for the vector $\boldsymbol{v}$. Now, if this equation is premultiplied by $U = U_1$ or U_2, then $\boldsymbol{v}$ also satisfies

$$UC\boldsymbol{v} = U\boldsymbol{d}. \tag{21}$$

With the change of variables $\boldsymbol{v} = U^*\boldsymbol{q}$, substitution into (21) results in

$$UCU^*\boldsymbol{q} = U\boldsymbol{d}. \tag{22}$$

Consequently, the solution vector $\boldsymbol{v}$ of (20) can be obtained from the solution vector $\boldsymbol{q}$ of (22) or $\boldsymbol{q}$ can be obtained from $\boldsymbol{v}$.

Note that if a system of real equations is Toeplitz-plus-Hankel $(T + H)$, where T is symmetric Toeplitz and H is skew-centrosymmetric Hankel, then the equations may be transformed into Hermitian Toeplitz and solved with $1.25n^2 + O(n)$ complex multiplies or $3.75n^2 + O(n)$ real multiplies. This is a significant improvement in complexity over the approach of [8] which requires $12n^2 + O(n)$ real multiplies, and is slightly lower in complexity than the approach found in [9] which uses an entirely different development and requires $6n^2 + O(n)$ real multiplies.

III. Conclusion

In this correspondence, we have shown that constant unitary matrices exist which transform a Hermitian Toeplitz matrix into a real Toeplitz-plus-Hankel structure. As a consequence of this property, some real symmetric Toeplitz-plus-Hankel matrices may be converted to Hermitian Toeplitz matrices and vice versa. The unitary matrices presented are different from the one given in [2] and have two advantages: i) they transform Hermitian Toeplitz matrices into real matrices also possessing a special form, i.e., Toeplitz-plus-Hankel, (the result of [2] does not) and ii) the unitary matrices are simple to express mathematically, thus making it easier to manipulate and interpret results based on them for analytical purposes. A consequence of the unitary transformation U_1 presented in this correspondence is a relationship between the real and imaginary parts of eigenvectors of Hermitian Toeplitz matrices. This relationship also differs from the eigenvector symmetry property often given in the literature, e.g., [1].

As a practical observation on eigenspace decomposition consider the following. If all eigenvalues and eigenvectors are required, it is more efficient to transform a Hermitian Toeplitz matrix to a real matrix (using either the result of this paper or that of [2]) and use a standard algorithm such as the QR algorithm or Jacobi rotations [5]. If only a few eigenvalue/eigenvector pairs are needed, however, it will be more efficient to work on with the Hermitian Toeplitz matrix using a fast algorithm such as the Toeplitz eigensystem solver (TESS) [10] that exploits the special structure and can find specific eigenpairs.

References

[1] M. J. Goldstein, "Reduction of the eigenproblem for Hermitian persymmetric matrices," *Math. Computation*, vol. 28, no. 125, pp. 237–238, Jan. 1974.

[2] A. Lee, "Centrohermitian and skew-centrohermitian matrices," *Linear Alg. Appl.*, vol. 29, pp. 211–216, 1980.

[3] N. Levinson, "The Wiener rms (root-mean-square) error criterion in filter design and prediction," *J. Math. Phys.*, vol. 25, pp. 261–278.

[4] H. Krishna and S. D. Morgera, "The Levinson recurrence and fast algorithms for solving Toeplitz systems of liner equations," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, no. 6, pp. 839–848, June 1987.

[5] E. H. Golub and C. Van Loan, *Matrix Computations*. Baltimore, MD: John Hopkins University Press, 1983.

[6] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*. New York: Clarendon, 1965.

[7] A. Cantoni and P. Butler, "Eigenvalues and eigenvectors of symmetric centrosymmetric matrices," *Linear Alg. Its Appl.*, pp. 275–288, 1976.

[8] G. A. Merchant and T. W. Parks, "Efficient solution of a Toeplitz-plus-Hankel coefficient matrix system of equations," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, no. 1, pp. 40–44, Feb. 1982.

[9] I. Gohberg and I. Koltracht, "Efficient algorithm for Toeplitz-plus-Hankel matrices," *Integral Equations Oper. Theory*, vol. 12, no. 1, pp. 136–142, 1989.

[10] Y. H. Hu and S.-Y. Kung, "Toeplitz eigensystem solver," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, no. 4, pp. 1264–1271, Oct. 1985.

An FPT Algorithm with a Modularized Structure for Computing Two-Dimensional Discrete Fourier Transforms

Ja-Ling Wu and Yuh-Ming Huang

***Abstract*—In this correspondence, the fast polynomial transform (FPT) for computing two-dimensional (2-D) discrete Fourier transforms (DFT's) is modularized into identical modules. In this new method, only FPT's and FFT's of the same length are required. As a consequence, the architecture is more regular and naturally suitable for multiprocessor and VLSI implementations.**

I. Introduction

Polynomial transforms, defined in rings of polynomials, have been shown to give efficient algorithms for the computation of 2-D convolutions [1], [2] and to map multidimensional DFT's into 1-D DFT's [3]. In this approach, the $Z^N - 1$ polynomial ring is first factorized into several irreducible cyclotomic polynomials and then operated on by the polynomial transforms. Thus, one has to use FFT's and FPT's with different lengths through the FPT stages. This fact complicates the control of multiprocessor and/or VLSI FPT.

Recently, Truong *et al.* [4] proposed the modularized FPT structure for computing 2-D cyclic convolutions. In this approach, the 1-D cyclic polynomial convolution is decomposed into cyclic convolutions of polynomials, all of the same length. The regularity of the new algorithm makes it of great practical value. Based on the ideas in [4], in this correspondence, we modularized the FPT algorithm for computing 2-D DFT's. In this new method, only FPT's of length N_1 and FFT's of length $N_2/2^r$ are required.

II. The Modularized FPT Algorithm for 2-D DFT'S

Without loss of generality, let us consider the 2-D DFT of the $N_1^* N_2$ complex sequence x_{n_1,n_2} defined by

$$\bar{X}_{k_1,k_2} = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x_{n_1,n_2} W_{N_1}^{n_1 k_1} W_{N_2}^{n_2 k_2} \tag{1}$$

where $N_i = 2^{t_i}$, and $W_{N_i} = e^{-j2\pi/N_i}$, for $i = 1, 2$, and $t_1 \le t_2$.

Manuscript received February 6, 1989; revised February 12, 1991.

The authors are with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 10764, Republic of China.

IEEE Log Number 9101527.

In order to compute $\bar{X}_{k_1,k_2}$ by the polynomial transform (PT) [3], we represent once again (1) by a set of three polynomial equations

$$\bar{X}_{k_1}(Z) = \sum_{n_1=0}^{N_1-1} X_{n_1}(Z) W_{N_1}^{n_1 k_1} \bmod (Z^{N_2} - 1) \tag{2a}$$

$$X_{n_1}(Z) = \sum_{n_2=0}^{N_2-1} x_{n_1,n_2} Z^{n_2} \tag{2b}$$

$$\bar{X}_{k_1,k_2} = \bar{X}_{k_1}(Z) \bmod (Z - W_{N_2}^{k_2}). \tag{2c}$$

Based on [4], if the polynomial $Z^{N_2} - 1$ is decomposed as

$$Z^{N_2} - 1 = \prod_{k=0}^{2^r-1} (Z^{N_2/2^r} - \alpha^k) \tag{3}$$

where $1 \le r \le t_2$ and $\alpha = e^{j2\pi/2^r}$, i.e., the 2^rth root of unity.

Then the index k_2 can be partitioned into 2^r sections by

$$k_2 = 2^r \cdot d + t \tag{4}$$

where $0 \le t \le 2^r - 1$ and $0 \le d \le N_2/2^r - 1$.

Furthermore, it can easily shown that

$$Z - W_{N_2}^{2^r \cdot d + t} \mid Z^{N_2/2^r} - \alpha^{2^r - t} \tag{5}$$

where "$a|b$" means a is divided by b.

Let

$$X_{n_1}^t(Z) = \sum_{n_2=0}^{N_2/2^r-1} x_{n_1,n_2}^t Z^{n_2} \equiv X_{n_1}(Z) \bmod (Z^{N_2/2^r} - \alpha^{2^r - t}). \tag{6}$$

Derivations and discussions of the FPT algorithm for 2-D DFT's are given in [3] and are not repeated here. From [3] and (3)–(6) it follows that, for k_2 odd ($k_2 = 2^r \cdot d + t$, t: odd), (2) can be reformulated as

$$\bar{X}_{k_2 k_1}^t(Z) = \sum_{n_1=0}^{N_1-1} X_{n_1}^t(Z) (Z^{N_2/N_1})^{n_1 k_1} \bmod (Z^{N_2/2^r} - \alpha^{2^r - t}) \tag{7a}$$

$$\bar{X}_{k_2 k_1, k_2} = \bar{X}_{k_2 k_1}^t(Z) \bmod (Z - W_{N_2}^{k_2}). \tag{7b}$$

Notice that, since $(k_2, N_1) = 1$ $k_2 k_1 \bmod N_1$ is just only a permutation of k_1. Now with $k_2 = 2^r \cdot d + t$ and let

$$\bar{X}_{(2^r \cdot d + t)k_1}^t(Z) = \sum_{n=0}^{N_2/2^r-1} y_{k_1,n}^t Z^n \tag{8a}$$

then

$$\begin{aligned} \bar{X}_{(2^r \cdot d + t)k_1, (2^r \cdot d + t)} &= \sum_{n=0}^{N_2/2^r-1} y_{k_1,n}^t (W_{N_2}^{(2^r \cdot d + t)n} \\ &= \sum_{n=0}^{N_2/2^r-1} (y_{k_1,n}^t \cdot W_{N_2}^{nt}) W_{N_2/2^r}^{nd}. \end{aligned} \tag{8b}$$

Thus, for fixed k_1, the evaluation of (7b) becomes $2^{r-1} N_2/2^r$-point 1-D DFT's as described in (8). Further, since k_2 is odd here, (8) can be computed very efficiently in a FFT-like manner by using the generalized FFT algorithm developed in [5] and [6]. And the calculations of remainders mod $(Z^{N_2/2^r} - \alpha^{2^r - t})$, required in (6) and (7a), can be accomplished by a butterfly-like structure as described in [4]. Now, let us change our attention to the case of k_2 is even. For k_2 even ($k_2 = 2^r \cdot d + t$, t: even). Since

$$Z - W_{N_2}^{k_2+1} \mid Z^{N_2/2^r} - \alpha^{2^r - t - 1}$$

the same as (7), for k_2: even, (2) can also be reformulated as

$$\begin{aligned} \bar{X}_{(k_2+1)k_1}^t(Z) = \sum_{n_1=0}^{N_1-1} [X_{n_1}^t(Z) W_{N_2}^{-n_2}] (Z^{N_2/N_1})^{n_1 k_1} \\ \cdot \bmod (Z^{N_2/2^r} - \alpha^{2^r - t - 1}) \end{aligned} \tag{9a}$$

$$\bar{X}_{(k_2+1)k_1, k_2} = \bar{X}_{(k_2+1)k_1}^t(Z) \bmod (Z - W_{N_2}^{k_2+1}),$$

$$k_2 : 2^r \cdot d + t,\ t \text{ even}. \tag{9b}$$

In this case, if follows that

$$k_2 + 1 = 2^r \cdot d + t + 1 = 2^r \cdot d + t'$$

where $1 \le t'$: odd $\le 2^r - 1$, and

$$Z - W_{N_2}^{2^r \cdot d + t'} \mid Z^{N_2/2^r} - \alpha^{2^r - t'}.$$

Hence by comparing (7) with (9), it is easy to conclude that, for k_2 even, the FPT algorithm for 2-D DFT's can also be modularized following the same procedures for the case of k_2 odd.

III. Conclusions and Discussions

In highly parallel computational environments, multiprocessor, or VLSI systems, the system's throughput rate depends not only on the number of total arithmetic operations but also on the regularity of the algorithm. The modularized FPT algorithms developed in [4] and extended in this correspondence can simplify the problems of control, memory management, load balancing, etc., although more arithmetic operations are needed than with the original ones.

References

[1] H. J. Nussbaumer, "Fast polynomial transform algorithms for digital convolution," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 205–215, Apr. 1980.

[2] T. K. Truong *et al.*, "On the application of a fast polynomial transform and the Chinese remainder theorem to compute a two-dimensional convolution," *IEEE Trans. Acoust., Speech, Signal Processing.*, vol. ASSP-29, pp. 91–99, Feb. 1981.

[3] H. J. Nussbaumer and Quandalle, "Fast computation of discrete Fourier transforms using polynomial transforms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 169–181, Apr. 1979.

[4] T. K. Truong *et al.*, "An FPT algorithm with a modularized structure for computing 2-D cyclic convolutions," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-36, pp. 1540–1542, Sept. 1988.

[5] P. Corsimi and G. Frosini, "Properties of the multidimensional generalized discrete Fourier transform," *IEEE Trans. Comput.*, vol. C-28, pp. 819–830, 1979.

[6] I. S. Reed *et al.*, "An improved FPT algorithm for computing two-dimensional cyclic convolution," *IEEE Trans. Acoust., Speech, Signal Processsing*, vol. ASSP-31, pp. 1048–1050, Aug. 1983.