A New Algorithm for the Discrete Cosine Transform of Arbitrary Number of Points

MEGHANAD D. WAGH AND H. GANESH

Abstract—An alternate algorithm to compute the discrete cosine transform (DCT) of sequences of arbitrary number of points is proposed. The algorithm consists of partitioning the DCT kernel into submatrices which by proper row and column shuffling and negations can be made equivalent to the group tables (or parts of them) of appropriate Abelian groups. The computations pertaining to the submatrices can be carried out using multidimensional cyclic convolutions. Algorithms are also developed to perform the computations associated with the submatrices that are parts of larger group tables. The new algorithms are more versatile and generally better in terms of the computational complexity in comparison with the existing algorithms.

Index Terms—Computational complexity, cyclic convolution, discrete cosine transform.

I. INTRODUCTION

THE discrete cosine transform (DCT) defined by Ahmed et al. [1] in 1974 has recently found a number of applications in the area of digital image processing [2]–[8]. The DCT approximates the optimal Karhunen-Loeve transform better than most other orthogonal transforms including the discrete Fourier transform (DFT) [1], [8], [9]. However, the utility of a transform is governed not only by its optimality but also by its computational simplicity and therefore there have been several attempts in recent years to find efficient algorithms for the DCT [10]–[13].

The DCT of an N-point sequence \( y(i), i = 0, 1, \ldots, N - 1 \) is defined as [1]

\[
Y(0) = \frac{\sqrt{N}}{2} \sum_{i=0}^{N-1} y(i), \\
Y(j) = \frac{\sqrt{N}}{2} \sum_{i=0}^{N-1} M(j, i) y(i), \quad j = 1, 2, \ldots, N - 1,
\]

where \( M(j, i) = \frac{2}{N} \cos(j(2i + 1)\pi/2N) \).

We restrict ourselves to real data sequences. Furthermore, since a scaling of every transform component by a constant factor does not alter its applicability, we ignore the factor \( (2/N) \) in \( M(j, i) \) and compute \( Y(0) \) as \( 1/\sqrt{2} (y(0) + y(1) + \cdots + y(N - 1)) \).

Among all the methods available to compute (1), those proposed in [11]–[13] appear to be best for various sequence lengths. When \( N \) is a power of 2, the algorithm [11] computes DCT directly in \( N \log_2 N - 3N/2 + 4 \) real multiplications and \( 3N/2 (\log_2 N - 1) + 2 \) real additions. The procedures of [12] and [13], on the other hand, compute DCT through a DFT of real data. It is assumed here that two real L-point DFT’s are computed through one complex L-point DFT and L complex additions [14], and that the complex DFT is implemented through the WFTA [15]. Each real L-point DFT then requires \( M_w(L) \) real multiplications and \( A_w(L) + L \) real additions where \( M_w(L) \) and \( A_w(L) \) denote the number of multiplications and additions, respectively, in Winograd's algorithm [15] of length \( L \). The DCT algorithm of [12] applicable for even \( N \), calls for a real \( N \)-point DFT followed by \( (N/2 - 1) \) complex multiplications, thus requiring a total of \( M_w(N) + 2N - 4 \) real multiplications and \( A_w(N) + 2N - 2 \) real additions. The algorithm of [13], applicable for arbitrary \( N \), requires the evaluation of the real part of a real 2N-point DFT followed by \( N \) real multiplications. It thus involves \( M_w(2N) + N \) real multiplications and \( A_w(2N) + N \) real additions.

In this paper, a new DCT algorithm based on the short-length cyclic convolutions is proposed. This algorithm is generally more efficient than the algorithms mentioned earlier and is applicable to sequences with arbitrary number of points.

Section II of this paper presents the required group theoretic fundamentals and briefly reviews the earlier work in this direction. The DCT algorithms are constructed in Section IV using the theoretical background developed in Section III. The construction is illustrated throughout with an algorithm of a 10-point DCT. Finally, the computational complexity of the algorithm developed is analyzed and compared with that of the conventional algorithms [11]–[13] in Section V.

II. GROUP TABLES AND TRANSFORM COMPUTATION

In this paper we will be concerned with Abelian groups of positive integers relatively prime to \( N \) and less than \( N \) under the operation of multiplication modulo \( N \) for integer \( N \). The group and the group operation will be denoted by \( \mathcal{A}(N) \) and \( \oplus \). The inverse of \( g \) will be denoted by \( \Theta g \), \( ng \) will mean \( g \oplus \cdots \oplus n \) times, and \( h \oplus g = h \oplus (\Theta g), g, h \in \mathcal{G} \).

Example 1: The integers \( \{1, 3, 7, 9, 11, 13, 17, 19, 21, 23, 27, 29, 31, 33, 37, 39\} \) from a group \( \mathcal{A}(40) \) under the operation of multiplication modulo 40. In this group, \( 17 \oplus 37 = 629 \mod 40 = 29, 9 \oplus 11 = 99 \mod 40 = 19, \text{etc.} \)

The structure of \( \mathcal{A}(N) \) is fully determined by the value of \( N \). In general, one has the following results (see [16, Theorems III.2.m and III.2.p]).
1) \( A(r_1 \cdot r_2) \cong A(r_1) \times A(r_2) \) when \( \gcd (r_1, r_2) = 1 \), where \( \cong \) denotes isomorphism and \( \times \), the direct product of groups.
2) \( A(p^n) \cong C_{p^{n-1}p^m-1} \) when \( p \) is an odd prime and \( C_p \) denotes a cyclic group of order \( p \).
3) \( A(2^n) \cong C_2 \times C_{2^{n-2}} \) if \( n \geq 3 \), \( A(2) = \{1\} \) and \( A(2^2) = C_2 \).

**Example 1 (Cont'd.):** \( A(40) \cong A(8) \times A(5) \cong C_2 \times C_2 \times C_4 \).

Furthermore, if \( A(N) \cong C_{n_1} \times C_{n_2} \times \cdots \times C_{n_r} \) it is obvious that one can find subgroups in \( A(N) \) that are isomorphic to \( C_{n_1}, C_{n_2}, \ldots, C_{n_r} \), such that the group \( A(N) \) is an internal direct product of these subgroups. Any element of \( A(N) \) can then be expressed uniquely as a sum of the elements of these subgroups. The subgroups then are called the splitting subgroups.

**Example 1 (Cont'd.):** \( A(40) = \{1, 19\} \times \{1, 21\} \times \{1, 3, 9, 27\} \) where it is easy to verify that the three subgroups are cyclic groups of orders 2, 2, and 4.

**Note:** \( 19 \oplus 19 = 1, 21 \oplus 21 = 1, 3 \oplus 3 = 9, 9 \oplus 9 = 27, 27 \oplus 3 = 1 \) and any \( h \in A(40) \) can be obtained as \( h = g_1 \oplus g_2 \oplus g_3 = g_1g_2g_3 \) mod where \( g_1 \in \{1, 19\}, g_2 \in \{1, 21\} \) and \( g_3 \in \{1, 3, 9, 27\} \). For example, \( 17 = 19 \oplus 1 \oplus 3, 31 = 19 \oplus 21 \oplus 9, 21 = 1 \oplus 21 \oplus 1 \), etc.

One can index a sequence \( u(i) \) of length \( N = n_1 \cdot n_2 \cdots n_r \) with reference to any group \( G = C_{n_1} \times C_{n_2} \times \cdots \times C_{n_r} \). Let \( a_1 \) denote the generator of \( C_{n_1} \), i.e., \( \langle a_1 \rangle = C_{n_1} \), \( 1 \leq 1 \leq r \). Then a sequence component may be interchangeably referred to as \( u(i, i_1, i_2, \ldots, i_r) \), \( 0 \leq i_1, i_2, \ldots, i_r \leq n_1, n_2, \ldots, n_r \) and \( i \) has the unique representation \( i = i_1n_1 + i_2n_2 + \cdots + i_rn_r \). For example, a sequence \( u(i) \) of length 18 will be indexed with reference to the groups \( C_3 \times C_2 \times C_2 \) as, \( u(0), u(1), u(2), u(3), u(4), u(5), u(6) \), \( u(7), u(8), u(9), u(10), u(11), u(12), u(13), u(14), u(15), u(16), u(17) \), where \( u \) is the group operation.

We now define a generalized convolution with respect to an Abelian group \( G \) [17]. Index the sequences \( u \) and \( v \) by the elements of \( G \). Then their convolved sequence \( w \) is given by

\[
w(h) = \sum_{g \in G} u(g)v(h \oplus g).\]

It has been shown earlier [18] that this convolution with respect to \( G = C_{n_1} \times C_{n_2} \times \cdots \times C_{n_r} \) can be computed through \( W = U \cdot V \) where \( U, V \), and \( W \) are \( r \)-dimensional arrays defined as

\[
U(i_1, i_2, \ldots, i_r) = u(i_1a_1 \oplus i_2a_2 \oplus \cdots \oplus i_ra_r),
\]
\[
V(i_1, i_2, \ldots, i_r) = v(i_1a_1 \oplus i_2a_2 \oplus \cdots \oplus i_ra_r),
\]
\[
W(i_1, i_2, \ldots, i_r) = w(i_1a_1 \oplus i_2a_2 \oplus \cdots \oplus i_ra_r).\]

**Theorem 1:** Consider the computation of \( Y(j) = \Sigma_{i \in M(j)} Y(i, j) \), \( i, j \in G \). If there exists a group \( G \) and the functions \( \psi_1: G \rightarrow B, \psi_2: G \rightarrow A; \delta_1, \delta_2: G \rightarrow \{1, -1\} \) and \( f: G \rightarrow \{1, -1\} \) such that

i) \( \psi_1 \) and \( \psi_2 \) are one-one and onto,

ii) \( M(\psi_1, \psi_2) = \delta_1(h)\delta_2(g)/(h \oplus g) \), \( \forall h, g \in G \),

then \( Y(\psi_1, \psi_2) = \delta_1(h)w(h), h \in G \) where \( w \) is the convolution with respect to \( G \) of the sequences \( u \) and \( v \) defined as \( u(g) = f(g), v(g) = \delta_2(\Theta g); v(\psi_1(h \oplus g)), g \in G \).

**Proof:** Refer to [18].

**III. GENERALIZED CONVOLUTION OF SEQUENCES WITH REDUNDANCIES**

Sometimes it might not be possible to find \( G, \psi_1, \psi_2 \), etc., satisfying the conditions of Theorem 1. Fortunately, in DCT computation, whenever this happens, index sets \( A \) and \( B \) can be extended to find a suitable group and the functions. Theorem 1 can then be used to compute the DCT as a convolution over a larger group \( G \). This is proved in Theorem 2 of this section. The extensions of sequences however, introduce redundancies which may be exploited to reduce the computational complexity as shown in Theorems 3 and 4.

Consider a group \( G \) containing an order 2 element \( a \). The set of coset representatives of \( [1, a] \) in \( G \) will be denoted by \( G_a \).

**Example 2:** \( A(50) = \{1, 3, 9, 27, 31, 43, 49, 47, 41, 23, 19, 7, 21, 13, 39, 17\} \) and contains \( \alpha = 49 \). Then \( G_\alpha = \{1, 3, 9, 27, 31, 43, 49, 47, 41, 23, 19, 7, 21, 13, 39, 17\} \) and \( G_\alpha \) contains \( \alpha = 49 \). (Note that \( G_\alpha \) is a coset.)

**Theorem 2 (Extension of Theorem 1):** If a group \( G \) and functions \( \psi_1: G_a \rightarrow B, \psi_2: G_a \rightarrow A; \delta_1, \delta_2: G_a \rightarrow \{1, -1\} \) and \( f: G_a \rightarrow \{1, -1\} \) exist such that,

i) \( \psi_1, \psi_2 \) are one-one and onto,

ii) \( f(\alpha \oplus \Theta g) = f(g), g \in G \), and

iii) \( M(\psi_1, \psi_2) = \delta_1(h)\delta_2(g)/(h \oplus g) \), \( \forall h, g \in G_a \), then \( Y(\psi_1, \psi_2) = \delta_1(h)w(h), h \in G_a \), where \( w \) is the convolution with respect to \( G \) of the sequences \( u \) and \( v \) defined as \( u(g) = f(g)/2, g \in G \) and \( v(g) = \delta_2(g^\prime)\psi(\psi_2g^\prime), g \in G \).

where

\[
g^\prime = \Theta g \text{ if } \Theta g \in G_a, \quad = \alpha \oplus \Theta g \text{ if } \Theta g \notin G_a.
\]

**Proof:** Extend the functions \( \psi_1, \psi_2, \delta_1, \) and \( \delta_2 \) to \( G \) as, \( \psi_1(g \oplus \alpha) = \psi_1(g), \psi_2(g \oplus \alpha) = \psi_2(g), \delta_1(g \oplus \alpha) = \delta_1(g) \) and \( \delta_2(g \oplus \alpha) = \delta_2(g) \), \( g \in G_a \). Then index sets \( B \) and \( A \) get extended to \( B \cup B^\prime \) and \( A \cup A^\prime \), respectively, where

\[
B = \{\psi_1g \mid g \in G_a\}, B^\prime = \{\psi_1g \mid g \in G, g \notin G_a\},
\]
\[
A = \{\psi_2g \mid g \in G_a\}, A^\prime = \{\psi_2g \mid g \in G, g \notin G_a\}.
\]

These extended index sets and functions satisfy the requirements of Theorem 1 giving the stated result.
Often, one comes across situations in which conditions i) and iii) of Theorem 2 are satisfied and in place of ii) one has

ii$'$ $f(g \otimes \alpha) = -f(g)$, $g \in G$.

Procedure of Theorem 2 can then still be used to compute $Y$ but for the redefinition of the sequence $v$ as

$$v(g) = \delta_{\alpha}(g)\delta_{g'}(g')v(\psi g'), \quad g \in G$$

where $g' = \Theta g$, $\delta_{\alpha}(g) = 1$, if $\Theta g \in G_{\alpha}$ and $g' = \alpha \otimes g$, $\delta_{g'}(g) = -1$ otherwise.

Note that now, $v(g \otimes \alpha) = -v(g)$, $g \in G$, whereas in Theorem 2, $v(g \otimes \alpha) = v(g)$, $g \in G$.

**Theorem 3:** Let $\alpha \in G$ be an order 2 element such that $u(g) = u(g \otimes \alpha)$, $v(g) = v(g \otimes \alpha)$, $\forall g \in G$. Then their convolution $\omega$ can be computed as a multidimensional cyclic convolution of patterns with only $|G|/2$ points each.

**Proof:** One can express $G$ as $G \simeq C_{2n} \times C_{n_3} \times C_{n_3} \times \cdots \times C_{n_r}$, where $\alpha \in C_{2n}$. Then in (3), one has $U(i_1, i_2, \ldots, i_r) = U(i_1 + 2^{n-1}, i_2, \ldots, i_r) V(i_1, i_2, \ldots, i_r) = V(i_1 + 2^{n-1}, i_2, \ldots, i_r)$

$$0 \leq i_1 \leq 2^{n-1} - 1, \ 0 \leq i_t \leq n_t - 1, \quad 2 \leq t \leq r.$$

Consequently,

$$W(i_1, i_2, \ldots, i_r) = W(i_1 + 2^{n-1}, i_2, \ldots, i_r)$$

and $W' = 2 \cdot U' * V'$

where $U'$, $V'$, and $W'$ are identical to $U$, $V$, and $W$, respectively, except that their first index $i_1$ varies only from 0 to $2^{n-1} - 1$. Thus $W'$ can be computed as an $r$-dimensional cyclic convolution of two patterns with $2^{n-1} \times n_3 \times n_3 \times \cdots \times n_r$ points.

**Example 3:** Convolution of $u = (0123101231)$ and $v = (2346023460)$ with respect to $C_4 \times C_3$. Then from (2) and (3)

$$U = \begin{bmatrix} 0 & 2 & 0 & 2 \\ 7 & 1 & 1 & 1 \end{bmatrix}, \quad V = \begin{bmatrix} 2 & 4 & 2 & 4 \\ 3 & 0 & 3 & 0 \end{bmatrix}.$$.

Therefore,

$$U' = \begin{bmatrix} 0 & 2 \\ 7 & 1 \end{bmatrix}, \quad V' = \begin{bmatrix} 2 & 4 \\ 3 & 0 \end{bmatrix}, \quad W = \begin{bmatrix} 86 & 114 & 86 & 114 \\ 110 & 66 & 100 & 66 \\ 92 & 144 & 92 & 144 \end{bmatrix}.$$

Finally,

$$w = [86 \ 110 \ 92 \ 114 \ 66 \ 144 \ 86 \ 110]$$

and $w = [86 \ 110 \ 92 \ 114 \ 66 \ 144 \ 86 \ 110].$

Note that the computation of the convolution was effected through $U' * V'$ where each $U'$ and $V'$ has half the number of points of those of $U$ and $V$.

**Theorem 4:** Let $\alpha \in G$ be an order 2 element such that $u(g) = -u(g \otimes \alpha)$, $v(g) = -v(g \otimes \alpha)$, $\forall g \in G$. Then the number of operations required to evaluate their convolution with respect to $G$ can be considerably reduced.

**Proof:** Proceeding as in the earlier proof, one now has

$$U(i_1, i_2, \ldots, i_r) = -U(i_1 + 2^{n-1}, i_2, \ldots, i_r)$$

$$V(i_1, i_2, \ldots, i_r) = -V(i_1 + 2^{n-1}, i_2, \ldots, i_r)$$

$$0 \leq i_1 \leq 2^{n-1} - 1, \ 0 \leq i_t \leq n_t - 1, \quad 2 \leq t \leq r.$$

Thus, $W(i_1, i_2, \ldots, i_r) = -W(i_1 + 2^{n-1}, i_2, \ldots, i_r)$ and one needs to compute $W$ only for $0 \leq i_1 \leq 2^{n-1} - 1; 0 \leq i_t \leq n_t - 1, \ 2 \leq t \leq r$. The use of fast algorithms [19] to compute the $r$-dimensional cyclic convolution of $U$ and $V$ calls for a linear transformation of these patterns. It can be verified that for $U$, $V$ satisfying (4), many of the components of the transformed sequences are zero and operations due to them are saved.

Based on the algorithms of [19], one can easily construct modified algorithms as given in Appendix A to convolve sequences satisfying the conditions of Theorem 4 for $G = C_{2n}$, $n = 1, 2, 3$. When $G = C_{2n} \times C_{n_3} \times C_{n_3} \times \cdots \times C_{n_r}$, the modified algorithm of convolution with respect to $G$ can be obtained by combining the modified algorithm for $C_{2n}$ with the cyclic algorithms of lengths $n_3, n_3, \ldots, n_r$. Table 1 compares the number of operations required to implement the convolution of sequences satisfying Theorems 3 and 4 with those of [19].

**Example 4:** Convolution of $u = (0123105767231)$ and $v = (2346023460)$ with respect to $C_4 \times C_3$. As in Example 3, one now has

$$U = \begin{bmatrix} 0 & 2 & 0 & -2 \\ 5 & 3 & -5 & -3 \\ 7 & 1 & -7 & -1 \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} 2 & 4 & -2 & -4 \\ 2 & 6 & -2 & -6 \\ 3 & 0 & -3 & 0 \end{bmatrix}.$$.

To compute $U * V$ as per [19], write $U$ and $V$ as $U = [u_0, u_1, -u_0, -u_1]$ and $V = [v_0, v_1, -v_0, -v_1]$ where $u_0, u_1, v_0, v_1$ stand for the columns. Applying the modified Algorithm 2 of Appendix A, one has

$$c_0 = \begin{bmatrix} 0 & 10 \\ 14 & 12 \end{bmatrix}, \quad c_1 = \begin{bmatrix} -4 & 4 \\ 16 & 16 \end{bmatrix}, \quad c_2 = \begin{bmatrix} 4 \end{bmatrix}.$$
The product \( m_t = c_t d_t \) is now to be interpreted as a length-3 cyclic convolution of \( c_t \) and \( d_t \) as per [19]. One then has

\[
\begin{bmatrix}
  \begin{array}{c}
    6 \\
    8 \\
    3 \\
  \end{array}
\end{bmatrix}, \quad \begin{bmatrix}
  \begin{array}{c}
    2 \\
    2 \\
    3 \\
  \end{array}
\end{bmatrix}, \quad \text{and} \quad \begin{bmatrix}
  \begin{array}{c}
    6 \\
    6 \\
    0 \\
  \end{array}
\end{bmatrix}.
\]

Finally, going back to Algorithm 2 of Appendix A,

\[
\begin{bmatrix}
  \begin{array}{c}
    142 \\
    102 \\
    164 \\
  \end{array}
\end{bmatrix}, \quad \begin{bmatrix}
  \begin{array}{c}
    28 \\
    36 \\
    20 \\
  \end{array}
\end{bmatrix}, \quad \text{and} \quad \begin{bmatrix}
  \begin{array}{c}
    112 \\
    88 \\
    160 \\
  \end{array}
\end{bmatrix}
\]

Thus

\[
\begin{bmatrix}
  \begin{array}{c}
    30 \\
    14 \\
    4 \\
  \end{array}
\end{bmatrix}, \quad \begin{bmatrix}
  \begin{array}{c}
    114 \\
    66 \\
    144 \\
  \end{array}
\end{bmatrix}, \quad \begin{bmatrix}
  \begin{array}{c}
    -30 \\
    -14 \\
    -4 \\
  \end{array}
\end{bmatrix}, \quad \begin{bmatrix}
  \begin{array}{c}
    -114 \\
    -66 \\
    -144 \\
  \end{array}
\end{bmatrix}
\]

or \( w = [30 \ 14 \ 114 \ 144 \ 66 \ -30 \ -14 \ -114 \ -144 \ -66] \).

\[
\text{IV. DEVELOPMENT OF THE ALGORITHMS}
\]

The computation of all those DCT components for which \( \gcd(j, 2N) = t \) is done together. First consider the case \( t \mid N \), i.e., \( t \) dividing \( N \). Utilizing the properties of the cosine function, (1) can be written as

\[
Y(j) = \sum_{s \equiv A \pmod{N/r}} X_s(j),
\]

where

\[
X_s(j) = \sum_{i \in \mathbb{A}} M(j, i)x_i(i),
\]

\[
A = \{0 \leq i < N/2t : \gcd(2i + 1, N/t) = s\},
\]

\[
x_i(i) = z_i(i) - z_i(N/t - 1 - i),
\]

\[
i = 0, 1, \cdots, \lfloor N/2t \rfloor - 1,
\]

\[
z_i(i) = \sum_{d = 0}^{r-1} (-1)^d y(i + dN/t),
\]

\[
i = 0, 1, \cdots, N/t - 1.
\]

\[
\lfloor \cdot \rfloor \text{ denotes the integer part. Unless the context demands } x_i \text{ will be referred to merely as } x_i. \text{ Further, because } X_s(j) = X_s(j_2) \text{ if } (j_1 + j_2)/(2N/s) \text{ or } (j_1 - j_2)/(2N/s) \text{ is an even integer and } X_s(j_1) = X_s(j_2) \text{ if it is an odd integer, one may partition the set of } j_i \text{ under consideration into subsets such that } j_1, j_2 \text{ will belong to the same subset iff } (2N/s)(j_1 + j_2) \text{ or } (j_1 - j_2). \text{ Equation (5) then needs to be evaluated only for } j \in B \text{ where } B \text{ denotes the representatives of these subsets. It will be shown later that } A \text{ and } B \text{ have the same number of elements.}
\]

\[
\text{Example 5: Computation of the DCT of length } N = 10 \text{ of } y = (0 \ 2 \ 1 \ 1 \ 3 \ -1 \ 0 \ 0 \ 2 \ -1). \text{ Possible values of } t \text{ dividing } N \text{ are } 1, 2 \text{ and } 5.
\]

When \( t = 1, Y(1), Y(3), Y(7), \text{ and } Y(9) \text{ are being computed. One has, then, } Y(j) = X_1(j) + X_3(j) \text{ where }
\]

\[
X_1(j) = \sum_{i \in \{0, 1, 2, 3, 4\}} M(j, i)x_1(i), \quad j \in \{1, 3, 7, 9\},
\]

\[
X_3(j) = \sum_{i \in \{2\}} M(j, i)x_3(i), \quad j \in \{1\},
\]

and \( x_1 = (10114) \). Note that for \( (t, s) = (1, 5), \text{ since } A \text{ is a one element set so should be } B. \text{ The set of } j's \text{ is } \{1, 3, 7, 9\}. \text{ One can use the value of } 2N/s = 4 \text{ to show that all the elements of this set are in the same subset as defined earlier. Thus only one may take } B = \{1\} \text{ in (9) and, } X_3(5) = X_3(1) = (3 + 1)/(2N/s) \text{ is odd and similarly } X_3(7) = X_3(9) = X_3(1) = (7 + 1)/(2N/s) \text{ and } (9 - 1)/(2N/s) \text{ are even. When } t = 2, Y(2), \text{ and } Y(6) \text{ are computed as } Y(j) = X_1(j) \text{ where }
\]

\[
X_1(j) = \sum_{i \in \{0, 1, 2, 3, 4\}} c_j(i)x_2(i), \quad j \in \{2, 6\}
\]

and \( x_2 = (-3, 3) \).

When \( t = 5, Y(5) \text{ is being computed as } Y(j) = X_1(j) \text{ where }
\]

\[
X_1(j) = \sum_{i \in \{0, 1, 2, 3, 4\}} c_j(i)x_5(i), \quad j \in \{5\}
\]

and \( x_5 = (5) \).

It will now be shown that a certain choice of \( G, \psi_1, \psi_2, \delta_1, \delta_2, \text{ and } f \text{ converts (5) for } j \in B \text{ into a convolution with respect to } G. \text{ Let } L = N/st \text{ and } H = A(4L). \text{ Then } \alpha = 2L + 1 + \beta = 2L + 1 \in H \text{ and are of order } 2. \text{ The choice of } G \text{ is dictated by the nature of } (\alpha) \text{ and } (\beta).
\]

First, at least one of \( (\alpha) \text{ or } (\beta) \text{ should be a splitting subgroup of } H \text{ because if neither of them is splitting subgroups, then both } \alpha \text{ and } \beta \text{ should have square-roots modulo } 4L. \text{ But } \alpha \beta = -1 \text{ (mod } 4L). \text{ Thus } -1 \text{ should have a square-root modulo } 4L \text{ which is impossible. So we have the following two cases.}
\]

Case 1: If both \( (\alpha) \text{ and } (\beta) \text{ are splitting subgroups of } H, \text{ i.e., } H \cong (\alpha) \times (\beta) \times G, \text{ one can establish a one-one correspondence between the elements of } G \text{ and } B \text{ as under. For every } g \in G, \text{ there exist integers } n_1 \text{ and } d_1 \text{ such that}
\]

\[
g = (-1)^{n_1} + d_1 \cdot 2L.
\]
for some \( j \in B \). If (12) is satisfied, \( j \) is defined as \( \psi_1 g \). The one-one and onto nature of \( \psi_1 \) is proved in Appendix B. Equation (12) also defines \( \delta_1 \) as \( \delta_1(g) = (-1)^{d_1} \). Similarly, one may find integers \( n_2 \) and \( d_2 \) satisfying

\[
g = (-1)^{n_2}(2i + 1)/s + d_2 \cdot 2L
\]

where \( g \in G, i \in A \). This establishes a one-one correspondence between the elements of \( G \) and \( A \). The proof for this is omitted as it runs parallel to that of (12). We then define

\[
\psi_2 g = i \quad \text{and} \quad \delta_2(g) = (-1)^{d_2}.
\]

Furthermore, by choosing \( f \) as

\[
f(g) = \cos(g \pi/2L),
\]

it can be verified that the requirements of Theorem 1 are satisfied and (5) may be computed by defining sequences \( u \) and \( v \) as \( u(g) = \cos(g \pi/2L) \) and \( v(g) = \delta_2(\Theta g)x_1(\psi_2(\Theta g)) \). In \( G \) and convolving them with respect to \( G \) to get the sequence \( W \) related to \( X_2 \) as \( X_2(\psi^j h) = \delta_1(h)w(h), h \in G \), where \( W \) is the convolution with respect to \( G \) of sequences \( u \) and \( v \) defined as

\[
u(g) = -v(\Theta g) = \delta_2(g')\delta_2(\Theta g)x_1(\psi_2(g')), \quad g \in G_a
\]

where

\[
g' = \Theta g \quad \text{if} \quad \Theta g \in G_a
\]

\[
= \alpha g \quad \text{otherwise}.
\]

Note that \( u \ast v \) can be evaluated efficiently as in Theorem 4.

Example 5 (Cont'd.): For \(( t, s) = (1, 1) \) (computation of (10)), \( L = 5 \), \( \alpha = 9 \), and \( \beta = 11 \). \( A(4L) = \{1, 11\} \times \{1, 3, 9, 7\} = \{\alpha\} \times \{\beta\} \times G \). Thus \( G_a = \{1, 3\} \). Equations (12) and (13) are satisfied by following values of \( n_1 \), \( d_1 \), \( n_2 \), and \( d_2 \). Each \( \delta_2(g') \) and \( \psi_2(g') \) is computed for \( g' \) rather than for \( g \) because one needs \( \delta_2(g') \) and \( \psi_2(g') \).

Finally, a convolution with respect to \( G \) (which is a cyclic convolution of length 4 as \( G = C_4 \)) of \( U \) and \( V \) gives \( W \) where

\[
U = \cos(\pi/20), \cos(3\pi/20), \cos(9\pi/20), \cos(27\pi/20),
\]

\[
V = (x_1(0), -x_1(3), x_1(4), x_1(1)) = (-1, -1, 0, 0)
\]

\[
W = U \ast V = \left( X_1(1), X_1(3), X_1(9), -X_1(7) \right) = (2.068, -1.913, 3.216, 2.954).
\]

For \(( t, s) = (1, 3) \) (computation of (9)), \( L = 2 \), \( H = A(8) = C_2 \times C_2 = \{1, 3\} \times \{1, 1\} \). \( X_1(2) = \{\alpha\} \times \{\beta\} \). Thus \( G = \{1\} \) and \( X_1(3) \) is a convolution with respect to \( G \) (i.e., a simple multiplication) of \( U = \cos(\pi/4) \) and \( x_3(2) = (1) \). Thus \( X_1(1) = 0.707 \). Similarly, for \(( t, s) = (5, 1) \) (computation of (11)), \( G = \{1\} \) and \( X_1(5) = \cos(\pi/4) \times x_5(0) = 3.535 \).

Case 2: If only one of \( \{\alpha\} \) or \( \{\beta\} \) is a splitting subgroup of \( H \), then \( H \simeq \{\alpha\} \times G \) or \( H \simeq \{\beta\} \times G \). We assume here that \( H \simeq \{\beta\} \times G \) but the case of \( H \simeq \{\alpha\} \times G \) may be worked out similarly.

Note that now \( \alpha \in G \). It may be verified that defining \( \psi_1, \psi_2, \delta_1, \delta_2, \) and \( f \) as in Case 1, the requirements of Theorem 2 are satisfied except that

\[
f(g \oplus \alpha) = \cos((\alpha \oplus \alpha) \pi/2L)
\]

\[
= \cos(\alpha g \pi/2L) \quad \text{as} \quad \oplus \text{denotes mult.mod} \, 4L
\]

\[
= \cos((2L - 1)g \pi/2L)
\]

\[
= -\cos(g \pi/2L) = -f(g).
\]

Therefore, following the remark after Theorem 2, one gets the algorithm in this case, as \( X_2(\psi^j h) = \delta_1(h)w(h), h \in G_a \), where \( w \) is the convolution with respect to \( G \) of sequences \( u \) and \( v \) defined as

\[
u(g) = -u(\Theta g) = \delta_2(g')\delta_2(\Theta g)x_1(\psi_2(g')), \quad g \in G_a
\]

where

\[
g' = \Theta g \quad \text{if} \quad \Theta g \in G_a
\]

\[
= \alpha g \quad \text{otherwise}.
\]

Note that \( u \ast v \) can be evaluated efficiently as in Theorem 4.

Example 5 (Cont'd.): For \(( t, s) = (2, 1) \) (computation of (10)), \( L = 5 \), \( \alpha = 9 \), and \( \beta = 11 \). \( A(4L) = \{1, 11\} \times \{1, 3, 9, 7\} = \{\beta\} \times G \). Thus \( G_a = \{1, 3\} \). Equations (12) and (13) are satisfied by following values of \( n_1 \), \( d_1 \), \( n_2 \), and \( d_2 \). Each \( g' \) is computed for \( g' \) rather than for \( g \) because one needs \( \delta_2(g') \) and \( \psi_2(g') \).

Finally, a convolution with respect to \( G \) (which is a cyclic convolution of length 4 as \( G = C_4 \)) of \( U \) and \( V \) gives \( W \) where

\[
U = \cos(\pi/10)/2, \cos(3\pi/10)/2, \cos(\pi/10)/2, \cos(3\pi/10)/2
\]

\[
V = (x_2(0), -x_2(1), -x_2(0), x_2(1)) = (-3, -3, 3, 3).
\]

Thus \( U \ast V = W = (-1.089, -4.617, 1.089, 4.617) \) or \((-1.089, -4.617) = (X_1(2), X_1(3)) \).

Note that since \( U \) and \( V \) satisfy conditions of Theorem 4, one can use modified Algorithm 2 of Appendix A to implement this convolution.

For \( t \not\mid N, (1) \) can be reduced to

\[
Y(j) = (-1)^{\lfloor j/N \rfloor} X_j(j) + \sum_{i \equiv odd \{j/N\}} X_i(j)
\]

where

\[
X_j(j) = \sum_{i \in A} M(j, i) x_i(i).
\]

\[
x_i(i) = z_i(i) + z_i(2N/t - 1 - i),
\]

\[
i = 0, 1, \ldots, [N/t] - 1
\]

\[
z_i(i) = \sum_{d=0}^{u/2-1} y(i + d N/t),
\]

\[
i = 0, 1, \ldots, 2N/t - 1
\]

\[
A = \{0 \leq i < N/t \mid \gcd(2i + 1, 2N/t) = s\}.
\]
Furthermore, because of the arguments similar to those in the case of \( t \mid N \), (16) need be computed only for \( j \in B \), where \( B \) is a set defined as earlier.

**Example 5 (Cont'd.):** The only value of \( t \mid N \) but \( t \mid 2N \) is 4 and for this \( Y(4) \) and \( Y(8) \) are being computed. One then has

\[
Y(4) = -z_4(2) + X_1(4)
\]

\[
Y(8) = z_4(2) + X_1(8)
\]

where

\[
X_1(j) = \sum_{i \in [0,1]} M(j, i) x_4(i), \quad j \in [4, 8] \tag{19}
\]

\[
z_4 = (-1, 2, 4, 3, 2), \quad x_4 = (1, 5).
\]

Let \( L = 2N/4 \) and \( H = A(2L) \). Then \( \alpha = 2L - 1 \in H \) is of order 2. The choice of \( G \) is dictated by the nature of \( \alpha \) and the following two cases arise:

**Case 3:** If \( \alpha \) is a splitting subgroup of \( H \), \( H \simeq \langle \alpha \rangle \times G \). One can then establish a one-one onto relation between the group elements \( g \) and the elements \( j \) of \( B \) as

\[
g = (-1)^{n_j} \begin{pmatrix} 1 \\ j \end{pmatrix} + d_1 L \tag{20}
\]

Then \( \psi_1(g) \) and \( \delta_1(g) \) are defined as \( j \) and \( (-1)^d_1 \), respectively. \( \psi_2 \) can be obtained as in (13) and \( \delta_2 \) may be taken as \( \delta_2(g) = 1 \), \( \forall \ g \in G \). Finally, choosing \( f \) as

\[
f(g) = \cos(g \pi/2) \tag{21}
\]

requirements of Theorem 1 may be shown to be satisfied. Consequently, the required algorithm for (16) will read as

\[
X_1(\psi_1 h) = \delta_1(h) w(h), \quad h \in G
\]

where \( w \) is the convolution with respect to \( G \) of the sequences \( u \) and \( v \) defined as

\[
u(g) = x_1(\psi_2(g \oplus g)), \quad \forall \ g \in G.
\]

**Case 4:** When \( \langle \alpha \rangle \) is not a splitting subgroup of \( H \), let \( G = H \). Now \( \alpha \in G \). By defining \( \psi_1, \psi_2, \delta_1, \), and \( \delta_2 \) and as in Case 3, it can be easily verified that the requirements of Theorem 2 are fulfilled. In particular, now \( f(g \oplus \alpha) = f(g) \), \( g \in G \).

(Compare with Case 2.) The algorithm in this case thus reads as

\[
u(g) = u(g \oplus \alpha) = \cos(g \pi/2) / 2
\]

\[
v(g) = v(g \oplus \alpha) = x_2(\psi_2(\alpha g))
\]

where

\[
g' = \alpha \oplus g, \quad \text{if} \ \Theta g \in G_n \]

\[
g' = g, \quad \text{otherwise}
\]

Theorem 3 may be utilized to evaluate this convolution efficiently.

**Example 5 (Cont'd.):** For \( t = (1, 1) \) [computation of (19)], one has \( L = 5 \), \( H = A(2L) = C_5 \). Thus \( G = C_4 = \{3, 1, 39, 7\} \). Note that \( \alpha = 9 \in G \). Corresponding to (20) and (13) we have (13) is evaluated for \( g' \) rather than for \( g \) as one needs \( \psi_2(g') \)

\[
g \in G_n \quad n_1 \quad d_1 \quad \psi_1 g \in B \quad \Theta g \quad g' \quad n_2 \quad d_2 \quad \psi_2 g' \in A
\]

\[
\begin{align*}
1 & 0 & 0 & 4 & 1 & 1 & 0 & 0 & 0 \\
3 & 1 & 1 & 8 & 7 & 3 & 0 & 0 & 1
\end{align*}
\]

Thus a convolution with respect to \( C_4 \) of \( W \) and \( V \) gives \( X_1(\psi_1) \), where

\[
U = (\cos(\pi/3), \cos(3\pi/3)/2, \cos(\pi/5)/2, \cos(3\pi/5)/2)
\]

\[
V = (x_4(0), x_4(1), x_4(0), x_4(1))
\]

\[
= (1, 5, 1, 5)
\]

\[
W = U \ast V = (X_1(\psi_1), -X_1(\psi_1), X_1(\psi_1), -X_1(\psi_1))
\]

from Theorem 3, \( U \ast V \) can be evaluated as a length-2 cyclic convolution of \( (\cos(\pi/5), \cos(3\pi/5)) \) and \( (1, 5) \) to give \( \{X_1(\psi_1), -X_1(\psi_1)\} = (-0.736, 3.736) \).

**V. Computational Complexity**

The algorithm developed thus has three computational stages.

i) For every \( t \subset \langle \alpha \rangle \subset \langle \alpha \rangle \), compute \( x_r \) sequences through (6), (7), (17), and (18).

ii) Use the \( x_r \)'s so computed in various convolutional algorithms.

iii) Recombine the results of the convolutions as per (5) and (15) to get the transform.

We now analyze the computational complexity of each of these stages.

i) No multiplications are involved at this stage. The number of additions involved is substantially reduced by adopting the following scheme (proofs omitted as they can be easily constructed).

**To Compute \( x_r \), for \( t \mid N \)**

Let \( p \) be the largest integer satisfying \( 2^p < N, 2^p \mid N \). Then one needs to compute \( x_r, n = 0, 1, \ldots, p \). Define \( r_0(i) = y(i), i = 0, 1, \ldots, N - 1 \),

\[
r_n(i) = r_{n-1}(i) + r_{n-1}(N/2^n - 1 - i), \quad i = 0, 1, \ldots, N/2^n - 1, \quad 1 \leq n \leq p
\]

Then

\[
x_r(i) = r_n(i) - r_n(N/2^n - 1 - i), \quad i = 0, 1, \ldots, [N/2^{p+1}] - 1, \quad 0 \leq n \leq p
\]

These involve \( 2N(1 - 1/2^p) + [N/2^{p+1}] \) additions. To compute \( x_r \) from \( x_r \) where \( t_1, t_2 \mid N \) and \( t = k \) for an odd integer \( k \), one may use

\[
x_r(i) = \sum_{q=0}^{k/2^j} (-1)^q x_r(i + Nq/t_1)
\]

\[
- \sum_{q=0}^{k/2^j+1} (-1)^q x_r(N/2^n - 1 - i - Nq/t_1)
\]

requiring only \((k - 1) [N/2^j] \) additions.

**To Compute \( x_r \), for \( t \mid N \)**

Such \( t \)'s exist iff \( N \) is not a power of 2. Then
TABLE II
A COMPARISON OF THE COMPUTATIONAL COMPLEXITY OF THE
New DCT Algorithm with the Best of the Conventional

<table>
<thead>
<tr>
<th>Sequence length</th>
<th>New DCT Multiplications</th>
<th>New DCT Additions</th>
<th>Conventional DCT Multiplications</th>
<th>Conventional DCT Additions</th>
<th>WFTA Multiplications</th>
<th>WFTA Additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>24</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>9</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>15</td>
<td>15</td>
<td>5</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>14</td>
<td>23</td>
<td>100</td>
<td>8</td>
<td>43</td>
</tr>
<tr>
<td>8</td>
<td>14</td>
<td>32</td>
<td>16</td>
<td>26</td>
<td>2</td>
<td>34</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
<td>46</td>
<td>29</td>
<td>126</td>
<td>10</td>
<td>54</td>
</tr>
<tr>
<td>13</td>
<td>21</td>
<td>110</td>
<td>53</td>
<td>300</td>
<td>20</td>
<td>137</td>
</tr>
<tr>
<td>21</td>
<td>47</td>
<td>195</td>
<td>73</td>
<td>384</td>
<td>24</td>
<td>171</td>
</tr>
<tr>
<td>29</td>
<td>96</td>
<td>381</td>
<td>219</td>
<td>1018</td>
<td>95</td>
<td>480</td>
</tr>
<tr>
<td>50</td>
<td>53</td>
<td>259</td>
<td>90</td>
<td>250</td>
<td>34</td>
<td>222</td>
</tr>
<tr>
<td>51</td>
<td>81</td>
<td>418</td>
<td>191</td>
<td>1080</td>
<td>80</td>
<td>509</td>
</tr>
<tr>
<td>60</td>
<td>65</td>
<td>374</td>
<td>184</td>
<td>562</td>
<td>68</td>
<td>504</td>
</tr>
<tr>
<td>65</td>
<td>151</td>
<td>1048</td>
<td>315</td>
<td>2190</td>
<td>125</td>
<td>1050</td>
</tr>
<tr>
<td>99</td>
<td>325</td>
<td>1983</td>
<td>559</td>
<td>3630</td>
<td>230</td>
<td>1716</td>
</tr>
<tr>
<td>129</td>
<td>613</td>
<td>2575</td>
<td>1045</td>
<td>5748</td>
<td>458</td>
<td>2745</td>
</tr>
<tr>
<td>258</td>
<td>1229</td>
<td>6943</td>
<td>1628</td>
<td>6004</td>
<td>916</td>
<td>5748</td>
</tr>
<tr>
<td>511</td>
<td>3097</td>
<td>17955</td>
<td>6071</td>
<td>35236</td>
<td>2780</td>
<td>17107</td>
</tr>
<tr>
<td>631</td>
<td>8361</td>
<td>40626</td>
<td>17351</td>
<td>86296</td>
<td>8360</td>
<td>42517</td>
</tr>
<tr>
<td>1262</td>
<td>16723</td>
<td>85774</td>
<td>19240</td>
<td>87556</td>
<td>16720</td>
<td>86296</td>
</tr>
</tbody>
</table>

\[ x_{2p+1}(i) = r_p(i) + r_p(N/2^p - 1 - i). \]

This calls for \([N/2^{p+1}]\) additions.

Similarly, to get \(x_{t_1}\) from \(x_{t_2}\) where \(t_1, t_2\) divide \(N\) and \(t_1 = kt_2\), one may use

\[
x_{t_1}(i) = \sum_{q=0}^{\lfloor k/2 \rfloor} x_{t_2}(i + 2qN/t_1) + \sum_{q=0}^{k-1} x_{t_2}(2N/t_2 - 1 - i - 2qN/t_1) \quad i = 0, 1, \ldots, [N/t_1] - 1,
\]

requiring only \((k - 1)[N/t_1] + [k/2] + 1\) additions.

Note from (15) that when \(i = [N/t_1] + [N/t_1]\), also is required which could be computed as

\[ z_{2p+1}([N/2^{p+1}]) = r_p([N/2^p] - 1) \]

requiring no extra operations and for \(t_1 = kt_2\),

\[
z_{t_1}([N/t_1]) = \sum_{q=0}^{\lfloor k/2 \rfloor - 1} x_{t_2}([N/t_2] + 2qN/t_1) + z_{t_2}([N/t_2])
\]

requiring only \([k/2]\) additions.

Finally, to compute \(Y(0)\), one needs to sum up all the \(y\) components. Let \(m\) be the smallest odd prime dividing \(N\). Then

\[ y(0) + y(1) + \ldots + y(N-1) = \sum_{i=0}^{m/2 - 1} x_{2N/m}(i) + y([N/2]) \]

requiring only \([m/2]\) additions. When \(N\) is a power of 2, \(y(0) + y(1) + \ldots + y(N-1) = r_p(0) + r_p(1)\) requiring only one addition.

ii) To determine the computational complexity of stage ii), note that each \((t, s)\) pair leads to a multidimensional cyclic convolution decided by the group which in turn is determined by \(N/ts\). The complexity of this stage is therefore obtained by summing the complexities of these convolutions over all possible \((t, s)\) pairs.

iii) This stage also does not involve any multiplications. Let \(S_t\) denote the total number of \(s\) values possible for any \(t\) and \(J_t\) the number of \(j\)'s satisfying

\[ 1 \leq j \leq N - 1 | \text{gcd}(j, 2N) = t. \] (22)

Then the total additions required in this stage is

\[
\sum_{1 \leq t \leq N} (S_t - 1)J_t + \sum_{1 \leq t \leq N} S_tJ_t.
\]

where the two summations correspond to (5) and (15).

Example 6: For \(N = 10, p = 1, x_1\) and \(x_2\) can be obtained in \((2 \times 10) - (1 - 1/2) + [10/4] = 12\) additions. \(x_5\) can be obtained from \(x_1\) in \((5 - 1) [10/10] = 4\) additions. \(x_4\) can be obtained in \([10/4]\) = 2 additions. \(y(0) + \ldots + y(9)\) is obtained from \(x_4\) in \([5/2] = 2\) additions. Thus stage i) requires 20 additions. Possible \((t, s)\) pairs are \((1, 1), (1, 5), (2, 1), (5, 1), (4, 1)\). For \((t, s) = (1, 1)\), \(H = A(4N/tx) = C_2 \times C_2 \times C_4\). Thus \(G = C_4\). Convolution with respect to \(C_4\) requires 5 multiplications and 15 additions. Similarly for \((t, s) = (1, 5)\) or \((5, 1)\), one needs a convolution with respect to \(G = \{1\}\) requiring only a single multiplication. For \((t, s) = (2, 1), G = C_4\). Therefore Theorem 4 is applicable for this convolution. From Table I, this requires 2 multiplications and 3 additions. For \((t, s) = (4, 1), G = C_4\). Theorem 3 is now applicable and from Table I the convolution requires 2 multiplications and 4 additions. Stage ii) thus needs 12 multiplications and 22 additions. From the given \((t, s)\) pairs, \(S_1 = 2, S_2 = 1, S_4 = 1, S_5 = 1\). Also the number of \(j\)'s satisfying (22) gives \(J_1 = 4, J_2 = 2, J_3 = 1\) and \(J_4 = 2\). Thus stage iii) requires 6 additions. Therefore the DCT of length 10 can be evaluated using the algorithm of this paper in 12 multiplications and 48 additions. As against this, the algorithm of [12] requires 26 multiplications and 62 additions and that of [13] requires 30 multiplications and 118 additions.

The complexity of the new DCT algorithm is compared in Table I1 with that of the best of [11]-[13]. It also lists the complexity of the WFTA [15].
One-One: Substituting for \( g = \sigma \theta - 1 \) for some integers \( \sigma, g \), we get:

\[
(1 - 2^p + 2^p) \sigma = 1 - 2^p \sigma
\]

(1) \( \sigma = 2^p \) \( \Rightarrow \) \( (1 - 2^p + 2^p) \sigma = 1 \)

1. \( \sigma = 2^p \)

Algorithm 1

Algorithm 2

Algorithm 3

Algorithm 4

Theorem 4

For sequences satisfying the requirements of the modified cyclic convolution algorithms, the one-one property holds for all \( n \) and \( \omega \).

Proof: Assume that \( f(\omega) \) is a sequence satisfying the requirements of the modified cyclic convolution algorithms.

1. \( f(\omega) \) is one-one for all \( \omega \).

2. \( f(\omega) \) is one-one for all \( \omega \).

3. \( f(\omega) \) is one-one for all \( \omega \).

4. \( f(\omega) \) is one-one for all \( \omega \).

Theorem 5

For sequences satisfying the requirements of the modified cyclic convolution algorithms, the one-one property holds for all \( n \) and \( \omega \).

Proof: Assume that \( f(\omega) \) is a sequence satisfying the requirements of the modified cyclic convolution algorithms.

1. \( f(\omega) \) is one-one for all \( \omega \).

2. \( f(\omega) \) is one-one for all \( \omega \).

3. \( f(\omega) \) is one-one for all \( \omega \).

4. \( f(\omega) \) is one-one for all \( \omega \).

Theorem 6

For sequences satisfying the requirements of the modified cyclic convolution algorithms, the one-one property holds for all \( n \) and \( \omega \).

Proof: Assume that \( f(\omega) \) is a sequence satisfying the requirements of the modified cyclic convolution algorithms.

1. \( f(\omega) \) is one-one for all \( \omega \).

2. \( f(\omega) \) is one-one for all \( \omega \).

3. \( f(\omega) \) is one-one for all \( \omega \).

4. \( f(\omega) \) is one-one for all \( \omega \).
or \((g_1 - g_2)\). Thus, there are four possibilities:
\[
g_1 + g_2 = 2L \text{ or } g_1 = -g_2 + 2L = g_2 \oplus \alpha, \\
g_1 + g_2 = 4L \text{ or } g_1 = -g_2 + 4L = g_2 \oplus \alpha \oplus \beta, \\
g_1 - g_2 = 2L \text{ or } g_1 = g_2 + 2L = g_2 \oplus \beta, \\
g_1 - g_2 = 0 \text{ or } g_1 = g_2.
\]
where \(\oplus\) denotes the group operation of multiplication modulo 4L. The first three cases are not possible because \(\alpha, \beta \notin G\) but both \(g_1, g_2 \in G\).

**Onto**

Any \(b \in B\) gives \(\gcd(b/t, 2L) = 1\) or \(\gcd(b/t \text{ mod } 2L, 2L) = 1\).

Let \(b/t \text{ mod } 2L = (b/t) + k2L\).

Thus \((b/t) \text{ mod } 2L = (b/t) + k2L = h \in H\). There are four cases each of which satisfy (12) with the following value of \(n\) and \(d\):
\[
h \in G \quad n = 0 \quad d = 0 \\
h \oplus \alpha = -(b/t) + (1-k)2L \in G \quad n = 1 \quad d = 1 - k \\
h \oplus \beta = (b/t) + (1+k)2L \in G \quad n = 0 \quad d = 1 + k \\
h \oplus \alpha \oplus \beta = -(b/t) + (2-k)2L \in G \quad n = 1 \quad d = 2 - k.
\]

**REFERENCES**


Meghanad D. Wagh was born in Bombay, India, on September 23, 1948. He received the B.Tech. and Ph.D. degrees from the Indian Institute of Technology, Bombay, in 1971 and 1977, respectively. From 1971 to 1976 he was a Research Assistant and during 1977-1978 a Research Associate with the Department of Electrical Engineering, Indian Institute of Technology, Bombay. Currently he is with the Department of Electrical Engineering, Concordia University, Montreal, P. Q., Canada. His research interests include the application of group theoretic techniques to digital signal processing and computational algorithms.

H. Ganesh received the B.Sc. degree in electrical engineering from the University of Kerala, India, and the M.S. degree in electrical engineering from the South Dakota State University, Brookings. From 1976 to 1979 he was with the Department of Electrical Engineering, Indian Institute of Technology, Bombay, working for his Ph.D. degree. Currently, he is with the Faculty of Electrical Engineering, Calicut Regional Engineering College, Calicut, India.