THE MITRE CORPORATION
Bedford, Massachusetts

Working Paper W-6671
Sheet 1 of 12 Sheets

Subject: GRAMMARS OF NUMBER THEORY: SOME EXAMPLES (U) Project 702

To: Distribution List

From: A. M. Zwicky, Jr.

Dept: D-15

Issued at: Bedford, Massachusetts

Date: 20 November 1963

Contract No. AF19(628)2390

Release to: Distribution List per: D.E. Walker
D. E. Walker

Abstract

The language K' generated by a grammar is said to represent the set K of all true statements in a given number-theoretic statement-form if there is an isomorphism of a special type between K and K'. In this paper two kinds of representation and several classes of languages are distinguished on the basis of (1) differing definitions of language generated by a grammar and (2) differing conditions on the form of the rules in the grammars. The grammars presented generate languages representing some of the standard number-theoretic forms; the associated functions and predicates are all primitive recursive.

Foreword

During the summer of 1963 a group of linguists, logicians, and mathematicians joined the Information Sciences Subdepartment of D-15, System Sciences, to work on a number of well-defined tasks for Project 702, Language Processing Techniques. This paper, the result of one of these special studies, is in the area of mathematical formalizations. In our development of a program to establish natural language as an operational language for command and control, logico-mathematical formalism is basic to: (1) defining the complex aspects of linguistic structure in generative grammars; (2) developing translation algorithms that relate structural descriptions of sentences to representations of data; and (3) solving mathematical problems of translatability between formal languages of differing complexity.

The material presented in this paper is part of a broad investigation of the generative capacity of various kinds of grammars. The objective of investigating the sets represented by context-sensitive languages is to shed some light on the question of the equivalence of deterministic and nondeterministic linear-bounded automata (Kuroda, 1963; Myhill, 1960).

The author is indebted to Joyce Friedman for her suggestions and for her correction of an early draft of this paper.

GRAMMARS OF NUMBER THEORY: SOME EXAMPLES

I. Representation

Consider a statement-form of formalized elementary number theory --e.g., $\bar{x}_1 + \bar{x}_2 = \bar{x}_3$-- where for any positive integer M, $\bar{M}$ is the numeral $\underbrace{111\ldots1}_{M}$, and where $\bar{x}_i$ is the ith occurrence, in the statement-form, of a variable over numerals. A set K' of strings of symbols from the alphabet $R = \{\#, a_1, a_2, a_3, \ldots\}$ strongly represents the set K of all true statements in a given statement-form F (with variables $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n$) if the strings in K' are just those strings

$$\#a_1^{M_1}a_2^{M_2}\ldots a_n^{M_n}\# \text{ (i.e., } \#\underbrace{a_1\,a_1\ldots a_1}_{M_1}\,\underbrace{a_2\,a_2\ldots a_2}_{M_2}\ldots\,\underbrace{a_n\,a_n\ldots a_n}_{M_n}\#\text{)}$$

for which F_i is true, where F_i is the statement obtained from F by substituting $\bar{M}_i$ for $\bar{x}_i$ for each i, $1 \leq i \leq n$.

For any string α of symbols of R let $N'_i(\alpha)$ be the number of occurrences of a_i in α; for any statement β in a statement-form F let $N_i(\beta)$ be the number of occurrences of the numeral 1 in β in the place of $\bar{x}_i$ in F. Then K' represents K if there is an isomorphism between K' and K such that if $\alpha \in K'$ is associated with $\beta \in K$, then $N'_i(\alpha) = N_i(\beta)$ for all i. Clearly, if K' strongly represents K, then K' represents K. If K' represents K, but not strongly, then K' weakly represents K.

The particular sets K' that we shall consider are languages generated by phrase structure grammars (Chomsky, 1963). Let us distinguish two definitions of the language generated by such a grammar: In a blocking grammar the generated language is the set of all #S#-derivatives that cannot be rewritten by the rules of the grammar; in an ordinary grammar

the generated language is the set of all #S#-derivatives which contain only terminal symbols and which cannot be rewritten by the rules of the grammar. An ordinary grammar is unfortunate if there is a #S#-derivative which cannot be rewritten by the rules but which contains a nonterminal symbol; otherwise, the grammar is fortunate.

If a grammar G generates a language K that represents the set K' associated with a form F, then G (as well as K) can be said to represent K', and both G and K can be said to represent F.

In the examples given in Section II all grammars are ordinary and fortunate; in the examples given in Section III blocking grammars for K are presented first (since they are, in general, simpler than ordinary grammars); fortunate (ordinary) grammars, second. To avoid extensive use of subscripts, x,y,z,a,b, and c are used throughout in place of x_1, x_2, x_3, a_1, a_2, and a_3, respectively. Capital Latin letters are nonterminal symbols in ordinary grammars (Section II); in blocking grammars (Section III) all letters are capitalized. Examples of grammars generating some of the sets K are also provided in Section II.

II. Linear and Finite State Languages

A linear grammar is a context-free grammar in which no rule contains more than one nonterminal symbol in its right-hand side. A finite state grammar is a linear grammar in which nonterminal symbols in the right-hand side of rules are either all leftmost or all rightmost.

$\overline{x} + \overline{y} = \overline{z}$ is strongly represented by the linear grammar (1a)

$$S \longrightarrow a\begin{Bmatrix} S \\ T \end{Bmatrix} c$$

$$T \longrightarrow b(T)c$$

$\overline{x} + \overline{y} = \overline{z}$ is weakly represented by the following finite state grammar, where $K' = \{(ac)^x (bc)^y \mid x \geq 1 . y \geq 1\}$: (1b)

$$S \longrightarrow ac\begin{Bmatrix} S \\ T \end{Bmatrix}$$

$$T \longrightarrow bc(T)$$

$\{\overline{x} + \overline{y} = \overline{z} \mid x \geq 1 . y \geq 1 . x + y = z\}$ is generated by the linear grammar (1c)

$$S \longrightarrow 1\begin{pmatrix} S \\ +T \end{pmatrix}1$$

$$T \longrightarrow 1\begin{pmatrix} T \\ = \end{pmatrix}1$$

$\overline{x} < \overline{y}$ is strongly represented by the linear grammar (2a)

$$S \longrightarrow a\begin{Bmatrix} S \\ T \end{Bmatrix}b$$

$$T \longrightarrow (T)$$

$\overline{x} < \overline{y}$ is weakly represented by the following finite state grammar, where $K' = \{(ab)^x b^z \mid x \geq 1 . z \geq 1\}$: (2b)

$$S \longrightarrow ab\begin{Bmatrix} S \\ T \end{Bmatrix}$$

$$T \longrightarrow b(T)$$

$\{\overline{x} < \overline{y} \mid x \geq 1 . y \geq 1 . x < y\}$ is generated by the linear grammar (2c)

$$S \longrightarrow 1\begin{pmatrix} S \\ <T \end{pmatrix}1$$

$$T \longrightarrow (T)1$$

The forms $\overline{x} \leq \overline{y}$, $\overline{x} > \overline{y}$, and $\overline{x} \geq \overline{y}$ can be represented by grammars differing only slightly from the ones above.

$\overline{m} \cdot \overline{x} + \overline{n} = \overline{y}$ (m and n fixed and ≥ 1) is strongly represented by the linear grammar (3a)

$$S \longrightarrow a\begin{Bmatrix} S \\ b^n \end{Bmatrix}b^m$$

If $n = 0$, the form is $\overline{m} . \overline{x} = \overline{y}$ and the grammar reduces to

$$S \longrightarrow a(S)b^m$$

$\overline{m} \cdot \overline{x} + \overline{n} = \overline{y}$ is weakly represented by the following finite state grammar, where $K' = \{(ab^m)^x b^n \mid x \geq 1\}$: (3b)

$$S \longrightarrow ab^m \begin{Bmatrix} S \\ b^n \end{Bmatrix}$$

$\{\overline{m} \cdot \overline{x} + \overline{n} = \overline{y} \mid x \geq 1 \cdot y \geq 1 \cdot y = mx + n\}$ is generated by the linear grammar (3c)

$$S \longrightarrow 1^m \cdot T$$

$$T \longrightarrow 1 \begin{Bmatrix} T \\ +1^n \end{Bmatrix} = 1^n \; 1^m$$

III. Context-Sensitive Languages[1]

Rules in this section are all of the form $\varphi_1 \rightarrow \varphi_2$, where the length of φ_2 is no less than the length of φ_1. Hence, for each grammar, there is an equivalent context-sensitive grammar (Chomsky, 1959).

$\overline{x}^2 = \overline{y}$ is represented by the blocking grammar (4a)

$$S \longrightarrow (P)AB$$

$$P \longrightarrow D(P)$$

$$DA \longrightarrow \begin{Bmatrix} AEED \;/\; \_\_\_\_\_A \\ AAEEB \;/\; \_\_\_\_\_B \end{Bmatrix}$$

$$EA \longrightarrow AE$$

$$E \longrightarrow B \qquad / \; \_\_\_\_\_B$$

Explanation: The strings D^nAB (for $n \geq 0$) are generated. As each D moves right, two E's are produced for each A. When a D reaches the rightmost A, two E's and an extra A are produced, and the D becomes a B. E's

---

[1] In this section only strong representations are given. It is easy, for most of the forms, to show that no weak representation by context-free grammars is possible. See R. J. Parikh (1961) for a discussion of semi-linear sets of integers and their relation to context-free grammars.

become B's before B's, so that to a string with m A's, one A and 2m+1 B's are added. That is,

$$S \begin{cases} \rightarrow AB \\ \rightarrow D^nAB \end{cases} \qquad (n \geq 1)$$

$$D^nAB \Longrightarrow D^{n-1}AABB^2B = D^{n-1}A^2B^4$$

$$\Longrightarrow D^{n-m+1}A^mB^{m^2}$$

$$\Longrightarrow D^{n-m}A^mABB^{2m}B^{m^2} = D^{n-m}A^{m+1}B^{(m+1)^2}$$

$$\Longrightarrow DA^nB^{n^2}$$

$$\Longrightarrow A^{n+1}B^{(n+1)^2}$$

The preceding grammar can be made into a fortunate grammar by the addition of the rules (4b)

(*) $A \longrightarrow a \;/\; \begin{Bmatrix} \# \\ a \end{Bmatrix} \_\_\_\_$

(**) $B \longrightarrow b \;/\; \begin{Bmatrix} a \\ b \end{Bmatrix} \_\_\_\_$

Without the contexts in the rules (*) and (**), the grammar would be unfortunate.

Square ($\overline{x}$) is represented by the blocking grammar (5a)

$S \longrightarrow (B(P))A$

$P \longrightarrow C(P)$

$CA \longrightarrow ADDC$ $\qquad$ $BA \longrightarrow AAB$

$CD \longrightarrow DC$ $\qquad$ $BD \longrightarrow AB$

$C \longrightarrow A \;/\; \_\_\_\_\_ \#$ $\qquad$ $B \longrightarrow A \;/\; \_\_\_\_\_ \#$

Explanation: The strings A and BC^nA ($n \geq 0$) are generated. C's move to the right through A's and D's, producing two D's for each A but leaving D's as they are. The single B moves right, tripling each A and changing

D's to A's, but not moving through C's. The B and the C's both become A's when they reach the right-hand end of the string. Then

$$S \begin{cases} \to A \\ \to BA \longrightarrow A^3B \longrightarrow A^4 \\ \to BC^nA \quad (n \geq 1) \end{cases}$$

$$\begin{aligned} BC^nA &\Longrightarrow BA(DDC)^n \\ &\Longrightarrow A^3B(DDC)^n \\ &\Longrightarrow A^3BD^{2n}C^n \\ &\Longrightarrow A^3A^{2n}BC^{n-1}A = A^{2n+3}(BC^{n-1}A) \\ &\Longrightarrow A^{2n+3}A^{2(n-1)+3}(BC^{n-2}A) \\ &\Longrightarrow A^{\sum_{i=1}^{n}(2i+3)}BC = A^{n^2+4n}BC \\ &\Longrightarrow A^{n^2+4n}A^4 = A^{(n+2)^2} \end{aligned}$$

The addition of rule (*) makes the above grammar fortunate. (5b)

$\overline{n} \mid \overline{x}$ (n fixed) is represented by the blocking grammar (6a)

$$S \longrightarrow \begin{Bmatrix} A \\ C \end{Bmatrix} A^{n-1}$$

$$CA \longrightarrow BAC$$

$$CB \longrightarrow BC$$

$$C \longrightarrow \begin{Bmatrix} BE \\ BD \end{Bmatrix} / \_\_\_\_\_ \#$$

$BE \longrightarrow EA$ $\qquad$ $BD \longrightarrow DB$

$AE \longrightarrow EA$ $\qquad$ $AD \longrightarrow DA$

$E \longrightarrow A \ / \ \#\_\_\_\_\_$ $\qquad$ $D \longrightarrow C \ / \ \#\_\_\_\_\_$

Explanation: The strings A^n and CA^{n-1} are generated. C moves to the right through the A's, adding a B for each A. At the right-hand end of the string, one B is added, and C becomes either E or D. E moves back to the left, changing B's and C's into A's and finally becoming an A itself. D moves back to the left without changing any symbols, becoming a C when it reaches the left-hand end; then C moves right again, passing through B's and adding a B for each A. That is,

$$S \begin{cases} \nearrow A^n \\ \searrow CA^{n-1} \end{cases}$$

$$CA^{n-1} \Longrightarrow (BA)^{n-1}C \longrightarrow (BA)^{n-1}B \begin{Bmatrix} E \\ D \end{Bmatrix}$$

$$(BA)^{n-1}BE \Longrightarrow EA^{2n-1} \longrightarrow A^{2n}$$

$$(BA)^{n-1}BD \Longrightarrow D(BA)^{n-1}B \longrightarrow C(BA)^{n-1}B$$

$$\Longrightarrow (BBA)^{n-1}BC \longrightarrow (BBA)^{n-1}BB \begin{Bmatrix} E \\ D \end{Bmatrix}$$

$$(BBA)^{n-1}BBE \Longrightarrow EA^{3n-1} \longrightarrow A^{3n}$$

$$(BBA)^{n-1}BBD \Longrightarrow C(BBA)^{n-1}B^2$$

etc.

The preceding grammar can be made into a fortunate grammar by adding the rule (*). (6b)

$\overline{x} \mid \overline{y}$ is represented by the blocking grammar (7a)

$$S \longrightarrow \begin{Bmatrix} A(S)B \\ AC \end{Bmatrix}$$

$$CB \longrightarrow DBC$$

$$CD \longrightarrow DC$$

$$C \longrightarrow \begin{Bmatrix} DE \\ DF \end{Bmatrix} / \_\_\_\_\_ \#$$

$DE \longrightarrow EB$ $\qquad$ $DF \longrightarrow FD$

$BE \longrightarrow EB$ $\qquad$ $BF \longrightarrow FB$

$E \longrightarrow B$ / A _____ $\qquad$ $F \longrightarrow C$ / A _____

Explanation: The strings AB and A^nCB^{n-1} ($n \geq 1$) are generated, and CB^{n-1} is treated exactly as CA^{n-1} was in (6a).

The preceding grammar can be made into a fortunate grammar (7b) if the last two rules are replaced by

$E \longrightarrow b$ / a _____ $\qquad$ $F \longrightarrow C$ / a _____

and if the rules (*) and (**) are added.

$\overline{n}^{\overline{x}} = \overline{y}$ (n fixed) is represented by the blocking grammar (8a)

$$S \longrightarrow AB^n(P)$$

$$P \longrightarrow (P)A$$

$$BA \longrightarrow AB^n$$

Explanation: The strings AB^nA^m ($m \geq 0$) are generated. Each of A's on the left moves to the right, producing n B's for each B in the string. That is,

$$S \begin{cases} \longrightarrow AB^n \\ \longrightarrow AB^nA^m \qquad (m \geq 1) \end{cases}$$

$$AB^nA^m \Longrightarrow AB^{n-1}AB^nA^{m-1}$$

$$\Longrightarrow AA(B^n)^nA^{m-1} = A^2B^{n^2}A^{m-1}$$

$$\Longrightarrow A^iB^{n^i}A^{m-i+1}$$

$$\Longrightarrow A^{i}A(B^{n})^{i}A^{m-i} = A^{i+1}B^{n}{}^{i+1}A^{m-1}$$

$$\Longrightarrow A^{m}B^{n}{}^{m}A$$

$$\Longrightarrow A^{m+1}B^{n}{}^{m+1}$$

The preceding grammar can be made into a fortunate grammar by the addition of rule (*) and the rule (8b)

$$B \longrightarrow b \;/\; \_\_\_\_\_ \begin{Bmatrix} \# \\ b \end{Bmatrix}$$

$x \cdot y = z$ is represented by the blocking grammar (9a)

$$S \longrightarrow PQ$$

$$P \longrightarrow B(P)$$

$$Q \longrightarrow (Q)A$$

$$BA \longrightarrow ABC$$

$$CA \longrightarrow AC$$

$$CB \longrightarrow BC$$

Explanation: The strings $B^{m}A^{n}$ ($m \geq 1$, $n \geq 1$) are generated. The A's move left through the B's, generating a C for each B. The nm C's then move to the right-hand end of the string.

The grammar above cannot be adjusted in the simple way to make (9b) it fortunate. The following grammar, based on (9a), accomplishes this by the use of a number of extra marker symbols; the rules will be given in groups, with an explanation of their operation for each group.

$$S \longrightarrow PQ$$

$$P \longrightarrow D(M) \qquad\qquad Q \longrightarrow (N)E$$

$$M \longrightarrow B(M) \qquad\qquad N \longrightarrow (N)A$$

The strings $DB^{y-1}A^{x-1}E$ ($x \geq 1$, $y \geq 1$) are produced. The next group of rules cannot apply until this production is completed.

$$BA \longrightarrow ABC$$

$$DA \longrightarrow ADC$$

$$BE \longrightarrow EBC$$

$$CA \longrightarrow AC$$

$$CE \longrightarrow EC$$

$$CB \longrightarrow BC$$

D is simply the leftmost B in (8a), and E the rightmost A. Each B moves right through the A's and through E, producing a C as it does so; D moves right through the A's, also producing a C each time. C's can move right through A's, B's, or E. When none of the first five rules in this group applies, the string $DB^{y-1}A^{x-1}E$ has been transformed into $A^{x-1}DEX$, where X contains the original (y-1) B's and the newly created (xy-1) C's, although not necessarily in that order. The next set of rules cannot apply until this point is reached.

$$DE \longrightarrow abF$$

$$A \longrightarrow a \;/\; \_\_\_\_\_\; a$$

$$FB \longrightarrow bF$$

$$FC \longrightarrow CG$$

$$GC \longrightarrow CG$$

$$CGB \longrightarrow GBC$$

$$bGB \longrightarrow bbG$$

$$G \longrightarrow H \;/\; \_\_\_\_\_\; \#$$

$$CH \longrightarrow HC$$

$$H \longrightarrow C \;/\; b \;\_\_\_\_\_$$

The first rule in this group changes $A^{x-1}DEX$ into $A^{x-1}abFX$. All A's to the left of this a becomes a's themselves. F moves to the right through B's, changing them into b's. When F first comes across a C, F

moves through it and becomes a G. G then moves through any further C's and changes into an H at the end of the string (in which case all the B's and C's were in the right order to start with). If G comes across a B, however, this B is out of order, and the G moves back to the left (carrying the B with it) until it meets the rightmost b --at which point the B that has been carried along becomes the new rightmost b. G is then in position to move to the right again. When all the errant B's have been put in their places (and turned into b's), G can move to the right end of the string and become an H. H moves back to the left through C's, changing them into c's as it goes and finally becoming the (xy)th c when it meets the rightmost b. The result is the string $a^x b^y c^{xy}$.

Fibonacci number ($\overline{x}$) is represented by the blocking grammar (10a)

$$S \longrightarrow \begin{Bmatrix} A \\ AA \\ AAA \\ CAEE \end{Bmatrix}$$

$$CA \longrightarrow EC$$

$$CE \longrightarrow EC$$

$$BE \longrightarrow AEB$$

$$BA \longrightarrow EB$$

$$B \longrightarrow \begin{Bmatrix} D \\ F \end{Bmatrix} / \underline{\quad} \#$$

$AD \longrightarrow DA$ $\qquad$ $AF \longrightarrow FA$

$ED \longrightarrow DA$ $\qquad$ $EF \longrightarrow FE$

$D \longrightarrow A \ / \ \# \underline{\quad}$ $\qquad$ $F \longrightarrow CE \ / \ \# \underline{\quad}$

Explanation: The strings A, AA, AAA, and CAEE are generated. The C moves to the right, changing A's to E's. When C reaches the first E, it changes the E to an A and becomes a B. This B then moves through

the rest of the E's, producing an A for each E; B (like C) also changes A's to E's as it moves to the right through them. Suppose we consider B as representing an E. Then at this point in the derivation there are as many E's as there were A's and E's together, and as many A's as there were E's. Given an initial string in which there are F_nA's (where F_n is the nth Fibonacci number) and F_{n+1}E's, we have converted it into a string with F_{n+1}A's and F_{n+2}E's. At this point, B becomes either D or F. D moves back to the left, changing all symbols to A's and becoming an A itself at the left-hand end of the string; there are F_{n+3} A's in the resultant string. F simply moves back to the left-hand end, changing nothing and finally becoming CE, so that the C is ready to run through the F_{n+1}A's and F_{n+2}E's.

The addition of the rule (*) makes the above grammar fortunate. (10b)

$\bar{x}! = \bar{y}$ is represented by the fortunate grammar (11)

$$* \quad S \longrightarrow \begin{Bmatrix} ab \\ aabb \\ DBBAE \end{Bmatrix}$$

$$BA \longrightarrow ABC$$

$$* \quad BE \longrightarrow EBC$$

$$CA \longrightarrow AC$$

$$CE \longrightarrow EC$$

$$CB \longrightarrow BC$$

$$* \quad DA \longrightarrow AD$$

$$* \quad DE \longrightarrow \begin{Bmatrix} aa \\ FGH \end{Bmatrix}$$

$$AF \longrightarrow FG$$

$$F \longrightarrow K \ / \ \# \_\_\_\_\_$$

$$H \begin{Bmatrix} B \\ C \end{Bmatrix} \longrightarrow IH$$

$$H \longrightarrow J \ / \ \_\_\_\_\_ \ \#$$

$$GI \longrightarrow IG$$

$$KI \longrightarrow IK$$

$$GJ \longrightarrow JG$$

$$G \longrightarrow \left\{ \begin{array}{l} E \;/\; \underline{\quad} \;\; \# \\ A \;/\; \underline{\quad} \left\{ \begin{array}{l} E \\ A \end{array} \right\} \end{array} \right\} \qquad I \longrightarrow \left\{ \begin{array}{l} D \;/\; \# \;\underline{\quad} \\ B \;/ \left\{ \begin{array}{l} D \\ B \end{array} \right\} \underline{\quad} \end{array} \right\}$$

$$KJ \longrightarrow BA \;/\; B\underline{\quad}A$$

$$A \longrightarrow a \;/\; \underline{\quad}a$$

$$\left\{ \begin{array}{l} B \\ C \end{array} \right\} \longrightarrow b \;/ \left\{ \begin{array}{l} a \\ b \end{array} \right\} \underline{\quad}$$

Explanation: The first two strings, ab and aabb, are generated directly. DBBAE will be used to generate all other strings. The D is a marker, and the E is a representative of an A. A's and the E move left through the B's, multiplying as in (9a). A's move through D without any change. Beginning with $DB^{n!}A^{n-1}E$ we have obtained $A^{n-1}DEX$, where X contains n! B's and $n \cdot n!$ C's, in some order. If DE becomes aa, the initial A's become a's, and the B's and C's all become b's, so that there are n+1 a's and $n!+n \cdot n! = (n+1)n! = (n+1)!$ b's. Otherwise, DE becomes FGH, in which case F moves left, changing A's to G's and H moves right, changing B's and C's to I's. The result of this is the string $FG^{n}I^{(n+1)!}H$. F then becomes K, and H becomes J. All the I's move to the far left, the G's to the far right, producing $I^{(n+1)!}KJG^{n}$. The leftmost I becomes D, the others become A's; the rightmost G becomes E, the others become A's, so that we have $DB^{(n+1)!-1}KJA^{n-1}E$. Only when all the I's and G's have been rewritten can KJ become BA, at which point we have $DB^{(n+1)!}A^{n}E$ and the multiplication can begin again.

Power $(\overline{x}) = \overline{y}$ (i.e., y is a positive power of x) is represented by the fortunate grammar obtained from the one in (11) by replacing the starred rules by (12)

$$S \longrightarrow \begin{Bmatrix} a(M)b \\ DTE \end{Bmatrix}$$

$$M \longrightarrow a(M)b$$

$$T \longrightarrow B(T)A$$

$$DA \longrightarrow ADC$$

$$BE \longrightarrow EB$$

$$DE \longrightarrow \begin{Bmatrix} ab \\ FH \end{Bmatrix}$$

$$KJ \longrightarrow BA \;/\; B \_\_\_\_\_ \begin{Bmatrix} E \\ A \end{Bmatrix}$$

Explanation: The terminal strings $a^n b^n$ are generated separately from the strings $DB^{n-1}A^{n-1}E$, which will be rewritten. Multiplication proceeds as in (11), except that this time the E passes through B's without multiplying, whereas A's multiply when passing the D. Starting with a string $DB^{n^i-1}A^{n-1}E$ $(i \geq 1)$, we obtain $A^{n-1}DEX$, where X contains n^i-1 B's and $(n-1)n^i = n^{i+1}-n^i$ C's. If DE becomes ab, the A's become a's, and both the B's and the C's become b's, so that the result is $a^{n-1}abb^{n^i-1}b^{n^{i+1}-n^i} = a^n b^{n^{i+1}}$. If DE becomes FH, the productions proceed as in (11), so that the result is $DB^{n^{i+1}-2}KJA^{n-2}E \longrightarrow DB^{n^{i+1}-1}A^{n-1}E$.

Exp $(\overline{x}, \overline{y}) = \overline{z}$ (i.e., $y^x = z$) is represented by the following (13) fortunate grammar, which is obtained from (12) by: substituting new rules for the initial rules, the rule expanding DE, and the termination rules; changing nonterminal symbols; and slightly altering the contexts of some rules.

$$S \longrightarrow \begin{Bmatrix} a(M)bc \\ abNc \\ aDTE \end{Bmatrix}$$

$$M \longrightarrow (M)a$$

$$N \longrightarrow b(N)C$$

$$T \longrightarrow C(T)B$$

$$CB \longrightarrow BCX$$

$$DB \longrightarrow BDX$$

$$XB \longrightarrow BX$$

$$XE \longrightarrow EX$$

$$XC \longrightarrow CX$$

$$CE \longrightarrow EC$$

$$DE \longrightarrow \begin{Bmatrix} Ybc \\ ZFH \end{Bmatrix}$$

$$BZ \longrightarrow ZB$$

$$BY \longrightarrow Yb$$

$$\begin{Bmatrix} Y \\ Z \end{Bmatrix} \longrightarrow a \;/\; a \_\_\_\_\_$$

$$\begin{Bmatrix} C \\ X \end{Bmatrix} \longrightarrow c \;/\; c \_\_\_\_\_$$

$$BF \longrightarrow FG$$

$$F \longrightarrow K \;/\; a \_\_\_\_\_$$

$$H\begin{Bmatrix} C \\ X \end{Bmatrix} \longrightarrow IH$$

$$H \longrightarrow J \;/\; \_\_\_\_\_ \#$$

$$GI \longrightarrow IG$$

$$KI \longrightarrow IK$$

$$GJ \longrightarrow JG$$

G $\begin{Bmatrix} E \,/\, \underline{\quad} \,\# \\ B \,/\, \underline{\quad}\{E\}\{B\} \end{Bmatrix}$ I $\begin{Bmatrix} D \,/\, a\,\underline{\quad} \\ C \,/\, \{D\}\{C\}\underline{\quad} \end{Bmatrix}$

KJ CB / C $\underline{\quad}\{E\}\{B\}$

Explanation: The strings $a^n bc$ and $ab^n c^n$ ($n \geq 1$) are generated separately from the strings $aDC^{n-1}B^{n-1}E$, in which $DC^{n-1}B^{n-1}E$ is treated similarly to $DB^{n-1}A^{n-1}E$ in (12), except that DE becomes Ybc or ZFH after each multiplication. The Y or Z then becomes an a to indicate that another power has been taken.

Arnold M. Zwicky, Jr.

A. M. Zwicky, Jr.

AMZ:nmc
Attachments: References
Distribution List

References

Chomsky, N. On certain formal properties of grammars. Information and Control, 1959, 2, 137-167.

Chomsky, N. Formal properties of grammars. In R. D. Luce, R. R. Bush, and E. Galanter (Eds.), Handbook of mathematical psychology. Vol. II. New York: Wiley, 1963, pp. 323-418.

Kuroda, S.-Y. Type 1 grammars and linear-bounded automata. Quarterly Progress Report No. 69, Research Laboratory of Electronics, M. I. T. April 15, 1963, 199-200.

Myhill, J. Linear bounded automata. WADD Technical Note 60-165. Wright-Patterson Air Force Base, June, 1960.

Parikh, R. J. Language generating devices. Quarterly Progress Report No. 60, Research Laboratory of Electronics, M. I. T. January 15, 1961, 199-212.

Distribution List

MITRE

F, G, H, I

D-06

J. M. Bartlett

D-15

D. E. Walker (50)
All Staff
Consultants
- N. Chomsky (M. I. T.)
- C. A. Wogrin (Yale University)

Summer Staff (1963)
- B. Hall
- H. Herzberger
- S. Isard
- R. C. Jeffrey
- C. B. Qualls
- P. R. Rosenbaum
- J. R. Ross
- E. A. Tabellario
- A. M. Zwicky, Jr. (20)

D-19

J. J. Baker
F. Cataldo
J. A. Clapp
R. A. Vicksell

D-23

R. P. Mayer

D-25

S. J. Hauser, Jr.
A. Neuman

D-27

P. R. Bagley
D. M. Liss
S. Okada

ESRC

M. E. Conway
J. F. Egan
S. J. Keyser (3)

ESRHI

J. B. Goodenough
H. Rubenstein