On the Parsing of Deterministic Languages

MICHAEL A. HARRISON AND IVAN M. HAVEL

University of California, Berkeley, California

ABSTRACT. A parsing method for strict deterministic grammars is presented and a technique for using it to parse any deterministic language is indicated. An important characterization of the trees of strict deterministic grammars is established. This is used to prove iteration theorems for (strict) deterministic languages, and hence proving that certain sets are not in these families becomes comparatively straightforward. It is shown that every strict deterministic grammar is LR(0) and that any strict deterministic grammar is equivalent to a bounded right context (1, 0) grammar. Thus rigorous proofs that the families of deterministic, LR(k), and bounded right context languages are coextensive are presented for the first time.

KEY WORDS AND PHRASES: parsing, deterministic language, LR(k), strict deterministic, context-free grammars, pushdown automata

CR CATEGORIES: 4.12, 5.22, 5.23

Introduction

Despite an extensive literature on the parsing of general context-free languages, there have been no improvements beyond algorithms which require time of the order of n^3 where n is the length of the input.[1] Attempts to produce lower bounds greater than n have not been successful so that the real computational complexity of parsing is not known at this time.

One approach has been to restrict attention to the deterministic context-free languages and their subfamilies. This ensures that we are dealing with a linear time parsing algorithm. Attention is then focused on the constants in the linear function, the amount of space required, and the family of languages defined. Examples of such studies include [2–7, 10–13, 17, 19, 21].

In [17] we defined the family of strict deterministic grammars and showed that the strict deterministic languages are exactly the prefix-free deterministic languages. By "endmarking" a language, i.e. having a special endmarker so that one maps any language $L \subseteq \Sigma^*$ into $L\{\$\}$ where $\$ \notin \Sigma$, one can make any deterministic language L into a strict deterministic language $L\$$. In the present paper, we present a parsing method for any strict deterministic grammar. We also study the parse trees of these grammars and give an important characterization of them. Using this characterization we can prove iteration theorems without recourse to automata (cf. [23]).

This research was supported by NSF Grant GJ-474.

Authors' present addresses: M. A. Harrison, Department of Computer Science, University of California, Berkeley, CA 94720; I. M. Havel, ÚTIA-ČSAV, Vyšehradská 49, Praha 2, Czechoslovakia.

[1] Valiant [26] has recently shown that every context-free language may be parsed in time $n^{2.81}$ by reducing parsing to Boolean matrix multiplication. One can also work in the opposite direction to give a lower bound on the parsing time as a funtion of the checking time for Boolean matrix multiplication.

It turns out that every strict deterministic grammar is LR(0) and every strict deterministic language also has a bounded right context grammar with bounds (1, 0). Using these results, we can obtain simple proofs that the families of deterministic, bounded right context, and LR(k) languages are identical.

The present paper is organized as follows. The remainder of the Introduction introduces a semiformal treatment of trees of context-free grammars. In Section I, an interesting parsing method is indicated for strict deterministic grammars. Section II focuses on grammatical trees. An important result concerning the uniqueness of partial subtrees is proven (we call this the Left Part Theorem). The Left Part Theorem is used to prove an iteration theorem for strict deterministic languages and then an iteration theorem for deterministic languages. Section III presents some background on LR(k) and bounded right context grammars and indicates some areas in which there were gaps in earlier work. In Section IV, we prove that every strict deterministic grammar is LR(0). Section V is devoted to showing that every strict deterministic grammar is equivalent to a bounded right context grammar (with bounds (1, 0)). Section VI concludes by proving that the families of deterministic, LR(k), and bounded right context languages are coextensive.

Before it is possible to present our new results, it is necessary to establish some common notational conventions. Although some of the notation and definitions from [17] are reproduced here, many of the theorems and proofs will also be used in this paper. Thus [17] is a prerequisite to understanding our proofs.

It is necessary for us to describe trees more precisely than is customarily done in the literature. Standard graph-theoretic formalism[2] is not sufficient for our purposes since we need to distinguish the left and right directions in our trees. Even the inductive definition of derivation trees in mathematical language theory [8] is inconvenient for our purposes. We could choose another way, and define trees axiomatically as algebraic structures[3] and then prove all, otherwise obvious, results as consequences of these axioms. This would, however, sacrifice the intuitive insight and therefore we shall, as a compromise, give only a semiformal overview of all the concepts which we shall need in our theory, and we shall use pictures.

An example of a *tree* T is in Figure 1. It has one or more *nodes* ($x_0, x_1, \cdots, x_{10}$) one of which is the *root* (x_0). We denote the relation of *immediate descendancy* by Γ (thus in Figure 1(a), $x_2 \ \Gamma \ x_4$, read x_4 is an immediate descendant of x_2, but not $x_2 \ \Gamma \ x_{10}$). The *descendancy* relation is the reflexive and transitive closure Γ^* of Γ (now $x_2 \ \Gamma^* \ x_{10}$). If $x \ \Gamma^* \ y$ then the *path from x to y* (or *path to y* if x is the root) is the sequence of all nodes between and including x and y (e.g. x_0, x_2, x_4, x_8 is the path from x_0 to x_8 in Figure 1(a). If n is the number of nodes in one of the longest paths in T then the *height of T* is, by definition, $n - 1$ (the tree in Figure 1(a) has height 4). A node x is a *leaf* in T iff for no y in T, $x \ \Gamma \ y$. Nodes which are not leaves are called *internal*.

As mentioned above we pay attention to the left-right order of nodes. Let

$$y_1, y_2, \cdots, y_m, \quad m \geq 1, \tag{1}$$

be a sequence of all the leaves in T without repetition (there is no other restriction on this sequence except the informal assumption that sequence (1) represents the intuitive left-to-right order of leaves). Now we define a special relation $\llcorner$ between certain pairs of nodes in T.

For any x, y in T, $x \ \llcorner \ y$ iff (i) x and y are not on the same path, and (ii) for some leaves y_i, y_{i+1} ($1 \leq i < m$) in (1) we have $x \ \Gamma^* \ y_i$ and $y \ \Gamma^* \ y_{i+1}$. Thus, in particular, there is no leaf "between" x and y. (Thus, for instance, we have $x_2 \ \llcorner \ x_6$ but not $x_9 \ \llcorner \ x_6$

[2] The reader may consult [20, pp. 362–406] for an introduction to the theory of trees.

[3] Formally, we can define a "tree structure" over a set T as $\langle T, \Gamma^*, \llcorner^*, x_0 \rangle$ by means of the following axioms: (i) Γ^*, $\llcorner^*$ are two partial orders on T; (ii) $\Gamma^* \cup \llcorner^* \cup (\Gamma^* \cup \llcorner^*)^{-1} = T \times T$; (iii) $\llcorner^+ = (\Gamma^*)^{-1} \llcorner^+ \Gamma^*$; (iv) $x_0 \in T$, and for any $y \in T$, $x_0 \ \Gamma^* \ y$. In (iii) we used the notation $\llcorner^+ = \llcorner^* - \{(x, x) \mid x \in T\}$.

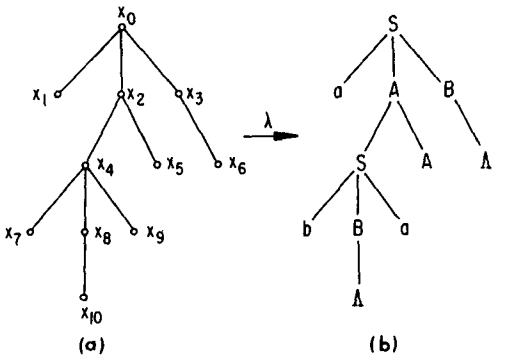

FIG. 1. A tree and its labeling

in Figure 1(a).) Note that relation $\llcorner$ is determined uniquely by (1). The *left-to-right order* is then the reflexive and transitive closure $\ulcorner^*$ of $\llcorner$. Note that we have $\ulcorner^+ \cap \llcorner^+ = \varnothing$. On the other hand, one of the two relations $\llcorner^*$ or $\ulcorner^*$ (or their inverses) holds between any two nodes of a tree (see Footnote 3, p. 526).

Every node $x \in T$ has a *label* $\lambda(x)$ from a given finite set of labels—in our cases it is always the set $V_\Lambda = V \cup \{\Lambda\}$ where V is the alphabet of a given grammar. The corresponding function $\lambda : T \to V_\Lambda : x \to \lambda(x)$ is the labeling (thus λ: Figure 1(a) $\to$ Figure 1(b)). Of special importance are the *root label of* T, $\mathbf{rt}(T)$, and the *frontier of* T, $\mathbf{fr}(T)$. The latter is defined as a concatenation of labels of leaves in (1): $\mathbf{fr}(T) = \lambda(y_1)\lambda(y_2)\cdots\lambda(y_m) \in V^*$. Note that some leaves may be labeled by Λ and thus $\mathbf{fr}(T)$ may be shorter than sequence (1) (for instance on Figure 1, $\mathbf{fr}(T) = abaA$).

For any internal node x in T the set $\{y \mid y = x \text{ or } x \ulcorner y\}$ defines an *elementary subtree of* T in an obvious way. A *cross section in* T is any maximal sequence $\xi = (x_1, x_2, \cdots, x_n)$ of nodes in T such that $x_1 \llcorner x_2 \llcorner \cdots \llcorner x_n$ $(n \geq 1)$. Thus, in particular, sequence (1) is a cross section in T.

Two trees T, T' are *structurally isomorphic*, $T \cong T'$, iff there is a bijection $T \to T' : x \to x'$, called *structural isomorphism from* T *to* T', such that $x \ulcorner y$ $(x \llcorner y)$ iff $x' \ulcorner y'$ $(x' \llcorner y')$ (intuitively, T and T' are "identical except for the labeling"). Where there is no danger of confusion, we write $T = T'$ when $T \cong T'$ and the labeling is preserved (this is in agreement with customary understanding that T and T' are "identical"— but not with the algebraic definition of trees). Also we use the same symbols $\ulcorner$, $\llcorner$, λ even for different trees.

We now define certain trees and sets of trees related to grammars. Let

$$G = \langle V, \Sigma, P, S\rangle \tag{2}$$

be a context-free grammar. We interpret the productions of G as trees (of height 1) in a natural way: the production $(A_0 \to A_1A_2\cdots A_n)$ corresponds to the tree shown in Figure 2 (or, formally, $x_0 \ulcorner x_1$, $x_0 \ulcorner x_2, \cdots, x_0 \ulcorner x_n$, $x_1 \llcorner x_2 \llcorner \cdots \llcorner x_n$ and $\lambda(x_i) = A_i$, $0 \leq i \leq n$). We make a convention that in this correspondence $\lambda(x_i) = \Lambda$ iff $i = n = 1$ and $A_0 \to \Lambda$ is in P.

Definition 0.1. Let G be a grammar of the form (2) and let T be a tree with labeling $\lambda : T \to V_\Lambda$. T is said to be a *grammatical tree of* G iff (i) for every elementary subtree T' of T there exists a production $p \in P$ corresponding to T', and (ii) $\mathbf{fr}(T) \in \Sigma^*$.

Consequently, the leaves in T are precisely the nodes labeled by letters in Σ (called the *terminal nodes*) or by Λ (the Λ-*nodes*). All other nodes are labeled by letters in $N = V - \Sigma$ (and are accordingly called the *nonterminal nodes*). Note that in particular, a trivial tree consisting of a single node labeled by $a \in \Sigma$ or by Λ is a grammatical tree of G.

We denote by $\mathfrak{I}_G$ the set of all grammatical trees of G and for a given $A \in V_\Lambda$ we define

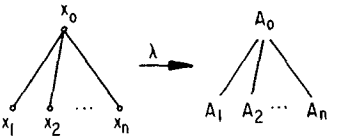

FIG. 2. An elementary subtree

$\mathfrak{I}_G(A) = \{T \in \mathfrak{I}_G \mid \mathbf{rt}(T) = A\}$. In particular, we call trees in $\mathfrak{I}_G(S)$ the *derivation trees of* G.

Remark. We can immediately observe a relationship between grammatical trees of a grammar G and the relation $\Rightarrow$ as defined on V^*. Thus for any $A \in V$ and $w \in \Sigma^*$, $A \Rightarrow^* w$ iff there is a tree $T \in \mathfrak{I}_G(A)$ such that $\mathbf{fr}(T) = w$. Moreover, if $\alpha \in V^*$, $A \Rightarrow {}^*\alpha \Rightarrow^* w$ iff there exists a tree $T \in \mathfrak{I}_G(A)$ such that $\mathbf{fr}(T) = w$ and $\alpha = \lambda(\xi)$ for some cross section ξ in T. This natural extension of the correspondence shown in Figure 2 to a correspondence between derivations in G on the one hand, and grammatical trees of G on the other hand, will often be used in the sequel.

Definition 0.2. A grammar G of the form (2) is said to be *unambiguous* iff for any T, $T' \in \mathfrak{I}_G(S)$, $\mathbf{fr}(T) = \mathbf{fr}(T')$ implies $T = T'$.

Now we recall the special type of grammars studied in [17].

Definition 0.3. Let G be a grammar of the form (2) and let π be a partition of the set V of terminal and nonterminal letters of G. Such a partition π is called *strict* iff (1) $\Sigma \in \pi$, and (2) for any A, $A' \in N$, and $\alpha, \beta, \beta' \in V^*$ if $A \to \alpha\beta$, $A' \to \alpha\beta'$, and $A \equiv A'$ (mod π), then either (i) both $\beta, \beta' \neq \Lambda$ and[4] ${}^{(1)}\beta \equiv {}^{(1)}\beta$ (mod π), or (ii) $\beta = \beta' = \Lambda$ and $A = A'$.

In most cases, the partition π will be clear from the context and we shall write simply $A \equiv B$ instead of $A \equiv B$ (mod π), and $[A]$ instead of $[A]_\pi = \{A' \in V \mid A' \equiv A \pmod{\pi}\}$.

Definition 0.4. Any grammar G of the form (2) is called *strict deterministic* iff there exists a strict partition π of V. A language L is called a *strict deterministic language* iff $L = L(G)$ for some strict deterministic grammar G.

As an example, let us consider the strict deterministic grammar G_2 taken from [17].

$$
\begin{aligned}
S &\to (E \\
E &\to T_1E \mid T_2 \\
T_1 &\to F_1T_1 \mid F_2 \\
T_2 &\to F_1T_2 \mid F_3 \\
F_1 &\to (E+ \mid a+ \\
F_2 &\to (E* \mid a* \\
F_3 &\to (E) \mid a)
\end{aligned}
$$

The blocks of a strict partition for G_2 are $\Sigma, \{S\}, \{E\}, \{T_1, T_2\}, \{F_1, F_2, F_3\}$.

I. *Parsing of Strict Deterministic Grammars*

Because of the special structure of strict deterministic grammars, very efficient parsing algorithms are possible. A "canonical" parsing algorithm will now be presented in an informal manner. A completely mathematical formulation implicitly appears in [17] (cf. [17, Def. 3.6, Lem. 3.4, Lem. 3.5.]). Our purpose in presenting the parsing method here is to illustrate the simplicity of the procedure. In view of [17], our presentation is intuitive and we shall not prove that the algorithm works. After additional results have been obtained, we will assess the practicality of this procedure.

Let $G = \langle V, \Sigma, P, S\rangle$ be a grammar with strict partition $\pi = \{\Sigma, V_1, V_2, \cdots, V_n\}$. One of the blocks of π, say V_1, contains S. We introduce a new set of letters

[4] For any string w and $n \geq 0$, ${}^{(n)}w$ is the prefix of w of length $\min\{\lg(w), n\}$. $w^{(n)}$ has the corresponding meaning for the suffix. Cf. [17].

$W = \{\bar{\Sigma}, \bar{V}_1, \cdots, \bar{V}_n, \perp\}$. Grammar G can be characterized by a table (or function) f which assigns to each pair $(\bar{V}_i, \alpha)$ such that $A \to \alpha\beta$ for some $A \in V_i$ and $\alpha, \beta \in V^*$, a value $f(\bar{V}_i, \alpha) \in W \cup V$ as follows:

$$f(\bar{V}_i, \alpha) = \begin{cases} \bar{V}_j \text{ (or } \bar{\Sigma}) & \text{if } \beta = B\beta' \text{ and } B \in V_j \text{ (or } B \in \Sigma); \\ A & \text{if } \beta = \Lambda. \end{cases}$$

Also define $f(\perp, S) = \bar{\Sigma}$ and $f(\perp, S\perp) =$ **accept**. (The singlevaluedness of f follows from the fact that π is a strict partition.)

The input to the algorithm has the form $w\perp$ where $w \in \Sigma^*$, and is scanned from the left to the right; the next symbol is read whenever requested (step 2). The algorithm always terminates and either accepts if $w \in L(G)$ (step 4) or rejects if $w \notin L(G)$ (step 5). In the case where $w \in L(G)$, the algorithm produces the sequence of productions used in $S \Rightarrow_R^* w$ but in the reverse order. This is a "canonical parse" of w.

Algorithm 1. The algorithm uses a pushdown store which initially contains only the pair $\perp \bar{V}_1$ (we assume that the top of the store is on the right; recall that $S \in V_1$).

At each subsequent stage of the algorithm the following is done. Let γ be a suffix (topmost substring) of the pushdown store such that $\gamma \in WV^*$. (The preceding steps have guaranteed that such a suffix always exists and its length never exceeds by more than one symbol the maximal length of a right-hand side of the productions of G.) Exactly one of the following five steps is executed. If one of the first three steps is executed, the whole process repeats.

Step 1. If $\gamma = \bar{V}_i\alpha$ and $f(\bar{V}_i, \alpha) = \bar{V}_j$, then $\bar{V}_j$ is entered on top of the pushdown store (i.e. γ is replaced by $\gamma\bar{V}_j$).

Step 2. If $\gamma = \bar{V}_i\alpha$ and $f(\bar{V}_i, \alpha) = \bar{\Sigma}$, then the next (initially the first) symbol of the input is transferred to the top of the pushdown store.

Step 3. If $\gamma = \bar{V}_i\alpha$ and $f(\bar{V}_i, \alpha) = A \in V$, then γ is replaced by A. Production $A \to \alpha$ is given as the output.

Step 4. If $\gamma = \perp S$ and the next input symbol is $\perp$, then the procedure halts. The input string is in $L(G)$.

Step 5. In all other cases an error is declared and the procedure halts. The input string is not in $L(G)$.

A flow diagram of the algorithm is given in Figure 3.

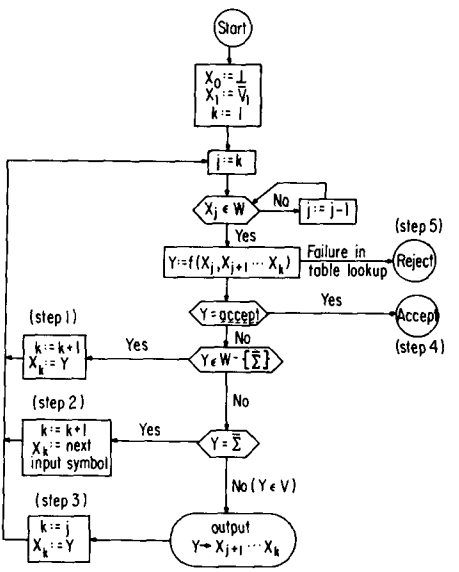

FIG. 3. Flow diagram of Algorithm 1. (Here array X represents the pushdown list, and X_k is its top.)

Remarks. (1) It is customary to distinguish between *detection phases* and *reduction phases* in parsing algorithms (cf. [12, 13]). Here a detection phase consists of a sequence of stages in which steps 1 or 2 are executed, except the last one corresponding to step 3. Step 3 itself has the role of a reduction phase too (note that the detection sequence need not be contiguous).

(2) The overall strategy of the above parsing procedure cannot be classified either as bottom-up or as top-down in the ordinary meaning of these terms (cf. [14]). In fact, it has some features of both classes of procedures. Analyzing how a derivation tree is traversed, we can specify a *working set* as a set of nodes that are currently under processing. The order in which nodes *enter* the working set is *preorder* [20], which is a top-down feature; on the other hand the nodes *exit* from the working set in *endorder*, which is a bottom-up feature.

(3) The algorithm is essentially a shift-reduce parsing algorithm in the sense of Aho and Ullman [1, 2] although their definition requires minor modifications as they do in the LR case [1]. Step 2 is a shift move while step 3 is a reduce move. The step 1 computations can be precomputed.

(4) The reader familiar with LR parsing will note a strong resemblance between this algorithm and the computations of LR tables. Also cf. Theorem 4.1. In [6], strict deterministic and LR(0) parsing are compared. Strict deterministic parsers can be optimized and it can be shown that they are as fast as LR(0) parsers and much smaller. We do not make this claim for the speed of the present algorithm which is a "canonical" one. Cf. [1] for the sense in which canonical is used.

(5) All information about the grammar needed for the parsing algorithm is included in the table of f (Table I). The table itself is relatively simple (never having more than $1 + |P| \cdot (m + 1)$ entries, where m is the maximal length of a right-hand side of a production in P). Moreover, it can be obtained immediately from P if π is known (or by a straightforward method, derived from Algorithm 1 of [17] if π is not known). Therefore even a parser-constructing algorithm would be relatively simple.

We now apply our parsing algorithm to a nontrivial example.

Example. Consider grammar G_2 from the Introduction. $\pi = \{\Sigma, V_S, V_E, V_T, V_F\}$ where $V_S = \{S\}$, $V_E = \{E\}$, $V_T = \{T_1, T_2\}$, $V_F = \{F_1, F_2, F_3\}$. The table of f for G_2 is shown in Table I.

An example of the computation with input string $(a*(a+a))$ is shown in Table II.

It should be noted that this parser is both small and simple. Since we know that by

TABLE I

	$\bar{V}_i$	α	$f(V_i, \alpha)$
1		Λ	$\bar{\Sigma}$
2	$\bar{V}_S$	(	$\bar{V}_E$
3		$(E$	S
4		Λ	$\bar{V}_T$
5		T_1	$\bar{V}_E$
	$\bar{V}_E$		
6		T_1E	E
7		T_2	E
8		Λ	$\bar{V}_F$
9		F_1	$\bar{V}_T$
10		F_1T_1	T_1
	$\bar{V}_T$		
11		F_1T_2	T_2
12		F_2	T_1
13		F_3	T_2
14		Λ	$\bar{\Sigma}$
15		(	$\bar{V}_E$
16		$(E$	$\bar{\Sigma}$
17		$(E+$	F_1
18		$(E*$	F_2
	$\bar{V}_F$		
19		(E)	F_3
20		a	$\bar{\Sigma}$
21		$a+$	F_1
22		$a*$	F_2
23		$a)$	F_3
24	$\perp$	S	$\bar{\Sigma}$
25	$\perp$	$S\perp$	**accept**

TABLE II

Entries used in table of f	History of pushdown list	Input	Output
		$(a*(a+a))\perp$	
	$\perp\bar{V}_S$	(	
1	$\perp\bar{V}_S($		
2	$\perp\bar{V}_S(\bar{V}_E$		
4	$\perp\bar{V}_S(\bar{V}_E\bar{V}_T$		
8	$\perp\bar{V}_S(\bar{V}_E\bar{V}_T\bar{V}_F$	a	
14	$\perp\bar{V}_S(\bar{V}_E\bar{V}_T\bar{V}_Fa$	$*$	
20	$\perp\cdots\bar{V}_T\bar{V}_Fa*$		
22	$\perp\cdots\bar{V}_E\bar{V}_TF_2$		$F_2 \to a*$
12	$\perp\cdots\bar{V}_ET_1$		$T_1 \to F_2$
5,4,8	$\perp\cdots\bar{V}_ET_1\bar{V}_E\bar{V}_T\bar{V}_F$	(	
14	$\perp\cdots\cdots\bar{V}_F($		
15	$\perp\cdots\cdots\bar{V}_F(\bar{V}_E$		
4,8	$\perp\cdots\cdots\bar{V}_F(\bar{V}_E\bar{V}_T\bar{V}_F$	a	
14	$\perp\cdots\cdots\cdots\bar{V}_T\bar{V}_Fa$	$+$	
20	$\perp\cdots\cdots\cdots\bar{V}_T\bar{V}_Fa+$		
21	$\perp\cdots\cdots\cdots\bar{V}_TF_1$		$F_1 \to a+$
9,8	$\perp\cdots\cdots\cdots\bar{V}_TF_1\bar{V}_T\bar{V}_F$	a	
14	$\perp\cdots\cdots\cdots\cdots\bar{V}_T\bar{V}_Fa$	)	
20	$\perp\cdots\cdots\cdots\cdots\bar{V}_T\bar{V}_Fa)$		
23	$\perp\cdots\cdots\cdots\bar{V}_TF_1\bar{V}_TF_3$		$F_3 \to a)$
13	$\perp\cdots\cdots\bar{V}_E\bar{V}_TF_1T_2$		$T_2 \to F_3$
11	$\perp\cdots\cdots\bar{V}_F(\bar{V}_ET_2$		$T_2 \to F_1T_2$
7	$\perp\cdots\cdots\bar{V}_T\bar{V}_F(E$	)	$E \to T_2$
16	$\perp\cdots\cdots\bar{V}_T\bar{V}_F(E)$		
19	$\perp\cdots\bar{V}_E\bar{V}_TF_3$		$F_3 \to (E)$
13	$\perp\cdots\bar{V}_ET_1\bar{V}_ET_2$		$T_2 \to F_3$
7	$\perp\bar{V}_S(\bar{V}_ET_1E$		$E \to T_2$
6	$\perp\bar{V}_S(E$		$E \to T_1E$
3	$\perp S$	$\perp$	$S \to (E$
24	$\perp S\perp$		
25	**accept**		

endmarking, i.e. mapping $L \subseteq \Sigma^*$ into $L\$$ where $\$ \notin \Sigma$, any deterministic language can be converted to a strict deterministic language [17], we can use this simple parsing method on the full class of deterministic languages. There is a difficulty in this approach, namely constructing an appropriate strict deterministic grammar.

Much of the rest of this paper will be devoted to characterizing parse trees of strict deterministic grammars and to relating the class of grammars to LR(k) and bounded right context grammars.

II. *Structure of Grammatical Trees*

In this section, the structural properties of derivation trees (also called parse trees) will be investigated. In addition to the property of unambiguity which, by definition, requires that every terminal string generated by the grammar has a unique derivation tree, the strict deterministic grammars have a unique "partial" tree for every prefix of a terminal string. We utilize more general objects than derivation trees, namely the grammatical trees (with roots labeled by any letter) in order to facilitate induction proofs. The uniqueness of the "partial" trees is the content of the Left Part Theorem. This theorem will be also used to give a new proof of a deterministic version of an Iteration Theorem [23].

We are now ready to define the left part of a grammatical tree. The reader should consult the Introduction for the formalism which is required in dealing with trees.

Definition 2.1. Let T be a grammatical tree of some grammar G. For any $n \geq 0$ we

define ${}^{(n)}T$, the *left n-part of* T (or the *left part* where n is understood) as follows. Let $(x_1, \cdots, x_m)$ be the sequence of all terminal[5] nodes in T (from the left to the right), i.e. $\{x_1, \cdots, x_m\} = \lambda^{-1}(\Sigma)$ and $x_1 \mathrel{\lrcorner}^+ x_2 \mathrel{\lrcorner}^+ \cdots \mathrel{\lrcorner}^+ x_m$. Then ${}^{(n)}T = \{x \text{ in } T \mid x \mathrel{\lrcorner}^* \Gamma^* x_n\}$ if $n \leq m$, and ${}^{(n)}T = T$ if $n > m$. We consider ${}^{(n)}T$ to be a tree under the same relations Γ, $\lrcorner$ and labeling λ as T.

Thus, for instance, the shaded area in Figure 4 (including the path to x) is the left n-part of T if x is the nth terminal node in T. Note that, in general, ${}^{(n)}T$ may not be a grammatical tree.

As immediate consequences of Definition 2.1 we have

Fact 2.1. ${}^{(n)}T = {}^{(n+1)}T$ iff ${}^{(n)}T = T$.

Fact 2.2. If ${}^{(n)}T = {}^{(n)}T'$ then ${}^{(j)}T = {}^{(j)}T'$ for any $j \leq n$.

Our principal result about the structure of grammatical trees of a strict deterministic grammar can be informally stated as follows. A reduced grammar G has a strict partition π iff, given any $V \in \pi$, then each prefix u of any string $w = uv \in \Sigma^*$ generated by some symbol $A \in V_i$ *uniquely* determines the left partial subtree of tree T (cf. Figure 4), where T corresponds to a derivation $A \Rightarrow^* w$ up to the path to the terminal node (x) labeled by the first letter of v. We allow that the labels of nodes on this particular path are determined modulo the partition π (in particular, $[\lambda(x)] = \Sigma$). In the case when $v = \Lambda$, the complete tree is specified uniquely (and then it cannot be a proper subtree of any other tree with an equivalent root label).

For the formal statement of the theorem we first introduce the "left part property" of a set of grammatical trees.

Definition 2.2. Let $\mathfrak{I} \subseteq \mathfrak{I}_G$ for some grammar $G = \langle V, \Sigma, P, S\rangle$ and let π be an arbitrary partition on V, not necessarily strict. We say that $\mathfrak{I}$ satisfies the *left-part property with respect to* π iff for any $n \geq 0$ and $T, T' \in \mathfrak{I}$ if $\mathbf{rt}(T) \equiv \mathbf{rt}(T') \pmod{\pi}$ and ${}^{(n)}\mathbf{fr}(T) = {}^{(n)}\mathbf{fr}(T')$ then

$$ {}^{(n+1)}T \cong {}^{(n+1)}T', \tag{1} $$

and, moreover, if $x \to x'$ is the structural isomorphism ${}^{(n+1)}T \to {}^{(n+1)}T'$ then for every x in ${}^{(n+1)}T$,

$$ \lambda(x) = \lambda(x') \text{ if } x \mathrel{\lrcorner}^+ y \text{ for some } y \in {}^{(n+1)}T \text{ or if } {}^{(n+1)}T = {}^{(n)}T \tag{2} $$

and

$$ \lambda(x) \equiv \lambda(x') \pmod{\pi} \text{ otherwise.} \tag{3} $$

Note that the condition "$x \mathrel{\lrcorner}^+ y$ for some $y \in {}^{(n+1)}T$" in (2) is equivalent to "x is not on the rightmost path in ${}^{(n+1)}T$."

An example of this kind of mapping can be obtained by considering grammar G_2 from the Introduction. Let us consider the trees shown in Figure 5. If we let $n = 2$, we see that $\mathbf{rt}(T) = S \equiv S = \mathbf{rt}(T')$ and ${}^{(2)}\mathbf{fr}(T) = (a = {}^{(2)}\mathbf{fr}(T')$. Thus ${}^{(3)}T \cong {}^{(3)}T'$ and the path from S to $*$ in T is "equivalent modulo π" to the path from S to) in T'.

Note that the left-part property is a *global property of trees* in distinction to the strictness of a grammar, which is a *local property* of grammatical trees (being a property of their elementary subtrees).

THEOREM 2.1 (Left Part Theorem). *Let $G = \langle V, \Sigma, P, S\rangle$ be a reduced grammar and let π be a partition on V such that $\Sigma \in \pi$. Then π is strict in G iff the set $\mathfrak{I}_G$ of all grammatical trees of G satisfies the left part property with respect to π.*

PROOF (The "if" direction). Let G and π be as in the assumption of the theorem and assume $\mathfrak{I}_G$ satisfies the left part property with respect to π. We shall show that π is strict.

Let $A, A' \in N$, $A \to \alpha\beta$, $A' \to \alpha\beta$, and $A \equiv A' \pmod{\pi}$. Since G is reduced we have $\alpha \Rightarrow^* w_\alpha$, $\beta \Rightarrow^* w_\beta$, and $\beta' \Rightarrow^* w_{\beta'}$ for some $w_\alpha, w_\beta, w_{\beta'} \in \Sigma^*$. Let us consider two grammatical trees T, T' corresponding to derivations $A \Rightarrow \alpha\beta \Rightarrow^* w_\alpha w_\beta$ and $A' \Rightarrow \alpha\beta' \Rightarrow^* w_\alpha w_{\beta'}$ respectively. Let $n = \lg(w_\alpha)$. Then $\mathbf{rt}(T) = A \equiv A' = \mathbf{rt}(T')$ and ${}^{(n)}\mathbf{fr}(T) =$

[5] Note that the Λ-nodes are not being indexed.

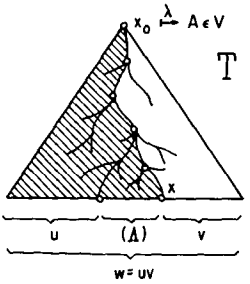

FIG. 4. The left part of a tree

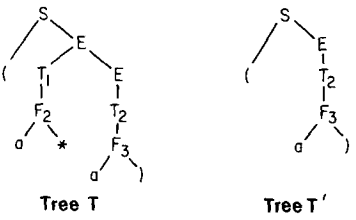

FIG. 5. Two trees from grammar G_2

${}^{(n)}\mathbf{fr}(T')$. Thus ${}^{(n+1)}T \cong {}^{(n+1)}T'$ by the left part property of $\mathfrak{Z}_G$. Let $x \to x'$ be the structural isomorphism from ${}^{(n+1)}T$ to ${}^{(n+1)}T'$. We distinguish two cases:

Case 1. $w_\beta \neq \Lambda$. Let x be the $(n+1)$-st terminal node of T (labeled by ${}^{(1)}w_\beta$) and y the $(\lg(\alpha)+1)$-st node among the immediate descendants of the root in T, counted from the left (i.e. $\lambda(y) = {}^{(1)}\beta$). Clearly y is in ${}^{(n+1)}T$ and by the structural isomorphism also y' is in ${}^{(n+1)}T'$, $\lambda(y') = {}^{(1)}\beta'$. Then ${}^{(1)}\beta \equiv {}^{(1)}\beta'$ by (2) or by (3) (depending on whether $y \mathrel{\lrcorner^+} x$ or not).

Case 2. $w_\beta = \Lambda$. Then ${}^{(n+1)}T = {}^{(n)}T$ and by (2), $\lambda(x) = \lambda(x')$ for all x in ${}^{(n+1)}T$. Thus ${}^{(n+1)}T = {}^{(n+1)}T'$. Then also ${}^{(n)}T = {}^{(n)}T'$ by Fact 2.2 and using Fact 2.1 we conclude $T = T'$. Hence $A = A'$ and $\beta' = \beta = \Lambda$.

(The "only if" direction). Let G be a reduced grammar with strict partition π. Let $T, T' \in \mathfrak{Z}_G$ be two grammatical trees such that

$$\mathbf{rt}(T) \equiv \mathbf{rt}(T') \pmod{\pi} \tag{4}$$

and

$${}^{(n)}\mathbf{fr}(T) = {}^{(n)}\mathbf{fr}(T') \tag{5}$$

for some $n \geq 0$. To show that ${}^{(n+1)}T$ and ${}^{(n+1)}T'$ satisfy (1), (2), and (3) we proceed by induction on height h of the larger one of the two trees. Assume, without loss of generality, that the taller tree is T.

Basis. $h = 0$. Then T consists of a single node labeled by some $a \in \Sigma$. (Note that we cannot have $a = \Lambda$ since in such a case, assumption (4) would be meaningless.) Then by (4) and since $\Sigma \in \pi$ also, T' has a single node labeled by some $a' \in \Sigma$. Thus $T \cong T'$. Clearly $a \equiv a'$ and, moreover, if $n \geq 1$ we have $a = a'$ from (5).

Inductive step. Let $h > 0$ and assume the result, i.e. (1), (2), and (3), true for all trees with heights smaller than h. Let us denote this assumption (A). Let x_0 and x_0' be the roots of T and T' respectively, and let $x_0 \mathrel{\Gamma} x_1, \cdots, x_0 \mathrel{\Gamma} x_m$, $x_0' \mathrel{\Gamma} x_1', \cdots, x_0' \mathrel{\Gamma} x_{m'}'$, $x_1 \mathrel{\lrcorner} x_2 \mathrel{\lrcorner} \cdots \mathrel{\lrcorner} x_m$, and $x_1' \mathrel{\lrcorner} x_2' \mathrel{\lrcorner} \cdots \mathrel{\lrcorner} x_{m'}'$, (cf. Figure 6).

Let us define for $1 \leq i \leq m$ and $1 \leq j \leq m'$, $T_i = \{x \text{ in } T \mid x_i \mathrel{\Gamma^*} x\}$, $T_j' = \{x \text{ in } T' \mid x_j' \mathrel{\Gamma^*} x\}$. Clearly all the T_i and T_j' are grammatical trees of G and because their heights are smaller than h the inductive hypothesis (A) is applicable. Denote $w_i = \mathbf{fr}(T_i)$, $w_j' = \mathbf{fr}(T_j')$.

Claim. If for some $k \leq m$, no tree among $T_1, \cdots, T_k$ contains the $(n+1)$-st terminal node of T (in particular, if no such node exists) then $m' \geq k$ and $T_i = T_i'$ for $1 \leq i \leq k$.

The argument is an induction on k. The *basis* (in which $k = 0$) is vacuously true.

Inductive step. Assume the premise of the claim for some $k > 0$ and, as inductive hypothesis (B), the conclusion for $k-1$, i.e. $m' \geq k-1$ and $T_i = T_i'$ $(1 \leq i \leq k-1)$. Then, in particular, $\lambda(x_i) = \lambda(x_i')$ for $1 \leq i \leq k-1$ and by the strictness of π, using (4), $\lambda(x_k) \equiv \lambda(x_k')$. Since $w_1 \cdots w_k$ is a prefix of ${}^{(n)}\mathbf{fr}(T) = {}^{(n)}\mathbf{fr}(T')$ and since $w_i \cdots w_{k-1} = w_1' \cdots w_{k-1}'$ by inductive hypothesis (B), either $\mathbf{fr}(T_k)$ is a prefix of $\mathbf{fr}(T_k')$, or conversely. But since $\mathbf{rt}(T_k) = \lambda(x_k) \equiv \lambda(x_k') = \mathbf{rt}(T_k')$ and using inductive hypothesis (A) we conclude that $w_k = w_k'$ and $T_k = T_k'$. Therefore the claim follows.

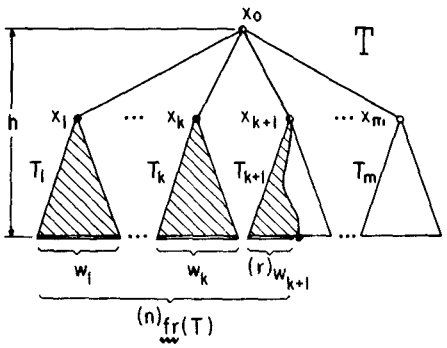

FIG. 6. An illustration of the inductive step

Now if $n \geq \lg(\mathbf{fr}(T))$ (i.e. if ${}^{(n)}\mathbf{fr}(T) = \mathbf{fr}(T)$) then by the claim $T_i = T_i'$ $(1 \leq i \leq m \leq m')$. But then $\mathbf{fr}(T) = w_1 \cdots w_m = w_1' \cdots w_m'$ and by the strictness of π we have $m = m'$, and since $\lambda(x_0) \equiv \lambda(x_0')$ we have then, again by the strictness of π, $\lambda(x_0) = \lambda(x_0')$. Hence ${}^{(n+1)}T = T = T' = {}^{(n+1)}T'$.

For the rest of the proof we assume $n < \lg(\mathbf{fr}(T))$ (cf. Figure 4). Let y be the $(n+1)$-st terminal node in T and let k and r be such that y is the $(r+1)$-st terminal node in T_{k+1}. By the claim $T_i = T_i'$ $(1 \leq i \leq k)$ and thus, by the strictness of π, $m' > k$ and $\mathbf{rt}(T_{k+1}) = \lambda(x_{k+1}) \equiv \lambda(x_{k+1}') = \mathbf{rt}(T_k')$. Moreover, $w_1 \cdots w_k = w_1' \cdots w_k'$ and since ${}^{(r)}w_{k+1}$ is a suffix of ${}^{(n)}\mathbf{fr}(T) = {}^{(n)}\mathbf{fr}(T')$ we have ${}^{(r)}w_{k+1} = {}^{(r)}(w_{k+1}' \cdots w_{m'}')$. The only possibility which is in agreement with the induction hypothesis (A) is ${}^{(r)}w_{k+1} = {}^{(r)}w_{k+1}'$.

Now ${}^{(n+1)}T$ consists of $x_0, T_1, \cdots, T_k$, and ${}^{(r+1)}T_{k+1}$; ${}^{(n+1)}T'$ consists of $x_0', T_1', \cdots T_k'$, and ${}^{(r+1)}T_{k+1}'$. Here ${}^{(r+1)}T_{k+1} \cong {}^{(r+1)}T_{k+1}'$ by the inductive hypothesis. Define a function $f : {}^{(n+1)}T \to {}^{(n+1)}T'$ such that f restricted to T_i is the structural isomorphism $T_i \to T_i'$ $(0 \leq i \leq k)$, f restricted to ${}^{(r+1)}T_{k+1}$ is the structural isomorphism ${}^{(r+1)}T_{k+1} \to {}^{(r+1)}T_{k+1}'$ and $f(x_0) = x_0'$. Now f is a structural isomorphism ${}^{(n+1)}T \to {}^{(n+1)}T'$. Hence (1). Let $x \in {}^{(n+1)}T$. If $x \mathrel{\mathrm{L}}^{+} y$ then $\lambda(x) = \lambda f(x)$ since either $x \in {}^{(r+1)}T_{k+1}$ and we can apply the inductive hypothesis; or else $x \in T_i$, $i \leq k$. Hence (2). Otherwise $\lambda(x) \equiv \lambda f(x)$ since either $x \in {}^{(r+1)}T_{k+1}$ and we can again apply the induction hypothesis or $x = x_0$ and $f(x) = x_0'$. Hence (3) and $\mathfrak{I}_G$ satisfies the left part property. Q.E.D.

Note that the assumption that G is reduced was not used in the "only if" part of the proof. On the other hand, in the "if" part, even if $\mathfrak{I}_G$ satisfies the left part property the strictness of π may be violated for some useless production which does not appear in any grammatical tree. By [17, Th. 2.1] the requirement of a reduced grammar causes no loss of generality.

For reduced grammars we have the following result.

Fact 2.3. Let G be a reduced grammar. The left part property is satisfied by $\mathfrak{I}_G$ iff it is satisfied by $\mathfrak{I}_G(S)$, the set of derivation trees in G.

The "only if" direction of this fact is trivial while the "if" direction follows from the observation that in a reduced grammar every grammatical tree is a subtree of some derivation tree.

The following two corollaries are immediate consequences of the Left Part Theorem.

COROLLARY 1. *Any strict deterministic grammar is unambiguous.*

COROLLARY 2. *If π is a strict partition on G then for every $U \in \pi$ the set $\{w \in \Sigma^* \mid A \Rightarrow^* w$ for some $A \in U\}$ is prefix-free.*

Thus, in particular, $L(G)$ is prefix-free, which we already know from [17, Th. 2.2].

Our next objective is to give a machine-independent proof of a deterministic iteration theorem. This deterministic variant appears in a slightly different formulation in [23] where the proof is sketched using a special type of pushdown automaton. Here we give another proof using the Left Part Theorem.

Definition 2.3. Let $w \in \Sigma^*$. Any sequence $\phi = (v_1, \cdots, v_n) \in (\Sigma^*)^n$, $n \geq 1$, such that $v_1v_2 \cdots v_n = w$ is called a *factorization of* w and any integer i, $1 \leq i \leq \lg(w)$, is called a *position in* w. Let K be a set of positions in w. Any factorization $\phi = (v_1, \cdots, v_n)$ of w induces a natural "partition" of K (it is possible that $K_i = \varnothing$ for some i): $K/\phi = \{K_1, \cdots, K_n\}$, where for $1 \leq i \leq n$, $K_i = \{k \in K \mid \lg(v_1 \cdots v_{i-1}) < k \leq \lg(v_1 \cdots v_{i-1}v_i)\}$.

THEOREM 2.2 (Iteration Theorem[6] for Δ_2). *Let $L \in \Delta_2$ and $L \subseteq \Sigma^*$. There is an integer $p(L)$ such that for any string $w \in L$ and any set K of positions in w, if $\mid K \mid \geq p(L)$ then there is a factorization $\phi = (v_1, \cdots, v_5)$ of w such that (1) $v_2 \neq \Lambda$; (2) for each n, $m \geq 0$, $u \in \Sigma^*$, $v_1v_2^{n+m}v_3v_4^nu \in L$ iff $v_1v_2^mv_3u \in L$; (3) if $K/\phi = \{K_1, \cdots, K_5\}$ then (i) K_1, K_2, $K_3 \neq \varnothing$ or K_3, K_4, $K_5 \neq \varnothing$, (ii) $\mid K_2 \cup K_3 \cup K_4 \mid \leq p(L)$.*

As a particular case of Property (2) (taking $m = 1$ and $u = v_4v_5$ and afterwards $m = 0$) we have: (2′) for each $n \geq 0$, $v_1v_2^nv_3v_4^nv_5 \in L$.

Before we prove Theorem 2.2 we present its version for Δ_0.

THEOREM 2.3 (Iteration Theorem for Δ_0). *Let $L \in \Delta_0$, $L \subseteq \Sigma^*$. There is an integer $p'(L)$ such that for every $w \in L$ and every set K as in Theorem 2.2 there exists a factorization $\phi = (v_1, v_2, v_3, v_4, v_5)$ of w satisfying Properties (1), (2′), and (3), and if $v_5 \neq \Lambda$ then also Property (2) holds.*

Remark. Using the well-known fact [15, 9] that Δ_0 is closed under complementation (i.e. if $L \in \Delta_0$ then $\Sigma^* - L \in \Delta_0$ for every alphabet Σ) and taking as $p(L)$ the larger number from $p(L)$ and $p(\Sigma^* - L)$, we can obtain a more general form of Theorem 2.3 where "for every $w \in L$" is replaced by "for every $w \in \Sigma^*$" and (2′) by: (2″) for all $n \geq 0$, $v_1v_2^nv_3v_4^nv_5 \in L$ iff $w \in L$.

PROOF OF THEOREM 2.2. (Part of the following proof is independent of the property of G being strict deterministic and follows exactly the lines of the standard proof of the "nondeterministic" iteration theorem—cf. [23, 16]. We present this part in a somewhat concise form.) Let $G = \langle V, \Sigma, P, S\rangle$ be a grammar generating L. Define $p(L) = p = r^{2\nu+3}$ where $r = \max\{2, \lg(\alpha) \mid A \to \alpha \text{ in } P\}$, $\nu = \mid N \mid$. Consider a derivation tree $T \in \mathfrak{I}_G(S)$ with $\mathbf{fr}(T) = w \in \Sigma^*$ and let $y_1, y_2, \cdots, y_{\lg(w)}$ be the sequence of all terminal nodes in T, $y_i \mathrel{\lrcorner}^+ y_{i+1}$, $1 \leq i \leq \lg(w)$. Define the following two sets of nodes in T:

$D = \{x \text{ in } T \mid x \mathrel{\Gamma}^* y_i \text{ for some } i \in K\}$,

$B = \{x \text{ in } T \mid x \mathrel{\Gamma} x_1 \text{ and } x \mathrel{\Gamma} x_2 \text{ for some } x_1, x_2 \in D, x_1 \neq x_2\}$.

(Nodes of the second set are called *B-nodes*). Consider a path q in T which contains the maximum possible number of B-nodes and let z_0 be the leaf of q. Since $\mid K \mid \geq p = r^{2\nu+3}$ this path contains at least $2\nu+3$ B-nodes. Let C be the set of the "lowest" $2\nu+3$ B-nodes on q, i.e. $C \subseteq B$, $\mid C \mid = 2\nu+3$, and for each $y \in B$, if $x \mathrel{\Gamma}^* y$ for some $x \in C$ and if y belongs to q, then also $y \in C$. Define two sets

$C_L = \{x \in C \mid x \mathrel{\Gamma} y \text{ and } y \mathrel{\lrcorner}^+ z_0 \text{ for some } y \in D\}$,

$C_R = \{x \in C \mid x \mathrel{\Gamma} y \text{ and } z_0 \mathrel{\lrcorner}^+ y \text{ for some } y \in D\}$.

Now $\mid C_L \cup C_R \mid = \mid C \mid = 2\nu + 3$ and hence either $\mid C_L \mid \geq \nu + 2$ or $\mid C_R \mid \geq \nu + 2$.

Case 1. $\mid C_L \mid \geq \nu + 2$. Let $C_L = \{x_1, \cdots, x_m\}$ where $x_i \mathrel{\Gamma}^+ x_{i+1}$ $(1 \leq i < m)$. Since $m \geq \nu + 2$ and there are only ν nonterminals in G,

$$\lambda(x_j) = \lambda(x_k) = A \text{ for some } 1 < j < k \leq m \text{ and } A \in N. \tag{6}$$

Therefore we can find a factorization $\phi = (v_1, \cdots, v_5)$ of w such that tree T corresponds to derivation

$$S \Rightarrow^* v_1Av_5 \Rightarrow^+ v_1v_2Av_4v_5 \Rightarrow^+ v_1v_2v_3v_4v_5 = w \tag{7}$$

and hence

$$S \Rightarrow^* v_1v_2^nv_3v_4^nv_5 \text{ for any } n \geq 0. \tag{8}$$

To prove the theorem let us take the above factorization ϕ. Let us first verify Property

[6] Δ_2 denotes the family of strict deterministic languages while Δ_0 is the family of all deterministic languages.

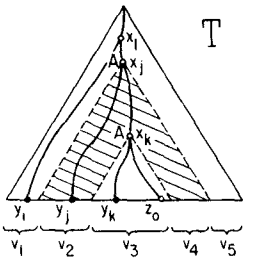

FIG. 7. A grammatical tree for derivation (7)

(3) from Theorem 2.2. Let $\{K_1, \cdots, K_5\} = K/\phi$. We use the definition of C_L and (6) to obtain three positions $i, j, k \in K$ such that $x_1 \ \Gamma^* y_i$, $x_j \ \Gamma^* y_j$, $x_k \ \Gamma^* y_k$, and $y_i \ \mathrm{L}^+ y_j \ \mathrm{L}^+ y_k \ \mathrm{L}^+ z_0$. By (7), $i \in K_1$, $j \in K_2$, $k \in K_3$ (cf. Figure 7).

Case 2. $|C_R| \geq \nu + 2$. This case is handled similarly to the first case. In particular, we obtain the same derivations as in (7) and (8). Then a symmetrical argument would yield $z_0 \ \mathrm{L}^+ y_k \ \mathrm{L}^+ y_j \ \mathrm{L}^+ y_i$ and $k \in K_3$, $j \in K_4$, $i \in K_5$.

Since q has the maximum number of B-nodes and exactly $2\nu+3$ nodes on q are descendants of x_1 no other path in the subtree of T with root x_j has more nodes than $2\nu+3$. Consequently, $|K_2 \cup K_3 \cup K_4| \leq p = r^{2\nu+3}$.

To verify Properties (1) and (2) we use the strict determinism of G. Then Property (1) is immediate from (7) since otherwise we would have $A \Rightarrow^+ Av_4$ which would contradict [17, Th. 2.3].

For Property (2), let $n, m \geq 0$. Using (7) we can write

$$S \Rightarrow^* v_1 v_2^m A v_4^m v_5 \Rightarrow^* v_1 v_2^{m+n} v_3 v_4^{n+m} v_5 = w_1 v_4^m v_5 \tag{9}$$

where $w_1 = v_1 v_2^{m+n} v_3 v_4^n \in \Sigma^*$. Assume now

$$S \Rightarrow^* v_1 v_2^{m+n} v_3 v_4^n u = w_1 u. \tag{10}$$

Let T, T' be two derivation trees corresponding to (9) and (10) respectively. Let $k = \lg(w_1)$. By the Left Part Theorem, ${}^{(k+1)}T \cong {}^{(k+1)}T'$. Let x be the node in T which corresponds to the indicated occurrence of A in sentential form $v_1 v_2^m A v_4^m v_5$. Clearly x is left of the $(k+1)$-st terminal node in T (labeled with ${}^{(1)}v_4$). By the left part property (2) the corresponding node in T' is also labeled with A. Therefore the following derivation corresponds[7] to T': $S \Rightarrow^* v_1 v_2^m A u \Rightarrow^* w_1 u$. Since $A \Rightarrow^+ v_2^{n'} v_3 v_4^{n'}$ for any $n' \geq 0$ we have $v_1 v_2^{m+n'} v_3 v_4^{n'} u \in L$, any $n' \geq 0$.

In particular, taking $n' = 0$ we obtain the "only if" part of Property (2); taking $n = 0$ in (9) and n' arbitrary we obtain the "if" direction of 2. Q.E.D.

PROOF OF THEOREM 2.3. Notice that in the first part of the last proof we have not used the strict determinism of G. Thus using the same factorization as above we obtain Property 3 and from (7) also 2′, for any $L \in \Delta_0$ (in fact for any context-free language).

For the last assertion of Theorem 2.3 let $L \in \Delta_0$. Then $L\$ \in \Delta_2$ and is subject to Theorem 2.2. Define $p'(L) = p(L\$)$ and let $w \in L$ with a set of positions K where $|K| \geq p'(L)$. Consider the factorization $\phi' = (v_1, v_2, v_3, v_4, v_5')$ of $w\$$ obtained from Theorem 2.2 applied to $L\$$. Since \$ doesn't occupy any position from K, there are only two possibilities where it can occur: either $v_3 \in \Sigma^+\$$ and $v_4 = v_5' = \Lambda$ or $v_3 \in \Sigma^+$ and $v_5' = v_5\$$ for some $v_5 \in \Sigma^*$. In any event (1) holds. Under the assumption $v_5 \neq \Lambda$ from Theorem 2.3 we can restrict ourselves to the second possibility. Take $\phi = (v_1, v_2, v_3, v_4, v_5)$ as the factorization of w. The result is now simple since for any $n, m \geq 0$ and $u \in \Sigma^*$ we have $v_1 v_2^{n+m} v_3 v_4^n u \in L$ iff $v_1 v_2^{n+m} v_3 v_4^n u\$ \in L\$$ iff $v_1 v_2^m v_3 u\$ \in L\$$ iff $v_1 v_2^m v_3 u \in L$. Q.E.D.

[7] For a completely precise formulation we would have to talk in terms of cross sections, but we want to avoid the tiresome details.

We conclude this section with one application of the Iteration Theorem. The example has a certain general importance as the Corollary will show.

Example 2.1. The language $L = \{ww^T \mid w \in \{a, b\}^*\}$ is not a finite union of deterministic languages. (This fact is mentioned without proof in [9].)

Proof. Assume for the sake of contradiction that $L = \bigcup_{i=1}^{N} L_i$ for some $N \geq 1$ and $L_i \in \Delta_0$. Let $p = \max_{1 \leq i \leq N} p'(L_i)$ where $p'(L_i)$ are the constants from Theorem 2.3 for every $i \leq N$. Consider the set $H = \{(ba^pb)^{2^n} \mid n \geq 1\} \subseteq L$. H is infinite and therefore for some L_i, $|H \cap L_i| \geq 2$ or, more specifically, there are two (even) numbers $n_1, n_2 \geq 2$ such that both $w = (ba^pb)^{n_1} \in L_i$ and $w' = w(ba^pb)^{n_2} \in L_i$. Applying the Iteration Theorem (2.3) to w, let $K = \{i \mid (n_1 - 1)(p + 2) < i < n_1(p + 2) = \lg(w)\}$ (i.e. K corresponds to the rightmost substring a^p in w). Clearly $|K| = p \geq p'(L_i)$. Let $(v_1, v_2, v_3, v_4, v_5)$ be the factorization of w from Theorem 2.3. By Property (3) and since strings in L_i are palindromes, we have $v_5 \in ba^+$, $v_5 \in a^+b$, and $v_2 = v_4 = a^q$ for some q, $1 \leq q < p$. Define $u = v_4v_5(ba^pb)^{n_2}$. Now taking $m = n = 1$ in 2 (note that $v_5 \neq \Lambda$), $v_1v_2v_3u = w' \in L_i$ implies

$$v_1v_2^2v_3v_4u = ba^{p+q}b(ba^pb)^{n_1-2}ba^{p+q}b(ba^pb)^{n_2} \in L_i,$$

which is not a palindrome since $q, n_2 \neq 0$. We have a contradiction with $L_i \subseteq L$. Q.E.D.

Since L is an unambiguous language, we have the following consequence.

COROLLARY. *There exists an unambiguous context-free language which is not a finite union of deterministic languages.*

III. *Background of LR and BRC Grammars*

In the present section, we summarize the background material on LR(k) grammars [19] and bounded right context grammars (BRC for short) [5]. A number of errors and omissions from the literature are discussed.

In order to introduce LR and BRC grammars, we must first introduce handles.

Definition 3.1. Let G be a grammar of the form

$$G = \langle V, \Sigma, P, S \rangle \tag{1}$$

and let $\alpha \in V^*$. Any pair (p, i) where $p \in P$ and $i \geq 0$ is called a *handle of* α *(in* G*)* iff for some $A \in N$, $\alpha', \beta \in V^*$, and $w \in \Sigma^*$ we have

(i) $S \Rightarrow_R^* \alpha' Aw \Rightarrow_R \alpha'\beta w = \alpha$;

(ii) p is a production of the form $A \rightarrow \beta$, and

(iii) $i = \lg(\alpha'\beta)$.

Note that the canonical sentential form S does not have a handle unless $S \Rightarrow_R^+ S$ in G.

The motivation behind the previous definition is that one can parse a string by finding and reducing a handle. The LR(k) grammars, introduced in [19], have the property that the handle of a string can be uniquely determined in a left to right parse with a lookahead of k letters.

Definition 3.2. Let G be a grammar of the form (1) with no derivations of the form $S \Rightarrow_R^+ S$. Let $k \geq 0$. Then G is called an LR(k) *grammar*[8] iff the following condition holds for all $\alpha \in V^*$, $w, w' \in \Sigma^*$, $p, p' \in P$, and $i' \geq 0$. If $(p, \lg(\alpha))$ and (p', i') are handles of αw and $\alpha w'$ respectively, and if ${}^{(k)}w = {}^{(k)}w'$, then $p' = p$ and $i' = \lg(\alpha)$. Grammar G is called an LR *grammar* iff it is LR(k) for some $k \geq 0$.

The next type of grammar, the bounded right context grammars, introduced in [5], allow for unique determination of a handle by looking l letters to the left and k to the right.

Definition 3.3. Let G be a grammar of the form (1) with no derivations of the form $S \Rightarrow_R^+ S$. Let $l, k \geq 0$. Then G is called a BRC(l, k) *grammar*[9] iff the following condition

[8] "LR(k)" stands for "parsable from *l*eft to *r*ight using k letters lookahead."

[9] "BRC(l, k)" stands for "parsable with *b*ounded *r*ight context using l letters to the left and k lookahead."

holds for all $\alpha, \alpha', \beta \in V^*$, $w, w' \in \Sigma^*$, $A \in N$, $p' \in P$, and $i' \geq \lg(\alpha'\beta)$. If $(A \to \beta, \lg(\alpha\beta))$ and (p', i') are handles of $\alpha\beta w$ and $\alpha'\beta w'$ respectively, and if $\alpha^{(i)} = \alpha'^{(i)}$ and ${}^{(k)}w = {}^{(k)}w'$, then p' has the form $A \to \beta$ and $i' = \lg(\alpha'\beta)$. Grammar G is called a BRC grammar iff it is BRC(l, k) grammar for some $l, k \geq 0$.

We have to comment on the particular forms of our definitions. First, the requirement of no derivation of the form $S \Rightarrow_R^+ S$, which is usually omitted,[10] is important since otherwise LR and BRC grammars would not be unambiguous grammars (the grammar $S \to S \mid a$ would be an ambiguous LR (0) grammar). For BRC grammars we require, as in [10], that the condition in Definition 3.3 be satisfied only if $i' \geq \lg(\alpha'\beta)$. This yields exactly the class of languages defined in [5]. (If we were to assume that $w, w' \in V^*$, then Λ-rules create problems with LR(0) and BRC$(l, 0)$ grammars.[11]) If one works with Λ-free grammars then some of these points do not arise.

Our definition of LR differs from that of Aho and Ullman [1]. It has been shown that the two definitions are equivalent for $k \geq 1$ [7]. For $k = 0$, the Aho-Ullman definition allows only strict deterministic languages. By choosing the present definition, we can capture all of the languages called LR(0) by Knuth [19]. It should be pointed out that the LR parser given by Knuth does not work on all the LR(0) grammars, but it is not difficult to modify it so that all LR grammars are LR parsable. We elaborate on these points further in [7].

As an immediate consequence of our definitions we have

Fact 3.1. For any $l, k \geq 0$ if G is LR(k) then G is LR(k') for all $k' \geq k$; if G is BRC(l, k) then it is LR(k) and also BRC(l', k') for all $l' \geq l$ and $k' \geq k$.

LR grammars were first introduced in [19] and BRC grammars in [5]. In both cases the intention was to give a formal characterization of a class of grammars suitable for deterministic bottom-up parsing. As a matter of fact, the class of LR grammars is one of the most general classes with a deterministic linear time parsing procedure currently known. It is no wonder that LR grammars have been widely used in practically oriented research (parsing and translation of programming languages, cf. [1, 3, 10–12, 14, 21, 25]). Despite this fact there are only a few general results in this area and much of the present theoretical knowledge about LR and BRC grammars can be summarized in the following eight propositions (the reader will note that almost all of them are from the original paper of Knuth [19]).

PROPOSITION 1. *For a given grammar G and given $k \geq 0$ it is decidable whether G is LR(k) or not.*

A procedure testing the LR(k) property is described in [1, 18, 19]. The test is by no means trivial due to the fact that the condition in Definition 3.2 actually depends on an infinite set (of all sentential forms in G). This is one of the arguments in favor of our approach using strict deterministic grammars.

PROPOSITION 2. *It is undecidable whether a given grammar is an LR grammar or not.*

The proof involves a reduction to the Post correspondence problem and appears in [19].

PROPOSITION 3. *Every LR grammar is unambiguous.*

This "fact" is usually stated as a direct consequence of the definition of LR grammars (cf. [18, 19]) but the restriction disallowing $S \Rightarrow_R^+ S$ is not mentioned.

PROPOSITION 4. *If G is an LR grammar then $L(G)$ is a deterministic language ($L(G) \in \Delta_0$). There is an effective procedure constructing the corresponding DPDA.*

A construction of DPDA for a given LR grammar is described in [11, 14].

PROPOSITION 5. *Any deterministic language has an LR(1) grammar.*

This result and a construction of the corresponding LR(1) grammar is presented without a rigorous proof in [19]. It appears also in [18] but the proof is not correct. Using properties of strict deterministic grammars we give below a proof (Theorem 6.1) which

[10] Rules of the form $S \to S$ are excluded in the LR definition of Salomaa [24].

[11] Assume $S \Rightarrow_R^+ \alpha A\beta \Rightarrow_R \alpha\beta$. Then the handle $(A \to \Lambda, \lg(\alpha))$ of $\alpha\beta$ would have to be a handle of $\alpha A\beta$ too, etc., and A would never be reduced.

was, to the authors' knowledge, the first complete proof of this important result. Lehmann [22] also attempts to prove Proposition 5. Although the proof of his Theorem II is informal, it can be expanded and the result is correct. However, his Theorem I is false and thus his approach fails to verify that the constructed grammar is LR(0). Also Aho and Ullman [1, 2] obtain Proposition 5 as a consequence of precedence techniques.

PROPOSITION 6. *Any LR grammar is equivalent to an LR(1) grammer.*

This is a direct consequence of Propositions 4 and 5. A direct transformation from LR(k) to LR(1) would be suitable (and important) but none is known. The proposition cannot be strengthened to LR(0): for instance the grammar $S \to a \mid \Lambda$ is LR(1) but has no equivalent LR(0) grammar.

PROPOSITION 7. *Any BRC(l, k) grammar ($l, k \geq 0$) is equivalent to a BRC(1, k) grammar.*

The proof and complete transformation procedure (preserving some other important properties of grammars) is in [10, 11].

PROPOSITION 8. *Any LR(k) grammar ($k > 0$) is equivalent to a BRC(1, k) grammar.*

A transformation from LR to BRC is indicated in [19] and is described in greater detail in [10, 11]. However, neither author provides a complete formal proof of its correctness. Again using strict deterministic grammars we will give a proof of the above proposition (Theorem 6.2). Incidentally, this result together with Fact 3.1 can serve as an alternative proof of Proposition 5.

It will be seen in the following sections that strict deterministic grammars can give simple proofs of these important propositions as well as other theorems about deterministic languages. The reader might wonder if it is not possible to give a finitary definition of a deterministic grammar which can generate all and only the deterministic languages. The LR(k) definition would fail to satisfy the finitary criterion since quantification over infinite sets is involved.

We now present such a finitary definition which is, as one might expect, a generalization of the existence of a strict partition.

Definition 3.4. A grammar $G = \langle V, \Sigma, P, S\rangle$ is *deterministic* iff there is a partition π of V and a subset $E \subseteq V - \Sigma$ such that $\Sigma \in \pi$ and for every $A, A' \in V - \Sigma$ and $\alpha, \beta, \beta' \in V^*$ (1) if $A \to \alpha\beta$ and $\alpha \in V^*E$ then $\beta = \Lambda$ and $A \in E$; (2) if $A \to \alpha\beta$, $A' \to \alpha\beta'$ and $A \equiv A' \pmod{\pi}$ then at least one of the following four cases is true:

(i) $\beta, \beta' \neq \Lambda$ and ${}^{(1)}\beta \equiv {}^{(1)}\beta' \pmod{\pi}$;
(ii) $\beta = \beta' = \Lambda$ and $A = A'$;
(iii) $\beta = \Lambda$ and $A \in E$;
(iv) $\beta' = \Lambda$ and $A' \in E$.

Note that condition (1) is equivalent to the requirement

$$P \subseteq (V \times (V - E)^*) \cup (E \times (V - E)^*E),$$

and that we obtain a strict deterministic grammar as a special case when $E = \varnothing$.

A detailed proof that deterministic grammars generate precisely the languages in Δ_0 will not be given here since we prefer not to use this rather cumbersome definition in the sequel. Intuitively the idea of Definition 3.4 is related to a nondeterministic pushdown automaton, the nondeterminism of which consists solely in the decision whether to erase all of the pushdown store and halt when the machine finds itself in a final state, or to continue the computation (the acceptance is by empty store).

We chose the concept of a strict deterministic grammar as a basis of our theory, rather than the above concept of deterministic grammar. The advantage of the former concept is in its simplicity and its convenience in mathematical proofs.

IV. *Strict Deterministic and LR Grammars*

Our present goal is to show that the existence of a strict partition on a grammar implies that the grammar has the LR property. The proof is based on certain properties of deriva-

tion trees, in particular the Left Part Theorem. First we need one additional definition and three lemmas.

Definition 4.1. Let T be any tree (in the sense of the Introduction). We define a *canonical cross section* (*of level* n, for $0 \leq n$) *in* T inductively as follows:

(1) The sequence $\xi = (x_0)$, consisting of the root x_0 of T, is a canonical cross section (*of level* 0) in T.

(2) Let

$$\xi_0 = (x_1, \ldots, x_k, \ldots, x_m) \tag{1}$$

be a canonical cross section of level h in which x_k is the rightmost node which is internal in T. Let $y_1, y_2, \ldots, y_r$ be all the immediate descendants of x_k in T and let $y_1 \llcorner y_2 \llcorner \cdots \llcorner y_r$. Then the sequence

$$\xi_1 = (x_1, \ldots, x_{k-1}, y_1, \ldots, y_r, x_{k+1}, \ldots, x_r) \tag{2}$$

is a canonical cross section (*of level* $h + 1$) in T.

Every canonical cross section is a cross section as defined in the Introduction. The following two facts are immediate consequences of the above definition.

Fact 4.1. No two distinct canonical cross sections in a tree can be of the same level.

Fact 4.2. Let T be a grammatical tree of any grammar G and let $\xi = (x_1, \ldots, x_m)$ and $\xi' = (x_1', \ldots, x_m')$ be two canonical cross sections in T, of level h and h' respectively. Define $\lambda(\xi) = \lambda(x_1) \ldots \lambda(x_m) \in V^*$, and similarly $\lambda(\xi')$. Then $\lambda(\xi) \Rightarrow_R \lambda(\xi')$ iff $h' = h + 1$. In particular, if T is a derivation tree in G then $\lambda(\xi)$ is a canonical sentential form in G, and conversely, for every canonical sentential form α there is a derivation tree with a canonical cross section ξ such that $\lambda(\xi) = \alpha$.

The next two lemmas relate canonical cross sections in grammatical trees with canonical cross sections in left parts of these trees. (Recall Definition 2.1 for the concept of a left part ${}^{(n)}T$ of a tree T.)

LEMMA 4.1. *Let T be a grammatical tree of some grammar, ${}^{(n)}T$ the left part of T for some $n \geq 0$, and $\xi = (x_1, \ldots, x_k, y_1, \ldots, y_m)$ a canonical cross section in T with exactly the nodes $x_1, \ldots, x_k$ in ${}^{(n)}T$. Then $\eta = (x_1, \ldots, x_k)$ is a canonical cross section in ${}^{(n)}T$.*

PROOF. The argument is an induction on the level h of cross section ξ.

Basis. ($h = 0$) is immediate since $\xi = \eta = (x_0)$ where x_0 is the common root of T and ${}^{(n)}T$.

Inductive step. Assume the lemma true for canonical cross sections of level $h \geq 0$. Let ξ have level $h + 1$ and let ξ' be the (unique) canonical cross section of level h in T. Using the inductive hypothesis let η' be the canonical cross section in ${}^{(n)}T$ which consists of all the nodes of ξ' which are in ${}^{(n)}T$. Now, to obtain ξ from ξ' we need the rightmost internal node (with respect to T), say x, in ξ'. There are two cases. If x is not in ${}^{(n)}T$ then none of its descendants are in ${}^{(n)}T$ and $\eta = \eta'$. Hence η is a canonical cross section in ${}^{(n)}T$. On the other hand, if x is in ${}^{(n)}T$ then it is the rightmost internal node not only in ξ' (with respect to T) but also in η' (with respect to ${}^{(n)}T$) and by Definition 4.1 we conclude that η is a canonical cross section in ${}^{(n)}T$. Q.E.D.

The next lemma is in a certain sense a converse of Lemma 4.1.

LEMMA 4.2 *Let T be a grammatical tree of some grammar and let ${}^{(n)}T$ be the left part of T for some $n \geq 0$. Let $\eta = (x_1, \ldots, x_k)$ be a canonical cross section in ${}^{(n)}T$ and let $y_1, \ldots, y_m$ be all the leaves in T which are right of x_k; accordingly we assume $x_k \llcorner y_1 \llcorner \cdots \llcorner y_m$. Then the sequence $\xi = (x_1, \ldots, x_k, y_1, \ldots, y_m)$ is a canonical cross section in T.*

PROOF. Again the argument is an induction, this time on the level h of canonical cross section η.

Basis. $h = 0$. Let x_0 be the root of ${}^{(n)}T$ (and of T). Then $\eta = (x_0)$ and the only possibility for ξ is $\xi = \eta = (x_0)$. Hence ξ is a canonical cross section (of level 0) in T.

Inductive step. Assume the lemma true for canonical cross sections of level $h \geq 0$. Under the same notation as in Lemma 4.1, let $\eta = (x_1, \ldots, x_k)$ be of level $h + 1$ and

let $\eta' = (x_1', \ldots, x_{k'}')$ be the (unique) canonical cross section of level h in ${}^{(n)}T$. By the induction hypothesis the sequence $\xi' = (x_1', \ldots, x_k', y_j, \ldots, y_m)$ for some $j \geq 1$ is a canonical cross section in T. Since none of the nodes y_i $(j \leq i \leq m)$ is an internal node in T we conclude directly[12] from Definition 4.1 that $\xi = (x_1, \ldots, x_k, y_1, \ldots, y_m)$ is a canonical cross section in T. Q.E.D.

The third lemma expresses the hardly surprising fact that in the case of unambiguous grammars the canonical cross sections are uniquely determined by the corresponding canonical sentential forms.

LEMMA 4.3. *Let T be a derivation tree of some unambiguous grammar G and let ξ, ξ' be two canonical cross sections in T. Then $\lambda(\xi) = \lambda(\xi')$ implies $\xi = \xi'$.*

PROOF. Let $\xi = (x_1, \ldots, x_m)$ and $\xi' = (x_1', \ldots, x_{m'}')$. Assume, for the sake of contradiction, that $\lambda(\xi) = \lambda(\xi')$ but $\xi \neq \xi'$. We can assume, without loss of generality, that ξ' has greater level than ξ. Then all Λ-nodes of ξ are in ξ' and hence ξ and ξ' differ in nodes which are not labeled by Λ. Let $(x_{i_1}, \ldots, x_{i_r})$ and $(x_{j_1}', \ldots, x_{j_s}')$ be the sequences of all non-Λ-nodes of ξ and ξ' respectively (in the left-to-right order). Since $\lambda(\xi) = \lambda(\xi')$ we have $r = s$ and for each k, $1 \leq k \leq r$,

$$\lambda(x_{i_k}) = \lambda(x_{j_k}'). \tag{3}$$

For any k, $1 \leq k \leq r$, let T_k be the grammatical subtree of T which has the root x_{i_k} and nodes $\{y \mid x_{i_k} \; \Gamma^* \; y\}$. Similarly, let T_k' be the grammatical subtree of T which has the root x_{j_k}' and nodes $\{y \mid x_{j_k}' \; \Gamma^* \; y\}$. Consider another tree T' obtained from T in the following way: for every k, $1 \leq k \leq r$, replace T_k by T_k'. (Such a replacement is trivial when $x_{i_k} = x_{j_k}'$; this occurs for example when $\lambda(x_{i_k}) = \lambda(x_{j_k}') \in \Sigma$.) By (3) it can be easily seen that T' is also a derivation tree in G. We shall show that $\mathbf{fr}(T') = \mathbf{fr}(T)$. Since every terminal node in T is a descendant of some x_{j_k}', $1 \leq k \leq r$, we can write $\mathbf{fr}(T) = \mathbf{fr}(T_1')\mathbf{fr}(T_2') \cdots \mathbf{fr}(T_r')$. By definition of T' we have immediately $\mathbf{fr}(T') = \mathbf{fr}(T_1')\mathbf{fr}(T_2') \cdots \mathbf{fr}(T_r') = \mathbf{fr}(T)$. Now by the assumption that G is unambiguous we have $T' = T$. On the other hand, $\xi \neq \xi'$ implies $x_{i_k} \neq x_{j_k}'$ for some k, $1 \leq k \leq r$, and therefore x_{i_k} is not in T' (being lost by the replacement). Thus $T \neq T'$ which is a contradiction. Q.E.D.

We are now prepared for the main theorem of this section.

THEOREM 4.1. *Any reduced strict deterministic grammar is also an LR(0) grammar.*

PROOF. Let $G = \langle V, \Sigma, P, S \rangle$ be a reduced strict deterministic grammar. First note that $S \Rightarrow_R^+ S$ cannot occur in P since that would contradict [17, Th. 2.3]. Now following Definition 3.2, assume that for some $\alpha \in V^*$; $w, w' \in \Sigma^*$; $p, p' \in P$; and $n' \geq 0$, $(p, \lg(\alpha))$ and (p', n') are handles of αw and $\alpha w'$ respectively. We want to prove that these two handles are identical. Informally summarized, we shall prove this equivalence by first showing that each of the two handles appears in a certain position in a left part of a given derivation tree, and second, using the Left Part Theorem, that this left part is unique.

Since G is reduced there is some $u \in \Sigma^*$ such that $\alpha \Rightarrow_R^* u$. Let T, T' be two derivation trees of G corresponding to derivations $S \Rightarrow_R^* \alpha w \Rightarrow_R^* uw$ and $S \Rightarrow_R^* \alpha w' \Rightarrow_R^* uw'$. Since G is unambiguous (cf. Corollary 1 after Theorem 2.1), each of the two trees is unique. Moreover, since αw and $\alpha w'$ are canonical sentential forms (by Definition 3.1 this is a necessary condition for a sentential form to have a handle) there are canonical cross sections ξ_1 in T and ξ_1' in T' such that $\lambda(\xi_1) = \alpha w$ and $\lambda(\xi_1') = \alpha w'$; each canonical cross section is also unique in its respective tree (cf. Fact 4.2). We shall study these cross sections in detail (Figure 8). Assume ξ_1 is of level h (in T) and ξ_1' of level h' (in T'). The existence of handles implies an existence of a canonical cross section ξ_0 of level $h - 1$ in T and another canonical cross section ξ_0' of level $h' - 1$ in T'. Correspondingly to (1) and (2)

[12] We need only use the following observation: If $(x_1, \ldots, x_p, \ldots, x_m)$ is a canonical cross section where $x_{p+1}, \ldots, x_m$ are leaves and if $x_1', \ldots, x_q'$ are all the leaves in the subtree below x_p then $(x_1, \ldots, x_{p-1}, x_1', \ldots, x_q', x_{p+1}, \ldots, x_m)$ is a canonical cross section.

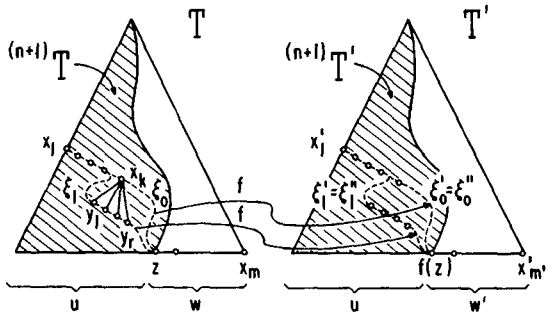

FIG. 8. The correspondence between cross sections in T and T'

in Definition 4.1 let us write for $1 \leq k \leq m$

$$\xi_0 = (x_1, \ldots, x_k, \ldots, x_m), \tag{4}$$

and for $r \geq 1$

$$\xi_1 = (x_1, \ldots, x_{k-1}, y_1, \ldots, y_r, x_{k+1}, \ldots, x_m), \tag{5}$$

where

$$\lambda(x_1) \cdots \lambda(x_{k-1})\lambda(y_1) \cdots \lambda(y_r) = \alpha, \qquad \lambda(x_{k+1}) \cdots \lambda(x_m) = w. \tag{6}$$

Similarly for ξ_0' and $1 \leq k' \leq m'$,

$$\xi_0' = (x_1', \ldots, x_{k'}', \ldots, x_{m'}'), \tag{7}$$

and for ξ_1' and $r' \geq 1$

$$\xi_1' = (x_1', \ldots, x_{k'-1}', y_1', \ldots, y_{r'}', x_{k'+1}', \ldots, x_{m'}'). \tag{8}$$

In (4) and (7), x_k and $x_{k'}'$ are the rightmost internal nodes in ξ_0 and ξ_0' respectively.

Next we shall apply the Left Part Theorem to T and T'. Assume first that $w \neq \Lambda$. Let $n = \lg(u)$ and let z be the $(n+1)$-st terminal node in T, i.e. $\lambda(z) = {}^{(1)}w$. From (6) we can see that z appears both in ξ_0 and ξ_1 or, more specifically, $z = x_l$ for some $l > k$. We have ${}^{(n)}\mathbf{fr}(T) = u = {}^{(n)}\mathbf{fr}(T')$ and the left part property of the derivation trees of G yields the existence of a structural isomorphism $f : {}^{(n+1)}T \to {}^{(n+1)}T'$ and, moreover, for any x in T,

$$x \; \mathrel{\mathsf{L}^+} z \text{ implies } \lambda(x) = \lambda(f(x)), \tag{9}$$

and for z itself

$$\lambda(z) \equiv \lambda(f(z)) = {}^{(1)}w'. \tag{10}$$

Define $\eta_0 = (x_1, \ldots, x_k, \ldots, z)$ and $\eta_1 = (x_1, \ldots, x_{k-1}, y_1, \ldots, y_r, x_{k+1}, \ldots, z)$. (Note that $x_k \neq z$, but $x_{k+1} = z$ is possible.) By Lemma 4.1 η_0 and η_1 are canonical cross sections in ${}^{(n+1)}T$. Because canonical cross sections are invariant under structural isomorphism, the following are canonical cross sections in ${}^{(n+1)}T'$ (cf. Figure 8): $f(\eta_0) = (f(x_1), \ldots, f(x_k), \ldots, f(z))$, $f(\eta_1) = (f(x_1), \ldots, f(x_{k-1}), f(y_1), \ldots, f(y_r), \ldots, f(z))$.

Extending $f(\eta_0)$ and $f(\eta_1)$ by all the leaves in T' on the right of $f(z)$ and using Lemma 4.2 we obtain two canonical cross sections ξ_0'' and ξ_1'' of T'. Now, by (9) and (10), $\lambda(\xi_1'') = \alpha w' = \lambda(\xi_1')$ and since G is unambiguous we have that $\xi_1'' = \xi_1'$ by Lemma 4.3.

Now $y_1, \ldots, y_r$ in (5) are the immediate descendants of x_k in T (and thus in ${}^{(n+1)}T$) and therefore, using the structural isomorphism ${}^{(n+1)}T \cong {}^{(n+1)}T'$, $f(y_1), \ldots, f(y_r)$ are the immediate descendants of $f(x_k)$ in ${}^{(n+1)}T'$ (and, since $f(x_k) \mathrel{\mathsf{L}^+} f(z)$, this is also true in T'). Moreover, $f(x_k)$ is the rightmost internal node in $f(\eta_0)$ and hence in ξ_0''. Consequently, ξ_0'' has level $h' - 1$ which is possible only if $\xi_0'' = \xi_0'$ (cf. Fact 4.1). Therefore in (7) and (8) we have $k' = k$, $r' = r$, $x_{k'}' = f(x_k)$, and $y_i' = f(y_i)$ for $i = 1, \ldots, r$. Using (9) we conclude that $p' = p$ and $\lg(\alpha) = k - 1 + r = k' - 1 + r' = n'$.

It remains to verify the case $w = \Lambda$. In such a case $T = T'$ by the Left Part Theorem and the result is an immediate consequence of Lemma 4.3. Q.E.D.

V. *Strict Deterministic and BRC Grammars*

Our main objective in this section is to establish the relationship of strict deterministic and BRC grammars. Specifically we will show that each strict deterministic language has a BRC (1, 0) grammar. First we need an additional lemma about cross sections in arbitrary trees.

LEMMA 5.1. *Let T be any tree and ξ a canonical cross section in T of level $h \geq 1$. Let x, y be two nodes in ξ such that $x \; \mathsf{L}^* \; y$ and y is internal in T. There exists a node z in T such that $z \; \Gamma \; x$ and $z \; \Gamma^+ \; y$.*

A typical form of a cross section implied by the lemma is in Figure 9. (The lemma is true under weaker assumptions on y but we shall only need just the present form.)

PROOF. The argument is an induction on the level h.

Basis. $h = 1$. Take $z = x_0$, the root of T.

Inductive step. Assume the lemma true for $h \geq 1$, and for $r \geq 1$ and $m \geq k \geq 1$ let $\xi = (x_1, \ldots, x_{k-1}, y_1, \ldots, y_r, x_{k+1}, \ldots, x_m)$ be a canonical cross section of level $h + 1$, and let $\xi_0 = (x_1, \ldots, x_k, \ldots, x_m)$ be the corresponding canonical cross section of level h (cf. Definition 4.1). Define two sets $X = \{x_1, \ldots, x_{k-1}\}$, $Y = \{y_1, \ldots, y_r\}$. Since nodes $x_{k+1}, \ldots, x_m$ are not internal, $y \in X \cup Y$. There are three cases:

Case 1. $x, y \in X$. They are both in ξ_0 and the result follows from the induction hypothesis.

Case 2. $x \in X$ and $y \in Y$. By the induction hypothesis applied to ξ_0 there is some z such that $z \; \Gamma \; x$ and $z \; \Gamma^+ \; x_k$. Hence $z \; \Gamma^+ \; y$.

Case 3. $x, y \in Y$. Take $z = x_k$. Q.E.D.

THEOREM 5.1. *Every strict deterministic grammar G is equivalent to some strict deterministic BRC $(l, 0)$ grammar G' for some $l \geq 0$.*

PROOF. Let $G = \langle V, \Sigma, P, S \rangle$ be a strict deterministic grammar with partition π.

The idea behind our construction of G' is that we add a new auxiliary letter to the righthand side of each production of G; this letter provides information about the equivalence class of the nonterminal on the left hand side of the production.

Construction. Define a new grammar $G' = \langle V \cup V', \Sigma, P', S \rangle$, where

$$V' = \mathbf{Alph}(\pi - \{\Sigma\})$$

is a set of *auxiliary letters*,

$$P' = P_1 \cup P_2,$$

$$P_1 = \{A \to \overline{[A]}\alpha \mid A \to \alpha \text{ in } P\}, \tag{1}$$

$$P_2 = \{A \to \Lambda \mid A \in V'\}. \tag{2}$$

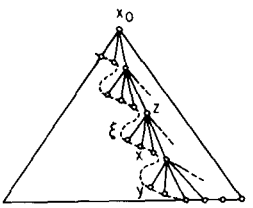

FIG. 9. The type of cross section described by Lemma 5.1

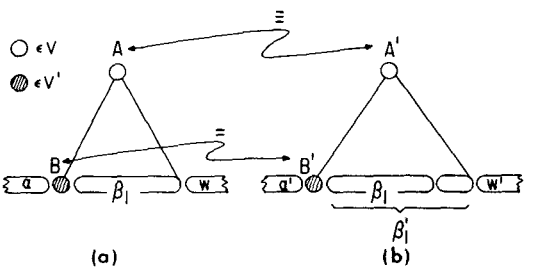

FIG. 10. Case 1: (a) handle of $\alpha\beta w$; (b) handle of $\alpha'\beta w'$

Claim 1. $L(G') = L(G)$. This is immediate from the construction.

Claim 2. G' is strict deterministic.

Proof of Claim 2. Define $\pi' = \pi \cup \{\{A\} \mid A \in V'\}$. We shall show that π' is a strict partition for G'. Clearly $\Sigma \in \pi'$ since $\Sigma \in \pi$. Let $A, A' \in V \cup V'$ and $A \equiv A' \pmod{\pi'}$. If $A \in V'$ or $A' \in V'$ then $A = A'$ and the strict deterministic condition is satisfied trivially. Let $A, A' \in V$, $A \to \alpha\beta$, and $A' \to \alpha\beta'$ in P' for some $\alpha, \beta, \beta' \in (V \cup V')^*$ and let $B = \overline{[A]} = \overline{[A']}$. If $\alpha = \Lambda$ then ${}^{(1)}\beta = B = {}^{(1)}\beta'$ and we are done. If $\alpha \neq \Lambda$ then $\alpha = B\alpha'$ for some $\alpha' \in V^*$ and we have $A \to \alpha'\beta$ and $A' \to \alpha'\beta'$ in P and since $A \equiv A' \pmod{\pi}$ the result follows from the strict determinism of G. This completes the proof of Claim 2.

Claim 3. Grammar G' is BRC$(l, 0)$ where l is the maximal length of a right-hand side of a production in G'.

Proof of Claim 3. Following Definition 3.3 let $\alpha, \alpha', \beta, \beta' \in (V \cup V')^*$, $w, w' \in \Sigma^*$, $A, A' \in (V \cup V') - \Sigma$, and $n' \geq \lg(\alpha'\beta)$. Let

$$(A \to_{G'} \beta, \lg(\alpha\beta)) \text{ be a handle of } \alpha\beta w, \tag{3}$$

let

$$(A' \to_{G'} \beta', n') \text{ be a handle of } \alpha'\beta w', \tag{4}$$

and let

$$\alpha^{(l)} = \alpha'^{(l)}. \tag{5}$$

We will distinguish four cases by the membership of A and A' in V or V'. The first case is the simplest:

Case 1. $A, A' \in V$. Define $B = \overline{[A]}$ and $B' = \overline{[A']}$. Then by (11) $\beta = B\beta_1$ and $\beta' = B'\beta_1'$ for some $\beta_1, \beta_1' \in V^*$ (cf. Figure 10). Since $\beta_1 \in V^*$, letter B (in position $\lg(\alpha') + 1$) is the rightmost occurrence of an auxiliary letter in $\alpha'\beta w'$. On the other hand, the occurrence of B' introduced by application of $A' \to_{G'} B'\beta_1'$ in the last step of the derivation of $\alpha'\beta w'$ is also the rightmost occurrence of an auxiliary letter in $\alpha'\beta w'$ (since the derivation is rightmost). Hence $B = B'$ and thus $A \equiv A' \pmod{\pi'}$. Moreover, either β is a prefix of β' (as in Figure 10) or conversely, but since G' is strict deterministic this is possible only if $\beta = \beta'$. Hence, again by the strict determinism, $A = A'$. Therefore $(A \to \beta) = (A' \to \beta')$ and $n' = \lg(\alpha'\beta)$.

We shall use an additional property of grammatical trees of G' for the remaining three cases. Let us call a node x in a grammatical tree an *auxiliary node* if $\lambda(x) \in V'$. The following observation is a direct consequence of Lemma 5.1 and of the fact that among the immediate descendants of any nonauxiliary internal node exactly the leftmost one is an auxiliary node (cf. (1)).

Observation. Let $T \in \mathfrak{I}_{G'}$ and let $\xi = (x_1, \ldots, x_r, \ldots, x_s, \ldots, x_m)$, for $1 \leq r < s \leq m$ be a canonical cross section in T where x_r and x_s are two auxiliary nodes but for $r < i < s$, x_i are not auxiliary nodes. Then there exist two distinct internal nodes z_1 and z_2 in T such that

$$z_1 \sqcap x_i \text{ for } r \leq i < s \quad \text{and} \quad z_1 \sqcap z_2 \sqcap x_s \tag{6}$$

(cf. Figure 11).

We shall proceed in our case analysis.

Case 2. $A \in V$, $A' \in V'$. Let $B = \overline{[A]}$ and $\beta = B\beta_1$ as in case 1. By (2) $\beta' = \Lambda \neq \beta$ and we seek a contradiction. Since $n' \geq \lg(\alpha'\beta)$ there exist $u, v \in \Sigma^*$ such that $uv = w'$ and $S \Rightarrow_R^* \gamma = \alpha'\beta uA'v \Rightarrow_R \alpha'\beta w'$. Consider a tree with canonical cross section ξ corresponding to canonical sentential form γ (cf. Figure 12(b)). Note that the level of ξ cannot be 0. Let x_r, x_s be two auxiliary nodes in ξ, x_r corresponding to ${}^{(1)}\beta = B$ (in position $\lg(\alpha') + 1$ in γ) and x_s corresponding to A' (in position $\lg(\alpha'\beta u) + 1$ in γ). Using the above Observation we obtain two internal nodes z_1, z_2 satisfying (6). But then we have the following two productions in G': $\lambda(z_1) \to B\beta_1 u\lambda(z_2)\delta$ in P' for some $\delta \in V^*$, and $A \to B\beta_1$ in P' from (3) (cf. Figure 12(a)). Since both right-hand sides start with

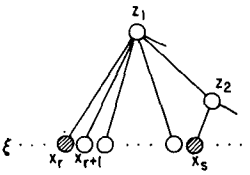

FIG. 11. An illustration of the situation in (6)

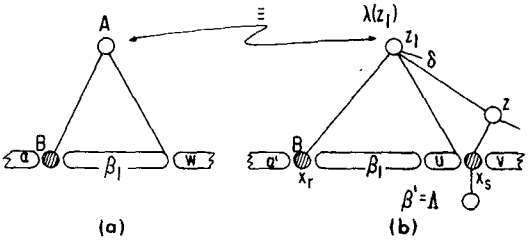

FIG. 12. Case 2: (a) handle of $\alpha\beta w$; (b) handle of $\alpha'\beta w'$ and part of the tree above it

the same letter from V' we have by (1) $\lambda(z_1) \equiv A \pmod{\pi'}$. These two productions contradict the strict determinism of G'.

Case 3. $A \in V'$, $A' \in V$. Let $B' = \overline{[A']}$ $\beta' = B'\beta_1$. This time $\beta = \Lambda \neq \beta'$ and we expect a contradiction again. We have $S \Rightarrow_R^* \alpha A w \Rightarrow_R \alpha w$, and let us consider a tree with canonical cross section ξ (of level greater than or equal to 1) corresponding to canonical sentential form $\alpha A w$ (Figure 12(a)). Since $n' \geq \lg(\alpha'\beta) = \lg(\alpha')$ the auxiliary letter B' (in position $n' - \lg(\beta_1)$ in $\alpha' w'$) occurs inside the suffix $\alpha'^{(l)}$ of α' (recall the definition of l) (Figure 13(b)). Thus $\alpha^{(l)} = \alpha'^{(l)} = \alpha_1 B' \alpha_2$ for some $\alpha_1 \in (V \cup V')^*$, $\alpha_2 \in V^*$, and we have two auxiliary nodes x_r and x_s in ξ such that $\lambda(x_r) = B'$, $\lambda(x_s) = A$. Using the Observation above we can find two nodes z_1, z_2 satisfying (6). Now we have

$$\lambda(z_1) \to B'\alpha_2\lambda(z_2)\delta \text{ in } P'$$

for some $\delta \in V^*$. The production $A' \to B'\beta_1$ can be written in the form $A' \to B'\alpha_2 w_1$, where $w_1 \in \Sigma^*$ is a prefix of w' (note that $n' \geq \lg(\alpha')$). Similarly as in the previous case we have by (1) $\lambda(z_1) \equiv A' \pmod{\pi'}$. Now either ${}^{(1)}w_1 \in \Sigma$ or $w_1 = \Lambda$, but $\lambda(z_2) \in V - \Sigma$ and we have again a contradiction of the strict determinism of G'.

Case 4. $A, A' \in V'$. Then $\beta = \beta' = \Lambda$. Since $n' \geq \lg(\alpha')$ we have for some w_1, $w_2 \in \Sigma^*$, $w_1 w_2 = w'$: $S \Rightarrow_R^* \alpha A w \Rightarrow_R \alpha w$, $S \Rightarrow_R^* \alpha' w_1 A' w_2 \Rightarrow_R \alpha' w'$.

First note that any canonical sentential form in our grammar G' which is not S or a terminal string has $\overline{[S]}$ as its leftmost letter. Assume first $\alpha = \Lambda$. Then $\alpha' = \Lambda$ (since $\alpha^{(l)} = \alpha'^{(l)}$) and $w_1 = \Lambda$ (since $w_1 \in \Sigma^*$). Thus $A = A' = \overline{[S]}$, $n' = \lg(\alpha)$, and we are done.

Next assume $\alpha \neq \Lambda$. Then we can write $\alpha^{(l)} = \alpha'^{(l)} = \alpha_1 B \alpha_2$ for some $\alpha_1 \in (V \cup V')^*$, $B \in V'$, and $\alpha_2 \in V^*$ (recall that l was defined to be the length of the longest right-hand side of a production of G'). Let ξ and ξ' be two canonical cross sections (in different trees, in general) for canonical sentential forms $\alpha A w$ and $\alpha' w_1 A' w_2$ (cf. Figure 14). Using the same arguments as in previous cases and applying our Observation to both ξ and ξ' we obtain the following relations (cf. Figure 14): $z_1 \sqcap x_r$ and $z_1 \sqcap z_2 \sqcap x_s$, $z_1' \sqcap x_r'$ and $z_1' \sqcap z_2' \sqcap x_s'$, where z_1, z_2, x_r, x_s are nodes in the tree with ξ and z_1', z_2', x_r', x_s' are nodes in the tree with ξ'. Moreover, $\lambda(x_r) = \lambda(x_r') = B$, $\lambda(x_s) = A$, and $\lambda(x_s') = A'$. Thus $\lambda(z_1) \equiv \lambda(z_1') \pmod{\pi'}$ and the following productions are in G':

$$\lambda(z_1) \to B\alpha_2\lambda(z_2)\delta \text{ in } P', \quad \lambda(z_1') \to B\alpha_2 w_1 \lambda(z_2')\delta' \text{ in } P'$$

for some $\delta, \delta' \in V^*$. Using the strict determinism of G' we obtain $w_1 = \Lambda$ and therefore $n' = \lg(\alpha')$. Further we obtain $\lambda(z_2) \equiv \lambda(z_2') \pmod{\pi'}$ and therefore $A = A'$. This concludes the proof of Claim 3 and of the theorem. Q.E.D.

COROLLARY. *Every strict deterministic grammar G is equivalent to some strict deterministic BRC (1, 0) grammar G'.*

PROOF. By the theorem, we may assume that G is a strict deterministic BRC$(l, 0)$ grammar for some $l \geq 0$. The transformations given in [11] produce a new grammar G' which is BRC(1, 0). Moreover G' is strict deterministic. The proof that these transforma-

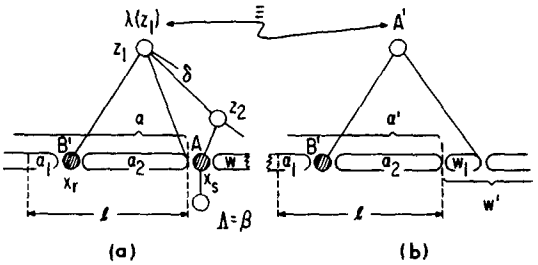

FIG. 13. Case 3: (a) handle of $\alpha\beta w = \alpha w$ and part of the tree above it; (b) handle of $\alpha' w'$

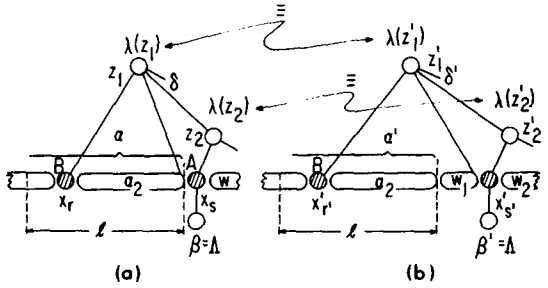

FIG. 14. Case 4: (a) handle of $\alpha\beta w = \alpha w$ and part of the tree above it; (b) handle of $\alpha' w'$ and part of the tree above it

tions preserve strict determinism is long and tedious but straightforward and will be omitted.

VI. *Deterministic = LR = BRC*

In the present section we utilize our earlier results to give rigorous proofs that the families of deterministic, LR, and BRC languages are coextensive.

One additional lemma is necessary.

LEMMA 6.1. *Let G be an LR(0) grammar of the form $G = \langle V \cup \{\$\}, \Sigma \cup \{\$\}, P, S\rangle$ where $P \subseteq N \times (V^* \cup V^*\$)$, $L(G) \subseteq \Sigma^*\$$, and there is no derivation $S \Rightarrow_R^+ S\$$ in G. Define another grammar $G' = \langle V, \Sigma, P', S\rangle$ where $P' = P_1 \cup P_2$, $P_1 = \{A \to \alpha \mid A \to \alpha$ in P and $\alpha \in V^*\}$, $P_2 = \{A \to \alpha \mid A \to \alpha\$$ in $P\}$. Then G' is an LR(1) grammar and $L(G')\$ = L(G)$.*

PROOF. Assume without loss of generality that G is reduced. Clearly there is no derivation $S \Rightarrow_R^+ S$ in G' since $S \Rightarrow_R^+ S\$$ cannot occur in G. We shall examine the handles in G'. Let $h = (A \to \beta, i)$ be a handle of γ in G' ($A \in N$, $\beta, \gamma \in V^*$, and $i \geq 0$). There are two possibilities. If $A \to \beta$ is in P_1 then h is also a handle of γ or $\gamma\$$ in G. On the other hand, if $A \to \beta$ is in P_2 then $(A \to \beta\$, i + 1)$ is a handle of $\gamma\$$ in G.

Now, following Definition 3.2, let $\alpha \in V^*$, w, $w' \in \Sigma^*$, p, $p' \in P'$, $i \geq 0$ and let $h = (p, \lg(\alpha))$ and $h' = (p', i')$ be handles of αw and $\alpha w'$ respectively. Moreover, let ${}^{(1)}w = {}^{(1)}w'$. We shall show that $h = h'$. Assume that p and p' have the form $A \to \beta$ and $A' \to \beta'$ respectively ($A, A' \in N$, $\beta, \beta' \in V^*$). We can distinguish four cases.

Case 1. $p, p' \in P_1$. Then h is also a handle of αw or $\alpha w\$$ in G and h' is also a handle of $\alpha w'$ or $\alpha w'\$$ in G. In all cases we have $h = h'$ by the LR(0) property of G.

Case 2. $p, p' \in P_2$. Then $h_1 = (A \to \beta\$, \lg(\alpha) + 1)$ is a handle of $\alpha w\$$ in G and $h_1' = (A' \to \beta'\$, i + 1)$ is a handle of $\alpha w'\$$ in G. Here ${}^{(1)}w = {}^{(1)}w'$ implies ${}^{(1)}(w\$) = {}^{(1)}(w'\$)$ and thus $h_1 = h_1'$ by the LR(0) property of G. Therefore also $h = h'$.

We shall see that the remaining two cases cannot occur. First, let us make the following observation which is a direct consequence of Definition 3.1: If $h = (p, \lg(\alpha))$ is a handle of αw and if $\alpha w'$ is a canonical sentential form (w and w' are terminal strings) then h is also a handle of $\alpha w'$. We proceed in our case analysis.

Case 3. $p \in P_1$, $p' \in P_2$. Then h is a handle of αw or $\alpha w\$$ in G and $h_1 = (A' \to \beta'\$, i + 1)$ is a handle of $\alpha w'\$$ in G. Here $\alpha w'\$$ is a canonical sentential form in G (otherwise it could not have a handle) and thus h is also a handle of $\alpha w'\$$ in G. Then by the LR(0) property $h = h_1$. But this is not possible since \$ does not occur in β.

Case 4. $p' \in P_1$, $p \in P_2$. Then h' is a handle of $\alpha w'$ or $\alpha w'\$$ and $h_1 = (A \to \beta\$, \lg(\alpha) + 1)$ is a handle of $\alpha w\$$. The form of h_1 implies that $w = \Lambda$. Since ${}^{(1)}w = {}^{(1)}w'$, also $w' = \Lambda$. Now h' is a handle of $\alpha\$$ (if it is a handle of α then it is also a handle of $\alpha\$$ since $\alpha\$$ is a canonical sentential form). But also h_1 is a handle of $\alpha\$$ and thus $h' = h_1$ which is again not possible.

We have proved that $h = h'$ and therefore G' satisfies the LR(1) property. The fact that $L(G')\$ = L(G)$ is immediate from the definition of G'. Q.E.D.

We can now prove two important consequences of our previous results.

THEOREM 6.1 (equivalent to Proposition 5). *Any deterministic language has an LR(1) grammar.*

FIRST VARIANT OF THE PROOF (using Theorem 4.1). Let $L \subseteq \Sigma^*$ and $L \in \Delta_0$. Then $L\$ \in \Delta_2$ and by [17, Th. 3.3] there exists a strict deterministic grammar $G = \langle V \cup \{\$\}, \Sigma \cup \{\$\}, P, S\rangle$ such that $L(G) = L\$$. Using [17, Ths. 2.1, 2.5] we may assume without loss of generality that G is reduced and Λ-free (note that $L(G) \neq \{\Lambda\}$). By Theorem 4.1, G is LR(0). We shall check that G satisfies assumptions of Lemma 6.1. First, $P \subseteq N \times (V^* \cup V^*\$)$, or in other words, each production in P has the form $A \to \alpha$ or $A \to \alpha\$$ where $A \in N$ and $\alpha \in V^*$. For, if $A \to \alpha\$\beta$ were in P for some $\beta \neq \Lambda$, the fact that G is reduced and Λ-free would imply that $u\$v \in L\$$ for some u, $v \in \Sigma^*$ and $v \neq \Lambda$ which is not possible. Second, $S \Rightarrow_R^+ S\$$ is impossible in G since that would contradict [17, Th. 2.3] and the strict determinism of G. Now we can apply Lemma 6.1 and obtain an LR(1) grammar G' such that $L(G') = L$. Q.E.D.

SECOND VARIANT OF THE PROOF (using Theorem 5.1). The proof proceeds in the same way as in the first variant until we obtain a reduced and Λ-free strict deterministic grammar G such that $L(G) = L\$$. This time we use Theorem 5.1 to obtain an equivalent BRC(1,0) grammar G'. This grammar is trivially LR(0) (cf. Fact 3.1). Moreover, we can obtain G' in a form which satisfies the assumptions of Lemma 6.1. For, the construction of BRC(1, 0) grammar in the proof of Theorem 5.1 preserves the property that \$ can occur only as a last letter of any production. Also, G' is strict deterministic and hence $S \Rightarrow_R^+ S\$$ cannot occur in G. Now we obtain an LR(1) grammar for L using Lemma 6.1 similarly as in the first variant. Q.E.D.

THEOREM 6.2 (equivalent to Proposition 8). *Any LR(k) grammar ($k > 0$) is equivalent to a BRC(1, k) grammar.*

PROOF. By Proposition 4 if G is LR(k) then $L(G) \in \Delta_0$. Hence $L(G)\$ \in \Delta_2$ and by [17, Th. 3.3] there is a strict deterministic grammar G' such that $L(G') = L(G)\$$. By Theorem 5.1 there is a BRC(1, 0) grammar G'' equivalent to G'. Using the arguments from the proof of Theorem 6.1 (the second variant) we can impose on G'' the assumptions of Lemma 6.1. The same construction as in Lemma 6.1 will yield a BRC(1, 1) grammar equivalent to G (this can be verified by a trivial repetition of the proof of Lemma 6.1 for the case of BRC property). Finally, any BRC(1, 1) grammar is also a BRC(1, k) in a trivial way (cf. Fact 3.1). Q.E.D.

These results may be combined as follows.

THEOREM 6.3. *The following families of languages are coextensive.* $\Delta_0 = LR = BRC$.

PROOF. We shall argue that $\Delta_0 =$ LR and LR = BRC. $LR \subseteq \Delta_0$ by Proposition 4 while $\Delta_0 \subseteq$ LR by Theorem 6.1. On the other hand, BRC $\subseteq$ LR by Fact 3.1 of Section III while Theorem 6.2 implies that LR $\subseteq$ BRC. Q.E.D.

REFERENCES

1. AHO, A. V., AND ULLMAN, J. D. *The Theory of Parsing, Translating, and Compiling: Vol. I, Parsing; Vol. II, Compiling.* Prentice-Hall, Englewood Cliffs, N.J., 1972.
2. AHO, A. V., DENNING, P. J., AND ULLMAN, J. D. Weak and mixed strategy precedence parsing. *J. ACM 19*, 2 (April 1972), 225–243.

3. DeRemer, F. L. Simple LR(k) grammars. *Comm. ACM 14*, 7 (July 1971), 453–460.
4. Earley, J. An efficient context-free parsing algorithm. *Comm. ACM 13*, 2 (Feb. 1970), 94–102.
5. Floyd, R. W. Bounded context syntactic analysis. *Comm. ACM. 7*, 2 (Feb. 1964), 62–66.
6. Geller, M. M., and Harrison, M. A. Strict deterministic versus LR(0) parsing. Conf. Rec. of the ACM Symp. on Principles of Programming Languages, 1973, pp. 22–31.
7. Geller, M. M., and Harrison, M. A. Characterizations of LR(0) languages. Conf. Rec. of the 14th Annual Symp. on Switching and Automata Theory, 73CH0-786-4-C, 1973, pp. 103–108.
8. Ginsburg, S. *The Mathematical Theory of Context-Free Languages*. McGraw-Hill, New York, 1966.
9. Ginsburg, S., and Greibach, S. A. Deterministic context-free languages. *Inform. and Contr. 9* (1966), 602–648.
10. Graham, S. L. Precedence languages and bounded right context languages. Ph.D. Thesis and Tech. Rep. CS-71-223, Stanford U., 1971.
11. Graham, S. L. On bounded right context languages and grammars. *SIAM J. of Computing* (to appear).
12. Gray, J. N., and Harrison, M. A. On the covering and reduction problems for context free grammars. *J. ACM 19*, 4 (Oct. 1972), 675–698.
13. Gray, J. N., and Harrison, M. A. Canonical precedence schemes. *J. ACM 20*, 2 (April 1973), 214–234.
14. Gries, D. *Compiler Construction for Digital Computers*. Wiley, New York, 1971.
15. Haines, L. H. Generation and recognition of formal languages. Ph.D. Thesis, M.I.T., 1965.
16. Harrison, M. A. On the relation between grammars and automata. In *Advances of Information Sciences 4*, J. T. Tou, Ed., Plenum Press, New York, 1972, pp. 39–92.
17. Harrison, M. A., and Havel, I. M. Strict deterministic grammars. *J. Comput. and Syst. Sci. 7* (1973), 237–277.
18. Hopcroft, J. E., and Ullman, J. D. *Formal Languages and Their Relation to Automata*. Addison-Wesley, Reading, Mass., 1969.
19. Knuth, D. E. On the translation of languages from left to right. *Inform. and Contr. 8* (1965), 607–639.
20. Knuth, D. E. *The Art of Computer Programming, Vol. I*. Addison-Wesley, Reading, Mass., 1968.
21. Korenjak, A. J. A practical method for constructing LR(k) processors. *Comm. ACM 12*, 11 (Nov. 1969), 613–623.
22. Lehmann, D. LR(k) grammars and deterministic languages. *Israel J. of Math. 10* (1971), 526–530.
23. Ogden, W. F. Intercalation theorems for pushdown store and stack languages. Ph.D. Thesis, Stanford U., 1968.
24. Salomaa, A. *Formal Languages*. Academic Press, New York, 1973.
25. Ullman, J. D. Applications of language theory to compiler design. In *Theoretical Computer Science*, A. V. Aho, Ed., Prentice-Hall, Englewood Cliffs, N. J., 1972.
26. Valiant, L. G. General context-free recognition in less than cubic time. Tech. Rep., Dep. of Computer Science, Carnegie-Mellon U., Pittsburgh, Pa., Jan. 1974.

RECEIVED JUNE 1972; REVISED NOVEMBER 1973