SIAM J. COMPUT.
Vol. 14, No. 4, November 1985

019

THE KNUTH-BENDIX COMPLETION PROCEDURE AND THUE SYSTEMS*

DEEPAK KAPUR† AND PALIATH NARENDRAN†

Abstract. The Knuth-Bendix completion procedure for term rewriting systems in many cases provides a decision procedure for equational theories and has been found to have many applications in various areas. We discuss the application of the Knuth-Bendix procedure to Thue systems. We use the notion of a reduced Thue system and show that for every Church-Rosser Thue system, there is a unique reduced Church-Rosser Thue system equivalent to it. Furthermore, the Knuth-Bendix completion procedure, when applied to a Thue system T, always produces the finite reduced Church-Rosser Thue system equivalent to T whenever such a system exists. Similar results can also be proved for almost-confluent Thue systems. Using properties of reduced Church-Rosser systems, we develop conditions under which a class of special Thue systems have equivalent finite Church-Rosser systems. In addition, we show that the completion procedure always terminates on finite parenthesized Thue systems, from which the termination of the completion procedure over ground-term-rewriting systems can be shown immediately. From the results discussed in this paper, we also obtain the termination of the Knuth-Bendix completion procedure for commutative Thue systems (commutative monoids) as a simple corollary.

Key Words. Thue systems, Knuth-Bendix completion procedure, Church-Rosser systems, rewriting systems, almost-confluent systems, ground terms, commutative monoids

1. Introduction.

There has been considerable interest recently in rewriting or transformation systems because of their applications to theorem proving, reasoning about specifications and programs, abstract data types, program transformation and synthesis, algebraic simplification, etc. [5],[7],[16],[17],[22],[24]. The main reason for this interest is the usefulness of the *Church-Rosser property*, which, in general, can be interpreted as implying that the order of applications of transformations makes no difference. Along with the *uniform termination property,* which ensures that every sequence of transformations eventually reaches a result that cannot be further transformed (called a "normal form"), we get a decision procedure for the equational theory induced by the transformations. Note that now any sequence of transformations on an object that produces a normal form will do; the Church-Rosser property ensures that all possible sequences of transformations on the object would produce the same normal form. Transformation systems that have both uniform termination and Church-Rosser properties are called *canonical* systems.

In most cases, however, the systems of transformations or rules that arise may not be canonical. This is where the notion of a completion procedure comes into play; we can attempt to get a canonical set of rules by adding and/or deleting existing rules without altering the underlying theory. Knuth and Bendix [17] introduced a completion procedure for term-rewriting systems in which objects under consideration are (first-order) terms or expressions and transformations are

*Received by the editors October 24, 1983, and in revised form October 1, 1984. A preliminary version of this paper appeared in the *Third Conference on Foundations of Software Technology and Theoretical Computer Science*, held in Bangalore, India, in December, 1983. This paper was typeset at General Electric Corporate Research and Development using *Troff* software developed for the Unix operating system.

†Computer Science Branch, General Electric Corporate Research and Development, Schenectady, New York 12345.

simply rewriting of terms using the rules corresponding to axioms of an algebraic system. Their completion procedure looks for representative terms that have *two* normal forms and then tries to "patch up" by adding a new rule involving the two normal forms. This method has been quite successful in many practical cases: free groups, commutative semigroups, and polynomial ideals are but a few of them. The Knuth-Bendix completion procedure has some other applications as well, such as establishing consistency of equational theories and proving inductive properties using what has become known in the literature as the *inductionless induction* method [23],[24].

In this paper, we discuss the application of the Knuth-Bendix completion procedure on Thue systems. Thue systems are rewriting systems specified using equations over strings (thus there are no variables, but concatenation satisfies the associativity property). It is our belief that understanding the behavior of the Knuth-Bendix completion procedure on Thue systems will also provide some insight into its behavior on term-rewriting systems. Further, Thue systems have recently been studied in their own right by Book and others in the context of formal language theory, monoid presentation, and word problems for finitely presented monoids. The application of the Knuth-Bendix completion procedure for such systems has not been studied so far.

We give a set of transformations that when applied on a Church-Rosser Thue system yields an equivalent reduced (or minimal) Church-Rosser system. We show that these transformations on Church-Rosser Thue systems themselves have the Church-Rosser property, and thus, for every Thue system that has a finite equivalent Church-Rosser Thue system, there is a unique finite reduced Church-Rosser Thue system equivalent to it.[1] Using properties of reduced Church-Rosser systems, we also develop conditions under which a class of special Thue systems have equivalent finite Church-Rosser systems. We also show that a version of the Knuth-Bendix completion procedure, when applied to such a Thue system, always terminates and results in the finite reduced Church-Rosser Thue system equivalent to the original system. We discuss how the completion procedure can be modified to generate almost-confluent Thue systems.

Using concepts and results of the paper, we are also able to prove, in a straightforward manner, that the procedure always halts when applied to commutative Thue systems (i.e., presentations of commutative monoids) [18],[20]. We further exhibit a sufficient condition for the input Thue systems that, when satisfied, guarantees that the procedure will eventually terminate. More specifically, it is shown that if the left-hand sides of rules satisfy a certain condition regarding overlaps, then the Knuth-Bendix completion procedure terminates. Ground-term-rewriting systems can be translated to Thue systems that satisfy this condition, so the result applies to ground-term-rewriting systems, thus giving another proof of termination of the Knuth-Bendix procedure for ground-term-rewriting systems (see also [7]). We are thus also able to give a uniform treatment of some the known results in rewrite rule theory.

The paper is organized as follows: The next section gives definitions and properties of Thue systems. We introduce a new property of Thue systems, called

---

[1]We have recently learned that Nivat and Benois [26] were the first to make this observation. A similar result was also obtained by Lankford and Ballantyne for term-rewriting systems, as reported in [18].

lexicographic confluence, which subsumes the Church-Rosser property. The lexicographic confluence property is used in later sections to study the Knuth-Bendix completion procedure. We prove some properties of the set of irreducible strings of a Thue system. In § 3, we introduce the notion of a reduced Thue system. We prove that if a Thue system T has an equivalent Church-Rosser system, then there exists a unique reduced Church-Rosser system equivalent to it. We also show that for a special Thue system that can be homomorphically mapped into a reduced Church-Rosser system, an equivalent Church-Rosser system exists if and only if the original system itself is Church-Rosser. In § 4, we discuss the application of the Knuth-Bendix completion procedure to Thue systems. First we prove that if there exists a Church-Rosser system equivalent to a Thue system T, then the Knuth-Bendix completion procedure, when applied on T, terminates with a reduced lexicographic confluent, and hence Church-Rosser, system equivalent to T. We also prove the termination of the completion procedure on commutative Thue systems (commutative monoids) as a corollary. Later, we modify the completion procedure to generate reduced almost-confluent systems. Section 5 discusses conditions under which the completion procedure terminates; this is used to show the termination of the completion procedure on parenthesized Thue systems and hence ground-term-rewriting systems.

2. Definitions. Let Σ be a finite alphabet. Σ^* is the monoid freely generated by Σ, or, in other words, the set of all finite strings over Σ. The empty string, which is the identity in the monoid, is denoted by λ, where λ is a symbol not in Σ. The length function on strings, denoted by $|u|$, can be defined as usual: $|\lambda| = 0$, $|ua| = |u| + 1$, where a is in Σ.

A *Thue system* T is a binary relation on Σ^*. The *Thue congruence* generated by T is the reflexive transitive closure $\longleftrightarrow_T^*$ of the relation $\longleftrightarrow_T$, which is defined as follows: for any u and v in Σ^* such that $<u, v>$ is in T or $<v, u>$ is in T, and any x, y in Σ^*, $x\,u\,y \longleftrightarrow_T x\,v\,y$. Two strings, w and z, are *congruent* mod T if $w \longleftrightarrow_T^* z$. Two Thue systems T_1 and T_2 are *equivalent* if and only if they generate the same congruence relation. A Thue system T is *commutative* if and only if for each $<xy, z>$ in $\longleftrightarrow_T^*$, $<yx, z>$ is also in $\longleftrightarrow_T^*$. A Thue system T is called *special* if and only if for every $<u, v>$ in T, either u or v is λ. Henceforth, we shall omit the subscript T whenever it is understood from the context.

Every element of a Thue system T is called an *equation* of T. Some equations of T can be oriented into rules depending upon the length of the sides of each equation. An equation $u = v$ is oriented into a rule $u \rightarrow v$ if $|u| > |v|$, or $v \rightarrow u$ when $|v| > |u|$; such a rule is called *length-reducing* or simply a *reduction*.

An equation $u = v$ in which u and v are of the same length cannot be oriented based on the length of u and v; it is written as $u \,|\!-\!|\, v$. Such an equation is called a *length-preserving rule*. Later, we will discuss another way of orienting rules for the case when a total ordering is introduced on Σ^*. In that case, it would also be possible to uniquely orient equations whose two sides have strings of the same length.

Based on the above classification of the rules, a Thue system T can be partitioned into two components: 1. a subset of length-preserving rules, which will be called LP, and 2. the remaining subset of reductions, which will be called R. Only the rules in the R subset of T will be used for reducing (rewriting) strings unless stated otherwise.

For any x, if there are u and v, as well as a rule $l \to r$ in R of T such that $x = u\, l\, v$, then $x \to u\, r\, v$, read as x *reduces to* $u\, r\, v$ using the rule $l \to r$ of R. The reflexive transitive closure of this relation is the *reduction relation* generated by R of T (also called the *reduction relation* generated by T) on Σ^*; this is denoted by $\to^*$ whereas $\to^+$ stands for the transitive closure of $\to$.

Two strings x and y are called *joinable* (under $\to$) if and only if there exists a z such that $x \to^* z$ and $y \to^* z$. Strings x and y are *almost-joinable* if and only if there exist u and v such that $x \to^* u$, $y \to^* v$, and $u \mid\!-\!\mid^* v$.

A Thue system T is called *Church-Rosser* if for each u, v, such that $u \leftrightarrow^* v$, u, and v are joinable. A Thue system T is called *confluent* if for each u, v, w, such that $u \to^* v$, $u \to^* w$, v, and w are joinable.

Note that the Church-Rosser property and the confluence property, though closely related, are not the same in case of Thue systems because Thue congruence $\leftrightarrow^*$ and the reflexive, symmetric, and transitive closure of the reduction relation $\to$, in general, do not coincide, as the latter does not take into account length-preserving rules. A Church-Rosser system is confluent but the converse does not hold, as shown by the simple example $\{<ab, cd>, <ab, a>, <cd, c>\}$, which is confluent but not Church-Rosser.

A Thue system T is called *almost confluent* if for each u, v such that $u \leftrightarrow^* v$, u and v are almost joinable.

For a commutative Thue system, a string x can be expressed as a k-tuple $<x_1, \ldots, x_k>$ (sometimes called a Parikh vector), where $\Sigma = \{a_1, \ldots, a_k\}$ and x_i is the number of times the letter a_i appears in x. A rule of a commutative Thue system can be expressed as a rule relating two such k-tuples.

2.1. Total ordering on strings and lexicographic confluence. We will introduce another property called "lexicographical confluence" that is not in the literature but, we think, is useful from theoretical as well as practical points of view.

Let $<$ be a total ordering on strings in Σ^* such that the following two properties hold:

1. $|x| < |y| \Rightarrow x < y$, and
2. $x < y \Rightarrow$ for any $u, v, u\, x\, v < u\, y\, v$.

The above properties are closely related to the properties of simplification orderings introduced by Dershowitz [6].

A family of total orderings satisfying the above two properties is the size and lexicographic ordering on strings induced by a total ordering on Σ defined as follows:

$x < y$ if and only if either

1. $|x| < |y|$ or
2. $|x| = |y|$, $x = a\, x'$, $y = b\, y'$, $a, b \in \Sigma$, and either $a < b$ or $a = b$ and $x' < y'$.

Such a total ordering $<$ on Σ^* can be used to orient equations whose two sides are of the same length. So, given a Thue system T, equations in the length-preserving component can also be oriented using $<$. Every rule in T is thus used for reduction. The symbol $\to$ will be used to denote this reduction relation also, as long as it is evident from the context that the whole T is being used for reduction. We shall use $\to'$ to specify the reduction relation induced by

T to distinguish it from the reduction relation $\rightarrow$ induced by R. However, in § 5 we shall drop the prime. The reflexive, symmetric, and transitive closure of the relation $\rightarrow'$ and the Thue congruence $\leftrightarrow^*$ are the same.

A Thue system T is called *lexicographically confluent* (with respect to $<$) if and only if for every u, v, and w such that $u \rightarrow'^* v$ and $u \rightarrow'^* w$, there is a z such that $v \rightarrow'^* z$ and $w \rightarrow'^* z$. We abbreviate a "lexicographically confluent system" to a "lex-confluent system."

THEOREM 2.1. *If T is Church-Rosser then T is lex-confluent.*

Proof. For every length-preserving rule $u \mid\!-\!\mid v$ in a Church-Rosser system T, there is a z such that $u \rightarrow^* z$ and $v \rightarrow^* z$. So, irrespective of the way the length-preserving rules are oriented for $\rightarrow'$, T is lex-confluent. □

Note that if T is lex-confluent, its R component (the set of length-reducing rules) need not be confluent, as the following example illustrates:

$$T = \{\, a\ b \rightarrow c,\ a\ b \rightarrow d,\ c \mid\!-\!\mid d \,\}\ .$$

T is also not Church-Rosser, as c and d do not reduce to the same string.

For any Thue system T, there need not exist an equivalent finite or infinite Church-Rosser system, but there is always an equivalent infinite lex-confluent system which can be obtained trivially from the congruence relation generated by T.

Once an ordering on strings is defined, it can be extended to rules in a natural way as follows:

$$l_1 \rightarrow r_1 < l_2 \rightarrow r_2 \text{ if and only if } l_1 < l_2 \text{ or } l_1 = l_2 \text{ and } r_1 < r_2\ .$$

This ordering on rules is used later in some proofs.

2.2. Irreducible sets for Thue systems. For a Thue system T, x is *irreducible* (mod T) if and only if x cannot be reduced further using the R component of T; x is *minimal* if and only if x is one of the smallest strings in its congruence class induced by T. Clearly, every minimal string is irreducible; but an irreducible string need not be minimal.

Let y be an irreducible string obtained by reducing x using rules of R; y is also called a *normal form* of x. Let $\bar{x}$ stand for a normal form of x under R. If a string x has a unique normal form under R, the normal form of x is also called its *canonical form.* It can be shown that every string has a unique normal form in a Church-Rosser system and thus a string is irreducible if and only if it is minimal.

Let $IRR\ (T)$ be the set of irreducible strings of T. Equivalent Thue systems can have different IRR sets; for example, $T_1 = \{\, abc \rightarrow ab, abc \rightarrow c \,\}$ and $T_2 = \{\, abc \rightarrow c,\ ab \rightarrow c \,\}$ are equivalent, but ab is in $IRR\ (T_1)$ but not in $IRR\ (T_2)$.

THEOREM 2.2. *If $T_1 \subseteq T_2$ and for every rule $l \rightarrow r$ in $T_2 - T_1$, l is reducible in T_1, then $IRR\ (T_1) = IRR\ (T_2)$.*

Proof. That $IRR\ (T_2) \subseteq IRR\ (T_1)$ is obvious. Since every string reducible modulo T_2 is reducible modulo T_1, $IRR\ (T_1) \subseteq IRR\ (T_2)$. □

It can be easily seen that equivalent Church-Rosser (lex-confluent, almost-confluent) systems have the same IRR set. The following also holds.

THEOREM 2.3. *Let T and T' be two equivalent Thue systems. If T is Church-Rosser (lex-confluent) and $IRR(T) = IRR(T')$, then T' is also Church-Rosser (lex-confluent).*

Proof. By contradiction. Assume T' is not Church-Rosser (lex-confluent), then there exists an x such that it has two distinct irreducible forms, say w_1 and w_2, in $IRR(T')$. Since $w_1 \leftarrow\!\!\rightarrow^* w_2$ and T is Church-Rosser, they have a common descendant, implying that at least one of w_1 and w_2 is reducible in T; so, $IRR(T) \neq IRR(T')$, which is a contradiction. □

A similar theorem about almost-confluent Thue systems is proved in [14].

For a Church-Rosser system, all irreducible elements are minimal and minimal elements are unique in their equivalence classes. These two properties can serve as an alternative characterization of Church-Rosser systems. For an almost-confluent system, only the first property holds.

3. Testing for the Church-Rosser property and critical pairs. Nivat and Benois [26] were the first to give a test for the Church-Rosser property and confluence of finite Thue systems. Book and O'Dunlaing [4] showed that this problem is tractable and gave a polynomial time algorithm. Kapur, Krishnamoorthy, McNaughton, and Narendran [13] improve on Book and O'Dunlaing's upper bound and give an $O(|T|^3)$ algorithm for this problem.

The conditions that a Thue system T must satisfy to be Church-Rosser can be stated as follows: We define *critical pairs* from rules; for T to be Church-Rosser, the two strings in each critical pair must be joinable.

1. For a length-preserving rule $l \mid\!-\mid r$ in T, the critical pair is $<l, r>$; l and r must be joinable.

2. Substring case: For rules $l_1 \rightarrow r_1$, $l_2 \rightarrow r_2$ in R, if l_2 is a substring of l_1, then for every u and v such that $l_1 = u\, l_2\, v$, both $p = u\, r_2\, v$ and $q = r_1$, which form a critical pair $<p, q>$, must be joinable.

3. Overlap case: For rules $l_1 \rightarrow r_1$, $l_2 \rightarrow r_2$, if $l_1 = u\, x$, $l_2 = x\, v$, then for every such u, v, and x, both $p = r_1\, v$ and $q = u\, r_2$, which also form a critical pair $<p, q>$, must be joinable. The rules $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$ are called *overlapping* rules.

A critical pair $<p, q>$ is called *nontrivial* if and only if $\bar{p} \neq \bar{q}$, where $\bar{p}, \bar{q}$ are, respectively, normal forms of p and q; otherwise, the critical pair $<p, q>$ is called *trivial*. We assume the existence of an algorithm for computing a normal form of a string, which we will denote by *normal_form*(x, R). A Thue system is Church-Rosser if and only if all its critical pairs are trivial. A pair $<p, q>$ is in *reduced form* with respect to R if p and q cannot be reduced further by R. Henceforth, whenever we refer to a nontrivial critical pair $<p, q>$, we assume that p and q are irreducible.

The test for lex-confluence is similar to that of the Church-Rosser property; we do not have case 1 because length-preserving rules are also oriented.

From the above definition of critical pairs, we have

LEMMA 3.1. *For any critical pair $<p, q>$ of a Thue system T, T is equivalent to $T \cup \{<p, q>\}$.*

4. Reduced Thue systems. A Thue system T is called *reduced* if for every rule in R and LP, neither its left-hand side (lhs) nor its right-hand side (rhs) can be rewritten using the remaining set of rules in R. Thus for every rule $l \rightarrow r$ in a reduced Thue system T, r is irreducible in T and l is irreducible in $T - \{l \rightarrow r\}$. In addition, we have

PROPOSITION 4.1. *For each rule $l \rightarrow r$ in a reduced Thue system T, every proper substring of l is irreducible in T.*

4.1. Church-Rosser Thue systems remain Church-Rosser under reduction. The definition implies that a reduced Church-Rosser system T cannot have an LP component. The following theorem states that Church-Rosser systems have an interesting property, namely that the transformation of stepwise reduction of Church-Rosser Thue systems itself has the Church-Rosser property.

THEOREM 4.2. *Given a Church-Rosser Thue system T, there is a unique reduced Church-Rosser system equivalent to T, effectively obtainable from T.*

We first prove that there is a finite reduced Church-Rosser Thue system equivalent to any finite Church-Rosser system T' and that this reduced system can be effectively obtained from T'. This proof is based on the following two lemmas, which state that the reduction of the lhs and rhs of rules in a Church-Rosser system does not affect the Church-Rosser property of the system. Later, we show the uniqueness of a finite reduced Church-Rosser Thue system.

LEMMA 4.3: *For a Church-Rosser Thue system T, if it has a rule* $w_1 \rightarrow w_2$ *whose lhs can be reduced using the remaining set of rules in T, then the system* $T' = T - \{w_1 \rightarrow w_2\}$ *is equivalent to T and is also Church-Rosser.*

Proof. Since w_1 can be reduced, there is a rule $u_j \rightarrow v_j$ in T' such that $w_1 = l_1 u_j r_1$, and $w_1 \rightarrow l_1 v_j r_1$. Since T is Church-Rosser, there is some irreducible z such that $w_2 \rightarrow^* z$ as well as $l_1 v_j r_1 \rightarrow^* z$. Since $| l_1 v_j r_1 | < w_1$, the rule $w_1 \rightarrow w_2$ of T is not applied in the reduction of either w_2 or $l_1 v_j r_1$ in T, thus implying that they reduce to z in T' also. Thus, w_1 and w_2 are congruent mod T', which means that T and T' are equivalent. The fact that T' is Church-Rosser follows from Theorems 2.2 and 2.3. □

LEMMA 4.4: *If there is a rule* $w_1 \rightarrow w_2$ *in a Church-Rosser system T, such that* w_2 *can be reduced using other rules, then* $T' = T - \{w_1 \rightarrow w_2\} \cup \{w_1 \rightarrow$ *normal-form* $(w_2, T - \{w_1 \rightarrow w_2\})\}$ *is equivalent to T and is Church-Rosser.*

Proof. For any reduction sequence $w_2 \rightarrow^+ z$ *in T*, since the rule $w_1 \rightarrow w_2$ *cannot be applied,* $w_2 \rightarrow^+ z$ *in T' also, which establishes the equivalence of T and T'.* It follows easily from Theorem 2.3 that T' is Church-Rosser. □

Proof of Theorem 4.2. Our algorithm consists of applying the two lemmas as often as possible. Clearly this procedure terminates in a reduced equivalent Church-Rosser system.

But we must also prove that two equivalent reduced Church-Rosser systems T_1 and T_2 are the same. To this end, let $w_1 \rightarrow w_2$ be a rule in T_1 which is not in T_2 (this must be possible, otherwise we are done). Since T_2 is Church-Rosser, $w_1 \rightarrow^+ w_2$ in T_2 because w_2 is an irreducible string of T_1 and also T_2 ($IRR(T_1) = IRR(T_2)$ by Theorem 2.2). So, there is a rule in T_2 that applies to w_1. Every proper substring of w_1 is irreducible (Lemma 4.1); the rule must be $w_1 \rightarrow w'$, where w' is also irreducible. We have w' and w_2 being equivalent in T_2, which is impossible because both are irreducible. □

The above proof easily extends to lexicographically confluent systems; instead of using the ordering on strings induced by their length in the proof of Lemma 4.3, a total ordering $<$ on strings as defined in §§ 2.2 is used.

THEOREM 4.5. *If there is a Church-Rosser Thue system equivalent to a reduced lex-confluent system T, then T itself is Church-Rosser.*

Proof. Since T is reduced, for every rule $l \rightarrow r$ in T, r is irreducible. In particular, if T has a rule $l' \rightarrow r'$, such that $| l' | = | r' |$, then r' is irreducible. But if there is a Church-Rosser system equivalent to T, then neither l' nor r' can be irreducible, implying that T cannot have rules whose two sides are the same length. Thus T is Church-Rosser. □

Thus an algorithm for reducing a Church-Rosser (or lex-confluent) system T is 1. repeatedly throw away a rule in T whose lhs is reducible by the remaining rules, and 2. for every rule in T whose rhs can be reduced using the remaining rules, replace the rhs by its normal form. Later we discuss an extension of this algorithm for almost-confluent Thue systems.

4.2. Special Church-Rosser systems. We discuss below conditions characterizing special Church-Rosser systems that extend some of the results reported in [2],[3]. These conditions are based on homomorphically mapping a special Thue system into a special reduced Church-Rosser Thue system.

Let Δ be a set of generators possibly different from Σ. Let ϕ be a homomorphism from Σ^* to Δ^*; so, $\phi(\lambda) = \lambda'$, where λ' is the identity of the monoid Δ^*, and $\phi(uv) = \phi(u)\phi(v)$. Furthermore, we require that the length function on Δ^*, also denoted as $|\,|$, satisfy the following property: $|u| > |v|$ if and only if $|\phi(u)| > |\phi(v)|$. The *version of T under ϕ*, denoted by T^ϕ, is defined as

$$T^\phi = \{(\phi(u), \phi(v)) \mid (u, v) \in T\}.$$

Again, note that $x \longleftrightarrow^* y \bmod T$ implies $\phi(x) \longleftrightarrow^* \phi(y) \bmod T^\phi$. A homomorphism ϕ is said to be *length-preserving* if $|\phi(a)| = 1$ for all $a \in \Sigma$.

THEOREM 4.6. *Let T be a special Thue system and ϕ be a length-preserving homomorphism such that T^ϕ is a reduced Church-Rosser system. Then T has an equivalent Church-Rosser system if and only if T itself is Church-Rosser.*

Proof. The "if" part is trivial. We prove the "only if" part by contradiction.

Assume T is not Church-Rosser. Then there must be rules $x \to \lambda$ and $y \to \lambda$ such that one of their critical pairs is nontrivial. There are two possibilities:

1. $x = uv$ and $y = vw$ for some $u, v, w \neq \lambda$ and the critical pair is $\langle u, w \rangle$. Clearly $u \neq w$. Note that $\phi(u) \longleftrightarrow^* \phi(w) \bmod T^\phi$. And, $\phi(u)$ must be equal to $\phi(w)$, since T^ϕ is a reduced Church-Rosser system and hence both $\phi(u)$ and $\phi(w)$ must be irreducible mod T^ϕ. Also, $|u| = |w|$. Now if T has an equivalent Church-Rosser system, then there must be a z shorter than both u and w such that $u \longleftrightarrow^* z \longleftrightarrow^* w$. This is obviously impossible since both $\phi(u)$ and $\phi(v)$ are irreducible mod T^ϕ.

2. $x = uyv$ for some u, v such that $uv \neq \lambda$ and the critical pair is $\langle uv, \lambda \rangle$. This case is evidently an impossibility, since $\phi(x)$ must be irreducible mod $T^\phi - (\phi(x), \lambda)$. □

From the above theorem, we immediately get the following results, already reported in [3],[4].

A Thue system T is called *homogeneous* if and only if for every $\langle u, v \rangle$, $\langle x, y \rangle$ in T, either $|u| = |x|$ and $|v| = |y|$, or $|u| = |y|$ and $|v| = |x|$.

COROLLARY 4.7. *Let T be a homogeneous special Thue system. Then T has an equivalent Church-Rosser system if and only if T itself is Church-Rosser.*

Proof. Define $\Delta = \{t\}$ and $\phi(a) = t$ for all a in the alphabet of T. A single-rule special Thue system with a rule $t^n \to \lambda$ is reduced Church-Rosser. □

COROLLARY 4.8. *If T is a special Thue system that has only a single rule, then T has an equivalent Church-Rosser system if and only if T is itself Church-Rosser.*

From the above theorem, we also get a generalization of Corollary 4.8 that suggests how to test whether a class of special Thue systems has equivalent Church-Rosser systems by mapping them to their commutative versions.

The *commutative version* of a Thue system T, denoted by T^C, is T with the commutative law built into it. As mentioned earlier, this can be represented as a

set of relations between k-tuples of integers (often referred to in the literature as Parikh-vectors) where k is the cardinality of Σ. For string w, let $\bar{w}$ stand for its Parikh-vector. Thus,

$$T^C = \{ (\bar{u}, \bar{v}) \mid (u, v) \in T \} .$$

Examples. Let $\Sigma = \{a, b\}$.

1. $T = \{(aba, bb)\}$. $T^C = \{((2,1), (0,2))\}$.
2. $T = \{(aba, \lambda), (baa, \lambda)\}$. $T^C = \{((2,1), (0,0))\}$.

Clearly, for all x, y, $x \longleftrightarrow^* y \bmod T$ implies $\bar{x} \longleftrightarrow^* \bar{y} \bmod T^C$.

We can define a homomorphism ϕ from T to its commutative version T^C as follows: Δ in this case is the set of k basis vectors, where a basis vector is a k-tuple in which exactly one component is nonzero and is 1. The concatenation operation on Δ^* is the vector addition; the identity is the zero vector, and the length of a vector is the sum of all components in the vector. It can be easily verified that ϕ is indeed a homomorphism that maps strings to vectors; furthermore, ϕ is length-preserving. Using Theorem 4.6 above, we have the following:

THEOREM 4.9. *Let T be a special Thue system such that T^C is a reduced Church-Rosser system. Then T has an equivalent Church-Rosser system if and only if T itself is Church-Rosser.*

5. The Knuth-Bendix completion procedure.

5.1. How to obtain a reduced Church-Rosser Thue system.

5.1.1. Completing a Thue system to get an equivalent lex-confluent system. For a finite Thue system T that is not lex-confluent, it is possible to generate an equivalent lex-confluent system by adapting the Knuth-Bendix completion procedure for term-rewriting systems to Thue systems. (Note that because we will discuss mostly lex-confluence in this section, $\rightarrow$ represents the reduction relation generated by all the rules, including the rules whose sides are of the same length.) If the test for lex-confluence fails, all nontrivial critical pairs generated during the lex-confluence test are added to T and the test for lex-confluence on the modified T is repeated. This transformation is performed until the resulting system is lex-confluent.

An inefficient version of the Knuth-Bendix completion procedure for Thue systems is given below in which redundant rules in T are not deleted. Later, a more efficient version will be presented in which redundant rules in T are deleted. The function $CP(T)$ generates all nontrivial critical pairs of T in normal form; if $CP(T)$ is empty, then T is lex-confluent. We do not need to generate critical pairs of rules in T_{i+1} that were also in T_i.

```
Knuth-Bendix Procedure (T):
  i := 0;
  T0 := Normalize(T);
  CE := CP(T0);
  while CE ≠ null do
     Ti+1 := Normalize(Ti ∪ CE);
     i := i+1;
     CE := CP(Ti)
  endwhile
  output(Ti);
```

```
Normalize (T):
 unmark all rules in T.
 while T has an unmarked rule l → r do
  T' := T − { l → r };
  <l', r'> := <normal_form(l, T'), normal_form(r, T') >;
  if <l, r> ≠ <l', r'> and l' ≠ r'
  then
          T := T ∪ { l' → r' };
          mark l' → r'
  endif,
  mark l → r
 endwhile
 return T;
```

Note that the procedure Normalize does not delete any rules. It merely adds rules that are obtained from further reducing the two sides of rules already in T without removing the original rule. The procedure Normalize is nondeterministic if strategies for 1. picking an unmarked rule and 2. computing a normal form of a string are not specified.

There is a possibility of the Knuth-Bendix completion procedure going on forever. We discuss below the conditions under which the Knuth-Bendix procedure is guaranteed to terminate.

Let T_∞ be T_k if the above procedure terminates after k iterations; otherwise, the union of T_i for all i.

LEMMA 5.2. *T and Normalize(T) are equivalent.*

Proof. Since T is a subset of Normalize(T), it needs to be shown that the extra rules in Normalize(T) do not introduce any extra congruence classes. Each rule $l \to r$ in Normalize(T) which is not in T is obtained by reducing some rule $l' \to r'$ in T. Since l = normal-form(l', $T - \{l' \to r'\}$) and r = normal-form(r', $T - \{l' \to r'\}$), and l' and r' are congruent in T, l and r are also congruent in T. □

From Lemmas 3.1 and 5.2, T is equivalent to T_i for each i. And also, T_∞ is equivalent to T.

LEMMA 5.3. *For any i, if $IRR(T_{i+1}) = IRR(T_i)$, then T_i is lex-confluent.*

Proof. By contradiction. If T_i is not lex-confluent, then there is a nontrivial critical pair $<p, q>$ where $p \neq q$ and p and q are in $IRR(T_i)$. Since the rule $p \to q$ or $q \to p$ is included in T_{i+1}, we get $IRR(T_i) \neq IRR(T_{i+1})$, which is a contradiction. □

It is obvious from the above lemma that the *IRR* set keeps decreasing in every iteration of the Knuth-Bendix procedure (since $T_i \subseteq T_{i+1}$, $IRR(T_{i+1}) \subseteq IRR(T_i)$). The procedure thus terminates when the *IRR* set becomes stable; i.e., for some i, $IRR(T_i) = IRR(T_{i+1})$. It can be noted that the set of reducible strings of a Thue system T form an ideal in the free semigroup Σ^*. For commutative semigroups, the irreducible set of strings always becomes stable because of the *finite ascending chain* (also called the *Noetherian*) condition for its ideals [9]. So, we get the result that for commutative Thue systems, the Knuth-Bendix procedure will always terminate as a corollary of the above lemma. A similar argument is used to show the termination of Buchberger's algorithm for finding a Grobner basis for polynomial ideals over a field [5], as well as polynomial ideals over a Euclidean ring [12] (also see references in Mayr and Meyer [20] as well as [11] for similar observations).

The correctness of the Knuth-Bendix completion procedure follows from the following theorem.

THEOREM 5.4. *T_∞ is lex-confluent.*

Proof. By contradiction. If T_∞ is not lex-confluent, it could fail because of any of the above two tests for lex-confluence. This is impossible because if any of the tests fails in an iteration i, then the corresponding nontrivial critical pairs are added in the $i+1^{th}$ iteration. □

For every finite Thue system T, the Knuth-Bendix procedure thus gives an equivalent lex-confluent system T_∞. If the Knuth-Bendix procedure terminates, then T_∞ is finite; otherwise, as we shall see, T_∞ is infinite.

5.1.2. The completion procedure without deletion of redundant rules.

THEOREM 5.5. *Given a finite Thue system T, if there exists a finite lex-confluent Thue system equivalent to T, then the Knuth-Bendix procedure terminates with a finite (not necessarily reduced) lex-confluent Thue system T'', which is equivalent to T.*

Proof. Assume that there exists a finite lex-confluent Thue system equivalent to T. Then using the results of § 3, there is a reduced finite lex-confluent system T' equivalent to T. Furthermore, T_∞ is equivalent to T' and both are lex-confluent, so the set of irreducible strings $IRR(T_\infty)$ and $IRR(T')$ are the same. It needs to be shown that $T_\infty = T_k$ for some k.

LEMMA 5.6. *For every rule $l \rightarrow r$ of T', there is some T_i having $l \rightarrow r$.*

Proof. By contradiction. Assume that the statement is not true. Pick the smallest rule $l \rightarrow r$ in T' that does not appear in any T_i; so, $l \rightarrow r$ is also not in T_∞. Let j be the maximum over the iteration numbers in which the rules in T' less than $l \rightarrow r$ get added; i.e., T_j has all rules of T' less than $l \rightarrow r$.

Since T' and T_∞ are equivalent as well as lex-confluent, l and r are congruent in T_∞, implying that l and r must reduce to the same string in T_∞. Since r is in $IRR(T') = IRR(T_\infty)$, l must reduce to r in T_∞, implying that T_∞ has a rule $l' \rightarrow r'$ that reduces l. By Proposition 4.1, no proper substring of l is reducible, implying that T_∞ must have a rule $l \rightarrow r'$, and both in T_∞ and T', $r' \rightarrow^* r$. Let j' be the iteration number when $l \rightarrow r'$ gets added. In T', $r' \rightarrow^* r$ using rules smaller than the rule $l \rightarrow r$ as $r' < l$, which are in T_∞. So, either in the i^{th} iteration, where $i = \max(j, j')$, or the $i+1^{th}$, r' would be reduced to r and the rule $l \rightarrow r$ would be added, leading to a contradiction. □

We now complete the proof of Theorem 5.5. Since a rule once added never gets deleted, Lemma 5.6 implies that T_∞ contains T'. But T' is finite, so after finitely many iterations of the loop in the procedure, all rules of T' get added into T_∞. But T' is lex-confluent, so after the iteration, say k^{th}, in which the last rule of T' is added to T_∞, the test for lex-confluence would succeed, implying that the procedure would terminate before the loop is executed the $(k+1)^{st}$ time.

5.1.3. The completion procedure with deletion of redundant rules. The above version of the Knuth-Bendix procedure is clearly inefficient, because the original rules from which the simplified rules are obtained in the normalization process are not discarded. However, its proof of termination is easier. An optimized version of the Normalize procedure follows in which redundant rules are discarded.

```
Normalize' (T):
  unmark all rules in T.
  while T has an unmarked rule l → r do
```

```
    T' := T − { l → r };
    < l', r' > := <normal_form(l, T'), normal_form(r, T') >;
    if l' = r'
    then T:= T'
    else
        T := T' ∪ { l' → r' };
        mark l' → r'
    endif
endwhile
return T;
```

We first illustrate the completion procedure on a simple example:

$$T = \{ bb \to a,\ bab \to a,\ baa \to aab \},$$

where $a < b$.

First iteration: $ba \to ab$ and $aab \to ab$ are added; $bab \to aab$ is deleted and $bab \to a$ is replaced by $aa \to a$, which results in the deletion of $aab \to ab$. $T_1 = \{bb \to a,\ aa \to a,\ ba \to ab\}$.

After this, the system T_1 is lex-confluent.

Using the above procedure, a stronger version of Theorem 5.5 can be proved. The termination proof requires an additional step, which is to show that once desired rules get added, they are never deleted. We first show that Normalize'(T) is equivalent to T and later that the *IRR* set of the system obtained using Normalize' is the same as the *IRR* set of the system obtained using Normalize.

LEMMA 5.7. *Normalize'(T) is equivalent to T.*

Proof. It is sufficient to show that in Normalize', the deletion of a rule does not affect the Thue system. There are two situations in which a rule gets deleted:
(a) For a rule $l \to r$ in T, l and r reduce to the same string in $T - \{ l \to r \}$. In this case, congruence relations generated by T and $T - \{l \to r\}$ is the same.
(b) For a rule $l \to r$ in T, if $l' \to r' \neq l \to r$, where $l' =$ normal_form$(l, T - \{l \to r\})$ and $r' =$ normal_form$(r, T - \{l \to r\})$, then $l \to r$ is replaced by $l' \to r'$. In $T - \{l \to r\}$, l' is congruent to l and r' is congruent to r, so $T - \{l \to r\} \cup \{l' \to r'\}$ is equivalent to T.

Note that T_i in the optimized version of the Knuth-Bendix procedure is always a subset of T_i of the unoptimized version. Furthermore, since Normalize'$(T) \subseteq$ Normalize(T), *IRR*(Normalize'(T)) $\supseteq$ *IRR*(Normalize(T)).

LEMMA 5.8. *IRR(Normalize(T)) = IRR(Normalize'(T)).*

Proof. If a rule $l \to r$ is deleted in Normalize', it means that l is reducible by some other rule $l' \to r'$. Thus, if a string is reducible by $l \to r$, then it is also reducible by $l' \to r'$. □

From Lemmas 3.1 and 5.7, we get that T_i is equivalent to T. So, T_∞ is equivalent to T. Further, we have

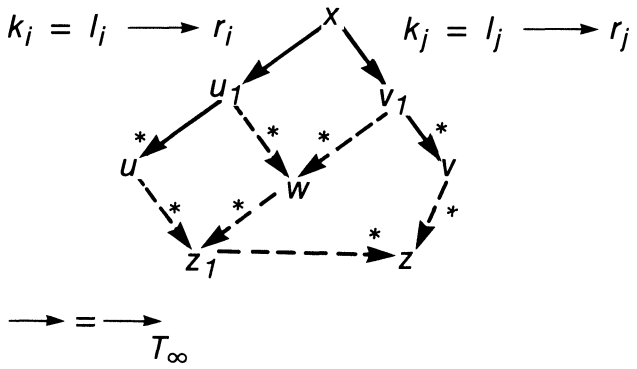
LEMMA 5.9. *T_∞ is lex-confluent.*

Proof. We need to show that for any string $x, u, v, x \to^*_{T_\infty} u$ in m steps, $m \geq 0$, and $x \to^*_{T_\infty} v$ in n steps, $n \geq 0$, there is w such that $u \to^*_{T_\infty} w$ and $v \to^*_{T_\infty} w$. Since T_∞ is Noetherian, the proof has the structure of the proof of Lemma 2.4 in [10], which states that a Noetherian relation is confluent if and only if it is locally confluent.

For $m=0$ or $n=0$, the above trivially holds. So, we assume that both m and $n>0$. Let $x \rightarrow_{T_\infty} u_1$ by a rule $l_i \rightarrow r_i$ and $x \rightarrow_{T_\infty} v_1$ by a rule $l_j \rightarrow r_j$ such that $u_1 \rightarrow^*_{T_\infty} u$ and $v_1 \rightarrow^*_{T_\infty} v$. Let T_{k_i} and T_{k_j} be the Thue systems generated in the Knuth-Bendix procedure having $l_i \rightarrow r_i$ and $l_j \rightarrow r_j$, respectively, such that $|k_i - k_j|$ is minimum.

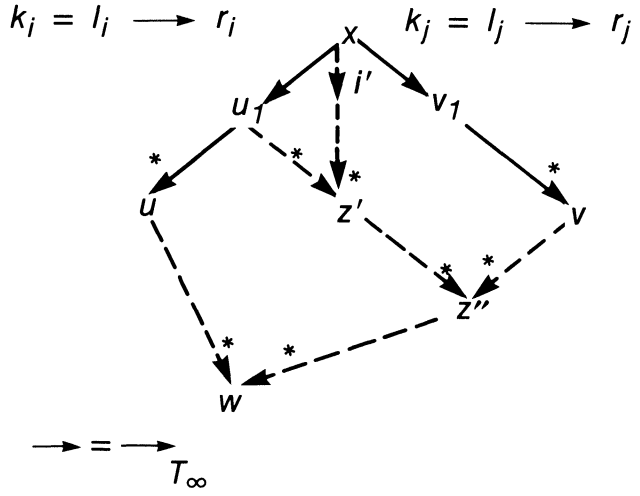
Let x be the smallest string under the ordering $<$ being used for lex-confluence such that there is no w with $u \rightarrow^*_{T_\infty} w$ and $v \rightarrow^*_{T_\infty} w$ and derive a contradiction by induction on $|k_i - k_j|$.

Basis: $|k_i - k_j| = 0$, in the iteration after $\max(k_i, k_j)$, all nontrivial critical pairs of $l_i \rightarrow r_i$ and $l_j \rightarrow r_j$ are generated, giving us the following diagram. But then both $u_1 < x$ and $v_1 < x$, which is a contradiction.



Inductive step: Assume for all $k < |k_i - k_j|$, to show for $|k_i - k_j|$:

Without any loss of generality, assume that $k_j > k_i$. The rule $l_i \rightarrow r_i$ disappears in some iteration i', such that $k_i < i' \leqslant k_j$. So, there is z such that in $T_{i'}$, l_i, and r_i reduce to z; this implies that x and u_1 reduce to some z' in $\rightarrow_{T_\infty}$, since $|k_j - i'| < |k_j - k_i|$, by the inductive hypothesis, there is a z'' such that $z' \rightarrow^*_{T_\infty} z''$ and $v \rightarrow^*_{T_\infty} z''$. Since $u_1 < x$, there is a w such that $u \rightarrow^*_{T_\infty} w$ and $z'' \rightarrow^*_{T_\infty} w$, which is a contradiction.



□

A similar result about the application of the Knuth-Bendix completion procedure on term-rewriting systems is proved in [11]. The above proof is simpler because we are considering Thue systems and also, the version of our completion procedure is not as efficient as in [11]. Using Lemma 5.9, we can prove a stronger version of Theorem 5.5.

THEOREM 5.10. *Given a finite Thue system T, if there exists a finite reduced lex-confluent Thue system T' equivalent to T, then the Knuth-Bendix procedure using the optimized Normalize procedure terminates with T'.*

One way to prove the above theorem is to show a stronger version of Lemma 5.6, that every rule of T' gets added in some iteration of the Knuth-Bendix procedure and never gets deleted afterward (Lemma 5.6 only states that every rule of T' gets added to some T_i in the i^{th} iteration). Theorem 5.10 then follows, because in the iteration k in which the last rule of T' is added, T_k would include T' and every rule other than those in T' would be deleted by the optimized Normalize procedure.

LEMMA 5.11. *Every rule in T' gets added in some iteration i and never gets deleted afterward.*

Proof. By contradiction. Similar to the proof of Lemma 5.6, let $l_j \rightarrow r_j$ be the smallest rule of T' with respect to the ordering $<$ that either

(a) never gets added in any iteration (so it is not in T_∞), or

(b) gets added in iteration i but later is deleted in iteration $i' > i$.

That is, all rules less than $l_j \rightarrow r_j$ in T' get added by some iteration m and never get deleted.

Case (a). The proof of this part is the same as the proof of Lemma 5.6, so it is omitted.

Case (b). Once the rule $l_j \rightarrow r_j$ is added into T_i, the only way it can get deleted in i'^{th} iteration, $i' > i$, is when Normalize' reduces l_j (because r_j is irreducible). And, the only way l_j can be reduced is if there is a rule $l_j \rightarrow r'_j$ in $T_{i'}$. Furthermore, this rule is still unmarked; otherwise, its lhs would have been reduced using the rule $l_j \rightarrow r_j$. There are two subcases: 1. l_j reduces to r_j, implying $r'_j \rightarrow^* r_j$ in $T_{i'}$: Normalize' would then replace the rule $l_j \rightarrow r'_j$ later by $l_j \rightarrow r_j$, meaning that $l_j \rightarrow r_j$ does not get deleted in i', which is a contradiction. 2. l_j reduces to l'_j, implying that $r'_j \rightarrow^* l'_j$ in $T_{i'}$: In this case also, Normalize' would first replace $l_j \rightarrow r_j$ by $l'_j \rightarrow r_j$. Later, Normalize' would also replace the rule $l_j \rightarrow r'_j$ by $l_j \rightarrow r_j$, again meaning that the rule under consideration does not get deleted in i', which is a contradiction. □

Using Theorem 4.5, which relates reduced finite lex-confluent systems and reduced finite Church-Rosser systems, and Theorem 5.10, we have:

THEOREM 5.12. *Given a finite Thue system T, if there exists a finite reduced Church-Rosser system T' equivalent to T, then the optimized Knuth-Bendix procedure terminates with T'.*

Huet in [11] discussed a version of the Knuth-Bendix completion procedure that is more efficient than the optimized version given above. Kapur and Sivakumar [16] have given an improvement over Huet's version. The proof discussed above extends to these versions of the Knuth-Bendix procedure, giving us a result similar to Theorem 5.12.

5.2 Reduced almost-confluent Thue systems. Nivat and Benois [26] also gave conditions under which a Thue system is almost-confluent. For length-reducing rules, the conditions are similar to the conditions for the Church-Rosser

property, except that for every critical pair $<p,q>$, p and q must be almost-joinable, and not necessarily joinable. In addition, interaction between length-reducing rules and length-preserving rules are also considered.

For every rule, we define *critical strings* whose interaction must be considered to check for the almost-confluence property: For a length-reducing rule $l \rightarrow r$, the critical string is l; for a length-preserving rule $b \mid\!-\!\mid s$, both b and s are critical strings. For every pair of rules such that at least one rule is length-reducing, the following two conditions must be met: Let (l_i, r_i) and (l_j, r_j) be such a pair of rules. Then if l_i and l_j are critical strings, then

(a) If $l_i = uv$, $l_j = vw$ for some u, v, w where $|u|, |v|, |w| > 0$ (i.e., the two rules properly overlap), then for every such u, v, and w, the critical pair $(u\, r_j, r_i\, w)$ must be almost-joinable.

(b) If $l_i = u\, l_j\, w$ for some u, w then for every such u, w, the critical pair $(r_i, u\, r_j\, w)$ must be almost-joinable.

Thus, one has to consider all possible overlaps between

(1) left-hand sides of length-reducing rules, and

(2) left-hand side of a length-reducing rule and *both* sides of a length-preserving rule.

It is shown in [14] that the test for the almost-confluence property is PSPACE-complete.

Just as in the case of Church-Rosser systems, given an almost-confluent Thue system, we can obtain a reduced almost-confluent Thue system from it by getting rid of redundancies.

1. For every length-reducing rule $l \rightarrow r$ in R, if r is reducible by other rules in R, then replace the rule by $l \rightarrow r'$, where r' is a normal form of r under R.

2. *(a)* For every length-reducing rule $l \rightarrow r$ in R, if l is reducible by other rules in R, then delete the rule from R. *(b)* For every length-preserving rule $b \mid\!-\!\mid s$, if either b or s is reducible by a rule in R, then delete the rule from T.

As stated in § 3, the above algorithm is a slight extension of the algorithm for obtaining a reduced Church-Rosser system; for a proof of correctness of the algorithm, see [14].

5.2.1. A normal form for almost-confluent systems. Using the above transformations, we can obtain an equivalent reduced almost-confluent system from every almost-confluent system. Note that unlike in the case of reduced Church-Rosser Thue systems, we do not have a unique reduced almost-confluent Thue system because of the length-preserving component. The following theorem tells us this property of reduced almost-confluent systems.

THEOREM 5.13. *Let T_1 and T_2 be two equivalent reduced almost-confluent systems.*

(a) $LP(T_1)$ and $LP(T_2)$ are equivalent; i.e., the congruence relations generated by the length-preserving components of T_1 and T_2 are the same.

(b) For every rule $l \rightarrow r$ in $R(T_1)$, there exists a rule $l \rightarrow r'$ in $R(T_2)$ such that $r \mid\!-\!\mid^* r'$ and vice versa.

Proof. *(a)* Assume there exists a rule $(x \mid\!-\!\mid y)$ in T_1 such that not $x \mid\!-\!\mid^* y$ in T_2. Because T_2 is almost-confluent, this implies that x and y are not irreducible and hence not minimal in T_2. Thus x and y are reducible in T_1 also, which contradicts the fact that T_1 is reduced. This implies that $LP(T_1) = LP(T_2)$ are equivalent.

(b) b and s are almost-joinable in T_2. Because all right-hand sides in a reduced system are irreducible, s is irreducible both in T_1 and T_2. Since no proper substring of b can be reducible, there must be a rule $(b \rightarrow t)$, t irreducible. Therefore s and t are equivalent and irreducible, and this implies $s \vdash\!\dashv^* t$ (by Lemma 2.2). □

It is possible to get rid of more redundancies in the LP component of a reduced almost-confluent Thue system to obtain an LP component that is minimal in the following sense: For any length-preserving rule $b \vdash\!\dashv s$, b and s are not related by other length-preserving rules in LP. Thus, for any length-preserving rule $b \vdash\!\dashv s$ in a reduced almost-confluent system T, if b and s are equivalent using the remaining length-preserving rules in $LP(T)$, then we can delete $b \vdash\!\dashv s$ from $LP(T)$.

In addition, if we assume a total ordering on strings that is an extension of the ordering induced by the length of strings (the size and lexicographic ordering discussed in § 2 is an example of such an ordering), we can orient the length-preserving rules also and use them as reductions for generating normal forms of reduced almost-confluent systems. In that case, the right-hand side of every rule in R can also be normalized with respect to the length-preserving rules. Despite these transformations, it is still not always possible to obtain a unique reduced almost-confluent system equivalent to a given almost-confluent system. For example, consider the following equivalent almost-confluent systems:

$$T_1 = \{\, cbc \vdash\!\dashv bba,\ cd \vdash\!\dashv ab,\ db \vdash\!\dashv bc \,\},$$

$$T_2 = \{\, abb \vdash\!\dashv bba,\ cd \vdash\!\dashv ab,\ db \vdash\!\dashv bc \,\}.$$

Both T_1 and T_2 are reduced. Even if the length-preserving rules are oriented using the length and lexicographic ordering induced by the ordering $a < b < c < d$ on the alphabet, T_1 and T_2 remain reduced.

As shown in § 4, for Church-Rosser systems, the transformations for obtaining a reduced Church-Rosser system have the Church-Rosser property; however, for almost-confluent systems, the transformations for obtaining a reduced almost-confluent system only have the almost-confluence property in the following sense: Given two equivalent almost-confluent systems, the above transformations for obtaining an equivalent reduced almost-confluent system may result in two distinct equivalent reduced almost-confluent systems. The following example illustrates this:

$$T_3 = \{\, cde \rightarrow ab,\ cde \rightarrow ba,\ ab \vdash\!\dashv ba \,\}.$$

The system T_3 is almost-confluent but is not reduced. Two different reduced almost-confluent systems can be obtained from T_3 by applying the above transformations depending upon which of two rules, $cde \rightarrow ab$ or $cde \rightarrow ba$, is considered first.

5.2.2. A completion procedure for almost-confluence. Given a Thue system T which is not almost-confluent, it is possible to generate from T a reduced almost-confluent system equivalent to T, if such a system exists, by a completion procedure which is in the spirit of the Knuth-Bendix completion procedure discussed earlier. In the test for almost-confluence, if for any pair of rules the conditions are not met, i.e., the normal forms of two strings generated from the superposition are not equivalent by the length-preserving rules, then we modify

the system by adding a rule that ensures that the critical pair under consideration is almost-joinable; in this way, we keep adding new rules to both R and LP components of T whenever the need arises until the almost-confluence test is met.

```
Knuth-Bendix Procedure (T):
  i := 0;
  T0 := Normalize(T);
  CE := CP(T0);
 while CE ≠ null do
   Ti+1 := Normalize(Ti ∪ CE);
   i:= i+1;
   CE := CP(Ti)
 endwhile
 output(T);

Normalize(T):
 unmark all rules in T.
 while T has an unmarked a rule l → r do
   T' := T − { <l, r> };
   <l', r' > := <normal_form(l, T'), normal_form(r, T') >;
   if almost_joinable(l', r', T')
   then T := T'
   else
     T := T' ∪ { <l', r'> };
     mark <l', r'>
   endif
 endwhile
 return T;
```

In the above procedures, the procedure CP generates all nontrivial critical pairs (which do not reduce to normal forms equivalent by $LP(T)$). The procedure normal_form generates a normal form of x using the length-reducing rules in T, whereas almost_joinable checks whether two strings are equivalent using the length-preserving and length-reducing rules of T. If strings in a nontrivial critical pair after normalization are of the same length, then the corresponding rule is added to the length-preserving component of T and is subsequently used to check for the almost-joinability condition. This completion procedure to generate almost-confluent systems is thus different from other uses of the Knuth-Bendix completion procedure discussed in the literature, because in this case the simplification theory generated by the length-preserving rules is also being extended; some of the new rules being generated are used as reduction while others are used as simplifications.

The procedure discussed above is not necessarily efficient, because critical pairs among various rules are being checked for again and again; an efficient implementation can be designed based on a version of the procedure given in [11],[16].

The results discussed in § 5.1 can be extended to almost-confluent Thue systems. Using the techniques and proofs developed there, we can show that for a given Thue system T, if there exists a finite almost-confluent Thue system T', then the Knuth-Bendix completion procedure terminates with a reduced almost-confluent Thue system T'' equivalent to T.

Examples.

1. $T = \{a \mid\!-\mid b,\ bab \to b\}$.

After the first iteration of the completion procedure, three rules are added: $baa \to b$, $aab \to b$, and $bbb \to b$. In the next iteration, three rules are added: $aaa \to b$, $bba \to b$, $abb \to b$. Subsequently, the rule $aba \to b$ is added, which makes the final system almost-confluent. The result is

$$\{a \mid\!-\mid b,\ aaa \to b,\ aab \to b,\ aba \to b,\ baa \to b,$$
$$abb \to b,\ bab \to b,\ bba \to b,\ bbb \to b\}.$$

2. $T = \{baa \mid\!-\mid aab,\ bab \to a,\ bb \to a\}$.

First iteration: $aa \to a$, $ab \mid\!-\mid ba$ added and $baa \mid\!-\mid aab$ deleted.
Second iteration: $aba \to ab$ added.
After this, the system

$$\{aa \to a,\ bb \to a,\ bab \to a,\ aba \to ab,\ ab \mid\!-\mid ba\}$$

is almost-confluent.

6. Termination of the Knuth-Bendix procedure on parenthesized Thue systems. Unfortunately it is undecidable whether there exists a finite Church-Rosser Thue system equivalent to a finite Thue system T [27]. Although the results in the previous section ensure that for Thue systems for which this question is decided in the affirmative, the Knuth-Bendix procedure is guaranteed to terminate, there is no way to say a priori by examining a given Thue system, whether the Knuth-Bendix procedure will terminate. Below, we discuss a property of Thue systems that can be checked and for which the Knuth-Bendix procedure terminates.

We define

$$DS(T) = \{x \mid l_i \to^* x \text{ for some } l_i \to r_i \text{ in } T\}.$$

(The letters DS represent "derived strings.")

Two not necessarily distinct strings x and y are called *nonoverlapping* if and only if there do not exist nonempty strings u, v, and w such that $x = u\,v$ and $y = v\,w$. This definition can be extended to a set of strings by requiring that every pair of identical and nonidentical strings from the set are nonoverlapping.

THEOREM. 6.1. *For a Thue system T, if $DS(T)$ is nonoverlapping, then the Knuth-Bendix procedure will terminate, generating a reduced lex-confluent system T' equivalent to T.*

Proof. Because of nonoverlapping of the left-hand sides of rules in T, all critical pairs generated in the first iteration must be due to the lhs of some rule being a substring of the lhs of another. These are less than the largest lhs of rules in T in the total ordering $<$ on strings being used. Furthermore, all critical pairs generated can be obtained by reducing lhs's of rules. The condition that all strings in $DS(T)$ are nonoverlapping implies strings in all critical pairs generated in any iteration are $<$ the greatest lhs in rules in T, which are finitely many. So, the extra rules that can be generated by the Knuth-Bendix completion procedure are only finitely many, implying that the Knuth-Bendix procedure will necessarily terminate. □

A Thue system T is called *parenthesized* if and only if every string in T is properly parenthesized with respect to "(" and ")." Some examples of parenthesized strings are (a), $(a\ b\ (c\ b)\ (b\ c)\ c)$, etc. This definition can be generalized so that two arbitrary strings can be used instead of "(" and ")."

For a parenthesized Thue system T, it is obvious that $DS(T)$ is nonoverlapping, so we have

COROLLARY 6.2. *For a parenthesized Thue system, the Knuth-Bendix completion procedure always terminates.*

Note that a term-rewriting system in which rules involve only ground terms is parenthesized, so it has the property that all its lhs's are nonoverlapping. So for ground-term-rewriting systems, the Knuth-Bendix procedure always terminates once we have an ordering on ground terms that satisfies the replacement and subterm properties (the analog for ground terms of the two properties of an ordering on strings discussed in §§ 2.1). As in parenthesized systems, critical pairs generated in the completion procedure are never going to be greater in size than the largest (with respect to the ordering on ground terms) lhs in a ground-term-rewriting system, and there are only finitely many such terms since the ordering is well founded. According to Dershowitz [7], Lankford was the first to observe that the completion procedure always terminates on ground-term-rewriting systems.

The condition stated above is a particular case of the following general condition: If the set of terms that can possibly serve as superpositions for generating critical pairs is finite, then the Knuth-Bendix completion procedure will always terminate, assuming that it does not abort because the two terms in a nontrivial critical pair cannot be made into a rule.

7. Conclusions. We have introduced the notion of a reduced Thue system. For reduced Thue systems, we have shown a number of properties. It was proved that if there exists a Church-Rosser system equivalent to a Thue system, then there is a unique reduced Church-Rosser Thue system equivalent to it. Using properties of reduced Church-Rosser systems, we have developed conditions under which a class of special Thue systems have equivalent finite Church-Rosser systems.

By using the fact that finite Church-Rosser (lex-confluent) Thue systems themselves have a canonical form, we have shown that if there exists a finite Church-Rosser (lex-confluent) system equivalent to a finite Thue system, then the Knuth-Bendix procedure is guaranteed to terminate with a reduced Church-Rosser (lex-confluent) system equivalent to the original system. The proof depends upon another crucial property, which is that all nontrivial critical pairs generated by the Knuth-Bendix completion procedure can be oriented to give rules; this property is easy to ensure for strings. We have also extended the completion procedure to generate reduced almost-confluent systems. This completion procedure is different from the completion procedures discussed in the literature, as in this case new rules are being added into the set of length-reducing rules, as well as into the set of length-preserving rules.

In addition, we have discussed two methods for showing the termination of the completion procedure. The first method uses the structure of the set of normal forms of a reduction system. For a class of reduction systems for which there cannot exist an infinite ascending chain of sets of reducible strings (by an

ascending chain, we mean that for every i, S_i properly contains S_{i+1}), the completion procedure would terminate because the sets of normal forms keep decreasing in every iteration of the completion procedure. The termination of the completion procedure on commutative Thue systems is shown using this method.

The second method is based on the property that can be informally stated as follows: For systems for which strings appearing in the critical pairs that can be generated by the Knuth-Bendix completion procedure constitute a finite set, the Knuth-Bendix procedure is guaranteed to terminate. The termination of the Knuth-Bendix procedure for ground rewriting systems turns out to be a corollary of this result.

To extend these results to term-rewriting systems, 1. a suitable canonical form for term-rewriting systems needs to be developed (the results in [21] are an attempt in that direction) and 2. orderings on terms have to be devised such that nontrivial critical pairs generated during the Knuth-Bendix completion procedure can always be oriented so that the Knuth-Bendix procedure does not have to be aborted because a critical pair cannot be converted to a rule. It will also be useful to extend the methods discussed above for showing termination of the completion procedure on term-rewriting systems. This would result in a nice characterization of a class of decidable theories.

Acknowledgments. We thank Ron Book, M.S. Krishnamoorthy, Bob McNaughton, Dave Musser and the referees for comments and suggestions which have improved the presentation of the paper.

REFERENCES

[1] R. Book, *Confluent and other types of Thue systems*, J. Assoc. Comput. Mach., 29 (Jan. 1982), pp. 171-182.

[2] ———, *A note on special Thue systems with a single defining relation*, Mathematical Systems Theory, 16 (1983), pp. 57-60.

[3] ———, *Homogeneous Thue systems and the Church-Rosser property*, Discrete Mathematics, 48 (1984), pp. 137-145.

[4] R. Book and C. O'Dunlaig, *Testing for the Church-Rosser property*, Theoretical Computer Science, 16 (1981), pp. 223-229.

[5] B. Buchberger and R. Loos, *Algebraic simplification*, Computing Suppl, 4, Springer Verlag (1982), pp. 11-43.

[6] N. Dershowitz, *Ordering for term rewriting systems*, Theoretical Computer Science, 17 (1982), pp. 279-301.

[7] ———, *Applications of the Knuth-Bendix Completion Procedure*, Laboratory Operation, Aerospace Corporation, Aerospace Report No. ATR-83(8478)-2, 15 May 1983.

[8] ———, *Existence and Construction of Rewrite Systems*, Laboratory Operation, Aerospace Corporation, Aerospace Report No. ATR-82(8478)-3, 1 December 1982.

[9] S. Eilenberg and M.P. Schutzenberger, *Rational sets in commutative monoids*, Journal of Algebra, 13 (1969), pp. 173-191.

[10] G. Huet, *Confluent reductions: abstract properties and applications to term rewriting systems*, J. Assoc. Comput. Mach., 27 (1980), pp. 797-821.

[11] ———, *A Complete Proof of Correctness of the Knuth-Bendix Completion Algorithm*, Journal of Computer and System Sciences, 23 (1981), pp. 11-21.

[12] A. Kandri-Rody and D. Kapur, *Computing the Grobner basis of a polynomial ideal over integers*, Third MACSYMA Users Conference, Schenectady, NY (July 1984), pp. 436-451.

[13] D. KAPUR, M.S. KRISHNAMOORTHY, R. MCNAUGHTON, AND P. NARENDRAN, *An $O(T^3)$ algorithm for testing the Church-Rosser property of Thue systems,* to appear in Theoretical Computer Science.

[14] D. KAPUR AND P. NARENDRAN, *Almost-Confluence and Related Properties of Thue Systems,* Report No. 83CRD258, General Electric Corporate Research and Development, Schenectady, NY, November, 1983.

[15] ———, *A finite Thue system with a decidable word problem and without equivalent finite canonical Thue system,* Theoretical Computer Science, 35 (1985), pp. 337-344.

[16] D. KAPUR AND G. SIVAKUMAR, *Experiments and architecture of RRL, a Rewrite Rule Laboratory,* in Proceedings of an NSF Workshop on the Rewrite Rule Laboratory (6-9 Sept. 1983), General Electric Report 84GEN008, Schenectady, NY, April 1984, pp. 33-66.

[17] D.E. KNUTH AND P.B. BENDIX, *Simple word problems in universal algebras,* in Computational Problems in Abstract Algebras (ed. J. Leech), Pergamon Press, 1970, pp. 263-297.

[18] D.S. LANKFORD AND G. BUTLER, *Experiments with Computer Implementations of Procedures which Often Derive Decision Algorithms for the Word Problem in Abstract Algebra,* Technical Report, MTP-7, Louisiana Tech. University, August 1980.

[19] P. LESCANNE, *Computer Experiments with the REVE Term Rewriting System Generator,* Proc. 10th ACM Symposium on Principles of Programming Languages (1983), pp. 99-68.

[20] E. MAYR AND A.R. MEYER, *The Complexity of the Word Problems for Commutative Semigroups and Polynomial Ideals,* MIT-LCS-TM-199, Lab. for Computer Science, MIT, Cambridge, June 1981.

[21] Y. METIVIER, *About the rewriting systems produced by the Knuth-Bendix completion algorithm,* Information Processing Letters, 16 (1983), pp. 31-34.

[22] D.R. MUSSER, *Abstract data type specification in the AFFIRM system,* IEEE Trans. Software Engg., Vol. 6 (1980), pp. 24-31.

[23] ———, *On proving inductive properties of abstract data types,* Proc. 7th ACM Symposium on Principles of Programming Languages (1980), pp. 154-162.

[24] D.R. MUSSER AND D. KAPUR, *Rewrite rule theory and abstract data type analysis,* in Computer Algebra: EUROCAM '82 (ed. J. Calmet), Lecture Notes in Computer Science, Springer Verlag, Vol. 144, (1982) pp. 77-90.

[25] P. NARENDRAN, *Church-Rosser and related Thue systems,* Ph.D. Dissertation, Dept. of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY (1984).

[26] M. NIVAT (with M. Benois), *Congruences parfaites et quasi- parfaites,* Seminaire Dubreil, 25^e Annee (1971-1972) 7-01-09.

[27] C. O'DUNLAING *Undecidable questions related to Church-Rosser Thue systems,* Theoretical Computer Science, 23 (1983), pp. 339-345.