Applying Probability Measures to Abstract Languages

TAYLOR L. BOOTH AND RICHARD A. THOMPSON

Abstract—**The problem of assigning a probability to each word of a language is considered. Two methods are discussed. One method assigns a probability to a word on the basis of particular measurable features of the language. The second method is applied to languages $L(G)$ generated by a grammar G. A probability is associated with each production of G. These in turn define the word probabilities of each word in the language. The conditions for this assignment to be a probabilistic measure are derived.**

Several properties of probabilistic languages are developed.

Index Terms—**Abstract languages, probabilistic languages, stochastic automata, word functions.**

I. Introduction

A CONSIDERABLE number of information system design problems involve developing systems that process strings of symbols from a set V_T. The strings of interest belong to a set $L \subseteq V_T^*$, where V_T^* represents the set of all symbol strings that can be formed from V_T. Current practice is to design a system without regard to the relative likelihood of occurrence of the different strings of L. In most situations all strings are not equally likely to occur. Thus, if a measure of importance can be assigned to each string in L, it becomes possible to take into account the relative importance of each string when carrying out a system design.

This paper considers some of the possible ways that a measure can be assigned to a set of strings L. In particular, the relationship between these measures and the way that the set L is defined is investigated. This discussion assumes that the reader is familiar with automata theory, abstract language theory, and probabilistic automata theory (see [2], [5], [8], and [9].

II. Measurable Properties of Languages

Each $v \in V_T^*$ is called a word. The set V_T^* is a denumerable set. In the following it is assumed that the words of V_T^* are ordered in dictionary order. If $V_T = \{a, b\}$ then the words of V_T^* are $\lambda, a, b, aa, ab, ba, bb$, etc., where λ is the null word.

Manuscript received May 22, 1972; revised November 28, 1972. An earlier version of this paper was presented at the 10th IEEE Symposium on Switching and Automata Theory, October 1969.

T. L. Booth is with the Computer Science Group, Department of Electrical Engineering, University of Connecticut, Storrs, Conn. 06268.

R. A. Thompson was with the Computer Science Group, Department of Electrical Engineering, University of Connecticut, Storrs, Conn. 06268. He is now with the Department of Electrical Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Va. 24061.

Following Paz [9], a numerical value can be assigned to each word of V_T^*.

Definition 1—Word Function: A word function is a mapping

$$f(v)\colon V_T^* \to RN$$

of V_T^* to the set of real numbers RN.

Paz presents a number of general properties of word functions that are associated with stochastic sequential machines and the languages accepted by these machines. This paper is concerned with a different class of word functions.

Definition 2—Measurable Word Function: A word function $f(v)$, $v \in V_T^*$, is a measurable word function if

$$\sum_{\forall v \in V_T^*} f(v) = N < \infty.$$

Definition 3—Probabilistic Word Function: A measurable word function $f(v)$ is a probabilistic word function if

$$0 \leqslant f(v) \leqslant 1$$

$$\sum_{\forall v \in V_T^*} f(v) = 1.$$

In the following $p(v)$ is used to denote a probabilistic word function and $p(v_i)$ is called the probability of the word v_i.

Since V_T^* contains an infinite number of words it is not possible to simply list the value of $f(v)$ for all $v \in V_T^*$. This leads to the following.

Definition 4—Computable Word Function: A word function $f(v)$ is a computable word function if there exists an algorithm that can be used to compute $f(v)$ for every $v \in V_T^*$.

Inherent in this definition is the fact that only a finite number of steps are needed to compute $f(v)$ for any v and that the steps of the computation may involve real functions as well as steps that relate to the structural properties of v.

For example, assume $V_T = \{a, b\}$ and let $n_a(v)$ and $n_b(v)$ represent, respectively, the number of times the symbol a and the symbol b appears in the word v. Then

$$f(v) = \tfrac{1}{2} + e^{-n_a(v)} \cos(n_b(v))$$

is a computable word function. Of particular interest are probabilistic word functions that are also computable word functions.

Measurable word functions provide a very useful tool that can be used to assign a measure of importance to words of a language. Following normal conventions any $L \subseteq V_T^*$ is a language. Let $f(v)$ be a word function such that

$$\sum_{\forall v \in L} f(v) = N < \infty;$$

then $f(v)$ is a measurable word function over L. If $p(v)$ is a probabilistic word function such that

$$\sum_{\forall v \in L} p(v) = 1, \quad p(v) = 0, \quad v \notin L$$

then $p(v)$ is said to be the probability measure assigned to L. This paper studies how probability measures can be assigned to languages and investigates the relationship between the structural properties of the language and the characteristics of the probability measure.

III. Language

Although any subset of V_T^* is a language, the greatest interest in languages is upon phrase structure languages generated by a grammar G [5], [8]. This section summarizes the conventions that are of particular importance when studying how probability and other related measures may be assigned to a language.

A grammar G is represented by $\langle V_T, V_N, R, \sigma \rangle$ where V_N, V_T, R are, respectively, the finite set of nonterminal symbols or variables, terminal symbols, and productions or rewrite rules that define G and σ is the start symbol. Lowercase letters from the beginning of the English alphabet are used to indicate elements of V_T and uppercase letters are used to indicate elements of V_N. Greek letters are used to indicate strings from V^*, where $V^* = (V_N \vee V_T)^*$, and lowercase letters from the end of the English alphabet indicate words from V_T^*. The language generated by G is denoted $L(G)$.

Any production from R can be represented by the ordered pair $\zeta \rightarrow \alpha$. The string ζ is called the *premise* and the string α is called the *consequence* of the production rule.

The grammar associated with a given language determines the syntactic structure of each word of the language. However, in any situation in which the words are used to transfer information, not all words are of equal importance. This difference in importance can be indicated by associating a probabilistic word function with the language. Unless the language L is finite, this word function must be a computable word function if it is to be of any use in system analysis.

In the literature, for example [9], the set of words accepted by a probabilistic automaton with cut point η is called a probabilistic language. The word functions associated with these languages are generally not probabilistic word functions. To avoid confusion, the languages discussed in the following sections are called probability measure-languages.

Definition 5–Probability Measure-Language (pm-Language): A language $L \subseteq V_T^*$ is a probability measure-language (denoted *pm*-language) if there is a probabilistic word function $p(v)$ defined on L.

There are many ways that a word function can be defined on a language.

Definition 6–Consistent Algorithm: Let F be an algorithm that defines a computable word function $f(v)$. Then F is a consistent algorithm for the language L if

$$\sum_{\forall v \in L} f(v) = 1$$

$$f(v) = 0, \quad v \notin L$$

(i.e., $f(v)$ is a probabilistic word function for L).

The structure of the words of a language may be more important than the grammar used to generate it. This is particularly true when the information content of the language is related to a particular structural feature instead of to its grammar.

Definition 7–Parameter Set: A language L is characterized by a parameter set A if there exists a 1 to 1 mapping $g(a)$ of A onto L.

For example, let L be the context-sensitive language $L = \{ 0^i 1^i 0^i \mid i = 0, 1, 2 \cdots \}$ and let A be the set of all nonnegative integers. Then the mapping $g(a)$ defined by

$$g(i) \rightarrow 0^i 1^i 0^i$$

satisfied the conditions of the above definition and A characterizes L.

Definition 8–Parameter Set Probabilistic Word Functions: Let $p(a)$ be a probabilistic word function defined for the parameter set A that characterizes the language L. This probabilistic word function is associated with L by the following correspondence:

$$p(v) = p(a) \text{ iff } g(a) = v$$

where $v \in L, a \in A, g(a)$ is the characterization mapping.

For example, let $L = \{ 0^i 1^i 0^i \mid i = 0, 1 \cdots \}$ and $A = \{ 0, 1, 2, \cdots \}$. For every i define

$$p(i) = e^{-\lambda} \left[\frac{\lambda^i}{i!} \right];$$

then

$$p(0^i 1^i 0^i) = e^{-\lambda} \left[\frac{\lambda^i}{i!} \right]$$

is the resulting probability measure defined on L.

The rest of this paper deals mainly with languages generated by context-free or regular grammars.

IV. Probabilistic Grammars

A word function can be associated with a language generated by a grammar G by assigning probabilities to the productions of G. Not all of these word functions are probabilistic word functions. They are computable word functions.

Definition 9–Probabilistic Grammar (p-Grammar): A probabilistic grammar (denoted *p*-grammar) is

$$G = \langle V_T, V_N, R, P, \sigma \rangle$$

where V_T, V_N, R, and σ have their normal meaning and P is a set of probabilities that are assigned by a 1 to 1 mapping to the productions of R.

In the general case the probabilities associated with the productions of R would depend upon both the deterministic properties of the grammar and the sequence in which the productions are applied during the generation of a given string of the language. In [10], Salomma considers one way to assign probabilities to the sequence of productions that generate a given string of the language. This approach generates a word function on L that will not be a probabilistic word function except under special conditions. For this paper a different method is used to assign probabilities to the productions of regular and context-free grammars.

For any $v \in L(G)$, G context-free, there exists one or more sequences of productions that can be used to generate v. Let $\{\alpha_i\}$ be the set of intermediate strings in the rth possible leftmost derivation sequence used to generate v. This sequence is indicated by

$$\sigma \to \alpha_1 \to \alpha_2 \to \cdots \to \alpha_n = v$$

where the production $r_{i,i+1}$ is used to generate α_{i+1} from α_i.

The simplest situation occurs when the application of any production depends only upon the value of the leftmost nonterminal symbol in the current intermediate string. For this situation the rule set P is defined as follows.

Assume

$$V_N = \{\sigma = A_1, A_2, \cdots A_k\}.$$

For each $A_j \in V_N$ let the productions that rewrite A_j be of the form

$$A_j \to \beta_{j,u}, \quad u = 1, 2, \cdots m_j$$

where m_j is the number of rules with premise A_j. The probability of this production is denoted by

$$p_{j,u} = p(\beta_{j,u}/A_j), \quad 0 \leqslant p_{j,u} \leqslant 1.$$

The set of all such probabilities associated with the productions of R form the desired rule set P.

For example let $G = \langle V_T, V_N, R, P, \sigma \rangle$ be defined by

$$V_T = \{a, b\} \quad V_N = \{\sigma, A_2\}$$

$$R$$

$$\sigma \to aA_2A_2 \qquad A_2 \to A_2\sigma$$
$$\sigma \to b \qquad\qquad A_2 \to a.$$

Then P is defined by assigning values to the probabilities

$$p_{1,1} = p(aA_2A_2/\sigma) \qquad p_{2,1} = p(A_2\sigma/A_2)$$
$$p_{1,2} = p(b/\sigma) \qquad\qquad p_{2,2} = p(a/A_2)$$
$$m_1 = 2 \qquad\qquad m_2 = 2.$$

Definition 10–Unrestricted Grammar: A grammar G with rule set P is an unrestricted grammar if the probability of the application of any rewrite rule depends only upon the presence of a given premise in a derivation and not upon how the premise was generated.

The context-free grammar discussed above is an example of an unrestricted grammar. Only unrestricted grammars are considered in this paper.

Definition 11–Proper Grammar: A p-grammar is proper if

$$\sum_{u=1}^{m_j} p_{j,u} = 1$$

for all $A_j \in V_N$.

The probability of any $v \in L(G)$ is determined by first finding all of the leftmost derivations of v. The probability of each derivation is then found by forming the product of the rule probabilities associated with the productions used in the derivation. The probability of v is then the sum of all the probabilities of the individual derivations. For example using the above grammar the probability of $v = aaba$ is

$$p(v) = p(aA_2A_2/\sigma)\, p(A_2\sigma/A_2)\, p(a/A_2)\, p(b/\sigma)\, p(a/A_2)$$
$$= p_{1,1}\, p_{2,1}\, p_{2,2}\, p_{1,2}\, p_{2,2}$$

since

$$\sigma \to aA_2A_2 \to aA_2\sigma A_2 \to aa\sigma A_2 \to aabA_2 \to aaba$$

is the only leftmost derivation of v.

This leads to the following observation. Let $L(G)$ be the language generated by the p-grammar G. This p-grammar induces the following word function $f(v)$ on $L(G)$. For each $v \in L(G)$

$$f(v) = \sum_{n=1}^{N(v)} \prod_{k=1}^{K(v,n)} q_{n,k}(v)$$

where

- $N(v)$ Number of distinct leftmost derivations of v.
- $K(v, n)$ Number of steps in the nth derivation.
- $q_{n,k}(v)$ Probability of the production used at the kth step of the nth derivation of v.

It should be noted that since R is a finite set there are only a finite number of distinct values that $q_{n,k}(v)$ can assume. The function $f(v)$ is zero if $v \notin L$.

Definition 12–Consistent Grammar: A p-grammar G is a consistent grammar if and only if the word function $f(v)$ induced by G is a probabilistic word function.

As will be shown in the next section, not all p-grammars are consistent grammars.

Theorem 1: There exist pm-languages that cannot be generated by proper unrestricted context-free p-grammars.

Proof: Let

$$L = \{0^i1^i \mid i = 0, 1, 2, \cdots\}.$$

L can be generated by a context-free grammar. Let a probabilistic word function be assigned to L by using the parameter set mapping

$$p(0^i1^i) = e^{-a}\left[\frac{a^i}{i!}\right]$$

where a is any real number. Under this word function, L is a pm-language. If $p(0^i1^i)$ is determined by an unrestricted context-free p-grammar then for all i it must have the form

$$p(0^i1^i) = e^{-a}\left[\frac{a^i}{i!}\right] = \sum_{j=1}^{u} \prod_{m=1}^{k} p_m^{r_{m,j}(i)}$$

where the $r_{m,j}(i)$ are integers depending upon m, j, and i; k equals the number of productions in R; and $0 < p_m \leqslant 1$. But it is not possible to represent $p(0^i1^i)$ in this manner for all i. Thus $p(0^i1^i)$ cannot be generated by a proper context-free p-grammar. Ellis [4] proves the same result using a much more complex counter example.

Two languages L_1 and L_2 are equal if and only if L_1 and L_2 correspond to the same set of strings. It should be noted that even when two languages are equal the grammars that generate L_1 and L_2 may be entirely different. For example, one might be context-free and the other could be a finite-state grammar. Consequently there is a very wide range of probabilistic word functions that may be assigned to any given language.

V. Conditions for a Probabilistic Grammar to be Consistent

It is quite easy to define a p-grammar. However, there is no guarantee that the given grammar is a consistent grammar. The conditions necessary for an unrestricted proper context-free p-grammar to be a consistent grammar will now be developed.

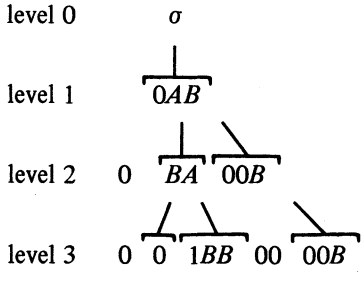
For a context-free language all the productions have the form $A_j \to \beta_{j,u}$. Even though the premise consists of a unique element the consequence may contain either zero, one, or a number of nonterminal symbols. The next theorem provides a test to determine if the resulting word function is a probabilistic word function. This discussion uses generating function techniques for representing branching processes [7] and closely parallels the development presented in [6]. To apply the theory of branching processes it is necessary to establish the idea of the level of the generation process used to generate a string $v \in L(G)$.

The zeroth level of a generation sequence will be taken as σ. The first level will be taken as γ_1 where γ_1 is the string generated by the production $\sigma \to \gamma_1$. The second level will correspond to the string γ_2 that is obtained from γ_1 by applying appropriate productions to *every* nonterminal element of γ_1. If γ_1 does not contain any nonterminal elements the process is terminated. Extending this idea, the ith level string γ_i is defined to be the string obtained from the string γ_{i-1} by applying appropriate productions to every nonterminal element of γ_{i-1}.

For example let

$$V_N = \{\sigma, A, B\} \quad V_T = \{0, 1\}$$
$$P = \{\sigma \to 0AB, A \to BA, A \to 1BB, B \to 00B, B \to 0\}$$

be given. Then a generation sequence for the string 00100000 in terms of levels would be



Since all nonterminal elements are considered simultaneously in going from level $i-1$ to level i, only the probabilities associated with the rewrite rules need be considered. The order in which these rules are applied is not important.

Definition 13–Production Generating Function: For each $A_j \in V_N$ define the k-argument generating function as

$$g_j(s_1, s_2, \cdots, s_k) = \sum_{\Gamma_{A_j}} p(\beta_{j,u}/A_j)\, s_1^{r_1(\beta_{j,u})} \cdots s_k^{r_k(\beta_{j,u})}$$

where

Γ_{A_j} Set of all productions with A_j as premise.
$r_n(\beta_{j,u})$ Number of times the variable A_n appears in the string $\beta_{j,u}$ of the production $A_j \to \beta_{j,u}$.

For example, for $V_N = \{\sigma, A\}$

$$R = \{\sigma \to 0\sigma A, \sigma \to 1, A \to 0AA, A \to 00\}$$
$$g_1(s_1, s_2) = p(0\sigma A/\sigma)\, s_1 s_2 + p(1/\sigma)$$
$$g_2(s_1, s_2) = p(0AA/A)\, s_2^2 + p(00/A).$$

As far as the statistical properties being considered are concerned, two ith level strings are equivalent if they contain the same number of nonterminal elements of each type. Thus the generating function $G_i(s_1, s_2, \cdots, s_k)$ for the ith level is defined as follows.

Definition 14: The ith level generating function $G_i(s_1, s_2, \cdots, s_k)$ is defined recursively as

$$G_0(s_1, s_2, \cdots, s_k) = s_1$$
$$G_1(s_1, s_2, \cdots, s_k) = g_1(s_1, s_2, \cdots, s_k)$$
$$G_i(s_1, s_2, \cdots, s_k) = G_{i-1}\,[g_1(s_1, s_2, \cdots, s_k), g_2(s_1, s_2, \cdots, s_k), \cdots, g_k(s_1, s_2, \cdots, s_k)].$$

For example, for the productions R defined after Definition 13

$$G_0(s_1, s_2) = s_1$$
$$G_1(s_1, s_2) = g_1(s_1, s_2) = p(0\sigma A/\sigma)\, s_1 s_2 + p(1/\sigma)$$
$$G_2(s_1, s_2) = p(0\sigma A/\sigma)[g_1(s_1, s_2)]\,[g_2(s_1, s_2)] + p(1/\sigma)$$
$$= p^2(0\sigma A/\sigma)\, p(0AA/A)\, s_1 s_2^3 + p^2(0\sigma A/\sigma)\, p(00/A)\, s_1 s_2 + p(0\sigma A/\sigma)\, p(1/\sigma)\, p(0AA/A)\, s_2^2 + p(0\sigma A/\sigma)\, p(1/\sigma)\, p(00/A) + p(1/\sigma).$$

Examining this example it is seen that $G_i(s_1, s_2, \cdots, s_k)$ can be expressed as

$$G_i(s_1, s_2, \cdots, s_k) = D_i(s_1, s_2, \cdots, s_k) + C_i$$

where the polynomial $D_i(\cdot)$ does not contain any constant term. The constant term C_i corresponds to the probability

associated with all strings $v \in L(G)$ that can be derived in i or fewer levels.

Lemma 1: An unrestricted proper context-free p-grammar is consistent if and only if

$$\lim_{i\to\infty} C_i = 1.$$

Proof: If the above limit does not equal 1 this means that there is a finite probability that the generation process can enter a generation sequence that has a finite probability of never terminating. Thus the word function defined upon $L(G)$ will always be less than 1 and will not be consistent.

On the other hand if the limit is 1 this means that no such infinite generation sequence exists since the limit represents the probability of all strings that are generated by the application of a finite number of production rules. Consequently the grammar is consistent.

Definition 15–First-Moment Matrix: The first-moment matrix E associated with the unrestricted proper p-grammar G is

$$E = [e_{ij}], \quad 1 \leqslant i,j \leqslant k$$

where

$$e_{ij} = \left.\frac{\partial g_i(s_1, s_2, \cdots, s_k)}{\partial s_j}\right|_{s_1, s_2, \cdots, s_k = 1}$$

represents the expected number of occurrences of the nonterminal symbol A_j in the set of production that have premise A_i.

The characteristic roots or eigenvalues of the first moment matrix E will be of particular importance. These roots will be indicated by $\rho_1, \rho_2, \cdots, \rho_k$ and ordered such that $|\rho_i| \geqslant |\rho_j|$ if $i < j$.

Theorem 2: A proper unrestricted p-grammar is consistent if $|\rho_1| < 1$ and it is not consistent if $|\rho_1| > 1$.

Comment: The case where one or more of the eigenvalues have a magnitude of 1 constitutes a special case. Because of the many special cases that must be considered for this situation it will be omitted. See [7] for discussion of this special case.

Proof: The coefficient of the term $s_1^{r_1} s_2^{r_2} \cdots s_k^{r_k}$ in the generating function $G_i(s_1, s_2, \cdots, s_k)$ corresponds to the probability that there will be r_1 nonterminal symbols $A_1, \cdots, r_k$ nonterminal symbols A_k in the ith level of the generation sequence. In particular, if the grammar is consistent this means that

$$\lim_{i\to\infty} G_i(s_1, s_2, \cdots, s_k) = \lim_{i\to\infty} [D_i(s_1, s_2, \cdots, s_k) + C_i] = 1.$$

This is true only if

$$\lim_{i\to\infty} D_i(s_1, s_2, \cdots, s_k) \to 0.$$

But if this limit exists then the expected value of r_u, $u = 1, \cdots, k$, must go to 0. The expected value of r_u at the ith level is

$$\bar{r}_{i,u} = \left.\frac{G_i(s_1, \cdots, s_k)}{\partial s_u}\right|_{s_1, s_2, \cdots, s_k = 1}.$$

Let

$$\bar{R}_i = [\bar{r}_{i,1}, \bar{r}_{i,2}, \cdots, \bar{r}_{i,k}];$$

then

$$\bar{R}_i = \left[\sum_{j=1}^{k} \frac{\partial G_{i-1}[g_1(s_1, \cdots, s_k), \cdots, g_k(s_1, \cdots, s_k)]}{\partial g_j} \cdot \frac{\partial g_j}{\partial s_u}(s_1, \cdots, s_k)\right]\Bigg|_{s_1, \cdots, s_k = 1}$$

$$= \bar{R}_{i-1} E.$$

Repeatedly applying this relationship gives

$$\bar{R}_i = \bar{R}_0 E^i = [1, 0, 0, \cdots, 0]\, E^i.$$

Thus

$$\lim_{i\to\infty} \bar{R}_i = 0 \text{ iff } \lim_{i\to\infty} E^i \to 0.$$

The matrix E will satisfy this condition if the magnitude of all the characteristic roots of E are less than 1. Similarly if one or more of these roots has a magnitude greater than 1 the limit diverges.

Definition 16–Strongly Consistent Grammar: An unrestricted proper p-grammar is called strongly consistent if and only if all of the eigenvalues of the E matrix have magnitudes less than 1.

As an example consider the set of productions R_1 defined after Definition 13. For this case

$$E = \begin{bmatrix} p(0\sigma A/\sigma) & p(0\sigma A/\sigma) \\ 0 & 2p(0AA/A) \end{bmatrix}.$$

The characteristic equation associated with E is

$$\phi(x) = [x - p(0\sigma A/\sigma)]\,[x - 2p(0AA/A)].$$

Thus this probabilistic representation will be consistent as long as

$$p(0\sigma A/\sigma) < 1 \text{ and } p(0AA/A) < 1/2.$$

Linear grammars (including regular grammars) are a special important case of context-free grammars. For a linear grammar all productions can be assumed to have one of the following forms

$$A \to uBv \quad A \to u$$

where

$$A, B \in V_N \text{ and } u, v \in V_T^*, \quad uv \neq \lambda.$$

(Regular grammars have productions of the form $A \to uB$, $A \to u$, $u \in V_T$.) These grammars are particularly easy to investigate since no intermediate string will contain more than one element from V_N.

A graph can be associated with every linear grammar. The nodes of the graph correspond to the elements of V_N and a special node n_T. Let $N = V_N \vee n_T$ denote the set of $k + 1$ nodes. There is a path between nodes A_i and A_j for each production of the form $A_i \to uA_jv$ and there is a path between nodes A_i and n_T for each production of the form

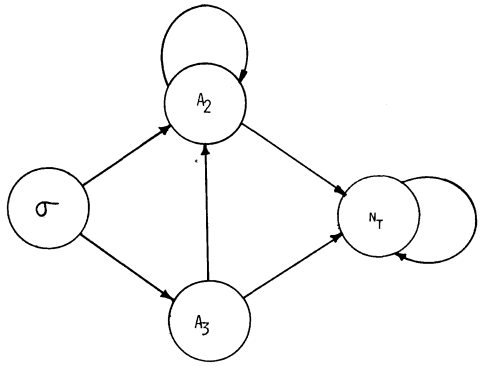


Fig. 1. Graph associated with G.

$A_i \to u$. The node corresponding to $\sigma = A_1$ is the initial node. For completeness it is assumed that there is a single self-loop associated with node n_T.

For every $v \in L(G)$ there must be at least one path through the graph from the initial node to the final node corresponding to the derivation $\sigma \Rightarrow v$. Thus if G is consistent the probability of reaching the terminal node from the initial node must be 1. Otherwise G would not be consistent.

The graph represents a $k + 1$ state Markov process where the state-transition probabilities are the probabilities associated with the production represented by each edge of the graph. The transition probability of the loop associated with n_T is 1.

The transition matrix M associated with this graph has the form

$$M = \begin{bmatrix} E & W \\ 0 & 1 \end{bmatrix}$$

where E is the $k \times k$ first-moment matrix associated with the grammar, W is a k-row column vector, the ith row of which corresponds to the probability that A_i is rewritten by a terminal symbol, and 0 is the k-column row null vector. This matrix defines a $(k + 1)$ state process where the first k states correspond to the elements of V_N and the $(k + 1)$st state corresponds to the terminal node. In particular, consider the $1, k + 1$ element, $m_{1,k+1}(n)$, of M^n. This term corresponds to the probability of the set $L(G, n)$ consisting of all strings of $L(G)$ that require the application of n or fewer productions in their derivation. This leads to the following lemma.

Lemma 2: Let G be a proper unrestricted linear p-grammar. G is a consistent probabilistic representation if and only if

$$\lim_{n\to\infty} m_{1,k+1}(n) = 1.$$

This lemma is important for two reasons. First, it shows that G is consistent if and only if n_T is the only absorbing state. Secondly, the properties of M^n can be used to establish a limit to the maximum length of strings from $L(G)$ that must be included in a finite-set approximation of $L(G)$. This application will be considered in detail in the next section.

Example: Let $G = \langle V_N, V_T, R, \sigma \rangle$ be defined by

$$V_N = \{\sigma, A_2, A_3\}, \quad V_T = \{0, 1\},$$

$$R = \{\sigma \to 0A_2, \sigma \to 1A_3, A_2 \to 0A_2, A_2 \to 0,$$

$$A_3 \to A_2 1, A_3 \to 1\}$$

$$P = \{p(A_2/\sigma) = 0.4, p(A_3/\sigma) = 0.6, p(A_2/A_2) = 0.5,$$

$$p(n_t/A_2) = 0.5, p(A_2/A_3) = 0.6,$$

$$p(n_t/A_3) = 0.4.$$

The graph associated with G is given in Fig. 1.

The matrix M becomes

$$M = \begin{bmatrix} 0 & p(A_2/\sigma) & p(A_3/\sigma) & 0 \\ 0 & p(A_2/A_2) & 0 & p(n_t/A_2) \\ 0 & p(A_2/A_3) & 0 & p(n_t/A_3) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0.4 & 0.6 & 0 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 1 \end{bmatrix};$$

then

$$\lim_{n\to\infty} M^n = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Thus $\lim m_{1,4}(n) = 1$ and G is consistent.

VI. Properties of Languages Derived from Consistent Grammars

Any information processing device that is designed to carry out some type of processing task on a language will require a storage space or computation time that is dependent upon the length of the word being processed. A language with an infinite number of terms contains words whose length exceeds any prespecified bound. Thus, if the system is designed to process all words in a language, provisions must be made to process words of any length. This approach completely neglects the fact that some words are much more likely to occur than other words. In this section the probabilistic properties of languages generated by strongly consistent p-grammars are used to study the properties of those words that have a non-negligible probability of occurring.

Let $l(v)$ denote the length of $v \in L$.

Definition 17–Average Word Length (AWL): The AWL of a language L is

$$\text{AWL} = \sum_{\forall v \in L} l(v)\, p(v)$$

where $p(v)$ is the probability measure associated with L.

For a p-grammar the probabilities associated with the productions are

$$p_{j,u} : A_j \rightarrow \beta_{j,u}, \quad u = 1, 2, \cdots, m_j$$

where m_j represents the number of productions that have A_j as a premise. Let

$t(\beta_{j,u})$	Number of terminal symbols in $\beta_{j,u}$.
$t_j = \sum_{u=1}^{m_j} p_{j,u} t(\beta_{j,u})$	Average number of terminal symbols generated when A_j is rewritten.
$T = [t_j]$	Column vector.

(It is noted that if G is in Greibach normal form then T is a column vector of ones.)

n_j	Average length of all words generated by G with A_j as the initial symbol (n_1 = AWL of a language if it exists).
$h_i(\beta_{j,u})$	Number of occurrences of A_i in the consequence $\beta_{j,u}$.
$E = [e_{ij}]$	First-moment matrix.
V_N	$= \{\sigma = A_1, A_2, \cdots, A_k\}$.

Theorem 3: Let $L(G)$ be generated by a strongly consistent context-free p-grammar G. Then the AWL of $L(G)$ is

$$\text{AWL} = [100\cdots0](I - E)^{-1}T.$$

Proof: For any j

$$n_j = \sum_{u=1}^{m_j} p_{j,u}\left[t(\beta_{j,u}) + \sum_{i=1}^{k} n_i h_i(\beta_{j,u})\right]$$

$$= \sum_{i=1}^{k}\left[\sum_{u=1}^{m_j} p_{j,u} h_i(\beta_{j,u})\right] n_i + \sum_{u=1}^{m_j} p_{j,u}\, t(\beta_{j,u}).$$

The term $[\sum_{u=1}^{m_j} p_{j,u} h_i(\beta_{j,u})]$ is the (j, i) term of the expectation matrix E. Thus, if N denotes the column vector $[n_i]$, the above expression gives the general matrix relationship

$$N = EN + T.$$

Solving for N gives

$$N = (I - E)^{-1}T$$

where $(I - E)^{-1}$ exists if E has no eigenvalues with unit magnitude. This condition is guaranteed if G is a strongly consistent context-free grammar. Premultiplying N by $[1\ 0\ 0\cdots0]$ removes all terms except n_1 which is the desired AWL.

Alternate Proof: Let $e_{i,j}(n)$ be the (i, j)th term of E^n. The term $e_{i,j}(n)$ corresponds to the average number of occurrences of A_j at the nth level of the generation sequence given that A_i is the initial nonterminal symbol at the zeroth level. Let $z_{i,j}$ indicate the average number of occurrences of A_j in all levels of a derivation beginning with A_i. Then

$$z_{i,j} = \sum_{n=0}^{\infty} e_{ij}(n)$$

where

$$e_{ij}(0) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j. \end{cases}$$

Extending this matrix form gives

$$Z = \sum_{n=0}^{\infty} E^n = I + E + E^2 + E^3 + \cdots$$

since $E^0 = I$. This matrix series converges to $(I - E)^{-1}$ if all of the eigenvalues of E have a magnitude less than 1. Thus

$$Z = (I - E)^{-1}.$$

Premultiplication by the row vector $[1\ 0\ 0\cdots0]$ yields a row vector whose ith term represents the average number of occurrences of A_i in a derivation starting with A_1. Multiplication of this row vector by the T matrix accounts for the average number of terminal symbols generated directly by each A_i. This then gives

$$\text{AWL} = [100\cdots0]\, ZT = [100\cdots0]\,[I - E]^{-1}T.$$

Example: Let $G = \langle V_T, V_N, R, P, \sigma\rangle$ be defined as

$$V_T = \{a, b, c\} \qquad V_N = \langle \sigma, B\rangle$$

$$R, P$$

$$0.5 : \sigma \rightarrow a \qquad 0.8 : B \rightarrow cb$$
$$0.5 : \sigma \rightarrow \sigma B \qquad 0.2 : B \rightarrow b\sigma.$$

The E matrix is

$$E = \begin{bmatrix} 0.5 & 0.5 \\ 0.2 & 0.0 \end{bmatrix}.$$

The eigenvalues are $\rho_1 = 0.65$, $\rho_2 = -0.15$. Thus G is strongly consistent. The T matrix is

$$T = \begin{bmatrix} 0.5 \\ 1.8 \end{bmatrix}.$$

Thus

$$\text{AWL} = [1\ \ 0]\begin{bmatrix} 2.5 & 1.25 \\ 0.5 & 1.25 \end{bmatrix}\begin{bmatrix} 0.5 \\ 1.8 \end{bmatrix} = 3.5 \text{ characters/word.}$$

In addition to the AWL associated with a language it is often desirable to know the character density of the language.

Definition 18–Character Density $d\{V_T\}$: The character density $d\{V_T\}$ of a language L is the relative number of times each character from V_T appears in the words of L.

Let $s_i(\beta_{j,u})$ denote the number of occurrences of $a_i \in V_T$ in the consequence of the production $A_j \rightarrow \beta_{j,u}$. Then let S_i be the column vector whose jth term is

$$\sum_{u=1}^{m_j} p_{j,u}\, s_i(\beta_{j,u}).$$

This term is interpreted as the average number of a_i generated directly from A_j.

Theorem 4: If $L(G)$ is produced by a strongly consistent context-free p-grammar G then

$$d(a_i) = [1\ 0\ 0\cdots0]\,[I - E]^{-1}S_i/\text{AWL}.$$

Proof: From Theorem 3

$$[1\ 0\ 0 \cdots 0]\,[I - E]^{-1} S_i$$

is seen to give the average number of times the symbol a_i appears in a word of L. Division by AWL normalizes this value since AWL is the average number of terminal symbols in a word in L.

Example: If G is the grammar given after Theorem 3 then

$$S_1 = \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} \quad S_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad S_3 = \begin{bmatrix} 0 \\ 0.8 \end{bmatrix}.$$

Note $S_1 + S_2 + S_3 = T$. Applying the above formula gives

$$d(a) = \frac{1.25}{3.5} = 0.375 \quad d(b) = \frac{1.25}{3.5} = 0.375 \quad d(c) = \frac{1.0}{3.5} = 0.286.$$

The above results indicate that the words of $L(G)$ decrease in probability as the length of the words increase. Thus in designing a processor that operates on the words from $L(G)$ it is often a realistic approach to design the processor so that it satisfactorily handles those sequences that have a high probability of occurring and may or may not process the other sequences correctly.

Assume that B is a subset of $L(G)$.

Definition 19–ϵ-Representation: The set B is an ϵ-representation of the language $L(G)$ if

$$B \subseteq L(G)$$

$$\sum_{v \in B} p(v) \geqslant 1 - \epsilon.$$

Definition 20–N_ϵ Bound: A language $L(G)$ has an N_ϵ bound if there is a B such that B is an ϵ-representation of $L(G)$ and $l(v) \leqslant N_\epsilon$ for all $v \in B$.

Theorem 5: Every strongly consistent context-free language $L(G)$ has a finite N_ϵ bound.

Proof: From the proof of Theorem 2, it is known that

$$\lim_{i \to \infty} \overline{R}_i = \lim_{i \to \infty} [1, 0, 0, \cdots, 0]\, E^i = 0$$

if $L(G)$ is strongly consistent. Now $1 - C_i$ is the probability that the generation process has not terminated by the ith level. But

$$1 - C_i \leqslant \overline{R}_i \begin{bmatrix} 1 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix} = [1, 0, 0, \cdots, 0]\, E^i \begin{bmatrix} 1 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix}.$$

However, if ρ_1 is the largest characteristic root of E, then E^i tends to zero at a rate determined by $|\rho_1|^i$. Thus there exists an $I_{\max}$ such that

$$1 - C_{I_{\max}} \leqslant \epsilon$$

or

$$C_{I_{\max}} \geqslant 1 - \epsilon.$$

Let B be the set consisting of all words requiring $I_{\max}$ or fewer levels to generate. Since only a finite number of terminal symbols are introduced at each level of a derivation and the above results show that no more than $I_{\max}$ levels are needed to generate any word in B it is concluded that an N_ϵ bound exists for the language $L(G)$.

The calculation of N_ϵ for a general strongly consistent context-free p-grammar is complicated by the fact that the number of nonterminal symbols present at any level is a random variable. However for linear grammars with the restriction on rewrite rules introduced for Lemma 2, it is possible to easily compute N_ϵ.

Lemma 3: For a consistent linear grammar the value of N_ϵ is bounded by

$$N_\epsilon \leqslant I_{\max} \cdot n_{\max}$$

where $n_{\max}$ is the maximum number of terminal symbols found in the consequence of any rewrite rule in the grammar.

Proof: Let M be the matrix defined in the proof of Lemma 2. The $m_{1,k+1}(i)$ element of M^i is the probability of all words that require the application of i or fewer rewrite rules in their production. A value $i = I_{\max}$ can be found so that

$$m_{1,k+1}(I_{\max}) \geqslant 1 - \epsilon.$$

Since a maximum of $n_{\max}$ terminal symbols can be added to a word by the application of a rewrite rule this means that no word derived in $I_{\max}$ or fewer levels contains more than $I_{\max} \cdot n_{\max}$ terminal symbols. Thus

$$N_\epsilon \leqslant I_{\max} \cdot n_{\max}.$$

It is noted that if $n_{\max} = 1$, then $N_\epsilon = I_{\max}$.

Lemma 4: For a strongly consistent context-free p-grammar in Chomsky normal form

$$N_\epsilon \leqslant 2^{(I_{\max} - 1)}.$$

Proof: From the proof of the *uvwxy* theorem [8] for context-free languages it is known that if a derivation requires j levels then the terminal word derived is of length no greater than 2^{j-1}. Letting $j = I_{\max}$ complete the proof.

VII. Conclusions

This paper has considered the problem of assigning a probabilistic measure to the words of a language that indicates the relative importance of the different words of the language. In particular, this measure deals with the probabilities of occurrence of each word in a language rather than with the probability of the word being accepted by a stochastic automaton. Consequently, the results obtained provide a way of working with random processes that occur as inputs to nonprobabilistic machines.

References

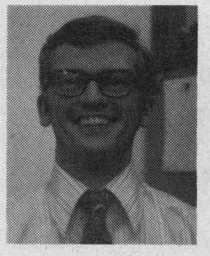
[1] A. T. Bharucha-Reid, *Elements of the Theory of Markov Processes and Their Applications.* New York: McGraw-Hill, 1960.
[2] T. L. Booth, *Sequential Machines and Automata Theory.* New York: Wiley, 1967.
[3] —, "Probabilistic representation of formal languages," in *IEEE Conf. Rec. 10th Symp. Switching and Automata Theory*, 1969.

[4] C. A. Ellis, "Probabilistic languages and automata," Ph.D. dissertation, Univ. Illinois, Urbana, 1969.
[5] S. Ginsburg, *The Mathematical Theory of Context-Free Languages*. New York: McGraw-Hill, 1966.
[6] U. Grenander, "Syntax-controlled probabilities," Div. Appl. Math., Brown Univ., Providence, R.I., Internal Rep., 1967.
[7] T. E. Harris, *The Theory of Branching Processes*. Berlin: Springer, 1963.
[8] J. E. Hopcroft and J. D. Ullman, *Formal Languages and Their Relation to Automata*. Reading, Mass.: Addison-Wesley, 1969.
[9] A. Paz, *Introduction to Probabilistic Automata*. New York: Academic, 1971.
[10] A. Salomma, "Probabilistic and weighted grammars," *Inform. Contr.*, vol. 15, pp. 529–544, 1969.
[11] R. A. Thompson, "Compact encoding of probabilistic languages," Ph.D. dissertation, Comput. Sci. Group, Dep. Elec. Eng., Univ. Connecticut, Storrs, 1971.

Taylor L. Booth (S'53-M'57-SM'70) was born in Middletown, Conn., on September 22, 1933. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Connecticut, Storrs, in 1955, 1956, and 1962, respectively.

From 1956 to 1959 he was an Analytical Engineer for the Air Arm Division of Westinghouse Electric Corporation, Baltimore, Md. In 1959 he joined the faculty of Electrical Engineering at the University of Connecticut where he is currently a Professor of Electrical Engineering and Chairman of the Department's Computer Science Group.

Dr. Booth was a member of the COSINE Committee of the National Academy of Engineering and is currently Associate Editor for Switching and Automata Theory of this TRANSACTIONS.



Richard A. Thompson (S'63-M'66) was born in Jersey City, N.J., on April 6, 1942. He received the B.S. degree in electrical engineering from Lafayette College, Easton, Pa., in 1964, the M.S. degree in electrical engineering from Columbia University, New York, N.Y., in 1966, and the Ph.D. degree in computer science from the University of Connecticut, Storrs, in 1971.

From 1964 to 1968 he was a member of the Technical Staff at Bell Telephone Laboratories, Inc., Holmdel, N.J., and from 1968 to 1969 he worked for the Royal Typewriter Division of Litton Industries, Inc., Hartford, Conn. In 1969 he joined the University of Connecticut holding the ranks of Graduate Assistant, Research Assistant, and Instructor. Since 1971 he has been an Assistant Professor in the Department of Electrical Engineering at the Virginia Polytechnic Institute and State University, Blacksburg.

Dr. Thompson is a member of Eta Kappa Nu, Tau Beta Pi, Phi Beta Kappa, Sigma Xi, the American Society for Engineering Education, and the Association for Computing Machinery.

On Symmetric Functions with Redundant Variables–Weighted Functions

BJØRN DAHLBERG

Abstract–**Any switching function may be transformed into a completely symmetric switching function with some of its variables repetitive. In this paper we shall discuss certain properties, easily detectable in decomposition charts, that may be applied to reduce the number of repeated variables when such a transformation is performed. An algorithm utilizing lookup tables based on these properties has been devised enabling us with relative ease to transform an arbitrary switching function of four variables into a completely symmetric switching function with the least number of variables. The algorithm may be generalized to *n*-variable switching functions provided that the corresponding lookup tables are made available.**

To eliminate the tediousness associated with repeating the variables, we introduce a novel class of functions–weighted functions–that also emphasize the similarity between completely symmetric functions and multithreshold threshold functions.

Index Terms–**Decomposition charts, lookup tables, multithreshold threshold logic, symmetric switching functions, symmetric switching functions with redundant variables, threshold logic.**

Manuscript received February 29, 1972; revised October 3, 1972.

The author is with the Systems Engineering Laboratories, Inc., Ft. Lauderdale, Fla., and is a graduate student at the University of Florida, Gainesville, Fla.

I. Introduction and Background

RELATIVELY few switching functions can be expressed as completely symmetric switching functions with no redundant variables. On the other hand, Kautz [1][1] has shown that any switching function can be transformed into a completely symmetric switching function of $2^n - 1$ variables with some of the variables repetitive. More elaborate formulas have been devised later, however, by which such a transformation can be executed with fewer than $2^n - 1$ variables if the function is partially symmetric [2]–[5]. To illustrate past progress in the theory of symmetric switching functions with redundant variables, and to aid the subsequent discussion, consider the following example.

Example 1: Consider the switching function

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = \sum (1\text{-}6, 9\text{-}14, 17\text{-}22, 24\text{-}26, 28, 33\text{-}38, 40\text{-}42, 44, 48\text{-}50, 52, 56\text{-}58, 60). \quad (1)$$

[1] Due to D. A. Huffman of M.I.T.