离散时间傅立叶变换（DTFT）：

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}$$

频域抽样 ⟵ 离散化

$$X(k)$$

时域周期延拓

有限长序列 $x(n) \longrightarrow \tilde{x}(n) \overset{\text{DFS}}{\longleftrightarrow} \tilde{X}(k)$

$$x(n) \Longleftrightarrow X(k)$$

有限长序列的离散傅立叶变换（**DFT**）

第五章　离散傅立叶变换（DFT）

$$x(n), \quad 0 \le n \le N-1$$

$$R_N(n) = \begin{cases} 1, & 0 \le n \le N-1 \\ 0, & \text{其它} \end{cases}$$

$$\tilde{x}(n) = \sum_{r=-\infty}^{\infty} x(n-rN)$$

$$\tilde{x}(n) = x((n))_N$$

$$x(n) = \tilde{x}(n) R_N(n)$$

离散时间傅立叶级数

令： $N\dot{A}_k = \tilde{X}(k)$　　$\boxed{W_N = e^{-j\frac{2\pi}{N}}}$

$$\begin{cases} \tilde{x}(n) = \sum_{k=0}^{N-1} \dot{A}_k e^{j\frac{2\pi}{N}kn} \\ \dot{A}_k = \frac{1}{N} \sum_{n=0}^{N-1} \tilde{x}(n) e^{-j\frac{2\pi}{N}kn} \end{cases}
\Rightarrow
\begin{cases} \tilde{x}(n) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{X}(k) W_N^{-kn} \\ \tilde{X}(k) = \sum_{n=0}^{N-1} \tilde{x}(n) W_N^{kn} \end{cases}$$

离散时间傅立叶级数（DFS）

$$\begin{cases} \tilde{x}(n) = \dfrac{1}{N}\sum_{k=0}^{N-1}\tilde{X}(k)W_N^{-kn} \\ \tilde{X}(k) = \sum_{n=0}^{N-1}\tilde{x}(n)W_N^{kn} \end{cases}$$

时域 $\xrightarrow{\text{DFS}}$ 频域

离散傅立叶变换（DFT）

$$x(n) \overset{\text{DFT}}{\longleftrightarrow} X(k)$$

$$\begin{cases} X(k) = \sum_{n=0}^{N-1}x(n)W_N^{kn} & (0 \le k \le N-1) \\ x(n) = \dfrac{1}{N}\sum_{k=0}^{N-1}X(k)W_N^{-kn} & (0 \le n \le N-1) \end{cases}$$

DFT与DFS之间的关系：

$$\begin{cases} \tilde{X}(k) = X((k))_N \\ X(k) = \tilde{X}(k)R_N(k) \end{cases}$$

$$X(e^{j\omega}) = \sum_{n=0}^{N-1} x(n) e^{-j\omega n}$$ 有限长序列的离散时间傅立叶变换

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}$$ 有限长序列的离散傅立叶变换

$$(W_N = e^{-j\frac{2\pi}{N}})$$ $$\boxed{X(k) = X(e^{j\omega})\Big|_{\omega=\frac{2\pi}{N}k}}$$ $$0 \le k \le N-1$$

DFT与DTFT之间的关系：

1. 有限长序列的离散傅立叶变换 (DFT)是该序列的离散时间傅立叶变换(DTFT)在一个周期内N个等间隔的样本，但是并不等同于该序列的频谱。

2. 为保证频域样本能够完全表征时域信号，样本点数必须大于或等于该有限序列的长度。

例： $x(n)=\cos\frac{\pi}{4}n \quad (0\le n\le 7)$ 求 $X(k)$

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

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

$$=\sum_{n=0}^{7}\cos\frac{\pi}{4}n\cdot e^{-j\frac{\pi}{4}kn}=\frac{1}{2}\sum_{n=0}^{7}(e^{j\frac{\pi}{4}n}+e^{-j\frac{\pi}{4}n})e^{-j\frac{\pi}{4}kn}$$

$$X(0)=\frac{1}{2}\sum_{n=0}^{7}(e^{j\frac{\pi}{4}n}+e^{-j\frac{\pi}{4}n})=0$$

$$X(1)=\frac{1}{2}\sum_{n=0}^{7}(e^{j\frac{\pi}{4}n}+e^{-j\frac{\pi}{4}n})e^{-j\frac{\pi}{4}n}=4$$

$$X(k)=\begin{cases}4 & k=1,7\\ 0 & \text{其它}k\end{cases}$$

离散傅立叶变换的性质：

1 圆周移位:

有限长序列的圆周移位：

$$x_1(n) = x((n-n_0))_N R_N(n)$$

$$\boxed{X_1(k) = W_N^{kn_0} X(k)}$$

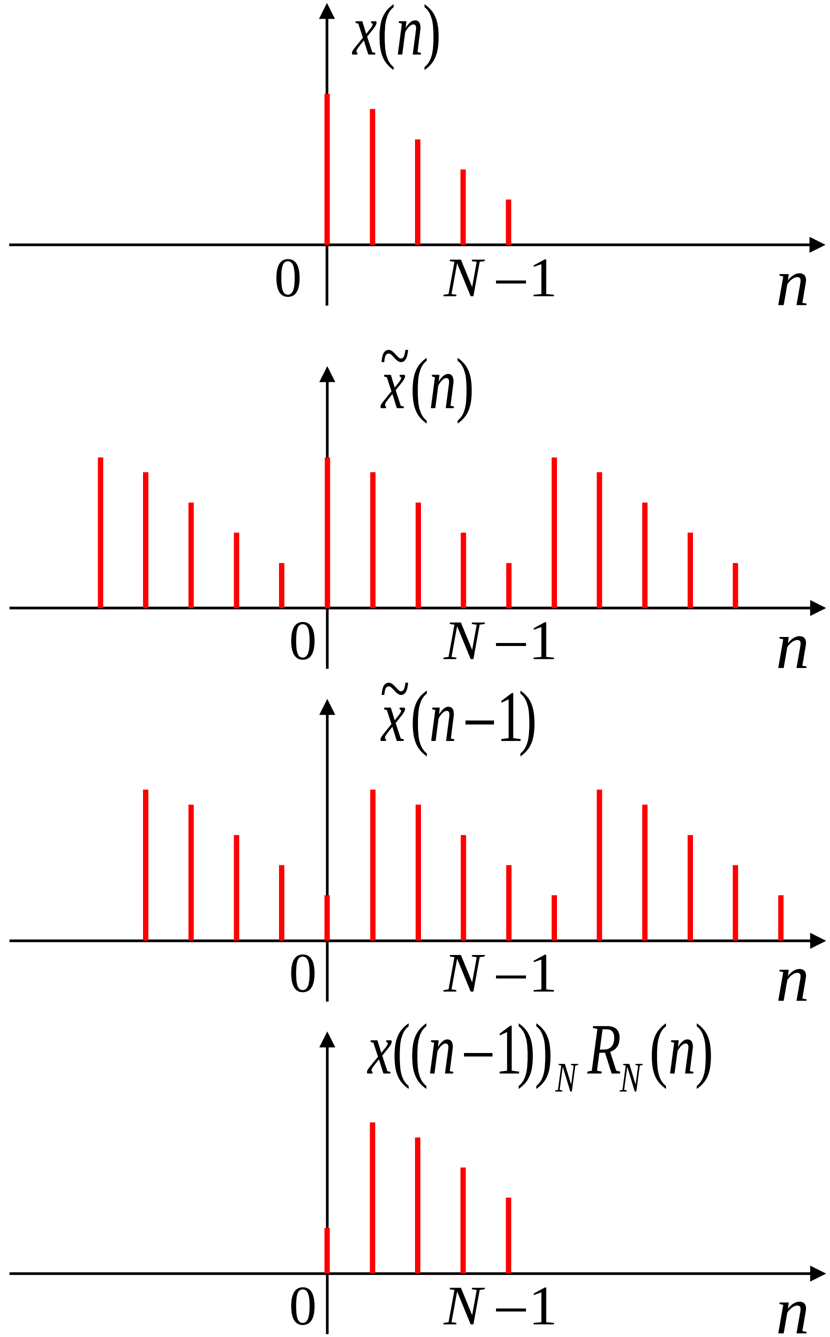
$$\tilde{x}(n-n_0) \xrightarrow{\text{DFS}} W_N^{kn_0} \tilde{X}(k)$$

$$X_1(k) = \text{DFT}[\tilde{x}(n-n_0)R_N(n)]$$

$$= \underline{W_N^{kn_0} \tilde{X}(k) R_N(k)}$$

$$= W_N^{kn_0} X(k)$$

$$\boxed{W_N^{-k_0 n} x(n) \leftrightarrow X((k-k_0))_N R_N(k)}$$



2 圆周卷积:

周期卷积: $\tilde{x}(n)$和$\tilde{y}(n)$都是以N为周期的序列

$$\boxed{\tilde{f}(n)=\tilde{x}(n)\circledast\tilde{y}(n)}=\sum_{m=0}^{N-1}\tilde{x}(m)\tilde{y}(n-m)=\sum_{m=0}^{N-1}\tilde{x}(n-m)\tilde{y}(m)$$

假设$x(n)$和$y(n)$分别是$\tilde{x}(n)$和$\tilde{y}(n)$的主值周期

$$\tilde{f}(n)=\sum_{m=0}^{N-1}x(m)y((n-m))_N=\sum_{m=0}^{N-1}x((n-m))_N\,y(m)$$

$$f(n)=\tilde{f}(n)R_N(n)=\sum_{m=0}^{N-1}x(m)y((n-m))_N\,R_N(n)$$

圆周卷积: $x(n)$和$y(n)$是长度相同的有限长序 列

$$f(n)=x(n)\circledast y(n)=\sum_{m=0}^{N-1}x(m)\underline{y((n-m))_N\,R_N(n)}$$ 圆周移位

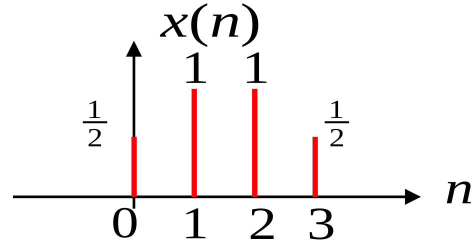
(1) 将两个有限长序列延拓成周期序列，并作周期卷积

(2) 对卷积结果取主值周期

圆周卷积与周期卷积过程实质是同样的

例: 如图所示

$x(n)$为$N=4$的有限长序列



计算$y(n)=x(n) \circledast x(n)$

解: $$y(n)=\sum_{m=0}^{N-1} x(m)x((n-m))_N R_N(n)$$

$$x(m)=\{\tfrac{1}{2},1,1,\tfrac{1}{2}\}$$

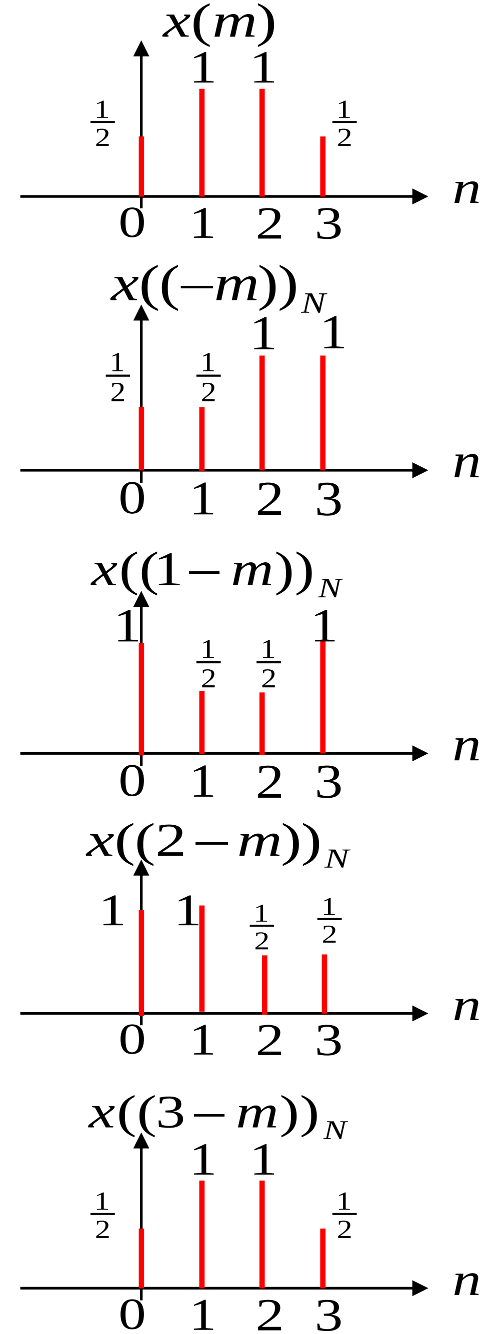
$$x((-m))_N=\{\tfrac{1}{2},\tfrac{1}{2},1,1\} \qquad y(0)=\frac{9}{4}$$

$$x((1-m))_N=\{1,\tfrac{1}{2},\tfrac{1}{2},1\} \qquad y(1)=2$$

$$x((2-m))_N=\{1,1,\tfrac{1}{2},\tfrac{1}{2}\} \qquad y(2)=\frac{9}{4}$$

$$x((3-m))_N=\{\tfrac{1}{2},1,1,\tfrac{1}{2}\} \qquad y(3)=\frac{5}{2}$$

$$y(n)=\{\tfrac{9}{4},2,\tfrac{9}{4},\tfrac{5}{2}\}$$



DFT圆周卷积的特性:

（1） 时域圆周卷积:

$$f(n)=x(n)\circledast y(n) \Rightarrow F(k)=X(k)Y(k)$$

$$\boxed{\tilde{f}(n)=\tilde{x}(n)\circledast\tilde{y}(n)} \Rightarrow \boxed{\tilde{F}(k)=\tilde{X}(k)\tilde{Y}(k)}$$

$$F(k)=\tilde{F}(k)R_N(k) =\tilde{X}(k)\tilde{Y}(k)R_N(k)$$

$$=\tilde{X}(k)R_N(k)\tilde{Y}(k)R_N(k)$$

$$=X(k)Y(k)$$

（2） 频域圆周卷积:

$$f(n)=x(n)y(n) \Leftrightarrow F(k)=\frac{1}{N}X(k)\circledast Y(k)$$

3 有限长序列的线性卷积与圆周卷积

$x(n)$ 长度为N　　$y(n)$ 长度为M

线性卷积： $f(n)=x(n)*y(n)=\sum_{m=-\infty}^{\infty}x(m)y(n-m)$

$$\left.\begin{array}{ll} x(n): & 0\le n\le N-1 \\ y(n): & 0\le n\le M-1 \end{array}\right\} \Longrightarrow f(n): \quad 0\le n\le N+M-2$$

两个有限长序列的线性卷积为一有限长序列： $N+M-1$

$$f(n)=x(n)\circledast y(n) \Longrightarrow F(k)=X(k)Y(k)$$

问题：能否利用圆周卷积计算线性卷积？

$x(n)$： 增加**L-N**个零值 $\Longrightarrow 0\le n\le L-1$

$y(n)$： 增加**L-M**个零值 $\Longrightarrow 0\le n\le L-1$

圆周卷积：

$$\tilde{y}(n)=\sum_{k=-\infty}^{\infty} y(n-kL) \qquad \tilde{y}(-m)=\sum_{k=-\infty}^{\infty} y(-m-kL)$$

$$y((n-m))_L=\tilde{y}(n-m)=\sum_{k=-\infty}^{\infty} y(n-m-kL)$$

$$\begin{aligned} f(n)=x(n)\circledast y(n) &= \sum_{m=0}^{L-1} x(m)\underline{y((n-m))_L} R_L(n) \\ &= \sum_{m=0}^{L-1} x(m)\underline{\sum_{k=-\infty}^{\infty} y(n-kL-m)R_L(n)} \\ &= \sum_{k=-\infty}^{\infty}\underline{\sum_{m=0}^{L-1} x(m)y(n-kL-m)R_L(n)} \\ &= \underline{\sum_{k=-\infty}^{\infty} f(n-kL)R_L(n)} \end{aligned}$$

结论：

1、 两个有限长序列的圆周卷积是将其线性卷积以L为周期延拓后的主值周期

2、 $\tilde{f}(n)R_L(n)$ 的前 **N+M-1** 点就是其线性卷积 $f(n)$

3、利用圆周卷积计算线性卷积，必须满足：$\boxed{L \geq N+M-1}$

4 共轭对称性(圆周共轭对称性)

$$x(n) \leftrightarrow X(k) \implies x^*(n) \leftrightarrow X^*(N-k)$$

$$x^*(n) \xrightarrow{\text{DFT}} \sum_{n=0}^{N-1} x^*(n) W_N^{kn} = [\sum_{n=0}^{N-1} x(n) W_N^{-kn}]^* = X^*(-k)$$

$$= X^*((-k))_N R_N(n) = X^*((N-k))_N R_N(n) = X^*(N-k)$$

$$k = 0, \quad X^*(N-k) = X^*(N) = X^*(0) \qquad (0 \le k \le N-1)$$

$$x(n) = x_{\text{Re}}(n) + j x_{\text{Im}}(n) \implies X(k) = X_{\text{e}}(k) + X_{\text{o}}(k)$$

$$x_{\text{Re}}(n) = \frac{1}{2}[x(n) + x^*(n)] \implies X_e(k) = \frac{1}{2}[X(k) + X^*(N-k)] \implies X_e(k) = X_e^*(N-k)$$

$$j x_{\text{Im}}(n) = \frac{1}{2}[x(n) - x^*(n)] \implies X_o(k) = \frac{1}{2}[X(k) - X^*(N-k)] \implies X_o(k) = -X_o^*(N-k)$$

圆周共轭偶对称：

$$X_e(k)=X_e^*(N-k) \Rightarrow \begin{cases} \mathrm{Re}[X_e(k)]=\mathrm{Re}[X_e(N-k)] & \text{圆周偶对称} \\ \mathrm{Im}[X_e(k)]=-\mathrm{Im}[X_e(N-k)] & \text{圆周奇对称} \end{cases}$$

圆周共轭奇对称：

$$X_o(k)=-X_o^*(N-k) \Rightarrow \begin{cases} \mathrm{Re}[X_o(k)]=-\mathrm{Re}[X_o(N-k)] & \text{圆周奇对称} \\ \mathrm{Im}[X_o(k)]=\mathrm{Im}[X_o(N-k)] & \text{圆周偶对称} \end{cases}$$

$$\left.\begin{aligned} &x(n)\text{为实序列：}\quad X(k)=X_e(k) \\ &x(n)\text{为虚序列：}\quad X(k)=X_o(k) \end{aligned}\right\}$$

利用对称性提高
DFT的运算效率

$$\boxed{x(n)=x_e(n)+x_o(n)}$$

圆周共轭偶部 + 圆周共轭奇部

$$x_e(n)=\frac{1}{2}[x(n)+x^*(N-n)] \Rightarrow \frac{1}{2}[X(k)+X^*(N-k)]=\mathrm{Re}[X(k)]$$

$$x_o(n)=\frac{1}{2}[x(n)-x^*(N-n)] \Rightarrow \frac{1}{2}[X(k)-X^*(N-k)]=j\,\mathrm{Im}[X(k)]$$

离散傅立叶变换的应用

1、在线性滤波中使用DFT

例：已知一个FIR 滤波器的单位脉冲响应为 $h(n)=\{1,2,3\}$。

利用DFT和IDFT计算该滤波器对输入序列 $x(n)=\{1,2,2,1\}$ 的响应。

$$y(n)=x(n)*h(n) \qquad y(n)=\{1,4,9,11,8,3\}$$

输入序列的长度为L=4，单位脉冲响应的长度为M=3， 这两个序列的线性卷积的长度为N=6，利用圆周卷积计算线性卷积，DFT大小至少为6。因此，将两个序列补零至6个点。

$$x(n)=\{1,2,2,1,0,0\} \qquad h(n)=\{1,2,3,0,0,0\}$$

$$y(n)=x(n)\otimes h(n)$$

利用DFT的圆周卷积性质

$$Y(k)=X(k)H(k)$$

可以利用DFT和IDFT计算该滤波器对输入序列的响应。

为简化起见，可计算8点DFT。

$$X(k)=\sum_{n=0}^{7}x(n)e^{-j2\pi kn/8}=1+2e^{-j\pi k/4}+2e^{-j\pi k/2}+e^{-j3\pi k/4},\quad k=0,1,\cdots,7$$

$$X(0)=6,\quad X(1)=\frac{2+\sqrt{2}}{2}-j(\frac{4+3\sqrt{2}}{2})$$

$$X(2)=-1-j,\quad X(3)=\frac{2-\sqrt{2}}{2}+j(\frac{4-3\sqrt{2}}{2})$$

$$X(4)=0,\quad X(5)=\frac{2-\sqrt{2}}{2}-j(\frac{4-3\sqrt{2}}{2})$$

$$X(6)=-1+j,\quad X(7)=\frac{2+\sqrt{2}}{2}+j(\frac{4+3\sqrt{2}}{2})$$

$$H(k)=\sum_{n=0}^{7}h(n)e^{-j2\pi kn/8}\quad=1+2e^{-j\pi k/4}+3e^{-j\pi k/2},\quad k=0,1,\cdots,7$$

$$H(0)=6,\quad H(1)=1+\sqrt{2}-j(3+\sqrt{2})$$

$$H(2)=-2-j2,\quad H(3)=1-\sqrt{2}+j(3-\sqrt{2})$$

$$H(4)=2,\quad H(5)=1-\sqrt{2}-j(3-\sqrt{2})$$

$$H(6)=-2+j2,\quad H(7)=1+\sqrt{2}+j(3+\sqrt{2})$$

$$Y(k)=X(k)H(k)$$

$$Y(0)=36,\quad Y(1)=-14.07-j17.48,\qquad Y(2)=j4,\quad Y(3)=0.07+j0.515,$$

$$Y(4)=0,\quad Y(5)=0.07-j0.515,\qquad Y(6)=-j4,\quad Y(7)=-14.07+j17.48,$$

最后，计算8点IDFT

$$y(n)=\sum_{k=0}^{7}Y(k)e^{j2\pi kn/8},\quad n=0,1,\cdots,7$$

$y(n)=\{1,4,9,11,8,3,0,0\}$　　　$y(n)$的前6点为求解的输出值。

2、长数据序列滤波：重叠相加法

利用DFT计算有限长序列的线性卷积是工程应用中相当有效的计算手段。但是在有些场合，例如对语音信号进行滤波时，需要做有限时宽序列与时宽很长序列间的线性卷积。理论上可以将整个信号存储下来，然后按照极大数量的样本点计算DFT，实现序列的卷积。这将产生两个问题：

（1）要求计算机的存储量过大

（2）等待输入的时间过长

解决此问题的基本思路：将长信号序列分割成固定尺寸的数据块。由于滤波是线性的，通过DFT，连续的数据块一次只处理一个，输出的数据块组合在一起就形成了总输出信号序列。

设FIR滤波器的长度为 M。输入数据序列分割成 L 点的数据块，不失一般性，假设 $L >> M$ 。

$$x(n) = \sum_{i=0}^{\infty} x_i(n) \qquad x_i(n) = \begin{cases} x(n), & iL \le n \le (i+1)L - 1 \\ 0, & \text{其它}n \end{cases}$$

$$y(n) = x(n) * h(n) = \sum_{i=0}^{\infty} x_i(n) * h(n)$$

因此每一个$x_i(n) * h(n)$ 都可用DFT计算线性卷积的方法计算。由于$x_i(n) * h(n)$为 $L+M-1$点，故先对 $x_i(n)$和 $h(n)$补零值点，补到$N \geq L+M-1$，然后做 N点DFT和IDFT。

取 $N = L+M-1$。

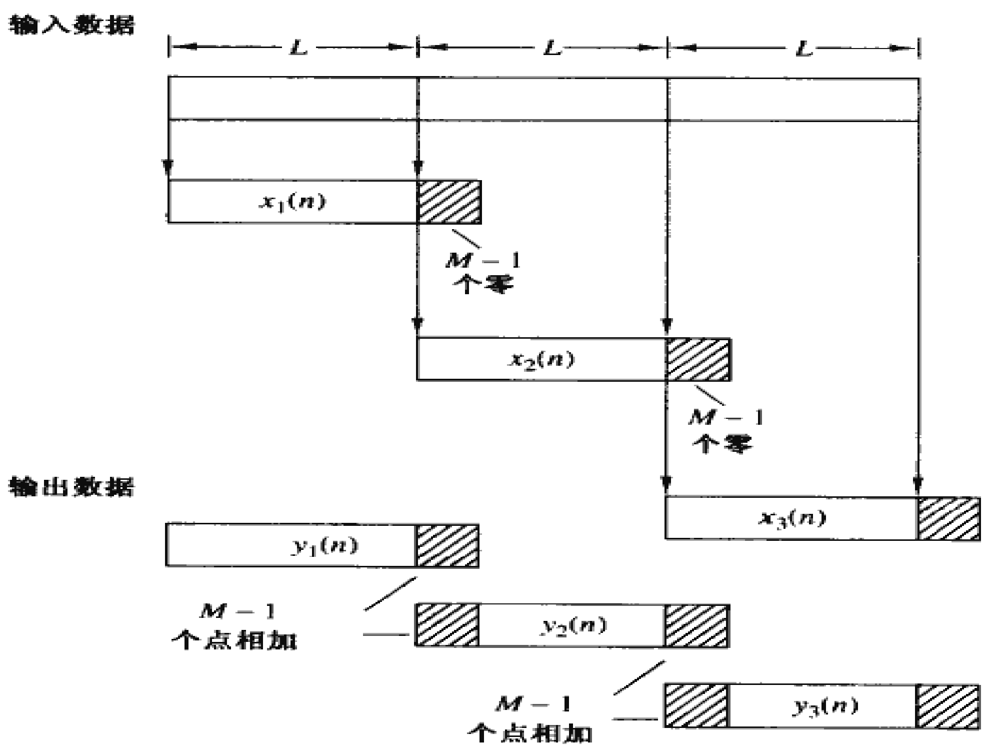
$$x_1(n) = \{x(0), x(1), \cdots, x(L-1), \underbrace{0,0,\cdots 0}_{M-1\text{个零}}\}$$

$$x_2(n) = \{x(L), x(L+1), \cdots, x(2L-1), \underbrace{0,0,\cdots 0}_{M-1\text{个零}}\}$$

$$x_3(n) = \{x(2L), \cdots, x(3L-1), \underbrace{0,0,\cdots 0}_{M-1\text{个零}}\}$$

$$Y_m(k) = H(k) X_m(k), \quad k = 0,1,\cdots, N-1$$

因为DFT和IDFT的长度为 $N = L+M-1$，并且通过对每个块补零以使序列长度增加到N 点，所以IDFT 得到的数据块长度也是N，而且不存在混叠。



利用重叠相加法进行线性 FIR 滤波

$$y(n)=\{y_1(0),y_1(1),\cdots,y_1(L-1),y_1(L)+y_2(0),y_1(L+1)+y_2(1),\cdots,$$
$$y_1(N-1)+y_2(M-1),y_2(M),\cdots\}$$

每个数据块以 $M-1$ 个零作为结尾，所以每个输出块的后 $M-1$ 个点必须要重叠并加到随后数据块的前 $M-1$ 个点上。

因此，这种长数据序列滤波方法称为**重叠相加法**。

快速傅立叶变换（FFT）

$$\begin{cases} X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} & 0 \le k \le N-1 \\ x(n) = \dfrac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn} & 0 \le n \le N-1 \end{cases}$$

计算 $X(k)$ 的一个点：　N次复数乘法，N-1次复数加法

计算 $X(k)$ 的全部N点：　N^2次复数乘法，N(N-1)次复数加法

DFT的运算复杂度：　N^2 数量级

$N \log_2 N$

DFT的运算特点：

1. W_N^{kn} 具有周期性

$W_N^{kn} = e^{-j\frac{2\pi}{N}kn}$ 关于 n或k 都是以N 为周期的

$$W_N^{k(n+N)} = W_N^{n(k+N)} = W_N^{kn}$$

2. W_N^{kn} 具有对称性

$$W_N^{k(N-n)} = W_N^{n(N-k)} = (W_N^{kn})^*$$

$W_N^N=1$ $\quad W_N^{N/2} = -1$ $\quad W_N^{(k\pm N/2)} = -W_N^k$ $\quad W_{N/2}^{kn} = W_N^{2kn}$

FFT算法的基本思想:

将长序列的DFT逐步分解成短序列的DFT, 并利用 W_N^{kn} 的周期性和对称性进行某些组合, 以减少运算量

基-2算法: 假定序列长度N是2的整数幂 $N = 2^M$

1. 按时间抽取的FFT算法 (Cooley-Tukey)

$$x(n) \to \begin{cases} x_1(r) = x(2r) \\ x_2(r) = x(2r+1) \end{cases} \qquad r = 0,1,\cdots,\tfrac{N}{2}-1$$

2. 按频率抽取的FFT算法 (Sand-Tukey)

$$x(n) \to \begin{cases} x_1(r) \\ x_2(r) = x_1(r + \dfrac{N}{2}) \end{cases} \qquad r = 0,1,\cdots,\tfrac{N}{2}-1$$

按时间抽取的FFT算法

$$x(n) \rightarrow \begin{cases} x_1(r) = x(2r) \\ x_2(r) = x(2r+1) \end{cases} \qquad r = 0,1,\cdots,\tfrac{N}{2}-1$$

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} = \sum_{r=0}^{\frac{N}{2}-1} x(2r) W_N^{2rk} + \sum_{r=0}^{\frac{N}{2}-1} x(2r+1) W_N^{(2r+1)k}$$

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

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

$$X(k) = X_1(k) + W_N^{k} X_2(k) \qquad 0 \le k \le \tfrac{N}{2}-1$$

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

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

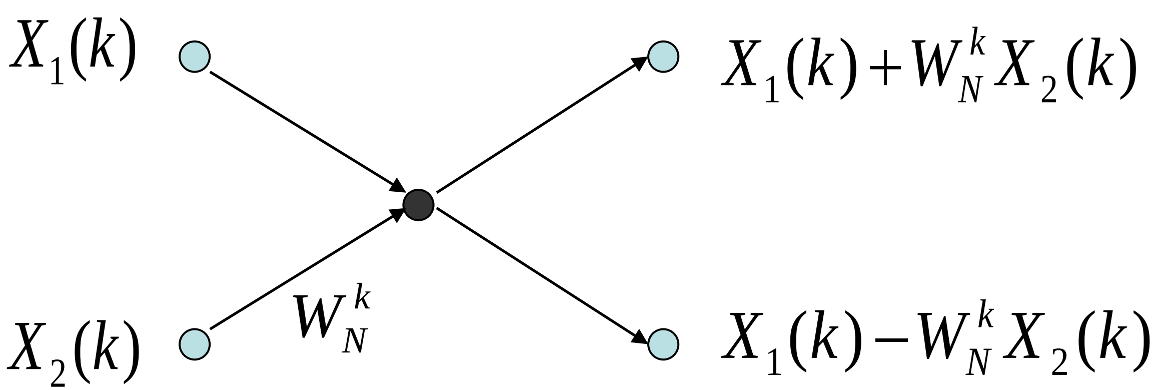
$$X_2(k+\tfrac{N}{2})=X_2(k)$$

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

$$X(k+\tfrac{N}{2})=X_1(k+\tfrac{N}{2})+W_N^{k+\frac{N}{2}}X_2(k+\tfrac{N}{2})=X_1(k)-W_N^kX_2(k)$$

$$\begin{cases}X(k)=X_1(k)+W_N^kX_2(k)\\[2ex] X(k+\frac{N}{2})=X_1(k)-W_N^kX_2(k)\end{cases}\qquad (0\le k\le \tfrac{N}{2}-1)$$

蝶形结运算



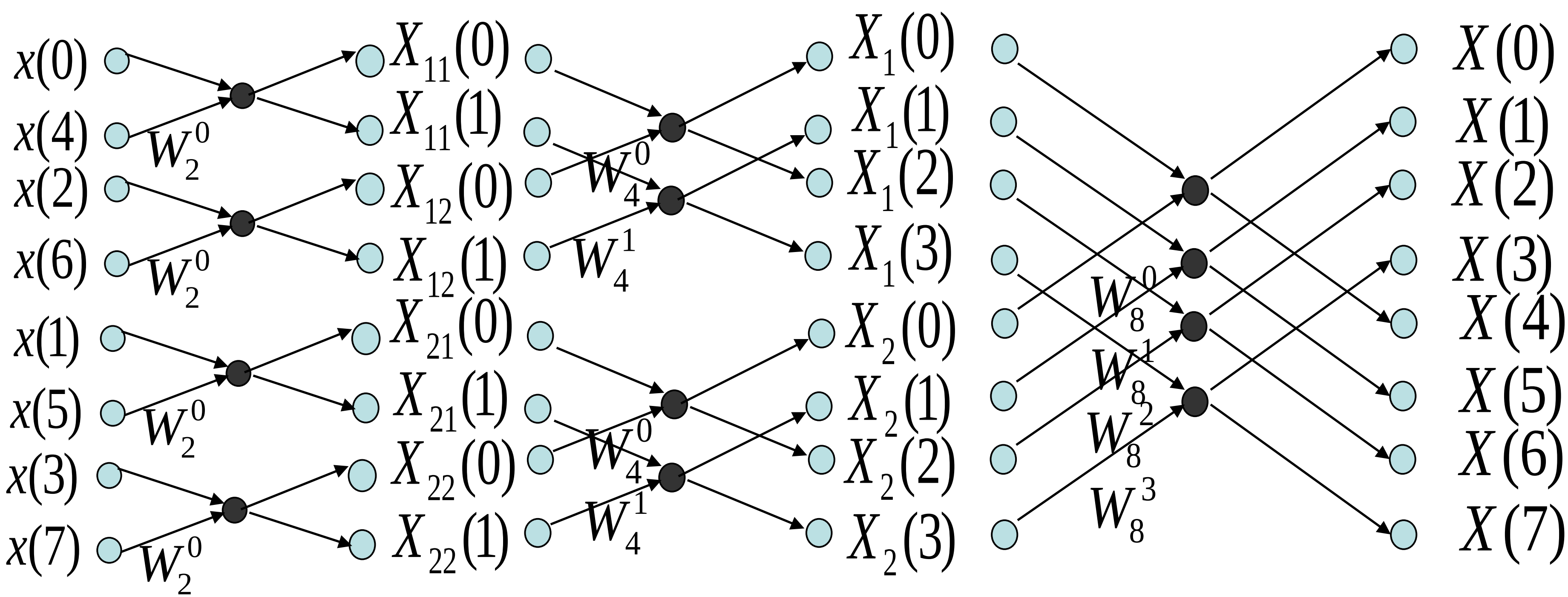
对于 $N=2^M$， **经过M级分解, 可使每一组只有两点**

$$X(k)=\sum_{n=0}^{1}x(n)W_2^{kn}$$

$$X(0)=x(0)+x(1)$$
$$X(1)=x(0)-x(1)$$

例: $N=8,\quad M=3$

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



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

FFT的运算复杂度： $N\log_2 N$ **数量级**

每一级蝶形结的个数： $\dfrac{N}{2}$

每一蝶形结的运算量： **1次复数乘 + 2次复数加**

M 级蝶形运算：

$$\begin{cases} \text{复数乘} & \dfrac{N}{2}\cdot M = \dfrac{N}{2}\log_2 N \\ \text{复数加} & N\log_2 N \end{cases}$$

$\mathrm{DFT} \to \mathrm{FFT}$： $N^2 \to N\log_2 N$

2. 按频率抽取的FFT算法 (Sand-Tukey)

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

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

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

$k=2r:$

$$X(2r)=\sum_{n=0}^{\frac{N}{2}-1}[x(n)+x(n+\tfrac{N}{2})]W_N^{2rn}=\sum_{n=0}^{\frac{N}{2}-1}\underbrace{[x(n)+x(n+\tfrac{N}{2})]}_{x_1(n)}W_{\frac{N}{2}}^{rn}$$

$$(0\le r\le \tfrac{N}{2}-1)$$

$k=2r+1:$

$$X(2r+1)=\sum_{n=0}^{\frac{N}{2}-1}[x(n)-x(n+\tfrac{N}{2})]W_N^{(2r+1)n}=\sum_{n=0}^{\frac{N}{2}-1}\underbrace{[x(n)-x(n+\tfrac{N}{2})]W_N^{n}}_{x_2(n)}W_{\frac{N}{2}}^{rn}$$

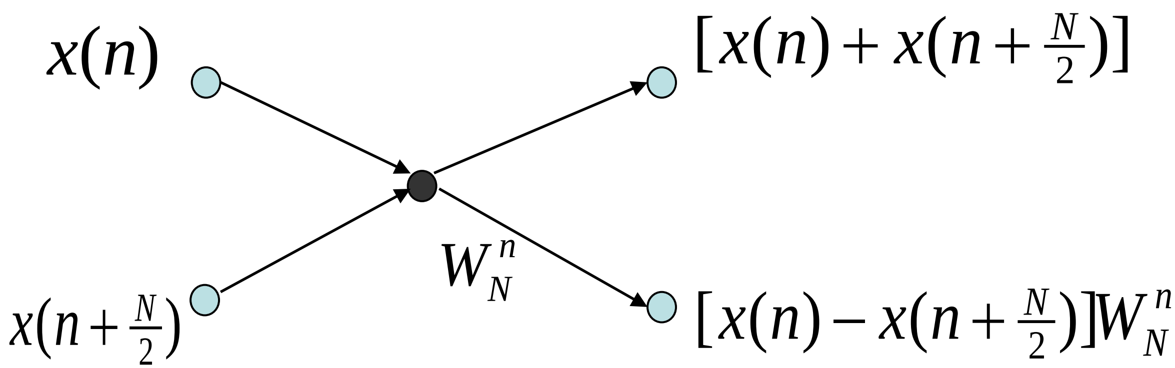
$$(0\le r\le \tfrac{N}{2}-1)$$

如果令: $x_1(n) = [x(n) + x(n+\frac{N}{2})]$

$$x_2(n) = [x(n) - x(n+\tfrac{N}{2})]W_N^n$$

则: $X(2r) = \text{DFT}\ [x_1(n)]$

$$X(2r+1) = \text{DFT}\ [x_2(n)]$$



IDFT的快速算法(IFFT):

$$\begin{cases} X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn} & 0\le k\le N-1 \\ x(n)=\frac{1}{N}\sum_{k=0}^{N-1}X(k)W_N^{-kn} & 0\le n\le N-1 \end{cases}$$

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

$$=\frac{1}{N}[\sum_{k=0}^{N-1}X^*(k)W_N^{kn}]^*$$

$$=\frac{1}{N}(\mathrm{DFT}[X^*(k)])^*$$

N为复合数的FFT（混合基算法）

基-2算法是以2为基数的FFT算法，即$N=2^M$。
该方法的优点：程序简单，效率高，使用方便。

实际使用时，有限长序列的长度N很大程度上有认为因素确定，多数场合可取$N=2^M$。

如果N的数值不是以2为基数的整数次幂，处理方法有两种：

1、**补零**：将x(n)补零，使$N=2^M$

例如，N=30，补x(30)=0，x(31)=0，使得$N=32=2^5$计算，直接采用以2为基数M=5的FFT程序。

有限长序列补零后并不影响其频谱$X(e^{j\omega})$，在许多场合这种处理是可接受的。

2、如要求准确的N点DFT值，可采用任意数为基数的FFT，其计算效率低于基-2FFT算法。

如果N为复合数，可分解为两个整数P与Q的乘积，其基本思想是将DFT的运算量尽量分小。因此，在N=PQ情况下，希望将N点的DFT分解为P个Q点DFT或Q个P点DFT，以减少计算量。

步骤: $\begin{cases} n = n_1 Q + n_0 \\ k = k_1 P + k_0 \end{cases}$

n_0，k_1分别为0，1，...，*Q*-1；

n_1，k_0分别为0，1，...，*P*-1。

N点DFT可以重新写成

$$X(k) = X(k_1P + k_0) = X(k_1, k_0)$$

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

$$= \sum_{n_0=0}^{Q-1} \sum_{n_1=0}^{P-1} x(n_1Q + n_0) W_N^{(k_1P+k_0)(n_1Q+n_0)}$$

$$X(k_1, k_0) = \sum_{n_1=0}^{Q-1} \sum_{n_0=0}^{P-1} x(n_1, n_0) W_N^{k_1n_1PQ} W_N^{k_0n_1Q} W_N^{k_1n_0P} W_N^{k_0n_0}$$

$$= \sum_{n_0=0}^{Q-1} \sum_{n_1=0}^{P-1} x(n_1, n_0) W_N^{k_0n_1Q} W_N^{k_1n_0P} W_N^{k_0n_0}$$

考虑到 $W_N^{k_0 n_1 Q} = W_P^{k_0 n_1}$，$W_N^{k_1 n_0 P} = W_Q^{k_1 n_0}$

$$X(k_1, n_0) = \sum_{n_0=0}^{Q-1} \left\{ \left[\sum_{n_1=0}^{P-1} x(n_1, n_0) W_P^{k_0 n_1} \right] W_N^{k_0 n_0} \right\} W_Q^{k_1 n_0}$$

令 $$X_1(k_0, k_1) = \sum_{n_1=0}^{P-1} x(n_1, n_0) W_P^{k_0 n_1}$$

$$X(k_1, k_0) = \sum_{n_0=0}^{Q-1} \left[X_1(k_0, n_0) W_N^{k_0 n_0} \right] W_Q^{k_1 n_0}$$

再令 $$X_1'(k_0, n_0) = X_1(k_0, n_0) W_N^{k_0 n_0}$$

$$X_2(k_0, k_1) = \sum_{n_0=0}^{Q-1} X_1'(k_0, n_0) W_Q^{k_1 n_0} \qquad X(k_1, k_0) = X_2(k_0, k_1)$$

以P=3，Q=4，N=12为例：

（1）先将$x(n)$通过$x(n_1Q+n_0)$改写成$x(n_1,n_0)$。

因为Q=4，n_1=0,1,2，n_0=0,1,2,3，故可得：

$x(0,0)=x(0)$ $x(0,1)=x(1)$ $x(0,2)=x(2)$ $x(0,3)=x(3)$

$x(1,0)=x(4)$ $x(1,1)=x(5)$ $x(1,2)=x(6)$ $x(1,3)=x(7)$

$x(2,0)=x(8)$ $x(2,1)=x(9)$ $x(2,2)=x(10)$ $x(2,3)=x(11)$

（2）求Q个P点的DFT

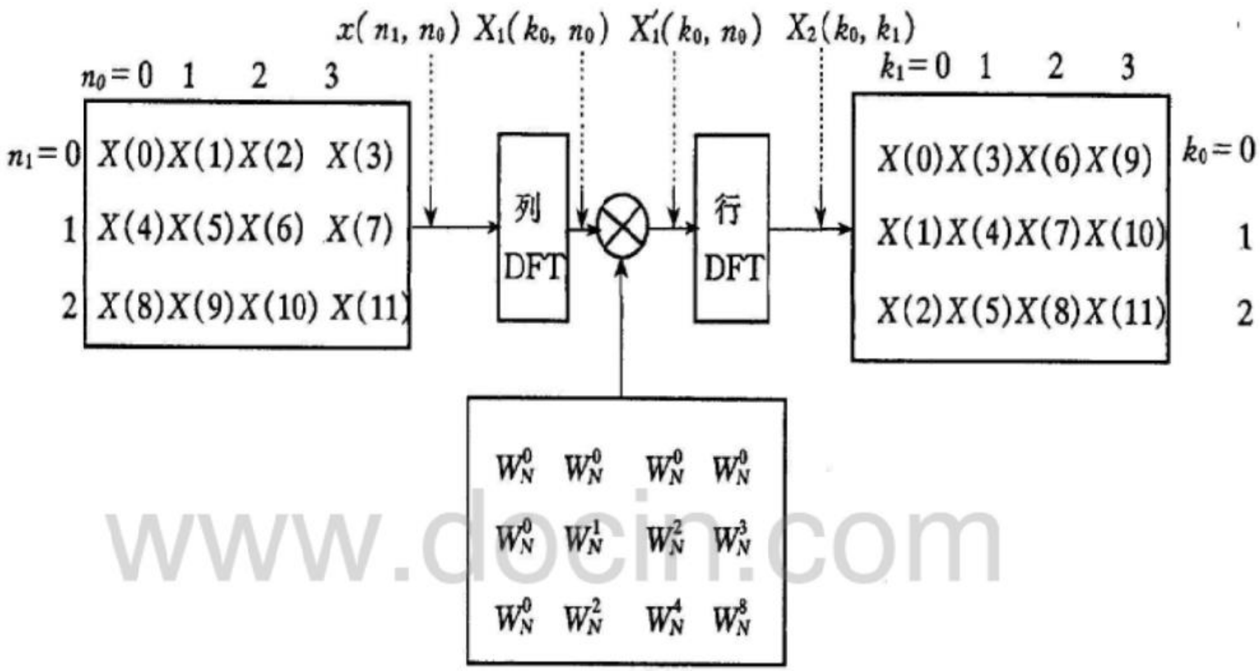
$$X_1(k_0,n_0)=\sum_{n_1=0}^{2}x(n_1,n_0)W_3^{k_0n_1}$$

（3）$X_1(k_0,n_0)$乘以$W_N^{k_0n_0}$得到$X_1'(k_0,n_0)$

（4）求P个Q点的DFT,参变量是k_0

$$X_2(k_0,k_1)=\sum_{n_0=0}^{3}X_1'(k_0,n_0)W_4^{k_1n_0}$$

（5）将$X_2(k_0,k_1)$通过$X(k_0+k_1P)$恢复为$X(k)$



计算量估算：

（1）求Q个P点DFT需要QP^2次附属乘和$QP(P-1)$复数加；

（2）乘N个W因子需要N次复数乘；

（3）求P个Q点DFT需要PQ^2次复数乘和$PQ(P-1)$次复数加

总的复数乘量：$QP^2+N+PQ^2=N(P+Q+1)$

总的复数加量：$QP(P-1)+PQ(Q-1)=N(P+Q-2)$

例：$N=23*29=667$，$N^2=444889$，$N(P+Q+1)=35351$

上述分解原则可以推广至任意技术的更加复杂的情况。

例如，如果N可分解为m个质数因子p1,p2,...pm，即N=p1p2p3...pm，则

第一步：可先把N分解为两个因子N=p1q1,q1=p2p3...pm,并用上述讨论方法将DFT分解为p1个q1点DFT；

第二步：将q1分解为q1=p2q2,q2=p3p4...pm,然后将每个q1点DFT再分解为p2个q2点DFT；

依次类推，通过m次分解，一直分到最少点数的DFT运算，从而获得最高的运算效率。其运算量近视为N(p1+p2+...+pm)次复数乘和复数加。但计算效率的提高是以编程的复杂性为代价的，一般较少用。

当p1=p2=...=pm=2,为基-2FFT算法。

当p1=p2=...=pm=4,为基-4FFT算法。

作业：

7.3 (a) (c) (e)

7.20