6.2 Decimation-in-Time FFT Algorithms

To achieve the dramatic increase in efficiency to which we have alluded, it is necessary to decompose the DFT computation into successively smaller DFT computations. In this process we exploit both the symmetry and the periodicity of the complex exponential \( W_N^k = e^{-j\frac{2\pi}{N}kn} \). Algorithms in which the decomposition is based on decomposing the sequence \( x(n) \), into successively smaller subsequences, are called decimation-in-time algorithms.

The principle of decimation-in-time is most conveniently illustrated by considering the special case of \( N \) an integer power of 2; i.e.,

\[ N = 2^r \]

Since \( N \) is an even integer, we can consider computing \( X(k) \) by separating \( x(n) \) into two \( N/2 \)-point sequences consisting of the even-numbered points in \( x(n) \) and the odd-numbered points in \( x(n) \). With \( X(k) \) given by

\[
X(k) = \sum_{n=0}^{N-1} x(n)W_N^kn, \quad k = 0, 1, \ldots, N - 1
\]

and separating \( x(n) \) into its even- and odd-numbered points we obtain

\[
X(k) = \sum_{n \text{ even}} x(n)W_N^kn + \sum_{n \text{ odd}} x(n)W_N^kn
\]

or with the substitution of variables \( n = 2r \) for \( n \) even and \( n = 2r + 1 \) for \( n \) odd,

\[
X(k) = \sum_{r=0}^{(N/2)-1} x(2r)W_N^{2rk} + \sum_{r=0}^{(N/2)-1} x(2r + 1)W_N^{(2r+1)k}
\]

\[
= \sum_{r=0}^{(N/2)-1} x(2r)(W_N^2)^{rk} + W_N^k \sum_{r=0}^{(N/2)-1} x(2r + 1)(W_N^2)^{rk}
\]

But \( W_N^k : W_{N/2} \) since

\[
W_N^k = e^{-j\frac{2\pi}{N}kn} = e^{-j\frac{\pi}{N/2}} = W_{N/2}^k
\]

Consequently Eq. (6.10) can be written as

\[
X(k) = \sum_{r=0}^{(N/2)-1} x(2r)W_{N/2}^{2rk} + W_N^k \sum_{r=0}^{(N/2)-1} x(2r + 1)W_{N/2}^{2rk}
\]

\[
= G(k) + W_N^kH(k)
\]
Figure 6.3 indicates the computation involved in computing \( X(k) \) according to Eq. (6.11) for an eight-point sequence, i.e., for \( N = 8 \). In this figure we have used the signal flow graph conventions that were introduced in Chapter 4 for representing difference equations [5,7]. That is, branches entering a node are summed to produce the node variable. When no coefficient is indicated, the branch transmittance is assumed to be one. For other branches, the transmittance of a branch is an integer power of \( W_N \). Thus we note in Fig. 6.3 that two four-point DFTs are computed, with \( G(k) \) designating the four-point DFT of the even-numbered points and \( H(k) \) designating the four-point DFT of the odd-numbered points. \( X(0) \) is then obtained by multiplying \( H(0) \) by \( W_N^0 \) and adding the product to \( G(0) \). \( X(1) \) is obtained by multiplying \( H(1) \) by \( W_N^1 \) and adding that result to \( G(1) \). For \( X(4) \) we would want to multiply \( H(4) \) by \( W_N^4 \) and add the result to \( G(4) \). However, since \( G(k) \) and \( H(k) \) are both periodic in \( k \) with period 4, \( H(4) = H(0) \) and \( G(4) = G(0) \). Thus \( X(4) \) is obtained by multiplying \( H(0) \) by \( W_N^4 \) and adding the result to \( G(0) \).
With the computation restructured according to Eq. (6.11), we can compare the number of multiplications and additions required with those required for a direct computation of the DFT. Previously we saw that for direct computation without exploiting symmetry, \( N^2 \) complex multiplications and additions were required.\(^\dagger\) By comparison, Eq. (6.11) requires the computation of two \( N/2 \)-point DFTs, which in turn requires \( 2(N/2)^2 \) complex multiplications and approximately \( 2(N/2)^2 \) complex additions. Then the two \( N/2 \)-point DFTs must be combined, requiring \( N \) complex multiplications, corresponding to multiplying the second sum by \( W_N^k \) and then \( N \) complex additions, corresponding to adding that product to the first sum. Consequently, the computation of Eq. (6.11) for all values of \( k \) requires \( N + 2(N/2)^2 \) or \( N + (N^2/2) \) complex multiplications and complex additions. It is easy to verify that for \( N > 2 \), \( N + (N^2/2) \) will be less than \( N^2 \).

Equation (6.11) corresponds to breaking the original \( N \)-point computation into two \( N/2 \)-point computations. If \( N/2 \) is even, as it always is when \( N \) is equal to a power of 2, then we can consider computing each of the \( N/2 \)-point DFTs in Eq. (6.11) by breaking each of the sums in Eq. (6.11) into two \( N/4 \)-point DFTs, which would then be combined to yield the \( N/2 \)-point DFTs. Thus \( G(k) \) and \( H(k) \) in Eq. (6.11) would be computed as indicated below:

\[
G(k) = \sum_{r=0}^{(N/2)-1} g(r)W_{N/2}^{kr} = \sum_{l=0}^{(N/4)-1} g(2l)W_{N/4}^{2kl} + \sum_{l=0}^{(N/4)-1} g(2l + 1)W_{N/4}^{2(2l+1)k} \\
\text{or} \\
G(k) = \sum_{l=0}^{(N/4)-1} g(2l)W_{N/4}^{2kl} + W_{N/4}^k \sum_{l=0}^{(N/4)-1} g(2l + 1)W_{N/4}^{2(2l+1)k} 
\]

(6.12)

Similarly,

\[
H(k) = \sum_{l=0}^{(N/4)-1} h(2l)W_{N/4}^{kl} + W_{N/4}^k \sum_{l=0}^{(N/4)-1} h(2l + 1)W_{N/4}^{2(2l+1)k} 
\]

(6.13)

Thus if the four-point DFTs in Fig. 6.3 are computed according to Eqs. (6.12) and (6.13), then that computation would be carried out as indicated in Fig. 6.4. Inserting the computation indicated in Fig. 6.4 into the flow graph of Fig. 6.3, we obtain the complete flow graph of Fig. 6.5. Note that we have used the fact that \( W_{N/2}^{N/2} = W_N^k \).

For the eight-point DFT that we have been using as an illustration, the computation has been reduced to a computation of two-point DFTs. The two-point DFT of, for example, \( x(0) \) and \( x(4) \), is depicted in Fig. 6.6. With the computation of Fig. 6.6 inserted in the flow graph of Fig. 6.5, we obtain...
Fig. 6.4 Flow graph of the decimation-in-time decomposition of an $N/2$-point DFT computation into two $N/4$-point DFT computations ($N = 8$).

Fig. 6.5 Result of substituting Fig. 6.4 into Fig. 6.3.

Fig. 6.6 Flow graph of a two-point DFT.

$W_z = W_{N/2}^{N/2} = -1$
For the more general case with $N$ a power of 2 greater than 3, we would proceed by decomposing the $N/4$-point transforms in Eqs. (6.12) and (6.13) into $N/8$-point transforms, and continue until left with only two-point transforms. This requires $v$ stages of computation, where $v = \log_2 N$. Previously we found that in the original decomposition of an $N$-point transform into two $N/2$-point transforms, the number of complex multiplications and additions required was $N + 2(N/2)^2$. When the $N/2$-point transforms are decomposed into $N/4$-point transforms, then the factor of $(N/2)^2$ is replaced by $N/2 + 2(N/4)^2$, so the overall computation then requires $N + N + 4(N/4)^2$ complex multiplications and additions. If $N = 2^v$, this can be done at most $v = \log_2 N$ times, so that after carrying out this decomposition as many times as possible the number of complex multiplications and additions is equal to $N \log_2 N$.

The flow graph of Fig. 6.7 displays the operations explicitly. By counting branches with transmittances of the form $W_N^k$, we note that each stage has $N$ complex multiplications and $N$ complex additions. Since there are $\log_2 N$ stages, we have, as before, a total of $N \log_2 N$ complex multiplications and additions. This is the substantial computational savings that we have previously indicated was possible. We shall see that the symmetry and periodicity of $W_N^k$ can be exploited to obtain further reductions in computation.

### 6.2.1 In-Place Computations

The computational flow graph of Fig. 6.7 describes an algorithm for the computation of the discrete Fourier transform. What is important in the