B is a diagonal matrix of multiplications,

C is a matrix with entries $0, \pm 1, \pm i$, defining the 'post-additions'.

The elements on the diagonal of **B** can be shown to be either real or pure imaginary, by the same argument as in Section 1.3.3.2.3.1. Matrices **A** and **C** may be rectangular rather than square, so that intermediate results may require extra storage space.

1.3.3.3. *Multidimensional algorithms*

From an algorithmic point of view, the distinction between one-dimensional (1D) and multidimensional DFTs is somewhat blurred by the fact that some factoring techniques turn a 1D transform into a multidimensional one. The distinction made here, however, is a practical one and is based on the dimensionality of the indexing sets for data and results. This section will therefore be concerned with the problem of factoring the DFT when the *indexing sets* for the input data and output results are multidimensional.

1.3.3.3.1. *The method of successive one-dimensional transforms*

The DFT was defined in Section 1.3.2.7.4 in an n-dimensional setting and it was shown that when the decimation matrix **N** is diagonal, say $\mathbf{N} = \mathrm{diag}(N^{(1)}, N^{(2)}, \ldots, N^{(n)})$, then $\bar{F}(N)$ has a tensor product structure:

$$\bar{F}(\mathbf{N}) = \bar{F}(N^{(1)}) \otimes \bar{F}(N^{(2)}) \otimes \ldots \otimes \bar{F}(N^{(n)}).$$

This may be rewritten as follows:

$$\begin{aligned}\bar{F}(\mathbf{N}) = {} & [\bar{F}(N^{(1)}) \otimes I_{N^{(2)}} \otimes \ldots \otimes I_{N^{(n)}}] \\ & \times [I_{N^{(1)}} \otimes \bar{F}(N^{(2)}) \otimes \ldots \otimes I_{N^{(n)}}] \\ & \times \ldots \\ & \times [I_{N^{(1)}} \otimes I_{N^{(2)}} \otimes \ldots \otimes \bar{F}(N^{(n)}],\end{aligned}$$

where the I's are identity matrices and $\times$ denotes ordinary matrix multiplication. The matrix within each bracket represents a one-dimensional DFT along one of the n dimensions, the other dimensions being left untransformed. As these matrices commute, the order in which the successive 1D DFTs are performed is immaterial.

This is the most straightforward method for building an n-dimensional algorithm from existing 1D algorithms. It is known in crystallography under the name of 'Beevers–Lipson factorization' (Section 1.3.4.3.1), and in signal processing as the 'row–column method'.

1.3.3.3.2. *Multidimensional factorization*

Substantial reductions in the arithmetic cost, as well as gains in flexibility, can be obtained if the factoring of the DFT is carried out in several dimensions simultaneously. The presentation given here is a generalization of that of Mersereau & Speake (1981), using the abstract setting established independently by Auslander, Tolimieri & Winograd (1982).

Let us return to the general n-dimensional setting of Section 1.3.2.7.4, where the DFT was defined for an arbitrary decimation matrix **N** by the formulae (where $|\mathbf{N}|$ denotes $|\det \mathbf{N}|$):

$$F(\mathbf{N}): \quad X(\mathbf{k}) = \frac{1}{|\mathbf{N}|} \sum_{\mathbf{k}^*} X^*(\mathbf{k}^*) e[-\mathbf{k}^* \cdot (\mathbf{N}^{-1}\mathbf{k})]$$

$$\bar{F}(\mathbf{N}): \quad X^*(\mathbf{k}^*) = \sum_{\mathbf{k}} X(\mathbf{k}) e[\mathbf{k}^* \cdot (\mathbf{N}^{-1}\mathbf{k})]$$

with

$$\mathbf{k} \in \mathbb{Z}^n/\mathbf{N}\mathbb{Z}^n, \quad \mathbf{k}^* \in \mathbb{Z}^n/\mathbf{N}^T\mathbb{Z}^n.$$

1.3.3.3.2.1. *Multidimensional Cooley–Tukey factorization*

Let us now assume that this decimation can be factored into d successive decimations, *i.e.* that

$$\mathbf{N} = \mathbf{N}_1\mathbf{N}_2 \ldots \mathbf{N}_{d-1}\mathbf{N}_d$$

and hence

$$\mathbf{N}^T = \mathbf{N}_d^T\mathbf{N}_{d-1}^T \ldots \mathbf{N}_2^T\mathbf{N}_1^T.$$

Then the coset decomposition formulae corresponding to these successive decimations (Section 1.3.2.7.1) can be combined as follows:

$$\begin{aligned}\mathbb{Z}^n &= \bigcup_{\mathbf{k}_1} (\mathbf{k}_1 + \mathbf{N}_1\mathbb{Z}^n) \\ &= \bigcup_{\mathbf{k}_1} \left\{ \mathbf{k}_1 + \mathbf{N}_1 \left[\bigcup_{\mathbf{k}_2} (\mathbf{k}_2 + \mathbf{N}_2\mathbb{Z}^n) \right] \right\} \\ &= \ldots \\ &= \bigcup_{\mathbf{k}_1} \ldots \bigcup_{\mathbf{k}_d} (\mathbf{k}_1 + \mathbf{N}_1\mathbf{k}_2 + \ldots + \mathbf{N}_1\mathbf{N}_2 \times \ldots \times \mathbf{N}_{d-1}\mathbf{k}_d + \mathbf{N}\mathbb{Z}^n)\end{aligned}$$

with $\mathbf{k}_j \in \mathbb{Z}^n/\mathbf{N}_j\mathbb{Z}^n$. Therefore, any $\mathbf{k} \in \mathbb{Z}/\mathbf{N}\mathbb{Z}^n$ may be written uniquely as

$$\mathbf{k} = \mathbf{k}_1 + \mathbf{N}_1\mathbf{k}_2 + \ldots + \mathbf{N}_1\mathbf{N}_2 \times \ldots \times \mathbf{N}_{d-1}\mathbf{k}_d.$$

Similarly:

$$\begin{aligned}\mathbb{Z}^n &= \bigcup_{\mathbf{k}_d^*} (\mathbf{k}_d^* + \mathbf{N}_d^T\mathbb{Z}^n) \\ &= \ldots \\ &= \bigcup_{\mathbf{k}_d^*} \ldots \bigcup_{\mathbf{k}_1^*} (\mathbf{k}_d^* + \mathbf{N}_d^T\mathbf{k}_{d-1}^* + \ldots + \mathbf{N}_d^T \times \ldots \times \mathbf{N}_2^T\mathbf{k}_1^* \\ &\quad + \mathbf{N}^T\mathbb{Z}^n)\end{aligned}$$

so that any $\mathbf{k}^* \in \mathbb{Z}^n/\mathbf{N}^T\mathbb{Z}^n$ may be written uniquely as

$$\mathbf{k}^* = \mathbf{k}_d^* + \mathbf{N}_d^T\mathbf{k}_{d-1}^* + \ldots + \mathbf{N}_d^T \times \ldots \times \mathbf{N}_2^T\mathbf{k}_1^*$$

with $\mathbf{k}_j^* \in \mathbb{Z}^n/\mathbf{N}_j^T\mathbb{Z}^n$. These decompositions are the vector analogues of the multi-radix number representation systems used in the Cooley–Tukey factorization.

We may then write the definition of $\bar{F}(\mathbf{N})$ with $d = 2$ factors as

$$\begin{aligned}X^*(\mathbf{k}_2^* + \mathbf{N}_2^T\mathbf{k}_1^*) = {} & \sum_{\mathbf{k}_1} \sum_{\mathbf{k}_2} X(\mathbf{k}_1 + \mathbf{N}_1\mathbf{k}_2) \\ & \times e[(\mathbf{k}_2^{*T} + \mathbf{k}_1^{*T}\mathbf{N}_2)\mathbf{N}_2^{-1}\mathbf{N}_1^{-1}(\mathbf{k}_1 + \mathbf{N}_1\mathbf{k}_2)].\end{aligned}$$

The argument of $e(-)$ may be expanded as

$$\mathbf{k}_2^* \cdot (\mathbf{N}^{-1}\mathbf{k}_1) + \mathbf{k}_1^* \cdot (\mathbf{N}_1^{-1}\mathbf{k}_1) + \mathbf{k}_2^* \cdot (\mathbf{N}_2^{-1}\mathbf{k}_2) + \mathbf{k}_1^* \cdot \mathbf{k}_2.$$

The first summand may be recognized as a twiddle factor, the second and third as the kernels of $\bar{F}(\mathbf{N}_1)$ and $\bar{F}(\mathbf{N}_2)$, respectively, while the fourth is an integer which may be dropped. We are thus led to a 'vector-radix' version of the Cooley–Tukey algorithm, in which the successive decimations may be introduced in all n dimensions simultaneously by general integer matrices. The computation may be decomposed into five stages analogous to those of the one-dimensional algorithm of Section 1.3.3.2.1:

(i) form the $|\mathbf{N}_1|$ vectors $\mathbf{Y}_{\mathbf{k}_1}$ of shape $\mathbf{N}_2$ by

$$Y_{\mathbf{k}_1}(\mathbf{k}_2) = X(\mathbf{k}_1 + \mathbf{N}_1\mathbf{k}_2), \quad \mathbf{k}_1 \in \mathbb{Z}^n/\mathbf{N}_1\mathbb{Z}^n, \quad \mathbf{k}_2 \in \mathbb{Z}^n/\mathbf{N}_2\mathbb{Z}^n;$$

(ii) calculate the $|\mathbf{N}_1|$ transforms $\mathbf{Y}^*_{\mathbf{k}_1}$ on $|\mathbf{N}_2|$ points:

$$Y^*_{\mathbf{k}_1}(\mathbf{k}^*_2) = \sum_{\mathbf{k}_2} e[\mathbf{k}^*_2 \cdot (\mathbf{N}_2^{-1}\mathbf{k}_2)] Y_{\mathbf{k}_1}(\mathbf{k}_2), \quad \mathbf{k}_1 \in \mathbb{Z}^n/\mathbf{N}_1\mathbb{Z}^n;$$

(iii) form the $|\mathbf{N}_2|$ vectors $\mathbf{Z}_{\mathbf{k}^*_2}$ of shape $\mathbf{N}_1$ by

$$Z_{\mathbf{k}^*_2}(\mathbf{k}_1) = e[\mathbf{k}^*_2 \cdot (\mathbf{N}^{-1}\mathbf{k}_1)] Y^*_{\mathbf{k}_1}(\mathbf{k}^*_2), \quad \mathbf{k}_1 \in \mathbb{Z}^n/\mathbf{N}_1\mathbb{Z}^n, \quad \mathbf{k}^*_2 \in \mathbb{Z}^n/\mathbf{N}_2^T\mathbb{Z}^n;$$

(iv) calculate the $|\mathbf{N}_2|$ transforms $\mathbf{Z}^*_{\mathbf{k}^*_2}$ on $|\mathbf{N}_1|$ points:

$$Z^*_{\mathbf{k}^*_2}(\mathbf{k}^*_1) = \sum_{\mathbf{k}_1} e[\mathbf{k}^*_1 \cdot (\mathbf{N}_1^{-1}\mathbf{k}_1)] Z_{\mathbf{k}^*_2}(\mathbf{k}_1), \quad \mathbf{k}^*_2 \in \mathbb{Z}^n/\mathbf{N}_2^T\mathbb{Z}^n;$$

(v) collect $X^*(\mathbf{k}^*_2 + \mathbf{N}_2^T\mathbf{k}^*_1)$ as $Z^*_{\mathbf{k}^*_2}(\mathbf{k}^*_1)$.

The initial $|\mathbf{N}|$-point transform $\bar{F}(\mathbf{N})$ can thus be performed as $|\mathbf{N}_1|$ transforms $\bar{F}(\mathbf{N}_2)$ on $|\mathbf{N}_2|$ points, followed by $|\mathbf{N}_2|$ transforms $\bar{F}(\mathbf{N}_1)$ on $|\mathbf{N}_1|$ points. This process can be applied successively to all d factors. The same decomposition applies to $F(\mathbf{N})$, up to the complex conjugation of twiddle factors, the normalization factor $1/|\mathbf{N}|$ being obtained as the product of the factors $1/|\mathbf{N}_j|$ in the successive partial transforms $F(\mathbf{N}_j)$.

The geometric interpretation of this factorization in terms of partial transforms on translates of sublattices applies in full to this n-dimensional setting; in particular, the twiddle factors are seen to be related to the residual translations which place the sublattices in register within the big lattice. If the intermediate transforms are performed *in place*, then the quantity

$$X^*(\mathbf{k}^*_d + \mathbf{N}_d^T\mathbf{k}^*_{d-1} + \ldots + \mathbf{N}_d^T\mathbf{N}_{d-1}^T \times \ldots \times \mathbf{N}_2^T\mathbf{k}^*_1)$$

will eventually be found at location

$$\mathbf{k}^*_1 + \mathbf{N}_1\mathbf{k}^*_2 + \ldots + \mathbf{N}_1\mathbf{N}_2 \times \ldots \times \mathbf{N}_{d-1}\mathbf{k}^*_d,$$

so that the final results will have to be *unscrambled* by a process which may be called 'coset reversal', the vector equivalent of digit reversal.

Factoring by 2 in all n dimensions simultaneously, *i.e.* taking $\mathbf{N} = 2\mathbf{M}$, leads to 'n-dimensional butterflies'. Decimation in time corresponds to the choice $\mathbf{N}_1 = 2\mathbf{I}, \mathbf{N}_2 = \mathbf{M}$, so that $\mathbf{k}_1 \in \mathbb{Z}^n/2\mathbb{Z}^n$ is an n-dimensional parity class; the calculation then proceeds by

$$Y_{\mathbf{k}_1}(\mathbf{k}_2) = X(\mathbf{k}_1 + 2\mathbf{k}_2), \quad \mathbf{k}_1 \in \mathbb{Z}^n/2\mathbb{Z}^n, \quad \mathbf{k}_2 \in \mathbb{Z}^n/\mathbf{M}\mathbb{Z}^n,$$

$$Y^*_{\mathbf{k}_1} = \bar{F}(\mathbf{M})[\mathbf{Y}_{\mathbf{k}_1}], \quad \mathbf{k}_1 \in \mathbb{Z}^n/2\mathbb{Z}^n;$$

$$X^*(\mathbf{k}^*_2 + \mathbf{M}^T\mathbf{k}^*_1) = \sum_{\mathbf{k}_1 \in \mathbb{Z}^n/2\mathbb{Z}^n} (-1)^{\mathbf{k}^*_1 \cdot \mathbf{k}_1} \times e[\mathbf{k}^*_2 \cdot (\mathbf{N}^{-1}\mathbf{k}_1)] Y^*_{\mathbf{k}_1}(\mathbf{k}^*_2).$$

Decimation in frequency corresponds to the choice $\mathbf{N}_1 = \mathbf{M}$, $\mathbf{N}_2 = 2\mathbf{I}$, so that $\mathbf{k}_2 \in \mathbb{Z}^n/2\mathbb{Z}^n$ labels 'octant' blocks of shape $\mathbf{M}$; the calculation then proceeds through the following steps:

$$Z_{\mathbf{k}^*_2}(\mathbf{k}_1) = \left[\sum_{\mathbf{k}_2 \in \mathbb{Z}^n/2\mathbb{Z}^n} (-1)^{\mathbf{k}^*_2 \cdot \mathbf{k}_2} X(\mathbf{k}_1 + \mathbf{M}\mathbf{k}_2)\right] \times e[\mathbf{k}^*_2 \cdot (\mathbf{N}^{-1}\mathbf{k}_1)],$$

$$\mathbf{Z}^*_{\mathbf{k}^*_2} = \bar{F}(\mathbf{M})[\mathbf{Z}_{\mathbf{k}^*_2}],$$

$$X^*(\mathbf{k}^*_2 + 2\mathbf{k}^*_1) = Z^*_{\mathbf{k}^*_2}(\mathbf{k}^*_1),$$

i.e. the 2^n parity classes of results, corresponding to the different $\mathbf{k}^*_2 \in \mathbb{Z}^n/2\mathbb{Z}^n$, are obtained separately. When the dimension n is 2 and the decimating matrix is diagonal, this analysis reduces to the 'vector radix FFT' algorithms proposed by Rivard (1977) and Harris *et al.* (1977). These lead to substantial reductions in the number M of multiplications compared to the row–column method: M is reduced to $3M/4$ by simultaneous 2×2 factoring, and to $15M/32$ by simultaneous 4×4 factoring.

The use of a non-diagonal decimating matrix may bring savings in computing time if the spectrum of the band-limited function under study is of such a shape as to pack more compactly in a non-rectangular than in a rectangular lattice (Mersereau, 1979). If, for instance, the support K of the spectrum Φ is contained in a sphere, then a decimation matrix producing a close packing of these spheres will yield an aliasing-free DFT algorithm with fewer sample points than the standard algorithm using a rectangular lattice.

1.3.3.3.2.2. *Multidimensional prime factor algorithm*

Suppose that the decimation matrix $\mathbf{N}$ is diagonal

$$\mathbf{N} = \text{diag } (N^{(1)}, N^{(2)}, \ldots, N^{(n)})$$

and let each diagonal element be written in terms of its prime factors:

$$N^{(i)} = \prod_{j=1}^{m} p_j^{\nu(i,\,j)},$$

where m is the total number of distinct prime factors present in the $N^{(i)}$.

The CRT may be used to turn each 1D transform along dimension i $(i = 1, \ldots, n)$ into a multidimensional transform with a separate 'pseudo-dimension' for each distinct prime factor of $N^{(i)}$; the number μ_i, of these pseudo-dimensions is equal to the cardinality of the set:

$$\{\, j \in \{1, \ldots, m\} | \nu(i,j) > 0 \text{ for some } i\}.$$

The full n-dimensional transform thus becomes μ-dimensional, with $\mu = \sum_{i=1}^{n} \mu_i$.

We may now permute the μ pseudo-dimensions so as to bring into contiguous position those corresponding to the same prime factor p_j; the m resulting groups of pseudo-dimensions are said to define 'p-primary' blocks. The initial transform is now written as a tensor product of m p-primary transforms, where transform j is on

$$p_j^{\nu(1,\,j)} \times p_j^{\nu(2,\,j)} \times \ldots \times p_j^{\nu(n,\,j)}$$

points [by convention, dimension i is not transformed if $\nu(i,j) = 0$]. These p-primary transforms may be computed, for instance, by multidimensional Cooley–Tukey factorization (Section 1.3.3.3.1), which is faster than the straightforward row–column method. The final results may then be obtained by reversing all the permutations used.

The extra gain with respect to the multidimensional Cooley–Tukey method is that *there are no twiddle factors between p-primary pieces corresponding to different primes p*.

The case where $\mathbf{N}$ is not diagonal has been examined by Guessoum & Mersereau (1986).

1.3.3.3.2.3. *Nesting of Winograd small FFTs*

Suppose that the CRT has been used as above to map an n-dimensional DFT to a μ-dimensional DFT. For each $\kappa = 1, \ldots, \mu$ [κ runs over those pairs (i, j) such that $\nu(i,j) > 0$], the Rader/Winograd procedure may be applied to put the matrix of the κth 1D DFT in the **CBA** normal form of a Winograd small FFT. The full DFT matrix may then be written, up to permutation of data and results, as

$$\bigotimes_{\kappa=1}^{\mu} (\mathbf{C}_\kappa \mathbf{B}_\kappa \mathbf{A}_\kappa).$$

A well known property of the tensor product of matrices allows this to be rewritten as

$$\left(\bigotimes_{\gamma=1}^{\mu}\mathbf{C}_\gamma\right)\times\left(\bigotimes_{\beta=1}^{\mu}\mathbf{B}_\beta\right)\times\left(\bigotimes_{\alpha=1}^{\mu}\mathbf{A}_\alpha\right)$$

and thus to form a matrix in which the *combined* pre-addition, multiplication and post-addition matrices have been *precomputed*. This procedure, called *nesting*, can be shown to afford a reduction of the arithmetic operation count compared to the row–column method (Morris, 1978).

Clearly, the nesting rearrangement need not be applied to all μ dimensions, but can be restricted to any desired subset of them.

1.3.3.3.2.4. *The Nussbaumer–Quandalle algorithm*

Nussbaumer's approach views the DFT as the evaluation of certain polynomials constructed from the data (as in Section 1.3.3.2.4). For instance, putting $\omega = e(1/N)$, the 1D N-point DFT

$$X^*(k^*) = \sum_{k=0}^{N-1} X(k)\omega^{k^*k}$$

may be written

$$X^*(k^*) = Q(\omega^{k^*}),$$

where the polynomial Q is defined by

$$Q(z) = \sum_{k=0}^{N-1} X(k)z^k.$$

Let us consider (Nussbaumer & Quandalle, 1979) a 2D transform of size $N \times N$:

$$X^*(k_1^*, k_2^*) = \sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1} X(k_1,k_2)\omega^{k_1^*k_1+k_2^*k_2}.$$

By introduction of the polynomials

$$Q_{k_2}(z) = \sum_{k_1} X(k_1,k_2)z^{k_1}$$

$$R_{k_2^*}(z) = \sum_{k_2} \omega^{k_2^*k_2}Q_{k_2}(z),$$

this may be rewritten:

$$X^*(k_1^*, k_2^*) = R_{k_2^*}(\omega^{k_1^*}) = \sum_{k_2} \omega^{k_2^*k_2}Q_{k_2}(\omega^{k_1^*}).$$

Let us now suppose that k_1^* is coprime to N. Then k_1^* has a unique inverse modulo N (denoted by $1/k_1^*$), so that multiplication by k_1^* simply permutes the elements of $\mathbb{Z}/N\mathbb{Z}$ and hence

$$\sum_{k_2=0}^{N-1} f(k_2) = \sum_{k_2=0}^{N-1} f(k_1^*k_2)$$

for any function f over $\mathbb{Z}/N\mathbb{Z}$. We may thus write:

$$\begin{aligned} X^*(k_1^*, k_2^*) &= \sum_{k_2} \omega^{k_1^*k_2^*k_2}Q_{k_1^*k_2}(\omega^{k_1^*}) \\ &= S_{k_1^*k_2}(\omega^{k_1^*}) \end{aligned}$$

where

$$S_{k^*}(z) = \sum_{k_2} z^{k^*k_2}Q_{k_2}(z).$$

Since only the value of polynomial $S_{k^*}(z)$ at $z = \omega^{k_1^*}$ is involved in the result, the computation of S_{k^*} may be carried out modulo the unique cyclotomic polynomial $P(z)$ such that $P(\omega^{k_1^*}) = 0$. Thus, if we define:

$$T_{k^*}(z) = \sum_{k_2} z^{k^*k_2}Q_{k_2}(z) \bmod P(z)$$

we may write:

$$X^*(k_1^*, k_2^*) = T_{k_1^*k_2^*}(\omega^{k_1^*})$$

or equivalently

$$X^*\left(k_1^*, \frac{k_2^*}{k_1^*}\right) = T_{k_2^*}(\omega^{k_1^*}).$$

For N an odd prime p, all non-zero values of k_1^* are coprime with p so that the $p \times p$-point DFT may be calculated as follows:

(1) form the polynomials

$$T_{k_2^*}(z) = \sum_{k_1}\sum_{k_2} X(k_1,k_2)z^{k_1+k_2^*k_2} \bmod P(z)$$

for $k_2^* = 0, \ldots, p-1$;

(2) evaluate $T_{k_2^*}(\omega^{k_1^*})$ for $k_1^* = 0, \ldots, p-1$;

(3) put $X^*(k_1^*, k_2^*/k_1^*) = T_{k_2^*}(\omega^{k_1^*})$;

(4) calculate the terms for $k_1^* = 0$ separately by

$$X^*(0, k_2^*) = \sum_{k_2}\left[\sum_{k_1} X(k_1,k_2)\right]\omega^{k_2^*k_2}.$$

Step (1) is a set of p 'polynomial transforms' involving no multiplications; step (2) consists of p DFTs on p points each since if

$$T_{k_2^*}(z) = \sum_{k_1} Y_{k_2^*}(k_1)z^{k_1}$$

then

$$T_{k_2^*}(\omega^{k_1^*}) = \sum_{k_1} Y_{k_2^*}(k_1)\omega^{k_1^*k_1} = Y_{k_2^*}^*(k_1^*);$$

step (3) is a permutation; and step (4) is a p-point DFT. Thus the 2D DFT on $p \times p$ points, which takes $2p$ p-point DFTs by the row–column method, involves only $(p+1)$ p-point DFTs; the other DFTs have been replaced by polynomial transforms involving only additions.

This procedure can be extended to n dimensions, and reduces the number of 1D p-point DFTs from np^{n-1} for the row–column method to $(p^n-1)/(p-1)$, at the cost of introducing extra additions in the polynomial transforms.

A similar algorithm has been formulated by Auslander *et al.* (1983) in terms of Galois theory.

1.3.3.3.3. *Global algorithm design*

1.3.3.3.3.1. *From local pieces to global algorithms*

The mathematical analysis of the structure of DFT computations has brought to light a broad variety of possibilities for reducing or reshaping their arithmetic complexity. All of them are 'analytic' in that they break down large transforms into a succession of smaller ones.

These results may now be considered from the converse 'synthetic' viewpoint as providing a list of procedures for assembling them:

(i) the building blocks are one-dimensional p-point algorithms for p a small prime;

(ii) the low-level connectors are the multiplicative reindexing methods of Rader and Winograd, or the polynomial transform reindexing method of Nussbaumer and Quandalle, which allow the construction of efficient algorithms for larger primes p, for prime powers p^ν, and for p-primary pieces of shape $p^\nu \times \ldots \times p^\nu$;

(iii) the high-level connectors are the additive reindexing scheme of Cooley–Tukey, the Chinese remainder theorem reindexing, and the tensor product construction;

(iv) nesting may be viewed as the 'glue' which seals all elements.

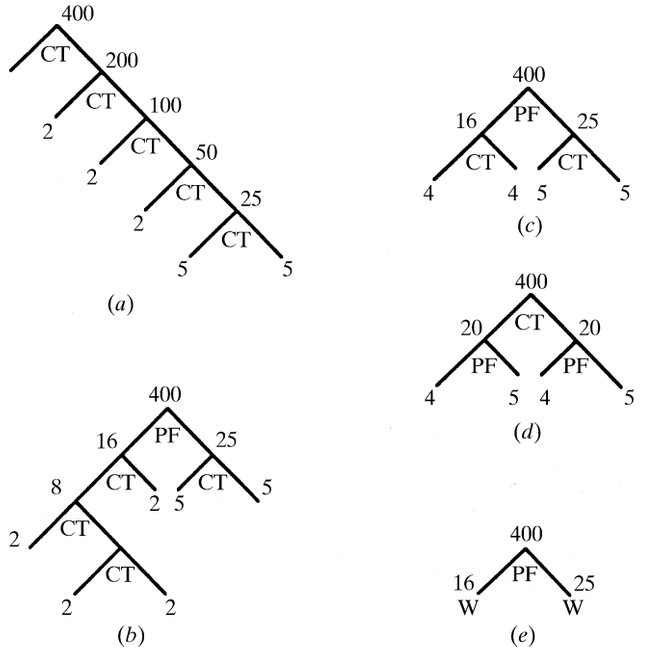

Fig. 1.3.3.1. A few global algorithms for computing a 400-point DFT. CT: Cooley–Tukey factorization. PF: prime factor (or Good) factorization. W: Winograd algorithm.

The simplest DFT may then be carried out into a global algorithm in many different ways. The diagrams in Fig. 1.3.3.1 illustrate a few of the options available to compute a 400-point DFT. They may differ greatly in their arithmetic operation counts.

1.3.3.3.3.2. *Computer architecture considerations*

To obtain a truly useful measure of the computational complexity of a DFT algorithm, its arithmetic operation count must be tempered by computer architecture considerations. Three main types of trade-offs must be borne in mind:

(i) reductions in floating-point (f.p.) arithmetic count are obtained by reindexing, hence at the cost of an increase in integer arithmetic on addresses, although some shortcuts may be found (Uhrich, 1969; Burrus & Eschenbacher, 1981);

(ii) reduction in the f.p. multiplication count usually leads to a large increase in the f.p. addition count (Morris, 1978);

(iii) nesting can increase execution speed, but causes a loss of modularity and hence complicates program development (Silverman, 1977; Kolba & Parks, 1977).

Many of the mathematical developments above took place in the context of single-processor serial computers, where f.p. addition is substantially cheaper than f.p. multiplication but where integer address arithmetic has to compete with f.p. arithmetic for processor cycles. As a result, the alternatives to the Cooley–Tukey algorithm hardly ever led to particularly favourable trade-offs, thus creating the impression that there was little to gain by switching to more exotic algorithms.

The advent of new machine architectures with vector and/or parallel processing features has greatly altered this picture (Pease, 1968; Korn & Lambiotte, 1979; Fornberg, 1981; Swartzrauber, 1984):

(i) *pipelining* equalizes the cost of f.p. addition and f.p. multiplication, and the ideal 'blend' of the two types of operations depends solely on the number of adder and multiplier units available in each machine;

(ii) integer address arithmetic is delegated to specialized arithmetic and logical units (ALUs) operating concurrently with the f.p. units, so that complex reindexing schemes may be used without loss of overall efficiency.

Another major consideration is that of data flow [see *e.g.* Nawab & McClellan (1979)]. Serial machines only have few registers and few paths connecting them, and allow little or no overlap between computation and data movement. New architectures, on the other hand, comprise banks of vector registers (or 'cache memory') besides the usual internal registers, and dedicated ALUs can service data transfers between several of them simultaneously and concurrently with computation.

In this new context, the devices described in Sections 1.3.3.2 and 1.3.3.3 for altering the balance between the various types of arithmetic operations, and reshaping the data flow during the computation, are invaluable. The field of machine-dependent DFT algorithm design is thriving on them [see *e.g.* Temperton (1983*a*,*b*,*c*, 1985); Agarwal & Cooley (1986, 1987)].

1.3.3.3.3.3. *The Johnson–Burrus family of algorithms*

In order to explore systematically all possible algorithms for carrying out a given DFT computation, and to pick the one best suited to a given machine, attempts have been made to develop:

(i) a high-level notation of describing all the ingredients of a DFT computation, including data permutation and data flow;

(ii) a formal calculus capable of operating on these descriptions so as to represent all possible reorganizations of the computation;

(iii) an automatic procedure for evaluating the performance of a given algorithm on a specific architecture.

Task (i) can be accomplished by systematic use of a tensor product notation to represent the various stages into which the DFT can be factored (reindexing, small transforms on subsets of indices, twiddle factors, digit-reversal permutations).

Task (ii) may for instance use the Winograd **CBA** normal form for each small transform, then apply the rules governing the rearrangement of tensor product $\otimes$ and ordinary product $\times$ operations on matrices. The matching of these rearrangements to the architecture of a vector and/or parallel computer can be formalized algebraically [see *e.g.* Chapter 2 of Tolimieri *et al.* (1989)].

Task (iii) is a complex search which requires techniques such as dynamic programming (Bellman, 1958).

Johnson & Burrus (1983) have proposed and tested such a scheme to identify the optimal trade-offs between prime factor nesting and Winograd nesting of small Winograd transforms. In step (ii), they further decomposed the pre-addition matrix **A** and post-addition matrix **C** into several factors, so that the number of design options available becomes very large: the N-point DFT when N has four factors can be calculated in over 10^{12} distinct ways.

This large family of nested algorithms contains the prime factor algorithm and the Winograd algorithms as particular cases, but usually achieves greater efficiency than either by reducing the f.p. multiplication count while keeping the number of f.p. additions small.

There is little doubt that this systematic approach will be extended so as to incorporate all available methods of restructuring the DFT.

1.3.4. Crystallographic applications of Fourier transforms

1.3.4.1. *Introduction*

The central role of the Fourier transformation in X-ray crystallography is a consequence of the kinematic approximation used in the description of the scattering of X-rays by a distribution of electrons (Bragg, 1915; Duane, 1925; Havighurst, 1925*a*,*b*; Zachariasen, 1945; James, 1948*a*, Chapters 1 and 2; Lipson & Cochran, 1953, Chapter 1; Bragg, 1975).