Unification Modulo an Equality Theory for Equational Logic Programming*

JIA-HUAI YOU

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

Received February 11, 1989; revised July 10, 1989

Equational logic programming is an extended programming paradigm of equational programming. Central to the notion of equational logic programming is the problem of solving equations, which is also called unification in equational theories. In this paper, we investigate the problem of solving equations in O'Donnell's equational language. We define an equality theory for this language which adequately captures the intended notion of equational programming in the original language. We present a novel technique of transforming narrowing derivations and show the effect of such a transformation on the generality of solutions. As the main result of this paper we show semantically and operationally that complete and minimal sets of solutions under this equality theory always exist and can be generated by a special class of narrowing derivations.

1. INTRODUCTION

In the last few years, the notion of solving equations has been widely used in the context of logic programming with an equational flavor [4, 5, 7, 10, 19, 22, 26, 28, 31] (also see [3] for a collection of articles addressing the subject). The problem of solving equations can be characterized in terms of unification modulo an equational theory, which is often called E-unification [29]. Many of the proposed systems that combine functional, equational, and logic programming adopt the classic equality theory, i.e., the one described by the axioms of reflexivity, transitivity, symmetry, and substitutivity.

Equality, which is often referred to as the classic equality, is difficult to handle. Although equality is so intuitive and notationally simple, there have been doubts about whether an efficient system, comparable to Prolog, can be actually built. In theorem proving, the handling of equality has long been suggested to use the E-unification method [25]. The same approach has been carried over to logic programming with equality. As pointed out by Gallier and Raatz [11], most proposed systems embody a mechanism of E-unification, either implicitly or

* Work supported by the Natural Sciences and Engineering Research Council of Canada under the Grant OGP9225.
explicitly. The difficulty in handling equality is further evidenced by nonexistence of a complete procedure based on narrowing for O'Donnell's equational language [32].

It has been observed that the functions definable in many functional languages are expressible in terms of the classic equality theory [2]. However, the fact that functions are a special case of equality is equally important, because the classic equality theory has made it difficult to define a faithful and effective operational semantics, and sometimes it is semantically desirable to allow equality to be held only among some meaningful terms.

To overcome the difficulty of equational reasoning in the classic equality theory, a number of researchers have considered variations of equality. One example is to divide function symbols into two disjoint subsets: defined functions and constructors. Defined function symbols are those that appear as the outermost function symbol in the left-hand side of an equational definition, and constructors are the remaining function symbols in the language. Fribourg was the first to use this notation in the context of equational logic programming [10]. He defined a relation which is a subset of the equality relation and considered the case where solutions to an equational query must be constructor-based; i.e., no substitutes should contain defined function symbols. A special strategy of narrowing, called innermost narrowing was proven complete under certain conditions. Yukawa [34] provided more arguments for a similar approach and showed a fixpoint characterization and a resolution proof procedure. On the other hand, some authors considered variants of equality without entirely relying on the division of constructors and defined functions. For example, van Emden and Yukawa [7] proposed a variant of equality based on the notion of canonicality of terms, which deals with confluent and noetherian term rewriting systems by a Prolog-like operational semantics. In our earlier work [32] we showed a completeness result for equational theories expressible by closed linear term rewriting systems [23]: narrowing is complete if we restrict equality to such a case that two terms are equivalent if they are both equivalent to a completely narrowed term (i.e., a non-narrowable term). Some other works rely more or less on a form of denotational semantics. Following the denotational approach to functional programming, Reddy [28] extended a functional language with constructors to a functional logic language and outlined a lazy narrowing strategy. To incorporate demand-driven computation of nonstrict functions and infinite data structure into their language, Levi et al. [19] used a nonstrict equality whose meaning is defined by a partial order semantics.

In this paper, we consider equational logic programming in O'Donnell's equational language, which is characterized by the left-linearity and nonoverlapping properties of equational programs. Equational programs satisfying these two properties are also called regular term rewriting systems [15, 16, 24]. The resulting language can be considered as a stand-alone language or a sublanguage which may be embedded into a general logic language by means of incorporating first-order theories into Horn clause theories, such as CLP [18], and into general clause theories, such as theory resolution [30]. Our goal is to build a logic system that
supports various first-order theories for possibly different applications. Because of
this, we have to confine our approach to the conventional logic framework and at
the same time preserve the expressive and computational power of the original
language, such as nonstrict functions, infinite data structures, and lazy evaluation.

These considerations prevent us from imposing the termination condition on an
equational program, as the original language allows nonterminating programs. It is
perhaps not very desirable that whether a program "makes sense" depends on an
undecidable condition—termination. In our opinion, the termination condition
should best be treated as a program property, which may be used to reduce search
space at run time, other than a basic requirement for a program to possess
meaning. Meaning, i.e., declarative semantics of a program, should be purely deter-
mined by the form (i.e., the syntactic structure) and the associated interpretations
of symbols in the form. (The confluence property can also be treated in this way
but a discussion of this will be out of the scope of this paper.) In this regard, Horn
clause logic programming serves as a good example: although terminating
programs are desirable, whether a program makes sense is entirely determined by
its form.

This semantic consideration, however, makes it difficult to design complete yet
effective proof procedures. The handling of the nontermination property is one of
the challenging tasks in equational reasoning. The elegant results by Fay and
Hullot [9, 17] on the use of narrowing to solve the E-unification problem were
obtained precisely because of the requirement of the termination property, in addi-
tion to the confluence property. It is well known that regular term rewriting systems
are confluent. However, narrowing becomes incomplete when nonterminating term
rewriting systems are allowed [32]. One of the aims of this paper is to show that
with a slightly more restrictive equality definition, nontermination can be handled
properly for the purpose of equational logic programming in an equational
language.

The desire to build a logic system that supports various first-order theories dis-
allows us to use a denotational approach, which can make it quite difficult, if not
entirely impossible, to embed such a language into Horn clause logic programming.
As indicated by Goguen and Meseguer [13], combining two logic languages can
best be approached by combining two underlying logics. This is especially true if we
want to build a logic system by the method of building-in theories without a semi-
tic reformalization of the system for each additional feature.

Under these considerations we define an equality theory called $E_n$-equality, which
is a restricted version of the well-known E-equality. The definition of $E_n$-equality
relies on a stronger notion of normal forms which we call S-normal forms. A term
$t$ is said to be in S-normal form if $t$ is completely narrowed, i.e., $t$ is non-
narrowable. In other words, a term $t$ is an S-normal form if it is either a constructor
term [34] (called data term in [19]), which is composed of constructors and
variables only, or a non-narrowable function term, which contains at least one
defined function symbol. Now two terms are $E_n$-equivalent if they are both
$E$-equivalent to an S-normal form. The computational effect of this definition is that
the classic equality applies everywhere except to those terms that have no way to terminate at an S-normal form. Thus, in our equality theory, $f(a)$ is $E_n$-equivalent to itself even if $f$ is “undefined” at $a$ (i.e., $f(a)$ is not reducible). However, if there is no way $f(a)$ can terminate at an S-normal form, $f(a)$ is not $E_n$-equivalent to anything, even itself. We will see that this definition of equality, while retaining the expressiveness of the original language, results in a more effective operational semantics than the classic equality theory. Under this semantics, a program will have the same meaning both in equational programming, as originally advocated in [15, 24], and in equational logic programming. The difference only stems from different, but compatible operational semantics based on reduction for the former and narrowing for the latter.

A non-narrowable function term is sometimes said to be “undefined.” The inclusion of these terms as S-normal forms is because in O’Donnell’s language, defined function symbols can also serve as constructors; they are allowed to appear in the inner part of the left-hand side of an equational definition. As indicated by O’Donnell [24], although in principle the appearances of defined function symbols in the inner part of the left-hand side of a rule can be transformed to constructors, the process may lead to longer and less clear equational programs. Nevertheless, $E_n$-equality defined here is more general than the constructor-based equality in [34]: the relation generated by the former is always a superset of that by the latter. Thus the results obtained here automatically apply to the systems based on the constructor-based equality.

The main purpose of this paper is to study the computational implications of such an equality definition. We show that the general problem of unification modulo this equality theory is still difficult. For example, similar to the case of $E$-unification, there exist theories for which complete and minimal sets of solutions may not always exist. We show, however, that complete and minimal sets of solutions do exist for theories expressible in O’Donnell’s language [24]. In addition, we show that complete and minimal sets of solutions can be enumerated effectively by a special class of narrowing derivations with the outer-before-inner property, which can be used to characterize various strategies of lazy narrowing (see, for example, [6, 19, 28, 33]). As Huet and Lévy [15] (also see [16]) have given a standardization theorem for reduction, we will give a standardization theorem for narrowing, which can be used to establish completeness of a narrowing strategy under possibly different semantics. The theorem is obtained by using a novel technique of transforming narrowing derivations.

It should be addressed that the minimality problem is known to be difficult in $E$-unification. A complete set of solutions is said to be minimal if no distinct solutions in the set are comparable; i.e., there is no redundancy in the set. The difficulty is twofold. First, complete and minimal sets of solutions may not always exist [8]. Second, even the existence question can in theory be answered positively for some special classes of systems, designing a proof procedure that can actually generate those sets of solutions can be extremely difficult, particularly for infinitary theories (the theories that may have an infinite number of most general solutions for some
input terms; see [1, 29] for details). One major contribution of this paper is to show that with a suitably defined equality theory, the minimality problem can be solved for a large class of equational programs, both semantically and operationally.

The paper is organized as follows. The next section recalls the basic notations used in term rewriting. The restricted equality theory will be formally defined in Section 3. It is shown in Section 4 that narrowing is complete, and complete and minimal sets of solutions always exist for equational programs expressible in O'Donnell's language. Section 5 describes a transformation process on narrowing derivations; based upon which a special class of narrowing derivations is defined and shown to generate complete and minimal sets of solutions. Finally, some remarks conclude the paper.

2. Preliminaries

We assume the well-known concept of terms and algebra [14]. \( T(F, V) \) denotes the set of terms generated from a set of function symbols \( F \) and an enumerable, disjoint set of variables \( V \). Terms are denoted by capital letters \( A, B, C \), etc., as well as by the lower case letters \( t, s \). Variables are denoted by \( x, y, z \), etc. 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 \( I^\ast \), to \( F \cup V \) such that its domain satisfies:

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

(ii) \( u \in D(t_i) \) iff \( i \cdot u \in D(f(t_1, \ldots, t_i, \ldots, t_n)) \), \( 1 \leq i \leq n \).

\( D(A) \) is called the set of occurrences of \( A \); \( O(A) \) denotes the nonvariable subset of \( D(A) \). The set of occurrences is partially ordered: \( u \leq v \) iff \( (\exists w) u \cdot w = v \), and \( u < v \) iff \( u \leq v \) and \( u \neq v \). When \( u < v \) we then say that \( u \) is outer to \( v \) and \( v \) is inner to \( u \). \( u \) and \( w \) are said to be independent, denoted by \( u \triangleleft w \) iff \( u \not< w \) and \( w \not> u \). The quotient \( u/v \) of two occurrences \( u \) and \( v \) is defined as: \( u/v = w \) iff \( u = v \cdot w \).

\( V(\Phi) \) denotes the set of variables occurring in an object \( \Phi \). \( A[u \leftarrow B] \) is the term \( A \) in which the subterm at occurrence \( u \) has been replaced by the term \( B \). The subterm of \( A \) at occurrence \( u \) is denoted by \( A/u \).

A substitution \( \sigma = \{ x_1/t_1, \ldots, x_n/t_n \} \), is a mapping from \( V \) to \( T(F, V) \), extended to an endomorphism of \( T(F, V) \). It applies to a term \( t \) by simultaneously replacing all occurrences of each \( x_i \) in \( t \) by \( t_i \). If \( \sigma \) is a substitution and \( A \) is a term, we write \( \sigma A \) for the application of \( \sigma \) to \( A \). The domain of a substitution \( \sigma \), denoted by \( D(\sigma) \), contains the variables that are not mapped to themselves. The set of all substitutions is denoted by \( \Omega \). The composition of substitutions \( \sigma \) and \( \theta \) is defined as a mapping: \( (\sigma \cdot \theta) x = \sigma(\theta x) \). We write \( \sigma|_W \) to restrict \( \sigma \) to the set of variables \( W \).

A term rewriting system (or rewrite system) is a finite set of directed equations \( R = \{ \alpha_i \rightarrow \beta_i \mid i \in N \} \) such that variables appearing in \( \beta_i \) must also appear in \( \alpha_i \). Such directed equations are also called rewrite rules. The reduction (or rewriting)
relation \( \rightarrow^R \) associated with \( R \) is the finest relation over \( T(F, V) \), containing \( R \) and closed by substitution and replacement. From now on we will use \( -+_R \) for \( \rightarrow^R \). We denote by \( \rightarrow^R \) the reflexive, transitive closure of \( \rightarrow \). We say that a term \( A \) reduces to \( B \) at occurrence \( u \) using \( \alpha_k \rightarrow \beta_k \in R \), and write \( A \rightarrow B \) if and only if there exists a substitution \( \eta \) and an occurrence \( u \) in \( O(A) \) such that \( A/u = \eta(\alpha_k) \) and \( B = A[u \leftarrow \eta(\beta_k)] \).

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

A term rewriting system \( R \) is said to be canonical iff \( \rightarrow \) is noetherian, i.e., there does not exist any infinite derivation sequence: \( A_0 \rightarrow A_1 \rightarrow \cdots \), and \( \rightarrow \) is confluent, i.e.,

\[
\forall A, B, C \quad (A \rightarrow^R B \land A \rightarrow^R C) \rightarrow \exists D(B \rightarrow^R D \land C \rightarrow^R D).
\]

When a term \( A \) reduces to a term \( B \) at occurrence \( u \) using the rule \( \alpha_k \rightarrow \beta_k \) in \( R \), we also write \( A \rightarrow_{[u, k]} B \) or \( A \rightarrow_{[u, \alpha_k \rightarrow \beta_k]} B \) 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 abbreviate a reduction derivation

\[
A_0 \rightarrow_{[u_0, k_0]} \cdots \rightarrow_{[u_{n-1}, k_{n-1}]} A_n
\]

by \( A_0 \rightarrow^*_{[U, K]} A_n \), where \( [U, K] \) denotes the sequence of \( [u_i, k_i] \).

Notice the difference between the reduction relation and reduction derivation. In particular, for any relation \( A_0 \rightarrow^* A_n \), 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 \( \rightarrow_{[u, k]} \). \( [u, k] \) is omitted only if no confusion arises.

A term \( A \) narrows to \( B \) at a nonvariable subterm \( A/u \) using the \( k \)th rewrite rule \( \alpha_k \rightarrow \beta_k \) in \( R \), denoted by \( A \rightarrow_{[u, \alpha_k \rightarrow \beta_k]} B \), if \( \rho \) is the most general unifier of \( A/u \) and \( \alpha_k \) and \( B = \rho(A[u \leftarrow \beta_k]) \). It is always assumed that the sets of variables in \( A \) and \( \alpha_k \) are disjoint, and the variables in the substitutes in \( \rho \) are renamed properly so they are distinct from those in \( A \) and \( \alpha_k \). We denote by \( \rightarrow^* \) the reflexive, transitive closure of \( \rightarrow \).

Notationally, we may use \( \rightarrow^*_{[U, K, \sigma]} \) to denote a narrowing derivation, where \( U \) denotes redexes, \( K \) the corresponding rule indexes, and \( \sigma \) the composition of all unifiers along the narrowing derivation.

An equational program \( R \) in O'Donnell's language is a regular term rewriting system satisfying two syntactic conditions: (i) every rule must be left-linear—i.e., no variable may appear more than once in the left-hand side; and (ii) there is no critical pair in \( R \)—i.e., for any two rules \( \alpha_i \rightarrow \beta_i \) and \( \alpha_j \rightarrow \beta_j \) in \( R \), \( \alpha_i/u \) and \( \alpha_j \), where \( u \in O(\alpha_i) \), have no common instance except when \( u = \varepsilon \) and \( i = j \). The second property is called nonoverlapping (also called nonambiguous or superposition-free in the literature). These two properties are syntactically checkable and guarantee the confluence property without resorting to the termination property.
3. An Equality Theory for Equational Logic Programming

Traditionally, an equational theory described by a term rewriting system $R$ is the set of equations $E$ obtained by replacing $\rightarrow$ by $=$; and $E$-equality, denoted by $=_{E}$, is defined as the finest congruence containing $E$, and closed under replacement and instantiation. That is, $=_{E}$ is generated from $E$ as the finest congruence containing all pairs $\sigma A = \sigma B$ for $A = B \in E$ and $\sigma \in \Omega$. It is known that the equality $=_{E}$ is the same as the symmetric closure of $\Rightarrow$ [14].

Once we have $E$-equality, we can define the Church-Rosser property: for any terms $A$ and $B$, $A =_{E} B$ if and only if there exists $C$ such that $A \Rightarrow C$ and $B \Rightarrow C$. It is well known that a term rewriting system is Church–Rosser iff it is confluent.

We restrict our attention to a subset of $=_{E}$, called $E_{n}$-equality, denoted by $=_{E_{n}}$.

**Definition 3.1.** A term $A$ is said to be an S-normal form if it is non-narrowable. Two terms $A$ and $B$ are said to be $E_{n}$-equivalent, denoted $A =_{E_{n}} B$ iff there exists an S-normal form $C$ such that $A =_{E} C$ and $B =_{E} C$.

It might be a slight abuse of terminology by calling the theory an “equality” theory, for it may not be an equivalence relation in the usual sense. The choice of this terminology, however, should not affect our discussion in this paper since we are primarily interested in a first-order semantics definition for equational logic programming.

From now on we will use the term $E_{n}$-equational theory instead of equational theory to reflect the fact that the underlying equality is $E_{n}$-equality.

Note that $E_{n}$-equality is defined completely in terms of the classic equality without resorting to any denotational means. This equality does not rely on the termination condition; in this semantic setting a term “makes sense” only if it has an S-normal form, even when there are nonterminating rewriting sequences starting from it. In other words, each non-singleton “congruence” class in $=_{E_{n}}$ must contain at least one S-normal form.

**Proposition 3.2.** Given a term rewriting system $R$,

(i) the set of S-normal forms is a subset of the set of normal forms;

(ii) if $R$ is confluent, then $\forall A, B \in T(F, V), \ A =_{E_{n}} B$ iff there exists an S-normal form $C$ such that $A \Rightarrow C$ and $B \Rightarrow C$; and

(iii) if $A =_{E_{n}} B$, then $A =_{E} B$.

**Proof.** Trivial. $lacksquare$

The essential discrepancy between $E$-equality and $E_{n}$-equality lies between the notion of S-normal form and the standard notion of normal form. It should be clear that this discrepancy has no effect on the features of equational programming in O’Donnell’s original language, such as nonstrict functions, infinite data structures, and lazy evaluation.
As stated in (ii) of Proposition 3.2, the proof method based on the reduction mechanism in conventional reasoning can still be utilized for proving $E_n$-equality with an additional predicate, true of $S$-normal forms.

Solving equations under this equality theory can be described in terms of $E_n$-unification. Two terms $A$ and $B$ are said to be $E_n$-unifiable iff there exists a substitution $\sigma$, such that $\sigma A = E_n \sigma B$.

Note that by definition substitutes in an $E_n$-unifier need not be constructor terms. For example, if we have a rewrite rule $f(x) \rightarrow a$ with $a$ being a constructor, then for any $\sigma$, $\sigma f(x) = E_n \sigma a$; i.e., any substitution can be an $E_n$-unifier of the terms $f(x)$ and $a$. As another example, suppose we have a rule $f(g(b)) \rightarrow g(c)$ with $g(c)$ being non-narrowable, where $g$ may or may not be a defined function symbol. Then, by definition, $\sigma = \{ y/g(b) \}$ is an $E_n$-unifier of $f(y)$ and $g(c)$.

To describe how $E_n$-unifiers compare with each other, we need to extend $E$-equality (not $E_n$-equality) to substitutions: $\sigma = E \theta [W]$ iff $\forall x \in W$, $\sigma x = E \theta x$, where $[W]$ is variable restriction. Comparison of substitutions is defined as: $\sigma \leq E \theta [W]$ iff $\exists \eta \cdot \sigma = E \theta [W]$. We then say that $\sigma$ is more general than $\theta$. We will sometimes omit $[W]$ if no confusion arises.

Note that $E$-equality is used only in the metalinguage for the purpose of comparing substitutions. Our system does not attempt to prove $E$-equality.

A set of substitutions $\Sigma$ is said to be a complete set of $E_n$-unifiers of two terms $A$ and $B$ iff every substitution in $\Sigma$ is an $E_n$-unifier of $A$ and $B$, and for any $E_n$-unifier $\theta$ of $A$ and $B$ (which may or may not be in $\Sigma$), there exists a substitution in $\Sigma$, which is more general than $\theta$. In addition, $\Sigma$ is said to be minimal iff no two unifiers in $\Sigma$ can compare by $\leq E$; i.e., $\forall \theta_1, \theta_2 \in \Sigma$, $\theta_1 \neq \theta_2 \Rightarrow \exists \zeta, \zeta \cdot \theta_1 \leq E \theta_2$.

The minimality property is important since when such an equational reasoning subsystem is embedded into another logic system, the use of non-most general solutions can cause a lot of redundant computations. This is analogous to the importance of the use of most general unifiers in resolution.

Similar to the case of $E$-unification [8], complete and minimal sets of $E_n$-unifiers may not always exist in our restricted equality.

Consider the following term rewriting system modified from [8]:

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

It can be shown that this system is a closed linear term rewriting system defined by O'Donnell [23]. However, there exist an infinite number of $E_n$-unifiers for $f(z)$ and $f(a)$:

$$\sigma_0 = \{ z/a \}$$

$$\sigma_1 = \{ z/g(x_1, a) \}$$

$$\sigma_2 = \{ z/g(x_1, g(x_2, a)) \}$$

$$\ldots$$

$$\sigma_i = \{ z/g(x_1, g(x_2, \ldots g(x_i, a) \ldots)) \}$$

$$\ldots$$
with each one being more general than the preceding one. It can be shown by induction that \( \sigma_i f(z) = E_n \sigma_i f(a) = E_n b \), for all \( i \), where \( b \) is a constructor term and therefore an S-normal form. Thus all of them are \( E_n \)-unifiers of \( f(z) \) and \( f(a) \).

Fortunately, complete and minimal sets of \( E_n \)-unifiers always exist for \( E_n \)-equational theories that can be described by a left-linear and nonoverlapping term rewriting system. We will prove this claim in the next section.

4. Existence of Complete and Minimal Sets of \( E_n \)-Unifiers

In this section we first show that given a left-linear, nonoverlapping term rewriting system, for any \( E_n \)-unifier of two terms, narrowing generates a more general \( E_n \)-unifier of the two given terms. Thus narrowing is complete for \( E_n \)-unification. This leads to the result that complete and minimal sets of \( E_n \)-unifiers always exist.

We first describe a completeness result of narrowing for \( E \)-unification, which was proven in [32] for the class of closed linear term rewriting systems which constitute a slightly larger class of systems than the class of regular term rewriting systems.

In the remainder of this paper, we will assume that the underlying term rewriting system \( R \) is always left-linear and nonoverlapping if not otherwise said. It is well known that term rewriting systems possessing these two properties are confluent [15]. Following [17], the binary constructor symbol \( H \) used in the rest of this paper is assumed not in \( F \) and is for the purpose of notationally combining two reduction/narrowing sequences.

**Theorem 4.1.** Let \( A \) be a term and \( C \) a term in S-normal form. For any \( E \)-unifier \( \xi \) of \( A \) and \( C \), there exists a narrowing derivation

\[
H(A, C) \xrightarrow{\xi} H(C', C),
\]

where \( C' \) is an S-normal form and \( C' \) and \( C \) unify by m.g.u. (the most general unifier) \( \xi \), such that \( \xi \cdot \theta \) is an \( E \)-unifier of \( A \) and \( C \), and \( \xi \cdot \theta \leq E \tau[V(A, C)] \).

A rephrasing of this result gives the completeness of narrowing for \( E_n \)-equality.

**Theorem 4.2.** For any \( E_n \)-unifier \( \sigma \) of two terms \( A \) and \( B \), there exists a narrowing derivation

\[
H(A, B) \xrightarrow{\sigma} H(C_1, C_2),
\]

where \( C_1 \) and \( C_2 \) are S-normal forms and unify by m.g.u. \( \xi \), such that \( \xi \cdot \theta \) is an \( E_n \)-unifier of \( A \) and \( B \), and \( \xi \cdot \theta \leq E \tau[V(A, B)] \).

**Proof.** Since \( \sigma \) is an \( E_n \)-unifier of \( A \) and \( B \), we have

\[
\tau A =_E \tau B =_E D
\]

for some S-normal form \( D \).
Let $D_1 = H(A, B)$ and Let $D_2 = H(D, D)$. Clearly, $D_2$ is an S-normal form. Also, $\tau$ is an $E$-unifier of $D_1$ and $D_2$, since $\tau D_1 = E \tau D_2$. Thus, by Theorem 4.1, a narrowing derivation issuing from $H(D_1, D_2)$ exists, which is essentially the same as the narrowing derivation $(*)$, that generates an $E_n$-unifier $\xi \cdot \theta$ of $A$ and $B$, such that $\xi \cdot \theta \leq_E \tau[V(A, B)]$.

**Corollary 4.3.** *All of the nonvariable substitutes in $\xi \cdot \theta |_{V(A, B)}$ are S-normal forms.*

**Proof.** Let the narrowing derivation $(*)$ in the theorem be

$$H(A, B) = A_0 \sim_{[\mu_0, k_0, \rho_0]} \ldots \sim_{[\mu_{n-1}, k_{n-1}, \rho_{n-1}]} A_n = H(C_1, C_2).$$

Since $R$ is left-linear, any substitute in any $\rho |_{V(A)}$ must be extracted from the left-hand side of the corresponding rule. By the nonoverlapping property, no nonvariable substitute so obtained may be unifiable with the left-hand side of any rule; thus, they must all be S-normal forms. This is also true of all nonvariable substitutes in $\xi$. It is easy to see that this is still true for any substitute in the composition $\xi \cdot \rho_{n-1} \cdot \ldots \cdot \rho_0 |_{V(A_0)}$.

We are now in a position to give the minimality result.

**Theorem 4.4.** *Complete and minimal sets of $E_n$-unifiers always exist for any $E_n$-unifiable terms.*

**Proof.** From Theorem 4.2 and Corollary 4.3, there exists a complete set $\Sigma$ generated by narrowing such that for any $E_n$-unifier $\tau$ in $\Sigma$, all nonvariable substitutes are S-normal forms. Based on this property it is easy to see that for any $\tau$, $\tau' \in \Sigma$, $\tau$ and $\tau'$ compare by $\leq_E$ iff $\tau$ and $\tau'$ compare by $\leq$. For any $\leq$ ordering, a lower bound always exists (in the extreme case it is a variable), which is unique up to variable renaming. Thus, the set of all lower bounds plus those uncompared $E_n$-unifiers form a complete and minimal set of $E_n$-unifiers.

Complete and minimal sets of $E_n$-unifiers are unique up to variable renaming (see [8] for more details). Notice that Theorem 4.4 only claims the existence, not finiteness. Consider the following regular term rewriting system

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

There exist an infinite number of uncompared $E_n$-unifiers for $f(x)$ and $e$, which are $\{x/d\}, \{x/(c(d))\}, \{x/(c(c(d)))\}, \ldots \{x/(c(\ldots(c(d)\ldots))\} \ldots$. It is this phenomenon that makes it impossible to use a filtering method to eliminate redundant solutions.
5. A Special Class of Narrowing Derivations

The narrowing method has been criticized for its high degree of non-determinism which often leads to generation of redundant solutions. The two most popular strategies adopted for reduction, innermost and outermost strategies, do not work well for narrowing. The following example shows that neither innermost narrowing, which ignores narrowing steps at outer occurrences, nor outermost narrowing, which ignores narrowing steps at inner occurrences can be complete even for \( E_n \)-equality defined in this paper. Consider

\[
R = \{ f(y, a) \rightarrow true, f(c, b) \rightarrow true, g(b) \rightarrow c \}.
\]

The system is regular. To unify the term \( f(g(x), x) \) with \( true \), innermost narrowing leads to

(i) \( f(g(x), x) \sim_{\{x/b\}} f(c, b) \sim_{\{\}} true \),

while outermost narrowing yields

(ii) \( f(g(x), x) \sim_{\{x/a\}} true \).

The innermost and outermost strategies generate uncompared results; therefore, neither is complete.

Fribourg [10] showed the completeness of the innermost narrowing strategy under certain sufficient conditions (basically, the functions should be totally defined). Dincbas and Hentenryck [6] investigated the termination and efficiency issues of several strategies, including innermost, outermost and a form of lazy strategy based on a procedural semantics of functional programming. Reddy [28] outlined a lazy narrowing strategy. Outer narrowing, suggested in [33], is operationally similar to lazy narrowing, but is complete for \( E \)-unification. (Lazy narrowing does not yield a complete procedure for \( E \)-unification.) Levi et al. [19] gave a different version of lazy narrowing on flattened equational programs for their partial order semantics.

We show that all of the above lazy strategies, though designed for different underlying semantics, can be described by a process of transformation of narrowing derivations. The observation is that the set of all narrowing derivations can be partitioned into classes in a way that derivations in the same class yield compared solutions. A complete and minimal procedure is then to generate one derivation from each such class that yields the most general solution. Two narrowing derivations belong to the same class if narrowing steps in one derivation can be rearranged to yield the other.

As an example, consider the following term rewriting system:

\[
R = \{ f(a, b, x) \rightarrow d(x), g(a) \rightarrow c \}
\]
and the narrowing derivation

\[ f(y, z, g(y)) \sim_{\{y/a\}} f(a, z, c) \sim_{\{z/b\}} d(c). \]

Notice that the second narrowing step can be performed right at the beginning, using the same rule, and at the same occurrence. "Moving" the second step to the front will give us

\[ f(y, z, g(y)) \sim_{\{y/a, z/b\}} d(g(a)) \sim_{\{\}} d(c). \]

Note that \( f(y, z, g(y)) \) is narrowable both at occurrence \( \varepsilon \) and at occurrence \( 3 \), where \( \varepsilon < 3 \). This seems to suggest that narrowing at an outer occurrence should always be carried out before narrowing at an inner occurrence, and if this is not the case, a transformation can be applied to produce a rearranged derivation that satisfies this property. Unfortunately, this type of transformation may not guarantee completeness when defined functions appear in the inner part of the left-hand side of a rule.

Consider the following nonoverlapping term rewriting system and narrowing derivation,

\[
R = \{ f(g(d)) \rightarrow \text{true}, \quad g(c) \rightarrow g(d) \}
\]

\[ f(g(x)) \sim_{\{x/c\}} f(g(d)) \sim_{\{\}} \text{true}. \]

The second narrowing step can be performed right at the beginning using the same rule and at the same occurrence, which will generate an uncomputed substitution \( \{x/d\} \). Therefore, the transformation should take care of this special case.

To characterize this rearrangement process precisely, we need a method to keep track of "copies" of subterms that are carried over by reduction (and therefore by narrowing). Intuitively, when a reduction is performed, the occurrences of a variable in the right-hand side of the rule denote the rearrangement of a subterm that is matched to an occurrence of the same variable in the left-hand side. The notion of residue map has been used by O'Donnell to study a closure property \[23\] (also see \[15, 27\]).

**Definition 5.1.** The residue map \( r \) with respect to a left-linear term rewriting system \( R \) is defined as: Let \( u, v \in D(A) \),

\[
r[A \rightarrow_{\{u, z_k \rightarrow \rho_k\}} B]v
definition as: Let \( u, v \in D(A) \),

\[
r[A \rightarrow_{\{u, z_k \rightarrow \rho_k\}} B]v
\]

\[
= \{u \cdot w \cdot (v/v') \mid \alpha_k(v'') \in V(\alpha_k) \land \alpha_k(v'') = \beta_k(w) \land v' = u \cdot v''\} \quad \text{if } v > u
\]

\[
= \{v\} \quad \text{if } (u \triangleleft v \text{ or } v < u)
\]

\[
= \emptyset \quad \text{otherwise.}
\]

The residue map \( n \) for narrowing is defined as

\[
n[A \rightarrow_{\{u, z_k \rightarrow \rho_k, \sigma\}} B]v = r[\sigma A \rightarrow_{\{u, z_k \rightarrow \rho_k\}} B]v.
\]
The definition says that if \( v \) is independent of \( u \), or \( v \) is outer to \( u \), the residue of \( v \) is itself and unique; otherwise, a residue of \( v \) is \( u \), at which the reduction occurs, concatenated by \( w \), the address of the variable in \( \beta_k \) that also occurs at \( v'' \) in \( \alpha_k \), and concatenated by \( v/v' \), the relative distance between \( u \cdot v'' \) and \( v \). The definition is illustrated in Fig. 1.

For example, with the rewrite rule \( f(c(x)) \rightarrow g(x, x) \) and reduction \( f(c(a)) \rightarrow g(a, a) \), both occurrences of \( a \) in \( g(a, a) \) are residues of \( a \) in \( f(c(a)) \).

The following proposition is easy to verify.

**Proposition 5.2.** (a) \( \forall w, w' \in r[A \rightarrow_{(u, \alpha_k \rightarrow \beta_k)} B]v, B/w = B/w' \).

(b) \( \forall w, w' \in n[A \rightarrow_{(u, \alpha_k \rightarrow \beta_k, \rho)} B]v, B/w = B/w' \).

The following proposition can be easily shown from the nonoverlapping property (see [15] for a proof). The proposition is illustrated in Fig. 2.

**Proposition 5.3.** Let \( A \) be a term. Then

\[
\forall u, v \in O(A)(v < u \& A \rightarrow_{[v, k]} B \& A \rightarrow_{[u, j]} C) \\
\Rightarrow \exists D \in T(F, V)(B \rightarrow_{[v, k]} D \& C \rightarrow_{[v, k]} D),
\]

where \( U = r[A \rightarrow_{[v, k]} B]u \) and the rule indexes in \( J \) are all \( j \).
We now extend Proposition 5.3 to narrowing, and call this outer-before-inner property. It shows that narrowing steps in a derivation sequence that leads to a solution can be rearranged to yield one that generates a more general solution.

**Lemma 5.4 (Outer-before-inner property).** Suppose

(a) \( A \rightarrow [u, \rho_1] C \rightarrow [v, k, \rho_2] D \), where \( v < u \).

If \( A \) is also narrowable at \( v \) using the \( k \)th rule \( \alpha_k \rightarrow \beta_k \) such that \( \alpha_k(u/v) \neq A(u) \), then there exists a narrowing derivation

(b) \( A \leadsto_{[v, k, \sigma_1]} [u, J, \sigma_2] B \rightarrow_{[v, \rho_1]} C \rightarrow [v, k, \rho_2] D' \)

such that \( \sigma_2 \cdot \sigma_1 \leq \rho_2 \cdot \rho_1 \) and \( \exists \tau, \tau(D') = D \), where \( U = r[A \rightarrow_{[v, k, \sigma_1]} B]u \) and the rule indexes in \( J \) are all \( j \).

**Proof.** From the hypothesis that \( A \) is narrowable at \( v \) using the rule \( \alpha_k \rightarrow \beta_k \) such that \( \alpha_k(u/v) \neq A(u) \), we claim that the following must be true

(c) \( u/v \notin D(\alpha_k) \vee \alpha_k(u/v) \in V(\alpha_k) \).

That is, when \( A \) is unified with \( \alpha_k \), either \( A/u \), or a superterm of it, must be matched to a variable in \( V(\alpha_k) \). This can be proved as follows. Suppose (c) is not true. We show that this will lead to the contradiction that \( A \) is not narrowable at \( v \) using the rule \( \alpha_k \rightarrow \beta_k \).

From the assumption that (c) is not true, we infer that \( \alpha_k(u/v) \) must be a function symbol. Since \( A \) is narrowable at \( u \), \( A(u) \) must be a defined function symbol. However, since \( \alpha_k(u/v) \neq A(u) \), \( A/v \) and \( \alpha_k \) must not unify. This contradicts the hypothesis that \( A \) is narrowable at \( u \) using the rule \( \alpha_k \rightarrow \beta_k \).

Now, let \( A_0 = \rho_2 \cdot \rho_1(A) \). It is easy to see that \( A_0 \) is reducible at \( v \) using the \( k \)th rule, i.e., for some \( B' \),

(i) \( A_0 \rightarrow_{[v, k]} B' \).

From the narrowing derivation (a) in the lemma, we get

(ii) \( A_0 \rightarrow_{[u, \rho_1]} C' \rightarrow_{[v, k]} D \).

From (i) and (ii) above and Proposition 5.3, we get

(iii) \( A_0 \rightarrow_{[v, k]} B' \rightarrow_{[v, \rho_1]} C' \rightarrow_{[v, k]} D \).

where, by the fact (c) above and the definition of the residue map for reduction, \( U = r[A_0 \rightarrow_{[v, k]} B']u \) and the rule indexes in \( J \) are all \( j \).

Since \( \rho_2 \cdot \rho_1 \cdot V(A) \) is normalized (see Corollary 4.3), we can use Hullot’s result [Hull 80]: corresponding to (iii) above there exists a narrowing derivation

\( A \rightarrow_{[v, k, \sigma_1]} B'' \rightarrow_{[u, J, \sigma_2]} D' \)

such that \( \sigma_2 \cdot \sigma_1 \leq \rho_2 \cdot \rho_1 \) and \( \exists \tau, \tau(D') = D \), where, by the fact (c) above and
the definition of the residue map for narrowing (Definition 5.1), $U = n[A \to_{[v, k, \sigma]} B'] u$ and the rule indexes in $J$ are all $j$. This completes the proof.  

**Corollary 5.5.** Let $A \to_{[u, j, \rho]} B$ and $A \to_{[v, k, \sigma]} C$, where $v < u$. If $\alpha_k(u/v) = A(u)$ then there does not exist a substitution $\eta$ such that $\eta \cdot \rho = \eta \cdot \sigma$.

**Proof.** Assume the contrary, i.e., $\exists \eta, \eta \cdot \rho = \eta \cdot \sigma$. We show that this contradicts the nonoverlapping property of the given term rewriting system. From

\[ A \to_{[u, j, \rho]} B \]
\[ A \to_{[v, k, \sigma]} C, \]

we have, respectively,

\[ \rho A \to_{[u, j]} B \]
\[ \sigma A \to_{[v, k]} C. \]

From which, we get

\[ \eta \cdot \rho A \to_{[u, j]} \eta B \]
\[ \eta \cdot \sigma A \to_{[v, k]} \eta C. \]

By the assumption $\eta \cdot \rho = \eta \cdot \sigma$, we have $\eta \cdot \rho A = \eta \cdot \sigma A$. Let $A' = \eta \cdot \rho A = \eta \cdot \sigma A$.

From (*) above, $\exists \tau_1$, $\tau_1 A_j = A'/u$; and from (**) above, $\exists \tau_2$, $\tau_2 \alpha_k/(u/v) = A'/u$. Thus, $\tau_1 A_j = \tau_2 \alpha_k/(u/v)$. With appropriate variable renaming so that $V(\alpha_k) \cap V(\alpha_j) = \emptyset$, we will have $D(\tau_1 \mid V(A')) \cap D(\tau_2 \mid V(A')) = \emptyset$. Let $\tau = \tau_1 \mid V(A') \cup \tau_2 \mid V(A')$. We get $\tau A_j = (\tau \alpha_k)/(u/v)$. By the conditions that $\alpha_k(u/v) = A(u)$ and $A$ is narrowable at $u$, $\alpha_k(u/v)$ must be nonvariable. This contradicts the nonoverlapping property.  

The second process in the transformation is based on the fact that rearranging narrowing steps at independent redexes does not affect completeness. Consequently, the branching factor for a set of independent redexes in any term is one. The reduction of the branching factor to one is guaranteed by a property similar to independent of computation rule in Prolog [20]. Without loss of generality, we fix the rule to be leftmost in this paper.

**Lemma 5.6 (Leftmost selection rule).** Let $\prec_{\text{lex}}$ denote the lexicographic ordering on $I^*$. Suppose $A \to_{[u, k, \rho_1]} C \to_{[v, j, \rho_2]} D$, where $v \diamond u$ and $v \prec_{\text{lex}} u$. Then there exists a narrowing derivation $A \to_{[u, k, \sigma_1]} B \to_{[v, k, \sigma_2]} D$, such that $\sigma_2 \cdot \sigma_1 = \rho_2 \cdot \rho_1$.

**Proof.** Easy and thus omitted.  

Now given a narrowing derivation, apply Lemmas 5.4 and 5.6 repeatedly to rearrange the given derivation. This is like a "sorting" process. It is easy to see that this process always terminates at a unique derivation.
Definition 5.7. A narrowing derivation is said to be a standard narrowing derivation iff no rearrangement can be made by either of the processes described in Lemmas 5.4 and 5.6.

We call these narrowing derivations standard, not only because there is a standardization theorem for narrowing, but also because they generate uncompared solutions for regular systems under the $E_n$-equality theory. We give the standardization theorem first.

Theorem 5.8. Given two terms $A$ and $B$ to be unified, for any narrowing derivation,

$$H(A, B) = A_0 \overset{[u_0, k_0, \rho_0]}{\rightarrow} \cdots \overset{[u_{n-1}, k_{n-1}, \rho_{n-1}]}{\rightarrow} A_n = H(C_1, C_2),$$

such that $C_1$ and $C_2$ unify by m.g.u. $\xi$, there exists a standard narrowing derivation,

$$H(A, B) = B_0 \overset{[v_0, j_0, \sigma_0]}{\rightarrow} \cdots \overset{[v_{m-1}, j_{m-1}, \sigma_{m-1}]}{\rightarrow} B_m = H(D_1, D_2),$$

such that $D_1$ and $D_2$ are unifiable. If $\delta$ is m.g.u. of $D_1$ and $D_2$, then $\delta \cdot \sigma_{m-1} \cdot \cdots \cdot \sigma_0 \leq \xi \cdot \rho_{n-1} \cdot \cdots \cdot \rho_0$.

Proof. Easy induction on the number of steps in the given narrowing derivation, using Lemmas 5.4 and 5.6. Observe that each application of either of these two lemmas results in a more general substitution.

Huet and Lévy [15] (see also [16]) described a standardization theorem for rewriting which says that it is always possible to compute in an outside-in manner. Standard narrowing derivation can be seen as an extension of outside-in reduction to narrowing. When restricted to rewriting, the transformation described in Lemmas 5.4 and 5.6 always produces an outside-in rewriting derivation.

It is not difficult to see from Lemmas 5.4 and 5.6 that whether a narrowing step contributes to a standard derivation can always be determined in advance.

Although various versions of lazy narrowing [6, 19, 28, 33] are different operationally (however, the difference is often slight), all of them can be viewed as computing standard narrowing derivations. The difference among these strategies reflects the difference of the underlying semantics. Completeness results can be obtained for each version once we have the standardization theorem. Thus the standardization theorem obtained here may serve as a general tool to establish completeness of a lazy narrowing strategy under specific semantics. For example, the completeness for $E_n$-unification is easy to obtain.

Corollary 5.9 (Completeness of standard narrowing for $E_n$-equality). The set of standard narrowing derivations yields a complete set of $E_n$-unifiers.

Proof. Narrowing is complete for $E_n$-equality (Theorem 4.4). The completeness of standard narrowing for $E_n$-equality only requires the terms $C_1$, $C_2$, $D_1$, and $D_2$ in Theorem 5.8 to be S-normal forms. Obviously, this is a special case.
For \( E_n \)-unification, standard narrowing also guarantees the minimality property. Before we show this let us give a lemma first.

**Lemma 5.10.** Let \( \sigma \) and \( \rho \) be normalized substitutions. Suppose \((\exists \xi) \xi \cdot \rho = \xi \cdot \sigma\). Then

(a) for any normalized substitutions \( \theta_1 \) and \( \theta_2 \), \((\exists \xi') \xi' \cdot \theta_1 \cdot \rho = \xi' \cdot \theta_2 \cdot \sigma\);

(b) \( \rho \) and \( \sigma \) cannot compare by \( \leq \).

**Proof.** (a) Assume \((\exists \xi') \xi' \cdot \theta_1 \cdot \rho = \xi' \cdot \theta_2 \cdot \sigma\). Then let \( \xi = \xi' \cdot \theta_1 \cup \xi' \cdot \theta_2 \). We hence get \( \xi \cdot \rho = \xi \cdot \sigma \). A contradiction.

(b) Similar and thus omitted.

We are now in the position to show the minimality result.

**Theorem 5.11 (Minimality of standard narrowing for \( E_n \)-equality).** Let \( A \) and \( B \) be two \( E_n \)-unifiable terms. For any two unrelated standard narrowing derivations (no one is a prefix of the other),

\[
H(A, B) = A_0 \leadsto_{[\nu_0, k_0, \rho_0]} \cdots \leadsto_{[\nu_{n-1}, k_{n-1}, \rho_{n-1}]} A_n = H(C_1, C_2)
\]

\[
H(A, B) = B_0 \leadsto_{[\nu_0, j_0, \sigma_0]} \cdots \leadsto_{[\nu_{m-1}, j_{m-1}, \sigma_{m-1}]} B_m = H(D_1, D_2),
\]

where \( C_1 \) and \( C_2 \) unify by m.g.u. \( \tau_1 \), \( D_1 \) and \( D_2 \) by m.g.u. \( \tau_2 \), and \( C_1, C_2, D_1, \) and \( D_2 \) are all S-normal forms, \( \tau_1 \cdot \rho_{n-1} \cdots \rho_0 \mid \nu(A_0) \) and \( \tau_2 \cdot \sigma_{m-1} \cdots \sigma_0 \mid \nu(B_0) \) do not compare by \( \leq E \).

**Proof.** Let \( \eta_1 = \tau_1 \cdot \rho_{n-1} \cdots \rho_0 \mid \nu(A_0) \) and \( \eta_2 = \tau_2 \cdot \sigma_{m-1} \cdots \sigma_0 \mid \nu(B_0) \). From the proof of Theorem 4.4 we know that \( \eta_1 \) and \( \eta_2 \) compare by \( \leq E \) iff \( \eta_1 \) and \( \eta_2 \) compare by \( \leq \). It therefore suffices to show that they do not compare by \( \leq \).

Since (i) and (ii) are unrelated derivations, branching will eventually occur. Let

(i) \( A_i \leadsto_{[\nu_i, k_i, \rho_i]} A_{i+1} \)

(ii) \( B_i \leadsto_{[\nu_i, j_i, \sigma_i]} B_{i+1} \)

denote the two steps corresponding to the very first branching, where \( A_i = B_i \). Note that because this is the first branching we have \( \rho_{i-1} \cdots \rho_0 = \sigma_{i-1} \cdots \sigma_0 \).

Clearly, \( u_i \not\leq v_i \) cannot hold because the selection rule is fixed; hence \( u_i \) and \( v_i \) must be dependent. There are two cases: \( u_i = v_i \) or \( u_i \neq v_i \).

**Case 1.** \( u_i = v_i \). Since \( u_i(A_i) = v_i(B_i) \), the \( k_i \)th rule and the \( j_i \)th rule must define the same function symbol. From the left-linearity and nonoverlapping properties, we have

\[
(\exists \xi) \xi \cdot \rho_i = \xi \cdot \sigma_i.
\]
It follows from \( p_{i-1} \cdots p_0 = \sigma_{i-1} \cdots \sigma_0 \) that
\[
(\exists \xi) \xi \cdot p_1 \cdots \cdot p_0 = \xi \cdot \sigma_1 \cdots \cdot \sigma_0.
\]
By Lemma 5.10, \( \eta_1 \) and \( \eta_2 \) do not compare by \( \preceq \).

**Case 2.** \( u_i \neq v_i \). Without loss of generality, assume \( v_i < u_i \). There are two subcases: (a) \( \alpha_j(u_i/v_i) = A_j(u_i) \) and (b) \( \alpha_j(u_i/v_i) \neq A_j(u_i) \).

**Subcase (a).** \( \alpha_j(u_i/v_i) = A_j(u_i) \). By Corollary 5.5, \( \tau \cdot \rho_1 = \tau \cdot \sigma_1 \), and then by of Lemma 5.10, \( \eta_1 \) and \( \eta_2 \) do not compare \( \preceq \).

**Subcase (b).** \( \alpha_j(u_i/v_i) \neq A_j(u_i) \). Let \( A_i/v_i \) be of the form \( f(t_0, \ldots, t_n) \). Since \( C_1 \), \( C_2 \), \( D_1 \), and \( D_2 \) are all S-normal forms and both narrowing sequences are standard, either the function symbol \( f \) will eventually be replaced in a latter narrowing step at \( v_j \), or the term will become a non-narrowable term \( f(t_0, \ldots, t'_{n}) \). We show the first and omit the second, which is simpler.

For the first case, we have a narrowing derivation,

\[
(iii) \quad A_i \sim_{[u_i, k_i, \rho_i]} A_{i+1} \sim_{[u_i, k_i, \rho_i]} \cdots \sim_{[u_i, k_i, \rho_i]} A_{i+1}, \quad i < n,
\]
such that \( u_i = v_i \), i.e., \( B_i(v_i) = A_i(u_i) = A_i(v_i) \) (the function symbol \( v_i(A_i) \) is eventually replaced by using a rule defining \( A_i(v_i) \)), and \( u_i > u_k \) for all \( k, i \leq k \leq i-1 \). By the outer-before-inner property, we must have \( k_i \neq k_i \), and therefore \( k_i \neq j_i \).

We now assume \( \exists \xi, \xi \cdot \rho_1 \cdots \cdot \rho_j = \xi \cdot \sigma_1 \), and show that this leads to a contradiction. From (iii) above we get
\[
(\rho_1 \cdots \cdot \rho_j(A_i) \ast \rho_j(A_i) \rightarrow_{[u_i, k_i]} A_{i+1}, \quad i < n,
\]
and from (ii) earlier we have
\[
(\sigma_j(B_j) \rightarrow_{[v_i, k_i]} B_{i+1}.
\]
From these two reduction sequences, we obtain the following two reduction sequences, respectively,
\[
(\xi \cdot \rho_1 \cdots \cdot \rho_j(A_i) \rightarrow_{[u_i, k_i]} \xi(A_{i+1})
\]
\[
(\xi \cdot \sigma_j(B_j) \rightarrow_{[v_i, k_i]} \xi(B_{i+1}).
\]
(From now on the reader may refer to Fig. 3 when reading the text.)

Let \( w \in \{u_i, u_{i+1}, \ldots, u_{i-1} \} \) such that for all \( k, i \leq k \leq i-1 \), \( w \leq u_k \) or \( w \nabla u_k \). Clearly, \( w \) must be inner to \( u_i \), i.e., \( w > u_i \). That is, \( w \) represents one of the largest subterms of \( A_i \) that are affected by the reductions. From the assumption that \( \xi \cdot \rho_1 \cdots \cdot \rho_j = \xi \cdot \sigma_j \), we have \( \xi \cdot \rho_1 \cdots \cdot \rho_j(A_i) = \xi \cdot \sigma_j(B_j) \). From the way \( w \) is chosen, for any \( w' \) such that \( w' < w \), we have
\[
(\xi \cdot \rho_j(A_i))(w') = (\xi \cdot \sigma_j(B_j))(w')
\]
but
\[
(\xi \cdot \rho_j(A_i))/w = (\xi \cdot \sigma_j(B_j))/w.
\]
(\( \ast \))
That is, any pair of symbols at the same occurrence in $\zeta \cdot \rho_i A_i$ and $\zeta \cdot \sigma_i B_i$ are identical with the possible exception that the subterms $(\zeta \cdot \rho_i A_i)/w$ and $(\zeta \cdot \sigma_i B_i)/w$ may not be identical; however, they are at least $E$-equivalent to each other. Without loss of generality, assume $(\zeta \cdot \rho_i A_i)/w \neq (\zeta \cdot \sigma_i B_i)/w$.

Since $u_i = v_i$, we let $u = u_i = v_i$. Notice that $(\zeta \cdot \rho_i A_i)/u$ is an instance of $\alpha_{k_i}$ and similarly $(\zeta \cdot \sigma_i B_i)/u$ is an instance of $\alpha_{j_i}$.

Now, since $k_i \neq j_i$ and $\alpha_{k_i}$ and $\alpha_{j_i}$ do not overlap, there exists $u' \geq w/u$, $\alpha_{k_i}(u') \in F$ and $\alpha_{j_i}(u') \in F$ such that $\alpha_{k_i}(u') \neq \alpha_{j_i}(u')$ and, for all outer addresses $v$, $v < u'$, and $\alpha_{k_i}(v) = \alpha_{j_i}(v)$.

Since $(\zeta \cdot \rho_i A_i)/u \cdot u'$ is an instance of $\alpha_{k_i}/u'$, we have $(\zeta \cdot \rho_i A_i)(u \cdot u') = \alpha_{k_i}(u')$. Similarly, $(\zeta \cdot \sigma_i B_i)(u \cdot u') = \alpha_{j_i}(u')$. These together imply that $(\zeta \cdot \rho_i A_i)(u \cdot u') \in F$, $(\zeta \cdot \sigma_i B_i)(u \cdot u') \in F$, $(\zeta \cdot \rho_i A_i)(u \cdot u') \neq (\zeta \cdot \sigma_i B_i)(u \cdot u')$, and, for all outer occurrence $v'$, $v' < u \cdot u'$, $(\zeta \cdot \rho_i A_i)(v') = (\zeta \cdot \sigma_i B_i)(v')$.

By the nonoverlapping property, for all $v \in O(\alpha_{k_i})$ and $e < v \leq u'$, $\alpha_{k_i}/v$ is not unifiable with the left-hand side of any rewrite rule (similarly for $\alpha_{j_i}$). This implies that for all $v'$, $u < v' \leq u'$, $(\zeta \cdot \rho_i A_i)/v'$ is not unifiable with the left-hand side of any rewrite rule (similarly for $\zeta \cdot \sigma_i B_i$). We then conclude that $(\zeta \cdot \rho_i A_i)/w$ and $(\zeta \cdot \sigma_i B_i)/w$ cannot reduce to an identical term. This plus (*) above contradicts the fact that the given term rewriting system is confluent and hence is Church-Rosser. Therefore, the contrary of the assumption must be true, i.e., $d_i \cdot i \cdot p \cdot \cdots \cdot p_i = \zeta \cdot \sigma_i$.

By Lemma 5.10, $\eta_1$ and $\eta_2$ cannot compare by $\leq$. This completes the proof.

Combining Corollary 5.9 and Theorem 5.11, we obtain the following result.

**Theorem 5.12.** For any $E_{\eta_i}$-unifiable terms $A$ and $B$, the set of $E_{\eta_i}$-unifiers generated by standard narrowing derivations issuing from $H(A, B)$ is a complete and minimal set of $E_{\eta_i}$-unifiers for $A$ and $B$.

The transformation process described in this section has been implicitly used in some of the proofs in [23] for proving properties of reduction. Réty [27] established a general commutation result of narrowing for arbitrary term rewriting systems in order to compare a number of narrowing strategies. The term commutation is similar to but more general than what we have called rearrangement or
transformation. To deal with non-left linear systems, Rétry 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. This definition 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 restriction that the underlying term rewriting systems be regular.

6. Final Remarks

In our previous work [32] we showed that narrowing is incomplete in general for E-unification even in the case of left-linear, nonoverlapping term rewriting systems. In this paper, we discovered that under a slightly tighter equality semantics narrowing is complete. An efficient implementation is made possible by the fact that there is a class of special narrowing derivations that guarantees no redundant solutions will be generated. This result is obtained by using a novel technique of transforming narrowing derivations. The standardization theorem given in this paper is a useful tool in proving completeness of narrowing strategies for other semantics.

Since $E_n$-equality is more general than the constructor-based equality in [19, 34], the results of this paper, particularly the minimality result, apply to the constructor-based equality. For example, although the underlying semantics is different and relies on a partial ordering, the operational semantics given in [19] should also possess the minimality property when dealing with regular term rewriting systems. The operational semantics given by Yukawa is complete but not minimal. Both Fribourg [10] and van Emden and Yukawa [7] use innermost narrowing, which supports total functions only. In general, innermost narrowing is faster than lazy narrowing since the former avoids repeated evaluations of common subterms. However, lazy narrowing can be faster than innermost narrowing when variable dropping rules are used (a rewrite rule $x \rightarrow \beta$ is variable-dropping if $V(\beta) \subseteq V(x)$).

Besides narrowing, there has been another approach based on transformation of systems of equations, initially explored by Martelli and Montanari [21]. This approach is so powerful that it has been incorporated into an E-unification procedure, complete for any equational theories [12]. The E-unification algorithm described in [22] for canonical term rewriting systems is superior to the naive use of narrowing. As a matter of fact, the relationship between this algorithm and standard narrowing can be precisely established to show that standard narrowing derivations can be enumerated effectively by a modified version of their algorithm. In this sense, this algorithm can also be used for nonterminating systems if $E_n$-equality is adopted.
REFERENCES


