Indexed Grammars—An Extension of Context-Free Grammars

ALFRED V. AHO

Bell Telephone Laboratories, Inc., Murray Hill, New Jersey

ABSTRACT. A new type of grammar for generating formal languages, called an indexed grammar, is presented. An indexed grammar is an extension of a context-free grammar, and the class of languages generated by indexed grammars has closure properties and decidability results similar to those for context-free languages. The class of languages generated by indexed grammars properly includes all context-free languages and is a proper subset of the class of context-sensitive languages. Several subclasses of indexed grammars generate interesting classes of languages.

KEY WORDS AND PHRASES: formal grammar, formal language, language theory, automata theory, phrase-structure grammar, phrase-structure language, syntactic specification, context-free grammar, context-sensitive grammar, stack automata

CR CATEGORIES: 5.22, 5.23

1. *Introduction*

A language, whether a natural language such as English or a programming language such as ALGOL, in an abstract sense can be considered to be a set of sentences.[1] One criterion for the syntactic specification of a language is that, invariably, a finite representation is required for an infinite class of sentences. There are several different approaches toward such a specification. In one approach a finite generative device, called a grammar, is used to describe the syntactic structure of a language. Another approach is to specify a device or algorithm for recognizing well-formed sentences. In this approach the language consists of all sentences recognized by the device or algorithm. A third possible method for the specification of a language would be to specify a set of properties and then consider a language to be a set of words obeying these properties.

In this paper a new type of grammar, called an indexed grammar, is defined. The language generated by an indexed grammar is called an indexed language. It is shown that the class of indexed languages properly includes all context-free languages and yet is a proper subset of the class of context-sensitive languages. Moreover, indexed languages resemble context-free languages in that many of the closure properties and decidability results for both classes of languages are the same.

Recently, there has been a great deal of interest in defining classes of recursive languages larger than the class of context-free languages. Programmed grammars are a recent example of a grammatical definition of such a class [16], and various

A summary of this paper was presented at the IEEE Eighth Annual Symposium on Switching and Automata Theory, October 1967.

[1] The words "sentence," "word," and "program" will be used synonymously as being an element in a language.

Journal of the Association for Computing Machinery, Vol. 15, No. 4, October 1968, pp. 647–671.

kinds of stack automata provide a device-oriented definition of such a class [8, 9, 14].

Part of this interest in larger classes of languages stems from the inadequacy of context-free grammars in specifying all of the syntactic structures found in many modern-day algorithmic programming languages, such as ALGOL [5]. For any automatic compiler writing system to be a practical reality, at the very least, some reasonable scheme for representing the complete syntactic structure of a language must be available. At present, however, there does not appear to be a reasonable grammar or automaton model less powerful than a context-sensitive grammar or linear-bounded automaton which is capable of providing such a specification.

In Section 2 of this paper the definitions of indexed grammar and indexed language are given. The basic closure properties and decidability results for indexed grammars and languages are presented in Sections 3 and 4. It is shown that the class of indexed languages possesses closure properties enjoyed by many other well-known classes of languages and, in fact, qualifies as an example of an abstract family of languages as defined in [7].

In Section 5 it is shown that the class of indexed languages includes all context-free languages and some context-sensitive languages, but yet is a proper subset of the class of context-sensitive languages.

Several restricted forms of indexed grammars capable of generating interesting subclasses of indexed languages are presented in Section 6. Two such classes of restricted forms of indexed grammars are equivalent in generative capability to the class of context-free grammars.

Each of the four classes of grammars in the Chomsky hierarchy, viz. type 0, context-sensitive, context-free, and regular grammars, generates a family of languages which corresponds exactly to that family of sets recognizable by a particular class of automata, viz. Turing machines, linear bounded automata, one-way nondeterministic pushdown automata, and finite automata, respectively [3]. Indexed languages also enjoy this dual representation. A new automaton model, called a nested stack automaton, provides this additional characterization, in the sense that a set is recognized by a (one-way, nondeterministic) nested stack automaton if and only if the set is an indexed language [1].

Both the pushdown automaton and the recently defined stack automaton are special cases of a nested stack automaton. Because of this characterization for indexed languages in terms of languages accepted by a class of devices which are a natural generalization of both pushdown automata and stack automata, we feel that the class of indexed languages assumes a natural position in the Chomsky hierarchy of languages.

Nested stack automata and the equivalence of indexed languages with (one-way, nondeterministic) nested stack automaton languages are discussed in [1].

2. *Indexed Grammars*

In this section the basic definition of an indexed grammar is presented. A few simple examples of indexed grammars are given, and it is shown how a derivation tree can be associated with the derivation of a sentence in an indexed grammar.

Definition. An *indexed grammar* is a 5-tuple, $G = (N, T, F, P, S)$, in which:

(a) N is a finite nonempty set of symbols called the *nonterminal alphabet.*

(b) T is a finite set of symbols called the *terminal alphabet.*

(c) F is a finite set each element of which is a finite set of ordered pairs of the form (A, χ), where A is in N and χ is in $(N \cup T)^*$.[2] An element f in F is called an *index* or *flag.* An ordered pair (A, χ) in f is written $A \to \chi$ and is called an *index production* contained in f.

(d) P is a finite set of ordered pairs of the form (A, α) with A in N and α in $(NF^* \cup T)^*$. Such a pair is usually written $A \to \alpha$; it is called a *production.*

(e) S, the *sentence symbol,* is a distinguished symbol in N.

Let $G = (N, T, F, P, S)$ be an indexed grammar. Roughly speaking, a derivation in G is a sequence of strings, $\alpha_1, \alpha_2, \cdots, \alpha_n$ with each α_i in $(NF^* \cup T)^*$, in which α_{i+1} is derived from α_i by the application of either one production or one index production. Except for the manner in which indices and strings of indices are manipulated, a derivation in G proceeds in exactly the same manner as in a context-free grammar.

However, in a derivation in G each nonterminal in N can be immediately followed by a string of indices in F^*. A string of symbols of the form $A\zeta$, where A is in N and ζ is in F^*, is called an *indexed nonterminal.* A string α in $(NF^* \cup T)^*$ is often referred to as a *sentential form.*

Formally, a sentential form α is said to *directly generate* (or *directly derive*) a sentential form β, written $\alpha \underset{G}{\Rightarrow} \beta$, if either:[3]

1. $\alpha = \gamma A \zeta \delta$,
   $A \to X_1\eta_1X_2\eta_2 \cdots X_k\eta_k$ is a production in P,
   $\beta = \gamma X_1\theta_1X_2\theta_2 \cdots X_k\theta_k\delta$ where, for $1 \leq i \leq k$, $\theta_i = \eta_i\zeta$ if X_i is in N, or $\theta_i = \epsilon$ if X_i is in T;

or

2. $\alpha = \gamma A f \zeta \delta$,
   $A \to X_1X_2 \cdots X_k$ is an index production in the index f,
   $\beta = \gamma X_1\theta_1X_2\theta_2 \cdots X_k\theta_k\delta$ where, for $1 \leq i \leq k$, $\theta_i = \zeta$ if X_i is in N, or $\theta_i = \epsilon$ if X_i is in T.

In both cases 1 and 2, the nonterminal A is said to be *expanded.* In case 2 the index f is said to be *consumed* by the nonterminal A. Observe that a terminal symbol can never have a string of indices immediately following it in any line of a derivation.

For example, if $A\zeta$ is an indexed nonterminal and $A \to aB\eta C\theta b$ is a production in P, then $A\zeta$ can directly derive the sentential form $aB\eta\zeta C\theta\zeta b$ in G. Here the index

[2] Let T be a set. T^+ is the set of all finite-length strings of elements of T, excluding ϵ, the empty string. $T^* = T^+ \cup \{\epsilon\}$.

[3] Whenever the reference is to a grammar $G = (N, T, F, P, S)$, unless otherwise stated, the following symbolic conventions are used:
(1) A, B, C, D are in N.
(2) a and b are in $T \cup \{\epsilon\}$ where ϵ is the null word.
(3) w and x are in T^*.
(4) χ and ω are in $(N \cup T)^*$.
(5) f, g, h are in F.
(6) ζ, η, θ are in F^*.
(7) $\alpha, \beta, \gamma, \delta$ are in $(NF^* \cup T)^*$.
(8) X is in $N \cup T$.

string ζ distributes over the indexed nonterminals $B\eta$ and $C\theta$ but not over the terminal symbols a and b in the production.

If $Af\zeta$ is an indexed nonterminal and the index f contains the index production $A \to aBC$, then $Af\zeta$ can directly derive the sentential form $aB\zeta C\zeta$ in G. Notice that in such a step the index f is removed from the next line of the derivation, and the index string ζ distributes over the nonterminals B and C but not over the terminal symbol a.

The reflexive and transitive closure of the relation $\underset{G}{\to}$ on $(NF^* \cup T)^*$ is defined as follows:

1. $\alpha \overset{0}{\underset{G}{\to}} \alpha$.
2. For $n \geq 0$, $\alpha \overset{n+1}{\underset{G}{\to}} \gamma$ if, for some β, $\alpha \overset{n}{\underset{G}{\to}} \beta$ and $\beta \underset{G}{\to} \gamma$.
3. $\alpha \overset{*}{\underset{G}{\to}} \beta$ iff $\alpha \overset{i}{\underset{G}{\to}} \beta$ for some $i \geq 0$.

We say that α *generates* or *derives* β if $\alpha \overset{*}{\underset{G}{\to}} \beta$. A sequence of sentential forms

$$\alpha_0 , \alpha_1 , \alpha_2 , \cdots , \alpha_n \tag{1.1}$$

such that $\alpha_i \underset{G}{\to} \alpha_{i+1}$, for $0 \leq i < n$, is called a *derivation* (of length n) of α_n from α_0 in G. If in each sentential form α_i the leftmost nonterminal is expanded in order to directly generate the string α_{i+1}, for $0 \leq i < n$, then the sequence of strings (1.1) is called a *leftmost derivation*. It should be clear that if there exists a derivation of β from α, then there always exists a leftmost derivation of β from α. Unless it is otherwise specified, the word "derivation" implies a leftmost derivation.

The language generated by an indexed grammar, $G = (N, T, F, P, S)$, denoted $L(G)$, is called an *indexed language* and is defined to be the set $\left\{w \text{ in } T^* \mid S \overset{*}{\underset{G}{\to}} w\right\}$.

Henceforth, the subscript G on the relations "$\underset{G}{\to}$" and "$\overset{*}{\underset{G}{\to}}$" is dropped whenever G is clearly understood.

Example 1. Let $G_1 = (\{S, A, B\}, \{a, b, c\}, \{f, g\}, P, S)$, where P consists of the productions $S \to aAfc$, $A \to aAgc$, and $A \to B$, and $f = [B \to b]$, $g = [B \to bB]$.

Applying the first production once, the second production $n - 1$ times, $n \geq 1$, and then the third production once, we have

$$S \to aAfc \to aaAgfcc \to \cdots \to a^nAg^{n-1}fc^n \to a^nBg^{n-1}fc^n.$$

Then, expanding B by consuming indices, we have

$$Bg^{n-1}f \to bBg^{n-2}f \to \cdots \to b^{n-1}Bf \to b^n.$$

Thus, $S \overset{2n+1}{\longrightarrow} a^nb^nc^n$. Moreover, a derivation in G_1 is "deterministic" except for the decision to expand A to B. Thus, it should be clear that the only strings of terminals which can be derived from S in G_1 are all of the form $a^nb^nc^n$, $n \geq 1$. Consequently, $L(G_1) = \{a^nb^nc^n \mid n \geq 1\}$.

As with context-free grammars the structural description of a word in an indexed language can be described in terms of a derivation tree or parsing diagram. A derivation tree in an indexed grammar $G = (N, T, F, P, S)$ is a tree with labeled nodes. Each node consists of a tuple of positive integers of the form $(i_1 , i_2 , \cdots , i_r)$,

$r \geq 1$, and the directed lines of the tree are ordered pairs of nodes of the form $((i_1, \cdots, i_r), (i_1, \cdots, i_r, i_{r+1}))$. Each node of the derivation tree is labeled by an indexed nonterminal or terminal symbol in $NF^* \cup T$.

Let l be the maximum number of nonterminals and terminals on the right-hand side of any production in P. With each node $n = (i_1, i_2, \cdots, i_r)$ with $1 \leq i_j \leq l$, we can associate the l-ary fraction $0.i_1 i_2 \cdots i_r$, which we call the node number for n. We say that $n_1 < n_2$ (or $n_1 \leq n_2$) if the node number for n_1 is less than (or less than or equal to) that for n_2. The relation "$\leq$" is a simple order on the set of nodes.

Suppose D is derivation tree and $\{\alpha_1 n_{i_1}, \alpha_2 n_{i_2}, \cdots, \alpha_k n_{i_k}\}$ is the set of labeled leaves[4] of D arranged such that $n_{i_j} < n_{i_{j+1}}$ for $1 \leq j < k$. Each node label, α_i, is in $NF^* \cup T$. The string $\alpha_1 \alpha_2 \cdots \alpha_k$ is called the *yield* of D.

A derivation tree in an indexed grammar $G = (N, T, F, P, S)$ with root $A\zeta_0$, A in N and ζ_0 in F^*, is a set of labeled nodes formally defined as follows:

1. The set containing exactly the labeled node $A\zeta_0$ (1), the root of the tree, is a derivation tree.

2. Suppose D is a derivation tree with yield $\beta B\zeta\gamma$ where β and γ are in $(NF^* \cup T)^*$ and $B\zeta$ is the node label of leaf $(i_1, i_2, \cdots, i_r)$. If $B \to X_1\eta_1 X_2\eta_2 \cdots X_k\eta_k$ is in P, then $D' = D \cup \{X_1\theta_1 m_1, X_2\theta_2 m_2, \cdots, X_k\theta_k m_k\}$ is a derivation tree where, for $1 \leq j \leq k$, $\theta_j = \eta_j\zeta$ if X_j is in N, or $\theta_j = \epsilon$ otherwise, and $m_j = (i_1, i_2, \cdots, i_r, j)$. The yield of D' is $\beta X_1\theta_1 X_2\theta_2 \cdots X_k\theta_k\gamma$.

3. Suppose D is a derivation tree with yield $\beta Bf\zeta\gamma$ where $Bf\zeta$ is the node label of leaf $(i_1, i_2, \cdots, i_n)$. If f contains $B \to X_1 X_2 \cdots X_k$, then $D' = D \cup \{X_1\theta_1 m_1, X_2\theta_2 m_2, \cdots, X_k\theta_k m_k\}$ is a derivation tree where, for $i \leq j \leq k$, $\theta_j = \zeta$ if X_j is in N, or $\theta_j = \epsilon$ otherwise, and $m_j = (i_1, i_2, \cdots, i_r, j)$. The yield of D' is $\beta X_1\theta_1 X_2\theta_2 \cdots X_k\theta_k\gamma$.

4. D is a derivation tree in G with root $A\zeta_0$ if and only if its being so follows from a finite number of applications of 1, 2, and 3.

It should be clear that, given an indexed grammar $G = (N, T, F, P, S)$, for each derivation in G there exists a corresponding derivation tree, and for each derivation tree in G there exists at least one derivation (and exactly one leftmost derivation).

THEOREM 2.1 *Given an indexed grammar $G = (N, T, F, P, S)$, $A\zeta \overset{*}{\underset{G}{\to}} \alpha$ if and only if there exists a derivation tree in G with root $A\zeta$ and yield α.*

The proof of Theorem 2.1 is similar to that for context-free grammars.

We conclude this section with an example of a derivation tree in an indexed grammar. For notational convenience each node of the derivation tree is represented only by its node label.

Example 2. Let $G_2 = (\{S, T, A, B, C\}, \{a, b\}, \{f, g\}, P, S)$, where P contains the productions

$$S \to Tf, \qquad T \to Tg, \qquad T \to ABA,$$

and where

$$f = [A \to a, B \to b, C \to b], \qquad g = [A \to aA, B \to bBCC, C \to bC].$$

[4] A leaf is a node $(i_1, \cdots, i_s)$ of D such that there is no node in D of the form $(i_1, \cdots, i_s, j)$ for any j.

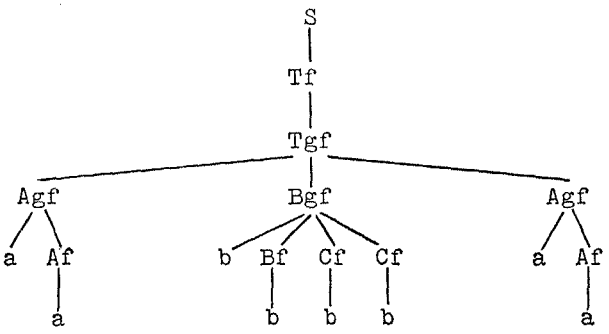

FIG. 1. Derivation tree for *aabbbbaa*

It is easy to show that

$$L(G_2) = \{a^n b^{n^2} a^n \mid n \geq 1\}.$$

The derivation tree corresponding to the derivation of $a^2b^4a^2$ is shown in Figure 1.

3. *Closure Properties*

A *class of languages* can be formally considered to be a pair $(T, \mathfrak{L})$, or $\mathfrak{L}$ whenever T is understood, where

(a) T is a countably infinite set of symbols,

(b) $\mathfrak{L}$ is a collection of subsets of T^*,

(c) For each L in $\mathfrak{L}$, there exists a finite subset T_1 of T such that $L \subseteq T_1^*$.

A large number of the classes of languages studied in language theory have many properties that are common to each class. A number of the closure properties enjoyed by many well-known classes of languages were selected in [7] as an axiom system for defining abstract families of languages. Specifically, a *full abstract family of languages* (full AFL) is a class of languages closed under the operations of union, concatenation, Kleene closure (*), homomorphism, inverse homomorphism, and intersection with regular sets. These properties,[5] in turn, imply a number of other closure properties including closure under left and right quotient by a regular set and closure under Init, Fin, and Sub [7].

In this section we show that the class of indexed languages qualifies as a full AFL and thus possesses all properties enjoyed by full AFL's. In addition, we show that indexed languages are also closed under the operations of word reversal and full substitution.

Before doing so, we use some preliminary results which make the representation of an indexed grammar notationally more convenient.

Two grammars G_1 and G_2 are said to be *equivalent* if $L(G_1) = L(G_2)$.

An indexed grammar $G = (N, T, F, P, S)$ is said to be in *reduced form* if:

(a) Each index production in each index in F is of the form $A \rightarrow B$, where both A and B are in N.

(b) Each production in P is of one of the forms

(1) $A \rightarrow BC$,

[5] Not all of the AFL operations themselves are independent. See [12] for details.

(2) $A \to Bf$, or

(3) $A \to a$,

where A, B, C are in N, f is in F, and a is in $T \cup \{\epsilon\}$.

THEOREM 3.1. *Given an indexed grammar $G = (N, T, F, P, S)$, an equivalent reduced form indexed grammar $G' = (N', T, F', P', S)$ can be effectively constructed from G.*

PROOF.

1. Suppose f is an index in F. A corresponding *reduced index* f' in F' is constructed as follows:

(a) If $A \to B$ is in f, then $A \to B$ is also in f'.

(b) If $A \to \chi$ is in f, where χ is not a single nonterminal, then $A \to C$ is in f' and $C \to \chi$ is in P', where C is a new nonterminal not in N.

2. Add to P' all productions in P replacing each index in F by the corresponding reduced index in F'.

3. In P' replace each production of the form $A \to X_1\eta_1 \cdots X_k\eta_k$ with $k \geq 2$, where η_i is a string of reduced indices, with the following productions:

$$A \to B_1C_1,$$

or, if $k = 2$,

$$\begin{aligned} A &\to B_1B_2, \\ C_i &\to B_{i+1}C_{i+1} \quad \text{for} \quad 1 \leq i \leq k-2, \\ C_{k-1} &\to B_{k-1}B_k. \end{aligned}$$

Here B_i and C_i are new symbols not in N, for $1 \leq i \leq k$, except as noted in (d) below. For each $i = 1, \cdots, k$:

(a) If $\eta_i = f_1' \cdots f_r'$ with $r \geq 2$, then add to P' the productions:

$$\begin{aligned} B_i &\to D_if_r', \\ D_j &\to D_{j-1}f_{j-1}' \quad \text{for} \quad 3 \leq j \leq r, \\ D_2 &\to X_if_1'. \end{aligned}$$

Again, D_j is a new symbol for $2 \leq j \leq r$.

(b) If $\eta_i = f'$, add to P the production

$$B_i \to X_if'.$$

(c) If X_i is a terminal symbol, add to P the production

$$B_i \to X_i.$$

(d) Otherwise, $B_i = X_i$. (This is the case where X_i is a nonterminal not followed by any indices.)

4. In P', each production of the form $A \to Bf_1 \cdots f_r'$ with $r \geq 2$ is replaced by a set of productions similar to those in 3(a).

5. The productions now in P' which are not of reduced form are those of the form $A \to B$. Call such a production a type 1 production. All type 1 productions are removed from P'. However, if $A \overset{*}{\underset{G}{\to}} B$ by type 1 productions only and $B \neq A$, then, for each production in P' of the form (i) $B \to CD$, (ii) $B \to Cf'$, or (iii) $B \to a$, add to P' the production (i) $A \to CD$, (ii) $A \to Cf'$, or (iii) $A \to a$, respectively.

Let $G' = (N', T, F', P', S)$ be the grammar constructed from G according to rules 1–5 given above, where N' is that set of nonterminals appearing in P'. G' is a reduced form indexed grammar, and a straightforward proof by induction on the length of a derivation yields that $Af_1 \cdots f_r \overset{*}{\underset{G}{\rightarrow}} w$ if and only if $Af_1' \cdots f_r' \overset{*}{\underset{G}{\rightarrow}} w$, where $r \geq 0$, A is in N', and f_i' is the reduced index constructed from f_i. From this, it immediately follows that $L(G') = L(G)$.

LEMMA 3.1. *The class of indexed languages is closed under the operations of union, concatenation, and Kleene closure.*

The proof of Lemma 3.1 is obvious.

The concept of a "finite" transducer mapping has many important applications in language theory.

Definition. A *nondeterministic finite transducer* (NFT, for short) is a 6-tuple $M = (Q, T, \Sigma, \delta, q_0, F)$ where:

(a) Q, T, Σ are finite sets of *states*, *input* symbols, and *output* symbols, respectively.

(b) δ is a mapping from $Q \times (T \cup \{\epsilon\})$ into finite subsets of $Q \times \Sigma^*$.

(c) q_0, in Q, is the *initial state*.

(d) $F \subseteq Q$ is the set of *final states*.

If δ is a mapping from $Q \times T$ into $Q \times \Sigma^*$, then M is a *deterministic* finite transducer (DFT).

$\hat{\delta}$ will be an extended mapping from $Q \times T^*$ into subsets of $Q \times \Sigma^*$ defined as follows:

(i) $\hat{\delta}(q, \epsilon)$ contains (q, ϵ).

(ii) If $\hat{\delta}(q_1, w)$ contains (q_2, x) and $\delta(q_2, a)$ contains (q_3, y), then $\hat{\delta}(q_1, wa)$ contains (q_3, xy) for w in T^*, a in $T \cup \{\epsilon\}$, x and y in Σ^*.

An NFT $M = (Q, T, \Sigma, \delta, q_0, F)$ can induce a mapping on a language $L \subseteq T^*$ in the following manner. For w in T^*, $M(w) = \{x \mid \hat{\delta}(q_0, w)$ contains (p, x) for some p in $F\}$. $M(L) = \bigcup_{w \text{ in } L} M(w)$.

We can also define an inverse NFT mapping. Let $M^{-1}(x) = \{w \mid M(w) = x\}$. Then, $M^{-1}(L') = \bigcup_{x \text{ in } L'} M^{-1}(x)$.

The definition of an NFT is sufficiently general for every inverse NFT mapping to be an NFT mapping. That is, given an NFT, $M = (Q, T, \Sigma, \delta, q_0, F)$, an NFT M' can be effectively constructed from M such that $M'(L) = M^{-1}(L)$ for any $L \subseteq \Sigma^*$.

LEMMA 3.2. *The class of indexed languages is closed under NFT mappings.*

PROOF. Let $G = (N, T, F, P, S)$ be a reduced form indexed grammar and let $M = (Q, T, \Sigma, \delta, q_0, K)$ be an NFT. We will construct an indexed grammar $G' = (N', \Sigma, F', P', S')$ such that $L(G') = M(L(G))$.

The nonterminals in N' will be of the form (p, X, q), where p and q are in Q and X is in $N \cup T \cup \{\epsilon\}$. The set of productions P' is constructed as follows.

1. If $A \rightarrow BC$ is in P, then P' contains the set of productions $(p, A, q) \rightarrow (p, B, r)\ (r, C, q)$ for all p, q, r in Q.

2. If $A \rightarrow Bf$ is in P, then P' contains the set of productions

$$(p, A, q) \rightarrow (p, B, q)f'$$

for all p, q in Q. The index f' contains the index productions $(r, C, s) \rightarrow (r, D, s)$ for all r, s in Q, if and only if f contains the index production $C \rightarrow D$.

3. If $A \to a$ is in P, then P' contains the set of productions $(p, A, q) \to (p, a, q)$ for all p, q in Q.

4. P' also contains the productions

$$(p, a, q) \to (p, a, r)\ (r, \epsilon, q),$$
$$(p, a, q) \to (p, \epsilon, r)\ (r, a, q),$$
$$(p, \epsilon, q) \to (p, \epsilon, r)\ (r, \epsilon, q),$$

for all a in T and p, q, r in Q.

5. P' contains the terminal production $(p, a, q) \to x$ if $\delta(p, a)$ contains (q, x) for a in $T \cup \{\epsilon\}$, x in Σ^*.

6. Finally, P' contains the initial productions $S' \to (q_0, S, p)$ for all p in K.

We now show that $(p, A, q)f_1' \cdots f_j' \overset{*}{\underset{G}{\to}} x$, $j \geq 0$, x in Σ^*, if and only if $Af_1 \cdots f_j \overset{*}{\underset{G}{\to}} w$ and $\hat{\delta}(p, w)$ contains (q, x) for some w in T^*. First, we show that:

(*) If $(p, A, q)f_1' \cdots f_j' \overset{k}{\underset{G'}{\to}} x$ is any derivation of length k, then $Af_1 \cdots f_j \overset{*}{\underset{G}{\to}} w$ and $\hat{\delta}(p, w)$ contains (q, x) for some w in T^*.

The proof of (*) will be by induction on k, the length of a derivation in G'. (*) is vacuously true for derivations of length 1.

Now suppose (*) is true for all $k < m$ with $m > 1$ and consider a derivation $(p, A, q)f_1' \cdots f_j' \overset{m}{\underset{G}{\to}} x$. Such a derivation can be of one of four following forms.

(i) $(p, A, q)f_1' \cdots f_j' \underset{G'}{\to} (p, B, r)f_1' \cdots f_j'(r, C, q)f_1' \cdots f_j'$

$(p, B, r)f_1' \cdots f_j' \overset{m_1}{\underset{G'}{\to}} x_1$ where $m_1 < m$,

$(r, C, q)f_1' \cdots f_j' \overset{m_2}{\underset{G'}{\to}} x_2$ where $m_2 < m$.

In this case, we have the following derivation in G:

$$Af_1 \cdots f_j \underset{G}{\to} Bf_1 \cdots f_jCf_1 \cdots f_j$$
$$\overset{*}{\underset{G}{\to}} w_1Cf_1 \cdots f_j$$
$$\overset{*}{\underset{G}{\to}} w_1w_2 .$$

Also, from the inductive hypothesis $\hat{\delta}(p, w_1)$ contains (r, x_1) and $\hat{\delta}(r, w_2)$ contains (q, x_2). Hence $\hat{\delta}(p, w)$ contains (q, x), where $w = w_1w_2$ and $x = x_1x_2$.

(ii) $(p, A, q)f_1' \cdots f_j' \underset{G'}{\to} (p, B, q)f'f_1' \cdots f_j'$
$\overset{m'}{\underset{G'}{\to}} x$ with $m' < m$.

In G, we have the derivation

$$Af_1 \cdots f_j \underset{G}{\to} Bff_1 \cdots f_j$$
$$\overset{*}{\underset{G}{\to}} w,$$

and from the inductive hypothesis, $\hat{\delta}(p, w)$ contains (q, x).

(iii) $(p, A, q)f_1' \cdots f_j' \underset{G'}{\to} (p, a, q)f_1' \cdots f_j'$
$\overset{m'}{\underset{G'}{\to}} x$ with $m' < m$.

Here, the indices disappear when terminals are introduced.

It is straightforward to show that $(p, a, q) \xrightarrow[G']{*} x$, a in $T \cup \{\epsilon\}$ if and only if $\hat{\delta}(p, a)$ contains (q, x).

Consequently, in G we have: $Af_1 \cdots f_j \xrightarrow[G]{} a$ and $\hat{\delta}(p, a)$ contains (q, x).

(iv) $(p, A, q)f_1'f_2' \cdots f_j' \xrightarrow[G']{} (p, B, q)f_2' \cdots f_j'$

$$\xrightarrow[G']{m'} x \quad \text{with } m' < m.$$

Here, the index f_1' contains the index production $(p, A, q) \rightarrow (p, B, q)$, and is consumed in the first line of the derivation.

In G we have the derivation

$$Af_1f_2 \cdots f_j \xrightarrow[G]{} Bf_2 \cdots f_j$$

$$\xrightarrow[G]{*} w \quad \text{for some } w \text{ in } T^*,$$

and from the inductive hypothesis $\hat{\delta}(p, w)$ contains (q, w).

These four cases represent all possible ways in which the first expansion in the derivation in G' can proceed. Thus, (*) holds for all $k \geq 1$.

A similar proof by induction on the length of a derivation in G yields that if $Af_1 \cdots f_j \xrightarrow[G]{*} w$ and $\hat{\delta}(p, w)$ contains (q, x), then $(p, A, q)f_1' \cdots f_j' \xrightarrow[G']{*} x$. The details are left for the reader.

Thus, we have: $S' \xrightarrow[G']{} (q_0, S, p) \xrightarrow[G']{*} x$ for p in K if and only if $S \xrightarrow[G]{*} w$, w in T^*, and $\hat{\delta}(q_0, w)$ contains (p, x). Thus, $L(G') = M(L(G))$.

Since an inverse NFT mapping is also an NFT mapping, we have:

COROLLARY 1. *The class of indexed languages is closed under inverse NFT mappings.*

We also have the following special cases.

COROLLARY 2. *The class of indexed languages is closed under (i) gsm mappings,*[6] *(ii) inverse gsm mappings, (iii) homomorphisms,*[7] *and (iv) inverse homomorphisms.*

COROLLARY 3. *The class of indexed languages is closed under intersection with regular sets.*

PROOF. Let $L \subseteq \Sigma^*$ be $L(G)$ for some indexed grammar G and let $R \subseteq \Sigma^*$ be any regular set. We can construct an NFT $M = (Q, \Sigma, \Sigma, \delta, q_0, F)$ such that $M(w) = \{w\}$ if w is in R and $M(w) = \varphi$, where φ is the empty set, otherwise. Then, $M(L) = L \cap R$.

From Lemma 3.1 and Corollaries 2(iii), 2(iv), and 3 of Lemma 3.2, we have:

THEOREM 3.2. *The class of indexed languages is a full abstract family of languages.*

Two important closure properties of the class of indexed languages which do not follow from its being a full AFL are closure under word reversal[8] and substitution.[9]

THEOREM 3.3. *The class of indexed languages is closed under word reversal.*

PROOF. Let L be $L(G)$ for some reduced form indexed grammar $G =$

[6] A *generalized sequential machine* (gsm) mapping is defined by a DFT of the form $(Q, T, \Sigma, \delta, q_0, Q)$ [6].

[7] A *homomorphism* is a mapping h of T^* into Σ^* such that $h(\epsilon) = \epsilon$ and $h(xa) = h(x)h(a)$ for x in T^*, a in T.

[8] $\epsilon^R = \epsilon$. If $w = a_1a_2 \cdots a_n$, then $w^R = a_n \cdots a_1$. $L^R = \{w^R \mid w \text{ is in } L\}$. w^R is called the *reversal* of w.

[9] Let T be a finite set and for each a in T let T_a also be a finite set. Suppose $\tau(a) \subset T_a^*$ for all a in T. τ is a *substitution* if $\tau(\epsilon) = \epsilon$ and $\tau(xa) = \tau(x)\tau(a)$ for x in T^*, a in T.

(N, T, F, P, S). Let $G' = (N, T, F, P', S)$ where P' contains the following productions.

1. If $A \to BC$ is in P, P' contains $A \to CB$.
2. All productions of the form $A \to Bf$ and $A \to a$ which are in P are also in P'.

A simple proof by induction of the length of a derivation yields that $A\zeta \overset{*}{\underset{G}{\to}} w$ if and only if $A\zeta \overset{*}{\underset{G'}{\to}} w^R$. It then follows $L(G') = (L(G))^R$.

Substitution of arbitrary indexed languages for the terminal symbols of words in an indexed language yields an indexed language.

THEOREM 3.4. *Let $T = \{a_1, \cdots, a_n\}$, T_i be a finite alphabet and $L_i \subseteq T_i^*$ be an indexed language for $1 \leq i \leq n$. If $L \subseteq T^*$ is an indexed language, then $L' = \{w_1 \cdots w_m \mid a_{i_1} \cdots a_{i_m}$ is in L and for $1 \leq j \leq m$ and $1 \leq i_j \leq n$, w_j is in $L_{i_j}\}$ is an indexed language.*

PROOF. Let L be $L(G)$ for $G = (N, T, F, P, S)$, and let L_i be $L(G_i)$ for $G_i = (N_i, T_i, F_i, P_i, a_i)$, $1 \leq i \leq n$. We can assume without loss of generality that $N_i \cap N_j = \varphi$ for $i \neq j$, $N \cap N_i = \varphi$, and $T \cap T_i = \varphi$ for $1 \leq i \leq n$.

Let $G' = (N', T', F', P', S)$ with

$$N' = N \cup \bigcup_{i=1}^{n} N_i, \qquad F' = F \cup \bigcup_{i=1}^{n} F_i,$$

$$T' = \bigcup_{i=1}^{n} T_i, \qquad P' = P \cup \bigcup_{i=1}^{n} P_i.$$

It should be clear that $L(G') = L'$.

In the next section we show that the class of indexed languages is not closed under intersection or complement.

The class of indexed languages very clearly includes all context-free languages and also, as we have seen, includes some context-sensitive languages which are not context-free, such as $\{a^n b^n c^n \mid n \geq 1\}$ and $\{a^n b^{n^2} a^n \mid n \geq 1\}$. However, indexed languages seem to more closely resemble context-free languages than context-sensitive languages in that many properties that hold for indexed languages also hold for context-free languages but do not necessarily hold for context-sensitive languages. For example, the class of context-sensitive languages is not a full AFL, whereas both the context-free and the indexed languages are. In Section 4 we show that decidability results for indexed languages also bring out this distinction.

4. *Decidability*

Let $\mathfrak{L}$ be a family of languages. The *emptiness problem* is said to be solvable for $\mathfrak{L}$ if, given any L in $\mathfrak{L}$, we can effectively determine whether $L = \varphi$. If L is specified as $L(G)$ for some grammar G, the emptiness problem for L becomes one of determining whether G generates any terminal strings.

The *membership problem* is solvable for $\mathfrak{L}$ if given any string w and given any L in $\mathfrak{L}$, we can effectively determine whether or not w is in L. The membership problem is solvable for $\mathfrak{L}$ if each language in $\mathfrak{L}$ is recursive.

The solvability of either the emptiness problem or the membership problem for a class of languages depends upon the representation of the class of languages. Here, we implicitly assume that the class of indexed languages is represented as some lexicographic listing of indexed grammars. Under this assumption we show that

both the emptiness problem and the membership problem are solvable for the class of indexed languages.

THEOREM 4.1. *Given an indexed grammar* $G' = (N, T, F, P', S)$, *the predicate* $L(G') = \varphi$ *is solvable.*

PROOF. Without loss of generality, assume that G' is in reduced form. Let $G = (N, \varphi, F, P, S)$ be the grammar derived from G by replacing each production in P of the form $A \to a$, where A is in N and a is in T by the production $A \to \epsilon$, and leaving all other productions unchanged. It should be clear that $L(G')$ is nonempty if and only if ϵ is in $L(G)$.

Let[10] $\#(P) = p$ and $\#(N) = n$.

From P, we construct a set Q of new productions. The symbols appearing in these productions are of the form (A, N_i), where A is in N and N_i is in[11] 2^N, and there is also a symbol of the form (ϵ, φ), where φ is the empty set. Moreover, all symbols on the left-hand side of index productions appearing in Q are in N.

Initially, Q is as follows:

1. If $A \to BC$ is in P, then Q contains the 2^{2n} productions $(A, N_i \cup N_j) \to (B, N_i)\ (C, N_j)$ for all values of N_i and N_j in 2^N.
2. If $A \to \epsilon$ is in P, then Q contains the production $(A, \varphi) \to (\epsilon, \varphi)$.
3. If $A \to Bf$ is in P and $f = [C_1 \to D_1, \cdots, C_k \to D_k]$, then Q contains for all values of N_{i_j}, $1 \leq j \leq k$, the productions $(A, N_i) \to (B, M)f'$, where

$$f' = [C_1 \to (D_1, N_{i_1}), \cdots, C_k \to (D_k, N_{i_k})]$$

such that $N_i = \bigcup_{j=1}^{k} N_{i_j}$. M is a set to which nonterminals in N appearing on the left-hand side of the index productions in f' may be added during the course of the algorithm to follow. Initially, $M = \varphi$. However, if it is determined that $D_j \xrightarrow[G]{*} \omega$, ω in $N_{i_j}^*$, and $C_j \to (D_j, N_{i_j})$ is an index production in f', then the nonterminal C_j is added to M.

The number of productions in Q is certainly no more than $p(2^{ln} + 2^{2n})$, where l is the maximum number of index productions in any index in F.

We now make a sequence of passes through Q. During each pass symbols in Q of the form (A, N_i) are marked or checked off in accordance with Algorithm 1 below. Upon termination of the algorithm a symbol of the form (A, N_i) is marked in Q if and only if $A \xrightarrow[G]{*} B_1B_2 \cdots B_k$ and $\bigcup_{j=1}^{k} \{B_j\} = N_i$. A symbol of the form (A, φ) is marked if and only if $A \xrightarrow[G]{*} \epsilon$. Thus, $L(G')$ is nonempty if and only if (S, φ) is marked in Q.

ALGORITHM 1

1. On the first pass through Q:
   (i) Mark all occurrences of symbols of the form $(A, \{A\})$ for all A in N.
   (ii) For all A in N mark the symbol (A, φ) in all productions of the form $(A, \varphi) \to (\epsilon, \varphi)$ in Q.
2. On each successive pass through Q:
   (i) For all A in N and all N_i in 2^N, if (A, N_i) is a marked symbol on the left side of a production in Q, mark all occurrences of the symbol (A, N_i) appearing on the right of productions and index productions in Q.

[10] $\#(E)$ = cardinality of the set E.

[11] Let M be any finite set with $\#(M) = k$. Then, $2^M = \{M_1, M_2, \cdots, M_{2^k}\}$ represents the set of subsets of M.

(ii) If $(A, N_i) \to (B, N_j)(C, N_k)$ is in Q and both (B, N_j) and (C, N_k) are marked in this production, then mark the symbol (A, N_i) in this production.

(iii) If $(A, N_i) \to (B, M)f'$ is in Q, such that (B, M) is marked, $M = \{C_1, \cdots, C_r\}$, $r \geq 0$, $C_j \to (D_j, N_{ij})$ is in f', and (D_j, N_{ij}) is marked for $1 \leq j \leq r$, and $N_i = \bigcup_{j=1}^{r} \{N_{ij}\}$, then mark the symbol (A, N_i) in this production.

(iv) If $(A, N_i) \to (B, M)f'$ is in Q and f' contains an index production $C \to (D, N_j)$ in which (D, N_j) is marked, add to M the nonterminal C. If M now contains the nonterminal B and $N_i = N_j$, then mark the symbol (A, N_i) in this production.

3. Repeat step 2 of this procedure until on a pass no more symbols are marked.

If on a given pass no more symbols are marked, then on each succeeding pass no more symbols would be marked. Since there are only a finite number of symbols in Q, it is obvious that this procedure must always halt.

The following lemma completes the proof of Theorem 4.1.

LEMMA 4.1. *A symbol (A, N_i) in Q, A in N, N_i in 2^N is marked by Algorithm 1 if and only if $A \xrightarrow[G]{*} \omega$ for some ω in N_i^*.*[12]

PROOF. (*) If (A, N_i) is the kth symbol to be marked in Q, then $A \xrightarrow[G]{*} \omega$ for some ω in N_i^*.

The statement (*) is proved by induction on k. (*) is obviously true for $k = 1$. Assuming that the statement is true for all $k < m$ with $m > 1$, consider the ways in which the mth symbol can be marked. Suppose that (B, N_j) is the mth symbol in Q to be marked.

(a) (*) is clearly true if (B, N_j) is marked in the first pass.

(b) If (B, N_j) is marked in step 2(i), (*) is trivally true.

(c) If (B, N_j) is marked in step 2(ii) in a production of the form $(B, N_j) \to (C, N_k)\ (D, N_l)$ where $N_j = N_k \cup N_l$, then $B \to CD$ is in P' and from the inductive hypothesis $C \xrightarrow[G]{*} \omega_1$ and $D \xrightarrow[G]{*} \omega_2$ from some ω_1 in N_k^* and ω_2 in N_l^*. Thus, $B \xrightarrow[G]{*} \omega_1\omega_2$ for some $\omega_1\omega_2$ in N_j^*.

(d) If (B, N_j) is marked by step 2(iii) in a production of the form $(B, N_j) \to (C, M)f'$, then $C \xrightarrow[G]{*} C_{i_1} \cdots C_{i_{m'}}$ with C_{i_k} in M and $C_{i_k} f \xrightarrow[G]{} D_{i_k} \xrightarrow[G]{*} \omega_k$, ω_k in $N_{i_k}^*$, $1 \leq k \leq m'$. Consequently, $B \xrightarrow[G]{} Cf \xrightarrow[G]{*} \omega_1 \cdots \omega_{m'}$ and $\omega_1 \cdots \omega_{m'}$ is in N_j^* since $N_j = \bigcup_{k=1}^{m'} N_{i_k}$.

(e) Suppose $(B, N_j) \to (C, M)f'$ is a production in Q such that (i) (B, N_j) is marked by step 2(iv), and (ii) f' contains an index production $C \to (D, N_j)$ in which (D, N_j) has been marked. Then, $B \xrightarrow[G]{} Cf \xrightarrow[G]{} D \xrightarrow[G]{*} \omega$ for some ω in N_j^*.

Thus, statement (*) is valid for all values of k. Now consider the following statement.

(**) If $A \xrightarrow[G]{k} B_1B_2 \cdots B_r$ and $\bigcup_{i=1}^{r} \{B_i\} = N_i$, then all occurrences of (A, N_i) on the right of productions and index productions in Q and at least one occurrence of (A, N_i) on the left of a production in Q are marked upon termination of Algorithm 1.

This statement is obviously true for $k = 1$. Assuming that it is true for all $k < m$, consider the derivation $B \xrightarrow[G]{m} C_1 \cdots C_s$ where $\bigcup_{i=1}^{s} \{C_i\} = N_j$, with $m > 1$. Such a derivation can be one of two forms:

[12] By convention, $\varphi^* = \{\epsilon\}$.

1. $B \underset{G}{\rightarrow} DE \overset{m_1}{\rightarrow} C_1 \cdots C_k E \overset{m_2}{\rightarrow} C_1 \cdots C_k C_{k+1} \cdots C_s$, where $m_1 < m$ and $m_2 < m$. Let $N_{j_1} = \bigcup_{i=1}^{k} \{C_i\}$ and $N_{j_2} = \bigcup_{i=k+1}^{s} \{C_i\}$. From the inductive hypothesis, (D, N_{j_1}) and (E, N_{j_2}) must be marked symbols in the production $(D, N_j) \rightarrow (D, N_{j_1})\ (E, N_{j_2})$ in Q, and from step 2(ii) of Algorithm 1, (D, N_j) is marked. From step 2(i) of Algorithm 1 all occurrences of (B, N_j) on the right of productions and index productions in Q are marked.

2. $B \underset{G}{\rightarrow} Df$, where $D \overset{m'}{\underset{G}{\rightarrow}} E_1 \cdots E_r$, $r \geq 0$, and $m' < m$, and $Ef \underset{G}{\rightarrow} F_i \overset{m_i}{\underset{G}{\rightarrow}} C_{i1} \cdots C_{ij_i}$, $m_i < m$, for $1 \leq i \leq r$ such that

$$C_{11} \cdots C_{1j_1} C_{21} \cdots C_{2j_2} \cdots C_{r1} \cdots C_{rj_r} = C_1 \cdots C_s .$$

Let $N_{j_i} = \bigcup_{k=1}^{j_i} \{C_{ik}\}$. There is a production $(B, N_j) \rightarrow (D, M)f'$ in Q in which f' contains the index productions $E_i \rightarrow (F_i, N_{j_i})$ for $1 \leq i \leq r$. From the inductive hypothesis each (F_i, N_{j_i}) is marked, and from step 2(iv) of the algorithm M will contain $\bigcup_{k=1}^{r} \{E_k\}$. From the inductive hypothesis, (D, M) is marked, and thus, by step 2(iii) of the algorithm, (B, N_j) is marked. If $m' = 0$, then (B, N_j) is checked by step 2(iv) of the algorithm.

This completes the proof of (**) and also the proof of Lemma 4.1. Theorem 4.1 now follows directly from the fact that $L(G')$ is nonempty if and only if (S, φ) is a checked symbol in Q upon termination of Algorithm 1.

Solvability of emptiness and closure under intersection with regular sets imply solvability of membership for any class of languages.

THEOREM 4.2. *Let $\mathfrak{L}$ be a family of languages for which the emptiness problem is solvable. If $\mathfrak{L}$ is effectively closed under intersection with regular sets, then the membership problem is solvable for $\mathfrak{L}$.*

PROOF. Let $L \subseteq T^*$ be any language in $\mathfrak{L}$ and let w be an arbitrary string in T^*. w is in L if and only if $L \cap \{w\}$ is not empty.

THEOREM 4.3. *The membership problem is solvable for the class of indexed languages.*

PROOF. Theorem 4.3 follows immediately from Corollary 3, Lemma 3.2, and Theorems 4.1 and 4.2.

COROLLARY. *Given an indexed grammar $G = (N, T, F, P, S)$, the predicate $A\zeta \overset{*}{\underset{G}{\rightarrow}} w$ for given w in T^*, A in N, and ζ in F^* is recursive.*

PROOF. Let $G' = (N \cup \{S'\}, T, F, P \cup \{S' \rightarrow A\zeta\}, S')$, where S' is a new symbol. w is in $L(G')$ if and only if $A\zeta \overset{*}{\underset{G}{\rightarrow}} w$.

Several nonclosure properties of indexed languages follow directly from some well-known results. Given any recursively enumerable set W, it is possible to find two deterministic context-free languages $L_1(W)$ and $L_2(W)$ and a homomorphism h such that $W = h(L_1(W) \cap L_2(W))$ [9].

If W is a recursively enumerable set which is not recursive [4], it must then follow that $L_1(W) \cap L_2(W) = \overline{\overline{L_1(W)} \cup \overline{L_2(W)}}$ is not an indexed language. $L_1(W)$ and $L_2(W)$ and their complements ($\bar{L}$ denotes the complement of L) are all deterministic context-free languages. Nonclosure of indexed languages under complement and intersection is thus evident.

THEOREM 4.4. *The class of indexed languages is not closed under intersection or complement.*

The language $L_1(W) \cap L_2(W)$ is a context-sensitive language. Thus, there can-

not be any class of recursive languages closed under homomorphism which includes all context-sensitive languages.[13]

At present, the class of indexed languages represents one of the largest known full abstract families of recursive languages.

It is often convenient to be able to remove ϵ-productions (productions of the form $A \rightarrow \epsilon$) from a grammar.

A *normal form* indexed grammar G is a reduced form indexed grammar (N, T, F, P, S) in which:

1. No production or index production contains the symbol S on the right-hand side.
2. No production is of the form $A \rightarrow \epsilon$, where A is in $N - \{S\}$.
3. The production $S \rightarrow \epsilon$ is in P if and only if ϵ is in $L(G)$.

THEOREM 4.5. *Given an indexed grammar G, an equivalent normal form indexed grammar G' can be effectively constructed from G.*

PROOF. Let $G = (N, T, F, P, S)$ be an indexed grammar in reduced form. From G, an equivalent ϵ-free intermediate indexed grammar $G' = (N', T, F', P', S')$ will be constructed. The nonterminals in N' will be of the form (A, M_i), where A is in N and M_i is in 2^N. For each M_i in 2^N, let m_i be the index containing the index production $B \rightarrow \epsilon$ if and only if B is in M_i. If $M_i = \varphi$, then $m_i = \epsilon$.

P' is constructed in the following manner:

1. If $A \rightarrow BC$ is in P, P' contains the productions $(A, M_i) \rightarrow (B, M_i)(C, M_i)$ for all values of M_i in 2^N.
   (i) If $Bm_i \overset{*}{\underset{G}{\rightarrow}} \epsilon$, P' also contains the production $(A, M_i) \rightarrow (C, M_i)$.
   (ii) If $Cm_i \overset{*}{\underset{G}{\rightarrow}} \epsilon$, P' contains the production $(A, M_i) \rightarrow (B, M_i)$.
2. If $A \rightarrow a$ is in P with a in T, P' contains $(A, M_i) \rightarrow a$ for all M_i in 2^N.
3. If $A \rightarrow Bf$ is in P, P' contains for all values of M_i the productions $(A, M_i) \rightarrow (B, M_j)f'$, where f' contains the index production $(C, M_j) \rightarrow (D, M_i)$ if and only if f contains $C \rightarrow D$ and where $M_j = \{C \mid Cfm_i \underset{G}{\rightarrow} Dm_i \overset{*}{\underset{G}{\rightarrow}} \epsilon\}$.
4. P' contains the initial production $S' \rightarrow (S, \varphi)$.
5. Finally, $S' \rightarrow \epsilon$ is in P' if and only if ϵ is in $L(G)$.

At this point it is convenient to define *corresponding indexed nonterminals* (c.i.n. for short) in the grammars G and G'. The definition is recursive.

1. A and (A, φ) are c.i.n. for all A in N.
2. If $A\zeta$ and $(A, M)\zeta'$ are c.i.n. and $A \rightarrow BC$ is in P, then (i) $B\zeta$ and $(B, M)\zeta'$ are c.i.n., (ii) $C\zeta$ and $(C, M)\zeta'$ are c.i.n.
3. If $A\zeta$ and $(A, M)\zeta'$ are c.i.n. and $A \rightarrow Bf$ is in P, then $Bf\zeta$ and $(B, M')f'\zeta'$ are c.i.n. where f' contains the index production $(C, M') \rightarrow (D, M)$ if and only if f contains $C \rightarrow D$ and where $M' = \left\{C \mid Cfm \underset{G}{\rightarrow} Dm \overset{*}{\underset{G}{\rightarrow}} \epsilon\right\}$.
4. If $Af\zeta$ and $(A, M)f'\zeta'$ are c.i.n. and f and f' contain the index productions $A \rightarrow B$ and $(A, M) \rightarrow (B, M')$, respectively, then $B\zeta$ and $(B, M')\zeta'$ are c.i.n.
5. $A\zeta$ and $(A, M)\zeta'$ are c.i.n. if and only if their being so follows from the rules above.

LEMMA 4.2. *If $A\zeta$ and $(A, M)\zeta'$ are c.i.n. in G and G', respectively, then $A\zeta \overset{*}{\underset{G}{\rightarrow}} w$, $w \neq \epsilon$, if and only if $(A, M)\zeta' \overset{*}{\underset{G'}{\rightarrow}} w$.*

[13] In fact, there cannot be any family of recursive languages closed under homomorphism which includes all languages of offline Turing machine [13] tape complexity $L(n)$, for any $L(n) \geq \log \log n$.

PROOF. A proof by induction on the length of a derivation is straightforward. The details are left for the reader.

We now have $S \overset{*}{\underset{G}{\Rightarrow}} w$, $w \neq \epsilon$, if and only if $S' \underset{G'}{\Rightarrow} (S, \varphi) \overset{*}{\underset{G'}{\Rightarrow}} w$. Also $S' \underset{G'}{\Rightarrow} \epsilon$ if and only if $S \overset{*}{\underset{G}{\Rightarrow}} \epsilon$. Thus, $L(G') = L(G)$.

To complete the proof of Theorem 4.5, apply the construction of Theorem 3.1 to obtain an equivalent reduced form grammar G'' from G'. G'' will also be an equivalent normal form grammar.

One undecidability result should be stated.

THEOREM 4.6. *Given an indexed grammar G, it is recursively unsolvable to determine whether:*

(a) $L(G) = R$ for some regular set R,

(b) $L(G) = L$ for some context-free language L.

PROOF. (a) follows directly from the fact that it is recursively unsolvable to determine whether a context-free grammar[14] generates a regular set [2]. (b) See [7] or [11] for what has become a standard proof for questions of this type.

5. *Containment Within Context-Sensitive Languages*

At first glance it may not be clear that the class of indexed languages is a subset of the class of context-sensitive languages. For example, in a normal form indexed grammar it is possible to have a derivation of the form (5.1) in which an

$$\begin{aligned} S \to A_1 f_1 \to A_2 f_2 f_1 \to \cdots \to A_i f_i \cdots f_1 \to \cdots \to A_k f_k \cdots f_1 \to \\ B_{k-1} f_{k-1} \cdots f_1 \to B_{k-2} f_{k-2} \cdots f_1 \to \cdots \to B_i f_i \cdots f_1 \to a \end{aligned} \tag{5.1}$$

intermediate sentential form of arbitrary length derives only one terminal symbol.

Let $G = (N, T, F, P, S)$ be a normal form indexed grammar. We begin by describing how an online nondeterministic Turing machine with one working tape can be effectively constructed from G to recognize exactly $L(G)$. We then modify M such that the length of working tape used in determining whether an input word w is in $L(G)$ is a linear function of the length of w.

We construct M to trace out, nondeterministically, on its working tape a leftmost derivation of any word in $L(G)$. The word that is being derived is compared symbol by symbol with w, the word on the input tape. If the two words are the same, M accepts w. Otherwise, M rejects w. Since M is nondeterministic, if there exists a leftmost derivation of w from S in G, M will find this derivation and accept w. If there exists no derivation of w from S in G, M will reject w.

M will retain only one copy of any string of indices on its working tape. For example, M will simulate a derivation of the form (5.2)

$$Af_1 \cdots f_k \to Bf_1 \cdots f_k C f_1 \cdots f_k \tag{5.2}$$

by first having on its working tape the string $\$Af_1 \cdots f_k\#$ and then replacing the nonterminal A by the nonterminals BC to create the string $\$BCf_1 \cdots f_k\#$. Thus, each nonterminal on the working tape of M will be considered to be indexed by all indices between that nonterminal and the right endmarker $\#$.

If the index f_1 contains the index production $B \to D$ and if derivation (5.2) pro-

[14] A context-free grammar $G = (N, T, P, S)$ is the indexed grammar (N, T, φ, P, S).

ceeds with the expansion of B by consuming the index f_1 as in (5.3), then M

$$Bf_1 \cdots f_k Cf_1 \cdots f_k \rightarrow Df_2 \cdots f_k Cf_1 \cdots f_k \tag{5.3}$$

rewrites its working tape as $\$Cf_1\$D\hat{\not\in}f_2 \cdots f_k\#$.

We call the symbol immediately to the right of the rightmost \$ sign on the working tape the *active symbol*. We use the symbol $\not\in$ as a right endmarker on the working tape for any intermediate sentential form derived by consuming an index.

For the sake of clarity, we describe the operation of M in terms of "composite moves." Each composite move consists of a number of elementary moves by M, and we leave it to the interested reader to supply the set of elementary moves to effect each composite move. We assume that at the start of each composite move, M is in the same state and the working tape head is scanning the active symbol on the working tape. A configuration of M can then be represented by a pair $(a_0a_1 \cdots \hat{a}_i \cdots a_n, Z_0 \cdots \hat{Z}_j \cdots Z_m)$, where

(i) a_k is in T for $0 < k < n$, $w = a_1 \cdots a_{n-1}$;
(ii) $a_0 = \not\in$, the left endmarker for the input; $a_n = \$$, the right endmarker for the input;
(iii) Z_k is in $N \cup T \cup F \cup \{\$, \not\in\}$ for $0 < k < m$;
(iv) $Z_0 = \$$ and $Z_m = \#$;
(v) M is scanning the input symbol a_i, $0 \leq i \leq n$;
(vi) Z_j is the active symbol. Z_j is in $N \cup T \cup F \cup \{\not\in\}$.

We write $(a_0 \cdots \hat{a}_i \cdots a_n, \alpha\hat{Z}\beta) \vdash (a_0 \cdots \hat{a}_{i+d} \cdots a_n, \alpha_1\hat{Z}_1\beta_1)$, d in $\{0, 1\}$ if there is a composite move which takes M from configuration $(a_0 \cdots \hat{a}_i \cdots a_n, \alpha\hat{Z}\beta)$ to configuration $(a_0 \cdots \hat{a}_{i+d} \cdots a_n, \alpha_1\hat{Z}_1\beta_1)$. We use the relation $\vdash^*$ over the set of configurations to denote the reflexive and transitive closure of $\vdash$.

M is constructed to execute the following composite moves.

1. If M is in any configuration of the form[15] $(x\hat{a}y, \alpha\$\hat{A}\beta)$ and if (i) $A \rightarrow BC$, (ii) $A \rightarrow b$, or (iii) $A \rightarrow Bf$ is a production in P, then M may enter the configuration (i) $(x\hat{a}y, \alpha\$\hat{B}C\beta)$, (ii) $(x\hat{a}y, \alpha\$\hat{b}\beta)$, or (iii) $(x\hat{a}y, \alpha\$\hat{B}f\beta)$, respectively.

2. If M is in configuration $(x\hat{a}y, \alpha\$\hat{A}\beta f\gamma)$ where f is the first index to the right of A and f contains an index production of the form $A \rightarrow B$, then M may enter the configuration $(x\hat{a}y, \alpha\$\beta f\$\hat{B}\hat{\not\in}\gamma)$.

3. If M is in configuration $(a_0 \cdots \hat{a}_i \cdots a_n, \alpha\$\hat{a}Z\beta)$, then M enters the configuration $(a_0 \cdots \hat{a}_{i+1} \cdots a_n, \alpha\$\hat{Z}\beta)$ if $a = a_i$, and M halts and rejects $a_1 \cdots a_{n-1}$ if $a \neq a_i$.

4. If M is in configuration $(x\hat{a}y, c\,\$\hat{f}Z\beta)$ where f is in F, then M enters the configuration $(x\hat{a}y, \alpha\$\hat{Z}\beta)$.

5. If M is in configuration $(x\hat{a}y, \alpha\$Z\beta\$\hat{\not\in}\gamma)$ where β contains no \$ symbols, then M enters configuration $(x\hat{a}y, \alpha\$\hat{Z}\beta\gamma)$.

6. If M is in configuration $(\not\in\hat{\$}, \$\hat{S}\#)$ and $S \rightarrow \epsilon$ is in P, then M enters configuration $(\not\in\hat{\$}, \$\hat{\#})$.

M initially enters the configuration $(a_0\hat{a}_1 \cdots a_n, \$\hat{S}\#)$ and then executes any

[15] We use the following conventions unless otherwise specified:
(1) xay is in $\not\in T^*\$$.
(2) α is any string of working-tape symbols.
(3) β and γ are strings of working-tape symbols not containing the symbol \$.
(4) Each string of symbols on the working tape begins with \$ and ends with #.
(5) Z is any working-tape symbol except \$.

sequence of possible composite moves. M accepts the input $w = a_1 \cdots a_{n-1}$ if and only if M can enter the final configuration $(a_0 \cdots \hat{a}_n , \$\hat{\#})$ after some sequence of composite moves.

A rather straightforward proof by induction on the length of a derivation and on the number of composite moves made by M yields that

$$(a_0 \cdots \hat{a}_i \cdots a_n , \alpha\$\hat{A}Z\beta_1 f_1 \beta_2 f_2 \cdots \beta_k f_k \beta_{k+1}\#) \vdash^* (a_0 \cdots \hat{a}_{i+m+1} \cdots a_n , \alpha\$\hat{Z}\beta_1 f_1 \cdots \beta_k f_k \beta_{k+1}\#), \quad k \geq 0,$$

where

(i) $Z\beta_1$ is[16] in $(N \cup \{\not\epsilon\})^*$ and

(ii) each β_j is in $(N \cup \{\not\epsilon\})^*$, $1 \leq j \leq k+1$

if and only if $Af_1 \cdots f_k \xrightarrow[G]{*} a_i \cdots a_{i+m}$.

Thus, $(a_0\hat{a}_1 \cdots a_n , \$\hat{S}\#) \vdash^* (a_0 \cdots \hat{a}_n, \$\hat{\#})$ if and only if $S \xrightarrow[G]{*} a_1 \cdots a_{n-1}$. The set of words accepted by M is then exactly $L(G)$.

The manner in which M operates is very similar to the operation of a device called a nested stack automaton, which is a generalized stack automaton in which the stack automaton can create and erase embedded stacks in very much the same way M can create and erase \$-$\not\epsilon$ pairs [1]. It is shown in [1] that a language L is an indexed language if and only if L is accepted by a (one-way, nondeterministic) nested stack automaton.

There has been no guarantee that the length of the string of symbols on the working tape in any intermediate configuration in a sequence of composite moves by M is linearly bounded by the length of the input. Each nonterminal appearing on the working tape derives at least one terminal symbol. However, an arbitrarily long string of indices may give rise to only one terminal, as in derivation (5.1).

An *augmented grammar* G is a normal form indexed grammar (N, T, F, P, S) such that if $A \xrightarrow[G]{*} B$ and if $B \to a$, $B \to Cf$, or $B \to CD$ is a production in P, then $A \to a$, $A \to Cf$, or $A \to CD$ respectively, is also a production in P. Moreover, if $A \xrightarrow[G]{*} B$ and if an index f in F contains the index production $C \to A$, then f also contains the index production $C \to B$.

In an augmented grammar, it is not necessary to generate any indices in a derivation in which the only net change is one nonterminal into another nonterminal. For example, in derivation (5.1), $A_i \xrightarrow{*} B_i$ and thus this derivation in an augmented grammar would proceed as shown in (5.4).

$$S \to A_1 f_1 \to A_2 f_2 f_1 \to \cdots \to A_i f_i \cdots f_1 \to a. \tag{5.4}$$

From Theorem 4.3 it immediately follows that the predicate $A \xrightarrow[G]{*} B$ is recursive for given $G = (N, T, F, P, S)$ and A and B in N. Thus, given a normal form indexed grammar G, an equivalent augmented grammar G' can be effectively constructed from G.

We now describe how M can be modified into a machine M' which will accept an input $w = a_1 \cdots a_{n-1}$ never using more than $6n$ squares of working tape if and only if w is in $L(G)$ for any augmented grammar (N, T, F, P, S).

In derivation (5.4) the indices $f_1, f_2, \cdots, f_i$ are not used and consequently need

[16] If $Z\beta = \epsilon$ and $Y \neq \epsilon$, then $(x\hat{a}y, \alpha\$\hat{Z}\beta Y\gamma) \equiv (x\hat{a}y, \alpha\$\hat{Y}\gamma)$.

ot have been generated in this particular derivation. Hence, M' will not print on s working tape any indices which are never used in the course of a derivation.

Moreover, M' will compress, wherever possible, strings of indices into single symols. For example, consider derivation (5.5),

$$A \to B_1 f_1 \to B_2 f_2 f_1 \to C f_2 f_1 D f_2 f_1 , \tag{5.5}$$

here

$$C f_2 f_1 \to E_1 f_1 \to E_2 \to a, \qquad D f_2 f_1 \to b.$$

n this particular derivation the index string $f_2 f_1$ could be condensed in a single ndex $h = (f_2 f_1)$ containing the index production $C \to E_2$.

In general we define a set F_c of compressed indices as follows.

1. All indices in F are in F_c.
2. If f and g are in F_c, F_c also contains the index (fg) containing the index production $A \to C$ if and only if $Afg \underset{G}{\to} Bg \underset{G}{\to} C$.

M' will print indices on the working tape in the form (f, d) where f is in F_c and d s in $\{0, 1, 2, 3\}$. If f is the first index generated by a nonterminal and if f is consumed during the derivation, then f will be printed on the working tape as $(f, 0)$. Any other index f which is consumed during the derivation will be printed as $(f, 2)$, or if the symbol immediately to the right of the active symbol is an index of the form (g, d) and no production of the form $A \to a$ or $A \to BC$ is ever used after f has been consumed but before g is consumed, then f will be combined with g in a symbol of the form (h, d) where $h = (fg)$.

If an index production, $A \to B$, is used from an index of the form $(f, 0)$ or $(f, 2)$ on the working tape and then B is expanded as $B \underset{G}{\to} a$ or $B \overset{*}{\underset{G}{\to}} C\zeta \underset{G}{\to} D\zeta E\zeta$ in the derivation, then the index will be rewritten on the working tape as $(f, 1)$ or $(f, 3)$, respectively. An index (f, d) will be erased from the working tape only if $d = 1$ or 3. Since G is an augmented grammar, this ensures that each index written on the working tape derives at least one additional terminal symbol.

M' will print a nonterminal A in N on its working tape as A' or A depending on whether there is, or is not, respectively, at least one index to the right of A on the working tape which will be consumed during the expansion of A.

The working tape vocabulary for M' will consist of the sets of symbols N, $N' = \{A' \mid A \text{ is in } N\}$, $F_c' = \{(f, d) \mid f \text{ is in } F_c, d \text{ is in } \{0, 1, 2, 3\}\}$ plus the special symbols \$, ¢, #.

M' will initially start off with the working tape blank and the string $\text{¢}a_1 \cdots a_{n-1}\$$ on the input tape. M' then marks off $6n$ squares on the working tape in such a fashion that if more than $6n$ symbols are ever written on the working tape, M' will halt and reject the input $w = a_1 \cdots a_{n-1}$.

M' then enters the configuration $(\text{¢}\hat{a}_1 \cdots a_{n-1}\$, \$\hat{S}\#)$ and will then be able to execute any sequence of the following composite moves. If α is a string of working-tape symbols, then α_+ is

(i) $\beta(f, 1)$ if $\alpha = \beta(f, 0)$,
(ii) $\beta(f, 3)$ if $\alpha = \beta(f, 2)$,
(iii) α otherwise.

1. If M' is in any configuration of the form $(x\hat{a}y, \alpha\$\hat{A}\beta)$ and if (i) $A \to BC$, (ii) $A \to b$, or (iii) $A \to Bf$ is in P, then M may enter the configuration

(i) $(x\hat{a}y, \alpha_+\$\hat{B}C\beta)$,
(ii) $(x\hat{a}y, \alpha_+\$\hat{b}\beta)$,
(iii) $(x\hat{a}y, \alpha\$\hat{B}\beta)$ or $(x\hat{a}y, \alpha\$\hat{B}'(f, 0)\beta)$, respectively.

2. If M' is in any configuration of the form $(x\hat{a}y, \alpha\$\hat{A}'Z\beta)$ and if (i) $A \to BC$ or (ii) $A \to Bf$ is in P, then M' may enter the configuration
(i) (a) $(x\hat{a}y, \alpha_+\$\hat{B}'C'Z\beta)$ or (b) $(x\hat{a}y, \alpha_+\$\hat{B}'CZ\beta)$ or
(c) $(x\hat{a}y, \alpha_+\$\hat{B}C'Z\beta)$,
(ii) (a) $(x\hat{a}y, \alpha\$\hat{B}'(f, 2)Z\beta)$ or,
(b) if Z is of the form (g, d) and the compressed index $h = (fg)$ is non-empty, $(x\hat{a}y, \alpha\$\hat{B}'(h, d)\beta)$.

3. If M' is in any configuration of the form $(x\hat{a}y, \alpha\$\hat{A}'\beta(f, d)\gamma)$ where β contains no symbol in F_c' and f contains the index production $A \to B$, then M' may enter the configuration
(a) $(x\hat{a}y, \alpha\$\beta(f, d)\$\hat{B}\not\epsilon\gamma)$ or,
(b) if $d = 2$ or 3, $(x\hat{a}y, \alpha\$\beta(f, d)\$\hat{B}'\not\epsilon\gamma)$.

4. If M' is in a configuration of the form $(a_0 \cdots \hat{a}_i \cdots a_n, \alpha\$\hat{a}Z\beta)$, $0 < i < n$, then M' enters the configuration $(a_0 \cdots \hat{a}_{i+1} \cdots a_n, \alpha\$\hat{Z}\beta)$ if $a = a_i$ and M' halts and rejects $w = a_1 \cdots a_{n-1}$ if $a \neq a_i$.

5. If M' is in a configuration of the form $(x\hat{a}y, \alpha\$(\hat{f}, d)Z\beta)$, then M' enters the configuration $(x\hat{a}y, \alpha\$\hat{Z}\beta)$ if $d = 1$ or 3, and halts and rejects $w = a_1 \cdots a_{n-1}$ if $d = 0$ or 2.

6. If M' is in a configuration of the form $(x\hat{a}y, \alpha\$Z\beta\$\hat{\not\epsilon}\gamma)$ where β contains no \$ symbols, then M' enters configuration $(x\hat{a}y, c\,\$\hat{Z}\beta\gamma)$.

7. If M' is in configuration $(\not\epsilon\hat{\$}, \$\hat{S}\#)$ and if $S \to \epsilon$ is in P, then M' enters configuration $(\not\epsilon\hat{\$}, \$\hat{\#})$.

The input $w = a_1 \cdots a_{n-1}$ will be accepted by M' if and only if M' can go from configuration $(\not\epsilon\hat{a}_1 \cdots a_{n-1}\$, \$\hat{S}\#)$ to configuration $(\not\epsilon a_1 \cdots a_{n-1}\hat{\$}, \$\hat{\#})$ by some sequence of composite moves during which the number of symbols on the working tape in any intermediate configuration never exceeds $6n$.

For example, M' would simulate derivation (5.5) with the following sequence of configurations.

$$
\begin{aligned}
(\not\epsilon\hat{a}b\$, \$\hat{A}\#) &\vdash (\not\epsilon\hat{a}b\$, \$\hat{B}_1(f_1, 0)\#)\\
&\vdash (\not\epsilon\hat{a}b\$, \$\hat{B}_2(h, 0)\#)\\
&\vdash (\not\epsilon\hat{a}b\$, \$\hat{C}D(h, 0)\#)\\
&\vdash (\not\epsilon\hat{a}b\$, \$D(h, 0)\$\hat{E}_2\not\epsilon\#)\\
&\vdash (\not\epsilon\hat{a}b\$, \$D(h, 1)\$\hat{a}\not\epsilon\#)\\
&\vdash (\not\epsilon a\hat{b}\$, \$D(h, 1)\$\hat{\not\epsilon}\#)\\
&\vdash (\not\epsilon a\hat{b}\$, \$\hat{D}(h, 1)\#)\\
&\vdash (\not\epsilon a\hat{b}\$, \$\hat{b}(h, 1)\#)\\
&\vdash (\not\epsilon ab\hat{\$}, \$(\hat{h}, 1)\#)\\
&\vdash (\not\epsilon ab\hat{\$}, \$\hat{\#})
\end{aligned}
$$

where $h = (f_2f_1)$.

It should be clear that $(a_0\hat{a}_1 \cdots a_n, \$\hat{S}\#) \vdash^* (a_0a_1 \cdots \hat{a}_n, \$\hat{\#})$ if and only if $S \xrightarrow[G]{*} a_1 \cdots a_{n-1}$. Moreover, each nonterminal printed on the working tape derives at least one terminal symbol. Also each index first written on the working tape in the form (f, d) is such that f contains at least one index production of the form $A \to B$ which is used in the simulation of the derivation of $w = a_1 \cdots a_{n-1}$ from S in G, and in which the nonterminal B is involved in a derivation of the form

$B \xrightarrow{*} a$ or $B \to C\zeta \to D\zeta E\zeta$ for some ζ in F^*. The index f can be associated with the terminal a or the first terminal derived from $C\zeta$, respectively. Thus, each index appearing on the working tape in any configuration can be associated with a distinct terminal symbol. Therefore, the number of indices on the working tape can never exceed the length of w. Since each symbol in N, N' or F_c' can at worst be surrounded by a \$–¢ pair, and since the working tape is delimited by \$–#, the maximum number of symbols on the working tape at any time can certainly be no more than $6n$.

It is straightforward to construct a linear bounded automaton (lba for short) M'' which simulates the behavior M'. Since a set L is accepted by some lba if and only if L is context-sensitive [15], we have shown the following result.

THEOREM 5.1 *Every indexed language is a context-sensitive language.*

We summarize results of Section 4 and Theorem 5.1 in the following theorem.

THEOREM 5.2. *The class of indexed languages is a proper superset of the class of context-free languages and is a proper subset of the class of context-sensitive languages.*

6. *Restricted Forms of Indexed Grammars*

Indexed grammars, as defined, are capable of generating a very large class of languages. In many situations it may be more convenient to consider smaller classes of languages. With this in mind some restricted forms of indexed grammars are defined in this section and the hierarchy of languages they generate is investigated.

A grammar is herein named according to the form of the index productions together with the form of the productions.

Definition. A *right linear indexed right linear* (RIR for short) grammar G is an indexed grammar (N, T, F, P, S) in which:

(1) Each index production in each index in F is of the form $A \to aB$ or $A \to a$. A and B are in N, a is in $T \cup \{\epsilon\}$.

(2) Each production in P is of the form $A \to aBf$, $A \to aB$, or $A \to a$. A and B are in N, f is in F, a is in $T \cup \{\epsilon\}$.

LEMMA 6.1. *Given a context-free grammar $G = (N, T, P, S)$, an equivalent RIR grammar G' can be effectively constructed from G.*

PROOF. Without loss of generality, assume that G is in Greibach normal 2-form [10].[17] Let $N' = N \cup \{(AB') \mid A \text{ is in } N, B \text{ is in } N\} \cup \{A' \mid A \text{ is in } N\}$.

Let P' be the set of productions constructed in the following manner:

For each production in P of the form (i) $A \to aBC$, (ii) $A \to aB$, or (iii) $A \to a$, add to P' productions of the form

(i) $A \to a(BB')[B' \to C]$, and $(AD') \to a(BB')[B' \to (CD')]$ for all D in N;

(ii) $A \to aB$, and $(AD') \to a(BD')$ for all D in N;

(iii) $A \to a$, and $(AD') \to aD'$ for all D in N;

respectively.

Let F be the set of indices appearing in the productions in P'. Let $G' = (N', T, F, P', S)$.

We now claim that:

$$\text{If } A \xrightarrow[G]{n} w, \quad \text{then} \quad (AD') \xrightarrow[G']{*} wD', \quad \text{for all } D \text{ in } N. \tag{6.1}$$

[17] That is, every production in P is of one of the following forms: (i) $A \to aBC$, (ii) $A \to aB$, or (iii) $A \to a$, where A, B, and C are in N, and a is in T. If ϵ is in $L(G)$, then $S \to \epsilon$ is also in P.

Statement (6.1) is clearly true for $n = 1$. Assuming that (6.1) is true for all $n < m$ with $m > 1$, consider a derivation $A \overset{m}{\underset{G}{\Rightarrow}} w$. Such a derivation in G can be of one of two forms:

(a) $A \underset{G}{\Rightarrow} aBC$
$\overset{m_1}{\underset{G}{\Rightarrow}} aw_1C$ with $m_1 < m$,
$\overset{m_2}{\underset{G}{\Rightarrow}} aw_1w_2$ with $m_2 < m$.

(b) $A \rightarrow aB$
$\overset{m_1}{\rightarrow} aw_1$ with $m_1 < m$.

From the constructions and the inductive hypothesis, the following derivations are possible in G'.

(a) $(AD') \rightarrow a(BB')[B' \rightarrow (CD')]$
$\overset{*}{\rightarrow} aw_1B'[B' \rightarrow (CD')]$
$\rightarrow aw_1(CD')$
$\overset{*}{\rightarrow} aw_1w_2D'$.

(b) $(AD') \rightarrow a(BD')$
$\overset{*}{\rightarrow} aw_1D'$.

Similarly, we can show that:

$$\text{If} \quad (AD') \overset{n}{\underset{G'}{\Rightarrow}} wD', \quad \text{then} \quad A \overset{*}{\underset{G}{\Rightarrow}} w. \tag{6.2}$$

In the same fashion it is easy to show that $A \overset{*}{\underset{G}{\Rightarrow}} w$ if and only if $A \overset{*}{\underset{G'}{\Rightarrow}} w$. It now follows that $L(G') = L(G)$.

A (one-way, nondeterministic) *pushdown automaton* P is a 6-tuple $(Q, T, \Gamma, \delta, q_0, Z_0)$ where:

(a) Q, T, and Γ are finite nonempty sets of *state* symbols, *input* symbols, and *stack* symbols, respectively.

(b) δ is a mapping from $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma$ into finite subsets of $Q \times \Gamma^*$.

(c) q_0 in Q is the initial state, and Z_0 in Z is the initial symbol on the stack.

A configuration of P is a triple $(q, aw, Z\gamma)$ where q is in Q, a is in $\Sigma \cup \{\epsilon\}$, w is in Σ^*, $Z\gamma$ is in Γ^*. q is the state of finite control, aw is the string remaining on the input tape, and $Z\gamma$ is the stack contents with Z the top symbol.

We write $(p, aw, Z\gamma) \vdash (q, w, c\gamma)$ if $\delta(p, a, Z)$ contains (q, α). The relation $\vdash^*$ over the set of configurations denotes the reflexive and transitive closure of $\vdash$. $N(P) = \{w \mid (q_0, w, Z_0) \vdash^* (q, \epsilon, \epsilon)$ for some q in $Q\}$. It is well known that L is $N(P)$ for some pushdown automaton P if and only if L is context-free [3].

LEMMA 6.2. *Given an RIR grammar $G = (N, T, F, P, S)$, a pushdown automaton $P = (Q, T, \Gamma, \delta, q_0, S)$ can be effectively constructed from G such that $N(P) = L(G)$.*

PROOF. P will be capable of deriving a leftmost derivation of any string w in $L(G)$. The set of pushdown tape symbols Γ will be $N \cup T \cup F$. $Q = \{q_0\} \cup \{q_A \mid A$ is in $N\}$. δ is constructed as follows:

1. If (i) $A \rightarrow aBf$, (ii) $A \rightarrow aB$, or (iii) $A \rightarrow a$ is a production in P, then $\delta(q_0, \epsilon, A)$ contains (i) (q_0, aBf), (ii) (q_0, aB), or (iii) (q_0, a), respectively.

2. If f contains the index production (i) $A \rightarrow aB$ or (ii) $A \rightarrow a$, then $\delta(q_A, \epsilon, f)$ contains (i) (q_0, aB) or (ii) (q_0, a), respectively.

3. $\delta(q_0, a, a) = \{(q_0, \epsilon)\}$ for all a in T.

4. $\delta(q_0, \epsilon, A)$ also contains (q_A, ϵ) for all A in N.
5. $\delta(q_0, \epsilon, f) = \{(q_0, \epsilon)\}$ for all f in F.

A straightforward proof by induction on the length of a derivation in G yields that if $A\zeta \xrightarrow{*} w$, then $(q_0, w, A\zeta) \vdash^* (q_0, \epsilon, \epsilon)$. A proof by induction on the number of moves made by P shows that if $(q_0, w, A\zeta) \vdash^* (q_0, \epsilon, \epsilon)$, then $A\zeta \xrightarrow{*} w$. Thus, $S \xrightarrow{*} w$ if and only if $(q_0, w, S) \vdash^* (q_0, \epsilon, \epsilon)$ and, hence, $N(P) = L(G)$.

Theorem 6.1 now follows immediately from Lemmas 6.1 and 6.2.

THEOREM 6.1. *The class of languages generated by RIR grammars is equivalent to the class of context-free languages.*

A *left linear indexed left linear* (LIL for short) grammar can be defined analogously. Then, the following lemma is obvious.

LEMMA 6.3. *Given an RIR grammar G, an LIL grammar G' can be effectively constructed from G such that $L(G') = (L(G))^R$.*

The class of context-free languages is closed under reversal. Thus:

THEOREM 6.2. *The class of languages generated by LIL grammars is also equivalent to the class of context-free languages.*

It is now natural to consider grammars in which productions are left linear in form and index productions are right linear, or vice versa.

Definition. *A left linear indexed right linear* (LIR) grammar G is an indexed grammar (N, T, F, P, S) in which:

(1) Each index production in each index in F is of the form $A \to Ba$ or $A \to a$.
(2) Each production in P is of the form $A \to aBf$, $A \to aB$, or $A \to a$.

Here, A and B are in N, f is in F, and a is in $T \cup \{\epsilon\}$.

Definition. A *right linear indexed left linear* (RIL) grammar G is an indexed grammar (N, T, F, P, S) in which:

(1) Each index production in each index in F is of the form $A \to aB$ or $A \to a$.
(2) Each production in P is of the form $A \to Bfa$, $A \to Ba$, or $A \to a$.

Example 3. Let

$$G_3 = (\{S, A, B, C, D\}, \{a, b\}, \{f_a, f_b, g_a, g_b, h_1, h_2\}, P, S),$$

be the LIR grammar where

$$\begin{aligned} f_a &= [C \to Ca], & g_a &= [D \to Da], \\ f_b &= [C \to Cb], & g_b &= [D \to Db], \\ h_1 &= [C \to \epsilon], & h_2 &= [D \to A], \end{aligned}$$

and P contains the productions

$$\begin{aligned} &S \to Ah_1, && A \to Bh_2, \\ &A \to aAf_a, && A \to bAf_b, \\ &B \to aBg_a, && B \to bBg_b, \\ &A \to C, && B \to D, \end{aligned}$$

$$L(G_3) = \{w_1x_1w_2x_2 \cdots w_nx_nw_1w_2 \cdots w_nx_nx_{n-1} \cdots x_1 \mid w_1, x_i \text{ are in } \{a, b\}^*, \quad 1 \leq i \leq n\}.$$

$L(G_3)$ is not a context-free language.

The construction in Lemma 6.1 also shows that each context-free language can be generated by some LIR grammar. Thus, the class of languages generated by

LIR grammars includes all context-free languages and some context-sensitive languages.

A similar remark holds for RIL grammars.

Perhaps one other restricted form of indexed grammar should be mentioned. It is possible to have two types of nonterminals in an indexed grammar, viz. those that are capable of generating new indices and those that are not. The former we call nonterminals, and the latter, intermediates. If all index productions contain only terminals and intermediates, then once an index is consumed, no new indices can be generated by these intermediates. We call such a grammar a restricted indexed grammar.

Formally, a *restricted indexed grammar* is a 6-tuple (N, I, T, F, P, S) where:

(a) N, I, T are finite disjoint sets of *nonterminal*, *intermediate*, and *terminal* symbols, respectively.

(b) F is a finite set of objects of the form $[B_1 \rightarrow \omega_1, \cdots, B_k \rightarrow \omega_k]$, where B_i is in I and ω_i is in $(I \cup T)^*$, $1 \leq i \leq k$.

(c) P is a set of productions of two forms: (i) $A \rightarrow \alpha$ where A is in N, α is in $(NF^* \cup IF^* \cup T)^*$; (ii) $A \rightarrow \beta$ where B is in I, β is in $(I \cup T)^*$. A production of type (ii) is called an *intermediate production*.

(d) S is a distinguished symbol in N.

A derivation in a restricted indexed grammar $G = (N, I, T, F, P, S)$ and the language generated by G are defined in exactly the same way as for the indexed grammar $(N \cup I, T, F, P, S)$. The derivation tree corresponding to the derivation has the property that once an intermediate appears in a path no new indices appear later in that path.

There are several interesting subclasses of restricted indexed grammars. Restricted right linear indexed right linear grammars and restricted left linear indexed left linear grammars can be defined analogously to RIR grammars and LIR grammars. It is not difficult to show that the class of languages generated by restricted RIR grammars and the class of languages generated by restricted LIL grammars are both exactly the class of linear context-free languages.

An R-grammar is a restricted indexed grammar in which all intermediate and index productions are right linear in form. A stack automaton [8, 9] which can read the stack in only one direction, from the top toward the bottom, is called a reading pushdown automaton (RPA) [1]. It is shown in [1] that the class of languages generated by R-grammars is equivalent to the class of languages recognized by (one-way, nondeterministic) RPA.

7. *Conclusions*

A new class of grammars, called indexed grammars, has been defined. These grammars generate a new class of languages, called indexed languages, properly containing all context-free languages and properly contained within the class of context-sensitive languages. The class of indexed languages represents a natural generalization of context-free languages and closely resembles this class of languages with similar closure properties and decidability results.

REFERENCES

1. AHO, A. V. Nested stack automata. (Submitted to a technical journal.)
2. BAR-HILLEL, Y., PERLES, M., AND SHAMIR, E. On formal properties of simple phrase

structure grammars. *Z. Phonetik, Sprachwiss. Kommunikationsforsch. 14* (1961), 143–172; in Bar-Hillel, Y., *Language and Information*, Addison-Wesley, Reading, Mass., 1965, pp. 116–150.

3. CHOMSKY, N. Formal properties of grammars. In Luce, R., Bush, R., and Galanter, E. (Eds.), *Handbook of Mathematical Psychology, Vol. II*, Wiley, New York, 1963.
4. DAVIS, M. *Computability and Unsolvability*. McGraw-Hill, New York, 1958.
5. FLOYD, R. W. On the nonexistence of a phrase structure grammar for ALGOL 60. *Comm. ACM 5*, 9 (Sept. 1962), 483–484.
6. GINSBURG, S. *The Mathematical Theory of Context Free Languages*. McGraw-Hill, New York, 1966.
7. GINSBURG, S., AND GREIBACH, S. A. Abstract families of languages. IEEE Conference Record of Eighth Annual Symposium on Switching and Automata Theory, IEEE pub. no. 16-C-56, Oct. 1967, pp. 128–139.
8. GINSBURG, S., GREIBACH, S. A., AND HARRISON, M. A. Stack automata and compiling. *J. ACM 14*, 1 (Jan. 1967), 172–201.
9. GINSBURG, S., GREIBACH, S. A., AND HARRISON, M. A. One-way stack automata. *J. ACM 14*, 2 (April 1967), 389–418.
10. GREIBACH, S. A. A new normal-form theorem for context-free phrase structure grammars. *J. ACM 12*, 1 (Jan. 1965), 42–52.
11. GREIBACH, S. A. A note on undecidable properties of formal languages. Tech. Rep. TM-738/038/00, System Development Corp., Santa Monica, Calif., Aug. 1967.
12. GREIBACH, S. A., AND HOPCROFT, J. E. Independence of AFL operations. Tech. Rep. TM-738/034/00, System Development Corp., Santa Monica, Calif., July 1967.
13. HARTMANIS, J., AND STEARNS, R. E. On the computational complexity of algorithms. *Trans. Amer. Math. Soc. 117* (May 1965), 285–306.
14. HOPCROFT, J. E., AND ULLMAN, J. D. Nonerasing stack automata. *J. Comp. Syst. Sci. 1*, 2 (Aug. 1967), 166–186.
15. KURODA, S. Y. Classes of languages and linear bounded automata. *Inform. Contr. 7* (June 1964), 207–223.
16. ROSENKRANTZ, D. J. Programmed grammars—a new device for generating formal languages. IEEE Conference Record of Eighth Annual Symposium on Switching and Automata Theory, IEEE pub. no. 16-C-56, Oct. 1967, pp. 14–20.

RECEIVED FEBRUARY, 1968; REVISED APRIL, 1968