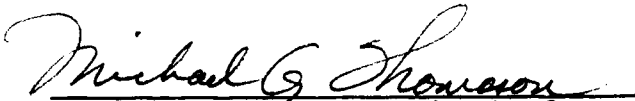
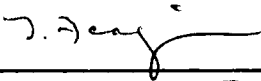
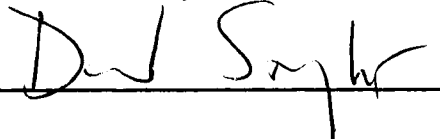


To the Graduate Council:

I am submitting herewith a thesis written by Richard H. Sady entitled "Probabilistic Grammars and Languages." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.


Michael G. Thomason, Major Professor

We have read this thesis
and recommend its acceptance:

Accepted for the Council:


Vice Chancellor
Graduate Studies and Research

PROBABILISTIC GRAMMARS AND LANGUAGES

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Richard H. Sady

December 1981

3055494

To
Carol

ABSTRACT

This work is a study of three areas in the theory of probabilistic grammars and languages. First, the relationship of branching processes to probabilistic context-free grammars is investigated. Second, a method is given for finding probabilities of certain strings in context-free languages given "regular-like" expressions representing the languages. Finally, it is shown that there is no natural way of extending Parikh's Theorem to a probabilistic version.

TABLE OF CONTENTS

SECTION	PAGE
I. INTRODUCTION.	1
II. PROBABILISTIC LANGUAGES AND GRAMMARS.	3
Basic Definitions and Notation.	3
Summary of Known Results.	7
Alternative Definitions	9
III. BRANCHING PROCESSES: APPLICATIONS TO PROBABILISTIC GRAMMARS.	11
Introduction to Multitype Branching Processes	11
Formal Definitions and Main Results in the Theory of Multitype Galton-Watson Branching Processes	13
Applications to Probabilistic Context-Free Grammars	17
Useful Results about the Spectral Radius.	20
Finding the Expected Derivation Length and Expected Word Length	21
Applications to Infinite Series	24
IV. FINDING PROBABILITIES	26
The Generating Function Approach for Regular Expressions.	26
A Generalization to Context-Free Languages.	27
Determining Production Probabilities.	31
V. PARIKH'S THEOREM--A PROBABILISTIC VERSION?.	33
Introduction.	33
Parikh's Approach	35
Pilling's Methods	38
Van Leeuwen's Extension	42
Some Other Ideas.	47
Conclusions	48
LIST OF REFERENCES.	49
VITA.	53

I. INTRODUCTION

One method of specifying formal languages is by means of grammars. In the Chomsky hierarchy restrictions are placed on the form of the productions of a grammar. One can also change the manner in which a grammar is allowed to generate words. That is, additional restrictions can be placed on the use of the productions. The study of probabilistic grammars and languages is one line of research that incorporates these ideas. We investigate here several topics in this area.

In section II we give the basic definitions and main results in the theory. The relationship of branching processes to probabilistic context-free grammars is investigated in section III. A possible application to infinite series is also discussed.

In a recent work Parker and McClusky (1978) show how to find probabilities of certain strings in a regular language directly from regular expressions and the probabilities of the individual symbols. Their method uses generating functions. We show how to extend their results to context-free languages. This problem is approached in two ways. Using Gruska's characterization of context-free languages in terms of "context-free expressions" (Gruska, 1971), the generating function method can be used. This is done in section IV.

In section V we attempt to find a probabilistic version of Parikh's Theorem. See, for example, Ginsburg (1966, p. 146) or Salomaa (1973, p. 63). If a probabilistic version were possible then, given a context-free language, one could find a regular language with the same symbols and the "same" probabilities. Then one could directly

apply the methods of Parker and McClusky. It is shown that there is no natural probabilistic extension of Parikh's Theorem.

II. PROBABILISTIC LANGUAGES AND GRAMMARS

1. Basic Definitions and Notation

We present here some fundamental concepts and notational conventions used throughout this work.

1.1 Definition. A grammar, G , is (V_N, V_T, P, S) , where

V_N = a finite set of "non-terminals,"

V_T = a finite set of "terminals,"

$S \in V_N$ is a distinguished symbol, the "start symbol," and

$P \subseteq V^*V_NV^* \times V^*$ are the "productions," where

$V = V_N \cup V_T$ = the "vocabulary," $V_N \cap V_T = \emptyset$, and

V^* = finite length sequences of elements from V .

$(\alpha, \beta) \in P$ is usually written $\alpha \rightarrow \beta$; α is the "premise," β is the "consequence." A language is any $L \subseteq V^*$. A context-free grammar (CFG) has $P \subseteq V_N \times V^*$. A string is any element of V^* . For two strings α, β we write $\alpha \Rightarrow \beta$ if $\alpha = \gamma A \delta$ and $\beta = \gamma \omega \delta$ and $A \rightarrow \omega$ is in P . Note that " $\Rightarrow$ " is a relation on V^* . Let $\stackrel{*}{\Rightarrow}$ denote the reflexive, transitive closure of $\Rightarrow$. We say " α derives β " if $\alpha \stackrel{*}{\Rightarrow} \beta$. A derivation of β from α is a sequence $\alpha = \alpha_0, \alpha_1, \dots, \alpha_n = \beta$ such that each $\alpha_i \in V^*$ and $\alpha_{i-1} \Rightarrow \alpha_i$ for all $i, 1 \leq i \leq n$. If G is a grammar, define the language generated by G to be $L(G) = \{x \mid S \stackrel{*}{\Rightarrow} x \text{ and } x \in V_T^*\}$. Let R denote the set of real numbers.

1.2 Definition. A probabilistic language over V (p-language) is (L, Φ) , where $L \subseteq V^*$ and $\Phi : V^* \rightarrow R$ is a function that satisfies:

a) if $x \in V^* - L$, then $\Phi(x) = 0$,

b) if $x \in L$, then $0 \leq \phi(x) \leq 1$,

c) $\sum_{x \in L} \phi(x) = 1$.

Note: ϕ will be called a "probabilistic word function."

1.3 Example. $V = \{a, b\}$, $L = \{a^i b^i \mid i \geq 0\}$, and $\phi(a^i b^i) = \frac{t^i}{e^{t_i} i!}$ for any fixed $t \in R$. Define $\phi(x) = 0$ if $x \in V^* - L$.

Intuitively, a probabilistic grammar is a grammar with probabilities "attached" to its productions. More formally, we have:

1.4 Definition. A probabilistic grammar (p-grammar) is (G, F) , where G is a grammar (V_N, V_T, P, S) and F is a function $F : P \rightarrow [0, 1]$.

For any production $\alpha \rightarrow \beta$ in P , we write $F(\alpha, \beta) : \alpha \rightarrow \beta$. $F(\alpha, \beta)$ is called the "probability" of the production $\alpha \rightarrow \beta$. Often the p-grammar (G, F) is denoted by G if F is understood from the context.

We assume that all p-grammars are "unrestricted": the probability of the application of any production depends only upon a given premise, not upon how the premise was generated. Intuitively, the probabilities weight the possible choices of a production. To insure that some choice must be made, we need the following concept.

1.5 Definition. A p-grammar is proper if for any premise α_i ,

$$\sum_{j=1}^{n_i} p_{ij} = 1, \text{ where}$$

n_i = number of productions of the form $\alpha_i \xrightarrow{p_{ij}} \beta_j$.

Unless otherwise stated, we assume that all p-grammars are proper.

Given a p-grammar, G , we would like to know if and under what conditions $L(G)$ becomes a p-language. First, we must be able to define a

probabilistic word function Φ on $L(G)$. There is a natural way to assign a probability to a derivation:

1.6 Definition. In a p-grammar the probability of a derivation is the product of the probabilities of the productions used in the derivation.

1.7 Example. $G = (\{S\}, \{a\}, P, S)$ where P is

$$p : S \rightarrow SS$$

$$q : S \rightarrow a, \text{ and } p + q = 1.$$

Then the derivation:

$S \Rightarrow SS \Rightarrow Sa \Rightarrow aa$ has the probability pq^2 . Because of possible ambiguity, one cannot define the probability of $x \in L(G)$ to be the probability of its derivation. All the "different" derivations of x must be considered. The following definition contains this idea.

1.8 Definition. Let G be a p-grammar and $x \in V_T^*$. The probability of x , denoted $P[x]$, is defined by:

$$P[x] = \sum_j p_j, \text{ where}$$

$$p_j = \text{the probability of the } j^{\text{th}} \text{ "different" derivation of } x.$$

We can choose any fixed method of distinguishing derivations. Here we consider only "right canonical" derivations--those in which the rightmost nonterminal is re-written at each step. A p-grammar, G , induces a function Φ on $L(G)$. Namely, for $x \in L(G)$, $\Phi(x) = P[x]$ as defined above. More precisely,

$$\Phi(x) = \sum_i \prod_j p_{ij}(x), \text{ where}$$

$$p_{ij}(x) = \text{the probability of the production used at the } j^{\text{th}} \text{ step of the } i^{\text{th}} \text{ derivation of } x.$$

But there is no reason to assume that $\sum_{x \in L(G)} \phi(x) = 1$. That is, ϕ might not be a probabilistic word function and so $(L(G), \phi)$ might not be a p-language. It is not true that every p-grammar generates a p-language.

1.9 Definition. A p-grammar, G , is consistent if $\sum_{x \in L(G)} \phi(x) = 1$. Thus a p-grammar is consistent iff it generates a p-language. Consistency is discussed in more detail in section III 3.

1.10 Examples.

1. $V_T = \{a, b\}$, $V_N = \{S\}$, P is

$$p : S \rightarrow aS$$

$$q : S \rightarrow b, \text{ where } p + q = 1.$$

$$G = (V_N, V_T, P, S)$$

$$L(G) = \{a^n b \mid n \geq 0\}$$

$$\phi(a^n b) = p^n q.$$

So $L(G)$ is a p-language iff $\sum_{x \in L(G)} \phi(x) = 1$, that is if $\sum_{n=0}^{\infty} p^n q = 1$. But

$$\begin{aligned} \sum_{n=0}^{\infty} p^n q &= q \sum_{n=0}^{\infty} p^n = q \cdot \frac{1}{1-p} \quad \text{if } p < 1 \\ &= q \cdot \frac{1}{q} = 1. \end{aligned}$$

2. $V_T = \{a, b\}$, $V_N = \{S, A\}$, P is

$$1 : S \rightarrow Ab$$

$$p : A \rightarrow AA$$

$$q : A \rightarrow a, \text{ } p + q = 1$$

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

In the next section (III 3), we will see that $\phi(a^{n+1} b) = \binom{2n}{n} \frac{1}{n+1} p^n q^{n+1}$ and that $(L(G), \phi)$ is a p-language if and only if $p \leq \frac{1}{2}$.

2. Summary of Known Results

Ignoring the probabilities, one can define the Chomsky hierarchy for p-grammars (G, F) . We say that (G, F) is of type i iff G is of type i , $i = 0, 1, 2, 3$. The usual results about "normal" forms, recognizers, etc. . . are valid. Some of these are given below.

2.1 Theorem. Any p - CFG is equivalent to a p - CFG in Chomsky normal form.

2.2 Note. Two p-grammars (G_1, F_1) and (G_2, F_2) are equivalent if $L(G_1) = L(G_2)$ and $\Phi_1(L(G_1)) = \Phi_2(L(G_2))$, where Φ_i denotes the induced probabilistic word function on $L(G_i)$, $i = 1, 2$.

The proof of Theorem 2.1 is straightforward. For example, in the non-probabilistic version, new non-terminals are introduced and a production of the form

$$A \rightarrow B_1 B_2 \dots B_k, B_i \in V_N$$

is broken down into $A \rightarrow B_1 C_1$

$$C_1 \rightarrow B_2 C_2$$

$$\vdots$$

$$\vdots$$

$$\vdots$$

$$C_{k-3} \rightarrow B_{k-2} C_{k-2}$$

$$C_{k-2} \rightarrow B_{k-1} C_k.$$

In the probabilistic version, the production

$$p : A \rightarrow B_1 B_2 \dots B_k$$

can be broken down into $p : A \rightarrow B_1 C_1$

$$1 : C_1 \rightarrow B_2 C_2$$

.

.

.

$$1 : C_{k-3} \rightarrow B_{k-2} C_{k-2}$$

$$1 : C_{k-2} \rightarrow B_{k-1} B_k .$$

Obviously, this preserves the probabilities of words. In these proofs one can also use the "equivalence transformations" of Thompson (1974, pp. 608-610). He gives several transformations on p-grammars which preserve the language generated by the grammar. We list here some other results.

2.3 Theorem. Any p - CFG is equivalent to a p - CFG in Greibach normal form.

Proof: See Hutchins (1972).

The concepts of stochastic finite automata and stochastic pushdown automata can be defined in a natural way. See, for example, Fu (1974, p. 134 ff) or Gonzalez and Thomason (1978, p. 190 ff) or Paz (1971).

2.4 Theorem. A p-language is generated by a p-regular grammar iff it is recognized by a stochastic finite automaton.

2.5 Theorem. A p-language is generated by a p - CFG iff it is recognized by a stochastic pushdown automaton.

3. Alternative Definitions

Our definitions of p-grammars and p-languages are due to Booth and Thompson (1973). Although these definitions seem very natural and useful, there have been other attempts to formulate the concept of p-grammar and to define the languages generated by p-grammars. We discuss some of these in this section.

The classical definition of a probabilistic language ("stochastic language") is due to Rabin (1963). It is a language acceptable by a finite probabilistic automaton with a given "cutpoint," λ , where λ is a real number. That is, if $p(x)$ denotes the probability that the word x was generated, then x is accepted iff $p(x) > \lambda$. Further details and results can be found in Fu (1974) and Santos (1973).

Probabilistic extensions of context-free grammars have been studied by Salomaa (1969). He suggests attaching to each production, f , a stochastic vector whose i^{th} component indicates the probability that the i^{th} production is applied after f . One can then define the probability of a derivation in the obvious way (section II 1.6). For each word x , the probability of x is found by summing up the probabilities of the "different" derivations of x . Salomaa offers three definitions for the language generated by a p-grammar, G : the set of pairs $(x, p(x))$ where $x \in L(G)$, the set of words $x \in L(G)$ such that $p(x) > \lambda$, the set of $x \in L(G)$ such that there exists at least one derivation of x with probability greater than λ . Here $0 \leq \lambda < 1$. These ideas are studied extensively in Salomaa (1969).

A complicated mathematical formulation of p-grammars has been considered by Santos (1972). His definitions are so involved that it is difficult to give even a simple example.

Knast (1972) generalizes the classical definition by replacing the cut point λ with a function. With this interpretation he shows that any probabilistic context-free language is probabilistic regular. His definitions are perhaps too general.

Peleg (1979) uses Salomaa's definition of p-grammars (see above) but defines the probability of a word, x , as the maximum probability over all derivations of x . He presents an algorithm for finding such "maximal derivations" in the case of probabilistic linear grammars.

III. BRANCHING PROCESSES: APPLICATIONS TO PROBABILISTIC GRAMMARS

1. Introduction to Multitype Branching Processes

Suppose we have a collection of "objects" from a "population" that can generate additional objects according to some stochastic laws. Assume that the objects consist of a finite number of different "types." An initial object (called generation 0) produces a random number of "offspring" of the various types (generation 1). Each of these in turn produce a random number of offspring of the various types (generation 2). This process continues as long as there are "live" objects in the population. This is a description of what is called a "multitype branching process." If all the objects contribute offspring to the population independently, then the above process is called a "multitype Galton-Watson branching process."

The generation of nonterminals in a p - CFG is an example of a multitype Galton-Watson process. Here the "population" consists of the finite set of non-terminals. The "initial object" is the start symbol. The nonterminals generate additional nonterminals through the probabilistic productions.

1.1 Example. $G = (V_N, V_T, P, S)$, where $V_N = \{S, A\}$, $V_T = \{a, b\}$, and P is

r : $S \rightarrow aSbA$

s : $S \rightarrow aAA$

t : $S \rightarrow b$

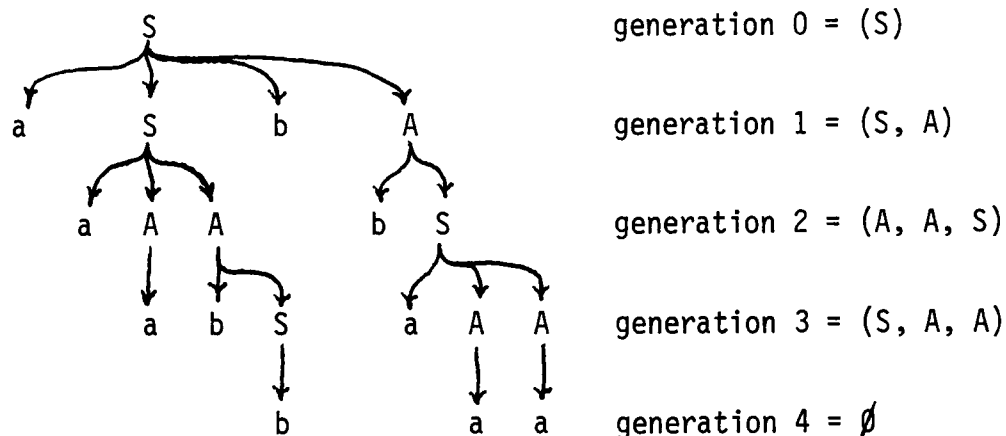
u : $A \rightarrow bS$

v : $A \rightarrow a$

where $r + s + t = 1$

and $u + v = 1$.

For example, a possible generation sequence for $a^3b^4a^3$ is:



In this example the population became "extinct" after four generations.

Some questions about p - CFG's can be phrased in terms of branching processes. For example, a p - CFG generates a p-language if the population eventually becomes "extinct." One can apply known results from the well-developed theory of branching processes to p - CFG's. However, this approach is weak since it makes no reference to the order of the terminals (and nonterminals) in a word. In the remainder of this section we shall examine some of the main results from the theory of multitype branching process and discuss how they apply to p - CFG's. For a more extensive treatment of branching processes see Athreya and Ney (1972).

2. Formal Definitions and Main Results in the Theory of Multitype Galton-Watson Branching Processes

2.1 Notation. We will use the following notation.

R = the set of real numbers

R^n = n -dimensional Euclidean space, R_+^n = the set of points in R^n with non-negative co-ordinates

I^n = the set of points in R^n with integer co-ordinates

I_+^n = the set of points in R_+^n with integer co-ordinates

If $j \in I_+^n$, say $j = (j_1, \dots, j_n)$, let $p^i(j)$ = the probability that a type i object produces j_k offspring of type k , $1 \leq k \leq n$. Let $p(j) = (p^1(j), p^2(j), \dots, p^n(j))$. A generating function $f(s)$ is defined as follows:

if $s = (s_1, s_2, \dots, s_n)$, where $0 \leq s_j \leq 1$ for $1 \leq j \leq n$, then

$$f^i(s) = \sum_{j \in I_+^n} p^i(j) s_1^{j_1} s_2^{j_2} \dots s_n^{j_n}.$$

Let $f(s) = (f^1(s), f^2(s), \dots, f^n(s))$.

We write $f(s) = \sum_j p(j) s^j$.

Whenever dealing with generating functions we assume that $0 \leq s_j \leq 1$ for $1 \leq j \leq n$.

2.2 Definition. A multitype Galton-Watson branching process (hereafter referred to as a "process") is a Markov chain $\{Z_n \mid n = 0, 1, 2, \dots\}$ on R_+^n with transition function $P(i, j) = P[Z_{n+1} = j \mid Z_n = i]$, $i, j \in R_+^n$,
 $\quad \quad \quad = \text{coefficient of } s^j \text{ in } [f(s)]^i$.

Let Z_{ki}^j = number of objects of type i in the k^{th} generation, assuming that $Z_0 = j$ for some $j \in I_+^n$. Here $Z_0 = (Z_{01}, Z_{02}, \dots, Z_{0n})$.

Let $Z_k^j = (Z_{k1}^j, Z_{k2}^j, \dots, Z_{kn}^j)$.

Let $f_k^{(j)}(s)$ denote the probability generating function for Z_k^j .

Leaving off the superscripts, we have the following standard result.

2.3 Theorem. $f_{k+n}(s) = f_k(f_n(s))$.

Proof: See Harris (1963).

One of the most important and useful concepts is the following.

2.4 Definition. The first moment matrix, M , is defined as follows:

$M = (m_{ij})$, where

$m_{ij} = E[Z_{ij}^1]$ = the expected number of type j offspring of a single object of type i .

An important result about the first moment matrix involves the expected number of offspring in the n^{th} generation.

2.5 Theorem. $E[Z_n | Z_0] = Z_0 M^n$.

Proof: This can be proved using generating functions. Details will be omitted here. See Mode (1971).

A process can be classified according to characteristics of its first moment matrix.

2.6 Definition. A multitype Galton-Watson branching process is called

a. irreducible if, for every pair of integers i, j , there exists a positive integer n such that $(M^n)_{ij} > 0$ (for a matrix A , its ij^{th} component is denoted A_{ij}),

b. positively regular if there exists a positive integer n such that all elements of M^n are positive. An irreducible process is called

singular if $f(s) = Ms^T$ for some Markov transition matrix M . ("T" denotes "transpose.")

Note that if a process is singular then it reduces to an ordinary finite Markov chain whose state space is the set of types and with transition probabilities given by M . In this case each object (nonterminal) has exactly one "offspring."

2.7 Assumption. All processes are assumed to be non-singular unless otherwise stated.

For any matrix M let $\rho(M)$ (or just ρ) denote its "spectral radius," defined by:

$$\rho(M) = \max \{ \|e\| : e \text{ is an eigenvalue of } M \}.$$

Here $\|\cdot\|$ denotes either real or complex norm. According to a theorem of Frobenius ρ always exists for an irreducible matrix. See Karlin and Taylor (1975). One can also define ρ for a reducible matrix by the following. If a matrix is reducible then it can be put into the following lower triangular form:

$$\begin{bmatrix} M(1) & 0 & \dots & \dots & 0 \\ M(2, 1) & M(2) & & & \vdots \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \\ M(k, 1) & & & M(k) & \end{bmatrix}$$

where each $M(i)$ is irreducible, $1 \leq i \leq k$. See, for example, Householder (1964). Let ρ_i denote the spectral radius of $M(i)$.

2.8 Definition. With the above notation, define $\rho(M) = \max \rho_i, 1 \leq i \leq k$.

Finally, one can also define non-singularity for a reducible process.

First, define an equivalence relation $\sim$ on the set of types by:

type $i \sim$ type j iff there exist non-negative integers m, n such that $M_{ij}^{(m)} > 0$ and $M_{ji}^{(n)} > 0$. Here $M_{ij}^{(n)}$ denotes the ij th component of M^n , where $M = (m_{ij})$ is the first moment matrix. In this case we say "type i communicates with type j ." This partitions the set of types into equivalence classes C_k . A process is reducible iff all types do not communicate. For a reducible process, whose first moment matrix is in the above lower triangular form, let $Z(k, n) = \{Z_j(n) : j \in C_k\}$. Then $Z(k, n)$ is an irreducible multitype Galton-Watson branching process with first moment matrix $M(k)$.

2.9 Definition. A reducible process is called non-singular if all the "subprocesses" $Z(k, n)$ are non-singular.

The problem of extinction of a process can now be precisely stated.

2.10 Definition. Let $q = (q^1, \dots, q^n)$, where

$$q^i = P[Z_m^i = 0 \text{ for some } m].$$

Thus, q^i = the probability of eventual extinction of the process that starts with a single object of type i . Using all the above notation and terminology, the major result about extinction can be put into a single theorem: (Here $f(s)$ is the generating function given in 2.1 and e_i denotes the standard unit vector in R^n , $e_i = (0, 0, \dots, 1, \dots, 0, 0)$, "1" in the i th position).

2.11 Theorem. (Harris, 1963)

a) If a non-singular multitype Galton-Watson branching process has $\rho \leq 1$, then $q = (1, 1, \dots, 1)$, given that $Z_0^i = e_i$ for $1 \leq i \leq n$.

b) Regardless of ρ , q is the smallest non-negative solution of the vector equation $q = f(q)$.

That is, if we begin in generation zero with a single object of each type and if $\rho \leq 1$, then the population becomes extinct with probability 1.

Of course the meaning of ρ depends on the particular type of process. If $\rho > 1$, using part b we can calculate q (in theory at least) and find the "probability of extinction." Some examples will be given in the next section.

3. Applications to Probabilistic Context-Free Grammars

Of fundamental importance in the theory of p-grammars is whether or not a given p-grammar (G, F) generates a p-language. That is, when is (G, F) consistent? (Definition 1.9, section II) One way to answer this question is to find the induced probability function ϕ on $L(G)$ and compute $\sum_{x \in L(G)} \phi(x)$. In general this is a difficult problem that can be solved only for certain grammars. Finding an explicit expression for $\phi(x)$ is itself usually difficult. For p - CFG's one can use the results of the previous section. Let $G = (V_N, V_T, P, S)$ be a p - CFG with $V_N = \{A_1, A_2, \dots, A_n\}$ where $A_1 = S$. Let $Z_0 = (1, 0, \dots, 0)$ and Z_{ki} = number of A_i 's in the k^{th} generation (see section III 1). Let $Z_0 = (1, 0, \dots, 0)$ and $Z_k = (Z_{k1}, Z_{k2}, \dots, Z_{kn})$. Then obviously $\{Z_n \mid n = 0, 1, 2, \dots\}$ is a multitype Galton-Watson branching process. To use the results in section II one must first characterize the type of branching process and then compute ρ . The computation of ρ can be difficult. An iterative algorithm is presented in Mode (1971). However, we are not too concerned with the exact value of ρ , only if $\rho \leq 1$. In the next section we list some known results about ρ that are useful in

determining if $\rho \leq 1$. The following is a rephrasing of part of Theorem 2.11 along with a partial intuitive proof that does not use the results of the previous section.

3.1 Theorem. A nonsingular p - CFG, G , is consistent iff $\rho(M) \leq 1$, where M = the first moment matrix of the process.

"Proof": We show that if $\rho < 1$ then G is consistent.

Let $\{A_1, A_2, \dots, A_n\}$ denote the set of nonterminals, with $A_1 = S$.

Let r_{ij} = expected number of A_j 's in the i^{th} generation. Let

$R_i = [r_{i1}, r_{i2}, \dots, r_{in}]$. Then G is consistent if $\lim_{i \rightarrow \infty} R_i = 0$

(that is, the process becomes "extinct"). Consider the product $R_{i-1} \cdot M$.

The j^{th} element of $R_{i-1} \cdot M = (R_{i-1} \cdot M)_j = \sum_{k=1}^n (R_{i-1})_k \cdot M_{kj} = (R_i)_j$.

Thus $R_i = R_{i-1} \cdot M = R_{i-2} \cdot M^2 = \dots = R_0 M^i = [1, 0, \dots, 0] M^i$.

So $\lim_{i \rightarrow \infty} R_i = 0$ iff $\lim_{i \rightarrow \infty} M^i = 0$, which is true iff $\rho < 1$. See Householder (1964).

3.2 Example. $G = (V_N, V_T, P, S)$, $V_N = \{S, A\}$, $V_T = \{a, b\}$, and P is

$$1 : S \rightarrow Ab$$

$$p : A \rightarrow AA$$

$$q : A \rightarrow a, \quad p + q = 1$$

Then $L(G) = \{a^n b \mid n \geq 1\}$. The first moment matrix is

$$M = \begin{array}{cc} & \begin{array}{cc} S & A \end{array} \\ \begin{array}{c} S \\ A \end{array} & \begin{bmatrix} 0 & 1 \\ 0 & 2p \end{bmatrix} \end{array}.$$

In this case we can find M^n explicitly:

$$M^n = \begin{bmatrix} 0 & (2p)^{n-1} \\ 0 & (2p)^n \end{bmatrix}$$

It is clear that $M^n \rightarrow 0$ iff $2p < 1$ or $p < \frac{1}{2}$. So, by Theorem 2.5, G is consistent if $p < \frac{1}{2}$. But M is reducible and can therefore be put into lower triangular form by rearranging the nonterminals:

$$\begin{array}{cc} & \begin{array}{cc} A & S \end{array} \\ \begin{array}{c} A \\ S \end{array} & \begin{bmatrix} 2p & 0 \\ 1 & 0 \end{bmatrix} \end{array}$$

The spectral radius of this matrix is found by considering the 1×1 diagonal matrices $[2p]$, $[0]$. Thus $\rho(M) = 2p$ (see Definition 2.8). We conclude that G is consistent iff $p \leq \frac{1}{2}$. Note that if $p = \frac{1}{2}$, then

$$M^n = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

for all n and it is not true that $M^n \rightarrow 0$. However, G is still consistent for $p = \frac{1}{2}$. Another way of proving this is by calculating the induced probabilistic word function $\phi(x)$. It can be shown that

$$\phi(a^{n+1}b) = \binom{2n}{n} \frac{1}{n+1} p^n q^{n+1}, \quad n \geq 0 \quad (\text{see IV 2.7})$$

$$\begin{aligned} \text{Hence } \sum_{x \in L(G)} \phi(x) &= \sum_{n=0}^{\infty} \phi(a^{n+1}b) \\ &= \sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} \left(\frac{1}{2}\right)^n \left(\frac{1}{2}\right)^{n+1}, \quad \text{if } p = q = \frac{1}{2}. \\ &= \frac{1}{2} \sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} \left(\frac{1}{4}\right)^n \end{aligned}$$

$$\text{But } \sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} x^n = \frac{2}{1 + \sqrt{1-4x}} \quad \text{if } x \leq \frac{1}{4}.$$

See Knuth (1969, p. 389).

$$\text{So } \sum_{x \in L(G)} \phi(x) = \frac{1}{2} \cdot \frac{2}{1+0} = 1 \text{ if } p = q = \frac{1}{2}.$$

3.3 Example. $G = (V_N, V_T, P, S)$, $V_N = \{S, B\}$, $V_T = \{a, b\}$ and P is

$$p : S \rightarrow aSBS$$

$$q : S \rightarrow aB$$

$$r : B \rightarrow bS$$

$$t : B \rightarrow b,$$

where $p > 0$, $r > 0$, $p + q = 1$, and $r + t = 1$. The first moment matrix is

$$M = \begin{array}{cc} & \begin{array}{cc} S & B \end{array} \\ \begin{array}{c} S \\ B \end{array} & \begin{bmatrix} 2p & 1 \\ r & 0 \end{bmatrix} \end{array}.$$

Note that M is positively regular since all elements of M^2 are positive.

The spectral radius $\rho(M)$ is found to be $p + \sqrt{p^2 + r}$. So G is consistent iff $p + \sqrt{p^2 + r} \leq 1$. This is true iff $p \leq \frac{1}{2}$ and $r \leq 1 - 2p$ (for example $p = \frac{1}{4}$, $q = \frac{3}{4}$, $r = \frac{1}{2}$, $t = \frac{1}{2}$). It is difficult to determine $L(G)$ and find the induced probabilistic word function $\phi(x)$. The probability generating function (see section 2.1) is $f(s) = f(s_1, s_2) = (ps_1^2s_2 + qs_2, rs_1 + t)$. If G is not consistent one can calculate the "probability of extinction" or the probability that G will generate a p -language. This is done by solving $f(s_1, s_2) = (s_1, s_2)$ which, in this example, is straightforward but non-trivial.

4. Useful Results About the Spectral Radius

We list here some of the most useful results about $\rho(A)$, the spectral radius of an $n \times n$ matrix $A = (a_{ij})$. Details can be found in Householder (1964) and Wielant (1967).

$$4.1 \quad \lim_{n \rightarrow \infty} A^n = 0 \quad \text{iff} \quad \rho(A) < 1.$$

$$4.2 \quad (I-A)^{-1} = I + A + A^2 + \dots \quad \text{iff} \quad \rho(A) < 1.$$

Here I denotes the $n \times n$ identity matrix.

4.3 For any matrix s ,

$$\rho(s^{-1}As) = \rho(A).$$

$$4.4 \quad \rho(A) = \rho(A^T).$$

Here "T" denotes "transpose."

4.5 If $B = (b_{ij})$ is an $n \times n$ matrix and $|a_{ij}| \leq b_{ij}$ for all i, j , then $\rho(A) \leq \rho(B)$.

$$4.6 \quad \rho(A) \leq \max_i \sum_{j=1}^n |a_{ij}|.$$

4.7 There exists an index i such that

$$|\rho(A) - a_{ii}| \leq \sum_{\substack{k=1 \\ k \neq i}}^n |a_{ik}|.$$

5. Finding the Expected Derivation Length and Expected Word Length

Suppose G is a consistent p - CFG with $V_N = \{A_1, \dots, A_k\}$. Let M denote its first moment matrix.

5.1 Definition. For any $A \in V_N$, let $EDL(A)$ denote the "expected derivation length" = the expected number of steps in any derivation (of an arbitrary $x \in V_T^*$) beginning with A .

The number of steps in a derivation is the same as the number of non-terminals introduced. The element M_{ij} is just the expected number of

A_j 's when rewriting A_i once (Definition 2.4). Hence the expected total number of nonterminals introduced (when rewriting A_i once) is the sum of the elements in the i^{th} row of M , that is $\sum_{j=1}^n M_{ij}$.

From Theorem 2.5 one concludes that $(M^n)_{ij}$ = the expected number of A_j 's when rewriting A_i in n steps. Hence the expected number of A_j 's after an arbitrary derivation beginning with A_i is the ij^{th} element of $I + M + M^2 + \dots = (I-M)^{-1}$ if $\rho(M) < 1$ (see 4.2). Therefore, we have the following.

5.2 Theorem. (Hutchins, 1972) Suppose G is a consistent p - CFG with $\rho < 1$ and $V_N = \{A_1, \dots, A_k\}$. Then $EDL(A_i)$ = sum of the elements in the i^{th} row of $(I-M)^{-1}$, where M = the first moment matrix of G .

Although concerned primarily with non-terminals, one can still make some conclusions about terminal strings.

5.3 Definition. For any $A \in V_N$, let $EWL(A)$ denote the "expected word length" = the expected number of terminal symbols in a terminal string derived from A .

Note that $EWL(S) = \sum_{x \in L(G)} |x|p(x)$, where $p(x)$ denotes the probability of x . Let W denote the $|V_N| \times |V_T|$ matrix with entries W_{ij} = expected number of occurrences of the j^{th} terminal after one rewriting of the i^{th} nonterminal. The following result can be proved in a manner similar to Theorem 5.2.

5.4 Theorem. (Sankoff, 1971) Suppose G is a consistent p - CFG with $\rho < 1$ and $V_N = \{A_1, \dots, A_k\}$. Then $EWL(A_i) =$ sum of the elements in the i^{th} row of $(I-M)^{-1} \cdot W$, where M = the first moment matrix of G and W is given above.

5.5 Example. $G = (V_N, V_T, P, S)$, $V_N = \{S, B\}$, $V_T = \{a, b\}$, P is

$$p : S \rightarrow aSB$$

$$q : S \rightarrow aB$$

$$1 : B \rightarrow b, \quad p + q = 1, \quad p < 1.$$

Then

$$M = \begin{array}{c} \begin{array}{cc} & \begin{array}{cc} S & B \end{array} \\ \begin{array}{c} S \\ B \end{array} & \begin{bmatrix} p & 1 \\ 0 & 0 \end{bmatrix} \end{array}$$

and $\rho(M) = p < 1$.

$$(I-M)^{-1} = \begin{bmatrix} \frac{1}{1-p} & \frac{1}{1-p} \\ 0 & 1 \end{bmatrix}, \quad W = \begin{array}{c} \begin{array}{cc} a & b \\ \begin{array}{c} S \\ B \end{array} & \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{array}.$$

$$\text{So } EDL(S) = \frac{2}{1-p}, \quad EWL(S) = \frac{2}{1-p}.$$

Intuitively, if p is close to 1 then we are much more likely to use the production $S \rightarrow aSB$ when rewriting S . This increases the length of a derivation. Note that if p is near 1 then $1-p$ is "small" so $\frac{2}{1-p}$ is "large."

5.6 Example. $G = (V_N, V_T, P, S)$, $V_N = \{S, B\}$, $V_T = \{a, b\}$ and P is

$$.4 : S \rightarrow aSB$$

$$.6 : S \rightarrow aBB$$

$$.3 : B \rightarrow abS$$

$$.7 : B \rightarrow bb.$$

In this case one can calculate:

$$\rho(M) \approx .92$$

$$EDL(S) \approx 21.7$$

$$EWL(S) = 35 .$$

6. Applications to Infinite Series

There are two ways of determining whether or not a p - CFG is consistent. One way involves determining the probabilistic word function $\Phi(x)$ and calculating $\sum_x \Phi(x)$. The second method deals with the spectral radius of the first moment matrix. Moreover, there exists an algorithm for calculating the spectral radius (Mode, 1971, p. 103). If it is possible to determine $\Phi(x)$ then one can derive a result about the infinite series (usually) $\sum_x \Phi(x)$.

6.1 Example. Consider the p - CFG given in section III, 3.2. The function $\Phi(x)$ is explicitly given. Moreover, G is consistent iff $p \leq \frac{1}{2}$.

Conclusion:

$$\sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} p^n (1-p)^{n+1} = 1 \quad \text{iff} \quad p \leq \frac{1}{2} .$$

or, rewriting the sum,

$$\sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} [p(1-p)]^n = \frac{1}{1-p} \quad \text{iff} \quad p \leq \frac{1}{2} .$$

6.2 Example. The above example can be generalized.

Let $G_k = (\{S\}, \{a\}, P, S)$, where P is given by:

$$p : S \rightarrow S^k$$

$$q : S \rightarrow a , \text{ with } p + q = 1 .$$

Then one calculates $L(G_k) = \{a^{nk-n+1} \mid n \geq 0\}$ and the induced probabilistic word function is

$$\phi(a^{nk-n+1}) = \binom{nk}{n} \frac{1}{nk-n+1} p^n q^{nk-n+1}.$$

The first moment matrix is just $[kp]$.

Conclusion:

$$\sum_{n=0}^{\infty} \binom{nk}{n} \frac{1}{nk-n+1} p^n q^{nk-n+1} = 1 \quad \text{iff} \quad p \leq \frac{1}{k}.$$

Here $k > 1$ and $q = 1 - p$.

IV. FINDING PROBABILITIES

1. The Generating Function Approach for Regular Expressions

Parker and McClusky (1978) approach some probability problems using regular expressions. Specifically, they show how to obtain multistep transition probabilities for a sequential circuit directly from the regular expressions. That is, one can find the probability of an automaton accepting a finite sequence of symbols given the probabilities of the individual symbols. Their approach can be formalized as follows. For any regular expression, E , they define a probability generating function $\text{gen}(E)$. The probability of accepting a string of length m is the co-efficient of t^m in $\text{gen}(E)$, when written as a power series in the variable t . More precisely, this probability is $\frac{1}{m!} \cdot \frac{d^m}{dt^m} \text{gen}(E)$. This function, $\text{gen}(E)$, is constructed as follows: let Σ denote any alphabet of symbols. Let $+$, $\cdot$, $*$ denote the regular operators union, concatenation, and Kleene star operation.

1.1 Definition. For $a \in \Sigma^*$, let $\text{gen}(a) = p(a)t^{|a|}$, where $p(a)$ = probability of a , $m = |a|$, and t is a dummy variable.

Let $\text{gen}(a^*) = \sum_{j=0}^{\infty} [p(a)t]^j = \frac{1}{1-p(a)t}$ in closed form.

For $a, b \in \Sigma^*$, define $\text{gen}(ab) = \text{gen}(a) \text{gen}(b)$ and $\text{gen}(a+b) = \text{gen}(a) + \text{gen}(b)$.

For an expression α , let $\text{gen}(\alpha^*) = \sum_{j=0}^{\infty} [p(\alpha)t]^j$.

1.2 Example. $E = ab^* + c$

$$\text{Then } \text{gen}(E) = xt \sum_{j=0}^{\infty} (yt)^j + zt = \frac{xt}{1-yt} + zt ,$$

where $x = p(a)$, $y = p(b)$, and $z = p(c)$.

2. A Generalization to Context-Free Languages

This generating function approach to probabilities can be extended to context-free languages. We first need a characterization of CFL's analogous to the characterization of regular languages via regular expressions. The key concept here is the notion of "nested iterated substitution."

2.1 Definition. Let Σ be a fixed alphabet. For any $a \in \Sigma$, let Σ_a be a finite alphabet and let $f(a) \subseteq \Sigma_a^*$. Then f is a substitution over Σ if $f(\lambda) = \{\lambda\}$ and $f(a_1 a_2 \dots a_k) = f(a_1) f(a_2) \dots f(a_k)$ for all $a_i \in \Sigma$. Note that f is a function, $f : \Sigma^* \rightarrow \text{subsets of } (\bigcup_a \Sigma_a)^*$. The function f is extended to sets by:

$$f(L) = \sum_{x \in L} f(x) \text{ for } L \subseteq \Sigma^* ,$$

An iterated substitution is f^* , defined by:

$$f^*(L) = \sum_{n \geq 0} f^n(L) .$$

f^* is a nested iterated substitution if $a \in f(a)$ for each $a \in \Sigma$.

The concept of nested iterated substitution plays the role of the star operation in the case of regular languages. Various forms of nested iterated substitution have been studied. There are Gruska's "regular-like" or "context-free" expressions (Gruska, 1971). There are the "substitution expressions" of McWhirter (1971). This is essentially a generalization of

Gruska in which concatenation is not a primitive operation but can be simulated by substitution. There are the "cap expressions" of Yntema (1971). This seems to be just a rephrasing of Gruska's work.

We shall consider Gruska's characterization of context-free language. First, we show how to get a probability generating function if we are given the productions of a context-free grammar. Let $G = (V_N, V_T, P, S)$ be a CFG. Assume G is "reduced" (every variable produces a terminal string and every non-terminal occurs in a sentential form). Let $V_N = \{A_1, A_2, \dots, A_n\}$, with $A_1 = S$. Define a homomorphism, Φ , as follows:

$$\Phi(A_i) = y_i, \text{ a (complex) variable;}$$

for $a \in V_T$, $\Phi(a) = xt$, where $x = p(a)$, the "probability of a " and t is a dummy (complex) variable. Extend Φ to any sequence of symbols by concatenation. That is, $\Phi(x_1 x_2 \dots x_n) = \Phi(x_1) \Phi(x_2) \dots \Phi(x_n)$, where each $x_i \in (V_N \cup V_T)^*$.

2.2 Definition. The probability generating equations of G are:

$$\Phi(A_i) = \sum_j \Phi(\alpha_{ij}) , \text{ where } A_i \rightarrow \alpha_{ij} \text{ is in } P, 1 \leq i \leq n.$$

One can show that these equations have unique analytical solutions $y_i = f_i(t)$, with $f_i(0) = 0$, on a disk about $t = 0$.

2.3 Definition. $g(t) = f_1(t)$ = the probability generating function of G .

The function $g(t)$ can be put into the form $\sum_{n=1}^{\infty} u(n)t^n$.

2.4 Note. If we set all probabilities = 1 (that is, for $a \in V_T$, set $\Phi(a) = t$), then $u(n)$ = number of derivations of words of length n generated by G . See, for example, Shamir (1968).

Let P_m denote the probability of a string of length m .

2.5 Theorem. $P_m = u(m) = \frac{1}{m!} \cdot \frac{d^m}{dt^m} g(t)$ evaluated at $t = 0$.

Proof: By construction, each terminal symbol in a string introduces another power of t in $g(t)$, together with the probability of that symbol. If $x = a_1 a_2 \dots a_m$, where each $a_i \in V_T$, then $p(x) = p(a_1) p(a_2) \dots p(a_m)$, where p denotes probability. In $g(t)$ this means that the coefficient of t^m is the sum of quantities of the form $p(x)$, where x ranges over all strings of length m . That is, $u(m) = P_m$. The second equality is well known.

Hence the probabilities can be calculated directly from $g(t)$. To find the expansion of $g(t)$ as a power series, the "Lagrange inversion formula" is sometimes useful. A consequence of this follows.

2.6 Theorem. If y is of the form:

$$y = z \cdot f(y) + a,$$

$$\text{then } y = a + \sum_{n=1}^{\infty} \frac{z^n}{n!} \cdot \frac{d^{n-1}}{da^{n-1}} [f(a)]^n.$$

Proof: See Harrison (1978).

2.7 Example. Consider $G = (\{S\}, \{a, b, c\}, P, S)$, where P is

$$S \rightarrow aSbS$$

$$S \rightarrow c$$

Note that G is unambiguous and not regular. The probability generating equation is

$$\Phi(S) = \Phi(aSbS) + \Phi(c)$$

or

$$y = pty \cdot qty + rt, \text{ where } p = p(a), q = p(b), r = p(c).$$

So $y = pqt^2y^2 + rt$.

By Lagrange's result, after simplification, we have:

$$y = g(t) = \sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} p^n q^n r^{n+1} t^{3n+1}.$$

For example, P_7 = the probability of a string of length 7 = the coefficient of $t^7 = \binom{4}{2} \cdot \frac{1}{3} p^2 q^2 r^3 = 2p^2 q^2 r^3$.

Note that there are two strings of length 7 in $L(G)$:

$a^2 c b c b c$

and

$a c b a c b c$, each with probability $p^2 q^2 r^3$.

Now, given a "context-free expression" (CFE) as defined by Gruska (1971), one can find its probability generating function as follows. If $a \in V_T$, then $\text{gen}(a) = xt$, where $x = p(a)$. For $a, b \in V_T$, let $\text{gen}(ab) = \text{gen}(a)\text{gen}(b)$. For expressions E_1, E_2 , $\text{gen}(E_1 + E_2) = \text{gen}(E_1) + \text{gen}(E_2)$. Finally, for any expression of the form

$$E = (\alpha_1 + \alpha_2 + \dots + \alpha_k)^S,$$

find $\text{gen}(E)$ by writing $S \rightarrow \alpha_1 + \alpha_2 + \dots + \alpha_k$ and use the methods previously given.

2.8 Example. Let $E = (aSb + ab)^S c + d$. First find $\text{gen}(E_0)$, where

$$E_0 = (aSb + ab)^S.$$

We have $y = pqt^2y + pqt^2$. In this case we can find y easily:

$$\begin{aligned} \text{gen}(E_0) = y &= \frac{pqt^2}{1-pqt^2} \\ &= \sum_{i=1}^{\infty} (pqt^2)^i, \text{ where } p = p(a), q = p(b) \end{aligned}$$

(as an infinite series).

$$\begin{aligned} \text{Hence, } \text{gen}(E) &= st + rt \cdot \sum_{i=1}^{\infty} (pqt^2)^i, \quad r = p(c), \quad s = p(d) \\ &= st + \frac{pqrt^3}{1-pqt^2}. \end{aligned}$$

3. Determining Production Probabilities

Probabilities for the productions of a p-grammar may be estimated by empirical methods. Suppose that $G = (V_N, V_T, P, S)$ is a p-grammar. Let $V_N = \{A_1, A_2, \dots, A_n\}$. Order the productions as follows:

$$\begin{aligned} &P_{11}, P_{12} \dots \dots P_{1n_1} \\ &P_{21}, P_{22} \dots \dots P_{2n_2} \\ &\vdots \\ &\vdots \quad \text{etc.} \end{aligned}$$

where P_{ij} is of the form $A_i \rightarrow \text{something}$. Let S be a "sample" drawn from $L(G)$, that is, a finite collection of words (repetition allowed).

Let f_{ij} = number of times p_{ij} was used in deriving elements of S , and
 N_i = number of times productions of the form $A_i \rightarrow \text{something}$ were used.

$$\text{Let } \hat{p}_{ij} = \frac{f_{ij}}{N_i}.$$

Then it seems that these "sample probabilities," $\hat{p}_{ij}$, should approach the "true" production probabilities, p_{ij} as S is enlarged. In fact, the following is true.

3.1 Theorem. For an unambiguous p-grammar, $\hat{p}_{ij}$ is the maximum likelihood estimator of p_{ij} .

Proof: Although this was first stated and proved by Fu and Booth (1975), it is a consequence of a standard statistical result about a "binomial experiment." See, for example, Mendenhall and Scheaffer (1973, p. 303).

3.2 Example. Let $V_N = \{S, B\}$, $V_T = \{a, b\}$ and P :

$$p_{11} : S \rightarrow aSB$$

$$p_{12} : S \rightarrow aB$$

$$1 : B \rightarrow b$$

where p_{11} and p_{12} are unknown probabilities. Clearly,

$$L(G) = \{a^i b^i \mid i \geq 1\}.$$

Suppose a sample S is drawn from $L(G)$, say

$$S = \{ab, aaabbb, aaaabbbb\}.$$

Then, in the above notation:

$f_{11} = 5$, $f_{12} = 3$, $f_{21} = 8$, $N_1 = 8$, $N_2 = 8$, so $\hat{p}_{11} = \frac{5}{8}$ and $\hat{p}_{12} = \frac{3}{8}$ are estimates of p_{11} and p_{12} , respectively.

These methods can be used to study the probability structure of a programming language. Choose S to be a sample of programs. Then parse each program and count the number of times each production was used. One can then find the estimators $\hat{p}_{ij}$ as above.

V. PARIKH'S THEOREM--A PROBABILISTIC VERSION?

1. Introduction

In 1961 R. J. Parikh proved a result that was later described as "among the most fundamental, yet subtly difficult to prove, in the theory [of context-free languages]" (Parikh, 1966). This result, now known as Parikh's Theorem, states that the commutative image of a context-free language is regular. That is, if the order of symbols is ignored, then context-free languages are indistinguishable from regular sets. We attempt to answer the question: Does there exist a "stochastic" or "probabilistic" version of Parikh's Theorem? First, we need to be more precise regarding the above statement.

1.1 Definition. Two words $x, y \in V^*$ are letter-equivalent (le) iff x is obtained from y by a permutation of symbols. (That is, for each symbol $a \in V$, both x and y contain the same number of occurrences of a .)

1.2 Definition. Two languages L_1 and L_2 are letter-equivalent iff, for each $x \in L_1$ there is a $y \in L_2$, and vice-versa.

Parikh's Theorem asserts that for any context-free language, there exists a le regular language. For example, the CFL $L_1 = \{a^n b^n \mid n \geq 1\}$ is le to the regular language $L_2 = \{(ab)^n \mid n \geq 1\}$. In fact, this simple example can be made "probabilistic." Consider the following p-grammars.

$$G_1 = (\{S, B\}, \{a, b\}, P_1, S)$$

where P_1 is

$$p : S \rightarrow aSB$$

$$q : S \rightarrow aB$$

$$1 : B \rightarrow b$$

with $p + q = 1$, $p < 1$. Also, $G_2 = (\{T\}, \{a, b\}, P_2, T)$ where P_2 is

$$p : T \rightarrow abT$$

$$q : T \rightarrow ab$$

with the same p, q as in P_1 . Then $L(G_1) = L_1$ and $P[a^n b^n] = p^{n-1} q$.

Also $L(G_2) = L_2$ and $P[(ab)^n] = p^{n-1} q$. So, regardless of the values of p and q , the probabilities of letter-equivalent words are "preserved."

That is, if x is le to y , then $P[x] = P[y]$. Certainly this is an ideal situation that would not seem to hold in all cases. In a sense, we are asking when a multitype Galton-Watson branching process (a probabilistic CFG) is "equivalent" to a simple Markov chain (a probabilistic regular grammar) to the extent that the probabilistic and occurrences of terminal symbols in realizations of the Galton-Watson process are preserved in realizations of the Markov process (see III 2.6). In the sequel we shall examine all known approaches to proving Parikh's original result and investigate the possibilities of extending these to probabilistic languages. More notation and terminology are needed.

Let N denote the set of non-negative integers and N^n denote the set of ordered n -tuples of non-negative integers. N^n is a commutative, associative semigroup with identity, in which addition and scalar multiplication are termwise.

1.3 Definition. A set $S \subseteq N^n$ is called linear iff it is of the form

$$S = \{v + \sum_{i=1}^k x_i w_i \mid x_i \in N\}, \text{ for some values of } k \text{ and } v, w_1, w_2, \dots, w_k \in N^n.$$

1.4 Example. $S = \{(1,1,1), (2,4,6), (3,5,7), (4,7,10), \dots\}$ is linear since $S = \{(1,1,1) + n(1,2,3) \mid n \geq 0\}$.

1.5 Definition. A set $S \subseteq N^n$ is semilinear iff it is a finite union of linear sets.

Let $V = \{a_1, a_2, \dots, a_n\}$. For any $a_i \in V$ and $x \in V^*$, let $N_{a_i}(x)$ denote the number of occurrences of a_i in x .

1.6 Definition. The Parikh mapping is the function

$$\psi : V^* \rightarrow N^n$$

given by

$$\psi(x) = (N_{a_1}(x), N_{a_2}(x), \dots, N_{a_n}(x)).$$

For any $L \subseteq V^*$, define $\psi(L) = \{\psi(x) \mid x \in L\}$. We say that L has a semilinear image if $\psi(L)$ is a semilinear set.

Note that ψ induces an equivalence relation "le" on languages and that $L_1 \text{ le } L_2$ iff $\psi(L_1) = \psi(L_2)$.

2. Parikh's Approach

Parikh (1966) proved the following.

2.1 Theorem. Let G be a context-free grammar and $L = L(G)$. Then

- a) $\psi(L)$ is semilinear and
- b) a canonical description of $\psi(L)$ can be found effectively from G .

We shall primarily be concerned with part a. The following result explains the relationship between semilinearity and letter-equivalence.

2.2 Theorem. A language $L \subseteq \Sigma^*$ has a semilinear image if and only if it is letter-equivalent to a regular set.

Proof sketch: The proof consists of a straightforward application of Kleene's Theorem on the characterization of regular sets: the family of regular sets over Σ is the smallest family which contains $\{a\}$ for each $a \in \Sigma$ and is closed under union, concatenation, and $*$. (Often, this is taken as the definition of regular sets). To show that any regular set has a semilinear image, one shows the following: if $\psi(L_1)$ and $\psi(L_2)$ are semilinear, then so are $\psi(L_1) \cup \psi(L_2)$, $\psi(L_1 L_2)$, and $\psi(L_1^*)$. The converse is proven constructively. If a language, L , has a semilinear image, one can easily construct a right-linear grammar G with $\psi(L(G)) = \psi(L)$.

In this work (and most of the literature) the following result is referred to as "Parikh's Theorem."

2.3 Corollary. (Parikh's Theorem) Any CFL L is le to a regular set.

Proof: Follows directly from Theorems 2.1 and 2.2.

We wish to extend these results to p-languages. To avoid certain pathological situations we make the following assumption (see III 3).

2.4 Assumption. Any probabilistic CFL can be generated by a probabilistic CFG.

The following seems to be a reasonable extension of "letter-equivalence."

2.5 Definition. Let (L_1, Φ_1) and (L_2, Φ_2) be p-languages. We say that L_1 is probabilistically letter-equivalent (ple) to L_2 if

a) L_1 is le to L_2 and

b) if $x \in L_1$ is le to $y \in L_2$, then $\phi_1(x) = \phi_2(y)$ (that is, letter-equivalence of strings preserves probabilities).

However, this is not a useful definition unless further restrictions are placed on L_1 and L_2 . The notion of ple is not even an equivalence relation as the following example shows.

2.6 Example. Let $G = (\{S\}, \{a, b\}, P, S)$, where P is

$$p : S \rightarrow aS$$

$$q : S \rightarrow ab$$

$$r : S \rightarrow ba, \quad p + q + r = 1, \quad p < 1, \quad q \neq r.$$

Then $L = L(G) = \{a^i b \mid i \geq 1\} \cup \{a^i ba \mid i \geq 0\}$.

Also: $a^i b$ is le to $a^{i-1} ba$ but, $P[a^i b] = p^{i-1} q$ and $P[a^{i-1} ba] = p^{i-1} r$.

Hence ple is not a reflexive relation since L is not ple to itself.

We restrict our discussion to the class of languages on which ple is reflexive:

2.7 Definition. A p-language (L, ϕ) will be called ψ -preserving if $\psi(x) = \psi(y)$ implies $\phi(x) = \phi(y)$, where ψ is the Parikh mapping.

That is, if $x, y \in L$ and x is le to y , then $\phi(x) = \phi(y)$. So the probability of a word depends only on the symbols used, not on their order in the word.

2.8 Assumption. Unless otherwise stated, all p-languages given below are ψ -preserving.

This is a strong, but necessary, assumption that must be made if we continue to use our definition of ple (2.5). The probabilistic version of Parikh's Theorem now takes the following form:

2.9 Conjecture. Any p - CFL, L, is ple to a p-regular set.

This conjecture is false. A counterexample is given below (3.6).

3. Pilling's Methods

D. L. Pilling has shown how to generalize Parikh's result while avoiding the concept of semilinearity. See Pilling (1973) or Conway (1971). His main result is the following theorem.

3.1 Theorem. The minimal solutions of a set of regular equations

$$x_i = f_i(x_1, \dots, x_n, A_1, \dots, A_m), 1 \leq i \leq n,$$

are regular.

To show that a CFL is le to a regular set, one proceeds as follows. Suppose we are given a CFG, G, that generates a CFL $L = L(G)$. If we assume that all the nonterminal and terminal symbols are free to commute with each other, then we can write down a corresponding system of regular equations. The above theorem implies that the solution is a regular set which is clearly le to L. The formal proof of Theorem 3.1 is constructive and hence very useful. An outline of the method follows.

Outline of proof of 3.1:

Consider a single equation $x_1 = f(x_1, x_2, \dots)$. Rewrite it as $x_1 = E + F(x_1) \cdot x_1$, where E is independent of x_1 . The minimal solution to this last equation is $F(E)^*E$. Use this as an induction step to eliminate the other variables one by one.

Thus, given a CFL, we can directly find a letter-equivalent set and give a regular expression for that set; it is not necessary to be

concerned about right-linear (regular) grammars. Therefore, any extension of Pilling's methods to p-languages should probably be phrased in terms of generating functions, since generating functions can be computed directly from regular expressions (section IV). The following example strengthens this claim.

3.2 Example. $G = (\{S\}, \{a, b, c\}, P, S), L = L(G)$ where P is

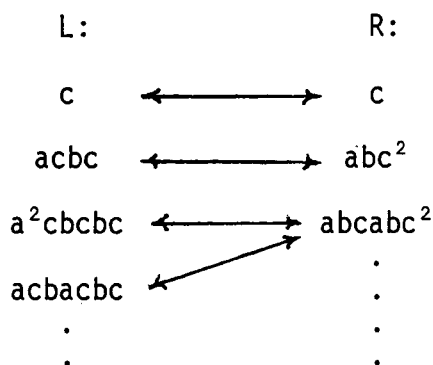
$$p : S \rightarrow aSbS$$

$$q : S \rightarrow c, \quad p + q = 1, \quad p < \frac{1}{2} \quad (\text{so that } G \text{ will be consistent}).$$

If ϕ is the probability function induced by G , then (L, ϕ) is a p-language. (It is easy to see that L is ψ -preserving). Using Pilling's methods, we obtain:

$$S = \underset{\substack{\uparrow \\ E}}{c} + \underbrace{(abS)}_{F(S)} S,$$

whose solution is $R = (abc)^*c$, a regular set that is le to L . We show that (L, ϕ) cannot be ple to $(R, \hat{\phi})$ for any probability function $\hat{\phi}$ on R . By consistency, $\sum_{x \in L} \phi(x) = 1$. Now examine the letter-equivalence between L and R :



If $(R, \hat{\phi})$ is a p-language, then $\sum_{y \in R} \hat{\phi}(y) = 1$, by definition.

$$\begin{aligned}
\text{But } 1 &= \sum_{x \in L} \phi(x) = \phi(c) + \phi(acbc) + \phi(a^2cbc bc) + \dots \\
&= \hat{\phi}(c) + \hat{\phi}(abc^2) + 2\hat{\phi}(abcabc^2) + \dots \\
&> \sum_{y \in R} \hat{\phi}(y) = 1,
\end{aligned}$$

a contradiction.

Hence, Pilling's methods do not immediately generalize to p-languages. The argument used in the above example can be generalized. For any p-language (L, ϕ) , let $\text{gen}_L(t)$ denote its probability generating function (see IV 1) with respect to ϕ .

3.3 Theorem. If (L_1, ϕ_1) is ple to (L_2, ϕ_2) , then $\text{gen}_{L_1}(t) = \text{gen}_{L_2}(t)$ (that is, for t sufficiently "small").

Proof: Let $P_{i,m}$ denote the probability of a word of length m in L_i , $i = 1, 2$. It suffices to show that $P_{1,m} = P_{2,m}$ (see IV 2.5). Let $x \in L_1$ be a word of length m . Let $\hat{x} \in L_2$ be ple to x . For any z , let $n_i(z)$ = number of words in L_i that are ple to z , $i = 1, 2$. Then, by an argument similar to that in Example 3.2, we have:

$$n_1(x) = n_2(\hat{x}).$$

Thus,

$$P_{1,m} = \sum n_1(x)\phi_1(x) = \sum n_2(\hat{x})\phi_1(x) = \sum n_2(\hat{x})\phi_2(\hat{x}) = P_{2,m}.$$

Each summation above is over representatives from each equivalence class obtained when ple is restricted to words of length m .

The converse of the above theorem is false, as the following example shows.

3.4 Example. Let L, G be as in Example 3.2. Let

$G_0 = (\{S\}, \{a, b, c\}, P_0, S)$, where P_0 is

$$p : S \rightarrow \text{SabS}$$

$$q : S \rightarrow c.$$

Let $L_0 = L(G_0)$ and Φ_0 be the probability function on L induced by G . Now $L_0 = (\text{cab})^*c$ is ple to L (Pilling, 1973). Also, it is clear that $\text{gen}_{L_0}(t) = \text{gen}_L(t)$, since both generating functions are obtained from the equation $S = \text{abS}^2 + c$. But L_0 is not ple to L ; let $x = \text{acbacbc}$ and $x_0 = \text{cabcbabc}$. Then $x \in L$, $x_0 \in L_0$, and $x \leq x_0$. But $\Phi(x) = p^2q^3$ and $\Phi_0(x_0) = 2p^2q^3$.

In the previous example, the reason that L_0 failed to be ple to L is that G_0 is ambiguous. This becomes apparent after the following results. First, in order to provide a counterexample to the probabilistic version of Parikh's Theorem (Conjecture 2.9), we need an interesting result, due to Takaoka. For any language, L , let $N(L, n)$ denote the number of words in L of length n . For any language $L \subseteq \Sigma^*$, define $\omega(L) = \sum_{t=0}^{\infty} N(L, t)r^{-t}$, where $r = |\Sigma|$.

3.5 Theorem. (Takaoka, 1978) If L is a regular language, $L \subseteq \Sigma^*$ and if $\omega(L)$ exists and is finite, then $\omega(L)$ must be a rational number.

3.6 Theorem. There exists a probabilistic CFL (L, Φ) that is not ple to any p-regular set.

Proof: Let $G = (\{S\}, \{a, b, c\}, P, S)$, where P is

$$S \rightarrow \text{aSbS}$$

$$S \rightarrow c.$$

Let $p = P[a]$, $q = P[b]$, $r = P[c]$ (see IV 1). Let $L = L(G)$ and ϕ be the probability function induced by p, q, r . We claim that (L, ϕ) cannot be ple to any p -regular set. Suppose R is a p -regular set. Assume that $R = L(\hat{G})$, for some unambiguous p -regular grammar $\hat{G}$. If L is ple to R then, by Theorem 3.3, $\text{gen}_L(t) = \text{gen}_R(t)$. But

$$\text{gen}_L(t) = \sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} p^n q^n r^{n+1} t^{3n+1},$$

which is the solution, y , to the equation $y = pqt^2y^2 + rt$ (see IV 2.7).

Then we have:

$$\begin{aligned} \binom{2n}{n} \frac{1}{n+1} &= \text{number of derivations of words of length } 3n+1 \text{ gener-} \\ &\quad \text{ated by } G \text{ (see IV 2.4)} \\ &= \text{number of derivations of words of length } 3n+1 \text{ gener-} \\ &\quad \text{ated by } \hat{G} \\ &= \text{number of words in } R \text{ of length } 3n+1, \text{ since } \hat{G} \text{ is} \\ &\quad \text{unambiguous} \\ &= N(R, 3n+1), \text{ by definition.} \end{aligned}$$

$$\text{But } \omega(R) = \sum_{n=0}^{\infty} N(R, 3n+1) 3^{-(3n+1)} = \sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} \frac{1}{3^{3n+1}},$$

The solution, y , to the equation $y = z^2y^2 + z$ is

$$\sum_{n=0}^{\infty} \binom{2n}{n} \frac{1}{n+1} z^{3n+1} \quad \text{or} \quad \frac{1 \pm \sqrt{1-4z^3}}{2z^2}.$$

When $z = \frac{1}{3}$, the second expression becomes a finite, irrational number, which, by Theorem 3.5, contradicts the regularity of R .

4. Van Leeuwen's Extension

Another generalization of Parikh's Theorem is due to J. van Leeuwen (1974), whose results lead to an algorithm for directly converting an arbitrary CFL into a letter-equivalence regular language.

Although his methods are entirely algebraic and seem complicated, they are similar to those of Pilling--but without using any "equations." One begins with a CFG generating a CFL and ends up with a letter-equivalent regular set. To do this the grammar is modified inductively. If we start with a p - CFG we will see to what extent the probabilities can be carried along during this modification. Some new notation and terminology is needed here.

4.1 Definition. A collection of languages F is called a family if it is non-empty, closed under isomorphism, and if all languages it contains have an alphabet chosen from a fixed set.

4.2 Definition. Let F be a family. A (context-free) F-grammar is a 4-tuple $G = (V_N, V_T, P, S)$, where V_N, V_T, S are as usual, and P consists of a finite set of productions of the form $A \rightarrow M$, where $A \in V_N$, $M \in F$, $M \subseteq (V_N \cup V_T)^*$. (G is sometimes called a "relative grammar").

4.3 Definition. $x \Rightarrow y$ means there exists x_1, x_2, A, ω such that

$$x = x_1 A x_2$$

$$y = x_1 \omega x_2$$

and there is a production $A \rightarrow M$ with $\omega \in M$.

We ignore temporarily the problem of assigning a probability to $x \Rightarrow y$, given probabilities for the productions.

4.4 Definition. For any language $L \subseteq \Sigma^*$ the set $\psi(L)$ is called the (Σ^-) character of L . Here ψ is the Parikh mapping. The character of a family is the collection of characters of its language.

The following can be considered as a corollary to the main theorem of van Leeuwen.

4.5 Parikh's Theorem. The family of context-free languages has the same character as the family of regular languages.

This is generalized by van Leeuwen. He replaces the phrase "family of context-free languages" by an arbitrary family, F , closed under certain operations. The phrase "family of regular languages" is replaced by "algebraic extension of F ." See van Leeuwen (1974) for details. The main ideas in van Leeuwen's proof can be extracted and stated as an algorithm.

4.6 Definition. Let $L \subseteq \Sigma^*$, $A \in \Sigma$. Define the operators c_A , d_A , e_A as follows:

$$c_A(L) = \{w \in L \mid A \text{ occurs in } w\}$$

$$d_A(L) = \{w \in L \mid A \text{ does not occur in } w\}$$

$$e_A(L) = \{wz \mid wAz \in L\}.$$

The following notation is due to Gruska (1971).

4.7 Definition. If L_1, L_2 are languages and A denotes any symbol, then $L_1 \overset{A}{\uparrow} L_2$ denotes the result of substituting all words of L_2 for occurrences of A in L_1 .

4.8 Algorithm.

Input: A context-free language $L = L(G)$, where $G = (V_N, V_T, P, S)$.

Output: A letter equivalent regular language, R .

Method: Assume $V_N = \{A_1, A_2, \dots, A_n\}$, with $A_1 = S$.

$G_1 \leftarrow G; P^{(1)} \leftarrow P; V_1 \leftarrow V_N \cup V_T;$

if $n > 1$ then do

for $i \leftarrow 2$ to n do

$V_i \leftarrow V_{i-1} - \{A_i\}$

$P_i \leftarrow \{A_i \rightarrow \alpha \mid A_i \rightarrow \alpha \in P^{(i-1)}\}$

$G_i \leftarrow (\{A_i\}, V_i, P_i, A_i)$

$L_i \leftarrow L(G_i)$

$P^{(i)} \leftarrow \{A \rightarrow \alpha \stackrel{A_i}{\uparrow} L_i \mid A \neq A_i, A \rightarrow \alpha \in P^{(i-1)}\}$

end;

end;

Let L_S be the language generated by all productions of the form:

$T \rightarrow e_S(c_S(\alpha))T$

$T \rightarrow c_S(\alpha)$, where $S \rightarrow \alpha \in P^{(n)}$.

Let R be $\bigcup_{\alpha} d_S(\alpha) \cup (L_S \stackrel{S}{\uparrow} (\bigcup_{\alpha} d_S(\alpha)))$. Then R is a regular language and $R \models L$.

A simple example might prove enlightening.

4.9 Example. $L = \{a^n b^n \mid n \geq 1\}$, $G = (\{S, B\}, \{a, b\}, P, S)$, where $P = P^{(1)}$ is

$S \rightarrow aSB$

$S \rightarrow aB$

$B \rightarrow b$

Then we have $V_1 = \{a, b, S, B\}$, $A_1 = S$, $A_2 = B$

$V_2 = \{a, b, S\}$

$P_2 = \{B \rightarrow b\}$

$G_2 = (\{B\}, \{a, b, S\}, P_2, B)$

$$\begin{aligned}
 L_2 &= \{b\} \\
 p^{(2)} &= \{S \rightarrow aSB \overset{B}{\uparrow} L_2, S \rightarrow aB \overset{B}{\uparrow} L_2\} \\
 &= \{S \rightarrow aSb, S \rightarrow ab\}.
 \end{aligned}$$

We consider the productions:

$$T \rightarrow abT$$

$$T \rightarrow aSb.$$

Note $e_S(c_S(aSb)) = e_S(aSb) = ab$. Then $L_S = (ab)^* aSb$. Now $d_S(ab) = ab$.

So $R = ab + (ab)^* aSb \overset{S}{\uparrow} \{ab\} = ab + (ab)^* a^2 b^2$. By inspection, $R \leq L$.

Now suppose we begin with a p -CFL. To what extent can the probabilities be carried along in Algorithm 4.8? First, we consider Example 4.9 with probabilities attached to the productions:

$$p : S \rightarrow aSB$$

$$q : S \rightarrow aB$$

$$1 : B \rightarrow b.$$

These can be carried through to the calculation of $p^{(2)}$. In general, how to proceed from there is a problem. In order to avoid the problem mentioned after Definition 4.3 and to have a chance of assigning probabilities at this point, we make the following assumptions:

1. each L_i is a finite set,
2. if $A \rightarrow \alpha \in p^{(i-1)}$ then α contains only one occurrence of

$A_i, i \geq 2$.

This insures that the productions obtained for $p^{(n)}$ are of the form

$S \rightarrow \beta_1 + \beta_2 + \dots + \beta_k$, which might be broken up into

$S \rightarrow \beta_1, S \rightarrow \beta_2, \dots, S \rightarrow \beta_k$. In Example 4.9, the natural choices

for productions generating R are

$$p : T \rightarrow abT$$

$$p : T \rightarrow aSb$$

$$l : S \rightarrow ab$$

$$q : T \rightarrow ab$$

or

$$p : T \rightarrow abT$$

$$p : T \rightarrow a^2b^2$$

$$q : T \rightarrow ab.$$

Unfortunately, this grammar does not generate R . There seems to be no way to proceed. Using various ad hoc methods to assign probabilities, it can be seen that $R \neq L$: Let $G = (\{S, A, B\}, \{a, b\}, P, S)$, where P is

$$l : S \rightarrow aA$$

$$p^2 : A \rightarrow baB$$

$$q : A \rightarrow b$$

$$pq : A \rightarrow ab^2$$

$$p : B \rightarrow baB$$

$$q : B \rightarrow ab^2.$$

Then G is proper and $L(G) = ab + (ab)a^2b^2 = R$. Also, it is not difficult to prove that $R \neq L$.

5. Some Other Ideas

There have been several other approaches to proving Parikh's theorem. We briefly discuss some of these in this section.

Greibach (1972) uses operations on languages to prove a generalization of Parikh's result. She studies families of languages with the "semilinear property": each language in the family has a semilinear image (see Definition 1.6). Her results are based on Gruska's characterization of context-free languages and are phrased in terms of nested iterated substitution. As a consequence, it is difficult to see where probabilities might be incorporated.

Goldstine (1977) offers a "simplified proof" of Parikh's theorem using only standard techniques from the study of context-free languages. His proof is a consequence of a strengthened form of the classical pumping lemma. The proof is essentially a neat and clear rephrasing of Parikh's original proof. Hence it is unlikely that his results can be extended to a probabilistic version.

Shamir (1968) interprets context-free languages as power series in non-commuting variables. In this context he gives a representation theorem for context-free languages in terms of "Dyck-like" languages. Parikh's theorem is a corollary to this result. However, according to Goldstine (1977), the proof of this corollary is "faulty."

6. Conclusions

Our definition of ple (2.5) seems to be a natural probabilistic extension of "letter equivalence." However, with this definition, Parikh's Theorem cannot be probabilistically extended (3.6). There is no obvious and useful way of altering the definition of ple. Hence, to extend the results of Parker and McCluskey, the methods we give in section IV are more useful in most cases. The approach taken in this section does not always work.

LIST OF REFERENCES

LIST OF REFERENCES

1. Athreya, K. B. and Ney, P. E. (1972). Branching Processes. Springer-Verlag, New York.
2. Booth, T. L. and Thompson, R. A. (1973). "Applying Probability Measures to Abstract Languages," IEEE Trans. Comput., vol. C-22, no. 4, pp. 442-450.
3. Conway, J. H. (1971). Regular Algebra and Finite Machines. Chapman and Hall Ltd., London.
4. Fu, K. S. (1974). Syntactic Methods in Pattern Recognition. Academic Press, New York.
5. Fu, K. S. and Booth, T. L. (1975). "Grammatical Inference: Introduction and Survey - Part II," IEEE Trans. Syst., Man, Cyb., vol. SMC-5, no. 4, pp. 409-423.
6. Ginsburg, Seymour (1966). The Mathematical Theory of Context Free Languages. McGraw-Hill, New York.
7. Goldstine, J. (1977). "A Simplified Proof of Parikh's Theorem," Discrete Math., vol. 19, pp. 235-239.
8. Gonzalez, R. C. and Thomason, M. G. (1978). Syntactic Pattern Recognition, An Introduction. Addison-Wesley, Reading, MA.
9. Greibach, S. A. (1972). "A Generalization of Parikh's Semilinear Theorem," Discrete Math., vol. 2, pp. 347-355.
10. Gruska, J. (1971). "A Characterization of Context-Free Languages," J. Comp. Syst. Sci., vol. 5, pp. 353-364.
11. Harris, T. E. (1963). The Theory of Branching Processes. Springer-Verlag, New York.
12. Harrison, M. A. (1978). Introduction to Formal Language Theory. Addison-Wesley, Reading, Massachusetts.
13. Householder, A. S. (1964). The Theory of Matrices in Numerical Analysis. Blaisdell Pub. Co., New York.
14. Hutchins, S. E. (1972). "Moments of String and Derivation Lengths of Stochastic Context-Free Grammars," Inform. Sci., vol. 4, pp. 179-191.
15. Karlin, S. and Taylor, H. M. (1975). A First Course in Stochastic Processes (second edition). Academic Press, New York.

16. Knast, R. (1972). "Finite-State Probabilistic Languages," Inform. Contr., vol. 21, pp. 148-170.
17. Knuth, D. E. (1969). The Art of Computer Programming, vol. 1, Addison-Wesley, Reading, Massachusetts.
18. McWhirter, I. P. (1971). "Substitution Expressions," J. Comp. Syst. Sci., vol. 5, pp. 629-637.
19. Mendenhall, W. and Scheaffer, R. L. (1973). Mathematical Statistics with Applications. Wadsworth, Belmont, California.
20. Mode, C. J. (1971). Multitype Branching Processes, Theory and Applications. American Elsevier Pub. Co., New York.
21. Parikh, R. J. (1966). "On Context-Free Languages," J.A.C.M., vol. 13, no. 4, pp. 570-581.
22. Parker, K. P. and McCluskey, E. J. (1978). "Sequential Circuit Output Probabilities from Regular Expressions," IEEE Trans. Comput., vol. C-27, no. 3, pp. 222-231.
23. Paz, A. (1971). Introduction to Probabilistic Automata. Academic Press, New York.
24. Peleg, S. (1979). "Maximal Derivations for Probabilistic Strings in Stochastic Languages," Inform. Contr., vol. 42, pp. 290-304.
25. Pilling, D. L. (1973). "Commutative Regular Equations and Parikh's Theorem," J. London Math. Soc. (2), vol. 6, pp. 663-666.
26. Rabin, M. O. (1963). "Probabilistic Automata," Inform. Contr., vol. 6, no. 3, pp. 230-245.
27. Salomaa, A. (1969). "Probabilistic and Weighted Grammars," Inform. Contr., vol. 15, pp. 529-544.
28. Salomaa, A. (1973). Formal Languages. Academic Press, New York.
29. Sankoff, D. (1971). "Branching Processes with Terminal Types: Application to Context-Free Grammars," J. Appl. Prob., vol. 8, pp. 233-240.
30. Santos, E. S. (1972). "Probabilistic Grammars and Automata," Inform. Contr., vol. 21, pp. 27-47.
31. Santos, E. S. (1973). "Regular Probabilistic Languages," Inform. Contr., vol. 23, pp. 58-70.
32. Shamir, E. (1968). "Algebraic, Rational, and Context-Free Power Series in Noncommuting Variables," in Algebraic Theory of Machines, Languages, and Semigroups (Michael A. Arbib, ed.), pp. 329-341. Academic Press, New York.

33. Takaoka, Tadao (1978). "A Definition of Measures Over Language Space," J. Comp. Syst. Sci., vol. 17, no. 3, pp. 376-387.
34. Thompson, R. A. (1974). "Determination of Probabilistic Grammars for Functionally Specified Probability-Measure Languages," IEEE Trans. Comput., vol. C-23, no. 6, pp. 603-614.
35. Van Leeuwen, J. (1974). "A Generalization of Parikh's Theorem in Formal Language Theory," in Automata, Languages, and Programming (J. Loeckx, ed.), pp. 17-26. Springer-Verlag, New York.
36. Wielant, H. (1967). Topics in the Analytic Theory of Matrices. Dept. Math., Univ. Wisconsin, Madison, Wisconsin.
37. Yntema, M. K. (1971). "Cap Expressions for Context-Free Languages," Inform. Contr., vol. 18, pp. 311-318.

VITA

Richard H. Sady was born in Springfield, Massachusetts on October 2, 1949. He attended the College of the Holy Cross in Worcester, Massachusetts, and in June, 1971 received a Bachelor of Arts degree in mathematics. In the fall of 1971 he accepted a teaching assistantship at the University of Notre Dame. He studied there for five years and, in June, 1976, he received a Ph.D. degree in mathematics.

In September 1976, the author accepted a position at Pembroke State University, a campus of the University of North Carolina. He was an Assistant Professor of Mathematics until June, 1979, when he resigned from that position. In September, 1979 he entered The University of Tennessee to begin studies in computer science. He received a Master of Science degree in December, 1981.

The author is a member of the American Mathematical Society, the Mathematical Association of America, the American Association for the Advancement of Science, and the Association for Computing Machinery. He is presently employed by The University of Tennessee as an Assistant Professor of Computer Science.