清華大學
Tsinghua University

互连与通信
(Interconnection and Communication)

汪东升（Prof. Dongsheng Wang）

wds@tsinghua.edu.cn

互连与通信

- 互连网络概念
- 静态网络
- 动态网络
- 通信问题

互连网络

定　　义：由开关元件按一定拓扑结构和控制方式构成的网络以实现计算机系统内部多个处理机或多个功能部件间的相互连接

操作方式：

同步通信（Synchronous Communication)

异步通信（Asynchronous Communication)

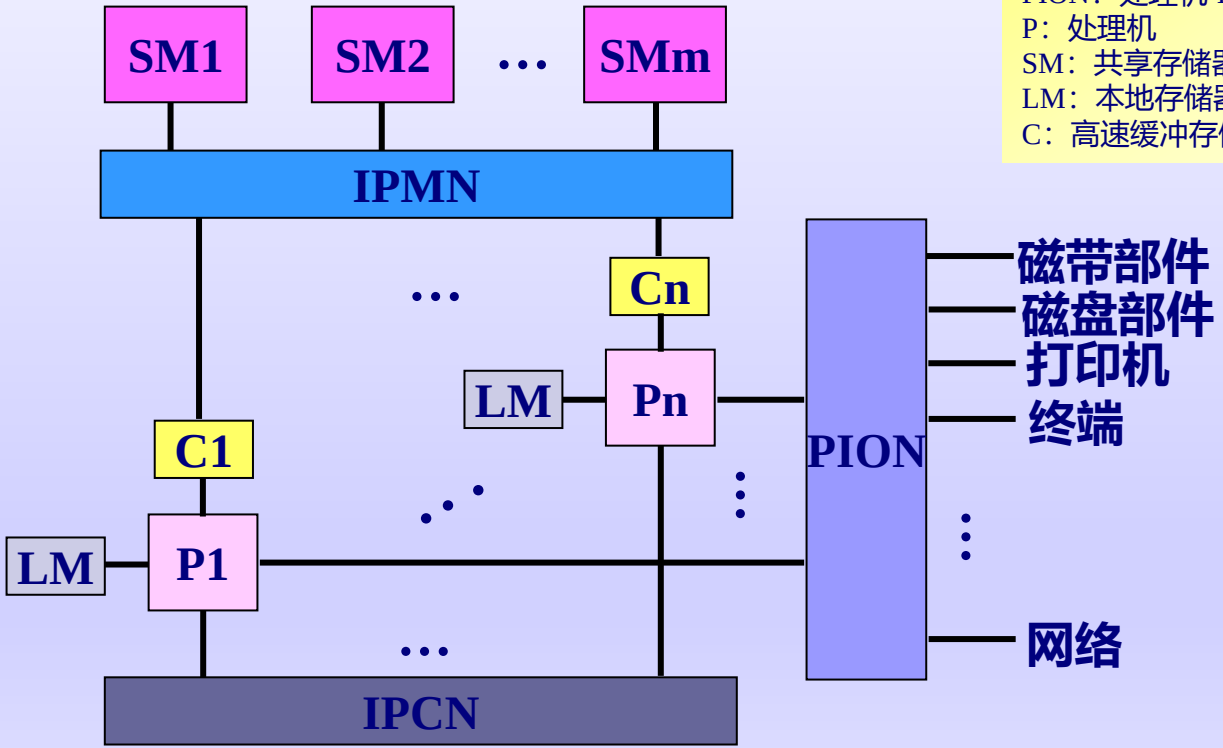
控制策略：

集中控制(Centralized control)

分布控制(Distributed control)

一般多处理机系统的互连结构

IPMN：处理机-存储器网络
IPCN：处理机间网络
PION：处理机-I/O网络
P：处理机
SM：共享存储器
LM：本地存储器
C：高速缓冲存储器



交换方式：

- 电路交换(Circuit switching)
  - 源结点和目的结点之间的物理通路在整个数据传送期间一直保持连接。
- 分组交换(Packet switching)
  - 把信息分割成许多组（又称为包），将它们分别送入互连网络。这些数据包可以通过不同的路径传送，到达目的结点后再拼合成原来的数据。结点之间不存在固定连接的物理通路。
- Wormhole交换(Wormhole switching)

网络拓扑结构：

静态网络(Static network)

动态网络(Dynamic network)

互连函数

排列：N个数的每一种有确定次序的放置方法叫做一个N排列

置换：把一个N排列变成另一个N排列的变换叫做N阶置换。

在有N个输入端和N个输出端的网络中，输入端和输出端的连接关系可以用置换来表示（输入端与输出端一一对应）。

一些常见的置换方式可以用下面的函数表示：

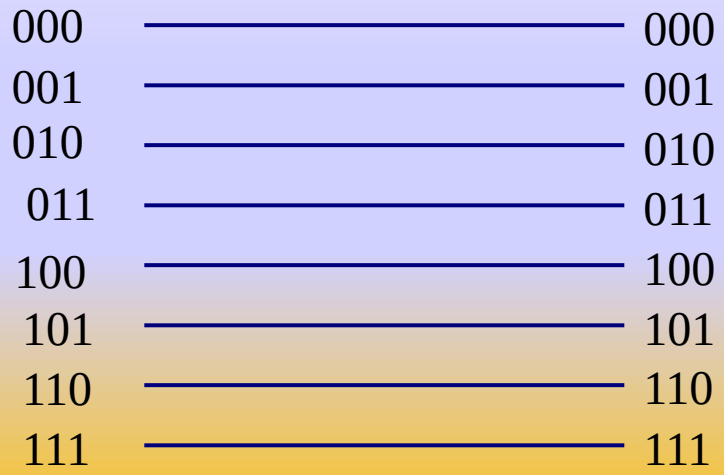
循环表示法：（$X_0, X_1, X_2 \ldots X_n$）

f(X0)=x1, f(X1)=X2,… f(Xn-1)=Xn

1. 恒等函数

$$f_e(X_{n-1}X_{n-2}\cdots X_k\cdots X_0)=(X_{n-1}X_{n-2}\cdots X_k\cdots X_0)$$

其中，$X_{n-1}X_{n-2}\ldots X_k\ldots X_0$是PE的地址（通常为二进制）。n为3时的恒等函数的连接情形如下：

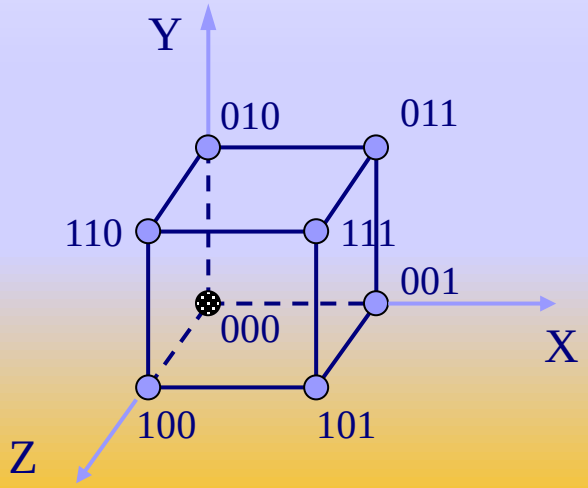


2. 方体函数（$cube_0$，$cube_1$，…，$cube_{n-1}$）

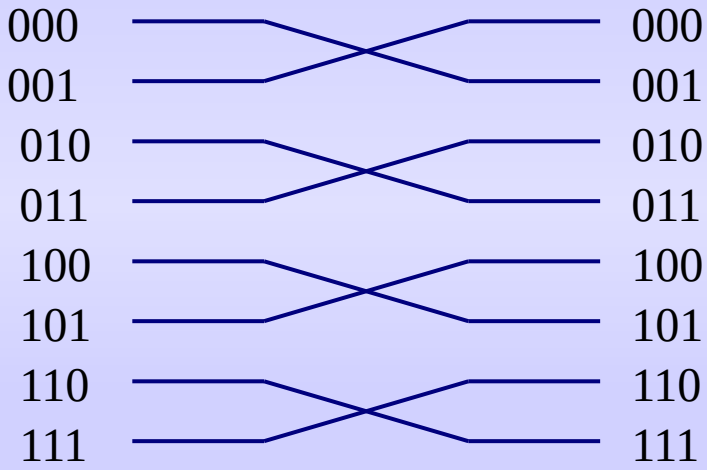
$$cube_k(X_{n-1}X_{n-2}\cdots X_k\cdots X_0)=(X_{n-1}X_{n-2}\cdots \overline{X_k}\cdots X_0)$$

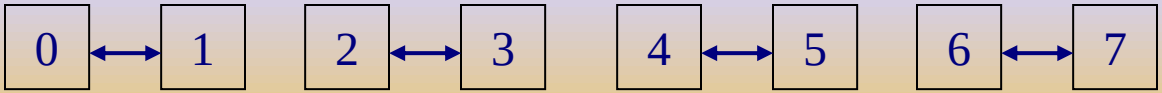
方体函数是由n个互连函数组成，其中0≤k≤n。

比如，n为3时，3-立方体各结点地址如下：



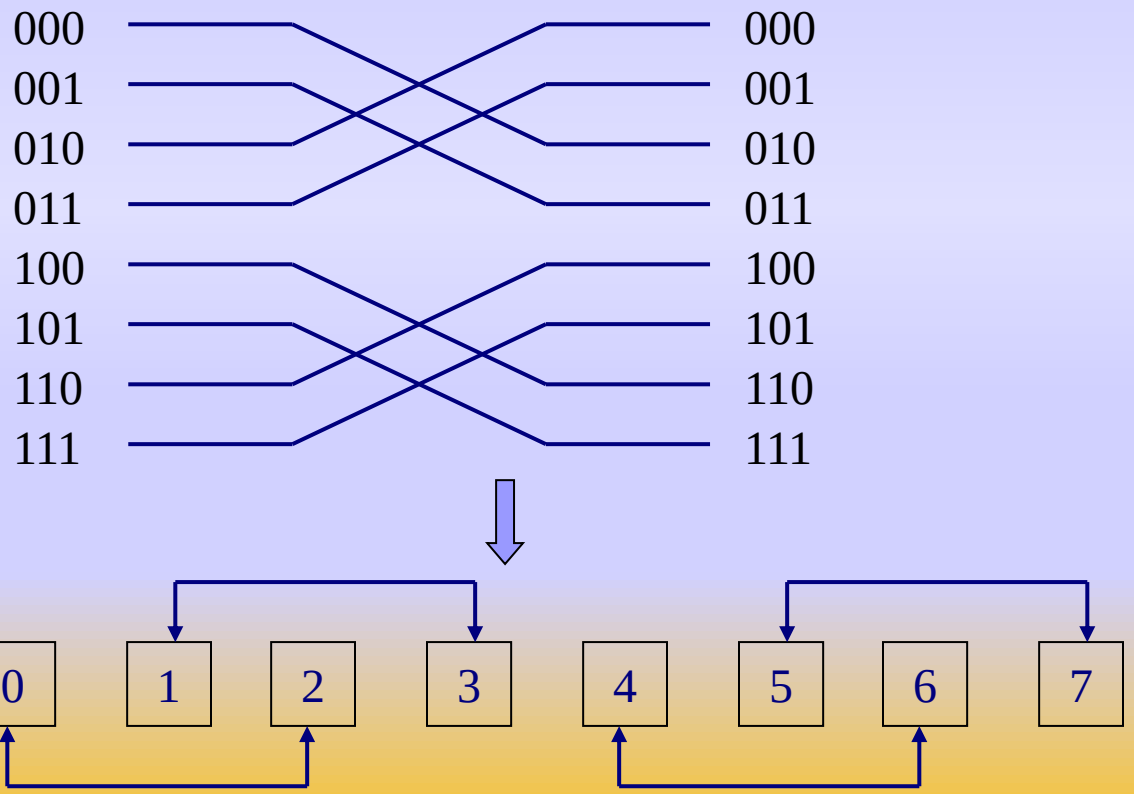
Cube$_0$: $cube_0(X_2X_1X_0) = (X_2X_1\overline{X_0})$



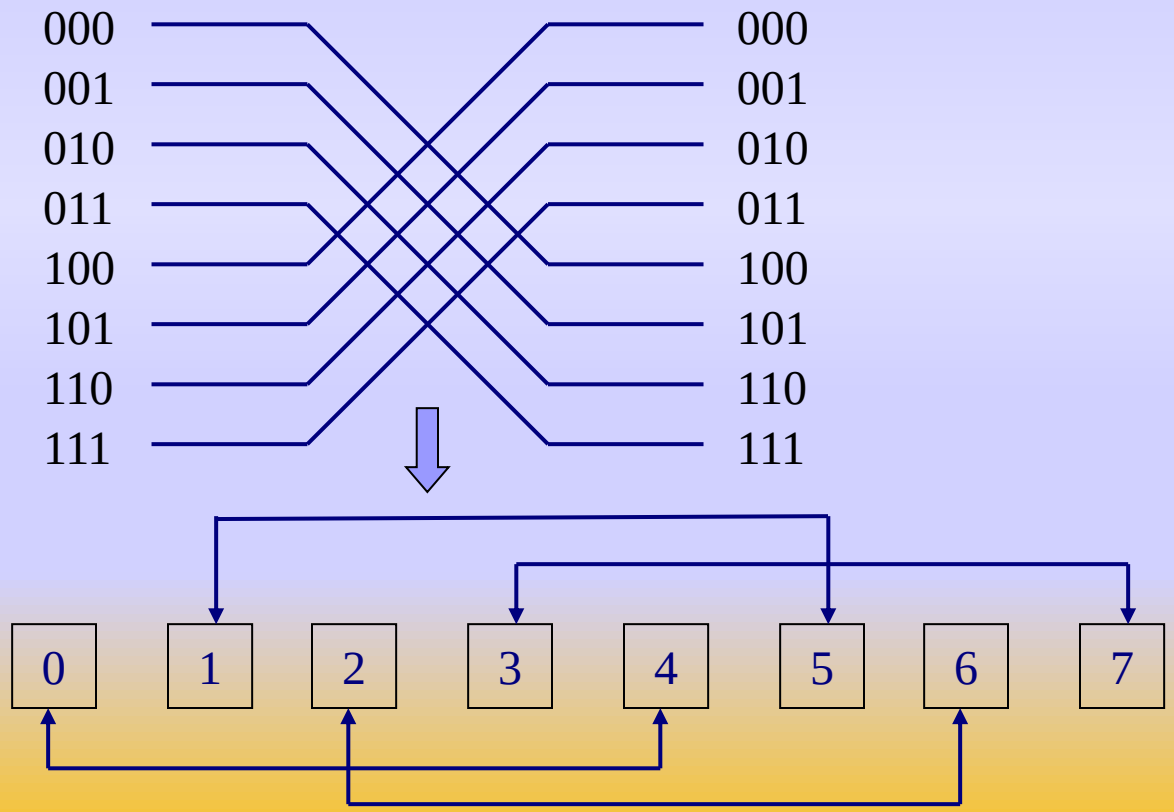


$Cube_1$:

$$cube_1(X_2X_1X_0) = (X_2\overline{X}_1X_0)$$

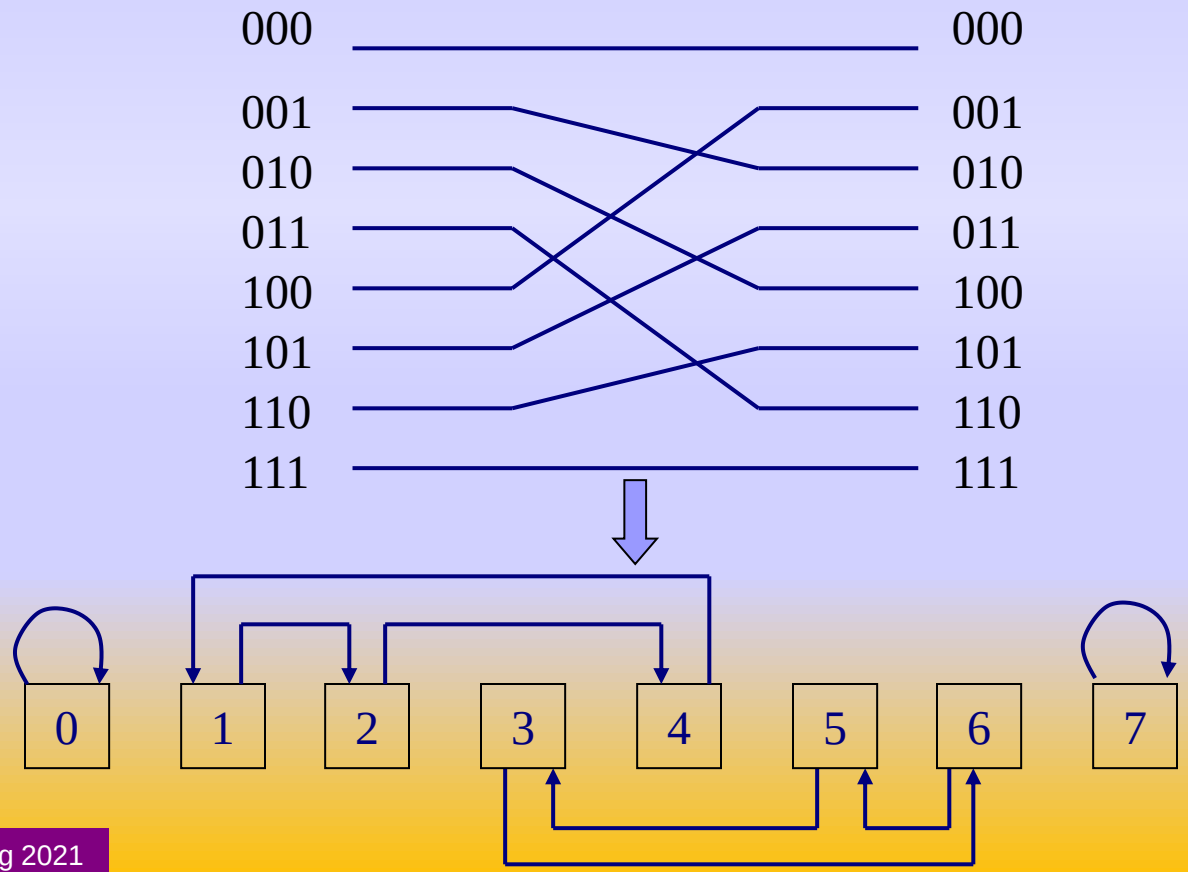


Cube$_2$: $cube_2(X_2X_1X_0) = (\overline{X}_2X_1X_0)$



3. 洗牌函数

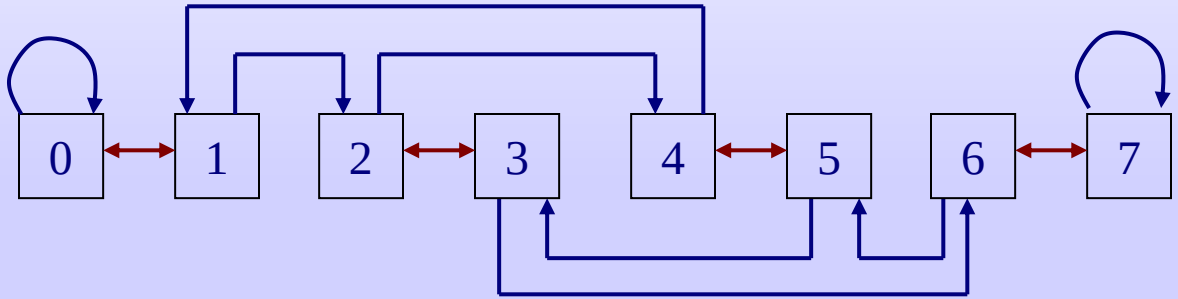
$$Sh(X_{n-1}X_{n-2}\cdots X_k\cdots X_0)=(X_{n-2}\cdots X_k\cdots X_0X_{n-1})$$



洗牌函数的变形：

a. 均匀洗牌（Shuffle-Exchange)

是洗牌函数与$Cube_0$函数的组合。



b. 第k个子洗牌

$$Sh_k(X_{n-1}\cdots X_{k+1}X_kX_{k-1}\cdots X_0)=(X_{n-1}\cdots X_{k+1}X_{k-1}\cdots X_0X_k)$$

即最低k位循环左移一位。

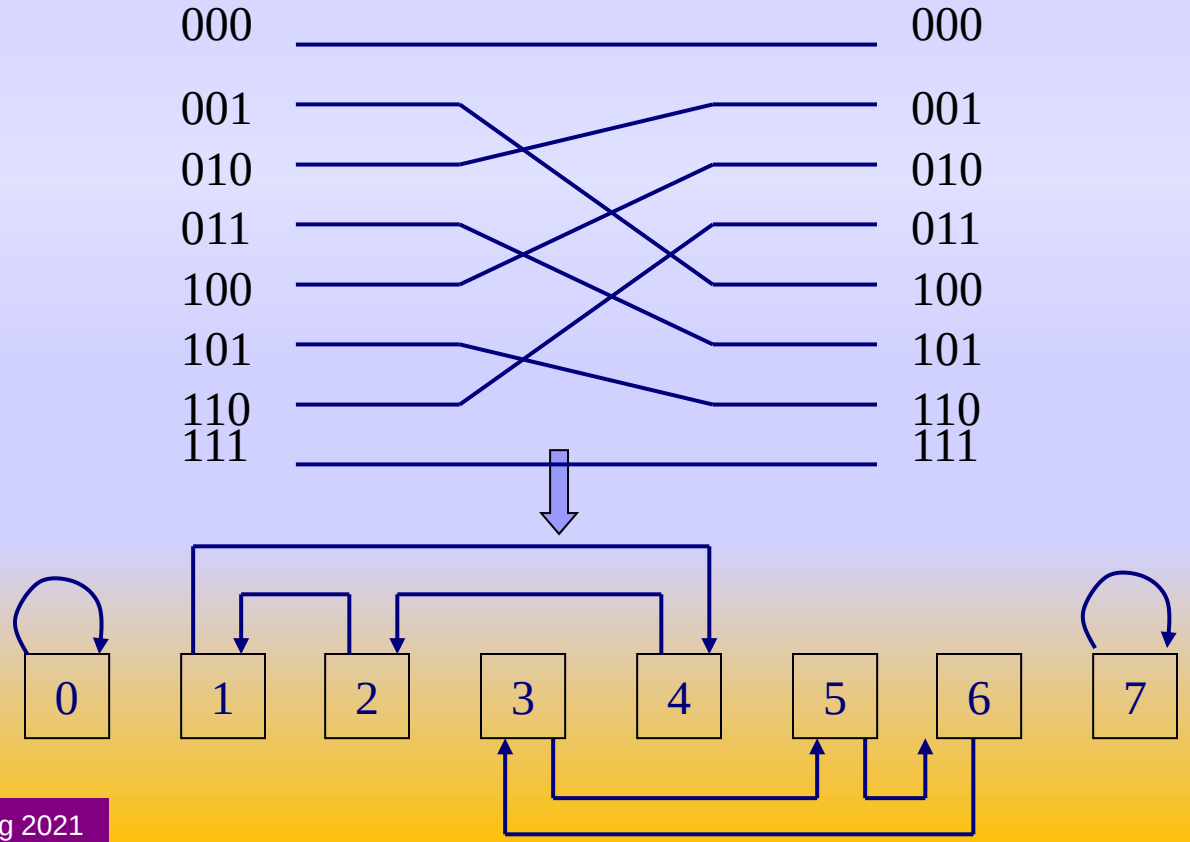
c. 第k个超洗牌

$$Sh^k(X_{n-1}X_{n-2}\cdots X_{n-k+1}X_{n-k}\cdots X_0)=(X_{n-2}\cdots X_{n-k+1}X_{n-1}X_{n-k}\cdots X_0)$$

即最高k-1位循环左移一位。

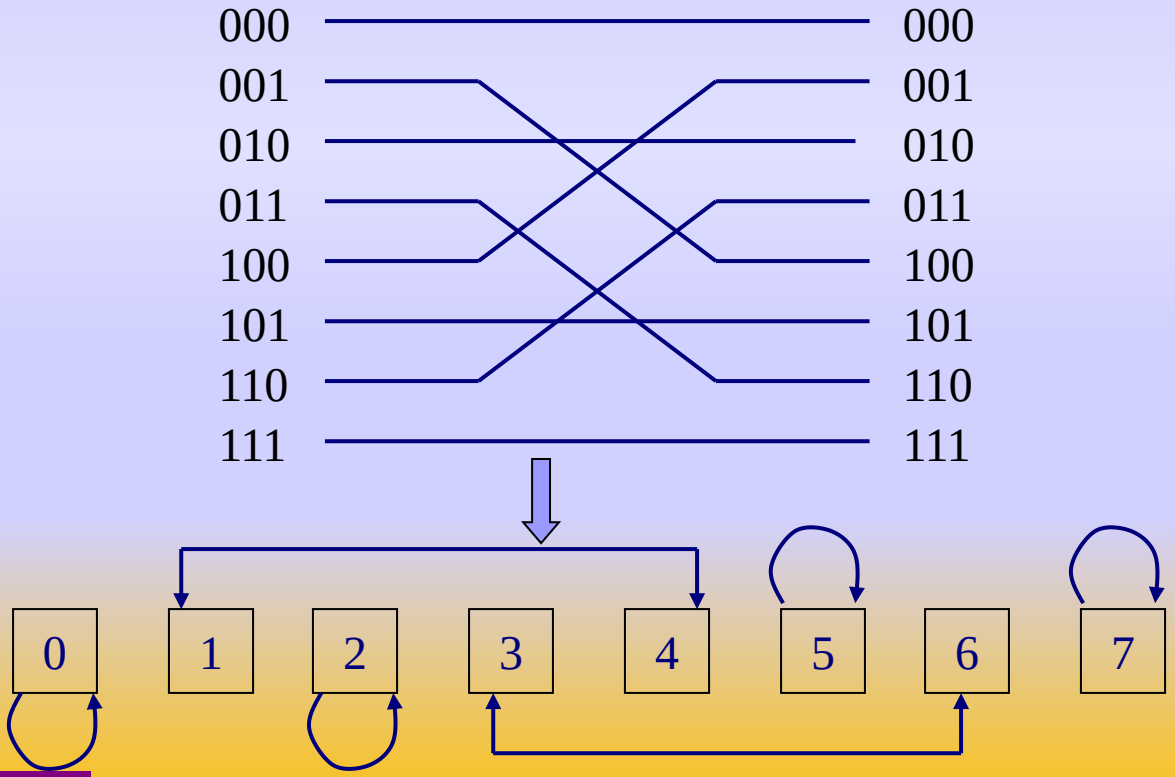
4. 逆洗牌函数

$$Sh^{-1}(X_{n-1}\cdots X_1X_0)=(X_0X_{n-1}\cdots X_1)$$



5. 蝶式

$$\beta(X_{n-1}X_{n-2}\cdots X_1X_0)=(X_0X_{n-2}\cdots X_1X_{n-1})$$



6. PM2I函数（加减2^i）

共有2n个互连函数，对N个结点的网络为

$$\begin{cases} PM2_{+i}(j) = j + 2^i \bmod N \\ PM2_{-i}(j) = j - 2^i \bmod N \end{cases}$$

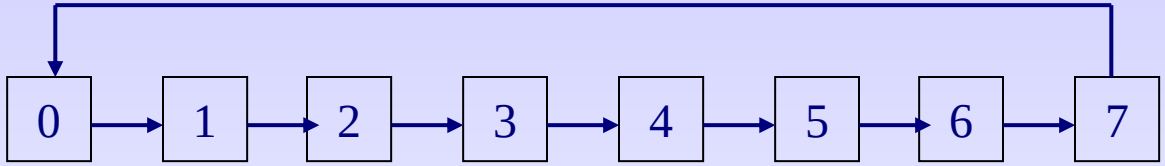
其中，$0 \le j \le N-1, 0 \le i \le n-1$，$n = \log_2 N$

例：

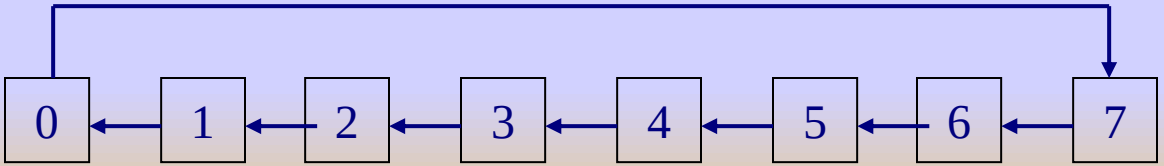
N = 8（8个结点），则$n = \log_2 8 = 3$，所以：i = 0，1，2；j = 0，1，…，7。

6个PM2I函数如下：

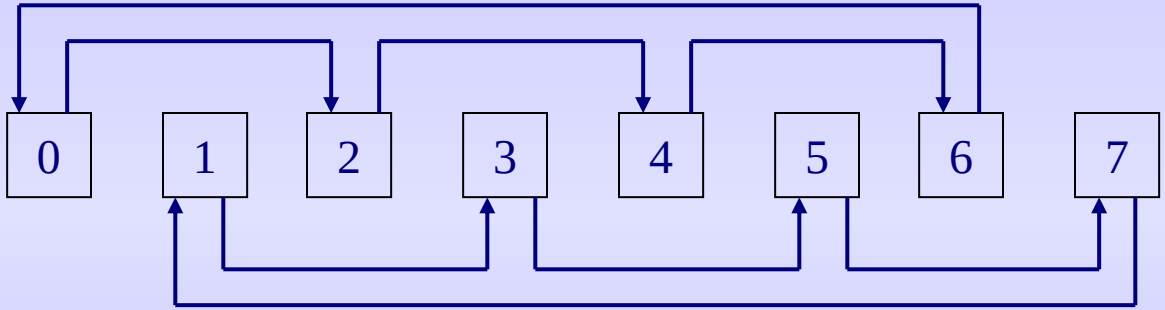
$PM2_{+0}$: (0 1 2 3 4 5 6 7)



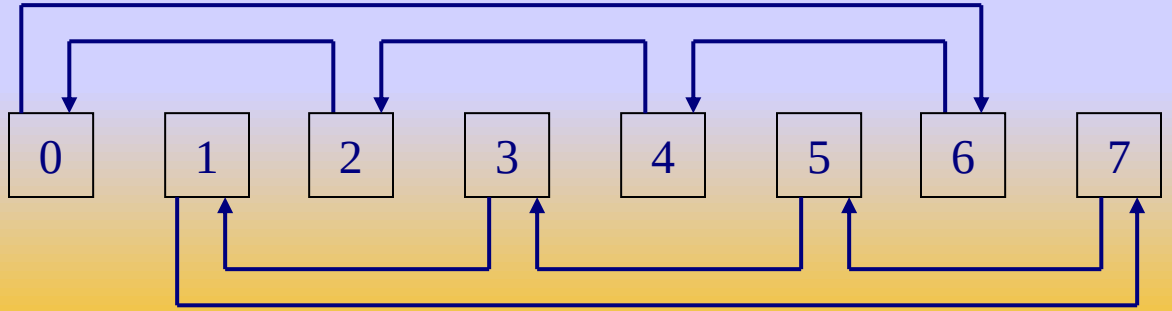
$PM2_{-0}$: (7 6 5 4 3 2 1 0)



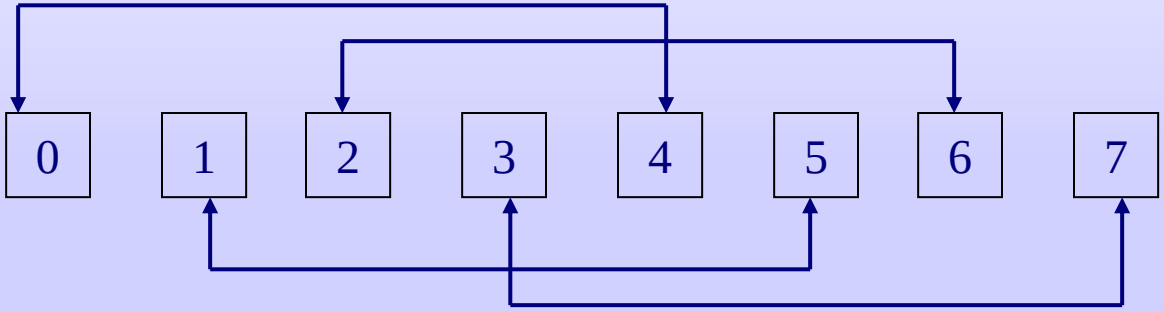
$PM2_{+1}$: (0 2 4 6) (1 3 5 7)



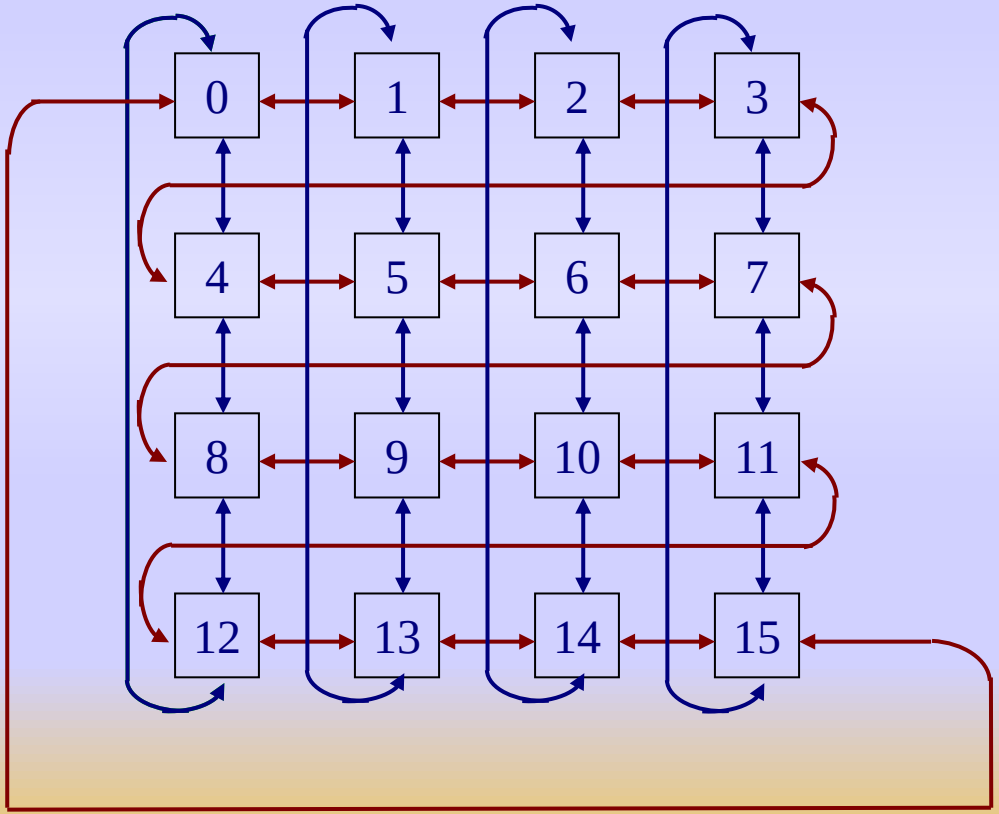
$PM2_{-1}$: (6 4 2 0) (7 5 3 1)



$PM2_{\pm 2}$: （0 4）（1 5）（2 6）（3 7）



例：



上面的网络可以用四个PM2I函数表示。

$PM2_{+0}$: (0 1 2 … 15)

$PM2_{-0}$: (15 14 13 … 0)

$PM2_{\pm2}$: (0 4) (1 5) (2 6) (3 7)
(4 8) (5 9) (6 10) (7 11)
(8 12) (9 13) (10 14)
(11 15) (12 0) (13 1) (14 2) (15 3)

互连与通信

互连网络概念

静态网络

静态网络的特点与指标

典型的静态网络

动态网络

通信问题

静态网络

静态网络的特点与指标

1.静态网络的特点

静态网络由点—点直接相连而成，这种连结方式在程序执行过程中不会改变。

如果用图来表示，结点代表开关，边代表通信链路，则

- 结点间的链路无源，不能重构
- 开关元件与处理机相连
- 不直接相连结点间的通信需通过中间结点中转。

2.静态网络的指标

结点度：与结点相连接的边（链路或通道）数，表示节点所需要的I/O端口数，**模块化要求结点度保持恒定**。根据通道到结点的方向，结点度可以进一步表示为：

结点度 = 入度 + 出度

其中入度是进入结点的通道数，出度是从结点出来的通道数

距离：与两个结点之间相连的最少边数。

网络直径：网络中任意两个结点间距离的最大值。

网络规模：网络中结点数，表示该网络功能连结部件的多少

等分宽度：某一网络被切成相等的两半时，沿切口的最小边数称为该网络的等分宽度

结点间的线长：两个结点间的线的长度

对称性：从任何结点看，拓扑结构都一样，这种网络实现和编程都很容易

结点是否同构

通道是否有缓冲

典型的静态网络

1.线性阵列

对N个结点的线性阵列，有N-1条链路，直径为N-1（任意两点之间距离的最大值），度为2，不对称，等分宽度为1。

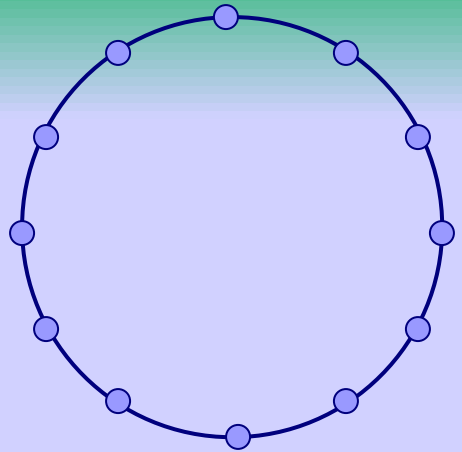
N很大时，通信效率很低。

线性阵列与总线的区别：

线性阵列：允许不同的源结点和目的结点对并发使用系统的不同部分

总线：通过切换与其相连的许多结点来实现时分特性，同一时刻只有一对结点在传送数据。

2.环



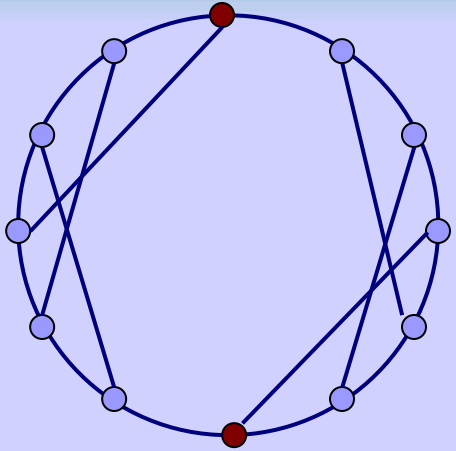
对N个结点的环，考虑相邻结点数据传送方向：

双向环：链路数为N，直径⌊N/2⌋，度为2，对称，等分宽度为2。

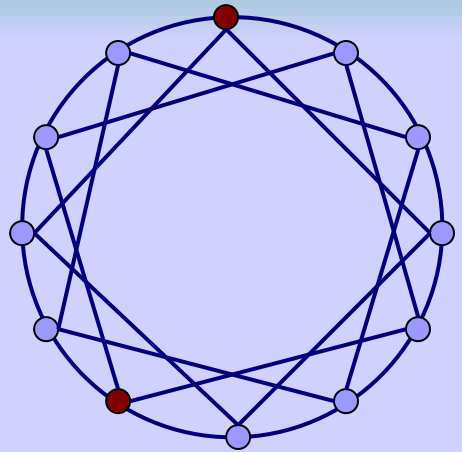
单向环：链路数为N，直径 N-1，度为2，对称，等分宽度为2。

3.带弦环

度为3的带弦环



度为4的带弦环



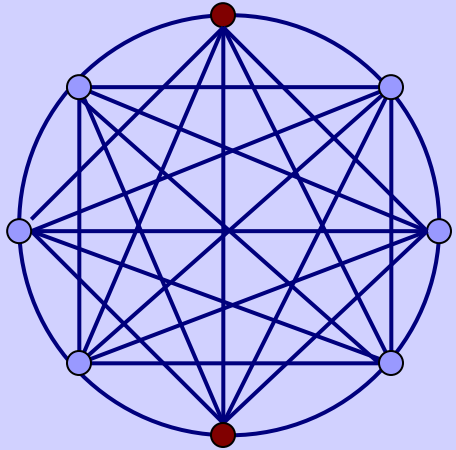
对上图中12个结点的带弦双向环，

结点度为3：链路数为18，直径4（比如红色结点），度为3，不对称，等分宽度为2

结点度为4：链路数为24，直径3（比如红色结点），度为4，对称，　等分宽度为8

4.链接

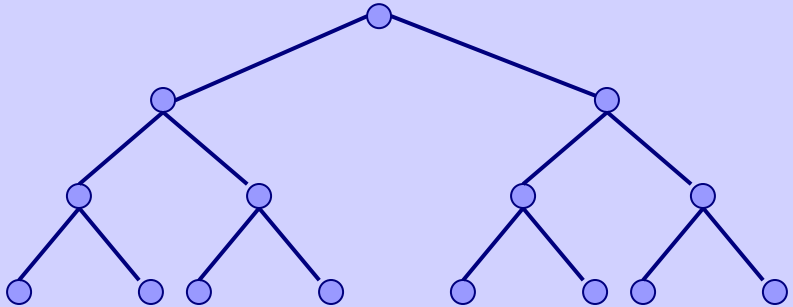
链接是带弦环的一种特殊情形。链接中的每个结点和其他结点之间都有单一的直接链路。如下图中8个结点的链接：



有28条链路，直径为1，度为7，对称，等分宽度为16

5.树形

4层的二叉树

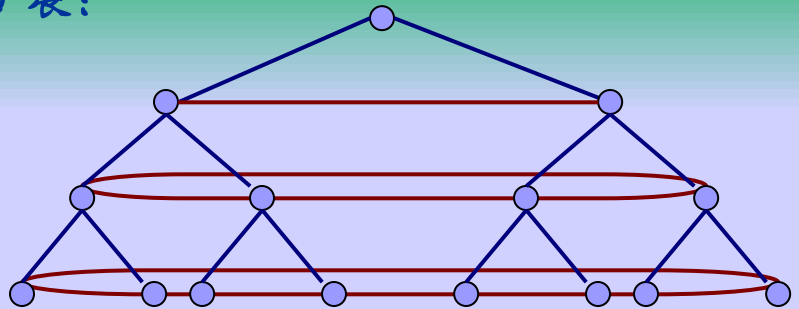


一棵K层完全二叉树应有$N = 2^K - 1$个结点，对大结点度为3，直径为2（K - 1）（即右边任意一个叶子结点到左边任意一个叶子结点）。不对称，等分度为1。

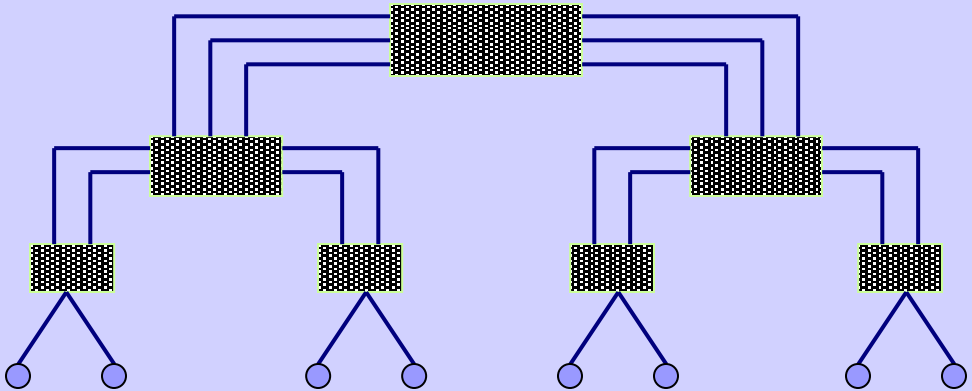
由于结点度为常数，所以树是一种可扩展的系统结构。

树形的扩展：

带环树

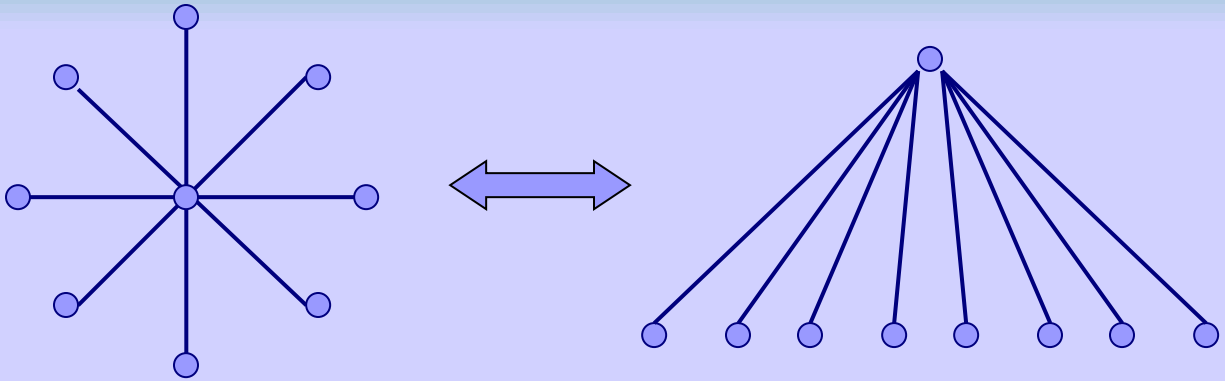


二叉胖树



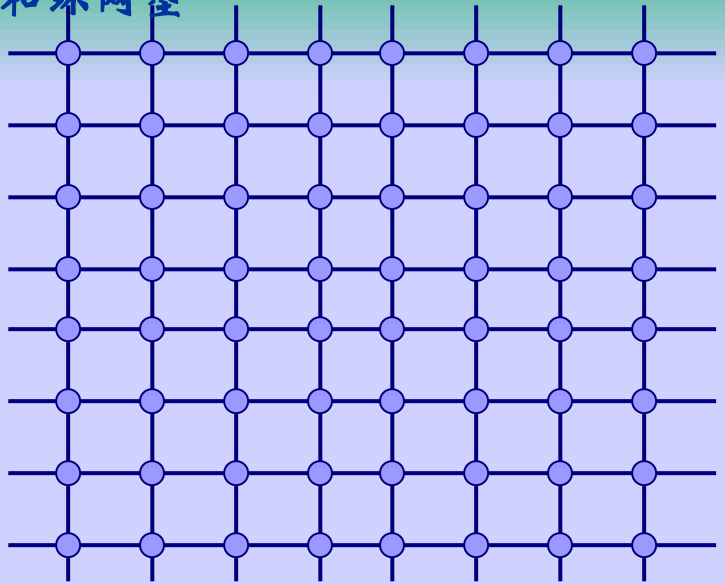
这两种结构都可以缓解根结点的瓶颈问题。

6. 星形



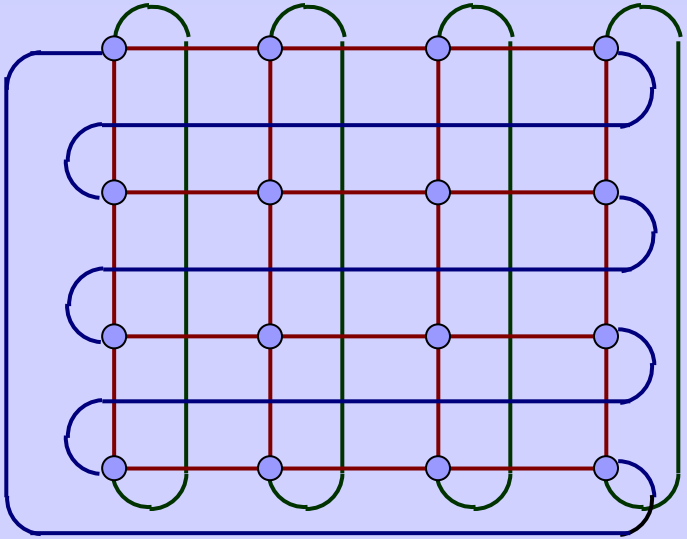
星形实际上是一种二层树（如右图）。有N个结点的星形网络，有N - 1条链路，直径为2，最大结点度为N - 1，非对称

7.网格型和环网型



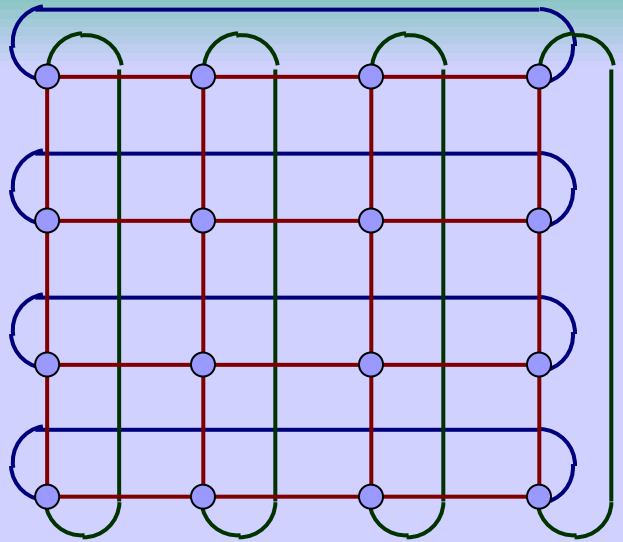
有N个结点的r×r网（其中 $r = \sqrt{N}$ ），有2N - 2r条路，直径为2（r-1），结点度为4，非对称，等分宽度为r

a. Illiac 网



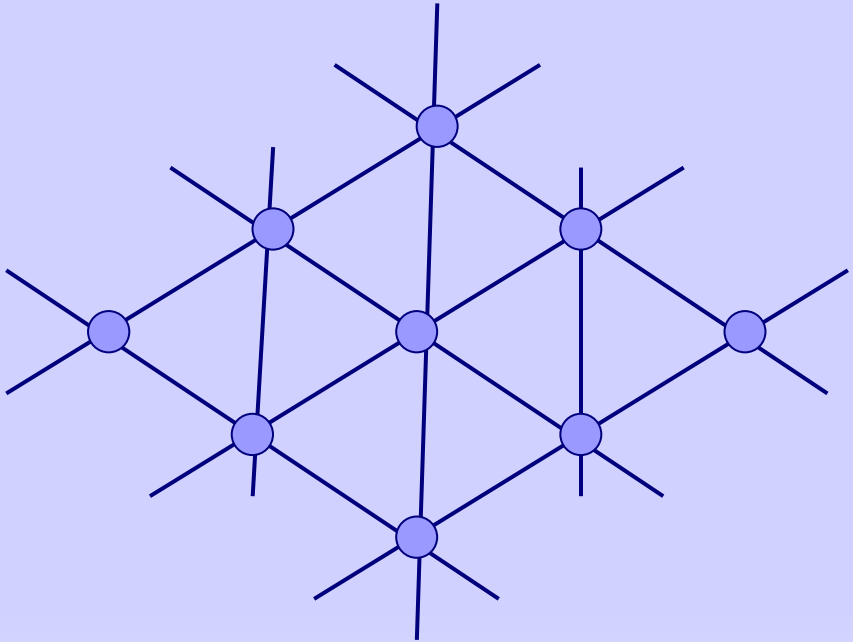
有N个结点的r×r网（其中 $r = \sqrt{N}$ ），有2N条链路，直径为r-1，结点度为4。

b. 环形网（2D—Torus）

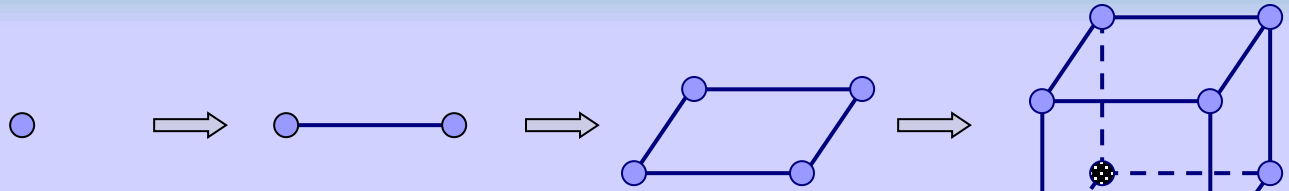


有N个结点的r×r网（其中 $r = \sqrt{N}$ ），有2N 条链路，直径为2⌊r/2⌋，结点度为4，对称。

c. 搏动式阵列（Systolic Array）

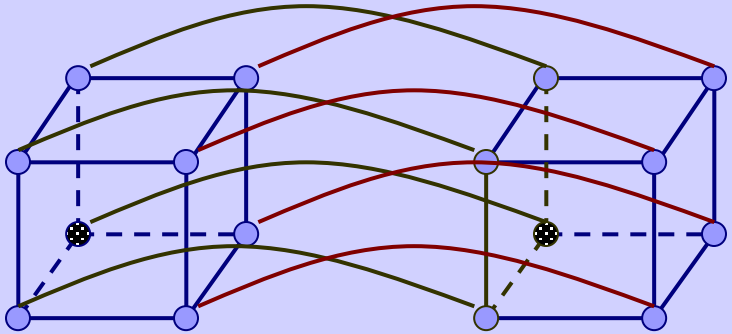


8.超立方体



0-立方体　　1-立方体　　2-立方体

3-立方体



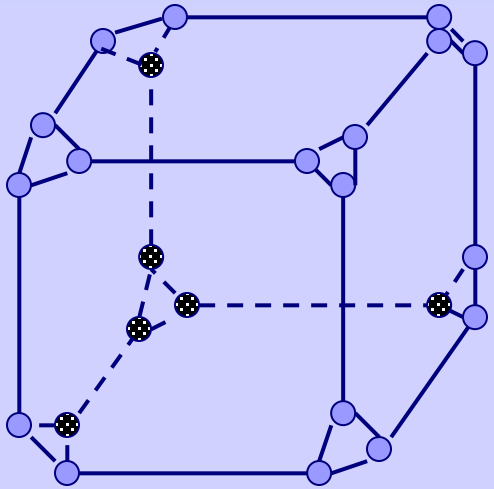
4-立方体

一个n-立方体由$N = 2^n$个结点构成，它们分布在n维上，每维有两个结点。直径为n，结点度为n，对称。由于结点度随维数线性增加，所以超立方体不是一种可扩展结构。

例子：

Intel的iPSC/1、iPSC/2、nCUBE

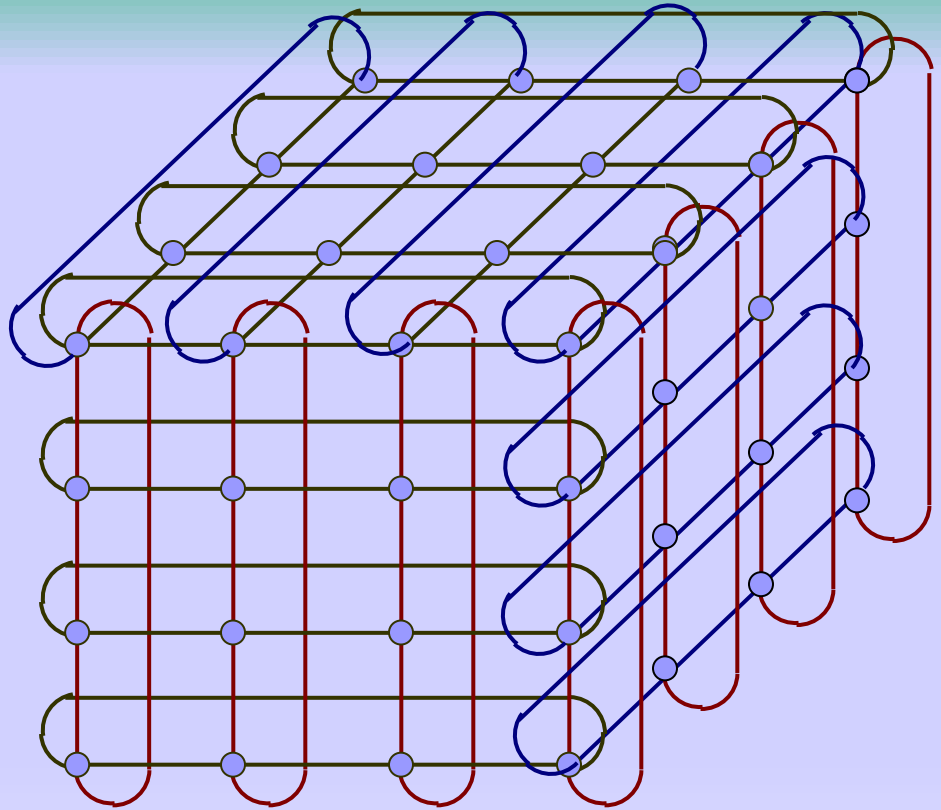
9.带环立方体



带环3-立方体

一个带环n-立方体由$N = 2^n$个结点环构成，每个结点环是一个有n个结点的环，所以结点总数为$n\ 2^n$个。直径通常为2n，结点度为3，对称

10.k元n-立方体网络



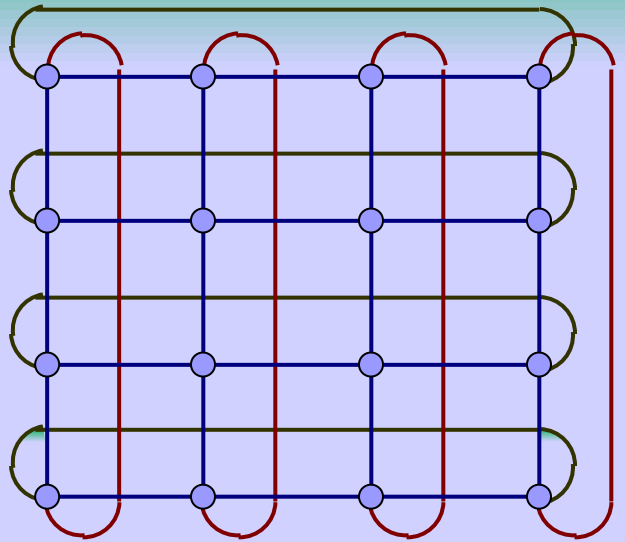
4元3-立方体（隐藏的结点与连接没有画出)

在一个k元n-立方体网络中，结点的数目$N = k^n$，即：

$$k = \sqrt[n]{N},$$
$$n = \log_k N$$

其中，k称为基数（radix），n称为维数（dimension）

k元n-立方体的结点可以用基数为k的n位地址$A = a_0\, a_1\, a_2 \cdots a_n$来表示，其中$a_i$代表第i维结点的位置



传统的环网等价于4元2-立方体。

互连与通信

互连网络概念

静态网络

动态网络

通信问题

动态网络

特点：

网络的开关元件有源，链路可通过设置这些开关的状态来重构。

只有在网络边界上的开关元件才能与处理机相连。

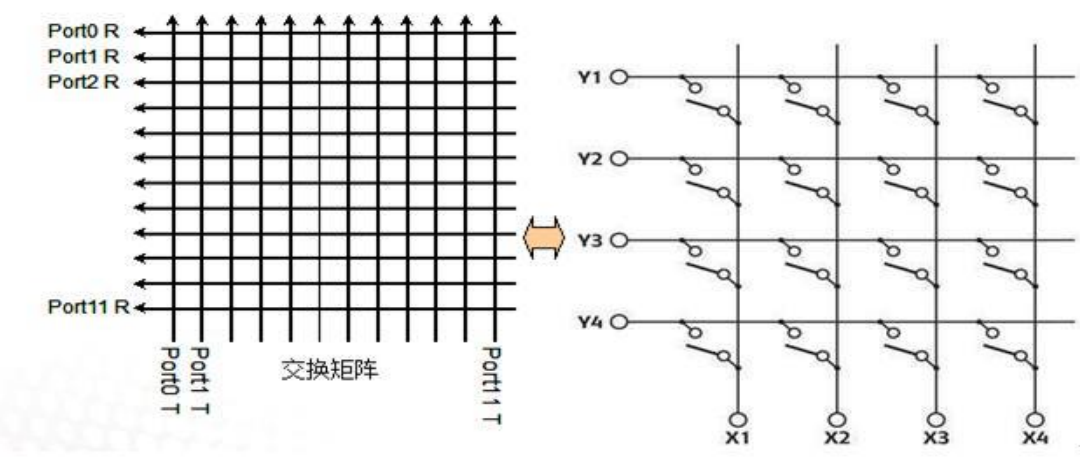
总线

总线是连接各个部件的信息传输线，是各个部件共享的传输介质

总线的分类

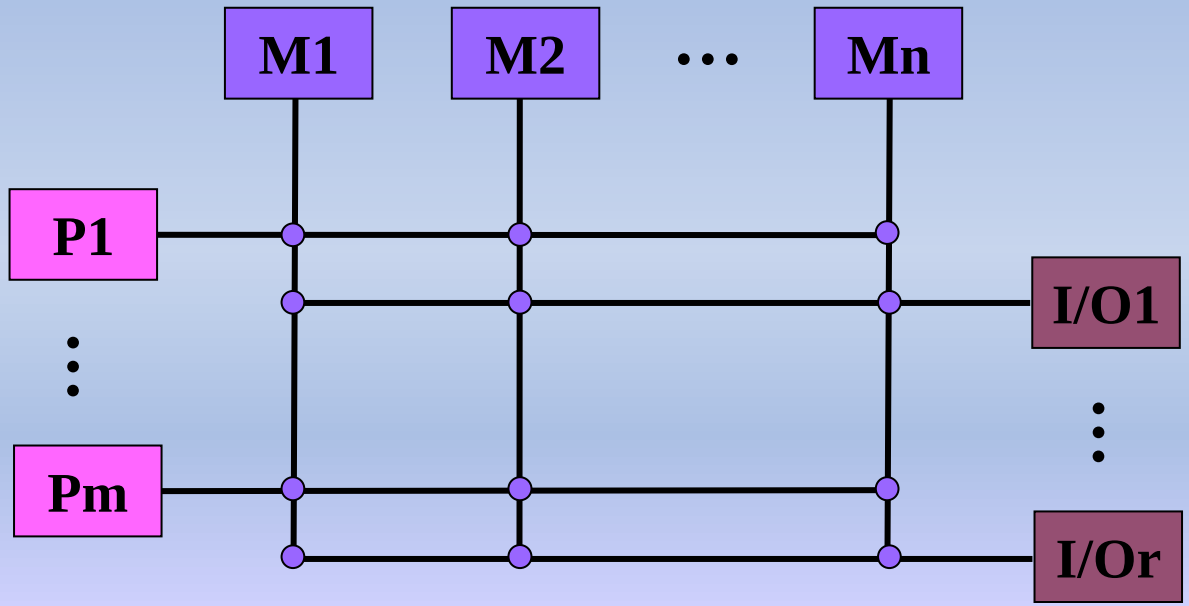
1. 片内总线（在芯片内部）
2. 系统总线（各个部件之间的信息传输线）
   - 数据总线： 双向 与机器字长、存储字长有关
   - 地址总线：单向 与存储地址、IO地址有关
   - 控制总线：有出 有入（从CPU的角度来说的），如存储器读、写 总线允许，终端确认；中断请求，总线请求
3. ++通讯总线用于计算机系统之间或计算机系统与其他系统（如控制仪表、移动通信等）之间的通信

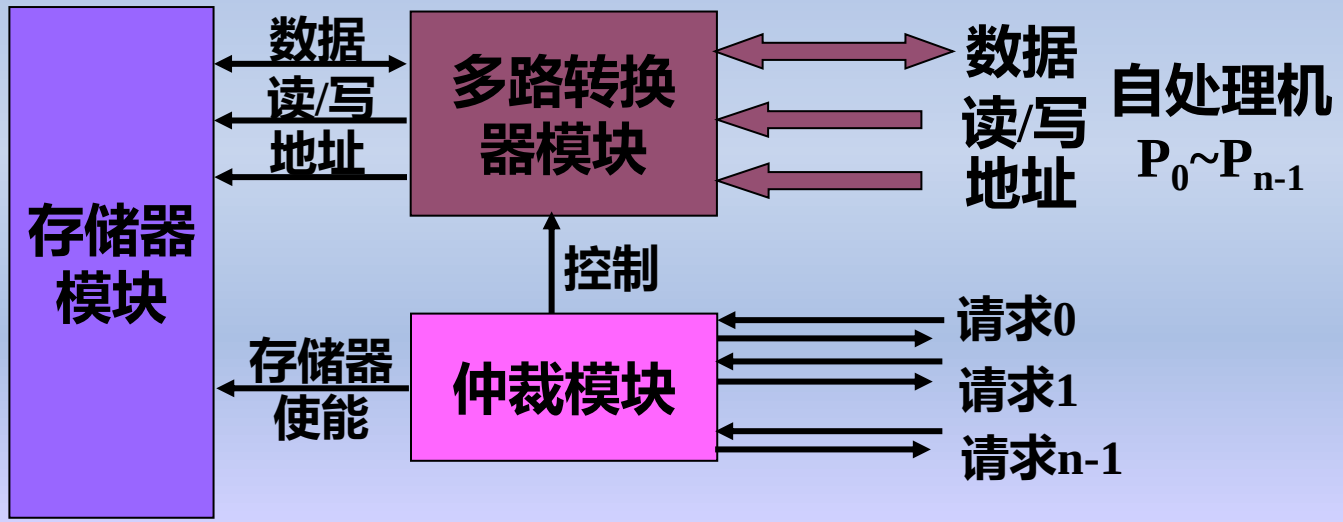
交叉开关



交叉开关

- 大大展宽了互连传输频带，提高了系统的效率；但交叉开关设备量过大，成本过高（一般$n \leq 16$）
- 采用按空间分配的机制


M1
M2
Mn
P1
Pm
I/O1
I/Or


存储器
模块
数据
读/写
地址
多路转换
器模块
数据
读/写
地址
自处理机
P_0~P_{n-1}
控制
存储器
使能
仲裁模块
请求0
请求1
请求n-1

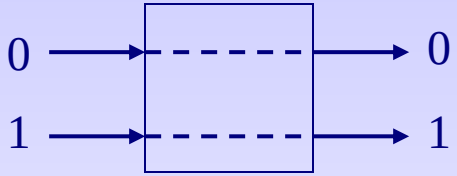
结点开关的结构

多级互连网络

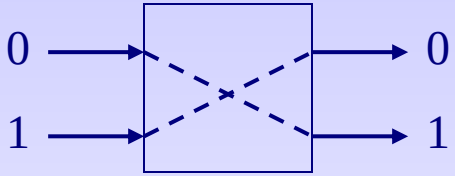
1. 多级网络的三要素

（1）开关单元：a个输入a个输出的开关单元记做a×a的开关单元，其中，a是2的整数倍。常见的有2×2、 4×4、8×8等

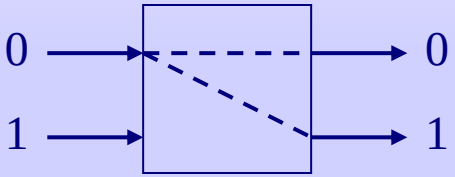
根据开关单元功能的多少， 2×2又可以分为两功能和四功能开关



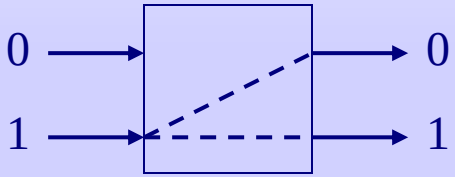
直送



交叉



上播



下播

（2）级间互连模式（InterStage Connection）：

均匀洗牌、蝶式、多路洗牌（比如四路洗牌即是把牌平均分成4份，然后4堆分别进行均匀洗牌）、纵横开关（Cross Switch)及立方体连结等。

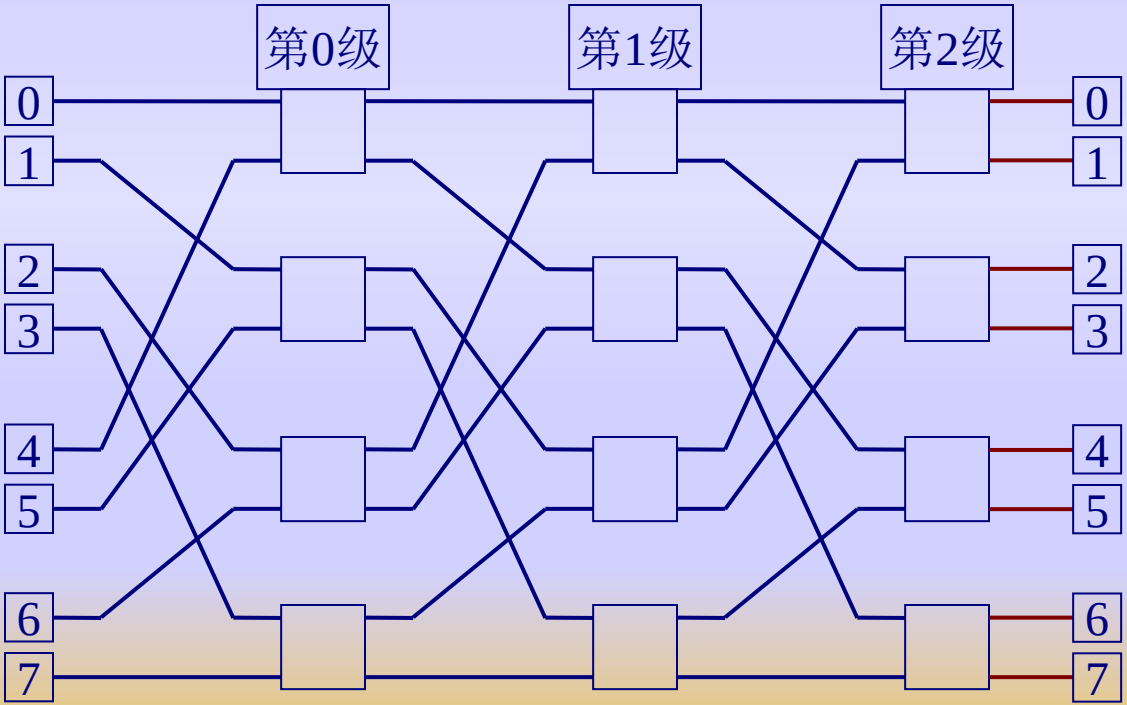
（3）控制方式

级控制：每级只有一个控制信号

单元控制：每个开关一个控制信号

部分级控制：几个开关合用一个控制信号

2. Ω网



Ω网的特点：

开关单元：2×2四功能开关

ISC：洗牌变换+恒等变换

控制方式：采用单元控制方式

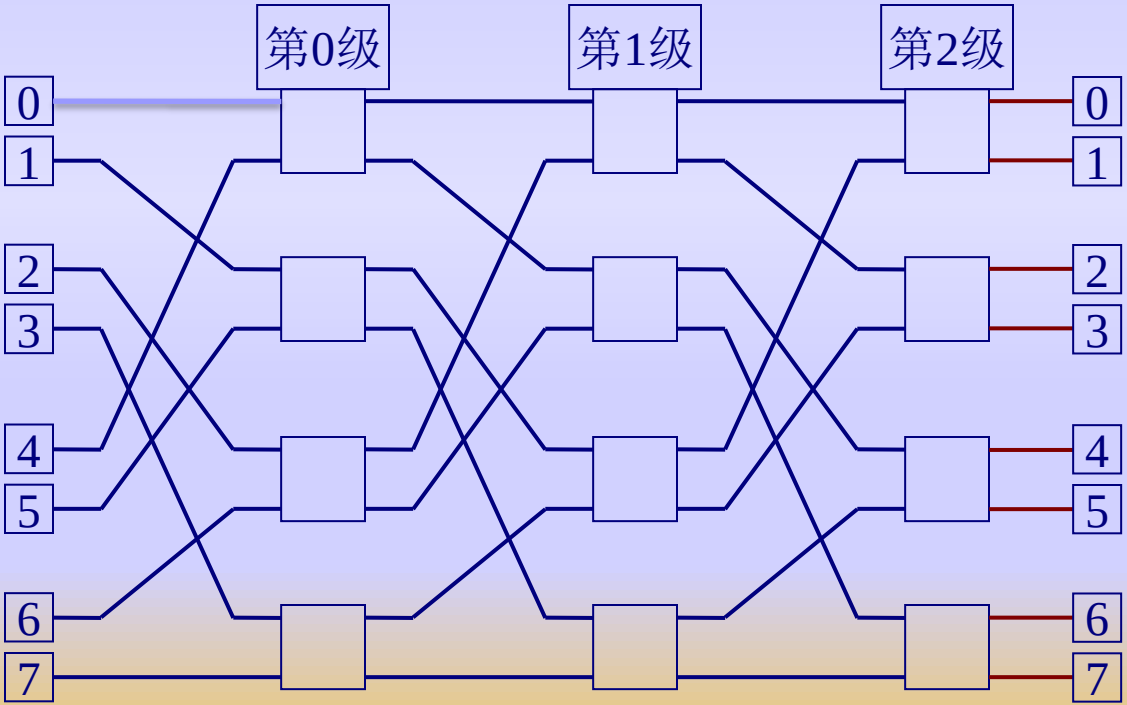
例子：

UIUC的Cedar

IBM的RP3

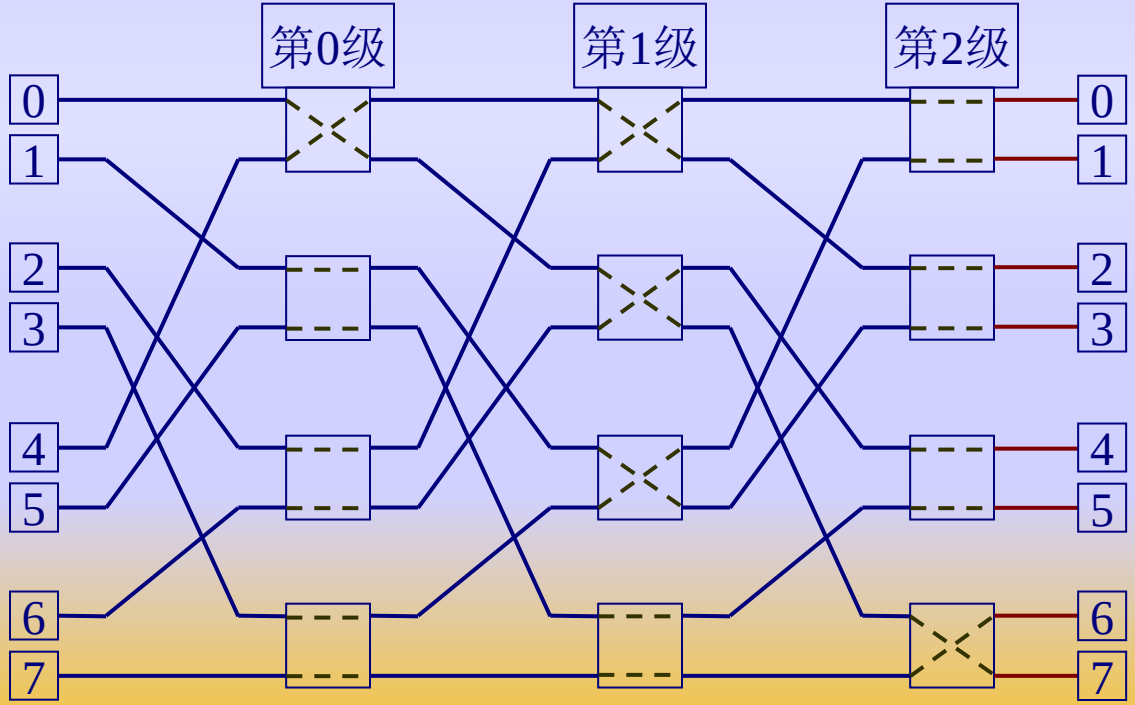
NYU的Ultracomputer

2. Ω网

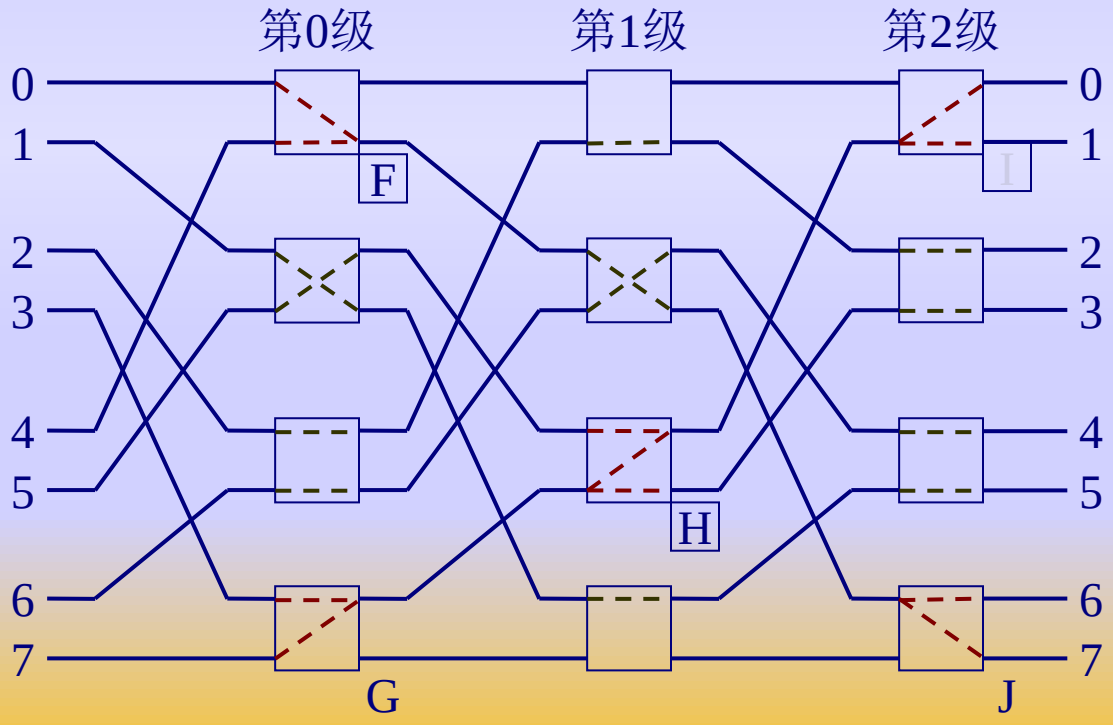


无阻塞的实现置换

π_1=（0 7 6 4 2）（1 3）（5）



置换π_2=（0 6 4 7 3）（1 5）（2）



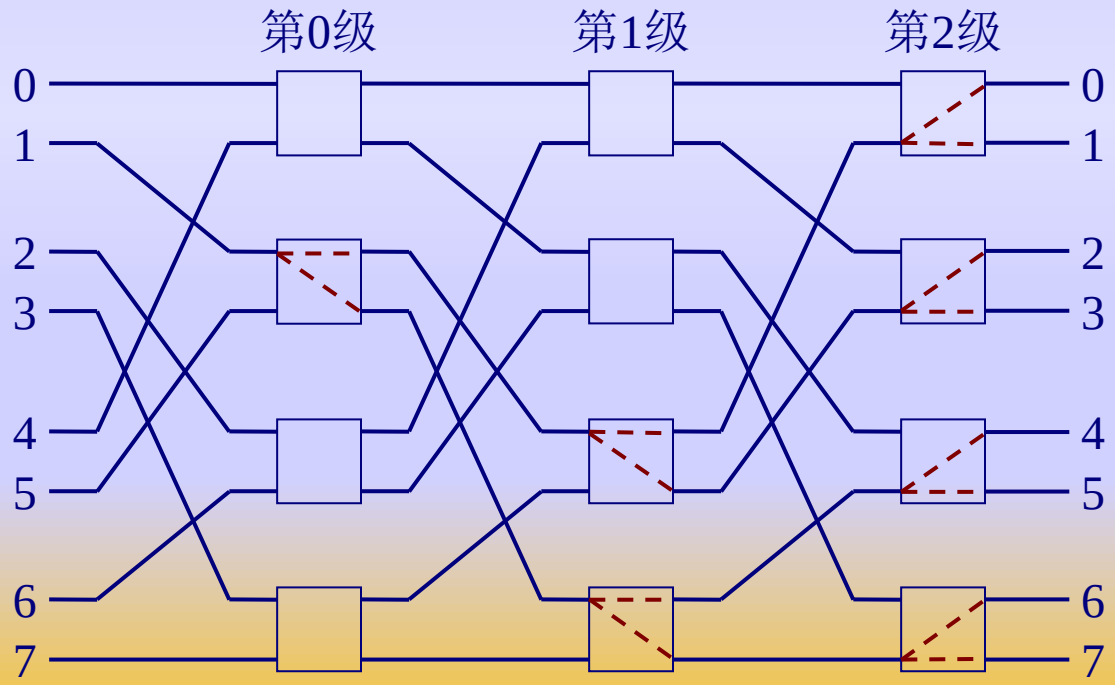
Ω网的特点(2)：

并不是所有的置换在Ω网中一次通过便可以实现。

Ω网是阻塞网络：出现冲突时，可以采用几次通过的方法来解决冲突。

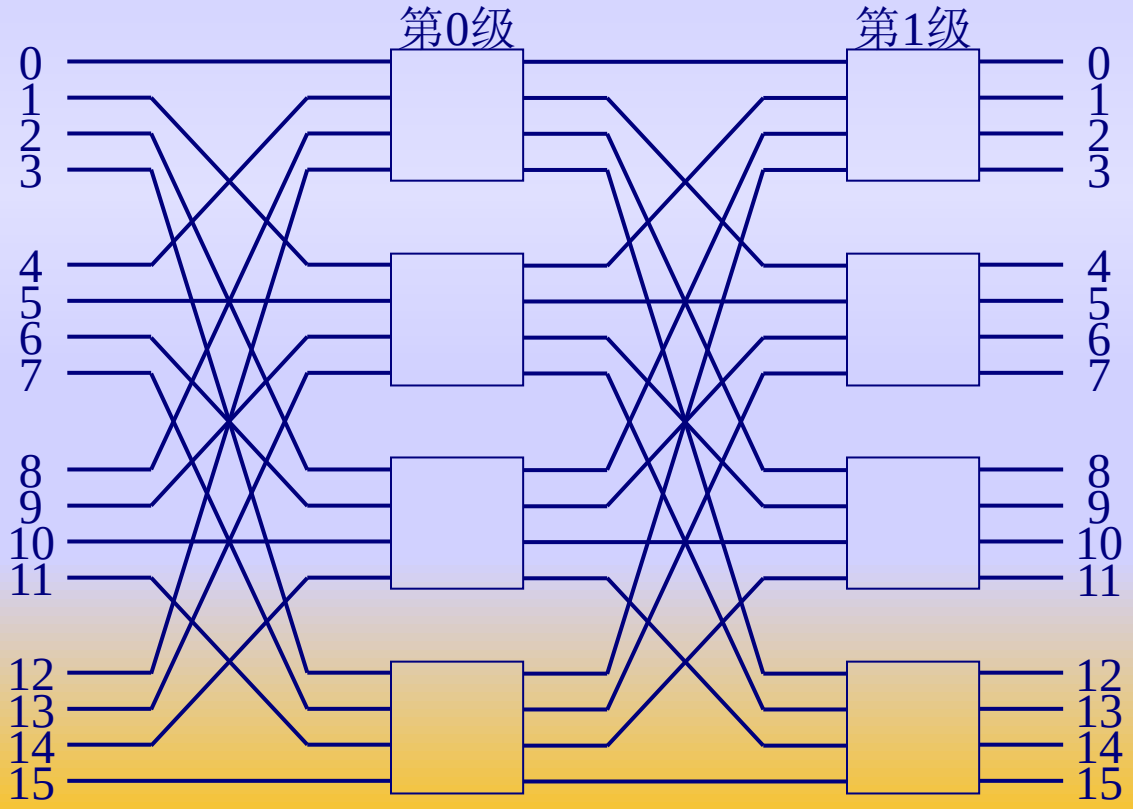
Ω网的广播功能：

001→8个输出端



4×4开关构成的Ω网：多路洗牌

如16输入4路洗牌：网路级数为$\log_4 16 = 2$



Ω网的特点(3)：

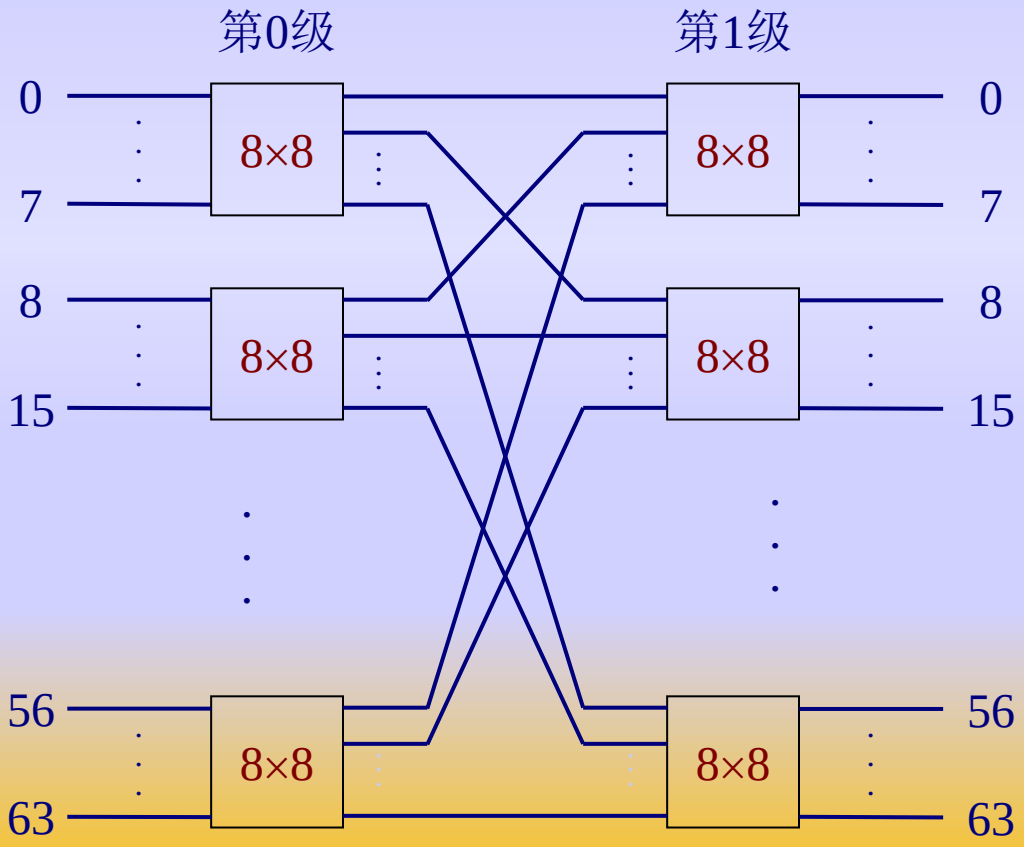
当采用k×k开关元件时，则可以定义k路洗牌函数来构造更大的级数为$\log_k n$的Ω网络。

3. 蝶式网络（Butterfly switch network）

蝶式网络的开关不允许广播功能，它实际上是Omega网的一个子集。

两级64 × 64的蝶式网络如下图所示：它采用16个8 × 8交叉开关构成，两级间采用8路洗牌连接。

两级64 × 64的蝶式网络



互连与通信

互连网络的作用

静态网络

动态网络

通信问题

- 基本术语与性能指标
- 寻径算法
- 虚拟通道与死锁
- 包冲突的解决
- 维序寻径
- 通信模式

通信问题

基本术语与性能指标

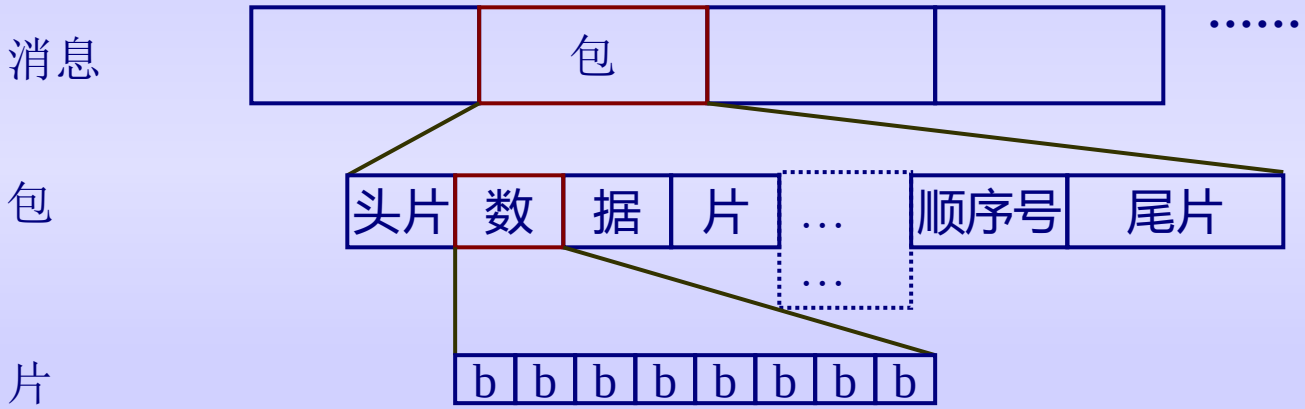
1. 消息、包和片

消息（Message）：是在多计算机系统的处理接点之间传递包含数据和同步消息的信息包。它是一种逻辑单位，可由任意数量的包构成。

包（Packet）：包的长度随协议不同而不同，它是信息传送的最小单位，64-512位。

片（Flit）：片的长度固定，一般为8位。

它们的相互关系如下图：


消息
包
……
包
头片
数
据
片
…
…
顺序号
尾片
片
b b b b b b b b

2.互连网络

互连网络用来在多计算机系统的处理结点之间传递消息

互连网络的描述：

- 拓扑（Topology）
- 寻径算法（Routing）
- 流控制（Flow Control）

互连网络性能的两个重要指标：

- 传输时延（Transmission Latency）
- 吞吐量（Throughput）

3.传输时延与吞吐量

一个**消息的传输时延**：从它在源结点进行发送初始化到它在目的结点完整的被接收所耗费的时间

一个**网络的传输时延**：在一定条件下发送消息的平均时延

网络的吞吐量：单位时间内网络所能传输的消息数目或长度

4.传输时延的公式

$$T = T_s + T_n + T_b$$

其中，T_s称为建立时延，T_n称为网络时延，T_b称为阻塞时延

建立时延T_s：一个消息在源结点和目的结点上装配和分解、从存储器拷贝到通信缓冲区以及正确性验证等所耗费的时间。它和机器本身的硬件、软件技术有关。

$$T_s = T_{ss} + T_{sd}$$

其中：

T_{ss}称为源结点时延：从发送进程开始消息发送初始化到消息的头部进入网络所经历的时间。

T_{sd}称为目的结点时延：从消息的尾部到达目的结点到消息完全被接收进程接收所经历的时间。

网络时延T_n：消息头部从源结点进入网络到消息的尾部到达目的结点的时间间隔。

$$T_n = T_p \times D + L/B$$

其中：

$T_p \times D$称为结点时延：其中T_p是消息在它所经过的路径上的每个中间结点上的平均时延，D为中间结点或源结点与目的结点之间的距离。

L/B称为线路时延：其中L为消息长度，B为结点之间的通道带宽。

阻塞时延T_b：消息传递过程中其他所有可能的时延（主要原因是资源冲突）。

5. 网络的拓扑结构

第一代并行计算机：**HyperCube**

第二代并行计算机：**n—Mesh**

其他...

6.网络的寻径算法

决定发送一个消息到其目的地所经过的路径。

可以分为：

最短路径算法

非最短路径算法

或者：

确定性算法：路径的选择只依赖于它所发送的消息的源结点和目的结点。

自适应算法：消息从结点A到结点B可以由几条不同的路径

7.网络的流控制

当一个消息在网络中沿着某条路径传送时，互连网络如何来为它分配通道和缓冲器。

寻径算法

存储转发（Store-and-Forward）

虚拟直通（Virtual cut through）

线路交换（Circuit Switching）

Wormhole交换（Wormhole Switching)

1.存储转发

当一个消息到达中间结点A时，A把整个消息放入其通信缓冲器中，然后在寻径算法的控制下选择下一个相邻结点B，当从A到B的通道空闲并且B的通信缓冲器可用时，把消息从A发向B。

缺点：

每个结点必须对整个消息进行缓冲，缓冲器较大。

网络时延与发送消息所经历的结点数成正比

$$T_n = T_p \times D + L/B = (L/B) \times D + L/B = (L/B) \times (D+1)$$

2.虚拟直通

中间结点没有必要等到整个消息全部被缓冲后再作出路由选择，只要消息的目的信息域可用后，就可以作出路由选择。

$$T_n = T_p \times D + L/B = (L_h/B) \times D + L/B = (L_h \times D + L)/B$$

其中，L_h为消息头部开始到其目的信息域的长度，显然有$L >> L_h$，所以D的影响比较小。

而当通向下一结点的通道忙或结点的缓冲器非空闲时，必须把整个消息缓冲起来，这时和存储转发一样。

3.线路开关

在传递一个消息之前，就为它建立一条从源结点到目的结点的物理通道。在传递的全部过程中，线路的每一段都被占用，当消息的尾部经过网络后，整条物理链路才被废弃。

$$T_n = T_p \times D + L/B = (L_c/B) \times D + L/B = (L_c \times D + L)/B$$

其中，L_c是为消息建立物理通路所传递的控制信息的长度。当$L >> L_c$时，D的影响较小。

缺点：

物理通道非共享

传输过程中物理通道一直被占用

4.Wormhole

Dally于1986年提出。

首先把一个消息分成许多片，消息的头片包含了这个消息的所有寻径信息。尾片是一个其最后包含了消息结束符的片。中间的片均为数据片

片是最小信息单位。每个结点上只需要缓冲一个片就能满足要求。

用一个头片直接开辟一条从输入链路到输出链路的路径的方法来进行操作。每个消息中的片以流水的方式在网络中向前“蠕动”每个片相当于Worm的一个节，“蠕动”以节为单位顺序的向前爬行

当消息的头片到达一个结点A的寻径器后，寻径器根据头片的寻径信息立即做出路由选择：

（1）如果所选择的通道空闲而且所选择的结点B的通信缓冲器可用，那么这个头片就不必等待，直接通过结点A传向下一个结点B；随后的其它片跟着相应的向前“蠕动”一步。当消息的尾片向前“蠕动”一步后，它刚才所占用的结点就被放弃了。

（2）如果所选择的通道非空闲或者所选择的结点的通信缓冲器非可用，那么这个头片就必须在此结点的通信缓冲器中等待，直到上述两者都可用为止；其它片也在原来的结点上等待。此时，被阻塞的消息不从网络中移去，片不放弃它所占有的结点和通道。这是Wormhole技术和其它流控制技术都不同的地方。

$$T_n = T_p \times D + L/B = (L_f/B) \times D + L/B = (L_f \times D + L)/B$$

其中，L_f是片的长度，通常很小。

优点：

（1）每个结点的缓冲器的需求量小，易于用VLSI实现。

（2）较低的网络传输延迟。所有的片以流水方式向前传送，时间并行性。而在存储转发中，消息是整个的从一个结点“跳”向另一个结点，通道的使用时串行的。所以它的传输延迟基本上正比于消息在网络中传输的距离。

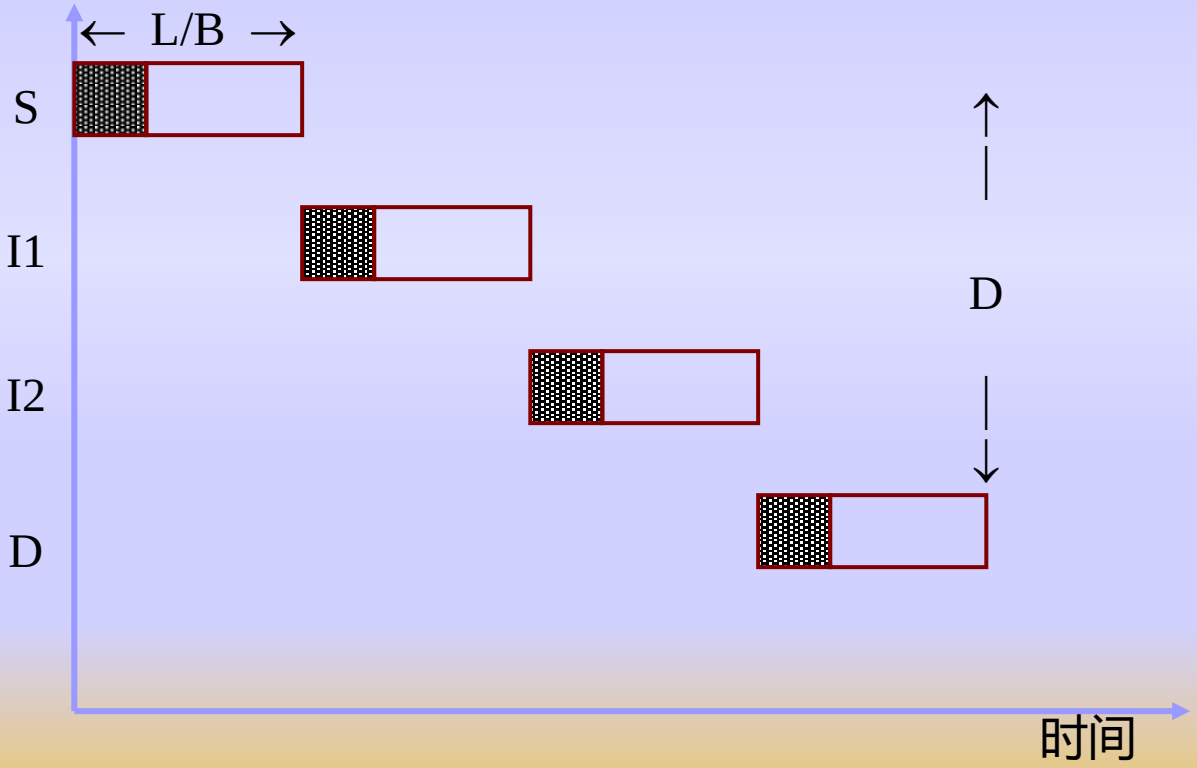
Wormhole与线路开关的网络传输延迟正比于消息包的长度，传输距离对它的影响很小（消息包较长时的情况）。

（3）通道共享性好、利用率高。对通道的预约和释放是结合在一起的一个完整的过程：占有一段新的通道后将立即放弃用过的一段旧通道。

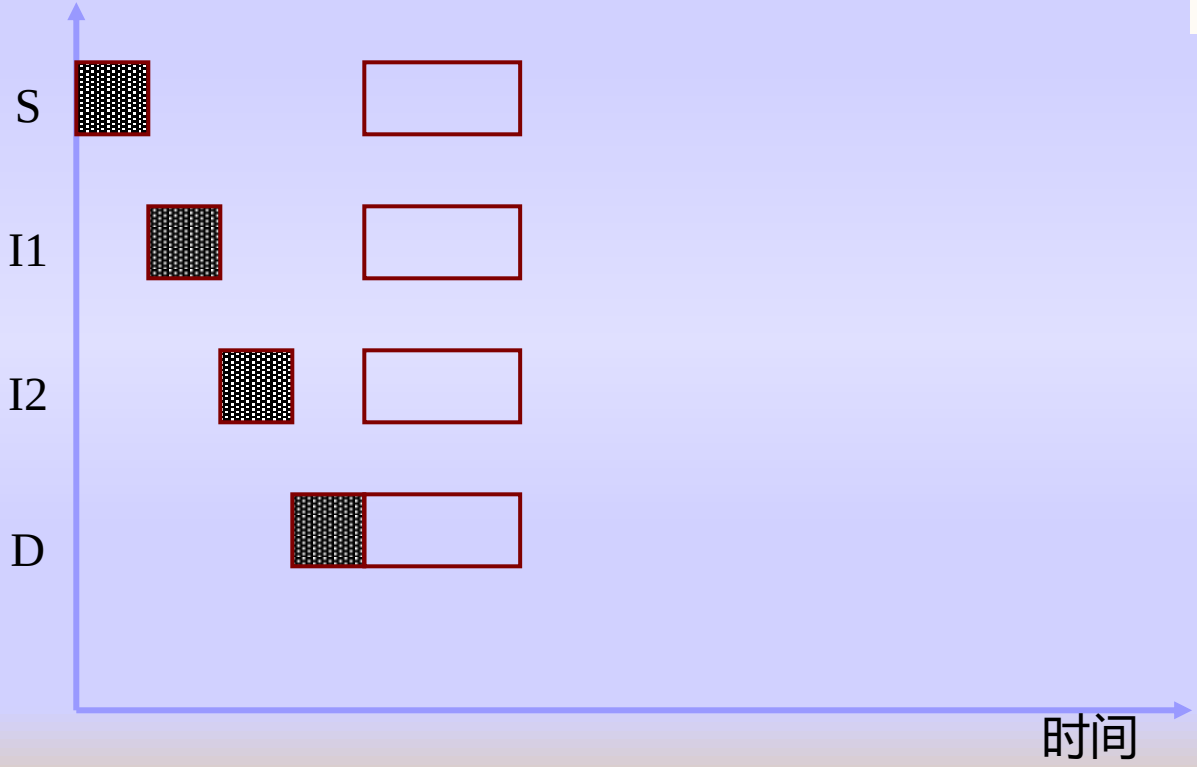
（4）易于实现Multicast和Broadcast。允许寻径器复制消息包的片并把它们从多个输出通道输出。

Wormhole方式中，同一个包中所有的片象不可分离的同伴一样以流水方式顺序的传送。包可看作是一列火车，由火车头（头片）和被牵引的车厢（数据片）组成。

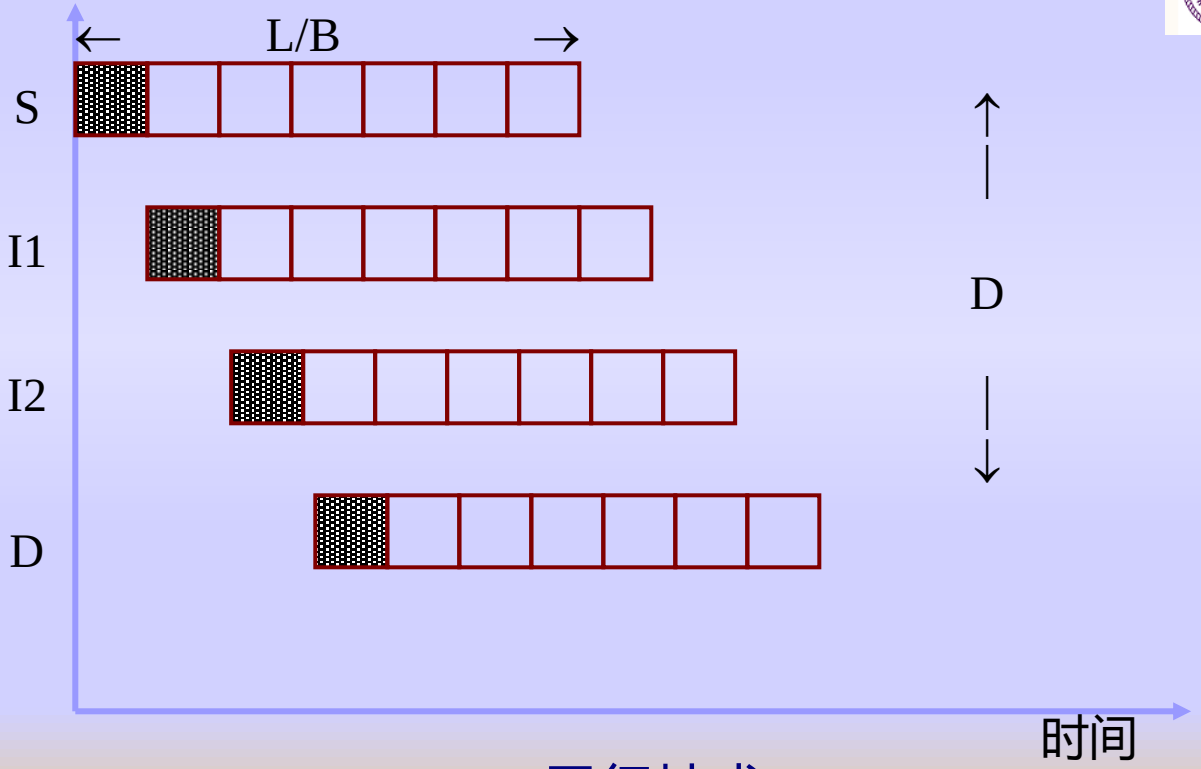
5.几种寻径技术的时空图



存储转发（Store-and-forward）寻径技术



电路开关寻径技术



Wormhole寻径技术

死锁和虚拟通道

1.虚拟通道

特点：

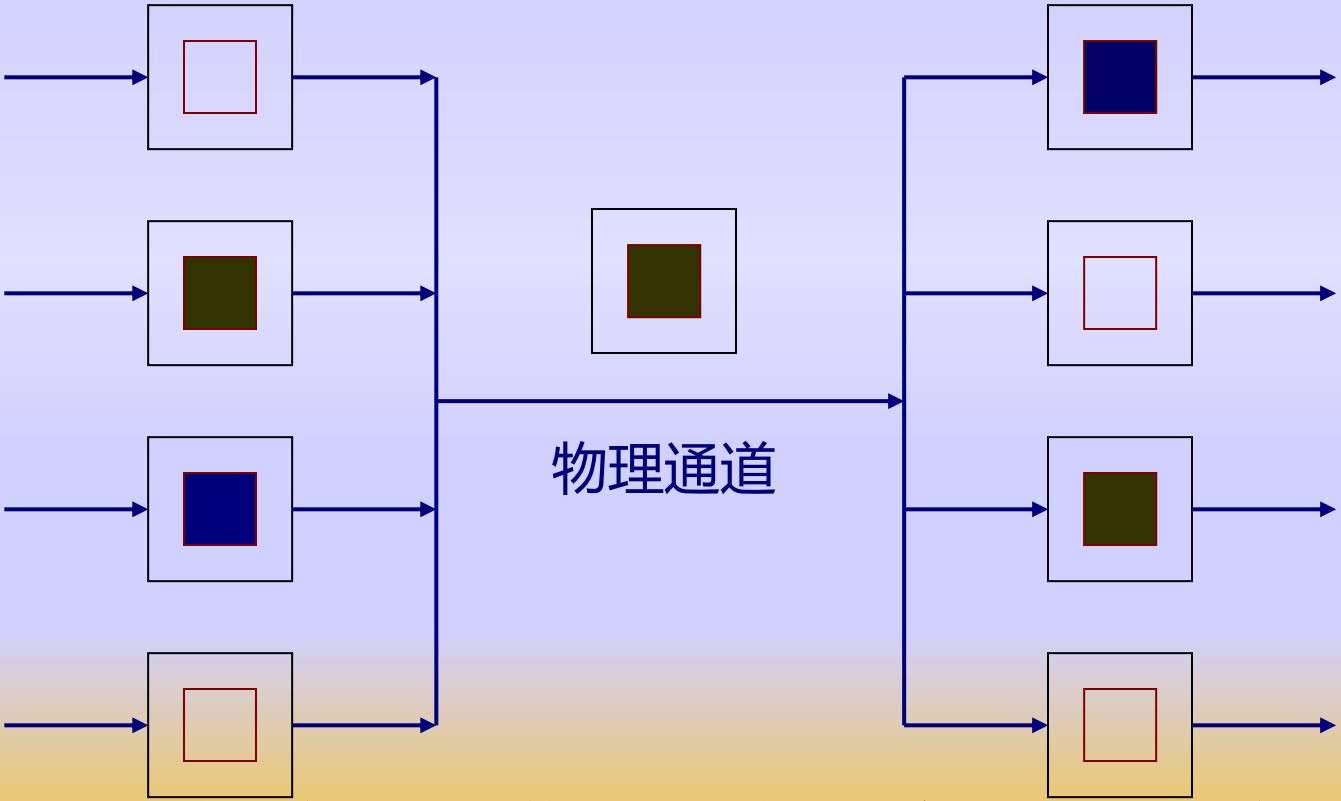
是两个结点间的逻辑链。

由源结点的片缓冲区、结点间的物理通道和接收结点的缓冲区组成。

物理通道由所有虚拟通道分时共享。比如在下例中，四条虚拟通道以片传递为基础分时的共享一条物理通道。

源结点中的片缓冲区

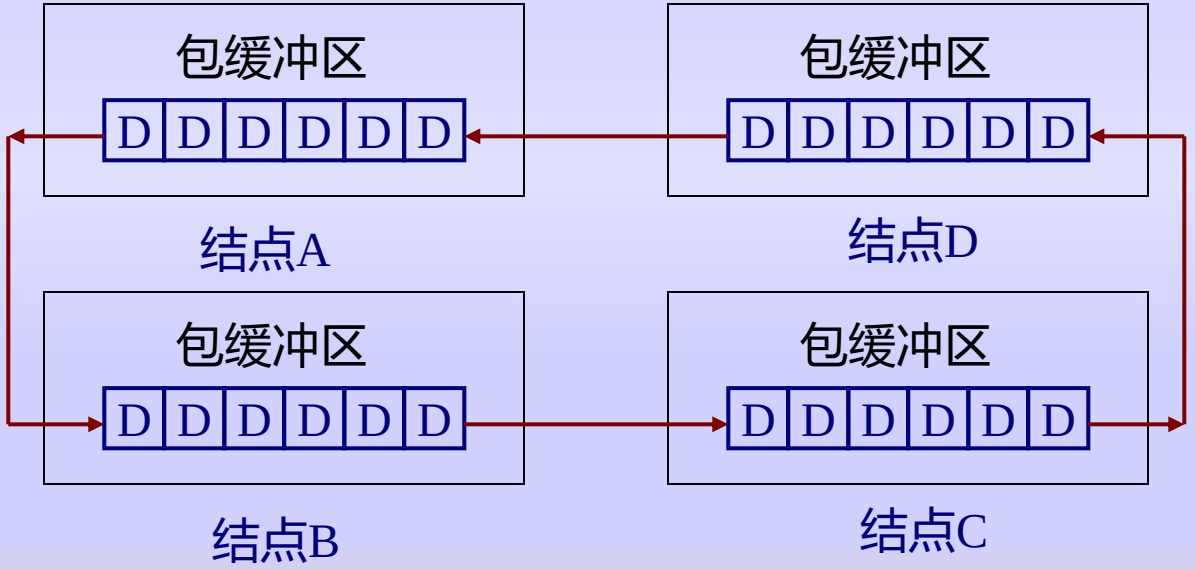
目的结点中的片缓冲区



四条虚拟通道以片传递为基础分时共享一条物理通道

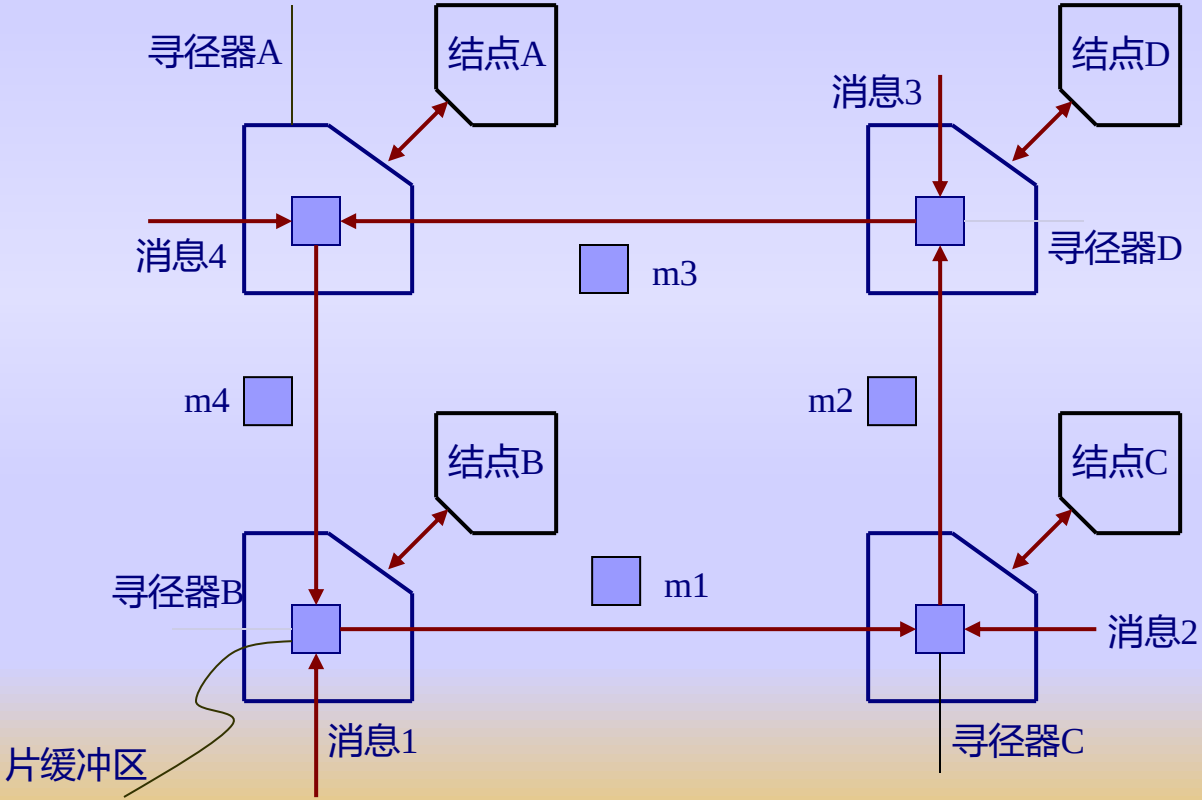
2.死锁的产生

缓冲区产生死锁：



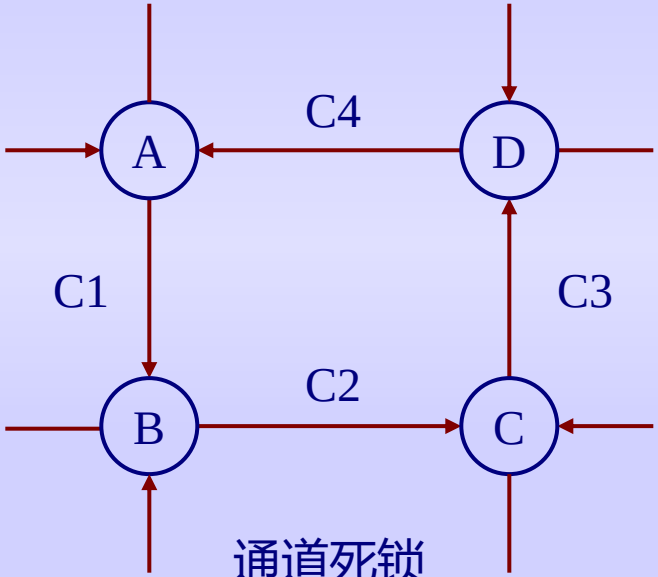
采用存储转发，四个包占用了四个结点的四个缓冲区，将导致循环等待。

通信产生死锁：

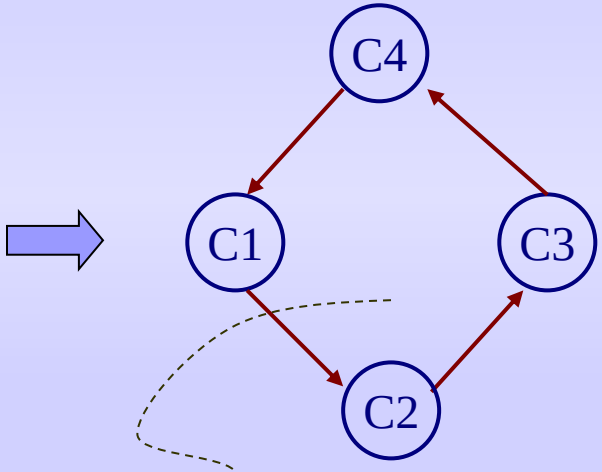


采用Wormhole，四个结点之间出现通道死锁，4个消息的四个片同时占用了4个通道。

2.死锁的避免

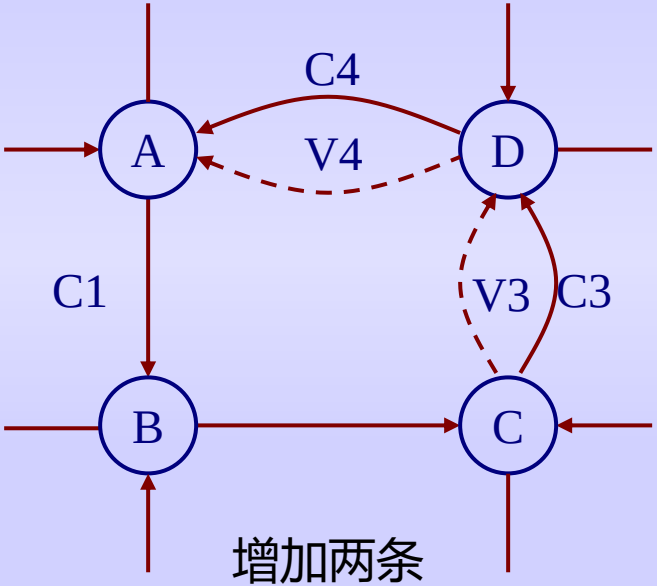


通道死锁

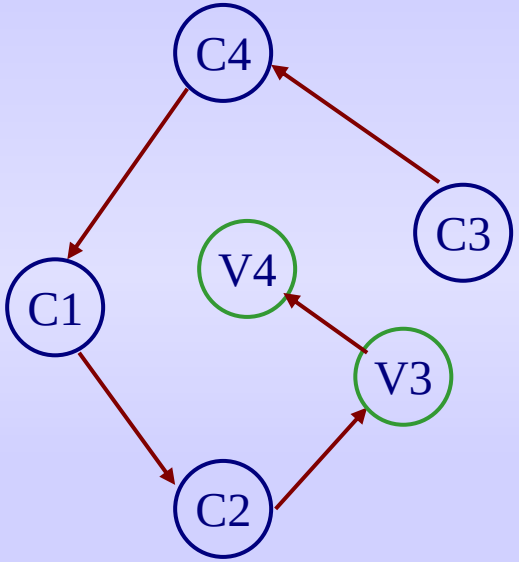


包含循环的通道相关图

结点表示通道，带方向的箭头表示通道之间的依赖关系。



增加两条
虚拟通道
V3,V4



利用虚拟通道后
的通道相关图

利用虚拟通道将通道相关图中的环路变成螺旋线来避免死锁

包冲突的解决

1.问题的提出

两个相邻结点间要传送包，必须具备下列三个条件：

（1）源缓冲区已存该包

（2）通道已分配好

（3）接收缓冲区准备接收

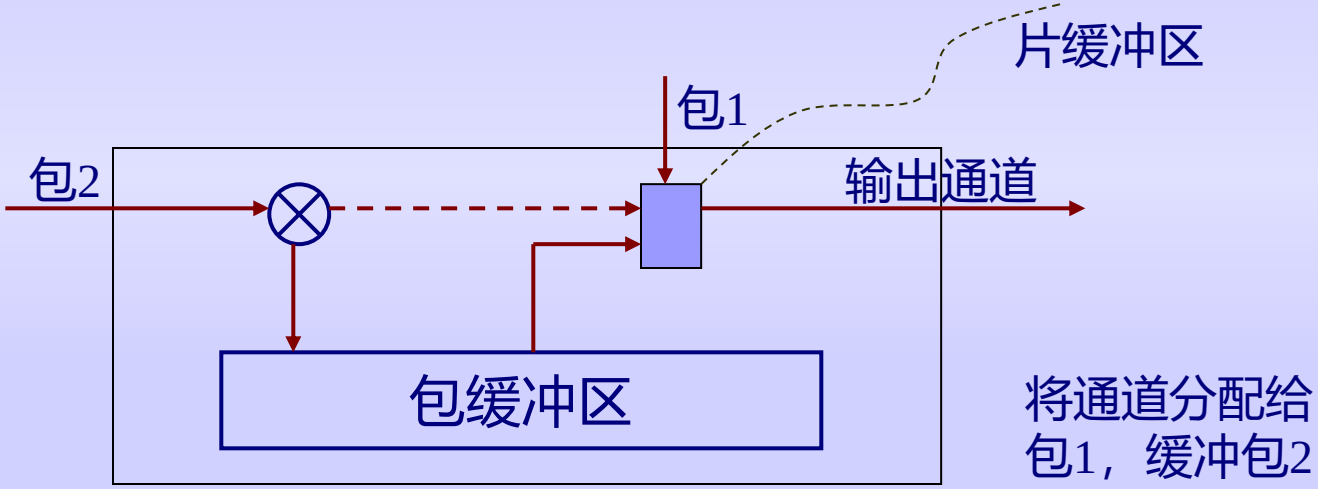
当两个包到达同一个结点时，可能请求同一个接收缓冲区或用同一个输出通道：

（1）把通道分配给哪个包？

（2）没有分配到通道的包怎么办？

2.四种解决方法

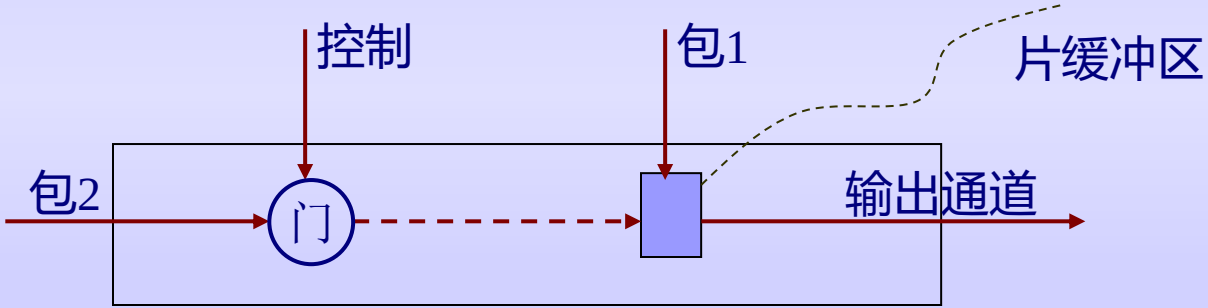
（1）用缓冲实现虚拟直通寻径



好处：不会浪费已经分配的资源

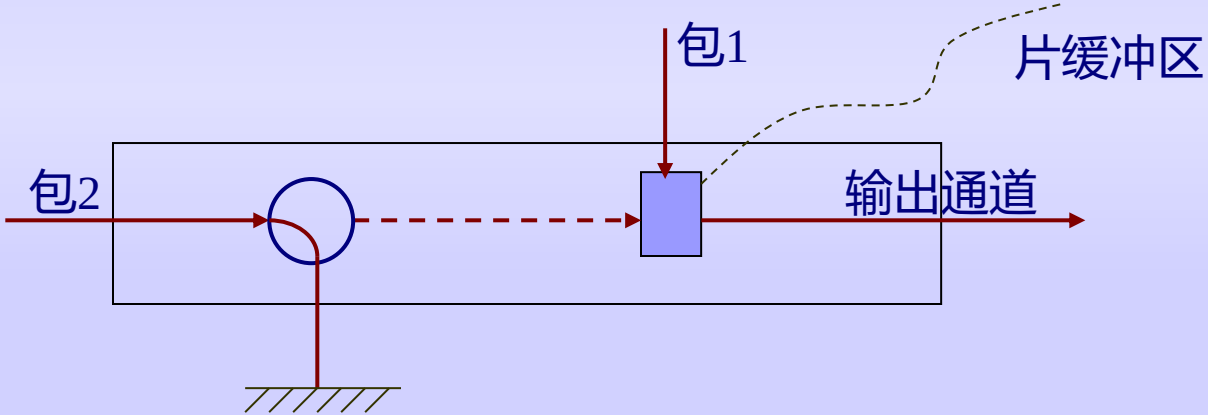
缺点：需要一个能存放整个包的缓冲区，包缓冲区不可能做在寻径芯片上，要用存储器作为缓冲区，会有较大的存储延迟。

（2）阻塞流控制（Wormhole寻径）



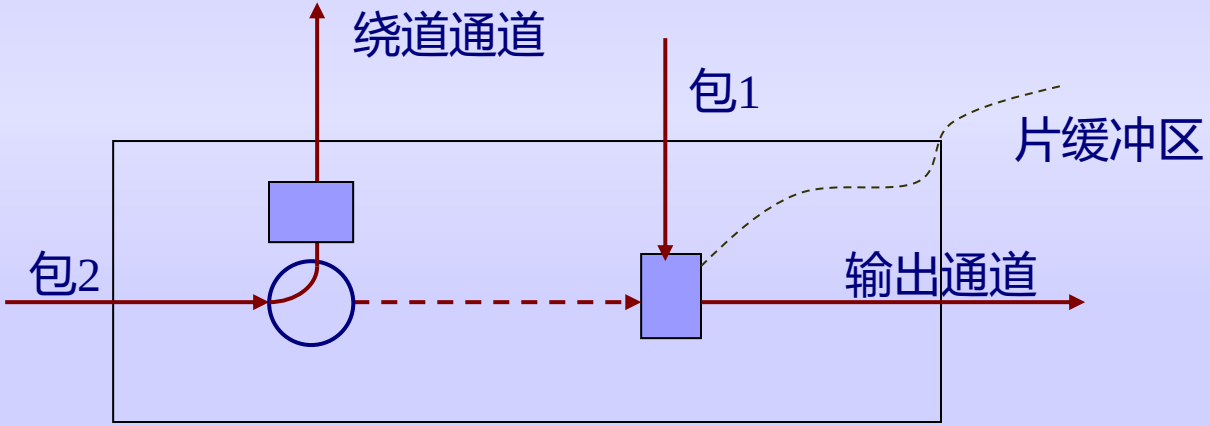
第二个包被阻塞不再前进，但没有被扬弃

（3）扬弃并重发



第二个包被扬弃

（3）阻塞后绕道



第二个包绕道：被转发到其它的寻径器

维序寻径

1.寻径方式

确定寻径（deterministic routing）：

通信路径完全由源和目的地址确定。（换句话说，寻找的路径是预先唯一确定的，与网络的状况无关）。

自适应寻径（adaptive routing）：

与网络的状况有关，可能会有几条路径。（需要消除死锁的算法）。

2.两种确定寻径算法（维序寻径）

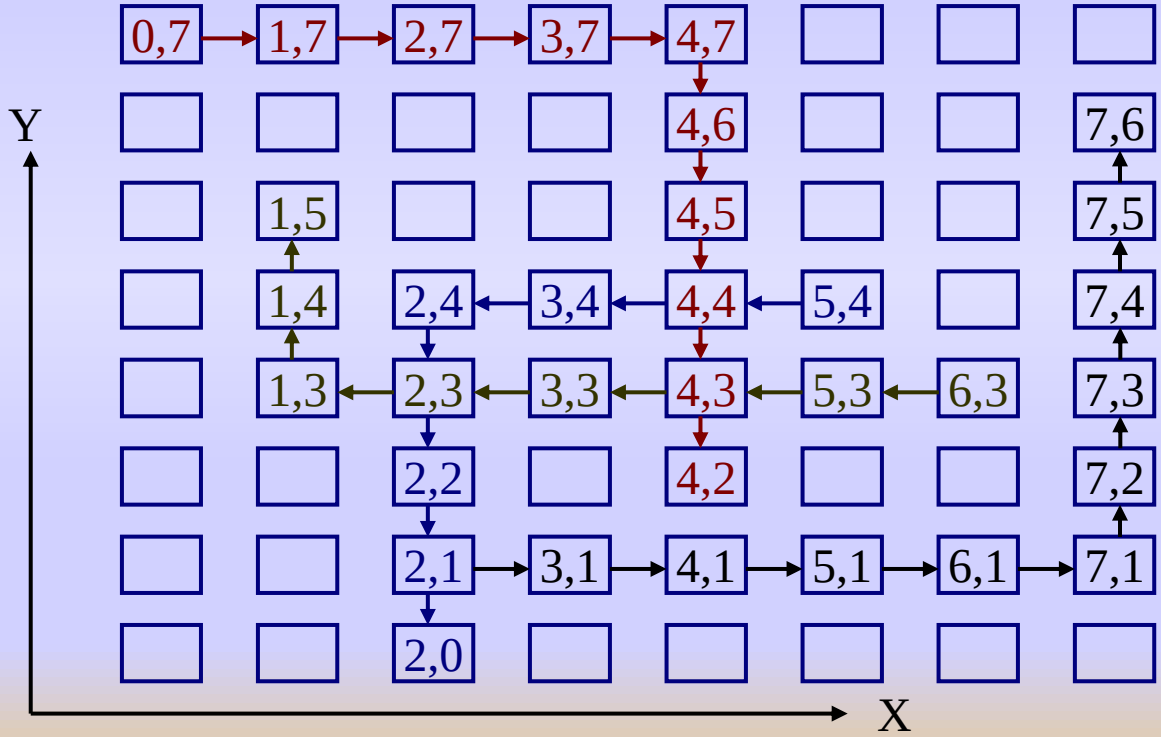
（1）二维网格中的X-Y寻径：

首先沿着X维方向确定路径，然后沿着Y维方向选择路径。

假定从任意源结点s = (X1 Y1)到任意目的结点 d = (X2 Y2)。寻径从s开始，首先沿着X方向前进一直到d所在的第X2列为止，然后沿Y方向前进直到d。

四种模式：

东—北，东—南，西—北，西—南



东—北：(2,1)→(7,6)　　西—南：(5,4)→(2,0)

东—南：(0,7)→(4,2)　　西—北：(6,3)→(1,5)

特点：

总是先沿X维方向寻径，然后再沿Y维方向寻径，寻径不会出现死锁或循环等待现象

可以扩充到n维网络，如X-Y-Z等等。

可用于存储转发或Wormhole寻径网络，在源和目的结点之间形成一条距离最短的路径。

（2）立方体网络中的E立方体寻径：

假设有一个$N = 2^n$个结点的n方体。每个结点的二进制编码为：

$$b = b_{n-1}\, b_{n-2} \ldots b_1\, b_0$$

$$s = s_{n-1}\, s_{n-2} \ldots s_1\, s_0$$

$$d = d_{n-1}\, d_{n-2} \ldots d_1\, d_0$$

如何确定一条从s到d的步数最小的路径？

将n维表示成$i = 1, 2, \ldots n$，其中第i维对应结点地址中的第$i-1$位。设

$$v = v_{n-1}\, v_{n-2} \ldots v_1\, v_0$$

是路径中的任一结点。

方法：

（1）计算方向位。

$$r_i = S_{i-1} \oplus d_{i-1}, \text{其中} i = 1,2,\cdots n。$$

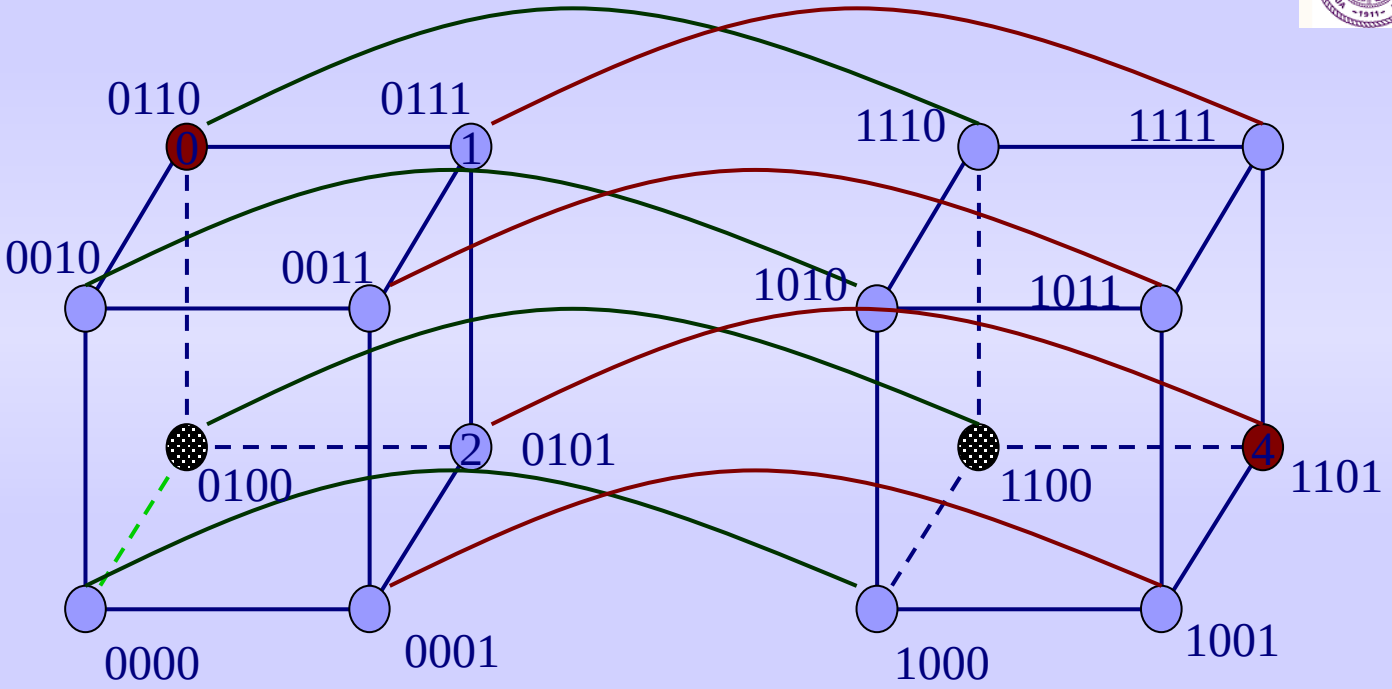
使 i = 1 , v = s， 开始下面的步骤。

（2）如果$r_i = 1$，则从当前结点v寻径到下一结点$v \oplus 2^{i-1}$ ；

如果$r_i = 0$，则跳过这一步。

（3）i = i + 1，如果 $i \le n$，则转第（2）步，否则退出。

如下面的例子：



4维立方体网络

n = 4，s = 0110，d = 1101

寻径：

（1）计算方向位。i = 1，v = s

$$\begin{array}{cccc} 0 & 1 & 1 & 0 \\ \oplus\ 1 & 1 & 0 & 1 \\ \hline 1 & 0 & 1 & 1 \end{array}$$

（ R= r_4 r_3 r_2 r_1 ）

(2) $r_1 = 1$,

$\therefore s$到$s \oplus 2^{1-1} = 0110 \oplus 0001 = 0111$

$i = i + 1 = 2$

(3) $r_2 = 1$,

$\therefore v = 0111$到$v \oplus 2^{2-1} = 0111 \oplus 0010 = 0101$

$i = i + 1 = 3$

(4) $r_3 = 0$,

$\therefore$ 跳过一步

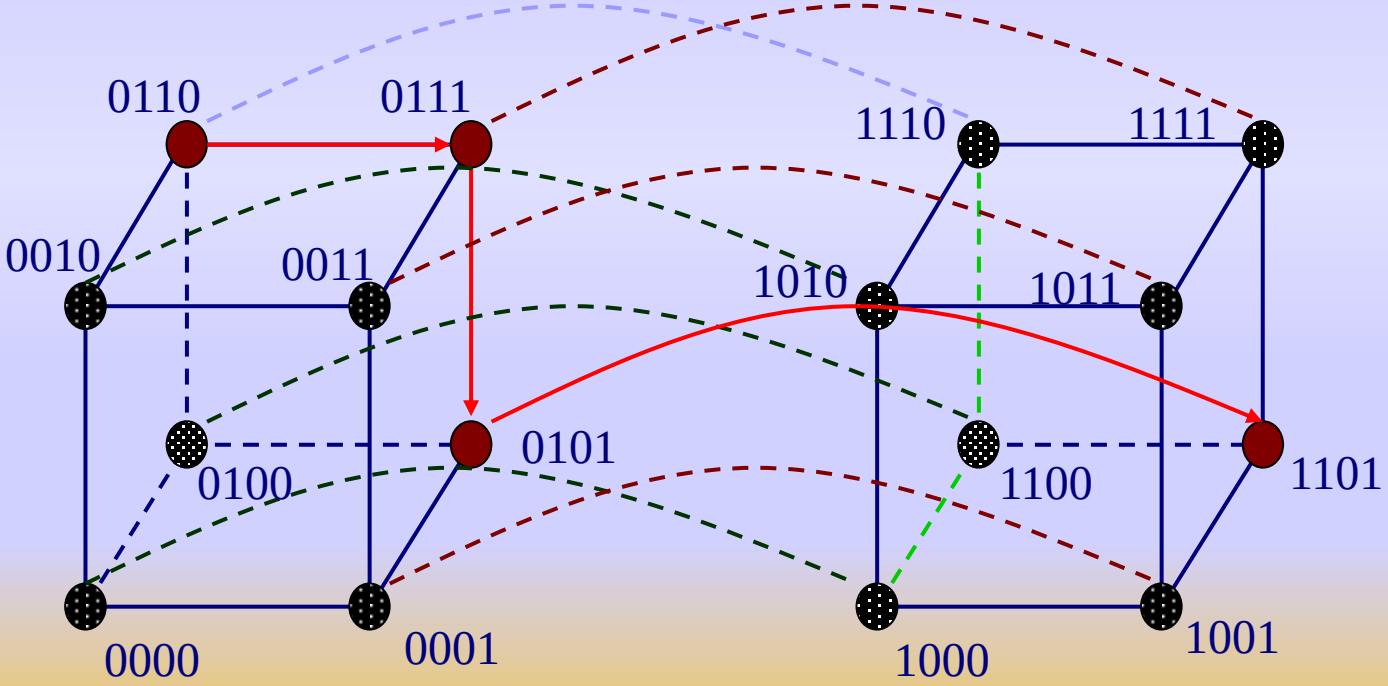
$i = i + 1 = 4$

(5) $r_4 = 1$,

$\therefore v = 0101$到$v \oplus 2^{4-1} = 0101 \oplus 1000 = 1101 = d$

$i = i + 1 = 5$, 结束。

路径为：0110 →0111→0101→1101



特点：

寻径按照从维1到维4的顺序进行。

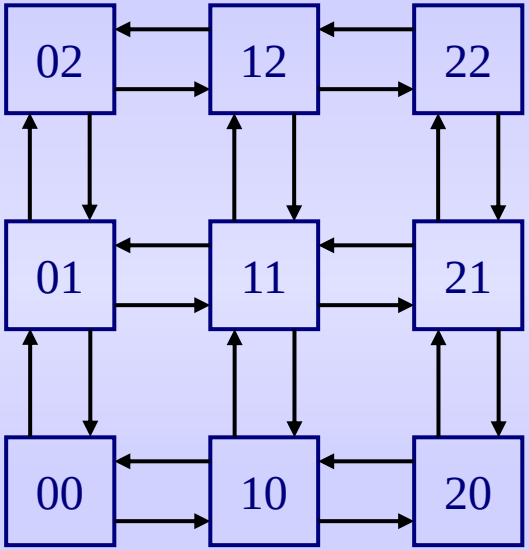
如果s和d的第i位相同，则沿维i的方向不需要寻径，否则从当前结点沿着这一维方向走向下一结点（立方体中，每维包括两个结点）。重复这一过程直到到达目的结点。

3.自适应寻径

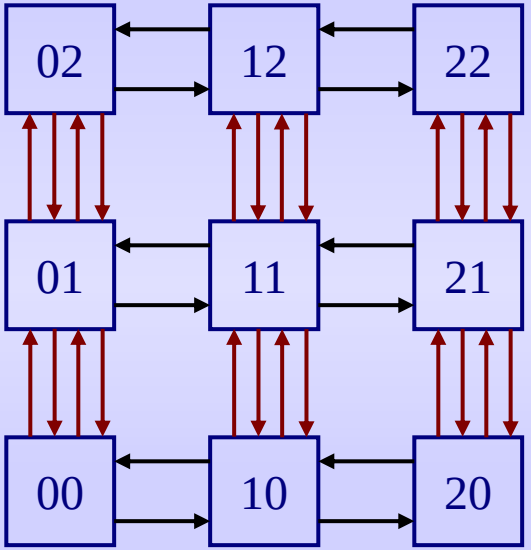
目的：避免死锁

虚拟通道：使实现自适应寻径更经济和更灵活。

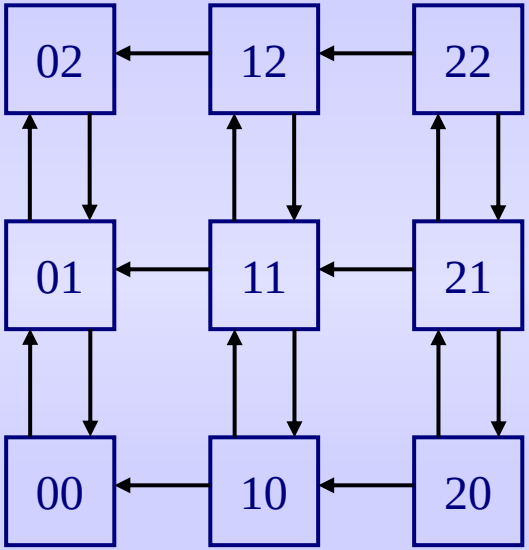
方法：网格网络中，同一维的所有连接都使用虚拟通道。



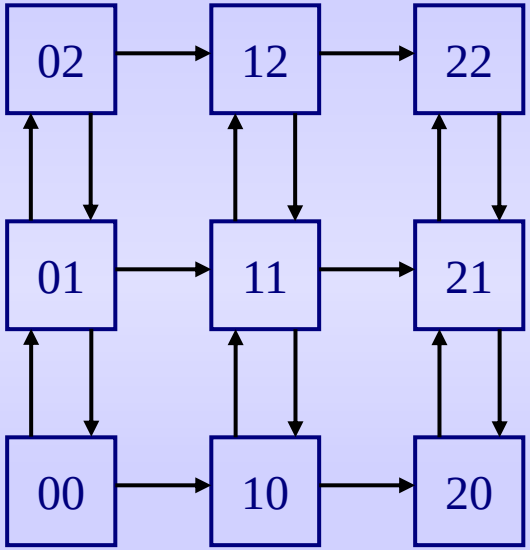
（a）没有虚拟通道的原型网络



（b）Y维方向有两对虚拟通道



（c）向西方向传递消息



（d）向东方向传递消息

通信模式

1.通信模式

单播模式（Unicast）：一个源结点——一个目的结点。

选播模式（Multicast）：一个源结点——多个目的结点。（多播、组播）

广播模式（Broadcast）：一个源结点——全体结点。

会议模式（Conference）：多个源结点——多个目的结点。

2.寻径效率

通信流量（Channel traffic）：用传输有关消息所使用的通道数来表示。

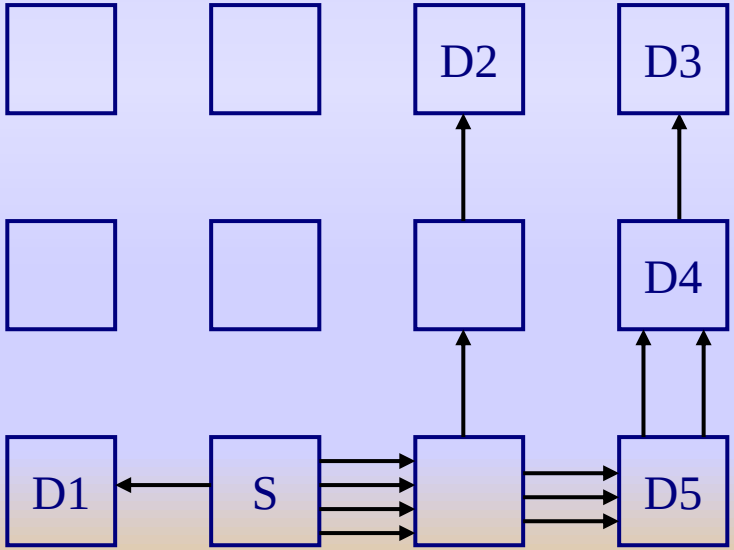
通信时延（Communication latency）：用包的最长传输时间来表示。

Wormhole寻径方式下，网络流量这个参数比较重要。

在存储转发网络中，时延是最重要的问题。

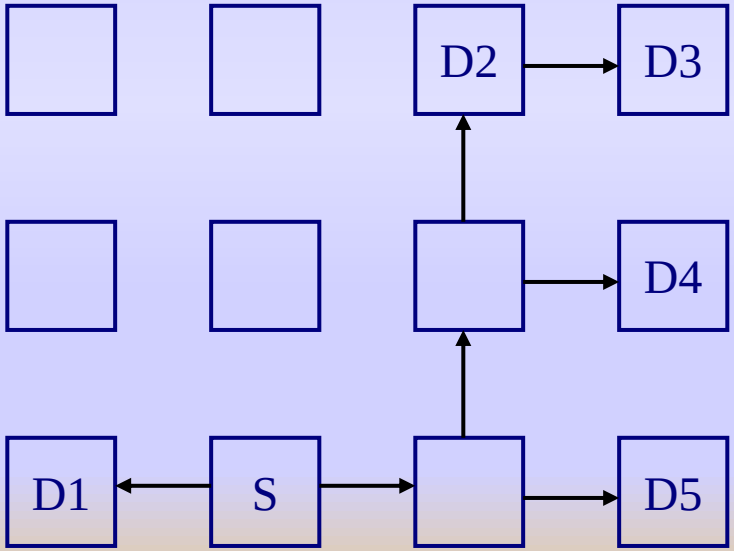
例：

网格连接计算机中的选播和广播。3×4网格上实现选播寻径

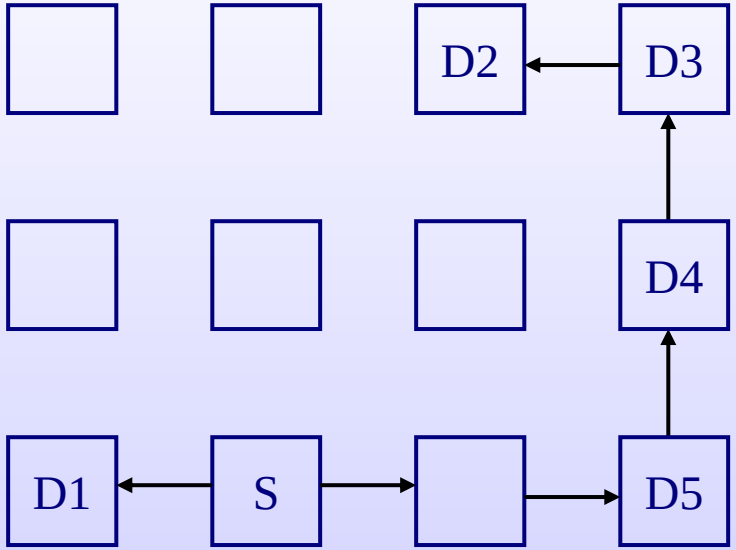


（a）5次单播，流量等于13，距离等于4

在一个中间结点上复制所传送的包，然后把该包的多个拷贝送到目的结点，这样可以减少通道流量。



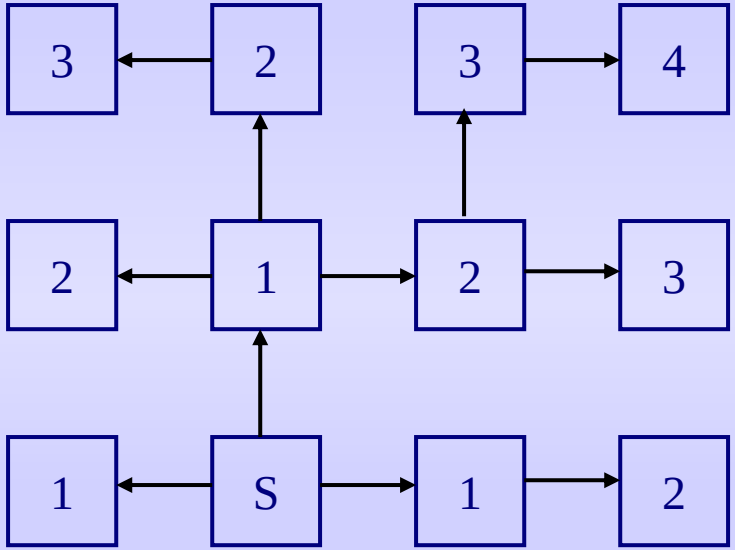
（b）一种流量等于7，距离等于4的选播



（c）流量等于6，距离等于5的选播

（b）中的选播方式对存储转发比较好

（c）中的选播方式对Wormhole寻径较好



(d) 通过树结构广播所有结点

结点中的数字表示树的层次号，该树称为广播树。

3.Active Message

由UC Berkeley开发。

背景：互连网络的硬件开销已经相当小了，通信和计算机不能相互重叠，消息发生和接收原语带来过大的系统软件开销。

互连网络接收源处理结点发来的消息包，并采用一定的寻径算法和流控制技术将它送到目的处理结点。消息包到达后，用一个中断信号去通知目的处理结点。此时目的处理结点会调用相应的中断处理函数来处理到达的消息包。——这是一个完全异步的过程。

传统的通信机制之所以会有比较大的软件开销，其实质上的原因就是过多的采用了与上述异步过程不相匹配的同步协议。

Active Message：消息发送方来指明消息处理函数的地址，当消息到达时，这一函数就自动的被调用。——简单得几乎没有任何协议的异步通信机制。

好处：

不需要任何缓冲

实现简单

Active Message 的消息格式