On Context-Free Languages

ROHIT J. PARIKH

Massachusetts Institute of Technology, Cambridge, Massachusetts

Abstract. In this report, certain properties of context-free (CF or type 2) grammars are investigated, like that of Chomsky. In particular, questions regarding structure, possible ambiguity and relationship to finite automata are considered. The following results are presented:

(a) The language generated by a context-free grammar is linear in a sense that is defined precisely.

(b) The requirement of unambiguity—that every sentence has a unique phrase structure—weakens the grammar in the sense that there exists a CF language that cannot be generated unambiguously by a CF grammar.

(c) The result that not every CF language is a finite automaton (FA) language is improved in the following way. There exists a CF language L such that for any $L' \subseteq L$, if L' is FA, an $L'' \subseteq L$ can be found such that L'' is also FA, $L' \subseteq L''$ and L'' contains infinitely many sentences not in L'.

(d) A type of grammar is defined that is intermediate between type 1 and type 2 grammars. It is shown that this type of grammar is essentially stronger than type 2 grammars and has the advantage over type 1 grammars that the phrase structure of a grammatical sentence is unique, once the derivation is given.

1. *Preliminaries*

Definition 1. By a phrase-structure grammar **G** is meant a set V of symbols and a set R of rules R_i of the form $R_i : \omega_i \rightarrow \eta_i$, where ω_i and η_i are strings (possibly null) composed of members of V.

Definition 2. The grammar **G** will be said to be of type 1 (a context grammar) if all the rules are of the type: $R_i = \varphi_i A_i \psi_i \rightarrow \varphi_i \omega_i \psi_i$, where A_i are individual symbols of V; φ_i, ω_i, ψ_i are some strings on V; and ω_i are not null. It will also be assumed that $S = A_i$ for at least one i.

Definition 3. A type 1 grammar **G** will be said to be of type 2 or context free (CF) if all the φ_i, ψ_i as given in the foregoing are null.

Definition 4. If **G** is a type 1 grammar, then by V_N is meant the subset $\{A_i\}$ of V. By V_T is meant $V - V_N$ (T means terminal; N means nonterminal).

Convention 1. Hereafter, when talking about type 1 grammars we will use the following convention. Capital letters denote strings on V_N, lower-case letters denote strings on V_T and Greek letters denote strings on V. Early letters of the alphabet denote individual symbols; late letters denote arbitrary (possibly empty) strings. The boundary symbol $\#$ will always belong to V_T (although it does not

This work was supported in part by the National Science Foundation.

Author's present address: Department of Mathematics, University of Bristol, Bristol, England.

Ed. Note. A preliminary version of this paper appeared in the Quarterly Progress Report #60 (January 15, 1961), Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, Mass., pp. 199–212. The theory of context-free languages has been extensively developed in the years since 1960, when this paper was written and when the first significant work on context-free languages was beginning; yet the results developed here (particularly Theorems 2 and 3) have remained among the most fundamental, yet subtly difficult to prove, in the theory. Since copies of the original paper are quite scarce, it is being reprinted here, with minor editorial revisions, at the suggestion of one of the editors.

belong to the alphabet). In discussions of type 2 grammars, this symbol will often be omitted.

Several results from papers by Chomsky and others [2–4] will be used. While this report does not presuppose acquaintance with those papers, they form the context of this paper.

Definition 5. By the set of φ-generable strings of a phrase-structure grammar **G** is meant the smallest set A_φ such that

(a) $\varphi \in A_\varphi$.

(b) If $\varphi_1\omega_i\varphi_2 \in A_\varphi$ and $\omega_i \rightarrow \eta_i$ is a rule, then $\varphi_1\eta_i\varphi_2 \in A_\varphi$.

If a string ψ belongs to this set we will call it φ-generable, and write $\varphi \Rightarrow \psi$. The set of generable strings will be the set of $\# S \#$-generable strings. A member of this set will be called generable.

Definition 6. The language L generated by **G** will be the set of those strings on V_T that are generable. Such strings will be referred to as sentences of **G** or L. Thus a sentence is a generable string which contains no nonterminal symbols. A language will be said to be of type X if it can be generated by a grammar of type X.

Note. Hereafter, all grammars will be type 1 grammars unless otherwise specified. For example, the A_i, ω_i of Definition 7 refer to Definition 2.

Definition 7. (R_i, j) will be said to be a φ-derivation of ψ if $\varphi = \eta_1\varphi_i A_i \psi_i \eta_2$, this A_i is the jth symbol of φ, and $\psi = \eta_1\varphi_i\omega_i\psi_i\eta_2$. The members of ω_i in ψ will be said to be descendents of A_i (here A_i refers not only to the particular member of V but also to the particular occurrence of it in φ) with respect to (R_i, j). The members of η_1, φ_i, ψ_i, η_2 in ψ will be said to be descendents of their counterparts of φ with respect to (R_i, j).

Definition 8. $D = (R_{i_1}, j_1), \cdots, (R_{i_n}, j_n)$ will be said to be a φ-derivation of ψ if there exists a sequence $\varphi = \varphi_0, \varphi_1, \cdots, \varphi_n = \psi$ such that (R_{i_k}, j_k) is a φ_{k-1} derivation of φ_k. β in ψ will be said to be a descendent of α in φ with respect to D if there exist $\alpha = \alpha_0, \cdots, \alpha_n = \beta$ such that α_l is a descendent of α_{l-1} with respect to (R_{i_l}, j_l).

Definition 9. Let φ be a generable string and let φ_1 be a substring of φ. Then φ_1 will be said to be a phrase of φ of type A with respect to D, where $D = (R_{i_1}, j_1), \cdots, (R_{i_n}, j_n)$, if there exists a corresponding sequence of strings $\# S \# = \varphi_0, \varphi_1, \cdots, \varphi_n = \varphi$ and an occurrence A_l of A in some φ_k such that φ_1 is the set of all descendants in φ of this A_l in φ_k with respect to the derivation $D' = (R_{i_{k+1}}, j_{k+1}), \cdots, (R_{i_n}, j_n)$. We will say "$\varphi_1$ is a phrase of φ of type A" if there exists a D as in the foregoing.

Remark. If two occurrences α and β of symbols in a string φ belong to the same phrase, then so do all the occurrences between these two.

Definition 10. A grammar **G** will be said to have unambiguous phrase structure if, given two derivations D, D' from $\# S \#$ of a member x of L, and a substring x' of x, x' is a phrase of x of type A with respect to D' if and only if x' is a phrase of x of type A with respect to D.

Definition 11. A grammar has unique phrase structure if, given any two phrases in a sentence, either they are disjoint or one is a part of the other.

THEOREM 1. *If a grammar has unambiguous phrase structure, it has unique phrase structure.*

PROOF. Let x_1 and x_2 be two subphrases of a sentence (that is, a generable string on V_T) x, and say x_1, x_2 are of types A_1 and A_2 with respect to derivations

D, D' of x. Now by unambiguity it may be assumed that $D = D'$. If x_1, x_2 are disjoint there is nothing to prove. So pick a in both x_1 and x_2. (*Caution: a refers not only to a member of V_T but to a particular occurrence of this member.*) Now a is descended from an occurrence α of A_1, and a is descended from an occurrence β of A_2. It is easy to see now that either β is descended from α, or α is descended from β, or $\alpha = \beta$. Hence either $x_1 \subseteq x_2$, or $x_2 \subseteq x_1$, or $x_1 = x_2$. Q.E.D.

Remark. Later an example will be given of a CF language that has no CF grammar with unique phrase structure. It follows that in that case ambiguity is unavoidable.

2. *Principal Results*

LEMMA 1. *Every CF-language L has a CF grammar $\mathbf{G}$ such that if $A \in V_n$ and $A \neq S$ then there exist terminal strings x, y and z, x not null, and at most one of y and z null, such that $A \to x$ and $A \to yAz$ are rules of $\mathbf{G}$. Moreover, if L has a CF grammar with unique phrase structure, then $\mathbf{G}$ can be assumed to have unique phrase structure.*

PROOF. If for some nonterminal A there is a terminal x such that $A \Rightarrow x$, then one adds the rule $A \to x$. If there is no such x, one eliminates A and every rule in which A occurs. This does not reduce the generable V_T strings because if any rule with A on the right-hand side is used, then the result cannot lead to a V_T string. However, some nonterminal symbols may become terminal. Then one eliminates these also. This process must have an end because each time one eliminates at least one symbol. Finally, for every $A \in V_N$ except S one has a rule $A \to x$, with x terminal. Now if there exists an A for which there is no rule $A \to \varphi_1 A \varphi_2$ with at least one of φ_1, φ_2 not null, then one eliminates A and for every rule of the form $B \to \varphi_1 A \varphi_2$ and for every rule $A \to \psi$ one replaces these two by the rule $B \to \varphi_1 \psi \varphi_2$ (φ_1, φ_2 may also contain A, in which case one repeats this process with $B \to \varphi_1 \psi \varphi_2$), and finally one only has symbols A such that there exist x, φ_1, φ_2 with $A \to x$ and $A \to \varphi_1 A \varphi_2$ as rules and at least one of φ_1, φ_2 not null. But then one must also have terminal y, z so that $\varphi_1 \Rightarrow y$, $\varphi_2 \Rightarrow z$ and not both y and z are null. So one adds the rule $A \to yAz$. This does not change the membership of L. In this entire process, a rule was never added that was not equivalent to a derivation. Hence, no new phrases were created and the new grammar must have unique phrase structure if the old one did. (If x_1, x_2 are phrases by the new grammar, then also by the old grammar they are phrases and then they must be disjoint or one is a part of the other.) Q.E.D.

LEMMA 2. *If a language L has a CF grammar, then it has a CF grammar in which $A \Rightarrow B$ is never true for A, B in V_N.*

PROOF. Let us define $A \equiv B$ if $A \Rightarrow B$ and $B \Rightarrow A$. Replacing all the congruence classes by one element each, one gets a grammar $\mathbf{G}'$ in which $A \Rightarrow B$ is a partial ordering of V_N. Now, for every minimal B in this ordering and every rule $A \to B$, one eliminates the rule $A \to B$ and replaces it by the rules $A \to \omega$ whenever $B \to \omega$ is a rule. This would not create any more rules of the form $A \to C$, since by minimality of B, $B \Rightarrow C$ (and hence $B \to C$) is impossible. Now one has reduced the number of rules of the form $C \to D$ without changing L. One continues until all such rules are eliminated. Now, every rule that does not increase length will replace a nonterminal symbol by a terminal one and $A \Rightarrow B$ is impossible. Q.E.D.

Remark. The constructions of Lemmas 1 and 2 can be applied successively to obtain a grammar satisfying all the conditions stated in these lemmas.

2.1 Commutative Images of CF Languages

Definition 12. Let J denote the nonnegative integers. Let J^n denote the direct product of J taken n times. Then J^n is a commutative associative semigroup with identity, under componentwise addition. (For example, in J^2: $(2, 3) + (5, 0) = (7, 3)$, etc.)

A subset Q of J^n will be said to be *linear* if there exist members $\alpha, \beta_1, \cdots, \beta_m$ of J^n such that

$$Q = \{x \mid x = \alpha + n_1\beta_1 + \cdots + n_m\beta_m, n_i \in J\}.$$

Q will be said to be *semilinear* if Q is the union of a finite number of linear sets.

Definition 13. Let L be any CF language on the terminal symbols $a_1 \cdots a_n$, $※$. Define Φ from L into J^n as follows:

$$\begin{aligned}
\Phi(a_1) &= (1, \cdots 0, 0, 0)\\
\Phi(a_2) &= (0, 1, 0, \cdots 0)\\
&\cdots\cdots\cdots\cdots\\
\Phi(a_n) &= (0 \cdots 0, \cdots 1)\\
\Phi(※) &= (0 \cdots 0, \cdots 0)\\
\Phi(xy) &= \Phi(x) + \Phi(y).
\end{aligned}$$

Then $\Phi(x)$ is called the commutative image of x and $\Phi(L)$ the commutative map of L. (Note that Φ depends on the order of the a_i; however, this fact will be ignored.)

Theorem 2. *Let* $\mathbf{G}$ *be a CF grammar generating the language* L. *Let* $\Phi(L)$ *be the commutative map of* L. *Then* $\Phi(L)$ *is a semilinear subset* Q *of* J^n *for the proper* n. *Moreover, a canonical description of* Q *in the form*

$$Q = Q_1 \cup Q_2 \cup \cdots \cup Q_m$$

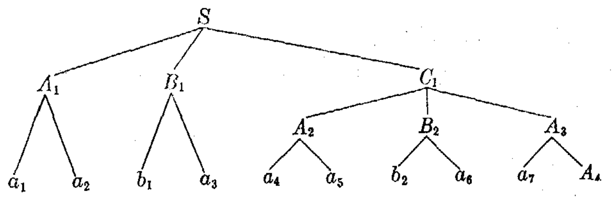
where

$$Q_j = \{x \mid x = \alpha_j + n_1\beta_{j1} + n_2\beta_{j2} + \cdots + n_{k_j}\beta_{jk_j}, n_i \in J\}$$

can be found effectively from $\mathbf{G}$.

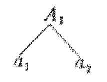
Proof. Let V' be a subset of V. Consider the set L' of all members x of L such that in some derivation D of x, the members of V' are precisely the symbols that are used. It is enough to find a canonical description for L', since L is a finite union of such L'. Obviously, L' is empty unless V' contains S. Since no rule involving some symbol outside V' can be used in such a D, it can be assumed without loss of generality that V' is V.

At this point, the notion of a tree is introduced by means of an illustration. Suppose one has the rules $S \rightarrow ABC$, $A \rightarrow aA$, $A \rightarrow aa$, $B \rightarrow ba$, $C \rightarrow ABA$. Then one could have the derivation $S \rightarrow ABC \rightarrow aaBC \rightarrow aabaC \rightarrow aabaABA \rightarrow aabaaaBA \rightarrow aabaaabaA \rightarrow aabaaabaaA$. This could be written diagrammatically as follows:

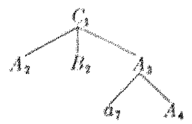


(The different occurrences of a, b, c, etc., are numbered for convenience.) The order in which the rules are applied is not preserved but nothing essential is lost. (It is possible to define a tree as an equivalence class of derivations, but in that case intuitively obvious facts would have to be proved. Here, it is enough to see that such a formal and more rigorous approach is possible.) Some notions are illustrated here.

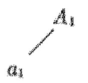
SA_1a_1, SB_1 are *chains*; SA_1b_1 is not; a_1 is descended from A_1 and S but not from B_1; A_4 is descended from A_3, hence A is descended from itself;



is a subtree and so is



but



is not. The string *aabaaabaaA* is the *product* of the tree.

Now, for every α in V_N we define two sets R_α and T_α. φ is said to be in R_α if (a) φ contains α and α is the only nonterminal symbol in φ; and (b) there is a tree with α at the vertex such that φ is the product of the tree and no symbol occurs more than n times in any chain of the tree, where n is the number of elements in V.

T_α is defined analogously except that condition (a) is replaced by the condition that φ be terminal. It is also required that every symbol of V appear in the tree of condition (b).

We claim that there is only a finite number of trees satisfying condition (b), since in any such tree the length of any chain cannot be greater than the square of the number of symbols in V. Hence R_α and T_α are finite and can be found effectively from G.

For each α, let $v_1^\alpha, v_2^\alpha, \cdots, v_r^\alpha$ be the vectors obtained by removing α from a member φ of R_α and then taking the image under Φ. (See Definition 13.) Let $u_1, \cdots, u_k$ be the images under Φ of the members of T_S. Set

$$Q_l = \{x \mid x = u_l + n_1v_1^\alpha + \cdots + n_rv_r^\alpha + n_1'v_1^\beta + \cdots + n_r'v_{r'}^\beta + \cdots ;\quad n_i, n_i', \cdots \in J, \quad \alpha, \beta, \cdots \in V_N\}.$$

Then $\Phi(L') = Q_1 \cup Q_2 \cup \cdots \cup Q_k$.

For, certainly, if some string y is in L' and $\alpha \in v_N$, then α must occur somewhere in a tree for y. Then in the place where α occurs one could imbed (for any v_i^α that one pleases) a tree with a product string φ, $\varphi \in R_\alpha$ and $\Phi(\varphi - \alpha) = v_i^\alpha$. Hence $\Phi(y) + v_i^\alpha$ is also in $\Phi(L')$.

On the other hand, if a string has a tree with more than n α's in a chain then, for some β, one can find $n + 1$ β's in a descending sequence $\beta_1, \cdots, \beta_{n+1}$ such that all of them occur in a chain which, moreover, has the property that there is no chain entirely below β_i which contains more than n occurrences of any symbol. Now suppose that we replace the tree following β_i by the tree following β_{i+1}. Then we have reduced the product of the entire tree by exactly a member of R_β. The new tree may not contain all the symbols from V. However, since there are only $n-1$ symbols in V apart from β, there must be a choice of i, $1 \leq i \leq n$, such that the new tree contains all the members of V if the old one did. This process is continued until one has a tree in which any chain has, at most, n occurrences of any one symbol, and its product must be a member of T_s. Q.E.D.

The converse of Theorem 2 is easy to verify. Assuming Q and Q_j have the stated form, let $y_j, y_{1j}, \cdots, y_{r_j j}$ be strings whose images under Φ are $\alpha_j, \beta_{1j}, \cdots, \beta_{r_j j}$. The rules

$$S \to A_j, \quad A_j \to y_j, \quad A_j \to A_j y_{ij} \quad (1 \leq j \leq m, \quad 1 \leq i \leq r_j)$$

generate a language L such that $\Phi(L) = Q$. These rules in fact constitute a finite-state (type 3) grammar, in the sense defined in what follows. Thus one has:

COROLLARY 1. *Every CF language is equivalent to a finite-state language modulo permutations.*

COROLLARY 2. *Let L be any CF language and $a \in V_T$. Define a map Θ from $L \to J$ by $\Theta(a) = 1$; $\Theta(b) = 0$, $b \neq a$; $\Theta(\alpha\beta) = \Theta(\alpha) + \Theta(\beta)$. Then there exist integers m, m', $n_1, \cdots, n_k$ such that if $n > m$, $n \in \Theta(L)$ if and only if $n \equiv n_i$ (mod m') for some i.*

PROOF. It is easy to see that $\Theta(L)$ will be a semilinear subset of the integers. Let $A_1, \cdots, A_r, A_{r+1}, \cdots, A_s$ be the linear sets whose union it is. Here we assume that $A_1 \cdots A_r$ are finite and $A_{r+1} \cdots A_s$ have each a smallest vector $\delta_i \neq 0$ such that if $x \in A_i$, then $x + \delta_i \in A_i$. Here δ_i is, of course, an integer. Take m to be bigger than all the elements of the finite sets $A_1, \cdots, A_r$ and the first elements of $A_{r+1}, \cdots, A_s$. Take m' to be the product of all the δ_i, and $n_1, \cdots n_k$, the residues modulo m' of the elements appearing among $A_{r+1}, \cdots, A_s$.

2.2 INHERENT AMBIGUITY

THEOREM 3. *There exists a CF-language L such that no CF grammar for L has unique phrase structure.*

In order to prove Theorem 3, we show, first, that the language

$$L = \{x \mid x = a^n b^m a^{n'} b^m \quad \text{or} \quad x = a^n b^m a^n b^{m'}, \quad \text{some } n, n', m, m' \in J - \{0\}\}$$

is CF. For, consider the rules:

$$S \to AB \qquad A \to aAa, \quad A \to aBa, \quad B \to b, \quad B \to bB,$$

$$S \to CD \qquad C \to bCb, \quad C \to bDb, \quad D \to a, \quad D \to aD.$$

The terminal descendants of B have the form b^n, $n > 0$. The terminal descendants of D have the form a^n, $n > 0$. Hence the terminal descendants of A must be $a^m b^n a^m$; the terminal descendants of C must be $b^m a^n b^m$. It is easy to see that these rules generate L.

Suppose that L has a grammar with unique phrase structure. By Lemma 1 it may be assumed that for every A in V_N there exist rules $A \to x$, $A \to yAz$ with x, y, z terminal, x not empty and, at most, one of y and z not empty. It may also be assumed that every A in V_N is descended from S because the others cannot contribute to L.

The intuitive idea behind the proof is as follows. L contains precisely the strings of the form $a^i b^j a^k b^l$ with either $i = k$ or $j = l$, or both. Now the strings $a^i b^j a^i b^j$ will have subphrases of the form $a^i b^j a^i$, while the strings $a^i b^j a^k b^j$ will have subphrases of the form $b^j a^k b^j$. Hence the strings $a^i b^j a^i b^j$ must contain both and will therefore have overlapping phrases. This is the essence of the proof. The details follow.

We *claim* that there are only eight types of nonterminal symbols A which can occur in V:

1a. There exist x and y such that $A \to xAy$ is a rule and $x = a^m$, $y = a^{m'}$, and no b's are ever descended from A.

1b. Same as 1a except that there are b's descended from A and $m \neq m'$ in at least one pair x, y. However, there is an integer l_A such that in any string descended from A there are less than l_A b's.

2a. Same as 1a with a and b interchanged.

2b. Same as 1b with a and b interchanged.

3a. Whenever $A \to xAy$ is a rule, $x = y = a^m$ for some m. There are b's descended from A, but the number of b's in a string from A is bounded by l_A.

3b. Whenever $A \to xAy$ is a rule $x = y = a^m$ for some m. There are integers l_A, $g_A = g$, $f_A = f$ such that $A \to a^g A a^g$ is a rule; some string descended from A has l_A b's; and if xby is a terminal string descended from A with at least l_A b's, then $xbb^f y$ is also descended from A.

4a. Same as 3a with a and b reversed.

4b. Same as 3b with a and b reversed.

Proof of Claim. First, it is easy to see that for every A either $A \Rightarrow xAy$ implies $x = a^m$, $y = a^{m'}$ for some $m, m' \geq 0$; or $A \Rightarrow xAy$ implies $x = b^m$, $y = b^{m'}$ for some $m, m' \geq 0$.

Anything else would contradict one of two requirements:

(a) Every sentence has exactly two groups of a's and two groups of b's.

(b) Either the groups of a's are identical, or else the groups of b's are.

Now, if $A \to xAy$ with $x = a^m$, $y = a^{m'}$ with $m \neq m'$, then A can only occur in the derivation of a string $a^i b^j a^k b^j$. Now the number of b's generated by A must be fixed. Otherwise one could not have matching of the groups of b's. Hence A is of type 1a or 1b. Now let us assume that $A \to xAy$ with x and y powers of a, and A is not of type 1a, 1b or 3a. It will be shown that it must belong to type 3b. It is already known that if $A \to xAy$, then $x = y$ must be true. Also, all strings generated from A must have the form $a^m b^k a^{m'}$ where $m - m'$ is constant.

Consider a string u descended from A which has more b's than the largest number occurring on the right-hand side of any rule. Then at the time in the derivation of u when the first b is generated, there must be a nonterminal symbol B left over. Now that string has the form $a^l \omega b \eta B \theta a^l$ or $a^l \omega B \eta b \theta a^l$. (The existence of the a^l at the two sides can be assumed because we could always have used the rule $A \to a^l A a^l$ before starting.) It is easy to see that if $B \to xBy$ is a rule, then x and y must be powers of b. Let $xy = b^{f_B}$. Let such an f_B be chosen for each B with a rule $B \to xBy$ attached to it and x and y powers of b. Now, in the string $a^l \omega b \eta B \theta a^l$ or $a^l \omega B \eta b \theta a^l$, no a's could possibly come from η. Hence if u has the form zbz' one can also get the string $zbb^{f_A} z'$ from A, where f_A is the product of all the f_B taken above. Hence A is of type 3b.

Types 2, 4 are handled in a similar manner. The claim is proved.

Proof of Theorem 3. Let p be a positive number divisible by all the f_A, g_A described in types 3b and 4b. Let $n/2$ be larger than all the l_A described in the foregoing. Consider the string $x_0 = a^{n+p} b^n a^{n+p} b^{n+2p}$.

Now no derivable string can contain more than three symbols of type 1b or 2b. The string x_t cannot have contained in its derivation any symbols of types 1a, 1b, 3a or 4b. On the other hand, not enough a's could come from type 2b or 4a. Hence there must have been an occurrence of a symbol A of type 3b, whose descendant is a phrase of the form zb^nz'. If one applies the rule $A \rightarrow xAy$ enough extra times, one can get another string in which the phrase coming from A is of the form $a^pzb^na'a^p$. This can be changed, as before, to $a^pzb^pb^{2p}z'a^p$. Thus one gets the string $x_1 = a^{n+2p}b^{n+2p}a^{n+2p}b^{n+2p}$ with the A-phrase containing at least $a^pb^{n+2p}a^p$, and bounded on both sides by a's. Similarly, by duality between a and b, there is a phrase of x_1 containing at least $b^pa^{n+2p}b^p$, and bounded on both sides by b's. But these phrases overlap, and yet one cannot include the other.

Hence G cannot have unique phrase structure. Q.E.D.

COROLLARY. *L is a CF language for which there is no CF grammar with unambiguous phrase structure.*

PROOF. The proof follows immediately from Theorem 1.

2.3 RELATION TO FA LANGUAGES

Definition 14. A finite-state grammar G consists of a finite set $\mathbf{S}$ (called the internal states of G), a finite set W (called the vocabulary of G), two distinguished elements S_0 and S_f of $\mathbf{S}$ and a subset $\mathbf{R}$ of $\mathbf{S} \times \mathbf{S} \times W'$ (called the rules of G), where $W' = W \cup \{\Lambda\}$ and Λ is the empty string.

Remark. Here we depart somewhat from the 1959 Chomsky definition [4] in that we do not require a symbol to be emitted at every interstate transition. It is not difficult to show, however, that the difference is unimportant and that the same class of languages is generated.

Definition 15. Let G be a finite-state grammar. Then it will be said that the sentence x is generated by G if there exists a sequence (S_0, S_1, x_0), $(S_1, S_2, x_1) \cdots (S_n, S_f, x_n)$ of members of $\mathbf{R}$ such that $x = x_0x_1 \cdots x_n$. The language generated by G is the set of all such sentences x.

THEOREM. *Every language generated by a finite-state grammar (FA language) is CF.*

PROOF. The proof has been given by Chomsky [4].

THEOREM 4. *There exists a CF language L such that given a grammar G' for an FA language L' with $L' \subseteq L$, one can effectively find a grammar G'' for an FA language L'' such that $L' \subseteq L'' \subseteq L$ and L'' has infinitely many sentences not in L'.*

Before this theorem is proved, two definitions are given and a lemma is proved.

Definition 16. A finite translator T consists of two finite sets V, V' (called the vocabularies of T), a set $\mathbf{S}$ (called the internal states of T) and a certain subset $\mathbf{R}$ (called the rules of T) of $\mathbf{S} \times V \times \mathbf{S} \times V'' \times \{0, 1\}$. Here $V'' = V' \cup \{\Lambda\}$ and Λ is the empty string. A member S_0 of $\mathbf{S}$ is distinguished and called the initial state of T.

Definition 17. Given a finite translator $T = \{V, V', \mathbf{S}, \mathbf{R}\}$ and a sentence $x = x_1 \cdots x_m$ on V, sentence z will be said to be a translation of x by T, if there exists a sequence

$$\langle S_0, y_1, S_1, z_1, i_1\rangle, \quad \langle S_1, y_2, S_2, z_2, i_2\rangle, \cdots, \langle S_n, y_{n+1}, S_{n+1}, z_{n+1}, i_{n+1}\rangle$$

of members of $\mathbf{R}$ such that $y_1 = x_1$. If $y_l = x_j$ and $i_l = 0$, then $y_{l+1} = x_j$; otherwise $y_{l+1} = x_{j+1}$. Furthermore, $y_{n+1} = x_m$, $i_{n+1} = 1$ and $z = z_1 \cdots z_{n+1}$ (where the z_i will, of course, be either Λ or members of V').

LEMMA 3. *Let L be an FA language on a vocabulary V with grammar $G = \langle V, \mathbf{S}, \mathbf{R}\rangle$. Let $T = \langle V, V', \mathbf{S}_1, \mathbf{R}_1\rangle$ be a finite-state translator. Then the set of all translations of members of L by T is an FA language L'' on V' and a grammar G'' for L'' can be found effectively from G and T.*

PROOF. For the vocabulary of G'' take the set V'. For $\mathbf{S}''$ take a set of ordered triples $\langle a, b, c\rangle$, where $a \in S$, $b \in S_1$ and $c \in V$ or $c = \Lambda$. $\mathbf{R}''$ is defined as follows:

(a) Whenever $\langle S_1, S_2, x\rangle$ is a rule of G, $x \in V$, and $\langle t_1, x, t_2, z, 0\rangle$ is a rule of T we introduce the rule $\langle\langle t_1, S_1, x\rangle, \langle t_2, S_1, x\rangle, z\rangle$ into $\mathbf{R}''$.

(b) Whenever $\langle S_1, S_2, x\rangle$ is a rule of G, $x \in V$, and $\langle t_1, x, t_2, z, 1\rangle$ is a rule of T we introduce the rules $\langle\langle t_1, S_1, x\rangle, \langle t_2, S_2, y\rangle, z\rangle$ into $\mathbf{R}''$ for every $y \in V'$ or $y = \Lambda$.

(c) If $\langle S_1, S_2, \Lambda\rangle$ is a rule of G, then for every t_1 we introduce the rules $\langle\langle t_1, S_1, \Lambda\rangle, \langle t_1, S_2, y\rangle, \Lambda\rangle$ for every $y \in V'$ or $y = \Lambda$.

We also introduce two more states I and F to be the initial and final states of G'', and the rules $\langle I, \langle t_0, S_0, y\rangle, \Lambda\rangle$, where $y \in V'$ or $y = \Lambda$, and $\langle\langle t, S_f, \Lambda\rangle, F, \Lambda\rangle$, where S_0, S_f are the initial and final states of G, t is any state of T and t_0 is the initial state of T.

Now it is easy to see that G'' produces exactly the translations of sentences produced by G. Consider the following cases:

(a) G is in state S_1, moves to state S_2 and produces x. The translator T in state t_1 translates x as z and the rule used is $\langle t_1, x, t_2, z, 0\rangle$. Then, correspondingly, G'' in state $\langle t_1, S_1, x\rangle$ produces z and moves to state $\langle t_2, S_1, x\rangle$. This continues until case (b) is obtained.

(b) G is in state S_1, moves to state S_2 and produces x. The translator T in state t_1 translates x as z and the rule used is $\langle t_1, x, t_2, z, 1\rangle$. This means, then, that the translator is finished translating x. Then G'' in state $\langle t_1, S_1, x\rangle$ produces z and may move to $\langle t_2, S_2, y\rangle$ for *any* y. Thus it is ready to translate the next symbol that G may produce.

(c) G moves from S_1 to S_2 and produces nothing. Then G'' moves from $\langle t_1, S_1, \Lambda\rangle$ to $\langle t_1, S_2, y\rangle$ for any y. The translator is unaffected.

Thus the second and third parts of the states of G'' trace out the states of G and symbols produced by G, while the first part traces out the reaction of T.

PROOF OF THEOREM 4. The language $L^\circ = \{a^n b^m a^n \mid n, m \in J\}$ can be easily shown to be CF but not FA. (See Chomsky [4].) Consider the language $L = \{A^n B^m A^n\}$, where each A has the form $ce^k c$ for some $k > 0$. Each B has the form $df^k d$ for some $k > 0$. Consider the translator T defined by $V = \{a, b\}$, $V' = \{c, d, e, f\}$, $\mathbf{S} = (S_0, S_1, S_2, S_3, S_4)$, $S_4 = S_f$, and the rules

$$(S_0, a, S_1, c, 0), \quad (S_1, a, S_1, e, 0), \quad (S_1, a, S_2, e, 0), \quad (S_2, a, S_0, c, 1).$$
$$(S_0, b, S_3, d, 0), \quad (S_3, b, S_3, f, 0), \quad (S_3, b, S_4, f, 0), \quad (S_4, b, S_0, d, 1).$$

The language L is the map of L° under T.

On the other hand, define T' by $V = \{c, d, e, f\}$, $V' = \{a, b\}$, $\mathbf{S} = \{S_0, S_1, S_2\}$ and the rules

$$\langle S_0, c, S_1, a, 1\rangle, \quad \langle S_1, e, S_1, \Lambda, 1\rangle, \quad \langle S_1, c, S_0, \Lambda, 1\rangle.$$
$$\langle S_0, d, S_2, b, 1\rangle, \quad \langle S_1, f, S_1, \Lambda, 1\rangle, \quad \langle S_1, d, S_0, \Lambda, 1\rangle.$$

Then L° is the map of L under T'.

Now consider any FA language $L' \subseteq L$. Then $T(L') \subseteq T(L) = L^\circ$. But $T(L')$ is FA and L° is not. Hence L° must contain a string x not in $T(L')$. A grammar for

$T(L')$ can be found effectively by Lemma 3, and it is easy to see how a grammar G° for $T(L') \cup \{x\}$ can be found effectively. (L° obviously has a decision procedure for membership. So do FA languages. We take the first x in $L^\circ - T(L')$ and construct the grammar, using x and the grammar for $T(L')$.)

But if $L^{\circ\prime}$ is the language generated by G°, then $L^{\circ\prime} \supset T(L')$. Hence $T'(L^{\circ\prime}) \supset T'T(L') \supset L'$ and, in fact, must contain the infinitely many sentences obtained from x which cannot be in L'. Q.E.D.

2.4. Special Type 1 Languages

Definition 18. A type 1 grammar G is said to be of type 1_A if there exists a function f from V into the nonnegative integers such that if $\varphi\alpha\psi \rightarrow \varphi\beta\psi$ is a rule with β in V, then $f(\beta) < f(\alpha)$.

Definition 19. A type 1 grammar is said to be of type 1_B if there are no rules of the form $\varphi A\psi \rightarrow \varphi B\psi$ with $A, B \in V_N$.

Corollary. *A type 1_B grammar is of type 1_A.*

Proof. Let $\begin{cases} f(\alpha) = 1 \text{ if } \alpha \in V_N, \\ f(\alpha) = 0 \text{ if } \alpha \in V_T. \end{cases}$ Q.E.D.

Note that if G is of type 1_B, $x \in L$, D is a derivation of x, x_1 is a phrase of x of type A with respect to D and x_1 is a phrase of x of type B with respect to D, then $A = B$. For if $A \neq B$, then we would have $\alpha A\omega \Rightarrow \alpha B\omega$ or $\alpha B\omega \Rightarrow \alpha A\omega$ for some α, ω. But this is impossible.

Remark. It is not difficult to show that there exist a type 1 grammar G and strings $\varphi AB\psi$, $\varphi BA\psi$ such that $\varphi AB\psi \Rightarrow \varphi BA\psi$. Such situations are obviously "unfortunate" from a grammatical point of view, and the following theorem shows this does not happen in type 1_A grammars.

Theorem 5. *Let L be a type 1_A language generated by a type 1_A grammar G. Let φ and ψ be two distinct strings on V such that $\varphi \Rightarrow \psi$. Then the commutative images of φ and ψ are distinct. (See definition 13.)*

Proof. Extend the function f of Definition 18 to all strings on V by taking $f(\omega\eta) = f(\eta\omega) = f(\omega) + f(\eta)$. Then f can be thought of as a function on $\Phi(L)$. But if φ and ψ have the same length and $\varphi \rightarrow \psi$, then $f(\varphi) > f(\psi)$. Hence if φ and ψ have the same length, $\varphi \neq \psi$ and $\varphi \Rightarrow \psi$, then $f(\varphi) > f(\psi)$. Hence $\Phi(\varphi) \neq \Phi(\psi)$. Q.E.D.

Theorem 6. *Every type 2 language is a type 1_B language.*

Proof. Theorem 6 is proved by Lemma 2. Q.E.D.

Theorem 7. *There are languages of type 1_B which are not of type 2.*

Proof:[1] Consider the rules

(a) $S \rightarrow cXd$
(b) $X \rightarrow CXXXXXD$

$$1 \begin{cases} XXDXX \rightarrow XXXEXX \qquad XXDXd \rightarrow XXXEXd \\ EXX \rightarrow EXDX \qquad EXd \rightarrow EXDd \\ XEXD \rightarrow XXXXXD \end{cases}$$

$$2 \begin{cases} XXCXX \rightarrow XXFXXX \qquad cXCXX \rightarrow cXFXXX \\ XXF \rightarrow XCXF \qquad cXF \rightarrow cCXF \\ CXFX \rightarrow CXXXXX \end{cases}$$

$$3 \begin{cases} cC \rightarrow cc \\ cX \rightarrow ce \\ eX \rightarrow ee \\ Dd \rightarrow dd \end{cases}$$

[1] There was an error in the original proof which was pointed out by several people.

Now notice that X can turn terminal only if preceded by c or e; similarly, C and D can turn terminal only in a very limited way, when they have moved to the extreme left or right, respectively. Careful analysis shows that the rules of group 1 allow a D to move right over an X only by quintupling it. Similarly, a C can move left across X's, quintupling them. But a D and C cannot cross; hence, given two applications of rule b, one of the applications must occur "within" the other, or else the C or D will get "stuck" and not lead to a terminal string. It follows that all terminal strings have the form

$$\# \, c^{n+1} e^{5^n} d^{n+1} \, \#, \qquad n \geq 0.$$

Furthermore, all of these strings can be generated, if one begins with a derivation of $\# \; c \; (CXX)^n \; X \; (XXD)^n \; d \; \#$ and moves the C's and D's to the ends.

This language is not CF, by Corollary 2 to Theorem 2. Q.E.D.

3. *A Remark on the Reduction of CF Grammars to a Question Regarding Free Rings*

Let G be a CF grammar with vocabulary V. Consider the free ring $\mathbf{R}$ generated by V. Define an operator Θ over $\mathbf{R}$ as follows:

(a) If a $\in V_T$, $\Theta(a) = a$.

(b) If $A \in V_N$ and $A \to \omega_i$ are the rules associated with A, then $\Theta(A) = \sum_i \omega_i$.

(c) $\Theta(\eta\eta') = \Theta(\eta)\Theta(\eta')$.

(d) $\Theta(\eta+\eta') = \Theta(\eta) + \Theta(\eta')$.

Then the generable strings are precisely the ones that appear as terms in some expression $\Theta^n(\# S \#)$ for some n.

For example, let $V = \{\#, a, b, A, S\}$; $S \to AS$, $A \to ab$, $A \to cd$, $S \to aAa$. Then

$$\Theta(S) = AS + aAa$$

$$\Theta(A) = ab + cd$$

$$\Theta(a) = a, \; \Theta(b) = b, \; \Theta(\#) = \#.$$

Now

$$\Theta(\# S \#) = \# AS \# + \# aAa \#,$$

$$\Theta^2(\# S \#) = \# abAS \# + \# cdAS \# + \# abaAa \# + \# cdaAa \# + \# aaba \# + \# acda \#, \quad \text{etc.,}$$

and every derivable string will eventually appear on the right-hand side. Every sentence will be always on the right-hand side after a certain point. (*Note*: Θ is a homomorphism of $\mathbf{R}$ into itself. Moreover, any homomorphism that fixes V_T comes from a CF grammar.)

REFERENCES

1. Bar-Hillel, Y., Perles, M., and Shamir, E. On formal properties of simple phrase structure grammars. *Z. Phonetik, Sprachwissen. Komm. 14* (1961), 143–172; in Y. Bar-Hillel, *Language and Information*, Addison-Wesley, Reading, Mass., 1965, pp. 116–150.

2. Chomsky, N. Three models for the description of language. *IRE Trans. IT-2*, 3 (1956), 113–124.
3. —— and Miller, G. A. Finite state languages. *Inform. and Contr. 1* (1958), 91–112.
4. ——. On certain formal properties of grammars. *Inform. and Contr. 2* (1959), 137–167.
5. Scheinberg, S. On Boolean properties of phrase structure grammars. *Inform. and Contr. 3* (1960), 372–375.
6. Schützenberger, M. P. Some remarks on Chomsky's context-free languages. MIT Quart. Progr. Rep. 63, Oct. 1961, p. 155ff.