A Computationally Efficient DFT Scheme for Applications With a Subset of Nonzero Inputs

Wei-Chieh Huang, Chih-Peng Li, *Member, IEEE*, and Hsueh-Jyh Li, *Senior Member, IEEE*

***Abstract*—Fourier transformation is a powerful analytical tool with wide-ranging applications in many fields. In certain cases, some of the inputs to the transformation function are zero, while the others are real or complex. For the case, where the nonzero inputs are complex, the transform decomposition (TD) method enables a significant reduction in the computational complexity. This letter proposes a modified TD (MTD) algorithm to further reduce the complexity when the nonzero input data are consecutive and real-valued. The analytical and numerical results confirm that the complexity of the MTD scheme is not only significantly lower than that of the original TD method, but also lower than that of the traditional split-radix fast Fourier transform (FFT) method when the length of the input sequence is short.**

***Index Terms*—Discrete Fourier transform (DFT), fast Fourier transform (FFT), transform decomposition (TD).**

I. Introduction

FOURIER transformation is widely applied in many different fields, ranging from communications and optics on the one hand to medical science and X-ray crystallography on the other. Advances in digital signal processing capabilities have prompted the development of powerful discrete Fourier transform (DFT) and inverse discrete Fourier transform (IDFT) algorithms. However, even with today's sophisticated processors, there is an urgent need to reduce the number of computations and storage elements. Consequently, intense research effort has been expended in developing more efficient numerical procedures. For most applications, the DFT and IDFT processes can be regarded as mapping an N-point long sequence in one domain to a corresponding N-point long sequence in another domain. In some applications, e.g., computing the high-resolution spectrum and transition-region (between the passband and stopband) response when designing a filter, some of the elements of the input sequence are zero, while the remainder are real or complex.

Various researchers have presented DFT algorithms for such applications [1]–[3]. One of the well-known examples is that of the pruning algorithm [1], [2], based upon the conventional one-butterfly radix-2 fast Fourier transform (FFT) scheme [4]. However, the transform decomposition (TD) method presented in [3] yields a significant reduction in the computational complexity when the nonzero inputs are complex. This letter modifies the TD scheme to further reduce the number of computational operations required when the nonzero inputs are consecutive and real-valued.

Manuscript received August 12, 2007; revised September 16, 2007. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Xiang-Gen Xia.

W.-C. Huang and H.-J. Li are with the Graduate Institute of Communication Engineering, National Taiwan University, Taipei, 106 Taiwan, R.O.C. (e-mail: d95942017@ntu.edu.tw; hjli@ew.ee.ntu.edu.tw).

C.-P. Li is with the Institute of Communications Engineering, National Sun Yat-Sen University, Kaohsiung, 804 Taiwan, R.O.C. (e-mail: cpli@faculty.nsysu.edu.tw).

Digital Object Identifier 10.1109/LSP.2007.911767

II. Modified Transform Decomposition (MTD) Method for Real-Valued Inputs

Let the (m,n)th element of the $N \times N$ DFT matrix $\mathbf{F}_N$ be given by $F_N(m,n) = W_N^{mn}$, where $W_N = \exp(-j2\pi/N)$. In general, the DFT operation on an arbitrary complex sequence $\mathbf{X}$ can be expressed in matrix form as

$$\mathbf{Y} = \mathbf{F}_N \mathbf{X}. \tag{1}$$

Assuming that only the first L consecutive elements of $\mathbf{X}$ are nonzero, (1) can be rewritten as

$$\mathbf{Y} = \mathbf{F}_{N,L} \mathbf{S} \tag{2}$$

where $\mathbf{F}_{N,L}$ is an $N \times L$ matrix containing the first L columns of the DFT matrix $\mathbf{F}_N$, and the $L \times 1$ vector $\mathbf{S}$ is extracted from the first L nonzero elements of $\mathbf{X}$, i.e., $\mathbf{S} = [X[0], X[1], \ldots, X[L-1]]^T$.

To simplify the computation of (2), consider the case where N is divisible by L, i.e., the variable Q, where $Q \equiv N/L$, has an integer value. Furthermore, if N is not divisible by the data length L', then $L - L'$ zeros can be padded at the end of the data sequence. Under these assumptions, the matrix $\mathbf{F}_{N,L}$ of size $N \times L$ can be expressed as

$$\mathbf{F}_{N,L} = \begin{bmatrix} \mathbf{F}_{N,L}(0) \\ \mathbf{F}_{N,L}(1) \\ \vdots \\ \mathbf{F}_{N,L}(N-1) \end{bmatrix} = \mathbf{P} \begin{bmatrix} \mathbf{G}_L^{(\beta)} \\ \mathbf{G}_L^{(\beta+1)} \\ \vdots \\ \mathbf{G}_L^{(\beta+Q-1)} \end{bmatrix} \tag{3}$$

where $\mathbf{F}_{N,L}(q)$ denotes the qth row of $\mathbf{F}_{N,L}$ and $\mathbf{G}_L^{(\beta)}$ is an $L \times L$ matrix constructed by extracting L equally spaced rows of $\mathbf{F}_{N,L}$, i.e.,

$$\mathbf{G}_L^{(\beta)} = \begin{bmatrix} \mathbf{F}_{N,L}(\langle \beta \rangle_N) \\ \mathbf{F}_{N,L}(\langle \beta + Q \rangle_N) \\ \vdots \\ \mathbf{F}_{N,L}(\langle \beta + (L-1)Q \rangle_N) \end{bmatrix} \tag{4}$$

where $\langle \cdot \rangle_N$ denotes the modulo of N and β is an arbitrary initial index. In addition, $\mathbf{P}$ is an $N \times N$ permutation matrix such that

(3) is fulfilled. Since $W_N^{\langle q \rangle_N} = W_N^q$ and $W_N^Q = W_L$, the matrix $\mathbf{G}_L^{(\beta)}$ has the form

$$\begin{aligned}\mathbf{G}_L^{(\beta)} &= \begin{bmatrix} W_N^0 & W_N^{\beta} & \cdots & W_N^{\beta(L-1)} \\ W_N^0 & W_N^{(\beta+Q)} & \cdots & W_N^{(\beta+Q)(L-1)} \\ \vdots & \vdots & \ddots & \vdots \\ W_N^0 & W_N^{\beta+(L-1)Q} & \cdots & W_N^{\{\beta+(L-1)Q\}(L-1)} \end{bmatrix} \\ &= \begin{bmatrix} W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^Q & \cdots & W_N^{Q(L-1)} \\ \vdots & \vdots & \ddots & \vdots \\ W_N^0 & W_N^{Q(L-1)} & \cdots & W_N^{Q(L-1)(L-1)} \end{bmatrix} \\ &\quad \times \begin{bmatrix} W_N^0 & 0 & \cdots & 0 \\ 0 & W_N^{\beta} & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & W_N^{\beta(L-1)} \end{bmatrix} \\ &\equiv \mathbf{F}_L \boldsymbol{\Delta}_L^{(\beta)}. \end{aligned} \tag{5}$$

As a result, the matrix $\mathbf{F}_{N,L}$ can be decomposed into QL-point DFTs, $\mathbf{F}_N$, i.e.,

$$\begin{aligned}\mathbf{F}_{N,L} &= \begin{bmatrix} \mathbf{F}_{N,L}(0) \\ \mathbf{F}_{N,L}(1) \\ \vdots \\ \mathbf{F}_{N,L}(N-1) \end{bmatrix} = \mathbf{P} \begin{bmatrix} \mathbf{G}_L^{(\beta)} \\ \mathbf{G}_L^{(\beta+1)} \\ \vdots \\ \mathbf{G}_L^{(\beta+Q-1)} \end{bmatrix} \\ &= \mathbf{P} \begin{bmatrix} \mathbf{F}_L \boldsymbol{\Delta}_L^{(\beta)} \\ \mathbf{F}_L \boldsymbol{\Delta}_L^{(\beta+1)} \\ \vdots \\ \mathbf{F}_L \boldsymbol{\Delta}_L^{(\beta+Q-1)} \end{bmatrix}. \end{aligned} \tag{6}$$

Note that a similar result was obtained in [3] for the particular case of $\beta = 0$. Clearly, different values of the initial index β result in different sets of transforming matrices $\mathbf{G}_L^{(\beta+i)}$, $i = 0, 1, \ldots, Q-1$.

In the fast Fourier transform (FFT) scheme, the computational complexity for real-valued inputs is half that for complex inputs [3], [5]. By contrast, the complexity of the conventional TD method is essentially independent of the input data type. However, this letter presents a modified TD algorithm (designated as MTD), in which the computational complexity for real-valued inputs is significantly reduced by specifying an appropriate value of the initial index β. For an even value of Q and $1 \le \alpha \le Q/2 - 1$, it can be shown that

$$\begin{aligned}\mathbf{G}_L^{(\alpha)} &= \mathbf{F}_L \boldsymbol{\Delta}_L^{(\alpha)} = \mathbf{F}_L \begin{bmatrix} W_N^0 & 0 & \cdots & 0 \\ 0 & W_N^{\alpha} & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & W_N^{\alpha(L-1)} \end{bmatrix} \\ &= \mathbf{F}_L \left[\boldsymbol{\Delta}_L^{(-\alpha)} \right]^*. \end{aligned} \tag{7}$$

In other words, if the value of the initial index in matrix $\mathbf{G}_L^{(\beta+i)}$ in (6), $i = 0, 1, \ldots, Q-1$, is specified as $\beta = -Q/2$, and $1 \le t \le Q/2 - 1$, then

$$\mathbf{F}_L \boldsymbol{\Delta}_L^{(\beta+Q-t)} = \mathbf{F}_L \left(\boldsymbol{\Delta}_L^{(\beta+t)} \right)^*. \tag{8}$$

Hence, the corresponding transformed sequence can be written as

$$\begin{aligned}\mathbf{Y}^{(\beta+t)} &= \mathbf{G}_L^{(\beta+t)} \mathbf{S} = \mathbf{F}_L \boldsymbol{\Delta}_L^{(\beta+t)} \mathbf{S} \\ &= \mathbf{F}_L \left\{ \mathrm{Re}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) + j \cdot \mathrm{Im}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \right\} \mathbf{S} \\ &= \mathbf{F}_L \mathrm{Re}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \mathbf{S} + j \cdot \mathbf{F}_L \mathrm{Im}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \mathbf{S} \end{aligned} \tag{9}$$

and

$$\begin{aligned}\mathbf{Y}^{(\beta+Q-t)} &= \mathbf{G}_L^{(\beta+Q-t)} \mathbf{S} = \mathbf{F}_L \boldsymbol{\Delta}_L^{(\beta+Q-t)} \mathbf{S} = \mathbf{F}_L \left(\boldsymbol{\Delta}_L^{(\beta+t)} \right)^* \mathbf{S} \\ &= \mathbf{F}_L \left\{ \mathrm{Re}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) - j \cdot \mathrm{Im}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \right\} \mathbf{S} \\ &= \mathbf{F}_L \mathrm{Re}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \mathbf{S} - j \cdot \mathbf{F}_L \mathrm{Im}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \mathbf{S}. \end{aligned} \tag{10}$$

For real-valued inputs $\mathbf{S}$, it can be observed from (9) and (10) that both $\mathbf{Y}^{(\beta+t)}$ and $\mathbf{Y}^{(\beta+Q-t)}$ can be obtained by executing just two real-valued L-point DFT operations on $\mathrm{Re}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \mathbf{S}$ and $\mathrm{Im}\left(\boldsymbol{\Delta}_L^{(\beta+t)} \right) \mathbf{S}$. In addition, when $t = Q/2$, the matrix $\boldsymbol{\Delta}_L^{(\beta+t)}$ becomes an $L \times L$ identity matrix $\mathbf{I}_L$, and thus $\boldsymbol{\Delta}_L^{(\beta+t)} \mathbf{S}$ is also real-valued. Consequently, for $1 \le t \le Q-1$, a total of $2(Q/2 - 1) + 1 = Q - 1$ L-point real-valued DFT operations are required. Note that when $t = 0$, a complex DFT operation is required since $\boldsymbol{\Delta}_L^{(\beta)}$ is a complex diagonal matrix. Therefore, when the nonzero inputs are real-valued, the computational complexity of the proposed MTD algorithm reduces to $Q - 1$ real DFT operations and 1 complex DFT operation. Note that the same result is also obtained when the value of the initial index is specified as $\beta = -Q/2 + 1$. By comparison, a total of Q complex DFT operations are required when the nonzero values are complex.

The analyses presented above are valid for the case where Q is even. However, when Q is odd, similar results can still be obtained by setting $\beta = -(Q-1)/2$. For odd values of Q, it is easily shown that the MTD algorithm requires just $Q - 1$ real-valued DFT operations when the inputs are real-valued.

III. Complexity Analyses

This section compares the computational complexity of the proposed MTD scheme with that of the split-radix FFT method and the conventional TD method for the case where N is a power of 2. The split-radix FFT [3], [6] requires a total of

$$\#\mathrm{M}_{\mathrm{FFT}}(N) = N \log_2 N - 3N + 4 \tag{11}$$

multiplications and

$$\#\mathrm{A}_{\mathrm{FFT}}(N) = 3N \log_2 N - 3N + 4 \tag{12}$$

additions to execute an N-point DFT operation.

The computational complexities of the original TD algorithm for complex and real-valued data are summarized in Table I [3]. It is noted that there is no significant reduction in the computational complexity of the original TD algorithm when the input data are real-valued.

If the input data are complex, the MTD algorithm requires a total of Q complex FFT operations. Hence, the computation involves a total of

$$\begin{aligned}\#\mathrm{M}_{\mathrm{MTD}} &= Q \cdot \#\mathrm{M}_{\mathrm{FFT}}(\mathrm{L}) + 2 \cdot 2L \left(\frac{Q}{2} - 1 \right) + 4L \\ &= N \log_2 L - 3N + 4Q + 2LQ \end{aligned} \tag{13}$$

TABLE I
COMPUTATIONAL COMPLEXITIES OF SPLIT-RADIX FFT METHOD, CONVENTIONAL TD ALGORITHM, AND PROPOSED MTD SCHEME

Split-radix FFT for complex data	Split-radix FFT for real data
$\#\mathrm{M}_{\mathrm{FFT}} = N\log_2 N - 3N + 4$	$\#\mathrm{M}_{\mathrm{RFFT}} = \frac{N}{2}\log_2 N - \frac{3N}{2} + 2$
$\#\mathrm{A}_{\mathrm{FFT}} = 3N\log_2 N - 3N + 4$	$\#\mathrm{A}_{\mathrm{RFFT}} = \frac{3N}{2}\log_2 N - \frac{5N}{2} + 2$
TD algorithm for complex data	**TD algorithm for real data**
$\#\mathrm{M}_{\mathrm{TD}} = N\log_2 L - 3N + 4Q + 4L(Q-1)$	$\#\mathrm{M}_{\mathrm{RTD}} = N\log_2 L - 3N + 4Q + 2L(Q-1)$
$\#\mathrm{A}_{\mathrm{TD}} = 3N\log_2 L - 3N + 4Q + 2L(Q-1)$	$\#\mathrm{A}_{\mathrm{RTD}} = 3N\log_2 L - 3N + 4Q$
MTD algorithm for complex data	**MTD algorithm for real data**
$\#\mathrm{M}_{\mathrm{MTD}} = N\log_2 L - 3N + 4Q + 2LQ$	$\#\mathrm{M}_{\mathrm{RMTD}} = \left(\frac{Q+1}{2}\right)(L\log_2 L - 3L + 4) + LQ$
$\#\mathrm{A}_{\mathrm{MTD}} = 3N\log_2 L - 3N + 4Q + 2L(Q-1)$	$\#\mathrm{A}_{\mathrm{RMTD}} = \left(\frac{Q+1}{2}\right)(3L\log_2 L - 3L + 4) + (Q-3)L$

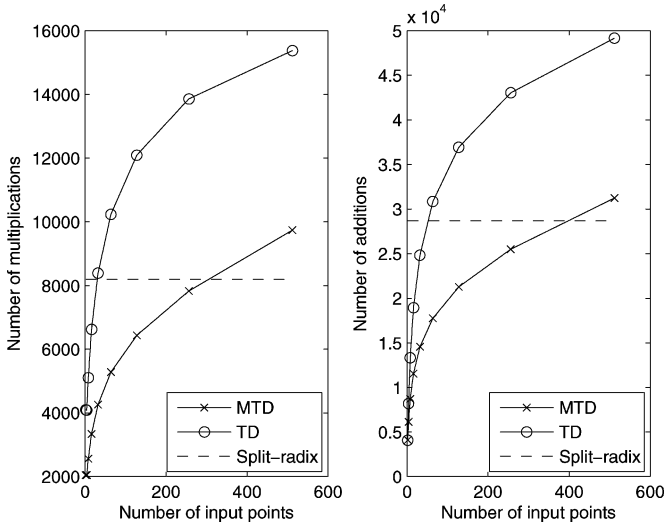

Fig. 1. Computational complexities of split-radix FFT method, conventional TD algorithm, and proposed MTD scheme for 2048-point DFT.

multiplication operations and

$$\begin{aligned}\#\mathrm{A}_{\mathrm{MTD}} &= Q\cdot\#\mathrm{A}_{\mathrm{FFT}}(\mathrm{L}) + 2\cdot 2L\left(\frac{Q}{2}-1\right) + 2L \\ &= 3N\log_2 L - 3N + 4Q + 2L(Q-1) \qquad (14)\end{aligned}$$

additions. In other words, the MTD algorithm requires fewer computations than the original TD scheme even when the input data are complex rather than real-valued. In evaluating the computational complexity of the MTD scheme for the case of real inputs, it is recalled that the N-point real-valued split-radix FFT requires [3], [5]

$$\#\mathrm{M}_{\mathrm{RFFT}}(N) = \frac{N}{2}\log_2 N - \frac{3N}{2} + 2 \qquad (15)$$

multiplications and

$$\#\mathrm{A}_{\mathrm{RFFT}}(N) = \frac{3N}{2}\log_2 N - \frac{5N}{2} + 2 \qquad (16)$$

additions, which is approximately one half the number of operations required when the input data are complex. Since the proposed MTD algorithm requires just $Q-1$ real FFT operations and one complex FFT operation when the input data are real-valued, the MTD algorithm requires just

$$\begin{aligned}\#\mathrm{M}_{\mathrm{RMTD}} &= (Q-1)\cdot\#\mathrm{M}_{\mathrm{RFFT}}(L) + \#\mathrm{M}_{\mathrm{FFT}}(L) \\ &\quad + 2\cdot L\left(\frac{Q}{2}-1\right) + 2L \\ &= \left(\frac{Q+1}{2}\right)(L\log_2 L - 3L + 4) + LQ \qquad (17)\end{aligned}$$

multiplications and

$$\begin{aligned}\#\mathrm{A}_{\mathrm{RMTD}} &= (Q-1)\cdot\#\mathrm{A}_{\mathrm{RFFT}}(L) + \#\mathrm{A}_{\mathrm{FFT}}(L) \\ &\quad + 2\cdot 2L\left(\frac{Q}{2}-1\right) \\ &= \left(\frac{Q+1}{2}\right)(3L\log_2 L - 3L + 4) + (Q-3)L \qquad (18)\end{aligned}$$

additions.

The analyses presented above clearly demonstrate the lower computational complexity of the proposed MTD algorithm compared to the original TD method for problems involving real-valued inputs. Fig. 1 presents the numerical results obtained for the split-radix FFT method, the conventional TD algorithm and the proposed MTD scheme when applied to a 2048-point DFT. It is observed that the MTD algorithm has a significantly lower computational complexity than the original TD algorithm. In addition, the MTD method is also more computationally efficient than the split-radix FFT method when the number of nonzero input data is small.

IV. CONCLUSIONS

This letter has proposed a MTD method for DFT operation in which L consecutive inputs to the transformation function are real, while the others are zero. The analytical and numerical results confirm that the complexity of the MTD scheme is significantly lower than that of the original TD method when the nonzero input data are real-valued. In addition, the computational complexity of the proposed scheme is also significantly lower than that of the traditional split-radix FFT method when the length of the input sequence is short.

REFERENCES

[1] J. D. Markel, "FFT pruning," *IEEE Trans. Audio Electroacoust.*, vol. AU-19, no. 4, pp. 305–311, Dec. 1971.
[2] D. P. Skinner, "Pruning the decimation in-time FFT algorithm," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-24, pp. 193–194, Apr. 1976.
[3] H. V. Sorensen and C. S. Burrus, "Efficient computation of the DFT with only a subset of input or output points," *IEEE Trans. Signal Process.*, vol. 41, no. 3, pp. 1184–1200, Mar. 1993.
[4] C. S. Burrus and T. W. Parks, *DFT/FFT and Convolution Algorithms*. New York: Wiley, 1985.
[5] H. V. Sorensen, D. L. Jones, M. T. Heideman, and C. S. Burrus, "Real-valued fast Fourier transform algorithms," *IEEE Trans Acoust., Speech, Signal Process.*, vol. ASSP-35, pp. 849–863, Jun. 1987.
[6] D. Takahashi, "An extended split-radix FFT algorithm," *IEEE Signal Process. Lett.*, vol. 8, no. 5, pp. 145–147, May 2001.