpha_k(v'') \in V(\alpha_k)\ \&\ \alpha_k(v'') = \beta_k(w)\ \&\ u.v'' = v'\} & \text{if } v > u \\ = \{v\} & \text{if } (u <> v \ or\ v < u) \\ = \varnothing & \text{otherwise} \end{array}$$

For example, with the rewrite rule $f(c(x)) \rightarrow g(x,x)$ and reduction $f(c(a)) \rightarrow_{\varepsilon} g(a,a)$, both occurrences of a in $g(a,a)$ are residues of a in $f(c(a))$. Note that if v is independent of u, or v is outer to u, then the residue of v is itself and unique. The definition given here treats redexes and other occurrences uniformly. There are two technical differences between this definition of residue and others such as the ones by O'Donnell (1977), Huet & Lévy (1979) and Réty (1987). First, we consider not only redexes, but any occurrence in a term. Secondly, an outer occurrence (which may not be a redex) remains to be a residue of itself.

The closure property, which was used by O'Donnell (1977) to ensure the confluence property, essentially says that inner and outer reductions can be switched.

3.2 DEFINITION. A term rewriting system R is *closed linear* iff R is left-linear and

$$\begin{array}{l} \text{(i) } \forall u,v \in O(A)\ (v < u\ \&\ A \rightarrow_{v} B\ \&\ A \rightarrow_{u} C) \\ \qquad\qquad \Rightarrow \exists D\ (B \overset{*}{\rightarrow}_{r[A \rightarrow_{v} B]u} D\ \&\ C \overset{*}{\rightarrow}_{r[A \rightarrow_{u} C]v} D) \\ \text{(ii) } \forall u \in O(A)\ (A \rightarrow_{u} B\ \&\ A \rightarrow_{u} C) \Rightarrow B = C \end{array}$$

The first clause essentially says that, if a term A can reduce to B at an outer redex and can also reduce to C at an inner redex, respectively, then there exists a term D such that D can be reduced from B by a sequence of inner reductions and from C by the reduction at the same outer redex. Note that either of these two sequences can be a null sequence if the corresponding set of residues is empty. The second clause says that if two different left hand sides match the same term, then the corresponding right hand sides must be the same.

It should be mentioned that O'Donnell's definition of closure is given in terms of subtree replacement systems and is more general in that the residue map r can be arbitrary as far as certain conditions are satisfied. The residue map r used in this paper is fixed to the preceding definition. From now on, when we refer to the term *closed linear* it is understood that the residue map is fixed as above. Notice also that a closed linear term rewriting system need not be nonoverlapping. For example, the following system, which has been used by Fages & Huet (1983) to show nonexistence of complete and minimal sets of E-unifiers in general, is closed linear:

$$R = \{ a * x \rightarrow x,\ f(x * y) \rightarrow f(y) \}.$$

3.3 PROPOSITION. *A term rewriting system is confluent if it is closed linear.*

PROOF. See O'Donnell (1977).

The residue map r can be extended to show how a set of occurrences is rearranged by a sequence of reductions (also see O'Donnell 1977; Huet & Lévy 1979; Réty 1987).

3.4 DEFINITION. Let $U = \{u_0, ..., u_{n-1}\}$ and S denote $A_0 \xrightarrow{*}_U A_n$. The extended residue map $\overset{*}{r}$ with respect to S is defined as follows:

for any $M \subseteq D(A_0)$

$$\overset{*}{r}[A_0 \xrightarrow{*}_U A_n]M = \overset{*}{r}[A_{n-1} \rightarrow_{u_{n-1}} A_n]\overset{*}{r}[A_0 \xrightarrow{*}_{U'} A_{n-1}]M$$

where $U' = \{u_0, ..., u_{n-2}\}$ & $n > 1$

$$\overset{*}{r}[A_0 \rightarrow_{u_0} A_1]M = \cup_{v \in M}\, r[A_0 \rightarrow_{u_0} A_1]v$$

We say that w is a residue of v, with respect to $A_0 \xrightarrow{*}_U A_n$, iff $w \in \overset{*}{r}[A_0 \xrightarrow{*}_U A_n]\{v\}$.

Intuitively, the above definition says that the residues of a set of occurrences are the union of the individual residues, and that a residue yielded by a sequence of reductions is the cascaded residue yielded by the individual residues in the sequence.

Before defining outer reduction we need to define an ordering on independent occurrences.

3.5 DEFINITION. Let A be a term. Two occurrences u and w in $O(A)$ are said to be *left independent*, denoted by $u <>_l w$, iff $u <> w$ and $u <_{lex} w$, where $<_{lex}$ is the lexicographic ordering on I^*.

For example, with the term $f(g(a), b)$, we will have $1 <>_l 2$ and $1.1 <>_l 2$. Left independence is a slightly generalized version of Prolog's leftmost computation rule which deals only with the flat structure at the literal level.

Outer reduction defined below requires that an inner reduction not be performed if it is followed (may or may not be immediately) by an outer reduction and it does not contribute to the reducibility of that outer reduction. In addition, independent reductions should be ordered by $<>_l$.

3.6 DEFINITION. A reduction derivation S: $A_0 \rightarrow_{[u_0, \alpha_0 \rightarrow \beta_0]} \cdots \rightarrow_{[u_{n-1}, \alpha_{n-1} \rightarrow \beta_{n-1}]} A_n$ is said to be *outer* iff

(a) for any u_i in S, if $\exists u_j$ in S, $j > i$, which is the *closest* to u_i, such that $\exists w \in O(A_i)$, $w < u_i$ and u_j is a residue of w; then $\exists v$ $\alpha_j((u_i/w).v) \in F$ & $A_i(u_i.v) \in F$ & $\alpha_j((u_i/w).v) \neq A_i(u_i.v)$; and

(b) suppose (a) is satisfied. If u_j is a residue of $w \in O(A_i)$ and $w <> u_i$, then $u_i <>_l w$.

Recall that $A_i(u_i)$ denotes the function symbol at occurrence u_i. $\alpha_j((u_i/w).v)$ thus denotes the function symbol in α_j which, by $\alpha_j((u_i/w).v) \neq A_i(u_i.v)$, disagrees with $A_i(u_i.v)$. Technically, Condition (a) says that if we try to reduce A_j at an outer occurrence w by the rule $\alpha_j \rightarrow \beta_j$, then there exists a function symbol at $u_i.v$, which would conflict with the one, denoted by $\alpha_j((u_i/w).v)$, at the corresponding position in α_j. Since we are dealing with left-linear rules where each variable in the left hand side occurs at most once, failure of matching or unification can only be caused by function symbol conflicts. Condition (b) simply orders independent reductions to further limit possible alternative reductions from a given term.

As an example, consider $R = \{ f(a, b, x) \rightarrow true,\ g(c) \rightarrow a,\ h(a) \rightarrow b \}$. The reduction sequence

$$f(a, b, g(c)) \rightarrow_3 f(a, b, a) \rightarrow_\varepsilon true$$

is not outer because it violates Condition (a). The reduction sequence

$$f(g(c), h(a), c) \rightarrow_2 f(g(c), b, c) \rightarrow_1 f(a, b, c)$$

is not outer either, since it violates Condition (b).

Note that outer reduction does not correspond to the well-known outermost reduction rule, by which only outermost redex occurrences can be used. For example, with the preceding term rewriting system, the following derivation is outer though it is not outermost (it does not even reduce the term to its normal form):

$$f(a, b, g(c)) \rightarrow_3 f(a, b, a).$$

The definition of outer reduction used in this paper is essentially equivalent to the one called *outside-in* reduction by Huet & Lévy (1979) for the class of left-linear, nonambiguous term rewriting systems. We present similar results below under a slightly less restrictive condition by the closure property.†

Our interest in outer reduction lies in the following fact: for any reduction derivation $A \overset{*}{\rightarrow}_{[U, K]} B$ of a closed linear system, there exists an outer reduction derivation $A \overset{*}{\rightarrow}_{[V, J]} B$. This can in fact be obtained from the closure property.

3.7 LEMMA. *Let R be a closed linear term rewriting system. For any pair of reductions*

$$S:\ A \rightarrow_{[u, i]} B \rightarrow_{[v, j]} C$$

there exists an outer reduction derivation $A \overset{*}{\rightarrow}_U C$.

PROOF. If S is already outer nothing needs to be done. If $v <>_l u$ then switch the two reductions. In the case $v < u$ and A is reducible at v by the j-th rule, using the closure property we get

$$S':\ A \rightarrow_{[v, j]} B' \overset{*}{\rightarrow}_{r[A \rightarrow_v B']u} C.$$

Since all of the occurrences in $r[A \rightarrow_v B']u$ are mutually independent, $r[A \rightarrow_v B']u$ can be ordered by $<>_l$.

3.8 LEMMA. *Let R be a closed linear term rewriting system. For every reduction derivation* $A_0 \overset{*}{\rightarrow}_{[U, K]} A_n$, *there exists an outer reduction derivation* $A_0 = B_0 \overset{*}{\rightarrow}_{[V, J]} B_m$ *such that* $A_n = B_m$.

PROOF. The outer reduction sequence $B_0 \overset{*}{\rightarrow}_{[V, J]} B_m$ is actually a "sorted" sequence of the given sequence $A_0 \overset{*}{\rightarrow}_{[U, K]} A_n$ with some reductions at inner occurrences duplicated, because reductions at outer occurrences are performed before inner reductions whenever possible. We show this by induction on the number of reduction steps in the given derivation. By definition any singleton is outer. Assume that $B_0 \overset{*}{\rightarrow}_{[V_l, J_l]} B_l$ is an outer sequence for $A_0 \overset{*}{\rightarrow}_{[U_i, K_i]} A_i$, where $A_i = B_l$, and show that there exists an outer sequence for $A_0 \overset{*}{\rightarrow}_{[U_i, K_i]} A_i \rightarrow_{[u_i, k_i]} A_{i+1}$. For this purpose, move the last reduction $B_l \rightarrow_{[u_i, k_i]} A_{i+1}$ in $B_0 \overset{*}{\rightarrow}_{[V_l, J_l]} B_l \rightarrow_{[u_i, k_i]} A_{i+1}$ to the left as far as possible using Lemma 3.7 (the number of steps is at most $l+1$). The rearranged sequence $B_0 \overset{*}{\rightarrow}_{[V_{l+1}, J_{l+1}]} A_{i+1}$ is obviously outer.

† Huet and Lévy (1979) mentioned the question about whether the nonambiguity condition was necessary in their approach.

3.2 SUBSUMPTION OF NARROWING BY OUTER NARROWING

In order to define outer narrowing and establish its relationship with outer reduction, we need to keep track of subterms that are rearranged by narrowing derivations. We thus define a residue map for narrowing derivations.

3.9 DEFINITION. The residue map n with respect to a narrowing step is defined as follows:

$$n[A -\!\!>_{[u,k,\rho]} B]v = r[\rho(A) \rightarrow_{[u,k]} B]v \qquad v \in O(A)$$

The extension of n to a narrowing derivation, denoted by $\overset{*}{n}$, is similarly defined:

$$\overset{*}{n}[A_0 \overset{*}{-\!\!>}_{[U,K,\sigma]} A_n]M = \overset{*}{r}[\sigma(A_0) \xrightarrow{*}_{[U,K]} A_n]M \qquad M \subseteq O(A_0).$$

We say that w is a residue of v, with respect to $A_0 \overset{*}{-\!\!>}_{[U,K,\sigma]} A_n$, iff

$$w \in \overset{*}{n}[A_0 \overset{*}{-\!\!>}_{[U,K,\sigma]} A_n]\{v\}.$$

Note that $\overset{*}{n}$ is well-defined since for every narrowing derivation $A_0 \overset{*}{-\!\!>}_{[U,K,\sigma]} A_n$ there always exists a reduction derivation of the form $\sigma(A_0) \xrightarrow{*}_{[U,K]} A_n$.

Outer narrowing is also defined with respect to narrowing derivations, which, as in the case of outer reduction, essentially says that no later narrowing steps can be performed earlier in the sequence.

3.10 DEFINITION. A narrowing derivation

$$S : A_0 -\!\!>_{[u_0, \alpha_0 \rightarrow \beta_0, \rho_0]} \cdots -\!\!>_{[u_{n-1}, \alpha_{n-1} \rightarrow \beta_{n-1}, \rho_{n-1}]} A_n$$

is said to be *outer* iff

(a) for any u_i in S, if $\exists u_j$ in S, $j > i$, which is the *closest* to u_i, such that $\exists w \in O(A_i)$, $w < u_i$ and u_j is a residue of w (with respect to S); then $\exists v \ \alpha_j((u_i/w).v) \in F \ \& \ A_i(u_i.v) \in F \ \& \ \alpha_j((u_i/w).v) \neq A_i(u_i.v)$; and

(b) suppose (a) is satisfied. If u_j is a residue of $w \in (A_i)$ and $w <> u_i$, then $u_i <>_l w$.

We give two terminologies below that will be frequently referenced in the rest of this paper.

3.11 DEFINITION. A term is called a *constructor term* if it contains constructors and variables only. A substitution $\sigma = \{x_1/t_1, ..., x_n/t_n\}$ is said to be *constructor-based* if all of t_i's are constructor terms.

3.12 LEMMA. *Consider a constructor-based term rewriting system R. Let $\sigma(B_0) = A_0$, where A_0 and B_0 are terms and σ a constructor-based substitution when restricted to $V(B_0)$. For any reduction derivation*

$$(i)\ A_0 \rightarrow_{[u_0, \alpha_0 \rightarrow \beta_0]} \cdots \rightarrow_{[u_{n-1}, \alpha_{n-1} \rightarrow \beta_{n-1}]} A_n,$$

there exist a narrowing derivation

$$(ii)\ B_0 -\!\!>_{[u_0, \alpha_0 \rightarrow \beta_0, \rho_0]} \cdots -\!\!>_{[u_{n-1}, \alpha_{n-1} \rightarrow \beta_{n-1}, \rho_{n-1}]} B_n,$$

and ζ_i, $0 \leq i \leq n$, such that ζ_i is constructor-based, $\zeta_i(B_i) = A_i$ and $\zeta_n \bullet \rho_{n-1} \bullet \cdots \bullet \rho_0 = \sigma$. Furthermore, if (i) is outer then (ii) is outer.

Conversely, for every narrowing derivation (ii), there exist a reduction derivation

$$(iii)\ \rho_{n-1} \bullet \cdots \bullet \rho_0(B_0) = C_0 \rightarrow_{[u_0, \alpha_0 \rightarrow \beta_0]} \cdots \rightarrow_{[u_{n-1}, \alpha_{n-1} \rightarrow \beta_{n-1}]} C_n$$

and ζ_i, $0 \leq i \leq n$, such that ζ_i is constructor-based, $\zeta_i(B_i) = C_i$, and $B_n = C_n$, i.e., ζ_n is the identity substitution. Furthermore, if (ii) is outer then (iii) is outer.

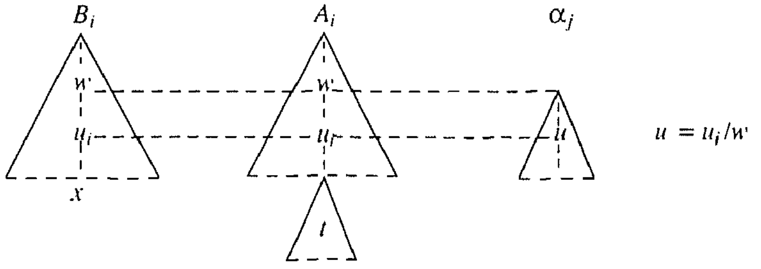
PROOF. The correspondence between reduction and narrowing derivations is a well-known result in some more general settings. The result proved by Hullot (Hullot 1980) which requires that σ be normalized is applicable here since σ is constructor-based and therefore normalized. Here we prove that the outer properties preserve.

($\Rightarrow$) Show that Condition (a) for outer narrowing preserves from that of outer reduction.

Let u_j be the closest to u_i such that u_j is a residue of $w' \in O(A_i)$ and $w' < u_i$. Then u_j is the closest to u_i such that u_j is a residue of $w' \in O(B_i)$ and $w' < u_i$, with respect to the narrowing derivation. Since (i) is outer and R is constructor-based, $\alpha_j(u_i/w')$ must be a constructor and $A_i(u_i)$ must be a defined function; thus $\exists v$, $v = \varepsilon$ such that $\alpha_j((u_i/w').v) \neq A_i(u_i.v)$, or simply $\alpha_j(u_i/w') \neq A_i(u_i)$. That is, matching A_i/w' with α_j would result in a conflict at u_i. We show that $\alpha_j(u_i/w') \neq B_i(u_i)$.

First of all, the symbol $A_i(u_i)$ must be a defined function symbol since A_i is reducible at u_i. Because ζ_i is constructor-based and $\zeta_i(B_i) = A_i$, we have $B_i(u_i) = A_i(u_i)$. That is, $B_i(u_i) = A_i(u_i) \neq \alpha_j(u_i/w')$. (The situation is illustrated in Figure 1.) Therefore, Condition (a) is satisfied for (ii), i.e., $\alpha_j(u_i/w') \neq B_i(u_i)$. We conclude that (ii) is outer by observing that Condition (b) trivially preserves.

($\Leftarrow$) Similar and thus omitted.



$\zeta_i = \{ \cdots, x/t, \cdots \}$

Figure 1: An illustration of the proof of Lemma 3.12.

We now come to the main result of this section: every narrowing derivation of a constructor-based term rewriting system is subsumed by an outer narrowing derivation.

3.13 THEOREM. *Let R be a constructor-based term rewriting system. For every narrowing derivation*

(a) $C_0 \overset{*}{\leadsto}_{[U, K, \sigma]} C_m$

there exists an outer narrowing derivation

(b) $C_0 \overset{*}{\leadsto}_{[U', K', \theta]} D_n$

such that $\exists\tau$, $\tau \bullet \theta = \sigma$ *and* $\tau(D_n) = C_m$.

PROOF. Let the narrowing derivation (a) be (ii) of Lemma 3.12 without the outer assumption. We then get (iii) from (ii). For which there exists an outer reduction derivation (Lemma 3.8)

$$A_0 \rightarrow_{[u_0, \alpha_0 \rightarrow \beta_0]} \cdots \rightarrow_{[u_{n-1}, \alpha_{n-1} \rightarrow \beta_{n-1}]} A_n,$$

where $\sigma(C_0) = A_0$ and $A_n = C_m$. Since R is constructor-based it is easy to show that the composition of all substitutions along a narrowing derivation, restricted to the set of variables in the initial term to be narrowed, is constructor-based. Thus $\sigma_{|V(C_0)}$ is constructor-based. By Lemma 3.12 again, we get an outer narrowing derivation

$$C_0 = D_0 \leadsto_{[u_0, \alpha_0 \rightarrow \beta_0, \rho_0]} \cdots \leadsto_{[u_{n-1}, \alpha_{n-1} \rightarrow \beta_{n-1}, \rho_{n-1}]} D_n \qquad (*)$$

and ζ_n, $\zeta_n(D_n) = A_n = C_m$ such that $\zeta_n \bullet \rho_{n-1} \bullet \cdots \bullet \rho_0 = \sigma$. The derivation (*) is therefore the desired derivation (b) where $\theta = \rho_{n-1} \bullet \cdots \bullet \rho_0$.

3.14 COROLLARY. Let Σ be the set of all E-unifiers generated by narrowing on two terms P and Q. Then outer narrowing generates Σ', $\Sigma' \subseteq \Sigma$ and for any σ in Σ there exists σ' in Σ', $\sigma' \leq_E \sigma$ $[V(P, Q)]$.

3.15 EXAMPLE. Consider the following system about sum and product over natural numbers:

$$
\begin{aligned}
R = \{ & \\
& (1)\quad 0 + x \rightarrow x \\
& (2)\quad s(x) + y \rightarrow s(x + y) \\
& (3)\quad 0 * x \rightarrow 0 \\
& (4)\quad s(x) * y \rightarrow y + x * y \\
\}. &
\end{aligned}
$$

The system is constructor-based. Suppose we want to unify the terms $x_1 * y_1$ and $s(x_1)$. One of the outer derivations is given below. We use a new binary operator == to iterate the narrowing process on the two terms.

$$
\begin{aligned}
& x_1 * y_1 == s(x_1) \rightsquigarrow_{[\varepsilon,\, 4,\, \{x_1/s(x_2)\}]} \\
& y_1 + x_2 * y_1 == s(s(x_2)) \rightsquigarrow_{[\varepsilon,\, 2,\, \{y_1/s(y_2)\}]} \\
& s(y_2 + x_2 * s(y_2)) == s(s(x_2)) \rightsquigarrow_{[1,\, 2,\, \{y_2/s(y_3)\}]} \\
& s(s(y_3 + x_2 * s(s(y_3)))) == s(s(x_2)) \rightsquigarrow_{[1.1,\, 1,\, \{y_3/0\}]} \\
& s(s(x_2 * s(s(0)))) == s(s(x_2)) \rightsquigarrow_{[1.1,\, 3,\, \{x_2/0\}]} \\
& s(s(0)) == s(s(0))
\end{aligned}
$$

The derivation generates $s(0)$ for x_1 and $s(s(0))$ for y_1. Note that from the term $y_1 + x_2 * y_1$, narrowing on the inner term $x_2 * y_1$ using either the third rule or the fourth rule can result in an infinite number of non-outer derivations, which, by Theorem 3.13, are all redundant.

It is important to mention that Lemma 3.12 and Theorem 3.13 may not be true of other classes of term rewriting systems, even though Lemma 3.8 is valid for all closed linear term rewriting systems.

3.16 COUNTEREXAMPLE. Consider the following nonoverlapping term rewriting system:

$$R = \{f(g(d)) \rightarrow true,\ g(c) \rightarrow g(d)\}.$$

With the term $f(g(x))$ and the substitution $\sigma = \{x/c\}$, we have $\sigma(f(g(x))) = f(g(c))$ and

$$\text{(i)}\ f(g(c)) \rightarrow f(g(d)) \rightarrow true$$

$$\text{(ii)}\ f(g(x)) \rightsquigarrow_{\{x/c\}} f(g(d)) \rightsquigarrow_{\{\}} true.$$

The reduction derivation (i) is outer while the corresponding narrowing derivation (ii) is not, since when $f(g(x))$ is unified with $f(g(d))$ no function symbol conflict exists. It is easy to check that this narrowing derivation is not subsumed by any outer narrowing derivation.

3.3 COMPARISON WITH OTHER NARROWING METHODS

LAZY NARROWING. Reddy (1985) defined a denotational semantics for the purpose of extending functional programming to functional logic programming. Under this semantics, the notion of equality is based on an equality function: two terms (called expressions) are equivalent if their denotations are the same. A lazy narrowing strategy was *outlined* in terms of a demand-driven unification algorithm. The key step in the algorithm is Step 2:

> "when one of the two expressions to be unified has a function application at the outermost level, then it is narrowed until it has a constructor at the outermost level."

We should notice that the phrase *two expressions* above may actually refer to subexpressions of the given expressions, and the outermost function symbol of a subexpression may not necessarily be an outermost redex of the whole expression at which narrowing is possible. Thus lazy narrowing is not the outermost narrowing strategy which ignores narrowing steps at all inner occurrences. It can be shown that a lazy narrowing derivation is an outer narrowing derivation but the reverse may not be true. Phrased in other words, lazy narrowing is outer narrowing with every defined function symbol eventually disappearing. As an example, consider

$$R = \{ f(a, y) \rightarrow y,\ g(c(x)) \rightarrow g(x),\ g(b) \rightarrow d \}$$

and unification of $f(z_1, z_2)$ and $g(z_3)$. From the following outer narrowing derivation

$$f(z_1, z_2) == g(z_3) \ {\sim}{>}_{[\varepsilon,\, f(a, y) \rightarrow y,\, \{z_1/a\}]} \ z_2 == g(z_3)$$

we get an E-unifier $\{z_1/a, z_2/g(z_3)\}$ for the two given terms. Because any narrowing on $g(z_3)$ can only generate compared but less general E-unifiers, these derivations can be dropped.†

The derivation above, however, is not a lazy narrowing derivation. First, $g(z_3)$ is not an expression having a constructor at the outermost level; and secondly, a solution containing $z_2/g(z_3)$ is not admissible by Reddy's denotational semantics. Thus, lazy narrowing will carry out narrowings on $g(z_3)$, producing an infinite number of solutions. As the reader can see, outer narrowing is operationally similar to lazy narrowing. The major difference between the two stems from the difference of the underlying semantics; lazy narrowing is aimed at computing denotations of expressions, while the purpose of outer narrowing is to generate unifiers modulo an equational theory. Lazy narrowing does not yield a complete procedure for E-unification and was not designed for that purpose. Our results show that a form of lazy strategy can be adopted for E-unification without losing completeness for constructor-based term rewriting systems.

INNERMOST NARROWING. We will restrict our discussion to the unit case corresponding to term rewriting systems, though Fribourg's results on innermost narrowing (Fribourg 1985) were obtained for the general case of equational Horn clauses.

A term $f(t_1, \ldots, t_n)$ is said to be *innermost* if all t_i's are constructor terms and f is a defined function symbol. Given a term, the innermost narrowing strategy selects an innermost term to narrow. The essential condition for completeness is called *well-innermost-reducing*, which requires that *every* ground innermost term be reducible. We have seen at the outset of this paper that

$$f(g(x), x) \ {\sim}{>}_{[\varepsilon,\, f(y, a) \rightarrow \mathit{true},\, \{x/a\}]} \ \mathit{true}$$

is not innermost, and its omission results in loss of a solution. The problem is that $f(g(a), a)$ is not well-innermost-reducing, because $g(a)$ is not reducible. Though a number of sufficient conditions were given by Fribourg to guarantee well-innermost-reducing, it appears difficult to mechanically check the satisfaction of these conditions in general, since all of them must be satisfied for all (ground) terms. The difficulty stems from the fact that functions, as defined by rewrite rules, are often *partial* functions over the domain of terms. For example, a rule such as $g(b) \rightarrow c$ defines a partial function on the set of terms. When such rules are present in the program, completeness can be easily lost. In practice, however, many programs do define total functions.

† Note that outer narrowing does not automatically prune this type of search space and should be used in combination with other methods such as recurrent description of sets of E-unifiers (Réty *et al.* 1985).

When innermost narrowing is complete, an innermost narrowing derivation is in general shorter than the corresponding outer narrowing derivation that generates the same solution, because innermost narrowing avoids repeated evaluation of inner terms. However, the converse may also be true when *variable-dropping* rules are present in the program (a rewrite rule $\alpha \rightarrow \beta$ is variable-dropping if $V(\beta) \subset V(\alpha)$). For example, with the rewrite system given earlier (Example 3.15), we have

innermost: $x * (0+y) \,{-}{>}_{[2,\, 1,\, \{\}]}\, x * y \,{-}{>}_{[\varepsilon,\, 3,\, \{x/0\}]}\, 0$
outer: $x * (0+y) \,{-}{>}_{[\varepsilon,\, 3,\, \{x/0\}]}\, 0$

BASIC NARROWING. The idea behind the concept of basic narrowing (Hullot 1980) is to avoid narrowing steps (and then subsequent sequences) on subterms that are introduced by instantiation. Réty (1987) discovered that the "basic" concept did not automatically extend to *normalized narrowing* (i.e., every narrowing step is followed by simplification), and showed a method to compute the basic occurrences which preserve the completeness of the solution set. Since we deal with constructor-based systems in this paper, all the substitutes generated by any narrowing step for the variables appearing in the given term must be constructor terms and thus not narrowable. Therefore, any narrowing derivation in this class is trivially basic.

In his paper, in order to compare various narrowing methods, Réty established a general *commutation* result of narrowing for arbitrary term rewriting systems. The term commutation is similar to what we have called rearrangement. To deal with non-left linear systems, Réty introduced the dual of residual notion, called *antecedent*, and extended it to narrowing. Our definition of residue map allows outer occurrences to be carried over to be residues when rewriting (or narrowing) is performed at an inner occurrence. Thus if we have a reduction $t \rightarrow_v t'$ where $u < v$ for some u in $O(t)$, a residue of u may not represent the same subterm as u; i.e., if $u' \in O(t')$ is a residue of u, t'/u' may not necessarily be identical to t/u. Réty's definition disallows u to be an antecedent of u'. Our definition of residue allows us to describe formally how narrowing steps at outer occurrences can be "moved" to the front of a narrowing sequence. However, it should be emphasized that this rearrangement is achieved under the strong restriction that the underlying term rewriting systems be constructor-based.

4. Complete and Minimal Sets of E-Matchers

In this section, we apply the result of last section to the following E-matching problem: given a constructor-based rewrite system R and two terms t_1 and t_2 with disjoint sets of variables, where t_2 is a constructor term, find a complete and minimal set of E-matchers from t_1 to t_2. This special case of E-matching may arise in practice; for example, compute pairs of values for x and y such that their sum is equal to a certain natural number by the familiar rewrite system:

$$R = \{\ 0+x \rightarrow x,\quad s(x)+y \rightarrow s(x+y)\ \}.$$

We first show that for this matching problem *unrelated* outer narrowing derivations generate uncompared E-matchers, i.e., no one is more general than the other. Based on a known result of closed linear term rewriting systems (You & Subrahmanyam 1986b) and the result of last section on the subsumption of narrowing by outer narrowing, we conclude that outer narrowing generates complete and minimal sets of E-matchers.

4.1 DEFINITION. Let A_0 be a term and σ_1 and σ_2 be normalized substitutions. Two reduction derivations $\sigma_1(A_0) \xrightarrow{*}_{[U_1, K_1]} C_1$ and $\sigma_2(A_0) \xrightarrow{*}_{[U_2, K_2]} C_2$ are said to be *unrelated* if $[U_1, K_1]$ and $[U_2, K_2]$ are no prefixes of each other.

Similarly, two narrowing derivations $A_0 \overset{*}{\rightsquigarrow}_{[U_1, K_1, \sigma_1]} C_1$ and $A_0 \overset{*}{\rightsquigarrow}_{[U_2, K_2, \sigma_2]} C_2$ are said to be *unrelated* if $[U_1, K_1]$ and $[U_2, K_2]$ are no prefixes of each other.

The two lemmas below are prepared for Theorem 4.4 in which we will show unrelated outer narrowing derivations must generate uncompared E-matchers.

4.2 LEMMA. *Let R be a constructor-based term rewriting system, P a term, and C a constructor-based term. For any normalized substitutions σ_1 and σ_2 and outer reduction derivations*

$$(i)\ \sigma_1(P) \xrightarrow{*}_{[U,K]} C$$
$$(ii)\ \sigma_2(P) \xrightarrow{*}_{[V,J]} C$$

(i) and (ii) are unrelated $\Rightarrow$ $\sigma_1 \not\geq \sigma_2$ *and* $\sigma_2 \not\geq \sigma_1$.

PROOF. Assume $\sigma_1 \leq \sigma_2$. The proof is similar for the symmetric case $\sigma_2 \leq \sigma_1$. From this assumption and the fact that σ_1 and σ_2 are normalized we have

$$\exists\tau,\ \tau \text{ is normalized and } \tau\bullet\sigma_1 = \sigma_2.$$

For any reduction derivation reducing a term to a constructor term, the outermost defined function symbol(s) must be reduced eventually, possibly repeatedly. Thus (i) and (ii) can be divided into sub-derivations of the form

$$(\text{i'})\ \sigma_1(P) = A_0 \xrightarrow{*}_{[U_0, K_0]} \cdots \xrightarrow{*}_{[U_{n-1}, K_{n-1}]} A_n = C$$
$$(\text{ii'})\ \sigma_2(P) = \tau\bullet\sigma_1(P) = B_0 \xrightarrow{*}_{[V_0, J_0]} \cdots \xrightarrow{*}_{[V_{m-1}, J_{m-1}]} B_m = C$$

such that the last reduction in each sub-derivation, say

$$A_i \xrightarrow{*}_{[U_i, K_i]} A_{i+1},$$

occurs at u', where u' is a (unique) residue of u, $A_i(u)$ is a defined function symbol and for any occurrence v, if $A_i(v)$ is a defined function symbol then $u <> v$ or $u < v$.

Because (i') and (ii') are unrelated, there exist nonempty sequences $[U_i, K_i]$ and $[V_i, J_i]$ such that $[U_i, K_i] \neq [V_i, J_i]$. Let $[U_i, K_i]$ and $[V_i, J_i]$ be the *first* such pair that occurred in the two derivations (i') and (ii'), respectively. Since the reduction steps before i are the same for both derivations (at the same occurrences and using the same rules), it is easy to show that $B_i = \tau(A_i)$. Thus, by the definition of outer reduction, the last reduction step in $A_i \xrightarrow{*}_{[U_i, K_i]} A_{i+1}$ and the last reduction step in $B_i \xrightarrow{*}_{[V_i, J_i]} B_{i+1}$ will take place at the same occurrence, say u, in order to change the same function symbol $A_i(u) = B_i(u)$. Let $f = A_i(u) = B_i(u)$. Now these two reductions will use either the same rule, or different rules defining f.

Case a: using two different rules.

Let the rules be $\alpha_j \rightarrow \beta_j$ and $\alpha_l \rightarrow \beta_l$. Because R is constructor-based (and thus left-linear and nonoverlapping), $\exists v$, $\alpha_j(v)$ and $\alpha_l(v)$ are two constructors such that $\alpha_j(v) \neq \alpha_l(v)$. Denote the subterms with these two constructors as the principal functors by $c(t)$ and $d(s)$, respectively, and assume they are a pair of outermost disagreed constructors, which always exists. By the definition of outer reduction again, there exist two outer reduction derivations

$$A_i/u \xrightarrow{*} c(t) \text{ and } B_i/u \xrightarrow{*} d(s).$$

From which it is easy to show by induction that for any τ, $\tau(A_i/u) \xrightarrow{*} \tau(c(t))$. From $B_i = \tau(A_i)$, we have $B_i/u = \tau(A_i/u)$. Therefore $\tau(c(t)) \overset{*}{\leftrightarrow} d(s)$, i.e., $\tau(c(t)) =_E d(s)$. However, the constructors c and d cannot be changed. That is, $\tau(c(t)) \neq d(s)$ and there do not exist reductions leading them to an identical term. This contradicts the Church-Rosser property.

Case b: using the same rule.

Let the rule be $\alpha_j \to \beta_j$. In this case, the two very last reductions take place at the same occurrence and using the same rule. By the definition of outer reduction and the assumption $[U_i, K_i] \neq [V_i, J_i]$, we will end up reducing both A_i and B_i at $u.v$, where $A_i(u.v)$ and $B_i(u.v)$ are the rightmost outermost defined function symbols such that $A_i(u.v) = B_i(u.v) \neq \alpha_j(v)$; otherwise we would have $[U_i, K_i] = [V_i, J_i] = \{[u, j]\}$. We thus have sub-derivations issuing from $A_i/u.v$ and $B_i/u.v$. Since there are only a finite number of reductions in $[U_i, K_i]$ and $[V_i, J_i]$, continuing in this way it will eventually either give us two identical sequences, which contradicts the assumption $[U_i, K_i] \neq [V_i, J_i]$, or lead to Case a.

Therefore, $\sigma_1 \leq \sigma_2$ cannot be true.

4.3 LEMMA. *Let R be a constructor-based term rewriting system, P a term, and C a constructor-based term. Let C_1 and C_2 be instances of C, and τ_1 and τ_2 be substitutions such that $\tau_1(C_1) = C$ and $\tau_2(C_2) = C$. Then for any two outer narrowing derivations*

(i) $P \overset{*}{\rightsquigarrow}_{[U_1, K_1, \sigma_1]} C_1$

(ii) $P \overset{*}{\rightsquigarrow}_{[U_2, K_2, \sigma_2]} C_2$

(i) and (ii) are unrelated $\Rightarrow$ $\tau_1 \bullet \sigma_1 \ngeq \tau_2 \bullet \sigma_2$ *and* $\tau_2 \bullet \sigma_2 \ngeq \tau_1 \bullet \sigma_1$.

PROOF. Notice that $\tau_1 \bullet \alpha_1$ and $\tau_2 \bullet \alpha_2$ are both constructor-based. (Recall that we have assumed that σ_1 and σ_2 are already restricted to $V(P)$.) Based on Lemma 3.12, for the two given outer narrowing derivations, we have two outer reduction derivations

$$\sigma_1(P) \xrightarrow{*}_{[U_1, K_1]} C_1 \text{ and } \sigma_2(P) \xrightarrow{*}_{[U_2, K_2]} C_2.$$

From which we can get

$$\tau_1 \bullet \sigma_1(P) \xrightarrow{*}_{[U_1, K_1]} \tau_1(C_1) = C$$
$$\tau_2 \bullet \sigma_2(P) \xrightarrow{*}_{[U_2, K_2]} \tau_2(C_2) = C.$$

By Lemma 4.2 we have $\tau_1 \bullet \sigma_1 \ngeq \tau_2 \bullet \sigma_2$ and $\tau_2 \bullet \sigma_2 \ngeq \tau_1 \bullet \sigma_1$.

4.4 THEOREM. *Let R be a constructor-based term rewriting system, P a term, and C a constructor-based term. Let C_1 and C_2 be instances of C, and τ_1 and τ_2 be such that $\tau_1(C_1) = C$ and $\tau_2(C_2) = C$. Then for any two outer narrowing derivations*

(i) $P \overset{*}{\rightsquigarrow}_{[U_1, K_1, \sigma_1]} C_1$

(ii) $P \overset{*}{\rightsquigarrow}_{[U_2, K_2, \sigma_2]} C_2$

(i) and (ii) are unrelated $\Rightarrow$ $\tau_1 \bullet \sigma_1 \ngeq_E \tau_2 \bullet \sigma_2$ $[V(P)]$ *and* $\tau_2 \bullet \sigma_2 \ngeq_E \tau_1 \bullet \sigma_1$ $[V(P)]$.

PROOF. Suppose $\tau_1 \bullet \sigma_1 \leq_E \tau_2 \bullet \sigma_2$ $[V(P)]$. The proof is similar for the symmetric case. Since $\tau_1 \bullet \sigma_1$ and $\tau_2 \bullet \sigma_2$ are constructor-based, $\exists \kappa$, κ is constructor-based and $\kappa \bullet \tau_1 \bullet \sigma_1 =_E \tau_2 \bullet \sigma_2$ $[V(P)]$. Now both $\kappa \bullet \tau_1 \bullet \sigma_1$ and $\tau_2 \bullet \sigma_2$ are constructor-based, we then must have $\kappa \bullet \tau_1 \bullet \sigma_1 = \tau_2 \bullet \sigma_2$, i.e., $\tau_1 \bullet \sigma_1 \leq \tau_2 \bullet \sigma_2$, which is not possible by Lemma 4.3.

It remains to show that outer narrowing generates complete sets of E-matchers. First, we have a completeness result for the class of closed linear rewrite systems.

4.5 LEMMA. *Let R be a closed linear term rewriting system, P_0 a term and C a constructor-based term. For any E-matcher σ from P_0 to C, there exists a narrowing derivation*

$$P_0 \rightsquigarrow_{[u_0, k_0, \rho_0]} \cdots \rightsquigarrow_{[u_{n-1}, k_{n-1}, \rho_{n-1}]} P_n$$

such that $\zeta \bullet \rho_{n-1} \bullet \ldots \bullet \rho_0 \leq_E \sigma$ $[V(P_0)]$, where ζ is a most general match from P_n to C.

If R is also terminating the lemma easily follows from Hullot's results (Hullot 1980). A closed linear term rewriting system, however, need not possess the termination property. In You & Subrahmanyam (1986b), we defined a transformation process whose termination guarantees the existence of the corresponding narrowing derivation. The transformation process trivially terminates for the case discussed here simply because C is not reducible and R is strictly left-linear. The details are omitted here.

4.6 LEMMA. *Let R be a constructor-based term rewriting system, P a term and C a constructor-based term. Let $P \overset{*}{\rightsquigarrow}_{\sigma} B$ be a narrowing derivation such that B is matchable to C. Let ξ_B be a most general match from B to C. Then there exists an outer narrowing derivation $P \overset{*}{\rightsquigarrow}_{\theta} D$, such that if ξ_D is a most general match from D to C then $\xi_D \bullet \theta \leq_E \xi_B \bullet \sigma$ $[V(P)]$.*

PROOF. By Theorem 3.13, the outer narrowing derivation exists and $\exists \tau$, $\tau \bullet \theta = \sigma$ and $\tau(D) = B$. Hence, $\xi_B \bullet \tau(D) = \xi_B(B) = C$. $\xi_B \bullet \tau$ is therefore a match from D to C. However, since ξ_D is a most general one, we have

$$\xi_D \leq \xi_B \bullet \tau \Rightarrow \xi_D \bullet \theta \leq \xi_B \bullet \tau \bullet \theta = \xi_B \bullet \sigma \Rightarrow \xi_D \bullet \theta \leq_E \xi_B \bullet \sigma \ [V(P)].$$

The interest of the lemma blow lies in the fact that for any pair of narrowing derivations, neither can be a prefix of the other; they must therefore be unrelated. The proof is obvious.

4.7 LEMMA. *Let R be a constructor-based term rewriting system, P be a term and C a constructor-based term. Assume P is matchable to C, i.e., there exists σ, $\sigma(P) = C$. Then P is constructor-based and not narrowable.*

We now give the main result of this section.

4.8 THEOREM. *Let R be a constructor-based term rewriting system, P be a term and C a constructor term such that $V(P) \cap V(C) = \emptyset$. Let W be a set of variables containing $V(P)$ but disjoint with $V(C)$. The set of all outer narrowing derivations issuing from P yields a complete and minimal set of E-matchers from P to C, away from W.*

PROOF. The correctness of outer narrowing for the E-matching problem considered here follows from Hullot (1980). Narrowing is complete (Lemma 4.5). For any E-matcher generated by a narrowing derivation, there is an outer narrowing derivation generating a more general E-matcher (Lemma 4.6); thus outer narrowing is complete. Any two E-matchers must be generated by two unrelated outer narrowing derivations (Lemma 4.7). Any two unrelated outer narrowing derivations must generate uncompared E-matchers (Theorem 4.4). Therefore, if Σ is the set of E-matchers from P to C, generated by outer narrowing derivations, then Σ is a complete set and

$$\forall \theta_1, \theta_2 \in \Sigma,\ \theta_1 \neq \theta_2 \Rightarrow \theta_1 \not\geq_E \theta_2 \ [V(P)].$$

We conclude the proof by observing that W can be easily protected by introducing disjoint sets of variables in each narrowing step.

Note that for this E-matching problem, the E-matchers obtained by outer narrowing are all constructor-based.

The following statement directly follows from the above result: given two terms P and C, where one of them is a *ground* constructor term, outer narrowing generates a complete and minimal set of E-unifiers for P and C.

5. A Pattern-Driven Procedure Enumerating Outer Narrowing Sequences

The procedure is presented in the form of solving systems of equations, following Martelli and Montanari (1982) (see also Herbrand 1971). However, the assumption of constructor-based term rewriting systems can simplify a number of processes.

An *equation* $t = s$ is in *solved form for matching* if t is a variable, and is in *solved form for unification* if either t or s is a variable. An *ordered* system of equations is in solved form for matching if every equation wherein is in solved form for matching and has no variable in the left hand sides appearing in other equations. An ordered system of equations is in solved form for unification if every equation wherein is in solved form for unification and has no variable appearing in other equations.

Given a term P and a constructor term C, we would like to find all outer narrowing derivations $P \overset{*}{\leadsto} C'$, where C' is syntactically matchable to C. It should be kept in mind that we are dealing with constructor-based systems, in which the left hand sides of rules are linear and not possibly narrowable. We first give three auxiliary procedures in Figure 2. The procedure **transform** syntactically transforms an ordered system of equations, preserving the ordering according to $<>_l$. From a solved system of equations, a match is obtained by **get-match** and a unifier by **get-unifier**.

```
transform($EQ$):
    where $EQ = \{t_1 = s_1, ..., t_n = s_n\}$;
    Repeat
        If the $i$-th equation in $EQ$ is of the form $g(l_1, ..., l_m) = g(l'_1, ..., l'_m)$, then
        $EQ := \{t_1 = s_1, ..., t_{i-1} = s_{i-1}, ..., l_1 = l'_1, ..., l_m = l'_m, ..., t_{i+1} = s_{i+1}, ..., t_n = s_n\}$;
        If $\exists x_i = s_i \in EQ$ where $x_j$ is a variable,
            then substitute $s_i$ into any other occurrences of $x_i$
    End repeat;
    return $EQ$.

get-match($EQ$):
    where $EQ$ is in solved form for matching;
    $\sigma := \{\}$;
    For each $x_i = t_i$ in $EQ$, compose $\{x_i/t_i\}$ into $\sigma$;
    return $\sigma$.

get-unifier($EQ$):
    where $EQ$ is in solved form for unification;
    $\sigma := \{\}$;
    For each $x_i = t_i$ or $t_i = x_i$ in $EQ$, where $x_i$ is a variable
        compose $\{x_i/t_i\}$ into $\sigma$;
    return $\sigma$.
```

Figure 2: Auxiliary procedures.

Since we are interested in the matching problem from a term P to a constructor term C of disjoint variables, it is safe to technically treat the variables in C as "constant" symbols. This allows us to use the procedure **transform** for both (syntactic) unification and matching purposes.

Note that in the procedure **transform** we do not need to consider substituting the variables in right hand sides of equations. This is due to the way that the procedure is used: the right hand side of an equation is extracted either from the left hand side of some rule which is linear, or from the given constructor term.

Before describing the procedure **solve** in Figure 3, we give a definition.

5.1 DEFINITION. Given a term rewriting system $R = \{\alpha_j \rightarrow \beta_j\}$, we denote by f_i the function symbol f defined by the i-th rewrite rule. Thus, if f is defined by more than one rule it will have different subscripts. We define a binary relation X_R as follows:

(a) Let $f(t_1, ..., t_n) \rightarrow g(s_1, ..., s_m)$ be the i-th rule in R. Then $<f_i, g> \in X_R$ if $g \in F_C$; and $<f_i, g_j> \in X_R$ if the j-th rule defines g.

(b) Let $f(t_1, ..., t_n) \rightarrow x$ be the i-th rule in R where x is a variable. Then for any $c \in F_C$, $<f_i, c> \in X_R$; and for any $g \in F_D$, $<f_i, g_j> \in X_R$ if the j-th rule defines g.

(c) X_R is the smallest relation satisfying (a) and (b).

We will denote by X^t_R the transitive closure of X_R.

Note that X^t_R is finite for finite R and F. It is clear that for any defined function symbol f and constructor c, an equation of the form $f(...) = c(...)$ has a solution, only if there exists $<f_i, c> \in X^t_R$, for some i. For example, with the following rewrite system

$$R = \{length([\ ]) \rightarrow 0,\ length(x.y) \rightarrow s(length(y))\}$$

we have $X^t_R = \{<length_1, 0>, <length_2, s>\}$.

The procedure **solve**(EQ, ρ, β) in Figure 3 tries to syntactically solve a system of equations; if the attempt is unsuccessful it then invokes the procedure **outer-narrow**(t, c) and finds the rules that can possibly narrow the term t to the one with c as the leftmost symbol. The parameters in **solve** are used for the following purposes: EQ holds the system of equations to be solved; ρ is the composition of the unifiers recursively obtained from syntactically solved (sub)systems of equations; and β the right hand side of the rule used by the procedure **outer-narrow**. Initially, the procedure is invoked by **solve**($\{P = C\}$, $\{\ \}$, $\perp$), where $\{P = C\}$ is the original system of equation for the E-matching problem from a term P to a constructor term C; $\{\ \}$ is the empty substitution; and $\perp$ indicates that no rewrite rule has been used.

Notice the pairs of elements returned by **solve** as intermediate results: the first component holds the right hand side (with the unifier applied) of the rule employed by **outer-narrow**, and the second the accumulated unifier generated from solving the corresponding (sub)system of equations.

When a system of equations is not syntactically solvable, the procedure **outer-narrow**(t, c) is called, which tries to change the term t to the one having c as the leftmost symbol. For each newly formed system of equations, if it is syntactically solvable the execution returns to **solve** to see if the modified system of equations is syntactically solvable; otherwise, **outer-narrow** will be called recursively. These two procedures recursively call each other until the generated systems of equations are either solved or cannot be solved for the reason of constructor conflict. The procedure **solve** is not guaranteed to terminate in general since we are dealing with a semi-decidable problem. Note that nondeterminism can only be introduced by the procedure **outer-narrow**. In addition, the relation X^t_R further cuts down the degree of nondeterminism.

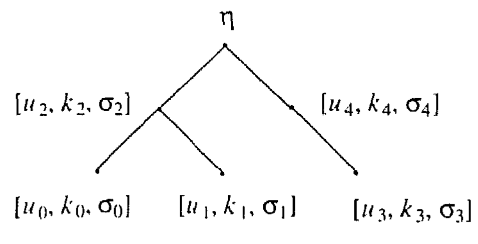
For each successfully computed E-matcher, we can construct a narrowing derivation as follows. A syntactically solved system of equations returned from each call of **outer-narrow** corresponds to a single narrowing step. Since we are using left-linear rules, it is easy to see that cyclic equations (i.e., a variable is equated to a term containing itself) cannot be generated and as a consequence, failure of unification can only result from function symbol conflict(s). Also, the unifier obtained from a syntactically solved system of equations is obviously most general for the term and the left hand side of the rule.

```
solve($EQ$, $\rho$, $\beta$):
    where $EQ$ is an ordered system of equations,
        $\rho$ the accumulated substitution and $\beta$ either a term or $\bot$;
    $EQ'$ := transform($EQ$);
    If $\beta = \bot$ and $EQ'$ is in solved form for matching then
        {
        $\sigma$ := get-match($EQ'$);
        return $\sigma \bullet \rho$                  /*an $E$-matcher generated*/
        };
    If $\beta \neq \bot$ and $EQ'$ is in solved form for unification then
        {
        $\sigma$ := get-unifier($EQ'$);
        return $\{\sigma(\beta), \sigma \bullet \rho\}$        /*an intermediate result obtained*/
        };
    Otherwise choose the leftmost equation $t_i = s_i$ in $EQ'$
                such that $\exists <t_i(\varepsilon)_j, s_i(\varepsilon)> \in X^t_R$ for some $j$
        {                  /*consider an equation which might be solvable*/
        $\Phi$ := outer-narrow($t_i$, $s_i(\varepsilon)$), where elements in $\Phi$ are of the form $\{\beta', \theta\}$;
        For each $\{\beta', \theta\} \in \Phi$, do
            {
            $EQ''$ := $\theta(EQ')$ with $\theta(t_i)$ replaced by $\beta'$;
            return solve($EQ''$, $\theta \bullet \rho$, $\beta$)
            }
        }
    If none of the above applies then stop and return FAILURE.

outer-narrow($t$, $c$):
    For each rule $\alpha_j \to \beta_j \in R$ such that $<t(\varepsilon)_j, c> \in X^t_R$,
    assuming variables are properly renamed,
    form an ordered system of equations: $EQ$ := $\{t = \alpha_j\}$;
    return solve($EQ$, $\{\}$, $\beta_j$).
```

Figure 3: The procedures **solve** and **outer-narrow**.

To be more precise, let us consider the calls of **o··ter-narrow** involved in the successful computation of an E-matcher. The order in which these calls were made can be represented by a tree (which may not necessarily be binary), and the sequence can be obtained by the depth-first search of the tree. In addition, a system of equations may be seen as an implicit representation of a term. Each subterm t in **outer-narrow**(t, c) can therefore be given a redex with respect to the top level system of equations. We will omit the complex but routine construction for this correspondence and leave it to the reader's intuition to see that this can be done. As an example, consider a successfully computed E-matcher by the following calls of **outer-narrow**:



We then have the following narrowing derivation:

$$P_0 \sim>_{[u_0,k_0,\sigma_0]} P_1 \sim>_{[u_1,k_1,\sigma_1]} P_2 \sim>_{[u_2,k_2,\sigma_2]} P_3 \sim>_{[u_3,k_3,\sigma_3]} P_4 \sim>_{[u_4,k_4,\sigma_4]} P_5$$

where σ_i is returned by **get-unifier** upon the return of the i-th call of **outer-narrow**, P_i is restored from the top level system of equations at that time. Since this is a successful derivation, P_5 must be matchable to C with the most general match η (returned by the first call of **solve**). The generated E-matcher is therefore

$$\eta \bullet \sigma_4 \bullet \sigma_3 \bullet \sigma_2 \bullet \sigma_1 \bullet \sigma_0 \,_{|V(P_0)}.$$

Note that each recursive call of **outer-narrow** is due to a disagreed function symbol. It is then obvious that A_i is not narrowable at u_j by the k_j-th rule if u_j is a residue of some w outer to u_i. Notice also that the order in a system of equations corresponds exactly to the one determined by $<>_l$ in an outer narrowing derivation. Therefore, **solve** enumerates only outer narrowing derivations.

Now, let us use **solve** as a reduction procedure. More precisely, let σ be a normalized substitution such that $\sigma(A_0) \overset{*}{\rightarrow}_U C$ is outer. It is not difficult to construct a calling sequence of **outer-narrow** and a calling sequence of **solve** connecting all reduction steps that reduce $\sigma(A_0)$ to C. From the exact same calling sequences a narrowing derivation starting from A_0 can be constructed. We therefore conclude that the procedure **solve** enumerates all and only outer narrowing derivations for the special matching problem considered in this paper.

The mechanism of solving systems of equations has also been employed in developing more general unification procedures without the minimality result. Gallier and Snyder (1987) developed a transformation system that is complete for all equational theories. The transformation system described in Martelli *et al.* (1986) aims at dealing with canonical term rewriting systems and employs a transformation rule (among others), called "outermost term rewriting" which has similar effect as outer narrowing. The procedure described in this section can be viewed as a simplified version of theirs because of the restrictions imposed on the rewrite systems. However, rewrite systems considered here are not required to be terminating in general.

6. Summary and Final Remarks

We have studied a special narrowing strategy, the outer narrowing strategy, which enumerates all outer narrowing derivations. Two results have been obtained. The first result shows that for the class of constructor-based term rewriting systems, every narrowing derivation is subsumed by an outer narrowing derivation. This result can be seen as an extension of Reddy's lazy narrowing strategy (Reddy 1985) in the purely equational framework. The second result is about complete and minimal sets of E-matchers for a restricted case of the general E-matching problem. It follows from this result that for this special matching problem, complete and minimal sets of E-matchers exist for equational theories that can be described by a constructor-based term rewriting system.

To our knowledge, all the known complete and minimal algorithms are based on the fact that they terminate. The minimal sets can then be obtained from the complete sets by a filtering process. These theories thus belong to *finitary* theories (Burckert *et al.* 1987; Siekmann 1984). It can be shown by an example that for the matching problem considered in this paper the class of constructor-based equational theories belong to the class of *infinitary matching* theories (Burckert *et al.* 1987; Siekmann 1984). Consider the following term rewriting system:

$$R = \{\, f(c(x)) \rightarrow f(x),\; f(d) \rightarrow e \,\}.$$

The system is constructor-based and even terminating. However, there are an infinite number of most general E-matchers from the term $f(x)$ to e, which are $\{x/d\}$, $\{x/c(d)\}$, $\{x/c(c(d))\}$,

... $\{x/c(...c(d)...)\}$.... The result presented in this paper appears to be the first one dealing with the minimality problem for classes of infinitary theories.

The approach used in this paper may be further extended with certain limitation. We claim that for terminating constructor-based systems, the general matching problem can be solved by the outer narrowing method. The difficulty to extend the current approach to the E-unification problem is that two compared E-unifiers may be generated by two narrowing derivations, one being a prefix of the other; a set of generated E-unifier is thus no longer guaranteed to be minimal. It appears that the handling of this situation is tricky and can be costly.

We noticed that a counterexample given in Fages & Huet (1983) was a closed linear term rewriting system; consequently, we restricted our attention to subclasses of closed linear systems. The following problem is still open: do complete and minimal sets of E-matchers or E-unifiers exist in general for equational theories that can be described by a constructor-based term rewriting system? We conjecture that the answer is *yes*. We further conjecture that complete and minimal sets of E-matchers or E-unifiers exist even for equational theories that can be described by a left-linear, nonoverlapping term rewriting system. We wish the work presented in this paper provides insights into solutions to these problems. However, even the existence problem can be positively answered, developing complete and minimal procedures can be difficult.

The rewrite systems considered in this paper are highly restricted; they only allow functions to be defined in terms of constructors. This is inadequate for other applications such as verifying properties of equational programs. This problem may be tackled along the same line as combining theories or building unification algorithms incrementally (see, for example, Huet & Oppen 1980, Jouannaud *et al.* 1983, Yellick 1985). Research in this direction is needed as most available methods work only for finitary theories.

Acknowledgments.
The author would like to thank Chung Sea Law for helpful feedback on an early draft of this paper, and the anonymous referees for the comments and suggestions that helped improve this paper.

References

Burckert, H., Herold, A., Schmidt-Schauβ, M. (1987). On equational theories, unification and decidability. *Proc. RTA '87, LNCS 256*. Springer-Verlag, New York.

Burstall, R.M., Sannella, D.T. (1980). *HOPE user's manual*. Dept. of Computer Science, University of Edinburgh.

Dershowitz, H., Plaisted, D. (1985). Logic programming can applicative programming. *Proc. International Symposium on Logic Programming*, pp. 54-67, Boston, Mass.

Dershowitz, H. (1985). Computing with rewrite rules. *Information and Control*, May/June, **65**, **2/3**.

Dincbas, M., van Hentenryck, P. (1987). Extended unification algorithms for the integration of functional programming into logic programming. *J. Logic Programming*, September, **4**, **3**, pp. 199-219.

Fages, F., Huet, G. (1983). Unification and matching in equational theories. *Proc. CAAP '83, LNCS 159*, pp. 205-220. Springer-Verlag.

Fay, M.J. (1979). First-order unification in an equational theory. *Proc. 4th Workshop on Automated Deduction*, pp. 161-167.

Fribourg, L. (1985). SLOG: A logic programming language interpreter based on clausal superposition and rewriting. *Proc. Symposium on Logic Programming*, pp. 172-184, Boston, Mass., 1985.

Gallier J.H., Snyder, W. (1987). A general complete E-unification procedure. *Proc. RTA '87, LNCS 256*, pp. 216-227. Springer-Verlag.

Goguen, J., Meseguer, J. (1984). Equality, types, modules and generics for logic programming. *J. Logic Programming*, **2**, pp 179-210.

Herbrand, J. (1971). Sur la Théorie de la Démonstration. In *Logical Writings*, W. Goldfarb (ed.), Cambridge.

Huet, G., Lévy, J.J. (1979). Computations in nonambiguous linear term rewriting systems (Part I and II). INRIA, Le Chesnay, France.

Huet, G., Oppen, D.C. (1980). Equations and rewrite rules: a survey. In *Formal Language Theory: Perspectives and Open Problems*, R.V. Book (ed.), pp. 349-405, Academic Press, New York.

Huet, G. (1986). Formal structures for computations and deduction. Course notes, Carnegie-Mellon University, May 1986.

Huet, G. (1987). A uniform approach to type theory. INRIA, Le Chesnay, France.

Hullot, J.M. (1980). Canonical forms and unification. *Proc. 5th Conference on Automated Deduction*, pp. 318-334.

Jouannaud, J.P., Kirchner, C., Kirchner, H. Incremental construction of unification procedures in equational theories. *Proc. 10th ICALP, LNCS 154*. Springer-Verlag.

Lankford, D.S. (1975). Canonical inference. Tech. Report ATP-32, Department of Mathematics and Computer Science, University of Texas at Austin.

Martelli, A., Montanari U. An efficient unification algorithm. *ACM TOPLAS*, **4**, **2**, pp. 258-282.

Martelli, A., Moiso, C., Rossi, G. (1986). An algorithm for unification in equational theories. *Proc. 1986 International Symposium on Logic Programming*, pp. 180-186, Salt Lake City, Utah.

Milner, R. (1984). A proposal for standard ML. *Proc. of 1984 ACM Symposium on Lisp and Functional Programming*, pp. 184-197.

Nutt, W., Réty, R., Smolka, G. (1987). Basic narrowing revisited. Tech. Report SR-87-07, Universitat Kaiserslautern.

O'Donnell, M. (1977). Computing in systems described by equations. *LNCS 58*, Springer-Verlag.

O'Donnell, M. (1985). *Equational Logic as a Programming Language*. The MIT Press, Cambridge, Massachusetts.

Plotkin, G. (1972). Building-in equational theories. In *Machine Intelligence* 7, pp. 73-90, Edinburgh University Press.

Reddy, U. (1985). Narrowing as the operational semantics of functional languages. *Proc. International Symposium on Logic Programming*, pp. 138-151, Boston, Mass.

Réty P., Kirchner, C., Kirchner, H., Lescanne, P. (1985). NARROWER: a new algorithm and its application to logic programming. *Proc. Rewriting Techniques and Applications, LNCS 202*, pp. 141-157.

Réty, P. (1987). Improving basic narrowing techniques. *Proc. RTA'87, LNCS 256*, pp. 228-241.

Siekmann, J. (1984). Universal unification. *Proc. 7th International Conference on Automated Deduction*, pp. 1-42, Napa, California.

Slagle, J.R. (1974). Automated theorem proving for theories with simplifier, commutativity, and associativity. *J. ACM*, Oct., **21**, **4**, pp. 622-642.

Subrahmanyam, P.A., You, J. (1986). FUNLOG: a computational model integrating logic and functional programming. In *Logic Programming: Functions, Relations, and Equations*, D. DeGroot and G. Lindstrom (eds.), Prentice-Hall, 1986.

Turner, D.A. (1979). SASL language manual. University of St. Andrews, 1979.

Yellick, K. (1985). Combining unification algorithms for confined equational theories. *Proc. RTA'85 LNCS 202*.

You, J., Subrahmanyam, P.A. (1986a). Equational logic programming: an extension to equational programming. *Proc. 13th POPL*, pp. 209-218, St. Petersburg, Florida.

You, J., Subrahmanyam, P.A. (1986b). A class of confluent term rewriting systems and unification. *J. Automated Reasoning*, December, **2**, 391-418.