Discrete Fourier Transform (DFT)

LECTURE 3

DR SANIKA WIJAYASEKARA

Discrete Fourier Transform (DFT)

- The DFT is the only kind of Fourier transform that can be evaluated on a computer.
- Since a computer can only process a finite amount of data, we must be able to represent every time signal and transform as a finite array

  - Another way to look at this problem is to consider the signals x [n] which can be expressed as a finite array of numbers.
  - We say x [n] is a finite-duration signal if there exists $N < \infty$ such that x [n] = 0 for $|n| > N$: Otherwise, x [n] is in finite-duration.

DFT of x[n]

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j\Omega_0 kn}, \quad 0 \le k \le N-1,$$

fundamental frequency. $\Omega_0 = \frac{2\pi}{N}$

Properties:

Linearity:

$$\begin{cases} \alpha x[n] \longleftrightarrow \alpha X[k] \\ x_1[n] + x_2[n] \longleftrightarrow X_1[k] + X_2[k] \end{cases}$$

Time Shift:

$$x[((n-n_0))_N] \longleftrightarrow e^{-j\Omega_0 n_0 k} X[k]$$

Frequency Shift:

$$e^{jin} x[n] \longleftrightarrow X[((k-i))_N]$$

Convolution:

$$x_1[n] * x_2[n] = \sum_{m=0}^{N-1} x_1[((n-m))_N]\, x_2[m]$$

$$x_1[n] * x_2[n] \longleftrightarrow X_1[k]\, X_2[k]$$

Multiplication:

$$X_1[k] * X_2[k] = \sum_{i=0}^{N-1} X_1[((k-i))_N]\, X_2[i]$$

$$x_1[n]\, x_2[n] \longleftrightarrow \frac{1}{N} X_1[k] * X_2[k]$$

Time Differencing:

$$x[n] - x[((n-1))_N] \longleftrightarrow \left(1 - e^{-j\Omega_0 k}\right) X[k]$$

Accumulation:

$$\sum_{m=0}^{n} x[m] \longleftrightarrow \frac{1}{1 - e^{-j\Omega_0 k}} X[k] \quad (\text{only for } X[0] = 0)$$

Properties:

Frequency Differencing:

$$\left(1 - e^{j\Omega_0 n}\right) x[n] \longleftrightarrow X[k] - X[((k-1))_N]$$

Conjugation:

$$x^*[n] \longleftrightarrow X^*[((-k))_N]$$

Reflection:

$$x[((-n))_N] \longleftrightarrow X[((-k))_N]$$

Real Time Signal

$$x[n] \text{ real} \iff \begin{cases} |X[((k))_N]| & \text{even} \\ \angle X[((k))_N] & \text{odd} \end{cases}$$

Even-Odd:

$$\begin{cases} x[((n))_N] \text{ even} \iff X[k] \text{ real} \\ x[((n))_N] \text{ odd} \iff X[k] \text{ imaginary} \end{cases}$$

Parseval's Theorem:

$$\frac{1}{N}\sum_{n=0}^{N-1} x_1[n]\, x_2^*[n] = \sum_{k=0}^{N-1} X_1[k]\, X_2^*[k]$$

Circular Convolution of x_1 [n] and x_2 [n]

$$x_1[n] * x_2[n] = \sum_{m=0}^{N-1} x_1[((n-m))_N]\, x_2[m].$$

Computational Complexity of DFT

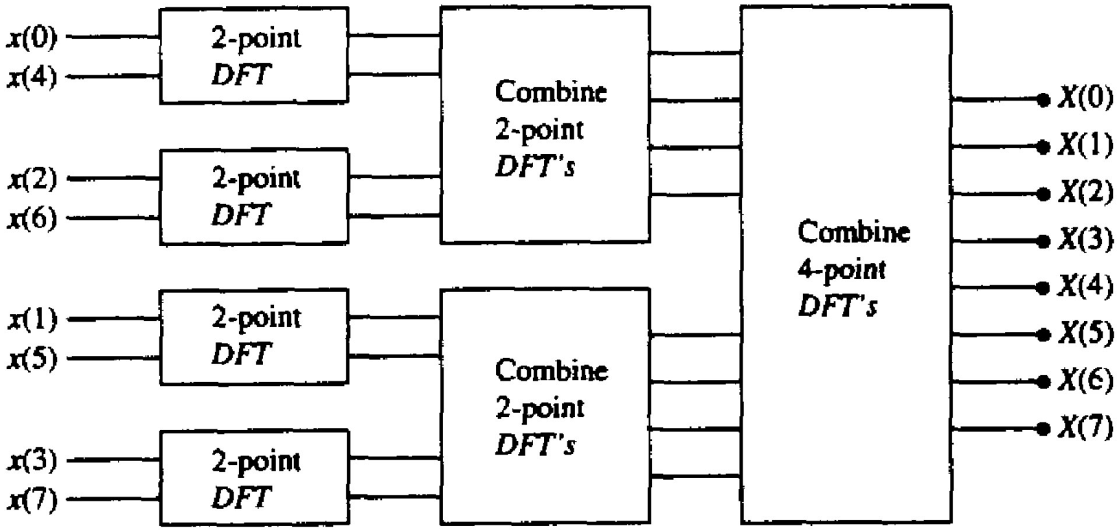
$$x = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}, \quad e = \begin{bmatrix} 1 \\ e^{-j\Omega_0 k} \\ \vdots \\ e^{-j\Omega_0 k(N-1)} \end{bmatrix}$$

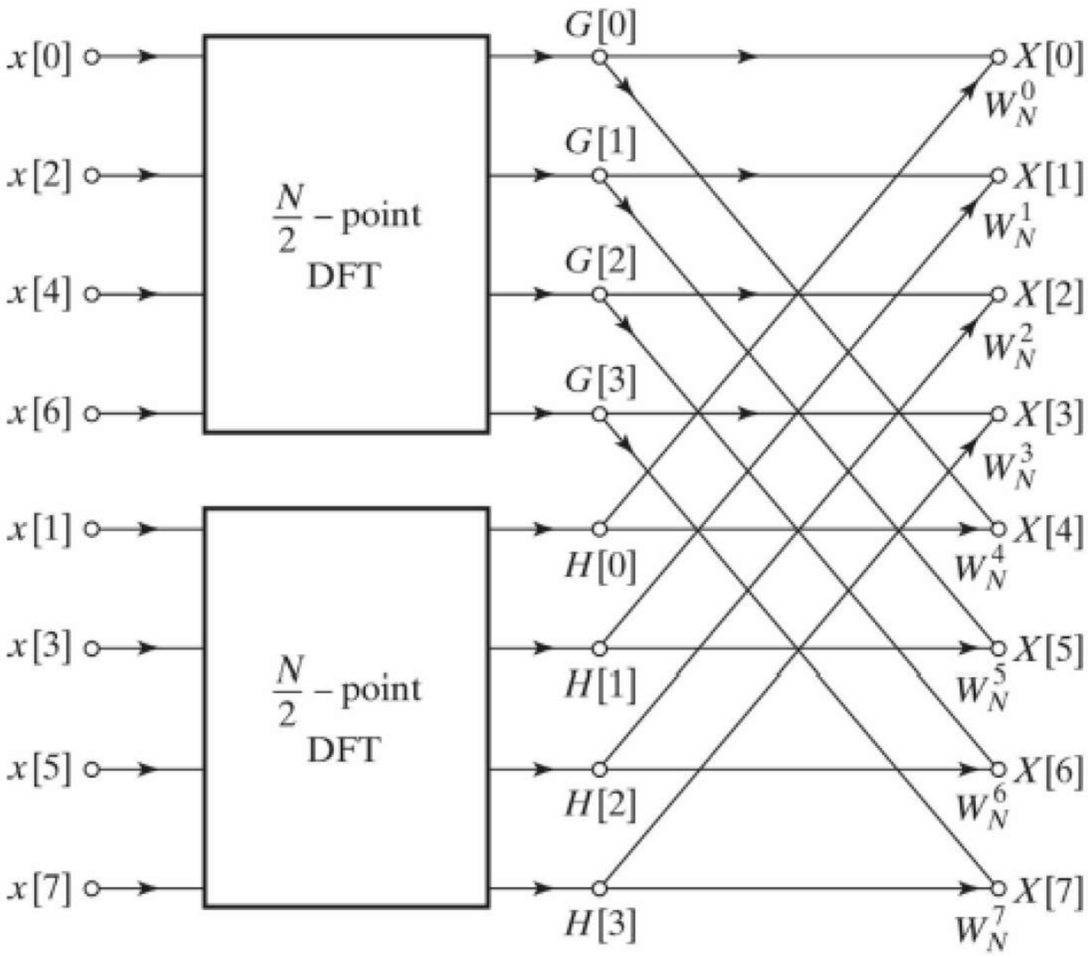
Fast Fourier Transform Algorithms (FFT)

Fast Fourier Transform is an efficient algorithm developed by Cooley & Tukey in 1965, used to compute the DFT with reduced computations.

Due to the efficiency of FFT, it is used for spectrum analysis, convolutions, correlations and linear filtering.

Decimation – In – Time (DIT) FFT algorithm





Butterfly diagram for DIT FFT algorithm for N =8