S. A. GREIBACH

The strong independence of substitution and homomorphic replication

RAIRO. Informatique théorique, tome 12, n° 3 (1978), p. 213-234

<http://www.numdam.org/item?id=ITA_1978__12_3_213_0>

© AFCET, 1978, tous droits réservés.

L’accès aux archives de la revue « RAIRO. Informatique théorique » implique l’accord avec les conditions générales d’utilisation (http://www.numdam.org/conditions). Toute utilisation commerciale ou impression systématique est constitutive d’une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme

Numérisation de documents anciens mathématiques

http://www.numdam.org/
THE STRONG INDEPENDENCE OF SUBSTITUTION
AND HOMOMORPHIC REPLICATION (*)

by S. A. GREIBACH (1)

Communicated by R. V. BOOK

Abstract. — The operations of homomorphic replication and substitution are strongly independent for full semi-AFLs in the sense that if \( \mathcal{L} \) is a full semi-AFL closed under neither homomorphic replication nor substitution, then the closure of \( \mathcal{L} \) under homomorphic replication \( \mathcal{A},(\mathcal{L}) \) is incomparable with the closure of \( \mathcal{L} \) under substitution \( \mathcal{A},(\mathcal{L}) \). The least full AFL containing a full semi-AFL \( \mathcal{L} \) and closed under homomorphic replication \( \mathcal{F},(\mathcal{L}) \) is closed under substitution if and only if either \( \mathcal{L} \subseteq \mathcal{F},(\text{REG}) \) (REG is the family of regular languages) or \( \mathcal{L} = \mathcal{F},(\mathcal{L}) = \mathcal{A},(\mathcal{L}) \).

1. The operation of homomorphic replication (first introduced in [13]) has been used in several recent papers, [3, 4, 5, 17, 19, 26, 27] to characterize a variety of classes of languages arising naturally in different situations—machines, grammars, string relations, complexity classes, etc. For example, the class of finite reversal checking automaton languages is the closure of the regular sets under homomorphic replication and also the closure of the regular sets under iteration of controls on linear context-free grammars [17, 19]. The class of recursively enumerable languages is the smallest class of languages containing the regular sets and closed under intersection and homomorphic replication [3].

The operation of substitution and some of its extensions and restrictions (such as nested iterated substitution [20], iterated substitution [2, 31], and insertion [21]) have also proved useful. For example, the class of derivation bounded languages is the substitution closure of the class of linear context-free languages [14].

Both substitution and homomorphic replication are syntactic operators in the sense that, if \( \mathcal{L} \) is a full semi-AFL not closed under the operator, a proper hierarchy is obtained by iterating applications of the operator and if the closure

(*) Received 24 January 1978.
1) Department of System Science, University of California, Los Angeles.
This work was supported in part by the National Science Foundation under Grants No. DCR74-15091, and MCS78-04725.
of a full semiAFL under the operator is not contained in some other full semiAFL, it cannot be contained in the corresponding AFL [21]. These are very useful properties in gaining strong results in a variety of situations without examining the details of particular machines or grammars, as shown in [15, 16, 17, 19, 20, 21 and 34]. For example, from the facts that the family of linear context-free languages is not closed under concatenation and that its substitution closure is the family of derivation-bounded (finite index) languages, one can conclude at once that the family of derivation-bounded languages is properly contained in the family of context-free languages [15]. Similarly, the fact that there are nonregular linear context-free languages allows one to conclude that iterating controls on linear context-free grammars, starting with the regular sets, produces a proper hierarchy [17, 24]. Perhaps the most impressive example of these techniques appears in Engelfriet’s proof of the tree transducer hierarchy [34], which uses duplication (a special case of homomorphic replication) and other operators.

In this paper, we compare the closure of \( \mathcal{L} \) under homomorphic replication \( \mathcal{H}_r(\mathcal{L}) \) with its closure under substitution \( \mathcal{H}_s(\mathcal{L}) \) for full semiAFLs \( \mathcal{L} \). The two operations are strongly independent for full semiAFLs in the sense that, if \( \mathcal{L} \) is a full semiAFL closed under neither operation, \( \mathcal{H}_r(\mathcal{L}) \) and \( \mathcal{H}_s(\mathcal{L}) \) are incomparable.

Turning to \( \mathcal{F}_r(\mathcal{L}) \), the least full AFL containing \( \mathcal{L} \) and closed under homomorphic replication, we find a different situation. For the family \( \text{REGL} \) of regular languages, \( \mathcal{F}_r(\text{REGL}) \) is closed under substitution. Further, for a full semiAFL \( \mathcal{L} \), \( \mathcal{F}_r(\mathcal{L}) \) is substitution closed if and only if either \( \mathcal{L} \subseteq \mathcal{F}(\text{REGL}) \) [so \( \mathcal{F}_r(\mathcal{L}) = \mathcal{F}_r(\text{REGL}) \)] or \( \mathcal{L} \) is closed under substitution and homomorphic replication. If \( \mathcal{L} \) is a full semiAFL closed under substitution but not homomorphic replication, \( \mathcal{F}_r(\mathcal{L}) \) is properly contained in \( \text{FINITE-VISIT}(\mathcal{L}) \), the closure of \( \mathcal{L} \) under deterministic two-way finite state transductions.

The paper is organized as follows. In section 2, we establish notation and give formal definitions. Section 3 compares \( \mathcal{H}_r(\mathcal{L}) \) and \( \mathcal{H}_s(\mathcal{L}) \). Section 4 discusses the closure of \( \mathcal{F}_r(\mathcal{L}) \) under substitution. Section 5 compares \( \mathcal{F}_r(\text{REGL}) \) and \( \text{FINITE-VISIT}(\mathcal{L}) \) by showing that a generalized Ogden’s lemma holds for the former but not for the latter.

2. It is assumed that the reader is familiar with the basic concepts of automata and formal languages as found, for example, in [10] or [30]. Some of the concepts that are most important for this paper are reviewed here and notation is established.

**Notation:** For a string \( w \), \( |w| \) denotes the length of \( w \). For a finite set \( S \), \( |S| \) denotes the number of members of \( S \). The reversal \( w^R \) of a string \( w \) is the string
obtained by writing \( w \) in reverse order. Let \( w^1 = w \), \( w^{n+1} = w w^n \). For a language \( L \), \( L^R = \{ w^R \mid w \text{ in } L \} \) and for a family of languages \( \mathcal{L} \), \( \mathcal{L}^R = \{ L^R \mid L \text{ in } \mathcal{L} \} \).

Kleene+ is the operation which takes a language \( L \) into \( L^+ = \{ w_1 \ldots w_n \mid n \geq 1, \text{each } w_i \in L \} \). We use \( e \) for the empty string. Kleene* is the operation taking \( L \) into \( L^* = L^+ \cup \{ e \} \). Inverse homomorphism is the operation determined by a homomorphism \( h \) taking \( L \) into \( h^{-1}(L) = \{ w \mid h(w) \in L \} \). By homomorphism, we mean monoid homomorphism, i.e., a function \( h: \Sigma^* \rightarrow \Delta^* \) such that for all \( x, y \in \Sigma^* \), \( h(xy) = h(x)h(y) \).

A homomorphism \( h \) is nonerasing if \( h(w) \neq e \) for \( w \neq e \). A homomorphism \( h \) is linear erasing on a language \( L \) if there is a \( k > 0 \) such that for all \( w \in L \) with \( |w| \geq k \), \( |w| \leq k \) \( |h(w)| \). A class \( \mathcal{L} \) of languages is closed under (nonerasing, linear erasing) homomorphism if for every language \( L \) and any homomorphism \( h \) (that is nonerasing, linear erasing on \( L \)), \( h(L) = \{ h(w) \mid w \in L \} \) is in \( \mathcal{L} \).

We shall reserve \( \Sigma \) with or without subscripts for finite alphabets.

Now we give the definitions and notation used for discussing semiAFLs.

**Definition**: A semiAFL is a family of languages containing at least one nonempty set and closed under union, nonerasing homomorphism, inverse homomorphism, and intersection with regular sets. A full semiAFL is a semiAFL closed under arbitrary homomorphisms. An AFL (full AFL) is a semiAFL (full semiAFL) closed under concatenation and Kleene+.

For a family \( \mathcal{L} \), we use the notation \( \mathcal{M}(\mathcal{L}) \) [respectively, \( \mathcal{M}(\mathcal{L}), \mathcal{F}(\mathcal{L}), \mathcal{F}(\mathcal{L}) \)] for the least semiAFL (respectively, full semiAFL, AFL, full AFL) containing \( \mathcal{L} \). If \( \mathcal{L} = \{ L \} \), we write \( \mathcal{M}(L) \) [respectively, \( \mathcal{M}(L), \mathcal{F}(L), \mathcal{F}(L) \)] and call it a principal semiAFL (respectively, full principal semiAFL, principal AFL, full principal AFL). For families \( \mathcal{L}_1 \) and \( \mathcal{L}_2 \), let

\[
\mathcal{L}_1 \cup \mathcal{L}_2 = \{ L_1 \cup L_2 \mid L_1 \in \mathcal{L}_1, L_2 \in \mathcal{L}_2 \}.
\]

**Definition**: Let \( \rho \) be a function from \( \{ 1, \ldots, n \} \) into \( \{ 1, R \} \) and for \( 1 \leq i \leq n \), let \( h_i \) be a homomorphism. The operation on languages defined by

\[
\langle \rho, h_1, \ldots, h_n \rangle(L) = \{ (h_1(w))^{\rho(1)} \ldots (h_n(w))^{\rho(n)} \mid w \in L \},
\]

is a homomorphic replication of type \( \rho \). It is nonerasing if each \( h_i \) is nonerasing. Let

\[
\mathcal{L}_\rho = \{ \langle \rho, h_1, \ldots, h_n \rangle(L) \mid L \in \mathcal{L}, h_1, \ldots, h_n \text{ homomorphisms} \}.
\]

We shall add the subscript \( r \) to specify a family closed under homomorphic replication of the appropriate type. Thus, \( \mathcal{M}_r(\mathcal{L}) \) [respectively, \( \mathcal{M}_r(\mathcal{L}), \mathcal{F}_r(\mathcal{L}), \mathcal{F}_r(\mathcal{L}) \)], is the least semiAFL (respectively, full semiAFL, AFL, full AFL)
containing \( \mathcal{L} \) and closed under nonerasing homomorphic replication. Clearly, a full semiAFL or AFL closed under nonerasing homomorphic replication is closed under homomorphic replication.

Two additional classes of operation we use are the \( a \)-transductions and the substitutions.

**Definition:** An \( a \)-transducer is a tuple \( M = (K, \Sigma, \Delta, H, q_0, F) \) where \( K \) is a finite set of states, \( q_0 \in K, F \subseteq K \), \( \Sigma \) is a finite input alphabet, \( \Delta \) is a finite output alphabet and \( H \) is a finite subset of \( K \times \Sigma^* \times \Delta^* \times K \). An ID (Instantaneous Description) of \( M \) is any member of \( K \times \Sigma^* \times \Delta^* \). If \( (q, uw, y) \) is an ID and \( (q, u, v, q') \in H \), then we write \( (q, uw, y) \vdash (q', w, yv) \). The relation \( \vdash * \) among ID's is the transitive reflexive extension of \( \vdash \).

For \( w \in \Sigma^* \),

\[
M(w) = \{ v \mid \exists q \in F, (q_0, w, e) \vdash^* (q, e, v) \}
\]

and for a language \( L \),

\[
M(L) = \{ v \mid \exists w \in L, v \in M(w) \}.
\]

We call \( M(L) \) an \( a \)-transducer mapping of \( L \) if

\[
H \subseteq K \times (\Sigma \cup \{ e \}) \times (\Delta \cup \{ e \}) \times K,
\]

we call \( M \) 1-bounded.

Intuitively, an \( a \)-transducer is a nondeterminstic one-way finite state transducer with accepting states; output is “legal” only when the machine is in an accepting state.

We shall use the fact that every full semiAFL is closed under \( a \)-transducer mapping and, more strongly, is characterized by union and \( a \)-transducer mapping [11, 12]. That is, for any family of languages \( \mathcal{L} \), containing at least one nonempty language,

\[
\tilde{\mathcal{M}}(\mathcal{L}) = \{ M_1(L_1) \cup \ldots \cup M_n(L_n) \mid L_1, \ldots , L_n \in \mathcal{L}, M_1, \ldots , M_n \text{ are } a\text{-transducers} \}
\]

and for a language \( L \),

\[
\mathcal{M}(L) = \{ M(L) \mid M \text{ is an } a\text{-transducer} \}.
\]

Further, restriction to 1-bounded \( a \)-transducers causes no loss in power [12]. Properties of semiAFLs and AFLs can be found in [11, 12]; \( a \)-transducers are also described in [7].

**Definition:** A substitution \( \tau \) on a finite alphabet \( \Sigma \) takes each \( a \) in \( \Sigma \) into a language \( \tau(a) \). We extend \( \tau \) to words by \( \tau(e) = \{ e \} \) and \( \tau(xy) = \tau(x) \tau(y) \) and to
languages \( L \) by \( \tau(L) = \{ u \mid \exists w \in L, u \in \tau(w) \} \). If \( \tau(a) \) is in \( \mathcal{L} \) for each \( a \) in \( \Sigma \), then \( \tau \) is an \( \mathcal{L} \)-substitution. If \( e \) is not in \( \tau(a) \) for each \( a \) in \( \Sigma \), then \( \tau \) is nonerasing. For families of languages \( \mathcal{L}_1 \) and \( \mathcal{L}_2 \), the families of languages obtained by substituting members of \( \mathcal{L}_2 \) into \( \mathcal{L}_1 \) are
\[
\mathcal{L}_1 \circ \mathcal{L}_2 = \{ \tau(L) \mid L \in \mathcal{L}_1, \tau \text{ is an } \mathcal{L}_2 \text{-substitution} \}
\]
and
\[
\mathcal{L}_1 \circ\sigma \mathcal{L}_2 = \{ \tau(L) \mid L \in \mathcal{L}_1, \tau \text{ is a nonerasing } \mathcal{L}_2 \text{-substitution} \}.
\]
If \( \mathcal{L} \circ \mathcal{L}_1 \subseteq \mathcal{L} \) (\( \mathcal{L} \circ \mathcal{L}_1 \subseteq \mathcal{L} \)), then \( \mathcal{L} \) is closed under substitution (nonerasing substitution) by \( \mathcal{L}_1 \); if \( \mathcal{L}_1 \circ \mathcal{L} \subseteq \mathcal{L} \) (\( \mathcal{L}_1 \circ \mathcal{L} \subseteq \mathcal{L} \)), then \( \mathcal{L} \) is closed under substitution (nonerasing substitution) into \( \mathcal{L}_1 \). If \( \mathcal{L} \circ \mathcal{L} \subseteq \mathcal{L} \) (\( \mathcal{L} \circ \mathcal{L} \subseteq \mathcal{L} \)), then \( \mathcal{L} \) is closed under substitution (nonerasing substitution).

We add the subscript \( \sigma \) to indicate closure under nonerasing substitution. Thus \( \mathcal{M}_\sigma(\mathcal{L}) \) [respectively, \( \mathcal{H}(\mathcal{L}), \mathcal{F}_\sigma(\mathcal{L}), \mathcal{F}_\sigma(\mathcal{L}) \)] is the least semiAFL (respectively, full semiAFL, AFL, full AFL) containing \( \mathcal{L} \) and closed under nonerasing substitution. A full semiAFL closed under nonerasing substitution is closed under substitution [22]. Full semiAFLs are closed under regular substitution (substitution by regular languages) while full AFLs are also closed under substitution into regular sets. A semiAFL closed under nonerasing substitution is an AFL, so \( \mathcal{F}_\sigma(\mathcal{L}) = \mathcal{M}_\sigma(\mathcal{L}) \) and we normally use just \( \mathcal{M}_\sigma(\mathcal{L}) \) or \( \mathcal{H}_\sigma(\mathcal{L}) \).

We let \( \text{REGL} \) denote the family of regular languages and \( \text{CF} \) the family of context-free languages. Two useful facts about substitutions are that substitution (and nonerasing substitution) are associative on semiAFLs [e.g., \( \mathcal{L}_1 \circ (\mathcal{L}_2 \circ \mathcal{L}_3) = (\mathcal{L}_1 \circ \mathcal{L}_2) \circ \mathcal{L}_3 \)] and that for any family of languages \( \mathcal{L} \), \( \mathcal{F}(\mathcal{L}) = \text{REGL} \circ \mathcal{H}(\mathcal{L}) \) [11, 12, and 22].

3. In this section, we establish the strong independence of substitution and homomorphic replication using “syntactic lemmas” akin to those in references [15] through [21], which state that languages of certain forms can only be built up in certain ways. The first syntactic lemma echoes example 3.1 of [21], the example on p. 27 of [20] and lemma 4.4 of [18]. It is similar to the “copying” theorems in [2, 8 and 9]; a close relative appears in [33].

**Lemma 3.1**: Let \( \mathcal{L} \) be a family of languages closed under concatenation by unit sets (i.e., sets of size 1), intersection with sets of the form \( \Sigma^* \) for \( \Sigma \) a finite vocabulary, and under union and (nonerasing) homomorphism. Let \( \mathcal{L} \subseteq \Sigma^* \) be a language with the following property:

(\( \star \)) if \( xyz \) is in \( L \) then either \( \#(L \cap \Sigma^* y \Sigma^*) \leq 1 \) or else \( \#(L \cap x \Sigma^* z) \leq 1 \).
If \( L = \tau(L_1) \) for \( L_1 \) in \( \mathcal{L} \) and \( \tau \) is an \( \mathcal{L} \)-substitution (a nonerasing \( \mathcal{L} \)-substitution), then \( L \) is in \( \mathcal{L} \).

**Proof:** Let \( L_1 \subseteq \Sigma^* \), let \( \Sigma_1 = \{ a \in \Sigma | \#\tau(a) = 1 \} \) and let \( \Sigma_2 = \Sigma - \Sigma_1 \). The language \( L_2 = \tau(L_1 \cap \Sigma^*_1) \) is in \( \mathcal{L} \). Suppose \( uav \) is in \( L_1 \) with \( \#\tau(a) \geq 2 \) and \( \tau(u) \neq \emptyset \neq \tau(v) \). Let \( x \) be in \( \tau(u) \), \( y \) in \( \tau(a) \) and \( z \) in \( \tau(v) \). Then \( xyz \) is in \( L \). If \( \#(L \cap \Sigma^* y \Sigma^*) \geq 2 \), then \( \#(L \cap x \Sigma^* z) = 1 \), so \( \tau(a) = \{ y \} \), a contradiction. Hence there are unique \( x_a \) and \( z_a \) such that \( x_a y z_a \) is in \( L \). Let \( \Sigma_3 = \{ a \in \Sigma_2 | (L_1 \cap \Sigma^* a \Sigma^*) \neq \emptyset \} \) and let \( L_3 = \bigcup_{a \in \Sigma_3} x_a \tau(a) y_a \).

Then \( L_3 \) is in \( \mathcal{L} \) and so \( L = L_2 \cup L_3 \) is in \( \mathcal{L} \). \( \square \)

Lemma 3.1 enables us to say that substitutions cannot "help" in building up certain types of homomorphic replications.

**Lemma 3.2:** Let \( \mathcal{L} \) be a semiAFL. Let \( A \subseteq \Sigma^* \), \( A \in \mathcal{L} \), and let \( c \) be a symbol not in \( \Sigma \). Then for any \( k \geq 3 \), \( t_1, \ldots, t_k \in \{ 1, R \} \) the language \( L = \{ w^1 c \ldots c w^k | w \) in \( A \} \) is in \( \mathcal{M}_\sigma(\mathcal{L}) \) if and only if it is in \( \mathcal{L} \), and if \( \mathcal{L} \) is a full semiAFL, \( L \) is in \( \mathcal{M}_\sigma(\mathcal{L}) \) if and only if it is in \( \mathcal{L} \).

**Proof:** The language \( L \) has property (*) of lemma 3.1, since for any \( xyz \) in \( L \) either \( y \) contains at least two \( c \)'s and uniquely determines \( x \) and \( z \) or else \( x \) contains at least one \( c \) and uniquely determines \( yz \) or \( z \) contains at least one \( c \) and uniquely determines \( xy \). For a semiAFL \( \mathcal{L} \), \( \mathcal{M}_\sigma(\mathcal{L}) \) can be obtained from \( \mathcal{L} \) by repeated nonerasing \( \mathcal{L} \)-substitutions and \( \mathcal{M}_\sigma(\mathcal{L}) \) by repeated \( \mathcal{L} \)-substitutions. But lemma 3.1 tells us that if \( L \) is not in \( \mathcal{L} \), this will not suffice to obtain \( L \) in \( \mathcal{M}_\sigma(\mathcal{L}) \) or, if \( \mathcal{L} \) is already a full semiAFL, to obtain \( L \) in \( \mathcal{M}_\sigma(\mathcal{L}) \). \( \square \)

**Theorem 3.3:** Let \( \mathcal{L} \) and \( \mathcal{L}_1 \) be semiAFLs with \( \mathcal{L} \) closed under linear erasing homomorphism

(1) \( \mathcal{M}_\tau(\mathcal{L}_1) \subseteq \mathcal{M}_{\sigma}(\mathcal{L}) \) if and only if \( \mathcal{M}_\tau(\mathcal{L}_1) \subseteq \mathcal{L} \)

and

(2) \( \mathcal{M}_\tau(\mathcal{L}_1) \subseteq \mathcal{M}_{\sigma}(\mathcal{L}) \) if and only if \( \mathcal{M}_{\tau}(\mathcal{L}_1) \subseteq \mathcal{L} \).

If \( \mathcal{L} \) is not closed under nonerasing homomorphic replication, then

\( \mathcal{M}_\tau(\mathcal{L}) - \mathcal{M}_{\sigma}(\mathcal{L}) \neq \emptyset \)

and if \( \mathcal{L} \) is also a full semiAFL,

\( \mathcal{M}_\tau(\mathcal{L}) - \mathcal{M}_{\sigma}(\mathcal{L}) \neq \emptyset \).

R.A.I.R.O. Informatique théorique/Theoretical Computer Science
(3) $\mathcal{M}_\sigma(L)$ is closed under nonerasing homomorphic replication if and only if $L = \mathcal{M}_\sigma(L) = \mathcal{M}_\sigma(L)$ and if $L$ is a full semiAFL, $\mathcal{M}_\sigma(L)$ is closed under homomorphic replication if and only if $L = \mathcal{M}_\sigma(L) = \mathcal{M}_\sigma(L)$.

Proof: Suppose $L_2 = \mathcal{M}_\sigma(L_1)$ is contained in $\mathcal{M}_\sigma(L)$. Let $A$ be any member of $L_2$ and $c$ a new symbol. Since $L_2$ is closed under nonerasing homomorphic replication, $\{wcwRcw|w \in A\}$ is in $\mathcal{M}_\sigma(L)$ and hence, by lemma 3.2, in $L$. Since $L$ is closed under linear erasing homomorphism, $A$ is in $L$. A similar argument applies to $\mathcal{M}_\sigma(L_1)$. This establishes (1). Statement (2) follows from (1), taking $L_1 = L$, and using the fact that if $L$ is a full semiAFL, $\mathcal{M}_\sigma(L) = \mathcal{M}_\sigma(L)$[22].

From (2) we know that if $L \notin \mathcal{M}_\sigma(L)$, $\mathcal{M}_\sigma(L)$ is not closed under nonerasing homomorphic replication. A full semiAFL is closed under homomorphic replication if and only if it is closed under nonerasing homomorphic replication, so if $L$ is a full semiAFL with $L \notin \mathcal{M}_\sigma(L)$, $\mathcal{M}_\sigma(L)$ is not closed under homomorphic replication. If $L \notin \mathcal{M}_\sigma(L)$, then $\mathcal{M}_\sigma(L)$ is not contained in $L$ [since $\mathcal{M}_\sigma(L) \subseteq \mathcal{M}_\sigma(L)$], and hence not in $\mathcal{M}_\sigma(L)$ by (1). This establishes (3).

The analogous result for nonerasing substitution is established only for full semiAFLs, using the fact that if $L$ is a full semiAFL not closed under homomorphic replication, $\mathcal{M}_\sigma(L)$ is not an AFL.

Theorem 3.4: Let $L$ be a full semiAFL and let $L_1$ be a semiAFL. Then $\mathcal{M}_\sigma(L)$ is closed under nonerasing substitution into $L_1$ if and only if $L = \mathcal{M}_\sigma(L) = \mathcal{M}_\sigma(L)$ and $L$ is closed under nonerasing substitution into $L_1$.

Proof: Now $L_1 \sigma \mathcal{M}_\sigma(L) = (L_1 \sigma \text{REGL}) \sigma \mathcal{M}_\sigma(L)$

$= L_1 \sigma (\text{REGL} \sigma \mathcal{M}_\sigma(L)) = L_1 \sigma \mathcal{F}_\sigma(M_\sigma(L))$,

so if $\mathcal{M}_\sigma(L)$ is closed under nonerasing substitution into $L_1$, then it is an AFL and equal to $\mathcal{F}_\sigma(L)$. But if $L \notin \mathcal{M}_\sigma(L)$, it is not an AFL [17, 19, 21].

Putting together theorems 3.3 and 3.4, we obtain our strong independence theorem.

Theorem 3.5: Substitution and homomorphic replication are strongly independent operations on full semiAFLs in the sense that if $L$ is a full semiAFL not closed under substitution or homomorphic replication, then $\mathcal{M}_\sigma(L)$ and $\mathcal{M}_\sigma(L)$ are incomparable.

Corollary: The family of derivation bounded languages is incomparable with the family of finite reversal checking automaton languages.

If we try to extend theorem 3.5 to compare $\mathcal{F}_\sigma(L)$ and $\mathcal{M}_\sigma(L)$, a different situation emerges. For any semiAFL $L$, $\mathcal{M}_\sigma(L)$ is an AFL while for a full...
semiAFL $\mathcal{L}$, $\hat{\mathcal{M}}_r(\mathcal{L})$ cannot be an AFL unless $\mathcal{L} = \hat{\mathcal{M}}_r(\mathcal{L}) = \hat{\mathcal{F}}_r(\mathcal{L})$. If $\mathcal{L} \neq \hat{\mathcal{M}}_r(\mathcal{L})$, then not only is $\hat{\mathcal{M}}_r(\mathcal{L})$ not an AFL but $\hat{\mathcal{F}}(\hat{\mathcal{M}}_r(\mathcal{L}))$ is not closed under homomorphic replication [17, 19, 21]. We can build $\hat{\mathcal{F}}_r(\mathcal{L})$ from $\mathcal{L}$ by alternative applications of homomorphic replication and AFL closure as

$$\hat{\mathcal{F}}_r(\mathcal{L}) = \bigcup_{i \geq 0} \mathcal{L}_i$$

for $\mathcal{L}_0 = \mathcal{L}$

and

$$\mathcal{L}_{i+1} = \begin{cases} \hat{\mathcal{M}}_r(\mathcal{L}) & \text{if } i \text{ is odd}, \\ \hat{\mathcal{F}}(\mathcal{L}) & \text{if } i \text{ is even} \end{cases}$$

and if $\mathcal{L}_0 \neq \mathcal{L}_1$, then $\mathcal{L}_i \neq \mathcal{L}_{i+1}$ for all $i$.

Examining the closure of $\hat{\mathcal{F}}_r(\mathcal{L})$ under substitution, a curious situation emerges. For $\hat{\mathcal{F}}_r(\text{REGL})$ is closed under substitution and this is basically the only situation in which substitution closure holds unless $\mathcal{L} = \hat{\mathcal{M}}_r(\mathcal{L}) = \hat{\mathcal{F}}_r(\mathcal{L})$, and $\mathcal{L}$ is substitution closed, for example, if $\mathcal{L}$ is the family of recursively enumerable languages which is closed under substitution and homomorphic replication [3]. We discuss this in the next section.

4. We first establish the closure of $\hat{\mathcal{F}}_r(\text{REGL})$ under substitution. The only construction needed already appears in [13], in the proof that $\mathcal{L}_p$ is a full semiAFL whenever $\mathcal{L}$ is a full semiAFL. We excerpt the basic idea as lemma 4.1.

**Lemma 4.1:** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be full semiAFLs and let $\rho: \{1, \ldots, k\} \to \{1, R\}$. Then

$$(\mathcal{L}_1)_\rho \cap \mathcal{L}_2 \subseteq (\mathcal{L}_1 \cap \mathcal{L}_2)_\rho.$$ 

**Proof:** Part (d) of the proof of theorem 3.1 of [13] shows how to express $\tau(\langle \rho, h_1, \ldots, h_k \rangle(L))$ as $\langle \rho, g_1, \ldots, g_k \rangle(\bar{\tau}(L))$ where $\tau$ and $\bar{\tau}$ are regular substitutions. The only property of REGL used is closure under inverse homomorphism, intersection with regular sets and concatenation. Since $(\mathcal{L}_1)_\rho \cap \mathcal{L}_2 = (\mathcal{L}_1)_\rho \cap \bar{\tau}(\mathcal{L}_2)$, we may as well assume that $\mathcal{L}_2$ is an AFL and so closed under concatenation. Hence any $\mathcal{L}_2$-substitution into a homomorphic replication of a language $L$ in $\mathcal{L}_1$ can be expressed as the homomorphic replication of a language obtained by some $\mathcal{L}_2$-substitution into $L$, whence the desired result. $\Box$

**Theorem 4.2:** Let $\mathcal{L}$, $\mathcal{L}_1$ and $\mathcal{L}_2$ be full semiAFLs such that $\mathcal{L}_1 \cap \mathcal{L}_2 \subseteq \hat{\mathcal{F}}_r(\mathcal{L})$. Then $\hat{\mathcal{F}}_r(\mathcal{L}_1) \cap \mathcal{L}_2 \subseteq \hat{\mathcal{F}}_r(\mathcal{L})$. In particular, $\hat{\mathcal{F}}_r(\mathcal{L})$ is closed under substitution into $\hat{\mathcal{F}}_r(\text{REGL})$.

**Proof:** Let $\mathcal{A}_0 = \mathcal{L}_1$ and for $i \geq 0$, let $\mathcal{A}_{i+1} = \hat{\mathcal{F}}(\hat{\mathcal{M}}_r(\mathcal{L})) = \text{REGL} \sigma \hat{\mathcal{M}}_r(\mathcal{L})$. 

R.A.I.R.O. Informatique théorique/Theoretical Computer Science
Then $\mathcal{F}_r(L_1) = \bigcup_{i \geq 0} L_i$.

We use lemma 4.1 to establish by induction on $i$, that $\mathcal{A}_i \sigma L_2 \subseteq \mathcal{F}_r(L)$ and hence $\mathcal{F}_r(L_1) \sigma L_2 \subseteq \mathcal{F}_r(L)$. The basis step, $i=0$, is true by hypothesis. Now assume that we have shown the inclusion to hold for $i \geq 0$. Then

$$\mathcal{A}_{i+1} \sigma L_2 = (\operatorname{REGL} \sigma \mathcal{M}_r(\mathcal{A}_i)) \sigma L_2 = \operatorname{REGL} \sigma (\mathcal{M}_r(\mathcal{A}_i) \sigma L_2),$$

$$\subseteq \operatorname{REGL} \sigma (\mathcal{M}_r(\mathcal{A}_i \sigma L_2)) \text{ by lemma 4.1,}$$

$$\subseteq \operatorname{REGL} \sigma \mathcal{F}_r(L) \text{ by the induction hypothesis,}$$

$$\subseteq \mathcal{F}_r(L).$$

If we take $L_1 = \operatorname{REGL}$ and $L_2 = \mathcal{F}_r(L)$, we can conclude that $\mathcal{F}_r(L)$ is closed under substitution into $\mathcal{F}_r(\operatorname{REGL})$. 

**Corollary:** $\mathcal{F}_r(\operatorname{REGL})$ is closed under substitution.

Now we wish to show that theorem 4.1 is the best result of its kind possible in the sense that, for full semiAFLs, $\mathcal{F}_r(L)$ is closed under substitution into $L_1$ if and only if either $L_1$ is contained in $\mathcal{F}_r(\operatorname{REGL})$ or $L$ is itself already an AFL closed under homomorphic replication and under substitution into $L_1$. Our approach is to establish a dichotomizing syntactic lemma which says that, if a particular type of substitution is in $\mathcal{F}_r(L)$, either one language is in $\mathcal{F}_r(\operatorname{REGL})$ or another is in $L \vee L^\sigma$. This requires some way of explicitly expressing members of $\mathcal{F}_r(L)$ in terms of members of $L$.

Unfortunately, we do not know of an attractive characterization of $\mathcal{F}_r(L)$. One can use a complex expression scheme generalizing the $\langle \rho, h_1, \ldots, h_n \rangle(L)$ formulation for $\mathcal{F}_r(L)$ or specialized machine formulations — for example, a nested restriction on the finite visit $L$-based automata [19] or adding to the finite reversal $L$-based automata a finite number of single reversal tapes in a nested fashion. We shall define a family of special language operators $\nu_{i,s}$ such that all members of $\mathcal{F}_r(L)$ can be obtained as $\nu$-transducer images of $\nu_{i,s}(L)$ for $L$ in $L$. The idea is to use brackets to nest alternate replications and applications of Kleene $+$. 

**Definition:** Let $L \subseteq \Sigma^+$ be a language and let $[1,1], \ldots, [k,k]$ be $2k$ reserved symbols. For $s \geq 1$, let $\nu_{0,s}(L) = L$ and for $i \geq 0$, let

$$\nu'_{i+1,s}(L) = [\{i+1 \omega_{i+1} w^R_{i+1}\}]^{i+1} w \text{ in } \nu_{i,s}(L)$$

and

$$\nu_{i+1,s}(L) = (\nu'_{i+1,s}(L))^+.$$

**Lemma 4.3:** Let $L$ be a full semiAFL. Let $L_0 = L = \mathcal{A}_0$ and for $i \geq 0$ let

$$L_{i+1} = \mathcal{F}_r(\mathcal{M}_r(L_i))$$

vol. 12, n° 3, 1978
and \( \mathcal{G}_{i+1} = \hat{\mathcal{H}}(\{ v_{i+1,s}(L) \mid s \geq 1, L \in \mathcal{L} \}) \).

The for all \( i \geq 0 \),
\[ \mathcal{L}_i = \mathcal{G}_i. \]

Proof: The proof proceeds by induction on \( i \). The basis step \( i = 0 \) is obvious.
Assume that the lemma is true for some \( i \geq 0 \). Clearly \( v_{i+1,s}(L) \) is in \( \hat{\mathcal{H}}(v_{i,s}(L)) \)
and \( v_{i+1,s}(L) \) is in \( \hat{\mathcal{F}}(v_{i+1,s}(L)) \). By the induction hypothesis, \( v_{i,s}(L) \) is in \( \mathcal{L}_i \) for
\( L \in \mathcal{L} \), and hence \( v_{i+1,s}(L) \) is in \( \mathcal{L}_{i+1} \), whence \( \mathcal{G}_{i+1} \subseteq \mathcal{L}_{i+1} \).

The outer brackets in \( v_{i+1,s}(L) \) serve as endmarkers, so \( \hat{\mathcal{F}}(v_{i+1,s}(L)) = \hat{\mathcal{H}}(v_{i+1,s}(L)) \) for any \( L \) [12]. Hence to show that \( \mathcal{L}_{i+1} \subseteq \mathcal{G}_{i+1} \), it
suffices to show that any language in \( \hat{\mathcal{F}}(\mathcal{L}_i) \) can be expressed as an \( a \)-transducer
mapping of \( v_{i+1,s}(L) \) for some \( L \in \mathcal{L} \) and \( s \geq 1 \).

Consider \( \mathcal{M}_r(\mathcal{L}_i) \). It suffices to consider a language of the form
\( L_1 = \langle \rho, h_1, \ldots, h_{2n} \rangle \) for \( L_2 \) in \( \mathcal{L}_i \), the \( h_j \) non-length-increasing homomorphisms and \( \rho(j) = 1 \) if and only if \( j \) is odd [17, 19, 21]. By the induction hypothesis,
\( L_2 = M_1(v_{i,s}(L_3)) \) for \( L_3 \) in \( \mathcal{L} \), \( s \geq 1 \) and an \( a \)-transducer
\( M_1 = (K_1, \Sigma_i, \Delta, H_1, q_0, F_1) \) where \( \Sigma_i = \Sigma \cup \{ l_j \mid 1 \leq j \leq i \} \) for a finite
vocabulary \( \Sigma \) not including any bracket symbol. We can assume that \( n \geq s \) (by adding, if needed, new \( \{w_1 \ldots w_s\} \) and new homomorphisms which only erase) and that
\( M_1 \) is 1-bounded [7, 12], and \( \# F_1 = 1 \). Further, since words in \( v_{i,s}(L_3) \) contain at
most \( 4(2^2 - 1) \) bracket symbols in a row, by using the states to rearrange transitions we can assume that \( M_1 \) gives empty output on bracket symbols (i. e.,
\( H_1 \) contains no transitions \( (p, [j, w, q) \) or \( (p, [j, w, q) \) with \( w \neq e) \), has no \( e \) input rules \( ((p, e, w, q) \) initially, terminally or in between bracket symbols and is
deterministic on bracket symbols in the sense that for \( b \) in \( \Sigma_i \) and \( q \) in \( K_1 \),
there is at most one \( p \) in \( K_1 \) with \( (q, b, e, p) \) in \( H_1 \) and at most one \( p' \) with
\( (p', b, e, q) \) in \( H_1 \). We shall alter \( L_2 \) and \( M_1 \) to obtain an \( a \)-transducer
deterministic everywhere, so that we can be sure that each \( h_j \) is applied to the
same \( w \).

We want to construct an \( a \)-transducer \( M_2 \) such that
\[ L_1 = \langle \rho, h_1, \ldots, h_{2n} \rangle (M_1(v_{i,s}(L_3))) = M_2(v_{i+1,n}(L)) \]
for some language \( L \) in \( \mathcal{L} \). We need some auxiliary definitions. Let \( \pi_i \) be the
projection on the \( i \)th coordinate of a tuple (i. e., \( \pi_i(a_1, \ldots, a_m) = a_i \) for \( m \geq i \)).
For \( a \) in \( \Sigma \cup \{ e \} \), let \( \Gamma_a \) be the collection of all possible \((2s)^i\)-tuples of members
of \( H_a = \{ (q, a, u, q') \mid (q, a, u, q') \} \) in \( H_1 \}; for \( a = e \) we assume that \( H_1 \) contains all
quadruples \( (q, e, e, q) \) for \( q \) in \( K_1 \).

Let \( g \) be the homomorphism defined by \( g(\gamma) = a \) for \( \gamma \) in \( \Gamma_a, a \) in \( \Sigma \cup \{ e \} \). Let
\( L = g^{-1}(v_{i,s}(L_3)) \). Clearly \( L \) is in \( \mathcal{L} \). What we have done is to encode each symbol of a

R.A.I.R.O. Informatique théorique/Theoretical Computer Science
word in $L_3$ by some guess as to the quadruples applied by $M_1$ in each of its $(2s)^t$
visits to that symbol during a successful computation of $M_1$ on that word; some
choices may be incorrect and will cause $M_2$ to block. Our special assumptions on
$M_1$ ensure that we do not have to specify $M_1$'s actions on brackets and so
brackets can remain brackets. The new $a$-transducer $M_2$ will act on $\nu_{i+1, n}(L)$.

The $a$-transducer $M_2$ will have state set $K_1 \times (\{1, 2, \ldots, 2n\})^{i+1}$. In a state
$(q_1, j_1, \ldots, j_i, j_{i+1})$, $q$ indicates the state of $M_1$ being simulated and $j_{i+1}$ the
replicate $h_{j_{i+1}}(w)$ currently under construction. The integer $j_i - 1$ is, roughly
speaking, the number of $[\ldots]$ subwords read since the last occurrence of $t_{i+1}$
or a reset $(q, e, e, q)$ (corresponding to an application of Kleene $+$), and
$(j_1, \ldots, j_i)$ gives base 2 $s$ the visit of $M_1$ to the symbol in $L_3$ currently simulated.
Machine $M_2$ must reconstruct $w$ 2$n$ times and output $h_{j_{i+1}}(w)$ or $(h_{j_{i+1}}(w))^R$.
Further complications occur because if $n \neq s$ there are “dummy” subwords to
skip and for the reversed replicates computations of $M_1$ must be simulated
backwards.

We define the transition set of $M_2$, $H_2$, as follows, for $M_2$ acting on input $a$:

1. Transitions to simulate $M_1$ on subwords of $a$ which are in $L_3$.

   Let $1 \leq j_i \leq 2s$ for $1 \leq i \leq t$, let $d$ be in $\Gamma_a$ for $a$ in $\Sigma \cup \{e\}$ and let
   \[
   r = 1 + \sum_{i=1}^{t} (j_i - 1)^{i-1}, \gamma = \pi_r(d) \text{ and } 1 \leq j_{i+1} \leq 2n. \]
   The integer $r$ indicates the visit of $M_1$ to the symbol a encoded by $d$, while $j_{i+1}$ indicates the current replicate simulated. For $j_{i+1}$ odd, the replicate is of the form $h_{j_{i+1}}(w)$, so if $M_2$ is in the state indicated by the $r$th coordinate of $d$, it outputs $h_{j_{i+1}}(\pi_3(\gamma))$ and changes to the indicated next state. For $j_{i+1}$ even, the replicate is of the form $h_{j_{i+1}}(w^R)$, so the actions of $M_1$ must be reversed. If $a = e$, $M_2$ simulates either a “real” action of $M_1$ on $e$ input or else a dummy $(q, e, e, q)$ transition:

   (A) For $j_{i+1}$ odd, $H_2$ contains
   \[
   ((\pi_1(\gamma), j_1, \ldots, j_{i+1}), d, h_{j_{i+1}}(\pi_3(\gamma)), (\pi_4(\gamma), j_1, \ldots, j_{i+1})).
   \]

   (B) For $j_{i+1}$ even, $H_2$ contains
   \[
   ((\pi_4(\gamma), j_1, \ldots, j_{i+1}), d, h_{j_{i+1}}(\pi_3(\gamma)), (\pi_1(\gamma), j_1, \ldots, j_{i+1})).
   \]

   If $n \neq s$, there are extra replicates of $w$ in $L$ which do not correspond to actions
of $M_1$ on $g(w)$ in $L_3$. Hence $M_2$ must skip such occurrences, using the next set of rules.

2. Transitions to skip extra subwords.

   If $j_i$ is greater than $2s$ for any $t \neq i + 1$, then $H_2$ contains for all $q$ in $K$, $a$ in
$\Sigma \cup \{e\}$, $d$ in $\Gamma_a$, the transition
   \[
   ((q, j_1, \ldots, j_{i+1}), d, e, (q, j_1, \ldots, j_{i+1})).
   \]
Now we define the transitions for reading brackets. Initially, \( M_2 \) must read a bracket sequence from \([i+1]\) to \([1]\) and simulate the initial transitions of \( M_1 \) while, at the end, \( M_2 \) is reading a reversed word and so simulates the initial transitions of \( M_1 \) in reverse order.

3. Initial and Final Transitions. Let

\[ w = ([j]^2 \ldots ([1]^2)^3 \quad \text{and suppose} \quad (q_0, w, e) \xrightarrow{*_{M_1}} (q, e, e). \]

Then \( H_2 \) contains the transitions

\[ ((q_0, 1, \ldots, 1), [i+1] \ [i+1] w, e, (q, 1, \ldots, 1)) \]

and

\[ ((q, 2n, \ldots, 2n), ([i+1])^2 (j)^2 \ldots ([1]^2)^3, e, (q_0, 2n, \ldots, 2n)). \]

We must also reinitialize the simulation of \( M_1 \) whenever \( M_2 \) passes from the \( j_{i+1} \)-th replicate to the next one, marked by reading the bracket sequence \([i+1]\) \([i+1]\). This is handled by the next sequence of rules.

4. Transitions to reinitialize the simulation of \( M_1 \). Let

\[ w_1 = ([1]^2 \ldots ([j]^2)^3 \quad \text{and} \quad w_2 = ([j]^2 \ldots ([1]^2)^3) \]

and suppose \((p, w_1, e) \xrightarrow{*_{M_1}} (f, e, e)\) for \( f \) in \( F \) and \((q_0, w_2, e) \xrightarrow{*_{M_1}} (q, e, e)\).

Let \( 1 \leq j_{i+1} \leq 2n - 1 \):

(A) If \( j_{i+1} \) is odd, then \( H_2 \) contains

\[ ((p, 2n, \ldots, 2n, j_{i+1}), w_1 [2n, e, (p, 2n, \ldots, 2n, j_{i+1} + 1)). \]

(B) If \( j_{i+1} \) is even, then \( H_2 \) contains

\[ ((q, 1, \ldots, 1, j_{i+1}), w_1 [i+1] w_2, e, (q, 1, \ldots, 1, j_{i+1} + 1)). \]

Finally, we need the transitions to simulate the action of \( M_1 \) on strings of bracket symbols which are not initial or final. These strings are of two kinds, marking transfer from one subword \( y \) of \( \alpha \) in some \( v_{i, n}(L_3) \) either into another subword of \( \alpha \) in \( v_{i, n}(L_3) \) (corresponding to an application of Kleene +) or into a replicate of \( y \). The first kind [represented in rule set (5) below] resets all \( j_m \) for \( m \leq t \) while the second kind [rule set (6) below] also increases \( j_{i+1} \).

5. Transitions to simulate \( M_1 \) on bracket strings within \( v_{i, n}(L_3) \).

Let \( 1 \leq t \leq i \), and let \( 1 \leq j_m \leq 2n \) for \( t + 1 \leq m \leq 2n \).

Let \( w = ([j]^2 \ldots ([j]^2)^3 ([l]^2 \ldots ([l]^2)^3 \text{and suppose that} \]

\[ (p, w, e) \xrightarrow{*_{M_1}} (q, e, e). \]
(A) For \( j_{i+1} \) odd, \( H_2 \) contains
\[
((p, 2n, \ldots, 2n, j_{t+1}, \ldots, j_{i+1}), w, e, (q, 1, \ldots, 1, j_{t+1}, \ldots, j_{i+1})).
\]
and
(B) For \( j_{i+1} \) even, \( H_2 \) contains
\[
((q, 1, \ldots, 1, j_{t+1}, \ldots, j_{i+1}), w, e, (p, 2n, \ldots, 2n, j_{t+1}, \ldots, j_{i+1})).
\]

(6) Transitions to simulate \( M_1 \) on bracket strings within \( \nu_{i+1, n}(L_3) \). Let \( 0 \leq t \leq i - 1 \) and \( w = (1)^2 \ldots (1)^2 j_{i+1} \). Suppose that
\[
(p, w, e) \xrightarrow{M_1} (q, e)
\]
and \( 1 \leq j_{i+1} \leq 2n - 1 \):

(A) For \( j_{i+1} \) odd, \( H_2 \) contains
\[
((p, 2n, \ldots, 2n, j_{t+1}, \ldots, j_{i+1}), w, e, (q, 1, \ldots, 1, j_{t+1} + 1, \ldots, j_{i+1})).
\]
(B) For \( j_{i+1} \) even, \( H_2 \) contains
\[
((q, 1, \ldots, 1, j_{t+1} + 1, j_{t+2}, \ldots, j_{i+1}), w, e, (p, 2n, \ldots, 2n, j_{t+1}, \ldots, j_{i+1})).
\]

This completes the construction. It can be shown that \( L_1 = M_2 (\nu_{i+1, s}(L)) \). □

Remark: Theorem 3.2 of [13] asserts that if \( \mathcal{L} \) is a full principal AFL, then \( \mathcal{F}(\mathcal{L}) \) is a full principal AFL. Lemma 4.3 can be considered a generalization, saying that a family of generators of \( \mathcal{L} \) correspond in a uniform way to a family of generators of \( \mathcal{F},(\mathcal{L}) \). The extra complication in the proof is needed to have one set of brackets suffice, and use languages in \( \mathcal{L} \) as building blocks.

Now we establish our lemma governing the existence of certain types of substitution languages in \( \mathcal{F},(\mathcal{L}) \).

Definition: For languages \( L_1 \) and \( L_2 \), \( L_1 \subseteq \Sigma_1^* \), let
\[
\tau(L_1, L_2) = \{ a_1 w_1 \ldots a_n w_n | a_i \in \Sigma_1 \text{ and } w_i \in L_2 \}
\]
\[
\cup \{ e | \text{if } e \text{ is in } L_1 \}.
\]

Clearly, \( \tau(L_1, L_2) = \tau(L_1) \) for the substitution \( \tau(a) = a L_2 \).

Lemma 4.4: Let \( L_1 \subseteq \Sigma_1^+, L_2 \subseteq \Sigma_2^+, \Sigma_1 \cap \Sigma_2 = / \emptyset \). Let \( \mathcal{L} \) be a full semiAFL. If \( \tau(L_1, L_2) \) is in \( \mathcal{F},(\mathcal{L}) \), either \( L_1 \) is in \( \mathcal{F},(\text{REGL}) \) or \( L_2 \) is in \( \mathcal{L} \setminus \mathcal{L}^R \).

Proof: Let \( L = \tau(L_1, L_2) \) be in \( \mathcal{F},(\mathcal{L}) \). By Lemma 4.4, there are \( k, s \geq 1 \), an \( a \)-transducer \( M \) and a language \( \overline{L} \) in \( \mathcal{L} \) such that \( L = M(\nu_{k,s}(\overline{L})) \).

vol. 12, n° 3, 1978
We shall now define from \( M \) and \( L \) two languages \( L_1 \) and \( L_2 \) such that \( L_1 \) is in \( \mathcal{F}_2(\text{REGL}) \), \( L_2 \) is in \( \mathcal{L} \lor \mathcal{L}^R \), and \( L_i \subseteq L_i \), \( i = 1, 2 \). Then we shall show that for words of the form \( \mu(a_1 \ldots a_n, w) = a_1 \ldots a_n w \) in \( L \) with \( n \geq 2 \), each \( a_i \) in \( \Sigma \) and \( w \) in \( L_2 \), either \( a_1 \ldots a_n \) is in \( L_1 \) or \( w \) is in \( L_2 \). Finally we shall be able to conclude that either \( L_1 = \overline{L_1} \) or \( L_2 = \overline{L_2} \).

Let \( \overline{L} \subseteq \Delta^* \) and let \( \Delta_k = \Delta \cup \{ [i,j] | 1 \leq j \leq k \} \). There are non-length-increasing homomorphisms \( g \) and \( h \) and a regular set \( R \) such that \( L = g(h^{-1}(v_{k,s}(L)) \cap R) \) \([11, 12]\). Let \( h : \Gamma^* \to \Delta_k^* \). Let

\[
B_1 = \{ b \in \Gamma | h(b) = [1] \} \quad \text{and} \quad B_2 = \{ b \in \Delta | h(b) = 1 \}.
\]

We can assume that \( B_1 \neq \emptyset \neq B_2 \) or else \( L \) would be empty. Since \( R \) is regular, there is a congruence relation on \( \Gamma^* \) with congruence classes \( C_1, \ldots, C_n \) such that \( R \) is the union of some of the \( C_i \)\([28]\). Let \( A_{i,j} = C_i \cap (C_j)^R \). For \( t \) in \( \{ 1, R \} \), say that \( (i, j, y) \) has property \( (\ast, t) \) if \( y^t \) is in \( h^{-1}(L) \cap A_{i,j} \) and there are \( u, v \) in \( \Gamma^* \), \( b_1 \) in \( B_1 \), \( b_2 \) in \( B_2 \) such that \( w = ub_1 y b_2 v \) is in \( R \) and \( h(w) = h(u)[1, h(y)] h(v) \) is in \( v_{k,s}(L) \). Suppose \( (i, j, y) \) has property \( (\ast, t) \) and \( x^t \) is in \( h^{-1}(L) \cap A_{i,j} \). Form \( w' \) from \( w \) by replacing all occurrences of \( b_1' y^t b_2' \), \( \alpha \) in \( \{ 1, R \} \) and \( b_1' \) in \( B_1 \), \( b_2' \) in \( B_2 \). Since \( h \) is non-length-increasing and \( h(x^t) \) is in \( \overline{L} \), \( h(w') \) is still in \( v_{k,s}(L) \); since \( x \) is congruent to \( y \) and \( x^R \) to \( y^R \), \( w' \) is still in \( R \). Thus, if \( (i, j, y) \) has property \( (\ast, t) \), \( (i, j, x) \) has property \( (\ast, t) \) for all \( x^t \) in \( h^{-1}(L) \cap A_{i,j} \), so we say that \( (i, j) \) has \( (\ast, t) \) if any \( (i, j, y) \) does. The crucial point is that if \( (i, j) \) has \( (\ast, t) \), all words in \( g(h^{-1}(L^t) \cap A_{i,j}) \) actually appear as subwords of words in \( L \). Let \( M_1 \) be the a-transducer acting on \( (\Sigma_1 \cup \Sigma_2)^* \) such that \( M_1(w) = \{ y \in \Sigma_2^* | w = uayb \alpha, \alpha \text{ in } \Sigma_1 \} \). Let

\[
\overline{L_2} = \bigcup_{(i,j) \text{ has } (\ast, 1)} M_1(g(h^{-1}(L) \cap A_{i,j}))
\]

\[
\cup \bigcup_{(i,j) \text{ has } (\ast, R)} M_1(g(h^{-1}(L^R) \cap A_{i,j})�).
\]

Since words in \( g(h^{-1}(L^t) \cap A_{i,j}) \) are subwords of words of \( L \) when \( (i, j) \) has \( (\ast, t) \), \( L_2 \subseteq L_2 \). Clearly \( L_2 \) is in \( \mathcal{L} \lor \mathcal{L}^R \). Now we must define \( L_1 \).

Let \( f \) be the homomorphism defined by \( f(a) = a, a \in \Sigma_1 \) and \( f(a) = e, a \in \Sigma_2 \). For \( a \in \Sigma_1 \cup \{ e \} \), let \( a, i, j \) be a new symbol if there is a \( y \) in \( h^{-1}(L) \cap A_{i,j} \) such that \( f(g(y)) = a \), and let \( T \) be the set of all such symbols. Extend \( g \) and \( h \) by \( h((a, i, j)) = (a, i, j) \) and \( g((a, i, j)) = a \). Let \( M_1 \) be the a-transducer which nondeterministically either acts as the identity on \( \Gamma^* \) or replaces a word in \( A_{i,j} \).
by a symbol \((a, i, j)\) in \(T\), and let \(R_x = M_x \{R\}\). Let \(L_x = f(g(h^{-1}(v_{k, x}(T)) \cap R_1))\).

Clearly \(\overline{L}_1\) is in \(\mathcal{F}_r(\text{REGL})\) since \(T\) is finite and so regular. If \(w\) is in \(v_{k, x}(T)\), each symbol \((a, i, j)\) in \(w\) can be replaced by a word \(h(y)\) in \(A_{i, j}\) and \(f(g(y)) = f(g((a, i, j))) = a\) in such a way that the resulting word \(w'\) is in \(v_{k, x}(\overline{L})\). If \(x\) is in \(h^{-1}(w)\), we can replace \((a, i, j)\) in \(x\) by \(y\) as above so that the resulting word \(x'\) is in \(h^{-1}(w')\) and if \(x\) is in \(R_1\), \(x'\) is in \(R\). Thus \(g(x')\) is in \(L\) and \(f(g(x)) = f(g(x'))\) is in \(f(L) = L_1\), so \(f(g(x))\) is in \(L_1\). Hence \(\overline{L}_1 \subseteq L_1\).

Consider a word \(\mu(a_1 \ldots a_n, w)\) in \(L\). For some \(x\) in \(v_{k, x}(L)\), \(z\) in \(h^{-1}(x) \cap R\), \(\mu(a_1 \ldots a_n, w) = g(z)\). We can decompose \(x\) as

\[
x = u_1[y_1, 1] \ldots u_m[y_m, 1] u_{m+1}
\]

the \(y_i\) in \(\overline{L} \cup \overline{L}^R\) and the \(u_i\) strings of brackets, and \(z\) as \(z = v_1 z_1 \ldots v_m z_m v_{m+1}\).

If \(g(f(g(z_i))) \in \{ e \} \) for all \(i, 1 \leq i \leq m\), we say that \(a_1 \ldots a_n\) splits in \(\mu(a_1 \ldots a_n, w)\). In that case, each \(y_i\) could be replaced by a symbol of \(T\), so \(a_1 \ldots a_n\) is in \(\overline{L}\). If \(n \geq 2\) and \(a_1 \ldots a_n\) does not split in \(\mu(a_1 \ldots a_n, w)\), then for some \(l, a_1 w_{a_{l+1}}\) is a subword of some \(g(z_k)\). For some \(i, j, t, (i, j, z_i^k)\) has property \((\star, t)\) and weAf so \(L = L_2\).

Lemma 4.4 has the following useful consequence.

**Lemma 4.5:** Let \(\mathcal{L}_x, \mathcal{L}_1\) and \(\mathcal{L}_2\) be full semiAFLs such that \(\mathcal{L}_1 \in \mathcal{L}_2\) is contained in \(\mathcal{F}_r(\mathcal{L})\). If \(\mathcal{L}_1\) is not contained in \(\mathcal{F}_r(\text{REGL})\), then \(\mathcal{L}_2\) is contained in \(\mathcal{L} \cup \mathcal{L}^R\) and if \(\mathcal{L}_2\) is not contained in \(\mathcal{L} \cup \mathcal{L}^R\), \(\mathcal{L}_1\) is contained in \(\mathcal{F}_r(\text{REGL})\).

**Proof:** Suppose \(\mathcal{L}_1\) is not contained in \(\mathcal{F}_r(\text{REGL})\). Then there is a language \(L_1\) in \(\mathcal{L}_1\) but not in \(\mathcal{F}_r(\text{REGL})\) such that \(L_1 \subseteq \Sigma_1^*\) for some vocabulary \(\Sigma_1\). Consider an arbitrary language \(L_2\) in \(\mathcal{L}_2\). Since full semiAFLs are closed under renaming and addition and deletion of the empty word, it suffices to consider \(L_2 \subseteq \Sigma_2^*\) with \(\Sigma_1 \cap \Sigma_2 = \emptyset\). Thus \(\tau(L_1, L_2)\) is in \(\mathcal{F}_r(\mathcal{L})\) so by lemma 4.4, \(L_2\) is in \(\mathcal{L} \cup \mathcal{L}^R\). The argument if \(\mathcal{L}_2\) is not contained in \(\mathcal{L} \cup \mathcal{L}^R\) is similar.

Now we are ready to draw some conclusions about the substitution of \(\mathcal{F}_r(\mathcal{L})\) into or by another full semiAFL.

**Theorem 4.6:** Let \(\mathcal{L}\) and \(\mathcal{L}_1\) be full semiAFLs:

1. \(\mathcal{F}_r(\mathcal{L})\) is closed under substitution into \(\mathcal{L}_1\) if and only if either \(\mathcal{L}_1\) is in \(\mathcal{L}_2\) and \(\mathcal{L}_2\) is in \(\mathcal{L} \cup \mathcal{L}^R\).

vol. 12, n° 3, 1978
(2) \( \mathcal{T}_r(L) \) is closed under substitution by \( L_1 \) if and only if either both \( L \) and \( L_1 \) are contained in \( \mathcal{T}_r(\text{REGL}) \), or \( \mathcal{M}_\sigma(L_1) \) is contained either in \( L \) or in \( L^R \) and \( L \cap L_1 \) is contained in \( \mathcal{T}_r(L) \).

**Proof:** First observe that if \( L_1 \) is contained in \( \mathcal{T}_r(\text{REGL}) \), then \( \mathcal{T}_r(L) \) is closed under substitution into \( L_1 \) by theorem 4.2. Next, assume that \( \mathcal{T}_r(L) \) is closed under substitution into \( L_1 \) and that \( L_1 \) is not contained in \( \mathcal{T}_r(\text{REGL}) \). Now \( L_1 \cap \mathcal{T}_r(L) \subseteq \mathcal{T}_r(L) \). By lemma 4.5, \( \mathcal{T}_r(L) \) is contained in \( L \vee L^R \) and hence \( \mathcal{T}_r(L) = L \vee L^R \). If \( L \neq L^R \), then \( L \vee L^R \) is not an AFL [15]. Thus \( \mathcal{T}_r(L) = L = L^R \) and \( L \) is closed under substitution into \( L_1 \). This establishes (1).

If \( L \) and \( L_1 \) are both contained in \( \mathcal{T}_r(\text{REGL}) \), then

\[
\mathcal{T}_r(L) = \mathcal{T}_r(\text{REGL}) = \mathcal{T}_r(L_1),
\]

so \( \mathcal{T}_r(L) \) is closed under substitution by \( L_1 \). On the other hand, if \( L \cap L_1 \subseteq \mathcal{T}_r(L) \), then \( \mathcal{T}_r(L) \) is closed under substitution by \( L_1 \).

Now suppose that \( \mathcal{T}_r(L) \) is closed under substitution by \( L_1 \). Then \( L \cap L_1 \subseteq \mathcal{T}_r(L) \), and \( \mathcal{T}_r(L) \cap \mathcal{M}_\sigma(L_1) \subseteq \mathcal{T}_r(L) \). If \( L \) is not contained in \( \mathcal{T}_r(\text{REGL}) \) then, by lemma 4.5, \( \mathcal{M}_\sigma(L_1) \) is contained in \( L \vee L^R \). Hence \( \mathcal{M}_\sigma(L_1) \) must be contained in either \( L \) or \( L^R \) [15]. Finally, suppose that \( L_1 \) is not contained in \( \mathcal{T}_r(\text{REGL}) \). Since \( \mathcal{M}_\sigma(L_1) = \mathcal{M}_\sigma(L_1) \cap \mathcal{M}_\sigma(L_1) \) is contained in \( \mathcal{T}_r(L) \), by lemma 4.5, \( \mathcal{M}_\sigma(L_1) \) is contained in \( L \vee L^R \) and hence in either \( L \) or \( L^R \). This establishes (2).

A consequence of theorem 4.6 worthy of special attention is the following. It is immediate from theorem 4.6(1), taking \( L_1 = L \).

**Theorem 4.7:** Let \( L \) be a full semiAFL. Then \( \mathcal{T}_r(L) \) is closed under substitution if and only if either \( L \) is contained in \( \mathcal{T}_r(\text{REGL}) \), so \( \mathcal{T}_r(L) = \mathcal{T}_r(\text{REGL}) \) or \( L \) is closed under both substitution and homomorphic replication.

5. If \( L \) is a full semiAFL, then the closure of \( L \) under deterministic two-way finite state transductions is \( \text{FINITE.VISIT}(L) \), the family of languages accepted by the one-way \( L \)-based preset Turing machines of [19] restricted to a finite number of visits per working tape square. (A one-way \( L \)-based preset Turing machine has a one-way input tape and one working tape preset to words in \( L \) for some \( L \) in \( L \).) If we restrict the number of reversals rather than visits of
\(\mathcal{L}\)-based preset Turing machines, we obtain \(\text{FINITE-REVERSAL}(\mathcal{L})\) which is equal to \(\hat{\mathcal{M}}_r(\mathcal{L})\) and also to the closure of \(\mathcal{L}\) under iterating controls on linear context-free grammars. For the special case \(\mathcal{L} = \text{REGL}\), \(\text{FINITE-REVERSAL}(\text{REGL})\) is the family of languages accepted by finite reversal checking automata, while \(\text{FINITE-VISIT}(\text{REGL})\) is the family of languages accepted by finite visit checking automata. If \(\mathcal{L}\) is a full AFL not closed under homomorphic replication, \(\text{FINITE-REVERSAL}(\mathcal{L})\) is not a full AFL while \(\text{FINITE-VISIT}(\mathcal{L})\) is an AFL [19]. Thus “most of the time”, \(\text{FINITE-VISIT}(\mathcal{L})\) property contains \(\text{FINITE-REVERSAL}(\mathcal{L})\), so visits are more powerful than reversals.

For any full AFL \(\mathcal{L}\), we have

\[
\text{Finite-Reversal}(\mathcal{L}) \subseteq \hat{\mathcal{F}}_r(\mathcal{L}) \subseteq \text{Finite-Visit}(\mathcal{L}).
\]

We shall now show that, if \(\mathcal{L}\) is closed under substitution but not homomorphic replication, these containments are always proper. Since \(\text{Finite-Visit}(\mathcal{L})\) is closed under substitution for any substitution closed full semiAFL \(\mathcal{L}\), this follows from theorem 4.7 when \(\mathcal{L}\) is not contained in \(\hat{\mathcal{F}}_r(\text{REGL})\).

It remains to show that \(\hat{\mathcal{F}}_r(\text{REGL})\) is properly contained in \(\text{Finite-Visit}(\text{REGL})\). We do so by showing that, although \(\text{Finite-Visit}(\text{REGL})\) does have stronger iterative properties than those established in [19], the iterative properties of \(\hat{\mathcal{F}}_r(\text{REGL})\) are still stronger.

First we show that if a generalized Ogden’s lemma [29] holds for a full semiAFL \(\mathcal{L}\), it does so for \(\hat{\mathcal{F}}_r(\mathcal{L})\). We call a language for which a generalized Ogden’s Lemma holds a strongly iterative language.

**Definition**: A language \(L\) is strongly \(k\)-iterative for \(k \geq 1\), if for each \(t \geq 1\) there are iteration numbers \(p, q\) such that if \(w\) is in \(L\), \(|w| \geq p\) and \(p\) or more positions of \(w\) are designated as distinguished, then we can factor \(w\) as

\[
w = u_1 v_1 \ldots u_k v_k u_{k+1},
\]

where:

1. For some \(i\), \(v_i\) contains at least \(t\) distinguished positions, \(u_i\) and \(u_{i+1}\) each contain at least one distinguished position and \(u_i v_i u_{i+1}\) contains at most \(q\) distinguished positions.
2. For each \(n \geq 0\), \(u_1 v_1^n \ldots u_k v_k^n u_{k+1}\) is in \(L\).

We call \(L\) strongly iterative if it is strongly \(k\)-iterative for some \(k \geq 1\). A family of languages is strongly iterative if each of its members is strongly iterative.

The property of being strongly iterative is preserved by homomorphic replication and by substitution. The next lemmas generalize results in [13, 17, 19, 24 and 25], and are similar to results in [35] for a slightly different property, “locally linear”

vol. 12, n° 3, 1978
LEMMA 5.1: If $\mathcal{L}$ is a strongly iterative full semiAFL, then $\mathcal{M}_r(\mathcal{L})$ is strongly iterative.

Proof: If $L_1$ is in $\mathcal{M}_r(\mathcal{L})$, we can express it as $L_1 = \langle p, h_1, \ldots, h_m \rangle (L)$ for $L$ in $\mathcal{L}$ and the $h_i$ non-length-increasing homomorphisms. Let $L$ be strongly $k$-iterative. We claim that $L_1$ is strongly $km$-iterative. For $t \geq 1$, let $p$, $q$ satisfy the definition of strongly $k$-iterative for $L$. Let $p_1 = mp$ and $q_1 = q$. Consider $w$ in $L_1$ with $|w| \geq p_1$. Then $w = (h_1(x))^{p(1)} \ldots (h_m(x))^{p(m)}$ for $x$ in $L$. Designate any $p_1$ or more positions in $w$ as distinguished. At least $p$ distinguished positions must lie in some $(h_j(x))^{p(j)}$ and, since $h_j$ is not length increasing, must correspond to at least $p$ positions in $x$ which we call distinguished positions in $x$. Thus there is a factorization $x = u_1 v_1 \ldots u_k v_k u_{k+1}$ satisfying (1) and (2) of the definition of strongly $k$-iterative. Then the factorization

$$w = (h_1(u_1) h_1(v_1) \ldots h_1(v_k) h_1(u_{k+1}))^{p(1)} \ldots (h_m(u_1) h_m(v_1) \ldots h_m(v_k) h_m(u_{k+1}))^{p(m)},$$

satisfies (1), (2) of the definition of strongly $km$-iterative with the $(h_j(x))^{p(j)}$ providing the $km$ iterative factors, since distinguished positions in $x$ and $h_j(x)$ are in one-one correspondence. □

LEMMA 5.2: If $\mathcal{L}_1$ and $\mathcal{L}_2$ are strongly iterative full semiAFLs, then $\mathcal{L}_1 \circ \mathcal{L}_2$ is strongly iterative.

Proof: Let $L_1 = \tau(L)$ for $L \subseteq \Sigma^*$, $L$ in $\mathcal{L}_1$ and each $\tau(a)$ in $\mathcal{L}_2$, $a$ in $\Sigma$. Let $L$ be strongly $k$-iterative and each $\tau(a)$ strongly $k_a$-iterative. We claim that $L_1$ is strongly $k_1$-iterative for $k_1 = \text{Max} \{ k \} \cup \{ k_a \mid a \in \Sigma \}$. For $t \geq 1$, let $p$, $q$ be the iteration numbers for $L$ and $p_a$, $q_a$ those for $\tau(a)$, $a$ in $\Sigma$. Let

$$p_1 = p \text{ Max} \{ \{ p_a \mid a \in \Sigma \} \}$$

and

$$q_1 = \text{Max} \{ \{ q(p_a - 1), q_a \mid a \in \Sigma \} \}.$$

Consider $w$ in $L$ with $|w| \geq p_1$ and call any $p_1$ or more positions in $w$ distinguished. We can write $w = w_1 \ldots w_m$, each $w_i$ in $\tau(a_i)$, $a_i$ in $\Sigma$ and $y = a_1 \ldots a_m$ in $L$. There are two cases. If some $w_i$ contains at least $p_a$ distinguished positions, we can use the factorization for $w_i$ inherited from $\tau(a_i)$ since we are free to substitute for $a_i$ any member of $\tau(a_i)$; if $k_a$ is less than $k_1$, we let the other $k_1 - k_a$ factors be the empty word. Suppose no $w_i$ contains $p_a$ distinguished positions. Thus there are at least $p$ subwords $w_i$ which contain at least 1 but no more than $p_a$ distinguished positions. If $w_i$ contains distinguished positions, let $a_i$ be distinguished in $y$. Hence we have a factorization

R.A.I.R.O. Informatique théorique/Theoretical Computer Science
y = u_1 v_1 \ldots u_k v_k u_{k+1} \text{ satisfying (1) and (2) of the definition of strongly } k\text{-iterative. Let}

\begin{equation}
w = \tau(u_1) \tau(v_1) \ldots \tau(u_k) \tau(v_k) \tau(u_{k+1})
\end{equation}

where if \( k > k, u_i = e \) for \( i \geq k + 2 \) and \( v_i = e \) for \( i \geq k + 1 \). Since \( u_1 v_1^* \ldots u_k v_k^* u_{k+1} \) is in \( L \) for each \( n \geq 0 \), \( \tau(u_1) (\tau(v_1))^n \ldots \tau(u_k) (\tau(v_k))^n \ldots \tau(u_{k+1}) \) is in \( L_1 \) for each \( n \geq 0 \). If \( u_i v_i u_{i+1} \) satisfies (1) of the definition for \( L \), then \( \tau(v_i) \) contains at least \( t \) distinguished positions, \( \tau(u_i) \) and \( \tau(u_{i+1}) \) contain distinguished positions and \( \tau(u_i v_i u_{i+1}) \) contains at most \( q \text{Max}(\{ p_a - 1 | a \text{ in } \Sigma \}) \) distinguished positions. \( \square \)

Since \( \mathcal{F}_r(L) \) is obtained from \( L \) by alternating closure under homomorphic replication and closure under substitution into regular sets and all regular sets are strongly 1-iterative, we have the next theorem.

**Theorem 5.3:** If \( L \) is a strongly iterative full semiAFL, so are \( \mathcal{F}_r(L) \) and \( \mathcal{A}_n(L) \).

Since every context-free language is strongly 2-iterative [29], we have the following corollary.

**Corollary:** \( \mathcal{F}_r(\text{REGL}) \) and \( \mathcal{F}_r(\text{CF}) \) are strongly iterative.

Showing that a particular member of \( \text{FINITE} \cdot \text{VISIT} \) (REGL) is not strongly iterative is long and tedious.

**Lemma 5.4:** Let \( L_1 = \{ a^n b^n \mid n \geq 1 \} \) and let \( L \) consist of all and only words of the form \( w_1 \$ w_2 \) where

\begin{align*}
w_1 &= dx_1 y_1 c y_1^R x_1^R d \ldots dx_m y_m c y_m^R x_m^R d \\
w_2 &= dy_1 x_2 c y_1^R x_2^R d \ldots dy_{m-1} x_m c y_{m-1}^R x_m^R d
\end{align*}

for \( m \geq 2 \), \( x_i, y_i \text{ in } L_1 \) for \( 1 \leq i \leq m \). Then \( L \) is not strongly iterative.

**Proof:** The idea is to find \( w \) for which the desired iterative factors exist but not as subwords of \( w \). Suppose that \( L \) were strongly \( k \)-iterative. Let \( t = 2 \) and let \( p, q \) be the required iteration numbers for \( L \).

Let \( w = w_1 \$ w_2 \) be in \( L \) where \( w_1 \) and \( w_2 \) are expressed as above, \( m = p + 1 \) and \( x_i \neq x_j \) and \( y_i \neq y_j \) for \( i \neq j \) and \( x_i \neq y_j \) for all \( i \). Call the second through the \((p+1)st\) \( d \)'s in \( w_1 \) distinguished. Then we have \( w_1 \$ w_2 = u_1 v_1 \ldots u_k v_k u_{k+1} \) and \( s, 1 \leq s \leq k \) such that \( v_s \) contains at least 2 distinguished positions. \( u_s \) and \( u_{s+1} \) each contain at least 1 distinguished position and \( w(n) = u_1 v_1^s \ldots u_k v_k^s u_{k+1} \) is in \( L \) for all \( n \geq 0 \). Thus \( v_s \) lies wholly in \( w_1 \) and does not contain the first two or the last two \( d \)'s in \( w_1 \).
Now $v_s$ cannot start and end with $d$ or else $w(2)$ would contain two $d$'s in a row. Thus $v_s = \beta \gamma d \alpha \beta$ for $\alpha \beta$ in $\{a, b, c\}^+$ and $w_1$ has a subword
\[ \ldots d\alpha_1 \beta \gamma d\alpha \beta_1 \ldots \]
with $\alpha_1 \beta$ and $\alpha \beta_1$ in
\[ L_2 = \{ a^n b^n a^n b^n \mid n_1, n_2 \geq 1 \} \].

Further, if $n_1, n_2$ are the exponents for $\alpha_1 \beta$ and $n_3, n_4$ for $\alpha \beta_1$, no two of the four integers are the same. Now $L$ contains $w(0) = u_1 u_2 \ldots u_s u_{s+1} \ldots u_{k+1}$ which has a subword
\[ \ldots d\alpha_1 \beta d \ldots \]
and
\[ w(2) = u_1 v_1^2 \ldots u_s v_s^2 u_{s+1} \ldots u_{k+1} \]
which has a subword
\[ \ldots d\alpha_1 \beta d\gamma d\alpha \beta d \gamma d\alpha \beta_1 d \ldots \]
so $\alpha_1 \beta_1$ and $\alpha \beta$ are also in $L_2$. Either $\alpha$ or $\beta$ contains $c$; suppose it is $\beta$, then $\alpha = \alpha_1$. If $\alpha$ contains any $b$'s, this determines $n_1$ and so $n_3 = n_1$, a contradiction. Hence $\alpha$ is in $a^*$ and $\beta$ is a member of $L_2$ with $|\alpha|$'s removed from the left end.

A similar argument applies if $\alpha$ contains $c$.

Let $y \setminus z = y'$ if $z = yy'$. We can write
\[ v_s = (a^t \setminus x_i) y_i c y_i^R x_i^R d x_i+1 y_{i+1}^R c y_{i+1}^R x_{i+1}^R d \ldots d x_i+t, y_i+t, c y_i+t, x_i+t, d a^t \]
with $t \geq 1$. Write $w(n) = w_1(n) \# w_2(n)$.

Now $w_2$ has a subword
\[ \ldots d y_{i-1} x_i c x_i^{-1} y_i^{-1} d y_i x_i+1 c x_i+1 d y_{i+1} x_i+1 c x_{i+1} x_{i+1}^R y_{i+1} d \ldots \]
For $n \geq 2$, $w_1(n)$ has a subword
\[ d(x_i, y_i c y_i^R x_i^R d x_i+1 y_{i+1} c y_{i+1}^R x_{i+1}^R d \ldots d x_i+t, y_i+t, c y_i+t, x_i+t, d)^n \]
which by the definition of $L$ means that $w_2(n)$ has a subword
\[ d(y_i x_{i+1} c x_{i+1} y_{i+1}^R d \ldots d y_{i+t-1} x_{i+t} c x_{i+t} y_{i+t}^R d y_{i+t} x_i c x_i^R y_i^R d)^{n-1} \]
But since all the $x_i$ and $y_i$ in $w$ are distinct, $d y_{i+t} x_i c x_i^R y_i^R d$ cannot be a subword of $w$ and so this iterative factor in $w_2(n)$ cannot be obtained as a subword of $w$. This is a contradiction. Hence $L$ is not strongly iterative. □

R.A.I.R.O. Informatique théorique/Theoretical Computer Science
Corollary: There is a language $L$ in $\text{FINITE. VISIT (REGL)}$ which is not strongly iterative.

Proof: It remains to show that the language $L$ in lemma 5.4 is in $\text{FINITE. VISIT (REGL)}$. Clearly $L_1$ and so $(L_1 L_1 d)^+ S$ are in $\text{FINITE. VISIT (REGL)}$. A two-way deterministic finite state transducer acting on $(L_1 L_1 d)^+$. $S$ can interpret an input string $S x_1 y_1 d \ldots dx_m y_m d$ $S$ as giving the $x_i$ and $y_i$ for words in $L$ and first go right and then left over $x_i y_i$ to produce $dx_i y_i c y_i^R x_i^R d$, on reading the right hand $S$ output $S$ and then return to the first $S$, now scanning $y_i dx_{i+1}$ to give $dy_i x_{i+1} c x_i^R y_i^R d_{i+1}$, and finally finish on again scanning the right hand $S$. Since $\text{FINITE. VISIT (REGL)}$ is closed under two-way deterministic finite state transducer mapping, it contains $L$. □

We can state the main result of this section.

Theorem 5.5: If $\mathcal{L}$ is a full AFL closed under substitution but not homomorphic replication

$$\mathcal{L} \subsetneq \text{FINITE. REVERSAL (}\mathcal{L}) \subsetneq \mathcal{F_r}(\mathcal{L}) \subsetneq \text{FINITE. VISIT (}\mathcal{L}).$$

References