第四章
快速傅里叶变换(FFT)
Fast Fourier Transforming

一、直接计算DFT的痛点——为什么需要FFT?

1.1 离散傅里叶变换的计算量

对于一个长度为N的有限长序列x(n),其DFT为：

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn} \qquad (4.2.1)$$

$$k=0,1,\cdots,N-1$$

旋转因子： $X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn}$

$$k=0,1,\cdots,N-1$$

$$W_N^{kn}=\mathrm{e}^{-j\frac{2\pi}{N}kn}$$

将旋转因子进行进一步展开：

利用欧拉公式 $e^{jx}=\cos(x)+j\sin(x)$

旋转因子：$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn}$$

$$k=0,1,\cdots,N-1$$

$$W_N^{kn}=\mathrm{e}^{-j\frac{2\pi}{N}kn}=?$$

$$e^{-j\frac{2\pi}{N}\mathrm{kn}}=\cos(\frac{2\pi}{N}\mathrm{kn})-j\sin(\frac{2\pi}{N}\mathrm{kn})$$

DFT的计算量：

$$X(k)=\sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad (4.2.1)$$

$$k=0,1,\cdots,N-1$$

固定

公式中的变量有两个：n和k

DFT的计算量：

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn} \qquad (4.2.1)$$

$$k=0,1,\cdots,N-1$$

对应一个k需要复数乘： N次

复数加： （N-1）次

DFT的计算量：

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn} \qquad (4.2.1)$$

$$k=0,1,\cdots,N-1$$

所有N个k需要复数乘： $N*N$次

复数加： $N*(N-1)$次；

当N》0时，总的需要复数乘和加： N^2次

例如：一个长度为10240的序列，其运算量是多少？

N^2=10240*10240　　上亿次加和乘

普通 CPU 需 0.1 秒以上，无法满足实时信号处理，（如 20ms / 帧的音频滤波）

FFT 的核心是利用旋转因子的对称性和周期性，将 N 点 DFT “分拆” 成多个短点 DFT（如 2 点、4 点）

$$N\log_2 N = 1.4*10000$$

计算量仅为 DFT 的 1/700，普通 CPU 可在 1ms 内完成，完全满足实时需求。

二、如何减少DFT的运算量？

答：将长的序列分解成短的序列

基础条件：旋转因子W_N^{kn}的周期性和对称性、可约性

旋转因子的对称性、周期性：

1） W_N^m 的周期性：

$$W_N^{m+lN}=e^{-j\frac{2\pi}{N}(m+lN)}=e^{-j\frac{2\pi}{N}m}=W_N^m \qquad (4.2.2)$$

2） W_N^m 的对称性：

$$W_N^{m+\frac{N}{2}}=-W_N^m \qquad \left(W_N^{km}\right)^*=W_N^{-km} \qquad (4.2.3)$$

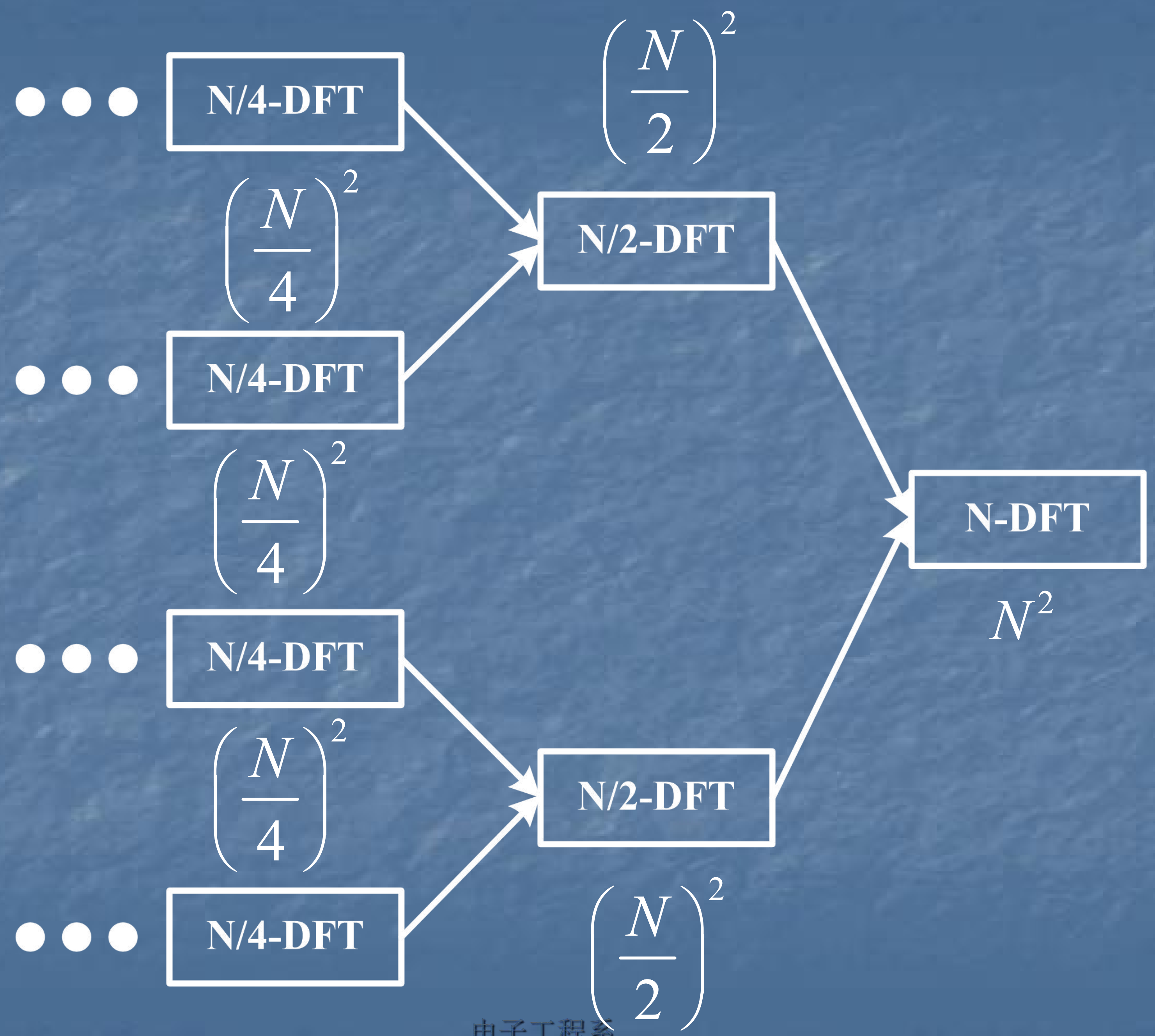
3） W_N^m 的可约性：

$$W_N^{mk}=W_{MN}^{Mmk} \qquad W_N^{mk}=W_{N/M}^{mk/M}$$

还有 $W_N^{N/2}=-1,\qquad W_N^{N/4}=-j$

$W_N^0=1,$

利用这些特性，DFT中有些项可以合并;
可以将长序列的DFT分解为短序列的DFT，
提高运算速度。


N/4-DFT
N/4-DFT
N/4-DFT
N/4-DFT
N/2-DFT
N/2-DFT
N-DFT
$\left(\frac{N}{2}\right)^2$
$\left(\frac{N}{4}\right)^2$
$\left(\frac{N}{4}\right)^2$
$\left(\frac{N}{4}\right)^2$
$\left(\frac{N}{2}\right)^2$
N^2

4.2.2 基2时域抽取法原理

设序列$x(n)$的长度为N,

$$N = 2^M$$

$$M = \log_2(N)$$

N点 DFT

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

将$x(n)$分为$x_1(n)$和$x_2(n)$，分成<u>偶数</u>和<u>奇数</u>序列项

则$N/2$点的序列为：

$$x_1(r)=x(2r), \qquad r=0, 1, \ldots, N/2-1$$

$$x_2(r)=x(2r+1), \quad r=0, 1, \ldots, N/2-1$$

则$x(n)$的DFT：

$$X(k)=\sum_{n=偶数} x(n)W_N^{kn}+\sum_{n=奇数} x(n)W_N^{kn}$$

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

$$=\sum_{r=0}^{N/2-1} x_1(r)W_N^{2kr}+W_N^k\sum_{r=0}^{N/2-1} x_2(r)W_N^{2kr}$$

将X(k)由X_1(k)和X_2(k)来表达

$$X_1(k)=\sum_{r=0}^{N/2-1} x_1(r)W_{N/2}^{kr}$$

$$X_2(k)=\sum_{r=0}^{N/2-1} x_2(r)W_{N/2}^{kr}$$

$$X(k)=\sum_{r=0}^{N/2-1} x_1(r)W_N^{2kr}+W_N^k\sum_{r=0}^{N/2-1} x_2(r)W_N^{2kr}$$

$W_{N/2}^{kr}$ 与 W_N^{2kr} 相等？

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

$$X(k) = \sum_{r=0}^{N/2-1} x_1(r) W_N^{2kr} + W_N^k \sum_{r=0}^{N/2-1} x_2(r) W_N^{2kr}$$

$$= X_1(k) + W_N^k X_2(k) \qquad k = 0, 1, \cdots, N-1$$

为了实现蝶形运算进一步减少计算量，我们将$X(k)$再进行分解。

$$X(k) \quad k=0,1,\cdots,N-1$$

$$\begin{cases} X(k) & k=0,1,\ldots,\frac{N}{2}-1 \\ X\left(k+\frac{N}{2}\right) & k=0,1,\ldots,\frac{N}{2}-1 \end{cases}$$

因此，可写出两个$N/2$点的方程:

$$\begin{cases} X(k) = X_1(k)k + W_N^k X_2(k) \\ \qquad k = 0,1,\ldots,\dfrac{N}{2}-1 \qquad (4.2.7) \\ X\left(k+\dfrac{N}{2}\right) = X_1\left(k+\dfrac{N}{2}\right) + W_N^{k+\frac{N}{2}} X_2\left(k+\dfrac{N}{2}\right) \end{cases}$$

$$k = 0,1,\ldots,\frac{N}{2}-1 \qquad (4.2.8)$$

$$X\left(k+\frac{N}{2}\right)=X_1\left(k+\frac{N}{2}\right)+W_N^{k+\frac{N}{2}}X_2\left(k+\frac{N}{2}\right)$$

$$k=0,1,\ldots,\frac{N}{2}-1$$

$$X_1\left(k+\frac{N}{2}\right)=\sum_{r=0}^{\frac{N}{2}-1}x_1(r)W_{N/2}^{\left(k+\frac{N}{2}\right)r}$$

$$=\sum_{r=0}^{\frac{N}{2}-1}x_1(r)W_{N/2}^{kr}=X_1(k)$$

$$W_{N/2}^{\left(k+\frac{N}{2}\right)r}=e^{-j\frac{2\pi}{N/2}(k+\frac{N}{2})r}$$

$$=\cos[\frac{2\pi}{N/2}(k+\frac{N}{2})r]-j\sin[\frac{2\pi}{N/2}(k+\frac{N}{2})r]$$

$$=\cos[\frac{2\pi}{N}(2k+N)r]-j\sin[\frac{2\pi}{N}(2k+N)r]$$

$$=\cos[\frac{2\pi}{N}2kr+2\pi r]-j\sin[\frac{2\pi}{N}2kr+2\pi r]$$

$$=\cos[\frac{2\pi}{N}2kr]-j\sin[\frac{2\pi}{N}2kr]=e^{-j\frac{2\pi}{N/2}kr}=W_{N/2}^{kr}$$

$$X\left(k+\frac{N}{2}\right)=X_1\left(k+\frac{N}{2}\right)+W_N^{k+\frac{N}{2}}X_2\left(k+\frac{N}{2}\right)$$

$$k=0,1,\ldots,\frac{N}{2}-1$$

$$=X_1(k)+W_N^{k+\frac{N}{2}}X_2(k)$$

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

$$= e^{-j\frac{2\pi}{N}k} e^{-j\frac{2\pi}{N}\frac{N}{2}}$$

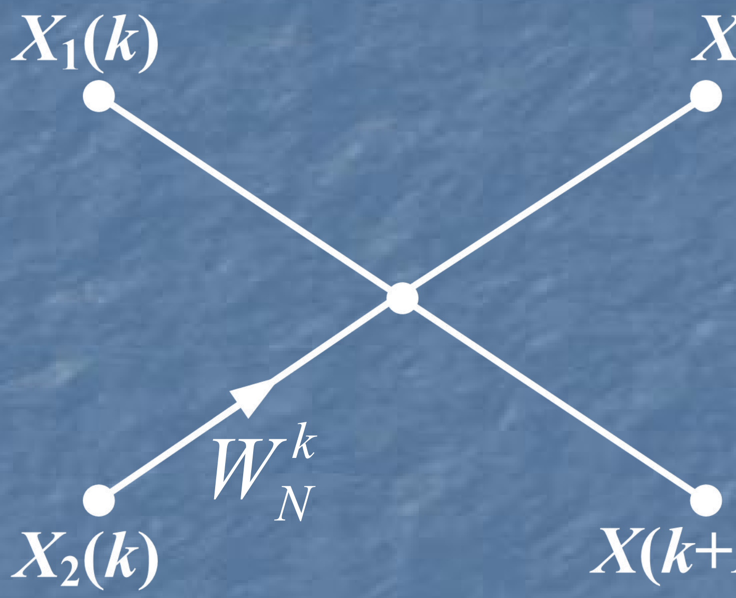
$$= W_N^k[\cos(\pi) - j\sin(\pi)] = \text{-}W_N^k$$

由于 $W_N^{k+\frac{N}{2}} = -W_N^k$

因此，可写出两个$N/2$点的方程:

$$\begin{cases} X(k) = X_1(k) + W_N^k X_2(k) \\ \qquad k = 0,1,\ldots,\dfrac{N}{2}-1 & (4.2.7) \\ X\left(k+\dfrac{N}{2}\right) = X_1(k) - W_N^k X_2(k) \\ \qquad k = 0,1,\ldots,\dfrac{N}{2}-1 & (4.2.8) \end{cases}$$

表示上述算法可用蝶形结（ butterfly）



$$X(k) = X_1(k) + W_N^k X_2(k)$$

$$X(k+N/2) = X_1(k) - W_N^k X_2(k)$$

图4.2.1 蝶形运算符号

时域按奇偶分→频域按前后半分→短序列算短 DFT→蝴蝶运算组合

作图要素：

(1)左边两路为输入

(2)右边两路为输出

(3)中间以一个小圆表示加、减运算（右上路为相加输出、右下路为相减输出)

(4)如果在某一支路上信号需要进行相乘运算，则在该支路上标以箭头，将相乘的系数标在箭头旁。

(5)当支路上没有箭头及系数时，则该支路的传输比为1。

例子：求 N=8点FFT变换

1、先将8点x(n)分为两个4点$x_1(n)$和$x_2(n)$，

$x_1(n)$: $x(0),x(2),x(4),x(6)$为偶子序列

$x_2(n)$: $x(1),x(3),x(5),x(7)$为奇子序列

2、再对两个4点序列$x_1(n)$和$x_2(n)$做4点的DFT得到$X_1(k)$和$X_2(k)$。

$$X(k)=X_1(k)+W_N^k X_2(k) \qquad k=0,1,\ldots,N-1$$

3、然后进行蝶形运算

$$\begin{cases} X(k)=X_1(k)+W_N^k X_2(k) \\ X\left(k+\frac{N}{2}\right)=X_1(k)-W_N^k X_2(k) \end{cases} \qquad k=0,1,\ldots,\frac{N}{2}-1$$

1、比较直接DFT与上页分解一次的FFT运算量

(a) N=8点的直接DFT的计算量为：
复数相乘：N^2次(64次），
复数相加：N(N-1)次(8＊(8-1)=56次)。共120次。

(b)分解一次的FFT运算量：两次4点DFT和4次蝶形运算

(b1) 分解的两个4点DFT的运算量

$$X(k) = DFT\{x(2r)\} + DFT\{x(2r+1)\}$$

偶数

其复数相乘为 $\left(\frac{N}{2}\right)^2$

复数相加为 $\frac{N}{2}\left(\frac{N}{2}-1\right)$

$\oplus$

奇数

其复数相乘为 $\left(\frac{N}{2}\right)^2$

复数相加为 $\frac{N}{2}\left(\frac{N}{2}-1\right)$

复数相乘为 $\frac{N^2}{2}=32$次，复数相加为 $N\left(\frac{N}{2}-1\right)=24$次，

共计为56次。

(b2)要运算一个蝶形结，需要乘法 $W_N^k X_2(k)$ 一次，加法两次。

$$\begin{cases} X(k) = X_1(k) + W_N^k X_2(k) \\ X(k+N/2) = X_1(k) - W_N^k X_2(k) \end{cases}$$

4次蝶形运算，需要乘法4次，加法8次。

所以8点一次分解的FFT求$X(k)$共需32+4=36次复数乘法，24+8 = 32次复数加法，共计为68次。看出仅做一次分解就可以节省约一半计算量。

(c) 将N=8点分解成2个4点的DFT的信号流图

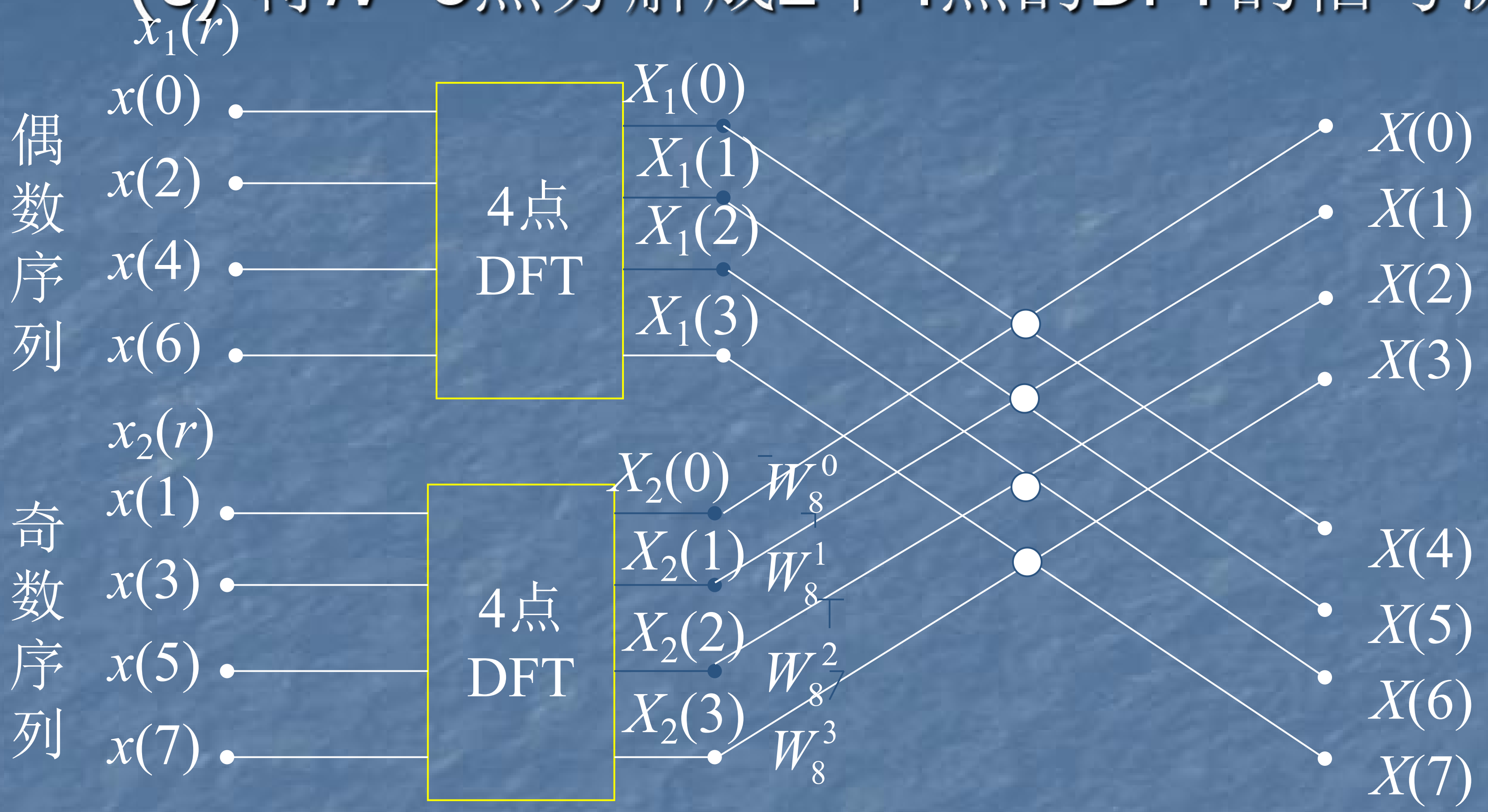


图4.2.2 N点DFT的一次时域抽取分解图(N=8)

如：$X(0)=X_1(0)+X_2(0)W_8^0$　$X(1)=X_1(1)+X_2(1)W_8^1$

$X(2)=X_1(2)+X_2(2)W_8^2$　$X(3)=X_1(3)+X_2(3)W_8^3$

$X(4)$~$X(7)$

同学们自己写

2、4点DFT再分解

因为4点DFT还是比较麻烦，所以再继续分解。

将两个4点子序列按奇/偶分解成四个2点子序列。

即将$x_1(r)$和$x_2(r)$分解成奇、偶四个 2点的子序列。

$$x_1(r):\begin{cases} x(0)、x(4) & \text{偶序列} \\ x(2)、x(6) & \text{奇序列} \end{cases} \quad \text{同理:}\ x_2(r):\begin{cases} x(1)、x(5) & \text{偶序列} \\ x(3)、x(7) & \text{奇序列} \end{cases}$$

$$\text{若设:}\begin{cases} x_1(2L) = x_3(L) & \text{偶序列} \\ x_1(2L+1) = x_4(L) & \text{奇序列} \end{cases} \left(L = 0 \cdots \frac{N}{4} - 1 \right)\text{，在此}L = 0,1$$

$$\text{同理:}\begin{cases} x_2(2L) = x_5(L) & \text{偶序列} \\ x_2(2L+1) = x_6(L) & \text{奇序列} \end{cases} \left(L = 0 \cdots \frac{N}{4} - 1 \right)\text{，在此}L = 0,1$$

$x_1(r) \xrightarrow{\text{DFT}} X_1(k)$ 可分解为：

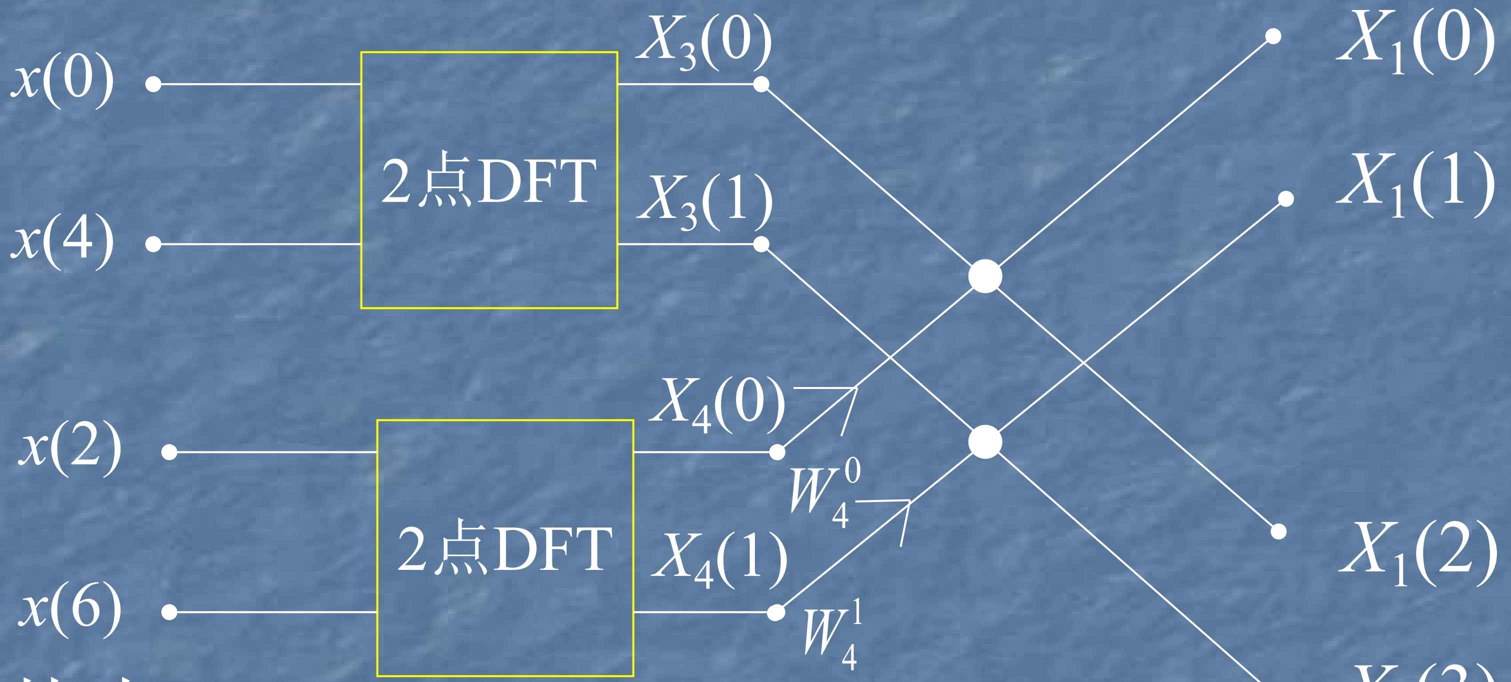
$$X_1(k)=\sum_{L=0}^{N/4-1} x_1(2L)W_{N/2}^{2Lk}+\sum_{L=0}^{N/4-1} x_1(2L+1)W_{N/2}^{(2L+1)k}$$

$$=X_3(k)+W_{N/2}^{k}X_4(k) \quad 其中k=0,1,\ ...\frac{N}{2}-1$$

对上式进行蝶形运算：

$$\begin{cases} X_1(k)=X_3(k)+W_{N/2}^{k}X_4(k) \\ X_1(k+N/4)=X_3(k)-W_{N/2}^{k}X_4(k) \end{cases} \quad 其中k=0,1,\ ...\frac{N}{4}-1$$

(a) 一个2点的DFT蝶形流图



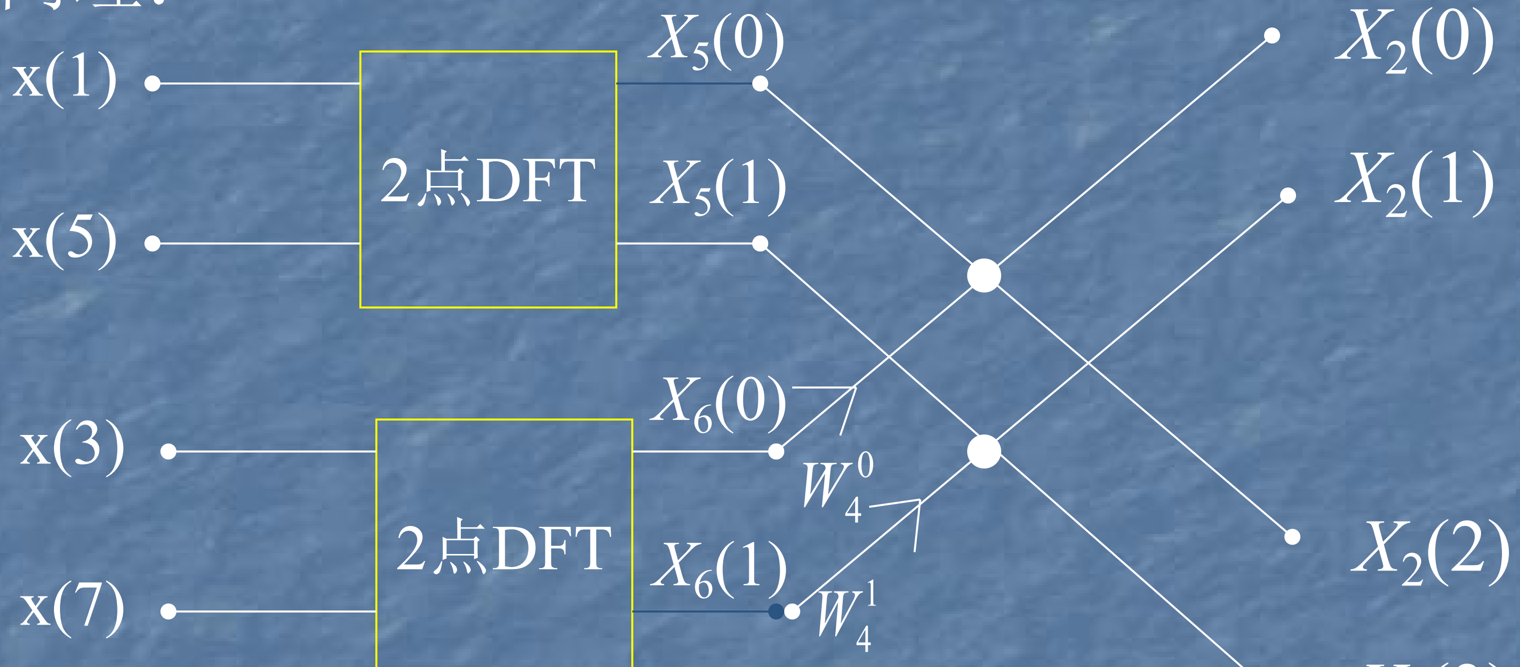
其中

$$X_1(0)=X_3(0)+W_4^0X_4(0) \quad X_1(1)=X_3(1)+W_4^1X_4(1)$$

$$X_1(2)=X_3(0)-W_4^0X_4(0) \quad X_1(3)=X_3(1)-W_4^1X_4(1)$$

(b) 另一个2点的DFT蝶形流图

同理：



其中：

$$X_2(0)=X_5(0)+W_4^0X_6(0) \qquad X_2(1)=X_5(1)+W_4^1X_6(1)$$

$$X_2(2)=X_5(0)-W_4^0X_6(0) \qquad X_2(3)=X_5(1)-W_4^1X_6(1)$$

(3) 将N/4(2点)DFT再分解成2个1点的DFT
(a) 求一点的DFT

一点DFT就等于输入信号本身

两点DFT可以用一个蝶形结表示

$$x_3(r):\begin{cases} x(0) & \text{偶序列} \\ x(4) & \text{奇序列} \end{cases}$$

$$x_3(r):\begin{cases} x(0) & 偶序列 \\ x(4) & 奇序列 \end{cases}$$

$$X_3(k)=\sum_{n=0}^{1}x(n)W_2^{nk}$$:

$$X_3(0)=x(0)W_2^0+x(4)W_2^0=x(0)W_2^0+x(4)W_2^0$$

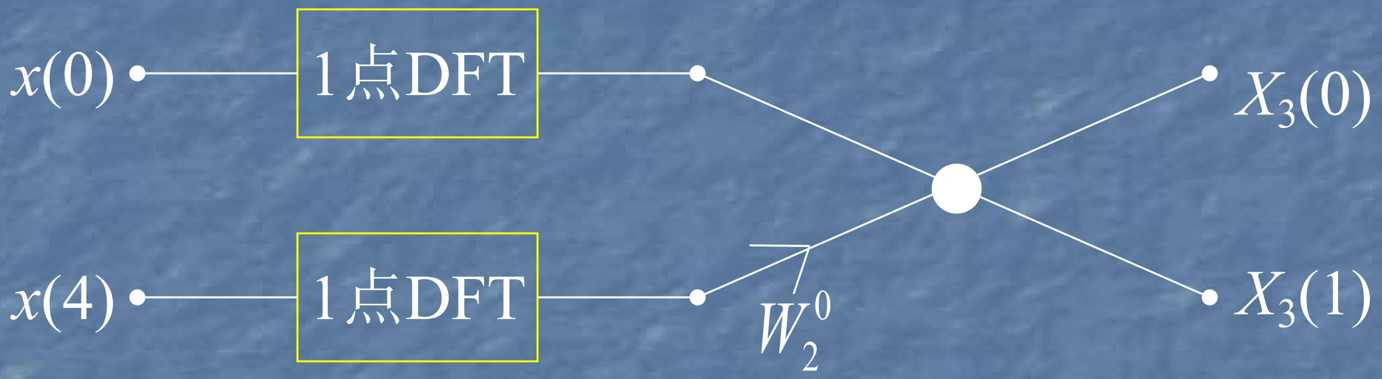
$$X_3(1)=x(0)W_2^0+x(4)W_2^1=x(0)W_2^0-x(4)W_2^0$$

这是一个蝶形结，这里用到对称性：

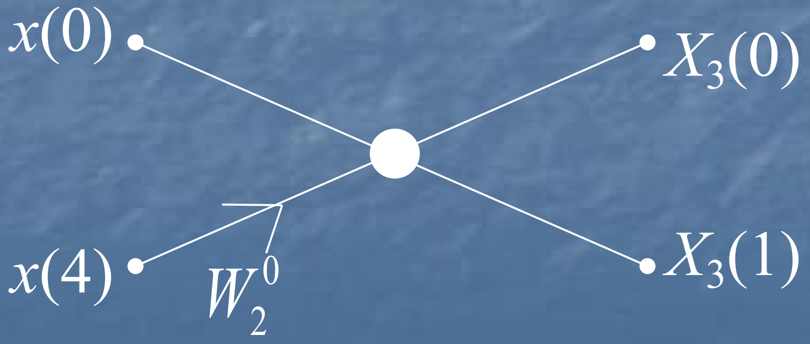
$W_N^{nk\pm\frac{N}{2}}=-W_N^{nk}$，则 $W_2^{nk\pm1}=-W_2^{nk}$，其中 $n=0,1;k=0,1$

$\therefore W_2^{0\pm1}=-W_2^0$

(b) 2个1点的DFT蝶形流图



进一步简化为蝶形流图：



其中：

$$X_3(0) = x(0) + W_2^0 x(4)$$
$$= x(0) + x(4)$$
$$X_3(1) = x(0) - W_2^0 x(4)$$
$$= x(0) - x(4)$$

(4)一个完整N=8的按时间抽取FFT的运算流图

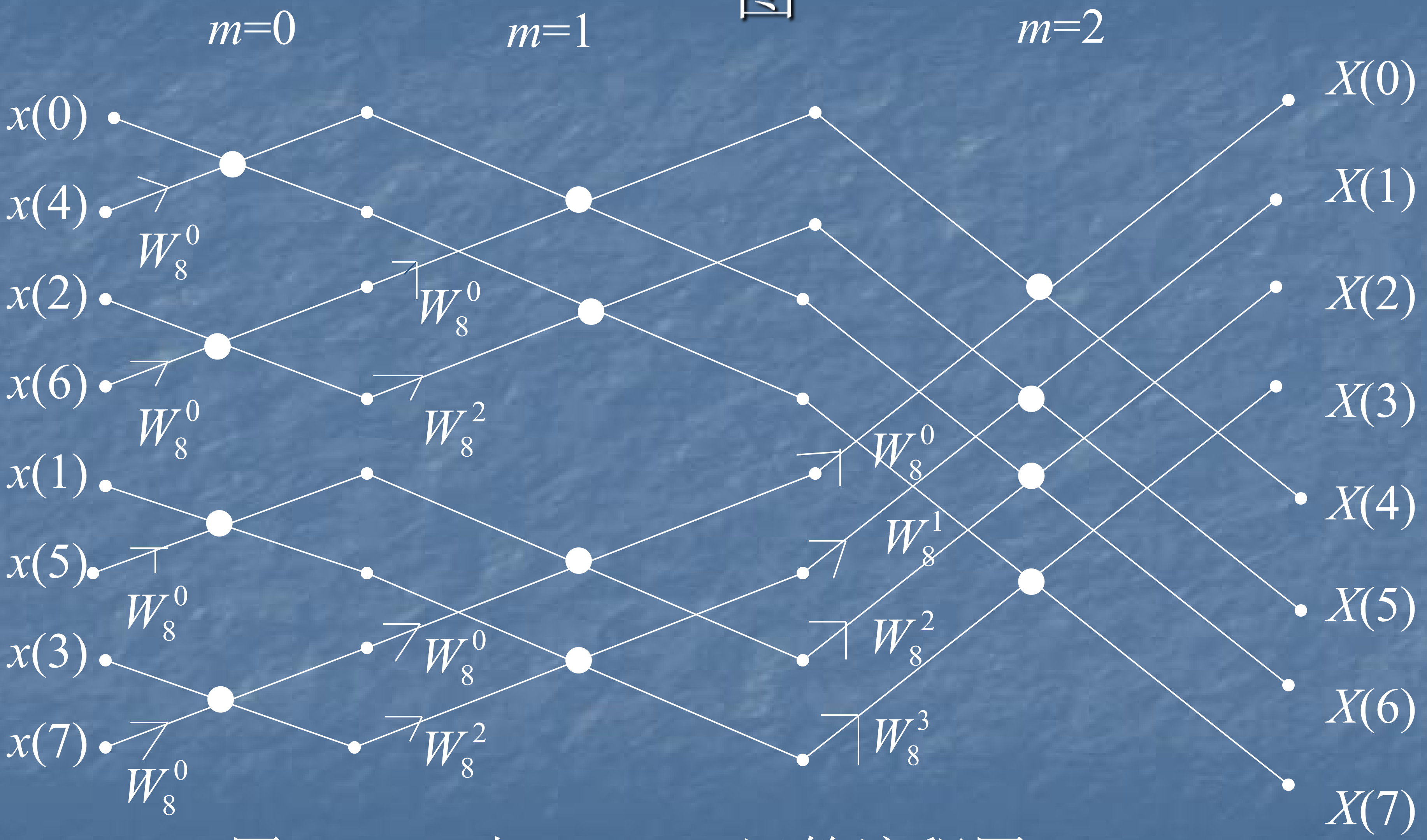


图4.2.4 N点DIT-FFT运算流程图(N=8)

4.2.3 DIT-FFT算法与直接计算DFT运算量的比较

由前面介绍的DIT-FFT运算流图可见：

每级都由$N/2$个蝶形单元构成，因此每一级运算都需要$N/2$次复乘和N次复加（每个结加减各一次）。这样（$N=2^M$）M级运算共需要：

复乘次数： $C_M(2)=\frac{N}{2}M=\frac{N}{2}\log_2 N$

复加次数： $C_A(2)=N\times M=N\log_2 N$

可以得出如下结论：

按时间抽取法所需的复乘数和复加数都是与 $N\log_2 N$ 成正比。而直接计算DFT时所需的复乘数与复加数则都是与N^2成正比.(复乘数N^2,复加数$N(N-1)\approx N^2$)

看N=8点和N=1024点时直接计算DFT与用基2-按时间抽取法FFT的运算量。

N	N^2	$N\log_2^N$	$N^2 / N\log_2^N$
8	64	24	2.7
2^{10}	2^{20}	$2^{10}\cdot 10$	102.4

看出：当N较大时，按时间抽取法将比直接法快一、二个数量级之多。