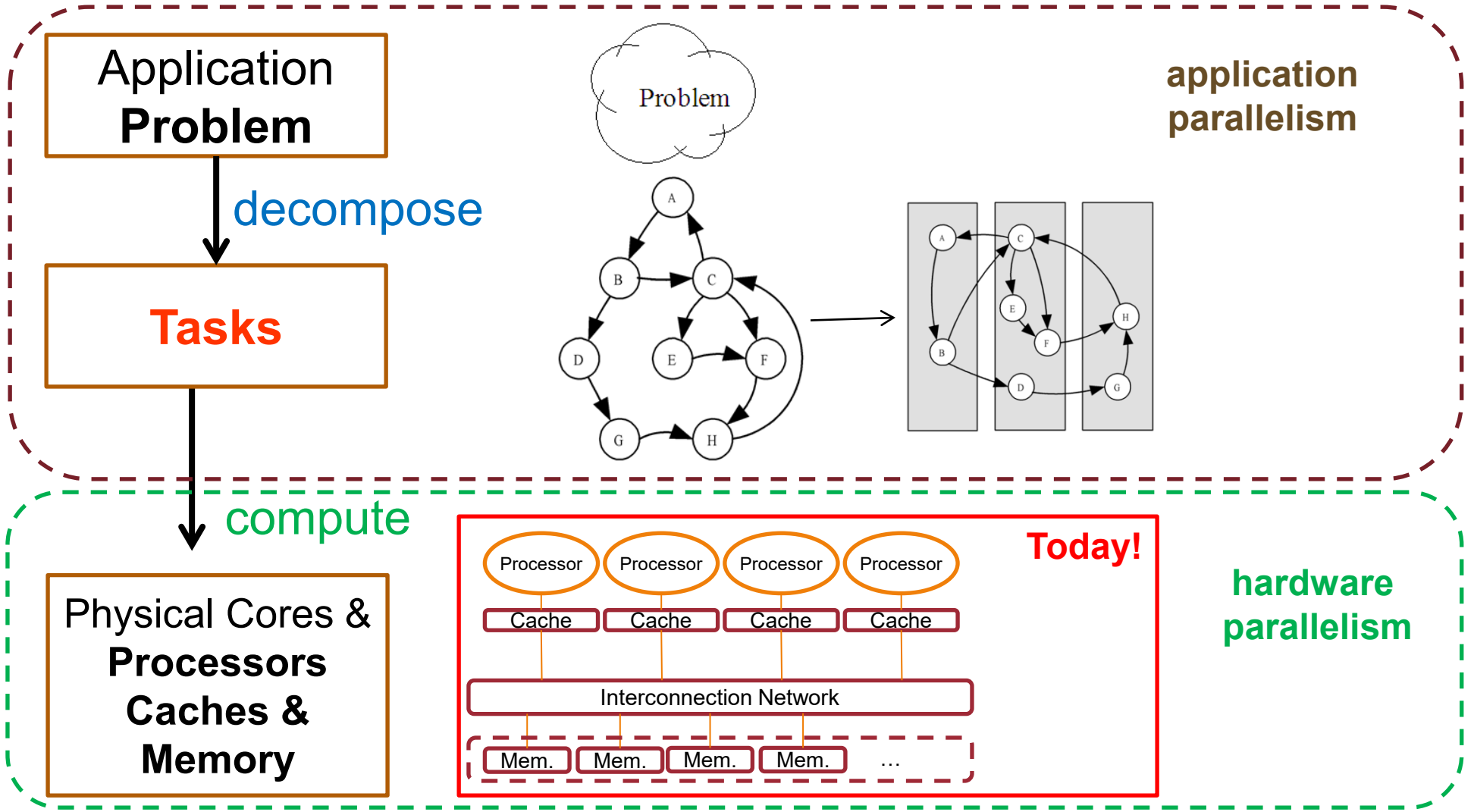
Interconnection Networks

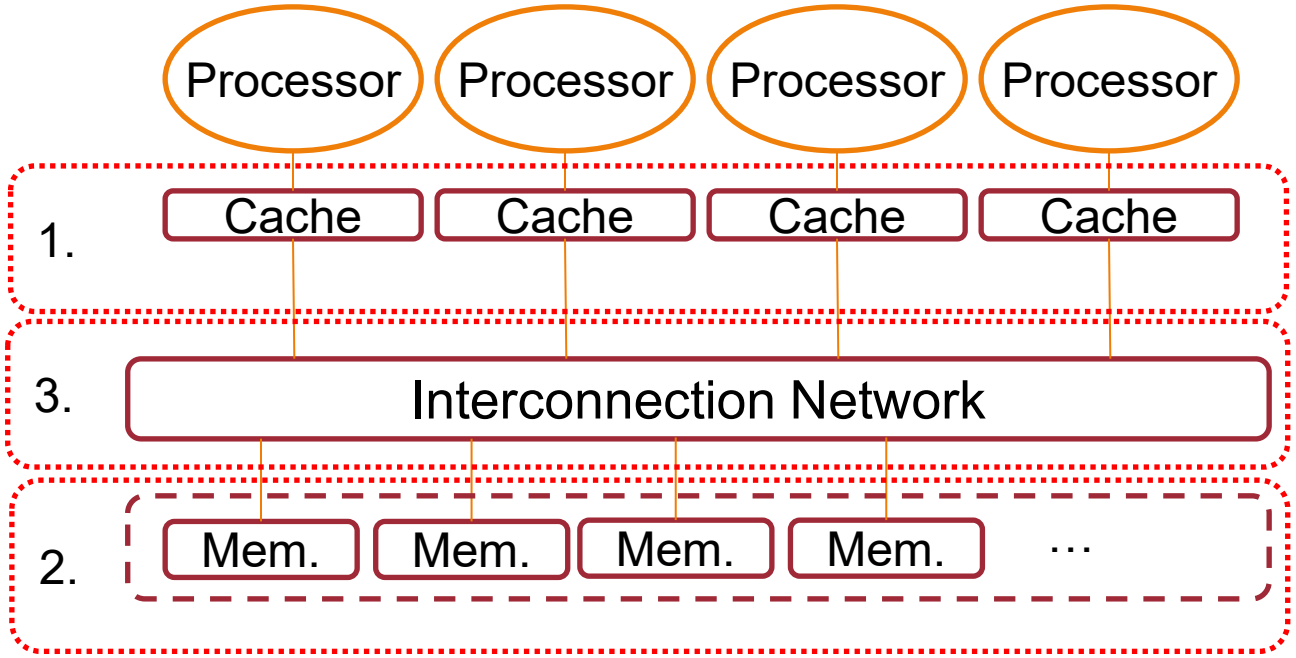
Lecture 11

Parallel Computing

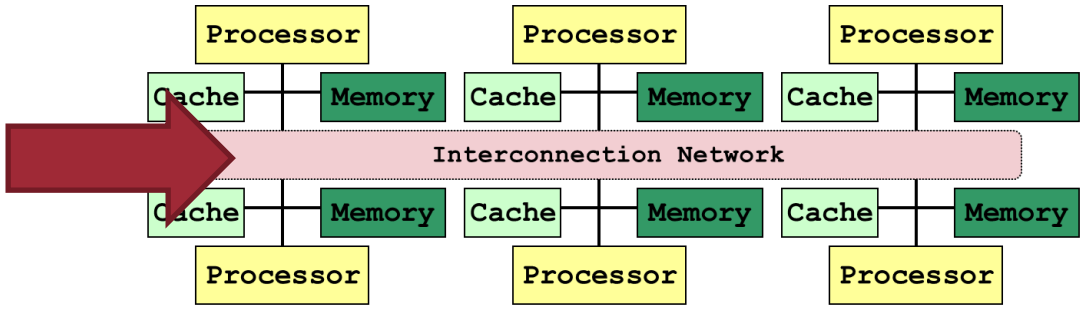


Outline

1. Cache Coherence
2. Memory Consistency
3. Interconnection Networks
   - Motivating Example
   - Topology
     - Metrics
     - Direct and Indirect Interconnection Networks
   - Routing
   - Current Trends



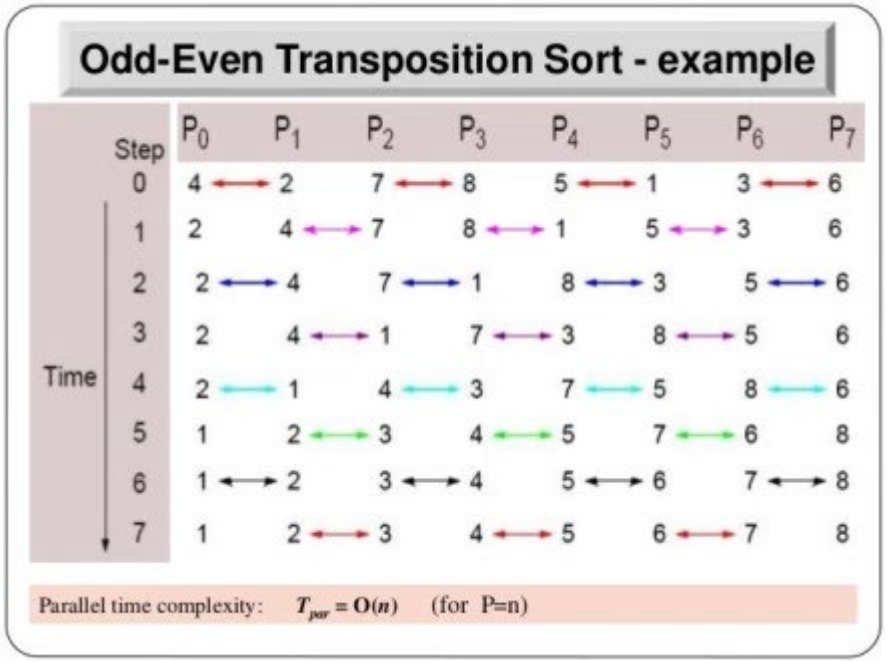
Interconnections: Motivating Example



- Interconnection forms the backbone of communication between:
  - Processors
  - Processor and memories
  - Processors and caches
  - I/O devices
- Let us first see two examples of interconnect network:
  - Using simple sorting as the focus point

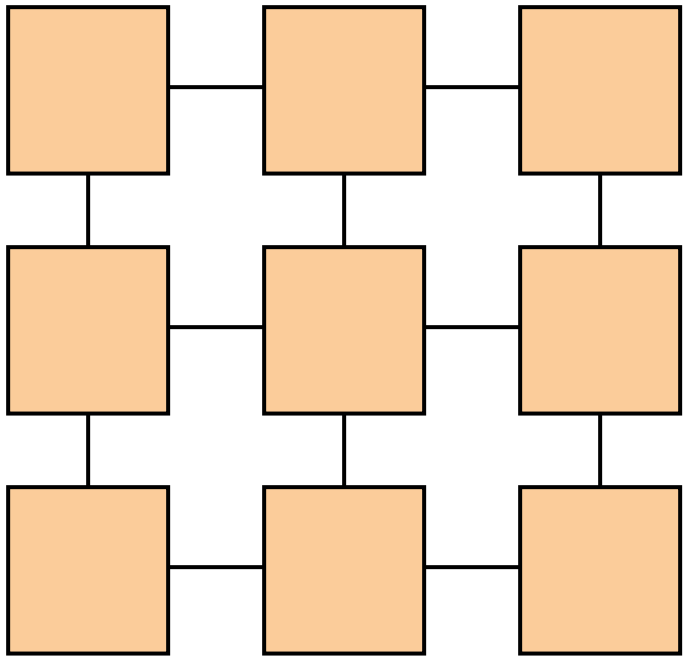
Sorting on Linear Array

Sort *N* numbers on *N*-PEs Linear Array



- Questions:
  - What is the number rounds needed?
  - Compare with sequential sorting algorithms?

Two Dimensional Mesh



3 x 3 PEs in a **Mesh**

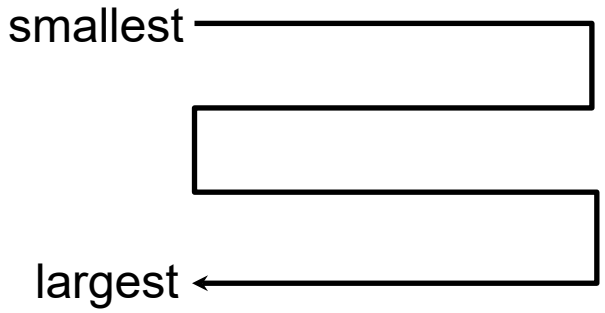
- Observe that the PEs have different number of communication links:
  - PE at the corners = 2 links
  - PE on the edges = 3 links
- If we wrap the left to right; top to bottom

  ➔ All PEs have four links ➔ known as **Torus**

Algorithm Essential

Sort *N* elements on *N*-PEs 2D Mesh

- Each PE has one number initially
- N-PEs arranged in a 2D Mesh:
  - √N rows
  - √N columns
- Sorted into “Snake-Like” order in the end:



Shear Sort Algorithm: Basic Idea

Phase 1: Row Sorting

- Odd Rows sort in **ascending** order
- Even Rows sort in **descending** order

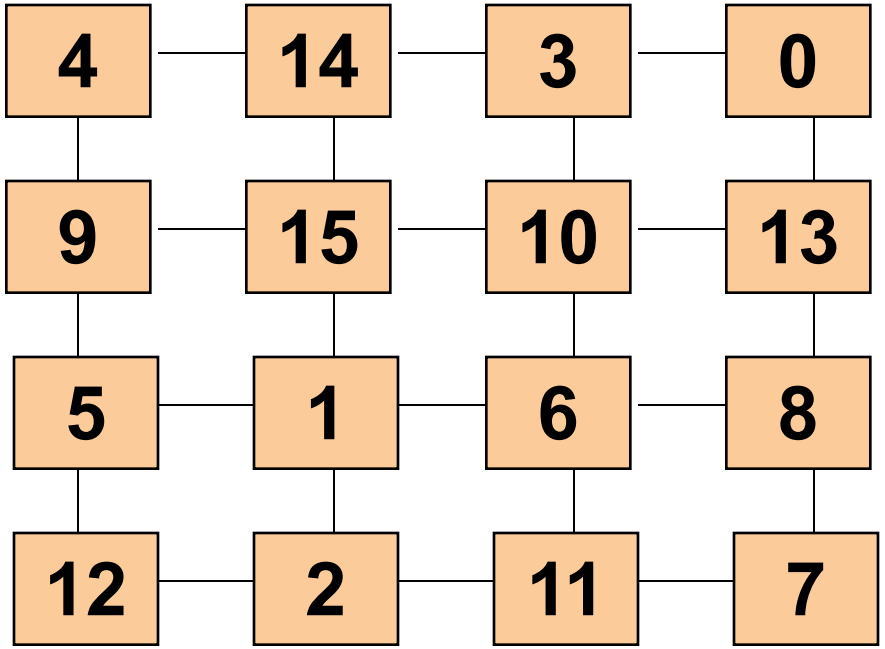
Phase 2: Column Sorting

- All columns sort in **ascending** order (top to bottom)

Repeat

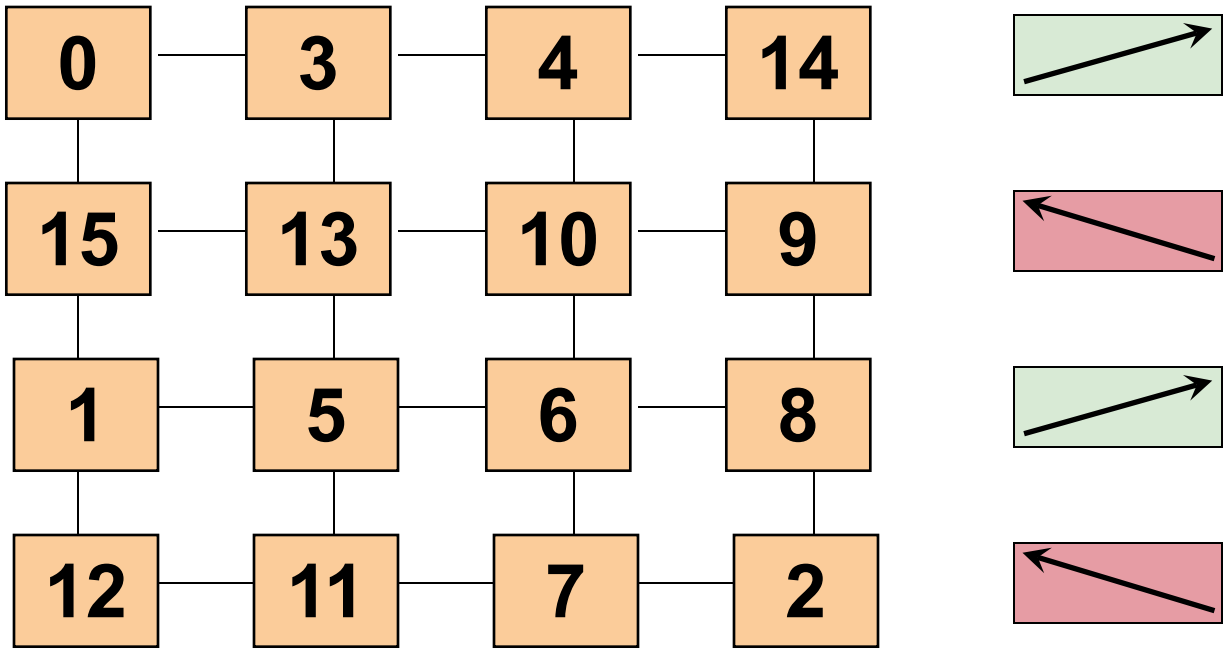
- Until sorted

Example: 16 Numbers – Initial Placement



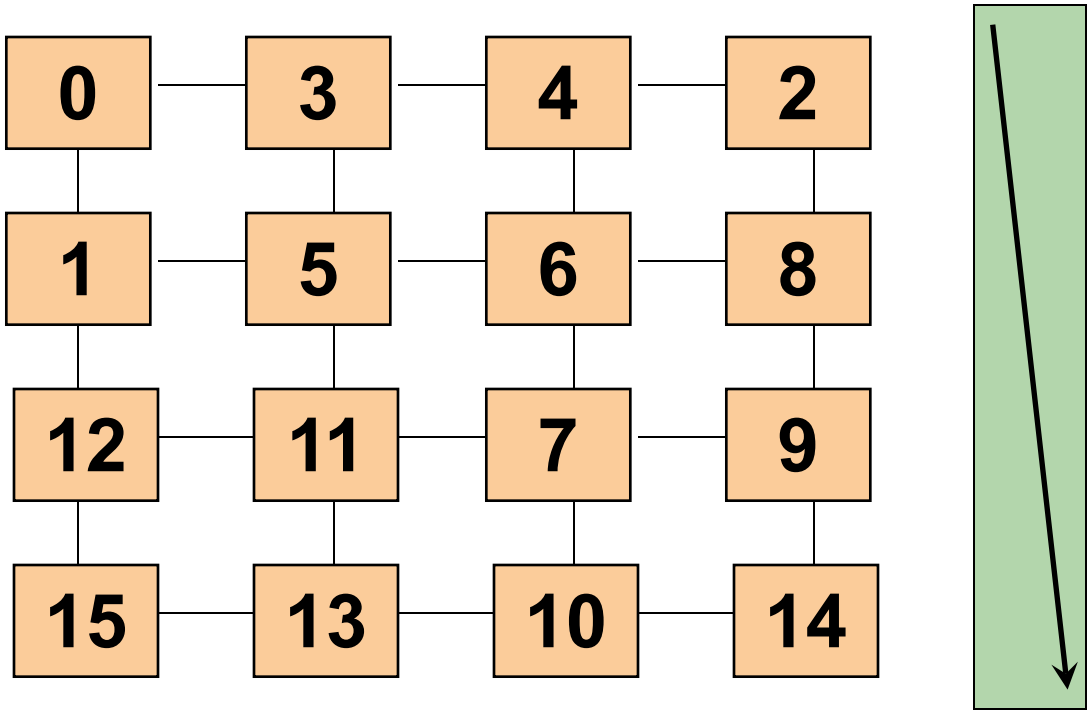
- Initial placement of 16 numbers (0 to 15)

Phase 1: **Row Sort**

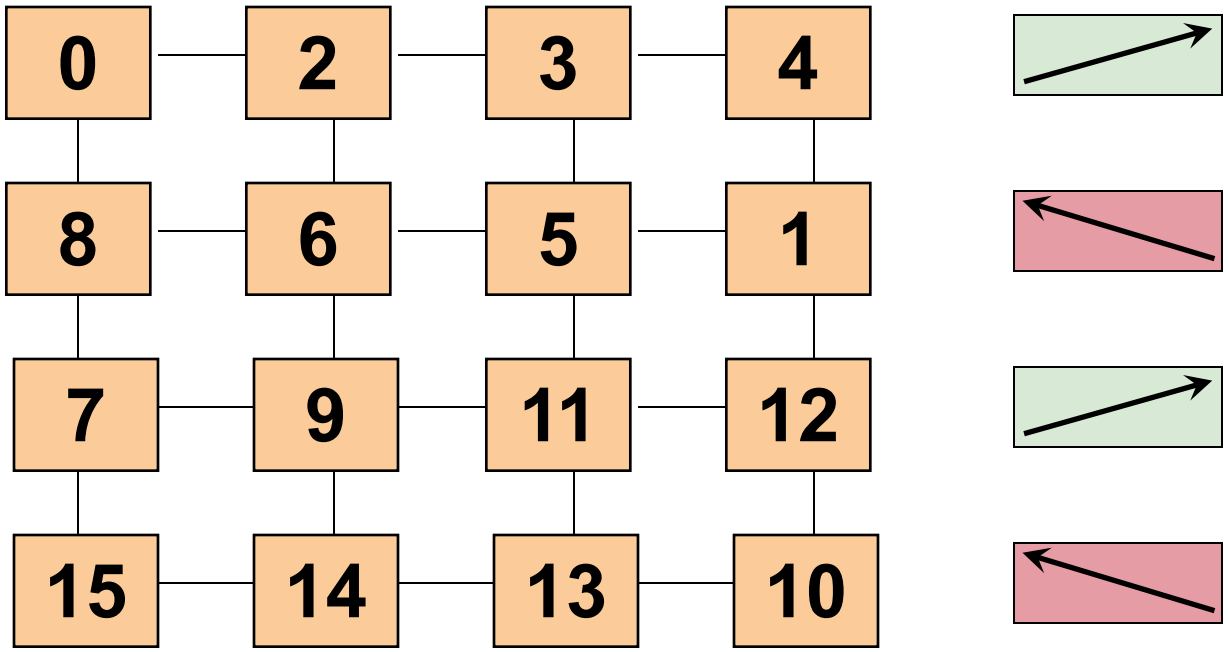


- Note the different order for each row

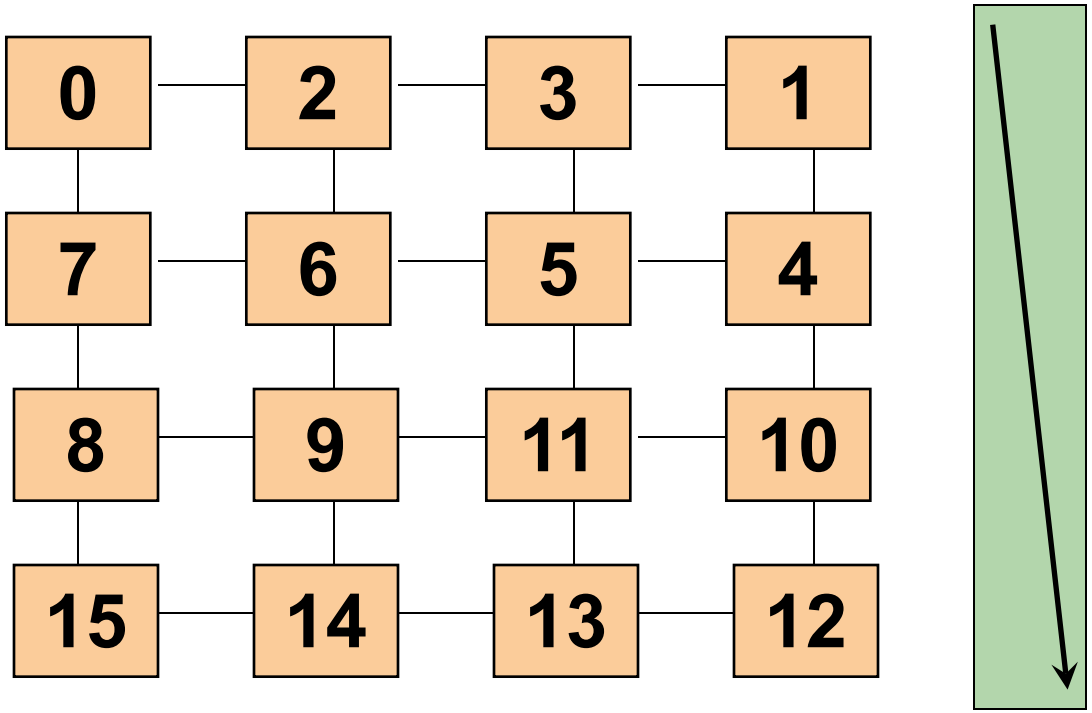
Phase 2: **Column Sort**



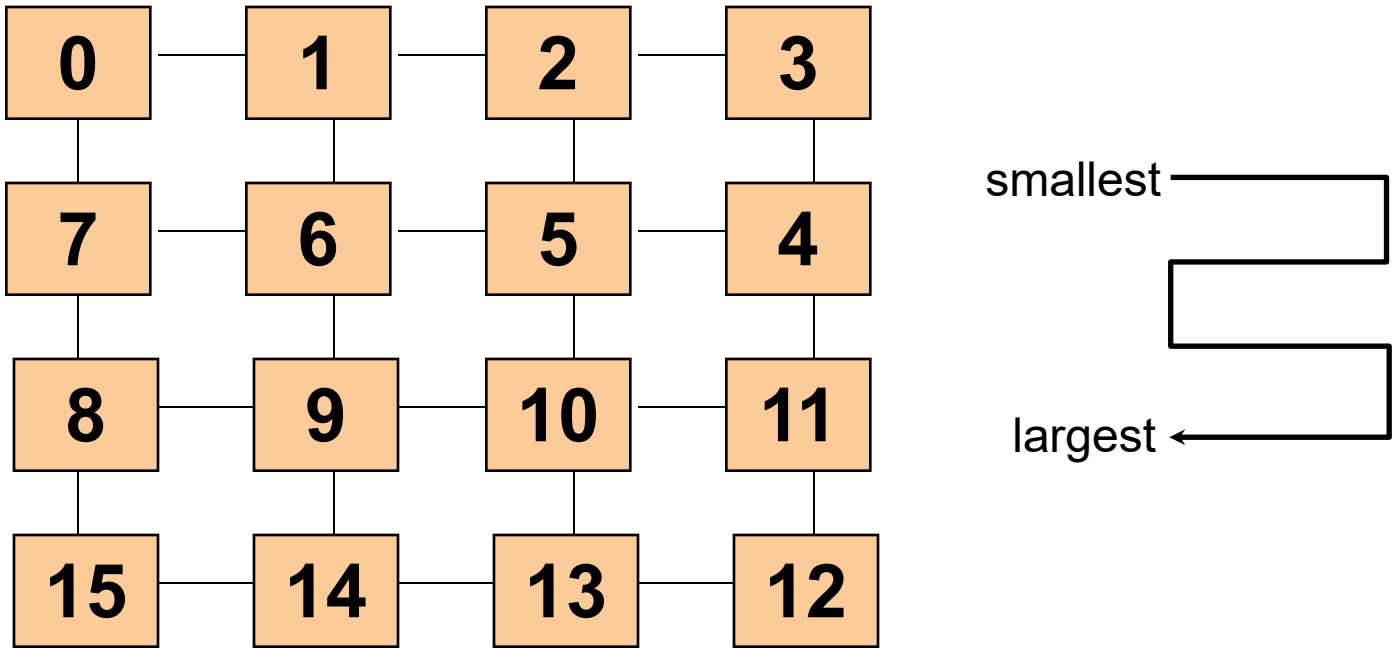
Phase 3: **Row Sort**



Phase 4: **Column Sort**



Phase 5: **Row Sort (Done!)**



- Sorted! Note the snake-like arrangement

Complexity

- A bold statement:

For **N** Numbers, **Log_2 N + 1** phases

- E.g. 16 Numbers = Log2 16 + 1
- = 5 phases

- How to perform each of the row/column sort?
  - Use … odd-even transposition sort
  - So, complexity is ……

We now know that

- Connection "patterns" enable different algorithms
- Connection "patterns" have a huge impact on execution
- Follow up questions:
  - What are the common ways to connect?
  - How to handle communication in these interconnection networks?

Interconnection Network Impact

- System scalability – size and extensibility
- System performance and energy efficiency
  - Communication speed
  - Latency to memory
  - Energy spent to communicate

Major Questions in Interconnections

Topology

- What is the geometrical shape of the connection?

Routing

- What is the path for a message to follow?

Switching

- How to transfer a message along a path?

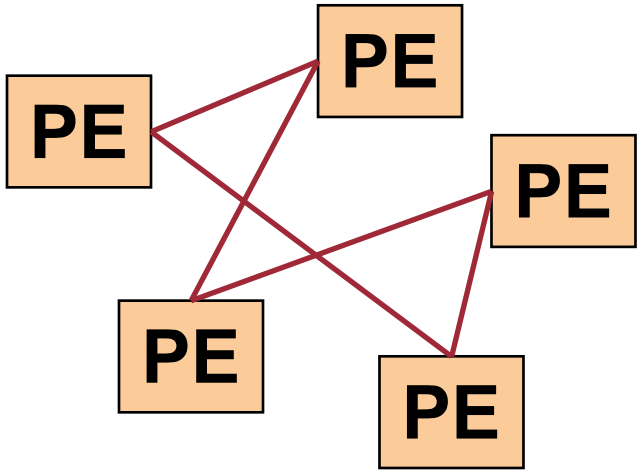
Flow Control

- How to handle concurrent messaging?

Beautiful patterns that induce headache….

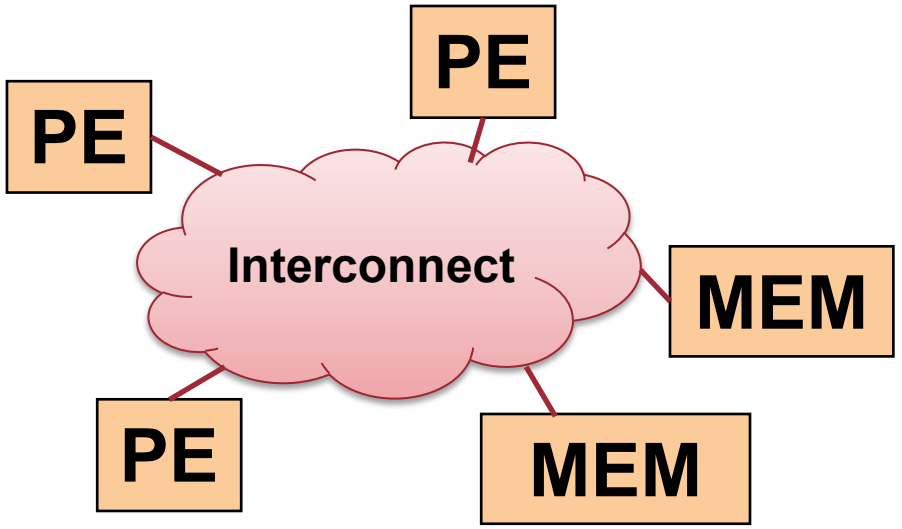
TOPOLOGY

Topology: Major Type



Direct Interconnection

- Also known as **Static** or **Point-to-Point**
- Usually endpoints are of the same type (core, memory)



Indirect Interconnection

- Also known as Dynamic
- Interconnect is formed by switches

Direct Interconnection: Overview

- Topology can be modeled as a Graph:
  - G = (V, E), V = vertices; E = edges
  - ➔ Can use graph theory to understand!
- However, some of the general cases are very hard to determine
  - ➔ Only study specific subcases

- Metrics:
  - Diameter
  - Degree
  - Bisection Width
  - Connectivity

Topology Metric - Diameter

- **Diameter δ(G):** maximum distance between any pair of nodes

$$\delta(G) = \max_{u,v \in V} \min_{\substack{\varphi \text{ path} \\ \text{from } u \text{ to } v}} \{k \mid k \text{ is the length of the path } \varphi \text{ from } u \text{ to } v\}.$$

- **Usefulness**:
  - Small diameter ensures small distances for message transmission

Topology Metric - Degree

- **Degree g(v):** number of direct neighbour nodes of node v
  - Degree g(G): maximum degree of a node in a network G

$$g(G) = \max\{g(v) \mid g(v) \text{ degree of } v \in V\}.$$

- **Usefulness:**
  - Small node degree reduces the node hardware overhead

Topology Metric - Bisection Width

- **Bisection width B(G):** minimum number of edges that must be removed in to divide the network into two equal halves

$$B(G) = \min_{\substack{U_1, U_2 \text{ partition of } V \\ ||U_1|-|U_2||\leq 1}} |\{(u, v) \in E \mid u \in U_1, v \in U_2\}|.$$

  - Bisection bandwidth BW(G): Total bandwidth available between the two bisected portion of the network

- **Usefulness:** a measure for the capacity of a network when transmitting messages simultaneously

Topology Metric - Connectivity

- **Node connectivity nc(G):** minimum number of nodes that must fail to disconnect the network

$$nc(G) = \min_{M \subset V} \{|M| \mid \text{there exist } u, v \in V \setminus M, \text{ such that there exists no path in } G_{V \setminus M} \text{ from } u \text{ to } v\}.$$

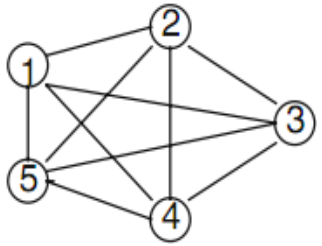
  - **Usefulness:** Determine the robustness of the network

- **Edge connectivity ec(G):** minimum number of edges that must fail to disconnect the network

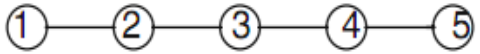
$$ec(G) = \min_{F \subset E} \{|F| \mid \text{there exist } u, v \in V, \text{ such that there exists no path in } G_{E \setminus F} \text{ from } u \text{ to } v\}.$$

  - **Usefulness:** Determine number of independent paths between any pair of nodes

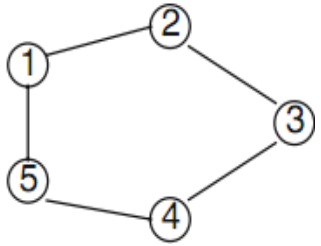
Examples



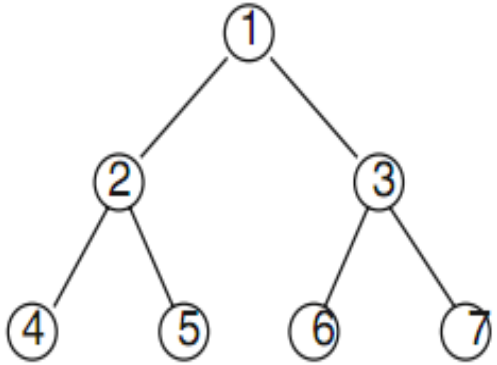
Complete network



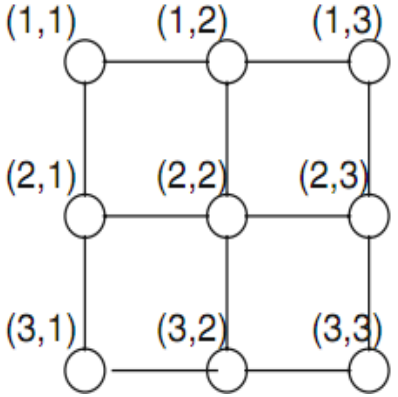
Linear array network



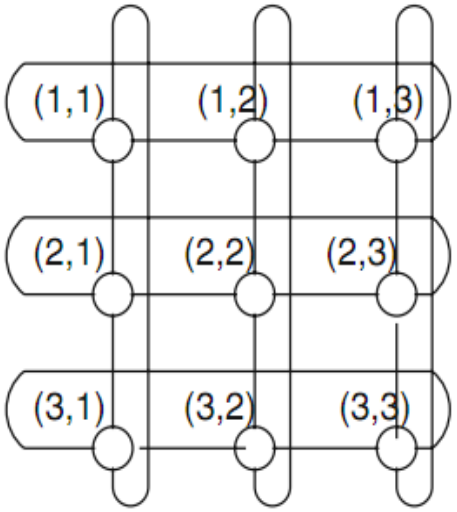
Ring network



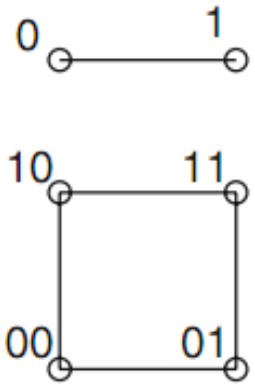
complete binary tree

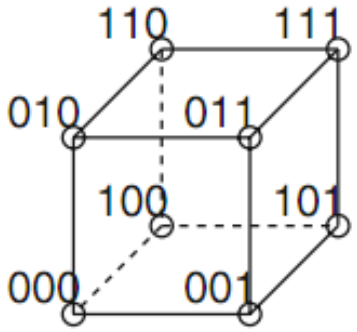


2-dimensional mesh

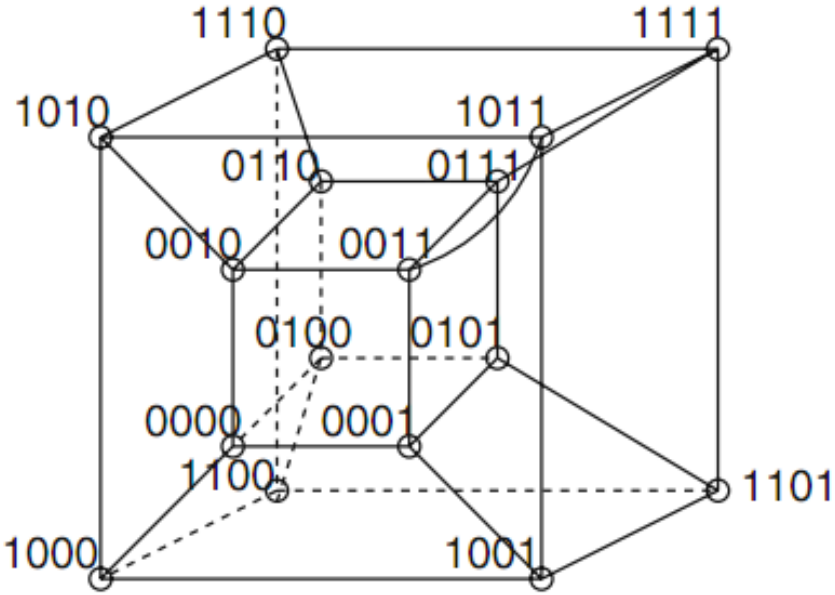


2-dimensional torus

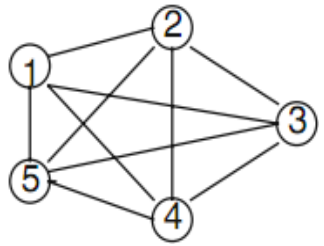




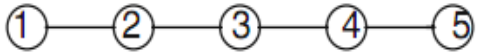
Hypercube



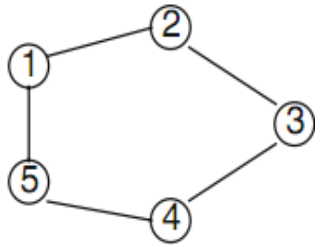
Complete, Linear Array & Ring



Complete network



Linear array network

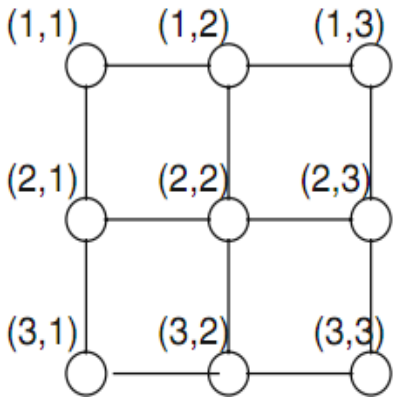


Ring network

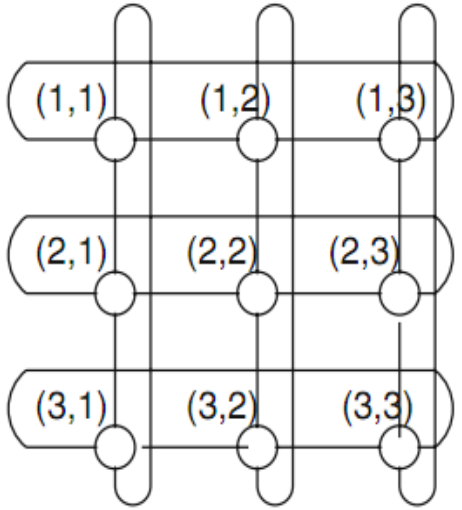
Degree **g(G)**	$n-1$	2	2
Diameter **δ (G)**	1	$n-1$	$\lfloor \frac{n}{2} \rfloor$
Edge Connectivity **ec(G)**	$n-1$	1	2
Bisection Width **B(G)**	$\left(\frac{n}{2}\right)^2$	1	2

2-Dimensional Mesh and Torus

$$n = r^2$$



2-dimensional mesh

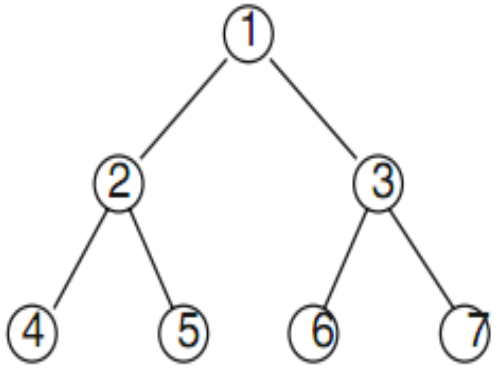


2-dimensional torus

Degree **g(G)**	**4**	**4**
Diameter **δ (G)**	**4**	**2**
Edge Connectivity **ec(G)**	**2**	**4**
Bisection Width **B(G)**	**4**	~~**8**~~ **6**

- Generalize to ***d***-dimension
  - Number of node $n = r^d$

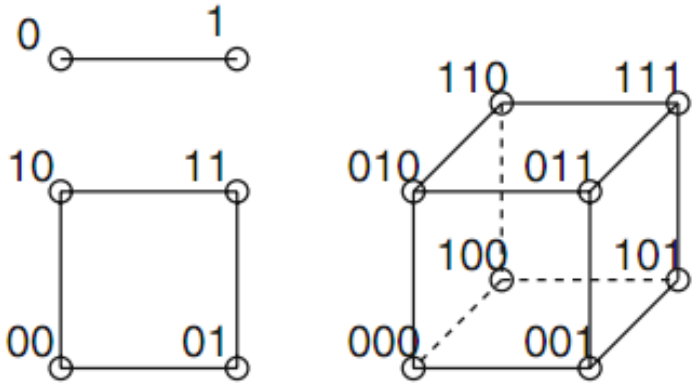
Complete Binary Tree



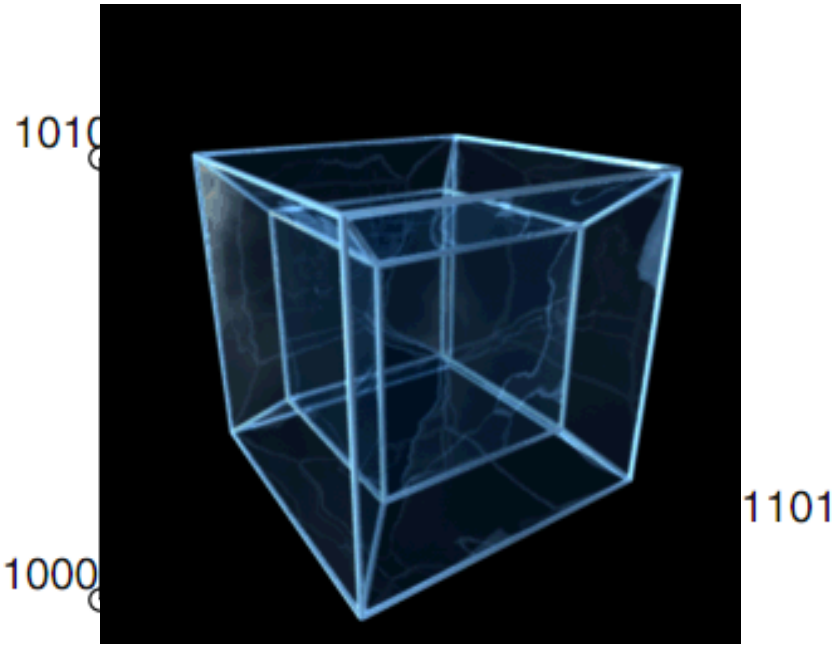
complete binary tree

Degree **g(G)**	**3**
Diameter **δ (G)**	**4**
Edge Connectivity **ec(G)**	**1**
Bisection Width **B(G)**	**1**

Hypercube



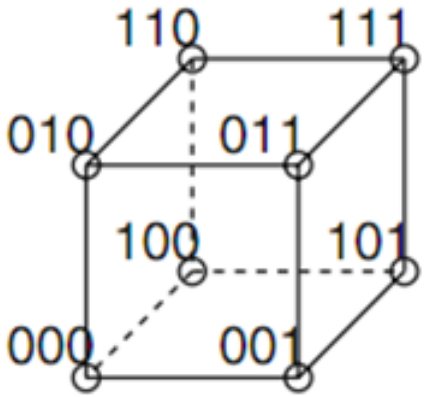
$$n = 2^k$$



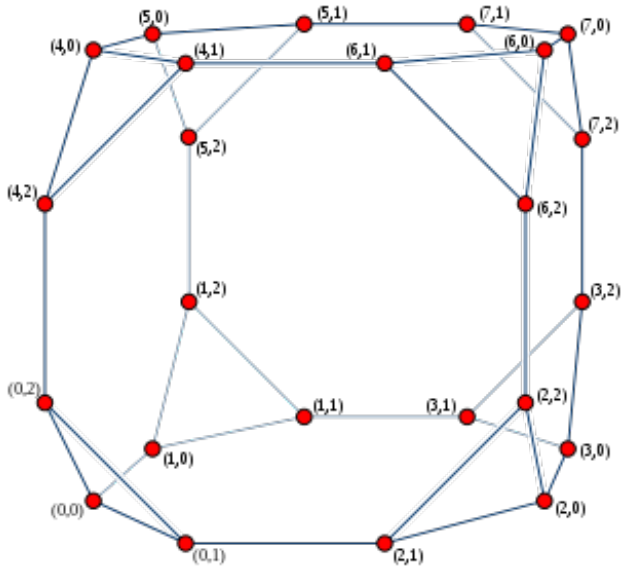
k-dimensional hypercube for k=1,2,3,4

Degree **g(G)**	**Log n**
Diameter **δ (G)**	**Log n**
Edge Connectivity **ec(G)**	**Log n**
Bisection Width **B(G)**	**n/2**

Cube-Connected-Cycles (CCC)



Hypercube, k = 3



Cube-Connected-Cycles network for k = 3

- From a k-dimensional hypercube (k ≥ 3)
  - Substitute each node with a cycle of k-nodes
  - Each of the k-nodes take one of the original k links
  - ➔ Total nodes = $k2^k$

CCC - Construction

- Each node in a k-dimensional CCC is labeled as (X, Y)
  - X = the corresponding node index in hypercube
  - Y = the position in the cycle, i.e. 0…k-1

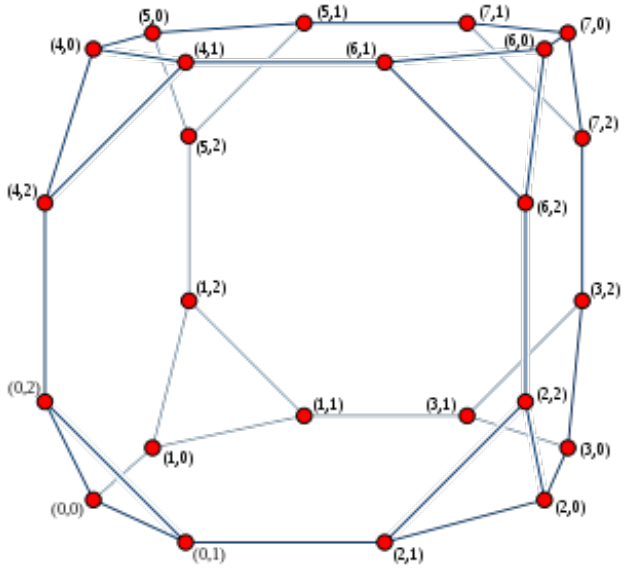
- Node (X, Y) is connected to:
  - (X, (Y+1) mod K)
  - (X, (Y-1) mod K)

  } The "Cycle Buddies"

  - $(X \oplus 2^y, Y)$

  } The link from the corresponding dimension in hypercube

CCC - Metrics



Degree **g(G)**	$\mathbf{3}$
Diameter **δ (G)**	$\mathbf{2k - 1 + \left\lfloor \frac{k}{2} \right\rfloor}$
Edge Connectivity **ec(G)**	$\mathbf{3}$
Bisection Width **B(G)**	$\mathbf{\frac{n}{2k}}$

Many others

- Generalized d-dimensional mesh
- Generalized d-dimensional torus
- Generalized k-ary d-dimensional cube
- Kautz Graph
- etc….

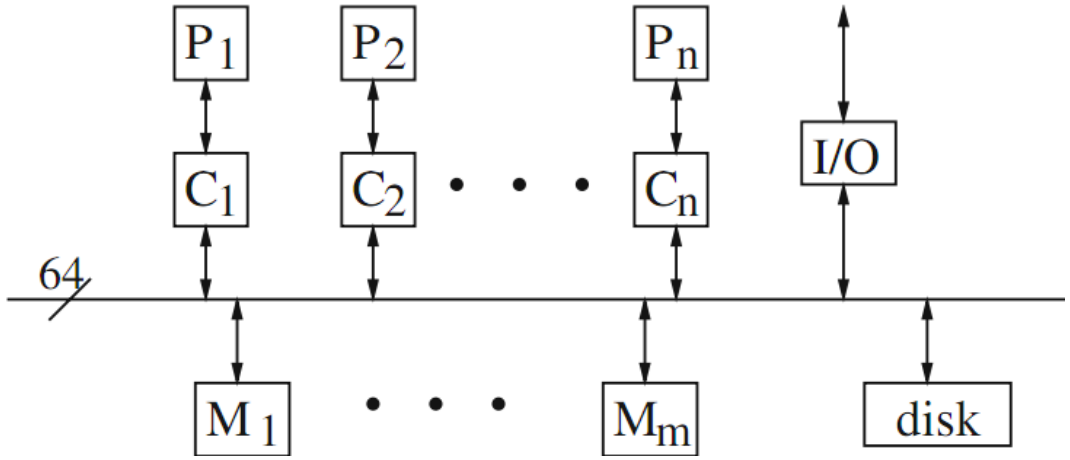
Summary of Metrics

network G with n nodes	degree $g(G)$	diameter $\delta(G)$	edge-connectivity $\mathrm{ec}(G)$	bisection bandwidth $B(G)$
complete graph	$n-1$	1	$n-1$	$\left(\frac{n}{2}\right)^2$
linear array	2	$n-1$	1	1
ring	2	$\lfloor \frac{n}{2} \rfloor$	2	2
d-dimensional mesh ($n = r^d$)	$2d$	$d(\sqrt[d]{n}-1)$	d	$n^{\frac{d-1}{d}}$
d-dimensional torus ($n = r^d$)	$2d$	$d\left\lfloor \frac{\sqrt[d]{n}}{2} \right\rfloor$	$2d$	$2n^{\frac{d-1}{d}}$
k-dimensional hyper-cube ($n = 2^k$)	$\log n$	$\log n$	$\log n$	$\frac{n}{2}$
k-dimensional CCC-network ($n = k2^k$ for $k \geq 3$)	3	$2k-1+\lfloor k/2 \rfloor$	3	$\frac{n}{2k}$
complete binary tree ($n = 2^k - 1$)	3	$2\log\frac{n+1}{2}$	1	1
k-ary d-cube ($n = k^d$)	$2d$	$d\lfloor \frac{k}{2} \rfloor$	$2d$	$2k^{d-1}$

Indirect Interconnection: Overview

- Why?
  - Reduce hardware costs by sharing switches and links
- How?
  - Switches provide indirect connection between nodes and can be configured dynamically
- What metric?
  - Cost (number of switches / links)
  - Concurrent connections

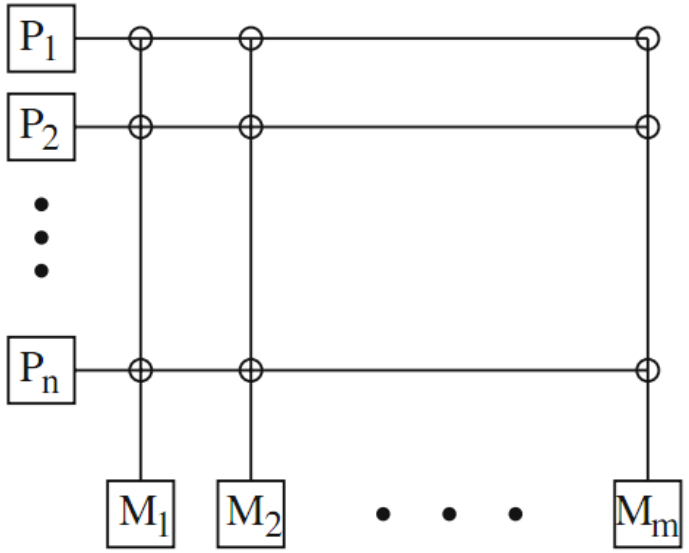
Bus Network



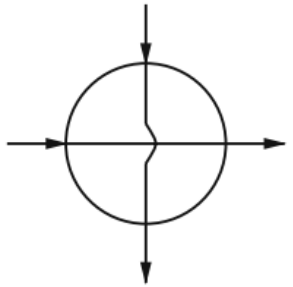
- A set of wires to transport data from a sender to a receiver
- Only one pair of devices can communicate at a time
  - A bus arbiter is used for the coordination
  - ➔ Typically used for a small number of processors

Crossbar Network

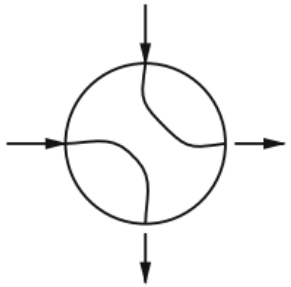
- A n × m crossbar network has n inputs and m outputs



n x m crossbar network



Straight state

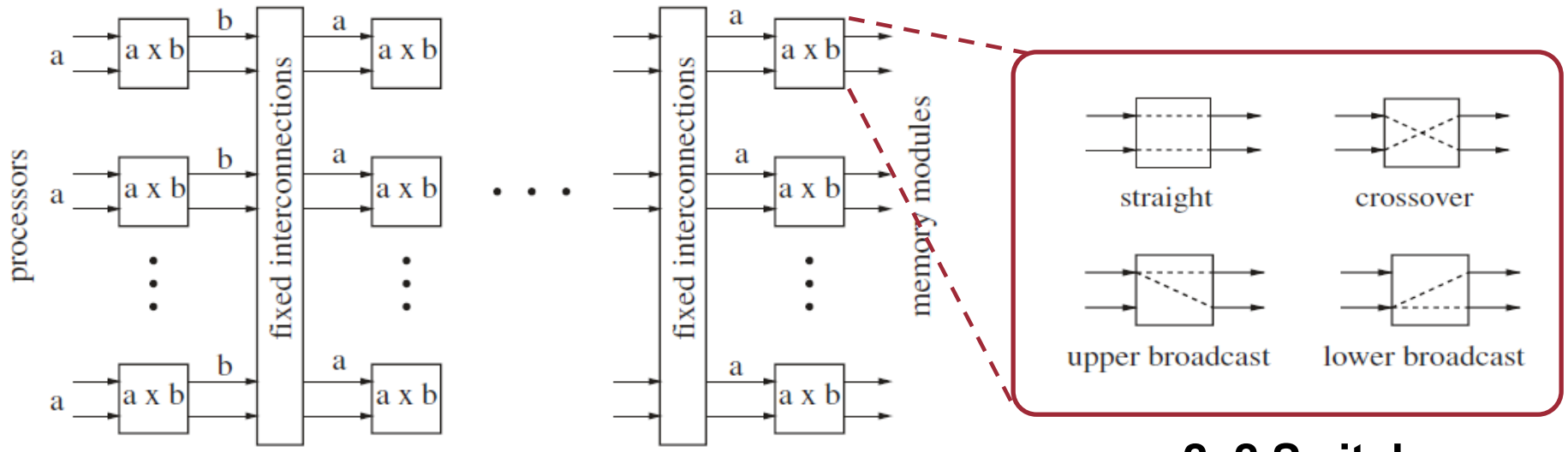


Direction change state

- Two states of a switch: straight or direction change
- Hardware is costly (n x m switches) → small number of processors

Multistage Switching Network

- Several intermediate switches with connecting wires between neighbouring stages
- **Goal:** obtain a small distance for arbitrary pairs of input and output devices

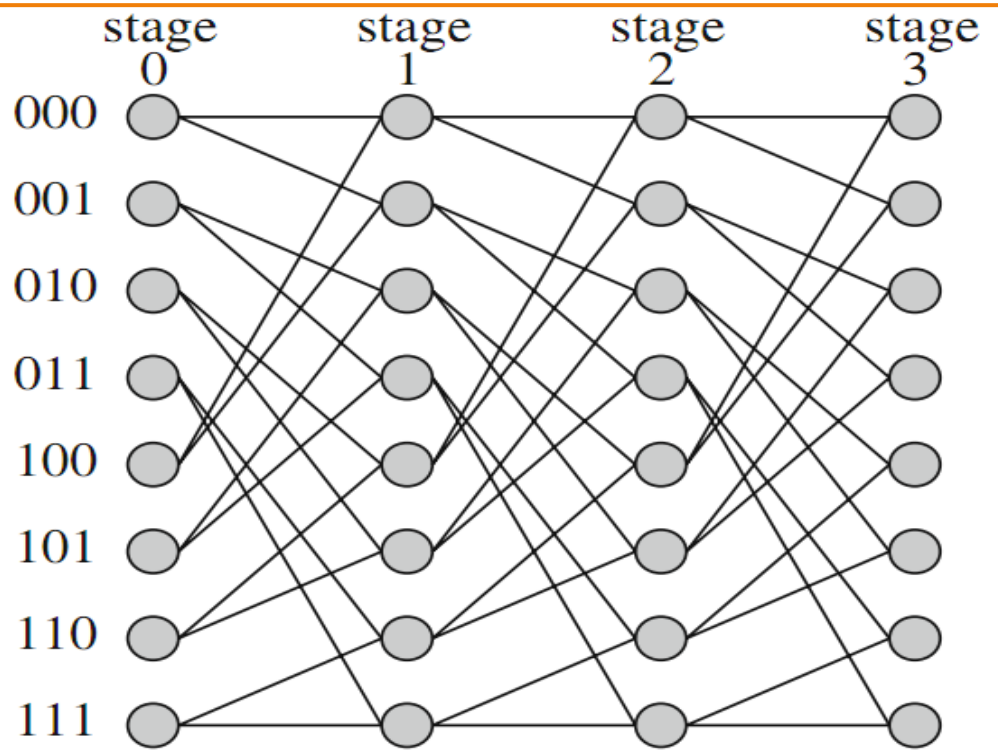


Multistage Networks using Regular 2x2 Crossbar Switches

2x2 Switch Settings

Omega Network

- One unique path for every input to output

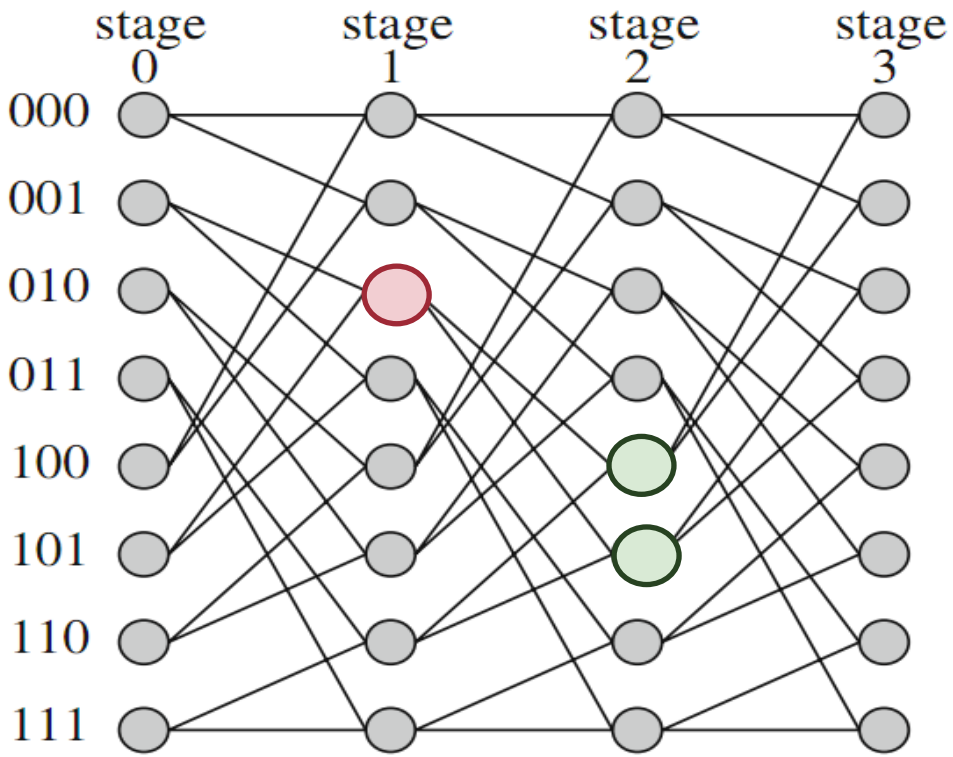


16 × 16 Omega Network

- An n × n Omega network has log n stages
  - n/2 switches per stage
  - Connections between stages are regular
  - Also known as (lg n – 1) – dimension Omega Network

Omega Network - Construction

- A switch position: (α, i)
  - α: position of a switch within a stage; i: stage number
- Has an edge from node (α, i) to two nodes (β, i + 1) where
  - β = α by a cyclic left shift
  - β = α by a cyclic left shift + inversion of the LSBit



Example:

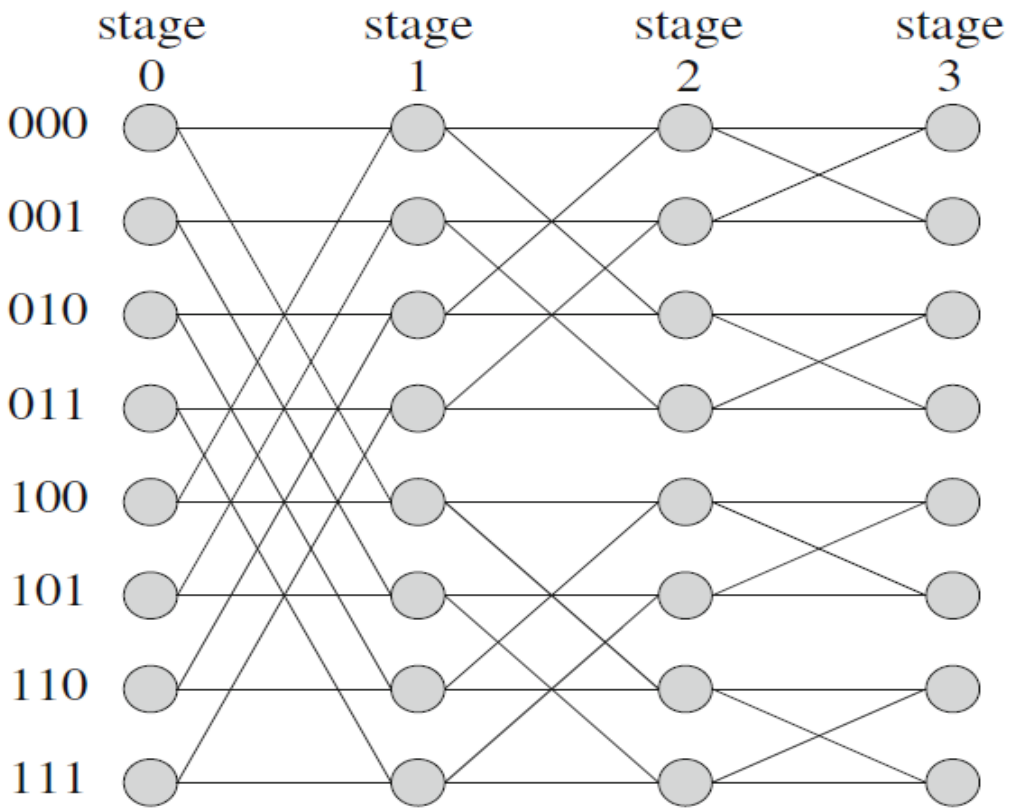
(010, 1)
➔ **(100, 2)** and
➔ **(101, 2)**

16 × 16 Omega Network using 2x2 switches

Omega Network vs Crossbar Switches

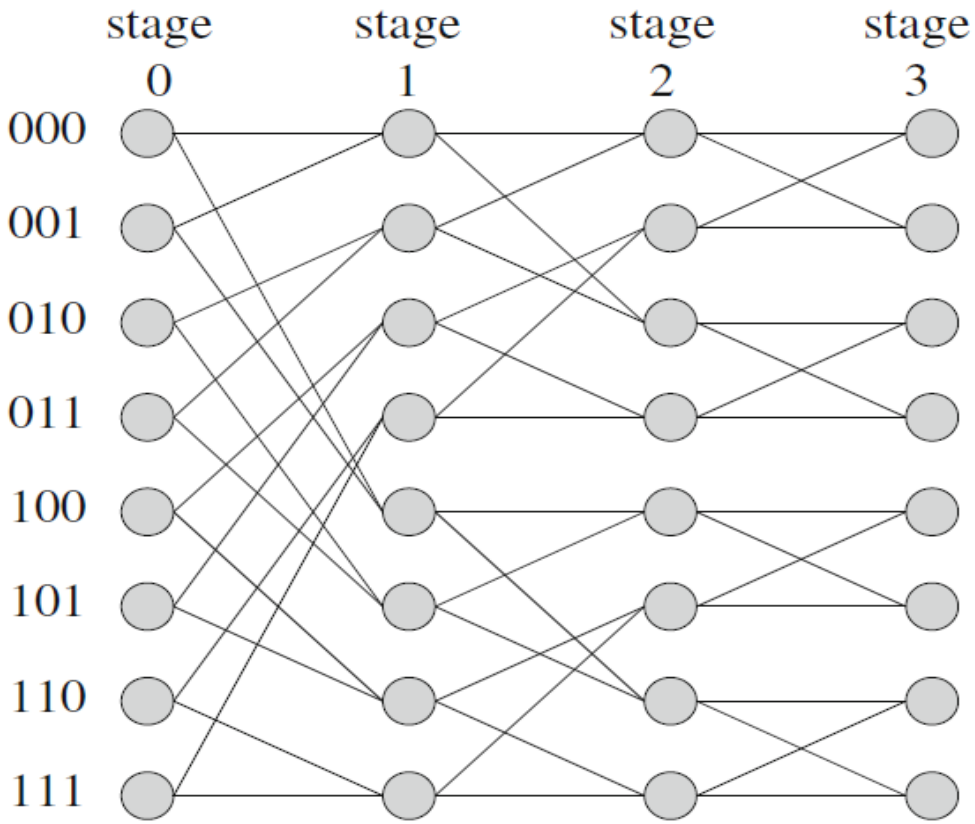
- To connect 16 processor nodes to 16 memory nodes
- Crossbar = 16 x 16 = 256 switches
- Omega: n=16 and using 2x2 switches
  - Total number of switches = **n/2** switches per stage x **log n** stages
  - 32 switches

Butterfly Network



- Node **(α, i)** connects to
  1. **(α, i+1)**, i.e. straight edge
  2. **(*α*', i+1)**, ***α*** and ***α'*** differ in the *(i* + 1)th bit from the left, i.e. cross edge

Baseline Network



- Node **(α, i)** to **two** nodes **(β, i + 1)** where
  1. β = cyclic right shift of last (k-i) bits of α
  2. β = inversion of the LSBit of α + cyclic right shift of last (k-i) bits

Many others

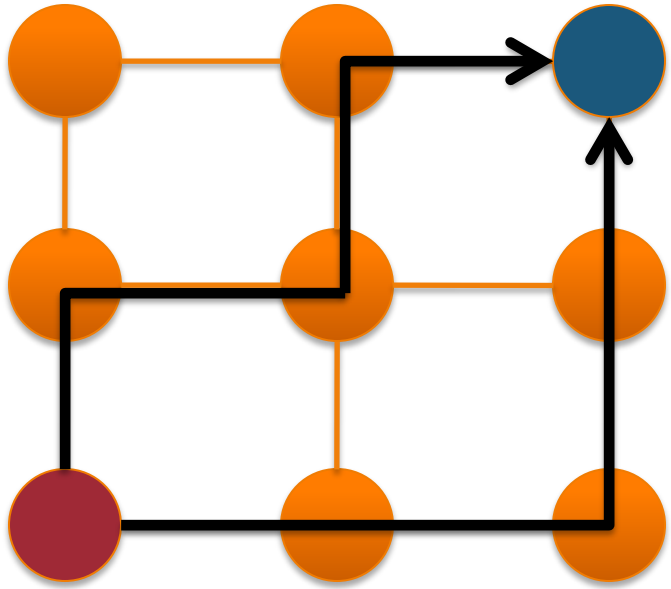
- Beneš network
- Clos network
- Fat-Tree
- Folded Butterfly
- etc…..

To go from Point A to Point B

ROUTING

Routing Overview

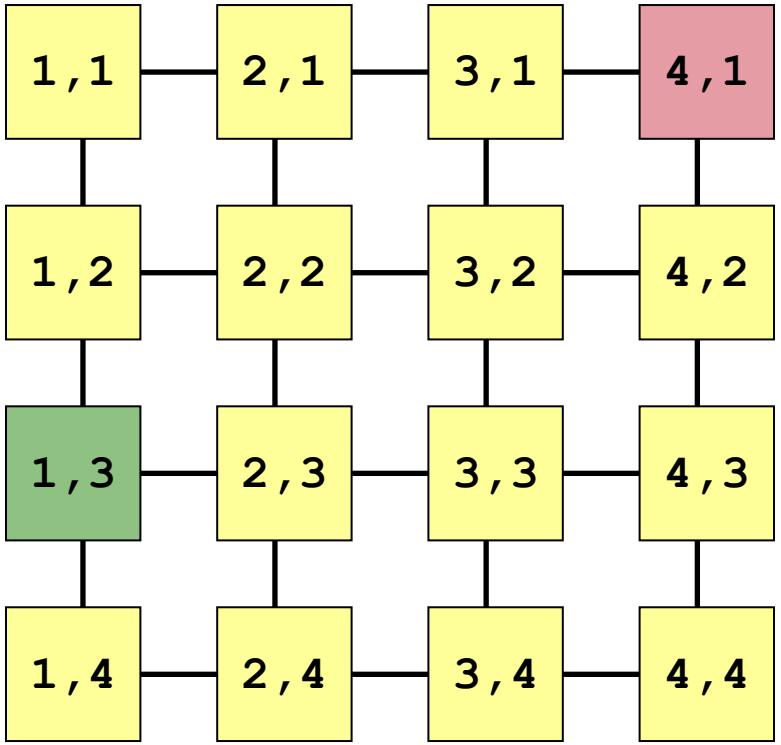
- Routing algorithm determines path(s) from source to destination
  - Within a given interconnection topology



Routing Algorithms Classification

- Based on **path length**:
  - **Minimal** or **Non-minimal** routing: whether the **shortest path** is always chosen
- Based on **adaptivity:**
  - **Deterministic**: Always use the same path for the same pair of (source, destination) node
  - **Adaptive:** May take into account of network status and adapt accordingly, e.g. avoid congested path, avoid dead nodes etc
- We look at three deterministic examples

XY Routing for 2D Mesh

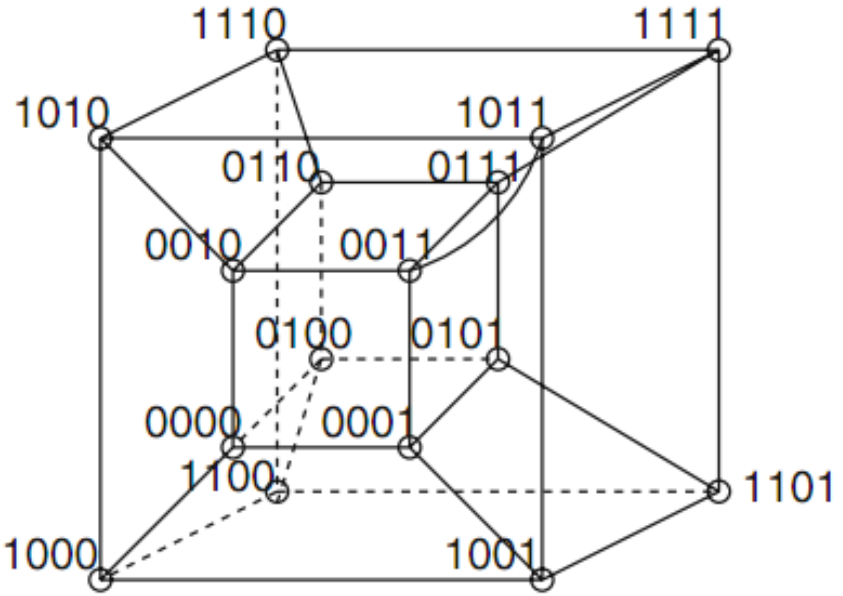


- (X_{src}, Y_{src}) to (X_{dst}, Y_{dst}) :
  - Move in X direction until X_{src} == X_{dst}
  - Move in Y direction until Y_{src} == Y_{dst}

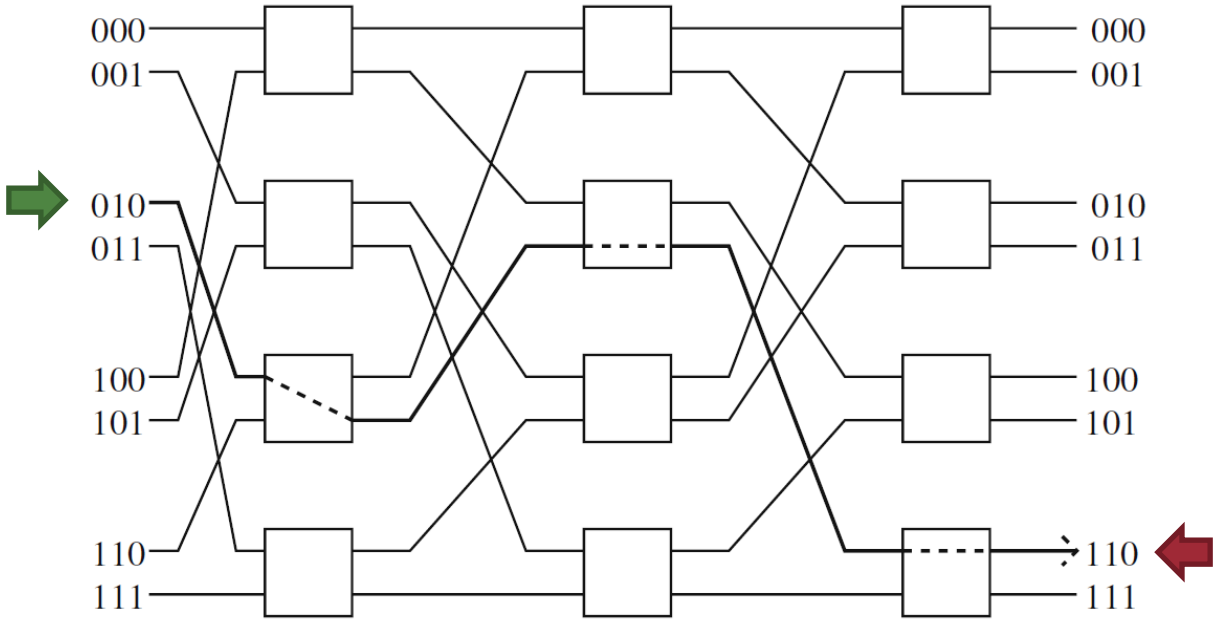
E-Cube Routing for Hypercube

- Let ($\alpha_{n-1}\ \alpha_{n-2\ldots}\ \alpha_1\ \alpha_0$) and ($\beta_{n-1}\beta_{n-2\ldots}\beta_1\beta_0$) be the bit representations of source and destination node address respectively:
  - Number of bits difference in source and target node address ➔ number of hops
    - Also known as ***hamming distance***
  - Start from MSB to LSB (or LSB to MSB)
    - Find the first different bit
    - Go to the neighboring node with the bit corrected

    ➔ **At most n hops**

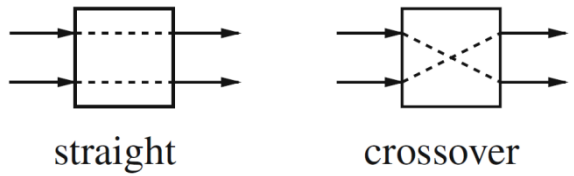


XOR-Tag Routing for Omega Network



8×8 Omega Network using 2x2 switches

- Let T = Source Id⊕ Destination Id
- At stage-**k:**
  - Go straight if bit k of T is 0
  - Crossover if bit k of T is 1



2x2 switches Settings

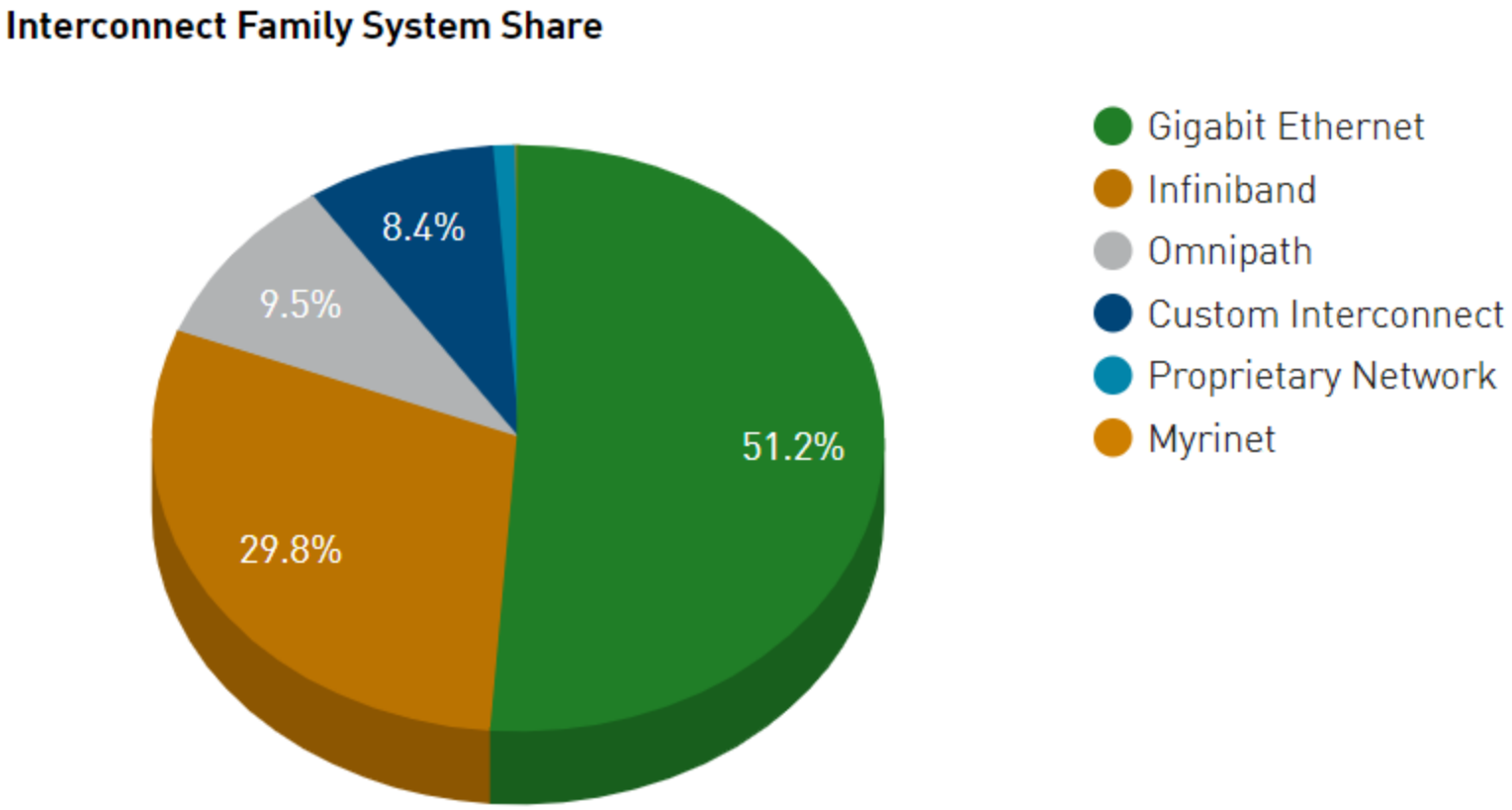
For your exploration

- Routing:
  - Adaptive Routing Algorithm
  - Deadlock and deadlock avoidance in routing
- Switching:
  - Protocols
  - Splitting and reconstruction of message
- Flow Control:
  - Mechanisms to avoid network congestion

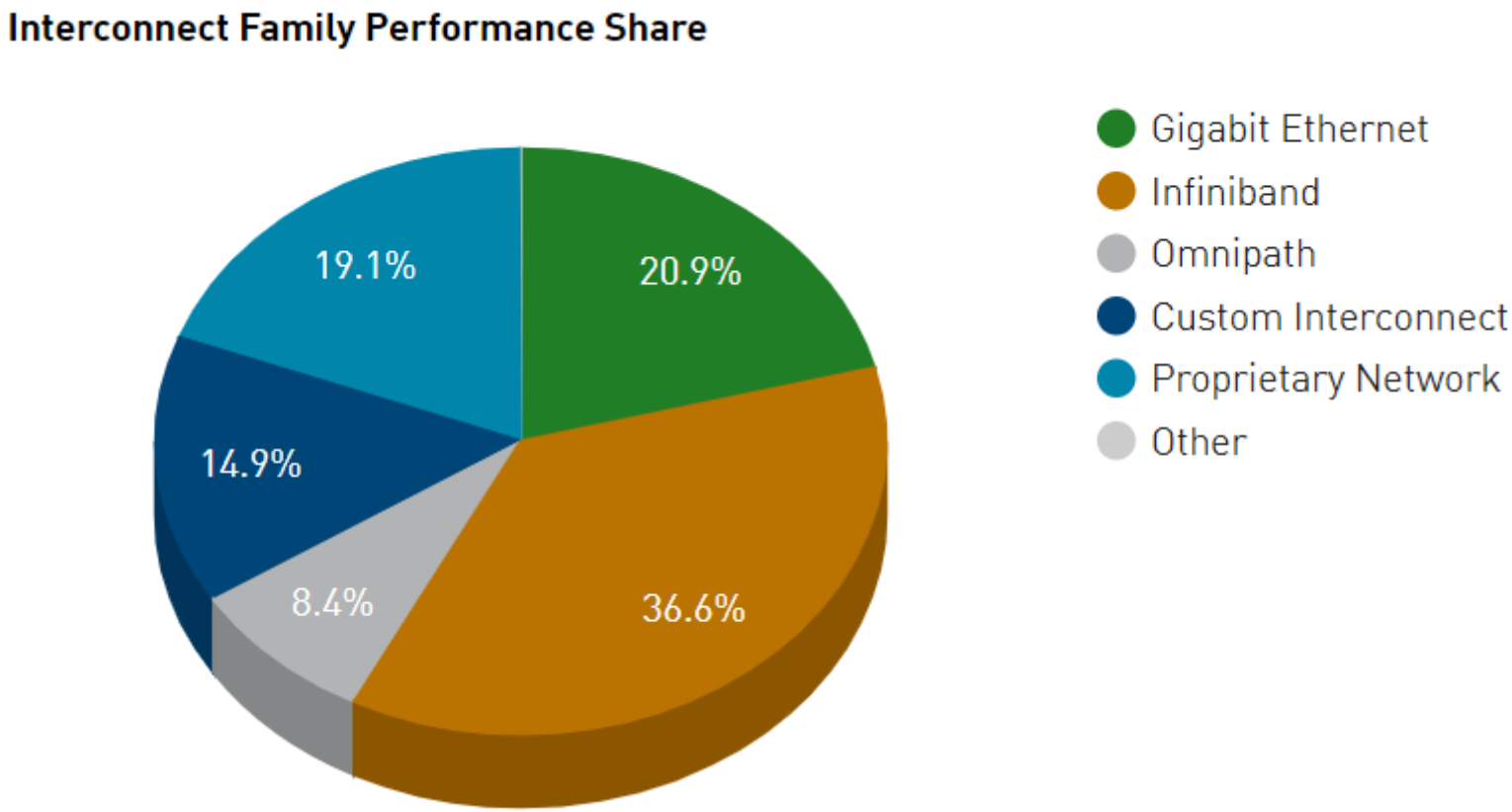
So, what's the "hotness" now?

CURRENT TREND

Top 500 Supercomputers: Interconnect



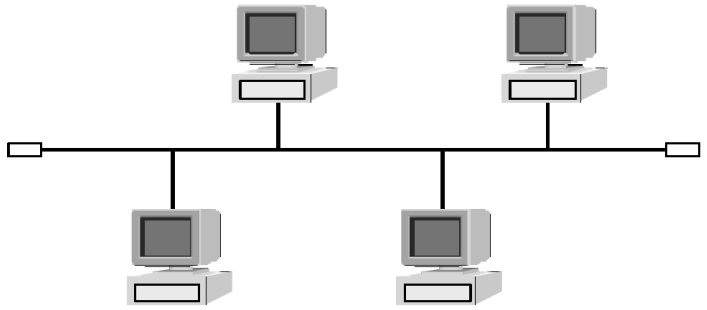
Interconnect Performance Share



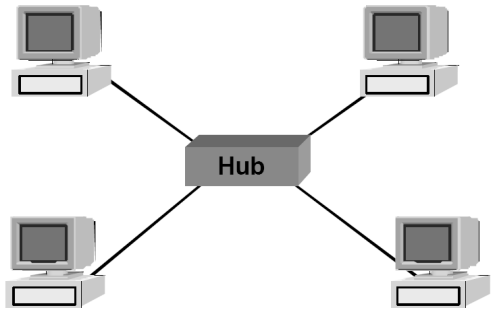
Ethernet

- Support *point-to-point* and *broadcast*

- Commonly-used topologies: physical bus, physical star with a logical bus, etc



Physical bus topology

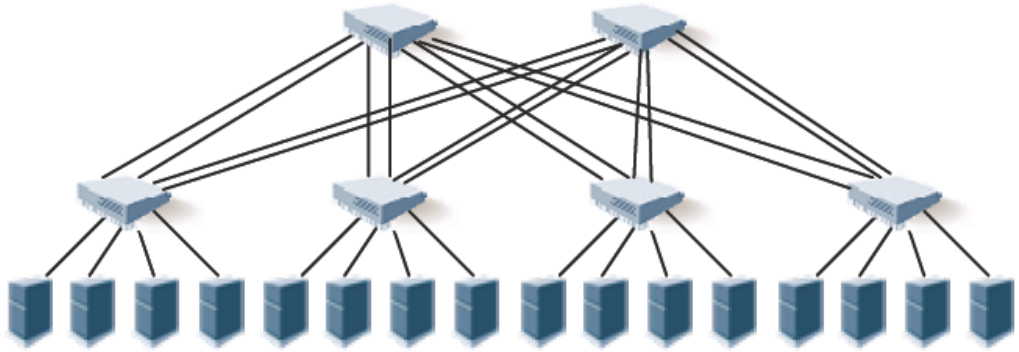


Physical star/logical bus topology

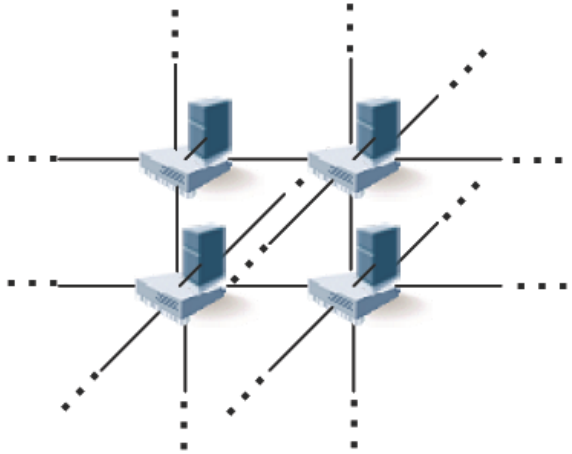
InfiniBand

- Support *point-to-point* and *multicast* (on top of point-to-point capabilities)

- Commonly-use topologies: Fat tree , torus, etc.

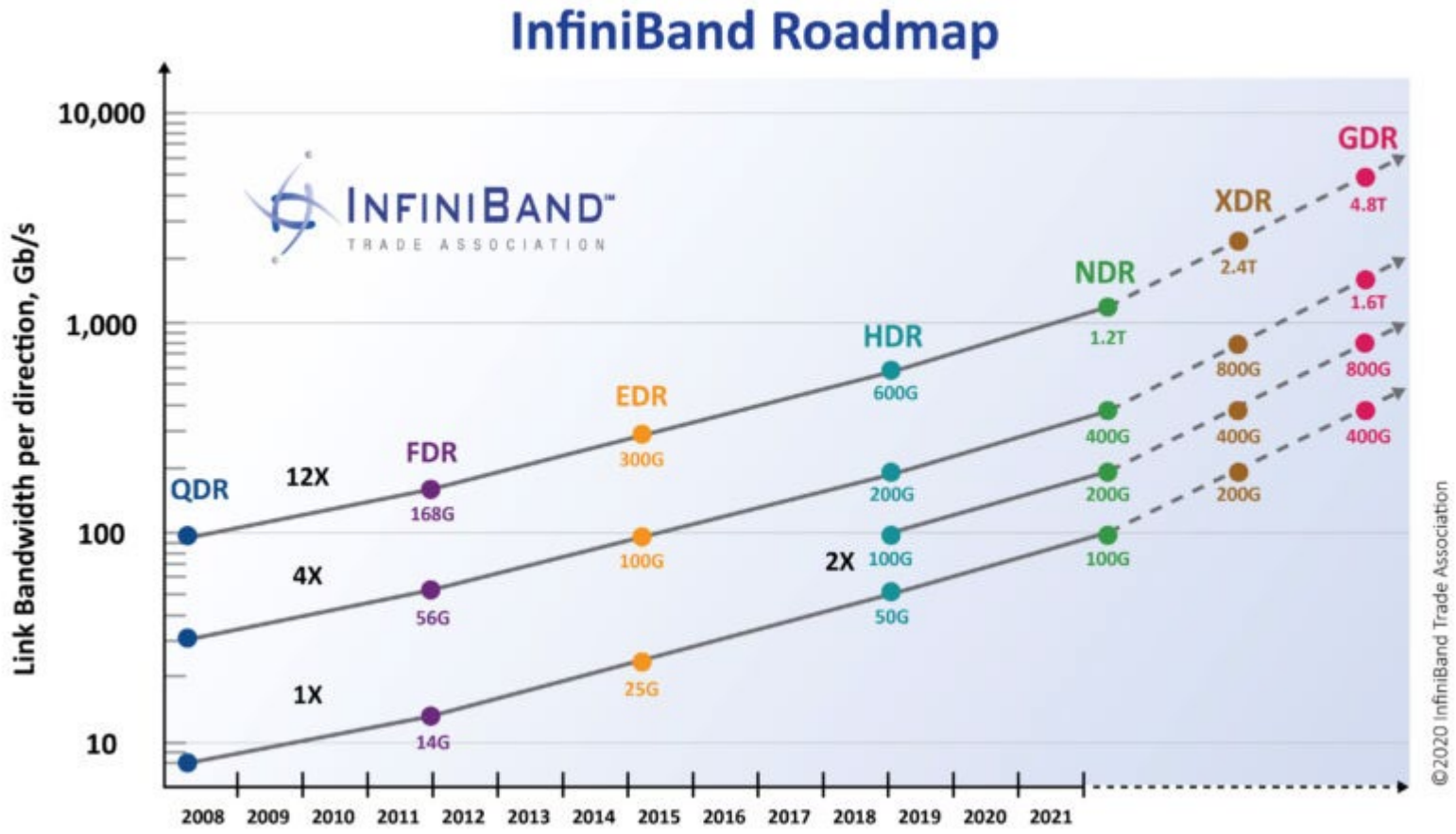


Two level fat tree



3D torus

InfiniBand Roadmap



Summary

- Interconnection networks
  - Design criteria and properties of interconnection networks
  - Direct and indirect interconnection networks

- References:
  - Infiniband Overview:
    - https://www.infinibandta.org/ibta-specification/