Fast Fourier Transform (FFT)

Borbala Hunyadi

Delft University of Technology, The Netherlands

Recap: Discrete Fourier Transform

Definition

The Discrete Fourier Transform (DFT) of a sequence $x[n]$ is

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{kn}{N}}, \text{ for } 0 \leq k \leq N-1$$

- Circular convolution of 2 sequences corresponds to the multiplication of their DFTs
- Linear convolution can be implemented using zero-padding and circular convolution
- Filtering can be achieved using DFT

Recap: Discrete Fourier Transform

Definition

The Discrete Fourier Transform (DFT) of a sequence $x[n]$ is

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{kn}{N}}, \text{ for } 0 \leq k \leq N-1$$

- Circular convolution of 2 sequences corresponds to the multiplication of their DFTs
- Linear convolution can be implemented using zero-padding and circular convolution
- Filtering can be achieved using DFT
- Is it worth the trouble?

Computational complexity of the DFT

Let's define $W_N = e^{-j2\pi/N}$! Then the DFT can be expressed as:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

Steps of the direct computation algorithm:

Stage 1:

Compute and store the values

$W_N^l = e^{-j2\pi l/N} = cos(2\pi l/N) - j \cdot sin(2\pi l/N)$

Stage 2:

for $k = 0 : N-1$

  $X[k] \leftarrow x[0]$

  for $n = 1 : N-1$

    $l = (kn)_N$

    $X[k] \leftarrow X[k] + x[n] W_N^l$

  end

end

Example: 4-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1, \text{ with } W_N = e^{-j2\pi/N}$$

$$X[k] = \sum_{n=0}^{3} x[n] W_4^{kn}, \text{ for } 0 \leq k \leq 3, \text{ with } W_4 = e^{-j2\pi/4} = e^{-j\pi/2}$$

Matrix notation:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^{0\cdot0} & W_4^{0\cdot1} & W_4^{0\cdot2} & W_4^{0\cdot3} \\ W_4^{1\cdot0} & W_4^{1\cdot1} & W_4^{1\cdot2} & W_4^{1\cdot3} \\ W_4^{2\cdot0} & W_4^{2\cdot1} & W_4^{2\cdot2} & W_4^{2\cdot3} \\ W_4^{3\cdot0} & W_4^{3\cdot1} & W_4^{3\cdot2} & W_4^{3\cdot3} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

Example: 4-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1, \text{ with } W_N = e^{-j2\pi/N}$$

$$X[k] = \sum_{n=0}^{3} x[n] W_4^{kn}, \text{ for } 0 \leq k \leq 3, \text{ with } W_4 = e^{-j2\pi/4} = e^{-j\pi/2}$$

Matrix notation:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

Example: 4-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \le k \le N-1, \text{ with } W_N = e^{-j2\pi/N}$$

$$X[k] = \sum_{n=0}^{3} x[n] W_4^{kn}, \text{ for } 0 \le k \le 3, \text{ with } W_4 = e^{-j2\pi/4} = e^{-j\pi/2}$$

Matrix notation:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

periodicity: $W_N^{l+N} = W_N^l$

Example: 4-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1, \text{ with } W_N = e^{-j2\pi/N}$$

$$X[k] = \sum_{n=0}^{3} x[n] W_4^{kn}, \text{ for } 0 \leq k \leq 3, \text{ with } W_4 = e^{-j2\pi/4} = e^{-j\pi/2}$$

Matrix notation:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^0 & W_4^2 \\ W_4^0 & W_4^3 & W_4^2 & W_4^1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

Example: 4-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1, \text{ with } W_N = e^{-j2\pi/N}$$

$$X[k] = \sum_{n=0}^{3} x[n] W_4^{kn}, \text{ for } 0 \leq k \leq 3, \text{ with } W_4 = e^{-j2\pi/4} = e^{-j\pi/2}$$

Matrix notation:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^0 & W_4^2 \\ W_4^0 & W_4^3 & W_4^2 & W_4^1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$W_4^0 = e^{-2\pi \cdot 0/4} = 1$$
$$W_4^1 = e^{-2\pi \cdot 1/4} = -j$$
$$W_4^2 = e^{-2\pi \cdot 2/4} = -1$$
$$W_4^3 = e^{-2\pi \cdot 3/4} = j$$

Example: 4-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1, \text{ with } W_N = e^{-j2\pi/N}$$

$$X[k] = \sum_{n=0}^{3} x[n] W_4^{kn}, \text{ for } 0 \leq k \leq 3, \text{ with } W_4 = e^{-j2\pi/4} = e^{-j\pi/2}$$

Matrix notation:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

Computational complexity of the DFT

Let's define $W_N = e^{-j2\pi/N}$! Then the DFT can be expressed as:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

Steps of the direct computation algorithm:

Stage 1:

Compute and store the values

$W_N^l = e^{-j2\pi l/N} = cos(2\pi l/N) - j \cdot sin(2\pi l/N)$

N evaluations of sin and cos functions

Stage 2:

for $k = 0 : N-1$

  $X[k] \leftarrow x[0]$

  for $n = 1 : N-1$

    $l = (kn)_N$

    $X[k] \leftarrow X[k] + x[n] W_N^l$

  end

end

Computational complexity of the DFT

Let's define $W_N = e^{-j2\pi/N}$! Then the DFT can be expressed as:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

Steps of the direct computation algorithm:

Stage 1:

Compute and store the values

$W_N^l = e^{-j2\pi l/N} = cos(2\pi l/N) - j \cdot sin(2\pi l/N)$

N evaluations of sin and cos functions

Stage 2:

for $k = 0 : N-1$

  $X[k] \leftarrow x[0]$

  for $n = 1 : N-1$

    $l = (kn)_N$

    $X[k] \leftarrow X[k] + x[n] W_N^l$

  end

end

N^2 complex multiplications and $N(N-1)$ complex additions

Computational complexity of the DFT

Let's define $W_N = e^{-j2\pi/N}$! Then the DFT can be expressed as:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

Steps of the direct computation algorithm:

Stage 1:

Compute and store the values

$W_N^l = e^{-j2\pi l/N} = cos(2\pi l/N) - j \cdot sin(2\pi l/N)$

N evaluations of sin and cos functions

Stage 2:

for $k = 0 : N-1$

  $X[k] \leftarrow x[0]$

  for $n = 1 : N-1$

    $l = (kn)_N$

    $X[k] \leftarrow X[k] + x[n] W_N^l$

  end

end

N^2 complex multiplications and $N(N-1)$ complex additions

\+ overhead: addressing, indexing...

Computational complexity of the DFT

Let's define $W_N = e^{-j2\pi/N}$! Then the DFT can be expressed as:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

Steps of the direct computation algorithm:

$O(N^2)$ - **very costly**

Stage 1:

Compute and store the values

$W_N^l = e^{-j2\pi l/N} = cos(2\pi l/N) - j \cdot sin(2\pi l/N)$

N evaluations of sin and cos functions

Stage 2:

for $k = 0 : N-1$
  $X[k] \leftarrow x[0]$
  for $n = 1 : N-1$
    $l = (kn)_N$
    $X[k] \leftarrow X[k] + x[n] W_N^l$
  end
end

N^2 complex multiplications and $N(N-1)$ complex additions

\+ overhead: addressing, indexing...

Fast Fourier Transform

- A family of computationally efficient algorithms to compute DFT
- Not a new transform!

Different working principles:

1. Divide and conquer approach
2. DFT as convolution: linear filtering approach
   - Goertzel algorithm
   - Chirp z-transform

Fast Fourier Transform

- A family of computationally efficient algorithms to compute DFT
- Not a new transform!

Different working principles:

1. **Divide and conquer approach**
2. DFT as convolution: linear filtering approach
   - Goertzel algorithm
   - Chirp z-transform

Divide and conquer FFT

Essential ingredients:

- Break down the N-point DFT to a cascade of smaller-size DFTs
- Exploit symmetries

Divide and conquer FFT

Essential ingredients:

- **Break down the N-point DFT to a cascade of smaller-size DFTs**
- Exploit symmetries

Guessing game: I am thinking of a random number between 1 and 16. Can you guess which number is it?

Divide and conquer FFT

Essential ingredients:

- Break down the N-point DFT to a cascade of smaller-size DFTs
- **Exploit symmetries**

$$W_N^k = W_N^{k+N}$$
$$W_N^{LK} = W_{N/L}^k$$

Radix-2 FFT

Radix-2 FFT can be used if $N = 2^r$. This can always be achieved using zero-padding.

Decimation in time (DIT) solution:

- Divide the N long sequence $x[n]$ to 2 N/2 long sequences
- The N-point DFT of $x[n]$ can be computed by properly combining the 2 N/2-point DFTs
- Repeat the subdivision until the sequences are 2 samples long (2-point DFT)

Radix-2 FFT

N-point DFT ($N = 2^r$) solved by a cascade of r stages:

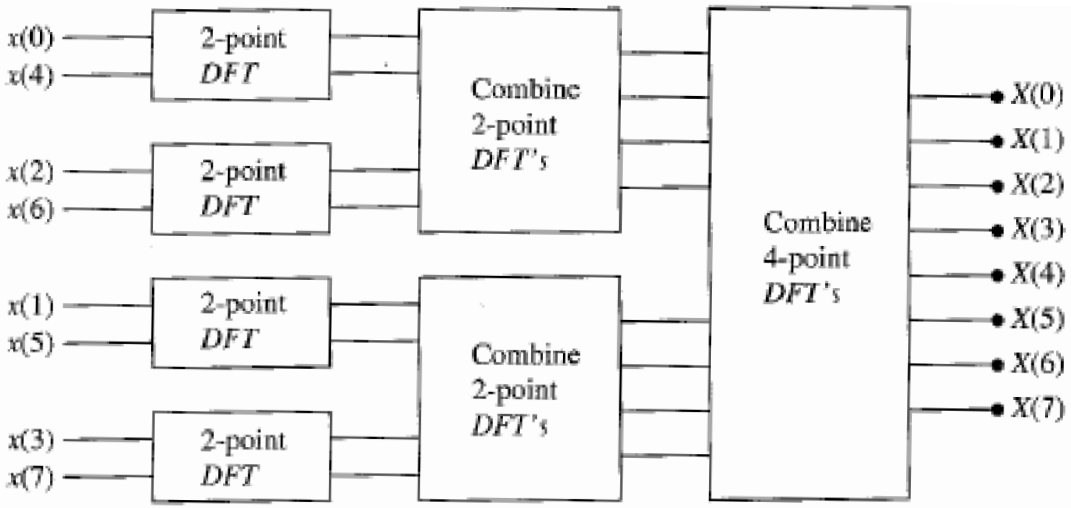


Figure 8.1.5 Three stages in the computation of an $N = 8$-point DFT.

2-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

N=2:

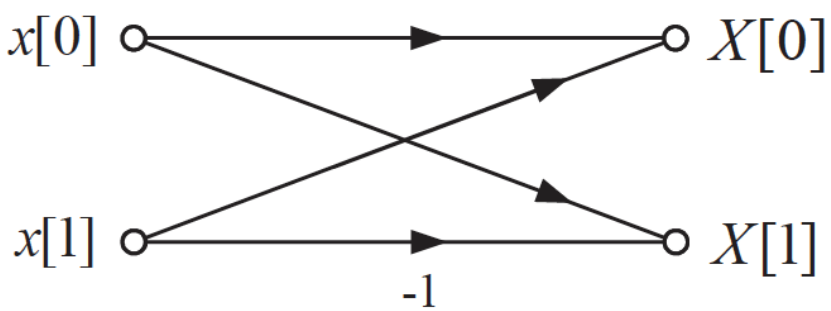
$$X[0] = x[0] + x[1] W_N^0 = x[0] + x[1]$$
$$X[1] = x[0] + x[1] W_N^1 = x[0] + x[1] e^{(-j2\pi/2)\cdot 1} = x[0] - x[1]$$

2-point DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

N=2:

$$X[0] = x[0] + x[1] W_N^0 = x[0] + x[1]$$
$$X[1] = x[0] + x[1] W_N^1 = x[0] + x[1] e^{(-j2\pi/2)\cdot 1} = x[0] - x[1]$$



4-point DFT from 2-point DFTs

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

$$X[k] = x[0] + x[1] W_N^k + x[2] W_N^{2k} + x[3] W_N^{3k}$$

4-point DFT from 2-point DFTs

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

$$\begin{aligned} X[k] &= x[0] + x[1]W_N^k + x[2]W_N^{2k} + x[3]W_N^{3k} \\ &= (x[0] + x[2]W_4^{2k}) + (x[1]W_4^k + x[3]W_4^{3k}) \end{aligned}$$

Decimation in time: divide the sum to a sum of even and a sum of odd samples

4-point DFT from 2-point DFTs

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

$$\begin{aligned} X[k] &= x[0] + x[1]W_N^k + x[2]W_N^{2k} + x[3]W_N^{3k} \\ &= (x[0] + x[2]W_4^{2k}) + (x[1]W_4^k + x[3]W_4^{3k}) \\ &= x[0] + x[2]W_4^{2k} + W_4^k(x[1] + x[3]W_4^{2k}) \end{aligned}$$

4-point DFT from 2-point DFTs

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

$$\begin{aligned} X[k] &= x[0] + x[1]W_N^k + x[2]W_N^{2k} + x[3]W_N^{3k} \\ &= (x[0] + x[2]W_4^{2k}) + (x[1]W_4^k + x[3]W_4^{3k}) \\ &= x[0] + x[2]W_4^{2k} + W_4^k(x[1] + x[3]W_4^{2k}) \\ &= x[0] + x[2]W_2^k + W_4^k(x[1] + x[3]W_2^k) \end{aligned}$$

using the property $W_N^{LK} = W_{N/L}^k$ $N = 4$ and $L = 2$

4-point DFT from 2-point DFTs

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \text{ for } 0 \leq k \leq N-1$$

$$\begin{aligned} X[k] &= x[0] + x[1]W_N^k + x[2]W_N^{2k} + x[3]W_N^{3k} \\ &= (x[0] + x[2]W_4^{2k}) + (x[1]W_4^k + x[3]W_4^{3k}) \\ &= x[0] + x[2]W_4^{2k} + W_4^k(x[1] + x[3]W_4^{2k}) \\ &= x[0] + x[2]W_2^k + W_4^k(x[1] + x[3]W_2^k) = G[k] + W_4^k H[k] \end{aligned}$$

$G[k] \equiv x[0] + x[2]W_2^k$ is the 2-point DFT of even samples

$H[k] \equiv x[1] + x[3]W_2^k$ is the 2-point DFT of odd samples

4-point DFT from 2-point DFTs

$$X[k] = G[k] + W_4^k H[k]$$

$$X[0] = G[0] + H[0]$$
$$X[1] = G[1] + W_4 H[1]$$
$$X[2] = G[2] + W_4^2 H[2]$$
$$X[3] = G[3] + W_4^3 H[3]$$

4-point DFT from 2-point DFTs

$$X[k] = G[k] + W_4^k H[k]$$

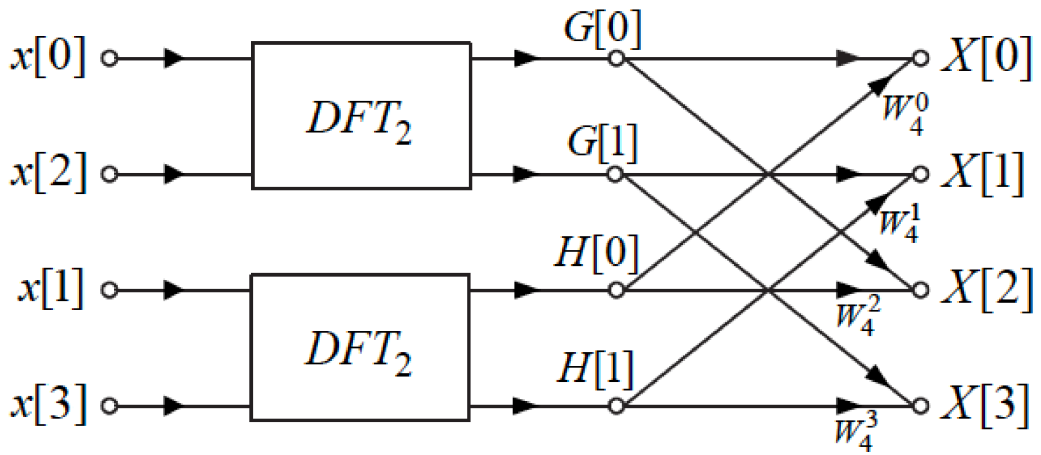
$$X[0] = G[0] + H[0]$$
$$X[1] = G[1] + W_4 H[1]$$
$$X[2] = G[2] + W_4^2 H[2] = G[0] + W_4^2 H[0]$$
$$X[3] = G[3] + W_4^3 H[3] = G[1] + W_4^3 H[1]$$

$G[k]$ and $H[k]$ are 2-point DFTs, hence, 2-periodic

4-point DFT from 2-point DFTs

$$X[k] = G[k] + W_4^k H[k]$$

$$X[0] = G[0] + H[0]$$
$$X[1] = G[1] + W_4 H[1]$$
$$X[2] = G[2] + W_4^2 H[2] = G[0] + W_4^2 H[0]$$
$$X[3] = G[3] + W_4^3 H[3] = G[1] + W_4^3 H[1]$$



General case

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}$$

General case

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}$$

$$= \sum_{r=0}^{N/2-1} x[2r] W_N^{2kr} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_N^{2kr}$$

Decimation in time: divide the sum to a sum of even and a sum of odd samples

General case

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}$$

$$= \sum_{r=0}^{N/2-1} x[2r] W_N^{2kr} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_N^{2kr}$$

$$= \sum_{r=0}^{N/2-1} x[2r] W_{N/2}^{kr} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_{N/2}^{kr}$$

using the property $W_N^{LK} = W_{N/L}^k$

General case

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}$$

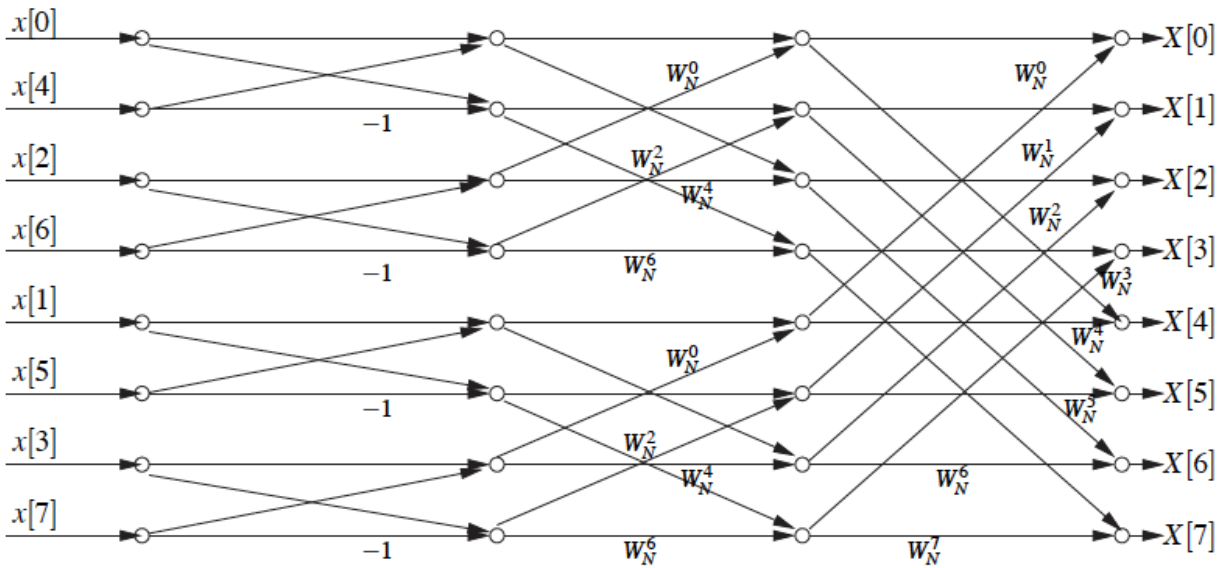
$$= \sum_{r=0}^{N/2-1} x[2r] W_N^{2kr} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_N^{2kr}$$

$$= \sum_{r=0}^{N/2-1} x[2r] W_{N/2}^{kr} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_{N/2}^{kr} = G[k] + W_N^k H[k]$$

$G[k]$ is the N/2-point DFT of even samples, hence N/2-periodic

$H[k]$ is the N/2-point DFT of odd samples, hence N/2-periodic

Example: 8-point FFT



Start with 2-point DFTs of properly arranged samples! Butterfly can be further simplfied with $W_N^{k+N/2} = -W_N^k$

Computational complexity of Radix-2 FFT

- $v = log_2 N$ stages
- per stage, there are N/2 butterflies
- per butterly, 1 complex multiplication and 2 complex additions

Total: $log_2 N \cdot N/2$ complex multiplications and $log_2 N \cdot N$ complex additions, i.e. $O(N \log_2 N)$.