Theoretical Computer Science 44 (1986) 237-245
North-Holland

NOTE

ON THE EQUIVALENCE PROBLEM FOR REGULAR THUE SYSTEMS

Paliath NARENDRAN
G.E. Corporate Research and Development Center, Schenectady, NY 12345, U.S.A.

Communicated by M. Nivat
Received October 1985

Abstract. A decision procedure is presented for the equivalence problem for regular almost-confluent Thue systems. On the other hand, the equivalence problem for regular preperfect systems is shown to be undecidable.

1. Introduction

There has been much interest in recent years in semigroup presentations or Thue systems both from algebraic and computational points of view [1, 2, 5, 9, 11]. Since Nivat initiated the study of Thue systems along with his colleagues Benoit and Cochet [9, 12], most of the studies so far have concentrated on *finite* Thue systems. Infinite Thue systems started getting attention only very recently, with the pioneering work of Book, Jantzen and Wrathall [3] and Ó'Dúnlaing [10]. Book, Jantzen and Wrathall consider *context-free* Thue systems and Ó'Dúnlaing considers *regular* Thue systems; these are infinite Thue systems finitely specified using context-free languages and regular sets respectively.

One of the problems considered by Ó'Dúnlaing is the following: under what conditions can we check two regular Thue systems for equivalence? It was shown that if the systems are monadic and Church-Rosser, then the equivalence problem is decidable. This result was extended in [7] where it was shown that the equivalence problem is decidable for regular Church-Rosser Thue systems. Here we extend this result to regular *almost-confluent* systems. On the other hand we show the problem to be undecidable for regular *preperfect* systems. (Note the contrast with the case where the Thue systems are *finite*: the equivalence problem is decidable for finite preperfect systems.)

2. Definitions

2.1. Strings over an alphabet

Let Σ be any finite alphabet and Σ^* the set of all possible strings over Σ, including the null string λ. For a string w in Σ^*, $|w|$ denotes its length. Given strings u and

v, their product uv is obtained by concatenating v onto u. A string x is said to be a *prefix* (respectively, *suffix*) of y if there exists z such that $y=xz$ (respectively, zx). A string x is a *proper* prefix (respectively, suffix) of y if x is a prefix (respectively, suffix) of y and $|x|<|y|$. Two strings are said to *overlap* if a proper prefix of one is a proper suffix of the other.

2.2. *Thue systems*

A Thue system T over Σ is a set of pairs of words over Σ^*. The elements of T are called *rules* (or *relations*). The Thue congruence $\leftrightarrow^*$ defined by T is the reflexive, transitive closure of the relation $\leftrightarrow$ defined as follows: if (u, v) is an element of T then, for all x, y, $xuy \leftrightarrow xvy$ and $xvy \leftrightarrow xuy$. If $x \leftrightarrow^* y$, then x and y are said to be *congruent modulo T*. We write

$$x \rightarrow y \quad \text{if } x \leftrightarrow y \text{ and } |x|>|y|,$$

$$x \vdash\!\dashv y \quad \text{if } x \leftrightarrow y \text{ and } |x|=|y|,$$

$$x \mapsto y \quad \text{if } x \leftrightarrow y \text{ and } |x|\geqslant|y|.$$

The reflexive, transitive closure of $\rightarrow$ ($\vdash\!\dashv$, $\mapsto$) is denoted by $\rightarrow^*$ (respectively, $\vdash\!\dashv^*$ $\mapsto^*$). When more than one Thue system is under consideration, we often use these relations with subscripts to avoid confusion.

Two strings x and y are said to be *joinable* if there exists a z such that $x \rightarrow^*$ z and $y \rightarrow^* z$. They are *almost-joinable* if there exist u and v such that $x \rightarrow^* u$, $y \rightarrow^* v$ and $u \vdash\!\dashv^* v$ and they are *resolvable* if there exists z such that $x \mapsto^* z$ and $y \mapsto^* z$. A Thue system T is said to be *Church–Rosser* (*almost-confluent, preperfect*) if $x \leftrightarrow^* y$ implies x and y are joinable (respectively, almost-joinable, resolvable).

A rule (u, v) in a Thue system T is said to be *length-reducing* if $|u| \neq |v|$; we often represent such a rule as $(u \rightarrow v)$ if $|u|>|v|$ and $(v \rightarrow u)$ otherwise. For a length-reducing rule $(u \rightarrow v)$, u is called the *left-hand side* (*lhs*) of the rule and v is called the *right-hand side* (*rhs*). A rule (u, v) is *length-preserving* if $|u|=|v|$ and is often represented as $(u \vdash\!\dashv v)$. Both these representations abuse the notation somewhat but make the exposition a lot easier.

A string x is said to be *reducible by the rule* $(L \rightarrow R)$ if there exist strings u, v such that $x=uLv$ and *reducible* (mod T) if it is reducible by some length-reducing rule in T. A string x is *irreducible* (mod T) if it is not reducible by any rule in T or, equivalently, there is no y such that $x \rightarrow y$. IRR(T) denotes the set of all strings that are irreducible modulo T. If $x \rightarrow^* y$ and $y \in \mathrm{IRR}(T)$, then y is called a *normal form* of x. It can be shown that in a Church–Rosser system every string has a unique normal form. This is not true in general for almost-confluent systems; however, the following holds: for every x, y, z such that y and z are normal forms of x mod T $y \vdash\!\dashv^* z$.

A Thue system T is said to be *reduced* if, for all $(u, v) \in T$, neither u nor v is reducible mod$(T-\{(u, v)\})$.

2.3. Regular Thue systems

A Thue system T is said to be *regular* if it can be represented as $\{(\Gamma_1, x_1), (\Gamma_2, x_2), \ldots, (\Gamma_n, x_n)\}$ where the Γ_i's are regular sets and x_i's are strings. It can be looked upon as a (possibly) infinite system, finitely specified using regular sets. Each rule (Γ_i, x_i) is a *meta-rule* which stands for 'every y in Γ_i is congruent to x_i'. It is easy to see that a regular Thue system can have only finitely many length-preserving rules. Let $\mathrm{LP}(T)$ be the set of length-preserving rules in T and $\mathrm{LD}(T)$ be the set of length-decreasing rules. By reversing all the length-increasing rules, of which there are only finitely many, T can be represented as $\mathrm{LP}(T) \cup \mathrm{LD}(T)$, where $\mathrm{LP}(T)$ is finite and $\mathrm{LD}(T)$ is regular.

Throughout the rest of this note, we assume a regular Thue system T to be specified in the following way:

$$\mathrm{LD}(T) = \{(\Gamma_1, x_1), (\Gamma_2, x_2), \ldots, (\Gamma_m, x_m)\},$$

where the Γ_i are regular expressions and

$$\forall i\, \forall y\colon [(\Gamma_i, x_i) \in \mathrm{LD}(T) \wedge y \in \Gamma_i \Rightarrow |y| > |x_i|],$$

and

$$\mathrm{LP}(T) = \{(u_1 \vdash\!\dashv v_1), \ldots, (u_n \vdash\!\dashv v_n)\}.$$

3. Almost-confluent Thue systems

We first review some important results about almost-confluent Thue systems and show how to apply them to regular almost-confluent Thue systems. The following lemmas were proved in [6].

Lemma 3.1. *Let T be an almost-confluent Thue system and $(L \to R)$ be a rule in it such that R is reducible. Then the Thue system $T' = (T - \{(L, R)\}) \cup \{(L, R')\}$ is equivalent to T and almost-confluent, where R' is a normal form of R.*

Lemma 3.2. *Let T be an almost-confluent Thue system and $(L \to R)$ be a rule in it such that L is reducible modulo $(T - \{(L, R)\})$. Then the Thue system $T' = (T - \{(L, R)\})$ is equivalent to T and almost-confluent.*

Lemma 3.3. *Let T be an almost-confluent Thue system and $(u \vdash\!\dashv v)$ be a rule in it such that u is reducible. Then the Thue system $T' = (T - \{(u, v)\})$ is equivalent to T and almost-confluent.*

Thus, we can conclude that the following theorem holds.

Theorem 3.4. *Every almost-confluent Thue system has an equivalent reduced almost-confluent Thue system.*

Proof. In the light of the above lemmas, such a reduced system is determined by the following steps:

(1) (REP) For every length-reducing rule $(L \to R)$, if R is reducible, then replace the rule by $(L \to R')$, where R' is a normal form of R.

(2) (REM1) For every length-preserving rule (u, v), if either u or v is reducible, then delete the rule.

(3) (REM2) Delete every length-reducing rule whose left-hand side is reducible by some other rule in the system.

(REP and REM stand for 'replacement' and 'removal' respectively.) □

The following important theorem is proved in [5, 6].

Theorem 3.5. *Two reduced almost-confluent Thue systems T_1 and T_2 are equivalent if and only if the following holds:*

(a) $\mathrm{LP}(T_1)$ *is equivalent to* $\mathrm{LP}(T_2)$;

(b) *for every rule $(L \to R)$ in $\mathrm{LD}(T_1)$ there exists a rule $(L \to R')$ in $\mathrm{LD}(T_2)$ such that $R \vdash\!\dashv^* R'$ and vice versa.*

Consider a regular Thue system T. Let $\mathrm{LD}(T) = \{(\Gamma_1, x_1), (\Gamma_2, x_2), \ldots, (\Gamma_m, x_m)\}$. Let $\mathrm{RED}(T)$ be the set of all reducible strings. Clearly,

$$\mathrm{RED}(T) = \Sigma^*\Gamma_1\Sigma^* \cup \cdots \cup \Sigma^*\Gamma_m\Sigma^*$$

and, hence, $\mathrm{RED}(T)$ (and $\mathrm{IRR}(T)$) is a regular set.

Theorem 3.6 ([10]). *There is an algorithm that, from a given regular Thue system T and string x, computes a normal form of x* mod *T.*

We now proceed to show that, given a regular almost-confluent Thue system T, we can effectively compute an equivalent reduced almost-confluent system. That the first step (REP) can be done easily follows from Theorem 3.6. Since every regular Thue system can have only finitely many length-preserving rules, REM1 can also be performed.

The third step (REM2) is the most crucial one. To do that, we first observe that the number of meta-rules is always finite. For two given meta-rules, say $(\Gamma_1 \to x_1)$ and $(\Gamma_2 \to x_2)$, we can remove all the rules in the former that are reducible by the latter in the following way: replace Γ_1 with its intersection with the complement of $\Sigma^*\Gamma_2\Sigma^*$. (Note that $\Sigma^*\Gamma_2\Sigma^*$ is the set of all strings that are reducible by the meta-rule $(\Gamma_2 \to x_2)$. Also, the order in which the meta-rules in the pair are considered is immaterial since if $z \in \Gamma_1 \cap \Gamma_2$, then it does not matter which of the two rules, $(z \to x_1)$ or $(z \to x_2)$, is removed.) This step can now be repeated with every such pair of meta-rules in a sequential manner. The important observation to be made here is that every pair of meta-rules has to be considered only once.

There is also another case we have to consider. Take for instance the meta-rule $(\Gamma_1 \rightarrow x_1)$. There may be rules *within this meta-rule* whose left-hand sides can be reduced by the left-hand sides of other rules within the same meta-rule. (An example would be $(ab^+ \rightarrow a)$, where every left-hand side of the form ab^i, $i \geq 2$, can be reduced by the rule $(ab \rightarrow a)$.) These can be removed by replacing Γ_1 with the intersection of the complement of $\Sigma^+\Gamma_1\Sigma^* \cup \Sigma^*\Gamma_1\Sigma^+$ with it. The following example will illustrate this step. Consider the meta-rule (a^+, λ) over the alphabet $\{a\}$. The complement of $a^+a^+a^* \cup a^*a^+a^+$ is the set $\{a, \lambda\}$. Thus, the new rule we get is $(a \rightarrow \lambda\}$, which is the only rule in the meta-rule whose lhs is not reducible by other rules.

Thus, the procedure for performing the step REM2 can be formulated as follows:

Procedure REM2PROC

```
    for all $i$ from 1 to $m$ do
        $\Gamma_i := \Gamma_i \cap \overline{\Sigma^+\Gamma_i\Sigma^* \cup \Sigma^*\Gamma_i\Sigma^+}$;
    endfor;
    for all $i$ from 1 to $m$ do
        for all $j$ from 1 to $m$ do
            if $i \neq j$ then
                $\Gamma_i := \Gamma_i \cap \overline{\Sigma^*\Gamma_j\Sigma^*}$;
        endfor;
    endfor;
```

It should not be hard to see that the procedure REM2PROC is indeed correct. The only important observation one has to make is the following property of reduction: if a string u is reducible by the rule $L \rightarrow R$ and L is reducible by the rule $L' \rightarrow R'$, then u is also reducible by the rule $L' \rightarrow R'$ (in other words, removing rules whose left-hand sides are reducible by other rules does not affect IRR(T)). Thus, we have the following theorem.

Theorem 3.7. *There is an algorithm that, from a given regular almost-confluent Thue system T, computes an equivalent reduced almost-confluent Thue system.*

The only other result we need is about the decidability of equivalence of reduced regular almost-confluent systems. Before we go into that we show how to perform another step which would make our task a little easier. We call this a '*compression*' step. What we do is this: if $\Gamma_1 \rightarrow x_1$ and $\Gamma_2 \rightarrow x_2$ are two meta-rules and $x_1 \vdash\!\dashv^* x_2$, then we merge the two meta-rules into one, namely, $(\Gamma_1 \cup \Gamma_2) \rightarrow x_1$ (or x_2). Clearly, this does not affect almost-confluence, reducedness or regularity.

Theorem 3.8. *Let T_1 and T_2 be two reduced regular almost-confluent systems that are also 'compressed'. Then they are equivalent if and only if*

(a) LP(T_1) *is equivalent to* LP(T_2) *and*

(b) *for every meta-rule $(\Gamma \rightarrow x)$ in* LD(T_1), *there exists a meta-rule $(\Gamma \rightarrow x')$ in* LD(T_2) *such that $x \vdash\!\dashv^* x'$ and vice versa.*

The proof is easy using Theorem 3.5. The important thing to note is this: suppose $(L_1 \to R)$ and $(L_2 \to R)$ belong to the same meta-rule in T_1. Then compression prevents their getting mapped onto $(L_1 \to R')$ and $(L_2 \to R'')$ respectively, say, where $R' \neq R''$.

Theorem 3.9. *Let T_1 and T_2 be two reduced regular almost-confluent systems. Then it is decidable whether* LP(T_1) *is equivalent to* LP(T_2).

Proof. Both LP(T_1) and LP(T_2) are finite, as mentioned before. Furthermore, the word problem for finite Thue systems that have only length-preserving rules is decidable since they are (trivially) almost-confluent. Hence the result. □

Thus, condition (a) in Theorem 3.8 can be effectively checked. Checking whether condition (b) holds is decidable too since equivalence of regular sets is decidable and, as mentioned before, LP(T_1) and LP(T_2) have decidable word problems. Thus, we have the following theorem.

Theorem 3.10. *The equivalence problem for regular almost-confluent Thue systems is decidable.*

4. Regular preperfect systems

Recall that a Thue system T is called *preperfect* if and only if, for all x and y, x congruent to y implies x and y are resolvable. In this section, we give a proof that the equivalence problem is undecidable for regular preperfect systems, by reducing the emptiness problem for languages accepted by deterministic linear bounded automata to it.

Throughout this discussion, familiarity with the concept and notation of Turing machines is assumed. A *deterministic linear bounded automaton* (DLBA) is a deterministic Turing machine with the additional property that the computation is restricted to that part of the tape where the input string is initially written. This is ensured by enclosing the input string between end-markers—say ¢ and \$—and not allowing the head to move left of the ¢ or right of the \$. In other words, a DLBA $\mathfrak{M}$ is an 8-tuple $(K, \Sigma, \Delta, ¢, \$, \delta, q_0, F)$, where K is the (finite) set of states including the unique initial state q_0 and the set of final states F; Σ is the set of input symbols, Δ is the set of tape symbols which include the left endmarker ¢ and the right endmarker \$, and δ is the transition function.

Configurations of a DLBA $\mathfrak{M}$ are represented by strings of the form uq_iv, where q_i stands for the state the machine is in, u the contents of the tape to the left of the symbol currently being scanned and v the contents of the tape to the right, including the current symbol. (Thus, $\mathfrak{M}$ is in state q_i scanning the first symbol of v.) A configuration is said to be *final* if the state symbol in it corresponds to a final state.

The *language* $L(\mathfrak{M})$ accepted by a DLBA $\mathfrak{M}$ is defined as the set of all strings w such that a final configuration can be reached from the initial state $q_0¢w\$$. In the discussion that follows we assume that none of the DLBA's under consideration accepts the empty string λ.

Theorem 4.1 ([4, p. 281]). *It is undecidable whether a language accepted by a* DLBA *is empty* (*and, hence, it is undecidable whether a language accepted by a* DLBA *over an input alphabet A is A^+*).

Starting from any DLBA $\mathfrak{M}_1$ we can construct an equivalent DLBA $\mathfrak{M}_2$ (i.e., $L(\mathfrak{M}_1) = L(\mathfrak{M}_2)$) with the following properties:

(1) $\mathfrak{M}_2$ has a unique final state q_f and comes to halt at the very left end of the tape (i.e., at the ¢ symbol) if the input is accepted.

(2) If the input is accepted, then $\mathfrak{M}_2$ reconstructs the input before coming to a halt. In other words, if a string w is accepted, then the final configuration will be $q_f¢w\$$.

Let $\mathfrak{M} = (K, \Sigma, \Delta, ¢, \$, \delta, q_0, F)$ be a DLBA with the above two properties. We show how to simulate $\mathfrak{M}$ with a Thue system T consisting of only (finitely many) length-preserving rules. The construction is similar to that given in [8]; the alphabet of T is $(\Delta \cup K)$ (We assume $\Delta \cap K = \emptyset$) and we have rules for every state and tape-symbol pair for which the transition function δ is defined. Motion of the tape-head is simulated by moving the state-symbols accordingly. Thus, we have

$$\begin{aligned} &(q_ia, bq_j) \in T && \text{if } \delta(q_i, a) = (q_j, b, \textsc{Right}),\\ &(cq_ia, q_jcb) \in T && \text{for all } c \in \Delta \text{ if } \delta(q_i, a) = (q_j, b, \textsc{Left}),\\ &(q_ia, q_jb) \in T && \text{if } \delta(q_i, a) = (q_j, b, \textsc{Same}). \end{aligned}$$

The above remarks are sufficient to verify the following claim.

Claim 4.2. *For all $w \in \Sigma^*$, $q_0¢w\$ \vdash\!\dashv^* q_f¢w\$$ if and only if $w \in L(\mathfrak{M})$.*

We now construct regular Thue systems T_1 and T_2 such that T_1 and T_2 are equivalent if and only if $\mathfrak{M}$ accepts every nonempty string over the input alphabet. Let C be a new symbol and let $\{C\} \cup \Delta \cup K$ be the alphabet over which the two Thue systems are defined. Define T_1 as

$$\mathrm{LD}(T_1) = \{(Cq_0¢\Sigma^+\$C \to CC)\} \quad \text{and} \quad \mathrm{LP}(T_1) = T$$

and T_2 as

$$\mathrm{LD}(T_2) = \{(Cq_f¢\Sigma^+\$C \to CC)\} \quad \text{and} \quad \mathrm{LP}(T_2) = T.$$

It is not hard to see that $\mathrm{LD}(T_1)$ and $\mathrm{LD}(T_2)$ are Church–Rosser. Furthermore, $\mathrm{LD}(T_1)$, $\mathrm{LD}(T_2)$, $\mathrm{LP}(T_1)$, and $\mathrm{LP}(T_2)$ neither create nor destroy C's. Note also that for both T_1 and T_2 the length-preserving rules do not interact with the C's. To put it more precisely, if $u = u_1Cu_2C...Cu_n$, where the u_i's do not contain the letter C, and $u \vdash\!\dashv^* v$, then v can be written as $v_1Cv_2C...Cv_n$, where $u_i \vdash\!\dashv^* v_i$ for $1 \leq i \leq n$. Furthermore, all that the length-reducing rules in T_1 (and T_2) do is to erase strings between two C's.

Claim 4.3. *T_1 and T_2 are preperfect.*

Proof. We only give the proof in the case of T_1. The case of T_2, being similar, is left to the reader.

Assume T_1 is not preperfect. Then it can be shown that there must be strings u, v, x, and y such that $u \vdash\!\dashv^* v$, $u \rightarrow x$, $v \rightarrow y$, and x and y are *not* resolvable (see [8]). The following cases have to be considered:

Case 1. $u = v$: This is clearly impossible since $\mathrm{LD}(T_1)$ is Church–Rosser.

Case 2. $u \neq v$: Assume u gets reduced to x by the rule $(C\alpha C \rightarrow CC)$ and v gets reduced by $(C\beta C \rightarrow CC)$, where neither α nor β contains C. Then there exist strings u_1 and u_2 such that $u = u_1C\alpha Cu_2$ and $x = u_1CCu_2$. Similarly, there exist strings β, v_1, v_2 such that $v = v_1C\beta Cv_2$ and $y = v_1CCv_2$.

Case 2(a). $u_1 \vdash\!\dashv^* v_1$: Then it must be that $C\alpha C \vdash\!\dashv^* C\beta C$, $\alpha \vdash\!\dashv^* \beta$, $u_2 \vdash\!\dashv^* v_2$, and thus, $x \vdash\!\dashv^* y$.

Case 2(b). $u_1 \not\vdash\!\dashv^* v_1$: Since C is an 'inert' symbol across which there can be no interaction, either

(i) $u_1 \vdash\!\dashv^* v_1C\beta Cv_{21}$ and $C\alpha Cu_2 \vdash\!\dashv^* v_{22}$ for some v_{21} and v_{22} where $v_{21}v_{22} = v_2$, or

(ii) $u_1C\alpha Cu_{21} \vdash\!\dashv^* v_1$ and $u_{22} \vdash\!\dashv^* C\beta Cv_2$, where $u_{21}u_{22} = u_2$.

Without loss of generality, assume (i). Then

$$x = u_1CCu_2 \vdash\!\dashv^* v_1C\beta Cv_{21}CCu_2 \rightarrow v_1CCv_{21}CCu_2$$

and

$$y = v_1CCv_{21}v_{22} \vdash\!\dashv^* v_1CCv_{21}C\alpha Cu_2 \rightarrow v_1CCv_{21}CCu_2$$

and we have reached a contradiction. □

Claim 4.4. *T_1 and T_2 are equivalent if and only if $L(\mathfrak{M}) = \Sigma^+$.*

Thus, we have the following theorem.

Theorem 4.5. *The equivalence problem for regular preperfect Thue systems is undecidable.*

Acknowledgment

I wish to thank Bob McNaughton for his guidance throughout the writing of this paper. I am also grateful to Dave Musser and Colm Ó'Dúnlaing for their comments and suggestions.

References

[1] R.V. Book, Confluent and other types of Thue systems, *J. Assoc. Comput. Mach.* **29** (1982) 171-182.
[2] R.V. Book, Thue systems and the Church-Rosser property: A survey, in: L.J. Cummings, ed., *Proc. 1st Internat. Conf. on Combinatorics of Words*, University of Waterloo, 1982.
[3] R.V. Book, M. Jantzen and C. Wrathall, Monadic Thue systems, *Theoret. Comput. Sci.* **19** (1982) 231-251.
[4] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory, Languages and Computation* (Addison-Wesley, Reading, MA, 1979).
[5] D. Kapur and P. Narendran, The Knuth-Bendix completion procedure and Thue systems, *SIAM J. Comput.*, **14** (1985) 1052-1072.
[6] D. Kapur and P. Narendran, Almost-confluence and related properties of Thue systems, Rept. No. 83CRD258, G.E. Corporate Research and Development Center, Schenectady, NY, 1983.
[7] P. Narendran, Church-Rosser and related Thue systems, Doctoral Dissertation, Rensselaer Polytechnic Institute, Troy, NY, 1984.
[8] P. Narendran and R. McNaughton, The undecidability of the preperfectness of Thue systems, *Theoret. Comput. Sci.* **31** (1984) 165-174.
[9] M. Nivat (with M. Benois), Congruences parfaites et quasi-parfaites, Séminaire Dubreil, 25e Année, 1971-1972.
[10] C. Ó'Dúnlaing, Finite and infinite regular Thue systems, Doctoral Dissertation, University of California at Santa Barbara, CA, 1981.
[11] F. Otto, Deciding algebraic properties of monoids presented by finite Church-Rosser Thue systems, *Proc. 1st Internat. Conf. on Rewriting Techniques and Applications*, Dijon, France, 1985.
[12] Y. Cochet and M. Nivat, Une généralisation des ensembles de Dyck, *Israel J. Math.* **9** (1971) 389-395.