JOURNAL OF COMPUTER AND SYSTEM SCIENCES 6, 168–204 (1972)

Writing Stack Acceptors*

JOSEPH ALPHONSO GIULIANO

Hughes Aircraft Co., Culver City, California 90232

Received September 15, 1971

INTRODUCTION

In recent years, automata theorists have devoted a great deal of effort to the study of two-way acceptors. Examples of such devices include the two-way pushdown acceptor [8], the time-bounded Turing acceptor [10], and the tape-bounded Turing acceptor [10]. A natural extension of these models is obtained by allowing the input head to print on the input tape. A trivial example of the "extended" model is the linear bounded acceptor (lba). Recently, a nontrivial example of the "extended" model has appeared in the literature [14]. The device, called a "writing pushdown acceptor," is essentially a two-way pushdown acceptor that can print on its input tape. In this paper, we introduce and study another example of the extended model, namely, the "writing stack acceptors" (WSA) and their associated family of languages, $\mathscr{L}_{\mathrm{WSA}}$. (As its name indicates, a WSA is essentially a two-way nondeterministic stack acceptor that can print on its input tape.) We also study the deterministic WSA (DWSA), the nonerasing WSA (NEWSA), and the nonerasing deterministic WSA (NEDWSA), as well as their associated families of languages $\mathscr{L}_{\mathrm{DWSA}}$, $\mathscr{L}_{\mathrm{NEWSA}}$, and $\mathscr{L}_{\mathrm{NEDWSA}}$, respectively. In particular, we characterize the four families of languages in terms of Turing machines and auxiliary pushdown Turing machines, both with exponential tape storage.

The paper is divided into four sections. In section one, the notion of a WSA is defined and its operation formalized. Also in section one, the $f(\alpha)$-tape-bounded auxiliary pushdown Turing machine ($f(\alpha)$-APTM) as introduced in [2] is recalled and its operation formalized. This device is essentially a $f(\alpha)$-tape-bounded Turing machine ($f(\alpha)$-TM), together with a pushdown storage, which is not memory limited. (In case the pushdown tape is nonerasing, the definition of $f(\alpha)$-APTM degenerates to that of $f(\alpha)$-TM.)

* This research was supported in part by the National Science Foundation under Grant No. GJ454.

The main results of the paper are that

$$\mathscr{L}_{\text{DWSA}} = \mathscr{L}_{\text{WSA}} = \bigcup_{c \geqslant 1} \mathscr{L}_{2^{c\alpha}\text{-APTM}}$$

and that

$$\mathscr{L}_{\text{NEDWSA}} = \mathscr{L}_{\text{NEWSA}} = \bigcup_{c \geqslant 1} \mathscr{L}_{2^{c\alpha}\text{-TM}}$$

Phrased otherwise, the main results of the paper provide a characterization of the exponential tape-bounded APTM and the exponential tape-bounded TM, each in terms of WSA. Sections two and three develop the machinery necessary to present the main results. Section four establishes the main results, as well as some AFL properties.

Throughout the paper we assume that the reader has a casual knowledge of formal language theory. The reader is referred to [12] for all unexplained definitions and notation.

1. Formalization

In this section we define a writing stack acceptor (WSA), together with several important subcases. We also recall the notion of an "auxiliary pushdown Turing machine" (APTM). A WSA may be informally illustrated as in Fig. 1. It consists of a two-way read-write input tape $¢a_2 \cdots a_{n-1}\$$; a finite state control (fsc); and a stack tape (as distinguished from a pushdown tape) $Y_t \cdots Y_1$, where the top of stack is the leftmost β, to the right of T_1. (β denotes the blank symbol.)

Definition. A *writing stack acceptor* (WSA) is an 8-tuple $S = (K, \Sigma, \Gamma, \delta, \delta_\beta, q_0, Z_0, F)$, where

(1) K and Σ are finite, nonempty sets (of *states* and *inputs*, respectively);

(2) Γ is an alphabet containing Σ, but not the seven distinguished symbols θ, -1, E, $¢$, $\$$, β, I (the elements of $\Gamma - \Sigma$ are called *stack* symbols);

(3) δ is a function from $K \times (\Gamma \cup \{¢, \$\}) \times \Gamma$ into the subsets of

$$K \times (\Gamma \cup \{¢, \$\}) \times \{-1, 0, 1\} \times \{-1, 0, 1\}$$

such that for each q in K and Z in Γ

(a) if $\delta(q, ¢, Z)$ contains (p, b, d_1, d_2) then $b = ¢$ and d_1 is in $\{0, 1\}$, and
(b) if $\delta(q, \$, Z)$ contains (p, b, d_1, d_2) then $b = \$$ and d_1 is in $\{-1, 0\}$;

(4) δ_β is a function from $K \times (\Gamma \cup \{¢, \$\}) \times \Gamma$ into the subsets of

$$K \times (\Gamma \cup \{¢, \$\}) \times \{-1, 0, 1\} \times (\Gamma \cup \{\theta, I, E\})$$

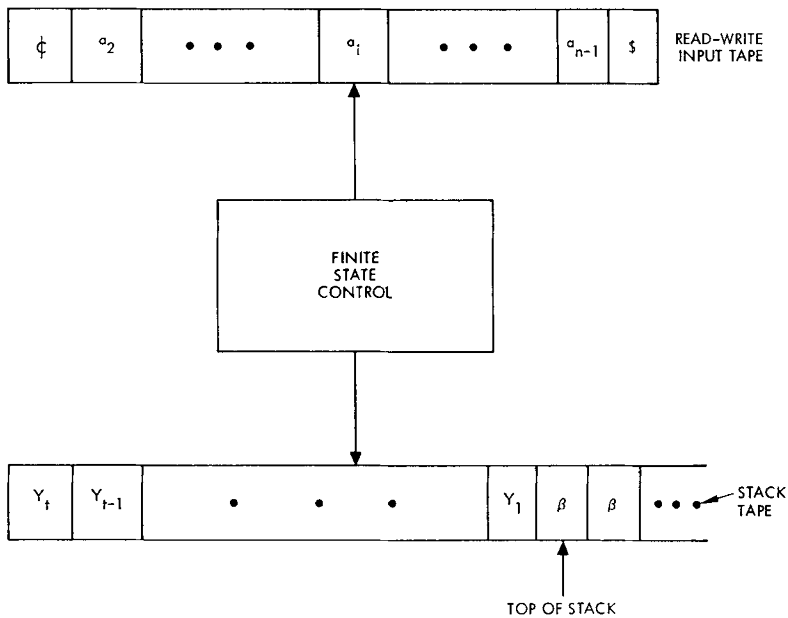

FIGURE 1

such that for each q in K and Z in Γ

(a) if $\delta_\beta(q, ¢, Z)$ contains (p, b, d_1, d_2) then $b = ¢$ and d_1 is in $\{0, 1\}$ and
(b) if $\delta_\beta(q, \$, Z)$ contains (p, b, d_1, d_2) then $b = \$$ and d_1 is in $\{-1, 0\}$;

(5) q_0 is in K (the *start* state), Z_0 is in Γ (the *initial* stack symbol), and $F \subseteq K$ (the set of *accepting* states).

The special character β is called a *blank*. The characters ¢ and \$ are called the *left* and *right end*-markers, for the input. Note that neither ¢ nor \$ occur in Σ. The initial input to a WSA is an element of $¢\Sigma^+\$$. The next move function when the stack head is not scanning the top of stack is denoted by δ. The next move function when the stack head is reading β at the top of stack is denoted by δ_β.

Agreement. The positions on the stack are numbered from right to left, beginning with the leftmost β at position 0. The symbol y, unless specified otherwise, will denote a word of the form $y = Y_t Y_{t-1} \cdots Y_1$,[1] with each Y_j in Γ, and will denote the stack of S.

DEFINITION. A WSA S is said to be a *deterministic* WSA (DWSA) if $\delta(q, a, Z)$ and $\delta_\beta(q, a, Z)$ each contain at most one element for all (q, a, Z) in $K \times (\Gamma \cup \{¢, \$\}) \times \Gamma$.

[1] $t = 0$ will denote the empty stack.

DEFINITION. A WSA S is said to be *nonerasing*, abbreviated NEWSA, if for each (q, a, Z) in $K \times (\Gamma \cup \{¢, \$\}) \times \Gamma$, (p, b, d, X) in $\delta_\beta(q, a, Z)$ implies $X \neq E$. A nonerasing DWSA is abbreviated as NEDWSA.

Notation. Let N denote the positive integers. For each positive integer n let $N_n = \{1,..., n\}$.

DEFINITION. A *configuration* of a WSA is any element of the set

$$\bigcup_{n \geqslant 3} (K \times ¢\Gamma^{n-2}\$ \times N_n \times \Gamma^* \times (N \cup \{0\})).$$

DEFINITION. Each configuration of the form $(q, w, j, y, 0)$ is called a *top* configuration.

DEFINITION. For each WSA S let $\vdash$ (or $\vdash_S$ when S is to be emphasized) be the relation on the set of configurations defined as follows (for $n \geqslant 3$, $w_1 = b_1 \cdots b_n$, $w_2 = b_1 \cdots b_{i-1}b'b_{i+1} \cdots b_n$, $y = Y_t \cdots Y_1$, and b' and each Y_j in Γ):

(1) $(p, w_1, i, y, j) \vdash (q, w_2, r, y, m)$ if $\delta(p, b_i, Y_j)$ contains (q, b', d_1, d_2), $r = i + d_1$, and $m = j + d_2$;

(2) Let $C = (p, w_1, i, y, 0)$, $r = i + d_1$, and let $\delta_\beta(p, b_i, Y_1)$ contain (q, b', d_1, X). Then

(a) $C \vdash (q, w_2, r, yZ, 0)$ if $X = Z$,
(b) $C \vdash (q, w_2, r, Y_t \cdots Y_2, 0)$ if $X = E$,
(c) $C \vdash (q, w_2, r, y, 0)$ if $X = \theta$,
(d) $C \vdash (q, w_2, r, y, 1)$ if $X = I$.

Thus (2) implies that if S is scanning β at the top of stack, δ_β will depend on Y_1, the symbol to the left of β.

Notation. Let

$$\overset{+}{\vdash} \text{ and } \overset{*}{\vdash} \ (\overset{+}{\vdash}_S \text{ and } \overset{*}{\vdash}_S,$$

when S is to be emphasized) be, respectively, the transitive and reflexive-transitive closure of $\vdash$.

DEFINITION. Each configuration C such that $(q_0, ¢u\$, 1, Z_0, 0) \overset{*}{\vdash}_S C$ is called an *S-confifuration.*

DEFINITION. A word u in Σ^+ is *accepted* by a WSA S if

$$(q_0, ¢u\$, 1, z_0, 0) \overset{*}{\vdash}_S (p, ¢v\$, j, y, m)$$

for some p in F and some S-configuration $(p, ¢v\$, j, y, m)$. The set of all words accepted by S is denoted by $T(S)$.

Notation. Let $\mathscr{L}_{\mathrm{WSA}}(\mathscr{L}_{\mathrm{DWSA}}, \mathscr{L}_{\mathrm{NEWSA}}, \mathscr{L}_{\mathrm{NEDWSA}})$ denote the family of all sets accepted by some WSA (DWSA, NEWSA, NEDWSA)S.

An APTM may be informally illustrated as in Fig. 2. It consists of a two-way read-only input tape $¢a_2 \cdots a_{n-1}\$$; a finite state control (fsc); a pushdown tape (pdt) $Y_1 \cdots Y_J$; and k two-way infinite cead/write work tapes.

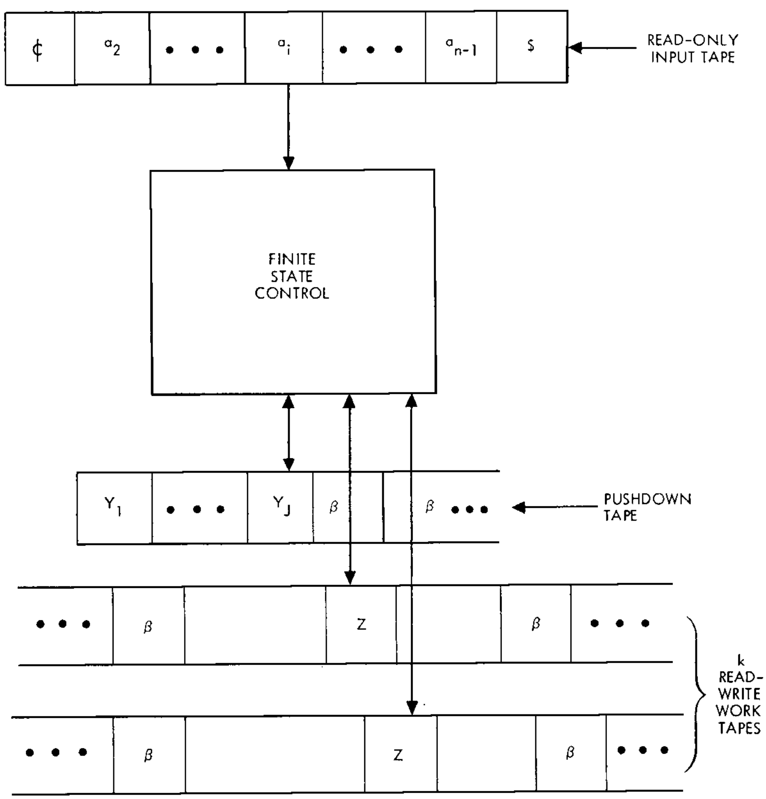

Figure 2

Notation. For each set X, let $X_\beta = X - \{\beta\}$.

Definition. An *Auxiliaty Pushdown Turing Machine* (APTM) is an 8-tuple $(K, \Sigma, W, \delta, q_0, Z_0, F, k)$, where

(1) K and Σ are finite, nonempty sets (of *states* and *input* symbols, respectively),

(2) W is an alphabet containing Σ, but not the special characters θ, E, ¢, and \$,
(3) k is a positive integer,
(4) δ is a function from[2] $K \times (\Sigma \cup \{¢, \$\}) \times W^{(k)} \times W_\beta$ into the subsets of $K \times \{-1, 0, 1\} \times (W_\beta \times \{-1, 0, 1\})^{(k)} \times (W_\beta \cup \{\theta, E\})$, such that for each p in K, B_i in W, $1 \leqslant i \leqslant k$, and Y in W_β,

(a) if $\delta(p, ¢, B_1, \dots, B_k, Y)$ contains $(q, d, \sigma_1, \dots, \sigma_k, X)$, then d is in $\{0, 1\}$, and

(b) if $\delta(p, \$, B_1, \dots, B_k, Y)$ contains $(q, d, \sigma_1, \dots, \sigma_k, X)$, then d is in $\{-1, 0\}$,

(5) q_0 is in K (the *start* state), Z_0 is in W_β, and $F \subseteq K$ (the set of *accepting* states).

The special characters ¢ and \$ are called the *left* and *right end-markers*, respectively, for the input. Elements of $W - \Sigma$ are called *working* symbols. Z_0 in (5) above is called the *initial* working symbol.

DEFINITION. A *deterministic* APTM, abbreviated DAPTM, is an APTM in which $\delta(p, a, B_1, \dots, B_k, Y)$ contains at most one element for all $(p, a, B_1, \dots, B_k, Y)$ in $K \times (\Sigma \cup \{¢, \$\}) \times W^{(k)} \times W_\beta$.

DEFINITION. Let A be an APTM and $\upharpoonleft$ a distinguised symbol which is not in W. Then a *configuration* is any element of

$$K \times H_1 \times H_2^{(k)} \times W^*,$$

where

$$H_1 = \upharpoonleft ¢\Sigma^+\$ \cup ¢\Sigma^+\$\upharpoonleft \cup ¢\Sigma^+ \upharpoonleft \Sigma^*\$ \cup ¢\Sigma^* \upharpoonleft \Sigma^+\$$$

and

$$H_2 = \upharpoonleft \beta W_\beta{}^* \cup W_\beta{}^* \upharpoonleft \beta \cup W_\beta{}^* \upharpoonleft W_\beta{}^+.$$

Agreement. Unless specified otherwise, the pdt will be denoted by the word $y = Y_1 \cdots Y_J$, $J \geqslant 0$, Y_j in W_β.

Notation. Let $\vdash$ (or $\vdash_A$ when A is to be emphasized) be the binary relation on arbitrary configurations defined as follows. Write

$$\begin{aligned}&(p, a_1 \cdots \upharpoonleft a_i \cdots a_n, u_1 \upharpoonleft v_1, \dots, u_k \upharpoonleft v_k, Y_1 \cdots Y_J)\\ &\qquad \vdash (q, a_1 \cdots \upharpoonleft a_{i+d} \cdots a_n, u_1' \upharpoonleft v_1', \dots, u_k' \upharpoonleft v_k', \gamma)\end{aligned}$$

if $(q, d, \sigma_1, \dots, \sigma_k, X)$ is in $\delta(q, a_i, B_1, \dots, B_k, Y_J)$, and the following two conditions are satisfied:

[2] $W^{(k)}$ is the k-fold cartesian product.

(1) Either

(a) $\gamma = Y_1 \cdots Y_J X$ if X is in W_β,

(b) $\gamma = Y_1 \cdots Y_J$ if $X = \theta$, or

(c) $\gamma = Y_1 \cdots Y_{J-1}$ if $X = E$,

(2) For each j, $1 \leqslant j \leqslant k$, with x_j and y_j in W_β^*, $\bar{B}_j$ in W_β, B_j in W, and $\sigma_j = (B_j', d_j)$,

(a) if (B_j, d_j) is in $W_\beta \times \{-1\}$ and

$$u_j \upharpoonright v_j = X_j \bar{B}_j \upharpoonright B_j y_j ,$$

then

$$u_j' \upharpoonright v_j' = X_j \upharpoonright \bar{B}_j B_j' y_j ,$$

(b) if (B_j', d_j) is in $W_\beta \times \{-1\}$ and

$$u_j \upharpoonright v_j = \upharpoonright B_j y_j ,$$

then

$$u_j' \upharpoonright v_j' = \upharpoonright \beta B_j' y_j ,$$

(c) if (B_j', d_j) is in $W_\beta \times \{0\}$ and

$$u_j \upharpoonright v_j = X_j \upharpoonright B_j y_j ,$$

then

$$u_j' \upharpoonright v_j' = X_j \upharpoonright B_j' y_j ,$$

(d) if (B_j', d_j) is in $W_\beta \times \{1\}$ and

$$u_j \upharpoonright v_j = X_j \upharpoonright B_j \bar{B}_j y_j ,$$

then

$$u_j' \upharpoonright v_j' = X_j B_j' \upharpoonright \bar{B}_j y_j ,$$

and

(e) if (B_j', d_j) is in $W_\beta \times \{1\}$ and

$$u_j \upharpoonright v_j = X_j \upharpoonright B_j ,$$

then

$$u_j' \upharpoonright v_j' = X_j B_j' \upharpoonright \beta .$$

Notation. Let $\vdash^+$ and $\overset{*}{\vdash}$ (or $\vdash^+$ and $\overset{*}{\vdash}_A$ when A is to be emphasized) denote, respectively, the transitive and reflective-transitive closure of $\vdash$.

DEFINITION. Each configuration C such that $(q_0, \upharpoonright ¢u\$, \upharpoonright \beta, \ldots, \upharpoonright \beta, Z_0) \overset{*}{\vdash}_A C$ is called an *A-configuration.*

DEFINITION. Let A be an APTM and let u be in Σ^+. Then A is said to *accept* u if $(q_0, \upharpoonright ¢u\$, \upharpoonright \beta,..., \upharpoonright \beta, Z_0) \overset{*}{\vdash}_A C$ for some configuration

$$C = (q, \nu_1 \upharpoonright \nu_2, u_1 \upharpoonright v_1, ..., u_k \upharpoonright v_k, Y_1 \cdots Y_J)$$

with q in F. Let $T(A)$ denote the set of all words accepted by A.

We now observe that by deleting the pdt component in the definitions of δ and $\vdash$ for APTM, we obtain a version of the familiar Turing machine. Specifically, we have the

DEFINITION. A *Turing Machine* (*with k work tapes*), abbreviated (k tape) TM, is a 7-tuple $T = (K, \Sigma, W, \delta, q_0, F, k)$, where

(1) K, Σ, W, q_0, F, and k are as in an APTM, and

(2) δ is a function from $K \times (\Sigma \cup \{¢, \$\}) \times W^{(k)} \times W_\beta$ into the subsets of $K \times \{-1, 0, 1\} \times (W_\beta \times \{-1, 0, 1\})^{(k)} \times \{\theta\}$.

The definition of deterministic TM (DTM) is obvious. We omit the formalization.

DEFINITION. Let f be a function from the positive integers into the positive integers. Let A be an APTM such that for each word w in $T(A)$, there exists some computation $(q_0, ¢ \upharpoonright w\$, \upharpoonright \beta,..., \upharpoonright \beta, Z_0) \vdash_A \cdots \vdash_A (q, v_1 \upharpoonright v_2, u_1 \upharpoonright v_1,..., u_k \upharpoonright v_k, Y_1 \cdots Y_J)$ with q in F and $| u_j \upharpoonright v_j | \leqslant f(| w |)$[3] for each j, $1 \leqslant j \leqslant k$. Then A is said to be an $f(\alpha)$-*tape-bounded* APTM ($f(\alpha)$-APTM).

Note that if a language L is accepted by some nonerasing $f(\alpha)$-APTM, then L is accepted by some $f(\alpha)$-TM.

Notation. Let $\mathscr{L}_{f(\alpha)-\mathrm{DAPTM}}$, $\mathscr{L}_{f(\alpha)-\mathrm{APTM}}$, $\mathscr{L}_{f(\alpha)-\mathrm{DTM}}$, and $\mathscr{L}_{f(\alpha)-\mathrm{TM}}$ be the families of languages accepted by, respectively, $f(\alpha)$-DAPTM, $f(\alpha)$-APTM, $f(\alpha)$-DTM, and $f(\alpha)$-TM.

2. SIMULATION OF $2^{c\alpha}$-TAPE-BOUNDED DAPTM BY DWSA

In Sections 2 and 3 we show that the following four statements are equivalent for an arbitrary language L:

(i) $L = T(A)$ for some $2^{c_1\alpha}$-tape-bounded DAPTM A for some integer c_1.

(ii) $L = T(A)$ for some DWSA S.

(iii) $L = T(A)$ for some WSA S.

(iv) $L = T(A)$ for some $2^{c_2\alpha}$-tape-bounded APTM A for some integer c_2.

[3] Since an APTM cannot erase, the length of each storage tape cannot decrease during a computation. For each word W, $| W |$ denotes the length of W.

In this section we prove that (i) implies (ii). It is trivial that (ii) implies (iii). In Section 3 we prove that (iii) implies (iv). That (iv) implies (i) is a known result [2].

In constructing new WSA or new APTM we shall usually describe these machines in an operational form only. It will be clear, however, from our description and from standard techniques that a formal specification can readily be made.

We now consider the proof of (i) implies (ii). We first ask the reader to observe that a DWSA can perform certain simple tasks.[4]

2.1. *Given any integer c, a* DWSA *can move its stack head exactly $2^{c|w|}$ positions into its stack, where w is the current input word.*

Proof. By marking its input tape as in a LBA, any DWSA can "count" to $2^{c|w|}$.

2.2. *Let c be any integer and D_1 any symbol not in Γ. A* DWSA *can print a word of the form*

$$w_1 = D_1^{2^{c|w|}}$$

on track two of the stack where w is the current input word.

Proof. Since any DWSA can "count" to $2^{c|w|}$, it can obviously print the word w_1.

2.3. *A* DWSA *S can be constructed with the following property. Let c be a given integer and w a given word. Let v denote the final subword on either track one or two of the stack, with $|v| = 2^{c|w|}$. Then S can print v on track one of the stack.*

Proof. In either case, S merely copies a block to the top of stack on track one using its ability to count to $2^{c|w|}$.

2.4. *A* DWSA *S can be constructed with the following property. Let c be an integer, w an input word, Δ a new symbol, and y and z track two stack words, with $|z| \geqslant |y| = 2^{c|w|}$. Then S, having $y\,\Delta z$ on track two of its stack, can determine whether or not z is of the form xy for same x.*

Proof. Here S again uses its ability to count to $2^{c|w|}$ and "compares" by repeatedly erasing final symbols of z that match with final symbols of y. In any case, S always erases up to symbol Δ.

In 2.5, Theorem 2.1 and occasionally in Section 3, we shall order the words over some alphabet. We thus recall the notion of lexicographical order.

DEFINITION. Let B be any set, simply ordered under $<$. The relation $\leqslant$, called the lexicographical order on B^+, is defined as follows. Let $u = u_{11} \cdots u_{1m}$ and

[4] The reader is referred to [7] for details.

$v = v_{21} \cdots v_{2n}$, $m \leqslant n$, with each u_{1i} and v_{2j} in B, $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$. Write $u \lll v$ if either

(a) $u_{1j} < v_{2j}$ for the smallest j such that $u_{1j} \neq v_{2j}$, or
(b) $u_{1j} = v_{2j}$, $1 \leqslant j \leqslant m$, and $u \neq v$.

Notation. Let c be given integer, $\mathscr{D}$ an alphabet, and $e_{\mathscr{D}}$ an enumeration $D_1, \ldots, D_{|\mathscr{D}|}$ of the elements of $\mathscr{D}$. Let $\lll$ be a lexicographical order on $\mathscr{D}^{2^{c|w|}}$.

2.5. *A* DWSA *S can be constructed with the following property. Let G_i', the i-th word in the ordering $\lll$ on $D^{2^{c|w|}}$, be at the right of the stack on track two. Then S can copy G_i' at the top of stack on track two, simultaneously replacing G_i' by G_{i+1}' on track two.*

Proof. S copies G_i' on track two as in 2.3; however, S replaces G_i' by G_{i+1}' by counting in base $|\mathscr{D}|$.

THEOREM 2.1. *For each $2^{C_1\alpha}$-tape-bounded* DAPTM *A, there exists a* DWSA *S such that $T(A) = T(S)$.*

Proof. Let u be in $T(A)$ and $n = |\mathop{¢} u\$|$. We shall construct S so that S simulates A. In order to describe the computation of S, we need to introduce some notation and concepts.

Let $\Delta_1, \ldots, \Delta_4$ be new symbols and

$$\mathscr{C}_A = \{C \mid (q_0, ¢ \upharpoonright u\$, \upharpoonright \beta, \ldots, \upharpoonright \beta, Z_0) \overset{*}{\underset{A}{\vdash}} C\}.$$

For each C in $\mathscr{C}_A$, we now define a *coded* configuration C'. Let

$$C = (q, a_1 \cdots \upharpoonright a_i \cdots a_n, u_1 \upharpoonright v_1, \ldots, u_k \upharpoonright v_k, Y_1 \cdots Y_J)$$

be an arbitrary element in $\mathscr{C}_A$, and let ν denote the word $\nu = \Delta_1 u_1 \upharpoonright v_1 \Delta_1 \cdots \Delta_1 u_k \upharpoonright v_k \Delta_1$. Let[5]

$$C' = q_1 a_1 \cdots \upharpoonright a_i \cdots a_n \nu Y_J \Delta_2{}^m,$$

where m is such that

$$|C'| = 2^{c_2 n}$$

for some integer C_2. Let $\mathscr{C}_A' = \{C' \mid C \text{ in } \mathscr{C}_A\}$ and $\mu = |\mathscr{C}_A'|$.

Let $\mathscr{D} = K \cup W_\beta \cup \{\upharpoonright, \Delta_1, \Delta_2\}$. Then,

$$\mu \leqslant |\mathscr{D}|^{2^{c_2 n}}.$$

[5] $Y_J = \epsilon$ if $J = 0$.

Let $e_{\mathscr{D}}$ be an enumeration $D_1, \ldots, D_{|\mathscr{D}|}$ of the elements of $\mathscr{D}$. For each i,

$$1 \leqslant i \leqslant |\mathscr{D}|^{2^{c_2 n}};$$

let G_i' be the i-th member of $\mathscr{C}_A'$ in a lexicographical ordering $\leqslant$ of $\mathscr{C}_A'$.

We now introduce notation to label certain elements in $\mathscr{C}_A'$. Let

$$C = (p, a_1 \cdots \upharpoonright a_{i_0} \cdots a_n, u_1 \upharpoonright v_1, \ldots, u_k \upharpoonright v_k, Y_1 \cdots Y_J),$$

$J \geqslant 1$ be in $\mathscr{C}_A$, and let π be the sequence $C_1, \ldots, C_r$, where

$$C_1 = (q_0, \text{¢} \upharpoonright u\$, \upharpoonright \beta, \ldots, \upharpoonright \beta, Y_1), Y_1 = Z_0, C_r = C,$$

and

$$C_j \vdash C_{j+1}$$

for $1 \leqslant j \leqslant r - 1$. For each j, $1 \leqslant j < J$, let $\Pi(j)$ be the two-element subsequence $C_{g(j)}, C_{g(j)+1}$, with $g(j)$ the largest integer, $1 \leqslant g(j) < r$ such that

(1) the pdt component in $C_{g(j)}$ is $Y_1 \cdots Y_j$ and
(2) the pdt component in $C_{g(j)+1}$ is $Y_1 \cdots Y_{j+1}$.

The stack of S is divided into two tracks. Let $\rho(\pi)$ denote the contents of the stack of S when A is in the configuration C_r. Let $\rho_1(\pi)$ and $\rho_2(\pi)$ denote tracks one and two, respectively, of $\rho(\pi)$. Let $\rho_i(\pi(j))$, $i = 1, 2$, denote the contents of track i when A is in the configuration $C_{g(j)+1}$, $1 \leqslant j < J$. Then $\rho_1(\pi(j))$ is of the form

$$\gamma_1 C'_{g(j)} \Delta_4 C'_{g(j)+1},$$

and $\rho_2(\pi(j))$ is of the form

$$\gamma_2 G'_{M(j)} \Delta_4 G_1'$$

for some γ_1 and γ_2 with $|\gamma_1| = |\gamma_2|$, and $M(j)$ is some integer, $1 \leqslant M(j) < \mu$.

Let $l = r - g(J-1) - 1$ and let π_l, when it exists, be the (not necessarily consecutive) sequence $C_{g(J-1)+2, \ldots,} C_r$, of elements in π with pdt component $Y_1 \cdots Y_J$. For $l = 0$, let

$$\rho_1(\pi_l) = \rho_2(\pi_l) = \epsilon.$$

for $l \neq 0$ let

$$\rho_1(\pi_l) = C'_{g(J-1)+2} \Delta_3 \cdots \Delta_3 C_r',$$

and

$$\rho_2(\pi_l) = G'_{m(g(J-1)+2)} \Delta_3 \cdots \Delta_3 G'_{m(r)}, \qquad 1 \leqslant m(i) < M(J), \quad g(J-1) + 2 \leqslant i < r.$$

The stack of S will record the A-computation to date, in the following sense. When A is in the configuration C_r, then

$$\rho_1(\pi) = \rho_1(\pi(1)) \cdots \rho_1(\pi(J-1))\rho_1(\pi_l)$$

and

$$\rho_2(\pi) = \rho_2(\pi(1)) \cdots \rho_2(\pi(J-1))\rho_2(\pi_l).$$

Note that each subword C' of $\rho_1(\pi)$ "lines up" with a corresponding G' in $\rho_2(\pi)$. We will sometimes denote subwords of $\rho(\pi)$ by

$$\begin{matrix} C'_{g(j)}\ \Delta_4, & C'_{g(j)+1}\,\Delta_3, & C_r', \\ & & & \text{etc.} \\ G'_{M(j)}\,\Delta_4, & G_1' \quad \Delta_3, & G'_{m(r)} \end{matrix}$$

Intuitively, $\rho(\pi)$ is a representation of pertinent information about the past behavior of A. It also contains "guesses" about the future behavior of A. In particular, for $j \geqslant 1$, each $G'_{M(j)}$ represents the latest guess at the A-configuration occurring in case A erases Y_{j+1} and thereby revisits Y_j. Thus the symbols

$$\begin{matrix} C'_{g(j)} \\ G'_{M(j)} \end{matrix}$$

are of special interest to S, and so are flanked (on the right) with the symbol

$$\begin{matrix} \Delta_4 \\ \Delta_4 \end{matrix}.$$

Using this notation, we now describe how S updates its stack word $\rho(\pi)$ for each of the three possible moves of A on its pdt. The DWSA S will have each member of the 8-tuple A and the sequence $e_{\mathscr{D}}$ in its fsc. Let

$$C_r' = q_1 a_1 \cdots \upharpoonright a_i \cdots a_n \nu_1 Y_J \Delta_2{}^m,$$

where

$$\nu_1 = \Delta_1 u_1 \upharpoonright v_1 \Delta_1 \cdots \Delta_1 u_k \upharpoonright v_k \Delta_1,$$

with

$$u_j \upharpoonright v_j = x_j \bar{B}_j \upharpoonright B_j Y_j$$

for $1 \leqslant j \leqslant k$, and

$$m = 2^{c_2 n} - |\, q_1 a_1 \cdots \upharpoonright a_i \cdots a_n \nu_1 Y_J \,|.$$

S proceeds as follows:

(3) S reads C_r' to obtain the $k+3$-tuple $T_1 = (q_1, a_i, B_1, ..., B_k, Y_J)$ (by means of the $\upharpoonright$ markers in ν_1 and in $a_1 \cdots a_n$) and stores T_1 in its fsc.

(4) S computes $\delta(T_1)$.

Suppose $\delta(q_1, a_i, B_1, \ldots, B_k, Y_J) = T_2$, where

$$T_2 = (q_2, d, \sigma_1, \ldots, \sigma_k, X)^6.$$

There are three possibilities for X:

(a) $X = Y_{J+1}$, a symbol in W_β. Then

$$C'_{r+1} = q_2 a_1 \cdots \upharpoonright a_{i+d} \cdots a_n \nu_2 Y_{J+1} \Delta_2^{m'},$$

with

$$m' = 2^{c_2 n} - | q_2 a_1 \cdots \upharpoonright a_{i+d} \cdots a_n \nu_2 Y_{J+1} |,$$

and

$$\nu_2 = \Delta_1 u_1' \upharpoonright v_1' \Delta_1 \cdots \Delta_1 u_k' \upharpoonright v_k' \Delta_1,$$

where each $u_j' \upharpoonright v_j'$ depends on $u_j \upharpoonright v_j$ and σ_j is as in the definition of $\vdash_{\Delta}$.

Using 2.2, 2.3 and standard techniques, S can be constructed so that it simultaneously prints

$$\Delta_4 C'_{r+1}$$
$$\Delta_4 G_1'$$

to the right of $\rho(\pi)$, given

$$C_r'$$
$$G'_{m(r)}$$

and T_2. Then S returns to (3) to continue this simulation.

(b) $X = E$.[7] Then

$$C'_{r+1} = q_2 a_1 \cdots \upharpoonright a_{i+d} \cdots a_n \nu_2 Y_{J-1} \Delta_2^{m'},$$

where ν_2 and m' are as in (a). Let $\Delta_{4,J-1}$ denote the rightmost Δ_4 symbol in $\rho_i(\pi(J-1))$, $1 \leqslant i \leqslant 2$. First S enters its stack scanning for the symbol

$$\Delta_{4,J-1}$$
$$\Delta_{4,J-1}$$

[6] Recall that d is in $\{-1, 0, 1\}$, and σ_j is in $W_\beta \times \{-1, 0, 1\}$ for each j, $1 \leqslant j \leqslant k$.

[7] Note that this is the only case involving erasing in the proof.

Next, S reads the symbol Y_{J-1} from $C'_{g(J-1)}$ and stores Y_{J-1} in its fsc. Then S uses T_2, Y_{J-1}, and C_r' to print

$$\begin{matrix} \Delta_3 C'_{r+1} \\ \Delta_3 C'_{r+1} \end{matrix}$$

to the right of its stack. Next, S checks if

$$C'_{r+1} = G'_{M(J-1)}$$

on track two of the stack. By 2.4, letting

$$\Delta_{4,J-1} = \Delta, \qquad Y = G'_{M(J-1)},$$

and letting C'_{r+1} be the final subword of Z, one of two possibilities must occur:

(i) $$G'_{M(J-1)} = C'_{r+1}.$$

By 2.4, S erases every symbol to the right of

$$\begin{matrix} \Delta_{4,J-1} \\ \Delta_{4,J-1} \end{matrix}$$

in the checking process. Then S erases

$$\begin{matrix} \Delta_{4,J-1} \\ \Delta_{4,J-1} \end{matrix}$$

and labels

$$\begin{matrix} C'_{g(J-1)} \\ G'_{M(J-1)} \end{matrix}$$

obsolete by printing

$$\begin{matrix} \Delta_3 \\ \Delta_3 \end{matrix}.$$

By 2.2, S can print G_1' to the right of $\rho_2(\pi)$. By 2.3, letting

$$v = G'_{M(J-1)},$$

S can copy $G'_{M(J-1)}$ from $\rho_2(\pi)$, printing $G'_{M(J-1)}$ to the right on track one. Thus S can be constructed so that S prints

$$\begin{matrix} G'_{M(J-1)} \\ G_1' \end{matrix}$$

to the right of the stack. Then S returns to (3) to continue the simulation.

(ii) $G'_{M(J-1)} \neq C'_{r+1}$. By 2.4, S erases every symbol to the right of

$$\begin{matrix} \Delta_{4,J-1} \\ \Delta_{4,J-1} \end{matrix}$$

in the checking process, leaving the rightmost subword of the stack in the form

$$\begin{matrix} C'_{g(J-1)}\, \Delta_{4,J-1} \\ G'_{M(J-1)}\, \Delta_{4,J-1} \,. \end{matrix}$$

Then S erases

$$\begin{matrix} \Delta_{4,J-1} \\ \Delta_{4,J-1} \end{matrix}$$

and labels

$$\begin{matrix} C'_{g(J-1)} \\ G'_{M(J-1)} \end{matrix}$$

obsolete by printing

$$\begin{matrix} \Delta_3 \\ \Delta_3 \,. \end{matrix}$$

By 2.5, S can print $G_{M(J-1)+1}$ to the right on track two. By 2.3, letting $C'_{g(J-1)} = v$, S can print $C'_{g(J-1)}$ to the right on track one. Thus S prints

$$\begin{matrix} C'_{g(J-1)} \\ G'_{M(J-1)+1} \end{matrix}$$

to the right of the stack. Then S returns to (3) to continue the simulation, beginning again with $C'_{g(J-1)}$.

(c) $X = \theta$. Then

$$C'_{r+1} = q_2 a_1 \cdots \upharpoonright a_{i+d} \cdots a_n \nu_2 Y_J \Delta_2^{m'},$$

where ν_2 and m' are as in (a). Next, S prints

$$\begin{matrix} \Delta_3 C'_{r+1} \\ \Delta_3 G_1', \end{matrix}$$

using the method of (a). Then S returns to (3) to continue the simulation.

If, in (a), (b), or (c), q_2 is in F then S accepts. Now S can surely write the initial A-configuration and initial "guess"

$$\begin{matrix} C_1' \\ G_1', \end{matrix}$$

where

$$C_1' = q_0 ¢ \upharpoonleft u\$(\Delta_1 \upharpoonleft B)^k \Delta_1 Z_0 \Delta_2^{2^{c_2 n}-(n+3k+4)}$$

and

$$G_1' = D_1^{2^{c_2 n}},$$

where B is a distinguished symbol treated as β.

Thus, by induction on the number of moves of A, S will accept $¢u\$$ if and only if A accepts $¢u\$$.

We now observe the following two facts:

(1) If L is an arbitrary language accepted by an arbitrary nonerasing $2^{c\alpha}$-tape-bounded DAPTM, then L is accepted by a $2^{c\alpha}$-tape-bounded DTM for the same constant c.

(2) In the proof of Theorem 2.1, S is nonerasing if A is nonerasing.

Observations (1) and (2) lead to

THEOREM 2.2. *For each $2^{c\alpha}$-tape-bounded* DTM M, *there exists a* NEDWSA *such that* $T(S) = T(M)$.

3. SIMULATION OF WSA BY $2^{c\alpha}$-TAPE-BOUNDED APTM

In this section we demonstrate the implication of statement (iv) from statement (iii) as asserted at the beginning of section two.

DEFINITION. For a given WSA S, a *state-input* of S is any member of

$$\bigcup_{n \geqslant 3} K \times ¢\Gamma^{n-2}\$ \times N_n .$$

Notation. Given $m \geqslant 1$, let

$$\overset{I}{\underset{m}{\vdash}}$$

be the relation defined as follows. For arbitrary configurations C and C', let

$$C \overset{I}{\underset{m}{\vdash}} C'$$

if there exist $C_1, \ldots, C_l$, with $C_i = (p_i, w_i, j_i, y_i, k_i)$ for each i, such that

(i) $C_1 = C$, $C_l = C'$,
(ii) $|\{k_i \mid k_i = 1\}| \leqslant m$, and
(iii) for each i, $i < l$, $C_i \vdash C_{i+1}$ and $k_i \geqslant 1$.

Let

$$C \overset{I}{\underset{+}{\vdash}} C',$$

if

$$C \overset{I}{\underset{m}{\vdash}} C'$$

for some integer $m \geqslant 1$.

Intuitively,

$$\overset{I}{\underset{m}{\vdash}}$$

relates the first and last configurations of an S computation in which (α) the stack head in each configuration, except possibly the last, is in the interior of the stack; and (β) throughout the total computation, the stack head scans position 1 at most m times.

Note that $C = C'$ if $k_1 = k_2 = m = 1$.

Notation. For each integer $m \geqslant 1$ and each S-configuration $(q, w, j, yZ, 1)$, with Z in Γ, let $R_m(q, w, j, yZ)$, written R_m, denote the set

$$\{(p, v, k) \mid (q, w, j, yz, 1) \overset{I}{\underset{m}{\vdash}} (p, v, k, yz, 1)\}.$$

Intuitively, R_m contains each state-input arising from the following computation. S starts in configuration $(q, w, j, yz, 1)$, always stays in the stack interior, reads position 1 at most m times, and ends in configuration $(p, v, k, yz, 1)$.

DEFINITION. Let $\mathscr{A}$ be a new symbol. Given S, w, and y in Γ^+, with $|w| = n$, the *transition matrix* $\mathscr{M}_{w,y}$ (or $\mathscr{M}_{S,w,y}$ when S is to be emphasized) is the function from $K \times N_n$ into the subsets of

$$\{\mathscr{A}\} \cup (K \times ¢\Gamma^{n-2}\$ \times N_n)$$

defined as follows for each (q, j) in $K \times N_n$:

(1) $\mathscr{M}_{w,y}(q, j) = \{\mathscr{A}\}$ if

$$(q, w, j, y, 1) \overset{I}{\underset{+}{\vdash}} (p, v, r, y, t),$$

with p in F, and $(q, w, j, y, 1)$ is an S-configuration.

(2) If (1) does not apply and $(q, w, j, y, 1)$ is an S-configuration, then

$$\mathscr{M}_{w,y}(q, j) = \{(p, v, k) \mid (q, w, j, y, 1) \underset{m}{\overset{I}{\vdash}} (p, v, k, y, 0)\}.$$

(3) If $(q, w, j, y, 1)$ is not an S-configuration, then $\mathscr{M}_{w,y}(q, j) = \phi$.

We shall only be concerned with $\mathscr{M}_{w,y}$ in which $(q, w, j, y, 1)$ is an S-configuration. Note that S accepts in (1).

Agreement. Hereafter in section four, w denotes a given word in $¢\Gamma^+\$$ and $n = |w| = |¢u\$| \geqslant 3$, where $¢u\$$ is the initial input to S. Sometimes $a_1 \cdots a_n$ is written in place of w. y denotes a given word in Γ^*.

Notation. Let $g = |\Gamma|$ and $s = |K|$. Let $\leqslant$ be a lexicographical order on $¢\Gamma^{n-2}\$$. Let $M_{w,y} = \{\mathscr{M}_{w,y}(q, j) \mid (q, j) \text{ in } K \times N_n\}$.

DEFINITION. The set

$$B_{S,n,y} = \{M_{w,y} \mid w \text{ in } ¢\Gamma^{n-2}\$\},$$

indexed by $\leqslant$ on the index w, is called a *block*.

We shall frequently write B_y instead of $B_{S,n,y}$ when S and n are understood.

Intuitively, given B_y and any top S-configuration, then A can "simulate" S for the case when S moves into its stack. That is, given that

$$(q_1, w_1, j_1, y, 0) \underset{S}{\vdash} (q_2, w_2, j_2, y, 1),$$

and given B_y, then to determine the future of S, A needs the element $M_{w_2,y}$, of B_y.

In what follows we shall refer to several common words ("encoded forms").

DEFINITION. For each m, 1 and each (q, j) in $K \times N_n$, the words $\mu(K \times N_n)$, $\mu(K \times ¢\Gamma^{n-2}\$ \times N_n)$, $\mu(R_m)$, $\mu(\mathscr{M}_{w,y}(q, j))$, $\mu(M_{w,y})$, and $\mu(B_y)$ are called the *encoded forms* of, respectively, $K \times N_n$, $K \times ¢\Gamma^{n-2}\$ \times N_n$, R_m, $\mathscr{M}_{w,y}(q, j)$, $M_{w,y}$, and B_y.

Notation. Let e_K be the enumeration $q_1, \ldots, q_s$ of the elements of K. Let

$$w_1', \ldots, w'_{g^{n-2}}$$

be the words of Γ^{n-2} in some order. For each i, let $w_i = ¢w_i'\$$. Let $\Delta_1, \ldots, \Delta_0$ be seven new symbols.

We now assemble all the necessary encoded forms in the

Notation. Let

$$\mu(K \times N_n) = \Delta_1 \nu(q_1, 1) \nu(q_1, 2) \cdots \nu(q_s, n) \Delta_1,$$

where

$$\nu(q_i, j) = q_i \Delta_5^j \Delta_6^{n-j}, \qquad 1 \leqslant i \leqslant s, \quad 1 \leqslant j \leqslant n.$$

Let

$$\mu(K \times ¢\Gamma^{n-2}\$ \times N_n) = \Delta_2 \nu_1(q_1, w_1, 1)\nu_1(q_1, w_1, 2) \cdots \nu_1(q_s, w_{g^{n-2}}, n)\,\Delta_2,$$

where

$$\nu_1(q_i, w_l, j) = q_i w_l\, \Delta_5^{\,j}\, \Delta_6^{n-j}, \qquad 1 \leqslant i \leqslant s, \quad 1 \leqslant l \leqslant g^{n-2}, \quad 1 \leqslant j \leqslant n.$$

Let $\mu(R_m(q, w, j, yz))$, abbreviated $\mu(R_m)$ denote the word

$$\mu(R_m) = \Delta_2 \nu_2(q_1, w_1, 1)\nu_2(q_1, w_1, 2) \cdots \nu_2(q_s, w_{g^{n-2}}, n)\,\Delta_2,$$

where

$$\nu_2(q_i, w_l, k) = {}_1(q_i, w_l, k)$$

if (q_i, w_l, k) is in R_m and

$$\nu_2(q_i, w_l, k) = \Delta_6^{2n+1}$$

otherwise.

For given y in Γ^+ and (q, j) in $K \times N_n$, let

$$\mu(\mathscr{M}_{w,y}(q, j)) = \nu_1(q, w, j) Y \nu_3(q_1, w_1, 1)\nu_3(q_1, w_1, 2) \cdots \nu_3(q_s, w_{g^{n-2}}, n),$$

where

(α) $Y = \mathscr{A}$ if $\mathscr{M}_{w,y}(q, j) = \{\mathscr{A}\}$, and $Y = \Delta_7$ otherwise
(β) $\nu_3(q_i, w_l, k) = \nu_1(q_i, w_l, k)$ if $Y = \Delta_7$ and (q_i, w_l, k) is in $\mathscr{M}_{w,y}(q, j)$, and

$$\nu_3(q_i, w_l, k) = \Delta_6^{2n+1}$$

otherwise.

For y in Γ^+, let $\mu(M_{w)y})$ denote the word

$$\Delta_4\, \Delta_4 \mu(\mathscr{M}_{w,y}(q_1, 1))\, \Delta_4 \mu(\mathscr{M}_{w,y}(q_1, 2))\, \Delta_4 \cdots \mu(\mathscr{M}_{w,y}(q_s, n))\, \Delta_4\, \Delta_4$$

For $y = \epsilon$, let $\mu(M_{w,y}) = \beta$, a distinguished symbol denoting the blank symbol.

For each y in Γ^+, let $\mu(B_y)$ denote the word

$$\Delta_3\, \Delta_3 \mu(M_{w_1}, y)\, \Delta_3 \mu(M_{w_2}, y)\, \Delta_3 \cdots \Delta_3 \mu(M_{w_{g^{n-2}}}, y)\, \Delta_3\, \Delta_3$$

For $y = \epsilon$, let $\mu(B_y) = \beta$.

From the form of $\mu(B_y)$ it is easy to derive a positive integer C_2 such that

$$|\mu(B_y)| < 2^{c_2 n}.$$

We now turn to the simulation of a WSA S by a $2^{c_2 \alpha}$-tape-bounded APTM A. For ease of presentation and comprehension, this is done by a sequence of lemmas, each of which modifies a construction given in a previous lemma. Since the pdt of the APTM A is not required until the final construction, the preliminary lemmas will refer only to the input tape and the work tapes of A. For simplicity, each encoded form required in the procedure is stored on a separate work tape.

Notation. Let T be a 10-tape TM and

$$\mathscr{C} = (p, u_0 \upharpoonright v_0, \ldots, u_{10} \upharpoonright v_{10})$$

and

$$\mathscr{C}' = (p', u_0' \upharpoonright v_0', \ldots, u_{10}' \upharpoonright v_{10}')$$

be T-configurations such that

$$\mathscr{C} \overset{*}{\vdash} \mathscr{C}'.$$

For each i, $0 \leqslant i \leqslant 10$, such that

$$u_i' \upharpoonright v_i' = u_i \upharpoonright v_i$$

(even though the i-th storage tape may have been altered and then reset during the computation), let

$$(i) = u_i' \upharpoonright v_i'.$$

The first lemma shows how A initially computes the words $\mu(K \times N_n)$ and

$$\mu(K \times ¢\Gamma^{n-2}\$ \times N_n).$$

LEMMA 3.1. *For each K, Σ, and Γ, there exists a positive integer C_1 and a $2^{C_1 \alpha}$-tape-bounded* TM $T_1 = (K_1, \Sigma, W_1, \delta_1, p_1, F_1, 10)$, *with W_1 containing Γ and with two distinguished states q_1 and q_2 in K_1, satisfying the following: For each word w in $¢\Gamma^+\$$, $w = a_1 \cdots a_n$, with $a_1 = ¢$ and $a_n = \$$, j_0 in N_n, and u in Σ^+,*

$$(q_1, \upharpoonright ¢u\$, \upharpoonright \beta, \upharpoonright \beta, a_1 \cdots \upharpoonright a_{j_0} \cdots a_n, \upharpoonright \beta \cdots \upharpoonright \beta)$$

$$\overset{*}{\underset{T_1}{\vdash}} (q_2, (0), (1), (2), (3), u_4' \upharpoonright v_4', (5), (6), \upharpoonright \mu(K \times N_n),$$

$$\upharpoonright \mu(K \times ¢\Gamma^{n-2}\$ \times N_n), {}^{(1)}u_9' \upharpoonright v_9', (10)),$$

where $u_4' \upharpoonright v_4'$ and $u_9' \upharpoonright v_9'$ are in H_2.[8]

Proof. We omit the straightforward proof.[9]

[8] Recall that

$$H_2 = \upharpoonright \beta W_\beta^* \cup W_\beta^* \upharpoonright \beta \cup W_\beta^* \upharpoonright W_\beta^+.$$

[9] See [7] for details.

Agreement. Hereafter in Section 4, we shall call the input tape of A tape 0 and denote the contents of tape 0 by $u_0 v_0 = ¢u\$$.

In Theorem 3.1, A must simulate S when S extends its stack, that is, when S prints some symbol Z at the top. A indirectly simulates S by, among other things, computing $\mu(B_{yz})$ from $\mu(B_y)$ where y is the contents of the stack. This portion of the procedure is developed over the next three lemmas.

Briefly, to compute $\mu(B_{yz})$, A computes $\mu(M_{w,yz})$ for each w in $¢\Gamma^{n-2}\$$. In turn, to compute $\mu(M_{w,yz})$, A computes $\mu(\mathcal{M}_{w,yz}(q,j))$ for each (q,j) in $K \times N_n$. In computing $\mu(\mathcal{M}_{w,yz}(q,j))$, A first computes $\mu(R\rho(q,w,j,yZ))$ where $\rho = sng^{n-2}$. Formally, we state this latter task as

LEMMA 3.2. *For each* WSA S, *there exists a positive integer* c_2 *and a* $2^{c_2\alpha}$*-tape-bounded* TM $T_2 = (K_2, \Sigma, W_2, \delta_2, p_2, F_2, 10)$, *with* W_2 *containing* Γ *and with two distinguished states* q_1 *and* q_2, *satisfying the following: For each word* w *in* $¢\Gamma^+\$$, $w = a_1 \cdots a_n$, *with* $a_1 = ¢$ *and* $a_n = \$$, $\nu(q,j)$ *in* $\nu(K \times N_n)$, j_0 *in* N_n, u *in* Σ^+, y *in* Γ^*, *and* Z *in* Γ,

$$\begin{aligned}&(q_1, \upharpoonleft ¢u\$, \upharpoonleft \mu(B_y), u_2 \upharpoonleft v_2, a_1 \cdots \upharpoonleft a_{j_0} \cdots a_n, u_4 \upharpoonleft v_4, u_5 \upharpoonleft v_5, u_6 \upharpoonleft v_6,\\ &\qquad \Delta_1 \nu(q_1, 1) \cdots \upharpoonleft \nu(q,j) \cdots \nu(q_s, n)\, \Delta_1, \upharpoonleft \mu(K \times ¢\Gamma^{n-2}\$ \times N_n),\\ &\qquad u_9 \upharpoonleft v_9, u_{10} \upharpoonleft v_{10})\\ &\underset{T_2}{\overset{*}{\vdash}} (q_2, 0, 1, 2, 3, u_4' \upharpoonleft v_4', u_5' \upharpoonleft v_5', \upharpoonleft \mu(R(q,w,j,yZ)\\ &\qquad \Delta_1 \nu(q_1, 1) \cdots \nu(q,j) \upharpoonleft \cdots \nu(q_s, n)\, \Delta_1, 8, 9, 10),\end{aligned}$$

where

$$u_2 \upharpoonleft v_2, u_6 \upharpoonleft v_6, u_9 \upharpoonleft v_9, u_{10} \upharpoonleft v_{10}, u_i \upharpoonleft v_i$$

and

$$u_i' \upharpoonleft v_i'$$

are in H_2 *for* $i = 4, 5$,

$$\max\{|u_2 v_2|, |u_4 v_4|, |u_5 v_5|, |u_6 v_6|, |u_9 v_9|, |u_{10} v_{10}|\} < 2^{c_2 n},$$

and

$$\rho = sng^{n-2}$$

Proof. The sequence of sets[10] $R_1, \ldots, R_\rho$ form an increasing chain of sets with the property that $R_i = R_{i+1}$ for some i, then $R_{j+1} = R_i$ for all $j \geqslant 1$. Since there are at most $\rho = sng^{n-2}$ distinct state-inputs for S, there exists a positive integer i_0, $1 \leqslant i_0 \leqslant \rho$ such that

$$R_{i_0} = R_{i_0+1} = \cdots = R\rho.$$

[10] Recall that for $m \geqslant 1$,

$$R_m = \{(p, v, k) \mid (q, w, j, yZ, 1) \vdash^{I}_{m} (p, v, k, yZ, 1)\}.$$

For each integer $m < i_0$, T_2 computes $\mu(R_{m+1})$, ultimately computing

$$\mu R_{i_0} = \mu(R\rho).$$

The procedure is such that T_2 has to remember at most

$$\max\{|\mu(R\rho)|, |\mu(B_y)|\} < 2^{c_2 n}$$

cells. Thus T_2 is a $2^{c_2 \alpha}$-tape-bounded TM.

We informally outline the operation of T_2. The TM T_2 stores the 8-tuple S together with Z in its fsc. For each m,[11] $1 \leqslant m < i_0$, T_2 computes (see next paragraph) $\mu(R_{m+1})$ on tape 6 using only $\mu(R_m)$ on tape 5, $\mu(B_y)$ on tape 1, and Z. Initially, T_2 prints $\mu(R_1) = (\mu\{(q, w, j)\})$ on tape 5. Upon computing $\mu(R_{m+1})$, T_2 checks if $\mu(R_m) = \mu(R_{m+1})$. If so [then $\mu(R_m) = \mu(R\rho)$ and $m = i_0$], T_2 enters the final configuration. Otherwise, T_2 replaces $\mu(R_m)$ by $\mu(R_{m+1})$ and m by $m + 1$ and returns again to compute the new $\mu(R_{m+1})$. In case $y = \epsilon$, T_2 computes $\mu(R_{m+1})$ using only $\mu(R_m)$ and Z. In this case, T_2 ignores one step in the procedure.

There remains to show how T_2 computes $\mu(R_{m+1})$. Let

$$\nu_2(q, w_t, k) \neq \Delta_6^{2n+1}$$

be a word in $\mu(R_m)$ such that $w_t = b_1 \cdots b_n$, and $\delta(q, b_k, Z)$ contains (q', b_k', d, X) for some X in $\{-1, 0, 1\}$. Let $x = b_1 \cdots b_{k-1} b_k' b_{k+1} \cdots b_n$. T_2 computes $\mu(R_{m+1})$ on tape 6 by executing the following steps for each

$$\nu_2(q, w_t, k) \neq \Delta_6^{2n+1}:$$

(1) $\nu_2(q, w_t, k)$ is copied[12] to tape 6;

(2) If $X = 0$ then $\nu_1(q', x, k + d)$ is copied[13] to tape 6;

(3) If $X = -1$ (impossible if $y = \epsilon$) then each $\nu_3(q'', w', l)$ is copied[14] to tape 6 from $\mu(\mathscr{M}_{x,y}(q', k + d))$ in $\mu(B_y)$, with

$$\nu_3(q'', w', l) \neq \Delta_6^{2n+1}.$$

The details for carrying out (1)–(3) are straightforward and are omitted.

In the next portion of the procedure, A computes $\mu(M_{w,yZ})$ by computing $\mu(\mathscr{M}_{w,yZ}(q, j))$ for each (q, j) in $K \times N_n$. A uses $\mu(R\rho(q, w, j, yZ), w, Z$ and $\mu(B_y)$ to accomplish this latter task.

[11] T_2 stores the integer m in unary form on tape 4.

[12] In the corresponding position as on tape 5.

[13] T_2 first locates the word on tape 8 to determine its position on tape 6.

[14] In the corresponding position to the right of Y in $\mathscr{M}_{x,y}(q', k + d)$.

LEMMA 3.3. *For each* WSA S, *there exists a positive integer* c_2 *and a* $2^{C_2\alpha}$*-tape-bounded* TM $T_3 = (K_3, \Sigma, W_3, \delta_3, p_3, F_3, 10)$, *with* W_3 *containing* Γ *and with two distinguished states* q_1 *and* q_2, *satisfying the following*: *For each word* w *in* $¢\Gamma^+\$$, $w = a_1 \cdots a_n$, *with* $a_1 = ¢$ *and* $a_n = \$$, Z *in* Γ, y *in* Γ^*, j_0 *in* N_n, *and* u *in* Σ^+,

$$(q_1, \upharpoonright ¢u\$, \upharpoonright \mu(B_y), u_2 \upharpoonright v_2, a_1 \cdots \upharpoonright a_{j_0} \cdots a_n, u_4 \upharpoonright v_4, u_5 \upharpoonright v_5, u_6 \upharpoonright v_6,$$
$$\upharpoonright \mu(K \times N_n), \upharpoonright \mu(K \times ¢\Gamma^{n-2}\$ \times N_n), u_9 \upharpoonright v_9, u_{10} \upharpoonright v_{10})$$
$$\overset{*}{\underset{T}{\vdash}} (q_2, ⓪, ①, \upharpoonright \mu(M_{w,yZ}), ③, u_4' \upharpoonright v_4', u_5' \upharpoonright v_5', u_6' \upharpoonright v_6', ⑦, ⑧, u_9' \upharpoonright v_9', ⑩),$$

where

$$u_2 \upharpoonright v_2, \qquad u_{10} \upharpoonright v_{10}, \qquad u_i \upharpoonright v_i$$

and

$$u_i' \upharpoonright v_i'$$

are in H_2 *for* i *in* $\{4, 3, 6, 9\}$, *and* $\max\{|u_i v_i|\} < 2^{c_2 n}$ *for* i *in* $\{2, 4, 5, 6, 9, 10\}$.

Proof. Let c_2 and the 10 tape TM T_2 be as given in Lemma 3.2. The work tapes of T_3 are those of T_2. The procedure is such that T_3 has to remember at most

$$|\mu(B_y)| < 2^{c_2 n}$$

cells. Thus T_3 is a $2^{C_2\alpha}$-tape-bounded TM. Intuitively, T_3 stores the 7-tuple T_2 together with S and Z in its fsc. T_3 computes $\mu(M_{w,yZ})$ by computing $\mu(\mathscr{M}_{w,yZ}(q,j))$ for each (q,j) in $K \times N_n$. To compute $\mu(\mathscr{M}_{w,yZ}(q,j))$, T_3 uses $\mu(R\rho(q, w, j, yZ))$ (first computing the latter by Lemma 3.2), S, and $\mu(B_y)$. In case $y = \epsilon$, T_3 computes $\mu(M_{w,Z})$ using only $\mu(R\rho(q, w, j, Z))$ and S. In this case, T_3 ignores one step (indicated below) in the procedure.

We now outline (steps (1)–(7)) the procedure for T_3. For each i, $1 \leqslant i \leqslant 5$, $7 \leqslant i \leqslant 9$, and d_2 in $\{-1, 0, 1\}$, let r_i and (r_6, d_2) be states in K_3. Let $\mathscr{C}_0$ be the start configuration stated in the lemma. For convenience, we assume that tapes 2 and 9 are initially blank. Thus,

$$\mathscr{C}_0 = (q_1, \upharpoonright ¢u\$, \upharpoonright \mu(B_y), \upharpoonright \beta, a_1 \cdots \upharpoonright a_{j_0} \cdots a_n, u_4 \upharpoonright v_4, \ldots, u_6 \upharpoonright v_6,$$
$$\upharpoonright \Delta_1 \nu(q_1, 1) \cdots \nu(q_s, n)\, \Delta_1, \upharpoonright \mu(K \times ¢\Gamma^{n-2}\$ \times N_n), \upharpoonright \beta, u_{10} \upharpoonright v_{10}).$$

(1) $\mathscr{C}_0 \vdash \mathscr{C}_1$, where

$$\mathscr{C}_1 = (r_1, ⓪, ①, \Delta_4 \upharpoonright \beta, ③, \ldots, ⑥, \Delta_1 \upharpoonright \nu(q_1, 1) \cdots \nu(q_s, n)\, \Delta_1, ⑧, ⑨, ⑩).$$

In steps (2)–(4), T_3 prepares to compute $\Delta_4 \mu(\mathscr{M}_{w,yZ}(q,j))$ on tape 2 for the next (first) $\nu(q, j)$ in $\nu(\times N_n)$. Thus, let[15] (q', j') be the element immediately preceding (q, j) in $K \times N_n$ in an ordering of $K \times N_n$ in an ordering of $K \times N_n$. Let

$$\lambda = \Delta_4 \mu(\mathscr{M}_{w,y} z(q_1, 1)) \Delta_4 \cdots \Delta_4 \mu(\mathscr{M}_{w,y} z(q', j')).$$

(2) $\mathscr{C}_1 \overset{*}{\vdash} \mathscr{C}_2$, where

$$\mathscr{C}_2 = (r_2, ⓪, ①, \Delta_4 \lambda \upharpoonright \beta, ③, \ldots, ⑥, \Delta_1 \nu(q_1, 1) \cdots$$
$$\upharpoonright \nu(q, j) \cdots (q_s, n) \Delta_1, ⑧, \upharpoonright \nu_1(q, w, j), ⑩).$$

(After printing ${}_1(q, w, j)$ on tape 9, T_3 begins a loop by entering r_2. Initially, $\nu(q, j) = \nu(q_1, 1)$.)

(3) $\mathscr{C}_2 \overset{*}{\vdash} \mathscr{C}_3$, where

$$\mathscr{C}_3 = (r_3, ⓪, ①, \Delta_4 \lambda \Delta_4 \upharpoonright \mu_e(\mathscr{M}_{w,y} z(q, j)), ③, \ldots, ⑩)$$

and

$$\mu_e(\mathscr{M}_{w,y} z(q, j)) = \nu_1(q, w, j) \Delta_7 \nu_4(q_1, w_1, 1) \cdots \nu_4(q_s, w_{g^{n-2}}, n)$$

each

$$\nu_4(q_i, w_t, k) = \Delta_6^{2n+1}.$$

(T_3 prints an "empty" copy of $\mu(\mathscr{M}_{w,y} z(q, j))$ on tape 2.)

(4) $\mathscr{C}_3 \overset{*}{\vdash} \mathscr{C}_4$, where

$$\mathscr{C}_4 = (r_4, ⓪, \ldots, ③, u_4' \upharpoonright v_4', u_5', \upharpoonright v_5', \upharpoonright \mu(R\rho), \Delta_1 \nu(q_1, 1) \cdots (q, j)$$
$$\upharpoonright \cdots \nu(q_s, n) \Delta_1, ⑧, ⑨, ⑩)$$

and the initial and final configurations are as in Lemma 3.2. (For the next (first) $\nu(q, j)$ in $K \times N_n$, with w, y, and Z fixed, T_3 computes $\mu(R\rho)$.)

The next step is executed for each

$$\nu_2(p, w_t, k) \neq \Delta_6^{2n+1}$$

in $\mu(R\rho)$.

(5) Let $\mu(\mathscr{M}'_{w,y} z(q, j))$ generically denote any word of the form

$$\nu_1(q, w, j) \Delta_7 \nu_4'(q_1, w_1, 1) \cdots \nu_4'(q_s, w_{g^{n-2}}, n),$$

where for each (q_i, w_t, k), either

$$\text{(i)} \quad \nu_4'(q_i, w_t, k) = \nu_4(q_i, w_t, k) \text{ or}$$

$$\text{(ii)} \quad \nu_4'(q_i, w_t, k) = q_i w_t \Delta_5{}^k \Delta_6^{n\ k}$$

[15] $(q', j') = \lambda = \epsilon$ if $(q, j) = (q_1, 1)$.

and $\nu_4'(q_i, w_t, k)$ has been inserted into $\mu(\mathscr{M}'_{w,yZ}(q,j))$ during the l-th pass, $0 \leqslant l \leqslant \rho$ of step 5. For $l = 0$, that is, initially,

$$\mu(\mathscr{M}'_{w,yZ}(q,j)) = \mu_e(\mathscr{M}_{w,yZ}(q,j)).$$

(a) $\mathscr{C}_4 \overset{*}{\vdash} \mathscr{C}_5$, where

$$\mathscr{C}_5 = (r_5, ⓪, ①, \Delta_4\lambda\, \Delta_4 \upharpoonright \mu(\mathscr{M}'_{w,yZ}(q,j)), ③,..., ⑤, \gamma_1 \upharpoonright \nu_2(p, w_t, k)\gamma_2, ⑦,..., ⑩), \nu_2(p, w_t, k)$$

is the next (first) subword in $\mu(R)$ such that

$$\nu_2(p, w_t, k) \neq \Delta_6^{2n+1},$$

and $\gamma_1\nu_2(p, w_t, k)\gamma_2 = \mu(R\rho)$.

(b) If p is in F, then $\mathscr{C}_5 \overset{*}{\vdash} \mathscr{C}_9$, where

$$\mathscr{C}_9 = (r_0, ⓪, ①, \Delta_4\lambda\, \Delta_4 \upharpoonright \nu_1(q, w, j)\mathscr{A}\nu_4(q_1, w_1, 1) \cdots \nu_4(q_s, w_{g^{n-2}}, n)\, ③,..., ⑤, \gamma_1\nu_2(p, w_t, k) \upharpoonright \gamma_2, ⑦, ⑧, \upharpoonright \Delta_6^{2n+1}, ⑩).$$

Then T_3 proceeds to step 7. (After setting $\mathscr{M}_{w,yZ}(q,j) = \{\mathscr{A}\}$, T_3 overprints tape 9 with Δ_6^{2n+1}.)

Next, let $w_t = b_1 \cdots b_n$, $\delta(p, b_k, Z)$ contain (q'', b_k', d_1, d_2), and

$$v = b_1 \cdots b_{k-1}b_k'b_{k+1} \cdots b_n.$$

(c) $\mathscr{C}_5 \overset{*}{\vdash} \mathscr{C}_6$, where

$$\mathscr{C}_6 = ((r_6, d_2), ⓪,..., ⑤, \gamma_1\nu_2(p, w_t, k) \upharpoonright \gamma_2, ⑦, ⑧, \upharpoonright \nu_1(q'', v, k + d_1), ⑩).$$

(After computing $\delta(p, b_k, Z)$, T_3 prints $\nu_1(q'', v, k + d_1)$ on tape 9 and enters state (r_6, d_2).)

Several cases arise. If q'' is not in F and $d_2 = 0$ then T_3 ignores this case since, by definition, $\nu_2(q'', v, k + d_1)$ would be a member of $\mu(R_{\rho+1}) = \mu(R_\rho)$. There remain three cases to consider, namely, q'' in F, q'' not in F and $d_2 = 1$, and q'' not in F and $d_2 = -1$.

(d) q'' is in F. Then $\mathscr{C}_6 \overset{*}{\vdash} \mathscr{C}_9$, where $\mathscr{C}_9$ is as in (5b).

(e) q'' is not in F and $d_2 = 1$. Then using

$$\mu(K \times \cent\Gamma^{n-2}\$ \times N_n),$$

T_3 copies $\nu_1(q'', v, k + d_1)$ from tape 9 to tape 2. That is, T_3 inserts $\nu_1(q'', v, k + d_1)$ into $\mu(\mathscr{M}'_{w,yZ}(q,j))$ (since $(q'', v, k + d_1)$ is a member of $\mathscr{M}_{w,yZ}(q,j)$), using tape 8 to determine its position. Then, T_3 proceeds to step 6.

(f) q'' is not in F and $d_2 = 1$. (This is not possible if $y = \epsilon$.) Then T_3 reads the symbol Y in $\mu(\mathscr{M}_{v,y}(q'', k + d_1))$ on tape 1 (obviously, T_3 can find $\mu(\mathscr{M}_{v,y}(q'', k + d_1))$ in $\mu(_Y)$ on tape 1). If $Y = \mathscr{A}$ then T_3 overprints the symbol Δ_7 in $\mu(\mathscr{M}'_{w,yZ}(q, j))$ with $\mathscr{A}$ and goes to step 7. If[16] $Y \neq \mathscr{A}$, T_3 goes to step 6.

(6) If $\nu_2(p, w_t, k)$ is not the last subword in $\mu(R_p)$ such that $\nu_2(p, w_t, k) \neq \Delta_6^{2n+1}$, then T_3 goes to step 5. If $\nu_2(p, w_t, k)$ is the last, T_3 goes to step 7.

(7) If the symbol scanned on tape 7 (see step 4) is not Δ_1, then T_3 returns to step 2. Otherwise, T_3 enters the final configuration stated in the lemma.

The next lemma says that A can compute $\mu(B_{YZ})$ given $\mu(B_Y)$ and Z. To accomplish this, A computes $\mu(M_{w,yZ})$ for each w in $¢\Gamma n - 2_\$$.

LEMMA 3.4. *For each* WSA S, *there exists a positive integer* C_2 *and a* $2^{C_2\alpha}$*-tape-bounded* TM $T_4 = (K_4, \Sigma, W_4, \delta_4, p_4, F_4, 10)$, *with* W_4 *containing* Γ *and with two distinguished states* q_1 *and* q_2, *satisfying the following*: *For each word* w *in* $¢\Gamma^+\$$, $w = a_1 \cdots a_n$, *with* $a_1 = ¢$ *and* $a_n = \$$, Z *in* Γ, y *in* *, j_0 *in* N_n, *and* u *in* Σ^+,

$$(q_1, \upharpoonright ¢u\$, \upharpoonright \mu(B_Y), u_2 \upharpoonright v_2, a_1 \cdots \upharpoonright a_{j_0} \cdots a_n, u_4 \upharpoonright v_4, u_5 \upharpoonright v_5, u_6 \upharpoonright v_6,$$
$$\upharpoonright \mu(K \times N_n), \upharpoonright \mu(K \times ¢\Gamma^{n-2}\$ \times N_n), u_9 \upharpoonright v_9, u_{10} \upharpoonright v_{10})$$
$$\overset{*}{\underset{T_4}{\vdash}} (q_2, ⓪, ①, u_2' \upharpoonright v_2', ③, u_4' \upharpoonright v_4', u_5' \upharpoonright v_5', u_6' \upharpoonright v_6', ⑦, ⑧, u_9' \upharpoonright v_9', \upharpoonright \mu(B_{YZ})),$$

where

$$u_{10} \upharpoonright v_{10}, \qquad u_i \upharpoonright v_i$$

and

$$u_i' \upharpoonright v_i'$$

are in H_2 *for* i *in* $\{2, 4, 5, 6, 9\}$, *and*

$$\max\{| u_i v_i |\} < 2^{c_2 n}$$

for i *in* $\{2, 4, 5, 6, 9, 10\}$.

Proof. Let c_2 and the 10 tape TM T_3 be as given in Lemma 3.3. The work tapes of T_4 are those of T_3. The procedure is such that each work tape of T_4 has to store at most

$$| \mu(B_Y)| < 2^{c_2 n}$$

cells. Thus, T_4 is a $2^{C_2\alpha}$-tape-bounded TM. Intuitively, T_4 stores the 7-tuple T_3 together with S and Z in its fsc. If $y \neq \epsilon$ then the initial contents of tapes 2 and 10

[16] All other values in $\mathscr{M}_{v,y}(q'', k + d_1)$ can be ignored. In particular, any state-inputs must, by definition be in $R\rho$.

are ignored, that is, overprinted with a symbol treated as β. It thus suffices to consider only the case where

$$u_2 \upharpoonright v_2 = u_{10} \upharpoonright v_{10} = \upharpoonright \beta,$$

that is, the case $y = \epsilon$.

For each j, $1 \leqslant j \leqslant g^{n-2}$, T_4 computes w_j on tape 9, T_4 computes $\mu(M_{w_j}, Z)$ on tape 2 (using Lemma 3.3), and then T_4 prints $\mu(M_{w_j}, Z)\, \Delta_3$ on tape 10. As usual, we omit the straightforward details.

THEOREM 3.1. *For each* WSA S *there exists a positive integer* C_2 *and a* $2^{C_2\alpha}$*-tape-bounded* APTM A *such that* $T(A) = T(S)$.

Proof. Let u be in $T(S)$ and $n = |\, ¢u\$\, | \geqslant 3$. The pdt of A is hereafter referred to as tape 11. Let C_2 be the positive integer and T_4 the 10 tape TM as given in Lemma 3.4. The APTM A will include the fsc and the 10 work tapes of T_4 together with tape 11. A stores the 8-tuple S in its fsc.

The definition of $f(\alpha)$-APTM together with the fact that T_4 is a $2^{C_2\alpha}$-tape-bounded TM implies that A is a $2^{C_2\alpha}$-tape-bounded APTM.

We now show how A "simulates" S on $¢u\$$. Let Π_α' be a sequence of S-configurations $C_1',..., C'$ where

$$C_1' = (q_0, ¢u\$, 1, Z_0, 0), \qquad C_1' = (p_i, w_i, j_i, y_i, k_i), \qquad 1 \leqslant i \leqslant \alpha,$$

and

$$C_i' \vdash_{\overline{S}} C_{i+1}'$$

for $1 \leqslant i \leqslant \alpha$. Let Π_r be the subsequence $C_1,..., C_r$, $1 \leqslant r \leqslant \alpha$, of top S-configurations in Π_α'. That is, C_l is the l-th $1 \leqslant l \leqslant r$, member of Π_r if and only if $k_l = 0$ and $|\{C_j' \mid k_j = 0, 1 \leqslant j \leqslant l\}| = l$.

Let $C_r = (q, w, j, y, 0)$, where $y = Y_t \cdots Y_1$, $t \geqslant 0$. Let

$$\zeta = Y_t\mu(B_{Y_t})Y_{t-1}\mu(B_{Y_tY_{t-1}}) \cdots Y_2\mu(B_{Y_t\cdots Y_2})Y_1\mu(B_Y)$$

for $t > 0$ and let $\zeta = \epsilon$ for $t = 0$. Corresponding to C_r, let $\mathscr{C}_r$ be the A-configuration

$$((q, Y_1), \upharpoonright ¢u\$, \upharpoonright \mu(B_Y), u_2 \upharpoonright v_2, a_1 \cdots \upharpoonright a_j \cdots a_n, u_4 \upharpoonright v_4, ..., u_{10} \upharpoonright v_{10}, \zeta),$$

where (q, Y_1) is a new state, $u_2 \upharpoonright v_2$ is in H_2, $u_m \upharpoonright v_m$ is in H_2 for $4 \leqslant m \leqslant 10$,

$$\max\{|\, u_2v_2 \,|, |\, u_mv_m \,| \; 4 \leqslant m \leqslant 10\} < 2^{c_2 n},$$

and as usual, $w = a_1 \cdots a_n$.

The APTM A indirectly "simulates" S in the following sense. When S is at the top configuration C_r, A is at the corresponding configuration $\mathscr{C}_r$. Also, A codes each component of C_r and the topmost symbol Y_1 of the stack of S as follows: q and Y_1 are in the fsc, w and position j are represented by

$$a_1 \cdots \upharpoonright a_j \cdots a_n$$

on tape 3, and y is represented both in $\mu(B_Y)$ on tape 1 and in ζ on tape 11.

We will show that, given the top S-configuration C_r, if S can accept u before or upon returning its stack head to the top in some configuration C_{r+1}, A will accept u. If on the other hand, without accepting, S chooses to enter a new top-configuration C_{r+1}, A will enter a configuration $\mathscr{C}_{r+1}$, simulating S in the above sense. Using a straightforward induction on r, the number of top S-configurations, it will then follow that $T(S) = T(A)$, proving our assertion.

Initially, A is in

$$\mathscr{C}_0 = (p_0, \upharpoonright ¢u\$, \upharpoonright \beta, \ldots, \upharpoonright \beta, Z_0)$$

and S is in $C_1 = (q_0, ¢u\$, 1, Z_0, 0)$. First, $\mathscr{C}_0 \overset{*}{\vdash}_A \mathscr{C}_1$, where

$$\mathscr{C}_1 = ((q_0, Z_0), \upharpoonright ¢u\$, \upharpoonright \mu(B_{Z_0}), u_2 \upharpoonright v_2, \upharpoonright ¢u\$, u_4 \upharpoonright v_4, \ldots, u_{10} \upharpoonright v_{10}, (Z_0\mu B_{Z_0}))$$

by copying the input to tape 3 and using $y = \epsilon$ in Lemma 3.4 to obtain the encoded block $\mu(B_{Z_0})$.

To see how A can simulate S from C_r to C_{r+1}, let (p, Z, i) denote a new state for each p in K, Z in Γ, and $i = 2, 3$. Let S and A be in the configurations, respectively, C_r and $\mathscr{C}_r$. A nondeterministically chooses a member of $\delta(q, a_j, Y_1)$ from its fsc (a_j is obtained from tape 3). There are four[17] possibilities for the fourth component of $\delta_\beta(q, a_j, Y_1)$.

(1) $\delta_\beta(q, a_j, Y_1)$ contains (q_1', a_j', d, θ). If q_1' is in F, then A also accepts. Otherwise,

$$\mathscr{C}_r \vdash_{\overline{A}} \mathscr{C}_{r+1}$$

where

$$\mathscr{C}_{r+1} = ((q_1', Y_1), (0), (1), (2), a_1 \cdots a_j' \upharpoonright a_{j+d} \cdots a_n, (4), \ldots, (10), \zeta)$$

(2) $\delta_\beta(q, a_j, Y_1)$ contains $(q_1', a_j', d, 1)$. Then S enters its stack in configuration $(q_1', x, j + d, y, 1)$. If S can accept before or upon returning its stack head to the top, then A will be made to accept. Otherwise, for each next possible top S-configuration of the form $C_{r+1} = (q_2', w_2, j_2, y, 0)$, A will simulate S by entering an associated configuration of the form

$$((q_2', Y_1), (0), (1), (2), b_1 \cdots \upharpoonright b_{j_2} \cdots b_n, (4), \ldots, (10), \zeta),$$

[17] Recall that δ_β is a function from $K \times (\Gamma \cup \{¢, \$\}) \times \Gamma$ into the subsets of

$$K \times (\Gamma \cup \{¢, \$\}) \times \{-1, 0, 1\} \times (\Gamma \cup \{\theta, I, E\}).$$

where

$$b_1 \cdots b_n = w_2 .$$

First, $\mathscr{C}_r \vdash_A \mathscr{C}_2'$, where

$$((q_1', Y_1, 2), ⓪, ①, ②, a_1 \cdots a_j' \upharpoonright a_{j+d} \cdots a_n, ④,..., ⑩, \zeta).$$

Next, to determine all possibilities for S, with S in configuration $(q_1', x, j + d, y, 1)$, A uses $\mu(\mathscr{M}_{x,y}(q_1', j + d))$ in $\mu(B_Y)$ on tape 1 (obviously, A can find $\mu(\mathscr{M}_{x,y}(q_1', j + d))$). Two cases arise for $\mu(\mathscr{M}_{x,y}(q_1', j + d))$:

(a) $Y = \mathscr{A}$, so that S accepts before or upon returning to a top-configuration. Then A is to accept.

(b) $Y = \Delta_7$. Then S cannot accept before or upon reaching the top of stack. S may or may not return to a top-configuration. In this case, A nondeterministically chooses some state input (q_2', w_2, j_2) to the right of Y in $\mu(\mathscr{M}_{x,y}(q_1', j + d))$. Thus, in case (b),

$$\mathscr{C}_2' \overset{*}{\underset{A}{\vdash}} \mathscr{C}_{r+1}$$

(After choosing (q_2', w_2, j_2), A copies w_2 to tape 3, moving tape 3 head to position j_2. Then A goes to state (q_2', Y_1).)

(3) $\delta_\beta(q, a_j, Y_1)$ contains (q_1', a_j', d, Z) where Z is in Γ. Then $C_r \vdash_S C_{r+1} = (q_1', x, j + d, yZ, 0)$. If q_1' is in F, then A also accepts. Otherwise, A simulates S as follows: First

$$\mathscr{C}_r \underset{A}{\vdash} \mathscr{C}_3',$$

where

$$\mathscr{C}_3' = ((q_1', Z, 3), ⓪, ①, ②, a_1 \cdots a_j' \upharpoonright a_{j+d} \cdots a_n, 4,..., ⑩, \zeta).$$

Next, holding tape 11 fixed, using the other 11 tapes and the states as in Lemma 3.4,

$$\mathscr{C}_3' \overset{*}{\underset{A}{\vdash}} \mathscr{C}_4',$$

where

$$\mathscr{C}_4' = (\overline{(q_1', Z)}, ⓪, ①, u_2' \upharpoonright v_2', ③, u_4' \upharpoonright v_4',..., u_6' \upharpoonright v_6', ⑦, ⑧, u_9' \upharpoonright u_9', \upharpoonright \mu(B_{YZ}), \zeta).$$

Finally,

$$\mathscr{C}_4' \overset{*}{\underset{A}{\vdash}} \mathscr{C}_{r+1},$$

where

$$\mathscr{C}_{r+1} = ((q_1', Z), ⓪,..., ⑩, \zeta Z(B_{YZ})).$$

(The $\mu(B_{YZ})$ is obtained from tape 10.)

(4) $\delta_\beta(q, a_j, Y_1)$ contains[18] (q_1', a_j', d, E). Then

$$C_r \vdash_{\overline{S}} C_{r+1} = (q_1', x, j + d, Y_t \cdots Y_2, 0)^{19}.$$

If q_1' is in F, then A also accepts. Otherwise,

$$\mathscr{C}_r \overset{*}{\vdash} \mathscr{C}_{r+1},$$

where

$$\mathscr{C}_{r+1} = ((q_1', Y_2), ⓪, \upharpoonright \mu B_{Y_t \cdots Y_2}, ②, a_1 \cdots a_j' \upharpoonright a_{j+d} \cdots a_n, ④, \ldots, ⑩, \rho' Y_2 \mu(B_{Y_t \cdots Y_2}))$$

with

$$\rho' = Y_t \mu B_{Y_t} Y_{t-1} \mu B_{Y_t Y_{t-1}} \cdots Y_3 \mu B_{Y_t \cdots Y_3}$$

for $t \geqslant 3$ and $\rho' = \epsilon$ for $t \leqslant 2$.

Thus, if

$$C_r \vdash_{\overline{S}} C_{r+1} = (q_1', x, j + d, Y_t \cdots Y_2, 0),$$

and S accepts, then A accepts. Otherwise, A simulates S by entering the associated configuration $\mathscr{C}_{r+1}$.

It follows that A will accept u if and only if S accepts u during any computation, and thus $T(A) = T(S)$.

Observe that in the proof of Theorem 3.1, A is nonerasing if S is nonerasing. By the comment made after the definition of $f(\alpha)$-APTM at the end of section one, if L is an arbitrary language accepted by an arbitrary nonerasing $2^{c\alpha}$-APTM, then L is accepted by a $2^{c\alpha}$-TM, for the same constant c. Hence there immediately follows:

THEOREM 3.2. *For each* NEWSA S, *there exists a positive integer* C_2 *and a* $2^{C_2\alpha}$-*tape-bounded* TM M *such that* $T(M) = T(S)$.

4. Main Results and Consequences

Using the results established in Sections 2 and 3, we now prove our main results. We also exhibit some AFL properties of $\mathscr{L}_{\mathrm{WSA}}$ and $\mathscr{L}_{\mathrm{NEWSA}}$.

[18] Note that this is the only occurrence of erasing in the construction of A.
[19] If $t = 1$ then $Y_t \cdots Y_2 = \epsilon$.

DEFINITION. Let $A = (K, \Sigma, W, \delta, q_0, F, k)$ be a DTM and let w be in $T(A)$. If $\mathscr{C}_0, \ldots, \mathscr{C}_m$ is a sequence of configurations such that

$$\mathscr{C}_0 = (q_0, \text{¢} \upharpoonright w\$, \upharpoonright \beta, \ldots, \upharpoonright \beta), \qquad \mathscr{C}_i \vdash \mathscr{C}_{i+1}$$

for each i, $0 \leqslant i < m$, the state-component of $\mathscr{C}_m$ is in F, and for each i, $0 \leqslant i < m$, the state-component of $\mathscr{C}_i$ is not in F, then $\mathscr{C}_0, \ldots, \mathscr{C}_m$ is called a *shortest accepting computation on w.*

Since A is deterministic, each word w in $T(A)$ has a unique shortest accepting computation.

DEFINITION. Let f be a nondecreasing function. A DTM $A = (K, \Sigma, W, \delta, q_0, F, k)$ is called a $f(\alpha)$-*time-bounded* DTM if it has the following property: For each word w in $T(A)$, if $\mathscr{C}_0, \ldots, \mathscr{C}_r$ is the shortest accepting computation on w, then $r \leqslant f(|w|)$.

Notation. For each nondecreasing function $f(\alpha)$, let

$$\mathscr{L}^{\mathrm{TIME}}_{f(\alpha)-\mathrm{DTM}}$$

denote the family of languages accepted by some $f(\alpha)$-time-bounded DTM.

Notation. For each nondecreasing function $f(\alpha)$, let Two $(f(\alpha)) = 2^{f(\alpha)}$.

Agreement. In this section, the symbol c (subscripted or not) always denotes a positive integer.

We need the following result from [2].

LEMMA 4.1[2].

$$\mathscr{L}_{f(\alpha)-\mathrm{DAPTM}} = \mathscr{L}_{f(\alpha)-\mathrm{APTM}} = \bigcup_{c_1 \geqslant 1} \mathscr{L}^{\mathrm{TIME}}_{\mathrm{Two}(c_1 f(\alpha))-\mathrm{DTM}}$$

for each nondecreasing function $f(n) \geqslant \log_2 n$.

COROLLARY 4.1.

$$\mathscr{L}_{\mathrm{Two}(c\alpha)-\mathrm{DAPTM}} = \mathscr{L}_{\mathrm{Two}(c\alpha)-\mathrm{APTM}} = \bigcup_{c_1 \geqslant 1} \mathscr{L}^{\mathrm{TIME}}_{\mathrm{Two}(c_1\,\mathrm{TWO}(c\alpha))-\mathrm{DTM}}$$

for each c.

Corollary 4.1, together with Theorems 2.1 and 3.1, leads to our first result.

THEOREM 4.1.

$$\mathscr{L}_{\mathrm{WSA}} = \mathscr{L}_{\mathrm{DWSA}} = \bigcup_{c \geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)-\mathrm{APTM}}\,.$$

Proof.

$$\begin{aligned}
\bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-DAPTM}} &\subseteq \mathscr{L}_{\mathrm{DWSA}}, \text{ by Theorem 2.1} \\
&\subseteq \mathscr{L}_{\mathrm{WSA}} \\
&\subseteq \bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-APTM}}, \text{ by Theorem 3.1} \\
&= \bigcup_{c0n1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-DAPTM}}, \text{ by Corollary 4.1.}
\end{aligned}$$

Thus

$$\mathscr{L}_{\mathrm{WSA}} = \mathscr{L}_{\mathrm{DWSA}} = \bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-ATPM}} \,.$$

To continue, we need the following result from [15].

LEMMA 4.2.[19]

$$\mathscr{L}_{f(\alpha)\text{-TM}} \subseteq \mathscr{L}_{(f(\alpha))^2\text{-DTM}}$$

for any nondecreasing function f such that $f(n) \geqslant \log_2 n$.

COROLLARY 4.2. *For each c, $\mathscr{L}_{\mathrm{Two}(c\alpha)\text{-}TM} \subseteq \mathscr{L}_{\mathrm{Two}(2c\alpha)\text{-DTM}}$.*
Corollary 4.2, together with Theorems 2.2 and 3.2, leads to our second result.

THEOREM 4.2.

$$\mathscr{L}_{\mathrm{NEWSA}} = \mathscr{L}_{\mathrm{NEDWSA}} = \bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-TM}} \,.$$

Proof.

$$\begin{aligned}
\bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-DTM}} &\subseteq \mathscr{L}_{\mathrm{NEDWSA}}, \text{ by Theorem 2.2,} \\
&\subseteq \mathscr{L}_{\mathrm{NEWSA}} \\
&\subseteq \bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-TM}}, \text{ by Theorem 3.2,} \\
&\subseteq \bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(2c\alpha)\text{-DTM}}, \text{ by Corollary 4.2,} \\
&\subseteq \bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{Two}(c\alpha)\text{-DTM}} \,.
\end{aligned}$$

[19] The notion of acceptance in [15] differs trivially from that given here, but the result still holds.

Thus

$$\mathscr{L}_{\text{NEDWSA}} = \mathscr{L}_{\text{NEWSA}} = \bigcup_{c \geqslant 1} \mathscr{L}_{\text{Two}(c\alpha)-\text{TM}} \cdot$$

Remark. If we generalize our model to one in which the input tape length can grow to $f(n)$, f any constructible[20] nondecreasing function, then similar results hold. Specifically, let $f(n)$ n be any constructible nondecreasing function. Define a "$f(n)$-WSA" to be a two-tape device in which the input tape is a two-way read-write $f(n)$-tape-bounded work tape and the second tape is a stack tape. Define acceptance by final state only. Denote the family of language accepted by $f(n)$-WSA by $\mathscr{L}_{f(n)-\text{WSA}}$. Then the results of the previous sections show that

$$\mathscr{L}_{f(n)-\text{DWSA}} = \mathscr{L}_{f(n)-\text{WSA}} = \bigcup_{c \geqslant 1} \mathscr{L}_{\text{Two}(cf(\alpha))-\text{APTM}}$$

and that

$$\mathscr{L}_{f(n)-\text{NEDWSA}} = \mathscr{L}_{f(n)-\text{NEWSA}} = \mathscr{L}_{\cup_{c \geqslant 1} \text{Two}(cf(\alpha))-\text{TM}} \cdot$$

Theorems 4.1 and 4.2 provide characterizations of $\mathscr{L}_{\text{WSA}}$ and $\mathscr{L}_{\text{NEWSA}}$ in terms of known families of languages. It is natural to inquire as to the closure properties of these two families, that is, their invariance under certain language operations. Recently, the notion of an "AFL"has been introduced [3] in order to unify the study of closure properties of families of languages. Thus it is natural to investigate these properties of $\mathscr{L}_{\text{WSA}}$ and $\mathscr{L}_{\text{NEWSA}}$ within the framework of "AFL" theory.

By way of introduction and for completeness, we recall the definition of an "AFL".

DEFINITION. An *abstract family of languages* (AFL) is a pair $(\Sigma, \mathscr{L})$, or $\mathscr{L}$ when Σ is understood, where

(1) Σ is an infinite set of symbols

(2) for each L in $\mathscr{L}$ there is a finite set $\Sigma_L \subseteq \Sigma$ such that $L \subseteq \Sigma_L{}^*$,

(3) $\mathscr{L}$ is closed under the operation of $\cup$, ., $+$, inverse homomorphism, ϵ-free homomorphism, and intersection with regular sets,

(4) $L \neq \varnothing$ for some L in $\mathscr{L}$.

In the sequel we shall assume that the reader has had a casual acquaintance with the subject of AFL theory.

We first consider whether $\mathscr{L}_{\text{WSA}}$ and $\mathscr{L}_{\text{NEWSA}}$ are principal[21] AFL. We shall show that both are principal AFL.

[20] Let a be a distinct symvol in W. A nondecreasing function f is said to be *constructible* if there exists a $f(n)$-DTM $A = (K, \Sigma, W, \delta, q_0, \{q\}, 1)$ such that for each integer $n \geqslant 3$,

$$(q_0, \not{c} \upharpoonright a^{n-2}\$, \upharpoonright \beta) \overset{*}{\vdash}_A (q, \not{c}a^{n-2} \upharpoonright \$, a^{f(n-2)} \upharpoonright \beta).$$

[21] A *principal* AFL is an AFL generated by a single language, i.e., it is the smallest AFL containing the given language.

We first consider $\mathscr{L}_{\mathrm{WSA}}$. The following result is noted after Corollary 3.5 in [1].

LEMMA 4.3. *Let f be any superadditive*[22] *deterministic time-constructible*[23] *nondecreasing function such that $f(2n)/f(n) \geqslant \max\{(1+c)^n, (f(n))^c\}$ for some integer c and all sufficiently large integers n. For each positive integer c let $\mathscr{L}_{\mathrm{DETTIME}(f(c\alpha))}$ be the family of all languages accepted within time-bound $f(\alpha)$*[23] *by a deterministic Turing machine M having a single tape. (Here M accepts by both final state and empty storage tape.) Then*

$$\bigcup_{c \geqslant 1} \mathscr{L}_{\mathrm{DETTIME}(f(c\alpha))}$$

is a principal AFL.

PROPOSITION 4.1. *$\mathscr{L}_{\mathrm{WSA}}$ is a principal* AFL.

Proof. By Corollary 4.1 and Theorem 4.1,

$$\mathscr{L}_{\mathrm{WSA}} = \bigcup_{c_1 \geqslant 1} \bigcup_{c_2 \geqslant 1} \mathscr{L}^{\mathrm{TIME}}_{\mathrm{Two}(c_1\,\mathrm{Two}(c_2\alpha))-\mathrm{DTM}}$$

$$= \bigcup_{c \geqslant 1} \mathscr{L}^{\mathrm{TIME}}_{\mathrm{Two}(\mathrm{Two}(c\alpha))-\mathrm{DTM}} \cdot$$

It is obvious that our $f(cn)$ is superadditive. As pointed out after Definition 3.2 of [1], "deterministic time-constructible" includes the "real-time" functions in [17]. In [17] it is shown that our $f(cn)$ is a "real-time" function. Clearly, $f(2n)/f(n) \geqslant \max\{(1+c)^n, (f(n))^c\}$ for some integer c and all sufficiently large n. Thus $f(cn)$ satisfies the hypothesis to Lemma 4.3. Thus, by Lemma 4.3, $\mathscr{L}_{\mathrm{WSA}}$ is a principal AFL.

We now consider $\mathscr{L}_{\mathrm{NEWSA}}$. The fact that $\mathscr{L}_{\mathrm{NEWSA}}$ is a principal AFL is a consequence of Theorem 3.3 in [1], restated here as

LEMMA 4.5. *Let f by any superadditive deterministic-tape-constructible*[24] *nondecreasing function. Let $\mathscr{L}_{\mathrm{DETTAPE}(f)}$ be the family of all languages accepted within tape-bound $f(\alpha)$ by a deterministic Turing machine M having a one-way read-only input tape without*

[22] A nondecreasing function f is *superadditive* if (i) $f(n) \geqslant n$ for some n_0 and all $n \leqslant n_0$; and (ii) for every n_1 and n_2, $f(n_1) + f(n_2) \leqslant f(n_1 + n_2)$.

[23] We are omitting the definitions of "deterministic time-constructible" and "accepted within time-bound" since a presentation would require a lengthy formalization which is not used in the body of our argument.

[24] We are omitting the definitions of "deterministic tape-constructible" and "accept within $f(\alpha)$-tape-bound" since a presentation would require a lengthy formalization which is not used in our argument.

endmarkers and a finite number of storage tapes. (Here M accepts by both final state and empty storage tapes.) Then

$$\bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{DETTAPE}(f(c\alpha))}$$

is a principal AFL.

PROPOSITION 4.2. $\mathscr{L}_{\mathrm{NEWSA}}$ *is a principal* AFL.

Proof. For each c, let $f(cn) = 2^{cn}$. Clearly, $f(cn)$ is superadditive. In [11, p. 149], it is noted that $f(cn)$ is "deterministic tape-constructible". Furthermore, using well-known techniques, it can be shown that

$$\bigcup_{c\geqslant 1} \mathscr{L}_{\mathrm{DETTAPE}(f(cn))} = \bigcup_{c\geqslant 1} \mathscr{L}_{f(cn)-\mathrm{DTM}}\,.$$

Thus, by Lemma 4.5, $\mathscr{L}_{\mathrm{NEWSA}}$ is a principal AFL.

LEMMA 4.6. *Let M' and M'' be* WSA(NEWSA). *Then there exists a* WSA(NEWSA)M *such that $T(M) = T(M') \cap T(M'')$.*

Proof. Since the construction of M involves a well-known argument[25], we omit the proof.

Combining Propositions 4.1 and 4.2 and Lemma 4.6, we have

THEOREM 4.3. $\mathscr{L}_{\mathrm{WSA}}$ *and* $\mathscr{L}_{\mathrm{NEWSA}}$ *are each intersection closed principal* AFL.

It is shown in the proof of Theorem 1.2 in [6] that an AFL which is intersection closed is also closed under ϵ-free substitution[26]. Thus we have

COROLLARY 4.3. $\mathscr{L}_{\mathrm{WSA}}$ *and* $\mathscr{L}_{\mathrm{NEWSA}}$ *are each ϵ-free substitution closed* AFL.

We conclude with several open questions:

(1) Is $\mathscr{L}_{\mathrm{WSA}} = \mathscr{L}_{\mathrm{NEWSA}}$ true?

(2) Does there exist a constructible tape function f that characterizes, in some sense, the family $\mathscr{L}_{\mathrm{WSA}}$?

(3) Is it true that for each WSA A there exists a halting[27] WSA A' such that $T(A) = T(A')$?

[25] For example, see Lemma 2.7 in [13].

[26] Let $L \subseteq \Sigma_1{}^*$. For each a in Σ_1, let $L_a \subseteq \Sigma^+$. Let s be the function defined by $s(\epsilon) = \{\epsilon\}$, $s(a) = L_a$ for each a in Σ_1, and $s(a_1 \cdots a_k) = s(a_1) \cdots s(a_k)$ for each $k \geqslant 1$ and a_i in Σ_1. Then s is called an *ϵ-free substitution.* A family $\mathscr{L}$ of languages is said to *be closed under ϵ-free substitution* if $s(L)$ is in $\mathscr{L}$ for each $L \subseteq \Sigma_1{}^*$ in $\mathscr{L}$ and each ϵ-free substitution s such that $s(a)$ is in $\mathscr{L}$ for each a in Σ_1.

[27] A WSA A is said to be *halting* if for each word w A either accepts or rejects w.

(4) Is $\mathscr{L}_{\mathrm{WSA}}$ closed under complementation?

(5) Let $\mathscr{L}_{\mathrm{2NSA}}$ denote the family of languages accepted by two-way nondeterministic stack acceptors (with a read-only input tape). Then,

(a) Is $\mathscr{L}_{\mathrm{2NSA}}$ properly contained in $\mathscr{L}_{\mathrm{WSA}}$?
(b) Are $\mathscr{L}_{\mathrm{2NSA}}$ and $\mathscr{L}_{\mathrm{NEWSA}}$ incomparable?

Note that if the answer to (1) is "yes" then the answer to (2) is "yes". Also if the answer to (3) is "yes" then the answer to (4) is "yes". It is easily shown that $\mathscr{L}_{\mathrm{NEWSA}}$ is closed under complementation. Thus if the answer to (4) is "no" then the answer to (1) is "no".

References

1. R. V. Book, S. A. Greinbach, and B. Wegbriet, "Tape- and Time-Bounded Turning Acceptors and AFL's," pp. 92–99, Second Annual ACM Symposium on Theory of Computing, May, 1970, Northampton, MA.
2. S. A. Cook, "Variations on Pushdown Machines," pp. 229–232, Proceedings of the ACM Symposium on Theory of Computing, May, 1969, Marina del Rey, CA.
3. S. Ginsbury and S. Greiback, Abstract families of languages, *in* "Studies in Abstract Families of Languages," pp. 1–32, Memoir No. 87, American Mathematical Society, Providence, RI, 1969.
4. S. Ginsburg and S. A. Greibach, Principal AFL, *J. Comput. System Sci.* **4** (1970), 308–338.
5. S. Ginsburg, S. A. Greibach, and M. A. Harrison, Stack automata and compiling, *J. Assoc. Comput. Mach.* **14** (1967), 172–201.
6. S. Ginsburg, S. Greibach, and J. Hopcroft, Pre-AFL, *in* "Studies in Abstract Families of Languages," pp. 41–51, Memoir No. 87, American Mathematical Society, Providence, RI, 1969.
7. J. A. Giuliano, "Writing Stack Acceptors," Thesis, University of Southern California, Los Angeles, CA, 1971.
8. J. N. Gray, M. A. Harrison, and O. Ibarra, Two-way pushdown automata, *Information and Control* **11** (1967), 30–70.
9. J. Hartmanis, P. M. Lewis II, and R. E. Stearns, "Computational Complexity of Recursive Sequences," pp. 82–90, Proceedings of the Fifth Annual Symposium on Switching Circuit Theory and Logical Design, Princeton, IEEE New Jersey, 1965.
10. J. Hartmanis and R. E. Stearns, On the computational complexity of algorithms, *Trans. Amer. Math. Soc.* **117** (1965), 285–306.
11. F. C. Hennie and R. E. Stearns, Two-tape simulation of multitape turning machines, *J. Assoc. Comput. Mach.* **13** (1966), 533–546.
12. J. Hopcroft and J. Ullman, "Formal Languages and Their Relation to Automata," Addison-Wesley, Reading, MA, 1969.
13. J. Hopcroft and J. D. Ullman, "Relations Between Time and Tape Complexities," *J. Assoc. Comput. Mach.* **15** (1968), 414–427.
14. G. Mager, Writing pushdown acceptors, *J. Comp. Syst. Sci.* **3** (1969), 276–318.
15. W. Savitch, Relationships between non-deterministic and deterministic tape complexities, *J. Comput. System Sci.* **4** (1970), 177–192.

16. R. Stearns, J. Hartmanis, and P. M. Lewis II, "Hierarchies of Memory Limited Computations," IEEE Conference Record on Switching Circuit Theory and Logical Design, 1965.
17. H. Yamada, Real-time computation and recursive functions not real-time computable, *IRE Trans.* **EC-11** (1962), 753–760.

Printed in Belgium