MATHEMATICS OF COMPUTATION
VOLUME 56, NUMBER 193
JANUARY 1991, PAGES 281–296

A POLYNOMIAL APPROACH TO FAST ALGORITHMS FOR DISCRETE FOURIER-COSINE AND FOURIER-SINE TRANSFORMS

G. STEIDL AND M. TASCHE

ABSTRACT. The discrete Fourier-cosine transform (cos-DFT), the discrete Fourier-sine transform (sin-DFT) and the discrete cosine transform (DCT) are closely related to the discrete Fourier transform (DFT) of real-valued sequences. This paper describes a general method for constructing fast algorithms for the cos-DFT, the sin-DFT and the DCT, which is based on polynomial arithmetic with Chebyshev polynomials and on the Chinese Remainder Theorem.

1. INTRODUCTION

In this paper, we use standard notation. By $\mathbb{N}, \mathbb{Z}, \mathbb{R}$ and $\mathbb{C}$, we denote the set of positive integers, the ring of integers, the field of reals and the field of complex numbers. For two polynomials X, Y we let $X \bmod Y$ signify the remainder of X divided by Y.

One of the most important tools in numerical analysis and digital signal processing is the fast Fourier transform (FFT), which efficiently computes the *discrete Fourier transform of length* N (DFT(N)), a mapping of a sequence $\mathbf{x} = (x_0, \ldots, x_{N-1}) \in \mathbb{C}^N$ to its spectrum $\hat{\mathbf{x}} = (\hat{x}_0, \ldots, \hat{x}_{N-1}) \in \mathbb{C}^N$ defined by

$$\hat{x}_k := \sum_{j=0}^{N-1} x_j w_N^{jk}, \qquad w_N := \exp(-2\pi i/N).$$

Using polynomial arithmetic, the formulation of many FFT-algorithms can be greatly simplified and their derivation seems more natural [1, 3, 10, 16]. Further, the polynomial notation can be utilized for considerations of the computational complexity of FFT's [6, 17].

In order to introduce a polynomial representation of the DFT, we represent the input sequence $\mathbf{x} \in \mathbb{C}^N$ of the DFT(N) as the polynomial

$$X(z) := \sum_{j=0}^{N-1} x_j z^j.$$

Received April 3, 1989; revised December 12, 1989.
1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 94A11, 42-04, 33A65.
Key words and phrases. Discrete Fourier-cosine transform, discrete Fourier-sine transform, discrete cosine transform, polynomial arithmetic, Chebyshev polynomials, computational complexity, discrete Fourier transform.

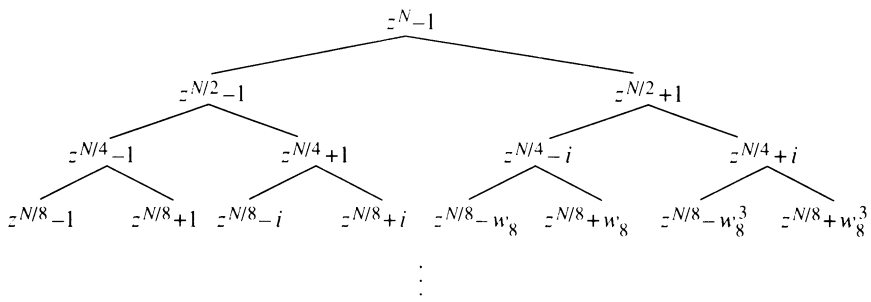

FIGURE 1
Factorization tree of $z^N - 1$ *with* $N = 2^r$ $(r \geq 3)$

Then we have $\hat{x}_k = X(w_N^k)$ $(k = 0, \ldots, N-1)$, i.e.,

$$\hat{x}_k = X(z) \bmod (z - w_N^k) \qquad (k = 0, \ldots, N-1). \tag{1.1}$$

Since

$$z^N - 1 = \prod_{k=0}^{N-1} (z - w_N^k),$$

we get a fast algorithm for the DFT(N) by the Chinese Remainder Theorem (CRT) [10, pp. 26–27], if we split $X(z) \bmod (z^N - 1)$ stepwise into equivalent simultaneous remainders, using the successive factorization of $z^N - 1$, such that we ultimately obtain the desired simultaneous remainders (1.1). We illustrate this by the radix-2 FFT of Cooley and Tukey (see [1]).

Let $N = 2^r$ $(r \in \mathbb{N})$. Then $z^N - 1$ can be decomposed successively as in Figure 1. This factorization is the foundation of the radix-2 FFT, which calculates the DFT(N) by the recursive reduction of the input polynomial $X(z)$ modulo the factors of $z^N - 1$ in Figure 1. The rth step of this reduction procedure yields the spectrum $\hat{\mathbf{x}} \in \mathbb{C}^N$.

Taking into account that most DFT's are taken on real data, many fast algorithms for real DFT's were published in recent years. These algorithms exploit directly the symmetries of the real DFT [14] or use transforms, which map a real-valued sequence to a real-valued spectrum as the discrete Hartley transform [13], the DCT, the cos-DFT and the sin-DFT [15]. Although the advantage of the polynomial arithmetic for the FFT is well known, there does not exist a convenient polynomial approach to the DCT, the cos-DFT and the sin-DFT up to now. This indeed is the task of our paper. Using Chebyshev polynomials, we define the DCT, the cos-DFT and the sin-DFT on a polynomial basis. We show that this representation leads to the descriptive derivation of fast algorithms for these transforms.

Section 2, where useful properties of Chebyshev polynomials are collected, has preliminary character. In §3, we suggest a new recursive algorithm for the DCT(2^r), which works with the same number of real operations as the best-known fast DCT's [7, 8, 15]. Introducing the cos-DFT and the sin-DFT, as well

as their reduced versions, we apply the polynomial arithmetic to decompose the reduced cos-DFT (reduced sin-DFT) of a length N divisible by 4 into a DCT($N/4$) and a reduced cos-DFT (reduced sin-DFT) of length $N/2$ in §5 (cf. [15]). Our fast algorithms can be used to compute the DFT(2^r) for real- and complex-valued sequences with the same computational complexity as the split-radix algorithm [3].

Although §§3 and 5 contain mainly the special case of fast algorithms for transforms of radix-2 length, the polynomial approach to fast algorithms for the DCT, the cos-DFT, and the sin-DFT of arbitrary highly factorizable lengths will be clear. We illustrate this idea by a fast DCT($3N$)-algorithm. Up to now, there do not exist fast algorithms for DCT's of such lengths. Note that, especially, the DCT has found wide applications in data compression and digital filtering.

2. Chebyshev polynomials

The polynomial approach to fast DCT's, cos-DFT's, and sin-DFT's is mainly based on known properties of Chebyshev polynomials, which we now summarize.

The *Chebyshev polynomials of first and second kind* can be defined recursively by

$$\begin{aligned} T_0(z) &:= 1, \qquad T_1(z) := z, \\ T_n(z) &:= 2zT_{n-1}(z) - T_{n-2}(z) \qquad (n = 2, 3, \dots), \\ T_n &= T_{-n} \qquad (n \in \mathbb{Z}), \end{aligned}$$

and by

$$\begin{aligned} U_0(z) &:= 1, \qquad U_1(z) := 2z, \\ U_n(z) &:= 2zU_{n-1}(z) - U_{n-2}(z) \qquad (n = 2, 3, \dots), \\ U_n &= -U_{-n-2} \qquad (n \in \mathbb{Z}), \end{aligned}$$

respectively [11, pp. 11–12]. From this it follows that

$$T_n(z) = \cos(n \arccos z) \qquad (|z| \le 1;\ n \in \mathbb{Z}), \tag{2.1}$$

$$U_n(z) = (1 - z^2)^{-1/2} \sin((n+1) \arccos z) \qquad (|z| < 1;\ n \in \mathbb{Z}), \tag{2.2}$$

and then

$$T_n(z) = 2^{n-1} \prod_{k=0}^{n-1} (z - \cos(\pi(2k+1)/2n)) \qquad (n \in \mathbb{N}), \tag{2.3}$$

$$U_n(z) = 2^n \prod_{k=1}^{n} (z - \cos(\pi k/(n+1))) \qquad (n \in \mathbb{N}). \tag{2.4}$$

We have [11, p. 24; 12, p. 5]

$$T_{mn} = T_m(T_n) = 2^{m-1} \prod_{k=0}^{m-1} (T_n - \cos(\pi(2k+1)/2m)) \qquad (m, n \in \mathbb{N}). \tag{2.5}$$

More generally, setting $y := n \arccos z \quad (|z| \le 1)$ in

$$\cos my - \cos\alpha = 2^{m-1} \prod_{k=0}^{m-1} (\cos y - \cos((\alpha + 2\pi k)/m)) \qquad (m \in \mathbb{N};\ \alpha \in \mathbb{R})$$

[5, p. 48], we obtain that

$$T_{mn} - \cos\alpha = 2^{m-1} \prod_{k=0}^{m-1} (T_n - \cos((\alpha + 2\pi k)/m)) \qquad (m, n \in \mathbb{N};\ \alpha \in \mathbb{R}). \tag{2.6}$$

Differentiation of (2.5) yields

$$\begin{aligned} U_{mn-1} &= U_{m-1}(T_n) U_{n-1} \\ &= 2^{m-1} \prod_{k=1}^{m-1} (T_n - \cos(\pi k/m)) U_{n-1} \qquad (m, n \in \mathbb{N}). \end{aligned} \tag{2.7}$$

Furthermore, we shall use the properties [11, p. 24]

$$T_m T_n = (T_{n-m} + T_{n+m})/2, \tag{2.8}$$

$$T_m(z) T_n(z) + (1 - z^2) U_{m-1}(z) U_{n-1}(z) = T_{n-m}(z), \tag{2.9}$$

$$U_{n-1} T_m + T_n U_{m-1} = U_{n+m-1}. \tag{2.10}$$

As in [4], we define the polynomials $W_n \quad (n \in \mathbb{N})$ by

$$\begin{aligned} W_1(z) &:= z - 2, \qquad W_2(z) := z + 2, \\ W_n(z) &:= \prod_{\substack{k=1 \\ (k,n)=1}}^{\lfloor n/2 \rfloor} (z - (w_n^k + w_n^{-k})) \\ &= \prod_{\substack{k=1 \\ (k,n)=1}}^{\lfloor n/2 \rfloor} (z - 2\cos(2\pi k/n)) \qquad (n = 3, 4, \ldots), \end{aligned}$$

where $\lfloor n/2 \rfloor := \max\{k \in \mathbb{Z} : k \le n/2\}$ and where (k, n) signifies the greatest common divisor of k and n. For further properties of W_n, especially the connection of W_n with Chebyshev polynomials, see [4]. Finally, let

$$V_{m+1}(z) := \prod_{d|n} W_d(2z) = 2^{m+1} \prod_{k=0}^{m} (z - \cos(2\pi k/n)) \tag{2.11}$$

with $m = \lfloor n/2 \rfloor$. If $n \in \mathbb{N}$ is even, then we have by (2.4) and (2.11) that

$$V_{m+1}(z) = 4(z^2 - 1) U_{m-1}(z). \tag{2.12}$$

3. Discrete cosine transform

The *discrete cosine transform of length* N (DCT(N)) is defined by the following mapping between $\mathbf{x} = (x_0, \dots, x_{N-1}) \in \mathbb{R}^N$ and $\tilde{\mathbf{x}} = (\tilde{x}_0, \dots, \tilde{x}_{N-1}) \in \mathbb{R}^N$:

$$\tilde{x}_k := \sum_{j=0}^{N-1} x_j \cos(\pi(2k+1)j/2N) \qquad (k = 0, \dots, N-1). \tag{3.1}$$

Note that our version of the DCT(N) is similar to the inverse DCT in [7, 9, 15].

In order to introduce a polynomial notation for the DCT, we represent the N-point sequence $\mathbf{x} \in \mathbb{R}^N$ as the polynomial

$$X := \sum_{j=0}^{N-1} x_j T_j. \tag{3.2}$$

Then, by (2.1), we have that (3.1) can be replaced by $\tilde{x}_k = X(\cos(\pi(2k+1)/2N))$ $(k = 0, \dots, N-1)$, i.e.,

$$\tilde{x}_k = X(z) \bmod (z - \cos(\pi(2k+1)/2N)) \qquad (k = 0, \dots, N-1). \tag{3.3}$$

By (2.3) and by the CRT, we obtain a fast decimation in frequency algorithm for the DCT, if we split $X \bmod T_N$ stepwise into the desired simultaneous remainders (3.3) by using polynomial factorizations of T_N.

In the following, let $N = 2^r$ $(r \in \mathbb{N})$. In this case, we get a successive factorization of T_N by applying the following

Lemma. *Let* $s \in \mathbb{N}$ $(s > 1)$, *and let* $a \in \mathbb{N}$ *be an odd integer with the bit representation*

$$a = (a_{s-1}, \dots, a_1, 1)_2 := 2^{s-1}a_{s-1} + \cdots + 2a_1 + 1 \qquad (a_i \in \{0, 1\}).$$

By $\oplus$, *we denote the addition modulo* 2. *Then, for any* $n \in \mathbb{N}$, *there holds*

$$T_{2n} - \cos(\pi a/2^s) = 2(T_n + \cos(\pi a/2^{s+1}))(T_n - \cos(\pi a/2^{s+1})), \tag{3.4}$$

$$T_{2n} - \cos(\pi a/2^s) = 2\prod_{j=0}^{1}(T_n - \cos(\pi(j, j \oplus a_{s-1}, \dots, j \oplus a_1, 1)_2/2^{s+1})).$$

Proof. Setting $m := 2$ and $\alpha := \pi a/2^s$ in (2.6), we get (3.4). The rest of the assertion follows from

$$\begin{aligned}
\cos(\pi a/2^{s+1}) &= -\cos(\pi(2^{s+1} - a)/2^{s+1}) \\
&= -\cos(\pi(2^s + 2^{s-1}(1 - a_{s-1}) + \cdots + 2(1 - a_1) + 1)/2^{s+1}) \\
&= -\cos(\pi(1, 1 - a_{s-1}, \dots, 1 - a_1, 1)_2/2^{s+1}) \\
&= -\cos(\pi(1, 1 \oplus a_{s-1}, \dots, 1 \oplus a_1, 1)_2/2^{s+1}). \quad \square
\end{aligned}$$

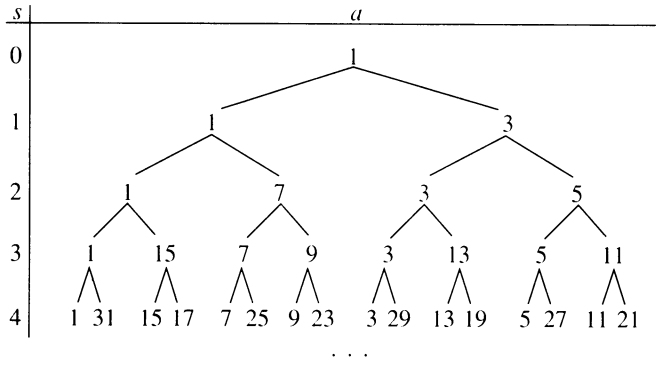

FIGURE 2
Factorization tree of T_N *with* $N = 2^r$ $(r \geq 4)$, *where* $a \in N$ *at the* s*th step signifies* $T_{N/2^s} - \cos(\pi a / 2^{s+1})$

The above lemma yields the following recursive factorization of $T_N = T_N - \cos(\pi/2)$:

1. $T_N = 2T^{(0)}T^{(1)}$ with

$$T^{(j_1)} := T_{N/2} - (-1)^{j_1}\sqrt{2}/2 \qquad (j_1 = 0, 1).$$

2. $T^{(j_1)} = 2T^{(j_1,0)}T^{(j_1,1)}$ for $j_1 = 0, 1$ with

$$T^{(j_1,j_2)} := T_{N/4} - (-1)^{j_2}\cos(\pi(j_1, 1)_2/8) \qquad (j_2 = 0, 1).$$

...

r. $T^{(j_1,\dots,j_{r-1})} = 2T^{(j_1,\dots,j_{r-1},0)}T^{(j_1,\dots,j_{r-1},1)}$ for $j_1, \dots, j_{r-1} = 0, 1$ with

$$\begin{aligned} T^{(j_1,\dots,j_r)} &:= T_1 - (-1)^{j_r}\cos(\pi(j_{r-1}, j_{r-1} \oplus j_{r-2}, \dots, j_{r-1} \oplus \cdots \oplus j_1, 1)_2/2N) \\ &= T_1 - \cos(\pi(j_r, j_r \oplus j_{r-1}, \dots, j_r \oplus \cdots \oplus j_1, 1)_2/2N) \qquad (j_r = 0, 1). \end{aligned}$$

The rth step contains the linear factors $T^{(j_1,\dots,j_r)}$ of T_N. Figure 2 describes the decomposition of T_N by the so-called transform factors $\cos(\pi(j_s, j_s \oplus j_{s-1}, \dots, j_s \oplus \cdots \oplus j_1, 1)_2/2^{s+1})$.

For a given input sequence $\mathbf{x} \in \mathbb{R}^N$, we consider the polynomial X introduced in (3.2). By the CRT, we have for every $s = 1, \dots, r$ that $X \bmod T_N$ is uniquely determined by its residues $X \bmod T^{(j_1,\dots,j_s)}$ $(j_1, \dots, j_s = 0, 1)$. This leads to the following recursive DCT(N)-algorithm:

1. Calculate $X \bmod T^{(j_1)}$ for $j_1 = 0, 1$. Observing that by (2.8)

$$T_{N/2+j} = 2T_{N/2}T_j - T_{N/2-j}, \tag{3.5}$$

we obtain

$$X^{(j_1)} := \sum_{j=0}^{N/2-1} x_j^{(j_1)} T_j = X \bmod T^{(j_1)}$$

with

$$x_j^{(j_1)} := \begin{cases} x_0 + (-1)^{j_1} x_{N/2}\sqrt{2}/2 & \text{for } j = 0, \\ x_j - x_{N-j} + (-1)^{j_1} x_{N/2+j}\sqrt{2} & \text{for } j = 1, \dots, N/2 - 1. \end{cases}$$

2. Calculate $X^{(j_1)} \bmod T^{(j_1, j_2)}$ for $j_1, j_2 = 0, 1$. Using (3.5), with $N/4$ instead of $N/2$, we get

$$X^{(j_1, j_2)} := \sum_{j=0}^{N/2-1} x_j^{(j_1, j_2)} T_j = X^{(j_1)} \bmod T^{(j_1, j_2)}$$

with

$$x_j^{(j_1, j_2)} := \begin{cases} x_0^{(j_1)} + (-1)^{j_2} x_{N/4}^{(j_1)} \cos(\pi(j_1, 1)_2/8) & \text{for } j = 0, \\ x_j^{(j_1)} - x_{N/2-j}^{(j_1)} + (-1)^{j_2} x_{N/4+j}^{(j_1)} 2\cos(\pi(j_1, 1)_2/8) & \\ \qquad \text{for } j = 1, \dots, N/4 - 1. & \end{cases}$$

. . .

r. Calculate $X^{(j_1, \dots, j_{r-1})} \bmod T^{(j_1, \dots, j_r)}$ for $j_1, \dots, j_r = 0, 1$. This yields the final result

$$X^{(j_1, \dots, j_r)} := x_0^{(j_1, \dots, j_r)} = X^{(j_1, \dots, j_{r-1})} \bmod T^{(j_1, \dots, j_r)}$$

with

$$x_0^{(j_1, \dots, j_r)} := x_0^{(j_1, \dots, j_{r-1})} + (-1)^{j_r} x_1^{(j_1, \dots, j_{r-1})} \cdot \cos(\pi(j_{r-1}, j_{r-1} \oplus j_{r-2}, \dots, j_{r-1} \oplus \cdots \oplus j_1, 1)_2/2N).$$

By (3.3) and by the decomposition of T_N, we see that $x_0^{(j_1, \dots, j_r)} = \tilde{x}_k$ for the index k with

$$k = (j_r, j_r \oplus j_{r-1}, \dots, j_r \oplus \cdots \oplus j_1)_2. \tag{3.6}$$

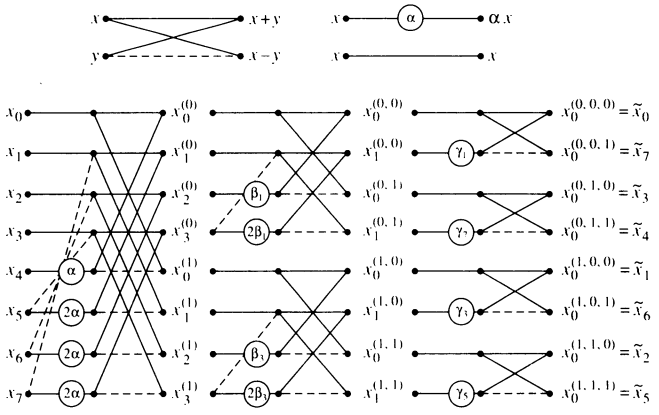

FIGURE 3

Flow graph for the DCT(8) *with* $\alpha := \sqrt{2}/2$, $\beta_j := \cos(\pi j/8)$, *and* $\gamma_j := \cos(\pi j/16)$

This describes the permutation of the output values. The $(j_1, \dots, j_r)_2$th component of our output sequence is $\tilde{x}_k$ with k as in (3.6). Figure 3 shows the flow graph of the DCT(8).

Simple considerations yield the computational complexity of our DCT-algorithm. The sth step of the algorithm requires $2^{s-1}(N/2^s)$ real multiplications and $2^{s-1}(3N/2^s - 1)$ real additions. Thus, our DCT(N)-algorithm with $N = 2^r$ $(r \in \mathbb{N})$ works with a total of

$$\widetilde{M}_N = \sum_{s=1}^{r} 2^{s-1}(N/2^s) = (N/2)r \tag{3.7}$$

real multiplications and

$$\widetilde{A}_N = \sum_{s=1}^{r} 2^{s-1}(3N/2^s - 1) = (N/2)(3r - 2) + 1 \tag{3.8}$$

real additions. Hence, our polynomial algorithm has the same computational complexity as the best-known DCT-algorithms [7, 8, 15].

4. Cos-DFT and sin-DFT

The image $\hat{\mathbf{x}} \in \mathbb{C}^N$ of the DFT(N) of a real-valued sequence $\mathbf{x} \in \mathbb{R}^N$ can be obtained by

$$\hat{x}_{c,k} := \sum_{j=0}^{N-1} x_j \cos(2\pi kj/N) \qquad (k = 0, \dots, N-1), \tag{4.1}$$

$$\hat{x}_{s,k} := \sum_{j=0}^{N-1} x_j \sin(2\pi kj/N) \qquad (k = 0, \dots, N-1), \tag{4.2}$$

and by

$$\hat{x}_k = \hat{x}_{c,k} - i\hat{x}_{s,k} \qquad (k = 0, \dots, N-1).$$

The mappings defined by (4.1) and (4.2) are called the *discrete Fourier-cosine transform of length* N (cos-DFT(N)) and the *discrete Fourier-sine transform of length* N (sin-DFT(N)), respectively. In this section, we present a polynomial approach to the cos-DFT and the sin-DFT, which suggests fast algorithms for both transforms.

Let $M := \lfloor N/2 \rfloor$. By

$$\cos(2\pi k(N-j)/N) = \cos(2\pi kj/N) \qquad (j, k \in \mathbb{Z}),$$

(4.1) can be rewritten as

$$\hat{c}_k := \hat{x}_{c,k} = \hat{x}_{c,N-k} = \sum_{j=0}^{M} c_j \cos(2\pi kj/N) \quad (k = 0, \dots, M) \tag{4.3}$$

with

$$c_j := \begin{cases} x_j & \text{for } j = 0 \text{ and } j = M \text{ if } 2|N, \\ x_j + x_{N-j} & \text{otherwise} \end{cases} \qquad (j = 0, \dots, M).$$

We call (4.3) the *reduced* cos-DFT(N). Similarly, by

$$\sin(2\pi k(N-j)/N) = -\sin(2\pi kj/N) \qquad (j, k \in \mathbb{Z}),$$

it follows from (4.2) that

$$\hat{s}_k := \hat{x}_{s,k} = -\hat{x}_{s,N-k} = \sum_{j=1}^{M} s_j \sin(2\pi kj/N) \qquad (k = 1, \ldots, M) \tag{4.4}$$

with

$$s_j :c= \begin{cases} 0 & \text{for } j = M \text{ if } 2|N, \\ x_j - x_{N-j} & \text{otherwise} \end{cases} \qquad (j = 1, \ldots, M).$$

Then (4.4) is said to be the *reduced* sin-DFT(N). Hence we can calculate the cos-DFT(N) (sin-DFT(N)) by $\lceil N/2 \rceil - 1$ additions and by the reduced cos-DFT(N) (sin-DFT(N)). Here, $\lceil N/2 \rceil := \min\{k \in \mathbb{Z} : k \geq N/2\}$. In the following, we deal with these reduced transforms.

In order to introduce a polynomial notation of the reduced cos-DFT, we represent the input sequence $\mathbf{c} = (c_0, \ldots, c_M) \in \mathbb{R}^{M+1}$ as the polynomial

$$C := \sum_{j=0}^{M} c_j T_j.$$

Then we have by (2.1) that $\hat{c}_k = C(\cos(2\pi k/N))$ $(k = 0, \ldots, M)$, i.e.,

$$\hat{c}_k = C(z) \bmod (z - \cos(2\pi k/N)) \qquad (k = 0, \ldots, M). \tag{4.5}$$

By (2.11), we obtain a fast algorithm for the reduced cos-DFT(N) if we split $C \bmod V_{M+1}$ stepwise into equivalent simultaneous remainders by using successive factorization of V_{M+1} together with the CRT, such that we get (4.5) in the last step.

Analogously, we represent the input sequence $\mathbf{s} = (s_1, \ldots, s_M) \in \mathbb{R}^M$ of the reduced sin-DFT as the polynomial

$$S := \sum_{j=1}^{M} s_j U_{j-1}. \tag{4.6}$$

Then we see by (2.2) that (4.4) can be expressed as

$$\hat{s}_k = \sin(2\pi k/N) S(\cos(2\pi k/N)) \qquad (k = 1, \ldots, M),$$

i.e.,

$$\hat{s}_k = \sin(2\pi k/N) S(z) \bmod (z - \cos(2\pi k/N)) \qquad (k = 1, \ldots, M). \tag{4.7}$$

It follows from (2.11) and from the CRT that we can deduce a fast sin-DFT(N) if we reduce $S(z) \bmod (V_{M+1}(z)/2(z-1))$ successively into equivalent simultaneous residues by applying polynomial factorizations of $V_{M+1}(z)/2(z-1)$, such that we ultimately obtain (4.7).

For even $N \in \mathbb{N}$, (4.6) and (4.7) can be simplified to

$$S = \sum_{j=1}^{M-1} s_j U_{j-1}, \tag{4.8}$$

$$(4.9)\quad \hat{s}_k = \sin(2\pi k/N)S(z) \bmod (z - \cos(2\pi k/N)) \qquad (k = 1, \dots, M-1),$$

since $\hat{s}_M = 0$. By (2.4), we get a fast algorithm for the reduced sin-DFT(N) with even $N \in \mathbb{N}$ by splitting $S \bmod U_{M-1}$ stepwise, such that we obtain (4.9) in the end.

5. Fast algorithms for reduced cos-DFT and sin-DFT

In this section, we assume $N \in \mathbb{N}$ divisible by 4. Set $M := N/2$. Based on the factorization

$$(5.1)\qquad U_{M-1} = U_1(T_{M/2})U_{M/2-1} = 2T_{M/2}U_{M/2-1},$$

which follows immediately from (2.7), we show that the reduced cos-DFT can be decomposed into the reduced cos-DFT($N/2$) and the DCT($N/4$). The reduced sin-DFT can be handled analogously. This verifies a result in [15] from the polynomial point of view.

By (2.12), (5.1), and by the CRT, $C \bmod V_{M+1}$ is completely determined by its residues $C \bmod T_{M/2}$ and $C \bmod V_{M/2+1}$. First we evaluate $C \bmod T_{M/2}$ by polynomial reductions. By $T_j = -T_{M-j} \bmod T_{M/2}$ $(j = 0, \dots, M/2-1)$, we verify that

$$(5.2)\qquad C^{(1)} := \sum_{j=0}^{M/2-1} c_j^{(1)} T_j = C \bmod T_{M/2}$$

with $c_j^{(1)} := c_j - c_{M-j}$ $(j = 0, \dots, M/2-1)$. Since by (2.12), (5.1) and (2.3),

$$\begin{aligned}\hat{c}_{2k+1} &= ((C \bmod V_{M+1}) \bmod T_{M/2}) \bmod (T_1 - \cos(2\pi(2k+1)/N)) \\ &= C^{(1)} \bmod (T_1 - \cos(\pi(2k+1)/M)) \qquad (k = 0, \dots, M/2-1),\end{aligned}$$

the output values with odd indices of the reduced cos-DFT(N) can be calculated by $M/2$ additions and by the DCT($M/2$) given by (5.2).

On the other hand, by (2.9), we have $T_j = T_{M-j} \bmod V_{M/2+1}$ $(j = 0, \dots, M/2-1)$, so that $C \bmod V_{M/2+1}$ is obtained by

$$(5.3)\qquad C^{(2)} := \sum_{j=0}^{M/2} c_j^{(2)} T_j = C \bmod V_{M/2+1}$$

with $c_j^{(2)} := c_j + c_{M-j}$ $(j = 0, \dots, M/2-1)$, $c_{M/2}^{(2)} := c_{M/2}$. Then we have by (2.12), (5.1), and (2.4) that

$$\begin{aligned}\hat{c}_{2k} &= ((C \bmod V_{M+1}) \bmod V_{M/2+1}) \bmod (T_1 - \cos(2\pi(2k)/N)) \\ &= C^{(2)} \bmod (T_1 - \cos(2\pi k/M)) \qquad (k = 0, \dots, M/2).\end{aligned}$$

Hence, we get the output values with even indices of the reduced cos-DFT(N) by $M/2$ additions and by the reduced cos-DFT(M) determined in (5.3).

Turning to the reduced sin-DFT, we take a similar approach. By (5.1) and by the CRT, $S \bmod U_{M-1}$ is completely determined by its residues $S \bmod T_{M/2}$ and $S \bmod U_{M/2-1}$. Considering that, by (2.10), $U_{j-1} = U_{M-j-1} \bmod T_{M/2}$ $(j = 1, \ldots, M/2)$, we find that $S \bmod T_{M/2}$ is given by

$$S^{(1)} := \sum_{j=1}^{M/2} s_j^{(1)} U_{j-1} = S \bmod T_{M/2}$$

with $s_j^{(1)} := s_j + s_{M-j}$ $(j = 1, \ldots, M/2 - 1)$, $s_{M/2}^{(1)} := s_{M/2}$. Instead of $S^{(1)}$ we consider

$$\tilde{S}^{(1)} := \sum_{j=0}^{M/2-1} \tilde{s}_j^{(1)} T_j \tag{5.4}$$

with $\tilde{s}_j^{(1)} := s_{M/2-j}^{(1)}$ $(j = 0, \ldots, M/2 - 1)$. Then from

$$\begin{aligned}\sin(2\pi(2k+1)/N) U_{j-1}(\cos(\pi(2k+1)/M)) &= \sin(2\pi(2k+1)j/N) \\ &= (-1)^k \cos(2\pi(2k+1)(M/2-j)/N) = (-1)^k T_{M/2-j}(\cos(\pi(2k+1)/M))\end{aligned}$$

one verifies that

$$\begin{aligned}\hat{s}_{2k+1} &= \sin(2\pi(2k+1)/N)((S \bmod U_{M-1}) \bmod T_{M/2}) \\ &\qquad \bmod(T_1 - \cos(2\pi(2k+1)/N) \\ &= \sin(2\pi(2k+1)/N) S^{(1)} \bmod(T_1 - \cos(\pi(2k+1)/M)) \\ &= (-1)^k \tilde{S}^{(1)} \bmod(T_1 - \cos(\pi(2k+1)/M)) \qquad (k = 0, \ldots, M/2 - 1).\end{aligned}$$

Consequently, we obtain the output values with odd indices of the reduced sin-DFT(N) by $M/2 - 1$ additions and by the DCT$(M/2)$ given by (5.4), where we have to change the sign of the output values with indices congruent 3 modulo 4.

Next, by (2.10), we have $U_{j-1} = -U_{M-j-1} \bmod U_{M/2-1}$ $(j = 1, \ldots, M/2 - 1)$. Using this property, we form

$$S^{(2)} := \sum_{j=1}^{M/2-1} s_j^{(2)} U_{j-1} = S \bmod U_{M/2-1} \tag{5.5}$$

with $s_j^{(2)} := s_j - s_{M-j}$ $(j = 1, \ldots, M/2 - 1)$. Now we conclude from

$$\begin{aligned}\hat{s}_{2k} &= \sin(2\pi(2k)/N)((S \bmod U_{M-1}) \bmod U_{M/2-1}) \bmod(T_1 - \cos(2\pi(2k)/N)) \\ &= \sin(2\pi k/M) S^{(2)} \bmod(T_1 - \cos(2\pi k/M)) \qquad (k = 1, \ldots, M/2 - 1),\end{aligned}$$

that the output values with even indices of the reduced sin-DFT(N) can be evaluated by $M/2 - 1$ additions and by the reduced sin-DFT(M) determined in (5.5).

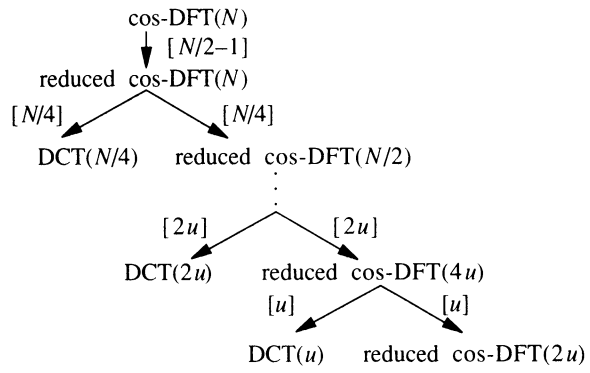

FIGURE 4
Recursive computation of the cos-DFT(N) *with* $N = 2^r u$ $(r \geq 2;\ u$ *odd*$)$ ($[\cdot]$ *signifies the number of additions per step*)

Let $N = 2^r$ $(r \geq 2)$. Then we can use the above reduction successively for the reduced cos-DFT(2^s) (reduced sin-DFT(2^s)) with $s = r, \ldots, 2$. This results in the computation of the cos-DFT(N) (sin-DFT(N)) using only DCT's and some additions. See Figure 4 with $u := 1$.

The numbers M_N^c and M_N^s of real multiplications and the numbers A_N^c and A_N^s of real additions to perform the cos-DFT(N) and the sin-DFT(N), respectively, follow directly from Figure 4, (3.7) and (3.8). For $N = 2^r$ $(r \geq 2)$, one obtains

$$M_N^c = M_N^s = \sum_{s=1}^{r-2} \widetilde{M}_{2^s} = (N/4)(r-3) + 1,$$

$$A_N^c = N/2 - 1 + 2\sum_{s=0}^{r-2} 2^s + \sum_{s=1}^{r-2} \widetilde{A}_{2^s} + 2 = (N/4)(3r-5) + r + 2,$$

$$A_N^s = N/2 - 1 + 2\sum_{s=0}^{r-2} (2^s - 1) + \sum_{s=1}^{r-2} \widetilde{A}_{2^s} = (N/4)(3r-5) - r + 2.$$

Consequently, the DFT(N) of a real-valued sequence computed by our method requires

$$M_N^r = M_N^c + M_N^s = (N/2)(r-3) + 2,$$
$$A_N^r = A_N^c + A_N^s = (N/2)(3r-5) + 4$$

real multiplications and real additions, respectively. Further, by

$$\hat{x}_k = \sum_{j=0}^{N-1} \mathrm{Re}(x_j)\cos(2\pi kj/N) - \sum_{\substack{j=1 \\ j\neq N/2}}^{N-1} \mathrm{Im}(x_j)\sin(2\pi kj/N)$$
$$+ i\left(\sum_{j=0}^{N-1} \mathrm{Im}(x_j)\cos(2\pi kj/N) + \sum_{\substack{j=1 \\ j\neq N/2}}^{N-1} \mathrm{Re}(x_j)\sin(2\pi kj/N)\right)$$
$$(k = 0, \ldots, N-1),$$

the number of real operations of the $\mathrm{DFT}(N)$ of a complex-valued sequence is given by

$$M_N = 2M_N^r = N(r-3)+4, \qquad A_N = 2A_N^r + 2(N-2) = 3N(r-1)+4.$$

Compared with other DFT-algorithms, we conclude that our polynomial algorithm works with the same computational complexity as the algorithm in [15] and the split-radix algorithm [3, 14].

6. Fast algorithm for the $\mathrm{DCT}(3N)$

The polynomial representations of the $\mathrm{DCT}(N)$, the cos-$\mathrm{DFT}(N)$, and the sin-$\mathrm{DFT}(N)$ in §§3 and 4 open new possibilities for the derivation of fast algorithms for these transforms for various lengths N by applying the CRT in combination with the factorizations (2.5), (2.6), and (2.7) of Chebyshev polynomials, or in combination with the factorization (2.11). The reductions of polynomials of the form (3.2) or (4.8) modulo Chebyshev polynomials in such algorithms can be performed only by (2.8), (2.9), and (2.10). In order to illustrate these general considerations, we suggest a new polynomial algorithm for the $\mathrm{DCT}(3N)$.

We consider the $\mathrm{DCT}(3N)$ $(N \in \mathbb{N})$, i.e., for given

$$X := \sum_{j=0}^{3N-1} x_j T_j \mod T_{3N},$$

we have to evaluate

$$\tilde{x}_k = X(z) \bmod (z - \cos(\pi(2k+1)/6N)) \qquad (k = 0, \ldots, 3N-1).$$

By (2.5), T_{3N} factors as

$$T_{3N} = 4(T_N - \sqrt{3}/2)T_N(T_N + \sqrt{3}/2),$$

so that $X \bmod T_{3N}$ is completely determined by the residues $X \bmod T_N$ and $X \bmod (T_N \pm \sqrt{3}/2)$. Considering that by (2.5) and (2.8)

$$T_{2N} = 2T_N^2 - 1,$$
$$T_{N+j} = 2T_N T_j - T_{N-j},$$
$$T_{2N+j} = (4T_N^2 - 1)T_j - 2T_N T_{N-j} \qquad (j = 1, \ldots, N-1),$$

we obtain the following recursive algorithm.

First, we calculate $X \bmod T_N$ by

$$X^{(0)} := \sum_{j=0}^{N-1} x_j^{(0)} T_j = X \bmod T_N \tag{6.1}$$

with

$$x_j^{(0)} := \begin{cases} x_0 - x_{2N} & \text{for } j = 0, \\ x_j - x_{2N-j} - x_{2N+j} & \text{for } j = 1, \dots, N-1. \end{cases}$$

Since by (2.3), T_N has the roots $\cos(\pi(2k+1)/2N) = c\ s(\pi(2(3k+1)+1)/6N)$ $(k = 0, \dots, N-1)$, we obtain

$$\tilde{x}_{3k+1} = X^{(0)} \bmod (T_1 - \cos(\pi(2k+1)/2N)) \qquad (k = 0, \dots, N-1).$$

Next, we form $X \bmod (T_N \pm \sqrt{3}/2)$ as follows:

$$X^{(1)} := \sum_{j=0}^{N-1} x_j^{(1)} T_j = X \bmod (T_N - \sqrt{3}/2) \tag{6.2}$$

with

$$x_j^{(1)} := \begin{cases} x_0 + x_{2N}/2 + x_N\sqrt{3}/2 & \text{for } j = 0, \\ x_j - x_{2N-j} + 2x_{2N+j} + \sqrt{3}(x_{N+j} - x_{3N-j}) & \text{for } j = 1, \dots, N-1, \end{cases}$$

and

$$X^{(2)} := \sum_{j=0}^{N-1} x_j^{(2)} T_j = X \bmod (T_N + \sqrt{3}/2) \tag{6.3}$$

with

$$x_j^{(2)} := \begin{cases} x_0 + x_{2N}/2 - x_N\sqrt{3}/2, & \text{for } j = 0, \\ x_j - x_{2N-j} + 2x_{2N+j} - \sqrt{3}(x_{N+j} - x_{3N-j}) & \text{for } j = 1, \dots, N-1. \end{cases}$$

By (2.6), the zeros of $T_N - \sqrt{3}/2$ and of $T_N + \sqrt{3}/2$ are given by

$$\begin{aligned} &\{\cos((\pi/6 + 2\pi k)/N) : k = 0, \dots, N-1\} \\ &\quad = \{\cos(\pi(2(6k)+1)/6N), \cos(\pi(2(6k'+5)+1)/6N) : \\ &\qquad\qquad k = 0, \dots, M;\ k' = 0, \dots, M'\} \end{aligned}$$

and by

$$\begin{aligned} &\{\cos(5\pi/6 + 2\pi k/N) : k = 0, \dots, N-1\} \\ &\quad = \{\cos(\pi(2(6k+2)+1)/6N), \cos(\pi(2(6k'+3)+1)/6N) : \\ &\qquad\qquad k = 0, \dots, M;\ k' = 0, \dots, M'\}, \end{aligned}$$

respectively, where $M := \lceil N/2 \rceil - 1$ and $M' := \lfloor N/2 \rfloor - 1$. Hence, we get by (6.2) and (6.3) that

$$\begin{aligned}
\tilde{x}_{6k} &= X^{(1)} \bmod(T_1 - \cos(\pi(12k+1)/6N)) \qquad (k = 0, \ldots, M),\\
\tilde{x}_{6k+5} &= X^{(1)} \bmod(T_1 - \cos(\pi(12k+11)/6N)) \qquad (k = 0, \ldots, M'),\\
\tilde{x}_{6k+2} &= X^{(2)} \bmod(T_1 - \cos(\pi(12k+5)/6N)) \qquad (k = 0, \ldots, M),\\
\tilde{x}_{6k+3} &= X^{(2)} \bmod(T_1 - \cos(\pi(12k+7)/6N)) \qquad (k = 0, \ldots, M').
\end{aligned}$$

As a result, we have decomposed the DCT($3N$) into the DCT(N) given by (6.1) and into the modified DCT(N)'s determined by (6.2) and (6.3), with a total of $2N$ multiplications and $6N - 2$ additions. Obviously, using (2.6) instead of (2.5), these modified DCT's can be handled similarly as the usual DCT. We have only to change the transform factors in the multiplications.

We now combine this idea with the developments of the previous sections. Let $N = 2^r$ $(r \geq 2)$. Then by (3.7) and (3.8), we can perform the DCT($3N$) with

$$\widetilde{M}_{3N} = 3\widetilde{M}_N + 2N = (N/2)(3r+4), \qquad \widetilde{A}_{3N} = 3\widetilde{A}_n + 6N - 2 = (N/2)(9r+6) + 1$$

real operations. Using the decompositions in §5 (see Figure 4), we obtain the following computational complexity for the cos-DFT($3N$) and for the sin-DFT($3N$):

$$M^c_{3N} = M^s_{3N} = \sum_{s=0}^{r-2} \widetilde{M}_{3\cdot 2^s} + 2 = (N/4)(3r-5) + 3,$$

$$A^c_{3N} = 3N/2 - 1 + 2\sum_{s=0}^{r-2} 3\cdot 2^s + \sum_{s=0}^{r-2} \widetilde{A}_{3\cdot 2^s} + 8 = (N/4)(9r-3) + r + 6,$$

$$A^s_{3N} = 3N/2 - 1 + 2\sum_{s=0}^{r-2} (3\cdot 2^s - 1) + \sum_{s=0}^{r-2} \widetilde{A}_{3\cdot 2^s} + 2 = (N/4)(9r-3) - r + 2.$$

Finally, we see that the DFT($3N$) requires

$$M^r_{3N} = (N/2)(3r-5) + 6, \qquad M_{3N} = N(3r-5) + 12$$

real multiplications and

$$A^r_{3N} = (N/2)(9r-3) + 8, \qquad A_{3N} = N(9r+3) + 12$$

real additions. This coincides with the number of real operations for the computation of the DFT($3\cdot 2^r$) $(r \geq 2)$ by combining the prime factor algorithm, the split-radix algorithm and the Rader algorithm [1, 14].

BIBLIOGRAPHY

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullmann, *The design and analysis of computer algorithms*, Addison-Wesley, Reading, Mass., 1976.
2. G. Bruun, *z-transform* DFT *filters and* FFTs, IEEE Trans. Acoust. Speech Signal Process. **26** (1978), 56–63.

3. P. Duhamel, *Implementation of "split-radix" FFT algorithms for complex, real and real-symmetric data*, IEEE Trans. Acoust. Speech Signal Process. **34** (1986), 285–295.
4. D. Garbe, *On the level of irreducible polynomials over Galois fields*, J. Korean Math. Soc. **22** (1985), 117–124.
5. I. S. Gradshtein and I. M. Ryzhik, *Tables of integrals, sums, series and products*, Nauka, Moscow, 1971. (Russian)
6. M. T. Heideman and C. S. Burrus, *On the number of multiplications necessary to compute a length-2^n DFT*, IEEE Trans. Acoust. Speech Signal Process. **34** (1986), 91–95.
7. H. S. Hou, *A fast recursive algorithm for computing the discrete cosine transform*, IEEE Trans. Acoust. Speech Signal Process. **35** (1987), 1455–1462.
8. B. G. Lee, *A new algorithm to compute the discrete cosine transform*, IEEE Trans. Acoust. Speech Signal Process. **32** (1984), 1243–1245.
9. M. J. Narasimha and A. M. Peterson, *On computing the discrete cosine transform*, IEEE Trans. Comm. **26** (1978), 934–936.
10. H. J. Nussbaumer, *Fast Fourier transform and convolution algorithms*, Springer, Berlin-Heidelberg-New York, 1981.
11. S. Paszkowski, *Numerical application of Chebyshev polynomials and series*, Nauka, Moscow, 1983. (Russian)
12. T. J. Rivlin, *The Chebyshev polynomials*, Wiley, New York-London-Sydney-Toronto, 1974.
13. H. V. Sorensen, D. L. Jones, C. S. Burrus, and M. T. Heideman, *On computing the discrete Hartley transform*, IEEE Trans. Acoust. Speech Signal Process. **33** (1985), 1231–1238.
14. H. V. Sorensen, D. J. Jones, M. T. Heideman, and C. S. Burrus, *Real-valued fast Fourier transform algorithms*, IEEE Trans. Acoust. Speech Signal Process. **35** (1987), 849–863.
15. M. Vetterli and H. J. Nussbaumer, *Simple FFT and DCT algorithms with reduced number of operations*, Signal Process. **6** (1984), 267–278.
16. S. Winograd, *On computing the discrete Fourier transform*, Math. Comp. **32** (1978), 175–199.
17. ———, *Arithmetic complexity of computations*, CBMS Regional Conf. Ser. in Math., vol. 33, SIAM, Philadelphia, 1980.

FACHBEREICH MATHEMATIK, UNIVERSITÄT ROSTOCK, UNIVERSITÄTSPLATZ 1, 0-2500 ROSTOCK, GERMANY