INFORMATIQUE THÉORIQUE ET APPLICATIONS

S. A. GREIBACH
C. WRATHALL

Single-tape reset machines

Informatique théorique et applications , tome 20, n° 1 (1986), p. 55-77
<http://www.numdam.org/item?id=ITA_1986__20_1_55_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 20, n° 1, 1986, p. 55 à 77)

SINGLE-TAPE RESET MACHINES (*)

by S. A. GREIBACH (1) and C. WRATHALL (2)

Communicated by J. GALLIER

Abstract. — *A* reset tape *can be written on once from left to right and then reread once, from left to right. A* circular tape *can be written once from left to right and then reread from left to right an arbitrary number of times. A* writing circular tape *is a circular tape which can be overprinted but not extended. An on-line nondeterministic Turing machine with one writing circular tape can accept any recursively enumerable set. A complete diagram is given and established for proper inclusions and equivalences among all other classes of nondeterministic on-line machines with one reset or circular or writing circular tape with or without restriction to finite-delay and with or without reinitialization. These classes are incomparable to the class of linear context-free languages.*

Résumé. — *Une classe d'automates non-déterministes qui parcourent le ruban mémoire seulement de gauche à droite est considérée. Un automate à un « reset » peut écrire une fois, et ensuite relire le ruban une seule fois, sans modifier le contenu de sa mémoire. Un automate à « mémoire circulaire » peut relire plusieurs fois, également sans modifier le contenu de sa mémoire. Enfin, un automate à « mémoire circulaire réécrivable » peut modifier le contenu du ruban mémoire, mais non sa longueur; sans restriction, il peut reconnaître tout langage recursivement énumérable.*

Nous étudions ces automates avec ou sans la restriction de « délai fini », et avec ou sans « recommencement ». Les inclusions strictes et les équivalences entre ces familles de langages sont démontrées, et représentées par un diagramme complet. Chacune de ces classes (sauf la famille de langages recursivement énumérables) et la famille des langages linéaires sont incomparables.

1. INTRODUCTION

One of the long-standing themes of research in formal language theory is the study of the generative power of specific languages under specified sets of operations. Perhaps the richest source of interesting examples in this area has been models of computation obtained by placing restrictions on Turing machines. Restrictions have been placed on resources such as time and space, and also on the access of the machine to its data, that is, on the type of storage tape allowed. Examples of this type of restriction include pushdown

(*) Received August 1984, revised April 1985.
This research was supported in part by the National Science Foundation under Grants No. MCS78-04725, MCS77-11360 and MCS83-12472.
(1) Department of Computer Science, University of California at Los Angeles, Los Angeles, Ca. 90024, U.S.A.
(2) Department of Mathematics, University of California at Santa Barbara, Santa Barbara, Ca. 93106, U.S.A.

Informatique théorique et Applications/Theoretical Informatics and Applications
ISSN en cours 86/01 55 23/$ 4.30/

stores, finite-turn pushdown stores and stacks. In a previous paper [3], the concept of a *reset tape* was introduced. In this paper we study on-line nondeterministic machines with a one-way tape that is either a reset tape or a simple extension of a reset tape.

A machine can write only once (say from left to right) on a reset tape and then can reread the tape only once, also from left to right. A reset tape is thus similar to a single-turn pushdown store, but instead of changing direction, the head is reset to the beginning of the tape for the second sweep.

If we allow the working tape to be reread any finite number of times, that is, the working tape head to be reset as desired from the right end of the tape to the left end, we have a *circular tape,* for we can envision the right end of the tape to be pasted to the left end. Thus in a circular tape the string written on the tape during the first pass can be reread any number of times, always from left to right. If we now allow the symbols on the circular tape to be overprinted, we obtain a *writing circular tape;* the length of tape available in any computation is determined by the string written in the first pass and, although the contents of the tape can be changed, the tape can only be accessed by sweeping from left to right. The model can also be extended by allowing *reinitialization* of the tape, that is, the tape is instantaneously erased and the computation can continue with an empty tape of the same type.

Machines with multiple tapes of these types were examined in the previous paper [3]. Any such machine can be simulated nondeterministically without loss of time by one with three (but not two [9]) reset tapes. For fixed $k \geqq 1$, machines with a single circular tape that is restricted to k sweeps accept the class of languages that has been studied under the names "k-equal matrix languages" [13, 15] and "k-right linear simple matrix languages" [16], and as the class derived by certain homomorphic replications of regular sets [14].

In this paper, we study the inclusions among families of languages accepted by on-line nondeterministic machines with one of these tapes, with or without restrictions on the time, and with or without reinitialization. In all cases, acceptance is by both final state and "empty store"; that is, the machine must be in a final state and the head on the work tape must be at the right end of the string written on the tape.

Each of these types of tapes can be described by a language which captures the essence of the restriction on access involved and which moreover is in a certain sense a generator of all the languages definable by that type of one-way machine. The language corresponding to a reset tape is

$$\text{COPY} = \{ ww : w \in \{ a, b \}^* \};$$

to a circular tape,

$$*\text{COPY} = \{(wc)^k : k \geqq 0,\ w \in \{a, b\}^*\};$$

and to a writing circular tape,

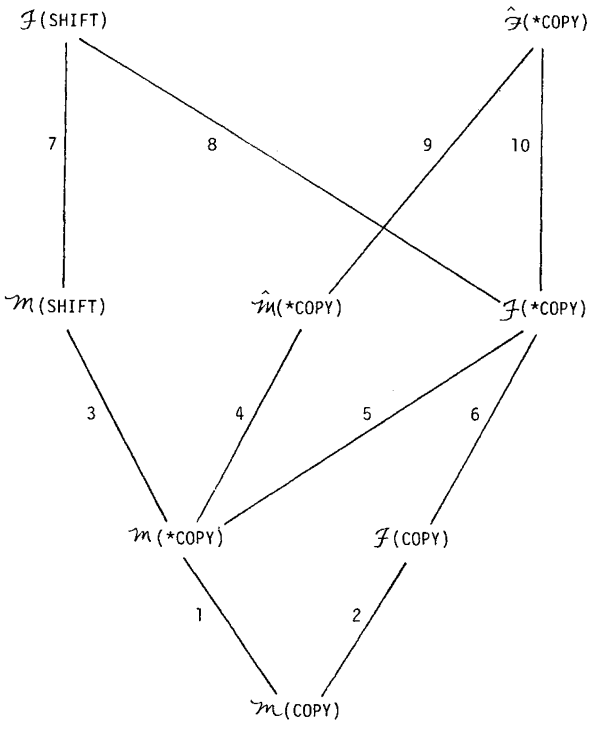
$$\text{SHIFT} = \{e\} \cup \{\langle x_1, x_2\rangle c \langle x_2, x_3\rangle c \ldots \langle x_m, x_{m+1}\rangle c : m \geqq 1,\ x_i \in \{a, b\}^*,\ |x_1| = \ldots = |x_m|\},$$

where $\langle x, y\rangle$ is a parallel encoding of equal-length strings. The class of languages accepted by nondeterministic finite-delay machines with a single tape of one of the three types is exactly the semiAFL (family of languages closed under nonerasing finite-state transductions) generated by the language corresponding to the tape. If the time restriction is dropped one has the full semiAFL (semiAFL closed under arbitrary homomorphism) generated by the appropriate language. Allowing the machine to reinitialize the tape when it reaches the right end gives rise to the AFL (semiAFL closed under Kleene*) or full AFL (full semiAFL closed under Kleene*) generated by the corresponding language.

In this paper we describe and study the families of languages accepted by these various types of one-way nondeterministic machines in terms of the semiAFL and AFL operations and the languages involved. For example, we use $\mathscr{M}$(COPY) for the family of languages accepted by one-way nondeterministic machines with a single reset tape and finite delay. Section 2 reviews the basic notation and definitions used.

The inclusions and noninclusions among these classes of languages are examined in section 3. For machines with one reset tape, the finite-delay restriction causes no loss of power (theorem 3.1 *a*). An on-line nondeterministic Turing machine with one writing circular tape can accept any recursively enumerable set (theorem 3.1 *b*). Otherwise, no inclusions hold except as implied by the definitions of the machines involved, and all such inclusions are proper. A complete diagram (*fig.* 1) is given and established for proper inclusions and equivalences among all these classes. For example, the semiAFLs form a strict hierarchy as the type of storage is extended from a reset tape to a circular tape to a writing circular tape, and, except as noted above, the restriction to finite delay is a restriction in power and the extension to reinitialization is an extension.

The action of a single-turn pushdown store can be described by the language $\text{PAL} = \{wcw^R : w \in \{a, b\}^*\}$ consisting of center-marked palindromes; that is, the class of linear context-free languages is the (full) semiAFL generated by $\text{PAL} \cup \{e\}$. The language COPY is not a (linear) context-free



language, nor can PAL be accepted by a machine with a single-reset tape. Section 4 contains a proof that PAL cannot be recognized either by a machine with a circular tape or by a finite-delay machine with a single writing circular tape.

2. PRELIMINARIES

This section contains a review of some basic definitions and notation.

For an alphabet (set of symbols) Σ, Σ^* denotes the free monoid generated by Σ, with identity e (the empty word); $\Sigma^+ = \Sigma^* - \{e\}$. The *length* of a string $x \in \Sigma^*$ is the number of occurrences of symbols in x and is denoted by $|x|$. The *reversal* of $x = a_1 \ldots a_n (a_i \in \Sigma)$ is $x^R = a_n \ldots a_1$. For strings w, x, y and z, if $w = xy$ then x is a *prefix* of w and y is a *suffix* of w, and if $w = xyz$ then y is a *factor* of w.

A language is a subset of Σ^* for some finite alphabet Σ. In addition to union ($\cup$) and intersection ($\cap$) of languages, we make use of the following operations:

(*a*) concatenation: $L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$;

(*b*) Kleene*: $L^* = \{ x_1 \ldots x_n : n \geqq 0, \text{ each } x_i \in L \}$;

(*c*) homomorphic image: $h(L_1) = \{ h(x) : x \in L_1 \}$ and inverse homomorphic image $h^{-1}(L_2) = \{ y \in \Sigma^* : h(y) \in L_2 \}$ where $h : \Sigma^* \to \Delta^*$ is a (monoid) homomorphism, $L_1 \subseteq \Sigma^*$ and $L_2 \subseteq \Delta^*$.

In (*c*), if $h(a) \neq e$ for each $a \in \Sigma$ then h is a *nonerasing* homomorphism. A language L is *bounded* if there exist $r \geqq 0$ and strings $w_1, \ldots, w_r$ such that $L \subseteq w_1^* \ldots w_r^*$. (For a string w, w^* is used to abbreviate $\{ w \}^*$.)

A *semiAFL (full semiAFL)* is a family of languages containing at least one nonempty language and closed under the operations of intersection with regular sets, union, inverse homomorphism and nonerasing homomorphism (resp., arbitrary homomorphism). An *AFL (full AFL)* is a semiAFL (resp., full semiAFL) that is additionally closed under product of languages and Kleene*. For a language L, $\mathscr{M}(L)$ denotes the smallest semiAFL containing L and $\hat{\mathscr{M}}(L)$, the smallest full semiAFL containing L. Similarly, $\mathscr{F}(L)$ and $\hat{\mathscr{F}}(L)$ denote, respectively, the smallest AFL and full AFL containing L.

A homomorphism $h : \Sigma^* \to \Delta^*$ is said to perform *linear erasing on* $L_1 \subseteq \Sigma^*$ if there is some constant k such that whenever $|x| \geqq k$ and $x \in L_1$, then $|x| \leqq k|h(x)|$. Let $\mathscr{M}^{\mathrm{LIN}}(L)$ denote the smallest semiAFL containing L that is also closed under "linear erasing": if $L_1 \in \mathscr{M}^{\mathrm{LIN}}(L)$ and h performs linear erasing on L_1, then $h(L_1) \in \mathscr{M}^{\mathrm{LIN}}(L)$.

We define:

$$\mathrm{COPY} = \{ ww : w \in \{ a, b \}^* \}$$

and

$$^*\mathrm{COPY} = \{ (wc)^k : k \geqq 0, \ w \in \{ a, b \}^* \}.$$

The language SHIFT can be defined as follows. Let:

$$\Sigma_0 = \{ a, b \} \times \{ a, b \} = \{ [a, a], [a, b], [b, a], [b, b] \}.$$

For strings $x = x_1 \ldots x_n$ and $y = y_1 \ldots y_n$ with $x_i, y_i \in \{ a, b \}$ (so $n = |x| = |y|$), let $\langle x, y \rangle = [x_1, y_1] \ [x_2, y_2] \ldots [x_n, y_n]$. Then $\mathrm{SHIFT} \subseteq (\Sigma_0 \cup \{ c \})^*$ is the language

$$\mathrm{SHIFT} = \{ e \} \cup \{ \langle w_1, w_2 \rangle c \langle w_2, w_3 \rangle c \ldots c \langle w_m, w_{m+1} \rangle c : m \geqq 1, \ w_i \in \{ a, b \}^*, \ |w_1| = |w_2| = \ldots = |w_{m+1}| \}.$$

For the homomorphism $h : \{ a, b, c \}^* \to (\Sigma_0 \cup \{ c \})^*$ determined by defining $h(a) = [a, a]$, $h(b) = [b, b]$, $h(c) = c$, we have $^*\mathrm{COPY} = h^{-1}(\mathrm{SHIFT})$, so $^*\mathrm{COPY} \in \mathscr{M}(\mathrm{SHIFT})$. Note also that COPY is a member of $\mathscr{M}(^*\mathrm{COPY})$.

We can now express the families of languages accepted by one-way nondeterministic machines with one working tape of the given kind in terms of the semiAFL and AFL operators and the languages above:

$\mathscr{M}$(COPY), reset tape, finite delay;

$\mathscr{F}$(COPY), reset tape, finite delay, with reinitialization;

$\hat{\mathscr{M}}$(COPY), reset tape;

$\hat{\mathscr{F}}$(COPY), reset tape, with reinitialization;

$\mathscr{M}$(*COPY), circular tape, finite delay;

$\mathscr{F}$(*COPY), circular tape, finite delay with reinitialization;

$\hat{\mathscr{M}}$(*COPY), circular tape;

$\hat{\mathscr{F}}$(*COPY), circular tape, with reinitialization;

$\mathscr{M}$(SHIFT), writing circular tape, finite delay;

$\mathscr{F}$(SHIFT), writing circular tape, finite delay, with reinitialization;

$\hat{\mathscr{M}}$(SHIFT), writing circular tape;

$\hat{\mathscr{F}}$(SHIFT), writing circular tape, with reinitialization.

Further explanation of the relationship between one-way nondeterministic machines and semiAFL and AFL generators and how to compute these generators can be found in [6, 17].

3. NON-INCLUSION RESULTS

There are twelve apparent classes determined by the three generators (COPY, *COPY, SHIFT) and the four operators ($\mathscr{M}$, $\mathscr{F}$, $\hat{\mathscr{M}}$, $\hat{\mathscr{F}}$), but three pairs of these classes are in fact equal. For machines with one reset tape, requiring operation with finite delay (i.e., "quasirealtime") does not cause any decrease in their power of acceptance, whether or not reinitialization of the tape is allowed between resets. Also, machines with one writing circular tape can perform any effective computation if sufficient time is allowed.

THEOREM 3.1:

(1) $\mathscr{M}(\text{COPY}) = \hat{\mathscr{M}}(\text{COPY})$ *and* $\mathscr{F}(\text{COPY}) = \hat{\mathscr{F}}(\text{COPY})$.

(2) $\hat{\mathscr{M}}(\text{SHIFT}) = \hat{\mathscr{F}}(\text{SHIFT})$ *is the class of all recursively enumerable languages.*

Proof: (1) A simple variation of the construction given in [7] for linear context-free languages (i.e., for one-turn pushdown automata) can be used to convert any machine with one reset tape to a nondeterministic machine, also with one reset tape, that operates with finite delay, so that

$\hat{\mathscr{M}}(\text{COPY}) \subseteq \mathscr{M}(\text{COPY})$. It follows from general principles [6, 8] that the AFL generated by COPY must also be full: $\hat{\mathscr{F}}(\text{COPY}) \subseteq \mathscr{F}(\text{COPY})$.

(2) Certainly each language in $\hat{\mathscr{F}}(\text{SHIFT})$ is recursively enumerable. On the other hand, any recursively enumerable language is the set accepted by some Turing machine with one-way input and a single one-way infinite tape. Such a machine M can be simulated by a machine M' that has a writing circular tape: M' begins its computations by guessing the amount of tape M will need and then makes successive sweeps of that portion of its tape while following a computation of M on the input. Thus any recursively enumerable language can be accepted by a nondeterministic machine with one writing circular tape that operates without time bound, and so is a member of $\hat{\mathscr{M}}(\text{SHIFT})$. (In contrast, a deterministic machine with a circular tape can use only linear space in any accepting computation.) □

Since it is clear that any member of $\mathscr{F}(\text{SHIFT})$ or $\hat{\mathscr{F}}(\text{*COPY})$ must be a recursive set, the following corollary to theorem 3.1 is immediate.

COROLLARY 3.2:

$$\mathscr{F}(\text{SHIFT}) \subsetneq \hat{\mathscr{M}}(\text{SHIFT}) \text{ and } \hat{\mathscr{F}}(\text{*COPY}) \subsetneq \hat{\mathscr{M}}(\text{SHIFT}).$$

Consider now the remaining eight classes, lying strictly below $\hat{\mathscr{M}}(\text{SHIFT})$. The inclusions among them that follow from their definitions are shown in figure 1. We will see that all the inclusions shown (by upward lines) are proper and that each pair of classes with no obvious inclusion (e. g., $\mathscr{M}(\text{*COPY})$ and $\mathscr{F}(\text{COPY})$) are in fact noncomparable classes.

The following technical lemma is useful for extending negative results, of the form $L \notin \mathscr{L}$, from semiAFLs to AFLs.

LEMMA 3.3 [2,8,11]: *Suppose $\mathscr{L}$ is a semiAFL (full semiAFL) containing $\{e\}$ and L_1 is a language in $\mathscr{F}(\mathscr{L})$ (resp., $\hat{\mathscr{F}}(\mathscr{L})$).*

(1) *If L_1 has the property that whenever $AB \subseteq L_1$, either A or B is finite, then $L_1 \in \mathscr{L}$.*

(2) *If $L_1 = \{0^n x 1^n : x \in L_2,\ n \geq 1\}$ for some language $L_2 \subseteq \Sigma^+$ where $0, 1 \notin \Sigma$, then $L_1 \in \mathscr{L}$.* □

We now proceed to identify some limits on the power of machines with the three types of tapes. Let $3\text{-COPY} = \{(uc)^3 : u \in \{a, b\}^*\}$, $E = \{a^n b^n : n \geq 1\}$ and $C_2 = \{ucudvcv : u, v \in \{a, b\}^+\}$. Notice that 3-COPY is a restriction of *COPY, E is a restriction of COPY and C_2 is similar to $(\text{COPY})^2$. Theorems 3.4-3.6 show that $3\text{-COPY} \notin \mathscr{F}(\text{COPY})$, $(cEd)^+ \notin \hat{\mathscr{M}}(\text{*COPY})$ and $C_2 \notin \mathscr{M}(\text{SHIFT})$. These facts will form the basis for deriving complete information about non-inclusions among the classes.

First, we show that a machine with one reset tape cannot check equality of three strings, even if reinitializations of the tape are allowed and (by virtue of theorem 3.1) no time bound is imposed.

THEOREM 3.4: The language 3-COPY $= \{(uc)^3 : u \in \{a, b\}^*\}$ does not belong to $\mathscr{F}$(COPY).

Proof: The language 3-COPY has the property used in lemma 3.3(1): if $AB \subseteq$ 3-COPY then either A or B is finite (and in fact must consist of at most one element). It is straightforward to derive an intercalation theorem for $\mathscr{M}$(COPY) from which it will follow that 3-COPY $\notin \mathscr{M}$(COPY); hence from lemma 3.3, 3-COPY $\notin \mathscr{F}(\mathscr{M}(\text{COPY})) = \mathscr{F}$(COPY). More general versions of this fact have been given by Ibarra ([16] theorem 2.3) and by Ginsburg and Spanier ([14], p. 387). □

Machines with one nonwriting circular tape that operate without time bound have more power than might appear from their definition. The class $\hat{\mathscr{M}}$(*COPY), for example, is closed under concatenation of languages, and also under the operation "chevron" of lemma 3.3(2), which takes a language $L \subseteq \Sigma^+$ to $\{0^n x 1^n : x \in L, n \geq 1\}$ when 0, $1 \notin \Sigma$. Simple constructions using machines with two-track work tapes can be used to show these closure properties. That $\hat{\mathscr{M}}$(*COPY) is not closed under Kleene* is a consequence of the following theorem.

THEOREM 3.5: *Let* $E = \{a^n b^n : n \geq 1\}$. *Then* $(c\,E\,d)^+ \notin \hat{\mathscr{M}}$(*COPY).

Proof: Suppose $(c\,E\,d)^+$ is accepted by a nondeterministic machine M with one nonwriting circular tape that operates without time bound. The "blocks" of a word in $(c\,E\,d)^+$ are its subwords of the form $ca^n b^n d$. M operates by sweeping its worktape while reading the symbols in a block, and the proof focuses on the state of M at the start and end of such sweeps during accepting computations. We may assume that at least one input symbol is read during each sweep. Call the sweep in which the first b of a block is read, the "center sweep" for that block.

We note first that not too many resets can be made while reading one block (claim 1); otherwise, the finite control of M would allow the sweeps of the worktape to be repeated, contradicting the fact that E contains no infinite regular set. Also, at least one reset must be made while reading a sufficiently long block (claim 2), or else a different worktape could be used to allow M to accept a block not in E.

In order to find a contradiction, we consider an accepting computation on a word with a large number of long blocks. For such a word, there must be two blocks for which the sweeps of the worktape do not overlap such that

exactly the same input is read from the blocks in all but the center sweeps. However, there is a linear constraint (claim 3) relating the total length of a block and the input read outside the center sweep, and we can force this to be violated by arranging for one block to be much longer than the other.

Let Γ be the tape alphabet of M and let k_0 be the size of its state set. For $y \in \Gamma^+$ let $S(y)$ be the set of pairs of states (p, q) such that (for some input) M can completely scan y on its worktape, beginning in state p and ending in state q. Define a relation $\equiv$ on Γ^+ by: $y_1 \equiv y_2$ if and only if $S(y_1) = S(y_2)$. Clearly $\equiv$ is a right-invariant equivalence relation of finite index, say k_1.

Let $k_2 = k_0 + 4$, $k_3 = 1 + k_1 (1 + k_0)^{k_2}$ and $k_4 = 2(1 + k_0)^{k_2 + 1}$.

If $x \equiv xy$ then for any computation C of M given work tape xyz and for any $n \geqq 0$ there is a computation C' of M given work tape $xy^n z$ such that C' contains the same number of sweeps of the work tape as C and each sweep begins and ends in the same state in C as it does in C'. In particular, if C is an accepting computation then C' is also accepting.

CLAIM 1: *In any accepting computation on a word w, M makes at most* $k_2 - 1$ *resets while reading each block of w.*

Proof: Suppose that during an accepting computation with work tape y, M makes r resets while reading a block $ca^n b^n d$. Since some input must be read during each sweep of the work tape, M must then have made at least $r-3$ full sweeps of y while only $a's$ and $b's$ are being read. If $r \geqq k_2$ then $r - 3 \geqq k_0 + 1$ so two of these sweeps begin in the same state, say the j-th and k-th sweeps, $k > j$. That part of the input, $v \neq e$, read from the start of the j-th sweep up to the start of the k-th sweep can then be deleted or repeated without affecting the computation during the other blocks. Hence $a^n b^n = xvy$, $v \neq e$, and for all $m \geqq 0$, M accepts a word with a block $cxv^m yd$, so $xv^* y \subseteq E$, which is impossible.

CLAIM 2: *In any accepting computation on a word w, M makes at least one reset while reading each block* $ca^n b^n d$ *of w for which* $n > k_0 k_1$.

Proof: Suppose that there is an accepting computation with work tape y in which no reset is made while a block $ca^n b^n d$ is read. Since the index of $\equiv$ is k_1 and M has k_0 states, if $n > k_0 k_1$ then there must be a decomposition $y = y_1 y_2 y_3$ such that (i) $y_1 \equiv y_1 y_2$ and (ii) during computation on this block M reads input a^s, $s > 0$, while scanning y_2 and is in the same state when it enters y_2 and when it enters y_3. But then $y_1 y_3 \equiv y_1 y_2 y_3$ and there is an accepting computation of M with work tape $y_1 y_3$ on an input that contains [using (ii)] a subword $ca^{n-s} b^n d \notin c E d$.

CLAIM 3: *Suppose in an accepting computation M reads $a^{n-s}b^{n-t}$ during the center sweep of its work tape for a block ca^nb^nd. Then $s \geqq \lfloor (n-t)/k_3 \rfloor$.*

Proof: Consider an accepting computation with work tape y in which ca^nb^nd is the $i-$th block of the input and during the center sweep for the $i-$th block, M reads $a^{n-s}b^{n-t}$ (and possibly c or d). Notice that s is the number of $a's$ read from the $i-$th block before the center sweep. The claim is clearly true if $n-t<k_3$, so let $m=\lfloor (n-t)/k_3 \rfloor$ and assume $m \geqq 1$.

For a prefix y_1 of y, let $\underline{\gamma}(y_1)$ be the sequence of states (from this accepting computation) in which M leaves y_1 during its sweeps of y while reading the $i-$th block (i. e., $\underline{\gamma}(y_1)$ is the $i-$th block crossing sequence at the end of y_1). The length of each such sequence is at most k_2 since (from claim 1) fewer than k_2 resets are made while the $i-$th block is being read. Define an equivalence relation E_i on prefixes of y by: $y_1 E_i y_1'$ iff $y_1 \equiv y_1'$ and $\underline{\gamma}(y_1)=\underline{\gamma}(y_1')$. The index of E_i is then at most $k_1(1+k_0)^{k_2}=k_3-1$.

Divide the $n-t\,b's$ read during the center sweep for the $i-$th block into sections of length m, the last section containing at most $m\,b's$. To each of the first k_3 sections ($k_3 \leqq (n-t)/m$) associate that prefix y' of y such that $y=y'y''$ and $M's$ head is on the first symbol of y'' as it begins to read that section. Since the index of E_i is less than k_3, two of these associated prefixes of y must be equivalent under E_i (possibly they are also equal). Therefore there is a decomposition $y=y_1y_2y_3$ such that $y_1 \equiv y_1y_2$ and $\underline{\gamma}(y_1)=\underline{\gamma}(y_1y_2)$, and b^{km}, $k \geqq 1$, is read while y_2 is scanned during the center sweep for the $i-$th block. It follows that there is an accepting computation of M with work tape $y_1y_2y_2y_3$ (since $y_1 \equiv y_1y_2$) in which, during the sweeps of the work tape corresponding to those for the $i-$th block, input ca^ub^vd is read with $u \leqq n+s$ and $v \geqq n+km$. For the bounds on u and v, we observe that (1) while reading $a's$ from the $i-$th block, M scans a prefix of y_1 during the center sweep and the second y_2 need only cause it to read double the number of $a's$ read up to that point; and (2) since $\underline{\gamma}(y_1)=\underline{\gamma}(y_1y_2)$, M can repeat, while scanning the second y_2 in the center sweep, the steps that read b^{km}. Thus, M accepts a word with a block ca^ub^vd, so $u=v$ and therefore $n+km \leqq n+s$, or $s \geqq km \geqq m$, as desired.

Now consider an accepting computation of M on a word

$$w=ca^{n_1}b^{n_1}d \ldots ca^{n_r}b^{n_r}d$$

for which $r \geqq k_4$, $n_i > k_0k_1$ and $n_{2i+1} \geqq (k_3+1)n_{2i-1}+k_3$. Since $n_{2i} > k_0k_1$, from claim 2 at least one reset is made while each even-numbered block is read, so the sweeps of the work tape while blocks $2i-1$ and $2i+1$ are read

do not overlap. For $1 \leqq i \leqq \lceil r/2 \rceil$, let α_{2i-1} be the sequence of states in which M begins the sweeps of its work tape during which symbols from block $2i-1$ are read, followed by the state in which the last such sweep is ended. From claim 1, each such sequence has length at most k_2+1. Since $\lceil r/2 \rceil Q \geqq (1/2)k_4 = (1+k_0)^{k_2+1}$, two of these sequences must be identical, say those for the j-th and k-th blocks, where j, k are odd and $j \leqq k-2$.

Since $\alpha_j = \alpha_k$, for each i, the $i-$th sweep in which symbols of block j are read and the $i-$th sweep in which symbols of block k are read must begin in the same state and lead to the same state for beginning the next sweep. Hence the $i-$th sweep for block j can be substituted for the $i-$th sweep for block k and an accepting computation will result. If the $i-$th sweep for one block reads input in ca^*, a^+, b^+ or $b^* d$ then exactly the same input must be read in the $i-$th sweep for the other; otherwise substitution of one sweep for the other will cause M to accept an input with a block not in cEd. For the same reason, if the $i-$th sweep for block j reads input in $ca^+ b^+$, $a^+ b^+$ or $a^+ b^+ d$ then the corresponding sweep for block k must read input of the same form. (Input in $ca^+ b^+ d$ is not possible since $n_k > k_0 k_1$.) Therefore there exist s, $t \leqq n_j$ such that for both blocks j and k, M reads $sa's$ up to the center sweep and $tb's$ after the center sweep, and (in particular) reads $a^{n_k - s} b^{n_k - t}$ during the center sweep for the k-th block. From claim 3, then, $s \geqq \lfloor (n_k - t)/k_3 \rfloor$ so $n_j \geqq \lfloor (n_k - t)/k_3 \rfloor$. But since $t \leqq n_j$ and $k \geqq j+2$, $n_k - t \geqq n_k - n_j \geqq n_{j+2} - n_j$ and $n_{j+2} - n_j \geqq k_3(n_j+1)$ so $\lfloor (n_k - t)/k_3 \rfloor \geqq n_j + 1$, a contradiction. □

The following lemma will be used in showing that:

$$C_2 = \{ ucudvcv : u, v \in \{a, b\}^+ \}$$

cannot be accepted by any finite-delay machine with one writing circular tape. It also reveals that, although $\hat{\mathscr{M}}$(*COPY) is closed under concatenation, $\mathscr{M}$(*COPY) is not.

LEMMA 3.6: *Suppose* A_1, A_2, B_1, $B_2 \subseteq \Sigma^+$, $\phi \notin \Sigma$, *and* $L \in \mathscr{M}$(*COPY) *satisfies* $B_1 \phi B_2 \subseteq L \subseteq A_1 \phi A_2$. *Then either* (i) *there is a regular set* R_1 *such that* $B_1 \subseteq R_1 \subseteq A_1$, *or* (ii) *there is a regular set* R_2 *such that* $B_2 \subseteq R_2 \subseteq A_2$. *Hence if* $L = A_1 \phi A_2$ *then at least one of* A_1, A_2 *is a regular set.*

Proof: Suppose L is accepted by a nondeterministic finite-delay machine M with one nonwriting circular tape. Let k be such that M takes at most km steps to read $m \geqq 1$ input symbols.

For $t \geqq 1$ define $L_t = \{ u_1 \in \Sigma^+ :$ there exists some u_2 such that M accepts $u_1 \phi u_2$ given a work tape of length at most $t \}$. Notice that $L_t \subseteq A_1$ and each L_t is a regular set.

Define $R_2 = \{ u_2 \in \Sigma^+ :$ there exists some u_1 such that M accepts $u_1 \not\in u_2$ without making a reset while reading $u_2 \}$, so $R_2 \subseteq A_2$. Also, R_2 is a regular set: if $u_2 \in R_2$ then there are strings u_1 and y such that M accepts $u_1 \not\in u_2$ with work tape y in a computation in which no reset is made while u_2 is being read and at most q resets are made while u_1 is being read, where q is the size of the state set of M. Therefore there is a finite-state machine that, given u_2, can (by keeping track of a q-tuple of states of M) guess successive symbols of a string y and test whether there is some u_1 such that M accepts $u_1 \not\in u_2$ given y without making a reset while reading u_2.

Now, if $B_2 \subseteq R_2$ then (ii) is true since $R_2 \subseteq A_2$ is a regular set. If $B_2 \not\subseteq R_2$ then let v be some string in $B_2 - R_2$, and let $\tau = k|v|$. In this case, $B_1 \subseteq L_\tau$ so since $L_\tau \subseteq A_1$ is regular, (i) is true. To see that $B_1 \subseteq L_\tau$, consider any string $u \in B_1$. Then $u \not\in v \in L$ so there is an accepting computation of M on $u \not\in v$ given some work tape y, but $v \notin R_2$ so M makes at least one reset while reading v. Since M takes at most $\tau = k|v|$ steps to read v, it follows that $|y| \leqq \tau$; thus M accepts $u \not\in v$ given a work tape of length at most τ, so $u \in L_\tau$. □

THEOREM 3.7: *Let* $C_1 = \{ ucu : u \in \{a, b\}^+ \}$, $C_2 = C_1 d C_1$ *and* $V_2 = \{ 0^n w 1^n : n \geqq 1, \quad w \in C_2 \}$. *Then* (i) $C_2 \notin \mathscr{M}$(SHIFT) *and* (ii) $V_2 \notin \mathscr{F}$(SHIFT).

Proof: (i) Suppose otherwise, that C_2 is accepted by a finite-delay machine M with one writing circular tape. Let α be such that M takes at most αn steps to read $n \geqq 1$ input symbols. Let β be such that for each $m \geqq 1$ there are less than $2^{\beta m}$ tape configurations of M with a work tape of length at most m (where a tape configuration consists of a state, a work-tape string and a position on the work tape).

For $u, v \in \{a, b\}^+$, let $w(u, v) = ucudvcv \in C_2$ and let $m(u, v)$ be the length of a shortest work tape used by M in an accepting computation on $w(u, v)$.

Some inputs must require long worktapes, in order for M to distinguish among them. In particular, there are infinite sets U and V such that for $u \in U$ and $v \in V$, $m(u, v)$ grows linearly with $|w(u, v)|$. With a long worktape, the number of resets M makes cannot be too large (because the time is bounded), and we are able to conclude that a fixed number of resets is used to accept strings $w(u, v)$ for $u \in U$, $v \in V$. However, lemma 3.6 then implies that C_1 contains an infinite regular set, a contradiction.

For strings $u_1 \neq u_2$, M cannot reach to same tape configuration (from its initial configuration) after reading u_1 and reading u_2, or else M would accept a string not in L. There are 2^n strings of length n but less than $2^{\beta(n/\beta)} = 2^n$ tape configurations of M with a worktape of length at most n/β; it follows that for each $n \geqq 1$, there is a string $u(n)$ of length n such that for every

$v \in \{a, b\}^+$, $m(u(n), v) > n/\beta$. Similarly, for each $n \geqq 1$ there is a string $v(n)$ of length n such that for every $u \in \{a, b\}^+$, $m(u, v(n)) > n/\beta$. Let $U = \{u(n) : n \geqq 1\}$ and $V = \{v(n) : n \geqq 1\}$; then $u \in U$ and $v \in V$ imply:

$$m(u, v) > (1/\beta) \max\{|u|, |v|\} \geqq (1/7\beta)|w(u, v)|.$$

Let $C_U = \{ucu : u \in U\}$ and $C_V = \{vcv : v \in V\}$.

Let S be the set of strings in C_2 that are accepted by M in a computation with fewer than $7\alpha\beta$ resets. A machine that accepts S with a nonwriting circular tape can be constructed from M, so $S \in \mathscr{M}$(*COPY). Suppose $u \in U$ and $v \in V$ and consider any accepting computation of M on $w(u, v)$. If the number of resets made is t, then the number of steps is bounded below by $(t+1)m(u, v) > (t+1)|w(u, v)|/7\beta$ and bounded above by $\alpha|w(u, v)|$ so $t+1 < 7\alpha\beta$; hence S includes the set $\{w(u, v) : u \in U, \quad v \in V\}$. Thus $C_U d C_V \subseteq S \subseteq C_1 d C_1$ with C_U and C_V infinite, so (from lemma 3.6) C_1 must contain an infinite regular set.

(ii) From lemma 3.3, if $V_2 \in \mathscr{F}$(SHIFT) $= \mathscr{F}(\mathscr{M}$(SHIFT)) then V_2, and hence C_2, would be in $\mathscr{M}$(SHIFT), contradicting (i). □

From corollary 3.2 and theorems 3.4-3.6 and the properties of the classes, the following conclusions can be drawn:

(*a*) 3-COPY $\in \mathscr{M}$(*COPY)-$\mathscr{F}$(COPY);

(*b*) $(cEd)^+ \in \mathscr{F}$(COPY)-$\hat{\mathscr{M}}$(*COPY);

(*c*) SHIFT $\in \mathscr{M}$(SHIFT)-$\hat{\mathscr{F}}$(*COPY);

(*d*) $C_2 \in (\hat{\mathscr{M}}$(*COPY) $\cap \mathscr{F}$(COPY))-$\mathscr{M}$(SHIFT);

(*e*) $V_2 \in \hat{\mathscr{M}}$(*COPY)-$\mathscr{F}$(SHIFT).

The ten inclusions shown in figure 1 are therefore all proper: numbers (1) and (6) because of (*a*); (2), (5) and (9) from (*b*); (3) and (8) from (*c*); (4) and (7) from (*d*); and (10) from (*e*). The three upper rows of the diagram contain classes that are not comparable: from (*c*) and (*e*), $\mathscr{F}$(SHIFT) and $\hat{\mathscr{F}}$(*COPY) are incomparable; from (*b*)-(*e*), the three classes $\mathscr{M}$(SHIFT), $\hat{\mathscr{M}}$(*COPY) and $\mathscr{F}$(*COPY) are mutually incomparable; and from (*a*) and (*b*), $\mathscr{M}$(*COPY) and $\mathscr{F}$(COPY) are incomparable. Finally, $\mathscr{F}$(COPY) is not comparable to either $\hat{\mathscr{M}}$(*COPY) or $\mathscr{M}$(SHIFT), from (*a*), and (*b*) and (*d*); $\mathscr{M}$(SHIFT) and $\hat{\mathscr{F}}$(*COPY) are incomparable, from (*c*) and (*d*); and $\mathscr{F}$(SHIFT) and $\hat{\mathscr{M}}$(*COPY) are incomparable, from (*c*) and (*e*). Thus, no inclusions hold among the classes except as shown in figure 1, and each of the inclusions shown is proper.

It is not difficult to show that the languages C_2 and V_2 belong to $\mathscr{M}^{\text{LIN}}$(*COPY), i.e., they can be defined from *COPY using linear-erasing

homomorphisms (and the semiAFL operations). Therefore:

$$\mathscr{M}(\text{SHIFT}) \subsetneqq \mathscr{M}^{\text{LIN}}(\text{SHIFT}) \subsetneqq \hat{\mathscr{M}}(\text{SHIFT}),$$

and $\mathscr{M}(*\text{COPY}) \subsetneqq \mathscr{M}^{\text{LIN}}(*\text{COPY})$; similar statements hold for the AFLs determined by the languages. Whether the inclusion between $\mathscr{M}^{\text{LIN}}(*\text{COPY})$ and $\hat{\mathscr{M}}(*\text{COPY})$ is proper remains open. Since, from theorem 1, $\hat{\mathscr{M}}(\text{COPY}) = \mathscr{M}(\text{COPY})$, certainly:

4. PALINDROMES

This section is devoted to showing that the set of palindromes $\text{PAL} = \{ wcw^R : w \in \{ a, b \}^+ \}$ cannot be accepted either by a machine with one non-writing circular tape or by a finite-delay machine with one writing circular tape, even if reinitialization of the tape is allowed. The class $\hat{\mathscr{M}}(\text{PAL})$ of linear context-free languages is therefore incomparable to the classes between $\mathscr{M}(\text{COPY})$ and $\mathscr{F}(\text{SHIFT})$. A technical result on bounded sets and noncommuting words leads to a proof that no unbounded set of palindromes can belong to the closure of the regular sets under homomorphic duplication. This fact, in turn, is the basis for the proof that PAL does not belong to $\mathscr{F}(\text{SHIFT})$.

LEMMA 4.1: *Consider an alphabet* Σ *and strings* $u, v \in \Sigma^*$.

(1) [1, 12]. *If* $\{ u, v \}^*$ *is not freely generated by* u *and* v *then there exists* $r \in \Sigma^*$ *such that* $u, v \in r^*$.

(2) [5]. *If* $uv \neq vu$ *and* $W \subseteq \Sigma^*$ *has the property that any string in* $\{ u, v \}^*$ *is a factor of some string in* W, *then* W *is an unbounded set.*

LEMMA 4.2: *Suppose* $s, t \in \Sigma^*$ *are distinct strings of the same length and* $B \subseteq \Sigma^*$ *is a bounded set. There exist* $x, y, \sigma, \tau \in \{ s, t \}^+$ *such that*

(i) $xy \neq yx$,

and

(ii) *for any* $z, z' \in B$ *and* $w, w' \in \{ x, y \}^+$ *if* $z \sigma w \tau$ *is a prefix of* $z' \sigma w' \tau$ *then* $w = w'$.

Proof: Let $n = |s| = |t| \geq 1$. Since $s \neq t$ also $st \neq ts$ so from lemma 4.1(2) there is a string $\varphi \in \{ s, t \}^*$ such that φ is not a factor of any string in B. Let $m = \max \{ |\varphi|, 4 \}$ and define:

$x = tsst$

$y = stts$

$\sigma = \varphi\, sts$

$\tau = t^m$.

Note that σ is not a factor of any string in B, and $|x\tau| = |y\tau| > |\sigma|$.

Since $s \neq t$ and they have the same length, $xy \neq yx$ so (i) holds and $\{x, y\}^*$ is freely generated by x and y. For (ii), suppose that $z\sigma w\tau v = z'\sigma w'\tau$ for some v, where $z, z' \in B$ and $w, w' \in \{x, y\}^+$; we must show that $w = w'$. Since σ is not a factor of z', $|z\sigma| > |z'|$ and hence:

$$|w'\tau| = |z\sigma w\tau v| - |z'\sigma| = |v| + (|z\sigma| - |z'|) + (|w\tau| - |\sigma|) > |v|.$$

It is convenient to distinguish five cases for $|v|$. In all but the first case ($v = e$), a contradiction will be found.

(1) If $v = e$ then $z\sigma w = z'\sigma w'$ with (say) $|w'| \geq |w|$. Since x, y freely generate $\{x, y\}^*$, $w' = \bar{w}w$ for some $\bar{w} \in \{x, y\}^*$ and $z\sigma = z'\sigma\bar{w}$. If $\bar{w} \neq e$ then it ends with x or with y; however, the suffix of σ of length $3n$ is sts, which is not a suffix of x or y, so in fact $\bar{w} = e$ and $w = w'$.

(2) If $0 < |v| < n$ then v is a suffix of $\tau = t^m$, hence of t, so write $t = uv$, $u \neq e$. Then $z\sigma wt^m = z'\sigma w' t^{m-1} u = z'\sigma w' t^{m-2} u(vu)$ so $t = vu = uv$ and $z\sigma wv = z'\sigma w'$. Applying lemma 4.1(1) to the equation $t = uv = vu$, there is a primitive string (i.e., one that is not a proper power of another string) r such that $t, u, v \in r^+$. Consider now the possible final segments (x or y) of w and w' -in each case we find that s and t must be equal.

(2.1) $w = w_1 x$, $w' = w_1' x$: the suffix of $z\sigma wv = z\sigma w_1 tsstv$ of length $4n$ is $usstv$, which must be equal to x, the suffix of $z'\sigma w'$ of the same length. Cancelling u from the left and $t = uv$ from the right we have $ssv = vss$. Since v is a power of the primitive string r, this implies [using lemma 4.1(1)] that $s \in r^+$ so (since $|s| = |t|$), $s = t$.

(2.2) $w = w_1 x$, $w' = w_1' y$: the suffix of $z\sigma wv$ of length n is $uv = t$ and the suffix of $z'\sigma w'$ of length n is s, so $s = t$.

(2.3) $w = w_1 y$, $w' = w_1' x$: taking the suffixes of length $2n$ and cancelling v from the right, we have $us = su$. Since $u \in r^+$ and r is primitive, s is also a power of r [by lemma 4.1(1)] so $s = t$.

(2.4) $w = w_1 y$, $w' = w_1' y$: taking the suffixes of length $n + |v|$, $sv = vs$ and it again follows that $s = t$.

(3) If $n \leqq |v| < 2n$, then, since $z\sigma w\tau v = z'\sigma w' t^{m-2} t^2$, $v = t_1 t$ where t_1 is a suffix of t, say $t = t_2 t_1$. Cancelling v from the right, we have $z\sigma wt^m = z'\sigma w' t^{m-2} t_2 = z'\sigma w' t_2 (t_1 t_2)^{m-2}$ so $t = t_1 t_2 = t_2 t_1$ and $z\sigma wtt_1 = z'\sigma w'$. If w' ends with y then the suffix of length n of $z'\sigma w'$ is s, but the suffix of length n of $z\sigma wtt_1$ is $t_2 t_1 = t$, so $s = t$. If $w' = w_1' x$ and $w = w_1 x$ then $z\sigma w_1 tssttt_1 = z'\sigma w_1' tsst$ so (taking the suffixes of length $2n$ and cancelling t from the right) $t_2 t_1 = s$ and $s = t$. If $w' = w_1' x$ and $w = w_1 y$

then (taking the suffixes of length $3n$) $t_2 stt_1 = sst$ so $t_2 st_1 = ss$. From this equation we see that t_2 is a prefix of s and t_1 is a suffix of s, so since $|t_2| + |t_1| = |t| = |s|$, we have $s = t_2 t_1 = t$.

(4) If $2n \leqq |v| < mn = |\tau|$ then (from $z\sigma w\tau v = z'\sigma w' t^m$) for some j, $0 \leqq j \leqq m-3$, $v = t_3 t^{j+2}$ where $t = t_4 t_3$. Write $t = t_5 t_6$ with $|t_5| = |t_3|$. Cancelling the suffixes of length mn, we have $z\sigma wt^{j+2} t_5 = z'\sigma w'$. The suffix of w' of length $2n$ is therefore $(t_6 t_5)^2$, but w' ends with st or ts so $s = t = t_6 t_5$.

(5) If $|v| \geqq |\tau|$ then τ is a suffix of v, say $v = v_1 \tau$. As noted above, $|v| < |w'\tau|$ so v_1 is a suffix of w': write $w' = \bar{w}uv_1$ with $\bar{w} \in \{x, y\}^*$ and u a prefix of x or of y, so that $z\sigma w\tau = z'\sigma \bar{w}u$. Since $|u| < |x| = |y| \leqq |\tau|$, u is thus a suffix of $\tau = t^m$. The overlap this implies between x or y and t^4 forces s and t to be equal. If $|u| \geqq 2n$ then $u = t_7 t^{j+2}$ where $0 \leqq j \leqq 1$ and $t = t_8 t_7$, so the prefix of u of length $2n$ is $(t_7 t_8)^2$. Since u is a prefix of $x = tsst$ or of $y = stts$, it follows that $s = t = t_7 t_8$. If $n \leqq |u| < 2n$ then $u = t_9 t$ where $t = t_{10} t_9$ so (from $z\sigma w\tau = z'\sigma \bar{w}u$) $z\sigma wt^{m-2} t_{10} = z'\sigma \bar{w}$. Taking the suffixes of length $2n$ from each side, $(t_9 t_{10})^2$ is equal to either st or ts, so $s = t = t_9 t_{10}$. Finally, if $|u| < n$ then $t = \bar{u}u$ for some $\bar{u}$ and $z\sigma wt^{m-1}\bar{u} = z'\sigma \bar{w}$, so $s = t = u\bar{u}$. □

Let $\mathscr{L}_{\text{DUP}}$ denote the class of languages formed by homomorphic duplications of regular sets, that is, languages of the form $\{h_1(x) \ldots h_k(x) : x \in R\}$ for some $k > 0$, homomorphisms $h_1, \ldots, h_k$ and regular set R. The class $\mathscr{L}_{\text{DUP}}$ is equal to the class of equal matrix languages [13], and to the union of the full semiAFLS generated by the languages:

$$k\text{-COPY} = \{(wc)^k : w \in \{a, b\}^*\}, \qquad k \geqq 0.$$

LEMMA 4.3: *Any subset of PAL that belongs to $\mathscr{L}_{\text{DUP}}$ is bounded.*

Proof: Suppose L_0 belongs to $\mathscr{L}_{\text{DUP}}$ and is a subset of PAL. Let $L_1 = \{w : wcw^R \in L_0\}$. It will be shown that if L_1 is unbounded then there is an a-transducer M such that $\text{PAL} = M(L_0)$. But $\mathscr{L}_{\text{DUP}}$ is closed under a-transductions, so this contradicts the fact that PAL does not belong to $\mathscr{L}_{\text{DUP}}$ [10] and hence L_1 and L_0 are bounded.

Since $L_0 \in \mathscr{L}_{\text{DUP}}$, also $L_1 \in \mathscr{L}_{\text{DUP}}$ so write $L_1 = \{h_1(z) \ldots h_N(z) : z \in R\}$ for a regular set R and homomorphisms $h_1, \ldots, h_N$. If L_1 is unbounded then so is some $h_i(R)$ so let $P = \min\{i : h_i(R) \text{ is unbounded}\}$.

Since $h_P(R)$ is unbounded, there exist strings u_0, s_0, t_0, v_1 such that $u_0\{s_0, t_0\}^* v_1 \subseteq R$ and $h_P(s_0)$, $h_P(t_0)$ do not commute [3]. Let $u_1 = u_0 s_0 t_0$, $s_1 = s_0 t_0$, $t_1 = t_0 s_0$, $s = h_P(s_1)$ and $t = h_P(t_1)$. Then $h_P(u_1) \neq e$, $|s| = |t|$ and $s \neq t$. Let $R_1 = u_1\{s_1, t_1\}^+ v_1 \subseteq R$. Notice that if $e \in h_i(R_1)$ then $h_i(R_1) = \{e\}$. The bounded set $h_1(R_1) \ldots h_{P-1}(R_1)\{h_P(u_1)\}$ is therefore contained in a set B of the form $B = z_1^+ \ldots z_r^+$ for $r \geqq 1$ and nonempty strings $z_1, \ldots, z_r$. Applying

lemma 2 to s, t and B, we obtain strings x, y, σ, $\tau \in \{s, t\}^+ = \{h_P(s_1), h_P(t_1)\}^+$ with the following properties:

(*a*) for any $w \in \{x, y\}^+$ there is a unique decomposition $w = w_1 \ldots w_n$ with $n \geqq 1$ and $w_i \in \{x, y\}$;

(*b*) for any z, $z' \in B$, any w, $w' \in \{x, y\}^+$ and any α, $\alpha' \in \{a, b\}^*$ if $z\sigma w\tau\alpha = z'\sigma w'\tau\alpha'$ then $w = w'$;

and

(*c*) for any $w \in \{x, y\}^+$ there exist $z \in B$ and $\bar{w} \in \{a, b\}^*$ such that $z\sigma w\tau\bar{w} \in L_1$.

The a-transducer M that produces PAL from L_0 operates as follows. Given a string wcw^R, M looks for and erases a prefix of w in $z_1^+ \ldots z_r^+$ followed by the marking string σ. It then decodes a string in $\{x, y\}^+$, putting out a for x and b for y. When the other marking string τ is encountered, it is erased along with any other letters before the center c. This process is reversed for w^R, resulting in output of the form ucu^R.

Let $Q = \{p_0, \ldots, p_r, q_0, q_1, q_2, \hat{q}_2, \hat{q}_1, \hat{q}_0, \hat{p}_r, \ldots, \hat{p}_0\}$ and let H be the set of transitions on Q given by:

$$
\begin{aligned}
H = \{ & (p_{i-1}, z_i, e, p_i), (p_i, z_i, e, p_i) : 1 \leqq i \leqq r \} \\
& \cup \{ (p_r, \sigma, e, q_0), (q_0, x, a, q_1), (q_0, y, b, q_1), \\
& (q_1, x, a, q_1), (q_1, y, b, q_1), (q_1, \tau, e, q_2), \\
& (q_2, a, e, q_2), (q_2, b, e, q_2), (q_2, c, c, \hat{q}_2), \\
& (\hat{q}_2, a, e, \hat{q}_2), (\hat{q}_2, b, e, \hat{q}_2), (\hat{q}_2, \tau^R, e, \hat{q}_1), \\
& (\hat{q}_1, x^R, a, \hat{q}_1), (\hat{q}_1, y^R, b, \hat{q}_1), (\hat{q}_1, x^R, a, \hat{q}_0), \\
& (\hat{q}_1, y^R, b, \hat{q}_0), (\hat{q}_0, \sigma^R, e, \hat{p}_r) \} \\
& \cup \{ (\hat{p}_i, z_i^R, e, \hat{p}_i), (\hat{p}_i, z_i^R, e, \hat{p}_{i-1}) : 1 \leqq i \leqq r \}.
\end{aligned}
$$

Let M be the a-transducer with state set Q, transitions H, initial state p_0 and single accepting state $\hat{p}_0$. M can only reach its accepting state on input of the form $\alpha_1 c \alpha_2^R$ with $\alpha_1 = z\sigma w\tau\beta_1$ and $\alpha_2 = z'\sigma w'\tau\beta_2$ where z, $z' \in B$ and w, $w' \in \{x, y\}^+$. The output produced from such an input is $u_1 cu_2^R$ where, for $f(a) = x$ and $f(b) = y$, $w = f(u_1)$ and $w' = f(u_2)$. By property (*a*), u_1 and u_2 are uniquely determined by w and w', respectively. Using property (*c*), it follows easily that $\mathrm{PAL} \subseteqq M(L_0)$. On the other hand, if $\alpha_1 = \alpha_2$ then [by property (*b*)] $w = w'$ and so $u_1 = u_2$; hence $M(L_0) \subseteqq \mathrm{PAL}$. □

The following property of $\hat{\mathscr{M}}$ (*COPY) allows us (since PAL has no infinite regular subset) to conclude from the previous lemma that no unbounded subset of PAL can be in $\hat{\mathscr{M}}$ (*COPY).

LEMMA 4.4: *If L is a language in $\hat{\mathscr{M}}$(*COPY) and L has no infinite regular subset then L is in $\mathscr{L}_{\text{DUP}}$.*

Proof: Let L be a language in $\hat{\mathscr{M}}$(*COPY) and let M be a nondeterministic machine with one nonwriting circular tape that accepts L. We may assume that M never makes a complete sweep of its work tape without reading input. Suppose M has k states. If for every x in L, M accepts x with fewer than k resets then M can be converted to a machine limited to k resets and so $L=L(M)$ is in the full semiAFL generated by k-COPY. Otherwise let x be some string such that M makes at least k resets in an accepting computation on x and consider the sequence of states reached by M just before the resets. Two of the states in this sequence must be the same, so x can be written as uvy for some strings u, v, y, and uv^*y is contained in L. The string v cannot be empty since M reads at least one input symbol between resets, so L contains an infinite regular set. □

THEOREM 4.5: *Any subset of PAL that belongs to $\hat{\mathscr{M}}$(*COPY) is bounded.*

COROLLARY: $\text{PAL} \notin \hat{\mathscr{F}}$(*COPY).

THEOREM 4.6: *PAL does not belong to $\mathscr{F}$*(SHIFT).

Proof: It is sufficient to show that $\text{PAL} \notin \mathscr{M}$(SHIFT): by virtue of lemma 3.1(1), if $\text{PAL} \in \mathscr{F}$(SHIFT) then $\text{PAL} \in \mathscr{M}$(SHIFT).

Suppose that PAL is accepted by a machine M with one writing circular tape that accepts in linear time. There is an unbounded subset of $L(M)$ for which the length of the worktape used in accepting grows at least linearly and so (since the time is bounded) the number of resets made cannot be large; but M restricted to a fixed number of resets can be simulated without writing on the worktape and hence accepts (by theorem 4.5) a bounded language.

Let α, β be constants such that each string in PAL of length n is accepted by M in some computation of at most αn steps, and M has fewer than $2^{\beta m}$ tape configurations with a worktape of length at most m. For $u \in \{a, b\}^+$ let $m(u)$ be the length of a shortest worktape used by M in an accepting computation on ucu^R of at most $\alpha(2|u|+1)$ steps.

For $n \geqq 1$, let $U(n)=\{u \in \{a, b\}^+ : |u|=n+1, \ m(u)>n/\beta\}$, and let $W=\{ucu^R : u \in U(n), \ n \geqq 1\}$. Then the cardinality of $U(n)$ is greater than 2^n: otherwise there would be strings $u_1 \neq u_2$ (of length $n+1$) such that during accepting computations, M reached the same tape configuration (with a worktape of length at most n/β) after reading u_1 and after reading u_2. It follows that for each $n \geqq 2$, there are more than 2^{n-1} strings in W of length $2n+1$, and therefore W cannot be a bounded set.

Let $L \subseteq$ PAL be the set of strings accepted by M in computations with at most $5\alpha\beta$-2 resets, so $L \in \hat{\mathscr{M}}$(*COPY). From theorem 4.5, then, L is bounded and so $W \not\subseteq L$; let $n \geq 1$ and $u \in U(n)$ be such that $ucu^R \notin L$. Since $ucu^R \notin L$, M makes at least $5\alpha\beta$ sweeps of its worktape in any accepting computation on ucu^R, so $\alpha|ucu^R| \geq (5\alpha\beta)m(u)$, or, since $|u| = n+1$, $2n+3 \geq (5\beta)m(u)$. Since $u \in U(n)$, $m(u) > n/\beta$ so $2n+3 > 5n$, a contradiction. □

ACKNOWLEDGMENTS

The authors wish to thank J. Berstel for his help in proving lemma 4.2.

APPENDIX

For completeness, we present here a proof, based on the work of Klingenstein [10] that PAL does not belong to $\mathscr{L}_{\text{DUP}}$. This fact will follow from the theorem below on the structure of the Parikh images of bounded languages in $\mathscr{L}_{\text{DUP}}$. (Basic information on these notions may be found in [14].)

NOTATION:

(1) For strings x, y : $x \underline{!}\, y$ if x is a subsequence of y.

(2) If $h = (h_1, \ldots, h_s)$ is a sequence of homomorphisms and x is a string then $h(x) = h_1(x) \ldots h_s(x)$. Let $\mathscr{R}_s$ denote the class of languages of the form $\{(h_1, \ldots, h_s)(x) : x \in R\}$ where R is a regular set.

(3) For $p = (p_1, \ldots, p_t) \in \mathbb{N}^t$, $\lambda(p) = \#\{i : 1 \leq i \leq t, p_i \neq 0\}$ is the number of nonzero entries in p, and for $1 \leq j \leq \lambda(p)$, $[p, j]$ is the index of the j-th nonzero entry in p. For $c \in \mathbb{N}^t$ and P a finite subset of $\mathbb{N}^t$, $L(c, P) = \{c + \sum_{p \in P} k_p . p : k_p \geq 0\}$ is the linear set with constant c and set of periods P.

DEFINITION: A set $P \subseteq \mathbb{N}^t$ is *s-interlaced* if $\lambda(p) \leq s$ for each $p \in P$ and there exist a linear order $<$ on P and for each $p \in P$ an increasing function $f_p : \{1, \ldots, \lambda(p)\} \to \{1, \ldots, s\}$ satisfying for all $p, q \in P$ and all i, j : if either $f_p(i) < f_q(j)$ or $p < q$ and $f_p(i) = f_q(j)$ then $[p, i] \leq [q, j]$.

THEOREM: *Let $a_1, \ldots, a_t$ be distinct letters and let ψ: $\{a_1, \ldots, a_t\}^* \to \mathbb{N}^t$ be the function taking a_i to the vector with 1 in the i-th position and zeros elsewhere. If $L \subseteq a_1^* \ldots a_t^*$ and $L \in \mathscr{R}_s$ then $\psi(L)$ is equal to a finite union of linear sets, each with an s-interlaced set of periods.*

Proof: Fix a sequence of homomorphisms $h = (h_1, \ldots, h_s)$, each $h_i : \Sigma^* \to \{a_1, \ldots, a_t\}^*$. Let Φ be the following property of subsets of

Σ^* : $\Phi(U) \Leftrightarrow$ there exist $n \geqq 0$; $c_1, \ldots, c_n \in \mathbb{N}^t$; $P_1, \ldots, P_n \subseteq \mathbb{N}^t$; an order $<$ that is linear on each P_i; and, for each $p \in \bigcup_i P_i$, an increasing function f_p: $\{1, \ldots, \lambda(p)\} \to \{1, \ldots, s\}$ and a string $w_p \in U$; such that $\psi h(U) = \bigcup_{i=1}^{n} L(c_i, P_i)$ and for all i:

(a) for all $p, q \in P_i$ and all j, k if either $f_p(j) < f_q(k)$ or $p < q$ and $f_p(j) = f_q(k)$ then $[p, j] \leqq [q, k]$; and

(b) for all $p \in P_i$ and all j, $1 \leqq j \leqq \lambda(p)$, $a_{[p,j]} \mid h_{f_p(j)}(w_p)$.

Note that if $\Phi(U)$ holds, then (in particular) $\psi h(U)$ is a finite union of linear sets $L(c_i, P_i)$, each P_i s-interlaced [by (a)]. It therefore suffices to show that $\Phi(R)$ holds whenever R is a regular subset of Σ^* and $h(R) \subseteq a_1^* \ldots a_t^*$. This is proved by induction on the structure of R. If R is a finite set then $\{c_1, \ldots, c_n\} = \psi h(R)$ and $P_1 = \ldots = P_n = \varnothing$ serve to establish $\Phi(R)$. If R is equal to $R_1 \cup R_2$, $R_1 R_2$ or R_1^* and $h(R) \subseteq a_1^* \ldots a_t^*$ then also $h(R_1)$, $h(R_2) \subseteq a_1^* \ldots a_t^*$. Suppose, then that $\Phi(R_1)$ and $\Phi(R_2)$ hold—we will see that $\Phi(R)$ does as well. Let $c_1, \ldots, c_n$, $P_1, \ldots, P_n$, $<_1$ and functions f_p and strings w_p ($p \in \bigcup P_i$) verify $\Phi(R_1)$, and similarly $d_1, \ldots, d_m$, $Q_1, \ldots, Q_m$, $<_2$ and f_p, w_p ($p \in \bigcup Q_j$) for $\Phi(R_2)$.

If $R = R_1 \cup R_2$ then:

$$\psi h(R) = \psi h(R_1) \cup \psi h \qquad (R_2) = \left(\bigcup_{i=1}^{n} L(c_i, P_i) \right) \cup \left(\bigcup_{j=1}^{m} L(d_j, Q_j) \right)$$

and this representation serves for $\Phi(R)$, with $< = <_1 \cup <_2$ and the functions f_p and strings w_p within each P_i and Q_j.

If $R = R_1 R_2$ then $\psi h(R) = \psi h(R_1) + \psi h(R_2) = \bigcup_{i,j} L(c_i + d_j, P_i \cup Q_j)$. Let $< = <_1 \cup <_2 \cup \{(p, q) : p \in \bigcup_i P_i, q \in \bigcup_j Q_j\}$ and associate f_p and w_p with $p \in (\bigcup_i P_i) \cup (\bigcup_j Q_j)$. For $\Phi(R)$, it is enough to show that each $P_i \cup Q_j$ satisfies (a) and (b). Since $c_i \in L(c_i, P_i) \subseteq \psi h(R_1)$, let $x_1 \in R_1$ be a string such that $\psi h(x_1) = c_i$; similarly let $x_2 \in R_2$ be such that $d_j = \psi h(x_2)$. Now (a) is satisfied (by the induction hypothesis) if $p, q \in P_i$ or $p, q \in Q_j$. For $p \in P_i$ and $q \in Q_j$, note that $w_p w_q \in R_1 R_2 = R$. If $f_p(l) < f_q(k)$ then:

$$a_{[p,l]} a_{[q,k]} \mid h_{f_p(l)}(w_p) h_{f_q(k)}(w_q) \mid h(w_p w_q) \in a_1^* \ldots a_t^*$$

so $[p, l] \leqq [q, k]$. If $f_q(l) < f_p(k)$ then again $a_{[q,l]} a_{[p,k]} \mid h(w_p w_q)$ so $[q, l] \leqq [p, k]$. Also, $p < q$ (but $q \not< p$) and if $f_p(l) = f_q(k) = f$ then

$$a_{[p,l]} a_{[q,k]} \mid h_f(w_p w_q) \mid h(w_p w_q)$$

so $[p, l] \leqq [q, k]$. For (b), if $p \in P_i$ then $w_p x_2 \in R$ and if $p \in Q_j$ then $x_1 w_p \in R$; in either case the condition on location of letters remains true.

Finally, suppose $R = R_1^*$, $h(R) \subseteq a_1^* \ldots a_t^*$. In this case there are indices $1 \leqq i(1) \leqq \ldots \leqq i(s) \leqq t$ such that $h_j(R_1) \subseteq a_{i(j)}^*$. (This follows from the observations that if $a_k \mid h_j(x)$ and $a_l \mid h_j(y)$ then $a_k a_l a_k \mid hh(xyx)$, and if $a_k \mid h_j(x)$ and $a_l \mid h_{j+i}(y)$ then $a_k a_l \mid h(xy)$.) For $c \in \{c_1, \ldots, c_n\}$ let $w_c \in R_1$ be a string such that $\psi h(w_c) = c$. Since $h(w_c) \in a_{i(1)}^* \ldots a_{i(s)}^*$, $\lambda(c) \leqq s$. Let f_c: $\{1, \ldots, \lambda(c)\} \to \{1, \ldots, s\}$ be the increasing function defined by $f_c(j) = \min \{i : a_{[c,j]} \circ h_i(w_c)\}$. Let $Q = \{c_1, \ldots, c_n\} \cup \bigcup_i P_i$. For $J \subseteq \{1, \ldots, n\}$, let $d_J = \sum_{j \in J} c_j$ and $Q_J = \{c_j : j \in J\} \cup \bigcup_{j \in J} P_j$, with $d_{\hat{\varphi}}$ the zero vector and $Q_{\hat{\varphi}} = \varnothing$. Then $\psi h(R) = \bigcup_J L(d_J, Q_J)$ and this representation serves for $\Phi(R)$, with any order on Q and functions f_q and strings w_q, $q \in Q$. Either by the definition above or because $\Phi(R_1)$ holds, for each $q \in Q$, f_q: $\{1, \ldots, \lambda(q)\} \to \{1, \ldots, s\}$ is an increasing function and w_q is a string in $R_1 \subseteq R$ such that $a_{[q,k]} \mid h_{f_q(k)}(w_q)$, $1 \leqq j \leqq \lambda(q)$, so that (b) is satisfied. For $p, q \in Q$, if $f_p(l) \leqq f_q(k)$ then

$$a_{[p,l]} a_{[q,k]} \mid h_{f_p(l)}(w_p) h_{f_q(k)}(w_q) \mid h(w_p w_q)$$

and $w_p w_q \in R_1^2 \subseteq R$ so $[p, l] \leqq [q, k]$; hence also (a) is satisfied. □

COROLLARY: $\mathrm{PAL} \notin \mathscr{L}_{\mathrm{DUP}}$.

Proof: If $\mathrm{PAL} \in \mathscr{L}_{\mathrm{DUP}}$ then for some $s \geqq 2$, $\mathrm{PAL} \in \mathscr{R}_s$. Since $\mathscr{R}_s$ is a full semiAFL, also:

$$L = \{a_1^{n(1)} a_2^{n(2)} \ldots a_t^{n(t)} a_{t+1}^{n(t)} \ldots a_{2t}^{n(1)} : n(i) \geqq 0\}$$

belongs to $\mathscr{R}_s$, where $t = s 2^{s-2}$. From the theorem, $\psi(L) = \bigcup_{i=1}^{n} L(c_i, P_i)$ where each P_i is s-interlaced, say by way of an order $<$ and functions f_p: $\{1, \ldots, \lambda(p)\} \to \{1, \ldots, s\}$. From the form of strings in L we see that each $P_i \subseteq \psi(L)$ and therefore for $p \in P_i$, $\lambda(p)$ is even and

$$[p, j] + [p, \lambda(p) + 1 - j] = 2t + 1, \qquad 1 \leqq j \leqq \lambda(p).$$

We may assume that $\lambda(p) > 0$.

For each i, let $Q_{i,1} \cup \ldots \cup Q_{i,m(i)}$ be the partition of P_i induced by the equivalence relation: $p \equiv q$ if $f_p = f_q$. There are $2^{s-1}-1$ increasing functions from sets $\{1, \ldots, 2l\}$, $1 \leqq l \leqq \lfloor s/2 \rfloor$, to $\{1, \ldots, s\}$, so $m(i) \leqq 2^{s-1}-1$. Also, all the vectors in $Q_{i,j}$ have the same set of nonzero coordinates: suppose $f_p = f_q$ and (say) $p < q$. Then for all l, $f_p(l) = f_q(l)$ so $[p, l] \leqq [q, l]$, and

$$f_p(\lambda(p)+1-l) = f_q(\lambda(q)+1-l)$$

so $[p, \lambda(p)+1-l] \leqq [q, \lambda(q)+1-l]$ and therefore

$$[p, l] = 2t+1-[p, \lambda(p)+1-l] \geqq 2t+1-[q, \lambda(q)+1-l] = [q, l].$$

Now, let α be larger than any entry in the constants $c_1, \ldots, c_n$ and consider the vector q with all entries equal to α. Since $q \in \psi(L)$ there is some i such that $q \in L(c_i, P_i)$, so there exist $k_p \in \mathbb{N}$ such that $q = c_i + \sum_{j=1}^{m(i)} \left(\sum_{p \in Q_{i,j}} k_p p \right)$. From the remark above, each inner sum contributes nonzero entries to a set of at most s coordinates of $q - c_i$. Since all the entires in $q - c_i$ are nonzero, at least $2t/s$ of the inner sums must be nonzero and so $m(i) \geqq 2t/s = 2^{s-1} > m(i)$. □

REFERENCES

1. E. K. BLUM, *A Note on Free Subsemigroups with Two Generators*, Bull. Amer. Math. Soc., Vol. 71, 1965, pp. 678-679.
2. L. BOASSON and M. NIVAT, *Sur diverses familles de langages fermés par transduction rationnelle*, Acta Inf., Vol. 2, 1973, pp. 180-188.
3. R. BOOK, S. GREIBACH and C. WRATHALL, *Reset Machines*, J. Comput. Syst. Sc., Vol. 19, 1979, pp. 256-276.
4. S. GINSBURG and E. H. SPANIER, *Bounded Regular Sets*, Proc. Amer. Math. Soc., Vol. 17, 1966, pp. 1043-1049.
5. S. GINSBURG and E. H. SPANIER, *Bounded Algol-Like Languages*, Trans. Amer. Math. Soc., Vol. 113, 1964, pp. 333-368.
6. S. GINSBURG and S. GREIBACH, *Principal AFL*, J. Comput. Syst. Sc., Vol. 4, 1970, pp. 308-338.
7. S. GREIBACH, *Erasable Context-Free Languages*, Inf. Control, Vol. 29, 1975, pp. 301-326.
8. S. GREIBACH, *Erasing in Context-Free AFLs*, Inf. Control, Vol. 21, 1972, pp. 436-465.
9. R. HULL, *Reset Languages*, Ph. D. dissertation, University of California at Berkeley, 1979.
10. K. KLINGENSTEIN, *Structures of Bounded Languages in Certain Families of Languages*, Ph. D. dissertation, University of California at Berkeley, 1975.
11. M. LATTEUX, *Cônes rationnels commutativement clos*, R.A.I.R.O.-Informatique théorique, Vol. 11, 1977, pp. 29-51.
12. M. LOTHAIRE, ed., *Combinatorics on Words*, Addison-Wesley, Reading, Mass., 1982.

13. R. SIROMONEY, *On Equal Matrix Languages*, Inf. Control, Vol. 14, 1969, pp. 135-151.
14. S. GINSBURG and E. H. SPANIER, *AFL with the Semilinear Property*, J. Comput. Syst. Sc., Vol. 5, 1971, pp. 365-396.
15. R. SIROMONEY, *Finite-Turn Checking Automata*, J. Comput. Syst. Sc., Vol. 5, 1971, pp. 549-559.
16. O. IBARRA, *Simple Matrix Languages*, Inf. Control, Vol. 17, 1970, pp. 359-394.
17. S. GINSBURG and S. GREIBACH, *On AFL Generators for Finitely Encoded AFA*, J. Comput. Syst. Sc., Vol. 7, 1973, pp. 1-27.
18. F.-J. BRANDENBURG, *Multiple Equality Sets and Post Machines*, J. Comput. Syst. Sc., Vol. 21, 1980, pp. 292-316.