Short Notes

The Relationship Between Two Fast Fourier Transforms

I. J. GOOD

Abstract—The purpose of this note is to show as clearly as possible the mathematical relationship between the two basic fast methods used for the calculation of discrete Fourier transforms and to generalize one of the methods a little further. This method applies to all those linear transformations whose matrices are expressible as direct products.

Index Terms—Algorithms, circulices, direct product of matrices, discrete Fourier transforms, fast Fourier transforms, frequency analysis, Hadamard transform, harmonic analysis, multidimensional linear transformation, sparse matrices.

A fast Fourier transform (FFT) is a fast method for the calculation of a discrete Fourier transform (DFT) whose definition we shall give below in order to make the paper adequately self-contained. For applications of FFTs and DFTs see, for example, [1]–[4]. See also the bibliographies [5], [6], [4]. Here we shall be concerned only with mathematical and computational aspects. The main purpose of this note is to show as clearly as possible the relationship between the two basic algorithms for FFTs [7], [8]. Other methods are refinements of these two. Each method has its own advantages; one method can be applied to a wider variety of moduli, the other is more appropriate for a wide class of multidimensional transforms, not necessarily Fourier.

A one-dimensional mod t DFT of a vector (a sequence of t numbers $a_0, a_1, \cdots, a_{t-1}$ assumed to be real in this paper) is another vector of t complex numbers $a_0^*, a_1^*, \cdots, a_{t-1}^*$ defined by the equations

$$a_s^* = \sum_{s=0}^{t-1} a_r \omega^{rs} \qquad (s = 0, 1, \cdots, t-1) \tag{1}$$

where $\omega = \exp(2\pi i/t)$, a tth root of unity. It has elegant analogs of the properties of an ordinary Fourier transform like an inversion formula, a formula for the transform of a convolution, and even a Poisson summation formula (for example, [9]–[11]). A *multidimensional* mod $(t_1, t_2, \cdots, t_n)$ DFT of the n-dimensional array of $t_1 t_2 \cdots t_n$ numbers a_r, where $\boldsymbol{r}$ is the vector $(r_1, r_2, \cdots, r_n)$ $(r_j = 0, 1, \cdots, t_j - 1;$ $j = 1, 2, \cdots, n)$, is the n-dimensional array of complex numbers a_s^* defined by the equations

$$a_s^* = \sum_{\boldsymbol{r}} a_r \omega_1^{r_1 s_1} \cdots \omega_n^{r_n s_n}$$

$$(r_j, s_j = 0, 1, \cdots, t_j - 1; j = 1, \cdots, n) \tag{2}$$

where $\omega_j = \exp(2\pi i/t_j)$ $(j = 1, \cdots, n)$. The arrays a_r and a_s^* can also be regarded as vectors if a rule is given for ordering the $\tau = t_1 t_2 \cdots t_n$ elements of each as sequences or "strings." When all the t_j's are equal to t, we can refer to the n-dimensional mod t DFT. The components of a multidimensional mod 2 DFT (of an array of real number a_r) are all real because in this case all the ω_j's are equal to -1. The mod 2 multidimensional DFT is also an example of a Hadamard transform [12]. The matrix of a Hadamard transform is orthogonal and each of its elements is $+1$ or -1. It is also known as an "anallagmatic pavement" [13].

A DFT, unidimensional or multidimensional, is, of course, a linear transformation of a vector (with t or $t_1 t_2 \cdots t_n$ components), a multiplication of a vector by a matrix, in which the matrix of the transformation is (ω^{rs}) or $(\omega_1^{r_1 s_1} \cdots \omega_n^{r_n s_n})$. In the latter matrix some convention is required for the ordering of the rows and columns. The simplest convention is to order the $t_1 t_2 \cdots t_n$ vectors $\boldsymbol{r} = (r_1, \cdots, r_n)$ lexicographically as if they were words in a dictionary, and similarly for the "columns" $\boldsymbol{s} = (s_1, \cdots, s_n)$. Then the components of a_r must be similarly ordered in order to convert it from an array into a vector.

The first FFT to be described here depends on the fact that a one-dimensional DFT modulo τ, where $\tau = t_1 t_2 \cdots t_n$, and $t_1, t_2, \cdots, t_n$ are mutually prime in pairs, can be expressed as a mod $(t_1, t_2, \cdots, t_n)$ multidimensional DFT. For the sake of completeness the procedure is quoted from Good [7].[1] We first set up (1–1) correspondences between the scalar r and the vector $\boldsymbol{r}$ and also between s and $\boldsymbol{s}$ in the following manner.

We begin by recalling the Chinese Remainder Theorem, known by Sun-Tsu in the first century A.D. [14], [15]. It states that if an integer is known modulo n integers (that is, its remainder is known when it is divided by each of these n integers), these integers being mutually prime in pairs (that is, each pair has highest common factor equal to 1), then it can be uniquely determined modulo the product. The solution can be expressed succinctly [9], [7] as follows. If $t_1, \cdots, t_n$ are mutually prime in pairs and $\tau = t_1 \cdots t_n$, and if the remainders are $s_1, \cdots, s_n$, or in the standard "congruence" notation, if

$$s \equiv s_1 (\bmod t_1), \cdots, s \equiv s_n (\bmod t_n),$$

then

$$s \equiv \frac{\tau s_1}{t_1}\left(\frac{\tau}{t_1}\right)_{t_1}^{-1} + \cdots + \frac{\tau s_n}{t_n}\left(\frac{\tau}{t_n}\right)_{t_n}^{-1} \pmod{\tau; 0 \le s < \tau; 0 \le s_1 < t_1, \text{etc.}} \tag{3}$$

where $(\tau/t_1)_{t_1}^{-1}$, for example, means the modulo t_1 reciprocal of the integer τ/t_1. This reciprocal exists and is unique modulo t_1 because the integers τ/t_1 and t_1 are mutually

Manuscript received December 18, 1969; revised August 21, 1970.

The author is with the Department of Statistics and Statistical Laboratory, Virginia Polytechnic Institute and State University, Blacksburg, Va. 24061.

[1] There are misprints in [7]. Four incorrect entries in the matrix at the foot of p. 363 have been found. On p. 364, line 10, $\delta_{r_{v-1}}$ should be $\delta_{r_{v+1}}$. The lower limit of the product at the foot of p. 365 should be 1. On p. 368, line 13, an asterisk was omitted.

prime. Hence the first term of this expression for s is unique modulo τ, and similarly the other terms are unique. The reader can verify at once that the value of s does satisfy the n given congruences. Moreover, each of the τ possible sets of n congruences has at least one solution given by (3). These solutions must be unique, since there are only τ residues modulo τ and none of them can correspond to more than one set of n congruences. Formula (3) is analogous to Lagrange's interpolation formula. We use it to set up a (1–1) correspondence between s and $\boldsymbol{s}=(s_1, \cdots, s_n)$ which I shall call the *Sino correspondence* or the *Sino representation* of s. For example, if $t_1=3$, $t_2=8$, $\tau=24$, then $s\equiv 16s_1+9s_2 \pmod{24}$. Once we fix $t_1, \cdots, t_n$ the coefficients (here 16 and 9) are known once for all. On the other hand, the (1–1) relationship between r and $\boldsymbol{r}$ is given by

$$r \equiv \frac{\tau}{t_1} r_1 + \cdots + \frac{\tau}{t_n} r_n \pmod{\tau, 0 \le r < \tau; 0 \le r_1 < t_1, \text{etc.}} \quad (4)$$

which I shall call the *Ruritanian correspondence*. Again, if $t_1=3$, $t_2=8$, $\tau=24$, this gives $r\equiv 8r_1+3r_2 \pmod{24}$.

I now assert that the equations

$$\hat{a}_s^* = \sum_{r=0}^{\tau-1} \hat{a}_r \omega^{rs} \qquad (\omega = e^{2\pi i/\tau})$$

and

$$a_s^* = \sum_{r_1=0}^{t_1-1} \cdots \sum_{r_n=0}^{t_n-1} a_r \omega_1^{r_1 s_1} \cdots \omega_n^{r_n s_n}$$

are equivalent, where $\hat{a}_r = a_r$ and $\hat{a}_s^* = a_s^*$ by definition, and where the correspondences are Sino and Ruritanian, respectively. To see this, note that

$$r \equiv (\tau/t_v) r_v \pmod{t_v}, \quad s \equiv \frac{\tau s_v}{t_v} \left(\frac{\tau}{t_v}\right)_{t_v}^{-1} \pmod{t_v}$$

so that $rs \equiv r_v s_v \tau/t_v \pmod{t_v}$ and hence, by Sun-Tsu's theorem in the form given above,

$$rs \equiv \frac{\tau r_1 s_1}{t_1} \cdot \frac{\tau}{t_1} \left(\frac{\tau}{t_1}\right)_{t_1}^{-1} + \cdots \equiv \sum_v r_v s_v \tau/t_v.$$

Moreover,

$$\omega_v = \omega^{\tau/t_v}$$

so

$$\hat{a}_s^* = \sum_r \hat{a}_r \omega_1^{r_1 s_1} \cdots \omega_n^{r_n s_n} = a_s^*,$$

as claimed.

A unidimensional mod $t_1 t_2 \cdots t_n$ DFT can therefore be computed with about the same speed as a multidimensional mod $(t_1, \cdots, t_n)$ DFT where $t_1, \cdots, t_n$ are mutually prime in pairs. Accordingly we now consider the calculation of a multidimensional DFT, or more generally, a linear transformation from a_r to a_s^* of the form

$$a_s^* = \sum_{r_1=0}^{t_1-1} \cdots \sum_{r_n=0}^{t_n-1} a_r K_1(r_1, s_1) \cdots K_n(r_n, s_n) \quad (5)$$

where we have replaced $\omega_1^{r_1 s_1}$ by $K_1(r_1, s_1)$, etc., and where we do not insist for the present on any constraints on $t_1, \cdots, t_n$. (The following discussion of (5) and (6) was given in [4].) We shall describe (5) as an *n-dimensional K-transform*. Although each of the functions $K_1, K_2, \cdots, K_n$ can be regarded as a square matrix, the right side of (5) is, of course, not as it stands in the form of the product of a vector by several matrices, although it can be regarded as the product of a vector by one large matrix whose rows are indexed by $\boldsymbol{r}$ and columns by $\boldsymbol{s}$. This large matrix will later be seen to be the "direct product" of the smaller matrices, and also to be the product of large matrices which correspond in turn to K_1, $K_2, \cdots, K_n$. For the moment we shall point out a simple procedure for summing (5) without matrix notation.

The summation can be performed by first summing over r_1, then over r_2, etc. The first summation is

$$\sum_{r_1} a_r K_1(r_1, s_1)$$

and the result is of the form $L_1(r_2, r_3, \cdots, r_n, s_1)$, that is, a function of $r_2, r_3, \cdots, r_n, s_1$. The next stage involves the sum

$$\sum_{r_2} L_1(r_2, r_3, \cdots, r_n, s_1) K_2(r_2, s_2)$$

and is of the form $L_2(r_3, \cdots, r_n, s_1, s_2)$, and so on. Thus at each stage we need store only an n-dimensional array, and altogether we need do only $(t_1+t_2+\cdots+t_n)\tau$ scalar multiplications. This is the basis of the method of [7] for forming a multidimensional DFT, except that it was there expressed in matrix notation. The method saves arithmetic as compared with the multiplication of the vector by the single large matrix mentioned above which would require τ^2 scalar multiplications. For example, with $\tau=5.7.8.9=2520$, the number of scalar multiplications is less than an eightieth of what would be required ($\tau^2=2520^2$) if a_r were multiplied by the $\tau \times \tau$ matrix $K_1(r_1, s_1) \cdots K_n(r_n, s_n)$.

We shall later discuss the inverse of the multidimensional K-transform, but first we consider a generalization of the above simple summation procedure which leads up to the second method for calculating an FFT.

We consider the analogous multiple sum

$$\sum_{r_1=0}^{t_1-1} \cdots \sum_{r_n=0}^{t_n-1} a_r K_1(r_1, s_1) K_2(r_2, s_1, s_2) \cdots K_n(r_n, s_1, s_2, \cdots, s_n) \quad (6)$$

and we find that the above argument goes through virtually unchanged for this more general summation. The summation of $a_r K_1$ with respect to r_1 leads to an expression of the form $L_1(r_2, \cdots, r_n, s_1)$; then the summation of $L_1 K_2$ with respect to r_2 leads to an expression of the form $L_2(r_3, \cdots, r_n, s_1, s_2), \cdots$, and finally the summation of $L_{n-1} K_n$ with respect to r_n to an expression of the form $L_n(s_1, s_2, \cdots, s_n)$. At each stage only an n-dimensional array needs to be stored, but the dimensions of the array vary from stage to stage unless the t_v's are all equal.

The method used by Cooley and Tukey [8] for computing a one-dimensional DFT is in effect to reduce it to the form (6). Their method applies to a modulus $\tau=t_1 t_2 \cdots t_{n-1}$, but it was first spelled out for the case $\tau=2^n$, that is $t_1=t_2=\cdots$

$=t_n=2$. The general case was explained in more detail by Bingham, Godfrey, and Tukey [16]. In this method the correspondence between the one-dimensional DFT and a multiple sum depends on setting up (1–1) correspondences between r and $\boldsymbol{r}$ and between s and $\boldsymbol{s}$ different from the Sino–Ruritanian correspondences or representations. Instead r and s are each represented in mixed radix form, but each with a different kind of reversal, as follows.

$$r = t_1t_2\cdots t_{n-1}r_1 + t_1t_2\cdots t_{n-2}r_2 + \cdots + t_1r_{n-1} + r_n$$
$$s = t_2t_3\cdots t_ns_n + t_3t_4\cdots t_ns_{n-1} + \cdots + t_ns_2 + s_1. \quad (7)$$

If we think of t_1 as analogous to 10, t_1t_2 to 100 and so on, we see that the representation of r is mixed radical, but with the *components* of $\boldsymbol{r}$ in the reverse order to the *digits* of r. We could call this the *reversed-digit* (mixed-radix) representation. On the other hand, the representation of s reverses the order of the radices $t_1, t_2, \cdots, t_n$ but leaves the components of $\boldsymbol{s}$ in the same order as the digits of s. Thus there are two different kinds of reversal so the representation of r and s together could be described as the *reversed-radical* or *reactionary* representation or correspondence. There are variants of the method, depending on various mixed-radix representations of integers, which, of course, are not mixed when $t_1=t_2=\cdots=t_n$.

The single summation

$$\sum_r a_r\omega^{rs} \qquad (\omega = e^{2\pi i/\tau})$$

can be regarded as a multiple summation over the digits of r in its mixed-radix representation. We also substitute the mixed-radix representation of s and obtain

$$\sum_{\boldsymbol{r}} a_{\boldsymbol{r}}\omega^{(t_1t_2\cdots t_{n-1}r_1+t_1t_2\cdots t_{n-2}r_2+\cdots+t_1r_{n-1}+r_n)(t_2\cdots t_ns_n+\cdots+t_ns_2+s_1)} = \sum_{\boldsymbol{r}} a_{\boldsymbol{r}}K_1(r_1,s_1)K_2(r_2,s_1,s_2)\cdots K_n(r_n,s_1,s_2,\cdots,s_n) \quad (8)$$

where (since $\omega^\tau=1$)

$$K_1(r_1,s_1) = \omega^{t_1t_2\cdots t_{n-1}r_1s_1}$$
$$K_2(r_2,s_1,s_2) = \omega^{t_1t_2\cdots t_{n-2}r_2(t_ns_2+s_1)}$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$K_n(r_n,s_1,\cdots,s_n) = \omega^{r_n(t_2\cdots t_ns_n+\cdots+t_ns_2+s_1)}. \quad (9)$$

If the elements of a_r are now permuted in accordance with the mixed-radical representation of r, the right side of (8) is of the form of (6) and can be computed by the simple procedure described above. Note that it would not be strictly correct to write $a_{\boldsymbol{r}}$ in place of a_r in the summation because we would then be using the symbol a to denote two different functions.

For this Cooley–Tukey or mixed-radix method the transformation from a unidimensional DFT to the multiple sum does not require that $t_1, t_2, \cdots, t_n$ be mutually prime in pairs. In this respect it has an advantage over the Sino–Ruritanian method, being more widely applicable when we are concerned with DFT's. It is perhaps used most often with the t_ν's all equal to 2, which is the simplest case to program for a binary computer. On the other hand, the multiple sum (5) is simpler than (6) and therefore more likely to occur (as a multidimensional K-transform) in contexts other than the DFT, especially with $K_1, \cdots, K_n$ all the same function. Some such contexts are mentioned, for example, by Andrews and Caspari [17]. Also, even for the DFT itself, if very fast special-purpose equipment is to be built, using fixed values of $t_1, t_2, \cdots, t_n$, the Sino–Ruritanian representation of r and s might turn out to be more convenient than the mixed-radical representation.

In order to discuss the algorithms in matrix notation it is necessary to make use of *direct products* (=*Kronecker products*) of matrices. (See, for example, [18]). The direct product $B\times C$ of two matrices $B=(b_{ij})$ $(i, j=0, 1, \cdots)$ and $C=(c_{\mu\nu})$ $(\mu, \nu=0, 1, \cdots)$, which need not be either square nor of related shapes and sizes, is defined as the block matrix

$$D = \begin{bmatrix} b_{00}C & b_{01}C\cdots \\ b_{10}C & b_{11}C\cdots \\ \cdots\cdots\cdots \end{bmatrix}. \quad (10)$$

this being of course an abbreviated notation for an ordinary larger matrix whose elements are scalars. This definition is equivalent to saying that the elements of $B\times C$ are

$$d_{\boldsymbol{r},\boldsymbol{s}} = b_{r_1,s_1}c_{r_2,s_2} \quad (11)$$

where $\boldsymbol{r}=(r_1, r_2)$ and $\boldsymbol{s}=(s_1, s_2)$ are ordered lexicographically. For example, if B and C are both 2×2,

$$\begin{array}{c|cccc} \boldsymbol{s} & 00 & 01 & 10 & 11 \\ \boldsymbol{r} & & & & \\ 00 & b_{00}c_{00}, & b_{00}c_{01}, & b_{01}c_{00}, & b_{01}c_{01} \\ 01 & b_{00}c_{10}, & b_{00}c_{11}, & b_{01}c_{10}, & b_{01}c_{11} \\ 10 & b_{10}c_{00}, & b_{10}c_{01}, & b_{11}c_{00}, & b_{11}c_{01} \\ 11 & b_{10}c_{10}, & b_{10}c_{11}, & b_{11}c_{10}, & b_{11}c_{11} \end{array} = B\times C. \quad (12)$$

Thus, for example, $b_{11}c_{00}$ corresponds to $r_1=1$, $s_1=1$, $r_2=0$, $s_2=0$. (The reader should now hold in mind Savage's metamathematical principle: he should sit bolt upright in a hard chair with a pencil in his hand [19]). The general definition for the direct product of several matrices is the obvious inductive one, which turns out to be associative, so that parentheses are not required. Moreover, formula (11) generalizes in the natural manner. The direct product of n matrices,

$$M^{(1)}\times M^{(2)}\times\cdots\times M^{(n)}$$

where $M^{(\nu)}=(m_{i,j})$ $(i=0, 1, \cdots, p_\nu-1;\ j=0, 1, \cdots, q_\nu-1)$ has its $(\boldsymbol{r}, \boldsymbol{s})$ element

$$\prod_{\nu=1}^{N} m^{(\nu)}_{r_\nu,s_\nu} \quad (13)$$

wherein the ordering of the rows is lexicographic as is that of the columns. This again is true even if the matrices are not square and are of various shapes and sizes.

If the ordinary matrix product $M^{(\nu)}N^{(\nu)}$ can be formed, that is, if the number of columns of $M^{(\nu)}$ is the same as the number of rows of $N^{(\nu)}$ $(\nu=1, 2, \cdots)$, then

$$(M^{(1)}\times M^{(2)}\times\cdots)(N^{(1)}\times N^{(2)}\times\cdots)$$
$$= (M^{(1)}N^{(1)})\times(M^{(2)}N^{(2)})\times\cdots. \quad (14)$$

In particular, if M is $t \times t$ and is nonsingular, then

$$(M \times M \times \cdots)(M^{-1} \times M^{-1} \times \cdots) = I_t \times I_t \times \cdots = I_{t^n}$$

where I_u is the $u \times u$ identity matrix. More briefly

$$(M^{[n]})^{-1} = (M^{-1})^{[n]} \tag{15}$$

where the symbol $[n]$ indicates the "direct nth power." Now the n-dimensional K-transform, as defined by (5), has as its matrix the direct product of the matrices $K_1, K_2, \cdots, K_n$, where, of course, the matrix K_1 has $K_1(r_1, s_1)$ as its elements, and so on. This is true provided that we order the rows and columns lexicographically. This explains the relevance of direct products of matrices to the previous discussion.

Now it is an interesting fact that a direct product of n matrices can also be expressed as an ordinary product of *sparse* matrices in accordance with the following theorem. (A sparse matrix is one consisting mainly of zeros.)

$$M = M^{(1)} \times M^{(2)} \times \cdots \times M^{(n)} = A^{(1)}A^{(2)} \cdots A^{(n)} \tag{16}$$

where

$$A^{(v)} = (a^{(v)}_{r,s})$$

where

$$a^{(v)}_{r,s} = m^{(v)}_{r_1,s_n}\delta^{s_1}_{r_2}\delta^{s_2}_{r_3} \cdots \delta^{s_{n-1}}_{r_n} \tag{17}$$

$$(0 \le r_1 < p_v, 0 \le r_2 < p_{v+1}, \cdots, 0 \le r_{n-v+1} < p_n,$$
$$0 \le r_{n-v+2} < q_1, \cdots, 0 \le r_n < q_{v-1}$$
$$0 \le s_1 < p_{v+1}, 0 \le s_2 < p_{v+2}, \cdots, 0 \le s_{n-v} < p_n,$$
$$0 \le s_{n-v+1} < q_1, \cdots, 0 \le s_n < q_v).$$

(The "Kronecker delta" is defined by $\delta^b_a = 1$ if $a = b$, otherwise $\delta^b_a = 0$.) Note that the nonzero elements of $A^{(v)}$ are, apart from replications, the same as those of $M^{(v)}$, but its *size* depends on all the t's.

Thus $A^{(1)}$ is a $(p_1p_2 \cdots p_n) \times (p_2p_3 \cdots p_nq_1)$ matrix having at most $p_1p_2 \cdots p_nq_1$ nonzero elements. $A^{(2)}$ is a $(p_2p_3 \cdots p_nq_1) \times (p_3p_4 \cdots p_nq_1q_2)$ matrix having at most $p_2p_3 \cdots p_nq_1q_2$ nonzero elements. $A^{(v)}$ is a $(p_v \cdots p_nq_1 \cdots q_{v-1}) \times (p_{v+1} \cdots p_nq_1 \cdots q_v)$ matrix having most $p_vp_{v+1} \cdots p_nq_1q_2 \cdots q_v$ nonzero elements. $A^{(n)}$ is a $(p_nq_1q_2 \cdots q_{n-1}) \times (q_1q_2 \cdots q_n)$ matrix having at most $p_nq_1q_2 \cdots q_n$ nonzero elements. $A^{(1)} \cdots A^{(n)}$ is a $(p_1p_2 \cdots p_n) \times (q_1q_2 \cdots q_n)$ matrix, namely M.

The product $\boldsymbol{a}'M$, where the components of $\boldsymbol{a}$ are $a_r (0 \le r_1 < p_1, \cdots, 0 \le r_n < p_n)$ can thus be effected as

$$(\cdots((\boldsymbol{a}'A^{(1)})A^{(2)}) \cdots A^{(n)}). \tag{18}$$

As an example, take $n=2, p_1=1, q_1=2, p_2=3, q_2=4$, and note that

$$\begin{array}{c|cccccc|cccccccc|c}
p_2q_1 & 00 & 01 & 10 & 11 & 20 & 21 & 00 & 01 & 02 & 03 & 10 & 12 & 13 & 14 & q_1q_2 \\
p_1p_2 & & & & & & & & & & & & & & & p_2q_1 \\
00 & a_{00} & a_{01} & 0 & 0 & 0 & 0 & b_{00} & b_{01} & b_{02} & b_{03} & 0 & 0 & 0 & 0 & 00 \\
01 & 0 & 0 & a_{00} & a_{01} & 0 & 0 & 0 & 0 & 0 & 0 & b_{00} & b_{01} & b_{02} & b_{03} & 01 \\
02 & 0 & 0 & 0 & 0 & a_{00} & a_{01} & b_{10} & b_{11} & b_{12} & b_{13} & 0 & 0 & 0 & 0 & 10 \\
 & & & & & & & 0 & 0 & 0 & 0 & b_{10} & b_{11} & b_{12} & b_{13} & 11 \\
 & & & & & & & b_{20} & b_{21} & b_{22} & b_{23} & 0 & 0 & 0 & 0 & 20 \\
 & & & & & & & 0 & 0 & 0 & 0 & b_{20} & b_{21} & b_{22} & b_{23} & 21
\end{array}$$

$$= \begin{bmatrix}
a_{00}b_{00} & a_{00}b_{01} & a_{00}b_{02} & a_{00}b_{03} & a_{01}b_{00} & a_{01}b_{01} & a_{01}b_{02} & a_{01}b_{03} \\
a_{00}b_{10} & a_{00}b_{11} & a_{00}b_{12} & a_{00}b_{13} & a_{01}b_{10} & a_{01}b_{11} & a_{01}b_{12} & a_{01}b_{13} \\
a_{00}b_{20} & a_{00}b_{21} & a_{00}b_{22} & a_{00}b_{23} & a_{01}b_{20} & a_{01}b_{21} & a_{01}b_{22} & a_{01}b_{23}
\end{bmatrix}$$

$$= A \times B.$$

In this example, if we were to multiply a horizontal 3-vector or a bolt-upright 8-vector by $A \times B$ it would involve $3 \times 8 \times 8 = 192$ scalar products, whereas multiplying first by $A^{(1)}$ and then by $A^{(2)}$, or vice versa, requires only $4 \times 8 + 3 \times 2 = 38$ scalar products. Generally, the reduction is from

$$p_1p_2 \cdots p_nq_1q_2 \cdots q_n$$

to

$$\sum_v p_vp_{v+1} \cdots p_nq_1q_2 \cdots q_v \tag{19}$$

scalar multiplications.

The identification of this procedure with that in (5), when $p_v = q_v = t_v$ $(v = 1, 2, \cdots, n)$, is exemplified by the following algebra for the case $t_1 = 2$, $t_2 = 3$.

$$\begin{array}{c|cccccc|c}
 & 00 & 01 & 10 & 11 & 20 & 21 & t_2t_1 \\
 & & & & & & & t_1t_2 \\
[a_{00}, a_{01}, a_{02}, a_{10}, a_{11}, a_{12}] & c_{00} & c_{01} & 0 & 0 & 0 & 0 & 00 \\
 & 0 & 0 & c_{00} & c_{01} & 0 & 0 & 01 \\
 & 0 & 0 & 0 & 0 & c_{00} & c_{01} & 02 \\
 & c_{10} & c_{11} & 0 & 0 & 0 & 0 & 10 \\
 & 0 & 0 & c_{10} & c_{11} & 0 & 0 & 11 \\
 & 0 & 0 & 0 & 0 & c_{10} & c_{11} & 12
\end{array}$$

$$= [a_{00}c_{00} + a_{10}c_{10}, a_{00}c_{01} + a_{10}c_{11}, a_{01}c_{00} + a_{11}c_{10}, a_{01}c_{01} + a_{11}c_{11}, a_{02}c_{00} + a_{12}c_{10}, a_{02}c_{01} + a_{12}c_{11}]$$

whose elements are $\sum_{r_1} a_{r_1,r_2} c_{r_1,s_1} = l_{r_2,s_1}$ just as after the first summation for (5). Further,

$$\begin{array}{c} \quad 00 \quad 01 \quad 02 \quad 10 \quad 11 \quad 12 \quad t_1 t_2 \\ \\ \left[l_{00}, l_{01}, l_{10}, l_{11}, l_{20}, l_{21}\right]\begin{bmatrix} d_{00} & d_{01} & d_{02} & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{00} & d_{01} & d_{02} \\ d_{10} & d_{11} & d_{12} & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{10} & d_{11} & d_{12} \\ d_{20} & d_{21} & d_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{20} & d_{21} & d_{22} \end{bmatrix} \begin{matrix} t_2 t_1 \\ 00 \\ 01 \\ 10 \\ 11 \\ 20 \\ 21 \end{matrix} \end{array}$$

$$= \left[l_{00}d_{00} + l_{10}d_{10} + l_{20}d_{20}, l_{00}d_{01} + l_{10}d_{11} + l_{20}d_{21}, \cdots, l_{01}d_{02} + l_{11}d_{12} + l_{21}d_{22}\right]$$

whose elements are $\sum_{r_2} l_{r_2,s_1} d_{r_2,s_2}$, as in the second summation for (5). The first term in this vector is $(a_{00}c_{00} + a_{10}c_{10})d_{00} + (a_{01}c_{00} + a_{11}c_{10})d_{10} + (a_{02}c_{00} + a_{12}c_{10})d_{20} = \sum_{r_1,r_2} a_{r_1,r_2} c_{r_1,0} d_{r_2,0}$, and similarly the other five terms also agree with (5).

The theorem given by (16) was proven for the case $p_1 = \cdots = p_n = q_1 = \cdots = q_n$ in [7], and the proof extends to the general case.

The inverse of a multidimensional K-transform exists if the matrices $M^{(v)}$, defined by $M^{(v)} = (K_v(r_v, s_v))$, are all square and nonsingular. This can be easily proved by verifying (29), which is given in the Summary and Discussion. We here express the proof in terms of direct products of matrices. If we write $\boldsymbol{a}$ and $\boldsymbol{a}^*$ for the column vectors whose components are $a_{\boldsymbol{r}}$ and $a^*_{\boldsymbol{s}}$, these components being arranged in the lexicographic order of $\boldsymbol{r}$ and $\boldsymbol{s}$, then (5) can be written

$$\boldsymbol{a}^{*\prime} = \boldsymbol{a}'(M^{(1)} \times M^{(2)} \times \cdots \times M^{(n)}) = \boldsymbol{a}'M. \tag{20}$$

Now $M^{(v)}(M^{(v)})^{-1} = I_{t_v}$, and we see readily from (14) that

$$\boldsymbol{a}' = \boldsymbol{a}^*\{(M^{(1)})^{-1} \times \cdots \times (M^{(n)})^{-1}\}. \tag{21}$$

It may be noted that the determinant of M

$$|M| = \Delta_1^{\tau/t_1} \cdots \Delta_n^{\tau/t_n} \tag{22}$$

where $\Delta_v = |M^{(v)}|$. This follows inductively from the case $n=2$ which is given by MacDuffee [18] who references Hensel, Netto, and von Sterneck. In the case where $M^{(v)}$ is independent of v and has order t and determinant Δ, we have

$$|M| = \Delta^{nt^{n-1}}. \tag{23}$$

If we wish to express the inverse multidimensional K-transform by means of ordinary matrix multiplications, we can apply (16) to the inverses of the $M^{(v)}$'s. Alternatively, we can denote the cofactors in $M^{(v)}$ by $M^{(v)}_{i,j}$ $(i,j = 0, 1, \cdots, t_{v-1})$ and write $B^{(v)} = b^{(v)}_{r,s}$ where

$$b^{(v)}_{r,s} = M^{(v)}_{s_1,r_n} \delta^{r_1}_{s_2} \delta^{r_2}_{s_3} \cdots \delta^{r_{n-1}}_{s_n} \tag{24}$$

$(s_1, r_n = 0, 1, \cdots, t_v - 1; s_2, r_1 = 0, 1, \cdots, t_{v+1} - 1; \cdots; s_{n-v+1}, r_{n-v} = 0, 1, \cdots, t_n - 1; \cdots; s_n, r_{n-1} = 0, 1, \cdots, t_{v-1} - 1)$.

Then

$$A^{(v)}B^{(v)} = \Delta_v I_\tau. \tag{25}$$

This is true even if $A^{(v)}$ is singular, in which case the right side of (25) is zero. If $A^{(v)}$ is nonsingular, it follows from (25) that $(A^{(v)})^{-1} = B^{(v)}/\Delta_v$. A proof of (25) is

$$\begin{aligned} \sum_{q} m^{(v)}_{r_1,q_n} \delta^{q_1}_{r_2} \delta^{q_2}_{r_3} \cdots \delta^{q_{n-1}}_{r_n} \cdot M^{(v)}_{s_1,q_1} \delta^{q_1}_{s_2} \delta^{q_2}_{s_3} \cdots \delta^{q_{n-1}}_{s_n} \\ = \sum_{q^n} m^{(v)}_{r_1,q_n} M^{(v)}_{s_1,q_n} \delta^{s_2}_{r_2} \delta^{s_3}_{r_3} \cdots \delta^{s_n}_{r_n} \\ = \Delta_v \delta^{s_1}_{r_1} \delta^{s_2}_{r_2} \cdots \delta^{s_n}_{r_n}. \end{aligned}$$

An example of (25) is

$$\begin{bmatrix} m_{00} & m_{01} & 0 & 0 \\ 0 & 0 & m_{00} & m_{01} \\ m_{10} & m_{11} & 0 & 0 \\ 0 & 0 & m_{10} & m_{11} \end{bmatrix} \begin{bmatrix} M_{00} & 0 & M_{10} & 0 \\ M_{01} & 0 & M_{11} & 0 \\ 0 & M_{00} & 0 & M_{10} \\ 0 & M_{01} & 0 & M_{11} \end{bmatrix} = \begin{bmatrix} \Delta & 0 & 0 & 0 \\ 0 & \Delta & 0 & 0 \\ 0 & 0 & \Delta & 0 \\ 0 & 0 & 0 & \Delta \end{bmatrix}$$

where

$$\Delta = m_{00}m_{11} - m_{01}m_{10}, M_{00} = m_{11}, M_{01} = -m_{10}, M_{10} = -m_{01}, M_{11} = m_{00}.$$

Note that $A^{(v)}$ and its inverse are equally sparse.

The determinants of $A^{(v)}$ can be neatly obtained from (22) by supposing that $\Delta_1 = \cdots = \Delta_{v-1} = \Delta_{v+1} = \cdots \Delta_n = 1$. It follows from this, together with the omitted argument for determining the sign, that

$$|A^{(v)}| = \Delta_v^u (-1)^{u(u-1)/2} \qquad (u = \tau/t_v) \tag{26}$$

and therefore, from (25), or from a familiar property of adjugates [20, p. 165]

$$|B^{(v)}| = \Delta_v^{\tau(\tau_v - 1)/\tau_v} (-1)^{u(u-1)/2}. \tag{27}$$

Of course if all the $M^{(v)}$'s are orthogonal (or else unitary), then their inverses are equal to their (conjugate) transposes,

and the multidimensional K-transform and its inverse involve nearly the same calculation. The matrix of the multidimensional DFT becomes unitary if divided by $\sqrt{\tau}$ (or if each $M^{(v)}$ is divided by $\sqrt{t_v}$), just as the multidimensional Fourier transform is unitary if a suitable normalizing factor is used, such as $1/\sqrt{2\pi}$ per integral. When each $M^{(v)}$ is square, with $p_v = q_v = t_v$ as before, then another way of factorizing $M^{(1)} \times \cdots \times M^{(n)}$ into n sparse matrices is [7]

$$M^{(1)} \times \cdots \times M^{(n)} = C^{(1)} C^{(2)} \cdots C^{(n)} \tag{28}$$

where

$$C^{(1)} = M^{(1)} \times I_{t_2} \times \cdots \times I_{t_n}$$
$$C^{(2)} = I_{t_1} \times M^{(2)} \times \cdots \times I_{t_n}$$
$$C^{(n)} = I_{t_1} \times I_{t_2} \times \cdots \times M^{(n)}.$$

This follows from (14) and might sometimes be more convenient to use than (16) although the patterns of $C^{(1)}, \cdots, C^{(n)}$ are less simple than those of $A^{(1)}, \cdots, A^{(n)}$.

Summary and Discussion

Yates [21] introduced a simple adding and subtracting algorithm for the calculation of the interactions in 2^n-factorial experiments. The fact that Yates' algorithm could be expressed as a multidimensional mod 2 discrete Fourier transform (DFT) was pointed out by Good [22]. Good [7] showed that the algorithm can be generalized to apply to any multidimensional DFT and even more generally. This depends on the first three of the following facts.

1) A multidimensional mod t DFT is a linear transformation and corresponds to multiplication by a matrix (of order t^n where n is the dimensionality) which can be expressed as the nth direct power of a matrix of order t.

2) The nth direct power of a matrix M is equal to the nth ordinary power of a matrix A defined as

$$A = \{M_{r_1, s_n} \delta^{s_1}_{r_2} \delta^{s_2}_{r_3} \cdots \delta^{s_{n-1}}_{r_n}\}.$$

3) Although A is t^n by t^n it is sparse and contains only t^{n+1} nonzero elements, so that multiplication by M involves only nt^{n+1} ordinary multiplications instead of t^{2n}. This, and the remarks under facts 1 and 2, extended readily to the case of unequal moduli $t_1, t_2, \cdots, t_n$ and even more generally. We thus have a fast algorithm for the computation of an n-dimensional K-transform, defined by (5), of which the n-dimensional DFT is a special case. The remarks following (5) give the algorithm in nonmatrix language.

When $K_1, \cdots, K_n$ in (5) are all the same function, the matrices $A^{(1)}, \cdots, A^{(n)}$ are all equal and this matrix formulation is then at its simplest. In equation (6) the K's cannot be identical unless the equation reduces to (5).

A mod 2 multidimensional DFT is an example of a Hadamard transform which is discussed and applied by Pratt, Kane, and Andrews [12]. Another example of a K-transform is the Andrews–Caspari transform [17] in which the orthogonal matrix of the transform has elements

$$m_{r,s} = \alpha^n \beta^{[r,s]} (-1)^{r \cdot s} \quad (t_v = 2;\ v = 1, 2, \cdots, n)$$

where α is real and positive,

$$\beta = \sqrt{1 - \alpha^2}/\alpha, \quad \text{and} \quad [\mathbf{r}, \mathbf{s}] = \sum_{\mu=0}^{n-1} (r_\mu \oplus s_\mu)$$

where $\oplus$ denotes mod 2 addition (EXCLUSIVE OR), but the summation is ordinary summation which happens to reduce to counting.

4) If, in (5), which defines a multidimensional K-transform, the functions $K_1(r_1, s_1)$, $K_2(r_2, s_2), \cdots$, when regarded as elements of matrices, are square and nonsingular, and if the inverse matrices have elements $J_1(r_1, s_1)$, $J_2(r_2, s_2)$, etc., then the inverse transform can be written

$$a_r = \sum_s a^*_s J_1(s_1, r_1) \cdots J_n(s_n, r_n). \tag{29}$$

This can be readily proved by using Kronecker deltas and is another way of expressing (21).

5) When one wishes to carry out a multiplication by a large matrix it can be worthwhile to look for a factorization into several sparse matrices so as to cut down on the work.

6) New coding methods, generalizing the DFT, can be invented ad lib by taking several sparse square matrices of equal orders and using their product as the matrix of the transformation or coding. In order that unique decoding should be possible, it is necessary that the sparse matrices should all be nonsingular and convenient if their inverses are sparse. Apart from the methods discussed in this paper, the matrices could, for example, be taken as permutation matrices, these being the simplest and sparsest possible nonsingular matrices. They might be too sparse for some purposes since they merely permute the elements of the vector on which they operate and so have no tendency to flatten the variation from one component to another. Such flattening is apt to be convenient for communication systems, but the permutation by itself gives some robustness against bursts of noise. It might also turn out to be useful to use functions $K_1, K_2, \cdots, K_n$ as in (6) instead of (5), although it is presumably more difficult to find elegant schemes of this kind.

7) Unidimensional DFTs can be expressed as multidimensional ones by either the Sino–Ruritanian method or the "reactionary" method, the latter being more widely applicable for this purpose. Once in the multidimensional form, the streamlining described above is applicable. The paper [7] provoked the influential paper by Cooley and Tukey in which the reactionary method was described. Their original acknowledgments to my paper were somewhat too generous since they gave the impression that the two methods were almost identical. Their methods more closely resembled the methods of Runge [23] and Danielson and Lanczos [24], whose papers had been generally overlooked by oceanographers, X-ray crystallographers, and many others who make frequent use of harmonic and spectral analysis. These authors wrote before the age of electronic

computers and were in this respect ahead of their time, although a fast Fourier transform would have been of use for more than 100 years. In the design of special-purpose equipment, both the Sino–Ruritanian and reactionary representations should be kept in mind.

8) One application of the DFT is for the inversion of a circulix, that is, a $t \times t$ matrix (c_{s-r}) where $s-r$ is taken modulo t [25]. A 1 000 000 × 1 000 000 circulix could easily be inverted since it requires only about 100 000 000 operations instead of some quintillion if the matrix were random. The eigenvalues of a circulix are given by the DFT of its top row, and the inverse DFT of their reciprocals gives the inverse circulix. Similarly a recursively blocked circulix, defined recursively as either a circulix or a block matrix whose elements are recursively blocked circulices, or equally defined as (a_{s-r}) where $\boldsymbol{r}$ and $\boldsymbol{s}$ are ordered lexicographically and $\boldsymbol{s}-\boldsymbol{r}$ is taken modulo $(t_1, t_2, \cdots, t_n)$, has eigenvalues given by the multidimensional DFT of its top row, and so can be inverted in a similar manner. (I have avoided the name "block circulix" tout court, because a block circulant is defined by Muir [20, p. 485] in a somewhat different sense, apart, of course, from its being a determinant and not a matrix.) The eigenvectors of a recursively blocked circulix are the columns of the matrix $(\omega_1^{r_1 s_1}\omega_2^{r_2 s_2}\cdots)$.

9) Let p be a prime number and g one its primitive roots; that is $1, g, g^2, \cdots, g^{p-2}$ are all distinct modulo p. Let α_r $(r = 0, 1, \cdots, p-2)$ be a vector of integers and let us define a *number-theoretic Fourier transform:*

$$\alpha_s^* = \sum_{r=0}^{p-2} \alpha_r g^{rs} \pmod{p}. \tag{30}$$

Then it is easily proved that the inverse transform is

$$\alpha_r = -\sum_{s=0}^{p-2} \alpha_s^* g^{-rs} \pmod{p} \tag{31}$$

and the transform of the convolution

$$\sum_{q=0}^{p-2} \alpha_q \beta_{r-q}$$

is $\alpha_s^* \beta_s^*$. A $(p-1)\times(p-1)$ circulix of integers has eigenvalues equal to the transform of its top row, and it can be inverted modulo p by using modulo p reciprocals of the eigenvalues, almost as in fact 8. Recursively blocked circulices of integers, modulo p, have their eigenvalues equal to multidimensional transforms of their top rows, these transforms being defined in an obvious way, modulo a *fixed* prime p. These multidimensional transforms could be used for a coding scheme for vectors of integers modulo p. The calculation can be done by the simple method following (5), or by the equivalent matrix methods, except that the arithmetic must of course be modulo p.

If p is replaced by a composite number m we can define

$$\alpha_s^* = \sum_{r<m}^{(r,m)=1} \alpha_r g^{rs} \pmod{m},$$

but this seems uninteresting because it does not have an elegant inverse even when m has a primitive root (whose powers take all values prime to m). But (30) can be generalized in a different direction, as we now indicate.

10) We can generalize (30) and its multidimensional form to a finite (Galois) field F. A finite field contains a prime power p^v of elements and always has a primitive root g for which $1, g, g^2, \cdots, g^{p^v-2}$ are distinct elements of the field [26]. If α_r $(r=0, 1, \cdots, p^v-2)$ is a vector of field elements we can define

$$\alpha_s^* = \sum_{r=0}^{p^v-2} \alpha_r g^{rs} \quad (s = 0, 1, \cdots, p^v - 2) \tag{32}$$

as its (unidimensional) *Fourier–Galois tranform.* Similarly, for an array $\alpha_{\boldsymbol{r}}(\boldsymbol{r}=(r_1, \cdots, r_n))$, we can define a multidimensional transform

$$\alpha_{\boldsymbol{s}}^* = \sum_{\boldsymbol{r}} \alpha_{\boldsymbol{r}} g_1^{r_1 s_1} \cdots g_n^{r_n s_n} \tag{33}$$

where $g_1, g_2, \cdots, g_n$ are primitive elements of F, not necessarily distinct. This can also be written in the form

$$\alpha_{\boldsymbol{s}}^* = \sum_{\boldsymbol{r}} \alpha_{\boldsymbol{r}} g^{c_1 r_1 s_1 + \cdots + c_n r_n s_n} \tag{34}$$

where $c_1, \cdots, c_n$ are certain constants prime to p^v-1. The inversion formula is

$$\alpha_{\boldsymbol{r}} = (-1)^n \sum_{\boldsymbol{s}} \alpha_{\boldsymbol{s}}^* g_1^{-r_1 s_1} \cdots g_n^{-r_n s_n} \tag{35}$$

and there is a convolution formula

$$\sum_{\boldsymbol{r}} \left(\sum_{\boldsymbol{q}} \alpha_{\boldsymbol{q}} \beta_{\boldsymbol{r}-\boldsymbol{q}} \right) g_1^{r_1 s_1} \cdots g_n^{r_n s_n} = \alpha_{\boldsymbol{s}}^* \beta_{\boldsymbol{s}}^*. \tag{36}$$

A recursively blocked circulix of group elements has, as eigenvalues, the multidimensional Galois–Fourier transform of its top row, and, as eigenvectors, the columns of the matrix $(g_1^{r_1 s_1}) \times \cdots \times (g_n^{r_n s_n})$. The transform could be used for coding "sentences" of "words" where each word, being an element of F, could be expressed by means of v modulo p integers, namely the coefficients of the word when the group is represented by means of polynomials modulo both p and a fixed irreducible polynomial [27]. (Finite fields have often been used in other ways in coding theory and in the design of statistical experiments.)

For the multidimensional transform we can again apply the simple algorithm following (5), or the equivalent matrix methods, the arithmetic being in the field. I can see no elegant transformation from unidimensional to multidimensional Fourier–Galois transforms.

References

[1] *IEEE Trans. Audio Electroacoust.*, vol. AU-15, June 1967.
[2] *IEEE Trans. Audio and Electroacoust.*, vol. AU-15, June 1969.
[3] I. J. Good, "Polynomial algebra: An application of the fast Fourier transform," *Nature*, vol. 222, 1969, p. 1302.
[4] ——, "The discrete Fourier transform: Work by I. J. Good," June, 1969, pp. 1–6, mimeographed.
[5] R. C. Singleton, "A short bibliography on the fast Fourier transform," in [2], pp. 166–169.
[6] I. J. Good, "The fast Fourier transform and discrete Fourier transform: Bibliography," June 1969, pp. 1–5, mimeographed.
[7] ——, "The interaction algorithm and practical Fourier analysis,"

J. Roy. Statist. Soc. Ser. B, vol. 20, 1958, pp. 361–372; vol. 22, 1960, pp. 372–375.
[8] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Computation*, vol. 19, 1965, pp. 297–301.
[9] I. J. Good, "Random motion on a finite Abelian group," *Proc. Cambridge Phil. Soc.*, vol. 47, 1951, pp. 756–762.
[10] ——, "The serial test and other tests for randomness," *Proc. Cambridge Phil. Soc.*, vol. 49, 1953, pp. 276–284.
[11] ——, "Analogues of Poisson's summation formula," *Amer. Math. Mon.*, vol. 69, 1962, pp. 259–266.
[12] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proc. IEEE*, vol. 57, Jan. 1969, pp. 58–68.
[13] W. W. R. Ball, *Mathematical Recreations and Essays*. London: Macmillan, 1940. (Revised by H. M. S. Coxeter.)
[14] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*. Oxford: Clarendon Press, 1938.
[15] P. Bachmann, *Niedere Zahlentheorie*, pt. 1. Leipzig: Teubner, 1902, p. 83.
[16] C. Bingham, M. D. Godfrey, and J. W. Tukey, "Modern techniques of power spectral estimation," in [1], pp. 56–66.
[17] H. C. Andrews and K. L. Caspari, "A generalized technique for spectral analysis," *IEEE Trans. Computers*, vol. C-19, Jan. 1970, pp. 16–25.
[18] C. C. MacDuffee, *The Theory of Matrices*. New York: Chelsea, 1956, p. 82.
[19] L. J. Savage, *The Foundations of Statistics*. New York: Wiley, 1954, p. *viii*.
[20] T. Muir, *A Treatise on the Theory of Determinants*. New York: Dover, 1960.
[21] F. Yates, *The Design and Analysis of Factorial Experiments*. Harpenden: Imperial Bureau of Soil Science, 1937.
[22] I. J. Good, review of M. H. Quenouille's "Design and analysis of experiment." in *Ann. Eugen.*, vol. 18, 1953, p. 263.
[23] C. Runge, "Über die Zerlegung empirisch gegebener periodischer Funktionen in Sinnuswellen," *Z. Math. Physik*, vol. 48, 1903, pp. 443–456; vol. 52, 1905, p. 117.
[24] G. C. Danielson and C. Lanczos, "Some improvements in practical Fourier analysis and their application to X-ray scattering from liquids," *J. Franklin Inst.*, vol. 233, 1942, pp. 365–380 and 435–452.
[25] I. J. Good, "On the inversion of circulant matrices," *Biometrika*, vol. 37, 1950, pp. 185–186.
[26] G. Birkhoff and S. MacLane, *A Survey of Modern Algebra*. New York: Macmillan, 1944, pp. 428–431.
[27] R. D. Carmichael, *Introduction to the Theory of Groups of Finite Order*. Boston: Ginn, 1937, p. 255.

A Universal Cellular Array

JUNG-CHANG HUANG, MEMBER, IEEE

Abstract—A new type of fixed-cell fixed-interconnection homogeneous cellular array that is capable of realizing any switching network is developed. The array is composed of identical combinational logic cells with three inputs and three outputs. The logic cells are arranged in a rectangular array with uniform interconnection structure. The signal flow is unilateral in the vertical direction and bilateral in the horizontal direction. Each *n*-column array is capable of realizing any set of *n* functions of *n* variables. The functional capabilities of the array can be changed by appropriately setting the parameters on the edges of the array. Algorithms for the realization of a set of *n* functions of *n* variables by using this type of array are presented.

Index Terms—Cellular arrays, combinational switching networks, iterative switching circuits, universal arrays.

Manuscript received September 12, 1969; revised July 1, 1970, and August 22, 1970. This work was supported by the U. S. Army Research Office (Durham) under Contract DA-31-124-ARO(D)-97 with the Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, Pa.

The author is with the Department of Computer Science, University of Houston, Houston, Tex. 77004.

I. Introduction

Most of the reported work on combinational cellular arrays considers two main types of arrays, namely, the fixed cell-function arrays [1] and the variable cell-function arrays. In fixed cell-function arrays the switching function realized by each cell is fixed. Realization of a logic function (or a set of functions) is accomplished by modifying the interconnection structure. In variable cell-function arrays the switching function realized by each cell can be changed by means of the cutpoint technique [2] or other methods. Realization of a logic function or a set of functions is accomplished by finding an appropriate configuration of logic cells in the array. In either type of array it is assumed that we have access to each cell in order to change the cell structure, the interconnection structure, or both. This is possible in present circuit construction techniques where the logic cells (integrated circuits) are mounted on the printed-circuit board (which provides the interconnections among the cells). However, these techniques may not apply to future generations of microelectronic modules.

With the rapid progress that is being made in the manufacturing techniques for monolithic integrated circuits, it appears that future generations of microelectronic devices will be the so-called large-scale integrated circuits. Since in such devices each digital component is fabricated on an extremely small area of the substrate, it will be impractical (i.e., very costly to implement), if not impossible, to change the cell functions or the interconnection structure of the array once the device is made. Thus it is deemed desirable to develop a design technique such that a single type of array can be used to realize many different logic functions without the necessity of changing the cell function or its interconnection. One way to accomplish this is to change the functional capabilities of a cellular array by appropriately setting the parameters on the edges of that array. This concept led us to the development of the new array design technique presented in this note. The main theoretical result is a homogeneous cellular array that can be used to realize any switching network.

II. Synthesis of the Array

Let us denote a sequential machine without output by $S=\langle Q, I, M\rangle$ as usual, where Q is the nonempty set of internal states, I is the input alphabet, and M is the next-state function. If $|I|=1$, then we say $S=\langle Q, I, M\rangle$ is a *one-column* (or *autonomous*) *state machine* (as defined in [3]).

It is well known [4] that the transition matrices of the machines $G_1^{(r)}$, $G_2^{(r)}$, and $G_3^{(r)}$ shown in Fig. 1 constitute a generator set that generates all the transition matrices of all r-state, state machines. Consequently, if N_1, N_2, and N_3 are three n-input ($n=\lceil \log_2 r \rceil$), n-output combinational networks that realize the next-state function of $G_1^{(r)}$, $G_2^{(r)}$, and $G_3^{(r)}$, respectively, then the next-state function of any one-column state machine with r states[1] can be realized by

[1] If r is not a power of 2, then this state machine can be realized as a submachine of a machine having 2^n states, where $n=\lceil \log_2 r \rceil$. Thus in this note we only have to deal with the cases in which r is a power of 2.