To illustrate the results of the computation, the well-formed words for $n = 5$ are:

111100000	110011000
111010000	110010100
111001000	101110000
111000100	101101000
110110000	101100100
110101000	101011000
110100100	101010100

The total number of well-formed words, for each $n = 2, 3, \ldots, 10$, is:

n	well-formed words
2	1
3	2
4	5
5	14
6	42
7	132
8	429
9	1430
10	4862

The computer program for generation, selection, and printing of these words is relatively short; it can be held entirely within the computing store of the Pegasus while the computation is taking place. As seen in the table above, the number of words becomes rather large when the number of components is greater than five. Because, in ordinary practice, mixtures of greater than five or six components are seldom encountered, it did not seem worth while to reproduce the full tables here; should any reader desire them, the tables or the program can be obtained from the authors.

APPENDIX

The Number of Nets

by E. S. Page

Let the number of nets defined in the preceding paper (Bernard and Mole, 1958) for a mixture of n components be u_n. If the first partition divides the mixture into k and $n - k$ components the number of nets with this first partition is $u_k u_{n-k}$. Hence

$$u_n = \sum_{k=1}^{n-1} u_k u_{n-k}. \tag{1}$$

Define the generating function $U(x) = \sum_{k=1}^{\infty} u_k x^k$; then

$$U^2(x) - U(x) + u_1 x = 0.$$

Since $u_1 = 1$, $u_2 = 1$, $U(x) = [1 - (1 - 4x)^{1/2}]/2$.

Hence
$$u_n = \frac{(2n-2)!}{n!(n-1)!}. \tag{2}$$

Stirling's formula for $n!$ gives

$$u_n \sim 2^{2n-2}/(\pi n^3)^{1/2}, \tag{3}$$

which shows the rate of increase of n_n as n increases.

This derivation of the number of nets suggests a method of enumeration different from that given by Bernard and Mole. The storage required limits its utility for a computer.

On Taking the Square Root of a Complex Number

The square root of a complex number provides an interesting example of the concealed dangers in using algebraic coding. If $(u + iv)^2 = x + iy$ the ordinary formulae for u and v are

$$u = \sqrt{\{\tfrac{1}{2}[\sqrt{(x^2 + y^2)} + x]\}}$$
$$v = \sqrt{\{\tfrac{1}{2}[\sqrt{(x^2 + y^2)} - x]\}}.$$

Any attempt to code these formulae using single-length floating arithmetic such as that normally used by automatic coding routines will lead to serious error whenever y is small. This is caused by the cancellation which occurs in one or other of the expressions $\sqrt{(x^2 + y^2)} \pm x$. If the error of the inner square root is ε, the error in the final result can be as much as $\sqrt{(\frac{1}{2}\varepsilon)}$. Thus, to get single-length accuracy from these formulae, it is not enough to find the single-length square root of a double-length number; the inner square root must itself be found to double-length accuracy.

Once this danger has been appreciated there is, of course, no real difficulty in getting round it. With some machines it is not difficult to find the inner square root to double-length accuracy, and this allows a very straightforward program. On others it may be more convenient to evaluate only one of u and v from the formula (the one in which no cancellation takes place) and to find the other from the equation $2uv = y$.

C. Strachey