INFORMATION AND CONTROL 18, 404–446 (1971)

Two-Way Sequential Transductions and Stack Automata

R. W. EHRICH* AND S. S. YAU

Departments of Electrical Engineering and Computer Sciences, Northwestern University, Evanston, Illinois 60201

A 2-way sequential transducer is a 2-way finite acceptor to which an output structure has been added. Such a device is considerably more complex than the usual 1-way sequential transducer, and it is shown that these transducers map both regular sets and context free languages into the context sensitive languages. Such a transducer is the simplest known which can map a list into its reversal or make duplicate copies of a list. In this paper unsolvability of equivalence is shown for a special class of nondeterministic 2-way sequential transducers, and it is shown that deterministic transducers are not so powerful as nondeterministic transducers. The properties of two-way transductions of regular-sets and context free languages are derived in detail and are shown to be similar to the corresponding properties of context free languages. Also, in this paper the intimate relationship between 2-way sequential transducers and 1-way stack automata is shown by presenting a new grammar which generates exactly the 1-way stack automata languages, and many of the properties of these grammars are derived.

1. INTRODUCTION

A 2-way sequential transducer, which will be referred to as a 2st, may be considered to be a primitive type of list processor or translator which reads lists or language strings of a given structure while writing outputs. Such a device uses potentially infinite tape since the entire input string may be read more than once; however during a mapping the transducer is still regarded as finite state. The essential source of its increased power compared with the 1-way sequential transducer (or simply 1st) seems to be in the spatial position of the read head, which enables the 2st to act in some sense as a counter.

The earliest mention of such a device was by Shepherdson (1959) who gave an informal example of a 2st which could map a regular set into a

* R. W. Ehrich is now with the Department of Electrical Engineering, University of Massachusetts, Amherst, Massachusetts.

context free language. The properties of the 1st are well known (Ginsburg (1966)), but there is evidence that some data and language processing operations are modeled more accurately by the 2st. For example, it is not possible for a 1st to map $\$x\phi$ in a language $\$L\phi$ into its reversal, $\phi x^R\$$.

The discussion of the 2st will be divided into three sections. In Section 2 basic concepts and definitions will be reviewed, the 2st defined, and a very general result concerning 2-way state machines derived as a preliminary for Section 3 and Section 4.

In Section 3 the basic properties of the 2st will be given. We will show that 2st mappings of regular sets or context free languages are properly contained within the context sensitive languages, and that 2st mappings are closed under most operations including substitution, reversal, and arbitrary sequential transducer mapping. We will also show that nondeterministic 2st mappings of regular sets properly contain deterministic 2st mappings of regular sets. Furthermore, the equivalence question will be shown to be unsolvable for a class of transducers, called complete nondeterministic 2st, hence for 2st in general.

In Section 4 the relationship between 1-way nondeterministic stack automata and nondeterministic 2st will be explored. A new grammar will be given which generates exactly the class of 1-way stack automata languages and does so by making explicit use of the 2st. A normal form for the grammars will be given, and we will show how several properties of the 1-way stack automaton are obtained easily from the grammars and from their 2st substructures. It will be shown that many properties of these grammars are similar to the corresponding properties of context free grammars, and the grammars can be used to specify structures in Algol 60 which were previously in the semanties of the language.

2. Languages and Transducers

When discussing language complexity, it is common to refer to a well-known hierarchy of languages with decreasing complexity (Chomsky, 1959).

Definition. A *phrase structure grammar* is a 4-tuple $G = (N, T, P, \sigma)$, where

(1) N is a finite nonempty set of nonterminal symbols.
(2) T is a finite nonempty set of terminal symbols.

(3) P is a set of pairs (u, v), $u \in N^+$, $v \in (N \cup T)^*$ which denote production rules, usually written in the form $u \to v$.[1]
(4) $\sigma \in N$ is called the starting symbol.

Let $G = (N, T, P, \sigma)$ be a phrase structure grammar. $x \Rightarrow y$ is used to denote a derivation in G if and only if there are strings $w_1, w_3 \in (N \cup T)^*$, $w_2 \in N^+$, and $w_4 \in (N \cup T)^*$ such that $x = w_1w_2w_3$, $y = w_1w_4w_3$, and (w_2, w_4) is a production in P. $x \overset{*}{\Rightarrow} y$ is used to denote the transitive closure of $\Rightarrow$ whenever there exist words $w_1, \ldots, w_k$ in $(N \cup T)^*$ such that $x = w_1$, $y = w_k$, and $w_i \Rightarrow w_{i+1}$, $1 \leqslant i \leqslant k - 1$, $k > 1$. A string y is in the language generated by G, denoted $y \in L(G)$, if and only if $y \in T^*$ and $\sigma \overset{*}{\Rightarrow} y$.

DEFINITION. A language L is said to be *context sensitive* if there is a phrase structure grammer $G = (N, T, P, \sigma)$ such that $L = L(G)$ and such that G has the property that for each pair $(u, v) \in P$ of the form $w_1w_2w_3 \to w_1w_4w_3$, $| w_2 | \leqslant | w_4 |$, where $| x |$ denotes the length of x, $w_1, w_3 \in N^*$, and $w_2 \in N$.

DEFINITION. A language L is said to be *context free* if there is a phrase structure grammar $G = (N, T, P, \sigma)$ such that $L = L(G)$ and such that for each pair $(u, v) \in P$ of the form $w_1w_2w_3 \to w_1w_4w_3$, $w_1 = w_3 = \epsilon$, $w_2 \in N$, and $| w_2 | \leqslant | w_4 |$.

DEFINITION. A language L is said to be *regular* if there is a phrase structure grammar $G = (N, T, P, \sigma)$ such that $L = L(G)$ and such that for each pair $(u, v) \in P$ of the form $w_1w_2w_3 \to w_1w_4w_3$, $w_1 = w_3 = \epsilon$, $w_2 \in N$, and either all w_4 are in $(T^* \cup NT^*)$ or all w_4 are in $(T^* \cup T^*N)$.

Following the conventions of other authors (Hopcroft and Ullman, 1969) we will permit regular, context free and context sensitive languages to contain the null string ϵ. It is well known that these families of languages may equally well be defined by corresponding classes of automata.

DEFINITION. A *finite state acceptor* is a 5-tuple $A = (K, \Sigma, \delta, s_0, F)$, where

(1) K is a finite, nonempty set of states.
(2) Σ is a finite, nonempty input alphabet.
(3) δ is a mapping of $K \times (\Sigma \cup \{\epsilon\})$ into subsets of K.
(4) $s_0 \in K$ is the initial state.
(5) $F \subseteq K$ is the set of final states.

[1] If X and Y are sets of words, $XY = \{xy \mid x \in X, y \in Y\}$. Letting $X^0 = \{\epsilon\}$ we denote by X^* the set $\bigcup_0^\infty X^i$, where $X^i = X^{i-1}X$, and by X^+ the set $\bigcup_1^\infty X^i = XX^*$.

δ is extended to $K \times \Sigma^*$ by defining $\delta(q, \epsilon) = q$ and $\delta(q, x_1 \cdots x_k) = q_k$, where $q_0 = \{q\}$ and $q_i = \{\bigcup \delta(s, x_i) \mid s \in q_{i-1}\}$ for $1 \leqslant i \leqslant k$, for each $q \in K$, and for each $x = x_1 \cdots x_k \in \Sigma^*$. A word x is accepted by A, denoted by $x \in T(A)$, if and only if $\delta(s_0, x) \cap F \neq \phi$. $L \subseteq \Sigma^*$ is regular if and only if $L = T(A)$ for some finite state acceptor A.

DEFINITION. A *1-way sequential transducer* is a 5-tuple $S_1 = (K, \Sigma, \Delta, H, s_0)$, where

(1) K is a finite, nonempty set of states.

(2) Σ is a finite, nonempty input alphabet.

(3) Δ is a finite, nonempty output alphabet.

(4) $s_0 \in K$ is the initial state.

(5) H is a finite subset of $K \times \Sigma^* \times \Delta^* \times K$.

Informally, $(p, x, y, q) \in H$ means that S_1 in state p may read $x \in \Sigma^*$ while writing $y \in \Delta^*$ and then transit to state q.

DEFINITION. A *2-way sequential transducer* is a 5-tuple $S_2 = (K, \Sigma, \Delta, H, s_0)$, where

(1) K is a finite, nonempty set of states.

(2) Σ is a finite, nonempty input alphabet.

(3) Δ is a finite, nonempty output alphabet.

(4) $s_0 \in K$ is the initial state.

(5) H is a finite subset of $K \times \Sigma \times \Delta^* \times K \times \{-1, 0, 1\}$.

Informally a 2st functions as follows. The 2st starts by reading the leftmost tape symbol while in state s_0, and a transduction terminates when the 2st moves right of the rightmost tape symbol to the blank tape where no further moves are defined. When $(p, a, y, q, e) \in H$, then S_2 in state p may read a, write y, transit to state q, and move e squares right on the input tape. It is evident that the transducer might never move from the right end of its input tape, and any output written under such circumstances is not considered a 2st mapping. If $R \subseteq \Sigma^*$ is a language, then $S_2(R) = \phi$ if and only if for all $x \in R$, S_2 never moves off the right end of x. ϵ is in $S_2(R)$ if $S_2(x) = \epsilon$, and by definition $S_2(\epsilon) = \epsilon$.

A 2st is deterministic if whenever (p, a, y, q, e) and (p, a, y', q', e') are in H then $y = y'$, $q = q'$, and $e = e'$. It is not hard to show that the class

of 1st is equivalent to the class of 2st with right endmarker (required in the forward direction) for which $(p, a, y, q, e) \in H$ implies $e \in \{0, 1\}$. Since the 2st reads exactly one symbol in each move, it is convenient to isolate its state behavior.

DEFINITION. Let $S_2 = (K, \Sigma, \Delta, H, s_0)$ be a 2st. Its *associated state machine* (asm) is the 3-tuple $A = (K, \Sigma, \delta)$, where δ is a mapping from $K \times \Sigma$ into subsets of $K \times \{-1, 0, 1\}$ defined as follows: $\delta(p, a)$ contains (q, e) if and only if $(p, a, y, q, e) \in H$ for some $y \in \Delta^*$. Thus, A simulates the state and spatial behavior of S_2 when both are started on the same square of tape in the same state.

DEFINITION. An *instantaneous description* (id) of a 2st or its associated state machine is an element of $\Sigma^* K \Sigma^* \cup K\{b\}\Sigma^*$, where b denotes a blank symbol and $K \cap \Sigma = \phi$. If $xqay$ is an id with $x, y \in \Sigma^*$, and $a \in \Sigma$, then S_2 is in state q reading a on tape xay. Similarly qbx indicates that S_2 is in state q reading the blank just to the left of tape x.

Let $A = (K, \Sigma, \delta)$ be an asm and let $xapa'y$ be an id with $x, y \in \Sigma^*$ and $a, a' \in \Sigma$. Then,

(1) $xapa'y \vdash xaa'qy$ if $(q, 1)$ is in $\delta(p, a')$.

(2) $xapa'y \vdash xaqa'y$ if $(q, 0)$ is in $\delta(p, a')$.

(3) $xapa'y \vdash xpaa'y$ if $(q, -1)$ is in $\delta(p, a')$.

If i and i' are id's of A, then $i \overset{*}{\vdash} i'$ is the transitive closure of $\vdash$ whenever there are id's $i_1, \ldots, i_k$ such that $i = i_1$, $i' = i_k$, and $i_1 \vdash i_2 \vdash \cdots \vdash i_k$. The following theorem generalizes the results of Rabin and Scott (1959) and Shepherdson (1959) to nondeterministic 2-way finite automata.

THEOREM 2.1. *Let* $A = (K, \Sigma, \delta)$ *be an* asm. *Then the sets*

$$T_1(p, q) = \{x \mid x_1 \cdots x_{n-1} p x_n \overset{*}{\vdash} x_1 \cdots x_n q\},$$

$$T_2(p, q) = \{x \mid p x_1 \cdots x_n \overset{*}{\vdash} x_1 \cdots x_n q\},$$

$$T_3(p, q) = \{x \mid p x_1 \cdots x_n \overset{*}{\vdash} q b x_1 \cdots x_n\},$$

and

$$T_4(p, q) = \{x \mid x_1 \cdots x_{n-1} p x_n \overset{*}{\vdash} q b x_1 \cdots x_n\}$$

are regular.

Proof. Define the set $\hat{K} = \{\hat{s} \mid s \in K\}$. For each $x \in \Sigma^*$ define the functions σ_x on $(\hat{K} \cup K) \times (\hat{K} \cup K)$ so that for all $p, q \in K$ and $\hat{p}, \hat{q} \in \hat{K}$,

$$\sigma_x(p, q) = 1 \quad \text{iff} \quad x_1 \cdots x_{n-1} p x_n \overset{*}{\vdash} x_1 \cdots x_n q \text{ and 0 otherwise,}$$

$$\sigma_x(\hat{p}, q) = 1 \quad \text{iff} \quad p x_1 \cdots x_n \overset{*}{\vdash} x_1 \cdots x_n q \text{ and 0 otherwise,}$$

$$\sigma_x(\hat{p}, \hat{q}) = 1 \quad \text{iff} \quad p x_1 \cdots x_n \overset{*}{\vdash} q b x_1 \cdots x_n \text{ and 0 otherwise,}$$

$$\sigma_x(p, \hat{q}) = 1 \quad \text{iff} \quad x_1 \cdots x_{n-1} p x_n \overset{*}{\vdash} q b x_1 \cdots x_n \text{ and 0 otherwise.}$$

Now $\sigma_x = \sigma_y$ implies that $\sigma_{xz} = \sigma_{yz}$ for all $z \in \Sigma^*$, for suppose $\sigma_x = \sigma_y$. Then

(1) $x_1 \cdots x_n z_1 \cdots z_{k-1} p z_k \overset{*}{\vdash} x_1 \cdots x_n z_1 \cdots z_k q$

$\Leftrightarrow y_1 \cdots y_r z_1 \cdots z_{k-1} p z_k \overset{*}{\vdash} y_1 \cdots y_r z_1 \cdots z_k q;$

(2) $p x_1 \cdots x_n z_1 \cdots z_k \overset{*}{\vdash} x_1 \cdots x_n z_1 \cdots z_k q$

$\Leftrightarrow p y_1 \cdots y_r z_1 \cdots z_k \overset{*}{\vdash} y_1 \cdots y_r z_1 \cdots z_k q;$

(3) $p x_1 \cdots x_n z_1 \cdots z_k \overset{*}{\vdash} q b x_1 \cdots x_n z_1 \cdots z_k$

$\Leftrightarrow p y_1 \cdots y_r z_1 \cdots z_k \overset{*}{\vdash} q b y_1 \cdots y_r z_1 \cdots z_k ;$

(4) $x_1 \cdots x_n z_1 \cdots z_{k-1} p z_k \overset{*}{\vdash} q b x_1 \cdots x_n z_1 \cdots z_k$

$\Leftrightarrow y_1 \cdots y_r z_1 \cdots z_{k-1} p z_k \overset{*}{\vdash} q b y_1 \cdots y_r z_1 \cdots z_k .$

Similarly, $\sigma_x = \sigma_y$ implies that $\sigma_{zx} = \sigma_{zy}$. Then

$$x \equiv y \quad \text{if and only if} \quad \sigma_x = \sigma_y \tag{2.1}$$

is a congruence relation of finite index at most $2^{4|K|^2}$, where $|K|$ is the number of states in K. Then

$$T_1(p, q) = \{x \in \Sigma^* \mid \sigma_x(p, q) = 1\},$$

$$T_2(p, q) = \{x \in \Sigma^* \mid \sigma_x(\hat{p}, q) = 1\},$$

$$T_3(p, q) = \{x \in \Sigma^* \mid \sigma_x(\hat{p}, \hat{q}) = 1\},$$

$$T_4(p, q) = \{x \in \Sigma^* \mid \sigma_x(p, \hat{q}) = 1\}$$

are all a union of the equivalence classes induced by (2.1) and are therefore regular. Furthermore, if $x = x_1 \cdots x_{n-1}$ and $x' = x_1 \cdots x_n$, then $\sigma_{x'}$ is

effectively calculable from x_n and σ_x so that a finite state acceptor for $T_i(p, q)$, $i = 1, 2, 3, 4$, can readily be constructed.

Any tapes x and y such that $\sigma_x = \sigma_y$ by (2.1) are indistinguishable by S_2 even though the output written by S_2 may differ when it is reading from tape x or from tape y. It is noted that both the 1st and the 2st have the possibility of writing strictly infinite tapes, and as in the case of the 1st this behavior is implicitly ignored.

DEFINITION. A 2st is said to *loop* if for some id i, $i \overset{*}{\vdash} i$. A 2st is said to *loop on* x if it loops and has no sequence of moves such that $s_0x \overset{*}{\vdash} xq$ for some state q and initial state s_0.

LEMMA 2.1. *Let R be a regular set (context free language) and let $L = S_2(R)$ for some* 2st $S_2 = (K, \Sigma, \Delta, H, s_0)$. *Then there is a regular set (context free language) R' such that $L = S_2(R')$ and $x \in R$ only if S_2 does not loop on x.*

Proof. Let $A = (K, \Sigma, \delta)$ be the asm corresponding to S_2. The set $T = \bigcup_{q \in K} T_2(s_0, q)$ is regular since it is a finite union of regular sets (Ginsburg, 1966) and T is the set of tapes $x \in \Sigma^*$ from which S_2 can ultimately leave the right end. Let $R' = R \cap T$; R' is a regular set (context free language) if R is because of the closure of these languages under intersection with regular sets (Ginsburg, 1966). In particular, if S_2 is deterministic and $s_0x' \overset{*}{\vdash} x'q$ for $x' \in R'$, then S_2 never loops and $s_0x' \overset{*}{\vdash} x'q$ is the only move it makes on x'.

LEMMA 2.2. *Let $L = S_2(R)$ be a deterministic* 2st *mapping of a language R, where $S_2 = (K, \Sigma, \Delta, H, s_0)$. If $y = S_2(x_1zx_2)$, then $| y' | \leqslant \alpha | z |$ for some constant α, where y' is the output S_2 writes while reading z.*

Proof. S_2 loops on x if there are id's $i_0, i_1, i_2, \ldots$ such that $i_0 = s_0x \vdash i_1 \vdash \cdots$ and gives $i_j = i_k$ for some $k \neq j$. Thus no tape square can be read more than once in the same state. If each tape square is read exactly $| K |$ times, then $| y | \leqslant \alpha | x |$, where $\alpha = | K | \max_y \{| y | | (p, a, y, q, e) \in H\}$.

Lemma 2.2 is essential in establishing the difference between deterministic and nondeterministic mappings of regular sets. It is added here that a 2st is also a finite state acceptor in disguise, since if we are willing to add endmarkers to a set R, S_2 may check the regularity of $\not{S}x\not{\phi}$ in $\not{S}R\not{\phi}$ before writing from x.

LEMMA 2.3. *Let $S_2 = (K, \Sigma, \Delta, H, s_0)$ be a* 2st. *Then the set of tapes $U_1(p, q) = \{x \mid x_1 \cdots x_{n-1}px_n \overset{*}{\vdash} x_1 \cdots x_nq$ on ϵ output$\}$ are regular.*

Proof. Define a new 2st $S_2' = (K \cup \hat{K}, \Sigma, \Delta, H', s_0)$, where $\hat{K} = \{\hat{s} \mid s \in K\}$. Define

$$H' = \{(p, a, y, q, e) \mid (p, a, \epsilon, q, e) \in H\}$$
$$\cup \{(\hat{p}, a, y, \hat{q}, e) \mid (p, a, y, q, e) \in H\}$$
$$\cup \{(p, a, y, \hat{q}, e) \mid (p, a, y, q, e) \in H, y \neq \epsilon\}, \qquad \forall p, q \in K, \quad \hat{p}, \hat{q} \in \hat{K},$$

and let $A = (K \cup \hat{K}, \Sigma, \delta)$ be its asm. By construction $S_2(x) = S_2'(x)$ for all $x \in \Sigma^*$, but A enters a state in $\hat{K}$ if S_2 writes non-ϵ output. Then let $U_1(p, q) = T_1(p, q)$, where T_1 is constructed from A as in Theorem 2.1. Lemma 2.3 is extended in the obvious way to U_2, U_3, and U_4 by using T_2, T_3, and T_4 of Theorem 2.1.

For convenience in Section 3 we prove the following:

LEMMA 2.4. *Let* $S_2 = (K, \Sigma, \Delta, H, s_0)$ *be a* 2st. *Then* $(p, a, y, q, e) \in H$ *implies* $|y| \leqslant 1$ *without loss of generality.*

Proof. Suppose otherwise. Define new states $t_1, \ldots, t_{n-1}$ and replace (p, a, y, q, e) by the moves

$$(p, a, b_1, t_1, 0), (t_1, a, b_2, t_2, 0), \ldots, (t_{n-1}, a, b_n, q, e),$$

where $y = b_1 \cdots b_n$. Determinism is preserved by the construction.

COROLLARY 2.1. *Let* $S_1 = (K, \Sigma, \Delta, H, s_0)$ *be a* 1st. *Then without loss of generality* $(p, a, y, q) \in H$ *implies* $|y| \leqslant 1$.

3. TRANSDUCERS AND MAPPINGS

In this section frequent reference is made to regular sets and to context free languages. Whenever it is unimportant which is intended we will refer simply to a "language." In order to show that a 2st mapping of a language is context sensitive it is sufficient to show that it is accepted by a linear bounded automaton.

DEFINITION. A *linear bounded automaton* (lba) is a 5-tuple $A = (K, \Sigma, \delta, q_0, F)$, where

(1) K is a finite, nonempty set of states.

(2) Σ is a finite, nonempty set of symbols.

(3) δ is a mapping of $K \times \Sigma$ into subsets of $K \times \Sigma \times \{1, 0, -1\}$.

(4) $q_0 \in K$ is the initial state.

(5) $F \subseteq K$ is the set of final states.

Informally a language R is accepted by an lba in the following way. $x \in R$ is written on the lba's working tape. A operates on this tape, reading and writing symbols. Thus the length of the working tape of A is bounded above by the length of the tape it is to recognize. More formally, if $(q, b, e) \in \delta(p, a)$, then A in state p reading a writes b over the a, enters state q, and moves e squares right on the working tape. If A leaves the tape on which x was written from the right while moving into a final state $f \in F$, then x is accepted by A, denoted by $x \in T(A)$. Let A be an lba and let $T(A)$ be the set accepted by A. $L \subseteq \Sigma^*$ is a context sensitive language if and only if $L = T(A)$ for some lba A. It is possible to show that every 2st mapping of a language may be obtained in such a way that the length of a mapping is bounded below by the length of the input string.

LEMMA 3.1. *Let $S_2 = (K, \Sigma, \Delta, H, s_0)$ be a* 2st *and let R be a language. It is decidable whether ϵ is in $S_2(R)$.*

Proof. Let $A = (K, \Sigma, \delta)$ be the asm of S_2, and suppose S_2 is constructed as in Lemma 2.3. Then $R' = \bigcup_{q \in K} U_2(s_0, q)$ is regular, and hence $R \cap R'$ is a language. Now, $x \in R \cap R'$ if and only if ϵ is in $S_2(x)$, and it is known to be decidable whether $R \cap R' = \phi$.

LEMMA 3.2. *Let R be a language and let $S_2 = (K, \Sigma, \Delta, H, s_0)$ be a* 2st *such that $L = S_2(R)$. Then there is a language R' and a new* 2st *$S_2' = (K', \Sigma', \Delta', H', s_0')$ such that $L = S_2'(R')$ and such that for each $x \in L$ where $x \in S_2(y)$ there is a $y' \in R'$ such that $x \in S_2'(y')$ and $| y' | \leqslant 2 | x | + 1$.*

Proof. The proof is in two parts. R' is defined in the first and S_2' in the second.

(a) First, a 1st $S_1 = (L, \Sigma, \Sigma', J, p_0)$ is defined such that $R' = S_1(R\not{S}) \cup T$, where $\not{S}$ is a special symbol not in Σ and $T = \{\epsilon\}$ if ϵ is in $S_2(R)$ and $T = \phi$ otherwise. By Lemma 3.1 it is decidable whether ϵ is in $S_2(R)$. Again we can assume that S_2 is constructed as in Lemma 2.3. Since the sets $U_i(p, q)$ constructed from its asm are all regular for all $p, q \in K$ and $i = 1, 2, 3, 4$, let $M_{ij}^k = (K_{ij}^k, \Sigma, \delta_{ij}^k, p_{ij}^k, F_{ij}^k)$, $0 \leqslant i, j \leqslant m$, $k = 1, 2, 3, 4$, be the finite state acceptors for $U_k(s_i, s_j)$, where $m = | K |$. As noted, these acceptors may be effectively constructed. Then let $\Sigma' = \{0, 1\}^{4m^2} \cup \Sigma$ and

$L = (K_{11}^1 \times \cdots \times K_{mm}^1 \times \cdots \times K_{mm}^4) \cup \{s_0, s_1\}$, and let J have the following moves:

(1) $\{(s, a, a, s_1), (s, a, a, (p_{11}^1, \ldots, p_{mm}^4)) \mid a \in \Sigma, s \in \{s_0, s_1\}\}$.

(2) $\{(s_0, a, \epsilon, (\delta_{11}^1(p_{11}^1, a), \ldots, \delta_{mm}^4(p_{mm}^4, a))) \mid a \in \Sigma\}$.

(3) $\{((r_{11}^1, \ldots, r_{mm}^4), a, \epsilon, (\delta_{11}^1(r_{11}^1, a), \ldots, \delta_{mm}^4(r_{mm}^4, a))) \mid a \in \Sigma, r_{ij}^k \in K_{ij}^k\}$.

(4) $\{((r_{11}^1, \ldots, r_{mm}^4), a, (t_{11}^1, \ldots, t_{mm}^4), s_1) \mid a \in \{\epsilon, \not\Delta\}, t_{ij}^k = 1$ if and only if $r_{ij}^k \in F_{ij}^k$ and $t_{ij}^k = 0$, otherwise$\}$.

(5) $\{(s_1, \not\Delta, \epsilon, s_1)\}$.

S_1 duplicates its input tapes with the modification that it can nondeterministically delete segments of these tapes. Whenever it deletes a segment of tape, however, it prints a special symbol in its place which describes how S_2 behaves on that segment while writing only ϵ output. Informally, rules (1) allow S_1 to duplicate tape and to nondeterministically initiate the skipping of a portion of tape from which S_2 is assumed to write only ϵ output. Rules (2) allow S_1 to skip the first symbol of its input tape. Rules (3) allow S_1 to read input tape without printing, while keeping complete record of all the state transitions S_2 can make on the deleted tape without writing output. Finally, rules (4) allow S_1 to terminate its ϵ output sequence by writing a special symbol which characterizes the behavior of S_2 on the deleted tape, and rule (5) allows S_1 to leave the input tape. R' is a language of the same type as R (Ginsburg, 1966).

(b) Now define S_2' in the following way. Let

$$K' = \{s_{iL}', s_{iR}' \mid s_i \in K\},$$

$$\Sigma' = \{0, 1\}^{4m^2} \cup \Sigma,$$

$$\Delta' = \Delta,$$

$$s_0' = s_{0R}$$

and let H' have the following moves:

1(a) $\{(p_L, a, y, s_R, e), (p_R, a, y, s_R, e) \mid (p, a, y, s, e) \in H, e \in \{0, 1\}\}$,

(b) $\{(p_L, a, y, s_L, -1), (p_R, a, y, s_L, -1) \mid (p, a, y, s, -1) \in H\}$;

2(a) $\{(s_{iR}, b, \epsilon, s_{jL}, -1) \mid b_{ij}^3 = 1, s_i, s_j \in K\}$,

(b) $\{(s_{iR}, b, \epsilon, s_{jR}, 1) \mid b_{ij}^2 = 1, s_i, s_j \in K\}$,

(c) $\{(s_{iL}, b, \epsilon, s_{jL}, -1) \mid b_{ij}^4 = 1, s_i, s_j \in K\}$,

(d) $\{(s_{iL}, b, \epsilon, s_{jR}, 1) \mid b_{ij}^1 = 1, s_i, s_j \in K\}$.

Informally, rules (1) allow S_2' to simulate S_2 on symbols of Σ. The subscripts R and L allow S_2' to remember the direction of its previous move. Using the fact that it moved to a special symbol in $\{0, 1\}^{4m^2}$ from the right or left, rules (2) allow S_2' to simulate S_2 on portions of tape from which S_2 writes only ϵ. We need only show now that $S_2'(R') = L$.

Let $x \in S_2(y)$, and suppose $y = b_0x_1b_1x_2 \cdots x_sb_s$, where $b_0, b_s \in \Sigma^*$ and $b_i, x_j \in \Sigma^+$, $0 < i < s$ and $1 \leqslant j \leqslant s$. b_j denotes a substring of y from which S_2 writes only ϵ as it scans across the segment during the computation of $S_2(y)$. Then there is a string $y' = b_0'x_1b_1'x_2 \cdots x_sb_s'$ in R', where $b_j' \in \{0, 1\}^{4m^2}$, $0 \leqslant j \leqslant s$, and $b_j' = \epsilon$ if $b_j = \epsilon$. By construction S_2' writes the same output sequences from x_j as S_2 does, and b_i' tells S_2' how S_2 behaves on b_i. Hence $S_2(R) \subseteq S'(R')$. Note that there may be longer tapes in R' which also produce x.

Conversely, let $x \in S_2'(y')$, where $y' = b_0'x_1b_1' \cdots x_sb_s'$. We can assume that S_2 writes at least one output symbol from each symbol of x_i. If some b_j' corresponds to a section b_j of tape $y \in R$ from which S_2 must write an output, S_2' blocks. However, there is then another tape in R' in which the segment from which S_2 writes output is not deleted. Then S_2' exactly simulates S_2, and then $S_2'(R') \subseteq S_2(R)$ so that $S_2(R) = S_2'(R')$.

It is not hard to see that S_2' is deterministic if S_2 is deterministic. By a standard coding procedure combining the symbols $b_0', \ldots, b_s'$ with the end symbols of $x_1, \ldots, x_s$ we obtain

COROLLARY 3.1. *If $L = S_2(R)$ is a* 2st *mapping of a language, then $L = S_2'(R')$ for a new* 2st *S_2' and language R', and $x \in S_2'(y')$ implies that there exists a tape $y'' \in R'$ such that $x \in S_2'(y'')$ and $|x| \geqslant |y''|$.*

Since the proof of the following theorem is uninformative, it is given in algorithmic form.

THEOREM 3.1. *If $L = S_2(R)$ for any language R, then L is context sensitive.*

Proof. All that is necessary is to show how to construct the lba A such that $T(A) = S_2(R)$. The working tape of A is divided into three channels by

augmenting the alphabet of A. Now suppose $x \in S_2(y)$ and $x \in S_2'(y'')$ in the notation of Corollary 3.1, then

(1) x is written on the channel 1 tape;

(2) y'' is written nondeterministically on the channel 2 tape as in Kuroda (1964).

(3) A simulates S_2' on y'', writing $S_2'(y'')$ on channel 3.

(4) A compares channel 1 with channel 3, accepting if they are identical, indicating that $x \in S_2'(y'')$.

A blocks if it attempts to write y'' longer than x on channel 2 so that the working tape length is bounded above by $|x|$. If ϵ is in $S_2(R)$, then ϵ is in R', $S_2'(\epsilon) = \epsilon$ by definition, and the starting state of A is also an accepting state so that A also accepts ϵ.

DEFINITION. Let $L \subseteq \Sigma^*$ be an element of a family of sets $\mathscr{T}$. For each $a \in \Sigma$ let $L_a \subseteq \Sigma_a{}^*$ be a language. Let τ be the function defined by $\tau(\epsilon) = \epsilon$ and $\tau(a) = L_a$ for each $a \in \Sigma$, and let $\tau(a_1 \cdots a_k) = \tau(a_1) \cdots \tau(a_k)$ for each $a_1 \cdots a_k \in \Sigma^+$. Let $\tau(L) = \bigcup_{x \in L} \tau(x)$. A family $\mathscr{T}$ is said to be *closed under substitution* τ if whenever $L \in \mathscr{T}$ and $L_a \in \mathscr{T}$ for all $a \in \Sigma$, then $\tau(L) \in \mathscr{T}$.

THEOREM 3.2. 2st *mappings of languages are closed under substitution.*

Proof. Let $L = S_2(R)$ for some language R and 2st $S_2 = (K, \Sigma, \Delta, H, s_0)$. Assume by Lemma 2.4 that if $(p, a, y, q, e) \in H$, then $|y| \leqslant 1$.

(1) For the remainder of the proof it will be assumed that

$$K = s_0, s_1, \ldots, s_r,$$

$$\Sigma = a_1, \ldots, a_u,$$

$$\Delta = b_1, \ldots, b_t.$$

For each i, $1 \leqslant i \leqslant t$, let L_{b_i} be a 2st mapping over Δ_{b_i} of a language $R_{b_i} \subseteq \Sigma_{b_i}^*$. Let $L_{b_i} = S_2^{b_i}(R_{b_i})$, where $S_2^{b_i} = (K_{b_i}, \Sigma_{b_i}, \Delta_{b_i}, H_{b_i}, s_0^{b_i})$. The remainder of the proof is devoted to showing that

$$\sigma(b_i) = L_{b_i} \qquad \text{and} \qquad \sigma(\epsilon) = \epsilon$$

is a substitution such that $\sigma(L) = S_2'(R')$ for some new 2st S_2' and language R'.

(2) For each $0 \leqslant i, k \leqslant r$, $1 \leqslant j \leqslant u$, and $e \in \{-1, 0, 1\}$, let

$$\Sigma_{ijke} = \{[_{ijke},]\} \cup \Delta \bigcup_{b_l \in \Delta} \Sigma_{b_l}$$

and let L_{ijke} be the language over Σ_{ijke} of the form

$$L_{ijke} = \{[_{ijke} b_l R_{b_l}] \mid (s_i, a_j, b_l, s_k, e) \in H\} \cup \{[_{ijke}] \mid (s_i, a_j, \epsilon, s_k, e) \in H\}.$$

Now let

$$\Sigma_{aj} = \{/_{a_j}\} \bigcup_{i,k,e} \Sigma_{ijke}$$

and let

$$L_{a_j} = /_{a_j} \left\{ \prod_{i,k,e} L_{ijke} \right\}^+,$$

where π denotes concatenation. By the closure properties of languages, L_{a_j} is a language if all the R_{b_i} are languages. Thus we can define a substitution τ by

$$\tau(\epsilon) = \epsilon \qquad \text{and} \qquad \tau(a_j) = L_{a_j},$$

and again $R' = \tau(R) = \bigcup_{x \in R} \tau(x)$ is a language by their closure properties under substitution (Ginsburg, 1966).

(3) The construction is now given for $S_2' = (K', \Sigma', \Delta', H', (s_0, 0))$ such that $\sigma(S_2(R)) = S_2'(\tau(R))$. Let

$$K' = K \cup (K \times \{-1, 0, 1\}) \bigcup_{b_l \in \Delta} K_{b_l} \cup \{q, \hat{q}\},$$

where we can assume $K \cap K_{b_i} = K_{b_i} \cap K_{b_j} = \phi$, $i \neq j$,

$$\Sigma' = \bigcup_{a_j \in \Sigma} \Sigma_{a_j},$$

$$\Delta' = \bigcup_{b_l \in \Delta} \Delta_{b_l}.$$

For convenience we also define the set $S = \bigcup_{a_j \in \Sigma} \{\Sigma_{a_j} - \{/_{a_j}\}\}$. H' contains the moves $H_{b_1} \cup \cdots \cup H_{b_t}$ in addition to the following:

1(a) $\{(s_i, a, \epsilon, s_i, 1) \mid s_i \in K, a \in S\}$,

(b) $\{(s_i, [_{ijke}, \epsilon, q, 1) \mid s_i \in K, [_{ijke} \in \Sigma'\}$;

2. $\{(q, b_l, \epsilon, s_0^{b_l}, 1) \mid b_l \in \Delta\}$;

3. $\{(p,], \epsilon, \hat{q}, -1) \mid p \in \{q\} \cup K_{b_l}\}$;

4. $\{(\hat{q}, a, \epsilon, \hat{q}, -1) \mid a \in \Sigma_{b_l} \cup \Delta\}$;

5(a) $\{(\hat{q}, [_{ijke}, \epsilon, (s_k, e), -1) \mid [_{ijke} \in \Sigma'\}$,

(b) $\{((s_k, e), a, \epsilon, (s_k, e), -1) \mid s_k \in K, a \in S, e \in \{-1, 0\}\}$,

(c) $\{((s_k, 1), a, \epsilon, (s_k, 1), 1) \mid s_k \in K, a \in S\}$;

6(a) $\{((s_k, e), /_{a_j}, \epsilon, s_k, 1) \mid s_k \in K, a_j \in \Sigma, e \in \{0, 1\}\}$,

(b) $\{((s_k, -1), /_{a_j}, \epsilon, (s_k, 0), -1) \mid s_k \in K, a_j \in \Sigma\}$.

Rules (1) are used to locate any $[_{ijke}$ for a given i and j, blocking if $a /_{a_j}$ is read. b_l is located by rule (2), and S_2' duplicates the behavior of $S_2^{b_l}$ which terminates by an application of rule (3). Then $[_{ijke}$ is located again by rule (4), and S_2' simulates the behavior of S_2 by rules (5) and (6). In all other cases, S_2' blocks so that indeed it produces the required substitution.

COROLLARY 3.2. *Deterministic* 2st *mappings of languages are closed under substitution.*

Proof. If S_2 is deterministic, it does not loop. Therefore L_{a_j} can be constructed to have the form

$$L_{a_j} = /_{a_j} \prod_i L_{ijke},$$

where again π has been used to denote concatenation. Further, in L_{a_j} there is only one $[_{ijke}$ with subscript i, so that rule 1(a) is made deterministic by specifying that $(s_i, a, \epsilon, s_i, 1) \in H'$ if $s_i \in K$, $a \in S - \{[_{ijke}\}$. Thus S_2' deterministically picks out the correct string $x_{b_l} \in R_{b_l}$, and since each $S_2^{b_l}$ is deterministic by hypothesis, it follows that $S_2'(\tau(R))$ is deterministic.

Since homomorphism[2] is a special case of substitution, we have the following corollary:

COROLLARY 3.3. *Deterministic and nondeterministic* 2st *mappings of languages are closed under arbitrary homomorphism.*

COROLLARY 3.4. *Deterministic and nondeterministic* 2st *mappings of languages are closed under the operations of* $\cdot$, $\cup$, *and* $*$.

[2] Let $L \subseteq \Sigma^*$. $\tau(L)$ is a *homomorphism* if $\tau(\epsilon) = \epsilon$, $\tau(a) = x$ for some $x \in \Sigma_a{}^*$, $\tau(a_1 \cdots a_k) = \tau(a_1) \cdots \tau(a_k)$ and $\tau(L) = \bigcup_{y \in L} \tau(y)$.

Proof. Let $\Sigma = \{a, b\}$. The languages $\{ab\}$, $\{a, b\}$, and $\{a^*\}$ are trivial deterministic 2st mappings of a regular set. The corollary follows by closure under substitution into these languages.

THEOREM 3.3. *It is decidable for an arbitrary* 2st S_2 *and language* R *whether* $S_2(R) = \phi$.

Proof. Let h be the homomorphism defined by $h(a) = \epsilon$ for all $a \in \Sigma$. $h(S_2(R))$ is a 2st mapping S_2' of a language R' by Corollary 3.3, and $S_2'(R') \neq \phi$ if and only if ϵ is in $S_2'(R')$ which is decidable by Lemma 3.1.

THEOREM 3.4. 2st *mappings of languages are properly contained in the family of context sensitive languages.*

Proof. Every recursively enumerable set E can be expressed as $E = h(D_1 \cap D_2)$, where D_1 and D_2 are context free languages and h is a homomorphism (Ginsburg, 1967a). Now there exists a recursively enumerable set T which is not context sensitive (Davis, 1958). Since $D_1 \cap D_2$ is context sensitive, $T = h(D_1 \cap D_2)$ contradicts that the context sensitive languages are closed under homomorphism. Since 2st mappings of languages are closed under homomorphism by Corollary 3.3, there must be a context sensitive language which is not a 2st mapping of a language.

Lemma 3.1 is strengthened by the following theorem.

THEOREM 3.5. 2st *mappings of languages are recursive.*

Proof. By Theorem 3.1 each 2st mapping of a language is context sensitive and the lba A accepting $S_2(R)$ can be effectively constructed. Furthermore, R' in Corollary 3.1 can be effectively determined from S_1 and R, and from A a context sensitive grammar $G = (N, T, P, \sigma)$ such that $L(G) = T(A)$ can be effectively determined (Kuroda, 1964). But it is solvable to determine whether $\sigma \overset{*}{\Rightarrow} x$, for $x \in \Sigma^*$ (Landweber, 1964).

THEOREM 3.6. *Let* $L = S_2(R)$ *be a nondeterministic* (*deterministic*) 2st *mapping of a language and let* R' *be a regular set. Then* $L \cap R' = S_2'(R\not{\emptyset})$ *is a nondeterministic* (*deterministic*) 2st *mapping of a language.*

Proof. Suppose $S_2 = (K, \Sigma, \Delta, H, s_0)$ and let $A = (J, \Delta, \delta, q_0, F)$ be the deterministic finite state acceptor for R', and by Lemma 2.4 assume that

if $(p, a, y, q, e) \in H$, then $|y| \leqslant 1$. Then $L \cap R' = S_2'(R\not{S})$, where $S_2' = (K', \Sigma', \Delta, H', s_0')$ and where

$$K' = (K \times J),$$
$$\Sigma' = \Sigma \cup \{\not{S}\},$$
$$s_0' = (s_0, q_0)$$

and where H' has the following moves:

(1) $\{((p, s), a, y, (q, \delta(s, y)), e) \mid (p, a, y, q, e) \in H, y \neq \epsilon, s \in J\}$;

(2) $\{((p, s), a, \epsilon, (q, s), e) \mid (p, a, \epsilon, q, e) \in H, s \in J\}$;

(3) $\{((p, s), \not{S}, \epsilon, (p, s), 1) \mid p \in K, s \in F\}$.

Intuitively, S_2' is not allowed to move right from the input tape if $S_2'(R\not{S})$ is not also in R'. $R\not{S}$ is clearly a language if R is.

THEOREM 3.7a. *Nondeterministic* 2st *mappings of languages are closed under arbitrary* 1st *mappings.*

Proof. Let $L = S_2(R)$, where $S_2 = (K_1, \Sigma_1, \Delta_1, H_1, s_0)$ and let $S_1 = (K_2, \Delta_1, \Delta_2, H_2, q_0)$ be a 1st. We show that $S_1(L) = S_2'(R\not{S})$, where $S_2' = (K_3, \Sigma_1 \cup \{\not{S}\}, \Delta_2, H_3, (s_0, q_0, \epsilon))$, and $K_3 = K_1 \times K_2 \times \Omega$, where

$$\Omega = \bigcup_{i=0}^{s} \Delta_1^i$$

and

$$s = \max_x \{|x| \mid (p, x, y, q) \in H_2\}.$$

By Lemma 2.4 we assume that if $(p, a, y, q, e) \in H$, then $|y| \leqslant 1$. Then H_3 has the following moves:

1. $\{((r, p, \epsilon), a, y, (r, q, \epsilon), 0) \mid (p, \epsilon, y, q) \in H_2, r \in K_1, a \in \Sigma_1 \cup \{\not{S}\}\}$;

2(a) $\{((r, p, \epsilon), a, y, (r, q, x_1 \cdots x_n), 0) \mid (p, x_1 \cdots x_n, y, q) \in H_2, r \in K_1, a \in \Sigma\}$;

(b) $\{((r, p, x_1 \cdots x_k), a, \epsilon, (s, p, x_2 \cdots x_k), e) \mid (r, a, x_1, s, e) \in H_1, p \in K_2, x_1 \cdots x_k \in \Omega\}$;

3. $\{((r, p, x), a, \epsilon, (s, p, x), e) \mid (r, a, \epsilon, s, e) \in H_1, p \in K_2, x \in \Omega\}$;

4. $\{((r, p, \epsilon), \not{S}, \epsilon, (r, p, \epsilon), 1) \mid r \in K_1, p \in K_2\}$.

If S_1 makes moves on ϵ input, then rules (1) allow S_2' to make corresponding moves. If S_1 reads non-ϵ inputs, then S_2' simulates S_2 as it writes the outputs S_1 reads by rules (2). If S_2 makes moves on ϵ outputs, S_2' has the moves (3), and rules (4) allow S_2' to leave the input tape whenever S_1 is not reading in the middle of an input sequence and S_2 has left its tape.

LEMMA 3.3. *Let* $S_1 = (K, \Sigma, \Delta, H, s_0)$ *be a* 1st, *and for each sequence* $(p, x_1, y_1, s_1)(s_1, x_2, y_2, s_2) \cdots (s_{n-1}, x_n, y_n, q)$ *of moves in* S_1 *let* $(r, p, x_1 \cdots x_n, q)\, y_1 \cdots y_n$ *be a string in* $L_p{}^r \subseteq (T \times K \times \Omega \times K)\Delta^*$, *where*

$$\Omega = \bigcup_{i=0}^{s} \Delta^i,$$

$$s = \max_x \{|\, x \,|\, | \, (p, x, y, q) \in H\},$$

and T *is a set. Then* $L_p{}^r$ *is regular.*

Proof. Let $G_p{}^r = (N, T, P, \sigma)$ be a right linear grammar defined as follows:

(1) $N = \{\sigma\} \cup \{[r, p, x, q] \mid p, q \in K, x \in \Omega\}$;

(2) $T = \{r\} \times K \times \Omega \times K$;

(3) P has the productions

$$\{\sigma \to (r, p, x_1 \cdots x_n, q)[r, p, x_1 \cdots x_n, q] \mid x \in \Omega, q \in K_2\},$$

$$\{[r, p, x_i \cdots x_j \cdots x_n, q] \to y[r, s, x_j \cdots x_n, q] \mid (p, x_i \cdots x_{j-1}, y, s) \in H\},$$

$$[r, q, \epsilon, q] \to \epsilon.$$

Clearly $L(G_p{}^r) = L_p{}^r$ is regular and $L_p{}^r \neq \phi$.

THEOREM 3.7b. *Deterministic* 2st *mappings of languages are closed under arbitrary* 1st *mappings.*

Proof. Again let $L = S_2(R)$, where $S_2 = (K_1, \Sigma_1, \Delta_1, H_1, s_0)$ and let $S_1 = (K_2, \Delta_1, \Delta_2, H_2, q_0)$ be a 1st. We show that $S_1(L) = S_2'(\tau(R\not{S}))$, where τ is a substitution. Let $S_2' = (K_3, \Sigma_2 \cup \{/_{\not{S}}\}, \Delta_2, H_3, (s_0, q_0, \epsilon, 0))$ and $K_3 = K_1 \times K_2 \times (\Omega \cup \{\not{S}, 0\}) \times \{T, \hat{T}, \not{S}, -1, 0, 1\}$, where

$$\Omega = \bigcup_{i=0}^{s} \Delta_1{}^i$$

and

$$s = \max_x \{ | x | \mid (p, x, y, q) \in H_2 \}.$$

By Lemma 2.4 we assume that if $(p, x, y, q, e) \in H_1$, then $| y | \leqslant 1$.

First the substitution τ is defined. Let $L_p{}^r$ be as in Lemma 3.3, where $T = K_1$, $K = K_2$, with $r \in K_1$, $p \in K_2$. Let L_a be the language

$$L_a = /_a \prod_{r \in K} \prod_{p \in K} L_p{}^r,$$

with $\Sigma_a = \{/_a\} \cup \{r, p, x, q) \mid r \in K_1, p, q \in K_2$, and $x \in \Omega\} \cup \Delta_2$. Here π is again used to denote concatenation of languages. Then $\tau(a) = L_a$ for all a in $\Sigma_1 \cup \{\mathcal{S}\}$ is a substitution of regular sets into a language which preserves languages by (Ginsburg, 1966).

Now for convenience define $\chi = \{/_a \mid a \in \Sigma_1\}$ and let $\Sigma_2 = \bigcup_{a \in \Sigma_1} \Sigma_a$. H_3 then has the following moves for all $r \in K_1$, $p \in K_2$:

1(a) $\{((r, p, 0, T), a, \epsilon, 1) \mid a \in \Sigma_2 - \{(r, p, x, q) \mid x \in \Omega, q \in K_2\}\}$,
(b) $\{((r, p, 0, T), (r, p, x, q), \epsilon, (r, q, x, T), 1) \mid x \in \Omega, q \in K_2\}$;

2(a) $\{((r, p, x, T), b, b, (r, p, x, T), 1) \mid b \in \Delta_2, x \in \Omega\}$,
(b) $\{((r, p, x, T), a, \epsilon, (r, p, x, T), -1) \mid a \in \Sigma_2 - \Delta_2, x \in \Omega\}$;

3(a) $\{((r, p, x, e), a, \epsilon, (r, p, x, e), -1) \mid a \in \Sigma_2 - \chi, e \in \{0, -1\}, x \in \Omega\}$,
(b) $\{((r, p, x, 1), a, \epsilon, (r, p, x, 1), 1) \mid a \in \Sigma_2 - \chi, x \in \Omega\}$,
(c) $\{((r, p, x_1 \cdots x_n, e), /_a, \epsilon, (s, p, x_2 \cdots x_n, e') \mid e \in \{-1, 0, 1, \hat{T}\}$,
$(r, a, x_1, s, e') \in H_1, x \in \Omega - \{\epsilon\}\}$,
(d) $\{((r, p, x, e), /_a, \epsilon, (s, p, x, e'), e') \mid e \in \{-1, 0, 1, \hat{T}\}$,
$(r, a, \epsilon, s, e') \in H_1, x \in \Omega\}$,
(e) $\{((r, p, \epsilon, e), /_a, \epsilon, (r, p, 0, T), 1) \mid /_a \in \chi, e \in \{-1, 0, 1\}$,
(r, a, ϵ, s, e') not in $H_1\}$;

4(a) $\{((r, p, \epsilon, 1), /_{\mathcal{S}}, \epsilon, (r, p, \mathcal{S}, T), 1)\}$,
(b) $\{((r, p, \mathcal{S}, T), a, \epsilon, (r, p, \mathcal{S}, T), 1) \mid a \in \Sigma_2$
$- \{(r, p, x, q) \mid x \in \Omega - \{\epsilon\}, q \in K_2\}\}$,
(c) $\{((r, p, \mathcal{S}, T), (r, p, \epsilon, q), \epsilon, (r, q, \mathcal{S}, \mathcal{S}), 1) \mid q \in K_2\}$,
(d) $\{((r, p, \mathcal{S}, \mathcal{S}), b, b, (r, p, \mathcal{S}, \mathcal{S}), 1) \mid b \in \Delta_2\}$,
(e) $\{((r, p, \mathcal{S}, \mathcal{S}), a, \epsilon, (r, p, \epsilon, 1), 1) \mid a \in \Sigma_2 - \Delta_2\}$.

Informally, when S_2 is in state r reading a and S_1 is in state p then S_2' scans L_a by rules (1) to see what the next moves of S_1 are on finite input. By rules (2), S_2' writes the output which S_1 writes on a finite input which is remembered in the state of S_2'. When S_2' has an input sequence of S_1 in its state, S_2' simulates S_2 by rules (3) until that sequence is written by S_2.

Rules (4) are the only ones which allow S_2' to leave the input tape. S_1 must not be in the middle of an input sequence when S_2 leaves its tape, and since S_1 may write outputs on ϵ input after leaving a tape, rules (4) allow S_2' to write the same outputs.

Let $a_1 \cdots a_n \in R$ and let $x_{a_i} \in L_{a_i}$, where a_i denotes the ith symbol of $a_1 \cdots a_n$, $1 \leqslant i \leqslant n$. S_2' scans x_{a_i} only if S_1 is not in the middle of an input sequence, S_2 has not further moves on ϵ output, and S_2' has simulated a move of S_2 since the last time it scanned x_{a_j}, $1 \leqslant j \leqslant n$. Since S_2 is deterministic, no x_{a_i} is scanned twice for the same entry. If the input of S_1 does not correspond exactly with the output of S_2, S_2' blocks by not finding moves in 3(c) or 3(d). Thus if $z \in S_2'(\tau(a_1 \cdots a_n \not{S}))$, then $z \in S_1(S_2(a_1 \cdots a_n))$.

On the other hand, if $y \in S_2(a_1 \cdots a_n)$, then $S_1(y) \in S_2'(\tau(a_1 \cdots a_n \not{S}))$. For suppose $y = y_1 \cdots y_t$ and suppose that y_j, $1 \leqslant j \leqslant t$, is the last symbol written after S_2 moves to a_j, $1 \leqslant i \leqslant n$, and y_j terminates an input sequence of S_1. Then there is an $x_{a_i} \in L_{a_i}$ specifying the next moves S_1 makes on the next finite segment of y, hence by induction on j, a tape z in $\tau(a_1 \cdots a_n \not{S})$ such that $S_2'(z) \in S_1(y)$.

THEOREM 3.8. *Deterministic and nondeterministic* 2st *mappings of languages are closed under* h^{-1}.[3]

Proof. Let $L = S_2(R)$ for some language R and 2st $S_2 = (K, \Sigma, \Delta, H, s_0)$, and let h be a homomorphism from Δ' into Δ^*. Let $S_1 = (\{s\}, \Delta', \Delta, H', s)$, where $H = \{(s, h(a), a, s) \mid a \in \Delta\}$. $x \in S_1(h(x))$ for all $x \in \Delta^*$ and by Theorem 3.7 there is a 2st S_2' and language R' such that $S_2'(R') = h^{-1}(S_2(R))$.

THEOREM 3.9a. *Nondeterministic* 2st *mappings of languages are closed under reversal.*[4]

Proof. Let $L = S_2(R)$ for some language R and 2st $S_2 = (K, \Sigma, \Delta, H, s_0)$ and define $S_2' = (K \cup \{p_0, f\}, \Sigma \cup \{\not{S}\}, \Delta, H', p_0)$ so that $[S_2(R)]^R = S_2'(R^R \not{S})$. $R^R \not{S}$ is a language if R is and H' has the following moves:

1. $\{(p, a, y^R, q, e) \mid (q, a, y, p, e) \in H\}$;
2. $\{(p_0, a, y^R, p, 0) \mid a \in \Sigma$ and $(p, a, y, q, 1) \in H\}$;

3(a) $\{(s_0, a, \epsilon, f, 1) \mid a \in \Sigma\}$,

(b) $\{(f, \not{S}, \epsilon, f, 1)\}$.

[3] Let h be a homomorphism. $h^{-1}(L) = \{x \mid h(x) \in L\}$ is called the inverse homomorphism of L.

[4] Let $x = x_1 \cdots x_k$, $x_i \in \Sigma$, $1 \leqslant i \leqslant k$. $x^R = x_k \cdots x_1$ is called the reversal of x. $L^R = \{x^R \mid x \in L\}$ is called the reversal of L.

Rules (1) allow S' to do the reverse of S_2, and rules (2) allow S_2' to start in the proper state. Rules (3) ensure that S_2' leaves the right end of tape only if S_2' has actually written the reversal of the corresponding string in $S_2(R)$, which can occur if and only if $p_0x_n \cdots x_1\mathcal{S} \overset{*}{\vdash} x_n \cdots x_2s_0x_1\mathcal{S} \vdash x_n \cdots x_1f\mathcal{S} \vdash x_n \cdots x_1\mathcal{S}f$.

The result for deterministic 2st is not so simple as for nondeterministic 2st since the predecessor of the id of a 2st may not be unique.

THEOREM 3.9b. *Deterministic* 2st *mappings of languages are closed under reversal.*

Proof. Let $L = S_2(R)$ for some language R and $S_2 = (K, \Sigma, \Delta, H, s_0)$ a deterministic 2st with asm $A_1 = (K, \Sigma, \delta_1)$. From $T_1(p, q)$ and $T_2(p, q)$ of Theorem 2.1 it is possible to calculate for any $x \in \Sigma^*$ the function $\tau_x : K \cup \{\hat{s}_0\} \to K \cup \{0\}$ so that $\tau_x(s_i) = s_j$ whenever $x = x_1 \cdots x_n$ and $x_1 \cdots x_{n-1}s_ix_n \overset{*}{\vdash} xs_j$, and $\tau_x(\hat{s}_0) = s_j$ whenever $s_0x \overset{*}{\vdash} xs_j$. There are at most $(n+1)^{(n+1)}$ functions τ_x. Note that τ_x is exactly the same as in Rabin and Scott (1959).

(1) Let $R_1 = S_1{}^1(R)$, where $S_1{}^1 = (K_1, \Sigma, \Sigma \times K_1, J_1, \tau_\epsilon)$ and where

$$K_1 = \{\tau_x \mid x \in \Sigma^*\},$$

$$J_1 = \{(s, a, (a, s), p) \mid s = \tau_{x_i}, p = \tau_{x_j}, a \in \Sigma, \text{ and } x_j = x_ia\}.$$

$S_1{}^1$ is constructable since τ_{x_j} depends only on τ_{x_i} and a.

(2) Define the asm $A_2 = (K, \Sigma, \delta_2)$ so that $\delta_2(p, a) = (q, -e)$ if and only if $\delta_1(p, a) = (q, e)$. Define the function $\rho_x : K \to K \cup \{0\}$ so that $\rho_x(s_i) = s_j$ whenever $s_ix \overset{*}{\vdash} s_jbx$. There are at most n^n functions ρ_x. Let $R_2 = S_1{}^2(R_1)$, where $S_1{}^2 = (K_2, \Sigma \times K_1, K_2 \times \Sigma \times K_1, J_2, \rho_\epsilon)$ and where

$$K_2 = \{\rho_x \mid x \in \Sigma^*\},$$

$$J_2 = \{(s, (a, b), (s, a, b), p) \mid s = \rho_{x_i}, p = \rho_{x_j}, a \in \Sigma, b \in K_1, x_j = ax_i\}.$$

Thus for each tape $y_1 \cdots y_k \cdots y_n$ in R there is a corresponding tape $[(\tau_\epsilon, y_1, \rho_{y_2 \cdots y_n}) \cdots (\tau_{y_1 \cdots y_{k-1}}, y_k, \rho_{y_{k+1} \cdots y_n}) \cdots (\tau_{y_1 \cdots y_{n-1}}, y_n, \rho_\epsilon)]^R$ in R_2.

(3) Next the construction of $S_2' = (K', K_2 \times \Sigma \times K_1, \Delta, H', p_0)$ such that $[S_2(R)]^R = S_2'(R_2)$ is given. Clearly, R_2 is a language if R is a language. Let $K' = (K \times \{-1, 0, 1\}) \cup \{p_0\}$. H' is constructed in the following way.

Suppose that S_2' is reading $x_k = (\tau_{y_1 \cdots y_{k-1}}, y_k, \rho_{y_{k+1} \cdots y_n})$ while in state (s_j, e). This indicates that S_2 is reading y_k and that its next transition is to

state s_j with direction e. What is needed is to determine from x_k and s_j the present state s_i of S_2 and the direction e' of its previous move; this will enable S_2' to trace deterministically in reverse the moves of S_2.

Now since S_2 is deterministic it can transit to y_k at most $m = |K|$ times. Thus there is a unique sequence $s_{i_1} = \tau_{y_1 \cdots y_{k-1}}(\hat{s}_0), s_{i_2}, \ldots, s_{i_r}$ of states in which S_2 transits to y_k, taken in order, $1 \leqslant r \leqslant m$, and such that $(s_{i_r}, y_k, y, s_j, e) \in H$. Furthermore, this sequence can easily be determined from x_k and (s_j, e). But in determining this sequence of states the direction e' of the transition to y_k in state s_{i_r} is known. Thus H' has the following moves:

1(a) $\{((s_j, e), (\tau_{y_1 \cdots y_{k-1}}, y_k, \rho_{y_{k+1} \cdots y_n}), y^R, (s_{i_r}, e'), e') \mid s_{i_r} \neq s_0$, when $y_1 \cdots y_{k-1} \neq \epsilon\}$;

(b) $\{((s_j, e), (\tau_\epsilon, y_1, \rho_{y_2 \cdots y_n}), y^R, (s_0, e), 1) \mid (s_0, y_1, y, s_j, e) \in H\}$.

Similarly the first move of S_2' is determined by the last two moves of S_2, and included in H' are the following moves:

2(a) $\{(p_0, (\tau_{y_1 \cdots y_{n-1}}, y_n, \rho_\epsilon), y^R, (s_i, e), e) \mid (s_i, y_n, y, \tau_{y_1 \cdots y_n}(\hat{s}_0), 1) \in H$, $s_i \neq s_0$ when $y_1 \cdots y_{n-1} \neq \epsilon$ and e s the direction of the 2nd last move of $S_2\}$;

(b) $\{(p_0, (\tau_\epsilon, y_1, \rho_\epsilon), y^R, (s_0, e), 1) \mid (s_0, y_1, y, s_j, 1) \in H\}$.

Regarding the construction just given notice that if S_2 does not leave the right end of $x \in R$, then S_2' does not leave the right end of the corresponding tape $x' \in R_2$, since $\tau_x(\hat{s}_0) = 0$.

THEOREM 3.10. *It is undecidable whether the intersection of two (deterministic) 2st mappings of a regular set is empty.*

Proof. Let $J = \{1, \ldots, n\}$ be the first n integers and let $\Sigma = J \cup \{\not\$, \not\phi\}$. Let $x_1, \ldots, x_n, y_1, \ldots, y_n \in \Delta^+$ be words over an alphabet Δ with at least 2 elements. Let R be the regular set $\not\$ J^+ \not\phi$ and define the set T as follows:

(1) $\{(s_0, \not\$, \epsilon, s_1, 1), (s_0, \not\phi, \epsilon, s_1, -1)\}$.

(2) $\{(s_0, i, i, s_0, 1), (s_1, i, \epsilon, s_1, -1) \mid i \in J\}$.

Next let

$$S_2^1 = (K_1, \Sigma, \Delta, H_1, s_0), \qquad \text{where} \quad K_1 = \{s_0, s_1, q\}$$

and let

$$S_2^2 = (K_2, \Sigma, \Delta, H_2, s_0), \qquad \text{where} \quad K_2 = \{s_0, s_1, r\},$$

where

$$H_1 = T \cup \{(s_1, \not\$, \epsilon, q, 1), (q, \not\phi, \epsilon, q, 1)\} \cup \{(q, i, x_i, q, 1) \mid i \in J\},$$
$$H_2 = T \cup \{(s_1, \not\$, \epsilon, r, 1), (r, \not\phi, \epsilon, r, 1)\} \cup \{(r, i, y_i, r, 1) \mid i \in J\}.$$

Now each word in $S_2^1(R)$ has the form $i_1 \cdots i_t x_{i_1} \cdots x_{i_t}$ and each word in $S_2^2(R)$ has the form $i_1 \cdots i_s y_{i_1} \cdots y_{i_s}$. But $S_2'(R) \cap S_2^2(R) \neq \phi$ if and only if there are integers $i_1 \cdots i_t$ such that $i_1 \cdots i_t x_{i_1} \cdots x_{i_t} = i_1 \cdots i_t y_{i_1} \cdots y_{i_t}$, i.e., such that Post's problem has a solution, which is unsolvable.

COROLLARY 3.5. *Nondeterministic and deterministic* 2st *mappings are not closed under intersection.*

Proof. Suppose the contrary. Then the emptiness question for 2st mappings of languages must not be solvable, contradicting Theorem 3.3.

DEFINITION. Two 2st, S_2 and S_2', are said to be *equivalent* if for all $x \in \Sigma^*$, $S_2(x) = S_2'(x)$. Let $S_2 = (K, \Sigma, \Delta, H, s_0)$ be a 2st. S_2 is said to be *nondeterministic complete* if $(p, a, y, q, e) \in H$ implies that $y \in \Delta$.

Since equivalence is unsolvable for nondeterministic generalized machines (Griffiths, 1968), equivalence is unsolvable for nondeterministic 2st. However, the equivalence problem is solvable for 1-way nondeterministic complete machines (Griffiths, 1968).

THEOREM 3.11. *It is unsolvable to determine for arbitrary nondeterministic complete* 2st A *and* B *whether* $A(x) = B(x)$ *for all* $x \in \Sigma^*$.

Proof. Let $J = \{1, \ldots, n\}$ and let $\Sigma = J \cup \{\not\$\}$. Let $x_1, \ldots, x_n, y_1, \ldots, y_n \in \Delta^+$ be words over an alphabet Δ with at least 2 elements, and let

$$s = \max(|x_1|, \ldots, |x_n|, |y_1|, \ldots, |y_n|).$$

Let $B = (Q, \Sigma, \Delta, H, q_0)$ be the 2st defined by

$$B(xi_1 \cdots i_k z) = \{*^\alpha y *^\beta \mid k \leqslant |y| \leqslant sk\}$$

for all $x \in J^*\mathcal{S}$, $z \in \mathcal{S}\Sigma^*$, and where $\alpha = |x| + 2k + 4$, $\beta = |z| + 1$. Let $Q = \{q_0, q_1, q_2, q_3, f\}$ and let H have the following moves:

1(a) $\{(q_0, j, *, q_0, 1), (q_1, j, *, q_1, 1), (q_2, j, *, q_2, -1) \mid j \in J\}$,
(b) $\{(q_0, \mathcal{S}, *, q_1, 1), (q_1, \mathcal{S}, *, q_2, -1)\}$,
(c) $\{(f, a, *, f, 1) \mid a \in \Sigma\}$;

2. $\{(q_2, \mathcal{S}, **, q_3, 1), (q_3, \mathcal{S}, *, f, 1)\}$;

3. $\{(q_3, j, x, q_3, 1) \mid x \in \Sigma^+ \text{ and } 1 \leqslant |x| \leqslant s\}$.

Rules (1) and (2) check whether the input tape t has written on it two $\mathcal{S}$ symbols. If not $B(t) = *^{|t|}$, and if so the desired output is written by rules (3).

Next define S_2^1 and S_2^2 so that

$$S_2^1(xi_1 \cdots i_k z) = B(xi_1 \cdots i_k z) - \{*^\alpha x_{i_1} \cdots x_{i_k} *^\beta\}$$

and

$$S_2^2(xi_1 \cdots i_k z) = B(xi_1 \cdots i_k z) - \{*^\alpha y_{i_1} \cdots y_{i_k} *^\beta\}$$

with $S_2^1(t) = S_2^2(t) = B(t)$ otherwise. Now $S_2^1 = (K, \Sigma, \Delta, L, q_0)$, where $K = Q \cup \{p_1, \ldots, p_n, +, -\}$. L has the moves (1) of H and the following additional moves:

(1) $\{(q_3, j, *, p_j, 1), (p_j, k, x_j, p_k, 1) \mid j, k \in J\}$.

(2) $\{(p_j, k, x, -, 1) \mid 2 \leqslant |x| \leqslant |x_j x_k|, x \neq x_j x_k, j, k \in J\}$.

(3) $\{(p_j, k, x, +, 1) \mid |x_j x_k| \leqslant |x| \leqslant 2s, x \neq x_j x_k, j, k \in J\}$.

(4) $\{(p_j, \mathcal{S}, x*, f, 1) \mid 1 \leqslant |x| \leqslant s, x \neq x_j, j \in J\}$.

(5) $\{(-, j, x, -, 1) \mid 1 \leqslant |x| \leqslant |x_j|, x \neq x_j, j \in J\}$.

(6) $\{(+, j, x, +, 1) \mid |x_j| \leqslant |x| \leqslant s, x \neq x_j, j \in J\}$.

(7) $\{(q_2, \mathcal{S}, *, q_3, 1), (q_3, \mathcal{S}, **, f, 1), (-, \mathcal{S}, *, f, 1), (+, \mathcal{S}, *, f, 1)\}$.

S_2^1 is identical to B except that it makes "mistakes" by rules (2) or (3), deciding at that point whether the output it writes from $i_1 \cdots i_k$ should be shorter or longer than that which B writes. $S_2^2 = (K', \Sigma, \Delta, L', q_0)$ is defined in exactly the same way.

Next define $A = (K \cup K', \Sigma, \Delta, M, q_0)$ so that $A(x) = S_2^1(x) \cup S_2^2(x)$, where we can take $M = L \cup L' \cup \{(q_0, \mathcal{S}, *, q', 1)\}$. $A(x) \neq B(x)$ for some $x \in \Sigma^*$ if and only if there are integers $i_1 \cdots i_k$ with $k \geqslant 1$ such that $x_{i_1} \cdots x_{i_k} = y_{i_1} \cdots y_{i_k}$, i.e., such that Post's problem has a solution. By

Lemma 2.4 both A and B can be made complete, since Lemma 2.4 introduces no ϵ rules.

COROLLARY 3.6. *It is unsolvable to determine for arbitrary nondeterministic complete* 2st A *and* B *whether* $A(\Sigma^*) = B(\Sigma^*)$.

Proof. Let A and B be as in Theorem 3.11 except replace Δ by $\Delta \cup J$ and change the 2nd rules in 1(a) of H from $(q_1, j, *, q_1, 1)$ to $(q_1, j, j, q_1, 1)$. Now

$$B(xi_1 \cdots i_k z) = (*^{\alpha_1} i_1 \cdots i_k *^{\alpha_2} y *^{\beta}),$$

where $k \leqslant |y| \leqslant sk$, $\alpha_1 = |x| + 1$, $\alpha_2 = k + 3$, $\beta = |z| + 1$. Then

$$S_2^1(xi_1 \cdots i_k z) = B(xi_1 \cdots i_k z) - \{*^{\alpha_1} i_1 \cdots i_k *^{\alpha_2} x_{i_1} \cdots x_{i_k} *^{\beta}\},$$

$$S_2^2(xi_1 \cdots i_k z) = B(xi_1 \cdots i_k z) - \{*^{\alpha_1} i_1 \cdots i_k *^{\alpha_2} y_{i_1} \cdots y_{i_k} *^{\beta}\}.$$

Then $A(\Sigma^*) \neq B(\Sigma^*)$ if and only if there are integers $i_1, \ldots, i_k$ with $k \geqslant 1$ such that $x_{i_1} \cdots x_{i_k} = y_{i_1} \cdots y_{i_k}$ which is unsolvable.

So far it has been shown that 2st transductions of languages are properly contained within the context sensitive languages. We would also like to show that these are fundamental differences between deterministic and nondeterministic transductions.

LEMMA 3.4. *Let* R *be a regular set such that* $R = T(A_1)$ *and* $A_1 = (K_1, \Sigma, \delta_1, s_0, F)$ *is its (deterministic) finite acceptor. Let* $S_2 = (K_2, \Sigma, \Delta, H, s_0)$ *be a* 2st *with* asm $A_2 = (K_2, \Sigma, \delta_2)$. *Let* σ *and* τ *be the functions defined as in Theorem* 2.1 *where*

$$\sigma : (K_1 \cup \hat{K}_1) \times (K_1 \cup \hat{K}_1) \to \{0, 1\} \text{ and } \tau : (K_2 \cup \hat{K}_2) \times (K_2 \cup \hat{K}_2) \to \{0, 1\}$$

are defined for A_1 *and* A_2, *respectively. Then*

$$x \equiv y \quad \text{if and only if} \quad \sigma_x = \sigma_y \quad \text{and} \quad \tau_x = \tau_y \tag{3.1}$$

is a congruence relation of finite index δ. *Furthermore,* $\delta \leqslant |K_1|^{|K_1|}(2|K_2|)^{2|K_2|}$ *if* S_2 *is deterministic and* $\delta \leqslant |K_1|^{|K_1|} 2^{4|K_2|^2}$ *if* S_2 *is nondeterministic.*

Proof. The proof is identical to that of Theorem 2.1. Since A_1 is one-way there are at most $|K_1|^{|K_1|}$ distinct functions σ_x. If A_2 is deterministic there are at most $(2|K_2|)^{2|K_2|}$ distinct functions τ so that δ has the upper bound given in the lemma.

In the following theorem let $e_x(z, a)$ denote the total number of symbols which a 2st writes while reading symbols of z as it computes $S_2(x)$ where $x = x_1 z x_2$ and $x_1, x_2, z \in \Sigma^*$.

LEMMA 3.5. *$L = \{a^{n^2} \mid n \geqslant 1\}$ is not a* 2st *mapping of a regular set.*

Proof. Suppose the contrary, that $L = S_2(R)$ for some 2st $S_2 = (K, \Sigma, \Delta, H, s_0)$ and a regular set R. Now if $a^{n^2} \in S_2(x)$ and S_2 writes a^t while in a loop on x, then $a^{kt}a^{n^2} \in L$ for all $k \geqslant 1$, so that $t = 0$. Therefore Lemma 2.2 holds here even though S_2 is nondeterministic, and $e_x(z, a) \leqslant \alpha \mid x \mid$. Now let $\equiv$ be the congruence (3.1) and let δ be as in Lemma 3.4. Suppose $x = x_1x_2 \in R$ is a tape with $k = e_x(x_1, a)$, $\alpha\delta < k \leqslant 2\alpha\delta$, and suppose $a^{n^2} \in S_2(x)$ with $n > k$. x_1 can easily be chosen in this way because of the definition of α. Now $\mid x_1 \mid > \delta$, and let $x_1' \equiv x_1$ be the minimal length sequence in its congruence class so that $\mid x_1' \mid \leqslant \delta$. Let $x' = x_1'x_2$. By hypothesis $S_2(x') \subseteq L$. But now $0 \leqslant e_{x'}(x_1', a) < k$ so that $S_2(x')$ has a tape y such that $n^2 > \mid y \mid \geqslant n^2 - k > (n-1)^2$, and y cannot be in L as assumed.

LEMMA 3.6. *$L = \{(0^n1^n0^n)^* \mid n \geqslant 1\}$ is not a deterministic* 2st *mapping of a regular set.*

Proof. Suppose the contrary, that $L = S_2(R)$ for some deterministic 2st $S_2 = (K, \Sigma, \Delta, H, s_0)$ and a regular set R. By Lemma 2.1 and the fact that S_2 is deterministic, S_2 never loops, and by Lemma 2.2, $e_x(z, 0) + e_x(z, 1) \leqslant \alpha \mid z \mid$. Now let $\equiv$ be the congruence (3.1) and let δ be as in Lemma 3.4. Suppose $x = x_1zx_2 \in R$ is a tape with $k' = e_x(z, 0) + e_x(z, 1)$, $2\alpha\delta > k' > \alpha\delta$, and suppose $(0^n1^n0^n)^k = S_2(x)$ for some fixed $n > 2\alpha\delta$ and some $k > \alpha\delta$. Now $\mid z \mid > \delta$ and let $z' \equiv z$ be the minimal length sequence in its congruence class so that $\mid z' \mid \leqslant \delta$. Let $x' = x_1z'x_2$. By hypothesis $S_2(x') \in L$ so that $S_2(x') = (0^{n'}1^{n'}0^{n'})^{k''}$ with $n' < n$ or $k'' < k$, since $\mid S_2(x') \mid < \mid S_2(x) \mid$. Now suppose first that $n' < n$. If n decreases then at least $3\alpha\delta > k'$ digits must change and $n' = n$. Then it must be that $k'' < k$. If k decreases then at least $6\alpha\delta > k'$ digits must change, so that $k'' = k$ contradicting that $S_2(x') \in S_2(R)$.

THEOREM 3.12. *Deterministic* 2st *mappings of regular sets are properly contained within the nondeterministic* 2st *mappings of regular sets.*

Proof. $\{(0^n1^n0^n)^* \mid n \geqslant 1\}$ is easily shown to be a nondeterministic 2st mapping of the regular set $\mathcal{S}1^*\mathcal{S}$.

4. Transducers and Stack Automata

Recently Ginsburg *et al.* (1967a,b) have proposed a class of automata called *stack automata* as models for modern programming language compilers. They are of interest not only because of their structure but because they define languages with context sensitive properties.

Definition. A 1-*way stack automaton* (1sa) is a 7-tuple[5]

$$A = (K, \Sigma, \Gamma, \delta, q_0, Z_0, F),$$

where

(1) K is a finite, nonempty set of states,

(2) Σ is a finite, nonempty set of inputs,

(3) Γ is a finite, nonempty set of stack symbols,

(4) δ is a mapping from $K \times (\Sigma \cup \{\epsilon\}) \times \Gamma$ into finite subsets of $K \times \{-1, 0, 1\} \times \Gamma^*$,

(5) $q_0 \in K$ is the initial state,

(6) $Z_0 \in \Gamma$ is the initial stack symbol,

(7) $F \subseteq K$ is the set of final states.

An instantaneous description is an element of $(K \times \Sigma^* \times \Gamma^* \uparrow \Gamma^*)$, where $\uparrow$ is a stack pointer not in Γ. The id $(p, aw, xb \uparrow y)$ denotes the fact that A is in state p, the read head is scanning a on the input string aw, and the stack head is reading b. A move of A is denoted by $\vdash$ as usual with $\overset{*}{\vdash}$ its transitive closure. For convenience we use a symbol $\bar{Z}_i$ instead of $Z_i \in \Gamma$ when it is specifically intended that Z_i is the top stack symbol. δ is restricted in the following way:

(1) If (q, e, Z') is in $\delta(p, a, Z)$ and $Z \neq \bar{Z}$ then $Z' = Z$. Thus A may not write on the stack while the stack pointer is embedded in the stack.

(2) If $(q, e, Z_1 \cdots Z_k)$ is in $\delta(p, a, Z)$ then

$$(p, aw, YZ\uparrow) \vdash (q, w, YZ_1 \cdots Z_k\uparrow)$$

if and only if $e = 0$. Hence $k > 1$ implies $e = 0$.

(3) If (q, e, Z') is in $\delta(p, a, Z)$ then $(p, aw, YZ'\uparrow) \vdash (q, w, Y\uparrow Z)$ if and only if $Z = Z'$ and $e = -1$.

[5] This model is a variant of that used by Hopcroft and Ullman (1967).

A word w is accepted by A by final state, denoted $w \in T(A)$, if and only if

$$(q_0, w, Z_0 \uparrow) \overset{*}{\vdash} (f, \epsilon, Y \uparrow Z),$$

for any f in F. A word is accepted by A by empty stack, denoted $w \in N(A)$, if and only if

$$(q_0, w, Z_0 \uparrow) \overset{*}{\vdash} (f, \epsilon, \uparrow)$$

for any f in K.

THEOREM 4.1 (Hopcroft, 1968). *The family of* 1sa *accepting by empty stack is equivalent to the family of* 1sa *accepting by final state.*

Because of Theorem 4.1, in what follows a 1sa will be referred to simply as a 6-tuple by deleting the set F of final states. Next a new grammar is defined, and it will be shown that these grammars generate the 1sa languages. One other grammar has been reported (Schkolnick, 1968) which is related but which is more complicated to use and does not exhibit the pushdown automaton and 2st which are the principal substructures of the 1sa. Our approach is to use flags as in Aho (1968) to pass contextual information down a generation tree.

DEFINITION. A *stack grammar* is a 6-tuple $G = (N, I, T, F, P, \sigma)$ where

(1) N is a finite, nonempty set of nonterminal symbols,

(2) $I \subset N$ is a finite set of intermediate symbols,

(3) T is a finite, nonempty set of terminal symbols,

(4) F is a finite set of special symbols f of the form

$$f = [(A_1 \to \varphi_1), \ldots, (A_k \to \varphi_k)]$$

called *flags*, where $A_i \in I$, $\varphi_i \in T^* \cup T^*(I \times \{-1, 0, 1\})$,

(5) P is a finite set of productions of the form $A \to \Phi$, where $A \in N - I$, $\Phi \in (NF^* \cup T)^*$,

(6) $\sigma \in N - I$ is the starting symbol.

Each nonterminal in a G derivation may be indexed on the right by a flag string $f \in F^*$. Each $A \in N - I$ may be expanded by a P production $A \to \Phi$ in such a way that the indexing flag string at A distributes over all the nonterminals in Φ. Thus if $\Phi = A_1 f_1 \cdots A_n f_n$ where $A_i \in N \cup T$, $f_i \in F^*$, and $f_i = \epsilon$ if $A_i \in T$, then $Ag \Rightarrow A_1 f_1 g \cdots A_n f_n g$ where $f_i g = \epsilon$ if $A_i \in T$.

Intermediates may be expanded by a flag production $A_i \to \varphi_i$ in the flag adjacent to A_i on the right. In this case the intermediate may become embedded in the flag string, moving left if $\varphi_i \in T^*(I, 1)$, remaining stationary if $\varphi_i \in T^*(I, 0)$, and moving right if $\varphi_i \in T^*(I, -1)$.

The notation $A_i \to x(B, e)$ for a flag production is used for convenience. Alternatively, one might use $A_i \xrightarrow{L} xB$ in the case $e = 1$, $A_i \xrightarrow{R} xB$ if $e = -1$, and $A_i \to xB$ if $e = 0$. Any terminals produced by a flag production appear to the left of an intermediate and its entire flag string.

Each derivation in a stack grammar has the form of a tree in a derivation in a context free grammar with the following exceptions:

(1) Each nonterminal $A \in N - I$ in the tree is indexed on its right by a flag string $f \in F^*$.

(2) A node N in a path in a derivation tree terminates the derivation by P productions when $N \in T \cup IF^*$.

(3) A path in a derivation tree containing an intermediate symbol terminates whenever a flag production $A \to x$ for some x in T^* is applied to an intermediate A or when the intermediate moves left of the leftmost flag. When such a path terminates, the intermediate and its flags are removed or "absorbed."

EXAMPLE 4.1. Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar where P contains the productions $\sigma \to aAf_1Bf_1 \cdots f_nC$ and $B \to aBf_1C$. Then

$$\sigma \Rightarrow aAf_1Bf_1 \cdots f_nC \Rightarrow aAf_1aBf_1f_1 \cdots f_nCf_1 \cdots f_nC$$

as illustrated in Fig. 1a.

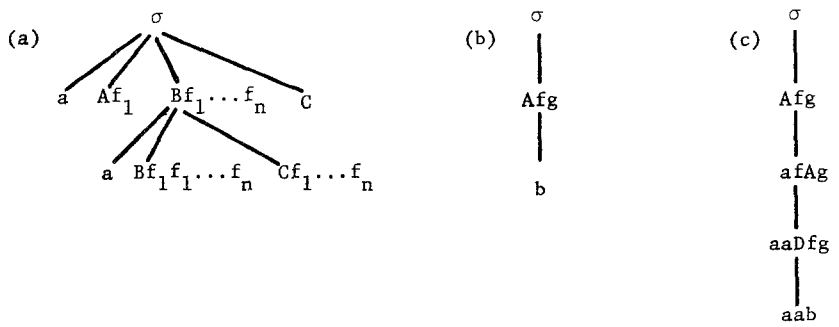

FIGURE 1

EXAMPLE 4.2. Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar where P contains the production $\sigma \to Afg$ and $f = [A \xrightarrow{L} bA, A \xrightarrow{R} aA, D \xrightarrow{L} b]$ and $g = [A \xrightarrow{L} aD]$ are the flags in F. Then

$$\sigma \Rightarrow Afg \Rightarrow b,$$
$$\sigma \Rightarrow Afg \Rightarrow afAg \Rightarrow aaDfg \Rightarrow aab$$

are G-derivations illustrated in Fig. 1b and 1c. If $g = [A \xrightarrow{R} aB]$, then $\sigma \Rightarrow b$ is the only G-derivation.

Now, a tree structure in a derivation brings up the possibility of making leftmost derivations as in a context free grammar. A derivation in a stack grammar $G = (N, I, T, F, P, \sigma)$ is a leftmost derivation if whenever

(1) $w_1 = xA\varphi$, $x \in T^*$, $A \in N - I$, $\varphi \in (N \cup F \cup T)^*$ and $w_1 \Rightarrow w_2$, then $w_1 \Rightarrow w_2$ by applying a P production $A \to \Phi$ to A.

(2) $w_1 = xAf\varphi$, $x \in T^*F^*$, $A \in I, f \in F$, $\varphi \in (N \cup F \cup T)^*$ and $w_1 \Rightarrow w_2$, then $w_1 \Rightarrow w_2$ by applying $A \to \psi$ in f to A.

Since two paths in a G-derivation tree are independent of each other except at their common vertices, it does not matter which path is completed first. Hence every sentence derivable in G is derivable by a leftmost derivation. Also, since there is no confusion, a node $A \in N - I$ of a derivation tree need not be indexed by all the flags appearing the nodes above it in the path of its derivation. At the first occurrence of an intermediate $B \in I$, the flags indexing B can readily be determined.

DEFINITION. Let $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0)$ be a lsa. A *stack scan* is a sequence of moves in A

$$(s_1, x_1w, X_1Y_1\uparrow) \vdash (s_2, x_2w, X_2\uparrow Y_2) \vdash \cdots \vdash (s_k, x_kw, X_kY_k\uparrow)$$

in which $x_k = \epsilon$, $X_iY_i \in \Gamma^*$, and $X_iY_i = X_jY_j$, $1 \leqslant i, j \leqslant k$.

DEFINITION. Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar. A derivation $w_1 \Rightarrow w_2$ in G is denoted by $w_1 \overset{F}{\Rightarrow} w_2$ if w_2 is obtained from w_1 by a flag production. A *flag scan* is a derivation $w_1 \overset{*F}{\Rightarrow} w_2$ in G where $w_1 = Af_1 \cdots f_n$ with $A \in I$, $f_i \in F$, $w_2 \in T^*$. A *P-derivation* of length n is a derivation $w_1 \overset{p}{\Rightarrow} w_2 \overset{p}{\Rightarrow} \cdots \overset{p}{\Rightarrow} w_n$ in G, where $w_i \overset{p}{\Rightarrow} w_{i+1}$, $1 \leqslant i \leqslant n - 1$, either by a P production or by a flag scan.

Let $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0)$ be a lsa. In the following lemma the symbol $[r, Z, q]$ will be used to denote the set of words $\alpha \in \Sigma^*$ such that $(r, \alpha, XZ\uparrow) \overset{*}{\vdash} (q, \epsilon, X\uparrow)$ in A for some X in Γ^*.

LEMMA 4.1. *Let* $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0)$ *be a* lsa. *Then a stack grammar* $G = (N, I, T, F, P, \sigma)$ *can be effectively constructed such that* $L(G) = N(A)$.

Proof. The vocabulary of G is defined by

$$N = \{\sigma\} \cup \{[r, Z, q] \mid r, q \in K \text{ and } Z \in \Gamma\} \cup I,$$

$$I = \{(r, q) \mid r, q \in K\},$$

$$F = \Gamma \cup \bar{\Gamma} = \{\bar{Z} \mid Z \in \Gamma\}.$$

P contains the productions

(1) $\sigma \to [q_0, Z_0, f]$ for each f in K.

(2) $[r, Z, s_1] \to a[q, Z_k, s_k] Z_{k-1} \cdots Z_1[s_k, Z_{k-1}, s_{k-1}] Z_{k-2} \cdots Z_1 \cdots [s_2, Z_1, s_1]$

for each sequence $s_2, \ldots, s_k$ of states in K if $\delta(r, a, Z)$ contains $(q, 0, Z_1 \cdots Z_k)$.

(3) $[r, Z, q] \to a$ if $\delta(r, a, Z)$ contains $(q, 0, \epsilon)$.

(4) $[r, Z, q] \to (r, p) \bar{Z}[p, Z, q]$ for all $r, p, q \in K$ and Z in Γ.

The flag Z contains the productions

(5) $(r, p) \to a((s, p), e)$ for all p in K if $\delta(r, a, Z)$ contains (s, e, Z).

The flag Z contains the productions

(6) $(q, q) \to ((q, q), 1)$ for all q in K;

(7) $(r, p) \to a((s, p), e)$ for all p in K if $\delta(r, a, Z)$ contains (s, e, Z) and e is in $\{-1, 0\}$.

Whenever $(r, w, X_1 \cdots X_m Z\uparrow)$ is an id of A, then $[r, Z, q]$ or (r, p) is the leftmost nonterminal of a corresponding leftmost G-derivation for some $p, q \in K$, and its indexing flag strings are $X_m \cdots X_1$ and $\bar{Z} X_m \cdots X_1$, respectively.

(A) $(r, w, X_1 \cdots X_m Z\uparrow) \overset{*}{\vdash} (p, \epsilon, X_1 \cdots X_m Z\uparrow)$ is a stack scan if and only if there is a production $[r, Z, q] \to (r, p) \bar{Z}[p, Z, q]$ in P and a flag scan $(r, p)\bar{Z}X_m \cdots X_1 \overset{*F}{\Rightarrow} w$ in G such that $[r, Z, q]X_m \cdots X_1 \overset{p}{\Rightarrow} w[p, Z, q]X_m \cdots X_1$ for all $q \in K$.

(B) $[r, Z, q] X_t \cdots X_1 \overset{*p}{\Rightarrow} \alpha$, $\alpha \in (NF^* \cup T)^*$, $X_1 \cdots X_t \in F^*$, by a leftmost P-derivation if and only if either

(a) $\alpha = wy$ for $w \in T^*$, $y \in ((N - I)F^*)^+$ and there are states $p_1, \ldots, p_{m+1}$ in K with $p_1 = q$ and flags $Z_1, \ldots, Z_m$ in Γ such that either

(i) $y = [p_{m+1}, Z_m, p_m] Z_{m-1} \cdots Z_1 X_t \cdots X_1 [p_m, Z_{m-1}, p_{m-1}] Z_{m-2} \cdots X_1 \cdots [p_2, Z_1, p_1] X_t \cdots X_1$, and in A,

$$(r, w, X_1 \cdots X_t Z \uparrow) \overset{*}{\vdash} (p_{m+1}, \epsilon, X_1 \cdots X_t Z_1 \cdots Z_m \uparrow);$$

(ii) $y = (p_{m+1}, s) \bar{Z}_m Z_{m-1} \cdots Z_1 X_t \cdots X_1 [s, Z_m, p_m] Z_{m-1} \cdots Z_1 X_t \cdots X_1 \cdots [p_2, Z_1, p_1] X_t \cdots X_1$, $wy \overset{*F}{\Rightarrow} w'y'$ where $y' = [s, Z_m, p_m] Z_{m-1} \cdots Z_1 X_t \cdots X_1 \cdots [p_2, Z_1, p_1] X_t \cdots X_1$, $w' = w\hat{w}$, and in A $(r, w', X_1 \cdots X_t Z \uparrow) \overset{*}{\vdash} (p_{m+1}, \hat{w}, X_1 \cdots X_t Z_1 \cdots Z_m \uparrow) \overset{s}{\vdash} (s, \epsilon, X_1 \cdots X_t Z_1 \cdots Z_m \uparrow)$.

(b) $\alpha \in T^*$ and in A, $(r, w, X_1 \cdots X_t Z_1 \uparrow) \overset{*}{\vdash} (q, \epsilon, X_1 \cdots X_t \uparrow)$.

Thus a (i) corresponds to the case where the last move is a production of form (2), a (ii) to that where the last move is a production form (4) followed by a flag scan, and (b) to the case where the last move is a production with form (3). The proof follows by some straightforward but tedious inductions and can be found in (Ehrich, 1969).

Next the converse to Lemma 4.1 is given. It is necessary to show how the information in the flags is organized on the stack of a corresponding lsa, which involves interleaving flags and the pushdown symbols of the pda which accepts the context free language associated with the P productions of an arbitrary stack grammar.

LEMMA 4.2. *Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar. Then a* lsa *$A = (K, T, \Gamma, \delta, q_0, Z_0)$ such that $N(A) = L(G)$ can be effectively constructed.*

Proof. If $A \to xB$ is a flag production with $B = \epsilon$ or $B = (C, e)$, assume $|x| \leqslant 1$. Otherwise, if $x = x_1 \cdots x_t$ let $C_1 \cdots C_{t-1}$ be new intermediates and replace $A \to xB$ by productions $A \to x_1(C_1, 0)$, $C_1 \to x_2(C_2, 0), \ldots, C_{t-2} \to x_{t-1}(C_{t-1}, 0)$, $C_{t-1} \to x_tB$. Now let

$$K = \{q_0, q\} \cup I \times \{U, D, T\} \times \{F/, FRN, R\},$$
$$\Gamma = F \cup N \cup T \cup \{Z_0, *, /\}.$$

The components of the states in K have the following significance:

D (U): Previous move on stack was down (up);
T: Move to the top of the stack;
$F/$ (FRN): Find a $/$ (rightmost nonterminal);
R: Normal read mode.

δ can now be defined by the following moves:

1(a) $(q, 0, Z_0, \sigma) = \delta(q_0, \epsilon, Z_0)$,
(b) $(q, 0, \epsilon) = \delta(q, \epsilon, Z_0)$;

2(a) $(q, 0, /\Phi^R) \in \delta(q, \epsilon, A)$, for all $A \to \Phi \in P$,
(b) $(q, 0, \epsilon) = \delta(q, a, a)$, for all $a \in T$,
(c) $(q, 0, \epsilon) = \delta(q, \epsilon, f)$, for all $f \in F \cup \{/\}$;

3(a) $((B, D, R), 0, *) = \delta(q, \epsilon, B)$, for all $B \in I$,
(b) $((B, D, R), -1, *) = \delta((B, D, R), \epsilon, *)$, for all $B \in I$;

4(a) $((C, D, R), d, g) \in \delta((B, e, R), a, g)$, for all $B \to a(C, d)$ in g, $d \in \{0, -1\}$, $e \in \{U, D\}$,
(b) $((C, U, R), 1, g) \in \delta((B, e, R), a, g)$, for all $B \to a(C, 1)$ in g, $e \in \{U, D\}$,
(c) $((C, T, R), 1, g) \in \delta((B, e, R), a, g)$, for all $B \to a$ in g, $e \in \{U, D\}$;

5(a) $((B, D, R), -1, /) = \delta((B, D, a), \epsilon, /)$, $a \in \{R, F/\}$, for all $B \in I$,
(b) $((B, D, F/), -1, A) = \delta((B, D, R), \epsilon, A)$, for all $A \in N$, $B \in I$,
(c) $((B, D, F/), -1, A) = \delta((B, D, F/), \epsilon, A)$, for all $A \in F \cup N \cup T$, $B \in I$;

6(a) $((B, U, F/), 1, /) = \delta((B, U, R), \epsilon, /)$, for all $B \in I$,
(b) $((B, U, F/), 1, A) = \delta((B, U, F/), \epsilon, A)$, for all $A \in F \cup N \cup T$, $B \in I$,
(c) $((B, U, FRN), -1, /) = \delta((B, U, F/), \epsilon, /)$, for all $B \in I$,
(d) $((B, U, FRN), -1, f) = \delta((B, U, FRN), \epsilon, f)$, for all $B \in I, f \in F$,
(e) $((B, U, R), 1, a) = \delta((B, U, FRN), \epsilon, a)$, for all $B \in I$, $a \in N \cup \{/\}$,
(f) $(q, 0, \epsilon) = \delta((B, U, F/), \epsilon, *)$, for all $B \in I$;

7(a) $(q, 0, \epsilon) = \delta((B, e, R), \epsilon, *)$, for all $B \in I$, $e \in \{U, T\}$,
(b) $((B, T, R), 1, a) = \delta((B, , R), \epsilon, a)$, for all $a \in \Gamma - \{*\}$, $B \in I$.

Informally, rules (1) are the first and last moves of A, rules (2) govern P productions, rules (3) initialize a stack scan (flag scan), and rules (4) allow A to simulate a flag scan. (5) and (6) are rules A uses when locating flags and moving down or up, respectively. Rules (7) require that A move to the top of the stack after a production of the form $A \to a$.

Now let $\sigma = w_0 \overset{p}{\Rightarrow} w_1 \overset{p}{\Rightarrow} \cdots \overset{p}{\Rightarrow} w_m$ be a leftmost P-derivation in G and let

$(q_0, x_1 \cdots x_m, Z_0\uparrow) \overset{*}{\vdash} (q, x_2 \cdots x_m, Z_1\uparrow) \vdash \cdots \vdash (q, \epsilon, Z_m\uparrow)$ be a corresponding sequence of id's of A. Let $f_{ij} \in F^*$ and $A_{ij} \in N \cup T$, let

$$g_{ij} = f_{ij}^R f_{i-1,k_{i-1}}^R \cdots f_{1k_1}^R \quad \text{if} \quad A_{ij} \in N,$$
$$g_{ij} = \epsilon \quad \text{if} \quad A_{ij} \in T,$$

and let $\varphi_i = f_{i1}A_{i1} \cdots f_{ik_i-1}A_{ik_i-1}f_{ik_i}$, where

$$f_{ij} \in F^* \quad \text{if} \quad A_{ij} \in N,$$
$$f_{ij} = \epsilon \quad \text{if} \quad A_{ij} \in T.$$

Then $w_m = x_1 \cdots x_m A_{sk_s} g_{sk_s} \cdots A_{s1} g_{s1} \cdots A_{11} g_{11}$ if and only if

$$Z_m = Z_0/\varphi_1/\varphi_2/ \cdots /\varphi_s A_{sk_s}, A_{sk_s} \in N.$$

If w_m has the flags shown, then A can simulate a flag scan by rules (4), (5), and (6) if the information on the stack is given by Z_m. It is therefore necessary to show by an induction on the length of a P-derivation in G that a P-derivation produces w_m if and only if in A, Z_m is the corresponding information on the stack. Thus the organization of the stack is invariant, and the topmost nonterminal of A is the leftmost nonterminal in G. The details of the induction can be found in (Ehrich, 1969).

Consequently, we have proved the following theorem.

THEOREM 4.2. *The class of stack grammars generates exactly the family of languages accepted by the* 1*-way nondeterministic stack automata.*

The definition given in Section 4 for stack grammars is more general than necessary. The following theorem presents a minimally complex form for a class of grammars which still generates all the stack languages.

THEOREM 4.3 (Normal Form). *Let* $G = (N, I, T, F, P, \sigma)$ *be a stack grammar. A grammar* $G' = (N', I', T, F', P', \sigma)$ *can be effectively constructed such that* $L(G) = L(G')$ *and such that*

(1) *P' productions have the form* $A \to a$, $A \to BC$, *or* $A \to Bf$ *for* $A, C \in N' - I'$, $B \in N'$, $f \in F'$, *and* $a \in T \cup \{\epsilon\}$.

(2) *Flag productions have the form* $A \to a(B, e)$ *for* $A, B \in I'$, $a \in T \cup \{\epsilon\}$.

Proof. Let $N' \supseteq N$, $I' \supseteq I$, and $F' = \{f' \mid f \in F\}$, and note that from (1) productions of the form $A \to B$ can be obtained for any $B \in N'$. Now let

$A \to \beta_1\theta_1 \cdots \beta_n\theta_n$ be any P production with $\beta_i \in N \cup T$, $\theta_i \in F^*$, and $\theta_i = \epsilon$ if $\beta_i \in T$. If $A \to \beta_1\theta_1 \cdots \beta_n\theta_n$ has the form (1) place it in P'. Otherwise add to $N' - I'$ the distinct, new nonterminals $A_1, \ldots, A_n$, $B_1, \ldots, B_{n-2}$ and $\{F_{ij} \mid f_{ij} \neq \epsilon\}$, where $\theta_i = f_{i1} \cdots f_{ik_i}$. Then add to P' the following productions:

(a) $A \to A_1B_1$, $B_{n-2} \to A_{n-1}A_n$, $\{B_i \to A_{i+1}B_{i+1} \mid 1 \leqslant i \leqslant n-3\}$.

(b) $\{A_i \to \beta_i \mid \theta_i = \epsilon\}$.

(c) $\{A_i \to F_{i1}f'_{i1} \mid f_{i1} \neq \epsilon\}$.

(d) $\{F_{ij} \to F_{ij+1}f'_{ij+1} \mid f_{ij+1} \neq \epsilon\}$.

(e) $\{F_{ij} \to \beta_i \mid j = k_i\}$.

Clearly G' contains $A \overset{*}{\Rightarrow} \beta_1\theta_1 \cdots \beta_n\theta_n$, and we need only check that $Af' \overset{*}{\Rightarrow} \beta_1\theta_1 f' \cdots \beta_n\theta_n f'$ for any $f' \in F'^*$ if and only if $Af \Rightarrow \beta_1\theta_1 f \cdots \beta_n\theta_n f$ in G. But this is easily seen since if A is indexed by f, so is each A_i and B_j, so that indeed $Af' \overset{*}{\Rightarrow} \beta_1\theta_1 f' \cdots \beta_n\theta_n f'$.

Now suppose $f \in F$ contains $A \to a_1 \cdots a_k(B, e)$ for some $A, B \in I$, $a_i \in T$. If $k = 1$ place $A \to a_1(B, e)$ in $f' \in F'$. Otherwise add to I' distinct new intermediates $B_1, \ldots, B_{k-1}$ and let f' have the productions

(a) $A \to a_1(B_1, 0)$.

(b) $B_{k-1} \to a_k(B, e)$.

(c) $\{B_{i-1} \to a_i(B_i, 0) \mid 2 \leqslant i \leqslant k-1\}$.

If $A \to a_1 \cdots a_k$ is in f then add to I' the new distinct intermediates $B_1, \ldots, B_{k-1}, Z$, add to each $g \in F$ the productions $Z \to (Z, 1)$, and let f' have the additional productions $B_{k-1} \to a_k(Z, 1)$ and (a) and (c) above. It is not difficult to see that $L(G) = L(G')$.

Next we formalize the connection between the 2st and stack automata.

THEOREM 4.4. *Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar. Then there is a regular set R and a* 2st $S_2 = (K, \Sigma, \Delta, H, s_0)$ *such that $Af_1 \cdots f_n \overset{*F}{\Rightarrow} w$ is a flag scan in G if and only if $w \in S_2(Af_1 \cdots f_n\not{S})$ for all $Af_1 \cdots f_n\not{S}$ in R.*

Proof. We can assume that G has normal form since Theorem 4.3 does not essentially change flag scans. Then it is easily seen that each flag string of the form $Af_1 \cdots f_n$ with $A \in I$ is derived by a left linear grammar, and the set S of strings $Af_1 \cdots f_n$ derivable in G is regular. Then $R = S\not{S}$ is regular.

Now S_2 is defined as follows:

(1) $K = I \cup \{s_0, s\}$.
(2) $\Sigma = F \cup I \cup \{\$\}$.
(3) $\Delta = T$.

H contains the moves

(1) $\{(B, f_i, a, C, -e) \mid B \to a(C, e) \in f\}$.
(2) $\{(s_0, A, \epsilon, A, 1), (A, B, \epsilon, s, 1) \mid A, B \in I\}$.
(3) $\{(s, f, \epsilon, s, 1) \mid f \in F \cup \{\$\}\}$.

Then it is easily seen that $Af_1 \cdots f_n \overset{*F}{\Rightarrow} w$ if and only if $w \in S_2(Af_1 \cdots f_n\$)$. Therefore a stack grammar behaves as a 2st during a flag scan, and by Lemma 4.1 a lsa can be viewed as though it makes a 2st mapping of the stack contents whenever it makes a stack scan, nondeterministically producing the symbols which the lsa is reading in.

It has already been pointed out that except for the flag scans, a derivation in a stack grammar looks much like a derivation in a context free language. Since in a strict sense type 1, 2, and 3 grammars do not contain ϵ rules, we show next how ϵ rules and ϵ flag scans can be removed from a stack grammar to produce an ϵ-free language just as context free languages can be modified to be ϵ-free. Furthermore, a stack grammar, such as one derived by the construction in Lemma 4.1, may contain blocking sequences, that is, sequences which cannot result in a terminal sequence. We show here that these can also be removed although G' may still make a "wrong" move during a flag scan which will block the scan. If each flag scan in G is deterministic, then no sequence derivable in G' is a blocking sequence.

THEOREM 4.5. *Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar. Then*

(1) *There is a grammar $G^1 = (N^1, I^1, T, F^1, P^1, \sigma^1)$ such that $L(G) = L(G^1)$ and such that $Af \overset{*F}{\Rightarrow} y$ is a flag scan in G^1 implies $y \neq \epsilon$.*

(2) *There is a grammar $G^2 = (N^2, I^2, T, F^2, P^2, \sigma^2)$ satisfying* (1) *such that $A \to a$ is a production in P^2 only if $a \neq \epsilon$ and such that $L(G^2) = L(G) - \{\epsilon\}$.*

(3) *There is a grammar $G^3 = (N^3, I^3, T, F^3, P^3, \sigma^3)$ such that $L(G) = L(G^3)$ and such that $A \to \Phi$ is a production in P^3 if and only if whenever $f \in F^{3*}$ is a flag string indexing A in a G^3-derivation then $Af \Rightarrow \Phi f \overset{*p}{\Rightarrow} x$ for some $x \in T^*$.*

(4) *There is a grammar* $G^4 = (N^4, I^4, T, F^4, P^4, \sigma^4)$ *such that* $L(G) = L(G^4)$ *and such that* $A \in N^4 - I^4$ *if and only if* $\sigma^4 \overset{*p}{\Rightarrow} \varphi A \psi$ *for some* $\varphi, \psi \in (N^4 \cup F^4 \cup T)^*$.

(5) *There is a grammar* $G^5 = (N^5, I^5, T, F^5, P^5, \sigma^5)$ *satisfying any or all of conditions* (1)–(4).

Proof. We make use in this proof of a simpler equivalence relation than has been used before. Let $A = (K, \Sigma, \delta)$ be an asm and let $x = x_1 \cdots x_n$ and $y = y_1 \cdots y_t$. Then

$$\begin{aligned} &x \equiv y \text{ if and only if } x_1 \cdots x_{n-1} p x_n \overset{*}{\vdash} xq \text{ whenever} \\ &y_1 \cdots y_{t-1} p y_t \overset{*}{\vdash} yq \text{ and conversely for all } p, q \in K \end{aligned} \tag{4.1}$$

is a right invariant equivalence relation of finite index $\delta \leqslant 2^{|K|^2}$.

Part 1. Let $A = (J, F, \delta)$ be the asm defined by $J = I \cup \{U\}$ and let δ be defined by

(a) $(B, e) \in \delta(A, F)$ iff $A \to (B, e)$ is in f.

(b) $(U, 1) \in \delta(A, f)$ iff $A \to \epsilon$ is in f.

(c) $(U, 1) \in \delta(U, f)$ for all $f \in F$.

Now $Af_m \cdots f_1 \overset{*F}{\Rightarrow} \epsilon$ is a flag scan in G if and only if in $A, f_1 \cdots f_{m-1} A f_m \overset{*}{\vdash} f_1 \cdots f_m B$ for some $B \in J$. Let $E = \{0, 1, \ldots, \delta - 1\}$ be the equivalence classes of (4.1) where $0 = \{\epsilon\}$ and define

$$\begin{aligned} N^1 &= ((N - I) \times E) \cup I^1, \\ I^1 &= I \cup \{\hat{A}, \tilde{A} \mid A \in I\}. \\ F^1 &= \{f' \mid f \in F\} \cup \{*\}. \\ \sigma^1 &= (\sigma, 0). \end{aligned}$$

The flags $f' \in F^1$ are defined in the following way. $*$ contains $\{A \to (\hat{A}, 1)$ and $\tilde{A} \to (\tilde{A}, 1) \mid A \in I\}$. f' contains

$$\begin{aligned} &\{\hat{A} \to (\hat{B}, e) \mid A \to (B, e) \in f\} \cup \\ &\{\hat{A} \to a(\tilde{B}, e) \mid A \to a(B, e) \in f,\ a \in T^+\} \cup \\ &\{\tilde{A} \to a \mid A \to a \in f,\ a \in T^+\} \cup \\ &\{\tilde{A} \to a \mid A \to a \in f,\ a \in T^*\} \cup \\ &\{A \to a(B, e) \mid A \to a(B, e) \in f,\ a \in T^*\}. \end{aligned}$$

During a flag scan an intermediate has a hat until non-ϵ output is written, at which time $\hat{}$ becomes $\tilde{}$. Now if $Af_1 \cdots f_m \overset{*F}{\Rightarrow} x$ is a flag scan in G, $A * f_1' \cdots f_m' \overset{*F}{\Rightarrow} x$ is a flag scan in G^1 except when $x = \epsilon$, in which case the scan blocks in G^1. P^1 is constructed as follows.

(1) Let $A \to \Phi$ be a production in P and let $A_1 f_1, \ldots, A_t f_t$ be the occurrences of intermediates and their indexing flags $f_1, \ldots, f_t$ in F^*. First replace each $A_i f_i$ by $A_i * f_i'$. Let $i_1, \ldots, i_k$ be the indices such that $A_{i_r} f_{i_r} x^R \overset{*F}{\Rightarrow} \epsilon$ is a flag scan in G, where $x \in \{j\}$ and $\{j\}$ is the equivalence class j. $i_1, \ldots, i_k$ are effectively calculable since for each i, $1 \leqslant i \leqslant t$, j_i such that $x^R f_i^R \in \{j_i\}$ is calculable from j and f_i. Then $(A, j) \to \Phi'$ are productions in P^1, where Φ' is obtained from Φ by replacing $0, 1, \ldots, k$ occurrences of $A_1 * f_1', \ldots, A_k * f_k'$ by ϵ.

(2) For each production $(A, j) \to \Phi'$ obtained in (1) if $Bg \in (N - I)F^*$ is an indexed nonterminal in Φ and $x \in \{j\}$, then replace Bg in Φ' by $(B, i)g'$ where $xg^R \in \{i\}$. $L(G^1) = L(G)$ and every G^1-derivation proceeds without the necessity of having to make an ϵ flag scan.

Part 2. We can assume that G has property (1). Let $S^0 = \{A \mid A \to \epsilon \text{ in } P\}$ and let $S^{i+1} = S^i \cup \{A \mid A \to \Phi \text{ in } P,\ \Phi \in (S^i F^*)^*\}$. Since N is finite, $S^{|N-I|} = S^{|N-I|+1} = \cdots$ and S^i represents the set of nonterminal symbols which generate ϵ by P productions in a tree of height at most $i + 1$. By (1) ϵ is never produced in a flag scan. Thus G^2 is defined by taking

$$N^2 = N$$

$$I^2 = I$$

$$F^2 = F$$

$$\sigma^2 = \sigma$$

and letting P be the set of all rules $A \to \Phi$ with $\Phi \neq \epsilon$ constructed from P by deleting from the right hand sides of the P rules $0, 1, 2, \ldots$ occurrences of nonterminals in $S^{|N-I|}$ together with their indexing flags. $L(G^2) = L(G) - \{\epsilon\}$.

Part 3. Let $A = (J, F, \delta)$ be the asm defined by $J = I \cup \{U\}$ and let δ be defined by

(a) $(B, e) \in \delta(a, f)$ iff $A \to a(B, e)$ is in f, $a \in T^*$.

(b) $(U, 1) \in \delta(A, f)$ iff $A \to a$ is in f, $a \in T^*$.

(c) $(U, 1) \in \delta(U, f)$ for all f in F.

Now $Af_m \cdots f_1 \overset{*F}{\Rightarrow} w$ is a successful flag scan in G if and only if in A, $f_1 \cdots f_{m-1} A f_m \overset{*}{\vdash} f_1 \cdots f_m B$ for some $B \in J$. Let $E = \{0, 1, \ldots, \delta - 1\}$ be the equivalence classes of (4.1), where $0 = \{\epsilon\}$, and define

$$N^3 = ((N - I) \times E) \cup I^3$$

$$I^3 = I$$

$$F^3 = F$$

$$\sigma^3 = (\sigma, 0),$$

and let P^3 be constructed as follows.

(1) Let $A \to \Phi$ be a production in P and $j \in E$, and suppose $x \in \{j\}$. Place in P^3 the production $(A, j) \to \Phi'$ where Φ is constructed by replacing each occurrence of $Bg \in (N - I)F^*$ in Φ by (B, i) where $xg^R \in \{i\}$.

(2) Let $(A, j) \to \Phi'$ be a rule in P^3 obtained in step (1), and let $A_1 f_1 \cdots A_t f_t$ be the occurrences of intermediates and their indexing flag strings in Φ'. Suppose $x \in \{j\}$. If for any r, $1 \leqslant r \leqslant t$, $A_r f_r x^R \overset{*F}{\Rightarrow} \phi$, then remove the rule $(A, j) \to \Phi'$ from P^3.

(3) Let $S^0 = \{A \mid A \to \Phi' \text{ in } P^3,\ \Phi' \in (IF \cup T)^*\}$. Let $S^{i+1} = S^i \cup \{A \mid A \to \Phi' \text{ in } P^3,\ \Phi' \in (S^i F^* \cup IF^* \cup T)^*\}$. S^i is the set of nonterminals which can generate some sequence in $(IF^* \cup T)^*$ in a tree of height at most $i + 1$. Since N^3 is finite, $S^{|N^3 - I^3|} = S^{|N^3 - I^3|+1} = \cdots$, and all productions in P^3 having an element in $N^3 - I^3 - S^{|N^3 - I^3|}$ on the right are removed since such rules always block. Thus $L(G^3) = L(G)$ and $(A, j) \to \Phi$ is a production in P^3 if and only if when $f \in F^*$ is a flag string indexing (A, j) in a G^3-derivation then $f^R \in \{j\}$ and $(A, j)f \Rightarrow \Phi f \overset{*T}{\Rightarrow} x$ for some $x \in T^*$.

Part 4. Let $S^0 = \{A \mid \sigma \to \varphi A \psi \text{ in } P,\ A \in N - I,\ \varphi, \psi \in (N \cup F \cup T)^*\}$. Let $S^{i+1} = S^i \cup \{A \mid S \to \varphi A \psi \text{ in } P,\ S \in S^i,\ A \in N - I,\ \varphi, \psi \in (N \cup F \cup T)^*\}$. S^{i+1} represents the nonterminals derivable in $i + 1$ steps in G. Since N is finite, $S^{|N-I|} = S^{|N-I|+1} = \cdots$ so that no nonterminal in $N - I$ is derivable in G if it is not in $S^{|N-I|}$. Let

$$N^4 = S^{|N-I|} \cup I^4$$

$$I^4 = I$$

$$F^4 = F$$

$$\sigma^4 = \sigma,$$

and let P^4 have all the productions in P not having an element of $N - S^{|N-I|}$ on its left side. Clearly $L(G^4) = L(G)$.

Part 5. None of (2)–(4) generate ϵ flag scans, neither (3) nor (4) produces any ϵ rules, and (4) does not produce any blocking rules so that a grammar may have any or all of the properties (1)–(4) by constructions in that order.

The complete proof of Theorem 4.5 may be found in (Ehrich, 1969).

COROLLARY 4.1. *Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar such that for any flag $f \in F$ an intermediate $A \in I$ appears at most once on the left side of a production in f. Then there is a grammar $G' = (N', I', T, F', P', \sigma)$ such that $L(G') = L(G) - \{\epsilon\}$ and every step in a G'-derivation leads to a terminal sequence in T^*.*

Proof. Take $G' = G^2$ in Theorem 4.5. Since in G every flag scan is deterministic a flag scan is either ϕ or in T^*, but not both. The construction of G^2 preserves this determinism while removing any blocking scans so that no step in a G'-derivation can block.

It is known that a stack automaton which never reads more than some fixed number, say k, of input symbols during a stack scan defines a context free language (Aho, 1968). Next we give an analogous result for stack grammars, which sharpens the assertions (1) and (3) of Theorem 4.5.

THEOREM 4.6. *Let $G = (N, I, T, F, P, \sigma)$ be a stack grammar with the property that $Af_1 \cdots f_n \overset{*F}{\Rightarrow} w$ is a flag scan in G implies $|w| \leqslant k$ for some fixed $k \geqslant 0$. Then $L(G)$ is context free.*

Proof. Informally, a context free grammar G' is defined such that $L(G') = L(G)$. The nonterminals in G' are constructed to keep track of the terminal sequences G can write during a flag scan.

As in Theorem 4.4 we can assume that G is in normal form, and for convenience define $S = \bigcup_{i=0}^{k} T^i$. Let $K = (I \times S) \cup (I \times \{*\})$ and define the asm $A = (K, F, \delta)$ in the following way.

(1) $((Q, xa), e) \in \delta((P, x), f)$ if f contains $P \to a(Q, e)$, $xa \in S$.

(2) $((Q, *), e) \in \delta((P, x), f)$ if f contains $P \to a(Q, e)$ and xa is not in S.

(3) $((Q, *), e) \in \delta((P, *), f)$ if f contains $P \to a(Q, e)$.

It can easily be seen that $Af_1 \cdots f_n \overset{*F}{\Rightarrow} w$ if and only if $f_n \cdots f_2(A, \epsilon)f_1 \overset{*}{\vdash} f_n \cdots f_1(B, w)$ for some $B \in I$ when $|w| \leqslant k$. Let $E = \{0, 1, \ldots, \delta - 1\}$ be the equivalence classes of the equivalence relation (4.1) with $0 = \{\epsilon\}$. Define

the context free grammar $G' = (N \times E, T \cup \{*\}, P', (\sigma, 0))$. P' contains the productions

1) $\{(A, j) \to a \mid A \to a \in P \text{ and } j \in E\}$.

2) $\{(A, j) \to (B, j)(C, j) \mid A \to BC \text{ in } P, j \in E\}$.

3) $\{(A, j) \to (B, i) \mid A \to Bf \text{ in } P, x \in \{j\} \text{ implies } xf \in \{i\}\}$.

4) $\{(A, j) \to w \mid A \in I \text{ and } x \in \{j\} \text{ implies } Ax^R \overset{*F}{\Rightarrow} w, w \in S\}$.

5) $\{(A, j) \to * \mid A \in I \text{ and } x \in \{j\} \text{ implies } Ax^R \overset{*F}{\Rightarrow} w, w \text{ not in } S\}$.

Productions (3) and (5), in particular, can be effectively determined and if the hypothesis is satisfied productions (5) are never used so that $L(G') = L(G)$. G' may have blocking rules and unreachable nonterminals which may readily be removed by Theorem 4.5 (3) and (4). The productions (5) were included so that the following corollary follows easily.

COROLLARY 4.2. *Given a stack grammar $G = (N, I, T, F, P, \sigma)$ it is decidable whether there is ever a flag scan of the form $Af \overset{*F}{\Rightarrow} w$ for some $w \in T^*$ with $| w | > k$ during the derivation of a sentence in $L(G)$.*

Proof. Let G' be the context free grammar in Theorem 4.6 from which all blocking rules have been removed. Then $\sigma \overset{*}{\Rightarrow} x$, $x \in T^*$, by a flag scan $Af \overset{*F}{\Rightarrow} w$, $w \in T^*$ and $| w | > k$, if and only if $(\sigma, 0) \overset{*}{\Rightarrow} \varphi * \psi$ in G' with $\varphi, \psi \in T^*$, which is decidable (Landweber, 1964).

Next we briefly compare stack automata and stack languages with related automata.

THEOREM 4.7. *The class of 1-way stack automata languages properly contain the context free languages and* 2st *transductions of regular sets.*

Proof. Let $G = (N, I, T, F, P, \sigma)$ be a normal form stack grammar. If $F = I = \phi$ then G is the Chomsky normal form (Chomsky, 1959) for an arbitrary context free grammar.

Now suppose that R is a regular set and $S_2 = (K, F, T, H, s_0)$ is a 2st such that $L = S_2(R)$. Let $G = (N, F, P, \sigma)$ be a left linear grammar such that $R = L(G)$. We construct a stack grammar

$$G_s = (N \cup K \cup \{\sigma_s, \hat{s}_0\}, K \cup \{\hat{s}_0\}, T, F \cup \{\$, \not\phi\}, P_s, \sigma_s)$$

such that $L = L(G_s)$ as follows. Let P_s have the productions

1(a) $\{\sigma_s \to Af_s \cdots f_1\not{\epsilon} \mid \sigma \to Af_s \cdots f_1 \in P\}$.
(b) $\{\sigma_s \to \hat{s}_0 \not{S} f_s \cdots f_1\not{\epsilon} \mid \sigma \to f_s \cdots f_1 \in P\}$.

2(a) $\{A \to Bf_s \cdots f_1 \mid A \to Bf_s \cdots f_1 \in P\}$.
(b) $\{A \to \hat{s}_0 \not{S} f_s \cdots f_1 \mid A \to f_s \cdots f_1 \in P\}$.

Let the flags in $F \cup \{\not{S}, \not{\epsilon}\}$ be defined in the following way.

$$\not{S} = [\hat{s}_0 \to (s_0, 1)]$$
$$\not{\epsilon} = [\{s \to \epsilon \mid s \in K\}].$$
$$f = [\{s \to x(q, -e) \mid (s, f, x, q, e) \in H\}].$$

The rules in P_s form a left linear grammar for generating $\hat{s}_0 \not{S} R \not{\epsilon}$ and the flags simulate S_2 so that indeed $L(G_s) = S_2(R)$.

Now by Lemma 3.5, $\hat{L} = \{a^{n^2} \mid n \geqslant 1\}$ is not a 2st transduction of a regular set and $\hat{L}$ is not context free (Ginsburg, 1966). However, $\hat{L}$ is easily shown to be generated by the stack grammar $\hat{G} = (\{\hat{\sigma}, S\}, \{I\}, \{a\}, \{f, g\}, \hat{P}, \hat{\sigma})$ where $\hat{P}$ contains the productions

$$\hat{\sigma} \to SgIg$$
$$S \to SffIff$$
$$S \to \epsilon$$

and where $f = [I \xrightarrow{R} aI]$ and $g = [I \to a]$. Thus the theorem is proved.

THEOREM 4.8. *Let A be a* lsa *which never erases stack symbols until it makes a stack scan, after which it erases its stack. $N(A)$ is the* 2st *mapping of a regular set.*

Proof. Let $A = (K, \Sigma, \Gamma, \delta, s_0, Z_0)$ be the lsa and define the left linear grammar $G = (N, T, P, \sigma)$

(1) $N = (K \times \Gamma) \cup \{\sigma\}$.

(2) $T = \Gamma \cup \Sigma \cup \{\hat{Z}_p \mid Z \in \Gamma, p \in K\} \cup \{*\}$.

(3) $P = \{(s, Z) \to (q, Z_k) Z_{k-1} \cdots Z_1 a \mid (q, 0, Z_1 \cdots Z_k) \in \delta(s, a, Z), k \geqslant 1\}$
$\cup \{(p, Z) \to \hat{Z}_p \mid (q, -1, Z) \in \delta(p, a, Z), q \in K\} \cup \{\sigma \to (s_0, Z_0)*\}$.

Each string in $L(G)$ contains the contents of the stack of A and the input which A read while writing on its stack. Now let $S_2 = (L, T, \Sigma, H, q_0)$ where

$L = K \cup K_L \cup K_R \cup \{q_0, q_1\}$, $K_L = \{s_L \mid s \in K\}$, $K_R = \{s_R \mid s \in K\}$, and H has the moves

1(a) $\{(q_0, Z, \epsilon, q_0, 1) \mid Z \in T - \{*\}\}$.
(b) $\{(q_0, *, \epsilon, q_1, -1)\}$.

2(a) $\{(q_1, a, a, q_1, -1) \mid a \in \Sigma\}$.
(b) $\{(q_1, a, \epsilon, q_1, -1) \mid a \in \Gamma\}$.
(c) $\{(q_1, \hat{Z}_p, \epsilon, p_R, 0) \mid p \in K\}$.

3(a) $\{(p_L, a, \epsilon, p_L, -1) \mid a \in \Sigma\}$.
(b) $\{(s, a, \epsilon, s, 1) \mid a \in \Sigma, s \in K \cup K_R\}$.

4(a) $\{(p_R, a, b, q_L, -1), (p_L, a, b, q_L, -1) \mid (q, 1, Z) \in \delta(p, b, Z), a \in \{Z, \hat{Z}_p \mid p \in K\}\}$.
(b) $\{(p_R, a, b, q_R, e), (p_L, a, b, q_R, e) \mid (q, -e, Z) \in \delta(p, b, Z), e \in \{0, 1\}, a \in \{Z, \hat{Z}_p \mid p \in K\}\}$.

5(a) $\{(s, \hat{Z}_r, b, q, 1) \mid s \in \{p_L, p_R\}$ and $(q, 0, \epsilon) \in \delta(p, b, Z), r \in K\}$.
(b) $\{(p, *, \epsilon, p, 1) \mid p \in K\}$.
(c) $\{(p, Z, b, q, 1) \mid (q, 0, \epsilon) \in \delta(p, b, Z)\}$.

Now $S_2(L(G))$ is a 2st mapping of a regular set. Moves (1) position the read head over the rightmost tape symbol, moves (2) allow S_2 to write what A read as it wrote on its stack, and moves (3) require that S_2 skip symbols in Σ as S_2 simulates the stack scan of A by rules (4). Finally rules (5) allow S_2 to simulate A as it erases its stack. Thus $N(A) = S_2(L(G))$.

Remarks. The stack grammars proposed here can be used very neatly to specify some of the structures of Algol 60 which are usually defined in the semantics of the language. These include the declaration of variables before their use, type agreement in assignment statements, and the definition of variables for blocks interior but not exterior to a block containing a declaration. Since apparently not all of Algol 60 can be specified by a lsa the need still exists for better models.

We would like to know if there are any useful relationships between the 2st and the 2-way stack automata. Regarding the 2st we strongly believe that D_2, the Dyck language on 4 symbols, is not a 2st mapping of a regular set, hence that the context free languages are not 2st mappings of regular sets. Also it appears that equivalence may be solvable for deterministic 2st. Perhaps better characterization of 2st transductions may help solve these problems.

RECEIVED: August 20, 1969; REVISED: November 16, 1970

References

AHO, A. V. (1968), Indexed grammars—an extension of context-free grammars, *J. Assoc. Comp. Mach.* **15**, 647–671.

CHOMSKY, N. (1959), On certain formal properties of grammars, *Inform. Control* **2**, 137–167.

DAVIS, M. (1958), "Computability and Unsolvability," McGraw-Hill, New York.

EHRICH, R. W. (1969), "Two-Way Sequential Transducers and Stack Automata," Ph.D. Dissertation, Northwestern University, Evanston, Illinois, August, 1969. Available from University Microfilms, Ann Arbor, Michigan.

GINSBURG, S. (1966), "The Mathematical Theory of Context-Free Languages," McGraw-Hill, New York.

GINSBURG, S., GREIBACH, S., AND HARRISON, M. (1967a), Stack automata and compiling, *J. Assoc. Comp. Mach.* **14**, 172–201.

GINSBURG, S., GREIBACH, S., AND HARRISON, M. (1967b), One-way stack automata, *J. Assoc. Comp. Mach.* **14**, 389–418.

GRIFFITHS, T. V. (1968), The unsolvability of the equivalence problem for Λ-free nondeterministic generalized machines, *J. Assoc. Comp. Mach.* **15**, 409–413.

HOPCROFT, J. E. AND ULLMAN, J. D. (1967), Two Results on One-Way Stack Automata, *Proc. 8th Ann. Symp. on Switching and Automata Theory*, IEEE Publication, 37–44.

HOPCROFT, J. E. AND ULLMAN, J. D. (1969), "Formal Languages and Their Relation to Automata," Addison Wesley, Reading, Massachusetts.

KURODA, S. Y. (1964), Classes of languages and linear-bounded automata, *Inform. and Control* **7**, 207–223.

LANDWEBER, P. S. (1964), Decision problems of phrase-structure grammars, *IEEE Trans. on Elect. Comp.* **EC-13**, 354–362.

RABIN, M. AND SCOTT, D. (1959), Finite automata and their decision problems, *IBM J. Res. Development* **3**, 114–125.

SHEPHERDSON, J. C. (1959), The reduction of two-way automata to one-way automata, *IBM J. Res. Development* **3**, 198–200.

SCHKOLNICK, M. (1968), Two-Type Bracketed Grammars, *Proc. 9th Ann. Symp. on Switching and Automata Theory*, IEEE Publication, 315–326.