快速傅立叶变换 (FFT)

$$Y_i = \sum_{k=0}^{n-1} X_k \omega_n^{ki}, 0 \leqslant i < n$$

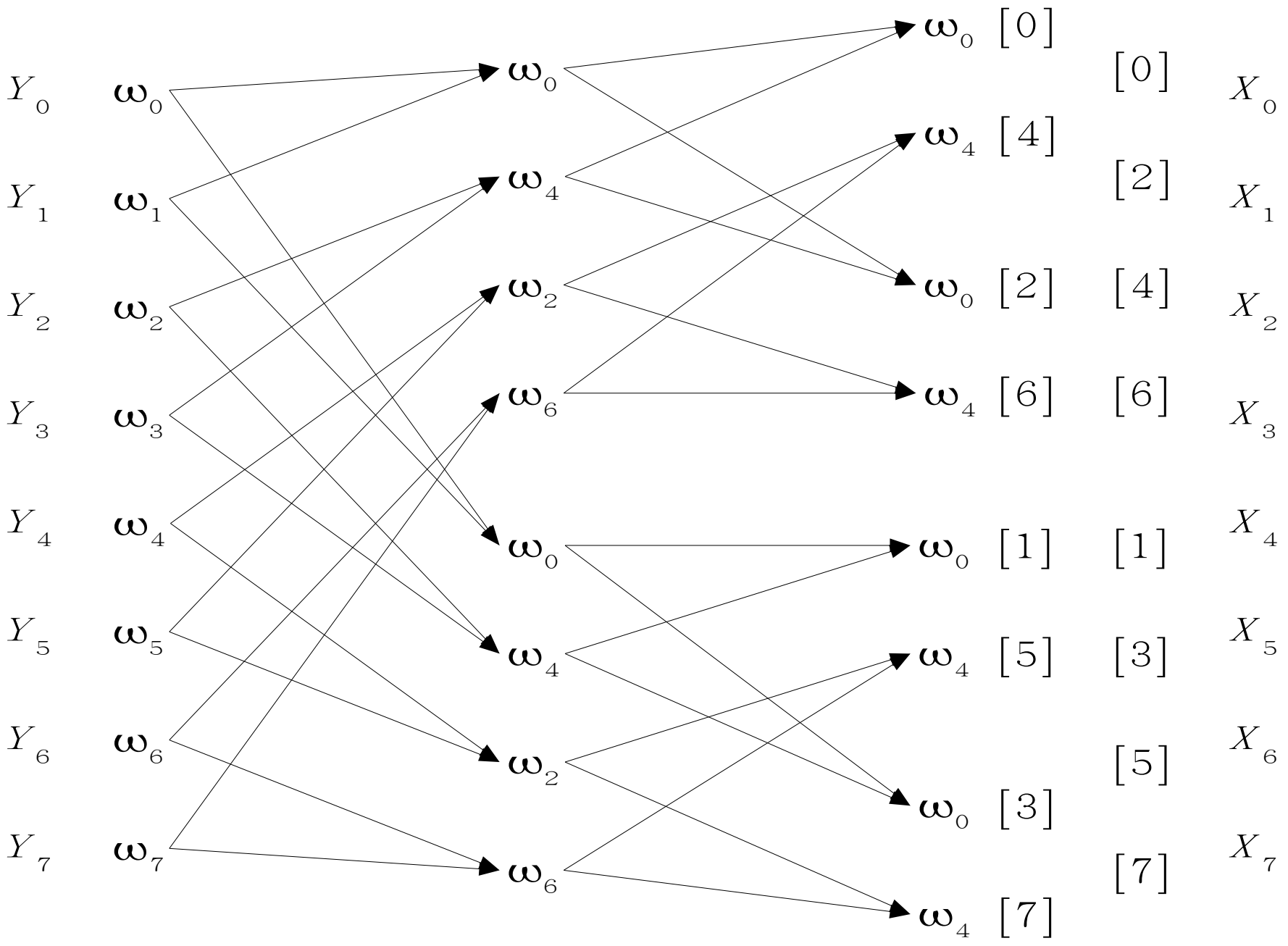
$$\omega_n = e^{2\pi\sqrt{-1}/n}$$

$$\begin{aligned}
Y_i &= \sum_{k=0}^{n/2-1} X_{2k} \omega_n^{2ki} + \sum_{k=0}^{n/2-1} X_{2k+1} \omega_n^{(2k+1)i} \\
&= \sum_{k=0}^{n/2-1} X_{2k} e^{2ki2\pi i/n} + \sum_{k=0}^{n/2-1} X_{2k+1} \omega_n^{i} e^{2ki2\pi i/n} \\
&= \sum_{k=0}^{n/2-1} X_{2k} e^{2\pi ikii/(n/2)} + \omega_n^{i} \sum_{k=0}^{n/2-1} X_{2k+1} e^{2\pi ikii/(n/2)} \\
&= \sum_{k=0}^{n/2-1} X_{2k} \omega_{n/2}^{ki} + \omega_n^{i} \sum_{k=0}^{n/2-1} X_{2k+1} \omega_{n/2}^{ki}
\end{aligned}$$

FFT 的算法（递归型）

```
procedure FFT(X, Y, n, omega)

    if (n=1) then Y[0]=x[0] e1se
    FFT((X[0], X[2], ..., X[n-2]),
        (Q[0], Q[1], ..., Q[n/2]), n/2, omega^2)
    FFT((X[1], X[3], ..., X[n-1]),
        (T[0], T[1], ..., T[n/2]), n/2, omega^2)
    for k=0,n-1 do
       Y[k]=Q[k%(n/2)]+omega^i T[k%(n/2)];
    end do
end procedure
```

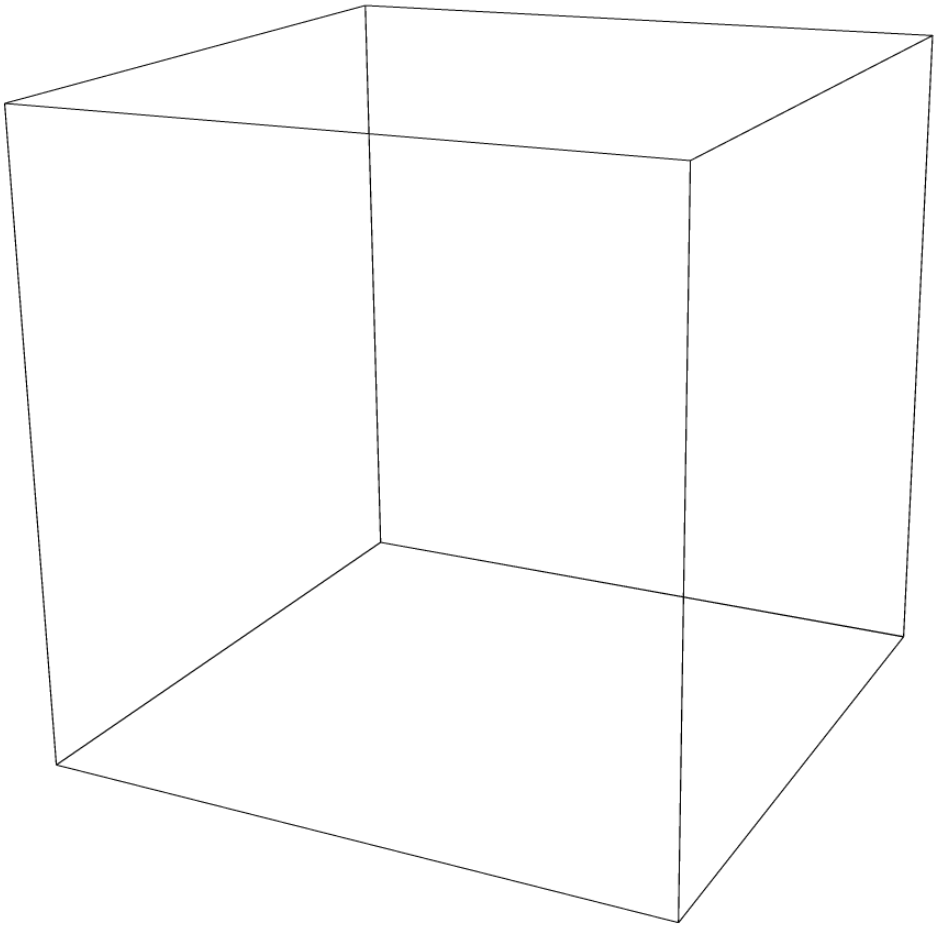

Y_0 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6 Y_7
ω_0 ω_1 ω_2 ω_3 ω_4 ω_5 ω_6 ω_7
ω_0 ω_4 ω_2 ω_6 ω_0 ω_4 ω_2 ω_6
ω_0 [0]
ω_4 [4]
ω_0 [2]
ω_4 [6]
ω_0 [1]
ω_4 [5]
ω_0 [3]
ω_4 [7]
[0] [4] [2] [6] [1] [3] [5] [7]
X_0 X_1 X_2 X_3 X_4 X_5 X_6 X_7

FFT 的算法（循环型）

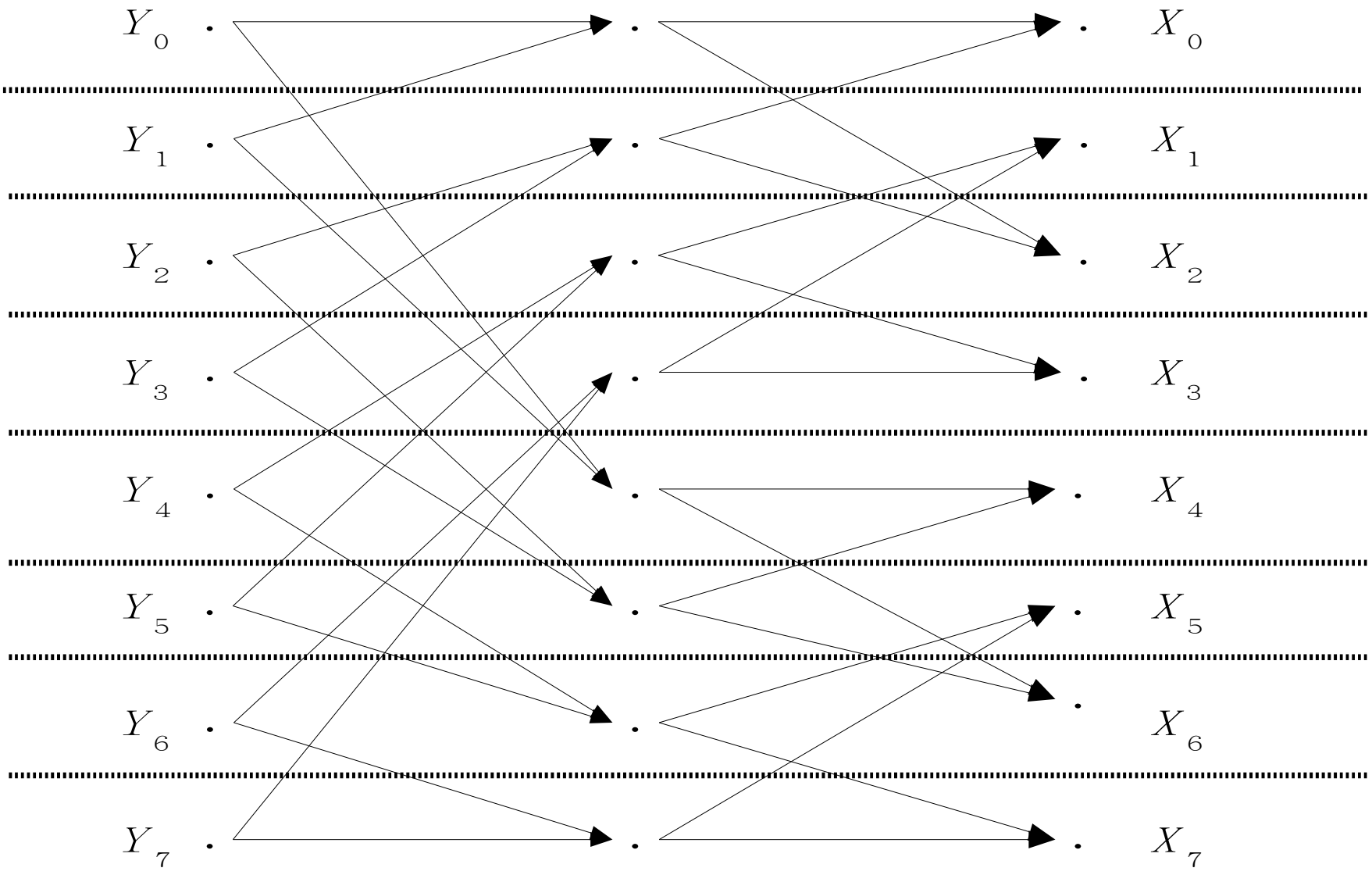
```
procedure FFT(X,Y,n)
    r = log(n);
    for i=0 to n-1 do R[i]=X[i]; end do
    for m=0 to r-1 do
        for i=0 to n-1 do S[i]=R[i]; end do
        for i=0 to n-1 do
            // 假设 i 具有二进制形式 (b_0 b_1 ··· b_{r-1})
            j := (b_0 ··· b_{m-1} 0b_{m+1} ··· b_{r-1});
            k := (b_0 ··· b_{m-1} 1b_{m+1} ··· b_{r-1});
            R[i] = S[j] + S[k] omega^(b_m b_{m-1} ··· b_0 0···0);
        end do
    end do
    for i=0 to n-1 do Y[i]=R[i]; end do
end procedure
```

Binary-Exchange Algorithm

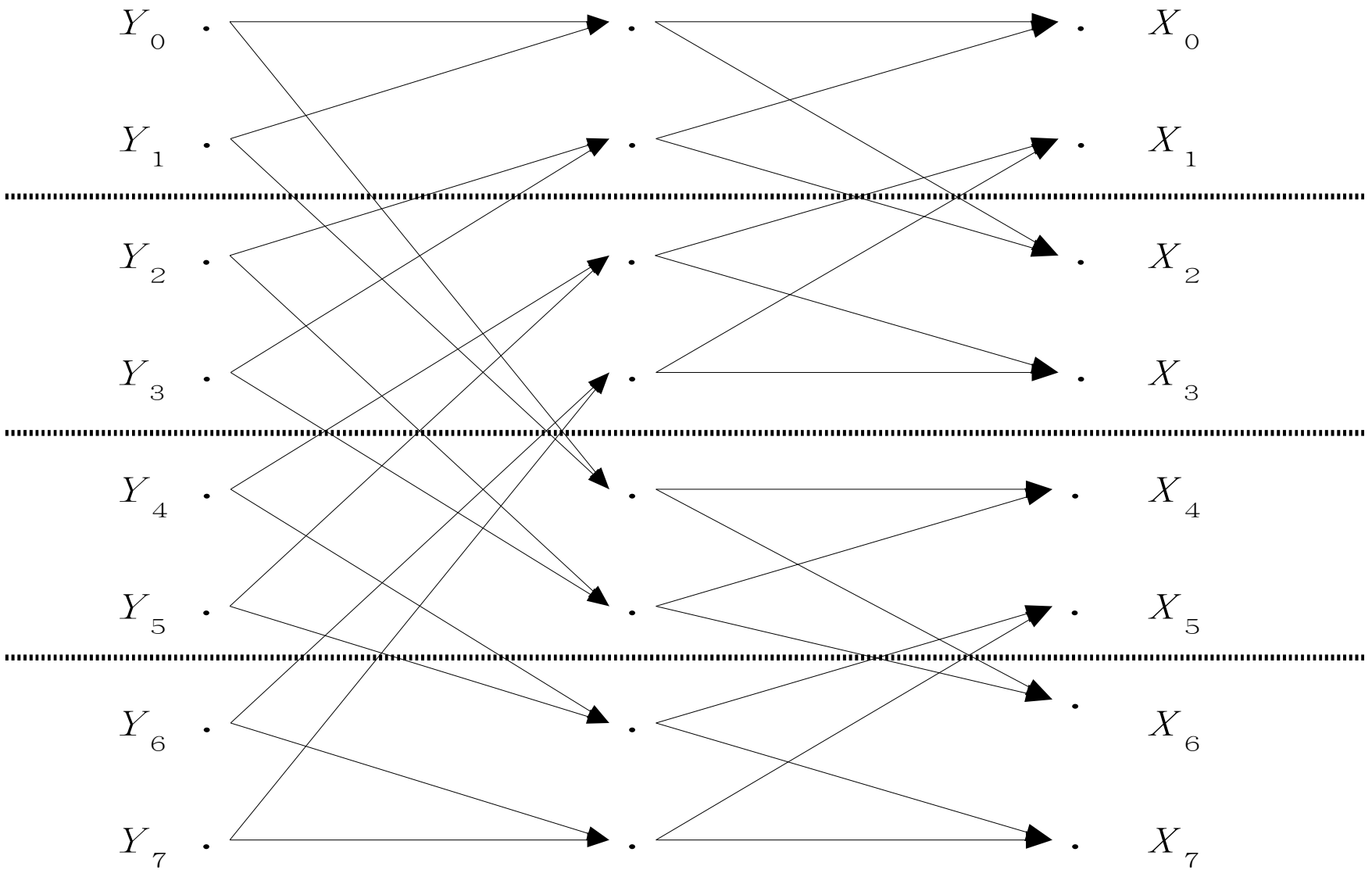
在超立方体结构的并行机上



n 个进程

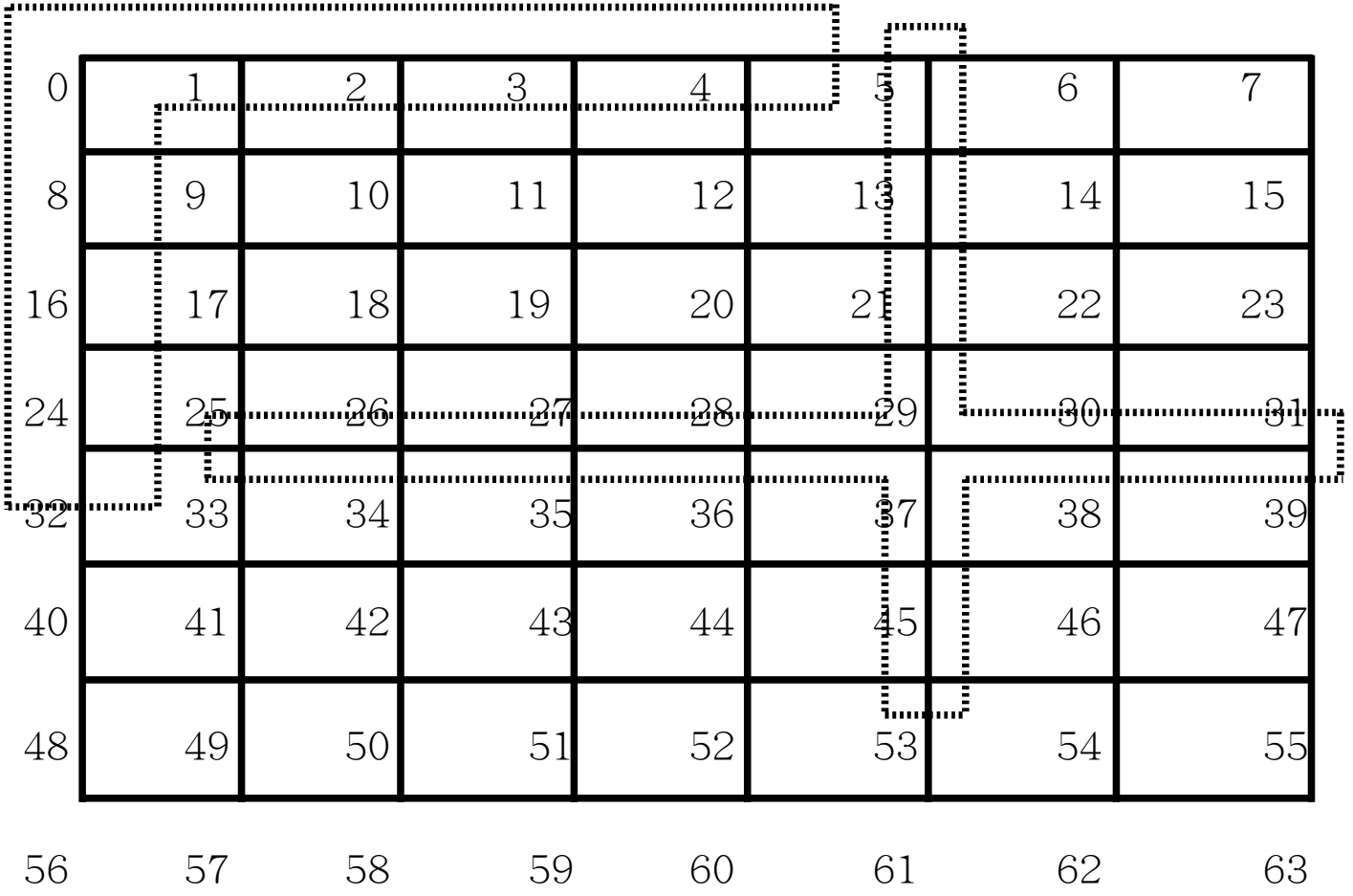


p 个进程($p=2^d<2^r=n$)



Binary-Exchange Algorithm

网格点结构的并行机



Binary-Exchange Algorithm

在网络带宽足够时，表现是最优的；

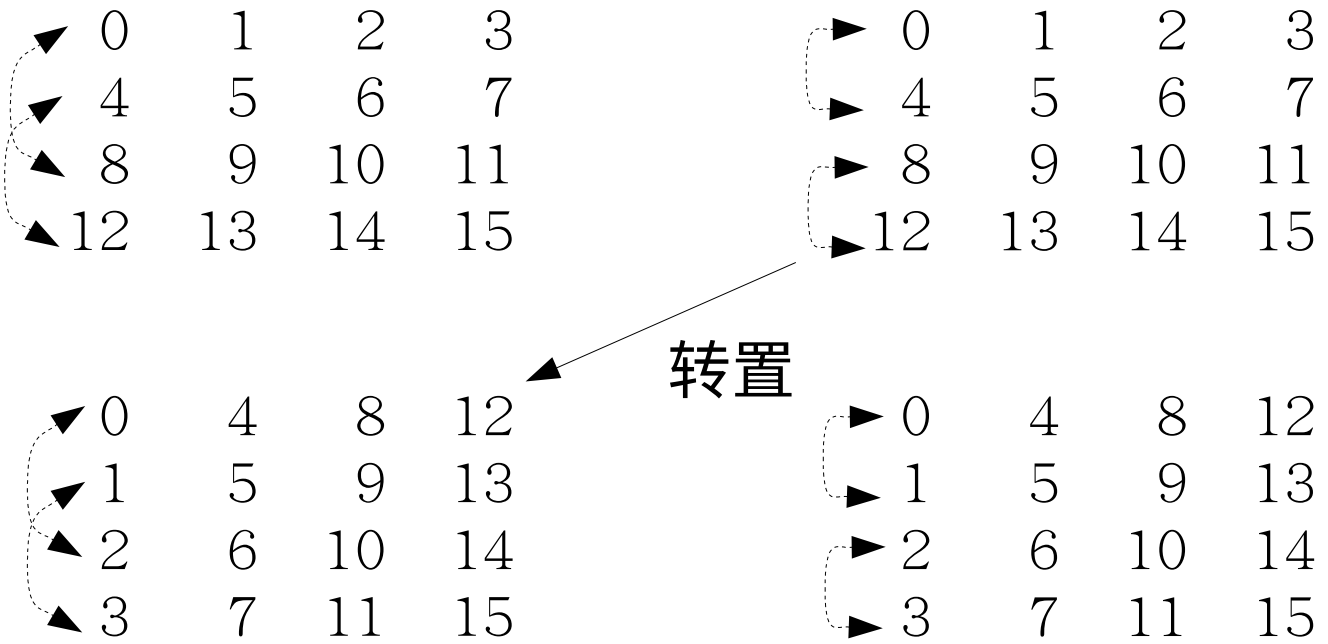
在带宽不足时，情况不太好，网络是瓶颈；

存在一个网络带宽阈值，在高于这个阈值时，算法表现优秀，在低于这个阈值时，表现直线下降。

转置算法
Transpose Algorithm

这个算法的特点在于对于网络的依赖要比 Binary Exchange Algorithm 要低，在网络带宽够大时，表现不如 Binary Exchange Algorithm，但是在网络带宽较小时，表现比 Binary Exchange Algorithm 要好得多。

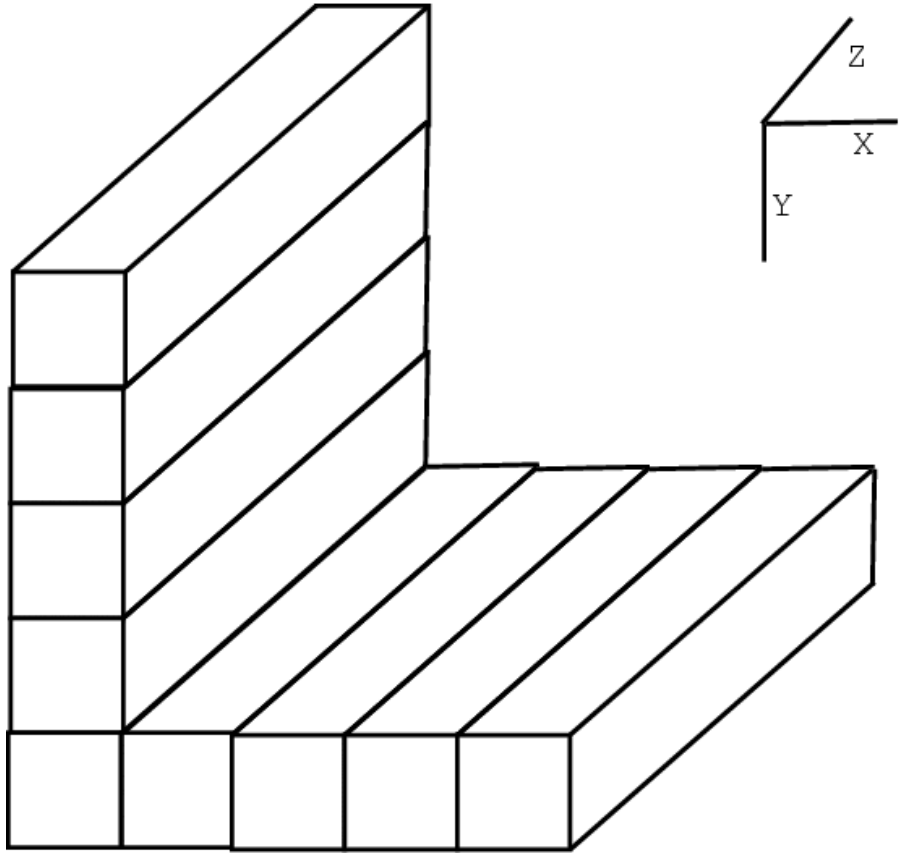
将总共$n=\sqrt{n}\times\sqrt{n}$,$\sqrt{n}=2^{r/2}$个数据按列分布在$\sqrt{n}$个进程上，前面一半的步骤事实上是完成每个进程上的$\sqrt{n}$个数据进行Fourier变换，后面的$\sqrt{n}$个步骤是对同一个行上的$\sqrt{n}$个数据进行Fourier变换，从而只是需要在中间进行一个转置操作使得两个操作能连接起来。



当进程个数 p 小于 $\sqrt{n}$ 时，每个进程上分配 $\frac{\sqrt{n}}{p}$ 个列,从而每个进程上只是进行本进程上数据的的 Fourier 变换，主要也是要进行一次转置。

Generalized Transpose Algorithm

数据按照 $n^{1/3} \times n^{1/3} \times n^{1/3}$ 的方式分布



算法：

1. 沿 Z 方向做局部 Fourier 变换；
2. 沿 XZ 平面进行矩阵转置操作；
3. 沿 Z 方向做局部 Fourier 变换；
4. 沿 YZ 平面进行矩阵转置操作；
5. 沿 Z 方向做局部 Fourier 变换；

并行程度更高，但是对于网络的依赖性略大一些。