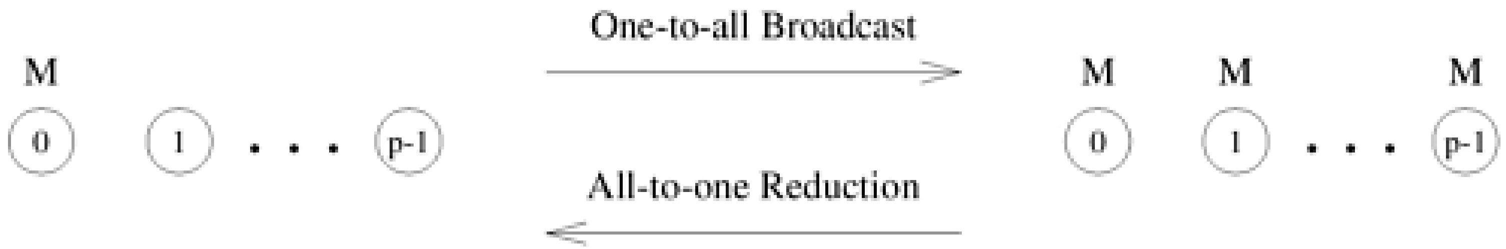
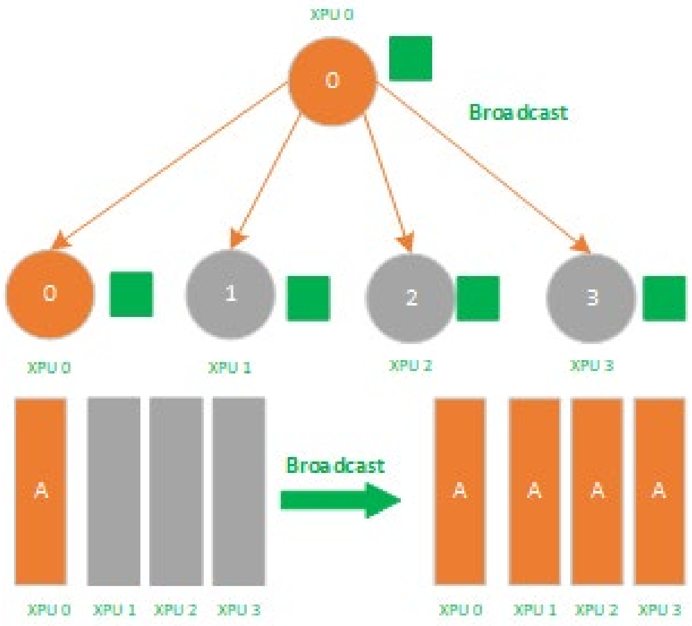
Chapter 4

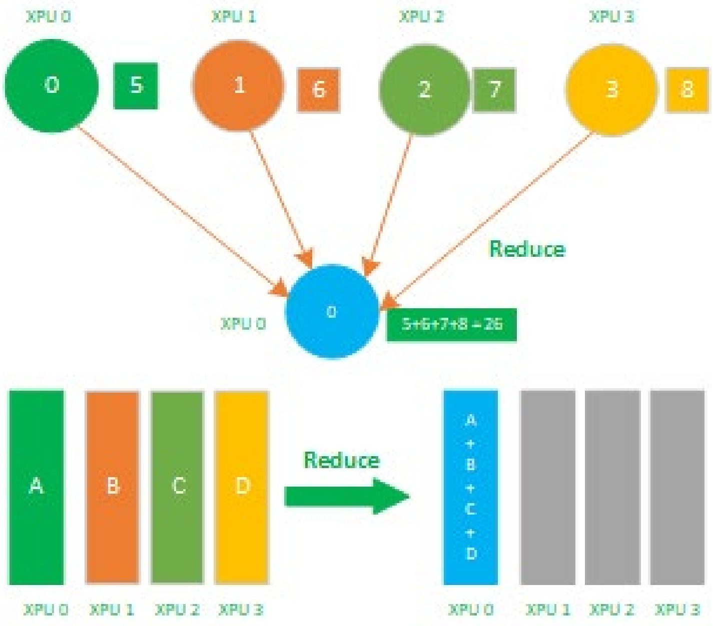
Basic Communication Operations

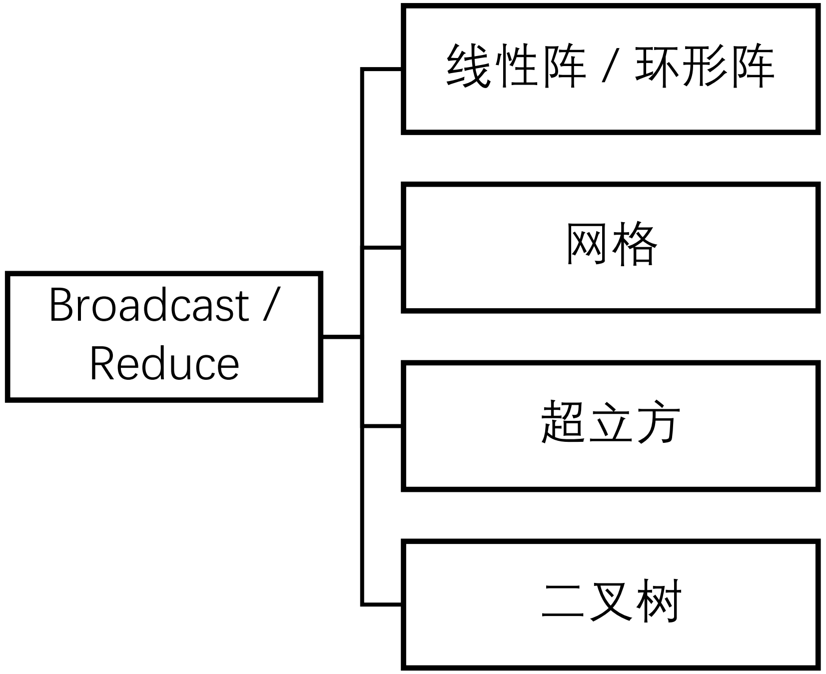
2024/06/24

4.1 One-to-All Broadcast / All-to-One Reduction



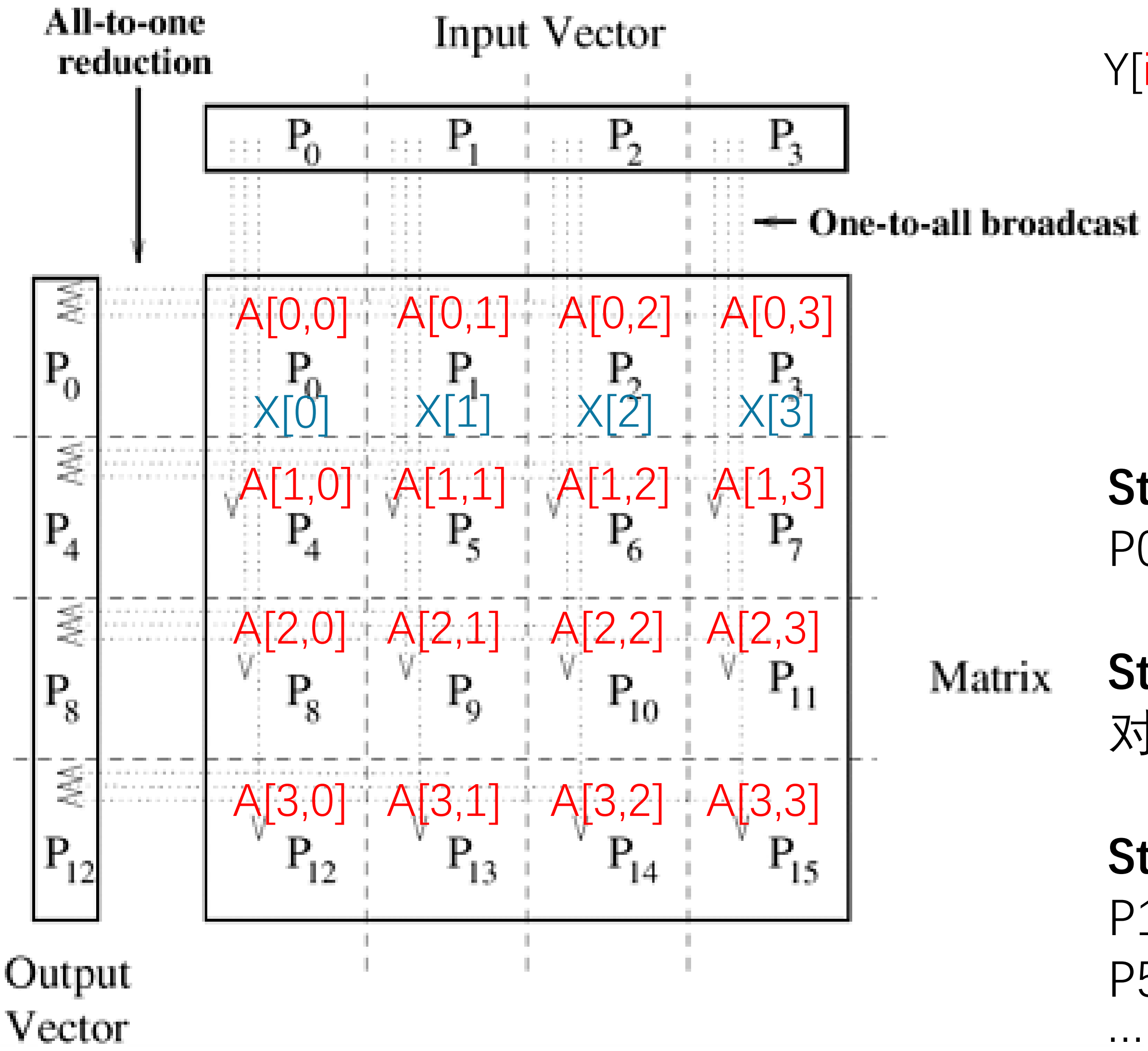






4.1 One-to-All Broadcast / All-to-One Reduction

Application Case: Matrix-Vector Multiplication



位于P0 位于P1

Y[i] = A[i,0]*X[0]+A[i,1]*X[1]+ A[i,2]*X[2]+A[i,3]*X[3]

… …

在P1， P5， P9， P13上计算

在P0， P4， P8， P12上计算

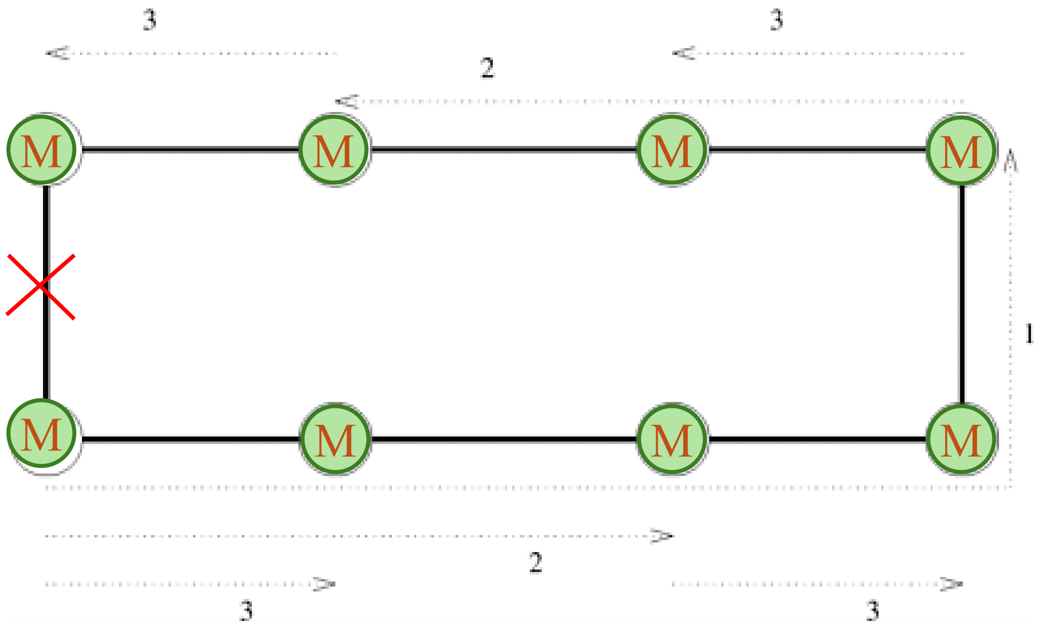
Step 1
P0/P1/P2/P3中对应的X数据Broadcast到对应的进程上

Step2
对应的进程进行计算

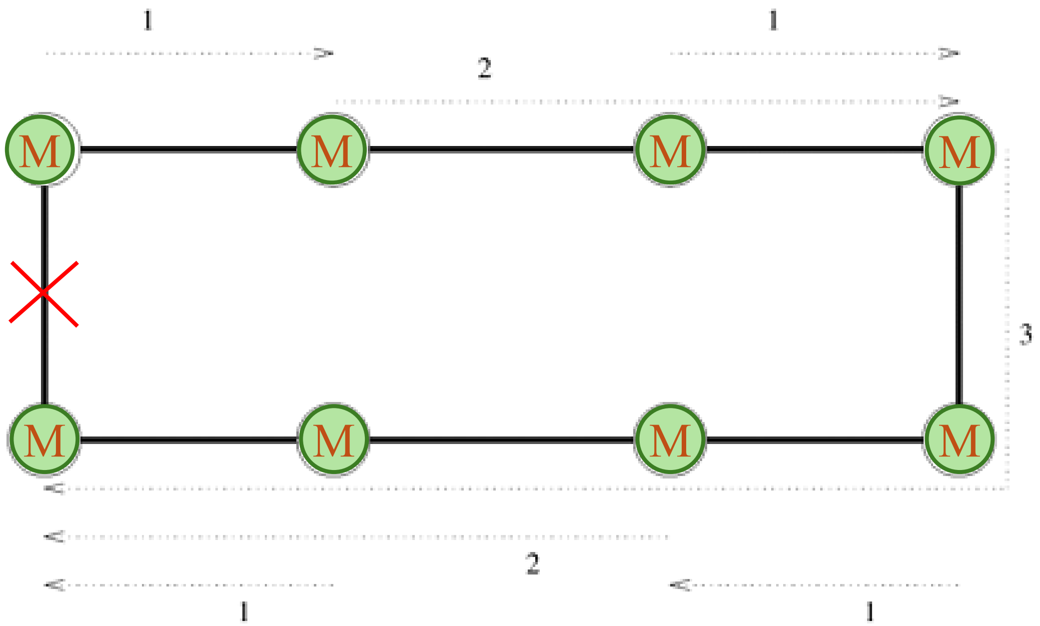
Step3
P1/P2/P3把计算结果Reduce到P0上
P5/P5/P7把计算结果Reduce到P4上
…

4.1 One-to-All Broadcast / All-to-One Reduction

Ring or Linear Array



Broadcast



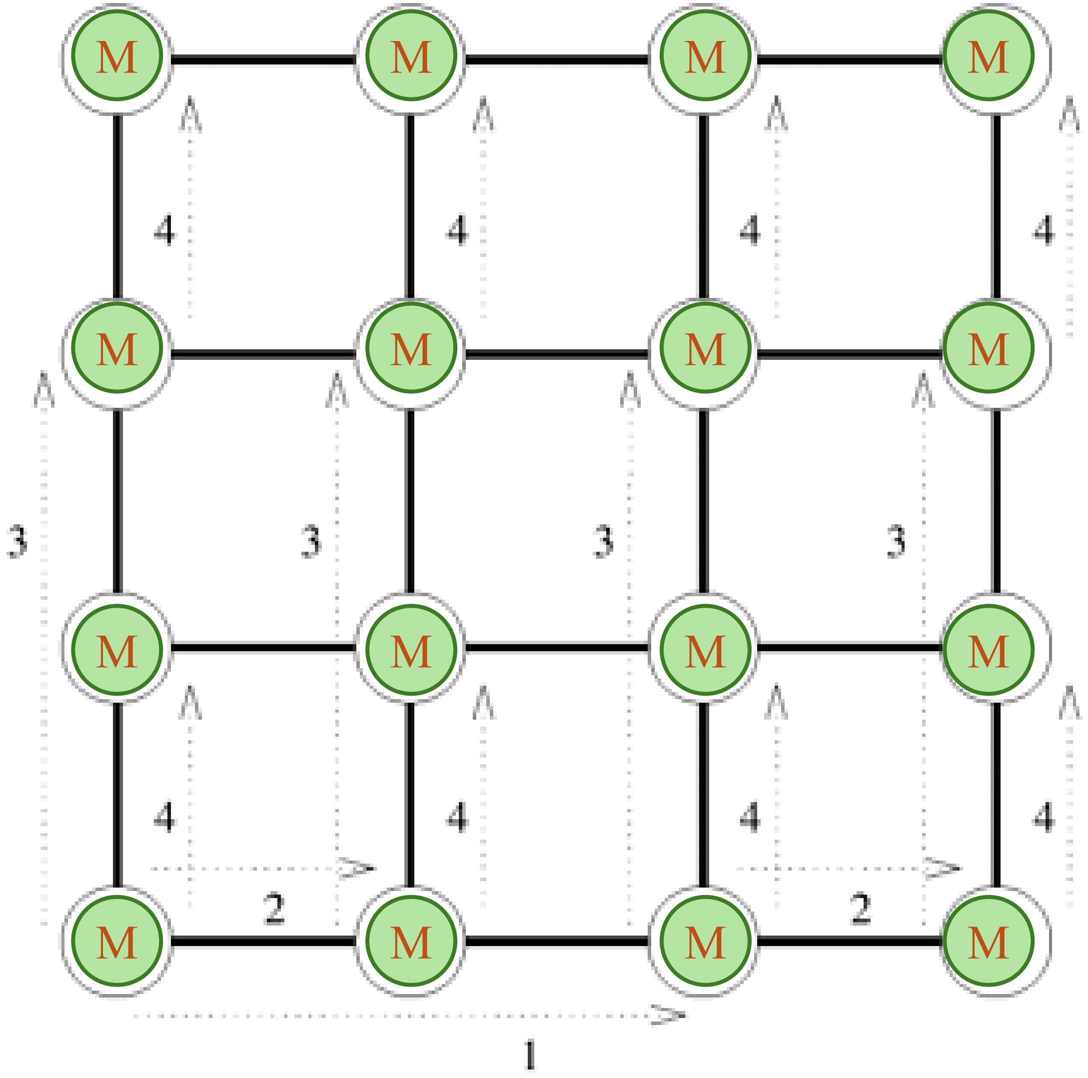
Reduce

$$T = (t_s + mt_w) \ +(t_s + mt_w) + (t_s + mt_w)$$

$$T = (t_s + mt_w)\log(p)$$

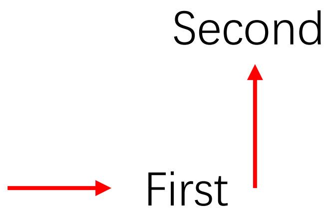
4.1 One-to-All Broadcast / All-to-One Reduction

Mesh Broadcast



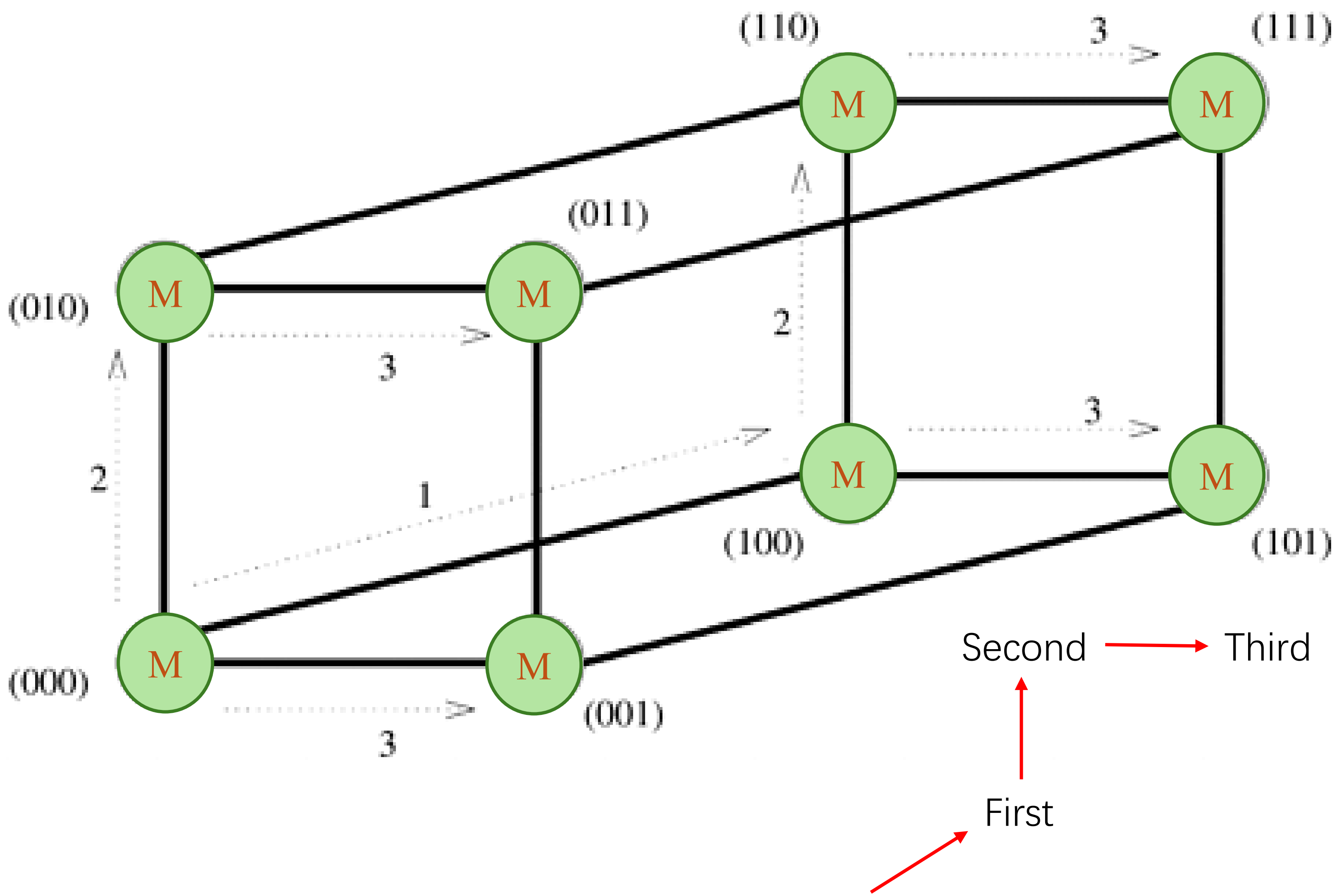
$$T = (t_s + mt_w) + (t_s + mt_w) + (t_s + mt_w) + (t_s + mt_w)$$

$$T = (t_s + mt_w)\log(p)$$



4.1 One-to-All Broadcast / All-to-One Reduction

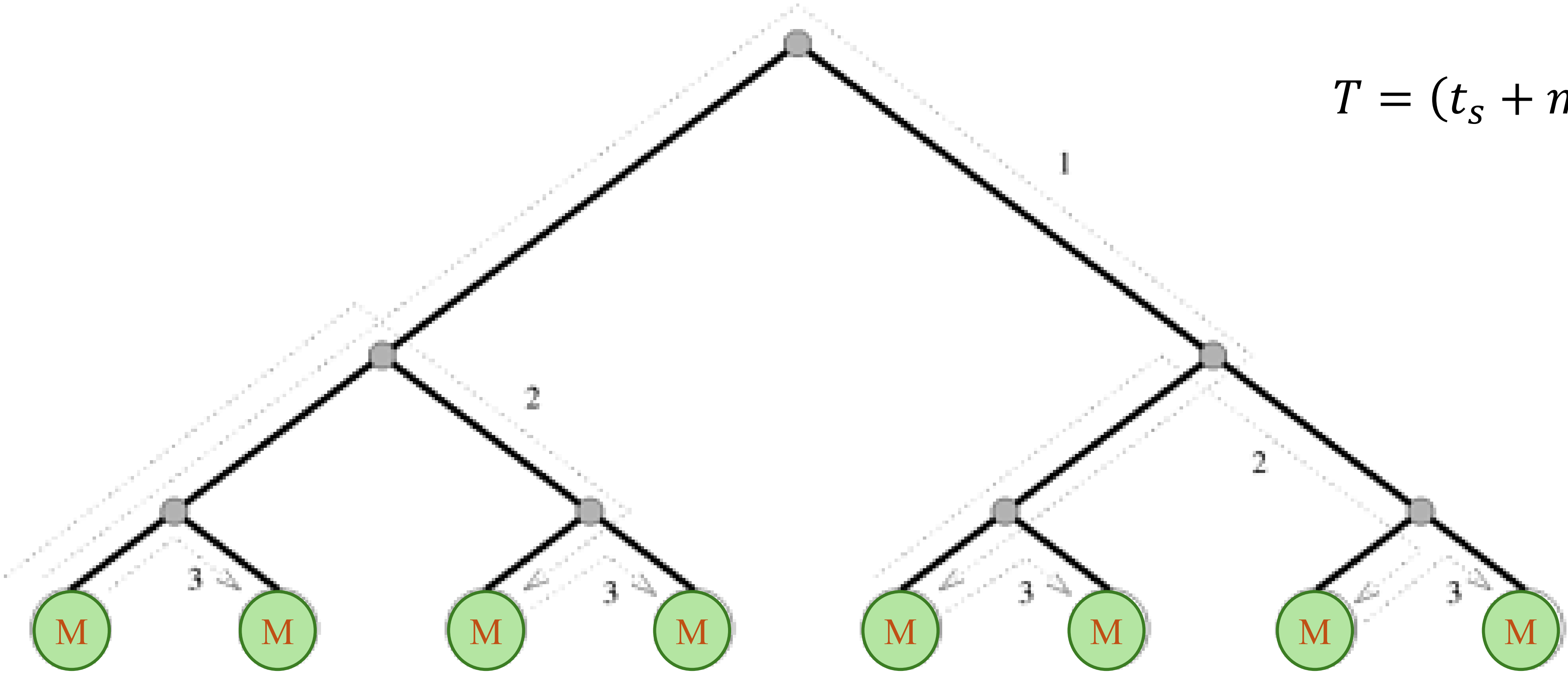
Hypercube Broadcast



$$T = (t_s + mt_w)\log(p)$$

4.1 One-to-All Broadcast / All-to-One Reduction

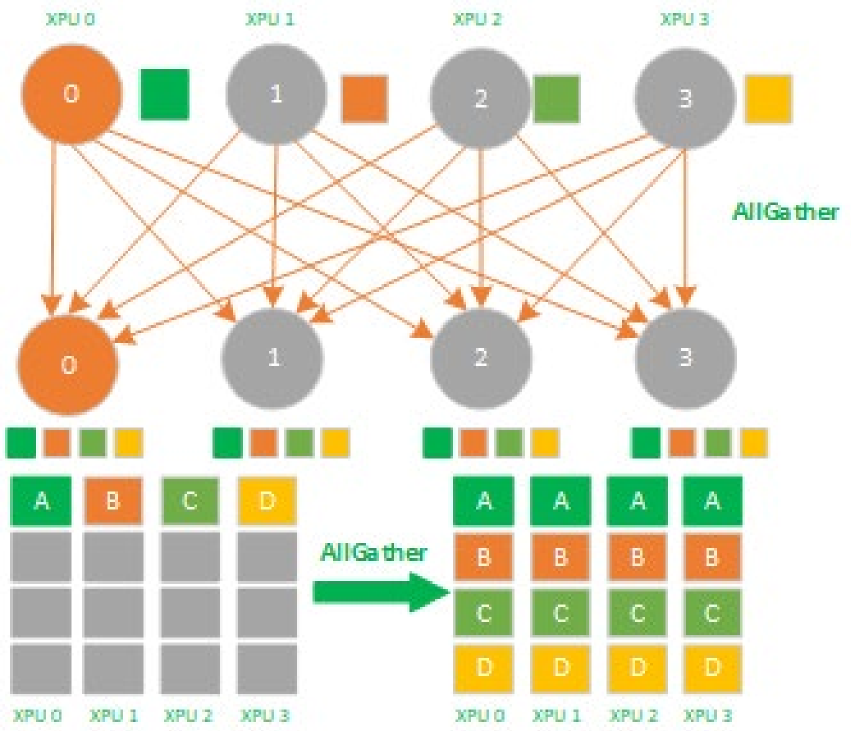
Balanced Binary Tree Broadcast

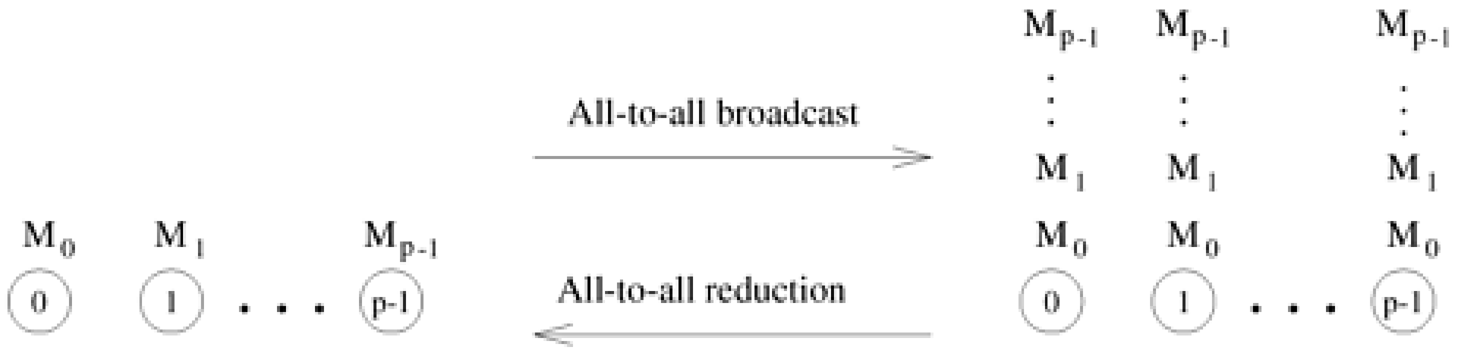


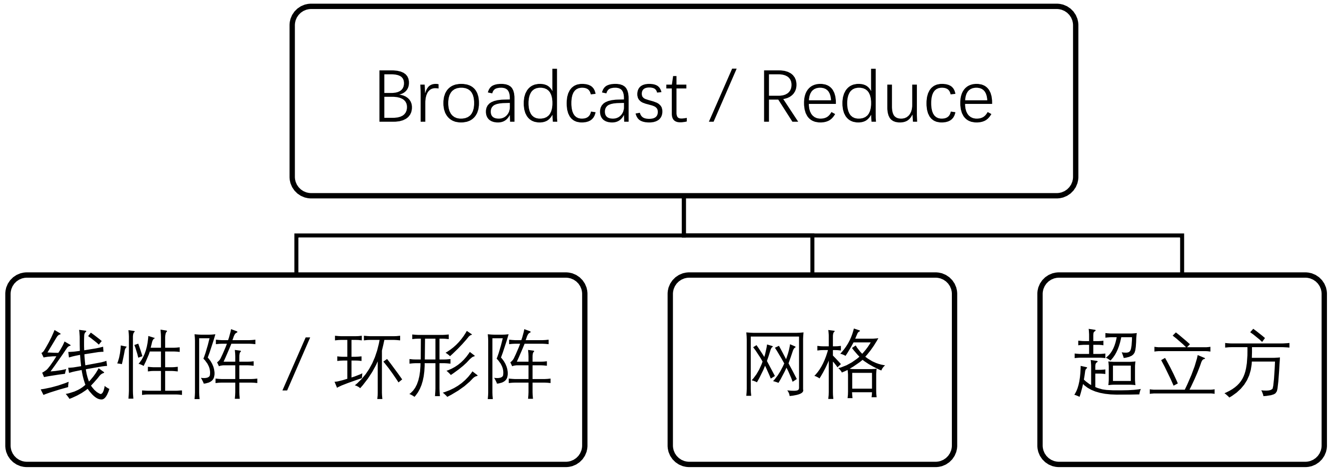
$$T = (t_s + mt_w)\log(p)$$

4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

All-to-all broadcast is a generalization of one-to-all broadcast in which all p nodes simultaneously initiate a broadcast



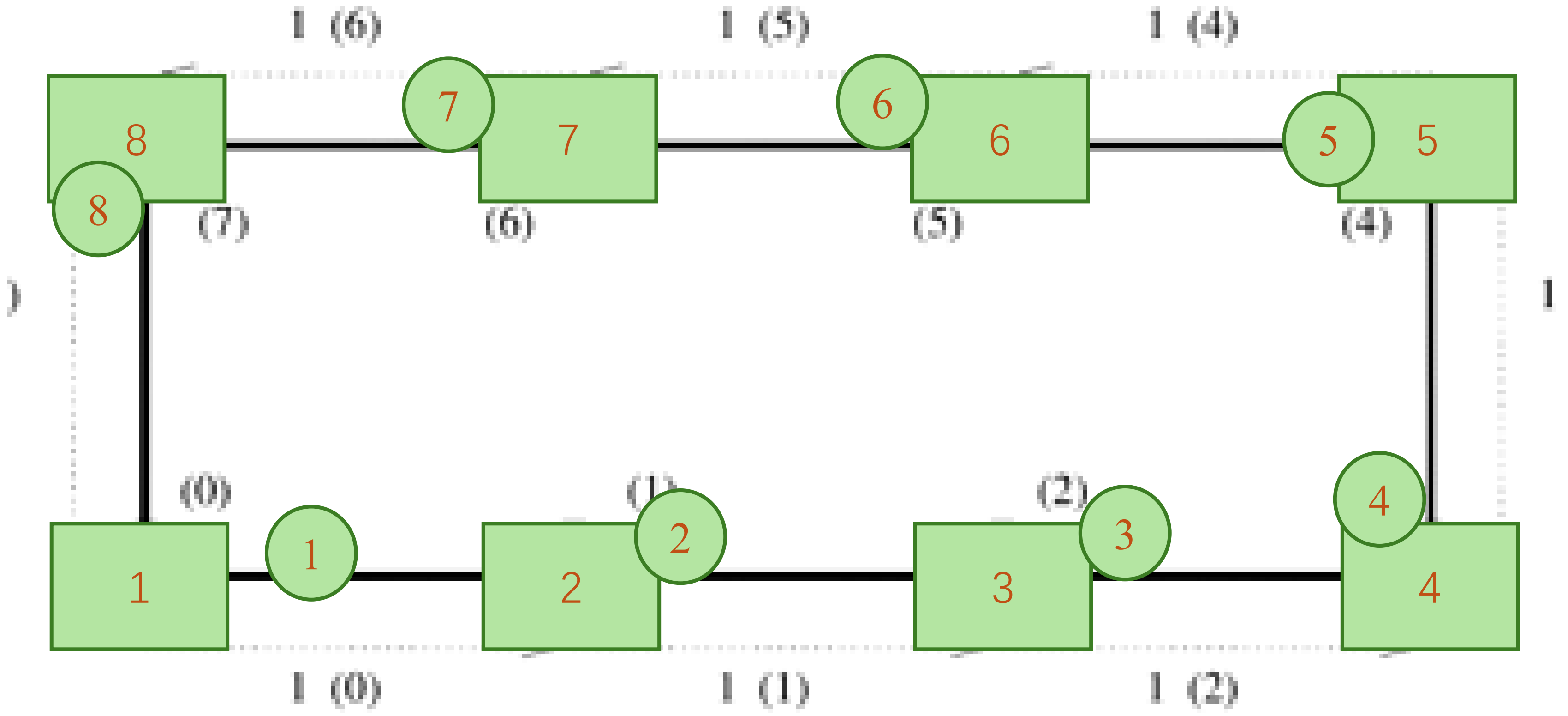




4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

Linear Array and Ring Broadcast

The same procedure would also work on a linear array with bidirectional links.

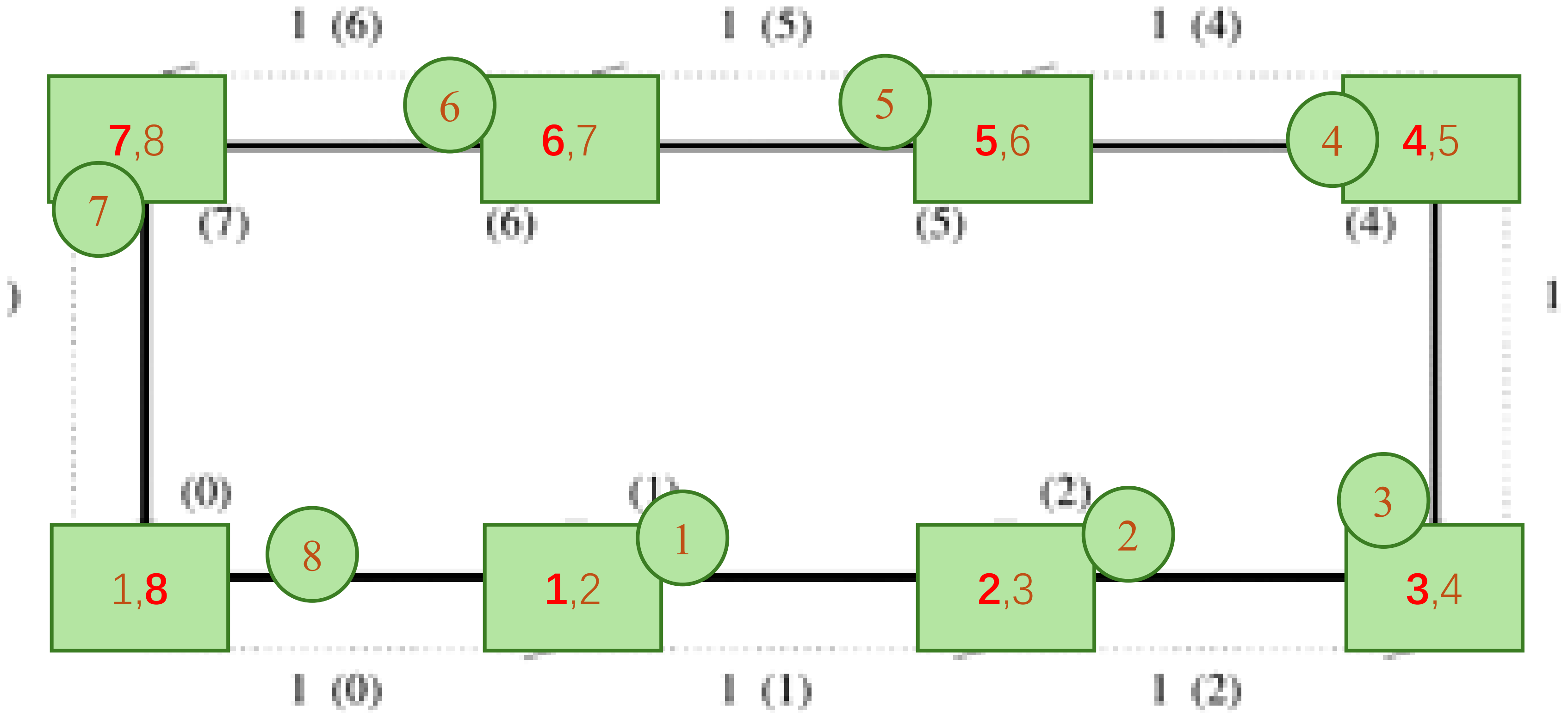


$$T = (t_s + mt_w)$$

4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

Linear Array and Ring Broadcast

The same procedure would also work on a linear array with bidirectional links.

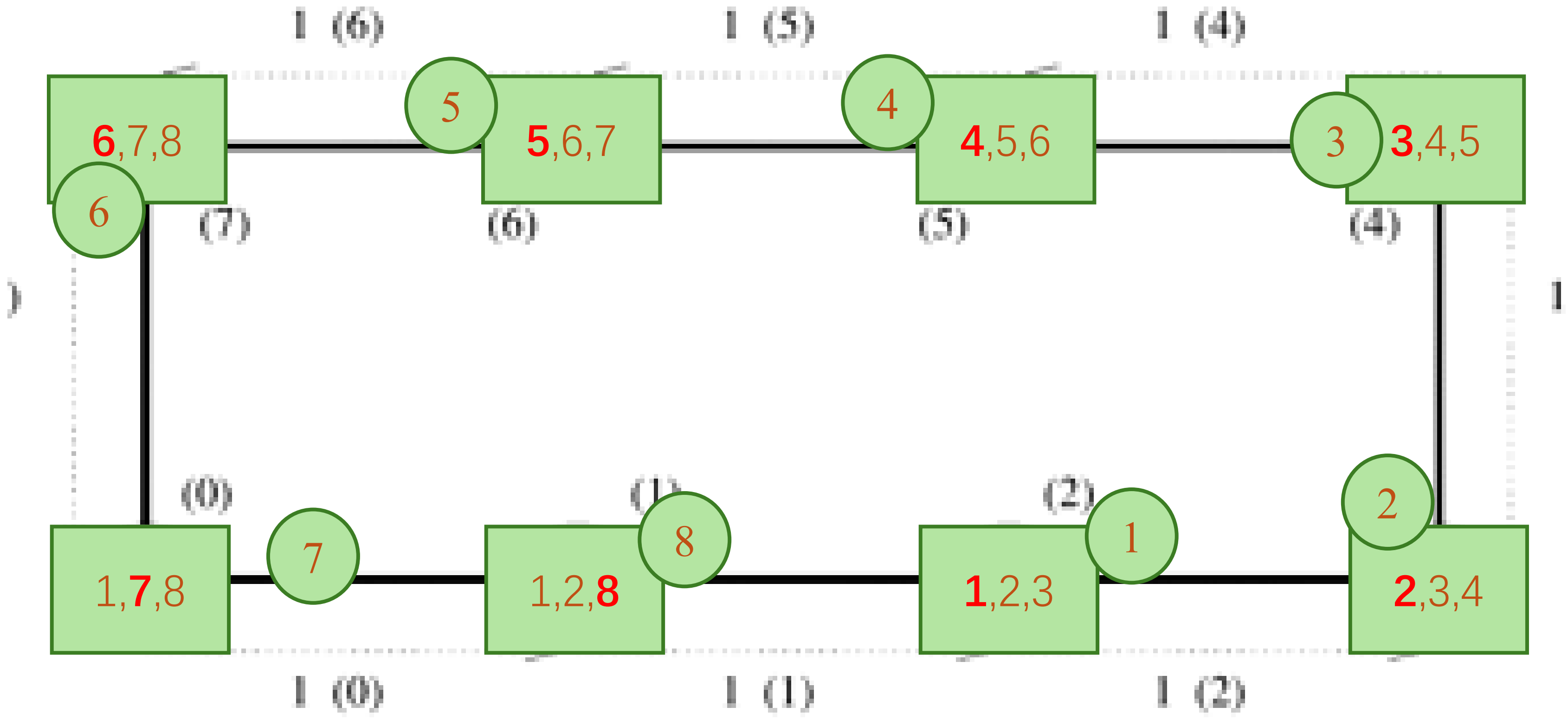


$$T = (t_s + mt_w) + (t_s + mt_w)$$

4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

Linear Array and Ring Broadcast

The same procedure would also work on a linear array with bidirectional links.

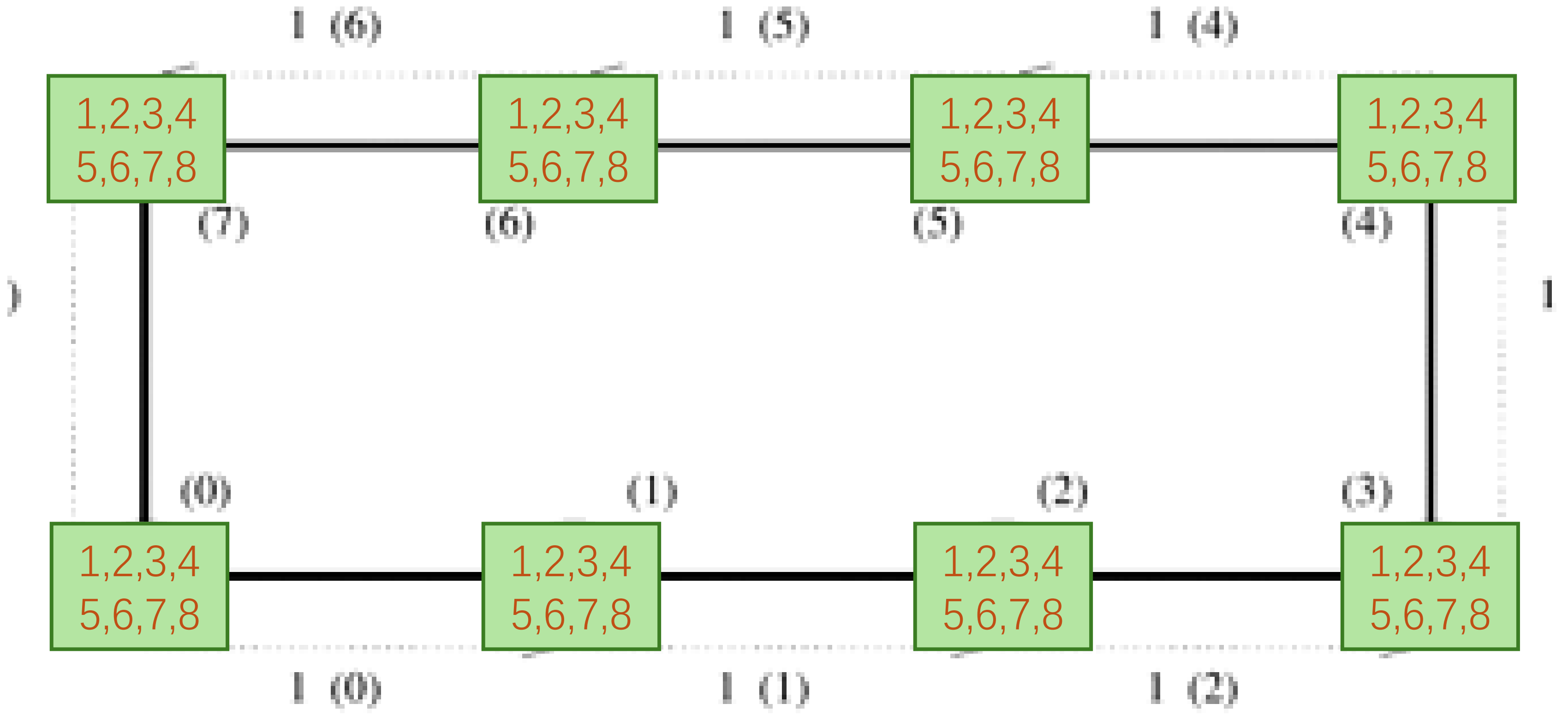


$$T = (t_s + mt_w) + (t_s + mt_w) + (t_s + mt_w)$$

4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

Linear Array and Ring Broadcast

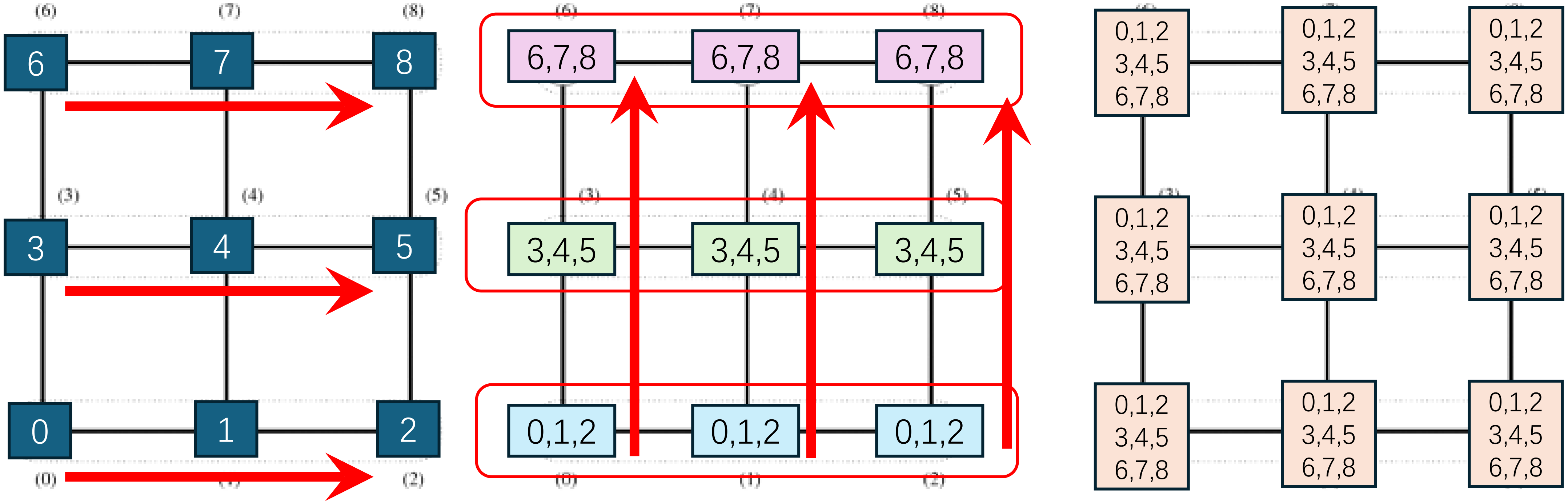
The same procedure would also work on a linear array with bidirectional links.



$$T = (t_s + m t_w)(p - 1)$$

4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

Mesh Broadcast



$$(t_s + m_1 t_w)(\sqrt{p} - 1)$$

$$m_1 = m$$

$$+$$

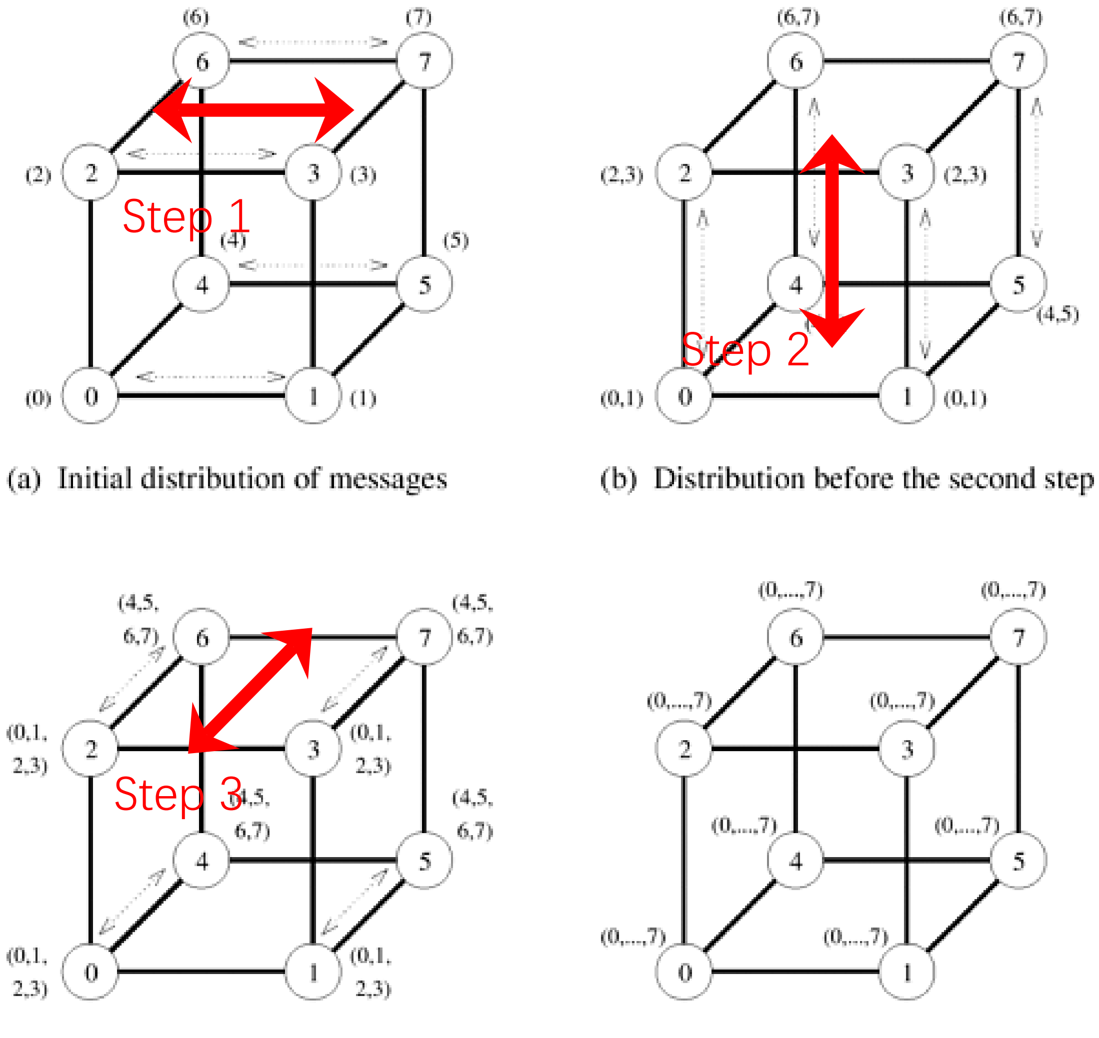
$$(t_s + m_2 t_w)(\sqrt{p} - 1)$$

$$m_2 = m_1 * \sqrt{p}$$

$$T = 2t_s(\sqrt{p} - 1) + mt_w(p - 1)$$

4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

Hypercube Broadcast



(a) Initial distribution of messages

(b) Distribution before the second step

(c) Distribution before the third step

(d) Final distribution of messages

Step 1

$$T_1 = (t_s + m_1 t_w), m_1 = m = 2^0 m$$

Step 2

$$T_2 = (t_s + m_2 t_w), m_2 = 2m_1 = 2^1 m$$

Step 3

$$T_3 = (t_s + m_3 t_w), m_3 = 2m_2 = 2^2 m$$

Step n

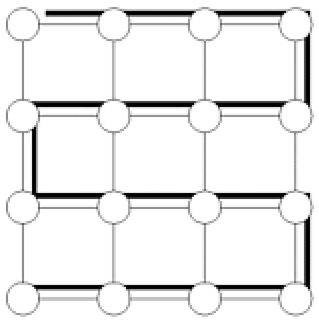
$$T_n = (t_s + m_n t_w), m_n = 2^{n-1} m$$

$$T = \sum_{i=1}^{\log p} (t_s + 2^{i-1} m t_w)$$

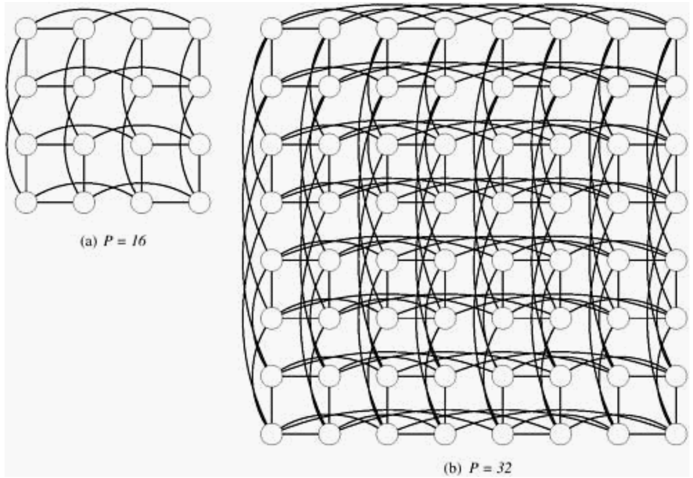
$$= t_s \log p + m t_w (p - 1)$$

4.2 All-to-All Broadcast (All Gather) / All-to-All Reduction (Reduce Scatter)

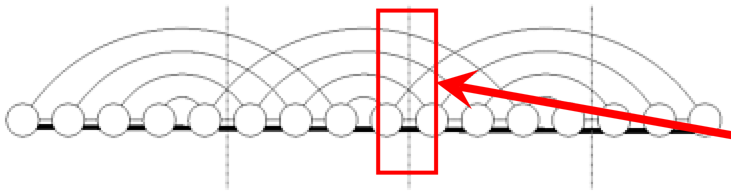
Caution



(a) Mapping a linear array into a 2D mesh (congestion 1).

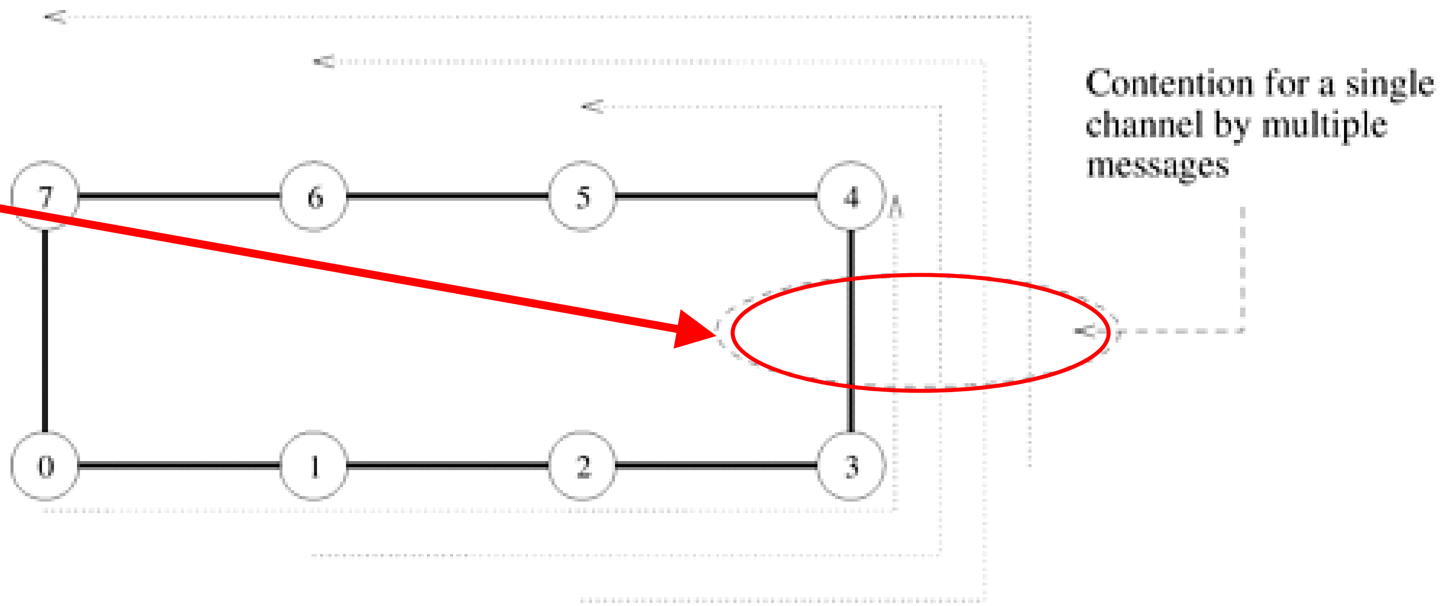


HyperCube到Mesh的映射



(b) Inverting the mapping – mapping a 2D mesh into a linear array (congestion 5)

Mesh到Linear的映射



4.3 All-Reduce / Prefix-Sum

All Reduce

> **All-to-All Reduction**：p个进程同时进行One-to-All Reduction，且Reduce的目的地不同
>
> **All Reduce**：All-to-All 的一种扩展

All Reduce = All-to-One Reduction + All-to-All Broadcast

= All-to-All Broadcat * Reduce

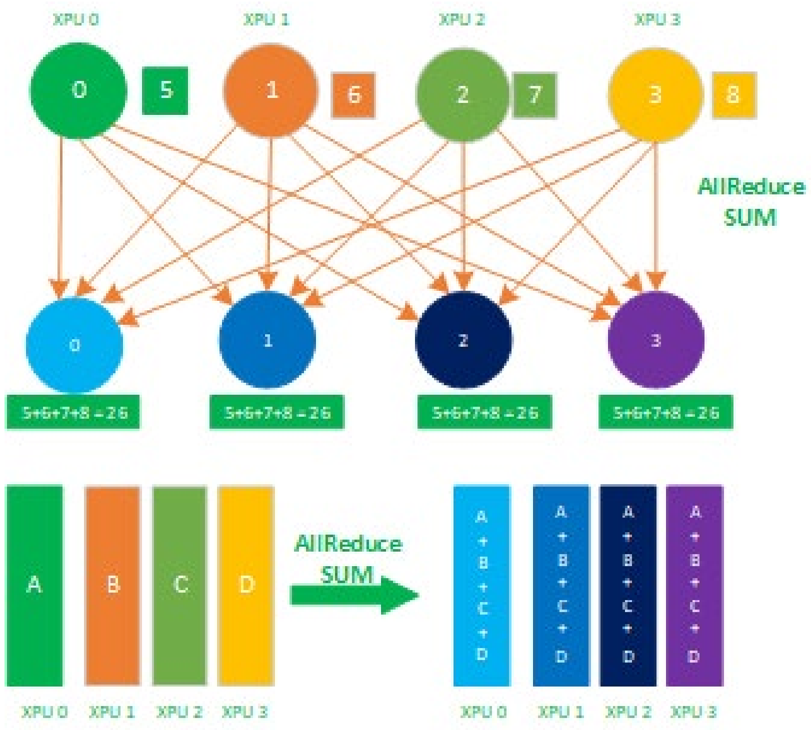
通信的消息量不翻倍

All-to-All Broadcast

$$T = \sum_{i=1}^{\log p} (t_s + 2^{i-1} m t_w)$$

All Reduce

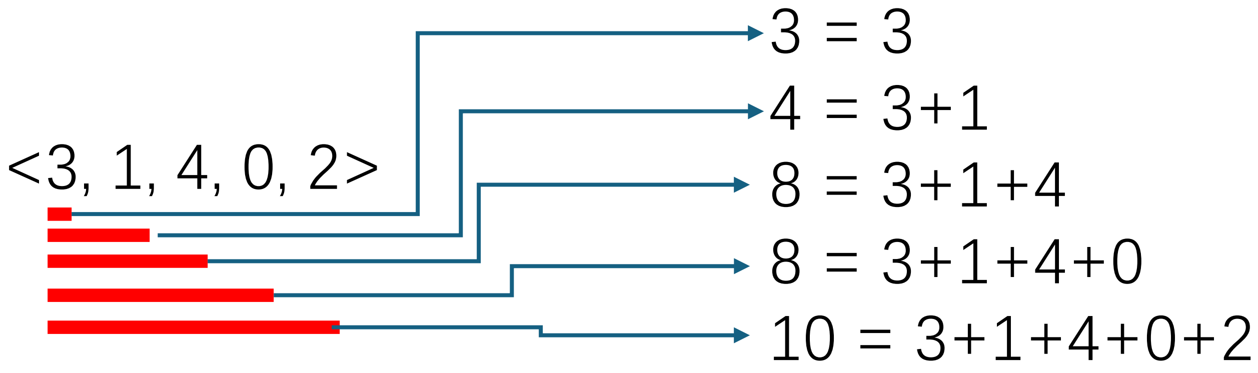
$$T = (t_s + m t_w) \log p$$



4.3 All-Reduce / Prefix-Sum

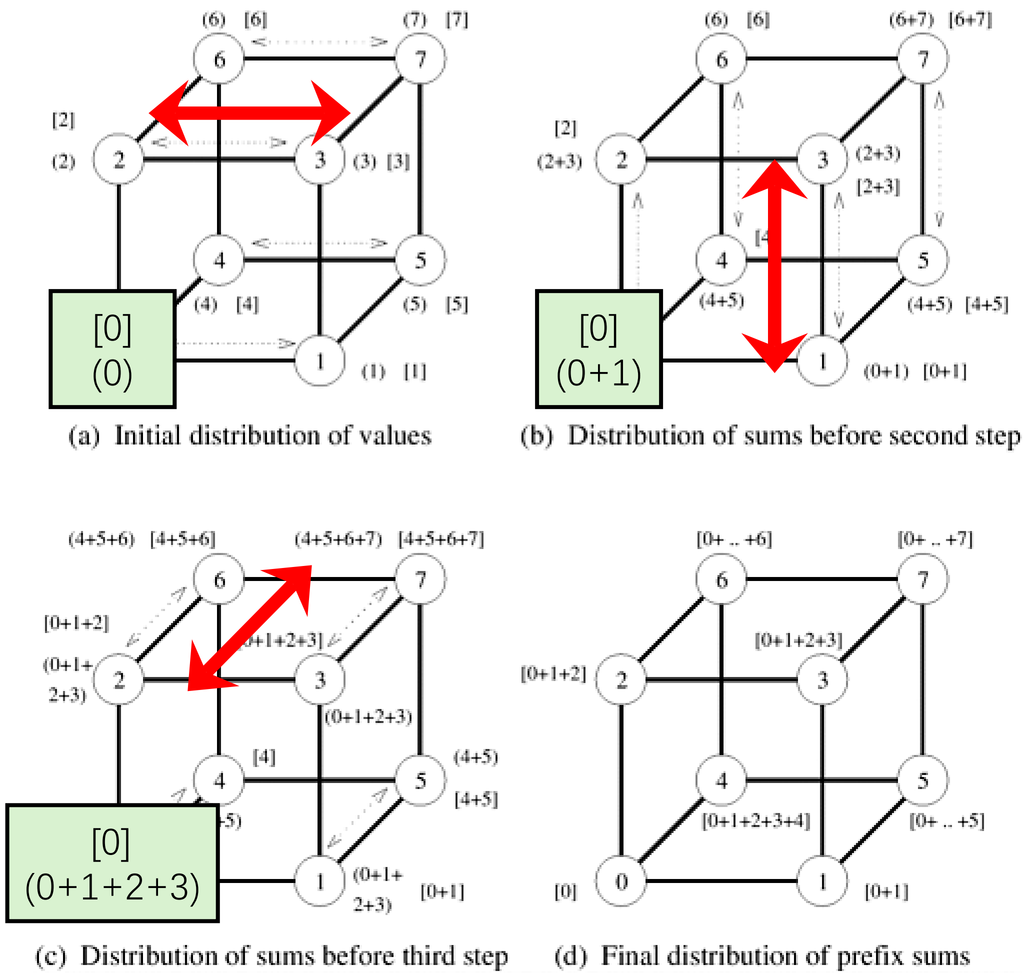
Prefix-Sum - Linear

Finding ***prefix sums*** (also known as the ***scan*** operation) is another important problem that can be solved by using a communication pattern similar to that used in all-to-all broadcast and allreduce operations.



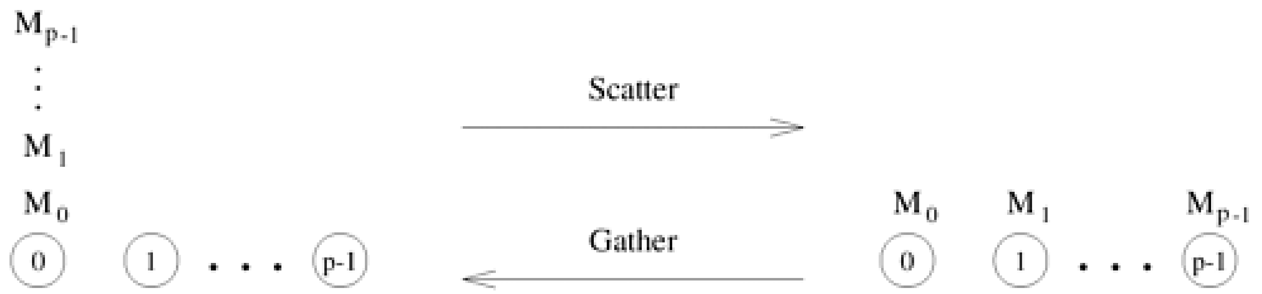
4.3 All-Reduce / Prefix-Sum

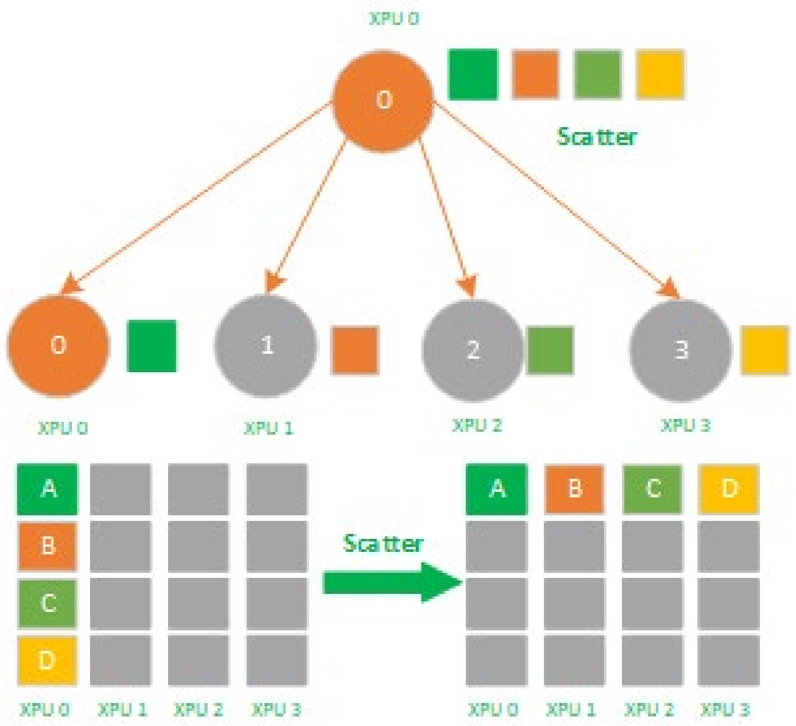
Prefix-Sum - Hypercube

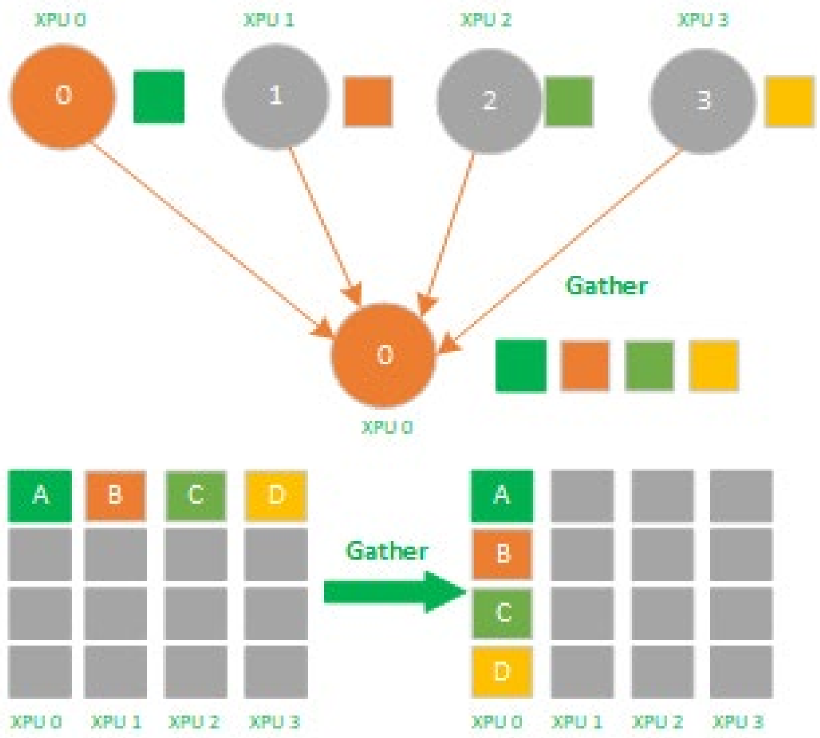


[方括号] 表示结果缓冲区中累积的本地Prefix-Sum
(圆括号) 表示下一步的传出信息缓冲区内容

4.4 Scatter (One-to-All Personalized Communication) / Gather

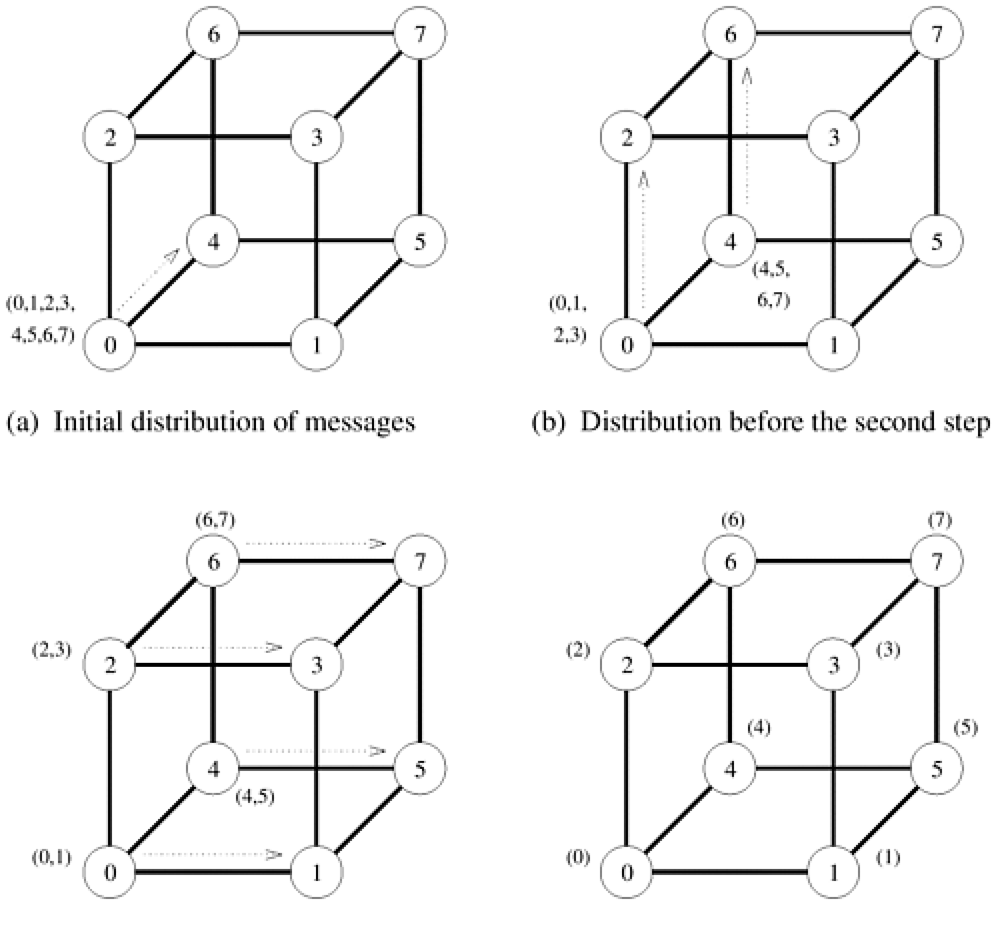






4.4 Scatter (One-to-All Personalized Communication) / Gather

Hypercube



(a) Initial distribution of messages

(b) Distribution before the second step

(c) Distribution before the third step

(d) Final distribution of messages

Step 1

$$T_1 = (t_s + m\frac{p}{2}t_w)$$

Step 2

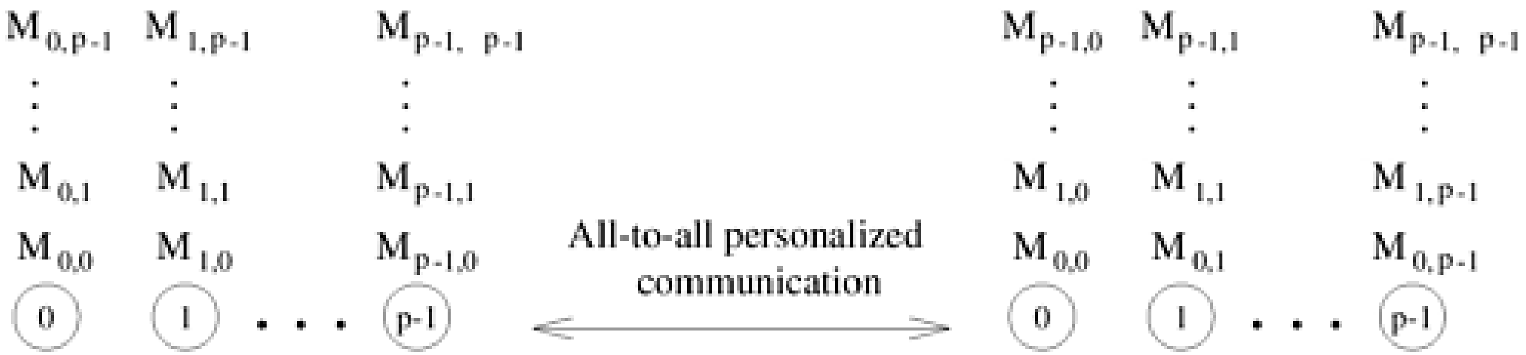
$$T_2 = (t_s + m\frac{p}{2^2}t_w)$$

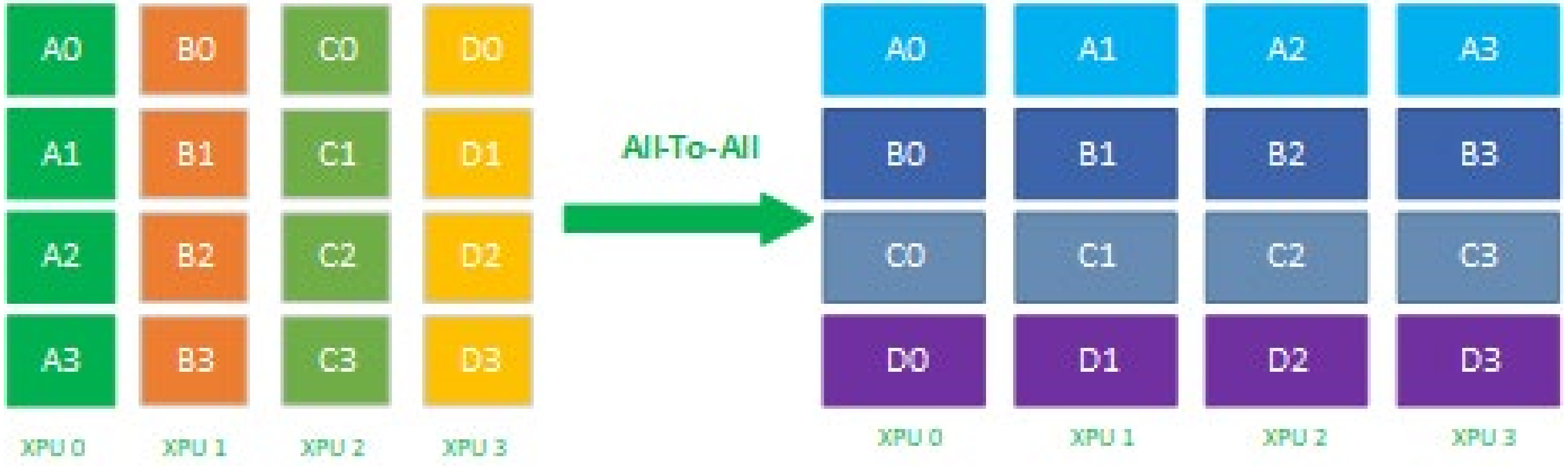
Step 3

$$T_3 = (t_s + m\frac{p}{2^3}t_w)$$

$$T = t_s \log p + m t_w (p-1)$$

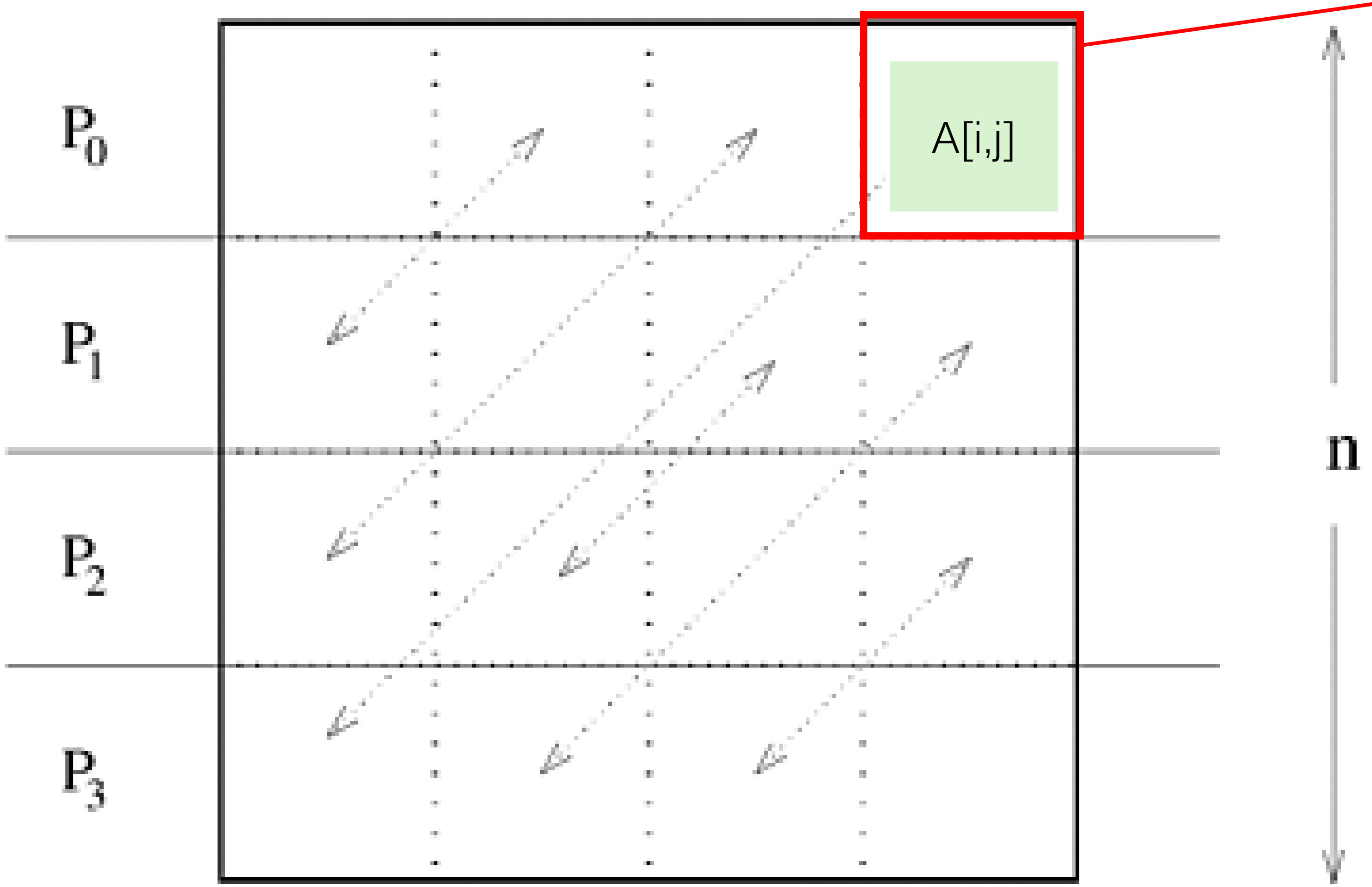
4.5 All-to-All Personalized Communication (Total Exchange)





4.5 All-to-All Personalized Communication (Total Exchange)

Application Case: Matrix Transposition

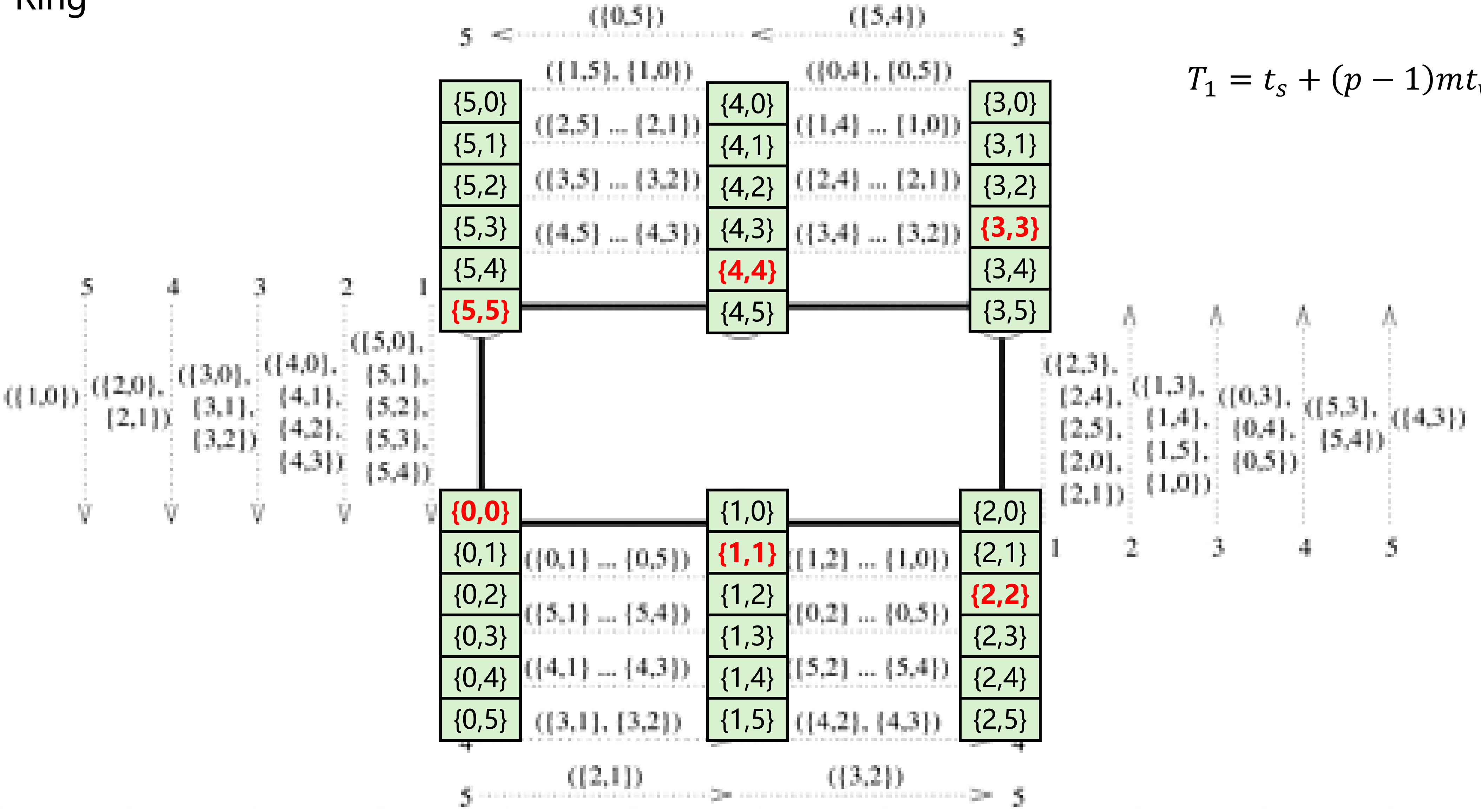


4x4 Matrix

矩阵的转置是针对 $\frac{n}{p} \times \frac{n}{p}$ 大小的数据块进行的

4.5 All-to-All Personalized Communication (Total Exchange)

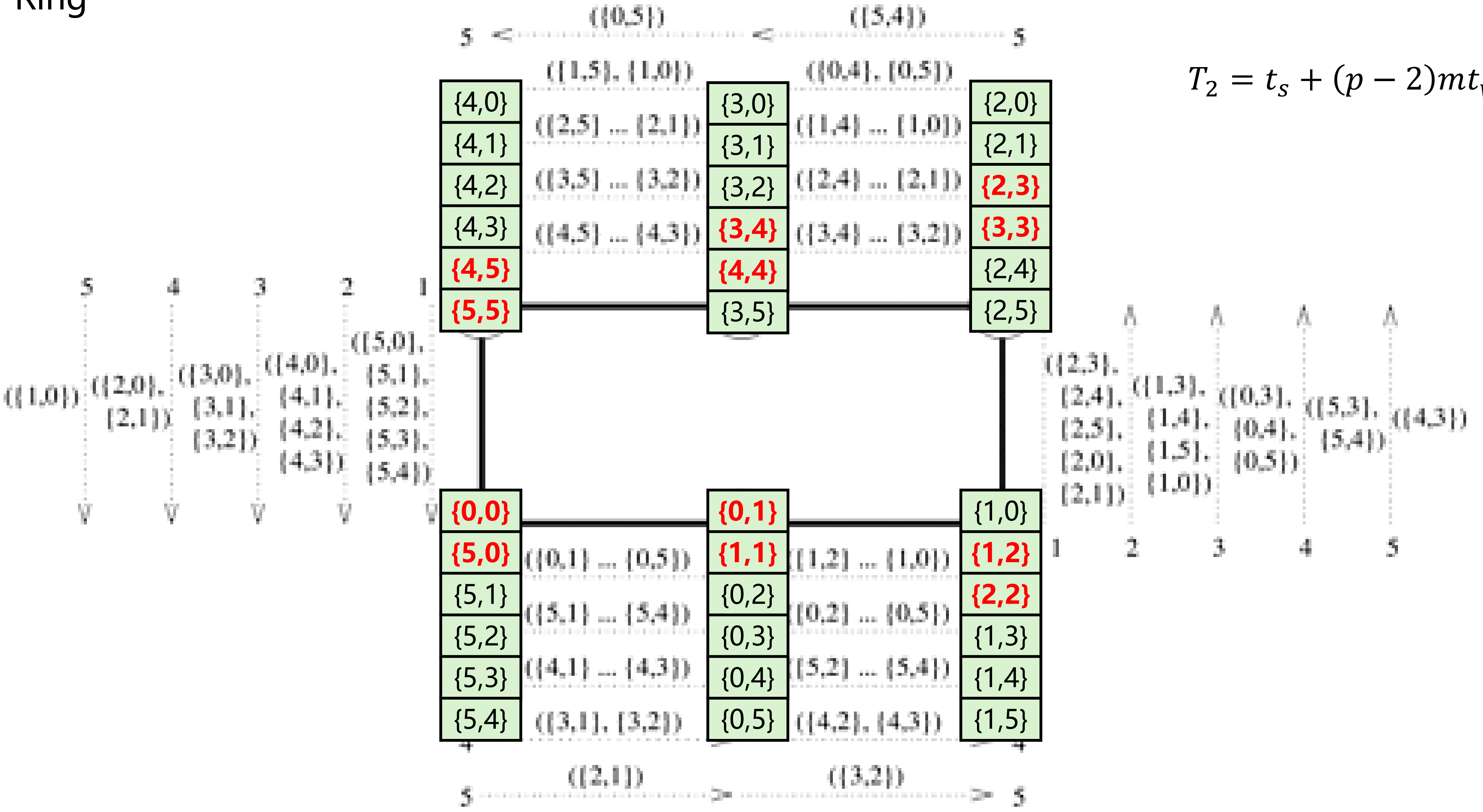
Ring



$$T_1 = t_s + (p-1)mt_w$$

4.5 All-to-All Personalized Communication (Total Exchange)

Ring

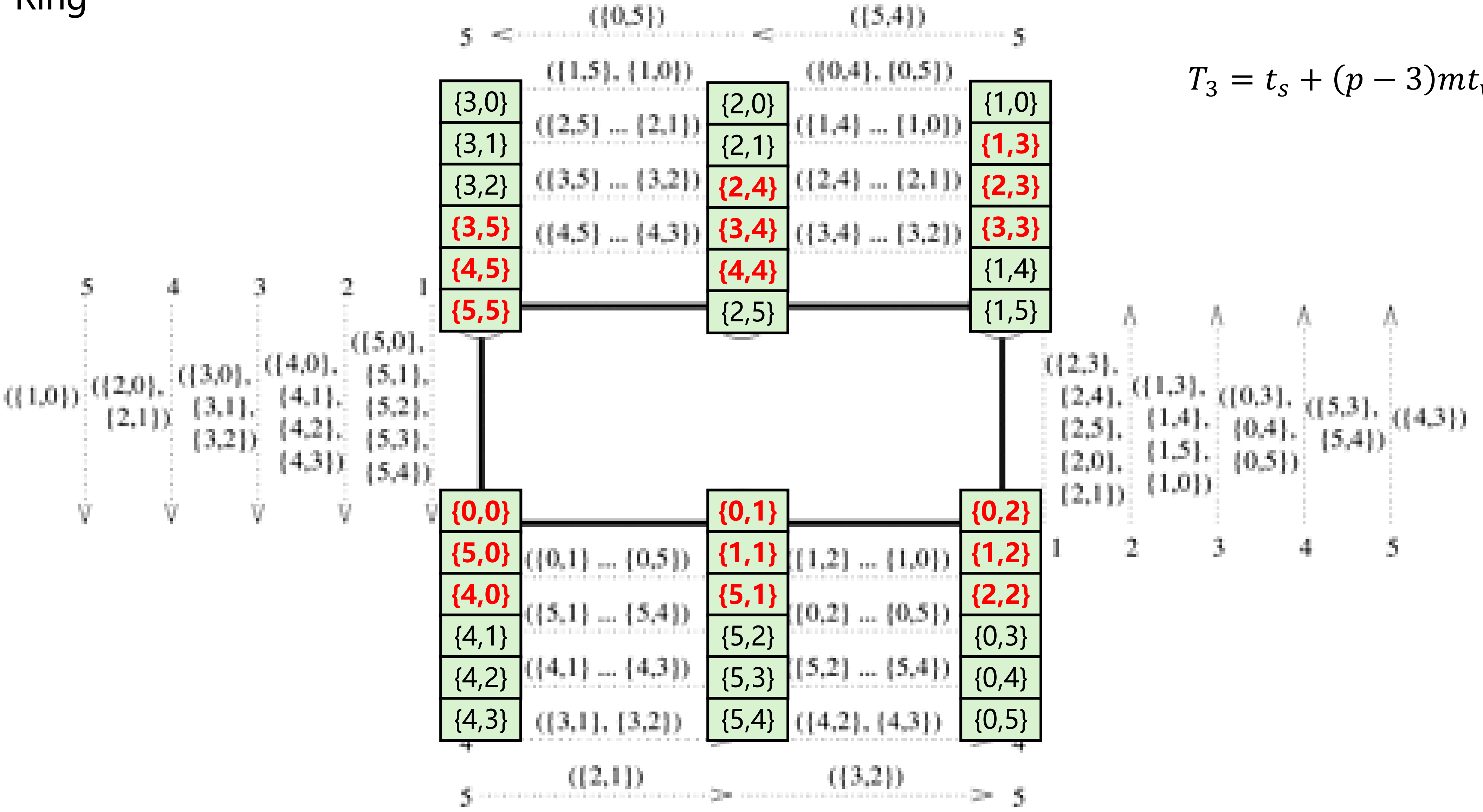


$$T_2 = t_s + (p-2)mt_w$$

4.5 All-to-All Personalized Communication (Total Exchange)

Ring

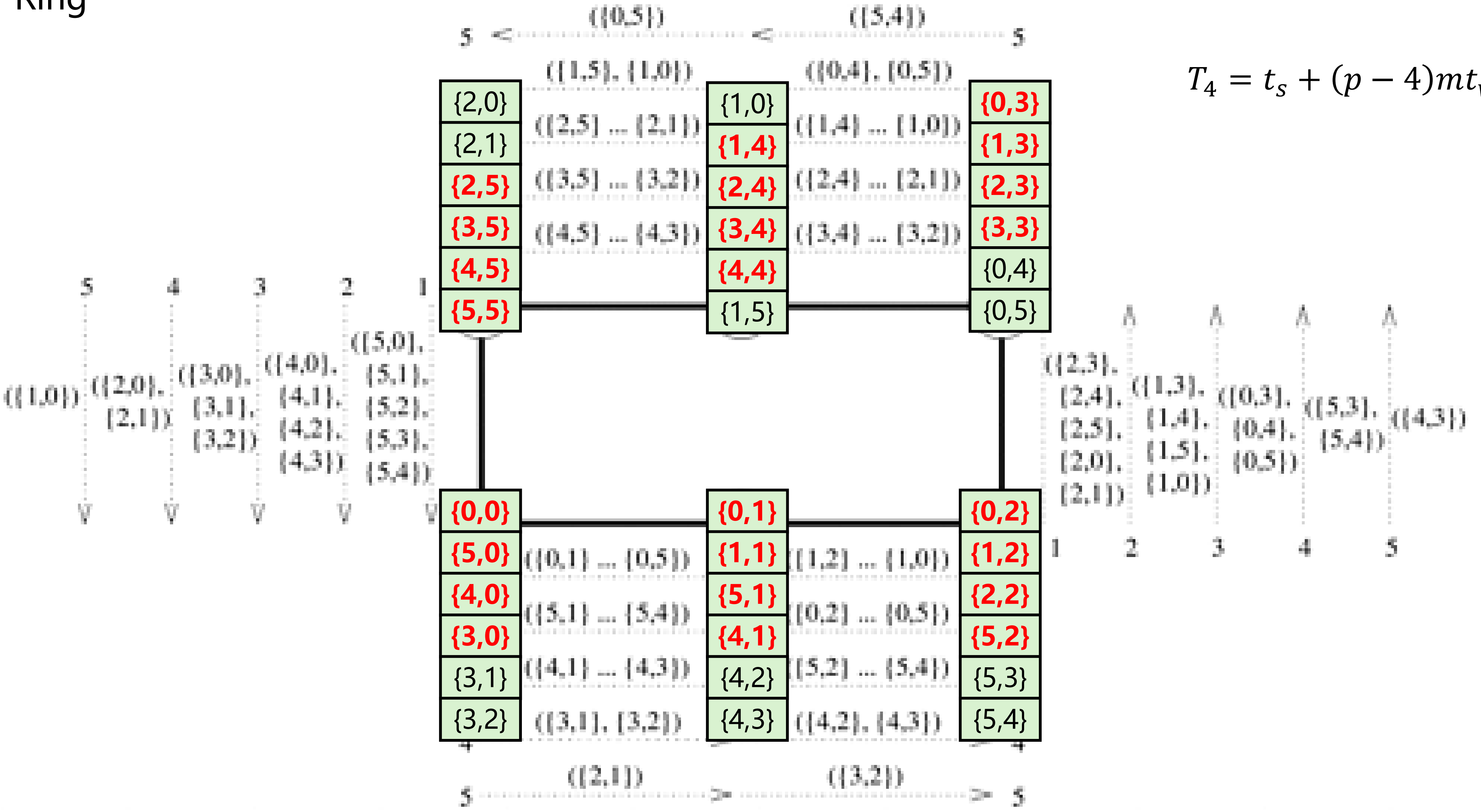
$$T_3 = t_s + (p-3)mt_w$$



4.5 All-to-All Personalized Communication (Total Exchange)

Ring

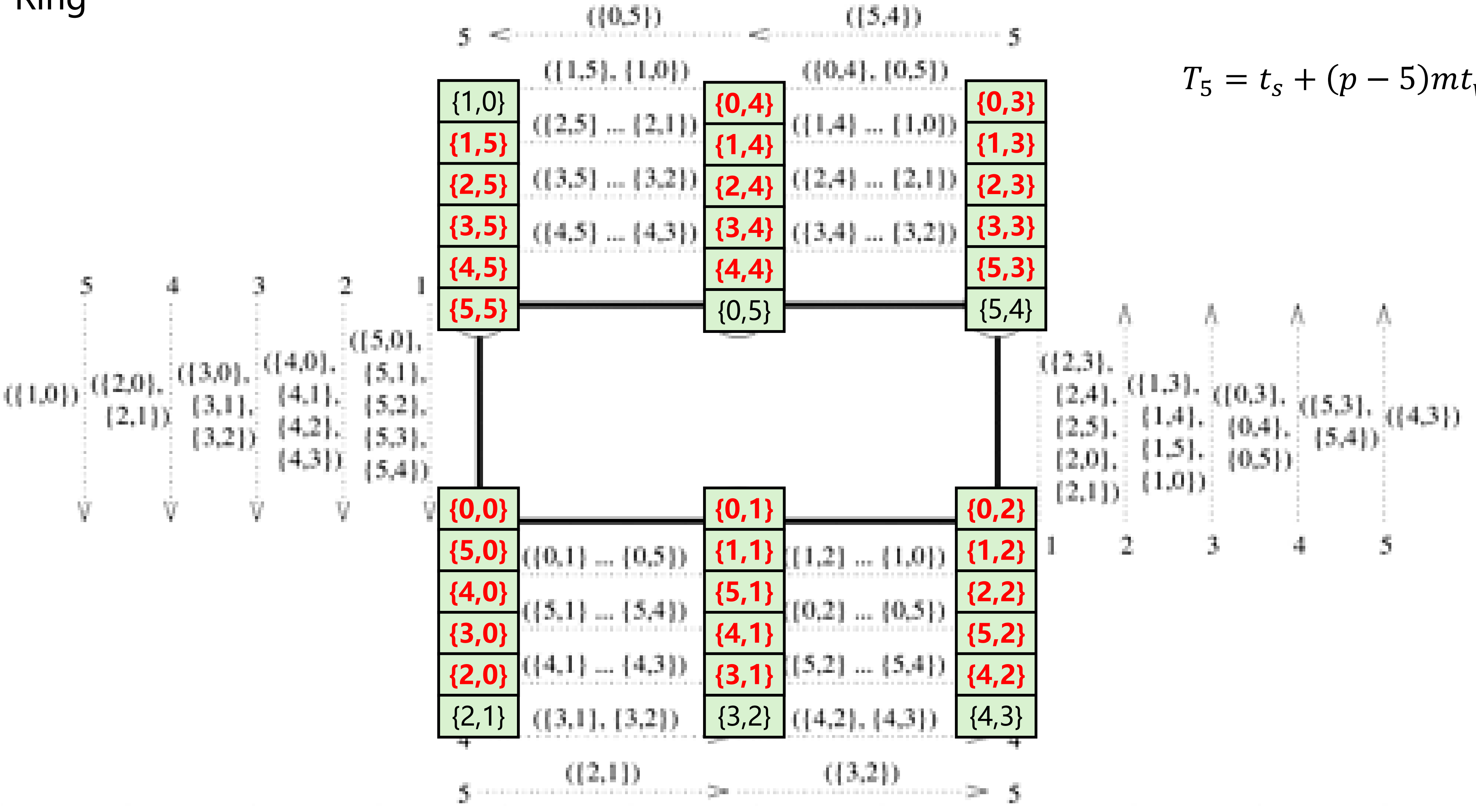
$$T_4 = t_s + (p-4)mt_w$$



4.5 All-to-All Personalized Communication (Total Exchange)

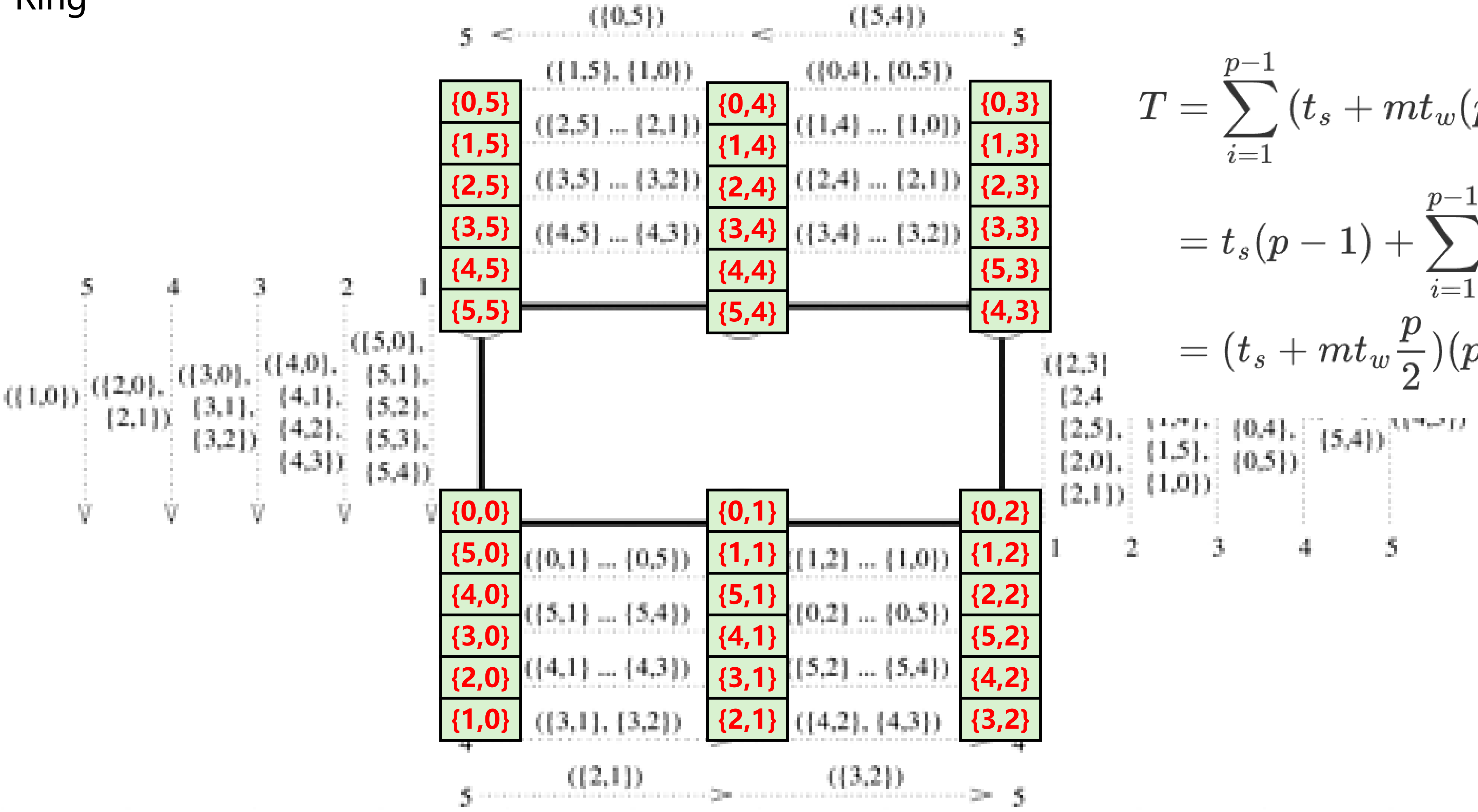
Ring

$$T_5 = t_s + (p-5)mt_w$$



4.5 All-to-All Personalized Communication (Total Exchange)

Ring



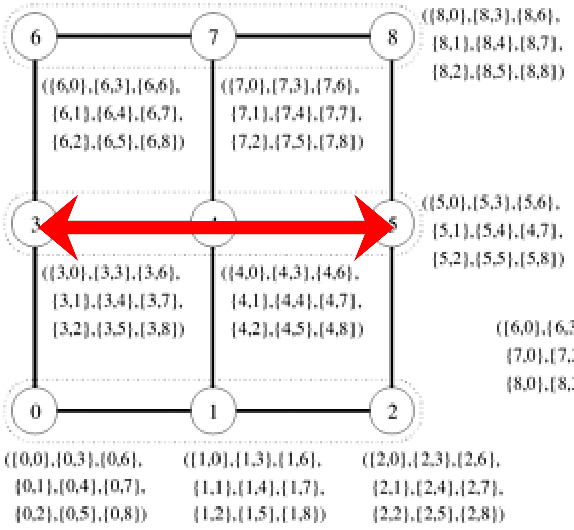
$$T = \sum_{i=1}^{p-1}(t_s + mt_w(p-i))$$

$$= t_s(p-1) + \sum_{i=1}^{p-1} imt_w$$

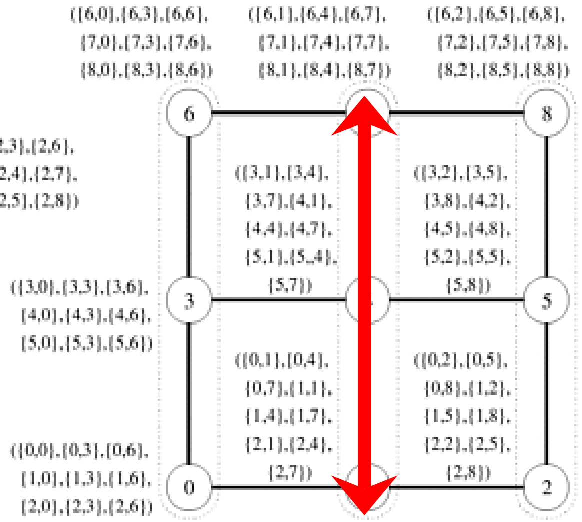
$$= (t_s + mt_w\frac{p}{2})(p-1)$$

4.5 All-to-All Personalized Communication (Total Exchange)

Mesh



(a) Data distribution at the beginning of first phase



(b) Data distribution at the beginning of second phase

Ring $T = \sum_{i=1}^{p-1} (t_s + mt_w(p-i))$

$$= t_s(p-1) + \sum_{i=1}^{p-1} imt_w$$

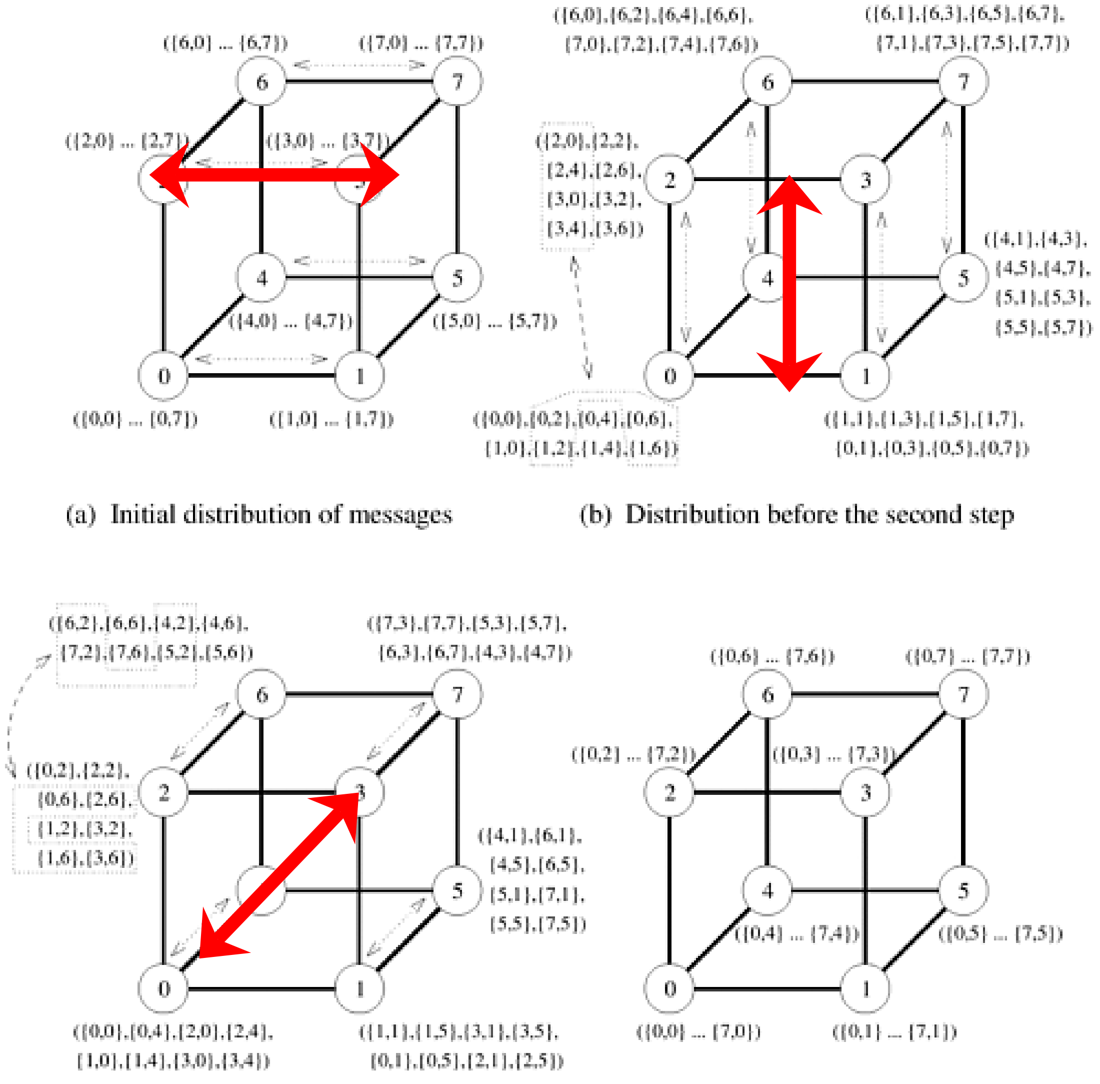
$$= (t_s + mt_w \frac{p}{2})(p-1)$$

$2 \times Ring$ and $p \rightarrow \sqrt{p}$

Mesh $T = (2t_s + mt_w p)(\sqrt{p} - 1)$

4.5 All-to-All Personalized Communication (Total Exchange)

Hypercube



(a) Initial distribution of messages

(b) Distribution before the second step

(c) Distribution before the third step

(d) Final distribution of messages

每步需要传输的数据量为$\frac{mp}{2}$

共计需要传输$\log p$步

总开销：$T = \left(t_s + \frac{mpt_w}{2}\right)\log p$

未考虑传输数据后的排序和索引的时间

问题

- 以上All-to-All算法在Hypercube是最优的吗?
  - 一共进行log p步
  - 每步传输数据大小为mp/2

4.5 All-to-All Personalized Communication (Total Exchange)

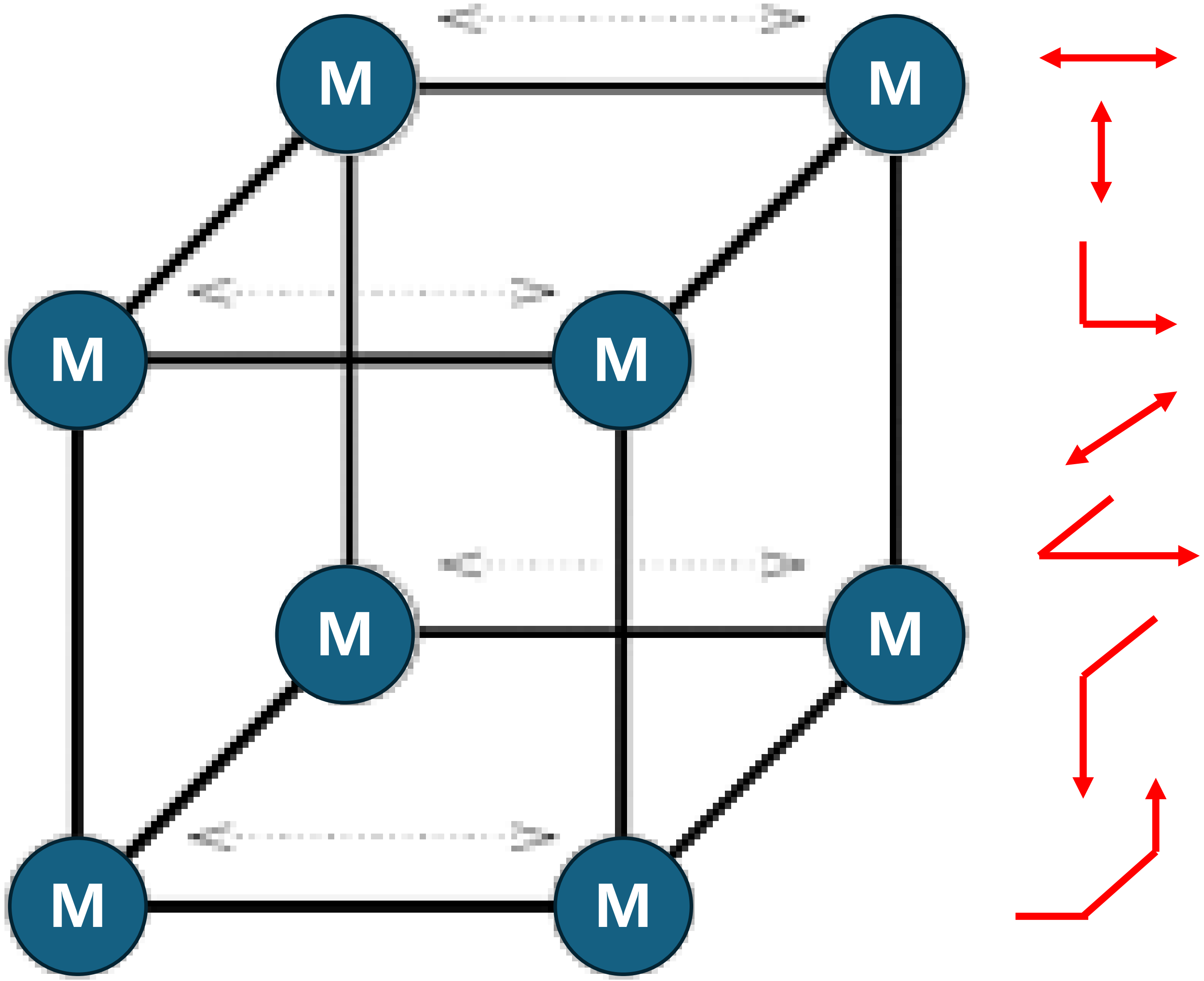
Hypercube - **An Optimal Algorithm**

All-to-All：

1. 每个节点与其他的p-1个节点通信，通信的大小为m个字节
2. 通信过程中没有阻塞

$$
\begin{aligned}
T = &(t_s + mt_w) \\
&+(t_s + mt_w) +(t_s + mt_w) \\
&+(t_s + mt_w) +(t_s + mt_w) \\
&+(t_s + mt_w) +(t_s + mt_w)
\end{aligned}
$$

$$T = (t_s + t_w m)(p-1)$$

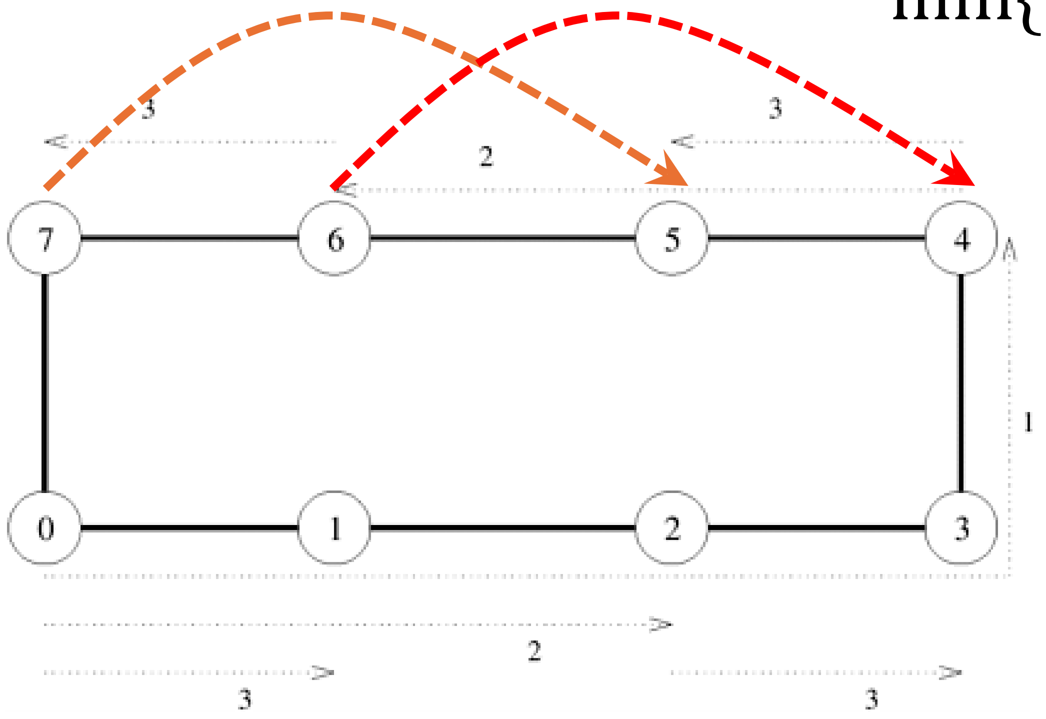


4.6 Circular Shift

We define a ***circular** q**-shift*** as the operation in which node ***i*** sends a data packet to node **(i + q) mod p** in a p-node ensemble

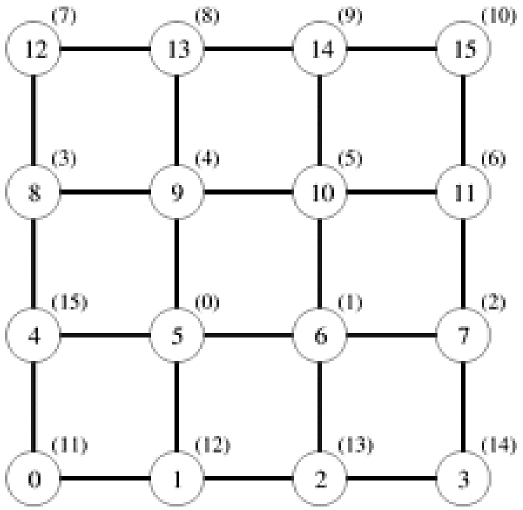
Ring

$$\min\{q, p-q\}$$



4.6 Circular Shift

Mesh - 4x4 Mesh 5-shift



循环移位的结果：

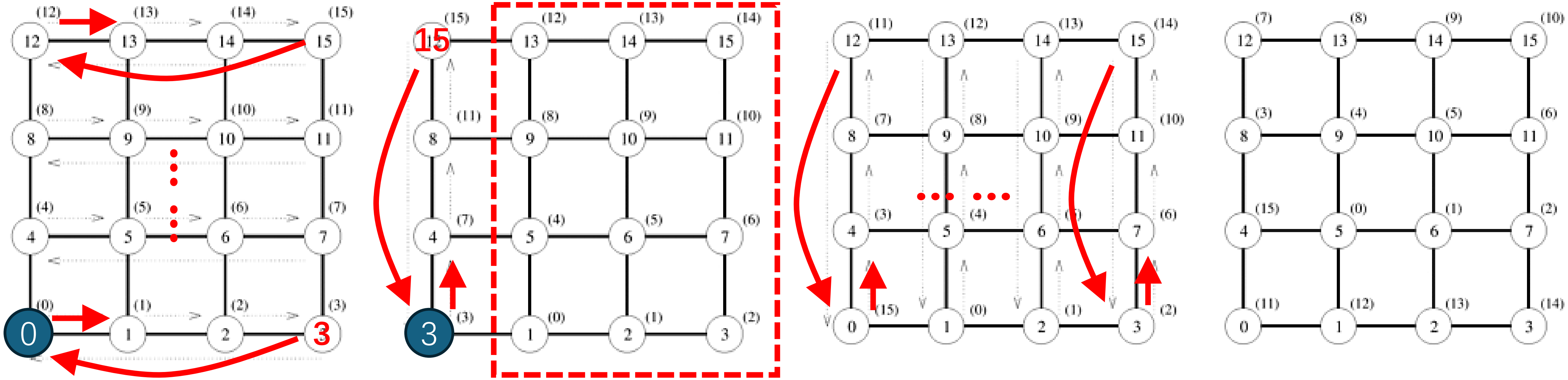
初始状态 <0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15>

5-Shift <11, 12, 13, 14, 15, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10>

在Mesh上如何实现？

4.6 Circular Shift

Mesh - 4x4 Mesh 5-shift



(a) Initial data distribution and the first communication step

(b) Step to compensate for backward row shifts

(c) Column shifts in the third communication step

(d) Final distribution of the data

沿着**行**方向移动$q \ mod \ \sqrt{p}$步

进位补偿

沿着列方向移动$\lfloor \frac{q}{\sqrt{p}} \rfloor$步

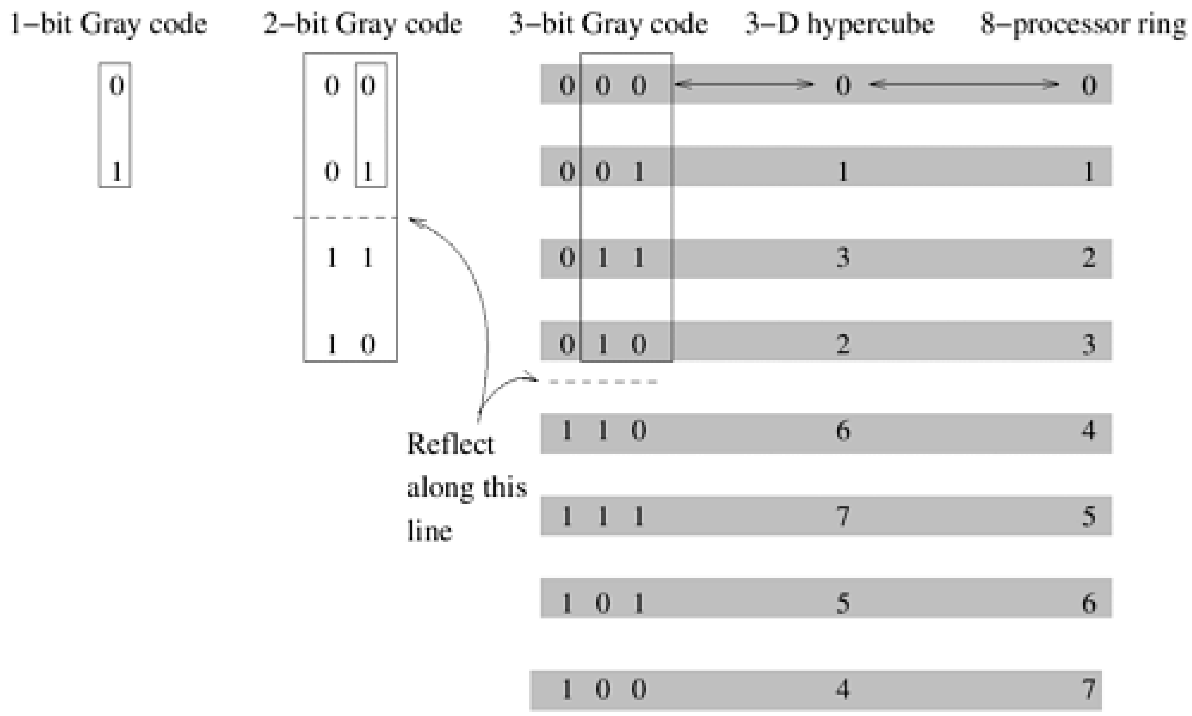
During the circular row shifts, some of the data traverse the wraparound connection from the highest to the lowest labeled nodes of the rows.

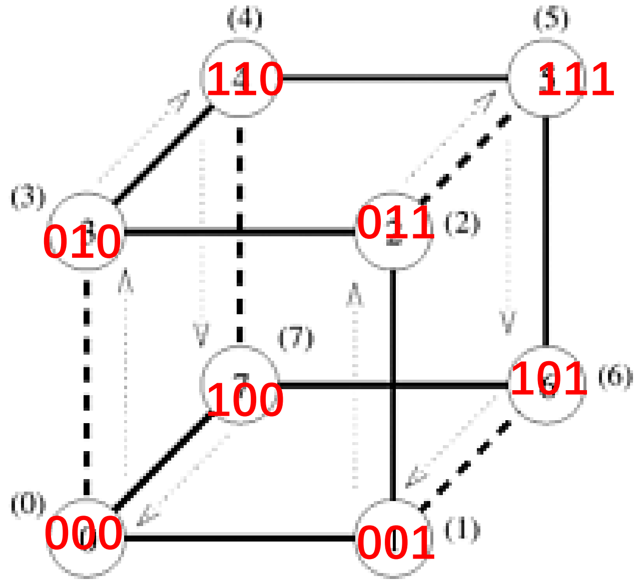
4.6 Circular Shift

Hypercube - 2^3Hypercube 5-Shift

初始状态<0，1，2，3，4，5，6，7>

5-Shift <3，4，5，6，7，0，1，2>

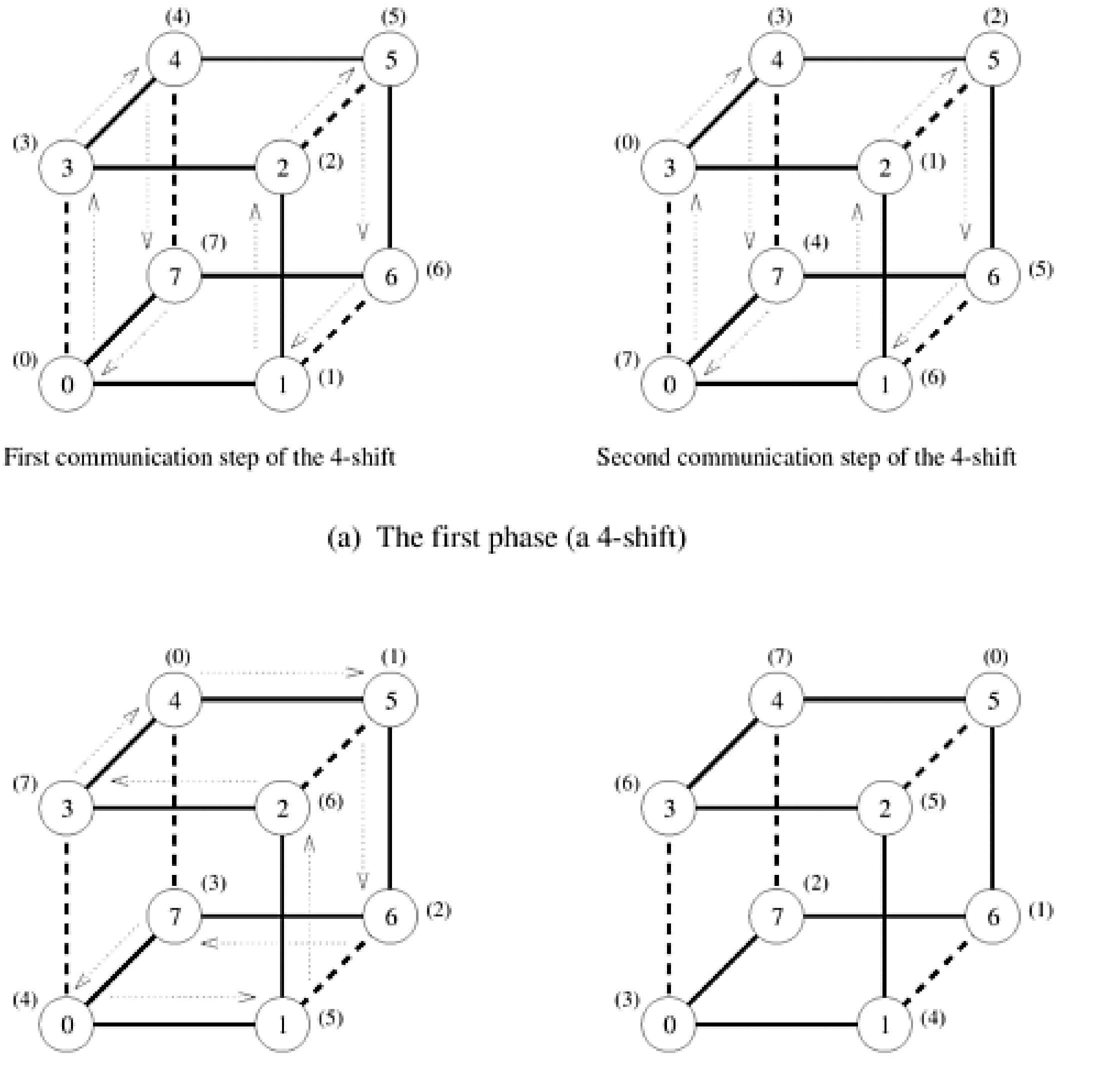




将Linear的序号使用**二进制反射格雷码**映射到Hypercube中

4.6 Circular Shift

Hypercube - 2^3Hypercube 5-Shift



First communication step of the 4-shift

Second communication step of the 4-shift

(a) The first phase (a 4-shift)

(b) The second phase (a 1-shift)

(c) Final data distribution after the 5-shift

2-Shift → 0b010 → 分为1个阶段进行

5-Shift → 0b101 → 分为2个阶段进行 ←

7-Shift → 0b111 → 分为3个阶段进行

通信过程

= 2^2移位 + 2^0移位

= 4移位 + 1移位

= (2次通信) + 1次通信 ← 除了1移位都是两次通信

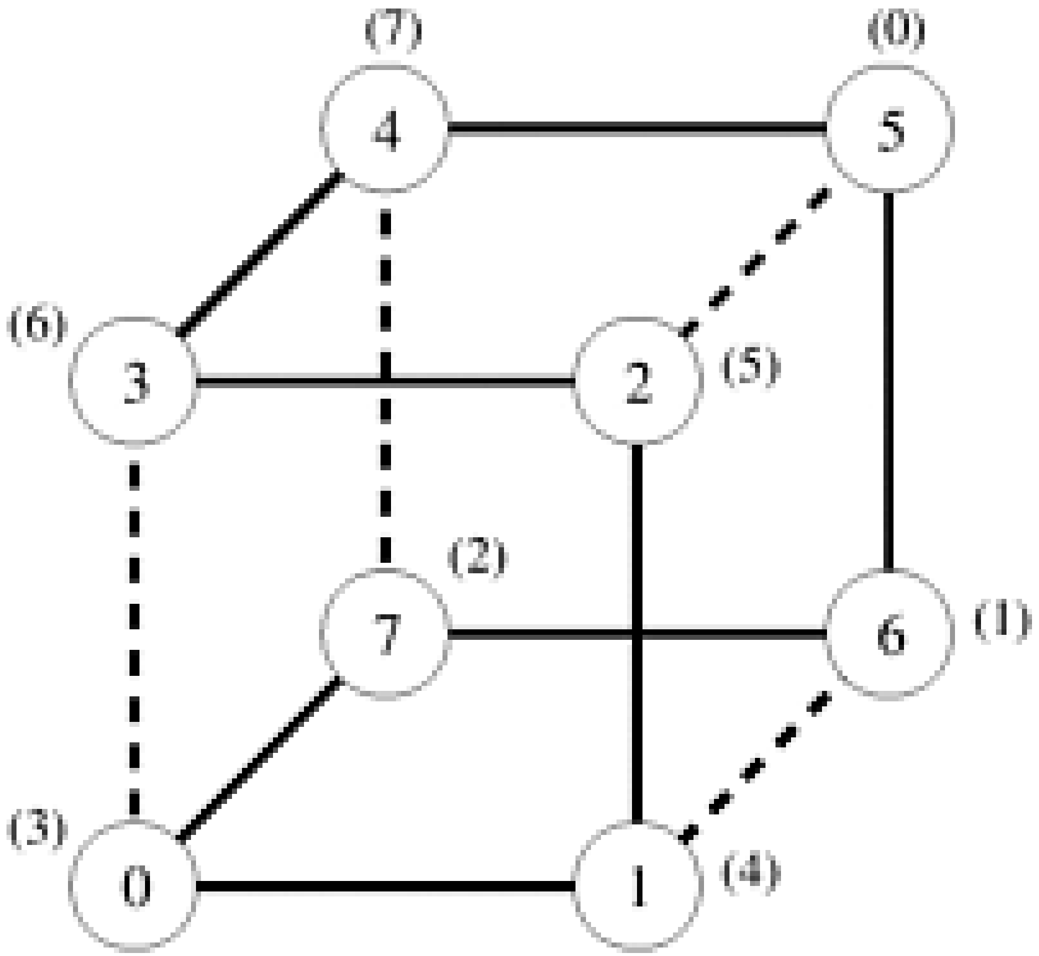
所有间隔为2^0的节点构成一个子矩阵循环一次

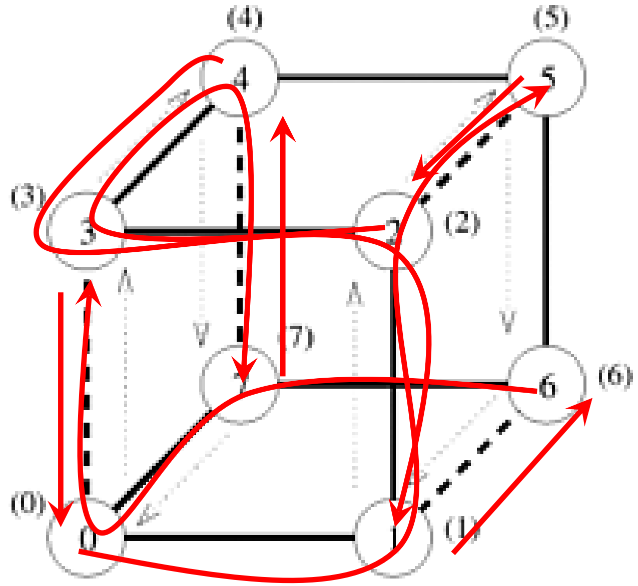
所有间隔为2^2的节点构成一个子矩阵循环两次

Ring

4.6 Circular Shift

Hypercube - 2^3Hypercube 5-Shift





4.7 Improving the Speed of Some Communication Operations

1. Splitting and Routing Messages in Parts

- Note that the algorithms of this section rely on ***m* being large enough** to be split into *p* roughly equal parts.

- there is **a cut-off value for the message size *m*** and only the messages longer than the cut-off would benefit from the algorithms in this section.

2. All-Port Communication

- an ***all-port communication*** model permits simultaneous communication on all the channels connected to a node.

4.7 Improving the Speed of Some Communication Operations

Splitting and Routing Messages in Parts

Hpyercube

One-to-All Broadcast and All-to-One Reduction

One-to-All Broadcast (m) = Scatter (m/p) + All-to-All Broadcast (m/p)

$$Hypercube\ \{T\} = t_s \log p + \frac{m}{p} t_w (p-1)$$ *Slide Page 14*

$$Hpyercube\ \{T\} = t_s \log p + \frac{m}{p} t_w (p-1)$$ *Slide Page 20*

$$T = 2 \times \left(t_s \log p + t_w (p-1) \frac{m}{p}\right)$$
$$\approx 2 \times (t_s \log p + \underline{m t_w})$$

$$\mathbf{2} m t_w$$

vs.

$$T = (t_s + \underline{m t_w) \log(p)}$$ *Slide Page 6*

$$\mathbf{\log p}\, m t_w$$

4.7 Improving the Speed of Some Communication Operations

Splitting and Routing Messages in Parts

Hpyercube

All Reduce

All Reduce (m) = All-to-All Reduction (m/p) + All-to-All Broadcast (m/p)

$Hypercube\ \{T\} = t_s \log p + \frac{m}{p} t_w (p-1)$ *Slide Page 14*

$Hypercube\ \{T\} = t_s \log p + \frac{m}{p} t_w (p-1)$ *Slide Page 14*

$$T = 2 \times (t_s \log p + t_w (p-1) \frac{m}{p})$$
$$\approx 2 \times (t_s \log p + \underline{m t_w})$$

vs.

$$T = (t_s + \underline{m t_w}) \log p$$ *Slide Page 16*

$2 m t_w$

$\log p \, m t_w$

4.8 Summary

各种操作在超立方互连网络上的通信时间汇总

Operation	**Hypercube Time**	**B/W Requirement**
One-to-All Broadcast / All-to-One Reduction	$\min((t_s + mt_w)\log p, 2(t_s \log p + mt_w))$	$\Theta(1)$
All-to-All Broadcast / All-to-All Reduction	$t_s \log p + t_w m(p-1)$	$\Theta(1)$
All Reduce	$\min((t_s + mt_w)\log p, 2(t_s \log p + t_w m))$	$\Theta(1)$
Scatter / Gather	$t_s \log p + t_w m(p-1)$	$\Theta(1)$
All-to-All Personalized	$(t_s + t_w m)(p-1)$	$\Theta(p)$
Circular Shift	$t_s + t_w m$	$\Theta(p)$

4.1 One-to-All Broadcast and All-to-One Reduction

3.1.2 任务交互 Task-Interaction

总结

问题

提示