Chapter 9
Computation of the DFT

Main Topics

1. Decimation-in-time FFT algorithm
2. Decimation-in-frequency FFT algorithm

Introduction

- DFT 是信号分析与处理中的一种重要变换。因直接计算 DFT 的计算量与变换区间长度 N 的平方成正比，当 N 较大时，计算量太大，所以在快速傅里叶变换（简称 FFT) 出现以前，直接用 DFT 算法进行谱分析和信号的实时处理是不切实际的。直到 1965 年发现了 DFT 的一种快速算法以后，情况才发生了根本的变化。

Direct computation

DFT of the sequence $\boldsymbol{x[n]}$ with the length $\boldsymbol{N}$

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

In matrix form：

$$\begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix} = \begin{bmatrix} W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^1 & \cdots & W_N^{N-1} \\ & & \vdots & \\ W_N^0 & W_N^{N-1} & \cdots & W_N^{(N-1)^2} \end{bmatrix} \bullet \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}$$

- For complex sequence $\boldsymbol{x[n]}$ ：

	Complex multiplication	Complex addition
One $X[k]$	N	N-1
N $X[k]$	N^2	$N(N$-1)

- 直接计算 DFT 的计算量与变换区间长度 N 的平方成正比。
  直接用 DFT 算法进行谱分析和信号的实时处理是不切实际的。
- 1965 年， J.W. Tukey 和 T.W. Coody 在《 Math. Computation 》杂志 Vol. 19 上发表著名的《机器计算傅里叶级数的一种算法》的论文。

$$DFT \xrightarrow{Fast\ Algorithm} FFT$$

- 可提高效率 1-2 个数量级！！

Approaches to improving the efficiency of computation of DFT

The symmetry and periodicity property of W_N^m

Periodicity
$$W_N^{m+lN} = e^{-j\frac{2\pi}{N}(m+lN)} = e^{-j\frac{2\pi}{N}m} = W_N^m$$

Symmetry
$$W_N^{-m} = W_N^{N-m} \quad \text{or} \quad [W_N^{N-m}]^* = W_N^m$$

$$W_N^{m+\frac{N}{2}} = -W_N^m$$

把 N 点 DFT 分解为几个较短的 DFT，可使乘法次数大大减少。

1 Decimation-in-time FFT algorithm (DIT-FFT in short)

The length of $\boldsymbol{x[n]}$ is N , considering the special case:

$$N = 2^M, \quad M \text{ is integer.}$$

Separating $\boldsymbol{x[n]}$ into two $\boldsymbol{N/2}$-point sequence.

$$g[r] = x[2r], \qquad r = 0, 1, \cdots \frac{N}{2} - 1$$

$$h[r] = x[2r+1], \quad r = 0, 1, \cdots \frac{N}{2} - 1$$

***g*[0]=*x*[0]**	***g*[1]=*x*[2]**	***g*[2]=*x*[4]**	**…**	***g*[*N*/2-1]=*x*[*N*-2]**
***h*[0]=*x*[1]**	***h*[1]=*x*[3]**	***h*[2]=*x*[5]**	**….**	***h*[*N*/2-1]=*x*[*N*-1]**

- The DFT of $x[n]$:

$$X[k]=\sum_{n=0}^{N-1}x[n]W_N^{kn}=\sum_{n\text{为偶数}}x[n]W_N^{kn}+\sum_{n\text{为奇数}}x[n]W_N^{kn}$$

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

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

$$W_N^{2kr}=e^{-j\frac{2\pi}{N}2kr}=e^{-j\frac{2\pi}{\frac{N}{2}}kr}=W_{\frac{N}{2}}^{kr}$$

$$X[k]=\sum_{r=0}^{\frac{N}{2}-1}x[2r]W_{\frac{N}{2}}^{kr}+W_N^k\sum_{r=0}^{\frac{N}{2}-1}x[2r+1]W_{\frac{N}{2}}^{kr}\qquad k=0,1,\cdots,N-1$$

$$=G[k]+W_N^kH[k]\qquad(9.14)$$

$G[k]$ and $H[k]$ are the N/2-point DFT of $g[r]$ 和 $h[r]$.

$$G[k] = \sum_{r=0}^{N/2-1} g[r]W_{N/2}^{kr} = DFT[g[r]]$$

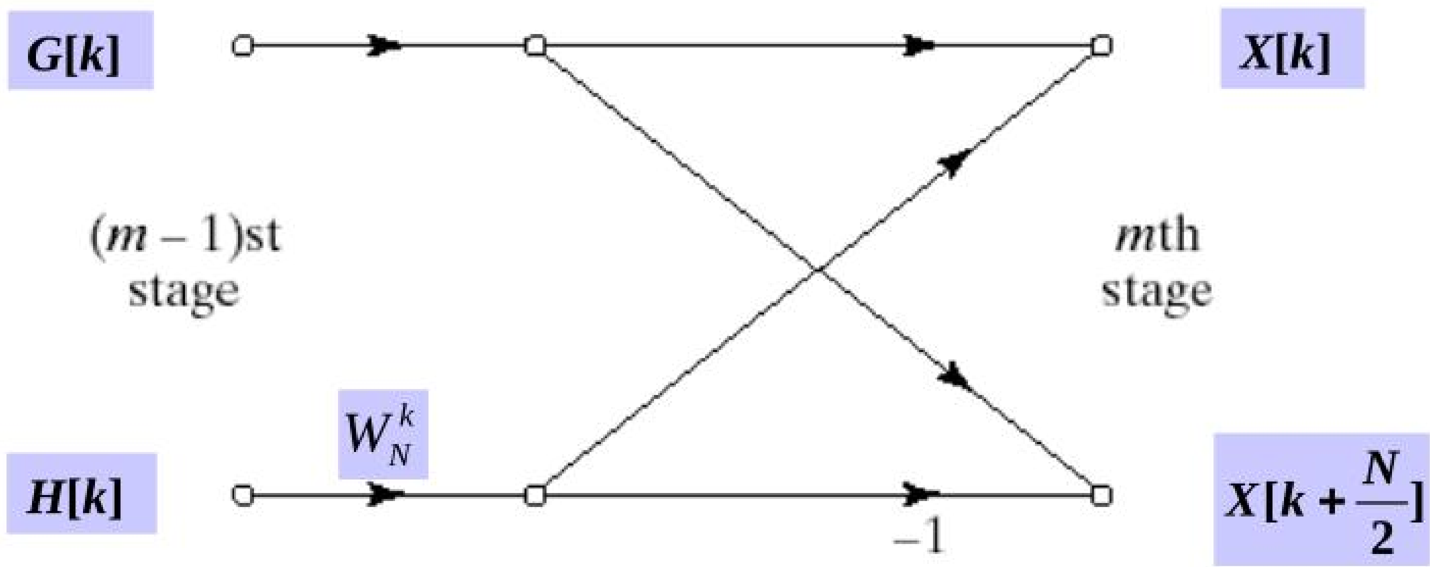
$$H[k] = \sum_{r=0}^{N/2-1} h[r]W_{N/2}^{kr} = DFT[h[r]]$$

***G*[*k*]** and ***H*[*k*]** has the period of ***N*/2** , $W_N^{k+\frac{N}{2}} = -W_N^k$

So

$$X[k] = G[k] + W_N^k H[k] \quad k = 0,1,\cdots\frac{N}{2}-1$$

$$X[k+\frac{N}{2}] = G[k] - W_N^k H[k] \quad k = 0,1,\cdots\frac{N}{2}-1$$

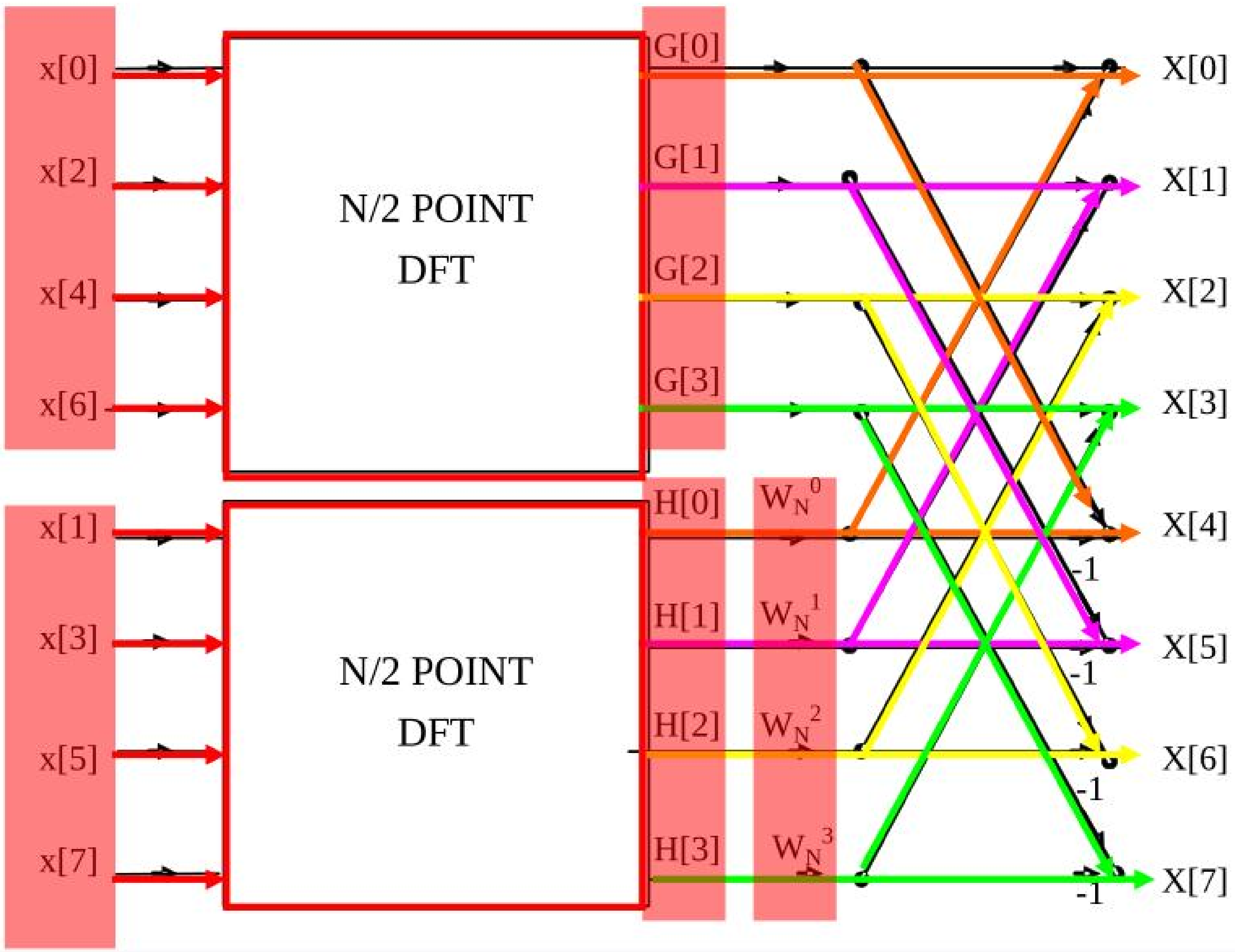


Butterfly computation

$$X[k] = G[k] + W_N^k H[k] \quad k = 0,1,\cdots \frac{N}{2} - 1$$

$$X[k+\frac{N}{2}] = G[k] - W_N^k H[k] \quad k = 0,1,\cdots \frac{N}{2} - 1$$

Decimation-in-time FFT Algorithms



- 与第一次分解相同，将 **$g[r]$** 按奇偶分解成两个 **$N/4$** 长的子序列 **$g_1[l]$** 和 **$g_2[l]$**，即

$$\left.\begin{aligned} g_1[l] &= g[2l] \\ g_2[l] &= g[2l+1] \end{aligned}\right\}, l = 0,1,\cdots,\frac{N}{4}-1$$

$g_1[0]=g[0]=x[0]$	**$g_1[1]=g[2]=x[4]$**	**…**
$g_2[0]=g[1]=x[2]$	**$g_2[1]=g[3]=x[6]$**	**….**

那么，**$G[k]$** 又可表示为

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

$$=\sum_{i=0}^{N/4-1} g[2l]W_{N/4}^{kl}+W_{N/2}^{k}\sum_{i=0}^{N/4-1} g[2l+1]W_{N/4}^{kl}$$

$$=G_1[k]+W_{N/2}^{k}G_2[k] \qquad\qquad k=0,1,\cdots N/2-1$$

式中

$$G_1[k] = \sum_{i=0}^{N/4-1} g_1[l] W_{N/4}^{kl} = DFT[g_1[l]]$$

$$G_2[k] = \sum_{i=0}^{N/4-1} g_2[l] W_{N/4}^{kl} = DFT[g_2[l]]$$

$N/4$ 点 DFT

由 **$G_1[k]$ 和 $G_2[k]$** 的周期性

$W_{N/2}^{k}$ 的对称性 $W_{N/2}^{k+N/4} = -W_{N/2}^{k}$

$$\left.\begin{aligned} G[k] &= G_1[k] + W_{N/2}^{k} G_2[k] \\ G(k+N/4) &= G_1[k] - W_{N/2}^{k} G_2[k] \end{aligned}\right\}, k = 0, 1, \cdots; N/4 - 1$$

- 用同样的方法可计算出

$$\left.\begin{aligned} H[k] &= H_1[k] + W_{N/2}^{k} H_2[k] \\ H(k+N/4) &= H_1[k] - W_{N/2}^{k} H_2[k] \end{aligned}\right\}, k = 0,1,\cdots N/4-1$$

其中

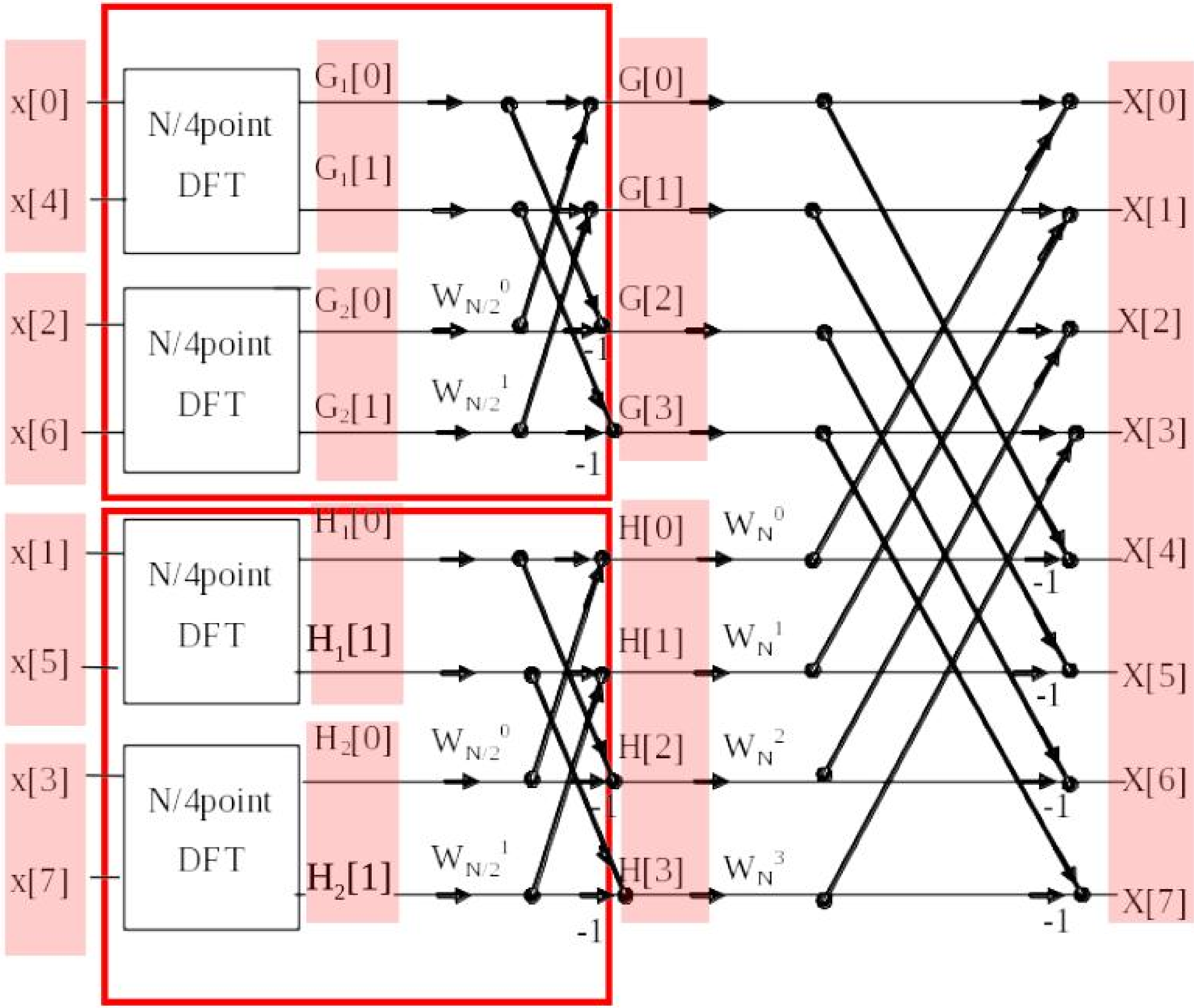
$$H_1[k] = \sum_{i=0}^{N/4-1} h_1[l] W_{N/4}^{kl} = DFT[h_1[l]]$$

$$H_2(k) = \sum_{i=0}^{N/4-1} h_2[l] W_{N/4}^{kl} = DFT[h_2[l]]$$

N/4 点 DFT

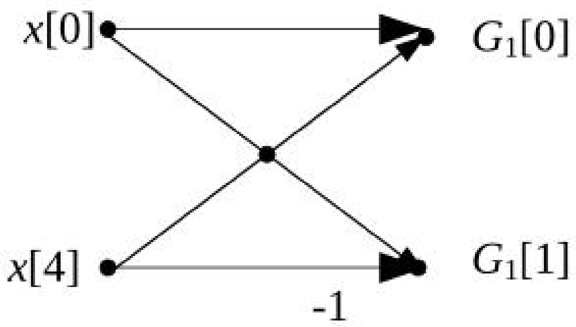
$$\left.\begin{aligned} h_1[l] &= h[2l] \\ h_2[l] &= h[2l+1] \end{aligned}\right\}, l = 0,1,\cdots N/4-1$$

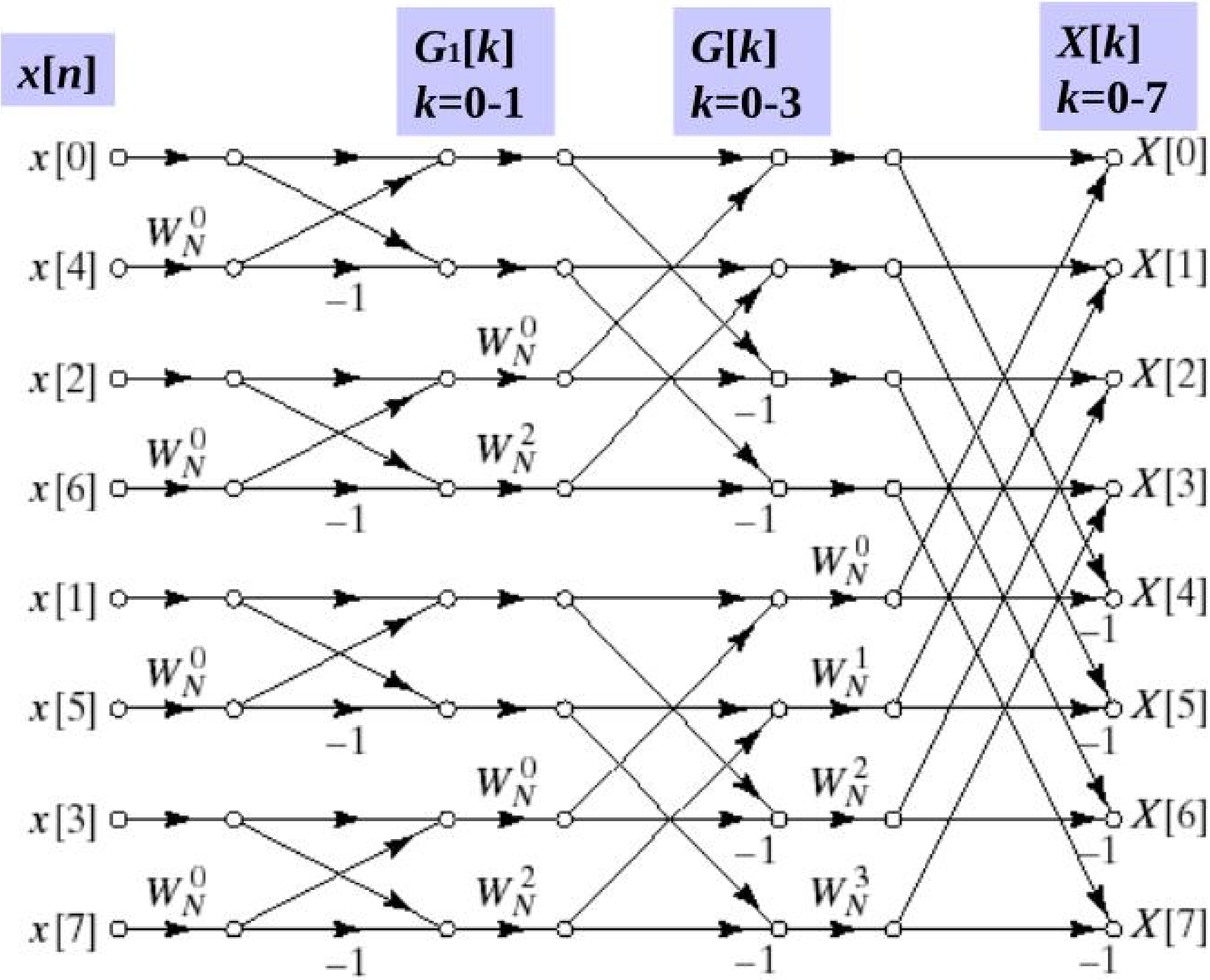
$h_1[0]=h[0]=x[1]$	$h_1[1]=h[2]=x[5]$	…
$h_2[0]=h[1]=x[3]$	$h_2[1]=h[3]=x[7]$	….


x[0]
x[4]
x[2]
x[6]
x[1]
x[5]
x[3]
x[7]
N/4point DFT
N/4point DFT
N/4point DFT
N/4point DFT
$G_1[0]$
$G_1[1]$
$G_2[0]$
$G_2[1]$
$H_1[0]$
$H_1[1]$
$H_2[0]$
$H_2[1]$
$W_{N/2}^0$
$W_{N/2}^1$
$W_{N/2}^0$
$W_{N/2}^1$
-1
G[0]
G[1]
G[2]
G[3]
H[0]
H[1]
H[2]
H[3]
W_N^0
W_N^1
W_N^2
W_N^3
X[0]
X[1]
X[2]
X[3]
X[4]
X[5]
X[6]
X[7]

- 重复分解过程，经过 M-1 次分解，最后分解成 N/2 个 2 点的 DFT：

$$G_1[0] = x[0] + x[4]$$
$$G_1[1] = x[0] - x[4]$$



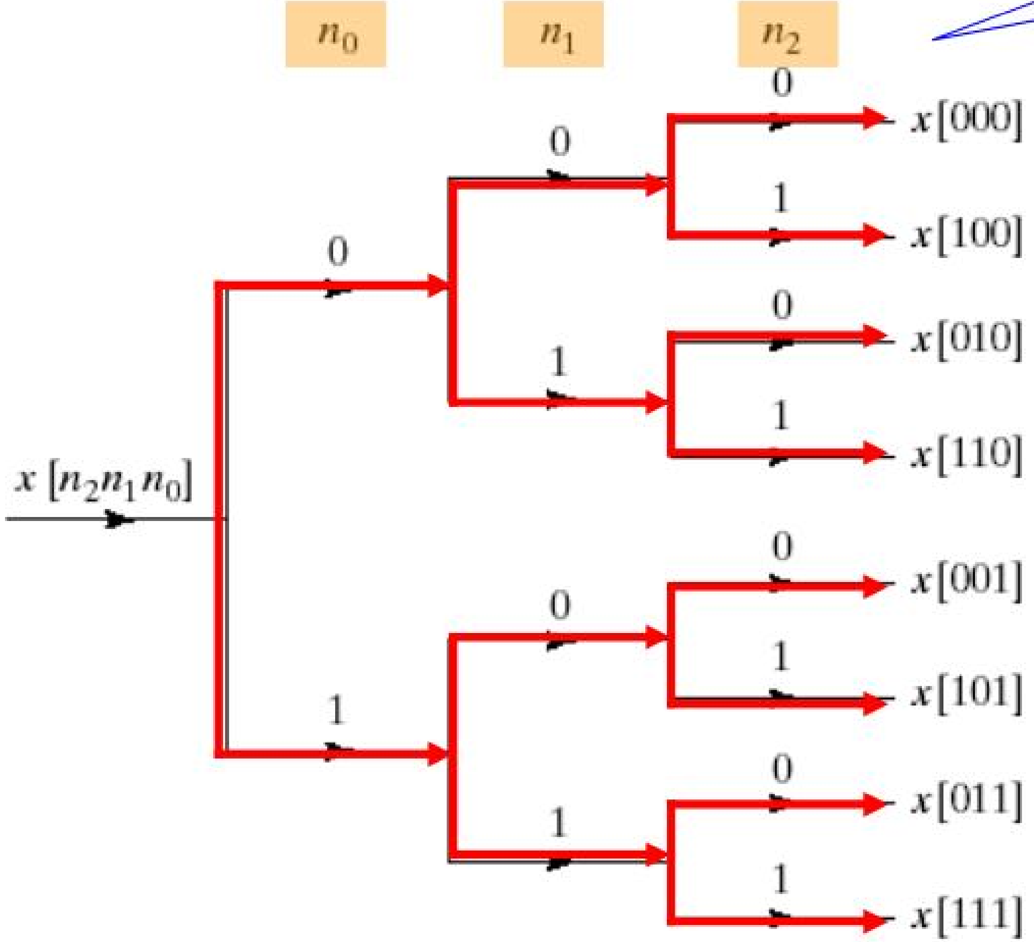


Strongpoint : in-place computations

Shortcoming : non-sequential access of data

Binary coding for position

cause of bit-reversed order



must padding 0 to

$$N = 2^M$$

DIT—FFT 算法的输入序列的倒序是很有规律的。

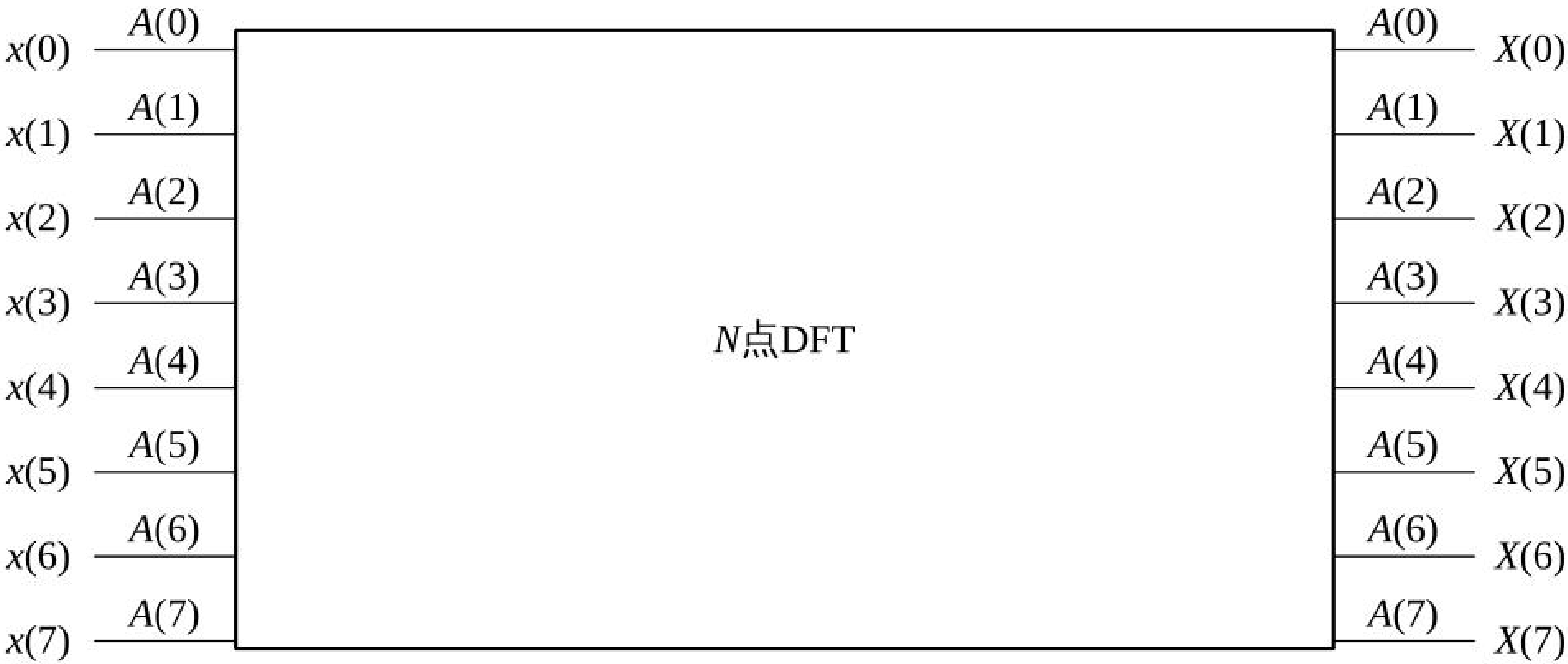
由于 $N=2^M$，所以顺序数可用 M 位二进制数 $(n^{M-1}n^{M-2}\ldots n^1n^0)$ 表示。

顺序和倒序二进制数对照表

顺序		倒序	
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

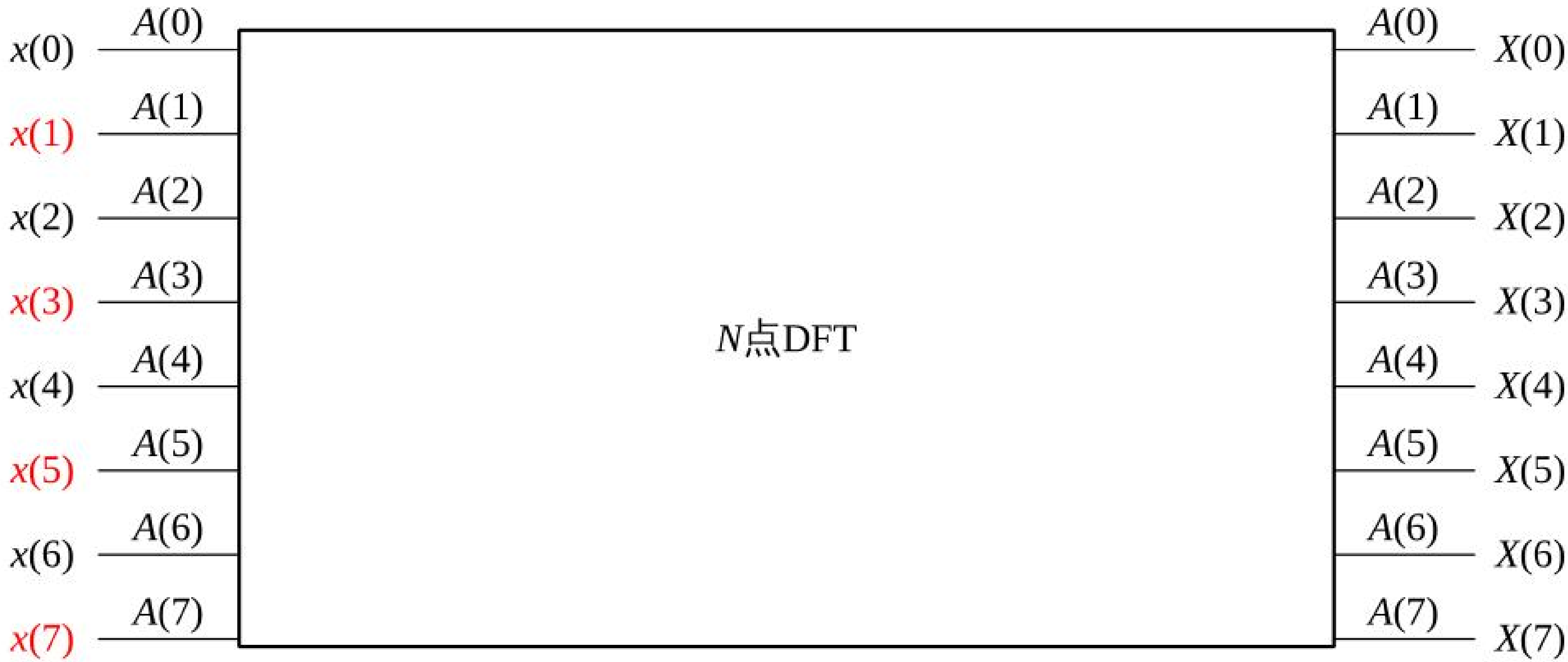
顺序		倒序	
0	0000	0000	0
1	0001	1000	8
2	0010	0100	4
3	0011	1100	12
4	0100	0010	2
5	0101	1010	10
6	0110	0110	6
7	0111	1110	14
8	1000	0001	1
9	1001	1001	9
10	1010	0101	5
11	1011	1101	13
12	1100	0011	3
13	1101	1011	11
14	1110	0111	7
15	1111	1111	15

FFT 的思路:

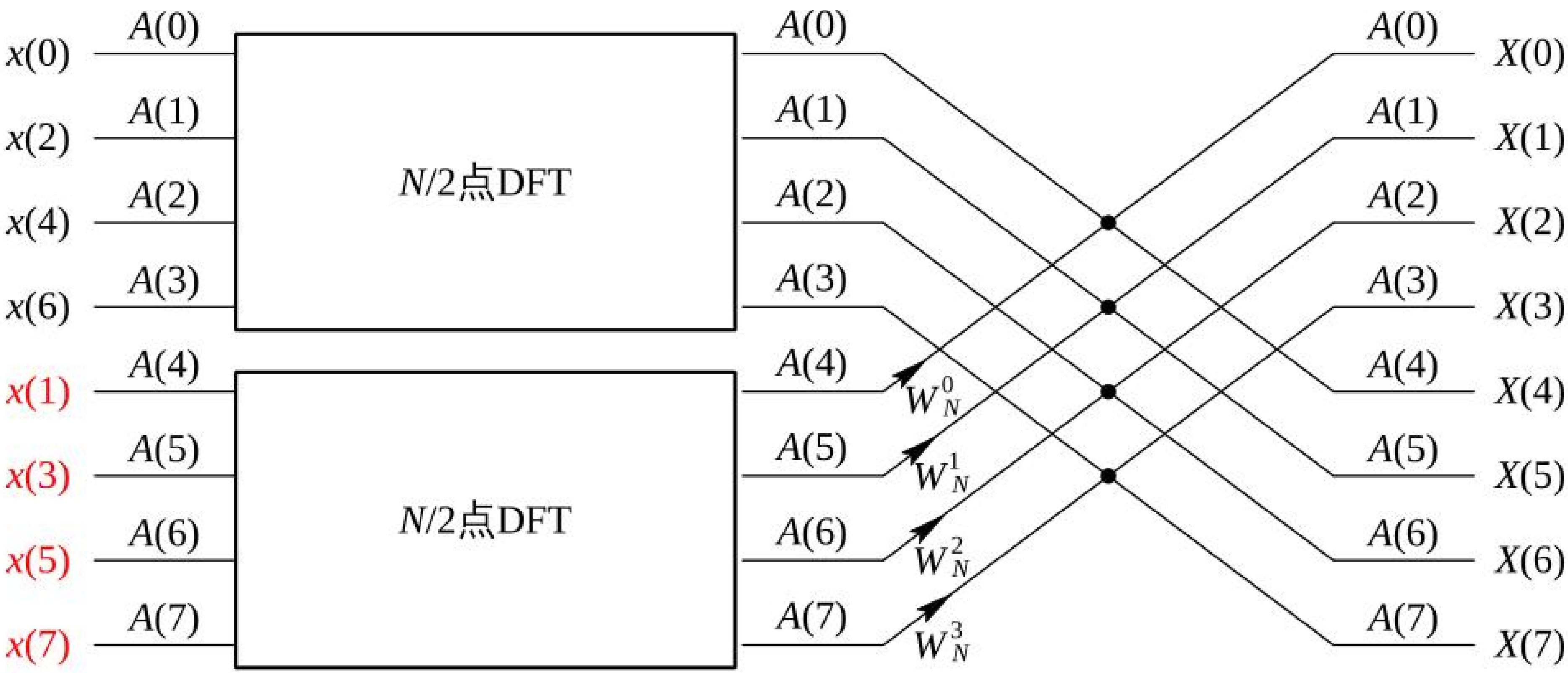


x(0)
x(1)
x(2)
x(3)
x(4)
x(5)
x(6)
x(7)
A(0)
A(1)
A(2)
A(3)
A(4)
A(5)
A(6)
A(7)
N点DFT
A(0)
A(1)
A(2)
A(3)
A(4)
A(5)
A(6)
A(7)
X(0)
X(1)
X(2)
X(3)
X(4)
X(5)
X(6)
X(7)

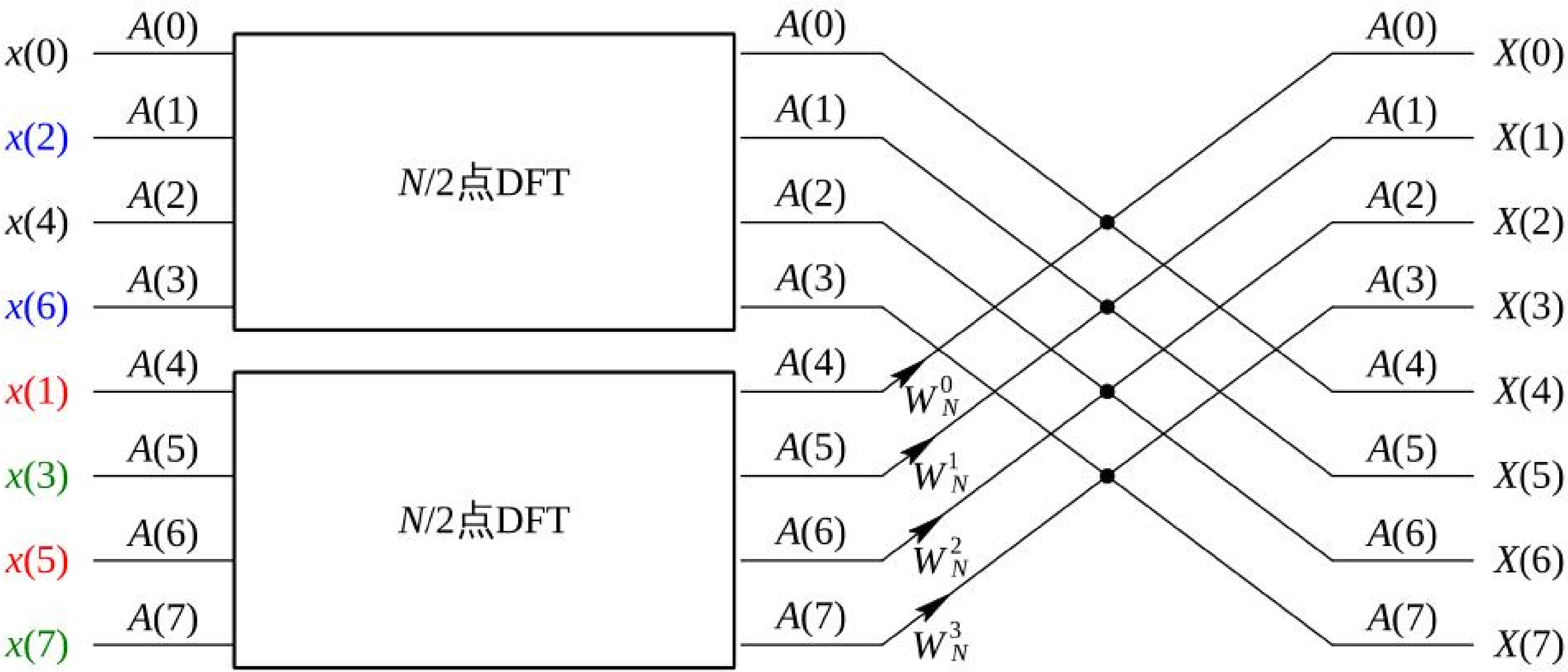
将时域序列按序号奇偶分成两组



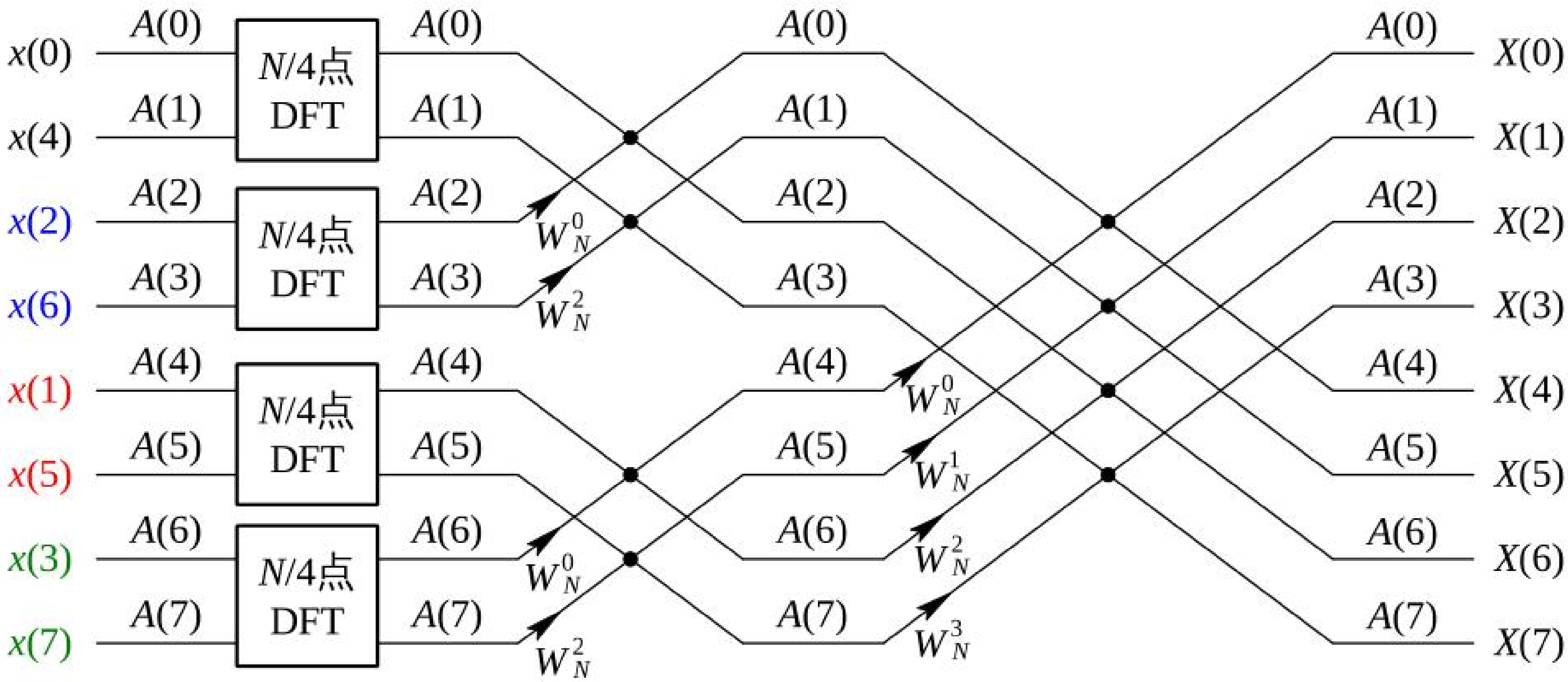
N 点 DFT 可分解成 2 个 N/2 点 DFT



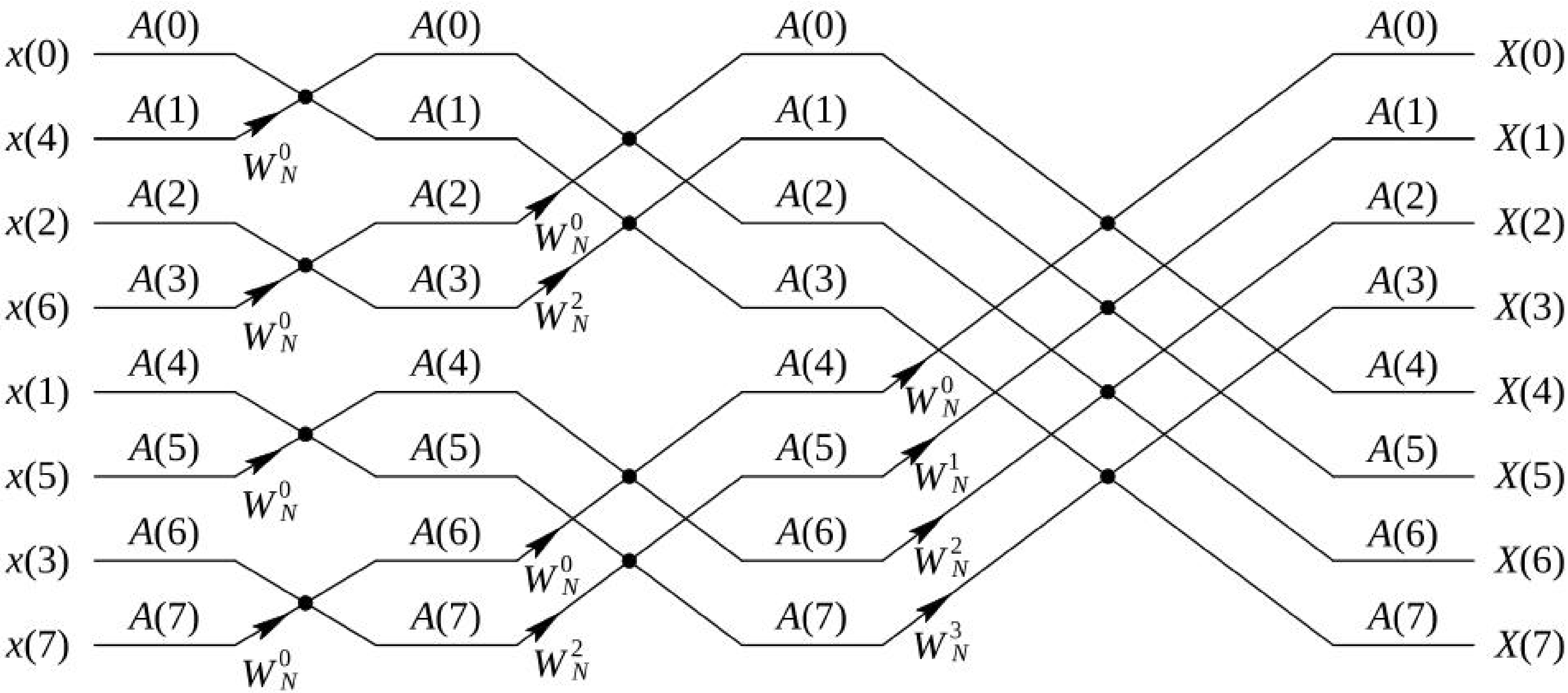
继续将每个 N/2 点 DFT 的时域序列按序号奇偶分成 2 组（共 4 组）



同理，每个 N/2 点 DFT 又可分解成 2 个 N/4 点 DFT



直至分解到 2 点的 FFT



DIT—FFT 算法与直接计算 DFT 运算量的比较

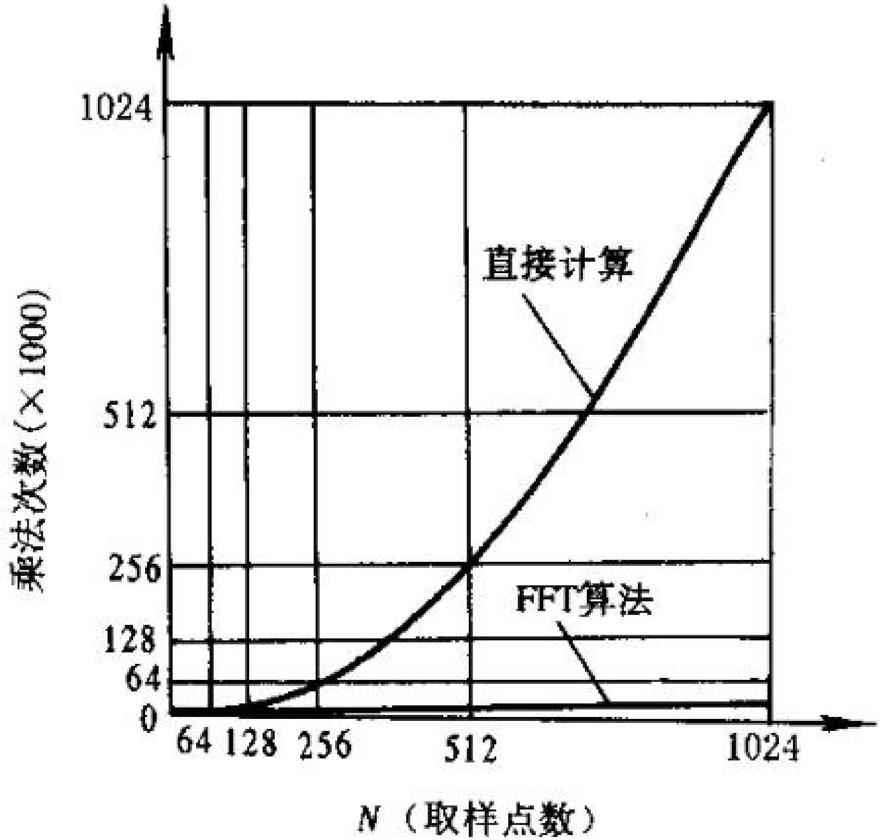
- 每一级运算都需要 $N/2$ 次复数乘和 N 次复数加（每个蝶形需要两次复数加法）。
- M 级运算总共需要的复数乘次数为

$$C_M(2) = \frac{N}{2} \cdot M = \frac{N}{2} \log_2 N$$

- 复数加次数为

$$C_A(2) = N \cdot M = N \log_2 N$$

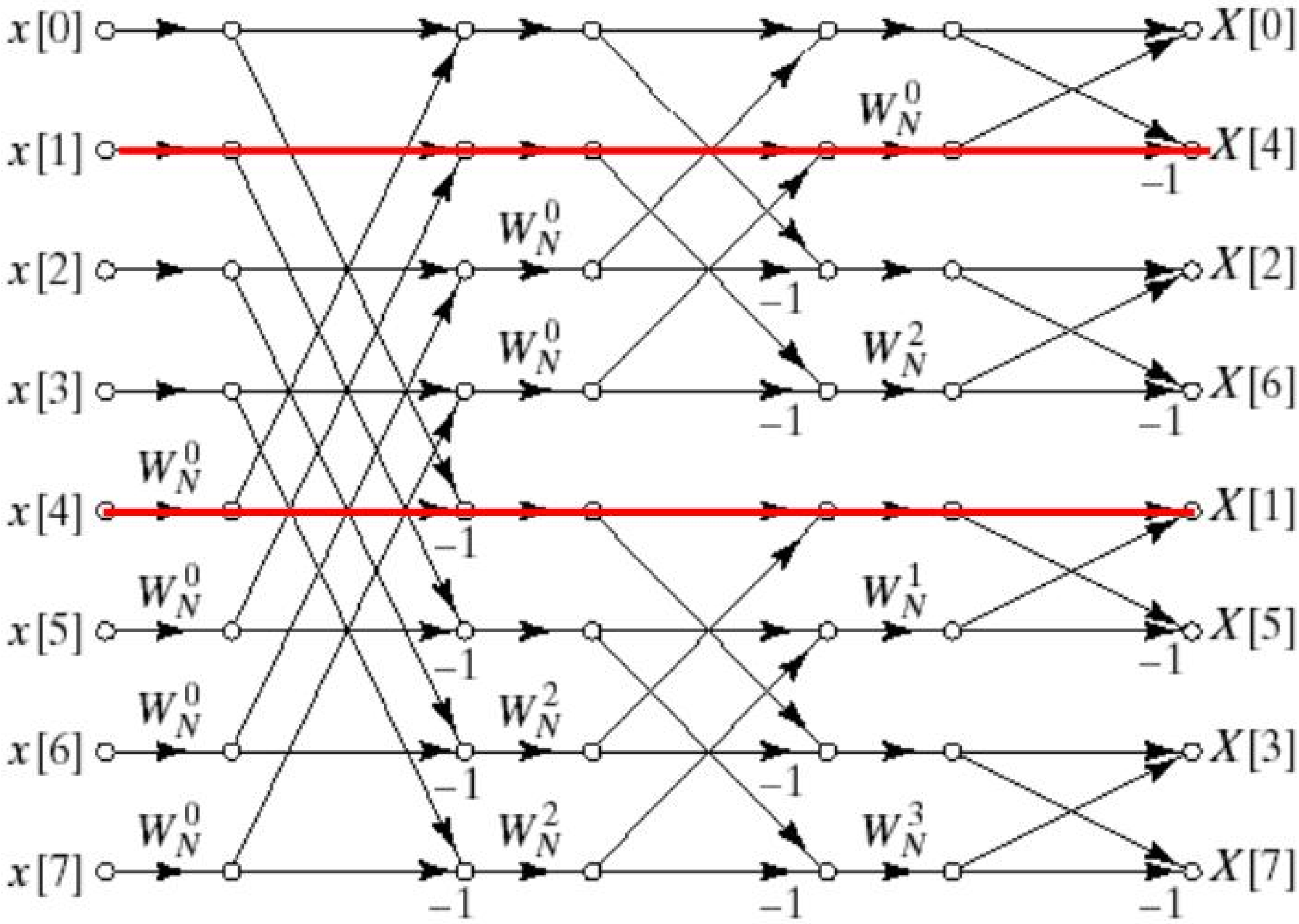
	复数乘次数	复数加次数
FFT	$\frac{N}{2}\log_2 N$	$N\log_2 N$
DFT	N^2	$N(N-1)$



N=2^{10}=1024 时， 复数乘的次数：

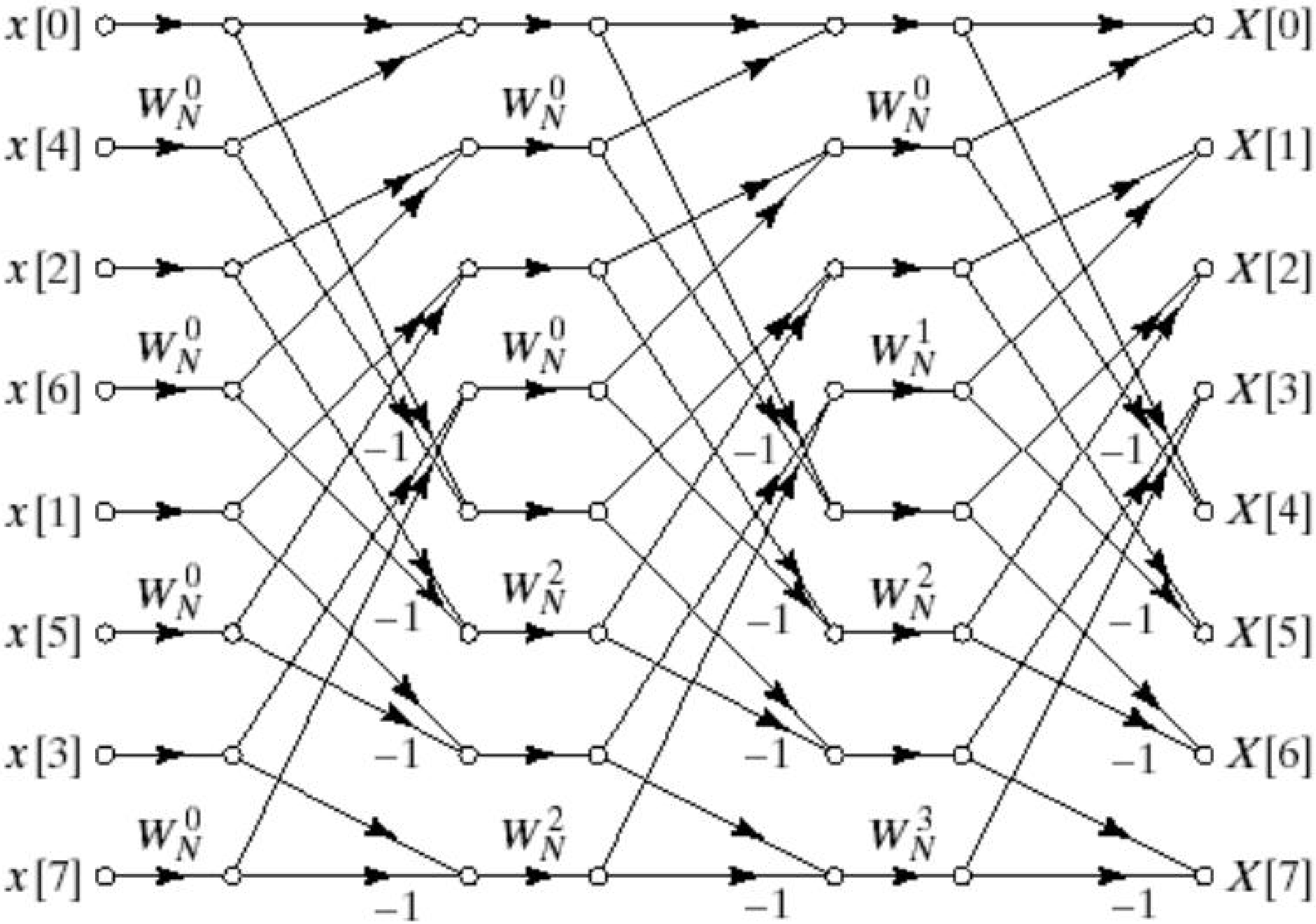
$$\frac{N^2}{(N/2)\log_2 N}=\frac{1048576}{5120}=204.8$$

Alternative forms：



Strongpoint : in-place computations

Shortcoming : non-sequential access of data



x[0]
x[4]
x[2]
x[6]
x[1]
x[5]
x[3]
x[7]
W_N^0
W_N^0
W_N^0
W_N^0
W_N^0
W_N^0
W_N^2
W_N^2
W_N^0
W_N^1
W_N^2
W_N^3
−1
X[0]
X[1]
X[2]
X[3]
X[4]
X[5]
X[6]
X[7]

2. Decimation-In-Frequency FFT algorithm (DIF-FFT)

- 在基 2 快速算法中，频域抽取法 FFT 也是一种常用的快速算法，简称 DIF-FFT 。
- 设序列 $x[n]$ 长度为 $N=2^M$，首先将 $x[n]$ 前后对半分开，得到两个子序列，其 DFT 为:

$$X[k]=\sum_{n=0}^{N-1}x[n]W_N^{kn} =\sum_{n=0}^{\frac{N}{2}-1}x[n]W_N^{kn}+\sum_{n=\frac{N}{2}}^{N-1}x[n]W_N^{kn}$$

$$=\sum_{n=0}^{\frac{N}{2}-1}x[n]W_N^{kn}+\sum_{n=0}^{\frac{N}{2}-1}x[n+\frac{N}{2}]W_N^{k\left(n+\frac{N}{2}\right)}$$

$$=\sum_{n=0}^{\frac{N}{2}-1}\left[x[n]+W_N^{\frac{kN}{2}}x[n+\frac{N}{2}]\right]W_N^{kn}$$

$$W_N^{kN/2}=(-1)^k=\begin{cases}1, & k=\text{偶数}\\ -1 & k=\text{奇数}\end{cases}$$

•当 k 取偶数 (k=2r, r =0,1,…, N/2-1) 时

$$X[2r]=\sum_{n=0}^{\frac{N}{2}-1}\left[x[n]+x[n+\frac{N}{2}]\right]W_{N/2}^{rn}$$

•当 k 取奇数 (k=2r+1, r =0,1,..., N/2-1) 时:

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

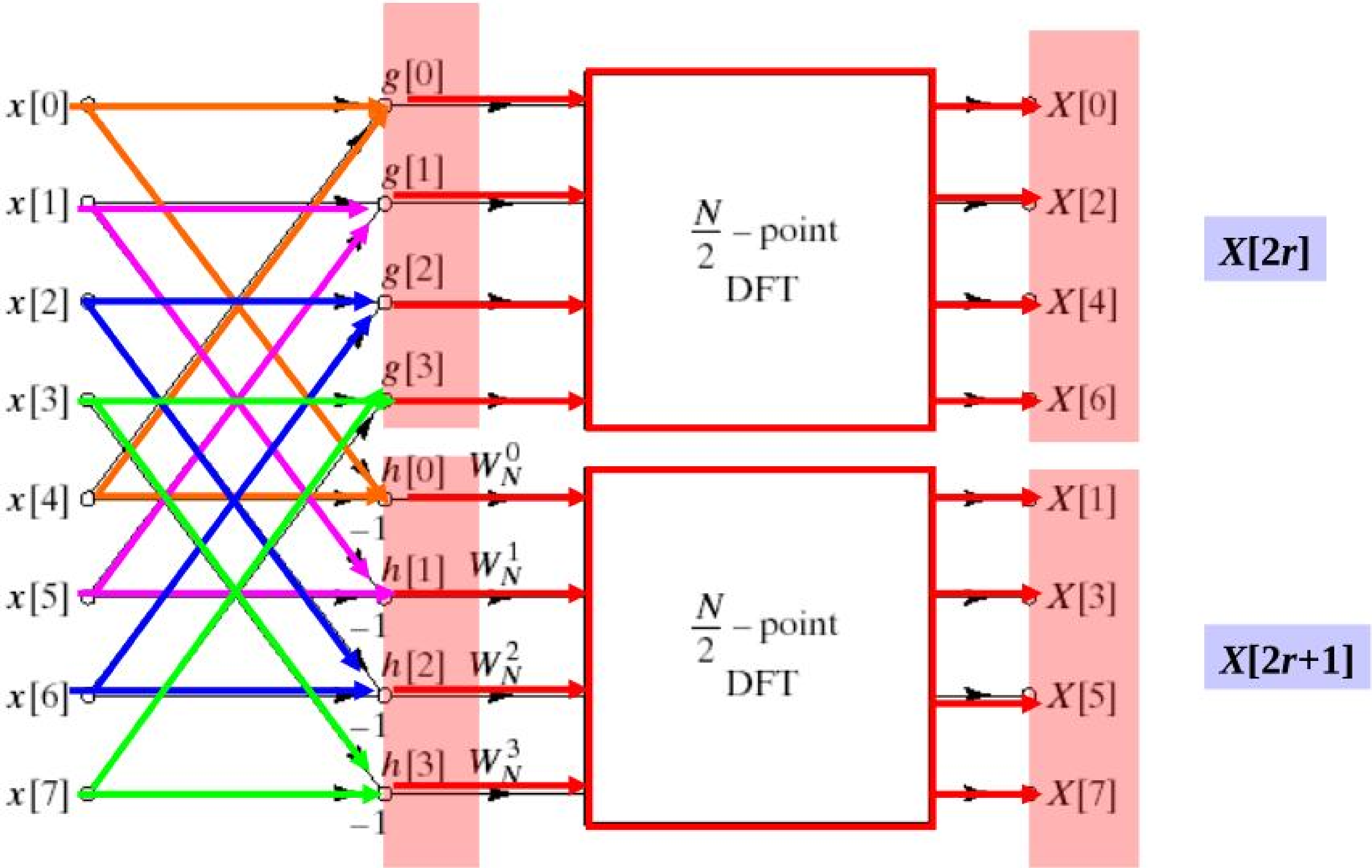
$$=\sum_{n=0}^{N/2-1}[x[n]-x[n+\frac{N}{2}]]W_N^{n}W_{N/2}^{nr}$$

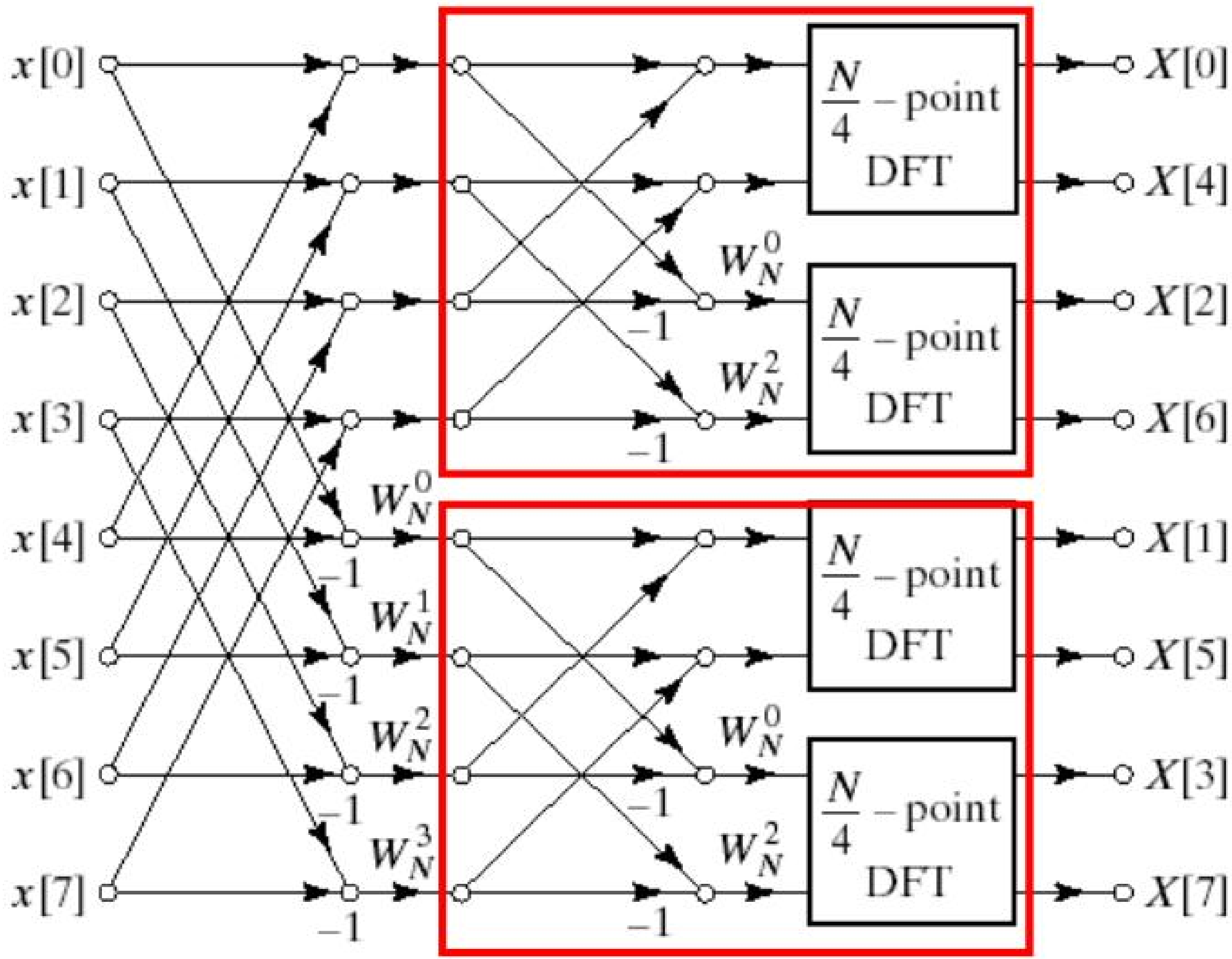
$$\begin{cases}X[2r]=\sum_{n=0}^{N/2-1}g[n]W_{N/2}^{rn}\\ X[2r+1]=\sum_{n=0}^{N/2-1}h[n]W_{N/2}^{nr}W_{N/2}^{rn}\end{cases}$$

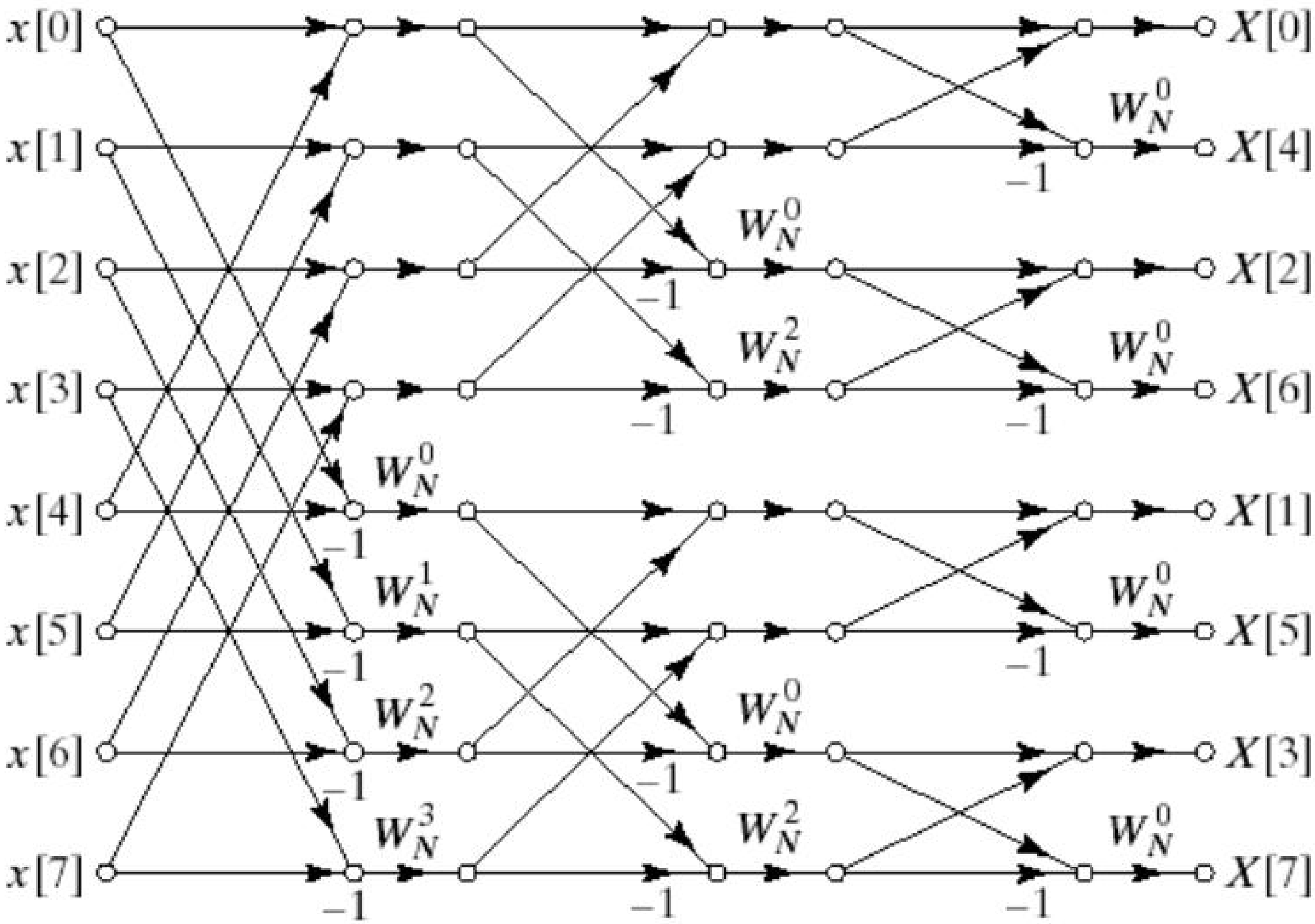
$$g[n]=x[n]+x[n+\frac{N}{2}]$$

$$h[n]=x[n]-x[n+\frac{N}{2}]$$

Decimation-in-frequency FFT Algorithms




x[0]
x[1]
x[2]
x[3]
x[4]
x[5]
x[6]
x[7]
W_N^0
W_N^1
W_N^2
W_N^3
–1
$\frac{N}{4}$ – point DFT
X[0]
X[4]
X[2]
X[6]
X[1]
X[5]
X[3]
X[7]


x[0]
x[1]
x[2]
x[3]
x[4]
x[5]
x[6]
x[7]
W_N^0
W_N^1
W_N^2
W_N^3
W_N^0
W_N^2
W_N^0
W_N^2
W_N^0
W_N^0
W_N^0
W_N^0
–1
X[0]
X[4]
X[2]
X[6]
X[1]
X[5]
X[3]
X[7]