INFORMATION AND CONTROL 42, 27–37 (1979)

Linearity is Polynomially Decidable for Realtime Pushdown Store Automata*

S. A. GREIBACH

Department of System Science, University of California, Los Angeles, Los Angeles, California 90024

If M is a realtime deterministic pushdown store acceptor, the language $L(M)$ accepted by M by final state and empty store is linear context-free if and only if a certain grammar obtained from M is linear context-free. Hence, it is polynomially decidable for realtime deterministic pushdown store automata M whether $L(M)$ is linear context-free. If M is a realtime deterministic pushdown store acceptor and $L(M)$ is linear context-free, we can construct a realtime single turn deterministic pushdown store automaton $\bar{M}$ with $L(M) = L(\bar{M})$. Hence "$L(M) = L$" is decidable for M a realtime deterministic pushdown store acceptor and L the language accepted by final state by a single turn deterministic pushdown store acceptor.

Many decision problems for deterministic pushdown store automata have remained open since they were first studied (Ginsburg and Greibach, 1966). The most notable is the equivalence problem; this problem has been shown equivalent to the equivalence problem for monadic recursion schemes (Friedman, 1977a) and various subcases have been solved (McNaughton, 1967; Rosenkrantz and Stearns, 1970; Korenjak and Hopcroft; 1966; Valiant, 1973, 1974; Valiant and Paterson, 1975; Beeri, 1976; Greibach and Friedman, 1978*a*, b; Friedman and Greibach, 1978; Oyamaguchi *et al.*, 1978). There are also the "containment" problems of the following nature: for a class of languages $\mathscr{C}$ which does not include all deterministic context-free languages and a deterministic pushdown store automaton M, is the language accepted by M a member of $\mathscr{C}$? The strongest result on those lines was obtained by Stearns (1967) and the timing of the algorithm improved by Valiant (1975): it is decidable whether a deterministic pushdown store automaton accepts a regular set. Some containment problems are also equivalent to problems on schemes: the containment problem with respect to the class of simple languages is equivalent to the problem of determining whether or not a monadic recursion scheme has a strongly equivalent

* This paper was supported in part by the National Science Foundation under Grants DCR74-15091 and MCS 78-04725.

0019-9958/79/070027-11$02.00/0

free scheme (Friedman, 1977b). Other containment problems are related to equivalence problems: the equivalence problem for deterministic pushdown automata is equivalent to the containment problem relative to the class of $LR(0)$ languages (Geller and Harrison, 1977).

These problems have been examined for the special subclass of real-time context-free languages, to see whether they could be solved in this case and, where algorithms were known in the general case, whether faster algorithms would be possible than in the general case. It has been shown recently that the method used by Valiant (1973) to decide equivalence for nonsingular automata will work for realtime pushdown store automata accepting by final state and empty store (Oyamaguchi *et al.*, 1978). Most of the containment problems remain open.

Harrison and Havel (1972) and Valiant (1975) observed that it is polynomially decidable whether a realtime pushdown store automaton accepts (by final state and empty store) a regular set. In this note, we extend this result to the acceptance of linear context-free languages. We show that it is polynomially decidable whether a realtime pushdown store automaton accepts by final state and empty store a linear context-free language. The restriction "by final state and empty store" is necessary since the method is no longer valid for acceptance "by final state alone", although we conjecture that the problem is decidable in that case.

The algorithm for deciding for a realtime pushdown store automaton M whether $L(M)$, the language M accepts by final state and empty store, is a linear context-free language is very simple. One obtains from M the canonical reduced context-free grammar G_M corresponding to M, turns nonterminals which are not self-embedding into terminals and simply examines whether the resulting grammar $\bar{G}_M$ is linear context-free. Then $L(M)$ is linear context-free if and only if $\bar{G}_M$ is already a linear context-free grammar.

The only work comes in showing that, if $\bar{G}_M$ is not linear, then $L(M)$ is not linear context-free. This involves using special pumping properties of linear context-free languages to show that certain classes of bounded languages are not linear context-free.

We start by establishing terminology for realtime pushdown store automata and context-free grammars. For our machines, we use a variant of the notation in (Valiant, 1973).

A *realtime pushdown store automaton* (*pda*) is denoted by $M = (K, \Sigma, \Gamma, H, q_0, Z_0, F)$ where K is a finite set of *states*, Σ is a finite set of *input symbols*, Γ is a finite set of *pushdown store* (*pds*) *symbols*, q_0 in K is the *initial state*, Z_0 in Γ is the *initial pushdown store symbol*, $F \subseteq K$ is the set of *accepting* or *final states* and the transition set H is a finite subset of $K \times \Gamma \times \Sigma \times K \times \Gamma^*$ obeying the following restrictions.[1] We write (q, A, a, p, y) in H as $(q, A) \to^a (p, y)$ and call (q, A) the *mode* of the rule with input a; our condition on H is that H contains,

[1] We use e for the empty word and let $L^+ = \{w_1 \cdots w_n \mid n \geqslant 1, w_i \text{ in } L\}$ and $L^* = L^+ \cup \{e\}$; we write $\{w\}^+$ as w^+.

for each input symbol a and mode (q, A), at most one rule with mode (q, A) and input a. We call M 1-*increasing* if for each rule (q, A, a, p, y) in H, $| y | \leqslant 2$.

A pair (q, yA), q in K, A in Γ, y in Γ^*, is a *configuration with mode* (q, A) while (q, e) is a *configuration with mode* (q, e). If $(q, A) \to^a (p, y)$ in H, then we write $(q, uA) \to^a (p, uy)$ for any u in Γ^* and call it a 1-*step computation*. If $c_1 \to^u c_2$ and $c_2 \to^v c_3$, we write $c_1 \to^{uv} c_2$ and let $c \to^e c$ trivially for any configuration c.

Our machines accept by final state and empty store so the *language accepted by* M is

$$L(M) = \{w \text{ in } \Sigma^* \mid \text{for some } f \text{ in } F,\ (q_0, Z_0) \xrightarrow{w} (f, e)\}.$$

Unlike Valiant, we do not allow the *pda* to operate with empty stack (no rules (q, e, a, p, y)). This avoids some complications in notation, but does not affect the classes of languages involved because we allow endmarkers.

For a configuration $c = (q, y)$, the state of c is state$(c) = q$ and the *stack height* of c is $| c | = | y |$.[2] If we have a series of 1-step computations $c_1 \to^{d_1} c_2 \to^{d_2} c_3 \to^{d_3} \cdots \to^{d_n} c_{n+1}$ with $| c_1 | \leqslant | c_i |, 2 \leqslant i \leqslant n + 1$, we write $c_1 \uparrow (a_1 \cdots a_n) c_{n+1}$. This is a *stacking computation*. On the other hand, if $| c_i | \geqslant | c_{n+1} |, 1 \leqslant i \leqslant n$, we write $c_1 \downarrow (a_1 \cdots a_n)\, c_{n+1}$ and call it a *popping computation*. Observe that, if $| c_1 | = | c_{n+1} |$, the computation can be both stacking and popping.

A *pda* makes a "turn" when it changes from stacking to popping; that is, if there is a subcomputation

$$c_1 \uparrow (v)\, c_2 \downarrow (u)\, c_3$$

with $| c_1 | < | c_2 |$ and $| c_2 | > | c_3 |$. If $| c_1 | = | c_3 |$, we can consider this to be a "hump" in the computation. A *pda* is *single turn* if no computation from the initial configuration (q_0, Z_0) contains more than one turn; it is *finite turn* if there is a $k \geqslant 1$ such that no computation from the initial configuration contains more than k distinct turns.

DEFINITION. A language L is *realtime* if there is a realtime *pda* M such that either $L = L(M)$ or $L\$ = L(M)$ for some symbol $\$$.

We write a context-free grammar as $G = (V, \Sigma, P, S)$ where V is a finite vocabulary, $\Sigma \subset V$, S is in $V - \Sigma$ and P is a finite set of rules of the form $Z \to y$, $Z \in V - \Sigma$, $y \in V^*$. Members of Σ are called *terminals* and members of $V - \Sigma$, *nonterminals*. For u, v in V^*, and $Z \to y$ in P, we write $uZv \Rightarrow uyv$ and, if $u \in \Sigma^*$, $uZv \Rightarrow^L uyv$. We let $\overset{*}{\Rightarrow}$ ($\Rightarrow^{L*}$) be the transitive reflexive extension of $\Rightarrow$ ($\Rightarrow^L$). The *language generated by* G is

$$L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}.$$

[2] We denote the length of a word w by $| w |$.

Context-free grammar G is *linear* if, for each rule $Z \to y$ in P, y contains at most one nonterminal. A language L is a *linear context-free language* if $L = L(G)$ for some linear context-free grammar G. The grammar G is *reduced* either if $G = (\{S\}, \varnothing, \varnothing, S)$ or if, for each $A \in V$, there are $x, y \in V^*$ with $S \overset{*}{\Rightarrow} xAy$ and, for each $A \in V - \Sigma$, there is a word $w \in \Sigma^*$ with $A \overset{*}{\Rightarrow} w$. For each grammar $G = (V, \Sigma, P, S)$, there is a reduced subgrammar $G' = (V', \Sigma', P', S)$ with $V' - \Sigma' \subseteq V - \Sigma$, $\Sigma' \subseteq \Sigma$, $P' \subseteq P$, and $L(G) = L(G')$ (Bar-Hillel, Perles and Shamir, 1961).

We need a lemma giving some of the special pumping or iterative properties of a linear context-free language. Let us establish some special notation for ease of presentation. Let w be a nonempty word in a language L. We say that a nonempty word v is an *iterative subword* of w with respect to L if $w = uvz$ and uv^nz is in L for all $n \geqslant 0$. A pair (v, y) with $vy \neq e$ is an *iterative pair in w with respect to L* if $w = uvxyz$ and uv^nxy^nz is in L for all $n \geqslant 0$; if $v \neq e \neq y$, then (v, y) is a *proper iterative pair*. Usually, we omit "with respect to L" since the language L is clear from the context.

The following pumping lemma for linear context-free languages can be proven by standard methods; the proof is omitted.

LEMMA 1. *Let L be a linear context-free language. There is an integer $k > 1$ with the following properties. If w is in L and $| w | \geqslant k$, then w can be decomposed as*

$$w = u_1v_1 \cdots u_tv_txy_tz_t \cdots y_1z_1$$

where:

(1) $t \leqslant k$ *and* $| u_1u_2 \cdots u_txz_t \cdots z_2z_1 | < k$,

(2) *for* $1 \leqslant i \leqslant t$, $v_iy_i \neq e$ *and* (v_i, y_i) *is an iterative pair in w, and*

(3) *for* $1 \leqslant i \leqslant t$, *if* $| v_i | \geqslant k | y_i |$, *then v_i contains a subword iterative in w and if* $| y_i | \geqslant k | v_i |$, *then y_i contains a subword iterative in w.*

The heart of the proof lies in the following technical lemma which says that bounded languages with certain properties cannot be linear context-free. In our main lemma (Lemma 3), we shall show that if $\bar{G}_M$ (to be defined formally later) is not linear, then there is a finite state transduction of $L(M)$ which has the properties of the language in Lemma 2 and so is not linear context-free; whence, $L(M)$ is not linear context-free.

LEMMA 2. *Let $L \subseteq a^+b^+c^+d^+$ be a language such that*

(1) $a^nb^nc^rd^r \in L$ *for all* $n, r \geqslant 1$,

(2) *if* $a^nb^nc^rd^s$ *is in L, then $r \leqslant s$, and*

(3) *there are integers* $t_1, t_2 \geqslant 1$ *such that, if* $a^nb^mc^rd^s$ *is in L and $n > m$, then* $(n - m)\, t_1 \leqslant (r + s)\, t_2$.

Then L is not linear context-free.

Proof. Assume that L is linear context-free. Let k be the integer in Lemma 1 and consider a word

$$w = a^n b^n c^r d^r$$

with $n > (2t_2/t_1 + 1)k$ and $r > 2kn + 3k$. Condition (1) guarantees that $w \in L$.

We can make some observations on iterative pairs and iterative subwords in L. Condition (2) implies that w cannot contain an iterative subword c^p and that, if (c^p, d^q) is an iterative pair in w, then $p \leqslant q$. Similarly, Condition (3) implies that w cannot contain an iterative subword a^p and that, if (a^p, d^q) is an iterative pair, then $q \geqslant (t_1/t_2)\, p$. If (a^p, c^q) or (b^p, c^q) is an iterative pair in w, then $q \leqslant kp$ (or else w would have an iterative subword in c^q).

Now consider the factorization

$$w = u_1 v_1 \cdots u_t v_t x y_t z_t \cdots y_1 z_1$$

given by Lemma 1. Since $n > k$ and $r > k$, the a's and c's must be parts of proper iterative pairs (v_i, y_i); clearly, $v_i, y_i \in a^* \cup b^* \cup c^* \cup d^*$. Let us examine the possibilities.

If, for any i, $v_i \in a^+$ and $y_i \in c^+$, then all but (at most) k c's must form proper iterative pairs with the a's and b's. But then $r - k \leqslant 2kn$, a contradiction. If, for any i, $v_i \in a^+$ and $y_i \in b^+$, then the c's must enter iterative pairs with the a's and we obtain again a contradiction. If the b's enter iterative pairs with the c's, then again the c's can enter pairs only with the a's and b's and we obtain a contradiction. Finally, if none of the above occurs, the a's, b's and c's all enter iterative pairs only with d's (the b's can form iterative subwords). Hence (ignoring the b's), the number of d's compared to the number of a's and c's must satisfy

$$r \geqslant \frac{t_1}{t_2}(n - k) + (r - k) > \frac{t_1}{t_2}\left(2\,\frac{t_2}{t_1}\,k\right) + (r - k) > r + k$$

a contradiction.

So L cannot be linear context-free. ∎

Let $M = (K, \Sigma, \Gamma, H, q_0, Z_0, F)$ be a *pda*. The *canonical grammar associated with* M, denoted G_M, is the equivalent reduced subgrammar of the grammar with nonterminal set

$$V = (K \times \Gamma \times K) \cup \{S\}$$

for S a new symbol and production set

$$\{S \to (q_0, Z_0, f) \mid f \text{ in } F\} \cup \{(p, Z, q) \to a \mid (p, Z, a, q, e) \text{ in } H\}$$
$$\cup \{(p, Z, q_2) \to a(q_1, Y, q_2) \mid (p, Z, a, q_1, Y) \text{ in } H,\ Y \text{ in } \Gamma,\ q_2 \text{ in } K\}$$
$$\cup \{(p, Z, q_n) \to a(p_1, Y_1 q_1)(q_1, Y_2, q_2) \cdots (q_{n-1}, Y_n, q_n) \mid (p, Z, a, p_1, Y_n \cdots Y_1) \text{ in } H,\ n \geqslant 2,\ Y_i \text{ in } \Gamma,\ q_i \text{ in } K\}.$$

Not only is $L(M) = L(G_M)$, but left-to-right derivations in G_M are closely related to computations in M (Ginsburg, 1966).

Let h be the homomorphism defined by $h((p, Z, q)) = Z$. If

$$(p, Z, p_{n+1}) \overset{L^*}{\Rightarrow} w_0(p_1, Y_1, q_1)\alpha_1{}^R \cdots (p_n, Y_n, q_n)\alpha_n{}^R$$

with w_0 in Σ^*, α_i in $(K \times \Gamma \times K)^*$, then in M

$$(p, Z) \xrightarrow{w_0} (p_1, h(\alpha_n)Y_n \cdots h(\alpha_1)Y_1)$$

and if

$$\alpha_i{}^R \overset{*}{\Rightarrow} w_i, \qquad (p_i, Y_i, q_i) \overset{*}{\Rightarrow} x_i$$

with w_i and x_i in Σ^*, then in M

$$(p_i, Y_i) \xrightarrow{x_i} (q_i, e)$$

and

$$(q_i, h(\alpha_i)) \xrightarrow{w_i} (p_{i+1}, e).^3$$

A nonterminal Z in $G = (V, \Sigma, P, S)$ is *self-embedding* if $Z \overset{*}{\Rightarrow} uZv$ for some nonempty terminal strings u and v in Σ^+. Let us call Z *regular* if Z is not self-embedding and $Z \overset{*}{\Rightarrow} y$ implies that y does not contain any self-embedding nonterminals. If Z is regular, then the language generated by Z

$$L(G, Z) = \{w \text{ in } \Sigma^* \mid Z \overset{*}{\Rightarrow} w\}$$

is regular (Chomsky, 1959). The converse is generally *not* true. However, it is true if G is the canonical reduced grammar for a realtime *dpda.* (cf. Harrison and Havel, 1972).

To any reduced grammar $G = (V, \Sigma, P, S)$, we can associate its *regular reduction* $\bar{G} = (\bar{V}, \bar{\Sigma}, \bar{P}, \bar{S})$ which is essentially G except that symbols regular in G are treated as terminals in $\bar{G}$. Formally, if all nonterminals in G are regular, then the start symbol S is regular. In this case, we let $\bar{S}$ be new and set $\bar{G} = (V \cup \{\bar{S}\}, V, \{\bar{S} \to S\}, \bar{S})$; otherwise, we let $S = \bar{S}$, $\bar{V} = V$, $\bar{\Sigma} = \Sigma \cup \{Z \text{ in } V - \Sigma \mid Z \text{ is regular in } G\}$, and $\bar{P} = P - \{Z \to y \mid Z \text{ is regular in } G\}$. We define the corresponding regular substitution[4] τ_G by $\tau_G(a) = \{a\}$, a in Σ and $\tau_G(Z) = L(G, Z)$ for Z in $\bar{\Sigma} - \Sigma$. Clearly, $L(G) = \tau_G(L(\bar{G}))$. The family of linear context-

[3] Let w^R be the *reversal* of w; so $e^R = e$ and, if $a_1, \ldots, a_n$ are symbols, $(a_1 \cdots a_n)^R = a_n \cdots a_1$.

[4] For each symbol a in a finite alphabet Δ, let $\tau(a)$ be a language. Extend τ to Δ^* by $\tau(e) = \{e\}$ and $\tau(xy) = \tau(x)\,\tau(y)$ and for $L \subseteq \Delta^*$, let $\tau(L) = \bigcup_{w \text{ in } L} \tau(w)$. We call τ a *substitution*; it is a *regular substitution* if $\tau(a)$ is regular for each a in Δ. A family $\mathscr{L}$ is *closed under regular substitution* if $\tau(L)$ is in $\mathscr{L}$ for each L in $\mathscr{L}$ and regular substitution τ on L.

free languages is closed under regular substitution (Ginsburg and Spanier, 1966) so, if $\bar{G}$ is linear, so is $L(G)$. The converse is not generally true but does hold in the case of interest here.

We also use the fact that the family of linear context-free languages is closed under finite state transductions (Ginsburg and Spanier, 1966). For this purpose, we define a finite state transducer as an *a-transducer* $T = (K, \Sigma, \Delta, H, q_0, F)$ where K is a finite set of states, Σ and Δ are the finite input and output vocabularies, q_0 in K is the initial state, $F \subseteq K$ is the set of final or accepting states, and H is a finite subset of $K \times \Sigma^* \times \Gamma^* \times K$, the transition set. A configuration of T is a member of $K \times \Sigma^* \times \Gamma^*$. If (q, u, v, p) is in H and (q, uw, z) is a configuration, we write $(q, uw, z) \vdash (p, w, zv)$; we let $\overset{*}{\vdash}$ be the transitive reflexive extension of $\vdash$. We define

$$T(w) = \{z \mid \exists f \text{ in } F, (q_0, w, e) \overset{*}{\vdash} (f, e, z)\}$$

and for a language L, the *tranduction of L by T* is

$$T(L) = \bigcup_{w \text{ in } L} T(w).$$

Now we are ready for our main lemma.

LEMMA 3. *Let M be a realtime dpda. If $\bar{G}_M$ is not linear context-free, then $L(M)$ is not linear context-free.*

Proof. Let $M = (K, \Sigma, \Gamma, H, q_0, Z_0, F)$. Since $\bar{G}_M$ is not linear context-free, we must have in G_M:

$$\begin{aligned} S &\overset{L^*}{\Rightarrow} u(p_1, Y_1, q_1)\gamma, \\ \gamma &\overset{L^*}{\Rightarrow} x_2(p_2, Y_2, q_2)\eta, \\ \eta &\overset{*}{\Rightarrow} z, \\ (p_1, Y_1, q_1) &\overset{*}{\Rightarrow} x_1, \\ (p_2, Y_2, q_2) &\overset{*}{\Rightarrow} x_3, \end{aligned}$$

where $x_1, x_2, x_3 \in \Sigma^+$, $u, z \in \Sigma^*$ and (p_1, Y_1, q_1) and (p_2, Y_2, q_2) are self-embedding in G_M.

Since (p_1, Y_1, q_1) and (p_2, Y_2, q_2) are self-embedding in G_M (and self-embedding derivations can be repeated as needed) and M is realtime, G_M must have derivations:

$$\begin{aligned} (p_1, Y_1, q_1) &\overset{L^*}{\Rightarrow} v_1(p_1, Y_1, q_1)\alpha, \\ (p_2, Y_2, q_2) &\overset{L^*}{\Rightarrow} v_2(p_2, Y_2, q_2)\beta, \\ \alpha &\overset{*}{\Rightarrow} y_1, \\ \beta &\overset{*}{\Rightarrow} y_2 \end{aligned}$$

with $v_1, v_2, y_1, y_2 \in \Sigma^+$, $|\alpha|, |\beta| \geqslant |ux_1x_2x_3z|$, $|y_1| = |v_2|$ and $|y_1|$ is a multiple of $|x_2|$.

Let h be the homomorphism given by $h((p, Z, q)) = Z$, and let $\gamma_1 = h(\gamma^R)$, $\eta_1 = h(\eta^R)$, $\alpha_1 = h(\alpha^R)$ and $\beta_1 = h(\beta^R)$. In M, we have:

$$
\begin{aligned}
&(q_0, Z_0) \xrightarrow{u} (p_1, \gamma_1 Y_1),\\
&(p_1, Y_1) \uparrow (v_1)(p_1, \alpha_1 Y_1),\\
&(p_1, Y_1) \uparrow (x_1)(q_1, e),\\
&(q_1, \alpha_1) \downarrow (y_1)(q_1, e),\\
&(q_1, \gamma_1) \xrightarrow{x_2} (p_2, \eta_1 Y_2),\\
&(p_2, Y_2) \uparrow (v_2)(p_2, \beta_1 Y_2),\\
&(p_2, Y_2) \uparrow (x_3)(q_2, e),\\
&(q_2, \beta_1) \downarrow (y_2)(q_2, e),\\
&(q_2, \eta_1) \xrightarrow{z} (f, e)
\end{aligned}
$$

for some f in F.

Let R_1 be the regular set

$$R_1 = uv_1^+x_1y_1^+x_2v_2^+x_3y_2^+z$$

and $L_1 = L(M) \cap R_1$. We shall show that L_1 is not linear context-free and hence $L(M)$ is not linear context-free.

Let T_1 be the finite state transducer with state set $\{1, 2, 3, 4, 5, 6\}$, initial state 1 and final state 6, and transition set:

$$
\begin{aligned}
\{&(1, u, e, 2), (2, v_1, a, 2), (2, x_1, e, 3), (3, y_1, b, 3), (3, x_2, e, 4),\\
&(4, v_2, c, 4), (4, x_3, e, 5), (5, y_2, d, 5), (5, z, e, 6)\}.
\end{aligned}
$$

We shall show that $L = T_1(L_1)$ satisfies the hypotheses of Lemma 2 and so L, L_1 and $L(M)$ are not linear context-free.

Clearly, for all $n, r \geqslant 1$, $uv_1^nx_1y_1^nx_2v_2^rx_3y_2^rz$ is in L_1, and $a^nb^nc^rd^r$ in L. If $a^nb^nc^rd^s$ is in L, then $uv_1^nx_1y_1^nx_2v_2^rx_3y_2^sz$ is in L_1. Thus, input $uv_1^nx_1y_1^nx_2v_2^rx_3$ takes M from (q_0, Z_0) to $(q_2, \eta_1\beta_1^r)$. If $s < r$, then y_2^s takes M to $(q_2, \eta_1\beta_1^{r-s})$. But, since $|\beta_1| > |z|$, input z cannot erase the store and bring M to an accepting configuration. Hence, we must have $s \geqslant r$. If $a^nb^mc^rd^s \in L$, with $n > m$, then $uv_1^nx_1y_1^mx_2v_2^rx_3y_2^sz \in L_1$. Input $uv_1^nx_1y_1^m$ takes M from (q_0, Z_0) to $(q_1, \gamma_1\alpha_1^{n-m})$. Hence

$$|x_2x_3z| + r|v_2| + s|y_2| \geqslant |\gamma_1| + (n - m)|\alpha_1|.$$

Let $t_1 = |\alpha_1|$ and $t_2 = |v_2| + |y_2| + |x_2x_3z|$. Then

$$t_2(r + s) \geqslant |x_2x_3z| + r|v_2| + s|y_2| \geqslant |\gamma_1| + (n - m)t_1 \geqslant (n - m)t_1.$$

So L satisfies Lemma 2 and thus $L(M)$ is not linear context-free. ∎

If $\bar{G}_M$ is linear context-free, this implies that in M no accepting computation has more than one "hump" of size exceeding $k = h(\#K)^2(\#\Gamma)^2$, where $h = \text{Max}\{|\, y \,| \mid M \text{ has a transition } (q, A, a, p, y)\}$.[5] Thus, we can build from M an equivalent deterministic *pda* $\overline{M}$ which "smooths out" the small humps and so is single turn. Machine $\overline{M}$ stores in its finite state control the last $k + 1$ pushdown store symbols and, having stored Ay, $|\, y \,| = k$, pushes A. When M pops, $\overline{M}$ does not pop until it runs out of stored symbols, in which case it pops and stores the next $k + 1$ symbols (or less, if the store is smaller). Machine $\overline{M}$ also records the number of "large" humps in the computation of M, which is equal to the number of times $\overline{M}$ has had to pop after storing $k + 1$ symbols. So a "turn" in $\overline{M}$ corresponds to a large hump in M and, if two occur, $\overline{M}$ blocks. Thus $\overline{M}$ is single turn. It is not necessarily realtime, but makes at most $k + 1$ moves before advancing the input tape and hence can be recoded to be realtime and single turn. So we have the following corollary.

COROLLARY. *For a realtime pda M, if $L(M)$ is linear context-free, then we can construct from M a realtime single turn pda $\overline{M}$ such that $L(M) = L(\overline{M})$. Hence a linear context-free language L can be accepted by final state and empty store by a realtime pda if and only if it can be accepted by final state and empty store by a single turn realtime pda.*

This corollary does not hold if M does not accept by final state and empty store or is not realtime. For example, if

$$L = \{a^{n_1}c^{m_1} \cdots a^{n_r}b^{m_r} \mid r \geqslant 1,\ n_i, m_i \geqslant 1, \text{ there exists } j, \quad 1 \leqslant j \leqslant r \text{ such that } m_j = n_j\},$$

then L is linear context-free and can be accepted by final state by a realtime *pda* but no finite turn deterministic pushdown store automaton can accept L.

Using Lemma 3, we see that it is polynomially decidable whether $L(M)$ is linear context-free for M realtime.

THEOREM 1. *It is polynomially decidable for a realtime pda M whether $L(M)$ is linear context-free.*

Proof. Valiant (1973) has shown that one can go from a realtime *pda* M to an equivalent 1-increasing realtime *pda* M_1 and thence to G_{M_1} with only polynomial increase in size. Reducing a grammar G, testing whether a nonterminal is self-embedding or whether one nonterminal is reachable from another, can clearly be done in time polynomial in the time it takes to test whether $L(G)$ is empty and the latter is polynomially decidable (Valiant, 1973). Thus, one can go from G_{M_1} to $\bar{G}_{M_1}$ in polynomial time and certainly test whether $\bar{G}_{M_1}$ is linear context-free

[5] For a finite set A, $\#A$ is the number of elements in A.

in linear time. So one can, given M, construct M_1 and $\bar{G}_{M_1}$ and test whether $\bar{G}_{M_1}$ is linear context-free in polynomial time. If $\bar{G}_{M_1}$ is linear context-free then, as we have observed above, $L(M) = L(M_1) = \tau_{G_1}(L(\bar{G}_{M_1}))$ is linear context-free. If $\bar{G}_{M_1}$ is not linear context-free, then by Lemma 3, $L(M) = L(M_1)$ is not linear context-free. ∎

The class of linear context-free languages is closed under addition or deletion of endmarkers, so one can allow endmarkers on our *pda*.

COROLLARY. *It is polynomially decidable whether L is linear context-free if L is given by $L\$ = L(M)$ for a realtime pda M.*

Using the corollary to Lemma 3, we obtain the following consequence of Theorem 1.

THEOREM 2. *"$L(M_1) = L(M_2)$" is decidable in time bounded by*

$$2^{2^{2^{p(n)}}}$$

for M_1 a realtime pda, M_2 a single turn deterministic pda, and $p(n)$ a polynomial in the size of M_1 and M_2.

Proof. If $L(M_1)$ is not linear context-free, then certainly $L(M_1) \neq L(M_2)$. We can decide whether $L(M_1)$ is linear context-free in polynomial time by Theorem 1. If it is, we can construct (by the corollary to Lemma 3) a single turn realtime *pda* M_3 with $L(M_1) = L(M_3)$. Further, the construction can be carried out so that the size of M_3 is bounded by $2^{q(n)}$ for $q(n)$ a polynomial in the size of M_1. Equivalence is decidable for single turn deterministic *pda* in time bounded by $2^{2^{\bar{p}(n)}}$ for $\bar{p}(n)$ a polynomial in the size of the machines (Beeri, 1976). ∎

Remarks. The decidability of the problem in Theorem 2 is implied by results in (Oyamaguchi *et al*, 1978) and (Greibach and Friedman, 1978b). However, the bound in the first case is unknown and, in the second case, involves one more exponential than the bound in Theorem 2.

We conjecture that it is polynomially decidable whether $L(M)$ is finite turn for M realtime and that in fact $L(M)$ is finite turn if and only if $\bar{G}_M$ is nonterminal bounded and Theorem 2 holds for M_2 finite turn. However, in the more general case, the proof of the analog of Lemma 2 is unpleasant; the analog of Lemma 3 is long but straightforward. Lemma 3 no longer holds if M accepts by final store alone or if M is an arbitrary deterministic pushdown store automaton; however, we conjecture that it is still decidable whether the language accepted by M is linear context-free.

RECEIVED: March 17, 1978; REVISED: December 27, 1978

REFERENCES

BAR-HILLEL, Y., PERLES, M., AND SHAMIR, E. (1961), On formal properties of simple phrase structure grammars, *Z. Phonetik. Sprach. Kommun.* **14**, 143–172.

BEERI, C. (1976), An improvement on Valiant's decision procedure for equivalence of deterministic finite turn pushdown machines, *Theor. Comput. Sci.* **3**, 305–320.

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

FRIEDMAN, E. P. (1977a), Equivalence problems for deterministic context-free languages and monadic recursion schemes, *J. Comput. System. Sci.* **14**, 344–359.

FRIEDMAN, E. P. (1977b), Simple context-free languages and free monadic recursion schemes, *Math. Syst. Theory* **11**, 9–28.

FRIEDMAN, E. P. (1978), A note on nonsingular deterministic pushdown automata, *Theor. Comput. Sci.* **7**, 333–340.

FRIEDMAN, E. P., AND GREIBACH, S. (1978), Superdeterministic PDAs: The method of accepting does affect decision problems, *J. Comput. System Sci.*, in press.

GELLER, M., AND HARRISON, M. A. (1977), On LR(k) grammars and languages, *Theor. Comput. Sci.* **3**, 245–276.

GINSBURG, S. (1968), "The Mathematical Theory of Context-Free Languages," McGraw–Hill, New York.

GINSBURG, S., AND GREIBACH, S. (1966), Deterministic context-free languages, *Inform. Contr.* **9**, 620–648.

GINSBURG, S., AND SPANIER, E. H. (1966), Finite-turn pushdown automata, *SIAM J. Contr.* **4**, 429–453.

GREIBACH, S. (1969), An infinite hierarchy of context-free languages, *J. Assoc. Comput. Mach.* **16**, 91–106.

GREIBACH, S., AND FRIEDMAN, E. P. (1978a), Superdeterministic PDAs: A subcase with a decidable inclusion problem, submitted for publication.

GREIBACH, S., AND FRIEDMAN, E. P. (1978b), Some results on the extended equivalence problem for classes of dpdas, in preparation.

HARRISON, M. A., AND HAVEL, I. M. (1972), Real-time strict deterministic languages, *SIAM J. Comput.* **1**, 333–349.

KORENJAK, A. J., AND HOPCROFT, J. E. (1966), Simple deterministic languages, *in* "Proc. IEEE 7th Annual Symp. on Switching and Automata Theory, Berkeley, Calif.," pp. 36–46.

MCNAUGHTON, R. (1967), Parenthesis grammars, *J. Assoc. Comput. Mach.* **14**, 490–500.

OYAMAGUCHI, M., HONDA, N., AND INAGAKI, Y. (1978), The equivalence problem for real-time strict deterministic languages, unpublished paper.

ROSENKRANTZ, D. J., AND STEARNS, R. E. (1970), Properties of deterministic top-down grammrs, *Inform. Contr.* **17**, 226–255.

STEARNS, R. E. (1967), A regularity test for pushdown machines, *Inform. Contr.* **11**, 323–340.

VALIANT, L. G. (1973), "Decision Procedures for Families of Deterministic Pushdown Automata," Ph. D. dissertation, University of Warwick, July 1973.

VALIANT, L. G. (1974), The equivalence problem for deterministic finite-turn pushdown automata, *Inform. Contr.* **25**, 123–133.

VALIANT, L. G. (1975), Regularity and related problems for deterministic pushdown automata, *J. Assoc. Comput. Mach.* **22**, 1–10.

VALIANT, L. G., AND PATERSON, M. S. (1975), Deterministic one-counter automata, *J. Comput. System. Sci.* **10**, 340–350.