NASA CR-158,230

JPL PUBLICATION 78-104

Faster Fourier Transformation: The Algorithm of S. Winograd

Shalhav Zohar

February 15, 1979

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL PUBLICATION 78-104

Faster Fourier Transformation: The Algorithm of S. Winograd

Shalhav Zohar

February 15, 1979

N79-19733#

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under NASA Contract No. NAS7-100.

NOTE REGARDING PUBLICATION DATE

The first version of this paper was completed in March 1977. Subsequently, following the publication of [10], a few changes were implemented. These are pointed out in the text and footnotes. The paper was finalized in its present form in May 1977. The delay in publication to February 1979 was due to factors beyond the author's control.

ABSTRACT

The new DFT algorithm of S. Winograd is developed and presented in detail. This is an algorithm which uses about $\frac{1}{5}$ of the number of multiplications used by the Cooley-Tukey algorithm and is applicable to any order which is a product of relatively prime factors from the following list: 2,3,4,5,7,8,9,16. The algorithm is presented in terms of a series of tableaus--one for each term in this list--which are convenient, compact, graphical representations of the sequence of arithmetic operations in the corresponding parts of the algorithm. Using these in conjunction with Tables 5-6 (pp. 80, 84), makes it relatively easy to apply the algorithm and evaluate its performance.

The organization of the paper allows skipping a large part of it on a first reading (see p. 3).

TABLE OF CONTENTS

FIGURES

FIGURES (Cont'd)

TABLES

I. Introduction

Ever since the discovery of the FFT algorithm [1], the following question must have been phrased in many minds: "Does the FFT algorithm represent the ultimate in the fast computation of the discrete Fourier transform (DFT) or is there a still faster algorithm yet to be discovered?" One answer was provided in 1968 by Yavne [2] who showed that the number of multiplications could be halved while leaving the number of additions unchanged. More recently (1976), a significant step along this path was taken by S. Winograd [3] who developed an algorithm which reduces the number of multiplications of the radix-2 FFT algorithm [1] by a factor of about 5. This reduction is accompanied by a small increase or decrease in the number of additions. In most cases, the increase does not exceed 20%.

Our basis for comparison both here and later on is the "nominal" performance of the Cooley-Tukey (FFT) algorithm, namely, the computation of an N-th order DFT of complex data with $\mathcal{M}_{CT}$ real multiplications and $\mathcal{A}_{CT}$ real additions where [1a]

$$\mathcal{M}_{CT} = 2N \log_2 N; \quad \mathcal{A}_{CT} = 1.5\mathcal{M}_{CT} \tag{1.1}$$

We adopt (1.1) as the basis for comparison for all N.

Winograd's algorithm then performs the above task using about $\frac{\mathcal{M}_{CT}}{5}$ real multiplications. We devote the rest of this section to a description of the capabilities and constraints of the algorithm so that the reader could assess its suitability to his needs before delving any deeper.

At its present state of development, the algorithm is applicable to any N satisfying

$$N = \prod_{k=1}^{K} N_k \tag{1.2}$$

in which the N_k's are relatively prime factors taken from the following list

$$N_k = 2,3,4,5,7,8,9,16 \tag{1.3}$$

The maximal N is therefore $16 \cdot 9 \cdot 7 \cdot 5 = 5040$. All of the N values satisfying the above prescriptions are listed in the summary of the algorithm presented in Table 6 (p. 84). The actual multiplication reduction factor is listed there for each N, under the heading G_∞. Note that the average of G_∞ for all $N > 140$ is about 5.5. with such a large reduction in the number of multiplications, it is quite obvious that the new algorithm will run faster than the Cooley-Tukey algorithm in most systems. To make a more specific claim, we must know the basic system parameter μ

[1a] Eqn. (1.1) is adopted as a convenient yardstick. It should be borne in mind that in addition to Yavne's algorithm, there are other FFT variants which are somewhat more efficient than (1.1).

which is the ratio of the time taken to execute one real multiplication to the time taken to execute one real addition. (The term "real" is used here as opposed to "complex"; not as opposed to "integer" in the Fortran language). For very large μ (microprocessors, software multipliers, etc.), the speed gain approaches G_∞ asymptotically. For lower values of μ, the gain would be smaller. Denoting the speed gain by G, it will be shown (pp. 85, 79) that

$$G(\mu) = G_\infty \frac{\mu + 1.5}{\mu + R} \tag{1.4}$$

where R is another parameter listed with G_∞ in Table 6. Obviously, it is now a trivial matter to compute the speed gain for any system and any permissible N.

Since $R > 1.5$ for all practical N values, it is obvious that the advantage of this new algorithm diminishes with decreasing μ. Yet, it is interesting to note that even in the extreme case of $\mu = 1$, the new algorithm is still the faster one for all N values of Table 6.

The main disadvantage of the algorithm is its need for a large memory. We express this in terms of the parameter $\mathcal{M}$ of Table 6. For the processing of complex data we have to have a storage array of $1.5\mathcal{M}$ real words. $\mathcal{M}$ varies from about 2N at the lower end of the table to about 4N at its upper end. Thus, for high N values, we require a storage array of size 6N. This is 4N more than the minimal requirements of 2N for storing the input vector.

Of the total memory requirement of $1.5\mathcal{M}$ real words, $0.5\mathcal{M}$ are needed for the storage of precomputed constants. Since not all of these constants are distinct, it is probably feasible to reduce this part by the use of a more involved addressing scheme.

Another probable disadvantage of the new algorithm is that, in comparison with the Cooley-Tukey algorithm, it might require more bits per word to maintain a prescribed level of precision. This effect is discussed in some more detail in section IX but the whole subject merits further study.

The development of the algorithm consists of two distinct parts. In the first part, fast DFT algorithms for the low orders listed in (1.3) are developed as a set of building blocks. The second part introduces a combining algorithm which integrates groups of these building blocks into the desired final structures, namely, DFT algorithms for orders N prescribed by (1.2).

The low-order algorithms of (1.3) are derivable from algorithms of orders 2,4,6 of another type of transformation called LCT (Left-Circulant Transformation). Hence, the following structure of the paper: Section II is devoted to the DFT-LCT interrelationship. This is followed by the three LCT algorithms in Section III and the seven DFT algorithms derived from them in Sections IV, V, VI. Section VII tackles the integration of these low order algorithms into the desired algorithm for order N satisfying (1.2). Section VIII is devoted to performance evaluation and Section IX concludes with an overview and some comments regarding "in-place" transformation.

The treatment of the subject is detailed and relatively complete, aiming to provide a sound basis for further development of the subject. Naturally, this demands a substantial investment of time. It should be pointed out, however, that a reasonable grasp of the basic ideas and their application can be obtained by skipping the detailed derivations of the low-order algorithms. If this is acceptable, the following parts of the paper may prove sufficient: Section II, first part of Section III (up to the treatment of the left circulant of order 4 on p. 18), the introduction of the η vector on p. 40, the tableau generalization in Section VI (portion bounded by eqns. (6.16), (6.17) on p. 49), Section VII, last part of Section VIII (following eqn. (8.9) on p. 83), and Section IX.

We conclude this introduction with a few words regarding the circumstances that initiated work on the present paper. Winograd's description of his algorithm [3] is very short ($1\frac{1}{2}$ pages) and, partly because of that, hard to follow. However, even with the necessary mathematical background and a full mastery of the paper, actual implementation would still be out of reach without a lot of additional hard work. This is partly due to the fact that the basic building blocks comprising the algorithm are only sketched in general outline--not actually constructed.[1b] It turns out that, for some of these, the transition from outline to end product is far from trivial.

The work reported here was started as an effort to understand and be able to apply what appeared to be--and is in fact--a highly promising, fascinating, algorithm. It was in the course of this work, as the various difficulties were being overcome and the different pieces of the puzzle were beginning to form a coherent overall picture, that the prospect of sharing the knowledge thus acquired, began to have merit. There is no question that the absence of a fuller treatment of the algorithm constitutes a serious obstacle to its wider dissemination and use. This is where the present paper comes in. We present Winograd's algorithm in detail, we construct the required building blocks and we show how to incorporate them in the overall algorithm. In short, we provide the missing links that would allow immediate implementation of the algorithm.

[1b] Since the submission of this paper for publication, a summary of the algorithms did appear in print [10], though still without derivation. This helped to shed light on two points of discrepancy between Winograd's stated results [3] and the first version of this paper. For further details see footnotes 8 (p. 26) and 9 (p. 41).

II. Strategy Development

The cornerstone of Winograd's algorithm is a theorem [4] providing the solution to a seemingly unrelated problem: Given the two polynomials $A(x)$, $B(x)$, what is the minimal number of multiplications required to compute

$$\{A(x)B(x)\} \bmod C(x) \tag{2.1}$$

where $C(x)$ is a given monic[1c] polynomial. The connection with the DFT consists of two links: Firstly, the DFT matrix is shown to be related to another special transformation in which the transforming matrix is a left-circulant (exact definition follows later). Secondly, the evaluation of this transformation is shown to be identical with the evaluation of (2.1). Thus, the minimization of the number of multiplications in (2.1) leads via the above two links to a minimization of the number of multiplications in the computation of the DFT.

We proceed now with some required definitions. A Hankel matrix is one in which the value of element a_{ij} is a function of $(i+j)$. In such a matrix, one encounters identical elements as one moves along any diagonal sloping down and to the left. Obviously, the matrix is completely determined by its first row and last column.

The matrix we will be concerned with here is a special case of a Hankel matrix, namely, a Hankel matrix for which (for order n and index range $0,1,\ldots,n-1$)

$$a_{\rho,n-1} = a_{0,\rho-1} \quad (1 \leq \rho \leq n-1) \tag{2.2}$$

that is, the last column is a trivial rearrangement of the elements of the first row. Hence, this matrix is completely prescribed by its first row. Indeed, the second row is obtained from the first one by a circular left shift (element shifted out on the left, reappears on the right), the third is derived the same way from the second and so on. We call such a matrix a left-circulant[2] or, equivalently,

---

[1c]If $C(x)$ is of degree n, it is said to be monic when the coefficient of x^n is 1.

[2]This is based on the term circulant which is commonly used to describe a matrix generated from its first row by circular right shifts.

an LC matrix. Similarly, the linear transformation effected by such a matrix will be referred to as an LCT (Left-Circulant-Transformation). Finally, a matrix which is not LC but contains a submatrix which is, will be called a quasi-left-circulant matrix (QLC).

We turn now to the LCT-DFT link. As will become obvious later on, we should concern ourselves with a trivial modification of the DFT defined as follows:

$$W = e^{-i\frac{2\pi}{N}} \tag{2.3}$$

$$F_u = \Omega \sum_{v=0}^{N-1} W^{uv} f_v \quad (u = 0,1,\ldots,N-1) \tag{2.4}$$

Ω in (2.4) is an arbitrary complex constant. When $\Omega = 1$, (2.4) reduces to the standard DFT.

Let N, the order of the DFT, satisfy

$$\left.\begin{array}{ll} N = p^k & (p \text{ prime; } k \text{ integer}) \\ k < \begin{cases} \infty \\ 3 \end{cases} & \begin{array}{l} (p \text{ odd}) \\ (p = 2) \end{array} \end{array}\right\} \tag{2.5}$$

We proceed to show now that for such N, eqn. (2.4) can be brought into the form of a QLC matrix[3]. The derivation follows [5] and is based on the number theoretic idea of a primitive root. g, the primitive root of N (satisfying (2.5)) is an integer whose integral powers (mod N) generate all integers in the interval (1,N) except multiples of p.

The number of multiples of p in the above interval is

$$\frac{N}{p} = p^{k-1} \tag{2.6}$$

Therefore, the number of integers generated by g is

$$n = p^k - p^{k-1} = (p-1)p^{k-1} \tag{2.7}$$

and we may say that the sequence

$$\{g^\rho \bmod N\} \quad (\rho = 0,1,\ldots,n-1) \tag{2.8}$$

is just a permutation of those integers in the interval (1,N) which are not multiples of p.

---

[3]This is true for a range wider than (2.5). However, (2.5) is sufficient for our purpose.

We use these ideas now to relabel the indices of (2.4) as follows. All indices which are not multiples of p will be represented as in (2.8). Specifically, denoting

$$\left.\begin{aligned} r &= g^{\rho} \mod N \\ s &= g^{\sigma} \mod N \end{aligned}\right\} \quad (\rho, \sigma = 0,1,\ldots,n-1) \tag{2.9}$$

we define

$$B_{\rho} = F_r; \; b_{\sigma} = f_s \quad (\rho, \sigma = 0,1,\ldots n-1) \tag{2.10}$$

The indices which are multiples of p are now used to define

$$B_{(i)} = F_i; \; b_{(i)} = f_i \qquad (i \mod p = 0) \tag{2.11}$$

With these definitions we embark now on the elimination of F_u, f_v from (2.4). In doing this, we split the summation of (2.4) into two parts. In the first part $v = mp$ (m integer), that is, $v \mod p = 0$. In the second part $v = s$. The result is[4]:

$$B_{(tp)} = F_{tp} = \Omega \sum_{m=0}^{N-n-1} W^{mtp^2} b_{(mp)} + \Omega \sum_{\sigma=0}^{n-1} W^{tpg^{\sigma}} b_{\sigma} \quad (t = 0,1,\ldots,N-n-1) \tag{2.12}$$

$$B_{\rho} = \hat{B}_{\rho} + B'_{\rho} \qquad (\rho = 0,1,\ldots,n-1) \tag{2.13}$$

where

$$\hat{B}_{\rho} = \Omega \sum_{m=0}^{N-n-1} W^{mpg^{\rho}} b_{(mp)} \qquad (\rho = 0,1,\ldots,n-1) \tag{2.14}$$

$$B'_{\rho} = \Omega \sum_{\sigma=0}^{n-1} W^{(g^{\rho+\sigma})} b_{\sigma} \qquad (\rho = 0,1,\ldots,n-1) \tag{2.15}$$

The term $(\rho+\sigma)$ identifies (2.15) as a Hankel transformation, that is, if we write down (2.15) with ρ and σ as the row and column indices, respectively, then the matrix transforming b into B' is a Hankel matrix. More than that, it is that special kind of a Hankel matrix referred to earlier as a left-circulant. To see this, note that in view of the LC condition (2.2) , our Hankel

---

[4]We take here advantage of the fact that (2.3) ensures $W^{(m \mod N)} = W^m$.

matrix would be an LC if

$$g^{\rho+n-1} = g^{\rho-1} \bmod N \tag{2.16}$$

But since the primitive root of N always satisfies[5]

$$g^{n} = 1 \mod N \tag{2.17}$$

It is obvious that (2.16) is indeed true and (2.15) is an LCT. We conclude that the permutations (2.9)-(2.11) will transform any DFT matrix of order N prescribed by (2.5), into a QLC matrix whose LC portion is of order n.

Of particular interest is the special case in which N is prime, that is,

$k=1$; $N=p$, so that (2.7) yields

$$n = N-1 \tag{2.18}$$

and the sequence (2.8) is just a permutation of the integers $1,2,\ldots,n$. In this case, eqns. (2.12), (2.14) simplify as follows:

$$B_{(0)} = \Omega\Big(b_{(0)} + \sum_{\sigma=0}^{n-1} b_\sigma\Big) \tag{2.19}$$

$$\hat{B}_\rho = \Omega b_{(0)} \qquad (\rho = 0,1,\ldots,n-1) \tag{2.20}$$

To illustrate these ideas, consider the case of $N=7$. The least positive primitive root of 7 is 3 [6]. Indeed, direct computation yields

ρ	0	1	2	3	4	5
3^ρ mod 7	1	3	2	6	4	5

(2.21)

Applying (2.9)-(2.11), we get the following form

---

[5]In the standard treatment of primitive roots, (2.8) is usually replaced by

$\{g^\rho \bmod N\}$ $(\rho=1,2,\ldots,n)$ (2.8').

Congruence (2.17) shows that the two formulations are equivalent. To prove (2.17) assume $g^m = 1 \bmod N$ for $m < n$. It follows then that $g^{m+1} = g^1 \bmod N$ and thus two of the terms in sequence (2.8') are equal. Hence the contradiction that g is not a primitive root. Conclusion: $m = n$.

$$\begin{bmatrix} F_0 \\ \hline F_1 \\ F_3 \\ F_2 \\ F_6 \\ F_4 \\ F_5 \end{bmatrix} = \begin{bmatrix} B_{(0)} \\ \hline B_0 \\ B_1 \\ B_2 \\ B_3 \\ B_4 \\ B_5 \end{bmatrix} = \Omega \left[\begin{array}{c|cccccc} W^0 & W^0 & W^0 & W^0 & W^0 & W^0 & W^0 \\ \hline W^0 & W^1 & W^3 & W^2 & W^6 & W^4 & W^5 \\ W^0 & W^3 & W^2 & W^6 & W^4 & W^5 & W^1 \\ W^0 & W^2 & W^6 & W^4 & W^5 & W^1 & W^3 \\ W^0 & W^6 & W^4 & W^5 & W^1 & W^3 & W^2 \\ W^0 & W^4 & W^5 & W^1 & W^3 & W^2 & W^6 \\ W^0 & W^5 & W^1 & W^3 & W^2 & W^6 & W^4 \end{array}\right] \bullet \begin{bmatrix} b_{(0)} \\ \hline b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} \Bigg\| \begin{bmatrix} b_{(0)} \\ \hline b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} f_0 \\ \hline f_1 \\ f_3 \\ f_2 \\ f_6 \\ f_4 \\ f_5 \end{bmatrix} \tag{2.22}$$

The LC structure is quite apparent here.

So far, we have established the first link to problem (2.1) for N satisfying (2.5). This constraint on N will be relaxed later on. We turn now to the second link, namely, showing that evaluation of an LCT is equivalent to evaluation of (2.1). We intend to establish this equivalence and, from it, derive algorithms for some general low-order LC transformations. It should be pointed out, however, that the LC matrix we are concerned with is one obtained by permuting a DFT submatrix and, as such, is still a function of only one variable (W) whereas the general LC matrix of order n is a function of n variables. This suggests further simplifications in our case which will indeed be realized later on.

The matrix multiplication we are considering is shown in (2.23) where the LC pattern is clearly visible.

$$\begin{bmatrix} t_m \\ t_{m-1} \\ t_{m-2} \\ \vdots \\ t_2 \\ t_1 \\ t_0 \end{bmatrix} = \begin{bmatrix} a_m & a_{m-1} & a_{m-2} & \cdots & a_2 & a_1 & a_0 \\ a_{m-1} & a_{m-2} & & & & a_0 & a_m \\ a_{m-2} & & & & & & a_{m-1} \\ \vdots & & & & & & \vdots \\ a_2 & & & & & & a_3 \\ a_1 & a_0 & & & & a_3 & a_2 \\ a_0 & a_m & a_{m-1} & \cdots & a_3 & a_2 & a_1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_{m-2} \\ b_{m-1} \\ b_m \end{bmatrix} \tag{2.23}$$

We introduce now the auxiliary polynomials. (NOTE: The polynomial subscript indicates its degree.)

$$A_m(x) = \sum_{i=0}^{m} a_i x^i \tag{2.24}$$

$$B_m(x) = \sum_{i=0}^{m} b_i x^i \tag{2.25}$$

$$T_m(x) = \sum_{i=0}^{m} t_i x^i \tag{2.26}$$

Consider the product polynomial

$$V_{2m}(x) = A_m(x)\ B_m(x) = \sum_{i=0}^{2m} v_i x^i \tag{2.27}$$

It is easy to see that the coefficients of this polynomial are obtainable by the matrix product (2.28).

$$\begin{bmatrix} v_{2m} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ v_{m+1} \\ v_m \\ v_{m-1} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ v_0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & a_m \\ 0 & 0 & & & & a_m & a_{m-1} \\ 0 & & & & & & a_{m-2} \\ \vdots & & & & & & \vdots \\ 0 & & & & & & a_2 \\ 0 & a_m & \cdots & \cdots & \cdots & a_2 & a_1 \\ a_m & a_{m-1} & a_{m-2} & \cdots & a_2 & a_1 & a_0 \\ a_{m-1} & a_{m-2} & & & & a_0 & 0 \\ a_{m-2} & & & & & & 0 \\ \vdots & & & & & & \vdots \\ a_2 & & & & & & 0 \\ a_1 & a_0 & & & & 0 & 0 \\ a_0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_{m-2} \\ b_{m-1} \\ b_m \end{bmatrix} \tag{2.28}$$

Comparing this to (2.23), we see that interchanging the two indicated triangular sections in (2.28) will transform the matrix of (2.28) into a trivial augmentation of the matrix of (2.23). This fact can be translated to the following relationship between $T_m(x)$ and $V_{2m}(x)$:

$$\begin{aligned} T_m(x) = V_{2m}(x) &- (a_m b_1 + a_{m-1} b_2 + \ldots + a_1 b_m)\ (x^{m+1}-1) \\ &- (a_m b_2 + \ldots + a_2 b_m)\ \ x(x^{m+1}-1) \\ &\ \ \vdots \\ &- (a_m b_{m-1} + a_{m-1} b_m)\ x^{m-2}(x^{m+1}-1) \\ &- a_m b_m\ x^{m-1}(x^{m+1}-1) \end{aligned} \tag{2.29}$$

$$\therefore\ T_m(x) = V_{2m}(x) - (x^{m+1}-1)\ F_{m-1}(x) \tag{2.30}$$

where $F_{m-1}(x)$ is the indicated polynomial of degree $(m-1)$. Hence

$$\frac{V_{2m}(x)}{x^{m+1}-1} = F_{m-1}(x) + \frac{T_m(x)}{x^{m+1}-1} \tag{2.31}$$

Note that, in the quotient on the right, the denominator degree is higher than the numerator degree. This means that $T_m(x)$ is just the remainder obtained when dividing $V_{2m}(x)$ by $(x^{m+1}-1)$. In other words,

$$T_m(x) = \{A_m(x)\ B_m(x)\} \quad \bmod\ (x^n-1) \tag{2.32}$$

We have made use here of the fact that the order of the LC matrix in (2.23) is

$$n = m+1 \tag{2.33}$$

Eqn. (2.32) is a prescription for performing the LCT of (2.23) through polynomial manipulations identical with those of (2.1). This, then, establishes the second link.

Consider now the number of multiplications required to evaluate (2.23). Straight matrix multiplication requires n^2 scalar multiplications. A much lower value is prescribed by Winograd's theorem [4]. In its narrower application to the present case, the theorem states that if (x^n-1) is representable as

$$x^n-1 = \prod_{i=1}^{k(n)} m_i(x) \tag{2.34}$$

where the $m_i(x)$ are distinct polynomials irreducible over the field of rationals, then the minimal number of multiplications is (2n-k), provided multiplications by rational numbers are not counted.

The exclusion of rational multiplications merits an explanation. Suppose we have a minimal realization of $T_m(x)$ of the following form

$$T_m(x) = \sum_{r=1}^{R} \left(\frac{J_r}{K_r}\right) F_r(x) \tag{2.35}$$

where J_r, K_r are integers and in which $F_r(x)$ involves no rational multiplications. According to the theorem, the F_r's will require a total of (2n-k) multiplications and we are essentially being told that the additional R rational multiplications appearing in (2.35) do not count. To see what is involved here, let us clear fractions in (2.35). Let K be the least common denominator and let

$$\frac{J_r}{K_r} = \frac{J_r'}{K} \tag{2.36}$$

Hence,

$$K\, T_m(x) = \sum_{r=1}^{R} J_r' \, F_r(x) \tag{2.37}$$

Each of the multiplications by J_r' can be implemented as $J_r'-1$ additions so that $KT_m(x)$ does not require any multiplications above the (2n-k) used in computing the F_r's. Finally, we compensate for the multiplication by K on the left, by prescaling the a matrix, that is, replacing a_i, $A_m(x)$ by

$$\hat{a}_i = \frac{a_i}{K} \; ; \quad \hat{A}_m(x) = \sum_{i=0}^{m} \hat{a}_i x^i \tag{2.38}$$

Thus, (2.32) is now replaced by

$$T_m(x) = K\{\hat{A}_m(x)\; B_m(x)\} \bmod (x^n-1) \tag{2.39}$$

and we see that multiplications by rationals can always be eliminated without an increase in the number of irrational multiplications.

It should be pointed out that while this argument is theoretically sound, practically, one should be concerned with the cost in terms of the extra additions introduced to eliminate the rational multiplications. Obviously, when these extra additions take longer than the multiplications they replace, we would be better off leaving the multiplications in. This will, indeed, be the case when n is large. Therefore, the algorithm taking advantage of Winograd's theorem, has to be constructed in such a way that a DFT of large order is broken down into many LC transformations of low order. As a matter of fact, all the DFT orders appearing in Table 6 ($N_{max} = 5040$) call for just three LC transformations of orders 2, 4, 6.

The factoring of (x^n-1) for these three cases is shown in Table 1 in which the last column gives the minimal number of multiplications as stated by Winograd's theorem. In the next section we develop the specific algorithms which realize these minima. These three algorithms serve as a foundation for the subsequent construction of DFT algorithms for all orders listed in (1.3).

III. The Basic LCT Algorithms

Our approach here is to present the general method first in sufficient detail so that its application to the three specific n values can be subsequently presented as a mostly self-explanatory sequence of equations.

The starting point is (2.39) in which K is left indeterminate till the very end of the derivation. The factoring of x^n-1 is spelled out in Table 1 which identifies the $m_i(x)$ factors of (2.34). With the m_i's available, we evaluate $T_m(x)$ in a two-phase scheme based on the polynomial version of the Chinese Remainder Theorem [7]. In phase 1 we compute

$$u_i(x) = \{\hat{A}_m(x)\ B_m(x)\} \bmod m_i(x) \tag{3.2}$$

for all i of (2.34). This is based entirely on results established in the Appendix and summarized in Tables A1, A2 there. In phase 2 we use the polynomial version of Garner's algorithm [7] to construct $T_m(x)$ from the u_i's. This calls for the auxiliary functions $c_{ij}(x)$, $v_i(x)$ introduced below. Their utilization in the construction of $T_m(x)$ is spelled out in (3.6).

$$\left[m_i(x)\ c_{ij}(x)\right] \bmod m_j(x) = 1 \qquad \text{(definition of } c_{ij}(x)) \tag{3.3}$$

$$v_1(x) = u_1(x) \tag{3.4}$$

$$v_i(x) = \{(\ldots\left\langle\left[(u_i(x)-v_1(x))c_{1,i}(x)-v_2(x)\right]c_{2,i}(x)-v_3(x)\right\rangle c_{3,i}(x)-,\ldots,- \\ -v_{i-1}(x))\ c_{i-1,i}(x)\} \bmod m_i(x) \tag{3.5}$$

$$T_m(x) = K\left\{v_1(x) + \sum_{i=2}^{k(m-1)} v_i(x) \prod_{j=1}^{i-1} m_j(x)\right\} \qquad \text{(k from Table 1)} \tag{3.6}$$

The computation of $c_{ij}(x)$ is trivial when $m_j(x)$ is of degree 1, that is, $m_j(x) = x-x_j$. From (A.5) in the Appendix, we see that, in this case,

Table 1

Rational Factorization of x^n-1

n	x^n-1	k	2n-k
2	$(x-1)(x+1)$	2	2
4	$(x-1)(x+1)(x^2+1)$	3	5
6	$(x-1)(x+1)(x^2+x+1)(x^2-x+1)$	4	8

$$m_i(x_j)c_{ij}(x_j) = 1 \tag{3.7}$$

Now, any $c_{ij}(x)$ satisfying (3.3) (and hence (3.7)) would do. Choosing the lowest degree we get

$$c_{ij}(x) = c_{ij} = \frac{1}{m_i(x_j)} \tag{3.8}$$

With these derivation outlines spelled out, we turn now to specific cases. To establish the evolving pattern, we follow these outlines even in the low order case (n=2) where direct derivation could be simpler.

<u>Left-Circulant Transformation of Order 2 (Fig. 1)</u>

$$\left.\begin{aligned}
&\begin{bmatrix} t_1 \\ t_0 \end{bmatrix} = \begin{bmatrix} a_1 & a_0 \\ a_0 & a_1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \\
&\hat{a}_i = \frac{a_i}{K}; \quad \hat{A}_1(x) = \sum_{i=0}^{1} \hat{a}_i x^i; \quad B_1(x) = \sum_{i=0}^{1} b_i x^i
\end{aligned}\right\} \tag{3.9}$$

<u>Phase 1</u>

$$T_1(x) = K\{\hat{A}_1(x)\ B_1(x)\} \mod [\underbrace{(x-1)}_{m_1}\underbrace{(x+1)}_{m_2}] \tag{3.10}$$

$$u_1(x) = \{\hat{A}_1(x)\ B_1(x)\} \mod (x-1)$$

$$= \underbrace{(\hat{a}_0+\hat{a}_1)}_{\alpha_1}\underbrace{(b_0+b_1)}_{\beta_1} \quad \text{(see (A.5))}$$

$$\delta_1 = u_1 = \alpha_1\beta_1$$

$$u_2(x) = \{\hat{A}_1(x)\ B_1(x)\} \mod (x+1)$$

$$= \underbrace{(\hat{a}_0-\hat{a}_1)}_{\alpha_2}\underbrace{(b_0-b_1)}_{\beta_2}$$

$$\delta_2 = u_2 = \alpha_2\beta_2$$

Phase 2

$$c_{12}(x) = \frac{1}{m_1(x_2)} = \frac{1}{m_1(-1)} = -\frac{1}{2}$$

$$v_1 = \delta_1$$

$$v_2 = \{(\delta_2-\delta_1)(-\tfrac{1}{2})\} \bmod (x+1) = \tfrac{1}{2}(\delta_1-\delta_2)$$

$$T_1(x) = K\left[\delta_1 + \frac{1}{2}(\delta_1-\delta_2)(x-1)\right] = \underbrace{\left[\frac{K}{2}(\delta_1-\delta_2)\right]}_{t_1} x + \underbrace{\left[\frac{K}{2}(\delta_1+\delta_2)\right]}_{t_0} \qquad (3.11)$$

The obvious choice here is $K=2$ so that the final result is

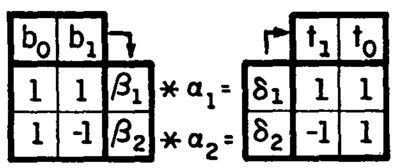
$$t_0 = \delta_1+\delta_2; \; t_1 = \delta_1-\delta_2$$

The algorithm is summarized graphically in the tableau of Figure 1. The conventions adopted here are quite simple and are also the ones adopted in the more complex tableaus presented later on. We defined β_i as a linear combination of the b_j's. The β_i row lists the (non-zero) coefficients of this linear combination. Similarly, we found that t_i is a linear combination of the δ_j's. The t_i column lists the coefficients of this linear combination. The left corner arrow indicates that the β_i's are derived from the b_j's and not the other way around. Usually there is no directional ambiguity so that the arrows may be omitted. The equations $\delta_i = \beta_i\alpha_i$ appear explicitly in the tableau using the Fortran multiplication symbol.

Finally, recall that the basic eqn. (2.39) is fully symmetric with respect to $\{\hat{a}_j\},\{b_j\}$. We have taken advantage of this in the terminology introduced here. Thus, coupled with each β_i which is a specific function of $\{b_j\}$ (say, $\phi_i(\{b_j\})$) spelled out in the tableau, there is the variable α_i which is exactly the same function of $\{\hat{a}_j\}$, that is,

$$\alpha_i = \phi_i\left(\left\{\frac{a_j}{K}\right\}\right)$$

This convention is adhered to throughout the paper. Practically, this means that



$$\alpha_i = \phi_i\left(\left\{\frac{a_j}{2}\right\}\right)$$

$$\beta_i = \phi_i\left(\left\{b_j\right\}\right)$$

$\underline{n = 2 \ \ (2M;\, 4A)}$

Fig. 1. Algorithm for LCT of order 2

the left part of the tableau has to be run through twice. First, with $\{\frac{a_j}{K}\}$ replacing $\{b_j\}$ and thus yielding $\{\alpha_i\}$ instead of $\{\beta_i\}$, then we run through the full tableau with $\{b_j\}$ as input. Note, however, that in spite of the mathematical symmetry between $\{\frac{a_j}{K}\}$ and $\{b_j\}$, practically, there is an important difference between them. a is considered a constant matrix transforming a number of different data vectors b. Therefore, the α_i's may be precomputed once and for all, their computation being ignored in accounting for the cost of transforming one data vector b. With this in mind, we count only the number of explicit arithmetic operations in the tableau, arriving at 2 multiplications and 4 additions, for which we adopt the designation (2M; 4A) appearing in the figure. Note that the 2 multiplications are the minimum prescribed by Winograd's theorem (right column in Table 1).

Left Circulant Transformation of Order 4 (Fig. 2)

In this and all other tableaus derived in this paper, it is suggested that the reader consider the tableau as he follows its derivation, noting the graphical representation of each mathematical statement as the algorithm evolves.

$$\begin{bmatrix} t_3 \\ t_2 \\ t_1 \\ t_0 \end{bmatrix} = \begin{bmatrix} a_3 & a_2 & a_1 & a_0 \\ a_2 & a_1 & a_0 & a_3 \\ a_1 & a_0 & a_3 & a_2 \\ a_0 & a_3 & a_2 & a_1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \qquad \left.\begin{aligned} \hat{a}_i &= \frac{a_i}{K} \\ \hat{A}_3(x) &= \sum_{i=0}^{3} \hat{a}_i x^i \\ B_3(x) &= \sum_{i=0}^{3} b_i x^i \end{aligned}\right\} \tag{3.13}$$

Phase 1

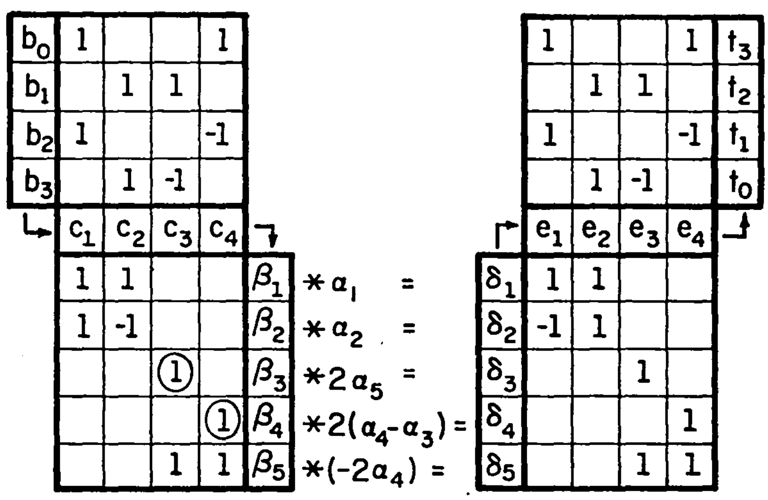
$$T_3(x) = K\,\{\hat{A}_3(x)B_3(x)\} \quad \mathrm{mod}\{\underbrace{(x^2+1)}_{m_1}\underbrace{(x+1)}_{m_2}\underbrace{(x-1)}_{m_3}\} \tag{3.14}$$

$$u_3(x) = \hat{A}_3(1)B_3(1) = \underbrace{(\hat{a}_0+\hat{a}_1+\hat{a}_2+\hat{a}_3)}_{\alpha_1}\underbrace{(b_0+b_1+b_2+b_3)}_{\beta_1}$$

$$c_1 = b_0+b_2; \quad c_2 = b_1+b_3$$

$$\therefore\ \beta_1 = c_1+c_2$$

$$\delta_1 = u_3 = \alpha_1\beta_1$$



$$\alpha_i = \phi_i \left\{ \frac{a_j}{4} \right\}$$

$$\beta_i = \phi_i \{ b_j \}$$

<u>n = 4 (5M; 15A)</u>

Fig. 2. Algorithm for LCT of order 4

$$u_2(x) = \hat{A}_3(-1)B_3(-1) = \underbrace{(\hat{a}_0-\hat{a}_1+\hat{a}_2-\hat{a}_3)}_{\alpha_2}\underbrace{(b_0-b_1+b_2-b_3)}_{\beta_2}$$

$$\therefore\ \beta_2 = c_1-c_2$$

$$\delta_2 = u_2 = \alpha_2\beta_2$$

$u_1(x)$ is evaluated in two steps:

$$\left.\begin{aligned} \hat{A}_3(x) \bmod (x^2+1) &= \underbrace{(\hat{a}_1-\hat{a}_3)}_{\alpha_3}x + \underbrace{(\hat{a}_0-\hat{a}_2)}_{\alpha_4} \\ B_3(x) \bmod (x^2+1) &= \underbrace{(b_1-b_3)}_{\beta_3}x + \underbrace{(b_0-b_2)}_{\beta_4} \end{aligned}\right\} \quad \text{(from Table A1)} \qquad (3.15)$$

Note that the tableau of Fig. 2 defines β_3, β_4 indirectly in terms of

$$c_3 = b_1-b_3; \quad c_4 = b_0-b_2$$

Thus,

$$\beta_3 = c_3; \quad \beta_4 = c_4$$

so that no arithmetic operations are involved in the last two equations. Whenever this is the case, we circle the relevant terms in the tableau to stress this fact.

Now we combine the two results of (3.15) using Table A2 ($\alpha_3 = p_1$; $\beta_3 = q_i$; etc.) getting

$$u_1(x) = \left[-(\alpha_4-\alpha_3)\beta_4 + \alpha_4(\beta_3+\beta_4)\right]x + \left[\alpha_4(\beta_3+\beta_4)-(\alpha_3+\alpha_4)\beta_3\right]$$

We introduce now

$$\alpha_5 = \alpha_3+\alpha_4; \quad \beta_5 = \beta_3+\beta_4 = c_3+c_4$$

$$\delta_3 = 2\alpha_5\beta_3; \quad \delta_4 = 2(\alpha_4-\alpha_3)\beta_4; \ \delta_5 = -2\alpha_4\beta_5$$

Hence,

$$u_1(x) = -\frac{1}{2}\underbrace{(\delta_4+\delta_5)}_{e_4}\ x - \frac{1}{2}\underbrace{(\delta_5+\delta_3)}_{e_3}$$

Phase 2

$$c_{12} = \frac{1}{m_1(x_2)} = \frac{1}{m_1(-1)} = \frac{1}{2};\quad c_{13} = \frac{1}{m_1(x_3)} = \frac{1}{m_1(1)} = \frac{1}{2};\quad c_{23} = \frac{1}{m_2(x_3)} = \frac{1}{m_2(1)} = \frac{1}{2}$$

$$v_1(x) = u_1(x) = -\frac{1}{2}(e_4x + e_3)$$

$$v_2 = \{(\delta_2 + \frac{1}{2}e_4x + \frac{1}{2}e_3)\frac{1}{2}\} \mod (x+1) = \frac{1}{4}(2\delta_2 - e_4 + e_3)$$

$$v_3 = \{\left[(\delta_1 + \frac{1}{2}e_4x + \frac{1}{2}e_3)\frac{1}{2} - \frac{1}{4}(2\delta_2 - e_4 + e_3)\right]\frac{1}{2}\} \mod (x-1) = \frac{1}{4}(\delta_1 - \delta_2 + e_4)$$

$$T_3(x) = -\frac{K}{2}(e_4x+e_3) + \frac{K}{4}(2\delta_2 - e_4 + e_3)(x^2+1) + \frac{K}{4}(\delta_1 - \delta_2 + e_4)(x^3+x^2+x+1)$$

Adopting

$$K = 4,$$

$$T_3(x) = (\delta_1-\delta_2+e_4)x^3 + (\delta_1+\delta_2+e_3)x^2 + (\delta_1-\delta_2-e_4)x + (\delta_1+\delta_2-e_3)$$

Introducing now

$$e_1 = \delta_1 - \delta_2;\quad e_2 = \delta_1 + \delta_2$$

we get the final result

$$T_3(x) = (e_1+e_4)x^3 + (e_2+e_3)x^2 + (e_1-e_4)x + (e_2-e_3) \tag{3.16}$$

A count of arithmetic operations in the tableau (excluding the α_i manipulations) yields 5 multiplications and 15 additions[6], again realizing the multiplication minimum of Table 1.

Left-Circulant Transformation of order 6 (Fig. 4)

$$\left.\begin{matrix} \begin{bmatrix} t_5 \\ t_4 \\ t_3 \\ t_2 \\ t_1 \\ t_0 \end{bmatrix} = \begin{bmatrix} a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ a_4 & a_3 & a_2 & a_1 & a_0 & a_5 \\ a_3 & a_2 & a_1 & a_0 & a_5 & a_4 \\ a_2 & a_1 & a_0 & a_5 & a_4 & a_3 \\ a_1 & a_0 & a_5 & a_4 & a_3 & a_2 \\ a_0 & a_5 & a_4 & a_3 & a_2 & a_1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} & \begin{matrix} \hat{a}_i = \frac{a_i}{K} \\ \hat{A}_5(x) = \sum_{i=0}^{5} \hat{a}_i x^i \\ B_5(x) = \sum_{i=0}^{5} b_i x^i \end{matrix} \end{matrix}\right\} \tag{3.17}$$

[6]The author wishes to acknowledge here the help of Dr. R. G. Lipes of JPL in reducing the number of additions from 16 to 15.

Phase 1

$$T_5(x) = K\{\hat{A}_5(x)B_5(x)\} \bmod\{\underbrace{(x^2-x+1)}_{m_1}\underbrace{(x^2+x+1)}_{m_2}\underbrace{(x+1)}_{m_3}\underbrace{(x-1)}_{m_4}\} \tag{3.18}$$

$$u_4 = \hat{A}_5(1)B_5(1) = \left(\sum_{i=0}^{5}\hat{a}_i\right)\left(\sum_{i=0}^{5} b_i\right)$$

$$c_1 = b_3+b_0; c_2 = b_5+b_2;\ c_3 = b_4+b_1$$

$$u_4 = \underbrace{\left(\sum_{i=0}^{5}\hat{a}_i\right)}_{\alpha_1}\ \underbrace{(c_1+c_2+c_3)}_{\beta_1}$$

$$\delta_1 = u_4 = \alpha_1\beta_1$$

$$u_3 = \hat{A}_5(-1)B_5(-1) = \underbrace{(-\hat{a}_0+\hat{a}_1-\hat{a}_2+\hat{a}_3-\hat{a}_4+\hat{a}_5)}_{\alpha_6}\underbrace{(-b_0+b_1-b_2+b_3-b_4+b_5)}_{\beta_6}$$

$$c_6 = b_3-b_0;\quad c_5=b_5-b_2;\ c_4 = b_4-b_1$$

$$u_3 = \alpha_6\underbrace{(c_6+c_5-c_4)}_{\beta_6}$$

$$\delta_6 = u_3 = \alpha_6\beta_6$$

$u_1(x)$, $u_2(x)$ are evaluated in two steps

$$B_5(x) \bmod(x^2+x+1) = \left[\underbrace{(b_4+b_1)}_{c_3}-\underbrace{(b_5+b_2)}_{c_2}\right]x + \left[\underbrace{(b_3+b_0)}_{c_1}-\underbrace{(b_5+b_2)}_{c_2}\right] \quad \text{(from Table A1)}$$

$$= \underbrace{(c_3-c_2)}_{\beta_2}x + \underbrace{(c_1-c_2)}_{\beta_3} = \beta_2 x + \beta_3$$

$$\therefore\ \hat{A}_5(x) \bmod(x^2+x+1) = \alpha_2 x + \alpha_3$$

$$\therefore\ u_2(x) = \{\hat{A}_5(x)B_5(x)\} \bmod (x^2+x+1) = \left[\alpha_3\beta_3-(\alpha_3-\alpha_2)(\beta_3-\beta_2)\right]x + \left[\alpha_3\beta_3-\alpha_2\beta_2\right] \tag{3.19}$$

(from Table A2)

We introduce now:

$$\left.\begin{aligned} \beta_7 &= \beta_3-\beta_2 = c_1-c_2+c_2-c_3 = c_1-c_3 \\ \alpha_7 &= \alpha_3-\alpha_2 \end{aligned}\right\} \tag{3.20}$$

and use these to transform (3.19) into[7]

$$u_2(x) = \underbrace{(\alpha_2\beta_3+\alpha_7\beta_2)}_{-e_3}x + \underbrace{(\alpha_7\beta_2+\alpha_3\beta_7)}_{e_2} \tag{3.21}$$

$$\begin{aligned} B_5(x) \bmod (x^2-x+1) &= -\left[\underbrace{(b_5-b_2)}_{c_5}+\underbrace{(b_4-b_1)}_{c_4}\right]x - \left[\underbrace{(b_3-b_0)}_{c_6}-\underbrace{(b_5-b_2)}_{c_5}\right] \\ &= -\underbrace{(c_5+c_4)}_{\beta_5}x - \underbrace{(c_6-c_5)}_{\beta_4} = -\beta_5 x - \beta_4 \end{aligned}$$

(from Table A1)

$$\therefore \hat{A}_5(x) \bmod (x^2-x+1) = -\alpha_5 x - \alpha_4$$

$$\therefore u_1(x) = \{\hat{A}_5(x)B_5(x)\} \bmod (x^2-x+1) = \left[(\alpha_4+\alpha_5)(\beta_4+\beta_5)-\alpha_4\beta_4\right]x + (\alpha_4\beta_4-\alpha_5\beta_5) \tag{3.22}$$

(from Table A2)

Introducing

$$\left.\begin{aligned} \beta_8 &= \beta_5+\beta_4 = c_5+c_4+c_6-c_5 = c_4+c_6 \\ \alpha_8 &= \alpha_5+\alpha_4, \end{aligned}\right\} \tag{3.23}$$

we transform (3.22) into[7]

$$u_1(x) = \underbrace{(\alpha_5\beta_4+\alpha_8\beta_5)}_{e_4}x + \underbrace{(\alpha_4\beta_8-\alpha_8\beta_5)}_{e_5} \tag{3.24}$$

<u>Phase 2</u>

$$\{m_1(x)c_{12}(x)\} \bmod m_2(x) = 1$$

Let $c_{12}(x). = \gamma_1 x + \gamma_0$

$$\therefore\ m_1(x)c_{12}(x) = (x^2-x+1)(\gamma_1 x+\gamma_0) = \gamma_1 x^3+(\gamma_0-\gamma_1)x^2+(\gamma_1-\gamma_0)x+\gamma_0$$

$$\therefore\ (\gamma_1-\gamma_0-\gamma_0+\gamma_1)x + (\gamma_0-\gamma_0+\gamma_1+\gamma_1) = 1 \qquad \text{(from Table A1)}$$

$$\therefore\ \gamma_1 = \gamma_0 = \tfrac{1}{2};\quad c_{12}(x) = \tfrac{1}{2}(x+1)$$

---

[7] The e_i's defined here, do not appear in the final tableau and are used only in the intermediate steps.

$$c_{13} = \frac{1}{m_1(x_3)} = \frac{1}{m_1(-1)} = \frac{1}{3}; \qquad c_{23} = \frac{1}{m_2(x_3)} = \frac{1}{m_2(-1)} = 1$$

$$c_{14} = \frac{1}{m_1(x_4)} = \frac{1}{m_1(1)} = 1; \quad c_{24} = \frac{1}{m_2(x_4)} = \frac{1}{m_2(1)} = \frac{1}{3}; \quad c_{34} = \frac{1}{m_3(x_4)} = \frac{1}{m_3(1)} = \frac{1}{2}$$

$$v_1 = e_4x + e_5$$

$$v_2 = \left[(-e_3x + e_2 - e_4x - e_5)\tfrac{1}{2}(x+1)\right] \bmod (x^2+x+1)$$

$$v_2 = \frac{1}{2}\left[-(e_4+e_3)x^2 + (-e_4-e_3+e_2-e_5)x + (e_2-e_5)\right] \bmod (x^2+x+1)$$

$$v_2 = \frac{1}{2}\left[(e_2-e_5)x + (e_2+e_3+e_4-e_5)\right] \qquad \text{(from Table A1)}$$

$$v_3 = \left\{(\delta_6 - e_4x - e_5)\frac{1}{3} - \frac{1}{2}\left[(e_2-e_5)x + (e_2+e_3+e_4-e_5)\right]\right\} \bmod (x+1)$$

$$= \frac{1}{6}(-3e_3 - e_4 - 2e_5 + 2\delta_6)$$

$$v_4 = \frac{1}{2}\left\{\left\langle(\delta_1 - e_4x - e_5) - \frac{1}{2}\left[(e_2-e_5)x + (e_2+e_3+e_4-e_5)\right]\right\rangle\frac{1}{3} - \frac{1}{6}(-3e_3 - e_4 - 2e_5 + 2\delta_6)\right\} \bmod (x-1)$$

$$= \frac{1}{6}(\delta_1 - e_2 + e_3 - e_4 + e_5 - \delta_6)$$

$$T_5(x) = K(e_4x+e_5) + \frac{K}{2}\left[(e_2-e_5)x + (e_2+e_3+e_4-e_5)\right](x^2-x+1) + \frac{K}{6}(-3e_3-e_4-2e_5+2\delta_6)(x^4+x^2+1) +$$

$$+ \frac{K}{6}(\delta_1 - e_2 + e_3 - e_4 + e_5 - \delta_6)(x^5+x^4+x^3+x^2+x+1)$$

Collecting equal power terms yields the desired t_i's. We make now the obvious choice K=6 and present the results in the tableau format in Fig. 3a. We note here that the coefficients of t_o and t_3 have identical magnitudes. On the left of the bisecting line, the coefficients themselves are identical whereas to the right of it, they have opposite signs. This holds true also for the pairs (t_1, t_4), (t_2, t_5). Taking advantage of these symmetries, we can reduce the number of additions as shown in Fig. 3b.

The final manipulations involve the elimination of the e_i's from Fig. 3b. This is based on their definitions (3.21), (3.24) and on a judicious application of (3.20), (3.23) as follows:

1	-1	1	-1	1	-1	t_5
1	-1	-2	-2	-1	1	t_4
1	2	1	-1	-2	-1	t_3
1	-1	1	1	-1	1	t_2
1	-1	-2	2	1	-1	t_1
1	2	1	1	2	1	t_0
δ_1	e_2	e_3	e_4	e_5	δ_6	

(a)

	1					-1	t_5
			1	-1			t_4
		1			-1		t_3
	1					1	t_2
			1	1			t_1
		1			1		t_0
	g_1	g_2	g_3	g_4	g_5	g_6	
δ_1	1	1	1				
e_2	-1	2	-1				
e_3	1	1	-2				
e_4				2	1	1	
e_5				1	2	-1	
δ_6				-1	1	1	

(b)

Fig. 3. Stages in the development of the algorithm for LCT of order 6

$$g_1-\delta_1=e_3-e_2=-2\alpha_7\beta_2-\alpha_2(\beta_7+\beta_2)-\alpha_3\beta_7=\underbrace{\beta_2(-\alpha_3-\alpha_7)}_{\delta_2}+\underbrace{\beta_7(-\alpha_2-\alpha_3)}_{\delta_7} \tag{3.25}$$

$$g_6-\delta_6=e_4-e_5=\alpha_5(\beta_8-\beta_5)+2\alpha_8\beta_5-\alpha_4\beta_8 = \underbrace{\beta_5(\alpha_8+\alpha_4)}_{\delta_5}+\underbrace{\beta_8(\alpha_5-\alpha_4)}_{\delta_8} \tag{3.26}$$

$$g_2-\delta_1=2e_2+e_3=\alpha_7(\beta_3-\beta_7)-\alpha_2\beta_3+2\alpha_3\beta_7=\underbrace{\beta_3(\alpha_7-\alpha_2)}_{\delta_3}-\underbrace{\beta_7(-\alpha_2-\alpha_3)}_{\delta_7} \tag{3.27}$$

$$g_5-\delta_6=2e_5+e_4=-\alpha_8(\beta_8-\beta_4)+\alpha_5\beta_4+2\alpha_4\beta_8=\underbrace{\beta_4(\alpha_8+\alpha_5)}_{\delta_4}-\underbrace{\beta_8(\alpha_5-\alpha_4)}_{\delta_8} \tag{3.28}$$

$$g_3-\delta_1=-e_2-2e_3=\alpha_7\beta_2+2\alpha_2\beta_3-\alpha_3(\beta_3-\beta_2)=-\underbrace{\beta_2(-\alpha_3-\alpha_7)}_{\delta_2}-\underbrace{\beta_3(\alpha_7-\alpha_2)}_{\delta_3} \tag{3.29}$$

$$g_4+\delta_6=e_5+2e_4=\alpha_8\beta_5+2\alpha_5\beta_4+\alpha_4(\beta_5+\beta_4)=\underbrace{\beta_4(\alpha_8+\alpha_5)}_{\delta_4}+\underbrace{\beta_5(\alpha_8+\alpha_4)}_{\delta_5} \tag{3.30}$$

This completes the derivation.[8]

---

[8] The corresponding tableau derived in the first version of this paper contained 6 extra additions. The subsequent appearance in print of the prescription of Winograd's algorithm [10], provided the clue to the specific manipulations ((3.25), etc.) utilized here to eliminate these extra additions.

	c_1	c_2	c_3	c_4	c_5	c_6
b_0	1					-1
b_1			1	-1		
b_2		1			-1	
b_3	1					1
b_4			1	1		
b_5		1			1	

g_1	g_2	g_3	g_4	g_5	g_6	
1					-1	t_5
		1	-1			t_4
	1			-1		t_3
1					1	t_2
		1	1			t_1
	1			1		t_0

c_1	c_2	c_3	c_4	c_5	c_6				g_1	g_2	g_3	g_4	g_5	g_6
1	1	1				β_1	$* \alpha_1 =$	δ_1	1	1	1			
	-1	1				β_2	$*(-\alpha_3-\alpha_7) =$	δ_2	1		-1			
1	-1					β_3	$*(\alpha_7-\alpha_2) =$	δ_3		1	-1			
				-1	1	β_4	$*(\alpha_8+\alpha_5) =$	δ_4				1	1	
			1	1		β_5	$*(\alpha_8+\alpha_4) =$	δ_5				1		1
			-1	1	1	β_6	$* \alpha_6 =$	δ_6				-1	1	1
1		-1				β_7	$*(-\alpha_2-\alpha_3) =$	δ_7	1	-1				
			1		1	β_8	$*(\alpha_5-\alpha_4) =$	δ_8					-1	1

$$\alpha_i = \phi_i\left(\left\{\frac{a_j}{6}\right\}\right)$$

$$\beta_i = \phi_i(\{b_j\})$$

<u>n = 6 (8M; 34A)</u>

Fig. 4. Algorithm for LCT of order 6

IV. The Basic DFT Algorithms for Prime N

In this section, we apply the tableaus just derived to obtain the DFT algorithms for the odd prime terms of list (1.3), namely, 3, 5, 7. As indicated in Section I, these will serve as building blocks for higher order DFT's.

We have seen in section II that with the proper relabeling, an N-th order DFT matrix displays an n-th order LC submatrix (2.7). The main part of the contribution of this submatrix to the overall transformation is spelled out in (2.15) repeated here

$$B'_\rho = \Omega \sum_{\sigma=0}^{n-1} W^{(g^{\rho+\sigma})} b_\sigma \qquad (\rho = 0,1,\ldots,n-1) \qquad (4.1)$$

On the other hand, the LC tableaus of the last section are based on the (2.23), (2.33) formuation of the n-th order LC transformation. Therefore, in applying the LCT tableaus to the LC transformation expressed in (4.1), we must adopt the following identifications

$$B'_\rho = t_{n-1-\rho} \qquad (\rho = 0,1,\ldots,n-1) \qquad (4.2)$$

$$a_\rho = \Omega W^{(g^{n-1-\rho})} \qquad (\rho = 0,1,\ldots,n-1) \qquad (4.3)$$

Note the effect of (4.3). The LCT tableaus, being general, provide only a prescription for the computation of the α_i's from the a_i's. However, since (4.3) provides an explicit formula for the a_i's, the α_i's may actually be computed. Specifically, the α_i's are expressible as

$$\alpha_i = \Omega \varepsilon_i \qquad (4.4)$$

in which the ε_i's are functions of i,N only and can thus be precomputed once and for all.

We copy now the remaining equations of the relabeled DFT ((2.13), (2.19), (2.20))

$$B_\rho = \hat{B}_\rho + B'_\rho \qquad (\rho = 0,1,\ldots,n-1) \qquad (4.5)$$

$$\hat{B}_\rho = \Omega b_{(0)} \qquad (\rho = 0,1,\ldots,n-1) \qquad (4.6)$$

$$B_{(0)} = \Omega (b_{(0)} + \sum_{\sigma=0}^{n-1} b_\sigma) \qquad (4.7)$$

Note that eqns. (4.6), (4.7) are valid only for the special case of prime N and are based on

$$n = N-1 \tag{4.8}$$

Eqns. (4.1)-(4.5), on the other hand, are quite general and will also be applied in the next two sections where N is not prime.

Our first step in the construction of the DFT tableau for prime N is the computation of the $\varepsilon_i(N)$ constants. This is done by evaluating the a_ρ's from (4.3) and then using them in the LCT tableau of order N-1 to compute the α_i's, and hence the ε_i's (4.4).

The next step is to copy the LCT tableau of order N-1 into the DFT tableau of order N. This is equivalent to the implementation of (4.1) and yields a tableau transforming the b_i's into the t_j's. Now, using (2.9)-(2.11), (4.2) we replace these variables by f_i's and F_j's. The input and output index sequences we get at this point will usually be non-monotonic. Therefore, we now permute the rows/columns of the input and output squares to achieve index monotonicity.

It should be pointed out that the variable changes (2.9)-(2.11) were introduced to expose the LC structure and thus allow the application of (2.39). Once this has been done, however, we want the resulting DFT tableau to be expressed in terms of the original variables F_u, f_v with standard monotonic index sequences since this simplifies the integration of the tableau into the algorithm for larger N.

We turn now to the implementation of (4.7). Since all three LCT tableaus satisfy

$$\beta_1 = \sum_{i=0}^{n-1} b_i, \tag{4.9}$$

eqn. (4.7) is equivalent to

$$F_0 = B_{(0)} = \Omega(b_{(0)}+\beta_1) = \Omega(f_0+\beta_1) \tag{4.10}$$

Finally, in order to efficiently realize (4.5), we examine the three LCT tableaus to see whether they contain a term which appears as a component with coefficient +1 of all t_i's. This term turns out to be δ_1.

Hence, replacing $\delta_1 = \alpha_1\beta_1$ by

$$\delta_1 = \Omega b_{(0)}+\alpha_1\beta_1 \tag{4.11}$$

will convert the output from B'_ρ to B_ρ. Note, however, that the term $\Omega b_{(0)}$ is already included in $B_{(0)}$ computed in (4.10). This raises the possibility of eliminating one of the two multiplications evident in (4.11). Indeed, combining (4.10) with (4.11) yields (using (4.4))

$$\delta_1 = B_{(0)} + (\varepsilon_1 - 1)\Omega\beta_1 = F_0 + (\varepsilon_1 - 1)\Omega\beta_1 \tag{4.12}$$

The multiplier of β_1 is seen here to be the product of Ω and a function of N. This turns out to be the general pattern for all β_i in all the DFT tableaus yet to be derived. Hence, we introduce now the notation

$$\xi_i = (\omega_i \Omega)\beta_i \tag{4.13}$$

where ω_i is a function of i,N only. β_i is always initially transformed as in (4.13). Hence one of our tasks in constructing the DFT tableaus is the determination of the numerical values of the ω_i constants. As we shall see in the course of the developments, the underlying LCT tableaus in conjunction with (4.4), (4.13), always uniquely determine the ω_i's. In the case of (4.12)

$$\omega_1 = \varepsilon_1 - 1; \quad \delta_1 = B_{(0)} + (\omega_1 \Omega)\beta_1 = F_0 + (\omega_1 \Omega)\beta_1 \tag{4.14}$$

So far, we have considered the LCT-DFT transition in general terms. We turn now to specific cases:

<u>DFT of order 3 (Fig. 5)</u>

Eqn. (4.10) is explicitly stated in the tableau. From (4.3) (with $W = e^{-i\frac{2\pi}{3}}$; $g = 2$) we get

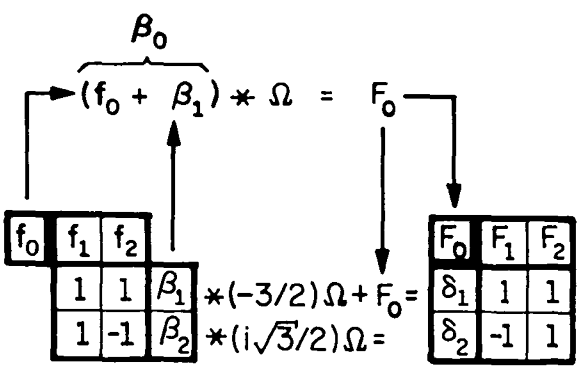
$$\begin{bmatrix} \frac{a_0}{2} \\ \\ \frac{a_1}{2} \end{bmatrix} = \frac{\Omega}{2} \begin{bmatrix} W^2 \\ \\ W^1 \end{bmatrix} = \frac{\Omega}{2} \begin{bmatrix} \bar{W}^1 \\ \\ W^1 \end{bmatrix} \tag{4.15}$$

where $\bar{W}$ is the complex conjugate of W. Hence, applying the second order LCT tableau (Fig. 1), we find

$\frac{\Omega}{2}\bar{W}^1$	$\frac{\Omega}{2}W^1$	
1	1	α_1
1	-1	α_2

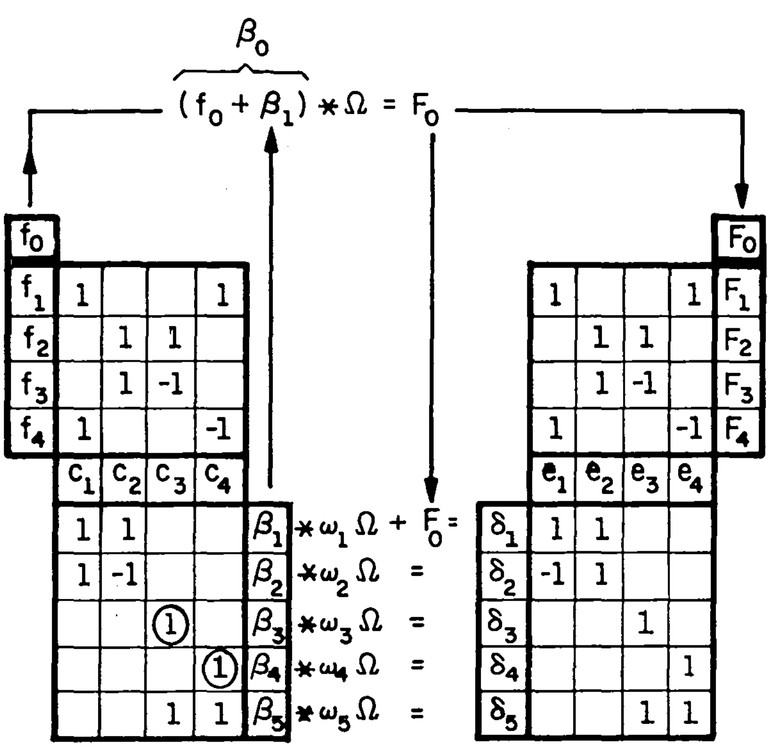
$$= -\frac{\Omega}{2}; \quad \varepsilon_1 = -\frac{1}{2}; \quad \omega_1 = \varepsilon_1 - 1 = -\frac{3}{2}$$

$$= i\frac{\sqrt{3}}{2}\Omega; \quad \varepsilon_2 = i\frac{\sqrt{3}}{2}; \quad \omega_2 = \varepsilon_2 = i\frac{\sqrt{3}}{2}$$



N =3 (3M; 6A)

Fig. 5. Algorithm for DFT of order 3



$\omega_1 = -\frac{5}{4}$
$\omega_2 = -\frac{1}{2}(\cos 36° + \cos 72°)$
$\omega_3 = i(\sin 36° + \sin 72°)$
$\omega_4 = i(\sin 36° - \sin 72°)$
$\omega_5 = -i \sin 36°$

N =5 (6M; 17A)

Fig.6 Algorithm for DFT of order 5

<u>DFT of order 5 (Fig. 6)</u>

$$W = e^{-i\frac{2\pi}{5}}; \quad g = 2. \quad \text{Hence, from (4.3),}$$

$$\frac{1}{4}\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \frac{\Omega}{4}\begin{bmatrix} W^3 \\ W^4 \\ W^2 \\ W^1 \end{bmatrix} = \frac{\Omega}{4}\begin{bmatrix} \overline{W}^2 \\ \overline{W}^1 \\ W^2 \\ W^1 \end{bmatrix} \tag{4.16}$$

The α_i's are determined from the following prescription of Fig. 2:

$$\begin{array}{c|cccc|}
\frac{\Omega}{4}\overline{W}^2 & 1 & & & 1 \\
\frac{\Omega}{4}\overline{W}^1 & & 1 & 1 & \\
\frac{\Omega}{4}W^2 & 1 & & & -1 \\
\frac{\Omega}{4}W^1 & & 1 & -1 & \\
\hline
 & c_1 & c_2 & c_3 & c_4 \\
\hline
 & 1 & 1 & & \\
 & 1 & -1 & & \\
 & & & 1 & \\
 & & & & 1 \\
 & & & 1 & 1 \\
\end{array}
\begin{array}{|c|}
\alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5
\end{array}$$

$$\left\{\begin{aligned}
c_1 &= -\frac{\Omega}{2}\cos 36^o \\
c_2 &= \frac{\Omega}{2}\cos 72^o \\
c_3 &= \frac{i\Omega}{2}\sin 72^o \\
c_4 &= \frac{i\Omega}{2}\sin 36^o
\end{aligned}\right.$$

$$\left.\begin{aligned}
\alpha_1 &= -\frac{\Omega}{4} \\
\alpha_2 &= -\frac{\Omega}{2}(\cos 36^o + \cos 72^o) \\
\alpha_3 &= \frac{i\Omega}{2}\sin 72^o \\
\alpha_4 &= \frac{i\Omega}{2}\sin 36^o \\
\alpha_5 &= \frac{i\Omega}{2}(\sin 36^o + \sin 72^o)
\end{aligned}\right\} \tag{4.17}$$

(4.4) and (4.17) prescribe the ε_i's. Finally, from (4.14) and the β_i multipliers in Fig. 2 we get

$$\left.\begin{aligned}
\omega_1 &= \varepsilon_1 - 1 = -\frac{5}{4} \\
\omega_2 &= \varepsilon_2 = -\frac{1}{2}\,(\cos 36^\circ + \cos 72^\circ) \\
\omega_3 &= 2\varepsilon_5 = i\,(\sin 36^\circ + \sin 72^\circ) \\
\omega_4 &= 2(\varepsilon_4 - \varepsilon_3) = i(\sin 36^\circ - \sin 72^\circ) \\
\omega_5 &= -2\varepsilon_4 = -i\,\sin 36^\circ
\end{aligned}\right\} \quad (4.18)$$

<u>DFT of order 7 (Fig. 8)</u>

$W = e^{-i\frac{2\pi}{7}}$; $g = 3$. Hence from (4.3)

$$\frac{1}{6}\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \frac{\Omega}{6}\begin{bmatrix} W^5 \\ W^4 \\ W^6 \\ W^2 \\ W^3 \\ W^1 \end{bmatrix} = \frac{\Omega}{6}\begin{bmatrix} \overline{W}^2 \\ \overline{W}^3 \\ \overline{W}^1 \\ W^2 \\ W^3 \\ W^1 \end{bmatrix} \quad (4.19)$$

All α_i's are expressible in terms of the angle

$$\theta = \frac{90^\circ}{7} \quad (4.20)$$

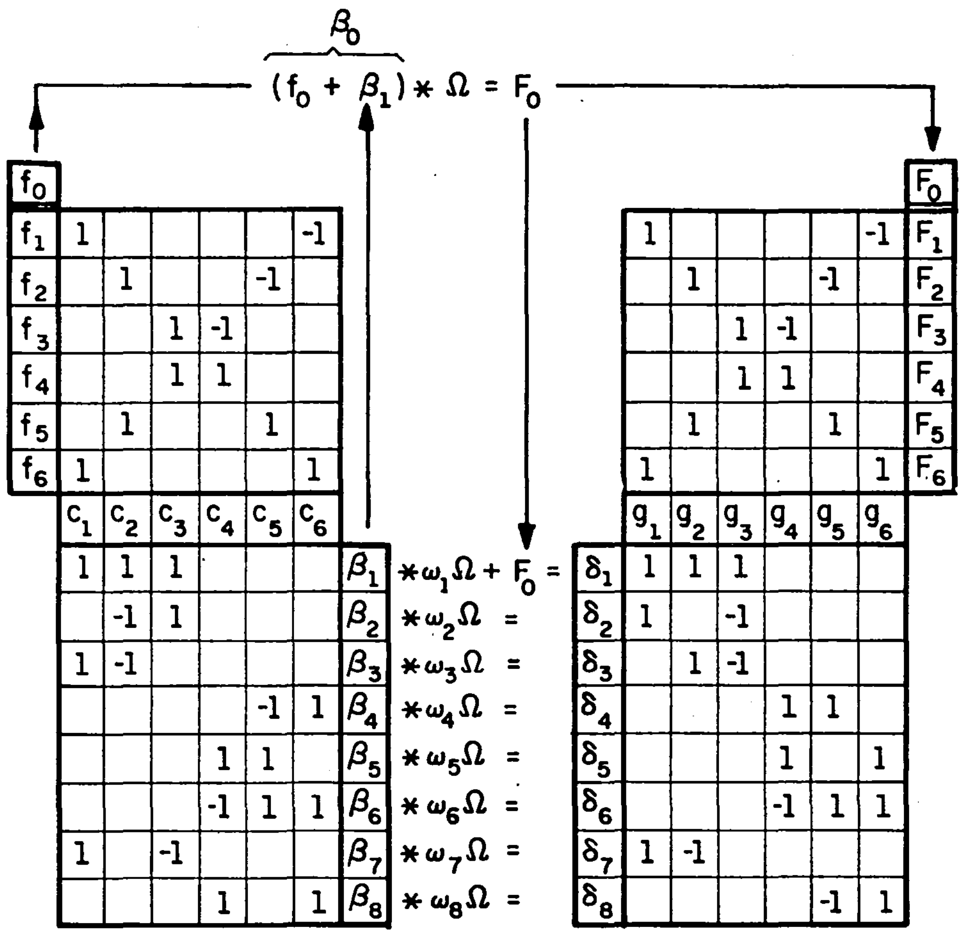
and its multiples. The prescription of Fig. 4 for the computation of the α_i's is shown in Fig. 7 and the final tableau based on that is shown in Fig. 8.

	c_1	c_2	c_3	c_4	c_5	c_6
$\frac{\Omega}{6}\overline{W}^2$	1					-1
$\frac{\Omega}{6}\overline{W}^3$			1	-1		
$\frac{\Omega}{6}\overline{W}^1$		1			-1	
$\frac{\Omega}{6}W^2$	1					1
$\frac{\Omega}{6}W^3$			1	1		
$\frac{\Omega}{6}W^1$		1			1	

$$
\left\{
\begin{aligned}
c_1 &= -\frac{\Omega}{3}\sin\theta \\
c_2 &= \frac{\Omega}{3}\cos 4\theta \\
c_3 &= -\frac{\Omega}{3}\cos 2\theta \\
c_4 &= -i\frac{\Omega}{3}\sin 2\theta \\
c_5 &= -i\frac{\Omega}{3}\sin 4\theta \\
c_6 &= -i\frac{\Omega}{3}\cos\theta
\end{aligned}
\right.
$$

c_1	c_2	c_3	c_4	c_5	c_6			
1	1	1				α_1	$=-\frac{\Omega}{6}$	; $\omega_1 = \frac{1}{\Omega}\alpha_1 - 1 = -\frac{7}{6}$
	-1	1				α_2	$=-\frac{\Omega}{3}(\cos 2\theta + \cos 4\theta)$	; $\omega_2 = -\frac{1}{\Omega}(\alpha_3 + \alpha_7) = \frac{1}{3}(2\sin\theta - \cos 2\theta + \cos 4\theta)$
1	-1					α_3	$=-\frac{\Omega}{3}(\sin\theta + \cos 4\theta)$	; $\omega_3 = \frac{1}{\Omega}(\alpha_7 - \alpha_2) = \frac{1}{3}(-\sin\theta + 2\cos 2\theta + \cos 4\theta)$
				-1	1	α_4	$=-i\frac{\Omega}{3}(\cos\theta - \sin 4\theta)$	; $\omega_4 = \frac{1}{\Omega}(\alpha_8 + \alpha_5) = -\frac{i}{3}(\cos\theta + 2\sin 2\theta + \sin 4\theta)$
			1	1		α_5	$=-i\frac{\Omega}{3}(\sin 2\theta + \sin 4\theta)$	; $\omega_5 = \frac{1}{\Omega}(\alpha_8 + \alpha_4) = -\frac{i}{3}(2\cos\theta + \sin 2\theta - \sin 4\theta)$
			-1	1	1	α_6	$=-i\frac{\Omega}{3}(\cos\theta - \sin 2\theta + \sin 4\theta)$	; $\omega_6 = \frac{1}{\Omega}\alpha_6 = -\frac{i}{3}(\cos\theta - \sin 2\theta + \sin 4\theta)$
1		-1				α_7	$=-\frac{\Omega}{3}(\sin\theta - \cos 2\theta)$	; $\omega_7 = -\frac{1}{\Omega}(\alpha_2 + \alpha_3) = \frac{1}{3}(\sin\theta + \cos 2\theta + 2\cos 4\theta)$
			1		1	α_8	$=-i\frac{\Omega}{3}(\cos\theta + \sin 2\theta)$	; $\omega_8 = \frac{1}{\Omega}(\alpha_5 - \alpha_4) = -\frac{i}{3}(-\cos\theta + \sin 2\theta + 2\sin 4\theta)$

Fig. 7. Computation of ω_i's for the DFT algorithm of order 7



N = 7 (9M;36A)

$\theta = \frac{90^\circ}{7}$

$\omega_1 = -\frac{7}{6}$; $\omega_5 = -\frac{i}{3}(2\cos\theta + \sin 2\theta - \sin 4\theta)$

$\omega_2 = \frac{1}{3}(2\sin\theta - \cos 2\theta + \cos 4\theta)$; $\omega_6 = -\frac{i}{3}(\cos\theta - \sin 2\theta + \sin 4\theta)$

$\omega_3 = \frac{1}{3}(-\sin\theta + 2\cos 2\theta + \cos 4\theta)$; $\omega_7 = \frac{1}{3}(\sin\theta + \cos 2\theta + 2\cos 4\theta)$

$\omega_4 = -\frac{i}{3}(\cos\theta + 2\sin 2\theta + \sin 4\theta)$; $\omega_8 = -\frac{i}{3}(-\cos\theta + \sin 2\theta + 2\sin 4\theta)$

Fig. 8. Algorithm for DFT of order 7

V. The Basic DFT Algorithms for $N=4,9$

From (1.2) we see that with the tableaus of the last section, the highest realizable N would be $105(=3\cdot 5\cdot 7)$. With this in mind, we add now the trivial algorithm for $N=2$ (Fig. 9) thus increasing the maximal N to 210.

If we want still higher N, we may either generate new tableaus for successively higher primes (11, 13, 17,...), or devise new tableaus for $N=p^k$ (p prime). We adopt the latter course here.

DFT of order 4 (Fig. 10)

$N=2^2$, yielding (2.7),

$$n=2 \tag{5.1}$$

The primitive root of 4 is 3. Hence,

ρ	0	1
$r=3^\rho \bmod 4$	1	3

(5.2)

The number of rows excluded from the LC pattern is 2. This number increases still further in the subsequent tableaus, reaching 8 for $N=16$. This calls for some new and some modified terminology to facilitate handling the non-LC part of the matrix.

First we extend the definition of r so that (2.10) will now include (2.11)

ρ	0	1	(0)	(2)
r	1	3	0	2

(5.3)

Similarly, for the column indices σ, s, we now have

σ	0	1	(0)	(2)
s	1	3	0	2

(5.4)

In subsequent derivations, here and in the next section, we shall assume without explicitly so stating that the definitions of r,s have been extended, as indicated here, to cover all indices. Next, we complement (2.13)-(2.15) with

f_0	f_1			
1	1	β_0	$* \Omega =$	F_0
1	-1	β_1	$* \Omega =$	F_1

N =2 (2M; 2A)

Fig. 9. Algorithm for DFT of order 2

$$F'_r = B'_\rho ; \quad \hat{F}_r = \hat{B}_\rho ; \quad F_r = \hat{F}_r + F'_r \tag{5.5}$$

Finally, recalling that the (r,s) element of the DFT matrix (2.4) is ΩW^{rs}, we introduce the "exponent matrix" E,

$$E_{r,s} = (r\,s) \bmod N \tag{5.6}$$

which we find very convenient in accounting for the contribution of the non-LC part.

In the present case

$$\begin{array}{ccccc} s \rightarrow & 0 & 2 & 1 & 3 \\ \sigma \rightarrow & (0) & (2) & 0 & 1 \end{array}$$

$$E = \left[\begin{array}{cc|cc} 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ \hline 0 & 2 & 1 & 3 \\ 0 & 2 & 3 & 1 \end{array}\right] \quad \begin{array}{cc} (0) & 0 \\ (2) & 2 \\ 0 & 1 \\ 1 & 3 \\ \uparrow & \uparrow \\ \rho & r \end{array} \tag{5.7}$$

Note that we have added here both kinds of row and column indices. One simple way of obtaining E is directly from its definition (5.6). In writing it down, we make sure that the non-bracketed ρ,σ indices would follow the sequence 0,1,2,... . This will bring out the LC structure. The sequence of the other indices is immaterial.

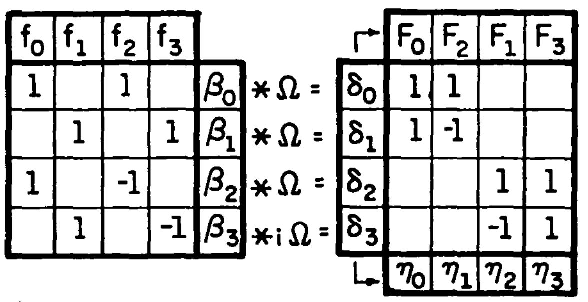
We observe that (5.7) displays a second order LC submatrix as was to be expected. We handle the effect of this submatrix with the LCT tableau of Fig. 1, starting with the evaluation of the α_i's

$$\frac{1}{2}\begin{bmatrix} a_0 \\ \\ a_1 \end{bmatrix} = \frac{\Omega}{2}\begin{bmatrix} W^3 \\ \\ W^1 \end{bmatrix} = \frac{\Omega}{2}\begin{bmatrix} i \\ \\ -i \end{bmatrix} \tag{5.8}$$

Hence (From Fig. 1)

$$\alpha_1 = 0; \quad \alpha_2 = i\Omega \tag{5.9}$$

We conclude that only the β_2 row contributes to F'_1 and we copy it into the β_3 row of Fig. 10.



N = 4 (4M; 8A)

Fig. 10. Algorithm for DFT of order 4

Note that the output vector F appears scrambled in Fig. 10. In section IX we point out the advantage of certain tableau structures which allow storage economies in implementing the algorithm. The tableaus generated up to this point have both this desirable structure and an unscrambled output F. From here on, it seems impossible to realize both these desirable features simultaneously and we opt for the more important memory conserving structure. Hence, the scrambled output. The scrambling here is quite simple but becomes complex for $N = 16$. To facilitate prescribing and handling of this, we have added the vector η to the affected tableaus. The scrambled F is identical with the unscrambled η. (see Fig. 10)

We return now to the realization of the remainder of the E matrix (5.7)

$$\left.\begin{aligned} F_r - F'_r &= \Omega\underbrace{(f_0 - f_2)}_{\beta_2} = \underbrace{\Omega\beta_2}_{\delta_2} \\ F_r &= F'_r + \delta_2 \end{aligned}\right\} (r = 1,3) \tag{5.10}$$

We have used here the fact that for $N = 4$, $W^2 = -1$. Equations like (5.10) can be read off directly from the E matrix. Such equations will henceforth be presented without any comment.

$$F_2 = \Omega\{\underbrace{(f_0+f_2)}_{\beta_0} - \underbrace{(f_1+f_3)}_{\beta_1}\} = \underbrace{\Omega\beta_0}_{\delta_0} - \underbrace{\Omega\beta_1}_{\delta_1} = \delta_0 - \delta_1$$

$$F_0 = \Omega\{\underbrace{(f_0+f_2)}_{\beta_0} + \underbrace{(f_1+f_3)}_{\beta_1}\} = \underbrace{\Omega\beta_0}_{\delta_0} + \underbrace{\Omega\beta_1}_{\delta_1} = \delta_0 + \delta_1$$

This completes the derivation.

DFT of Order 9 (Fig. 13)

$N = 3^2$. Hence (2.7),

$$n = 6 \tag{5.11}$$

The primitive root of 9 is the primitive root of 3, namely, 2. This leads to

ρ	0	1	2	3	4	5
$r = 2^\rho \bmod 9$	1	2	4	8	7	5

(5.12)

Hence, the following E matrix

$$
\begin{array}{cc}
 & \begin{array}{ccc|cccccc} 0 & 3 & 6 & 1 & 2 & 4 & 8 & 7 & 5 \end{array} \\
 & \begin{array}{ccc|cccccc} (0) & (3) & (6) & 0 & 1 & 2 & 3 & 4 & 5 \end{array}
\end{array}
\quad \leftarrow s, \ \leftarrow \sigma
$$

$$
s \to \; 0 \;\; 3 \;\; 6 \;\; 1 \;\; 2 \;\; 4 \;\; 8 \;\; 7 \;\; 5
$$

$$
\sigma \to \; (0) \;\; (3) \;\; (6) \;\; 0 \;\; 1 \;\; 2 \;\; 3 \;\; 4 \;\; 5
$$

$$
E = \left[\begin{array}{ccc|cccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 3 & 6 & 3 & 6 & 3 & 6 \\
0 & 0 & 0 & 6 & 3 & 6 & 3 & 6 & 3 \\
\hline
0 & 3 & 6 & 1 & 2 & 4 & 8 & 7 & 5 \\
0 & 6 & 3 & 2 & 4 & 8 & 7 & 5 & 1 \\
0 & 3 & 6 & 4 & 8 & 7 & 5 & 1 & 2 \\
0 & 6 & 3 & 8 & 7 & 5 & 1 & 2 & 4 \\
0 & 3 & 6 & 7 & 5 & 1 & 2 & 4 & 8 \\
0 & 6 & 3 & 5 & 1 & 2 & 4 & 8 & 7
\end{array}\right]
\qquad
\begin{array}{cc}
(0) & 0 \\
(3) & 3 \\
(6) & 6 \\
0 & 1 \\
1 & 2 \\
2 & 4 \\
3 & 8 \\
4 & 7 \\
5 & 5 \\
\uparrow & \uparrow \\
\rho & r
\end{array}
\tag{5.13}
$$

With $n = 6$, Fig. 4 will be applicable here. We consider the α_i's first. The E matrix (5.13) clearly displays the a_i sequence (this is also in agreement with (4.3)). Hence,

$$
\frac{1}{6}\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \frac{\Omega}{6}\begin{bmatrix} W^5 \\ W^7 \\ W^8 \\ W^4 \\ W^2 \\ W^1 \end{bmatrix} = \frac{\Omega}{6}\begin{bmatrix} \overline{W}^4 \\ \overline{W}^2 \\ \overline{W}^1 \\ W^4 \\ W^2 \\ W^1 \end{bmatrix}
\tag{5.14}
$$

The computation of the α_i's and ω_i's is shown in Fig. 11. Note that, in the terminology of Figs. 4, 11

$$
\alpha_1 = \alpha_6 = 0
\tag{5.15}
$$

This means that in the realization of F_i' which is effected by copying Figs. 4, 11 into Fig. 13, the terms associated with α_1, α_6, should be eliminated.[9] In the actual

[9] Fig. 13 seems to indicate that these terms have not been eliminated. This is misleading as the terms supporting this impression are those added later on. Winograd [3] appears to be unaware of (5.15). This leads to 2 extra multiplications in his algorithm for N = 9.

	c_1	c_2	c_3	c_4	c_5	c_6
$\frac{\Omega}{6}\overline{w}^4$	1					-1
$\frac{\Omega}{6}\overline{w}^2$			1	-1		
$\frac{\Omega}{6}\overline{w}^1$		1			-1	
$\frac{\Omega}{6}w^4$	1					1
$\frac{\Omega}{6}w^2$			1	1		
$\frac{\Omega}{6}w^1$		1			1	

$$\left\{\begin{aligned} c_1 &= -\frac{\Omega}{3}\cos 20^\circ \\ c_2 &= \frac{\Omega}{3}\cos 40^\circ \\ c_3 &= \frac{\Omega}{3}\sin 10^\circ \\ c_4 &= -i\frac{\Omega}{3}\cos 10^\circ \\ c_5 &= -i\frac{\Omega}{3}\sin 40^\circ \\ c_6 &= -i\frac{\Omega}{3}\sin 20^\circ \end{aligned}\right.$$

c_1	c_2	c_3	c_4	c_5	c_6		
1	1	1				α_1	$=\frac{\Omega}{3}(\sin 10^\circ - \cos 20^\circ + \cos 40^\circ) = 0$
	-1	1				α_2	$=\frac{\Omega}{3}(\sin 10^\circ - \cos 40^\circ)$; $\omega_2 = -\frac{1}{\Omega}(\alpha_3+\alpha_7) = \frac{1}{3}(\sin 10^\circ + 2\cos 20^\circ + \cos 40^\circ)$
1	-1					α_3	$=-\frac{\Omega}{3}(\cos 20^\circ + \cos 40^\circ)$; $\omega_3 = \frac{1}{\Omega}(\alpha_7 - \alpha_2) = -\frac{1}{3}(2\sin 10^\circ + \cos 20^\circ - \cos 40^\circ)$
				-1	1	α_4	$=-i\frac{\Omega}{3}(\sin 20^\circ - \sin 40^\circ)$; $\omega_4 = \frac{1}{\Omega}(\alpha_8 + \alpha_5) = -\frac{i}{3}(2\cos 10^\circ + \sin 20^\circ + \sin 40^\circ)$
			1	1		α_5	$=-i\frac{\Omega}{3}(\cos 10^\circ + \sin 40^\circ)$; $\omega_5 = \frac{1}{\Omega}(\alpha_8 + \alpha_4) = -\frac{i}{3}(\cos 10^\circ + 2\sin 20^\circ - \sin 40^\circ)$
			-1	1	1	α_6	$= i\frac{\Omega}{3}(\cos 10^\circ - \sin 20^\circ - \sin 40^\circ) = 0$
1		-1				α_7	$=-\frac{\Omega}{3}(\sin 10^\circ + \cos 20^\circ)$; $\omega_7 = -\frac{1}{\Omega}(\alpha_2 + \alpha_3) = \frac{1}{3}(-\sin 10^\circ + \cos 20^\circ + 2\cos 40^\circ)$
			1		1	α_8	$=-i\frac{\Omega}{3}(\cos 10^\circ + \sin 20^\circ)$; $\omega_8 = \frac{1}{\Omega}(\alpha_5 - \alpha_4) = -\frac{i}{3}(\cos 10^\circ - \sin 20^\circ + 2\sin 40^\circ)$

Fig. 11. Computation of ω_i's for the DFT algorithm of order 9

copying from these figures, we also apply some permutations in addition to the obvious ones involving the input and output variables. These permutations are spelled out in (Fig. 12). Note that the permutation $g_i \to g_j$ is shown in two sequential steps: $g_i \to e_j$; $e_j \to g_j$. (Example: $g_1 \to e_9 \to g_3$)[9a]

Figs. 4, 11 index (i)	0	1	2	3	4	5	6	7	8	
$b_i \to f_j$	1	2	4	8	7	5				Fig. 13 index (j)
$c_i \to c_j$		1	4	2	7	5	8			
$\omega_i \to \omega_j$; $\beta_i \to \beta_j$; $\delta_i \to \delta_j$			4	2	7	5		9	10	
$g_i \to e_j$		9	4	2	7	5	10			
$e_j \to g_j$		3	4	2	7	5	6			
$t_i \to F_j$	5	7	8	4	2	1				

Fig. 12. Index permutations in assembling Fig. 13.

We turn now to account for the rest of the matrix using (5.13) as our guide. In doing that we encounter the following constants

$$\Omega W^3 = \Omega(-\sin 30^\circ - i\cos 30^\circ) = -\Omega\left(\frac{1}{2} + i\,\frac{\sqrt{3}}{2}\right)$$

$$\Omega W^6 = \Omega \overline{W}^3 = -\Omega\left(\frac{1}{2} - i\,\frac{\sqrt{3}}{2}\right)$$

which we use in the following.

<u>r = 1; 4; 7</u>

$$F_r - F_r' = \Omega\{f_0 - \left(\frac{1}{2} + i\frac{\sqrt{3}}{2}\right) f_3 - \left(\frac{1}{2} - i\frac{\sqrt{3}}{2}\right) f_6\}$$

$$= \Omega\{\underbrace{f_0}_{\beta_0} - \frac{1}{2}\underbrace{(f_6+f_3)}_{\beta_3} + i\frac{\sqrt{3}}{2}\underbrace{(f_6-f_3)}_{\beta_6} \quad = \underbrace{\Omega\beta_0}_{\delta_0} - \underbrace{\frac{1}{2}\Omega\beta_3}_{\delta_3} - \underbrace{(-i\frac{\sqrt{3}}{2}\Omega\beta_6)}_{\delta_6}$$

$$= \underbrace{\delta_0 - \delta_3}_{e_3} - \underbrace{\delta_6}_{e_6}$$

9a In the case of $g_3 \to e_2 \to g_2$, we introduce two canceling sign changes.

$$\therefore\ F_r = F_r' + e_3 - e_6$$

We withhold implementation and proceed to the next group

<u>r = 2; 8; 5</u>

Compared to the previous case we have here interchange of W^3 with $W^6(=\overline{W}^3)$. Hence,

$$F_r = F_r' + e_3 + e_6$$

At this point we implement both groups as shown in Fig. 12.

Next we implement F_3, F_6. From (5.13),

$$F_3 = \Omega\left\{(f_0+f_3+f_6) - (\tfrac{1}{2} + i\tfrac{\sqrt{3}}{2})(f_1+f_4+f_7) - (\tfrac{1}{2} - i\tfrac{\sqrt{3}}{2})(f_2+f_8+f_5)\right\}$$

$$= \Omega\left\{\underbrace{f_0}_{\beta_0} + \underbrace{(f_6+f_3)}_{\beta_3} - \tfrac{1}{2}\left[\underbrace{(f_8+f_1)}_{c_1} + \underbrace{(f_7+f_2)}_{c_2} + \underbrace{(f_5+f_4)}_{c_4}\right] + \right.$$

$$\left. + i\tfrac{\sqrt{3}}{2}\left[\underbrace{(f_8-f_1)}_{c_8} - \underbrace{(f_7-f_2)}_{c_7} + \underbrace{(f_5-f_4)}_{c_5}\right]\right\}$$

$$= \underbrace{\Omega\beta_0}_{\delta_0} + \underbrace{\Omega\beta_3}_{2\delta_3} - \tfrac{1}{2}\Omega\underbrace{(c_1+c_2+c_4)}_{\beta_1} + i\tfrac{\sqrt{3}}{2}\,\Omega\underbrace{(c_5-c_7+c_8)}_{\beta_8} = \underbrace{(\delta_0+2\delta_3)}_{e_0} - \underbrace{(\tfrac{1}{2}\Omega\beta_1)}_{\delta_1} - \underbrace{(-i\tfrac{\sqrt{3}}{2}\Omega\beta_8)}_{\delta_8}$$

$$= e_0 - \underbrace{\delta_1}_{e_1} - \underbrace{\delta_8}_{e_8} = \underbrace{e_0 - e_1}_{g_1} - \underbrace{e_8}_{g_8} = g_1 - g_8$$

We turn now to F_6. The only difference here is the interchange of W^3 with $W^6(=\overline{W}^3)$. Hence

$$F_6 = g_1 + g_8$$

Finally,

$$F_0 = \Omega(f_0+f_3+f_6) + \Omega\beta_1 = \underbrace{\Omega\beta_0}_{\delta_0} + \underbrace{\Omega\beta_3}_{2\delta_3} + \underbrace{\Omega\beta_1}_{2\delta_1} = \underbrace{(\delta_0+2\delta_3)}_{e_0} + \underbrace{2\delta_1}_{e_1} = e_0 + 2e_1 = g_0$$

This completes the derivation.

With the two additional tableaus developed in this section, the maximal realizable N has been pushed to 1260 (4·5·7·9). We push it still higher (5040) with the tableaus of the next section.

	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
f_0	1								
f_1		1							-1
f_2			1					-1	
f_3				1			-1		
f_4					1	-1			
f_5					1	1			
f_6				1			1		
f_7			1					1	
f_8		1							1

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8			
1									β_0	$*\Omega$ =	δ_0
	1	1		1					β_1	$*(1/2)\Omega$ =	δ_1
	1			-1					β_2	$*\omega_2\Omega$ =	δ_2
			1						β_3	$*(1/2)\Omega$ =	δ_3
		1		-1					β_4	$*\omega_4\Omega$ =	δ_4
					1		1		β_5	$*\omega_5\Omega$ =	δ_5
						1			β_6	$*(-i\sqrt{3}/2)\Omega$ =	δ_6
					-1			1	β_7	$*\omega_7\Omega$ =	δ_7
					1		-1	1	β_8	$*(-i\sqrt{3}/2)\Omega$ =	δ_8
	1	-1							β_9	$*\omega_9\Omega$ =	δ_9
							1	1	β_{10}	$*\omega_{10}\Omega$ =	δ_{10}

e_0	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}	
1	2										g_0
1	-1										g_1
		-1	1								g_2
			1						1		g_3
			1	1							g_4
					1	1					g_5
						1				1	g_6
						-1	1				g_7
								1			g_8

	e_0	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}
δ_0	1			1							
δ_1		1									
δ_2			1		1						
δ_3	2			-1							
δ_4			1							1	
δ_5								1			1
δ_6							1				
δ_7						1		1			
δ_8									1		
δ_9					-1					1	
δ_{10}						-1					1

	F_0	F_3	F_2	F_1	F_4	F_5	F_8	F_7	F_6
g_0	1								
g_1		1							1
g_2			1					1	
g_3				1			1		
g_4					1	1			
g_5					-1	1			
g_6				-1			1		
g_7			-1					1	
g_8		-1							1
	η_0	η_1	η_2	η_3	η_4	η_5	η_6	η_7	η_8

$\omega_2 = -\frac{1}{3}(2\sin 10° + \cos 20° - \cos 40°)$; $\omega_7 = -\frac{i}{3}(2\cos 10° + \sin 20° + \sin 40°)$

$\omega_4 = \frac{1}{3}(\sin 10° + 2\cos 20° + \cos 40°)$; $\omega_9 = \frac{1}{3}(-\sin 10° + \cos 20° + 2\cos 40°)$

$\omega_5 = -\frac{i}{3}(\cos 10° + 2\sin 20° - \sin 40°)$; $\omega_{10} = -\frac{i}{3}(\cos 10° - \sin 20° + 2\sin 40°)$

N = 9 (11M; 44A)

Fig. 13. Algorithm for DFT of order 9

VI. The Basic DFT Algorithms for $N=8,16$

In developing the tableaus for N = 8, 16, we face a complication due to the fact that

$$N = 2^k \quad (k>2) \tag{6.1}$$

has no primitive roots. In other words, unlike the $N=4$ case, there is no integer whose powers (mod N) would generate all of the odd numbers in the interval (1,N). We can, however, generate half of these with powers (mod N) of the number 3 [9]. We proceed now to modify the relabeling scheme to handle this case. First we modify (2.9) to read as follows:

$$\left.\begin{array}{l} r = 3^{\rho} \bmod N \\ s = 3^{\sigma} \bmod N \end{array}\right\} \quad (\rho,\sigma = 0,1,\ldots \tfrac{N}{4}-1) \tag{6.2}$$

$$\left.\begin{array}{l} r = -3^{\rho} \bmod N \\ s = -3^{\sigma} \bmod N \end{array}\right\} \quad (\rho,\sigma = \overline{0},\overline{1},\ldots \overline{\tfrac{N}{4}-1}) \tag{6.3}$$

Note that we have introduced here (6.3) a new type of index so that we now have three kinds: $(i),\overline{i},i$. We illustrate this with the case $N=8$

ρ	$\overline{0}$	$\overline{1}$	0	1
$r = 3^{\rho} \bmod 8$	–	–	1	3
$r = -3^{\rho} \bmod 8$	7	5	–	–

(6.4)

It should be stressed that the bar or parentheses have no effect on the numerical value of the index. Their only effect is on the choice of the functional relationships connecting the new entities b_{σ},B_{ρ} to the original variables f_s,F_r. Consider for example the expression $g^{i}B_{i}$. If we evaluate it for $i=1$, its value is $g^{1}B_{1} = g^{1}F_{3}$ (see (6.4)). If, on the other hand, we want its value for $i=\overline{1}$ we get $g^{1}B_{\overline{1}} = g^{1}F_{5}$.

Eqns. (2.10), (2.11) now read

$$\left.\begin{array}{ll} F_r = B_{\rho};\ f_s = b_{\sigma} & (r,s \text{ odd}) \\ F_r = B_{(r)};\ f_s = b_{(s)} & (r,s \text{ even}) \end{array}\right\} \tag{6.5}$$

so that the overall r,ρ functional relationship can be summarized as follows

$$\begin{array}{c|cccccccc} \rho & \bar{0} & \bar{1} & 0 & 1 & (0) & (2) & (4) & (6) \\ \hline r & 7 & 5 & 1 & 3 & 0 & 2 & 4 & 6 \end{array} \tag{6.6}$$

with an identical table relating s to σ.

Eqns. (6.2)-(6.5) are now applied to eliminate F_u, f_v from (2.4). (See analogous treatment in section II.)

$$B_{(2t)} = F_{2t} = \Omega \sum_{m=(0),(2)\cdots}^{(N-2)} W^{2mt} b_m + \Omega \sum_{\sigma=\bar{0}}^{\overline{\frac{N}{4}-1}} W^{-2t3^\sigma} b_\sigma + \Omega \sum_{\sigma=0}^{\frac{N}{4}-1} W^{2t3^\sigma} b_\sigma$$

$$(t = 0,1,\ldots\tfrac{N}{2}-1) \tag{6.7}$$

$$B_\rho = \hat{B}_\rho + B'_\rho;\quad \hat{F}_r = \hat{B}_\rho;\quad F'_r = B'_\rho \qquad (r \text{ odd}) \tag{6.8}$$

$$\hat{B}_\rho = \begin{cases} \Omega \displaystyle\sum_{m=(0),(2),\ldots}^{(N-2)} W^{-m3^\rho} b_m & (\rho=\bar{0},\bar{1},\ldots,\overline{\tfrac{N}{4}-1}) \\ \Omega \displaystyle\sum_{m=(0),(2),\ldots}^{(N-2)} W^{m3^\rho} b_m & (\rho=0,1,\ldots,\tfrac{N}{4}-1) \end{cases} \tag{6.9}$$

$$B'_\rho = \begin{cases} \Omega \displaystyle\sum_{\sigma=\bar{0}}^{\overline{\frac{N}{4}-1}} W^{(3^{\rho+\sigma})} b_\sigma + \Omega \sum_{\sigma=0}^{\frac{N}{4}-1} W^{-(3^{\rho+\sigma})} b_\sigma & (\rho=\bar{0},\bar{1},\ldots,\overline{\tfrac{N}{4}-1}) \\ \Omega \displaystyle\sum_{\sigma=\bar{0}}^{\overline{\frac{N}{4}-1}} W^{-(3^{\rho+\sigma})} b_\sigma + \Omega \sum_{\sigma=0}^{\frac{N}{4}-1} W^{(3^{\rho+\sigma})} b_\sigma & (\rho=0,1,\ldots,\tfrac{N}{4}-1) \end{cases} \tag{6.10}$$

It is obvious that all four matrix products appearing in (6.10) are Hankel transformations $(\rho+\sigma)$ of order $\frac{N}{4}$. We prove now that they are also LCT's. To do this, we must show that (see (2.2), (2.16))

$$3^{\rho+\frac{N}{4}-1} = 3^{\rho-1} \bmod N \qquad (N = 2^k;\ k > 2) \tag{6.11}$$

or equivalently,

$$3^{\frac{N}{4}} = 1 \mod N \qquad (N = 2^k;\quad k > 2) \tag{6.12}$$

We prove (6.12) by induction on k. Assume it to be true for $N = n$, then $3^{\frac{n}{4}} = mn+1$ (m integer). Squaring yields $3^{\frac{n}{2}} = (m2\frac{n}{2})(2n) + m(2n)+1$ so that $3^{\frac{(2n)}{4}} = 1 \mod (2n)$ and (6.12) is true for $N = 2n$. Finally, (6.12) is obviously true for $k = 3$.

Thus, (6.10) involves four LC matrix products. Furthermore, these four LC matrices comprise a second order compound matrix which is an LC itself. All these features stand out quite clearly in the two cases to be developed now.

<u>DFT of Order 8 (Fig. 14)</u>

Applying the permutation of (6.6), we get the following E matrix

$$\begin{array}{l}
s \rightarrow \quad 0 \quad 4 \quad 2 \quad 6 \quad 7 \quad 5 \quad 1 \quad 3 \\
\sigma \rightarrow \; (0)(4)(2)(6)\; \bar{0} \quad \bar{1} \quad 0 \quad 1
\end{array}$$

$$E = \left[\begin{array}{cccc|cccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 4 & 4 & 4 & 4 \\
0 & 0 & 4 & 4 & 6 & 2 & 2 & 6 \\
0 & 0 & 4 & 4 & 2 & 6 & 6 & 2 \\
\hline
0 & 4 & 6 & 2 & 1 & 3 & 7 & 5 \\
0 & 4 & 2 & 6 & 3 & 1 & 5 & 7 \\
\cline{5-8}
0 & 4 & 2 & 6 & 7 & 5 & 1 & 3 \\
0 & 4 & 6 & 2 & 5 & 7 & 3 & 1
\end{array}\right]
\begin{array}{cc}
(0) & 0 \\
(4) & 4 \\
(2) & 2 \\
(6) & 6 \\
\bar{0} & 7 \\
\bar{1} & 5 \\
0 & 1 \\
1 & 3 \\
\uparrow & \uparrow \\
\rho & r
\end{array} \tag{6.13}$$

We turn now to an explicit representation of the submatrix corresponding to the lower right quarter of E. Denoting

$$w_k = \Omega W^k, \tag{6.14}$$

we get

$$\begin{array}{c}
\hat{t}_1\left\{\begin{array}{c} \\ \\ \end{array}\right. \\
\hat{t}_0\left\{\begin{array}{c} \\ \\ \end{array}\right.
\end{array}
\begin{bmatrix} F'_7 \\ F'_5 \\ \hline F'_1 \\ F'_3 \end{bmatrix}
=
\begin{array}{c}
\overbrace{\hspace{3em}}^{\hat{a}_1} \quad \overbrace{\hspace{3em}}^{\hat{a}_0} \\
\left[\begin{array}{cc|cc}
w_1 & w_3 & w_7 & w_5 \\
w_3 & w_1 & w_5 & w_7 \\
\hline
w_7 & w_5 & w_1 & w_3 \\
w_5 & w_7 & w_3 & w_1
\end{array}\right]
\end{array}
\begin{bmatrix} f_7 \\ f_5 \\ \hline f_1 \\ f_3 \end{bmatrix}
\begin{array}{c}
\left.\begin{array}{c} \\ \\ \end{array}\right\}\hat{b}_0 \\
\left.\begin{array}{c} \\ \\ \end{array}\right\}\hat{b}_1
\end{array} \tag{6.15}$$

We see here four second-order LC submatrices, two each, of types $\hat{a}_0$, $\hat{a}_1$. Using the terminology introduced here, we write down the equivalent second-order compound matrix

$$\begin{bmatrix} \hat{t}_1 \\ \hat{t}_0 \end{bmatrix} = \begin{bmatrix} \hat{a}_1 & \hat{a}_0 \\ \hat{a}_0 & \hat{a}_1 \end{bmatrix} \begin{bmatrix} \hat{b}_0 \\ \hat{b}_1 \end{bmatrix} \tag{6.16}$$

Note that (6.16) is also an LCT. We propose now to evaluate (6.16) through a direct application of the tableau of Fig. 1. Some elaboration is in order here. The tableaus in this paper have been derived with the implied assumption that the variables appearing in them are all scalars. This is not really necessary. Reviewing the derivations, one concludes that the tableaus are also valid under the following generalized interpretation:

1. The row (column) elements ($f_i, c_i, \beta_i, \delta_i, e_i, F_i, \ldots$ etc.) are column submatrices
2. a_i, α_i, Ω are square matrices
3. ω_i is still a scalar constant
4. The tableau entries

$$\beta_i * \alpha_i = \delta_i; \quad \beta_i * \omega_i \Omega = \delta_i$$

should be interpreted as the following matrix products[10]

$$\alpha_i \beta_i = \delta_i; \quad \omega_i \Omega \beta_i = \delta_i$$

We introduce this generalization here with the immediate goal of evaluating (6.16). However, its importance transcends this immediate application. It is this generalization which elevates the tableaus from the rather theoretical realm of fast algorithms for very low order DFT's into the very practical realm of high-order, high-speed DFT algorithms. The specific way in which this is done is presented in detail in section VII.

We return now to the evaluation of (6.16). Applying Fig. 1 we get

$$\begin{array}{|cc|c|ccc|c|cc|} \hline \hat{b}_0 & \hat{b}_1 & & & & & & \hat{t}_1 & \hat{t}_0 \\ \hline 1 & 1 & \hat{\beta}_1 & * & \hat{\alpha}_1 & = & \hat{\delta}_1 & 1 & 1 \\ 1 & -1 & \hat{\beta}_2 & * & \hat{\alpha}_2 & = & \hat{\delta}_2 & -1 & 1 \\ \hline \end{array} \tag{6.17}$$

[10]This "modified interpretation" can be avoided by using row matrices instead of column matrices. Note in this context that the Ω matrix we will be concerned with is symmetric.

$$\begin{array}{|cc|c}
\hline
\frac{1}{2}\,\hat{a}_0 & \frac{1}{2}\,\hat{a}_1 & \\
\hline
1 & 1 & \hat{\alpha}_1 \\
1 & -1 & \hat{\alpha}_2 \\
\hline
\end{array} \tag{6.18}$$

Hence, (denoting $\gamma = \frac{1}{\sqrt{2}}$),

$$\hat{\alpha}_1 = \frac{\Omega}{2}\begin{bmatrix} W^7+W^1 & W^5+W^3 \\ W^5+W^3 & W^7+W^1 \end{bmatrix} = \gamma\,\Omega \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}; \quad \hat{\beta}_1 = \begin{bmatrix} f_7+f_1 \\ f_5+f_3 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_3 \end{bmatrix}$$

$$\hat{\alpha}_2 = \frac{\Omega}{2}\begin{bmatrix} W^7-W^1 & W^5-W^3 \\ W^5-W^3 & W^7-W^1 \end{bmatrix} = i\,\gamma\,\Omega \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}; \quad \hat{\beta}_2 = \begin{bmatrix} f_7-f_1 \\ f_5-f_3 \end{bmatrix} = \begin{bmatrix} c_7 \\ c_5 \end{bmatrix}$$

$$\hat{\delta}_1 = \hat{\alpha}_1\hat{\beta}_1 = \gamma\Omega\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} c_1 \\ c_3 \end{bmatrix} = \gamma\,\Omega\begin{bmatrix} c_1-c_3 \\ c_3-c_1 \end{bmatrix} = \gamma\,\Omega\begin{bmatrix} \beta_1 \\ -\beta_1 \end{bmatrix} = \begin{bmatrix} \delta_1 \\ -\delta_1 \end{bmatrix}$$

$$\hat{\delta}_2 = \hat{\alpha}_2\hat{\beta}_2 = i\gamma\Omega\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} c_7 \\ c_5 \end{bmatrix} = i\gamma\Omega\begin{bmatrix} c_5+c_7 \\ c_5+c_7 \end{bmatrix} = i\gamma\Omega\begin{bmatrix} \beta_7 \\ \beta_7 \end{bmatrix} = \begin{bmatrix} \delta_7 \\ \delta_7 \end{bmatrix}$$

$$\begin{bmatrix} F'_7 \\ F'_5 \end{bmatrix} = \hat{t}_1 = \hat{\delta}_1-\hat{\delta}_2 = \begin{bmatrix} \delta_1-\delta_7 \\ -(\delta_1+\delta_7) \end{bmatrix} = \begin{bmatrix} g_7 \\ -g_1 \end{bmatrix}; \quad \begin{bmatrix} F'_1 \\ F'_3 \end{bmatrix} = \hat{t}_0 = \hat{\delta}_1+\hat{\delta}_2 = \begin{bmatrix} \delta_1+\delta_7 \\ -(\delta_1-\delta_7) \end{bmatrix} = \begin{bmatrix} g_1 \\ -g_7 \end{bmatrix}$$

We turn now to the realization of the remaining parts of (6.13)

$\underline{r = 1;5}$

$$F_r = F'_r + \Omega\{\underbrace{(f_0-f_4)}_{c_4} + i\underbrace{(f_6-f_2)}_{c_6}\} = F'_r + \Omega\underbrace{(c_4+ic_6)}_{\beta_4} = F'_r + \underbrace{\Omega\beta_4}_{g_4} = F'_r+g_4$$

$\underline{r = 3;7}$

$$F_r = F'_r + \Omega\underbrace{(c_4-ic_6)}_{\beta_6} = F'_r+g_6$$

-----

	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7
c_0	1				1			
c_1		1						1
c_2			1				1	
c_3				1		1		
c_4	1				-1			
c_5				-1		1		
c_6			-1				1	
c_7		-1						1

	e_0	e_1	e_2	e_3	e_4	e_5	e_6	e_7
c_0	1		1					
c_1		1		1				
c_2	1		-1					
c_3		-1		1				
c_4					1		1	
c_5						1		1
c_6					i		-i	
c_7						-1		1

e_0	e_1	e_2	e_3	e_4	e_5	e_6	e_7			
1			1					β_0	$*\Omega$ =	δ_0
	1							β_1	$*(1/\sqrt{2})\Omega$ =	δ_1
		1			-i			β_2	$*\Omega$ =	δ_2
1			-1					β_3	$*\Omega$ =	δ_3
				1				β_4	$*\Omega$ =	δ_4
		1			i			β_5	$*\Omega$ =	δ_5
						1		β_6	$*\Omega$ =	δ_6
							1	β_7	$*(i/\sqrt{2})\Omega$ =	δ_7

	g_0	g_1	g_2	g_3	g_4	g_5	g_6	g_7	
η_0	1								F_0
η_1		1			1				F_1
η_2			1						F_2
η_3				1					F_4
η_4		-1			1				F_5
η_5						1			F_6
η_6							1	-1	F_3
η_7							1	1	F_7

	g_0	g_1	g_2	g_3	g_4	g_5	g_6	g_7
δ_0	1							
δ_1		1						1
δ_2			1					
δ_3				1				
δ_4					1			
δ_5						1		
δ_6							1	
δ_7		1						-1

N = 8 (8M; 26A)

Fig. 14. Algorithm for DFT of order 8

$$F_6 = \Omega\{\underbrace{(f_0+f_4)}_{c_0} - \underbrace{(f_2+f_6)}_{c_2} - i\underbrace{(f_7-f_1)}_{c_7} + i\underbrace{(f_5-f_3)}_{c_5}\} = \Omega\{\underbrace{(c_0-c_2)}_{e_2} + i\underbrace{(c_5-c_7)}_{e_5}\}$$

$$F_6 = \Omega\underbrace{(e_2+ie_5)}_{\beta_5} = \Omega\beta_5 = g_5$$

$$F_2 = \Omega\underbrace{(e_2-ie_5)}_{\beta_2} = \Omega\beta_2 = g_2$$

$$F_4 = \Omega\{\underbrace{(f_0+f_4)}_{c_0} + \underbrace{(f_6+f_2)}_{c_2} - \underbrace{(f_7+f_1)}_{c_1} - \underbrace{(f_5+f_3)}_{c_3}\} = \Omega\{\underbrace{(c_0+c_2)}_{e_0} - \underbrace{(c_1+c_3)}_{e_3}\}$$

$$= \Omega\underbrace{(e_0-e_3)}_{\beta_3} = \Omega\beta_3 = g_3$$

$$F_0 = \Omega\underbrace{(e_0+e_3)}_{\beta_0} = \Omega\beta_0 = g_0$$

This concludes the derivation.

DFT of Order 16 (Fig. 17)

The permutation is controlled by the following table

ρ	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	0	1	2	3
$r = 3^\rho \bmod 16$	–	–	–	–	1	3	9	11
$r = -3^\rho \bmod 16$	15	13	7	5	–	–	–	–

(6.19)

Applying this we get the following E matrix

$$
\begin{array}{l}
\begin{array}{c|cccccccccccccccc}
s \to & 0 & 4 & 8 & 12 & 2 & 6 & 10 & 14 & 15 & 13 & 7 & 5 & 1 & 3 & 9 & 11 \\
\sigma \to & (0) & (4) & (8) & (12) & (2) & (6) & (10) & (14) & \bar{0} & \bar{1} & \bar{2} & \bar{3} & 0 & 1 & 2 & 3
\end{array}
\\[1em]
E = \left[\begin{array}{cccc|cccc|cccc|cccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 8 & 8 & 8 & 8 & 12 & 4 & 12 & 4 & 4 & 12 & 4 & 12 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\
0 & 0 & 0 & 0 & 8 & 8 & 8 & 8 & 4 & 12 & 4 & 12 & 12 & 4 & 12 & 4 \\
\hline
0 & 8 & 0 & 8 & 4 & 12 & 4 & 12 & 14 & 10 & 14 & 10 & 2 & 6 & 2 & 6 \\
0 & 8 & 0 & 8 & 12 & 4 & 12 & 4 & 10 & 14 & 10 & 14 & 6 & 2 & 6 & 2 \\
0 & 8 & 0 & 8 & 4 & 12 & 4 & 12 & 6 & 2 & 6 & 2 & 10 & 14 & 10 & 14 \\
0 & 8 & 0 & 8 & 12 & 4 & 12 & 4 & 2 & 6 & 2 & 6 & 14 & 10 & 14 & 10 \\
\hline
0 & 12 & 8 & 4 & 14 & 10 & 6 & 2 & 1 & 3 & 9 & 11 & 15 & 13 & 7 & 5 \\
0 & 4 & 8 & 12 & 10 & 14 & 2 & 6 & 3 & 9 & 11 & 1 & 13 & 7 & 5 & 15 \\
0 & 12 & 8 & 4 & 14 & 10 & 6 & 2 & 9 & 11 & 1 & 3 & 7 & 5 & 15 & 13 \\
0 & 4 & 8 & 12 & 10 & 14 & 2 & 6 & 11 & 1 & 3 & 9 & 5 & 15 & 13 & 7 \\
\hline
0 & 4 & 8 & 12 & 2 & 6 & 10 & 14 & 15 & 13 & 7 & 5 & 1 & 3 & 9 & 11 \\
0 & 12 & 8 & 4 & 6 & 2 & 14 & 10 & 13 & 7 & 5 & 15 & 3 & 9 & 11 & 1 \\
0 & 4 & 8 & 12 & 2 & 6 & 10 & 14 & 7 & 5 & 15 & 13 & 9 & 11 & 1 & 3 \\
0 & 12 & 8 & 4 & 6 & 2 & 14 & 10 & 5 & 15 & 13 & 7 & 11 & 1 & 3 & 9
\end{array}\right]
\begin{array}{cc}
(0) & 0 \\
(4) & 4 \\
(8) & 8 \\
(12) & 12 \\
(2) & 2 \\
(6) & 6 \\
(10) & 10 \\
(14) & 14 \\
\bar{0} & 15 \\
\bar{1} & 13 \\
\bar{2} & 7 \\
\bar{3} & 5 \\
0 & 1 \\
1 & 3 \\
2 & 9 \\
3 & 11 \\
\uparrow & \uparrow \\
\rho & r
\end{array}
\end{array}
\qquad (6.20)
$$

In view of the complexity of the present case, we derive the tableau in two stages. F_i', the contribution of the LC submatrices, is considered separately in the intermediate tableau of Fig. 15 which is then copied into the final tableau of Fig. 17.

Following the previous case, we write down explicitly the LC part

	f_1	f_3	f_5	f_7	f_9	f_{11}	f_{13}	f_{15}		e_5	e_7	e_{13}	e_{15}
	1							1	c_1		1		
		1					1		c_3	1			
			1			1			c_5	-1			
				1	1				c_7		-1		
				-1	1				c_9				1
			-1			1			c_{11}			1	
		-1					1		c_{13}			1	
	-1							1	c_{15}				1

e_5	e_7	e_{13}	e_{15}				g_5	g_7	g_{13}	g_{15}
1				β_5	$* \omega_5 \Omega =$	δ_5	1			
	1			β_7	$* \omega_7 \Omega =$	δ_7		1		
		1		β_{13}	$* \omega_{13} \Omega =$	δ_{13}			1	
			1	β_{15}	$* \omega_{15} \Omega =$	δ_{15}				1
1	1			β_{16}	$* \omega_{16} \Omega =$	δ_{16}	1	1		
		1	1	β_{17}	$* \omega_{17} \Omega =$	δ_{17}			1	1

g_5	g_7	g_{13}	g_{15}		F'_{11}	F'_9	F'_3	F'_5	F'_1	F'_{13}	F'_7	F'_{15}
1		1		h_5	-1		1					
	1		1	h_7		-1			1			
-1		1		h_{13}				1		-1		
	-1		1	h_{15}							1	-1

g_5	g_7	g_{13}	g_{15}		F'_{11}	F'_9	F'_3	F'_5	F'_1	F'_{13}	F'_7	F'_{15}
	1			$\hat{\delta}_{10}$					1			1
1				$\hat{\delta}_{11}$			1			1		
	-1			$\hat{\delta}_{12}$		1					1	
-1				$\hat{\delta}_{13}$	1			1				
			1	$\hat{\delta}_{20}$					1			-1
		1		$\hat{\delta}_{21}$			1			-1		
			-1	$\hat{\delta}_{22}$		1					-1	
		-1		$\hat{\delta}_{23}$	1			-1				

Fig. 15. Intermediate tableau for the computation of F'_i for the DFT of order 16

$$\begin{array}{r} \hat{t}_1\left\{\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right. \\ \hat{t}_0\left\{\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right. \end{array}
\left[\begin{array}{c} F'_{15} \\ F'_{13} \\ F'_{7} \\ F'_{5} \\ \hline F'_{1} \\ F'_{3} \\ F'_{9} \\ F'_{11} \end{array}\right] =
\begin{array}{c}
\begin{array}{cc} \hspace{4em}\hat{a}_1\hspace{4em} & \hspace{4em}\hat{a}_0\hspace{4em} \end{array} \\
\left[\begin{array}{cccc|cccc}
w_1 & w_3 & w_9 & w_{11} & w_{15} & w_{13} & w_7 & w_5 \\
w_3 & w_9 & w_{11} & w_1 & w_{13} & w_7 & w_5 & w_{15} \\
w_9 & w_{11} & w_1 & w_3 & w_7 & w_5 & w_{15} & w_{13} \\
w_{11} & w_1 & w_3 & w_9 & w_5 & w_{15} & w_{13} & w_7 \\
\hline
w_{15} & w_{13} & w_7 & w_5 & w_1 & w_3 & w_9 & w_{11} \\
w_{13} & w_7 & w_5 & w_{15} & w_3 & w_9 & w_{11} & w_1 \\
w_7 & w_5 & w_{15} & w_{13} & w_9 & w_{11} & w_1 & w_3 \\
w_5 & w_{15} & w_{13} & w_7 & w_{11} & w_1 & w_3 & w_9
\end{array}\right]
\end{array}
\left[\begin{array}{c} f_{15} \\ f_{13} \\ f_{7} \\ f_{5} \\ \hline f_{1} \\ f_{3} \\ f_{9} \\ f_{11} \end{array}\right]
\begin{array}{l} \left.\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right\} \hat{b}_0 \\ \left.\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right\} \hat{b}_1 \end{array} \tag{6.21}$$

that is,

$$\begin{bmatrix} \hat{t}_1 \\ \hat{t}_0 \end{bmatrix} = \begin{bmatrix} \hat{a}_1 & \hat{a}_0 \\ \hat{a}_0 & \hat{a}_1 \end{bmatrix} \begin{bmatrix} \hat{b}_0 \\ \hat{b}_1 \end{bmatrix} \tag{6.22}$$

We handle (6.22) via the tableaus (6.17), (6.18). It is obvious from (6.18) that $\hat{\alpha}_1, \hat{\alpha}_2$ will be LC matrices. Hence, it is sufficient to compute their first rows only. We express these in terms of the basic angle

$$\theta = \frac{360^\circ}{16} = 22.5^\circ \tag{6.23}$$

$$\left[\text{First row of } \hat{\alpha}_1\right] = \Omega \left[\cos\theta \quad \sin\theta \quad -\cos\theta \quad -\sin\theta\right] \tag{6.24}$$

$$\left[\text{First row of } \hat{\alpha}_2\right] = i\Omega \left[\sin\theta \quad \cos\theta \quad -\sin\theta \quad -\cos\theta\right] \tag{6.25}$$

Turning to (6.17), we get

$$\hat{\beta}_1 = \begin{bmatrix} f_{15}+f_1 \\ f_{13}+f_3 \\ f_7+f_9 \\ f_5+f_{11} \end{bmatrix} = \begin{bmatrix} c_1 \\ c_3 \\ c_7 \\ c_5 \end{bmatrix} ; \quad \hat{\beta}_2 = \begin{bmatrix} f_{15}-f_1 \\ f_{13}-f_3 \\ f_7-f_9 \\ f_5-f_{11} \end{bmatrix} = \begin{bmatrix} c_{15} \\ c_{13} \\ -c_9 \\ -c_{11} \end{bmatrix} \tag{6.26}$$

The next step in (6.17) calls for the evaluation of two LCT's of order 4, namely,

$$\hat{\delta}_i = \hat{\alpha}_i \hat{\beta}_i \quad (i = 1,2) \tag{6.27}$$

We adopt now the following notation for the components of $\hat{\delta}_i$

$$\hat{\delta}_i = \begin{bmatrix} \hat{\delta}_{i0} \\ \hat{\delta}_{i1} \\ \hat{\delta}_{i2} \\ \hat{\delta}_{i3} \end{bmatrix} \tag{6.28}$$

These will be determined now by applying the tableau of Fig. 2.

<u>Implementation of $\hat{\delta}_1 = \hat{\alpha}_1 \hat{\beta}_1$</u>

$-\frac{\Omega}{4}\sin\theta$	1			1		
$-\frac{\Omega}{4}\cos\theta$		1	1			
$\frac{\Omega}{4}\sin\theta$	1			-1		
$\frac{\Omega}{4}\cos\theta$		1	-1			
	0	0	$-\frac{\Omega}{2}\cos\theta$	$-\frac{\Omega}{2}\sin\theta$		
	1	1			0	
	1	-1			0	
			1		$-\frac{\Omega}{2}\cos\theta$	$=\varepsilon_5\Omega;\omega_5=2\varepsilon_{16}=-(\cos\theta+\sin\theta)$
				1	$-\frac{\Omega}{2}\sin\theta$	$=\varepsilon_7\Omega;\omega_7=2(\varepsilon_7-\varepsilon_5)=(\cos\theta-\sin\theta)$
			1	1	$-\frac{\Omega}{2}(\cos\theta+\sin\theta)$	$=\varepsilon_{16}\Omega;\ \omega_{16}=-2\varepsilon_7=\sin\theta$

(6.29)

Note that the vanishing of the first two multipliers means that only a portion of Fig. 2 is required, namely, the portion involving the rows of β_3,β_4,β_5. This is copied into rows of $\beta_5,\beta_7,\beta_{16}$, respectively, in Fig. 15 (Hence the adopted ω indices). Note further that Fig. 15 shows two alternative paths leading from

g_i to F'_r. The upper path should be ignored for now. The preceding discussion has derived that portion of the tableau leading to $\hat{\delta}_{1j}$. We turn now to $\hat{\delta}_{2j}$.

<u>Implementation of $\hat{\delta}_2 = \hat{\alpha}_2\hat{\beta}_2$</u>

$$
\begin{array}{c|cccc|c|l}
-\frac{i\Omega}{4}\cos\theta & 1 & & & 1 & & \\
-\frac{i\Omega}{4}\sin\theta & & 1 & 1 & & & \\
\frac{i\Omega}{4}\cos\theta & 1 & & & -1 & & \\
\frac{i\Omega}{4}\sin\theta & & 1 & -1 & & & \\
\hline
 & 0 & 0 & -\frac{i\Omega}{2}\sin\theta & -\frac{i\Omega}{2}\cos\theta & & \\
\hline
 & 1 & 1 & & & 0 & \\
 & 1 & -1 & & & 0 & \\
 & & & 1 & & -\frac{i\Omega}{2}\sin\theta & =\varepsilon_{13}\Omega;\omega_{13}=2\varepsilon_{17}=-i(\sin\theta+\cos\theta) \\
 & & & & 1 & -\frac{i\Omega}{2}\cos\theta & =\varepsilon_{15}\Omega;\omega_{15}=2(\varepsilon_{15}-\varepsilon_{13})=i(\sin\theta-\cos\theta) \\
 & & & 1 & 1 & -\frac{i\Omega}{2}(\sin\theta+\cos\theta) & =\varepsilon_{17}\Omega;\omega_{17}=-2\varepsilon_{15}=i\cos\theta
\end{array}
\tag{6.30}
$$

The situation here is very similar to the $\hat{\delta}_1$ case. The rows of β_3,β_4,β_5 of Fig. 2 are now copied into the rows of $\beta_{13},\beta_{15},\beta_{17}$, respectively, of Fig. 15.

With $\hat{\delta}_1,\hat{\delta}_2$ now available, we apply (6.17) to obtain $\hat{t}_1,\hat{t}_0$ as shown in the lower part of Fig. 15. The transition from g_i to F'_r is shown there requiring 8 additions. The upper part realizes the same transformation with only 4 additions and is the version copied into Fig. 17. The identity of the two paths can be easily verified by inspection. For example, the upper part prescribes $F'_1 = g_7+g_{15}$ but so does the lower part. Verifying such agreements for all 8 outputs, establishes the identity.

We implement now the remaining parts of (6.20). To bring out clearly the various symmetries we are exploiting here, we show an explicit form of the remaining part of the permuted DFT matrix in Fig. 16. It is shown here for convenience as the sum of two matrices and expressed in terms of the constant

$$\gamma = \frac{1}{\sqrt{2}} \tag{6.31}$$

$$
\begin{array}{c|cccc|cccc|cccc|cccc|}
s \rightarrow & 0 & 4 & 8 & 12 & 2 & 6 & 10 & 14 & 15 & 13 & 7 & 5 & 1 & 3 & 9 & 11 \\
\sigma \rightarrow & (0) & (4) & (8) & (12) & (2) & (6) & (10) & (14) & \bar{0} & \bar{1} & \bar{2} & \bar{3} & 0 & 1 & 2 & 3 \\
\hline
 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & i & -i & i & -i & -i & i & -i & i \\
 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -i & i & -i & i & i & -i & i & -i \\
\hline
 & 1 & -1 & 1 & -1 & -i & i & -i & i & \gamma & -\gamma & \gamma & -\gamma & \gamma & -\gamma & \gamma & -\gamma \\
 & 1 & -1 & 1 & -1 & i & -i & i & -i & -\gamma & \gamma & -\gamma & \gamma & -\gamma & \gamma & -\gamma & \gamma \\
 & 1 & -1 & 1 & -1 & -i & i & -i & i & -\gamma & \gamma & -\gamma & \gamma & -\gamma & \gamma & -\gamma & \gamma \\
Y - Y' = & 1 & -1 & 1 & -1 & i & -i & i & -i & \gamma & -\gamma & \gamma & -\gamma & \gamma & -\gamma & \gamma & -\gamma \\
\hline
 & 1 & i & -1 & -i & \gamma & -\gamma & -\gamma & \gamma & & & & & & & & \\
 & 1 & -i & -1 & i & -\gamma & \gamma & \gamma & -\gamma & & & & & & & & \\
 & 1 & i & -1 & -i & \gamma & -\gamma & -\gamma & \gamma & & & & & & & & \\
 & 1 & -i & -1 & i & -\gamma & \gamma & \gamma & -\gamma & & & & & & & & \\
\hline
 & 1 & -i & -1 & i & \gamma & -\gamma & -\gamma & \gamma & & & & & & & & \\
 & 1 & i & -1 & -i & -\gamma & \gamma & \gamma & -\gamma & & & & & & & & \\
 & 1 & -i & -1 & i & \gamma & -\gamma & -\gamma & \gamma & & & & & & & & \\
 & 1 & i & -1 & -i & -\gamma & \gamma & \gamma & -\gamma & & & & & & & & \\
\end{array}
$$

$$+$$

$$
\begin{array}{|cccc|cccc|cccc|cccc|cc}
0 & 4 & 8 & 12 & 2 & 6 & 10 & 14 & 15 & 13 & 7 & 5 & 1 & 3 & 9 & 11 & & \\
(0) & (4) & (8) & (12) & (2) & (6) & (10) & (14) & \bar{0} & \bar{1} & \bar{2} & \bar{3} & 0 & 1 & 2 & 3 & & \\
\hline
 & & & & & & & & & & & & & & & & (0) & 0 \\
 & & & & & & & & & & & & & & & & (4) & 4 \\
 & & & & & & & & & & & & & & & & (8) & 8 \\
 & & & & & & & & & & & & & & & & (12) & 12 \\
\hline
 & & & & & & & & i\gamma & i\gamma & i\gamma & i\gamma & -i\gamma & -i\gamma & -i\gamma & -i\gamma & (2) & 2 \\
 & & & & & & & & i\gamma & i\gamma & i\gamma & i\gamma & -i\gamma & -i\gamma & -i\gamma & -i\gamma & (6) & 6 \\
 & & & & & & & & -i\gamma & -i\gamma & -i\gamma & -i\gamma & i\gamma & i\gamma & i\gamma & i\gamma & (10) & 10 \\
 & & & & & & & & -i\gamma & -i\gamma & -i\gamma & -i\gamma & i\gamma & i\gamma & i\gamma & i\gamma & (14) & 14 \\
\hline
 & & & & i\gamma & i\gamma & -i\gamma & -i\gamma & & & & & & & & & \bar{0} & 15 \\
 & & & & i\gamma & i\gamma & -i\gamma & -i\gamma & & & & & & & & & \bar{1} & 13 \\
 & & & & i\gamma & i\gamma & -i\gamma & -i\gamma & & & & & & & & & \bar{2} & 7 \\
 & & & & i\gamma & i\gamma & -i\gamma & -i\gamma & & & & & & & & & \bar{3} & 5 \\
\hline
 & & & & -i\gamma & -i\gamma & i\gamma & i\gamma & & & & & & & & & 0 & 1 \\
 & & & & -i\gamma & -i\gamma & i\gamma & i\gamma & & & & & & & & & 1 & 3 \\
 & & & & -i\gamma & -i\gamma & i\gamma & i\gamma & & & & & & & & & 2 & 9 \\
 & & & & -i\gamma & -i\gamma & i\gamma & i\gamma & & & & & & & & & 3 & 11 \\
 & & & & & & & & & & & & & & & & \uparrow & \uparrow \\
 & & & & & & & & & & & & & & & & \rho & r \\
\end{array}
$$

Fig. 16. E matrix for the computation of $F_i - F'_i$ for the DFT of order 16

We define now

$$c_{14} = f_6 - f_{14};\; c_{12} = f_4 - f_{12};\quad c_{10} = f_2 - f_{10};\quad c_8 = f_0 - f_8$$

and proceed with the evaluation as follows:

r = 1;9

$$F_r - F_r' = \Omega\,\underbrace{(c_8 - ic_{12})}_{\beta_{12}} - \gamma\Big[\underbrace{(c_{14}-c_{10})}_{e_{10}} + i\underbrace{(c_{14}+c_{10})}_{e_{14}}\Big] = \underbrace{\Omega\beta_{12}}_{\delta_{12}} - i\gamma\Omega\underbrace{(e_{14} - ie_{10})}_{\beta_{10}}$$

$$= \underbrace{\delta_{12}}_{g_{12}} - \underbrace{i\gamma\Omega\,\beta_{10}}_{\delta_{10}} = g_{12} - \underbrace{\delta_{10}}_{g_{10}} = g_{12} - g_{10} = h_{12}$$

$$\therefore\; F_r = F_r' + h_{12}$$

r = 5;13

$$F_r = F_r' + \underbrace{(g_{12}+g_{10})}_{h_{10}} = F_r' + h_{10}$$

r = 3;11

$$F_r - F_r' = \Omega\Big\{\underbrace{(c_8 + ic_{12})}_{\beta_8} + \gamma\Big[\underbrace{(c_{14}-c_{10})}_{e_{10}} - i\underbrace{(c_{14}+c_{10})}_{e_{14}}\Big]\Big\} = \underbrace{\Omega\beta_8}_{\delta_8} - i\gamma\Omega\underbrace{(e_{14}+ie_{10})}_{\beta_{14}}$$

$$= \underbrace{\delta_8}_{g_8} - \underbrace{i\gamma\Omega\beta_{14}}_{\delta_{14}} = g_8 - \underbrace{\delta_{14}}_{g_{14}} = g_8 - g_{14} = h_8$$

$$F_r = F_r' + h_8$$

r = 7;15

$$F_r = F_r' + \underbrace{(g_8+g_{14})}_{h_{14}} = F_r' + h_{14}$$

We define now

$$c_0 = f_0 + f_8;\quad c_2 = f_2 + f_{10};\quad c_4 = f_4 + f_{12};\quad c_6 = f_6 + f_{14}$$

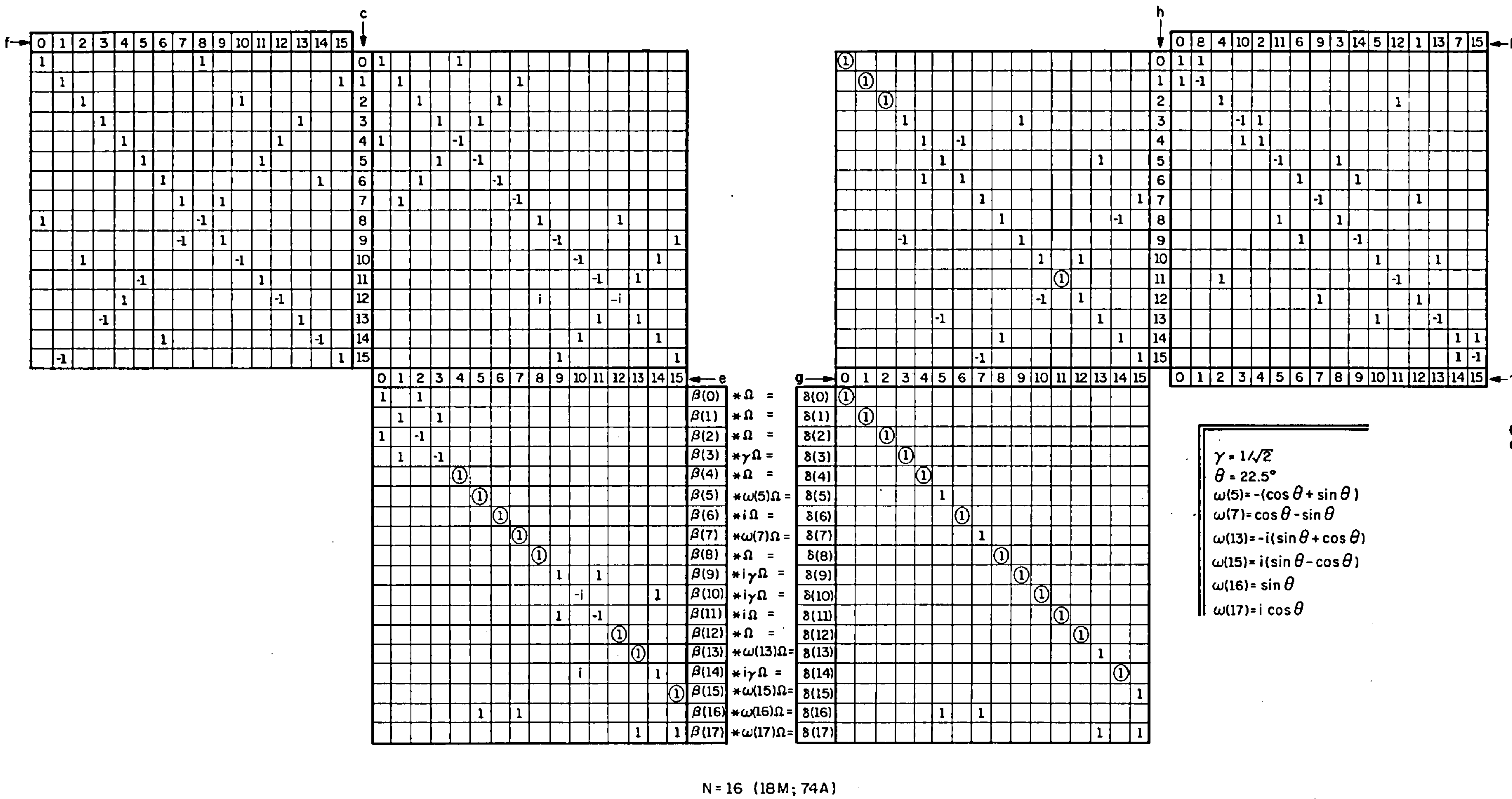


Fig. 17. Algorithm for DFT of order 16

$$g_4 = \delta_4 = \Omega\beta_4 = \Omega(c_0-c_4)$$

and turn to the next group of rows $(F_2, F_6, F_{10}, F_{14})$

$$F_2 = g_4 + \Omega\Big\{-i\underbrace{(c_2-c_6)}_{\beta_6} + \gamma\Big[\underbrace{(c_7+c_1)}_{e_1} - \underbrace{(c_5+c_3)}_{e_3} + i\underbrace{(c_{15}-c_9)}_{e_9} + i\underbrace{(c_{13}-c_{11})}_{e_{11}}\Big]\Big\}$$

$$= g_4 - \underbrace{i\Omega\beta_6}_{\delta_6} + \gamma\Omega\underbrace{(e_1-e_3)}_{\beta_3} + i\gamma\Omega\underbrace{(e_9+e_{11})}_{\beta_9} = g_4 - \underbrace{\delta_6}_{g_6} + \underbrace{\gamma\Omega\beta_3}_{\delta_3} + \underbrace{i\gamma\Omega\beta_9}_{\delta_9} = g_4 - g_6 + \underbrace{\delta_3}_{g_3} + \underbrace{\delta_9}_{g_9}$$

$$= \underbrace{(g_4-g_6)}_{h_4} + \underbrace{(g_9+g_3)}_{h_3} = h_4+h_3$$

Noting which columns in Fig. 16 are associated with each of the four g_i's comprising F_2, we observe that F_6, F_{10}, F_{14} involve the same g_i's but with different sign combinations. Specifically,

$$F_{10} = (g_4-g_6) - (g_9+g_3) = h_4-h_3$$

$$F_6 = \underbrace{(g_4+g_6)}_{h_6} + \underbrace{(g_9-g_3)}_{h_9} = h_6+h_9$$

$$F_{14} = (g_4+g_6) - (g_9-g_3) = h_6-h_9$$

Finally we consider the upper four rows

$$F_0 = \Omega\Big\{\underbrace{(c_0+c_4)}_{e_0} + \underbrace{(c_2+c_6)}_{e_2} + \underbrace{(c_7+c_1)}_{e_1} + \underbrace{(c_5+c_3)}_{e_3}\Big\} = \Omega\Big\{\underbrace{(e_0+e_2)}_{\beta_0} + \underbrace{(e_1+e_3)}_{\beta_1}\Big\}$$

$$= \underbrace{\Omega\beta_0}_{\delta_0} + \underbrace{\Omega\beta_1}_{\delta_1} = \underbrace{\delta_0}_{g_0} + \underbrace{\delta_1}_{g_1} = \underbrace{g_0}_{h_0} + \underbrace{g_1}_{h_1} = h_0 + h_1$$

$$\therefore\ F_8 = h_0 - h_1$$

$$F_4 = \Omega\{\underbrace{(e_0-e_2)}_{\beta_2} + i[\underbrace{(c_{15}-c_9)}_{e_9}-\underbrace{(c_{13}-c_{11})}_{e_{11}}]\} = \underbrace{\Omega\beta_2}_{\delta_2} + i\Omega\underbrace{(e_9-e_{11})}_{\beta_{11}} = \underbrace{\delta_2}_{g_2} + \underbrace{i\Omega\beta_{11}}_{\delta_{11}}$$

$$= \underbrace{g_2}_{h_2} + \underbrace{\delta_{11}}_{g_{11}} = h_2 + \underbrace{g_{11}}_{h_{11}} = h_2 + h_{11}$$

$$\therefore F_{12} = h_2 - h_{11}$$

This completes the derivation. Note that the size of this tableau imposes certain notational peculiarities. In particular, $\omega(i) = \omega_i$, $\beta(i) = \beta_i$, etc.

VII. The DFT Algorithm for $N = \prod_{k=1}^{\kappa} N_k$ [11a]

At this point, we finally are in possession of DFT tableaus for all orders listed in (1.3). Our immediate goal is their integration into a DFT algorithm of order

$$N = \prod_{k=1}^{\kappa} N_k \quad (N_k\text{'s relatively prime}) \tag{7.1}$$

In the subsequent derivations, we will also need the following partial products of these factors

$$\left.\begin{aligned} \nu_i &= \prod_{k=i+1}^{\kappa} N_k \qquad (0 \leq i < \kappa) \\ \nu_\kappa &= 1 \end{aligned}\right\} \tag{7.2}$$

$$n_i = \prod_{\substack{k=1 \\ k\neq i}}^{\kappa} N_k = \frac{N}{N_i} \tag{7.3}$$

Let us introduce now a set of matrices which figure prominently in the development of the algorithm. Consider a matrix whose order N satisfies (7.1). We denote by Ω_i its upper-left submatrix of order ν_i. Note that (7.2) implies the existence of a sequence of these submatrices

$$\Omega_0, \Omega_1, \ldots, \Omega_\kappa$$

The matrix order decreases to the right. Ω_0 on the left is of order N and is thus identical with the overall matrix. Ω_κ on the right is of order 1 and is thus the upper-left element of the overall matrix .

The second item we introduce here is the graphical representation of the tableaus shown in Figure 18. We describe this in terms of the generalized, compound-matrix, interpretation of the tableaus (see discussion following (6.16)). Let the tableau variables $f_i, \beta_i, \delta_i, F_i, \ldots$ etc. represent m-dimensional vectors. Then, the tableau of order N is applicable to any matrix transformation of order mN in which the transforming matrix, Y, when regarded as a compound matrix of order N, is the N-th order DFT matrix (2.4). In this

[11a] The basic idea underlying the developments in this section is commonly attributed to I.J. Good [11]. Here, we find it more convenient to avoid Good's explicit use of the Kronecker matrix product.

case, the Ω parameter of the tableau is the m-th order upper-left submatrix of Y.

Examination of the tableaus reveals that they all consist of three parts corresponding to three distinct phases of the algorithms they describe. In phase 1, the mN-dimensional input vector f is operated upon to yield the m-dimensional β_i vectors. In phase 2, a scalar multiple of the Ω submatrix transforms β_i into ξ_i according to (4.13) $(\xi_i = (\omega_i\Omega)\beta_i;\ i=0,1,\ldots,M-1)$.[11] M is the number of multiplications appearing in the tableau designation. Obviously, this is also the number of β_i vectors generated in phase 1. Finally, in phase 3, the m-dimensional ξ_i vectors are operated upon to yield the mN-dimensional output vector F. The three phases are represented schematically in Fig. 18. The conventions adopted here are as follows: All lines represent vectors. We may attach an integer to a line to indicate the dimensionality of the vector it represents (see Fig. 21). Phase 1 is represented by a circle, phase 3 by a square. In either case, the symbol inside designates the matrix transforming the "circle input" to the "square output". Note that the only arithmetic operations involved inside either the circle or the square, are additions and subtractions.

With these preliminaries out of the way, we are ready now to consider a set of scrambling guidelines that would allow a simple implementation of the DFT of order N satisfying (7.1). Denoting the scrambled matrix by Y, we propose the following set of sufficient conditions:

1. $Y(=\Omega_0)$ should be a compound DFT matrix of order N_1. This will allow the implementation of phase 1 of the N_1 tableau. Phase 2 calls for the determination of $(\omega_i\Omega_1)\beta_i$. Hence,
2. Ω_1 should be a compound DFT matrix of order N_2. We apply phase 1 of the N_2 tableau and are led, in phase 2 to a transformation involving Ω_2. Hence,
3. Ω_2 should be a compound DFT matrix of order N_3 and so on down to Ω_{K-1} which should be a DFT matrix of order N_K.

---

[11]For most i values, $\xi_i = \delta_i$ of the tableaus but not for all of them. For example, the N = 5 tableau implies $\xi_0 = F_0$, $\delta_1 = \xi_1+\xi_0$.

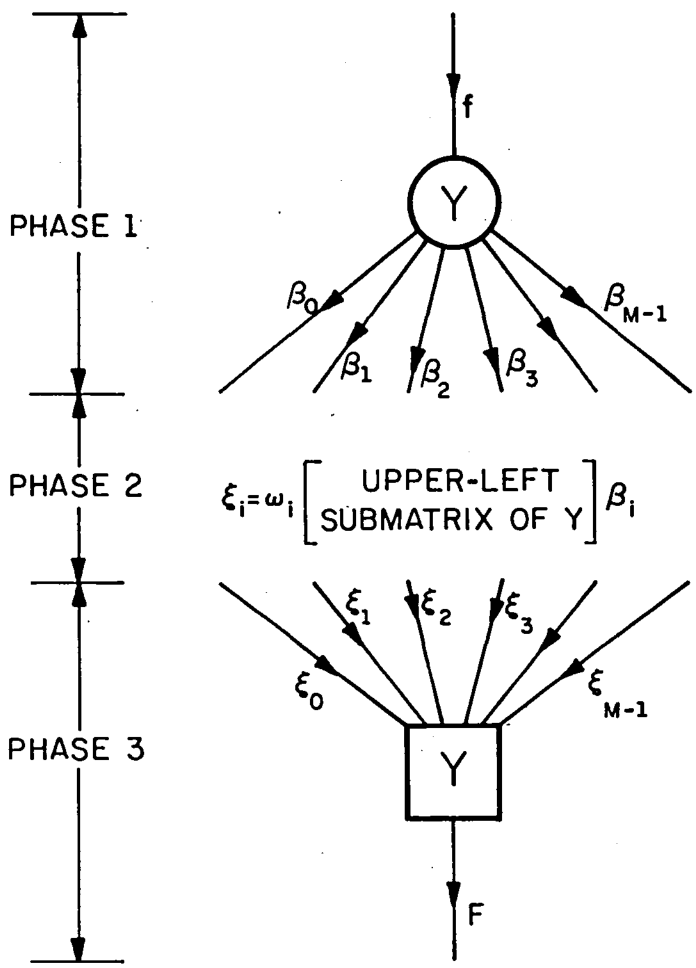


Fig. 18. Schematic representation of the basic DFT tableaus

All this can be summarized as follows: A convenient relabeling scheme would be one which satisfies the following set of κ constraints:

$$\left.\begin{array}{l}\text{Submatrix } \Omega_{k-1} \text{ of Y should be} \\ \text{a compound DFT matrix of order} \\ N_k. \text{ This should hold for all} \\ \qquad 1 \leq k \leq \kappa\end{array}\right\} \tag{7.4}$$

While an algorithm following the above outline would be quite convenient to implement, it is not at all clear that a scrambling scheme that would simultaneously satisfy all κ constraints of (7.4) does, in fact, exist.

We proceed now to develop a relabeling scheme which comes very close to (7.4), adding only a minor complication to the above algorithm outline. We start with the standard DFT matrix ((2.4) with $\Omega = 1$).[12]

$$\hat{F}_u = \sum_{v=0}^{N-1} W^{uv}\hat{f}_v \qquad (u=0,1,\ldots,N-1). \tag{7.5}$$

Let

$$Q_m = \hat{F}_u; \quad q_n = \hat{f}_v \tag{7.6}$$

where

$$u = \lambda(m); \; v = \lambda(n) \qquad (m,n = 0,1,\ldots,N-1) \tag{7.7}$$

and the function λ is yet to be specified. This transforms (7.5) into

$$Q_m = \sum_{n=0}^{N-1} W^{\lambda(m)\lambda(n)} q_n = \sum_{n=0}^{N-1} Y_{mn}q_n \quad (m=0,1,\ldots,N-1) \tag{7.8}$$

The function λ will be defined in terms of a modular representation [7] of u,v. In other words, the relabeling depends on the entities

$$\left.\begin{array}{l} u_k = u \bmod N_k \\ v_k = v \bmod N_k \end{array}\right\} \quad (k=1,2,\ldots,\kappa) \tag{7.9}$$

According to the Chinese Remainder Theorem [7], these remainders uniquely determine any $0 \leq (u,v) < N$. Hence, we may adopt the following representation for u,v

---

[12] We use here $\hat{F}_u, \hat{f}_v$ to distinguish these entities from the tableau variables F_u, f_v.

$$\left.\begin{array}{l} u = (u_1,u_2,\dots,u_\kappa) \\ v = (v_1,v_2,\dots,v_\kappa) \end{array}\right\} \tag{7.10}$$

The function λ is now defined in terms of the following combination of (7.7), (7.10)

$$v = (v_1,v_2,\dots,v_\kappa) = \lambda(n) \tag{7.11}$$

$\lambda(n)$ should be such that as n follows the sequence $0,1,\dots,N-1$, each v_k should follow a periodic repetition of the sequence $0,1,\dots,N_k-1$, starting with $v_k = 0$. v_k should be stepped with every ν_k-increment of n (see (7.2)). This means that v_1 varies very slowly, v_2 varies faster and so on, up to v_κ which varies in step with n.

To illustrate this scrambling, consider the following example

$$\kappa = 3;\ N_1 = 8;\ N_2 = 3;\ N_3 = 5;\ \therefore N = 120 \tag{7.12}$$

for which part of the index sequence would look as follows:

n	v
⋮	⋮
3	48 = (0,0,3)
4	24 = (0,0,4)
5	40 = (0,1,0)
6	16 = (0,1,1)
⋮	⋮
62	12 = (4,0,2)
63	108 = (4,0,3)
64	84 = (4,0,4)
65	100 = (4,1,0)
66	76 = (4,1,1)
⋮	⋮

One could use modular arithmetic subroutines to determine this sequence on a computer. Alternatively, it could be determined in a scheme using neither computers nor computations. All that is called for is some tedious writing down of repetitive sequences. We illustrate this method for our example (7.12) in Tables 2,3. First we write down the sequence $0,1,\ldots,N-1$ as shown in Table 2 under the heading v. Then we write down next to this column, under the heading v_k, a periodic repetition of the sequence $0,1,\ldots,N_k-1$ and repeat this for all $1 \leq k \leq \kappa$. This yields representation (7.10) for all v's. The particular arrangement in Table 2 saves some writing and is convenient for the next step which is just a reordering of Table 2 in the desired sequence.

To simplify the process, we let Table 2 determine the order in which Table 3 is being filled in. For example, the first v values copied from Table 2 into Table 3 are $0,40,80,8,48,88,16,56,\ldots$ etc. Table 3 then prescribes the index sequence of the scrambled input vector for (7.12), namely,

$$\hat{f}_0, \hat{f}_{96}, \hat{f}_{72}, \ldots, \hat{f}_{104}, \hat{f}_{105}, \ldots, \hat{f}_{89}, \hat{f}_{90}, \ldots, \hat{f}_{119}$$

To facilitate the analysis of the adopted scrambling, we introduce now E, the exponent matrix of Y. Recall that (7.8) implies

$$Y_{mn} = W^{\lambda(m)\lambda(n)} \tag{7.13}$$

Hence we define the exponent matrix E as follows:

$$E_{mn} = \lambda(m)\lambda(n) = uv \tag{7.14}$$

Similarly, paralleling the Ω_k submatrices of Y we define $\mathcal{E}_k$ as the upper-left ν_k-order submatrix of E.

Let us examine now the structure of $\mathcal{E}_{k-1}$, starting with the distribution of u_k, v_k. The scrambling prescribes that, starting with the value zero at the upper-left corner, u_k should be increased by one every ν_k rows. Similarly, v_k should increase by one every ν_k columns. This, then defines a subdivision of $\mathcal{E}_{k-1}$ into submatrices of order ν_k. $\mathcal{E}_{k-1}$ can now be regarded as a compound matrix of order $N_k\left(=\frac{\nu_{k-1}}{\nu_k}\right)$, whose (r,s) "element" is characterized by

$$u_k = r;\ v_k = s \tag{7.15}$$

The (0,0) element of this compound matrix is obviously identical with $\mathcal{E}_k$. Let us pick now an element in an arbitrary position in $\mathcal{E}_k$. Its u,v will have the form

$$\left.\begin{aligned} u &= (0,\ldots,0,u_{k+1},u_{k+2},\ldots,u_\kappa) \\ v &= (0,\ldots,0,v_{k+1},v_{k+2},\ldots,v_\kappa) \end{aligned}\right\} \tag{7.16}$$

Table 2

Modular Representation of v in Example (7.12)

$N_1 = 8;\ N_2 = 3;\ N_3 = 5$

v_1	v	v_2	v_3	v	v_2	v_3	v	v_2	v_3	v	v_2	v_3	v	v_2	v_3
0	0	0	0	8	2	3	16	1	1	24	0	4	32	2	2
1	1	1	1	9	0	4	17	2	2	25	1	0	33	0	3
2	2	2	2	10	1	0	18	0	3	26	2	1	34	1	4
3	3	0	3	11	2	1	19	1	4	27	0	2	35	2	0
4	4	1	4	12	0	2	20	2	0	28	1	3	36	0	1
5	5	2	0	13	1	3	21	0	1	29	2	4	37	1	2
6	6	0	1	14	2	4	22	1	2	30	0	0	38	2	3
7	7	1	2	15	0	0	23	2	3	31	1	1	39	0	4
0	40	1	0	48	0	3	56	2	1	64	1	4	72	0	2
1	41	2	1	49	1	4	57	0	2	65	2	0	73	1	3
2	42	0	2	50	2	0	58	1	3	66	0	1	74	2	4
3	43	1	3	51	0	1	59	2	4	67	1	2	75	0	0
4	44	2	4	52	1	2	60	0	0	68	2	3	76	1	1
5	45	0	0	53	2	3	61	1	1	69	0	4	77	2	2
6	46	1	1	54	0	4	62	2	2	70	1	0	78	0	3
7	47	2	2	55	1	0	63	0	3	71	2	1	79	1	4
0	80	2	0	88	1	3	96	0	1	104	2	4	112	1	2
1	81	0	1	89	2	4	97	1	2	105	0	0	113	2	3
2	82	1	2	90	0	0	98	2	3	106	1	1	114	0	4
3	83	2	3	91	1	1	99	0	4	107	2	2	115	1	0
4	84	0	4	92	2	2	100	1	0	108	0	3	116	2	1
5	85	1	0	93	0	3	101	2	1	109	1	4	117	0	2
6	86	2	1	94	1	4	102	0	2	110	2	0	118	1	3
7	87	0	2	95	2	0	103	1	3	111	0	1	119	2	4

Table 3

Scrambling for Example (7.12)

n vs. $v = (v_1, v_2, v_3)$

v_1								v_2	v_3
0	1	2	3	4	5	6	7		
0	105	90	75	60	45	30	15	0	0
96	81	66	51	36	21	6	111	0	1
72	57	42	27	12	117	102	87	0	2
48	33	18	3	108	93	78	63	0	3
24	9	114	99	84	69	54	39	0	4
40	25	10	115	100	85	70	55	1	0
16	1	106	91	76	61	46	31	1	1
112	97	82	67	52	37	22	7	1	2
88	73	58	43	28	13	118	103	1	3
64	49	34	19	4	109	94	79	1	4
80	65	50	35	20	5	110	95	2	0
56	41	26	11	116	101	86	71	2	1
32	17	2	107	92	77	62	47	2	2
8	113	98	83	68	53	38	23	2	3
104	89	74	59	44	29	14	119	2	4

Now, the adopted scrambling scheme guarantees that all elements of $\mathcal{E}_{k-1}$ occupying the same identical position in the other submatrices of $\mathcal{E}_{k-1}$, have u_i, v_i which differ from (7.16) only in u_k, v_k and these satisfy (7.15). This means that the difference between an element in submatrix (r,s) of $\mathcal{E}_{k-1}$ and the corresponding element in $\mathcal{E}_k$ (submatrix (0,0)) is just ([7] , (7.14))

$$\Delta E(r,s) = (0,\ldots,0,\underset{\substack{\uparrow \\ \text{k-th position}}}{rs \bmod N_k},0,\ldots,0). \tag{7.17}$$

We notice the important fact that the difference is independent of the specific common location of the two paired elements in their respective submatrices. Hence, the difference is constant throughout the (r,s) submatrix. In other words, the (r,s) submatrix of $\mathcal{E}_{k-1}$ could be generated from $\mathcal{E}_k$ simply by adding the constant $\Delta E(r,s)$ to all its elements.

We are interested in an explicit expression for this important constant. Considering its implicit formulation (7.17), we conclude that it must be some integral multiple of n_k (7.3). Specifically,

$$\Delta E(r,s) = \eta(r,s)n_k \tag{7.18}$$

where the integer η satisfies

$$\eta < N_k \tag{7.19}$$

$$n_k\eta - rs \equiv 0 \pmod{N_k}. \tag{7.20}$$

Eqn. (7.20) is a linear congruence for the unknown η for which there is an explicit solution [8] , namely,

$$\eta = (rs') \bmod N_k \tag{7.21}$$

where[13]

$$s' = (s\zeta_k) \bmod N_k \tag{7.22}$$

$$\zeta_k = n_k^{\phi(N_k)-1} \bmod N_k \tag{7.23}$$

The result we have established for the submatrices of $\mathcal{E}_{k-1}$ translates as follows for the submatrices of Ω_{k-1}: The (r,s) submatrix of Ω_{k-1}

---

[13] $\phi(n)$ is the Euler Totient function defined as the number of integers not exceeding, and relatively prime to, n.

is just $W^{\Delta E(r,s)}\Omega_k$. Applying (7.18), we find that

$$W^{\Delta E(r,s)} = e^{-i\frac{2\pi}{N_k}\eta(r,s)} \tag{7.24}$$

Denoting now

$$W_k = e^{-i\frac{2\pi}{N_k}}, \tag{7.25}$$

we get

$$W^{\Delta E(r,s)} = W_k^{\eta(r,s)} \tag{7.26}$$

so that the (r,s) "element" of the compound N_k-th order Ω_{k-1} is

$$W_k^{\eta(r,s)}\Omega_k = W_k^{(rs') \bmod N_k}\Omega_k = W_k^{rs'}\Omega_k$$

This is very similar to the (r,s) element of the DFT matrix (2.4) of order N_k with $\Omega = \Omega_k$. The only difference is that s' has now replaced s. This, however, is a trivial difference involving only column permutations. To prove this, it is sufficient to show that there is a one-to-one correspondence between s and s' so that when s goes through the values $0,1,\ldots,N_k-1$, s' goes through a permutation of them. This will, indeed, be the case if ζ_k (in (7.22)) is relatively prime to N_k. But this is guaranteed by (7.23) since n_k is relatively prime to N_k (see (7.3)).

We conclude that a trivial modification of the DFT tableau of order N_k, will evaluate the transformation effected by Ω_{k-1}. Specifically, we should modify the input square of the N_k tableau by permuting the f_i rows/columns so that the i sequence would be identical with the s' sequence (instead of the natural number sequence (s) of the unmodified tableau). We refer to such a tableau as a "modified tableau" and use this term from now on, only with this restricted, precise, meaning.

We illustrate now the tableau modification with $k=1$ in our example (7.12)

$$n_1 = N_2N_3 = 15; \quad \phi(N_1) = \phi(8) = 4$$
$$\zeta_1 = 15^{4-1} \bmod 8 = (-1)^3 \bmod 8 = 7$$

Hence

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	
1				1				c_0
	1						1	c_1
		1				1		c_2
			1		1			c_3
1				-1				c_4
			1		-1			c_5
		1				-1		c_6
	1						-1	c_7

Fig. 19. Input square of the 8-th order modified tableau for example (7.12)

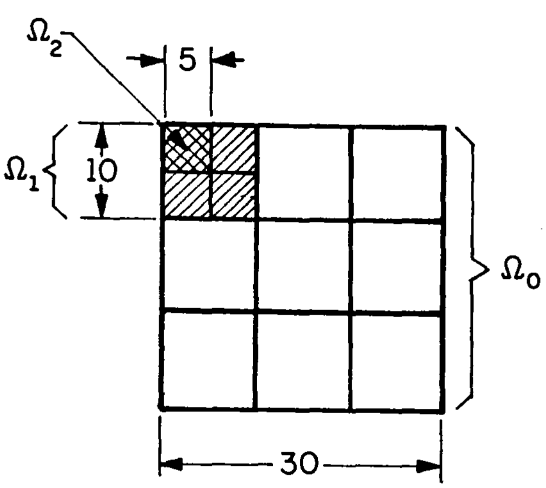


Fig. 20. Subdivision of the Y matrix for example (7.29)

s	0	1	2	3	4	5	6	7
$s'=(7s) \bmod 8$	0	7	6	5	4	3	2	1

(7.27)

The modified input square called for by (7.27) is shown in Fig. 19. Note that, in this case, the modification involves only sign changes.

The result we have established may be summarized as follows:

Submatrix Ω_{k-1} of Y is a column permutation of the compound DFT matrix of order N_k (2.4) with $\Omega = \Omega_k$. This is valid for all $1 \leq k \leq \kappa$. (7.28)

This differs from (7.4) in the column permutation. But, as we have just pointed out, this only means that in the algorithm for order N (7.1), the standard tableaus should be replaced by the modified tableaus. This is the minor complication referred to earlier.

We turn now to a simple example to illustrate the algorithm developed here. The example chosen has the following parameters:

$$\kappa = 3;\ N_1 = 3;\ N_2 = 2;\ N_3 = 5;\quad \therefore N = 30 \tag{7.29}$$

The first step is the scrambling of the input vector $\hat{f}$ to yield the vector q. This and the concurrent scrambling of the output $\hat{F}$, transform the DFT matrix into the Y matrix of (7.13). Y is now subdivided as indicated schematically in Fig. 20 (the indicated "measurements" refer to the number of rows/columns). The next step is to regard $Y(=\Omega_0)$ as a compound third-order matrix $(N_1 = 3)$ and apply the third order modified DFT tableau with the tableau's Ω identified with Ω_1 of Fig. 20. The application of the tableau is shown in Fig. 21 which utilizes the schematic tableau representation introduced in Fig. 18. Phase 1 is shown at the top (circled Ω_0), phase 3 is at the bottom (Ω_0 in a square) and phase 2 is all the region in between. Phase 1 requires separation of the input vector into its three "components". This is implemented in the following straightforward manner:

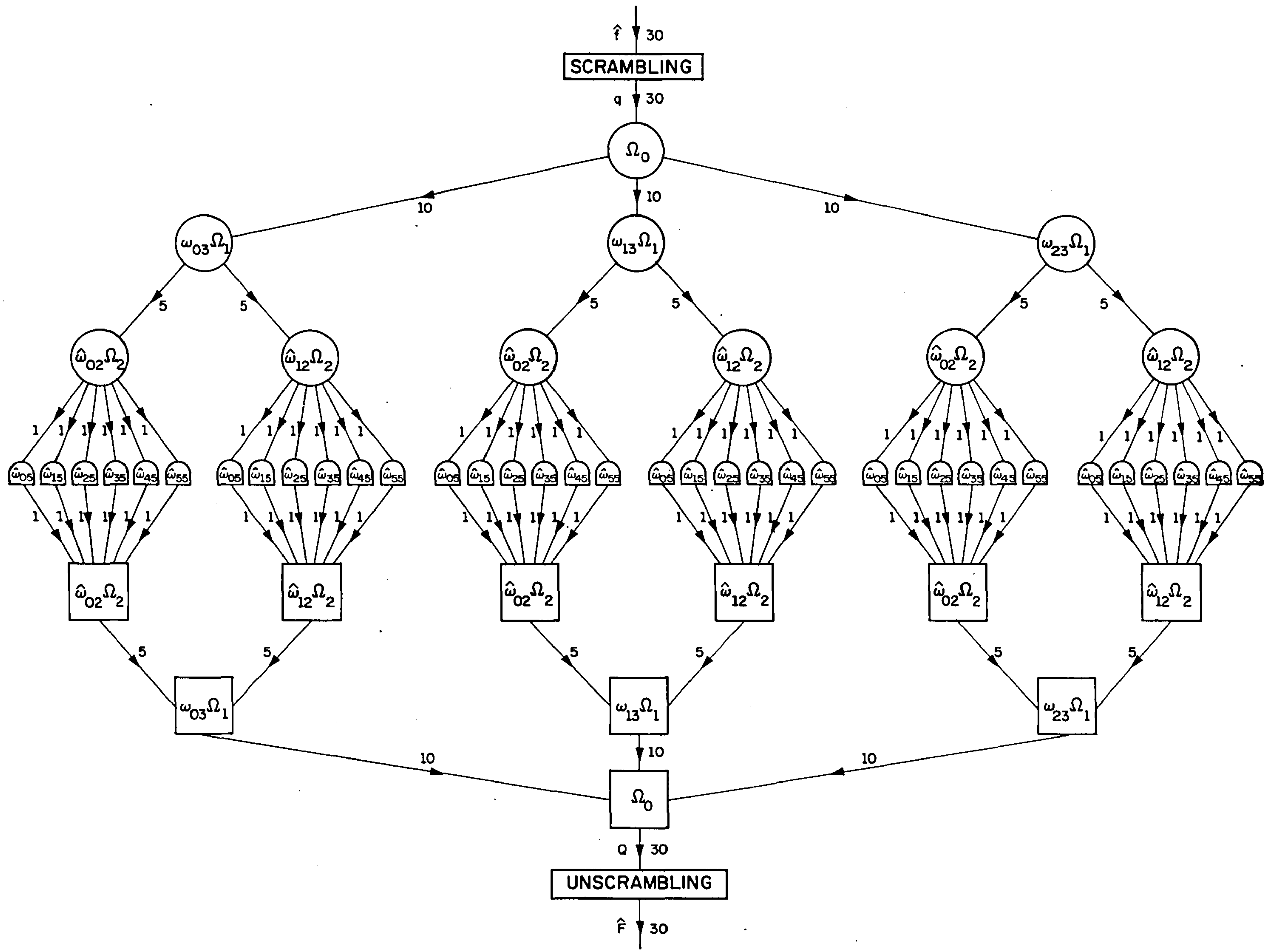


Fig. 21. Algorithm for DFT of order 30 (example (7.29))

$$f_0 = \begin{bmatrix} q_0 \\ \cdot \\ \cdot \\ \cdot \\ q_9 \end{bmatrix}; \quad f_1 = \begin{bmatrix} q_{10} \\ \cdot \\ \cdot \\ \cdot \\ q_{19} \end{bmatrix}; \quad f_2 = \begin{bmatrix} q_{20} \\ \cdot \\ \cdot \\ \cdot \\ q_{29} \end{bmatrix} \qquad (7.30)$$

The outputs of phase 1 are the three β_i vectors of dimensionality 10. Before considering phase 2, we have to introduce the generalized notation we use there for the ω_i's. ω_i for the DFT tableau of order N is referred to now as $\omega_{i,N}$. ($\hat{\omega}_{i,N}$, also appearing in Fig. 21, will be defined later). These constants are either explicitly listed in the tableaus or are inferred from the convention that every β_i is multiplied by $(\omega_i\Omega)$. For example, the DFT tableau of order 5 states explicitly $\omega_{1,5} = -\frac{5}{4}$. Implicitly, we infer $\omega_{0,5} = 1$. All the ω_{ij}'s are tabulated in Table 4 for convenience. The values there have been computed on a 10-digit calculator. For more precise values, one should refer to the exact expressions in the tableaus.

Returning now to Fig. 21, we find that phase 2 of the third order tableau requires the evaluation of $\xi_i = (\omega_{i3}\Omega_1)\beta_i$. Here we apply again result (7.28) which implies (with k = 2) that the transformation $(\omega_{i3}\Omega_1)\beta_i$ could be evaluated with the modified second order DFT tableau (with the tableau's Ω identified with $(\omega_{i3}\Omega_2)$ (see Fig. 20)). Each of the three 10-dimensional β_i's is therefore shown in Fig. 21 as the input to phase 1 of a second order tableau. In each of these applications of the second order tableau, phase 1 yields a pair of 5-dimensional β vectors. Consider now the specific 5-dimensional β vector on the extreme left of Fig. 21. It has been generated by phase 1 of the tableau implementing the transformation based on the matrix $(\omega_{03}\Omega_1)$. Therefore, phase 2 calls for its transformation by the matrix $\omega_{02}(\omega_{03}\Omega_2)$. Fig. 21 shows, instead, the matrix $\hat{\omega}_{02}\Omega_2$. We have adopted here the following somewhat unusual terminology:

$$\hat{\omega}_{ij} = \omega_{ij} * (\text{the first } \Omega_k \text{ multiplier met in moving against the arrows in the upper half of Fig. 21}) \qquad (7.31)$$

Thus, $\hat{\omega}_{ij}$ is only defined with respect to a specific diagram. Furthermore, the same symbol may have a different numerical value at a different location

Table 4

The DFT Tableau Multipliers ($\omega_{k,N}$)

k \ N	2	3	4	5	7	8	9	16
0	1	1	1	1	1	1	1	1
1	1	-1.5	1	-1.25	-1.1666667E-0	7.0710678E-1	0.5	1
2		i8.6602540E-1	1	-5.5901699E-1	5.5854267E-2	1	-1.7364818E-1	1
3			i	i1.5388418E-0	7.3430220E-1	1	0.5	7.0710678E-1
4				-i3.6327126E-1	-i8.7484229E-1	1	9.3969262E-1	1
5				-i5.8778525E-1	-i5.3396936E-1	1	-i3.4202014E-1	-1.3065630E-0
6					-i4.4095855E-1	1	-i8.6602540E-1	i
7					7.9015647E-1	i7.0710678E-1	-i9.8480775E-1	5.4119610E-1
8					-i3.4087293E-1		-i8.6602540E-1	1
9							7.6604444E-1	i7.0710678E-1
10							-i6.4278761E-1	i7.0710678E-1
11								i
12								1
13								-i1.3065630E-0
14								i7.0710678E-1
15								-i5.4119610E-1
16								3.8268343E-1
17								i9.2387953E-1

NOTE: Numbers shown in 3 digits or less are exact.
All other numbers are in Fortran E-format ($3.4E-1 = 3.4 \times 10^{-1}$).

in the diagram. For example, we have just seen that on the extreme left, $\hat{\omega}_{02} = \omega_{02}\omega_{03}$. This symbol appears in two other places to the right. The first one equals $\omega_{02}\omega_{13}$, the second equals $\omega_{02}\omega_{23}$.

Returning now to the main line of the argument, each one of the 5-dimensional β vectors should now be input to a modified DFT tableau of order 5. This time, the β "vectors" generated in phase 1 are of dimensionality 1, namely, scalars. They are to be multiplied by $\hat{\omega}_{i5}\Omega_3 = \hat{\omega}_{i5}$ (Ω_3 is the scalar 1). At this point, the multiplications are actually carried out and the numerical values of the multipliers are needed. Their determination is straightforward. Consider for example the multiplier on the extreme left of Fig. 21

$$\hat{\omega}_{05} = \omega_{05}\hat{\omega}_{02} = \omega_{05}\omega_{02}\omega_{03}$$

Similarly, for the multiplier on the extreme right

$$\hat{\omega}_{55} = \omega_{55}\hat{\omega}_{12} = \omega_{55}\omega_{12}\omega_{23}$$

and in general, the value of the multiplier at a specific node is the product of all the ω_{ij} terms (stripped of their circumflex) which one encounters in moving from that node up the (inverted) tree structure to the stem.

The multiplications are indicated by the half-circle, half-square shapes stringed along the center line of Fig. 21. (These can be regarded as representing the combined three phases of the DFT algorithm of order 1 with $\Omega = \hat{\omega}_{i5}$).

The 36 terms we get after performing the multiplications comprise 6 independent groups resulting from the 6 separate applications of the modified tableau of order 5. The terms of each of these groups are now combined as prescribed in phase 3 of the 5-th order tableau to yield six 5-dimensional vectors. Each of these is actually a term of the form $\hat{\omega}_{i2}\Omega_2\beta_i$ required to complete the computations in the three applications of the second order tableau. These computations now yield, as tableau outputs, three 10-dimensional vectors which are identical with $\omega_{i3}\Omega_1\beta_i$ of the third order tableau. Combining these as prescribed by phase 3 of this tableau yields the 30-dimensional Q vector which is just a scrambled version of the desired vector $\hat{F}$.

Fig. 21, though directly applicable to example (7.29) only, is characteristic of all N values. For larger N, there might be one more level of branching in each half of the diagram and the number of branches per node may be higher. Otherwise, the structure is identical with that of Fig. 21.

VIII. Speed Analysis

In sections IV-VI we constructed the basic DFT tableaus. In section VII we showed how to use them in an efficient algorithm for certain N values. Our purpose in this section is to determine just how fast the resulting algorithm is and present a summary (Table 6) of the pertinent parameters for all orders realizable with the constructed tableaus. The basis for these developments is Table 5 which presents a summary of the tableau parameters. These have been collected from the tableau designations and have been fully explained earlier. The values appearing here are in general agreement with Winograd's results[14] (Table 1 of [3]). The only difference is in the number of multiplications in the tableau of order 9. Winograd uses 13; we use 11. This means that, using this tableau in the computation of any DFT whose order is divisible by 9, will yield a 15% reduction in the number of multiplications as compared to Winograd's results (see (8.3)). In most of these cases there is also a reduction in the number of additions (see (8.9)).

Consider now the algorithm of order $N = \prod_{k=1}^{K} N_k$ as applied to complex data[15]. For each N_k, we read off from Table 5 the corresponding number of complex multiplications M_k and complex additions A_k. We are interested in two functions of these variables, namely, the total number of real multiplications $\mathcal{M}$ and the total number of real additions $\mathcal{A}$ for the overall algorithm realized in the order implied in (7.1) (phase 1 of the N_1 tableau realized first; phase 1 of the N_K tableau realized last). With this goal in mind, we turn now to a mathematical formulation of some of the characteristics of the algorithm structure which are quite evident in Fig. 21.

We note that the output of phase 1 of the N_1 tableau is a set of M_1 vectors (β_i) of dimensionality ν_1. Each of these now generates (at the output of phase 1 of the N_2 tableau) M_2 vectors of dimensionality ν_2, and so on. It is obvious therefore that

$$\left.\begin{array}{l} \text{The total output of phase 1 of all the} \\ \text{tableaus of order } N_k \text{ consists of} \\ \left(\prod_{i=1}^{k} M_i\right) \text{ vectors of dimensionality } \nu_k \text{ (7.2)} \end{array}\right\} \quad (8.1)$$

---

[14] Our $\mathcal{M}$ values, representing the total number of multiplications, should be compared to the sum of Winograd's two multiplication columns.

[15] The computation of the number of arithmetic operations for real data is more involved and will only be briefly discussed later on.

Table 5

Summary of Basic DFT Tableaus

Tableau Order	Total Number of Multiplications	Number of Multiplications by 1 or i	Number of Additions	Tableau Figure	Tableau Page
N	M	m	A		
2	2	2	2	9	37
3	3	1	6	5	31
4	4	4	8	10	39
5	6	1	17	6	31
7	9	1	36	8	35
8	8	6	26	14	51
9	11	1	44	13	45
16	18	8	74	17	60

As each one of these vectors is fed to phase 1 of an N_{k+1} tableau, this is also the number of tableaus of order N_{k+1} or, equivalently,

$$\left.\begin{array}{l}\text{The number of tableaus of} \\ \text{order } N_k \text{ is } \prod_{i=1}^{k-1} M_i\end{array}\right\} \tag{8.2}$$

The simplest application of these results is the determination of the number of multiplications. Let $\mathcal{M}_\kappa$ be the total number of (complex) scalar multiplications in phase 2 of the algorithm realized as a cascade of κ stages. The variables which are multiplied are the 1-dimensional β_i's generated in the last stage of the cascade (N_κ). From (8.1) we know that there are $\prod_{i=1}^{\kappa} M_i$ such terms. Hence

$$\mathcal{M}_\kappa = \prod_{i=1}^{\kappa} M_i \tag{8.3}$$

To get the total number of real multiplications in the overall cascade $(\mathcal{M})$, we note that in each of the counted multiplications, only one of the two factors is complex, the other being real or imaginary (see Table 4). Therefore,

$$\mathcal{M} = 2\mathcal{M}_\kappa \tag{8.4}$$

We have seen in Section VII that each of the overall multipliers for the cascade is a product of κ tableau multipliers, one from each tableau of the cascade. Since each tableau has, at least, one multiplier whose value is 1 or i ($m \geq 1$ in Table 5), some of the $\mathcal{M}_\kappa$ overall multipliers will be 1 or i. A consideration of the structure of Fig. 21 shows that the number of such multipliers is[16]

$$P_\kappa = \prod_{i=1}^{\kappa} m_i \tag{8.5}$$

---

[16]All permissible N values are expressible as $N = H2^r$ (H odd; $o \leq r \leq 4$). Using this with the values of m_i listed in Table 5, yields $P_\kappa = 2r + \delta_{o,r}$

Therefore, if one accepts the somewhat more complex programming involved in the special handling of these P_κ trivial multipliers, the DFT can actually be computed with the smaller number of real multiplications

$$\hat{\mathcal{M}} = 2\ (\mathcal{M}_\kappa - P_\kappa) \tag{8.6}$$

The summary in Table 6 covers both cases $\Big((8.4),\ (8.6)\Big)$.[17]

We turn now to the additions count. Let $\mathcal{A}_\kappa$ be the total number of (complex) scalar additions in a cascade of κ stages. Hence, the corresponding number of real additions is

$$\mathcal{A} = 2\mathcal{A}_\kappa \tag{8.7}$$

The simplest way to determine $\mathcal{A}_\kappa$ is through a recursive argument. Assume that we have the result for a cascade of length $\kappa - 1$ and we add to it one more stage at position κ. This has two effects. First, the additions in the first κ-1 stages will now involve vectors whose dimensionalities are N_κ times their previous values. Hence, the previous $\mathcal{A}_{\kappa-1}$ additions are transformed into $N_\kappa\ \mathcal{A}_{\kappa-1}$ additions. To this we should add the additions of the last stage. From (8.2), the number of tableaus in the last stage is $\prod_{i=1}^{\kappa-1} M_i = \mathcal{M}_{\kappa-1}$ (8.3). Each such tableau requires A_κ additions of scalars. Hence, the number of scalar additions in the last stage is $\mathcal{M}_{\kappa-1} A_\kappa$ and the total is

[17] One could argue that in a binary machine, multiplication by $\frac{1}{2}$ is also trivial and should be excluded from the multiplication count. If this attitude is adopted, then the 9th order tableau will have 3 trivial multiplications (m = 3). The effect of this will be quite pronounced for N = 144, reducing $\hat{\mathcal{M}}$ from 380 (Table 6) to 348.

$$\left.\begin{aligned} \mathcal{A}_K &= \mathcal{A}_{K-1}\, N_K + \mathcal{M}_{K-1}\, A_K \\ \mathcal{M}_K &= \mathcal{M}_{K-1}\, M_K \\ \mathcal{A}_1 &= A_1;\ \mathcal{M}_1 = M_1 \end{aligned}\right\} \qquad (8.8)$$

We have added here a recursive rephrasing of (8.4) as well as the initial conditions to provide a complete prescription for the simultaneous computation of both $\mathcal{A}_K$ and $\mathcal{M}_K$. (8.8) also yields explicit formulae for $\mathcal{A}_K$. For example

$$\begin{aligned} \mathcal{A}_4 = {} & A_1N_2N_3N_4+ \\ & +M_1A_2N_3N_4+ \\ & +M_1M_2A_3N_4+ \\ & +M_1M_2M_3A_4 \end{aligned} \qquad (8.9)$$

An important fact clearly indicated by (8.9) is that $\mathcal{A}_K$, unlike $\mathcal{M}_K$, is also a function of the order of the N_k's comprising N. Thus, it is important to find the cascade order minimizing $\mathcal{A}_K$.

With eqns. (8.3)-(8.8) and Table 5 at our disposal, we can now compute $\mathcal{A}, \mathcal{M}$ for any N satisfying (7.1). Table 6 presents the results of such computations implemented by a simple computer program which also provides information for the selection of the most efficient cascade ordering. This N_k sequence appears in the last columns of Table 6. We provide here room for two sequences since, in some cases, the same values of $\mathcal{M}, \mathcal{A}$ are obtained with two different N_k sequences. In this case, the choice could be governed by arguments other than efficiency.

Each N_k value appears with a bracketed number to its right. This is the ζ_k of (7.22). Thus, Table 6 provides both the sequence of tableaus to be realized and the permutation required in the input square of each tableau (see discussion preceding (7.27).

Note that adoption of the order prescribed in Table 6 is quite important. For example, with N = 240, Table 6 states $\mathcal{A}$ = 5016 with the prescribed order 3; 16; 5. If, instead, we adopt the order 5; 16; 3, the number of real additions jumps to 5592--an increase of 11%.

We turn now to the two remaining columns of Table 6, namely, G_∞, R. These are the two parameters mentioned in section I and are required to determine the speed advantage in a specific system.

TABLE 6

Summary of DFT Algorithm

$$\left(\mu = \frac{\text{time for one real multiplication}}{\text{time for one real addition}}\right)$$

Order of DFT	Multiplications by 1 and i, included in count: Speed-Gain Function Parameters $G(\mu)=G_\infty \frac{\mu+1.5}{\mu+R}$		Multiplications by 1 and i, included in count: Number of Real Multiplications	Multiplications by 1 and i, excluded from count: Speed-Gain Function Parameters $\hat{G}(\mu)=\hat{G}_\infty \frac{\mu+1.5}{\mu+\hat{R}}$		Multiplications by 1 and i, excluded from count: Number of Real Multiplications	Number of Real Additions	Tableau Realization Sequences $N_k(\zeta_k)$: k				k			
N	G_∞	R	$\mathcal{M}$	$\hat{G}_\infty$	$\hat{R}$	$\hat{\mathcal{M}}$	$\mathcal{A}$	1	2	3	4	1	2	3	4
2	1.000	1.000	4	–	–	0	4	2(1)							
3	1.585	2.000	6	2.377	3.000	4	12	3(1)							
4	2.000	2.000	8	–	–	0	16	4(1)							
5	1.935	2.833	12	2.322	3.400	10	34	5(1)							
6	2.585	3.000	12	3.877	4.500	8	36	3(2)	2(1)			2(1)	3(2)		
7	2.183	4.000	18	2.456	4.500	16	72	7(1)							
8	3.000	3.250	16	12.000	13.000	4	52	8(1)							
9	2.594	4.000	22	2.853	4.400	20	88	9(1)							
10	2.768	3.667	24	3.322	4.400	20	88	2(1)	5(3)						
12	3.585	4.000	24	5.377	6.000	16	96	3(1)	4(3)			4(3)	3(1)		
14	2.961	4.778	36	3.331	5.375	32	172	2(1)	7(4)						
15	3.256	4.500	36	3.447	4.765	34	162	3(2)	5(2)						
16	3.556	4.111	36	6.400	7.400	20	148	16(1)							
18	3.412	4.818	44	3.753	5.300	40	212	2(1)	9(5)						
20	3.602	4.500	48	4.322	5.400	40	216	4(1)	5(4)						
21	3.416	5.556	54	3.548	5.769	52	300	3(1)	7(5)						
24	4.585	5.250	48	6.113	7.000	36	252	3(2)	8(3)			8(3)	3(2)		
28	3.739	5.556	72	4.206	6.250	64	400	4(3)	7(2)						
30	4.089	5.333	72	4.330	5.647	68	384	3(1)	2(1)	5(1)		2(1)	3(1)	5(1)	
35	3.325	6.167	108	3.387	6.283	106	666	7(3)	5(3)						
36	4.230	5.636	88	4.653	6.200	80	496	4(1)	9(7)						
40	4.435	5.542	96	5.069	6.333	84	532	8(5)	5(2)						
42	4.194	6.333	108	4.355	6.577	104	684	3(2)	2(1)	7(6)		2(1)	3(2)	7(6)	
45	3.744	6.167	132	3.802	6.262	130	814	9(2)	5(4)						
48	4.964	5.889	108	5.828	6.913	92	636	3(1)	16(11)						
56	4.517	6.528	144	4.927	7.121	132	940	8(7)	7(1)						
60	4.922	6.167	144	5.212	6.529	136	888	3(2)	4(3)	5(3)		4(3)	3(2)	5(3)	
63	3.804	7.111	198	3.843	7.184	196	1408	9(4)	7(4)						
70	3.973	6.815	216	4.048	6.943	212	1472	2(1)	7(5)	5(4)					
72	5.048	6.659	176	5.417	7.146	164	1172	8(1)	9(8)						
80	4.683	6.259	216	5.058	6.760	200	1352	16(13)	5(1)						
84	4.972	7.111	216	5.163	7.385	208	1536	3(1)	4(1)	7(3)		4(1)	3(1)	7(3)	
90	4.426	6.848	264	4.494	6.954	260	1808	2(1)	9(1)	5(2)					
105	4.352	7.463	324	4.379	7.509	322	2418	3(2)	7(1)	5(1)					
112	4.706	7.198	324	4.951	7.571	308	2332	16(7)	7(4)						
120	5.756	7.208	288	6.006	7.522	276	2076	3(1)	8(7)	5(4)		8(7)	3(1)	5(4)	
126	4.440	7.747	396	4.485	7.827	392	3068	2(1)	9(2)	7(2)					
140	4.621	7.463	432	4.708	7.604	424	3224	4(3)	7(6)	5(2)					
144	5.214	7.364	396	5.434	7.674	380	2916	16(9)	9(4)						
168	5.750	8.083	432	5.914	8.314	420	3492	3(2)	8(5)	7(5)		8(5)	3(2)	7(5)	
180	5.108	7.530	528	5.187	7.646	520	3976	4(1)	9(5)	5(1)					
210	5.000	8.111	648	5.031	8.161	644	5256	3(1)	2(1)	7(4)	5(3)	2(1)	3(1)	7(4)	5(3)
240	5.857	7.741	648	6.005	7.937	632	5016	3(2)	16(15)	5(2)					
252	5.076	8.384	792	5.128	8.469	784	6640	4(3)	9(1)	7(1)					
280	5.269	8.273	864	5.343	8.390	852	7148	8(3)	7(3)	5(1)					
315	4.401	8.759	1188	4.409	8.774	1186	10406	9(8)	7(5)	5(2)					
336	5.802	8.580	972	5.899	8.724	956	8340	3(1)	16(13)	7(6)					
360	5.790	8.383	1056	5.856	8.479	1044	8852	8(5)	9(7)	5(3)					
420	5.648	8.759	1296	5.683	8.814	1288	11352	3(2)	4(1)	7(2)	5(4)	4(1)	3(2)	7(2)	5(4)
504	5.713	9.179	1584	5.756	9.249	1572	14540	8(7)	9(5)	7(4)					
560	5.260	8.831	1944	5.303	8.905	1928	17168	16(11)	7(5)	5(3)					
630	4.931	9.290	2376	4.940	9.305	2372	22072	2(1)	9(4)	7(6)	5(1)				
720	5.753	8.970	2376	5.792	9.031	2360	21312	16(5)	9(8)	5(4)					
840	6.296	9.569	2592	6.326	9.614	2580	24804	3(1)	8(1)	7(1)	5(2)	8(1)	3(1)	7(1)	5(2)
1008	5.644	9.727	3564	5.669	9.771	3548	34668	16(15)	9(7)	7(2)					
1260	5.462	9.820	4752	5.471	9.836	4744	46664	4(3)	9(2)	7(3)	5(3)				
1680	6.173	9.984	5832	6.190	10.011	5816	58224	3(2)	16(9)	7(4)	5(1)				
2520	5.992	10.483	9504	6.000	10.496	9492	99628	8(3)	9(1)	7(5)	5(4)				
5040	5.798	10.939	21384	5.802	10.948	21368	233928	16(3)	9(5)	7(6)	5(2)				

Let

$$\mu = \frac{\text{time for one real multiplication}}{\text{time for one real addition}}$$

in the specific system considered. We define the gain G, of the present algorithm over the (nominal) Cooley-Tukey algorithm by

$$G = \frac{\mu \mathcal{M}_{CT} + \mathcal{A}_{CT}}{\mu\mathcal{M} + \mathcal{A}} \tag{8.10}$$

where $\mathcal{M}_{CT}, \mathcal{A}_{CT}$ are the Cooley-Tukey parameters introduced in (1.1). Obviously, G is the ratio of the time required by the Cooley-Tukey algorithm to the time required by Winograd's algorithm. We refer to it as the speed gain. It is a function of μ and the four parameters appearing in (8.10). Eqn. (8.11) is a more convenient two-parameter formulation (see (1.1)).

$$G(\mu) = G_\infty \frac{\mu+1.5}{\mu+R} \tag{8.11}$$

where

$$G_\infty = \frac{\mathcal{M}_{CT}}{\mathcal{M}} \tag{8.12}$$

$$R = \frac{\mathcal{A}}{\mathcal{M}} \tag{8.13}$$

G_∞ is the asymptotic speed gain that is approached with large μ. R prescribes a pole of the $G(\mu)$ function (at $\mu = -R$) and thus determines the second asymptote. Adding to these two the zero of $G(\mu)$ (at $\mu = -1.5$) makes it very easy to sketch $G(\mu)$ and get a sufficiently precise estimate of it.[18]

We conclude with a few words regarding the real data case. The number of arithmetic operations here is not half the number in the corresponding complex data case. The main reason for that is that as the DFT of a real vector is, in general, complex, some of the intermediate entities will be complex too. For example, the 8-th order DFT tableau prescribes

$$\beta_5 = e_2 + ie_5 \tag{8.14}$$

and thus β_5 is complex even for real data. This means of course that the tableaus realizing $\xi_5 = \Omega\beta_5$ have complex data inputs.

Another factor to consider is the fact that the construction of a complex number, given its two components, does not involve any arithmetic addition in spite of the appearance of the plus symbol. In the example previously cited, both e_2 and e_5 are real when the f_i's are real. Hence, the "addition" appearing in (8.14) is free.

[18] In eqns. (8.10) - (8.13), replacing $\mathcal{M}$ of (8.4) by $\hat{\mathcal{M}}$ of (8.6), yields the circumflexed entities $\hat{G}$, $\hat{G}_\infty$, $\hat{R}$ appearing in Table 6.

IX. Concluding Remarks

We have tried to present here an orderly development of Winograd's DFT algorithm, starting with the general concept, continuing with the construction of the necessary building blocks and culminating in a detailed description of their incorporation into the overall algorithm.

In applying the algorithm, Table 6 (p. 84) is the starting point as it lists all the permissible N values with their associated performance parameters. Having chosen a particular N, the next step is to consult Table 5 (p. 80) in order to locate the specific tableaus called for in Table 6. In the actual implementation, one starts with the scrambling of the input vector and then applies phase 1 of the tableaus (in the order prescribed in Table 6) to smaller and smaller segments of the data vector in its various partially transformed states. This culminates in single component "segments" finally being multiplied by constants in phase 2. From here on, the process is reversed in the application of phase 3 of the tableaus: The scalars appearing at the output of phase 2 are combined into vectors of higher and higher dimensionality, finally culminating in an N-dimensional vector which is just a scrambled version of the transformed input vector.

The present paper contains sufficiently detailed information upon which one could base a direct, straightforward, implementation of the above process in either hardware or software. There are, however, some less obvious implementations which have various advantages. These are the subject of a forthcoming paper and will not be discussed here. Nevertheless, attention should be called to a certain feature of the tableaus specifically designed into them to facilitate the application of these special techniques. We are referring here to "in-place" transformation. Consider, for example, the 7-th order tableau (Fig. 8). Note that the input components f_2, f_5 are used to compute c_2, c_5 and nothing else. Hence, there is no need to assign additional storage for c_2, c_5. They may be stored back into the f array, overwriting f_2, f_5. The only requirement is for a temporary storage for, say, f_2 so that after we store c_2 we still have f_2 available for the computation of c_5. Note that even if f_2 represents a vector, we still need only a one-word temporary store since the computation is carried out one component at a time.

This property which we have illustrated here with the $(f_2, f_5) \rightarrow (c_2, c_5)$ transformation is common to all variables in the DFT tableaus of orders 2, 3, 5, 7.

It is also valid for the remaining tableaus, if we regard η as the output vector. The only deviation from the above pattern is that, in some cases, groups of 3 components (rather than 2) have to be considered. In the above tableau, the computation of $\beta_1, \beta_2, \beta_3$ is such an example. Note however, that, even in this case, a one-word temporary store is sufficient.

It should be pointed out that in those applications in which this "in-place" feature is not utilized, the tableaus of orders 4,9,8,16, may be simplified somewhat by permuting the F_i rows/columns in the output square to yield an unscrambled output vector F. In this case, of course, the vector η may be dispensed with.

We turn now to a brief consideration of the precision disadvantage mentioned in section I. We have already seen in Appendix A (last paragraph) that some of the manipulations generating the tableaus have a detrimental effect on precision. A similar situation afflicts the computation of δ_1 in some of the tableaus. Examination of eqns. (4.11), (4.12) reveals that the adopted formulation (4.12) involves addition and subtraction of $\Omega\beta_1$. Hence if $|\Omega\beta_1| >> |\delta_1|$ we are bound to have problems, namely, loss of significant bits in floating point arithmetic and tendency to overflow in fixed-point arithmetic. Similar addition-subtraction manipulations are dispersed in various disguises throughout the tableaus' derivations.

The effect of these peculiarities of the tableaus is that to guarantee a certain measure of precision in the transformation, we probably need more bits per word than in the Cooley-Tukey algorithm. We do not analyze this effect here but it should be pointed out that the structure of the algorithm as displayed in Fig. 21 makes such an analysis relatively simple.

Finally, we conclude with yet another important aspect of the algorithm brought forth in Fig. 21, namely, the suitability of its structure to the application of various schemes of parallel processing and pipelining. Indeed, there is fertile ground here for all sorts of ingenious designs and variations. As the algorithm becomes more widely known, more and more of these will undoubtedly materialize.

Acknowledgement

The author wishes to express his thanks to Dr. L. D. Baumert of JPL for his helpful comments regarding some number-theoretic aspects of this work.

Appendix: Polynomial Congruences

The derivations in section III require numerous evaluations of

$$R(x) = S_n(x) \bmod m(x) \tag{A.1}$$

where

$$S_n(x) = \sum_{k=0}^{n} s_k x^k \tag{A.2}$$

and $m(x)$ is a monic polynomial of degree 1 or 2, whose roots lie on the unit circle. We establish here all the needed results.

1. $m(x) = x-x_0 \quad (x_0 = \pm 1)$

R(x) must be of degree 0

$$R(x) = r_0 \tag{A.3}$$

and (A.1) is equivalent in this case to

$$S_n(x) = (x-x_0)Q_{n-1}(x) + r_0 \tag{A.4}$$

where $Q_{n-1}(x)$ is a polynomial of degree (n-1). Substitutiong $x = x_0$ in (A.4) we find

$$R(x) = r_0 = S_n(x_0) \tag{A.5}$$

Note that with $x_0 = \pm 1$, r_0 is a multiplication -free algebraic sum of the coefficients of $S_n(x)$.

2. $m(x) = x^2-2x\cos\theta+1 = (x-x_0)(x-\overline{x}_0)$

R(x) must be of degree 1

$$R(x) = r_0 + r_1 x \tag{A.6}$$

and (A.1) is equivalent to

$$S_n(x) = (x-x_0)(x-\overline{x}_0)Q_{n-2}(x)+(r_0+r_1 x) \tag{A.7}$$

Hence

$$S_n(x_0) = r_0+r_1 x_0 \tag{A.8}$$

$$S_n(\overline{x}_0) = r_0+r_1\overline{x}_0 \tag{A.9}$$

Note that

$$x_0 = \cos\theta + i\sin\theta = e^{i\theta} \tag{A.10}$$

Hence, subtracting (A.9) from (A.8) yields

$$2\,i\,r_1 \sin\theta = S_n(x_0) - S_n(\overline{x}_0) = \sum_{k=0}^{n} s_k (e^{ik\theta} - e^{-ik\theta})$$

$$\therefore r_1 = \frac{1}{\sin\theta} \sum_{k=0}^{n} s_k \sin k\theta$$

$$r_1 = \sum_{k=1}^{n} s_k U_{k-1}(\cos\theta) \qquad \text{(A.11)}$$

where $U_m(x)$ is the Chebyshev polynomial of the second kind.

To get r_0, we multiply (A.8) by $\overline{x}_0$ and (A.9) by x_0 and then subtract, getting

$$2i\, r_0 \sin\theta = x_0 S_n(\overline{x}_0) - \overline{x}_0 S_n(x_0) = \sum_{k=0}^{n} s_k (e^{-i(k-1)\theta} - e^{i(k-1)\theta})$$

$$= 2is_0 \sin\theta - 2i \sum_{k=2}^{n} s_k \sin(k-1)\theta$$

$$\therefore r_0 = s_0 - \sum_{k=2}^{n} s_k U_{k-2}(\cos\theta) \qquad \text{(A.12)}$$

The specific $m(x)$ polynomials for which $R(x)$ is required are listed in Table A1 with the corresponding θ values. Note that for these values of θ, $U_k(\cos\theta)$ takes only the values $0, \pm 1$. Hence, both r_0 and r_1 are multiplication-free algebraic sums of the coefficients of $S_n(x)$. The results for all required degrees of $S_n(x)$ are shown in Table A1.

A specific application of Table A1 is the following: Given

$$P(x) \bmod m(x) = p_1 x + p_0 \qquad \text{(A.13)}$$

$$Q(x) \bmod m(x) = q_1 x + q_0 \qquad \text{(A.14)}$$

Find

$$G(x) = g_1 x + g_0 = \{P(x)Q(x)\} \bmod m(x) \qquad \text{(A.15)}$$

G(x) can be expressed in terms of (A.13) (A.14) as follows:

Table A1: $S_n(x) \bmod m(x)$

$$S_n(x) = \sum_{k=0}^{n} s_k x^k$$

m(x)	θ	$S_2(x)$	$S_3(x)$	$S_5(x)$
x^2-x+1	60^o	$(s_1+s_2)x+(s_0-s_2)$	-----	$(s_1+s_2-s_4-s_5)x+(s_0-s_2-s_3+s_5)$
x^2+1	90^o	$s_1\ x+(s_0-s_2)$	$(s_1-s_3)x+(s_0-s_2)$	-----
x^2+x+1	120^o	$(s_1-s_2)x+(s_0-s_2)$	$(s_1-s_2)x+(s_0-s_2+s_3)$	$(s_1-s_2+s_4-s_5)x+(s_0-s_2+s_3-s_5)$

$$G(x) = \{[P(x) \bmod m(x)][Q(x) \bmod m(x)]\} \bmod m(x)$$

$$= \{(p_1q_1)x^2 + (p_1q_0+p_0q_1)x + p_0q_0\} \bmod m(x) \tag{A.16}$$

Identifying the bracketed polynomial with $S_2(x)$ in Table A1, we get the following results

$$G(x) = [p_1(q_0+q_1)+p_0q_1]x + (p_0q_0-p_1q_1) \qquad (m(x) = x^2-x+1) \tag{A.17}$$

$$G(x) = (p_1q_0+p_0q_1)x + (p_0q_0-p_1q_1) \qquad (m(x) = x^2+1) \tag{A.18}$$

$$G(x) = [p_1(q_0-q_1)+p_0q_1]x + (p_0q_0-p_1q_1) \qquad (m(x) = x^2+x+1) \tag{A.19}$$

Starting with p_i, q_i, each of these formulae requires 4 multiplications. However, with the proper rearranging of terms, we can replace one of these multiplications with an extra addition, thus getting faster computation. This is accomplished by adding and subtracting p_0q_0 from the g_1 term. This is sufficient for (A.17), (A.19). In (A.18) we also modify the g_0 term by adding and subtracting p_0q_1. The results are summarized in Table A2 and, as we see there, 3 multiplications are now sufficient. The x^2+1 case, however, seems to indicate that the price is higher than previously stated, namely, 3 extra additions. In general this is indeed true. However, we intend to apply this result to a situation where arithmetic operations involving p_0, p_1 only, do not count (precomputation). Under these conditions, the price is indeed 1 extra addition.

It should be noted that the higher speed realized by the formulae of Table A2 is accompanied by the disadvantage of requiring more bits per word. In fixed point arithmetic we will need more bits to prevent overflow of the intermediate results. In floating point arithmetic, we will need more bits to prevent loss of precision. Consider the extreme case in which $p_0q_0 >> g_1$. (A.17) would not be affected by that but, in floating point, Table A2 could yield a value for g_1 which would be pure noise.

Table A2: $\{P(x)Q(x)\} \bmod m(x)$

$$P(x) \bmod m(x) = p_1x + p_0$$

$$Q(x) \bmod m(x) = q_1x + q_0$$

$m(x)$	$\{P(x)Q(x)\} \bmod m(x)$
x^2-x+1	$\left[(p_1+p_0)(q_1+q_0)-p_0q_0\right]x + (p_0q_0-p_1q_1)$
x^2+1	$\left[(p_1-p_0)q_0 + (q_1+q_0)p_0\right]x + \ (q_1+q_0)p_0-(p_1+p_0)q_1$
x^2+x+1	$\left[(p_1-p_0)(q_0-q_1) + p_0q_0\right]x + (p_0q_0-p_1q_1)$

REFERENCES

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series", Math. of Comp. Vol. 19 (1965), pp. 297-301.

[2] R. Yavne, "An economical method for calculating the discrete Fourier transform", AFIPS Conference Proceedings, Vol. 33, part 1, pp. 115-125.

[3] S. Winograd, "On computing the discrete Fourier transform", Proc. Nat. Acad. Sci. USA, Vol. 73, No. 4 pp. 1005-1006, April 1976, Mathematics.

[4] S. Winograd, "The effect of the field of constants on the number of multiplications", Proceedings of the 16th annual symposium on foundations of computer science, 1975, pp. 1-2.

[5] C. M. Rader, "Discrete Fourier Transforms When the Number of Data Samples is Prime", Proc. IEEE, Vol. 56, pp. 1007-1008, June 1968.

[6] M. Abramowitz and I. Stegun, Handbook of Mathematical Functions, New York, Dover, 1965, p. 864.

[7] D. E. Knuth, "The art of computer programming", Vol. 2, Addison Wesley, 1969 (section 4.3.2).

[8] T. Nagel, "Introduction to Number Theory", John Wiley, 1951.

[9] W. J. LeVeque, "Topics in Number Theory", Vol. 1, Addison-Wesley, 1958.

[10] H. F. Silverman, "An introduction to programming the Winograd Fourier Transform Algorithm (WFTA)", IEEE Trans. Acoust. Speech, Signal Processing, Vol. ASSP-25, pp. 152-165, April 1977.

[11] I. J. Good, "The Interaction Algorithm and Practical Fourier Analysis", J. of the Royal Statistical Society, Series B, Vol. 20, pp. 361-372, 1958.

NASA–JPL–Coml., L.A., Calif.

End of Document