EE/CSCI 451: Parallel and Distributed Computation

Lecture #7

9/8/2020

Viktor Prasanna

prasanna@usc.edu

ceng.usc.edu/~prasanna

University of Southern California

Announcement

- Midterm 1 date: 9/25
  - Discussion session - 2hours

- HW2 due 9/10 (upcoming Thu.)
- PHW2 due 9/14 (+3 free late-days)
- HW1 and PHW1 grades are out

PHW1 Statistics	
Average	93.51
Median	98.00
Standard Deviation	18.86

HW1 Statistics	
Average	91.10
Median	96.00
Standard Deviation	15.39

Announcement: Midterm1 Logistics

- **Online Proctored** Exam:
  - Time: Week 6 discussion session – 2 hours: 3:30-5:30PM (Los Angeles time)
  - Format: Open-book, open-notes **[Attendence is required, no make-up given]**
  - Proctoring: 2 proctors watching different subgroups of students in separate Zoom meetings, links will be sent to students in advance
    - Require **camera-enabled** device
- Receiving and returning your exam:
  - Exam will be released on Piazza under resource page at/around 3:26 PM
  - You will submit the completed exam on Blackboard (a submission portal will be created in advance)
- Completing your exam:
  - Download the assignment pages (exam pages) as pdf files on to your tablet and annotate it with your answers. Only hand-annotated pdf files are acceptable.
    - Require a **writable tablet** device
- Coverage: Week 1-Week 5 contents (Week 6 contents - analytical modeling & communication primitives - not covered)
- Special note 1: Important - discussion **attendance is required** on Week 5 (Sept 18)!
  - 10-min midterm trial run to make sure all students are prepared for and comfortable with the exam process
- Special note 2:
  - We have created a Piazza poll [link] to collect info regarding your available resources/capabilities to complete the exam with writable tablet. Everyone is **required to participate in the poll**

Announcement

- Lecture slides will be released **before** each class for your reference

- Occasionally, lecture slides will be **updated** after the lecture
  - The updated slides will over raid the slides released before the class

- Lecture slides are available at https://piazza.com/usc/fall2020/eecsci451/resources

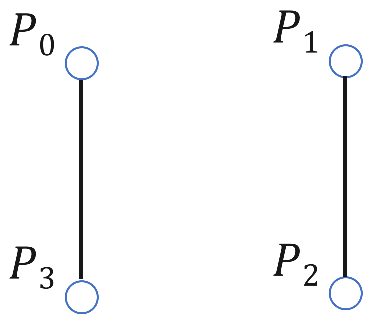
Outline

- From last class
  - Message Passing Programming Model
    - Asynchronous/Loosely Synchronous/SPMD
    - Send/Receive
    - Blocking/Non-blocking
    - Cannon's Algorithm for Matrix Multiplication
- Today
  - Interconnection networks (Chap. 2.4.3-2.4.5)
    - Crossbar network
    - Shuffle exchange network
    - Multistage network

Interconnection Networks

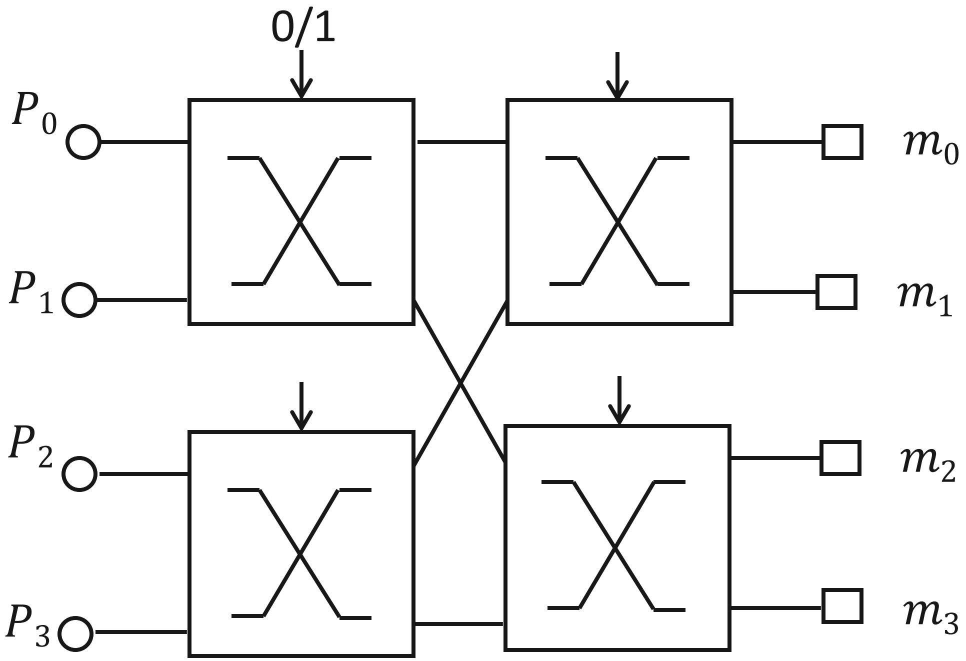
Data communication among processors and memory

Static (direct) networks



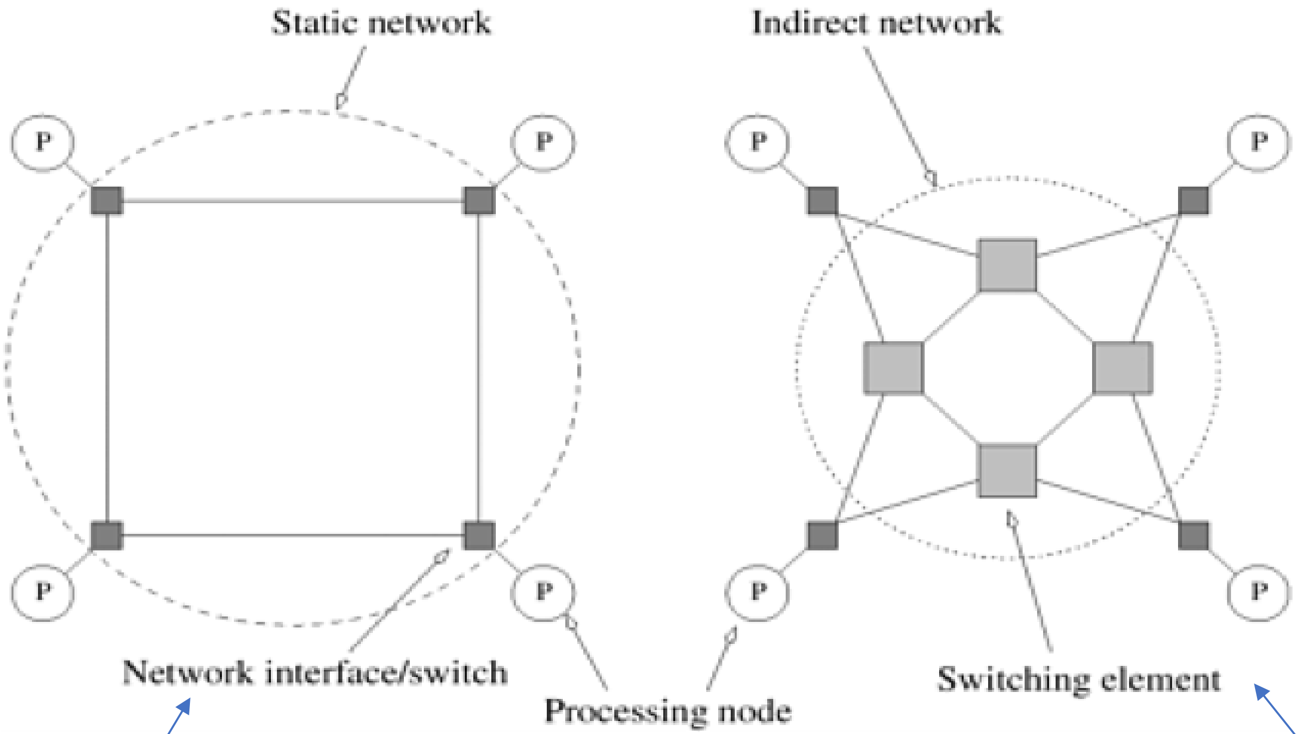
Direct connections (fixed)

Dynamic (indirect) networks
(Multistage network)



Network Topologies

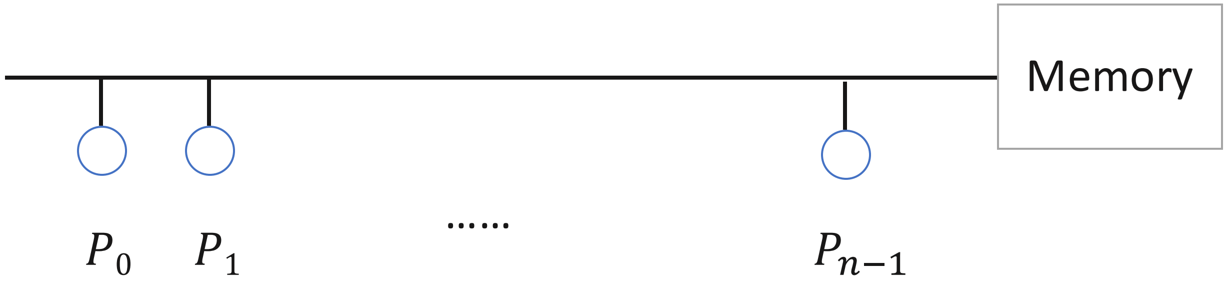
Examples:



A static network of four processing elements or nodes

A dynamic network of four nodes connected via a network of switches to other nodes

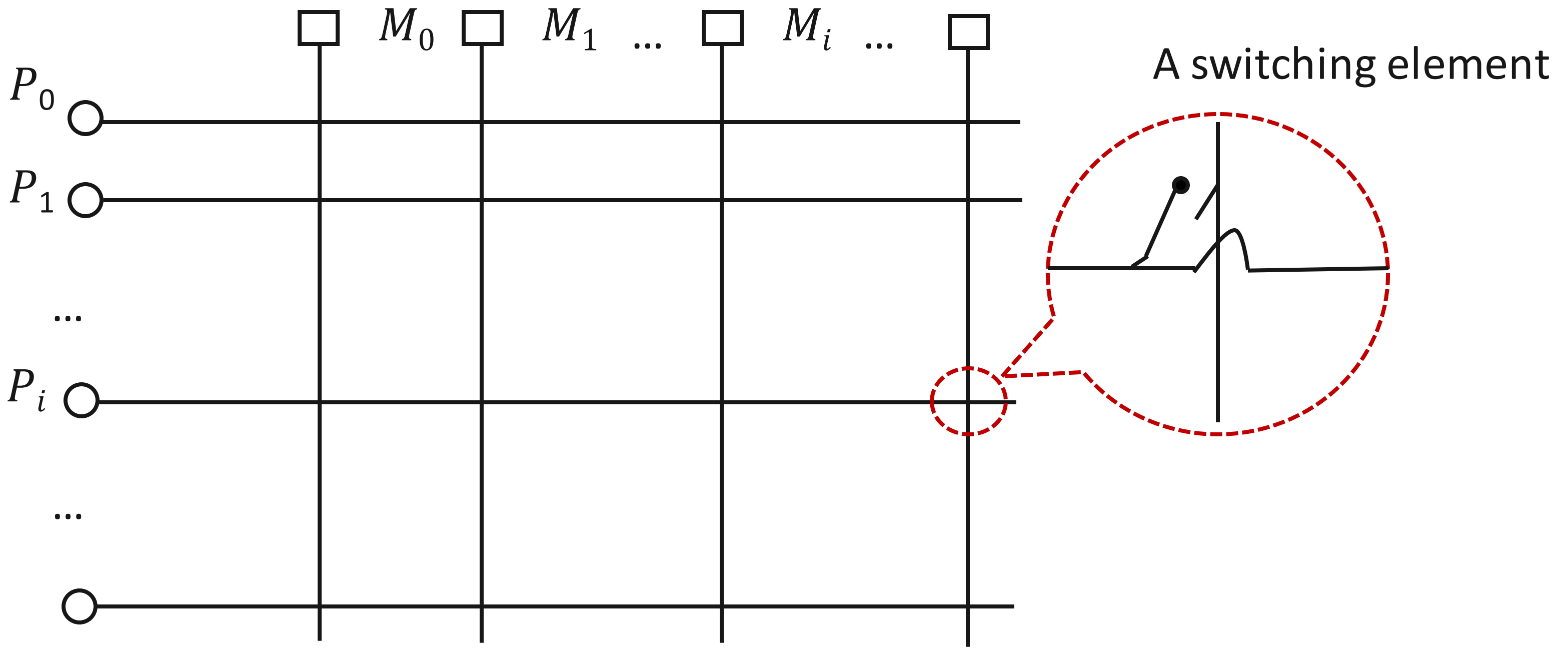
Bus-based Network



- Distance: $O(1)$
- Low cost
- Scalable?
- Bandwidth = bus width (bits) $\times$ clock rate (independent of n)
- Traffic on the bus can be reduced by using a cache in each node

One transaction at any time

Crossbar Network



- $p \times p$ crossbar: cost $\sim O(p^2)$ Number of switches
- To make a connection from P_i to M_j (permutation on p items):

  P_i broadcast $j \rightarrow$ switch $(i, j) \rightarrow$ close connection

Shuffle Network

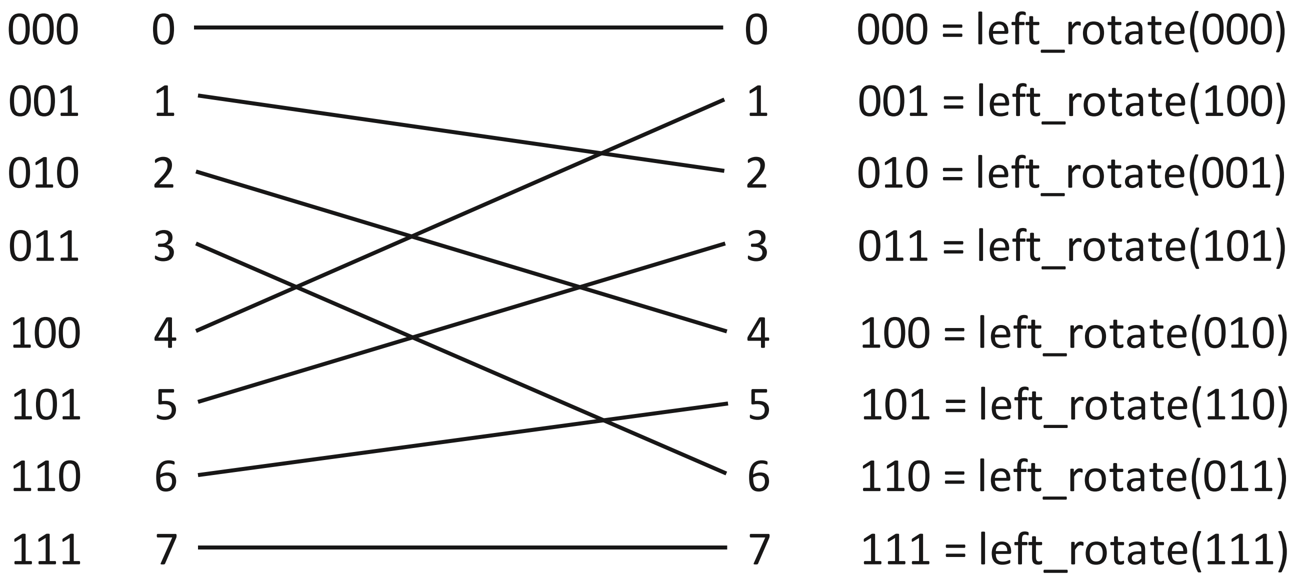
Perfect Shuffle (PS) connection

- A link exists between input i and output j if:

$$j = \begin{cases} 2i, & 0 \le i < \frac{p}{2} \\ 2i + 1 - p, & \frac{p}{2} \le i < p \end{cases}$$

Left rotation (circular left shift) of binary representation of i

p = power of 2



Shuffle Exchange Network

Perfect Shuffle (Shuffle) for $n = 8$

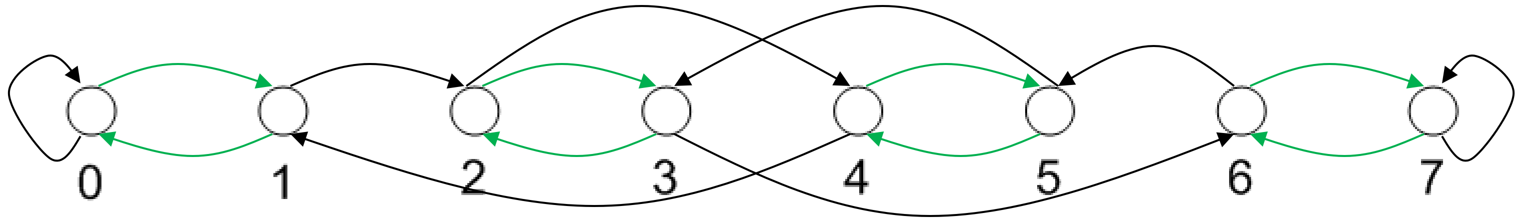
0	$\rightarrow$	0
1	$\rightarrow$	2
2	$\rightarrow$	4
3	$\rightarrow$	6
4	$\rightarrow$	1
5	$\rightarrow$	3
6	$\rightarrow$	5
7	$\rightarrow$	7

$i \rightarrow 2i \ \ mod \ \ n \qquad 0 \le i < \frac{n}{2}$

$i \rightarrow (2i+1) \ \ mod \ \ n \qquad \frac{n}{2} \le i < n$

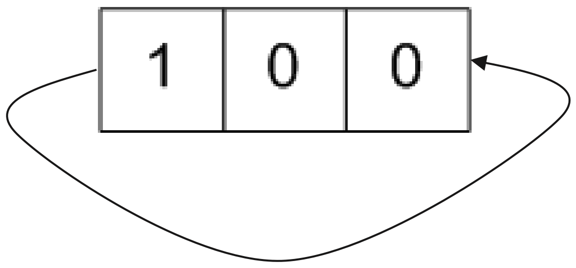
$2i \leftrightarrow 2i+1 \qquad 0 \le i < \frac{n}{2}$

Exchange



Shuffle Connection

Example: n=8 3 bit index

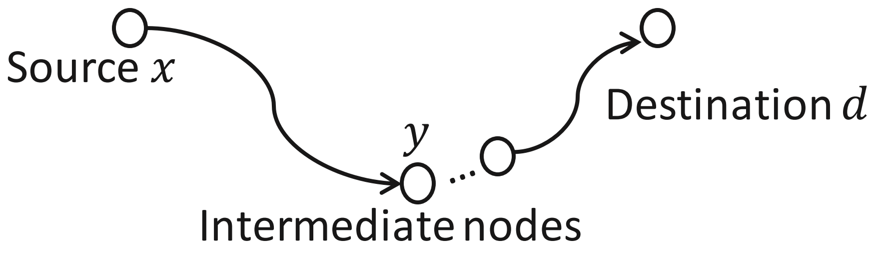


Circular left shift

- 1 0 0 $\rightarrow$ 0 0 1 (4 $\rightarrow$ 1)
- Exchange connection $2i \leftrightarrow 2i+1$
  Complement lsb
- Diameter: (discussed later) $O(\log n)$ $n = 2^k$

Routing in Shuffle Exchange Network (1)

Source $x = x_{k-1} \cdots x_0$ $\longrightarrow$ Destination $d = d_{k-1} \cdots d_0$



$y \leftarrow x$ {current location}

$i \leftarrow 1$

While $i \leq k$

    Shuffle y {Rotate left}

    Compare LSB of y with bit $(k - i)$ of destination (d)

    If bits are the same, then do not Exchange;
    else Exchange {Complement y_0}
    $i \leftarrow i + 1$

End

Total # of hops $\leq 2k\ (2\log_2 n)$

Routing in Shuffle Exchange Network (2)

Source $x_2x_1x_0$ (000) $\rightarrow$ Destination $d_2d_1d_0$ (110) $k = 3$

Example: $i = 1$

- Shuffle

$$000 \leftarrow 000$$

- Compare LSB of y with bit 2 of destination

$$y_0 = d_2?$$

  Same as

$$x_2 = d_2?$$

  Position at the end of first iteration: 001

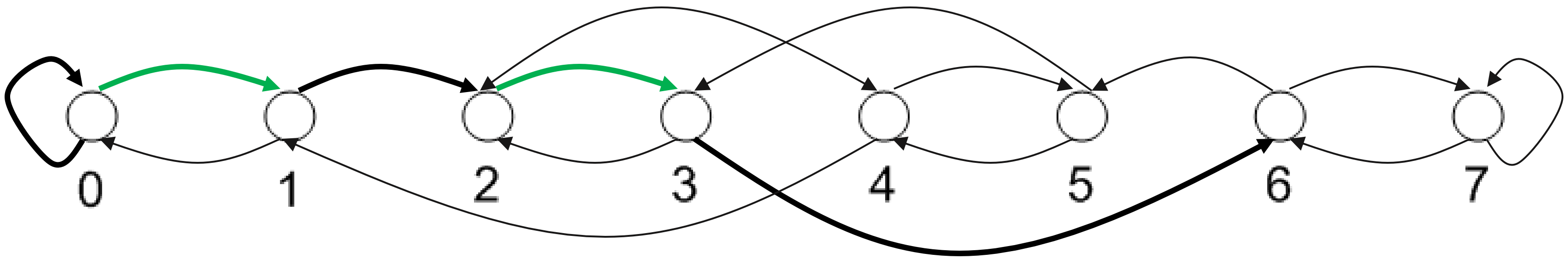
- End of i^{th} iteration: $y = x_{k-1-i} \dots x_0 d_{k-1} \dots d_{k-i}$

$x = 000$ $d = 110$

$i = 1$	000	S
	001	E
$i = 2$	010	S
	011	E
$i = 3$	110	S
	110	No E

Routing in Shuffle Exchange Network (3)

Source $x_2x_1x_0$ (000) $\rightarrow$ Destination $d_2d_1d_0$ (110) $k = 3$

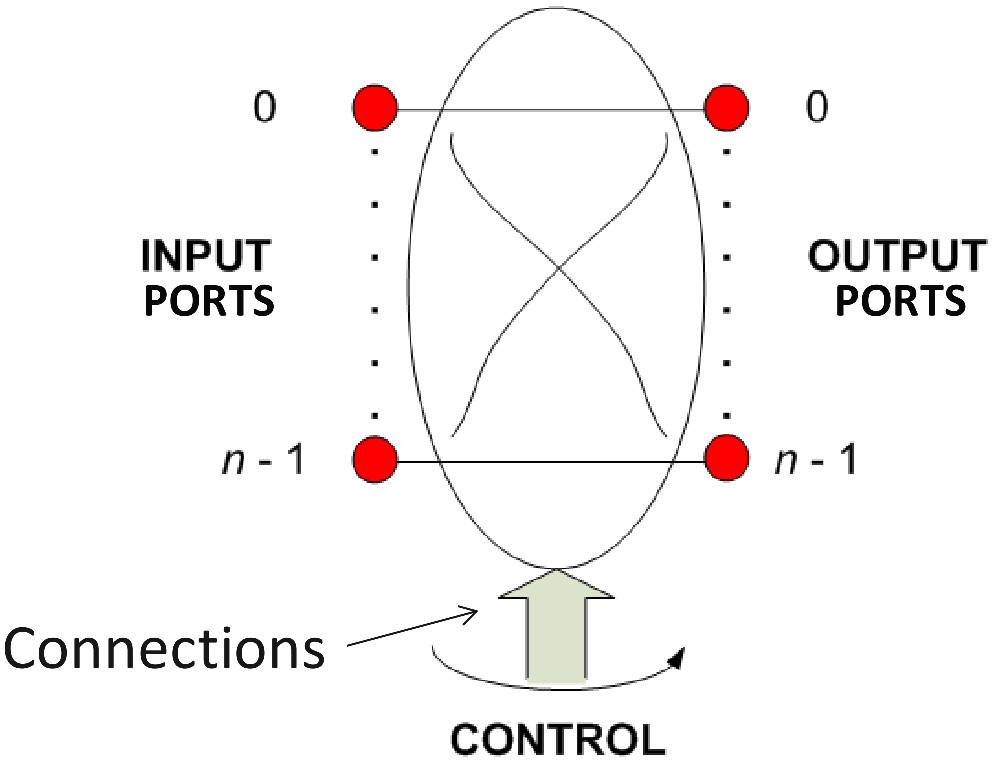


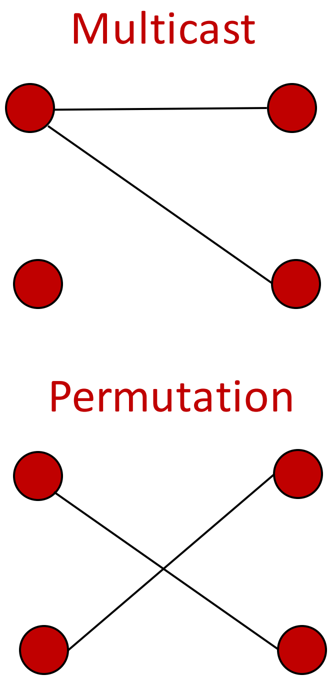
Routing in Shuffle Exchange Network (4)

- Theorem: In a shuffle exchange network with $n = 2^k$ nodes, data from **any** source to **any** destination can be routed in at most $2\log_2 n$ steps.

Multistage Network (1)

- Can realize rich set of connecting patterns from input to output





- Dynamic networks
- Multiple stages of switches and connections

Multistage Network (2)

Connecting Pattern (Data Communication Pattern)

n inputs / n outputs

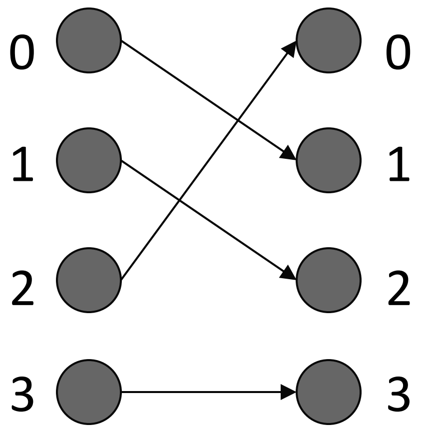
For each i, $0 \leq i < n$, connecting pattern specifies

output(s) j to which data from i is to be routed to

- Example:
    - Connecting Pattern is a permutation
    - Given n inputs and n outputs,

      total # of connecting patterns = $n!$

Example

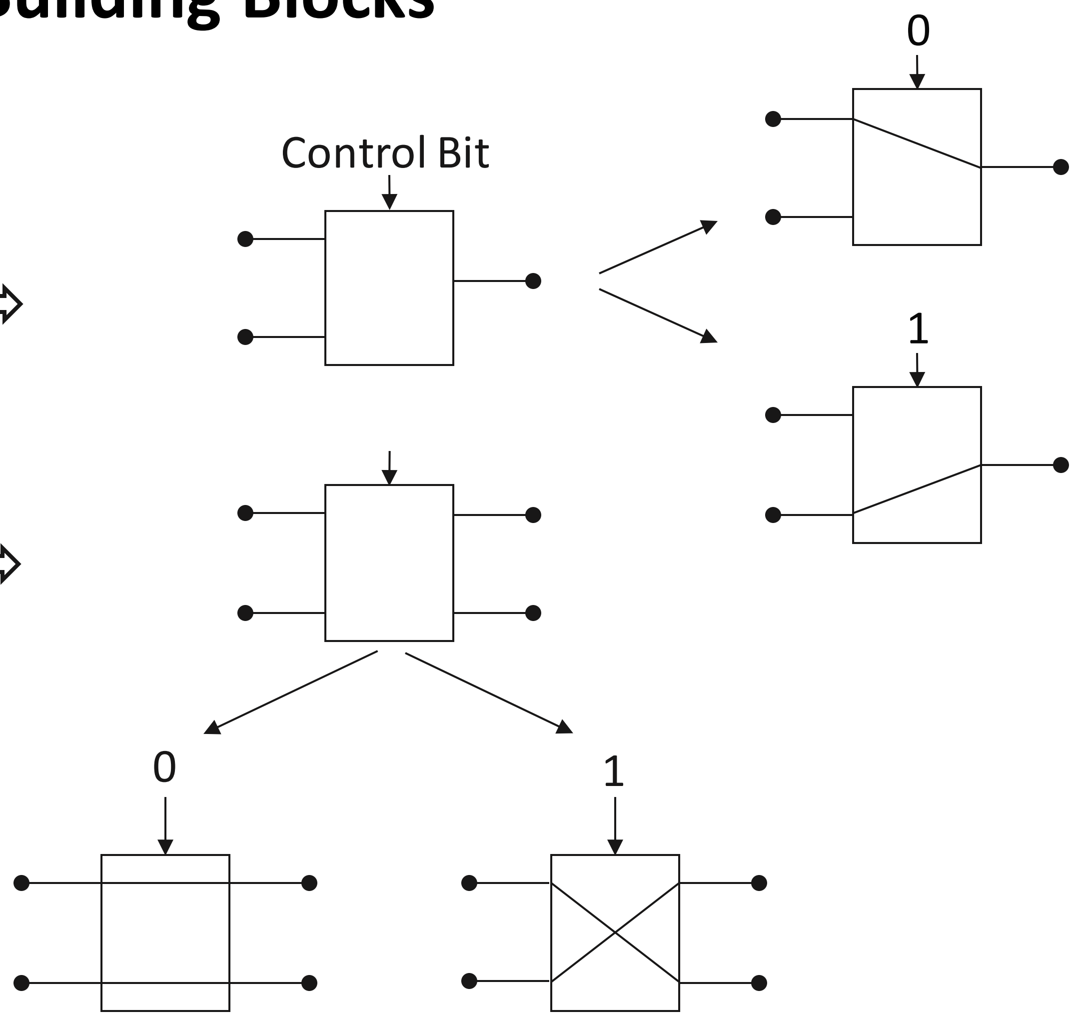


Multistage Network (3)

Building Blocks

$2 \rightarrow 1$ MUX ⇨

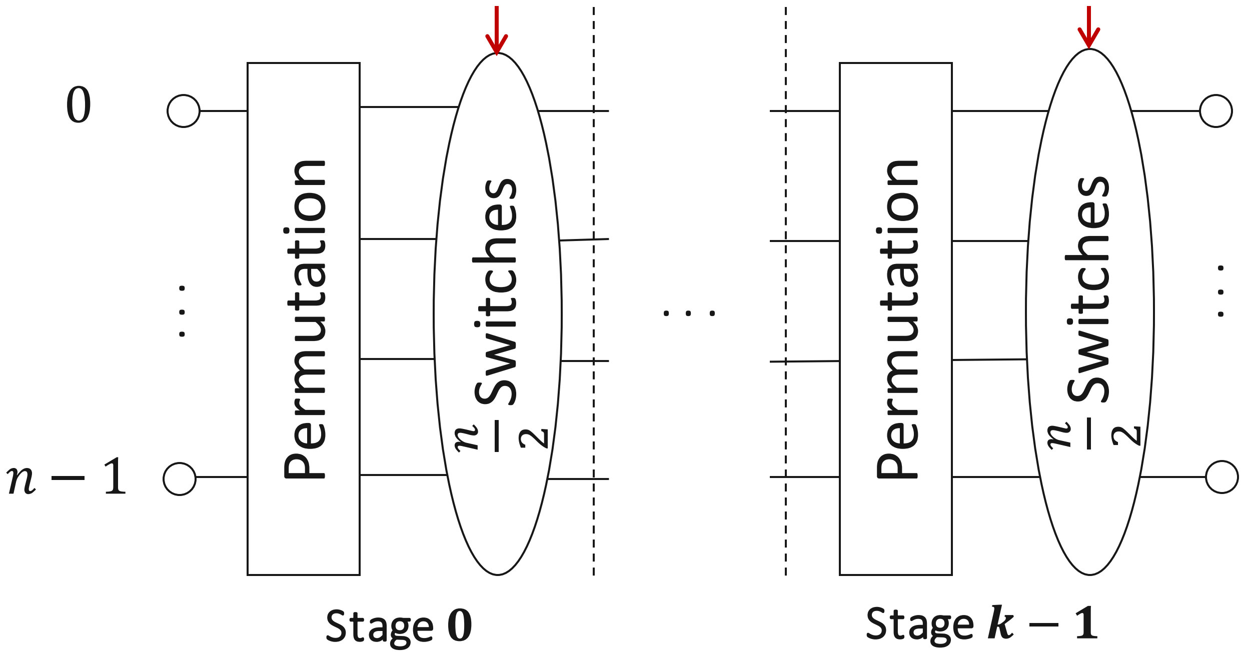
2×2 switch ⇨

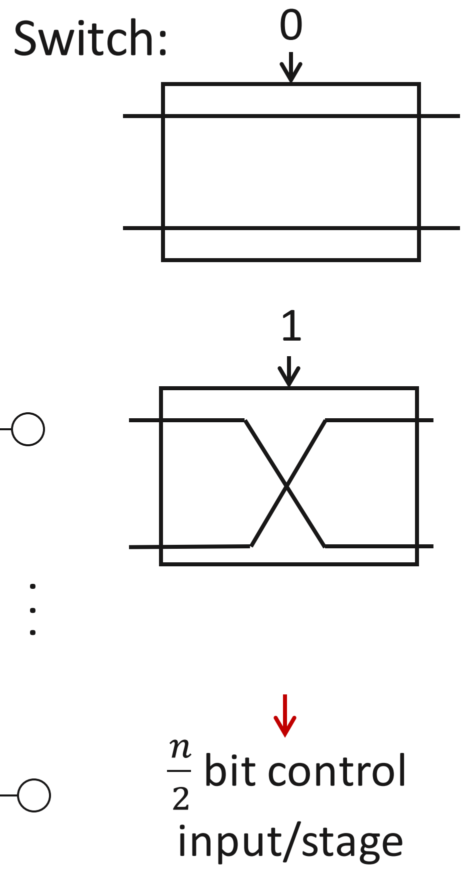


Multistage Network (4)

Multistage network

- k stages, $k \geq 1$
- $\frac{n}{2}$ switches in each stage
- delay = k





Performance of a Multistage Network

1 Switch = unit cost

unit delay

- Cost of a network: Total no. of 2×2 and 2×1 switches

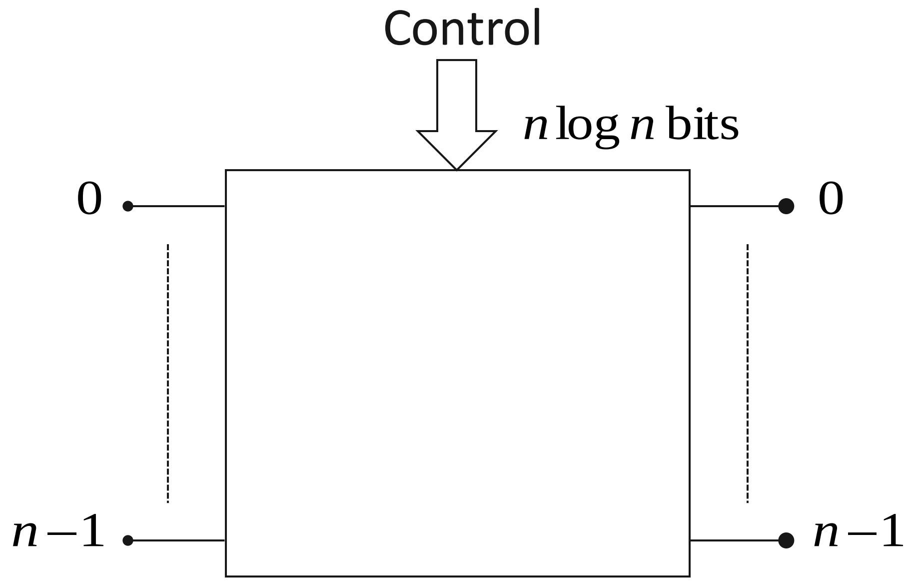
  Note: **Wiring cost ignored**

- Traditional metric
- Important metric
- Delay = Total # of stages

  Note: **Wire delay ignored**

Example Multistage Network (1)

CROSS BAR Switch



All $n!$ permutations can be realized

Example Multistage Network (2)

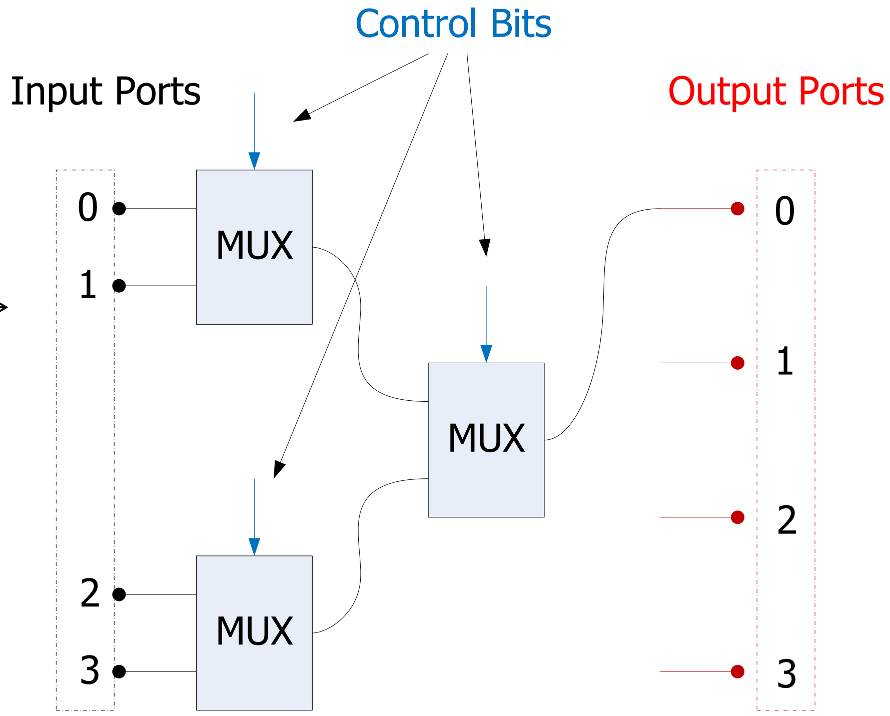
Example implementation using MUX

COST

- # of logic gates
- # of 2 → 1 MUX

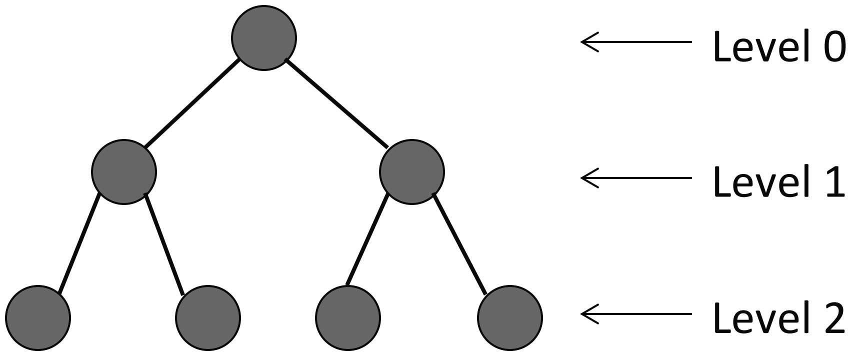
  = $\boldsymbol{O}(\boldsymbol{n} \cdot [\boldsymbol{n}])$

expensive



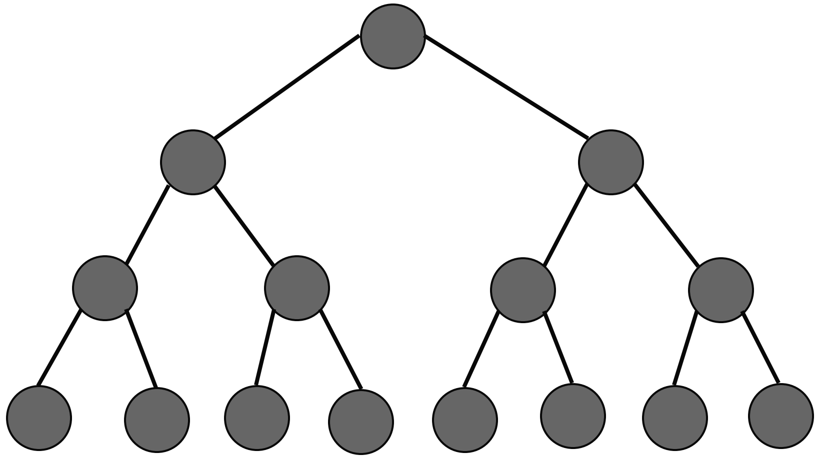
Total Number of Nodes in a Complete Binary Tree

$n = 4$



$$4 + 2 + 1 = 7 = 2 \cdot n - 1$$
$$= 2 \cdot (\text{\# of leaf nodes}) - 1$$

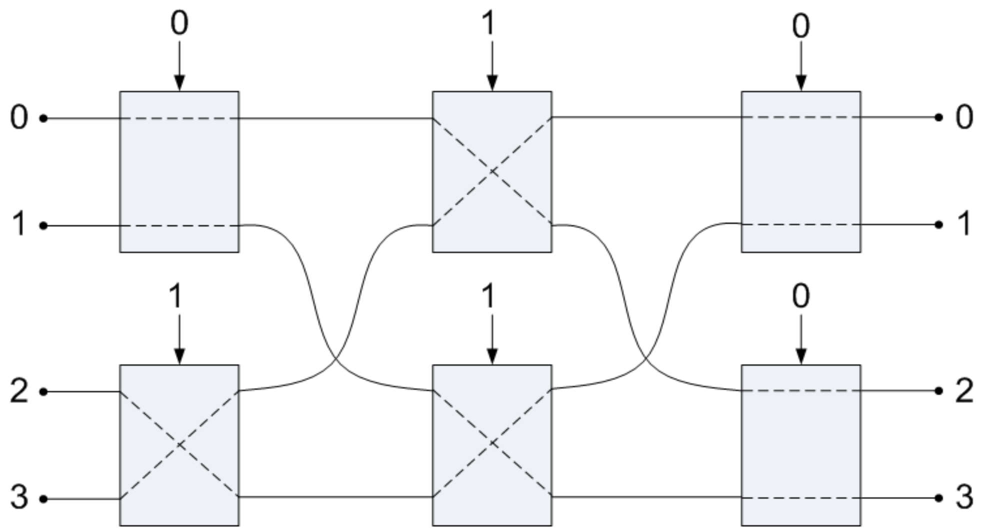
$n = 8$



$$T(n) = 2T\left(\frac{n}{2}\right) + 1 = 15$$

Routing (1)

Example of Multistage Network



Permutation

$0 \rightarrow 2$

$1 \rightarrow 3$

$2 \rightarrow 1$

$3 \rightarrow 0$

$\mathbf{2^6}$ combinations of switch settings

Note: All $4!$ Permutations can be realized by this network

- **Delay** = No. of stages
  = 3 stages
- **Cost** = No. of switches
  $= \frac{n}{2} \times$ (**number of stages**)
  $= 2 \times 3$

Routing (2)

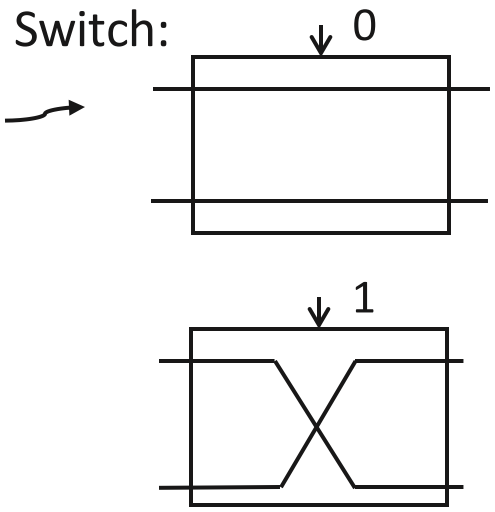
k stage, n input network

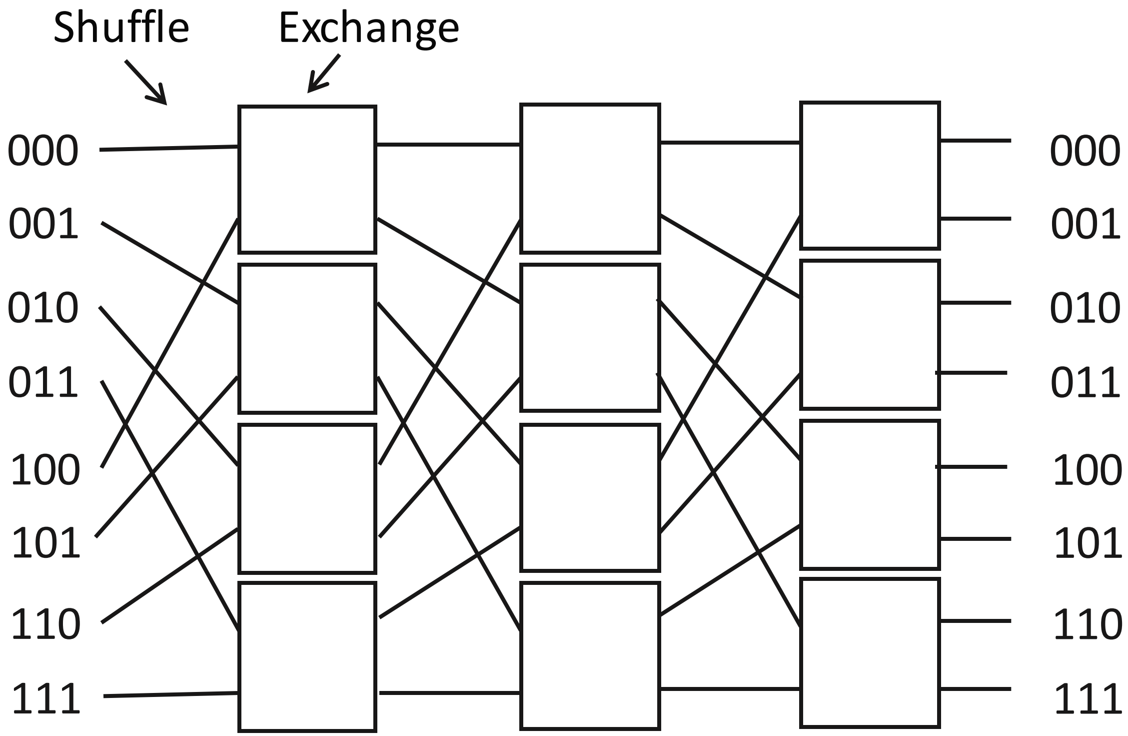
- Total number of switches: $\frac{n}{2} \cdot k$
- Total number of control bits: $\frac{n}{2} \cdot k$
  - Control bits specify a configuration of the network
    - Configuration → permutation from input to output
- Total number of permutations that can be realized: $\leq 2^{nk/2}$
- If we want all $n!$ permutations to be realized: $2^{nk/2} \geq n!$

$$\boxed{k = no.of\ stages = \Omega(\log n)}$$

Omega Network (1)

- p inputs, p outputs
- $\log_2 p$ stages, each stage having $\frac{p}{2}$ 2×2 switches





Omega Network (2)

Multistage network

Omega network properties

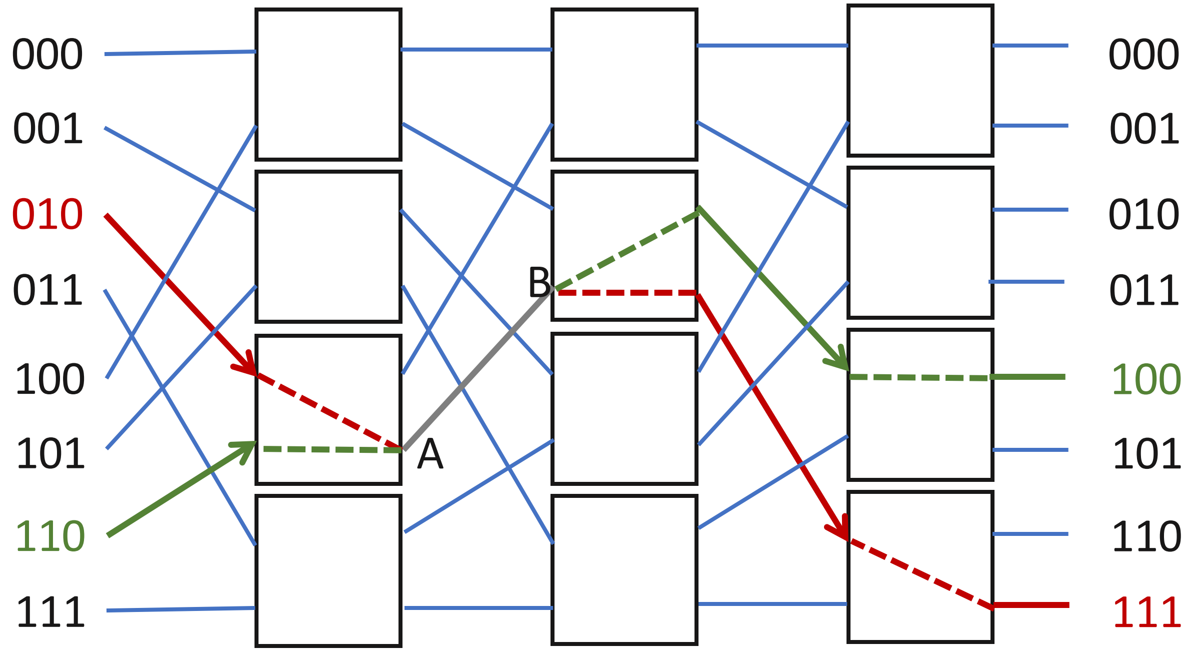
- Cost $= \frac{P}{2}\log_2 p$ (number of switches)
- Note: in actual hardware design, routing cost dominates !
- Omega network can do $2^{(\frac{p}{2}\log_2 p)} < p!$ Permutations

  $\rightarrow$ All $p!$ permutations **can not** be realized
- Unique (only one) path from any input to any output

Omega Network (3)

Example of Blocking

- one of the messages (010 to 111 or 110 to 100) is blocked at link AB



Blocking: To realize a given connecting pattern, two inputs to a switch need to go through the same output port of the switch

Omega Network and Shuffle Exchange Network

n node SE network $\equiv$ Omega network

(S+E) $\log_2 n$ times

with n inputs

Specify $\frac{n}{2}$ bits for each exchange step

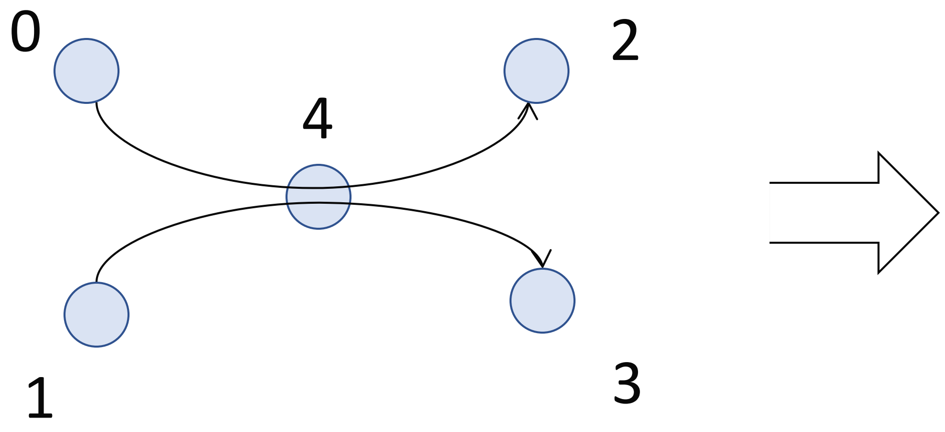
(We can pipeline here)

(Both networks realize the same set of permutations)

Congestion in a Network (1)

Given a routing protocol and data communication pattern (ex. a permutation)

Congestion at a node = Max. { # of paths passing through the node}



Congestion = 2

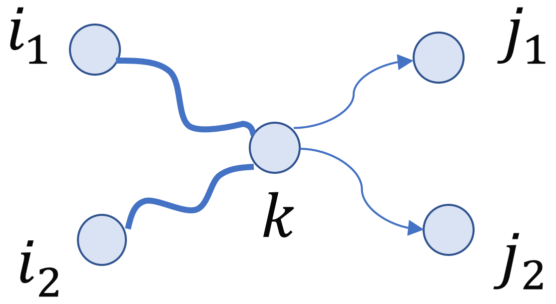
Congestion in a Network (2)

Interconnection Network = Graph + Routing algorithm

Assume the routing algorithm provides unique (exactly one path) communication from source i to destination j for all i, j

For a given permutation:

- Congestion at node k = # of paths (based on the routing protocol) that pass through k



Congestion in a Network (3)

- Congestion for a given permutation in the network $= \operatorname*{Max}_{\text{over nodes } k} \{\text{\# of paths that pass through } k\}$

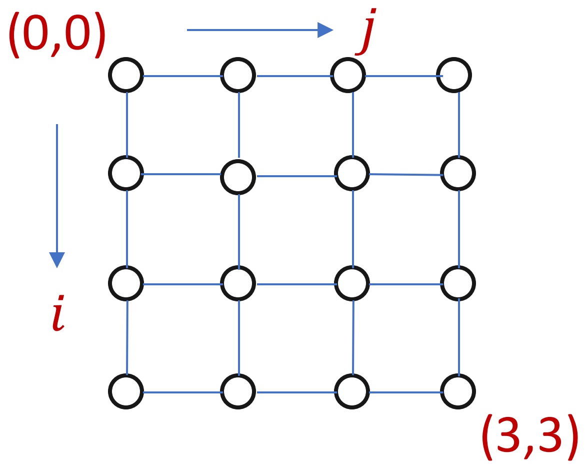
- Congestion in the network $= \operatorname*{Max}_{\text{all permutations}} \{\text{congestion in the network for a given permutation}\}$

Example Computation of Congestion (1)

2-D Mesh Architecture

$n \times n$ mesh

Data $A(i,j)$ in $PE(i,j)$, $\quad 0 \leq i,j < n$

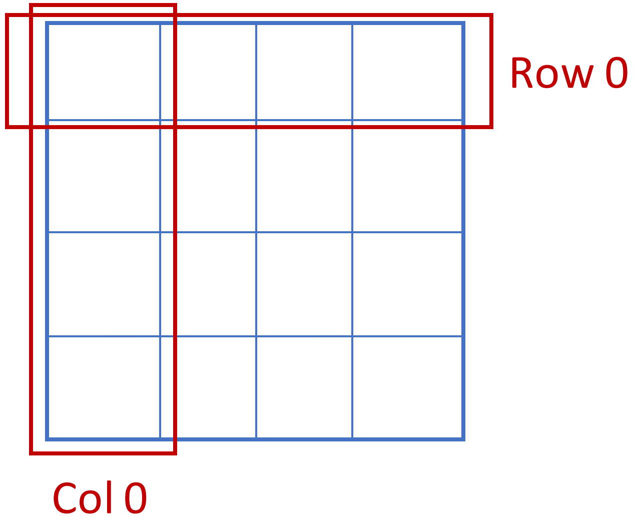


Example Computation of Congestion (2)

Permutation: Matrix Transpose

Data in $PE(i,j)\ [A(i,j)] \rightarrow PE(j,i) \quad 0 \leq i,j < n$

Row i → Col $i \quad 0 \leq i < n$



Example Computation of Congestion (3)

Routing

- Dimension ordered routing
- Widely used technique

For any permutation $(i, j) \rightarrow (d_i, d_j) \quad 0 \leq i, j < n$

1) Go along row i to destination column d_j $\leftarrow$ dimension 0
2) Go along column d_j to destination row d_i $\leftarrow$ dimension 1

Example Computation of Congestion (4)

Row 0 → Col 0

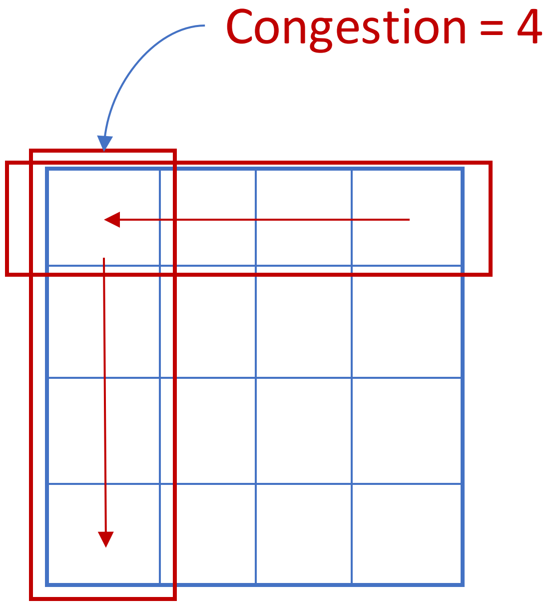
$n = 4$

$(0,0) \rightarrow (0,0)$

$(0,1) \rightarrow (1,0)$

$(0,2) \rightarrow (2,0)$

$(0,3) \rightarrow (3,0)$



Example Computation of Congestion (5)

Using dimension ordered routing, $\boldsymbol{n \times n}$ mesh

- For any permutation
  - Congestion at any node $\leq n$
- Matrix Transpose results in worst case congestion
  - Congestion at $PE(i, i) = n, \;\; 0 \leq i < n$
  - Congestion in the network $= n$

Summary

- Interconnection networks
  - Static/Dynamic network
  - Shuffle exchange network
  - Multistage network
  - Routing
  - Congestion