KYBERNETIKA — VOLUME 17 (1981), NUMBER 3

CHARACTERIZATIONS OF STACK UNIFORM STRICT DETERMINISTIC LANGUAGES*

JAN PITTL

Three new characterizations for the class of languages accepted with empty store by a stack uniform deterministic pushdown automaton are presented. This class is shown to coincide with the classes of languages generated by length uniform grammars and length uniform strict deterministic grammars respectively. The result is then used for proving the interesting fact that the power of nondeterministic stack uniform automata is the same, i.e. all the four devices mentioned above possess the same capability of defining languages.

1. INTRODUCTION

Among many other problems related to the theory of deterministic context-free languages there appeared two of particular significance, namely, the problems of deciding whether two given languages are equal and whether a language forms a subset of another one. They are often referred to as the equivalence and the inclusion problem, respectively. Recent investigations in this field have focused their attention on the family of languages accepted in real time with empty store by a deterministic pushdown automaton.

The equivalence problem for this class, having represented one of the well known open problems for a long time, has been recently solved by Oyamaguchi et al. [9]. As far as the inclusion problem is concerned, two subsets of the real time family have been found, each of them possessing quite different properties. Friedman [2] has shown that this problem is undecidable for simple deterministic languages (cf. Korenjak and Hopcroft [6]). On the other hand, the inclusion problem is decidable for the class of stack uniform strict deterministic languages (U_0 languages, see Linna [7]).

* A preliminary report on this research was presented at the 7th International Symposium on Mathematical Foundations of Computer Science, Zakopane, Poland, September 1978, cf. [10].

Most results mentioned above deal only with automata. In practice, languages are often defined by means of other generating devices, particularly by grammars. The use of grammars is of considerable interest. In many cases, the existence of a convenient grammatical characterization provides a deeper insight into the structure of the languages generated (compare [4, 5, 6]).

The purpose of this paper is to present and utilize a new grammatical concept, namely the concept of a *length uniform grammar*. These are context-free grammars in Greibach Normal Form satisfying the condition that the right hand sides of productions starting with the same terminal symbol agree in length. We first investigate a deterministic version of this grammar family, more precisely, length uniform grammars being in addition strict deterministic (cf. Harrison and Havel [4]). We show that such grammars generate exactly U_0 languages i.e. that the concept of a length uniform strict deterministic grammar provides a grammatical characterization of the class U_0 (none has been known till now).

The notion of stack uniformity, as introduced by Linna [7], can be generalized in a straightforward manner to deal with nondeterministic pushdown automata working in real time. It is a well known fact that for pushdown automata the assumption of determinism leads to a considerable restriction on the family of languages accepted. We show that for stack uniform machines it is not the case, i.e. that nondeterministic stack uniform pushdown automata accept merely U_0 languages. Thus we prove the existence of a nontrivial class of nondeterministic machines for which the inclusion problem is decidable.

It is necessary to have the terminology to deal consistently with pushdown automata and context-free grammars. We use a variant of the notation in Valiant [11], Harrison and Havel [4]. We adopt familiar conventions of Aho and Ullman [1] concerning strings and languages.

A *pushdown automaton* (abbreviated *PDA*) is a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ where Q is a finite nonempty *set of states*, Σ and Γ are two alphabets called the *input alphabet* and the *pushdown store alphabet* respectively, $q_0 \in Q$ is the *initial state*, $Z_0 \in \Gamma$ is the *initial pushdown symbol*, $F \subseteq Q$ is the set of *final states*, and δ is a function, mapping the set $Q \times \Sigma_\Lambda \times \Gamma$ into finite subsets of $Q \times \Gamma^*$ where Λ denotes the *empty word*, $\Sigma_\Lambda = \Sigma \cup \{\Lambda\}$.

We now describe how PDA's move. A *configuration* of the PDA M is a pair (q, γ) where $q \in Q$, $\gamma \in \Gamma^*$. For any $p, q \in Q$, $a \in \Sigma_\Lambda$, $\alpha, \beta \in \Gamma^*$, $Z \in \Gamma$ we write

$$(q, \alpha Z) \vdash_M^a (p, \alpha\beta) \quad \text{iff} \quad (p, \beta) \in \delta(q, a, Z)$$

read "M makes a *move* from configuration $(q, \alpha Z)$ to configuration $(p, \alpha\beta)$ while reading a". The symbol M will be omitted whenever there is no danger of misunderstanding. We extend this notation to *computations* (= finite sequences of moves) by writing

$$c_1 \vdash^{w_1 w_2} c_3 \quad \text{if} \quad c_1 \vdash^{w_1} c_2 \quad \text{and} \quad c_2 \vdash^{w_2} c_3$$

for $w_1, w_2 \in \Sigma^*$ and configurations c_1, c_2, c_3.

Our machines will accept by final state and empty store. The *language strictly accepted* by the PDA M is the set

$$L(M) = \{w \in \Sigma^* \mid (q_0, Z_0) \vdash_M^w (q, \Lambda) \text{ for some } q \in F\} .$$

The PDA M is said to be a *deterministic PDA* (*DPDA* for short) if for any $q \in Q$, $Z \in \Gamma$, $a \in \Sigma$ we have

$$|\delta(q, \Lambda, Z)| + |\delta(q, a, Z)| \leqq 1$$

where $|X|$ denotes the cardinality of the set X. The PDA M is called a *real-time PDA* if $\delta(q, \Lambda, Z) = \emptyset$ for all $q \in Q$, $Z \in \Gamma$.

A *context-free grammar* is a quadruple $G = (V, \Sigma, P, S)$ where V and Σ are two alphabets, $\Sigma \subseteq V$ (letters in Σ and in $N = V - \Sigma$ are called *terminals* and *nonterminals* respectively), $S \in N$ and P is a finite subset of $N \times V^*$ (the *set of productions*). As usual, we write $A \to \alpha$ instead of (A, α) for elements of P.

We define a relation $\Rightarrow \subseteq V^* \times V^*$ as follows. For any $\alpha, \beta \in V^*$, $\alpha \Rightarrow \beta$ if $\alpha = \alpha_1 A \alpha_2$, $\beta = \alpha_1 \gamma \alpha_2$ and $A \to \gamma$ is in P for some $A \in N$, $\alpha_1, \alpha_2, \gamma \in V^*$. The reflexive transitive closure of $\Rightarrow$ is denoted by $\Rightarrow^*$. The *language generated* by G is the set

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

The grammar G is said to be in *Greibach Normal Form*[1]) if $P \subseteq N \times \Sigma N^*$ i.e. any production in P is of the form $A \to a\alpha$ where $A \in N$, $a \in \Sigma$, $\alpha \in N^*$.

2. PRELIMINARIES ON STRICT DETERMINISTIC GRAMMARS

The family of strict deterministic grammars was introduced by Harrison and Havel [4]. We now review some fundamental definitions and several useful facts concerning these grammars that will be utilized later on.

Definition 2.1. Let X be a set. A set π of nonempty pairwise disjoint subsets of X is called a *partition of* X iff for any $a \in X$ there is some $Y \in \pi$ such that $a \in Y$. The elements of π are called *blocks* of the partition π. For $a, b \in X$ we write

$$a \equiv b \pmod{\pi} \quad \text{iff} \quad a \in Y \quad \text{and} \quad b \in Y \quad \text{for some} \quad Y \in \pi .$$

Strict deterministic grammars are context-free grammars possessing special properties with respect to some partition of the total alphabet.

Definition 2.2. Let $G = (V, \Sigma, P, S)$ be a context-free grammar, π a partition of V. Such a partition is called *strict* iff Σ forms a block of π and for any $A, A' \in N$, $\alpha, \beta, \beta' \in V^*$, if $A \to \alpha\beta$, $A' \to \alpha\beta'$ are in P, and $A \equiv A' \pmod{\pi}$ then either

[1]) Some authors allow GNF grammars to have $P \subseteq N \times \Sigma V^*$. Our definition is then referred to as Greibach Standard Form.

(i) both $\beta, \beta' \neq \Lambda$ and ${}^{(1)}\beta \equiv {}^{(1)}\beta' \pmod{\pi}$, or
(ii) $\beta = \beta' = \Lambda$ and $A = A'$,
where ${}^{(1)}\gamma$ denotes the first symbol of a word $\gamma \in V^+$.

Definition 2.3. A context-free grammar $G = (V, \Sigma, P, S)$ is called *strict deterministic* iff there exists some strict partition π of V.

There are two constructions related to the theory of strict deterministic grammars that are of fundamental importance for the development of our results. The former enables us to reduce the number of final states of some PDA to one without disturbing the properties of the automaton under consideration.

Definition 2.4. Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA. We define the PDA $\overline{M}$ as follows. $\overline{M} = (Q, \Sigma, \Gamma \cup \bar{\Gamma}, \bar{\delta}, q_0, \bar{Z}_0, \{q_f\})$ where $q_f \in F$ is chosen arbitrarily, $\bar{\Gamma} = \{\bar{Z} \mid Z \in \Gamma\}$ and the function $\bar{\delta}$ is defined below. For any $p, q \in Q$, $a \in \Sigma_\Lambda$, $Y, Z \in \Gamma$, $\gamma \in \Gamma^*$
(i) $\bar{\delta}(q, a, Z) = \delta(q, a, Z)$,
(ii) $(p, \bar{Y}\gamma) \in \bar{\delta}(q, a, \bar{Z})$ if $(p, Y\gamma) \in \delta(q, a, Z)$,
(iii) $(q_f, \Lambda) \in \bar{\delta}(q, a, \bar{Z})$ if $(p, \Lambda) \in \delta(q, a, Z)$ and $p \in F$.
Otherwise $\bar{\delta}(q, a, \bar{Z}) = \emptyset$.

The proof of the following proposition, together with the proofs of all other propositions involved in this section, can be found in Harrison and Havel [4].

Proposition 2.1. Let M be a PDA, let $\overline{M}$ be the PDA constructed in Definition 2.4. Then $L(M) = L(\overline{M})$.

The last principal construction we need transforms PDA's with a single final state into grammars. It represents the conventional "triple construction" commonly used in the proofs of the equivalence of pushdown automata and context-free grammars.

Definition 2.5. Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \{q_f\})$ be a PDA. We define the *canonical grammar* G_M for M as follows[2]). $G_M = (V, \Sigma, P, S)$ where $V = (Q \times \Gamma \times Q) \cup \Sigma$, $S = q_0 Z_0 q_f$ and P is described below. For any $a \in \Sigma_\Lambda$, $Z, Z_1, \ldots, Z_k \in \Gamma$, $p, q, q_1, \ldots, q_k \in Q$ and $k \geqq 1$

$$qZq_k \to a\, pZ_1q_1\, q_1Z_2q_2 \ldots q_{k-1}Z_kq_k \quad \text{is in } P \text{ if } \quad (p, Z_k \ldots Z_2Z_1) \in \delta(q, a, Z)$$

$$qZp \to a \quad \text{is in } P \text{ if } \quad (p, \Lambda) \in \delta(q, a, Z)$$

No other productions are in P.

Proposition 2.2. Let M be a PDA with a single final state, let G_M be the canonical grammar for M. Then $L(G_M) = L(M)$.

[2]) For the sake of simplicity we write ordered triples as sequences of symbols instead of the usual parenthetical notation.

In addition, it turns out that for deterministic machines the following result holds true.

Proposition 2.3. Let M be a DPDA with a single final state, let G_M be the canonical grammar for M. Then G_M is strict deterministic.

3. THE CONCEPT OF UNIFORMITY

Stack uniform deterministic pushdown automata have been introduced by Linna [7]. This class of machines forms a subclass of real-time DPDA's satisfying the condition that for any input symbol a all the moves of the automaton reading a have the same effect on the length of the pushdown store. The concept in question can be defined for nondeterministic devices as well.

Definition 3.1. Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA. For any word $\gamma \in \Gamma^*$ let us denote by $\lg(\gamma)$ the length of γ. The PDA M is said to be *stack uniform* iff M is a real-time PDA such that for any $p, p', q, q' \in Q$, $Z, Z' \in \Gamma$, $a \in \Sigma$, $\gamma, \gamma' \in \Gamma^*$,

$$(p, \gamma) \in \delta(q, a, Z) \quad \text{and} \quad (p', \gamma') \in \delta(q', a, Z') \quad \text{implies} \quad \lg(\gamma) = \lg(\gamma') .$$

Languages strictly accepted by stack uniform DPDA's are of particular importance since the inclusion problem for them has been shown to be decidable (cf. Linna [7] for the proof).

Definition 3.2. A language $L \subseteq \Sigma^*$ is called a *stack uniform strict deterministic language* (U_0 *language*) iff $L = L(M)$ for some stack uniform DPDA M.

Till now, no grammatical characterization has been known either for the class U_0 or for its nondeterministic variant. We next present the fundamental concept of this paper that will be shown to represent such a characterization[3]).

Definition 3.3. Let $G = (V, \Sigma, P, S)$ be a context-free grammar in Greibach Normal Form. The grammar G is called *length uniform* iff for any $A, A' \in N$, $a \in \Sigma$, $\alpha, \alpha' \in N^*$, if $A \to a\alpha$, $A' \to a\alpha'$ are in P then $\lg(\alpha) = \lg(\alpha')$.

In other words, a length uniform grammar is a context-free grammar in GNF such that there is a function h, mapping the terminal alphabet into integers, and satisfying the condition that for any terminal a occurring in the productions of the grammar, the length of the right hand side of any production starting with a is $h(a)$. To illustrate the above definition we now present an example[4]). It will also indicate the complexity of languages generated by length uniform grammars.

[3]) In this definition, we have abandoned the original notation of [10] in favour of a more convenient wording.

[4]) Our example is a grammatical counterpart of the example presented in the proof of Theorem 1(c) of Linna [7].

For any $n \geqq 1$ consider languages

$$L_n = \{a^m b^k c^m b^{k-1} c \mid m \geqq 1 \text{ and } 1 \leqq k \leqq n\} .$$

The language L_n is generated by the grammar $G_n = (V_n, \{a, b, c\}, P_n, S)$ where $V_n = \{S, A_1, \ldots, A_n, B_1, \ldots, B_n, a, b, c\}$ and P_n is formed by productions

$$\begin{aligned} S &\to aA_iB_i , && 1 \leqq i \leqq n , \\ A_i &\to aA_iB_1 , && 1 \leqq i \leqq n , \\ A_i &\to bB_i , && 1 \leqq i \leqq n , \\ B_i &\to bB_{i-1} , && 2 \leqq i \leqq n , \\ B_1 &\to c . \end{aligned}$$

Setting $h(a) = 3$, $h(b) = 2$, and $h(c) = 1$ we see that G_n is length uniform. In addition, using the same technique as in Lemma 4.3 of Harrison and Havel [4], it is possible to prove that any DPDA M such that $L_n = L(M)$ has at least n states.

Let us try to compare the power of stack uniform PDA's and length uniform grammars. First let us investigate the properties of canonical grammars corresponding to the class of automata in question.

Lemma 3.1. Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \{q_f\})$ be a stack uniform PDA, let $G_M = (V, \Sigma, P, S)$ be the canonical grammar for M. Then G_M is length uniform.

Proof. Since M is real-time, it follows immediately that G_M is in GNF. Let us define a homomorphism stack: $N^* \to \Gamma^*$ as follows. For any $p, q \in Q$, $Z \in \Gamma$ let stack $(qZp) = Z$. Clearly for any $\alpha \in N^*$ it holds that $\lg(\text{stack}(\alpha)) = \lg(\alpha)$.

Now assume that for some $A, A' \in N$, $a \in \Sigma$, $\alpha, \alpha' \in N^*$ we have $A \to a\alpha$, $A' \to a\alpha'$ in P. Then there are $p, p', q, q', r, r' \in Q$, $Z, Z' \in \Gamma$ and $\gamma, \gamma' \in \Gamma^*$ such that $A = qZp$, $A' = q'Z'p'$, $(r, \gamma) \in \delta(q, a, Z)$, $(r', \gamma') \in \delta(q', a, Z')$, stack $(\alpha) = \gamma$ and stack $(\alpha') = \gamma'$. Since M is stack uniform, we obtain $\lg(\gamma) = \lg(\gamma')$. Therefore we have

$$\lg(\alpha) = \lg(\text{stack}(\alpha)) = \lg(\gamma) = \lg(\gamma') = \lg(\text{stack}(\alpha')) = \lg(\alpha') . \qquad \square$$

The preceding lemma has indicated that length uniform grammars are powerfull enough to generate all languages strictly accepted by stack uniform PDA's.

Theorem 3.1. Let $L \subseteq \Sigma^*$ be a language. Then

(a) If $L = L(M)$ for some stack uniform PDA M then $L = L(G)$ for some length uniform grammar G,

(b) If $L = L(M)$ for some stack uniform DPDA M then $L = L(G)$ for some length uniform strict deterministic grammar G.

Proof. The proof directly parallels the technique developed in Harrison and Havel [4]. Assume $L = L(M)$ for some stack uniform PDA M, let $\overline{M}$ be the PDA

constructed in Definition 2.4. Since M is stack uniform, the same is true for $\overline{M}$ because the construction preserves both the real-time property and the length of the words placed on the pushdown store. Let $G_{\overline{M}}$ be the canonical grammar for $\overline{M}$. Then by Propositions 2.1 and 2.2 $L = L(\overline{M}) = L(G_{\overline{M}})$, and Lemma 3.1 implies that $G_{\overline{M}}$ is length uniform. Finally, to prove (b) it suffices to realize that $\overline{M}$ is deterministic if M is and to use Proposition 2.3. □

At this point we should be able to prove also the reverse implication to Theorem 3.1(a) by the help of the conventional construction relating GNF grammars to single state real-time nondeterministic pushdown automata. We shall obtain this result in Section 5 using a less straightforward but rather surprising argument.

We first study the deterministic version of that assertion. As we shall see the task is a little bit more difficult in this case.

4. THE CANONICAL AUTOMATON

We now wish to show that the notion of a length uniform strict deterministic grammar yields the required characterization of U_0 languages. Thus we need, for any such grammar, to construct a stack uniform DPDA strictly accepting the language generated by the grammar under consideration. Our approach follows a more general construction valid for so called "uniform" grammars (cf. [10]).

Definition 4.1. Let $G = (V, \Sigma, P, S)$ be a length uniform grammar, π a strict partition of V. Let $G' = (V', \Sigma, P', S')$ be a context-free grammar such that S' is a new symbol not included in V, $V' = V \cup \{S'\}$, $P' = P \cup \{S' \to S\}$. We define $\pi' = \pi \cup \cup \{\{S'\}\}$.

The set of *canonical states* for G is the set[5])

$$Q_G = \{[X, \alpha] \in \pi' \times V^* \mid A \to \alpha\beta \in P' \text{ for some } A \in X \text{ and } \beta \in V^*\} .$$

For any canonical state q we define a partial function $f_q : Q_G \to Q_G$ as follows. Let $[X, \alpha], [Y, \beta] \in Q_G$. Then

$$f_{[X,\alpha]}([Y, \beta]) = [X, \alpha A] \quad \text{if} \quad A \to \beta \in P \quad \text{and} \quad [X, \alpha A] \in Q_G \quad \text{for some} \quad A \in Y.$$

Otherwise $f_{[X,\alpha]}$ is undefined[6]).

Now we define the *canonical stack uniform DPDA* M_G for G as follows. $M_G = (Q_G, \Sigma, \Gamma_G, \delta, q_0, Z_0, \{q_f\})$ where Γ_G is the smallest set of partial functions closed under composition and containing the identity function on Q_G and functions f_q

[5]) Square brackets are used instead of parentheses to clearly distinguish canonical states from other ordered pairs.

[6]) Cf. the proof of Theorem 2.2 in Harrison and Havel [5] for the motives leading to the introduction of these functions.

for all $q \in Q_G$ (we denote by $\circ$ *composition of functions* and by id the *identity function*), $q_0 = [\{S'\}, \Lambda]$, $Z_0 = \mathrm{id}$, $q_f = [\{S'\}, S]$ and the function δ is defined below.

Let $q \in Q_G$, $a \in \Sigma$, $f \in \Gamma_G$. Suppose that $q = [X, \alpha]$ and there are $A \in N$, $\beta \in N^*$ such that $[X, \alpha A] \in Q_G$ and $A \to a\beta$ is in P. Denote $k = \lg(\beta)$, $\bar{A} = \{B \mid B \equiv A \pmod{\pi}\}$. Then

$$(f \circ f_q([\bar{A}, a]), \Lambda) \in \delta(q, a, f) \quad \text{if} \quad k = 0\,,$$

$$([\bar{A}, a], f \circ f_q \,\mathrm{id}^{k-1}) \in \delta(q, a, f) \quad \text{if} \quad k \geqq 1$$

where id^{k-1} denotes the string of symbols for the identity function of length $k - 1$. Otherwise $\delta(q, a, f) = \emptyset$.

First we have to verify that our definition is correct.

Lemma 4.1. For any length uniform strict deterministic grammar $G = (V, \Sigma, P, S)$ the object M_G described in the previous definition is a well defined stack uniform DPDA.

Proof. Let π be a strict partition of V. Since P and π are finite sets, Q_G and Γ_G are finite, too[7]) $(|\Gamma_G| \leqq (|Q_G| + 1)^{|Q_G|})$. However, it is necessary to check that the functions f_q are well defined. Clearly π' is a strict partition of V'. If two productions within the same block of a strict partition agree on the right hand side, their left hand sides are equal. Therefore the values of f_q are uniquely determined.

Now we show that M_G is deterministic. By the definition, M_G is real-time. Hence it suffices to verify that for any $q \in Q_G$, $a \in \Sigma$, $f \in \Gamma_G$ the set $\delta(q, a, f)$ contains at most one element. Let $q = [X, \alpha]$, and assume that there are $A, A' \in N$, $\beta, \beta' \in N^*$ such that $[X, \alpha A], [X, \alpha A'] \in Q_G$ and $A \to a\beta$, $A' \to a\beta'$ are in P. Then for some $B, B' \in X$, $\gamma, \gamma' \in N^*$ we have $B \to \alpha A\gamma$, $B' \to \alpha A'\gamma'$ in P'. From the strictness of π' it follows that $A \equiv A' \pmod{\pi'}$. Since $A, A' \in N$, by the definition of π' we have $A \equiv A' \pmod{\pi}$, i.e. $\bar{A} = \bar{A}'$. Finally, using the length uniformness of G we obtain $\lg(\beta) = \lg(\beta')$. Thus the moves of M_G based on $A \to a\beta$ and $A' \to a\beta'$ are identical.

It remains to show that M_G is stack uniform. Let $p, q \in Q_G$, $a \in \Sigma$, $f \in \Gamma_G$, $\gamma \in \Gamma_G^*$, and suppose that $(p, \gamma) \in \delta(q, a, f)$. Then there are $A \in N$, $\beta \in N^*$ such that $A \to a\beta$ is in P and we have

$$\gamma = \Lambda \quad \text{if} \quad \lg(\beta) = 0\,,$$

$$\gamma = f \circ f_q \,\mathrm{id}^{\lg(\beta)-1} \quad \text{if} \quad \lg(\beta) \geqq 1\,.$$

We conclude that $\lg(\gamma) = \lg(\beta)$. Now consider another move of M_G reading a and placing a word γ' on the pushdown store. From the above consideration it follows that there are $A' \in N$, $\beta' \in N^*$ such that $A' \to a\beta'$ is in P and $\lg(\gamma') = \lg(\beta')$. Since G is length uniform, we get $\lg(\beta) = \lg(\beta')$ i.e. $\lg(\gamma) = \lg(\gamma')$. □

[7]) Recall our convention that the cardinality of a set X is denoted by $|X|$.

The next part of this section is devoted to the proof of the fact that the canonical automaton strictly accepts the language generated by the underlying grammar. Although this assertion might seem to be intuitively obvious, its formal proof is somewhat complicated. For this reason we divide it into several steps.

Lemma 4.2. Let $G = (V, \Sigma, P, S)$ be a length uniform strict deterministic grammar, $A \in N$, $w \in \Sigma^*$, $f \in \Gamma_G$, $\gamma \in \Gamma_G^*$, $[X, \alpha]$, $[X, \alpha A]$, $f([X, \alpha A]) \in Q_G$. Then

$$A \Rightarrow^* w \quad \text{implies} \quad ([X, \alpha], \gamma f) \vdash_{M_G}^{w} (f([X, \alpha A]), \gamma) .$$

Proof. Let $A \Rightarrow^* w$. Then $A \Rightarrow a\beta \Rightarrow^{n-1} w$ for some $n \geqq 1$, $a \in \Sigma$, $\beta \in N^*$. The argument is an induction on n.

Basis. $n = 1$. Then $w = a$, $\beta = \Lambda$ i.e. $\lg(\beta) = 0$, and by the definition of M_G we have

$$([X, \alpha], \gamma f) \vdash^{a} (f \circ f_{[X,\alpha]}([\bar{A}, a]), \gamma) .$$

Since $A \to a$ is in P, we get $f_{[X,\alpha]}([\bar{A}, a]) = [X, \alpha A]$ which completes the proof of the basis.

Induction Step. $n > 1$. Assume that the assertion of the lemma is true for all derivations shorter than n. Let $k = \lg(\beta)$. Since $n > 1$, we have $k \geqq 1$, and there are $B_j \in N$, $1 \leqq j \leqq k$, such that $\beta = B_1 \dots B_k$. Thus we have $w = aw_1 \dots w_k$ and $B_j \Rightarrow^{n_j} w_j$ for some $w_j \in \Sigma^*$, $n_j < n$, $1 \geqq j \leqq k$. Define $q_j = [\bar{A}, aB_1 \dots B_j]$ for $0 \leqq j \leqq k$, and consider the configurations

$$c_j = (q_j, \gamma\, f \circ f_{[X,\alpha]}\, \mathrm{id}^{k-1-j}), \quad 0 \leqq j \leqq k - 1 .$$

As an immediate consequence of the definitions we obtain

$$([X, \alpha], \gamma f) \vdash^{a} c_0 \tag{1}$$

Now let $1 \leqq j \leqq k - 1$. Since $\mathrm{id}(q_{j-1}) \in Q_G$ and $B_j \Rightarrow^{n_j} w_j$ where $n_j < n$, from our induction hypothesis it follows that

$$c_{j-1} \vdash^{w_j} c_j , \quad 1 \leqq j \leqq k - 1 . \tag{2}$$

Finally, consider the configuration c_{k-1}. Since $A \to aB_1 \dots B_k$ is in P, we have

$$f \circ f_{[X,\alpha]}([\bar{A}, aB_1 \dots B_k]) = f \circ f_{[X,\alpha]}(q_k) = f([X, \alpha A]) \in Q_G .$$

Due to $B_k \Rightarrow^{n_k} w_k$, $n_k < n$, the induction hypothesis implies

$$c_{k-1} = (q_{k-1}, \gamma\, f \circ f_{[X,\alpha]}) \vdash^{w_k} (f \circ f_{[X,\alpha]}(q_k), \gamma) = (f([X, \alpha A]), \gamma) \tag{3}$$

Combining (1), (2), and (3) we obtain the required computation. □

The preceding lemma has related derivations in a grammar to certain computations of the canonical machine. Next we investigate a relationship in the reverse direction.

Lemma 4.3. Let $G = (V, \Sigma, P, S)$ be a length uniform strict deterministic grammar, $q, [X, \alpha] \in Q_G$, $w \in \Sigma^*$, $f \in \Gamma_G$. Then

$$([X, \alpha], f) \vdash_{M_G}^{w} (q, \Lambda) \quad \text{implies} \quad A \Rightarrow^* w \quad \text{and} \quad q = f([X, \alpha A])$$

for some nonterminal $A \in N$.

Proof. We use an induction on $\lg(w)$.

Basis. $\lg(w) = 1$. Then $w = a$ for some $a \in \Sigma$. By the definition, M_G pops the symbol f in one move iff there is $A \in N$ such that $A \to a$ is in P, $[X, \alpha A] \in Q_G$ and $q = f \circ f_{[X,\alpha]}([\bar{A}, a]) = f([X, \alpha A])$.

Induction Step. $\lg(w) > 1$. Then $w = aw'$ for some $a \in \Sigma$, $w' \in \Sigma^+$. Assume that the lemma is true for all computations on words shorter than w. Since $w' \in \Sigma^+$ the first move of M_G must not pop the symbol f from the pushdown store. Consequently there are $B \in N$, $\beta \in N^+$ such that $B \to a\beta$ is in P, $[X, \alpha B] \in Q_G$ and

$$([X, \alpha], f) \vdash^{a} ([\bar{B}, a], f \circ f_{[X,\alpha]} \mathrm{id}^{k-1}) \vdash^{w'} (q, \Lambda)$$

where $k = \lg(\beta) \geqq 1$. For $0 \leqq j \leqq k - 1$ let $c_j = (q_j, \gamma_j)$ be the configurations of M_G during the computation on the word w' such that $\lg(\gamma_j)$ becomes for the first time equal to $k - j$. Then

$$c_j = (q_j, f \circ f_{[X,\alpha]} \mathrm{id}^{k-1-j}), \quad 0 \leqq j \leqq k - 1 .$$

Let $w_j \in \Sigma^*$, $1 \leqq j \leqq k$ such that $w = aw_1 \ldots w_k$,

$$c_{j-1} \vdash^{w_j} c_j , \quad 1 \leqq j \leqq k - 1 , \quad \text{and} \quad c_{k-1} \vdash^{w_k} (q, \Lambda) .$$

By our choice of configurations c_j we have

$$q_0 = [\bar{B}, a] , \quad (q_{j-1}, \mathrm{id}) \vdash^{w_j} (q_j, \Lambda) , \quad 1 \leqq j \leqq k - 1 .$$

Since $\lg(w_j) < \lg(w)$, from the induction hypothesis it follows that there are $B_j \in N$, $1 \leqq j \leqq k - 1$ such that

$$q_j = [\bar{B}, aB_1 \ldots B_j] \quad \text{and} \quad B_j \Rightarrow^* w_j , \quad 1 \leqq j \leqq k - 1 .$$

Now let us consider the computation

$$c_{k-1} = ([\bar{B}, aB_1 \ldots B_{k-1}], f \circ f_{[X,\alpha]}) \vdash^{w_k} (q, \Lambda) .$$

Using the induction hypothesis again, we get

$$q = f \circ f_{[X,\alpha]}([\bar{B}, aB_1 \ldots B_k]) \quad \text{and} \quad B_k \Rightarrow^* w_k$$

for some $B_k \in N$. By the definition of $f_{[X,\alpha]}$ there is a nonterminal $A \in \bar{B}$ such that $A \to aB_1 \ldots B_k$ is in P and $q = f([X, \alpha A])$. Thus we conclude that

$$A \Rightarrow aB_1 \ldots B_k \Rightarrow^* aw_1 \ldots w_k = w . \qquad \square$$

Combining Lemmas 4.2 and 4.3, we are able to prove the desired assertion about the language strictly accepted by the canonical DPDA.

Theorem 4.1. Let $G = (V, \Sigma, P, S)$ be a length uniform strict deterministic grammar, let M_G be the canonical stack uniform DPDA for G. Then $L(M_G) = L(G)$.

Proof. Let $w \in \Sigma^*$. First recall our definitions $q_0 = [\{S'\}, \Lambda]$, $q_f = [\{S'\}, S]$, $Z_0 = \mathrm{id}$. Now assume $S \Rightarrow^* w$. Since $q_0, q_f, \mathrm{id}(q_f)$ are in Q_G, Lemma 4.2 implies $(q_0, \mathrm{id}) \vdash^w (\mathrm{id}(q_f), \Lambda)$ which proves the inclusion $L(G) \subseteq L(M_G)$. On the other hand suppose that $(q_0, \mathrm{id}) \vdash^w (q_f, \Lambda)$. By Lemma 4.3 there is a nonterminal $A \in N$ such that $A \Rightarrow^* w$ and $[\{S'\}, S] = \mathrm{id}([\{S'\}, A])$. Hence $A = S$, and consequently $L(M_G) \subseteq L(G)$. □

Thus we obtain the following characterization theorem for U_0 languages.

Theorem 4.2. Let $L \subseteq \Sigma^*$. L is a stack uniform strict deterministic language iff $L = L(G)$ for some length uniform strict deterministic grammar G.

Proof. Cf. Theorems 3.1(b) and 4.1. □

5. DETERMINISM VERSUS NONDETERMINISM

This final section of the paper is devoted to the relationship between the families of languages strictly accepted by deterministic and nondeterministic stack uniform PDA's respectively. As far as general PDA's are concerned, we know that the power of DPDA's is substantially weaker. Next we show that for stack uniform machines an opposite situation arises, i.e. that the corresponding families of languages coincide.

As we have mentioned in the introduction, the existence of a convenient grammatical characterization may help us to obtain results provable for automata themselves only with considerable difficulties. Indeed, we can use much simpler tools of the grammar theory to derive them. This is exactly the idea we will now follow.

First we need an additional definition.

Definition 5.1. Let $G = (V, \Sigma, P, S)$ be a context-free grammar, π a partition of V. Such a partition is called *invertible* iff for any $A, A' \in N$, $\alpha \in V^*$, if $A \to \alpha$, $A' \to \alpha$ are in P, and $A \equiv A' (\mathrm{mod}\ \pi)$ then $A = A'$.

Clearly any strict partition is invertible. The reverse implication does not hold in general. The key idea of our considerations is based on the following simple argument proving certain invertible partitions to be strict.

Lemma 5.1. Let $G = (V, \Sigma, P, S)$ be a length uniform grammar such that S does not occur in the right hand side of any production in P, let $\pi = \{\{S\}, N - \{S\}, \Sigma\}$ be an invertible partition of V. Then π is strict.

Proof. We shall show that the partition π satisfies the conditions of Definition 2.2. Let $A, A' \in N$, $\alpha, \beta, \beta' \in V^*$, $A \to \alpha\beta$, $A' \to \alpha\beta' \in P$, and $A \equiv A' (\mathrm{mod}\ \pi)$.

If $\alpha = \Lambda$ then both β, β' are in ΣN^* by the GNF property of G. Hence ${}^{(1)}\beta \equiv \equiv {}^{(1)}\beta' \pmod{\pi}$.

Thus we assume $\alpha \neq \Lambda$ in the remainder of the proof. Again, the GNF property implies that there are $a \in \Sigma$, $\gamma \in N^*$ such that $\alpha = a\gamma$ and $A \to a\gamma\beta$, $A' \to a\gamma\beta'$ are in P. Since G is length uniform, we get $\lg(\gamma\beta) = \lg(\gamma\beta')$ i.e. $\lg(\beta) = \lg(\beta')$. Next we distinguish two cases.

Case 1. $\lg(\beta) > 0$. Then both β, β' are in N^+. By our assumption, the symbol S can occur neither in β nor in β'. Therefore we have ${}^{(1)}\beta, {}^{(1)}\beta' \in N - \{S\}$, i.e. ${}^{(1)}\beta \equiv \equiv {}^{(1)}\beta' \pmod{\pi}$.

Case 2. $\lg(\beta) = 0$. Then $\beta = \beta' = \Lambda$. In other words, both $A \to \alpha$, $A' \to \alpha$ are in P. Since $A \equiv A' \pmod{\pi}$ and π is invertible, we conclude that $A = A'$. □

It remains to find a method of converting length uniform grammars to those satisfying the assumptions of Lemma 5.1 without affecting the language generated. Problems related to the notion of invertibility[8]) were first studied by McNaughton [8]. Gray and Harrison [3] further generalized his results to arbitrary context-free grammars. Our proof will utilize a modified version of the method presented in [3].

Theorem 5.1. Let $G = (V, \Sigma, P, S)$ be a length uniform grammar. Then there exists a length uniform strict deterministic grammar G' such that $L(G) = L(G')$.

Proof. Let $G' = (2^N \cup \{S'\} \cup \Sigma, \Sigma, P', S')$ where 2^N is the set of all subsets of N, S' is a new symbol. The set P' will be constructed below.

For any $n \geqq 0$, $A, A_1, \ldots, A_n \in N$, $a \in \Sigma$, if $A \to aA_1 \ldots A_n$ is in P then P' contains all productions $B \to aB_1 \ldots B_n$ such that $B_i \in 2^N$, $A_i \in B_i$, $1 \leqq i \leqq n$, and

$$(4) \qquad B = \{C \in N \mid C \to aC_1 \ldots C_n \in P \text{ and } C_i \in B_i,\ 1 \leqq i \leqq n\}.$$

In addition, if $S \in B$ then P' also contains the production $S' \to aB_1 \ldots B_n$. No other productions are in P'.

Clearly no right hand side of a production in P' includes S'. The grammar G' is length uniform because the construction of P' preserves the positions of terminals and nonterminals in the right hand sides of productions. Due to (4), the partition $\pi = \{\{S'\}, 2^N, \Sigma\}$ is invertible. Hence by Lemma 5.1 the grammar G' is strict deterministic. The fact $L(G) = L(G')$ follows from the two claims given below.

Claim 1. Let $B \in 2^N$, $w \in \Sigma^*$, $B \Rightarrow_{G'}^* w$. Then for all $A \in B$ we have $A \Rightarrow_G^* w$.

Claim 2. Let $A \in N$, $w \in \Sigma^*$, $A \Rightarrow_G^* w$. Then there is some $B \in 2^N$ such that $A \in B$ and $B \Rightarrow_{G'}^* w$.

The proofs of both the claims can be found in Gray and Harrison [3]. □

[8]) A context-free grammar $G = (V, \Sigma, P, S)$ is called *invertible* (or *backwards deterministic*) iff $\{N, \Sigma\}$ is an invertible partition of V.

Now we can summarize our results as follows.

Theorem 5.2. Let $L \subseteq \Sigma^*$. Then the following four statements are equivalent.
(a) $L = L(M)$ for some stack uniform DPDA M,
(b) $L = L(M)$ for some stack uniform PDA M,
(c) $L = L(G)$ for some length uniform grammar G,
(d) $L = L(G)$ for some length uniform strict deterministic grammar G.

Proof. (a) → (b) trivial, (b) → (c) Theorem 3.1(a), (c) → (d) Theorem 5.1, (d) → (a) Theorem 4.2. □

Note. The theorem proves the existence of a class of nondeterministic pushdown automata accepting nonregular languages for which the inclusion problem is decidable.

ACKNOWLEDGEMENT

The author wishes to thank Ing. I. M. Havel, CSc. for numerous valuable comments on the paper. Last but not the least, the author appreciates the support of the Research Institute for Mathematical Machines that enabled him to finish this research.

(Received September 24, 1980.)

REFERENCES

[1] A. V. Aho, J. D. Ullman: The Theory of Parsing, Translation, and Compiling, Vols. I, II. Prentice Hall, Englewood Cliffs, N. J. 1972, 1973.

[2] E. P. Friedman: The inclusion problem for simple languages. Theoretical Computer Science *1* (1976), 297—316.

[3] J. N. Gray, M. A. Harrison: On the covering and reduction problems for context-free grammars. J. ACM *19* (1972), 675—698.

[4] M. A. Harrison, I. M. Havel: Strict deterministic grammars. Journal of Computer and System Sciences *7* (1973), 237—277.

[5] M. A. Harrison, I. M. Havel: Real-time strict deterministic languages. SIAM Journal on Computing *1* (1972), 333—349.

[6] A. J. Korenjak, J. E. Hopcroft: Simple deterministic languages. IEEE Conference Record of the 7th Annual Symposium on Switching and Automata Theory (1966), 36—46.

[7] M. Linna: Two decidability results for deterministic pushdown automata. Journal of Computer and System Sciences *18* (1979), 92—107.

[8] R. McNaughton: Parenthesis grammars. J. ACM *14* (1967), 490—500.

[9] M. Oyamaguchi, N. Honda, Y. Inagaki: The equivalence problem for real-time strict deterministic languages. Information and Control *45* (1980), 90—115.

[10] J. Pittl: On two subclasses of real-time grammars. Mathematical Foundations of Computer Science 1978 (J. Winkowski, ed.). Lecture Notes in Computer Science 64, Springer-Verlag, Berlin 1978, 426—435.

[11] L. G. Valiant: Decision Problems for Families of Deterministic Pushdown Automata. University of Warwick, Computer Centre, Report No. 7, 1973.

RNDr. Jan Pittl, Výzkumný ústav matematických strojů (Research Institute for Mathematical Machines), Loretánské nám. 3, 118 55 Praha 1. Czechoslovakia.