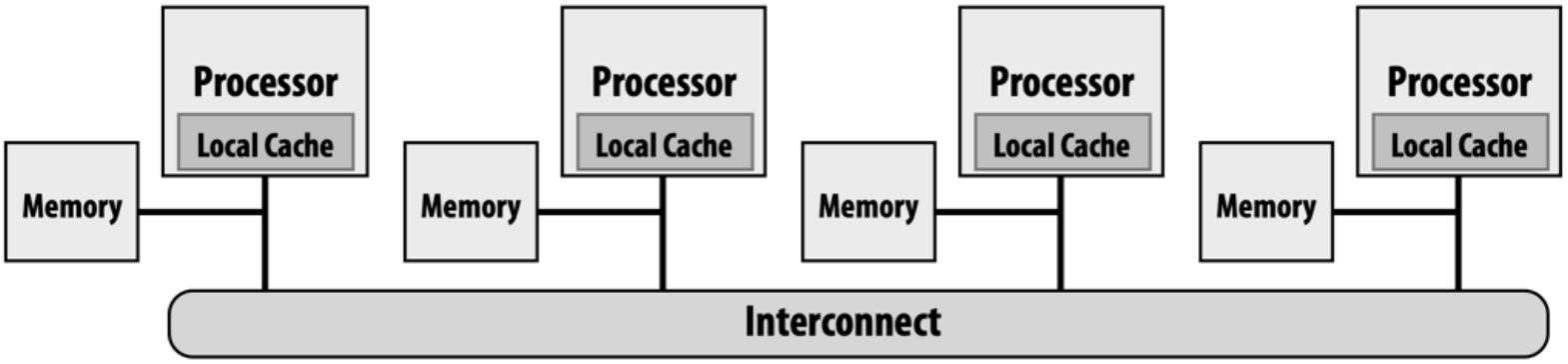
Interconnection Network

Interconnection Networks[互连网络]

- **An Interconnection Network (ICN) is a programmable system that transports data between terminals**
  - To hold our parallel machines together
    - at the core of parallel computer architecture
  - Share basic concept with LAN/WAN
  - but very different trade-offs due to very different time scale/requirements



Interconnection Networks[互连网络]

- **Interconnection networks can be grouped into four domains**
  - Depending on number and proximity of devices to be
    - **Wide-Area Networks**[广域网]
    - **Local-Area Networks**[局域网]
    - **System-Area Networks**[系统区域网络]
    - **On-Chip Networks**[片上网络]

Wide-Area Networks

- **Interconnect systems distributed across the globe**
  - Internetworking support is required
  - Millions of devices interconnected
- **Maximum interconnect distance**
  - many thousands of kilometers

Local-Area Networks

- **Interconnect autonomous computer systems**
  - Machine room or throughout a building or campus
  - Hundreds of devices interconnected
- **Maximum interconnect distance**
  - Few meters to tens of kilometers
  - Example (most popular):
    - Ethernet, with 10 Gbps over 40Km

System-Area Networks

- **Interconnects within one "machine"**
  - Interconnect in a multi-processor system
  - Interconnect in a supercomputer
  - Hundreds to thousands of devices interconnected
    - Tianhe-2 supercomputer (16K nodes)
- **Maximum interconnect distance**
  - Fraction to tens of meters (typical)
  - A few hundred meters (some)
    - InfiniBand: 120 Gbps over a distance of 300m

System-Area Networks

Perf	54.9PFlops / 33.86PFlops
Nodes	16000
Mem	1.4PB
Racks	125+8+13+24=170 (720m^2)
Power	17.8 MW (1.9GFlops/W)
Cool	Close-coupled chilled water cooling

国防科学技术大学
National University of Defense Technology

On-Chip Networks

- **Interconnect within a single chip**
  - Devices are micro-architectural elements
    - Caches, directories, processor cores
  - Currently, designs with tens of devices are common
    - Ex: IBM Cell, Intel multicores, Tile processors
- **Proximity: millimeters**

We are concerned with On-Chip and System-Area Networks

Why Study Interconnects?

- **Provide external connectivity from system to outside world**
  - Also, connectivity within a single computer system at many levels
- **Interconnection networks should be well designed**
  - To transfer the maximum amount of information
  - Within the least amount of time (and cost, power constraints)
- **Application:**
  - managing communication can be critical to performance

Why Study Interconnects? (cont.)

- **Trends:** high demand on communication bandwidth
  - increased computing power and storage capacity
  - switched networks are replacing buses

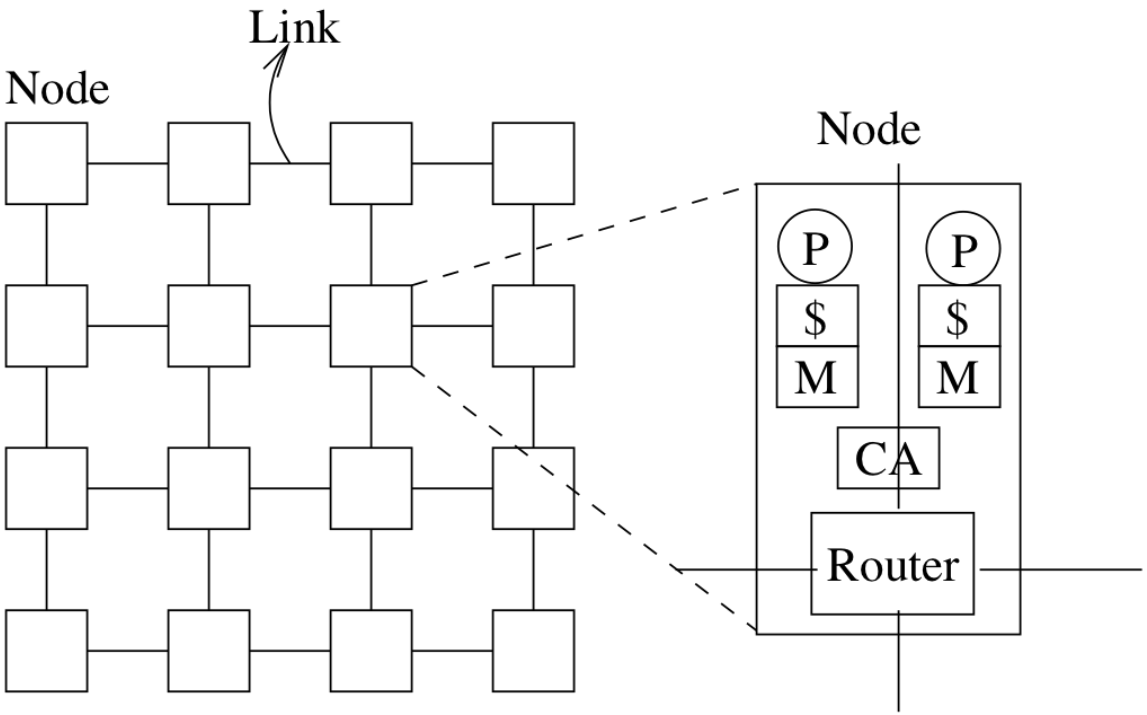
- **Computer architects/engineers must understand interconnect problems and solutions**
  - in order to effectively design and evaluate systems

Interconnection network

- **Characteristics of communication**
  - Latency must be low
  - Bandwidth must be high
  - Message characteristics
    - Many small messages with several fixed sizes
    - The largest message is one cache block size (64 or 128 bytes)

- **Implications**
  - Only two layers are sufficient: link level and node level
  - Communication protocol must be simple
    - no packet dropping, no elaborate flow control

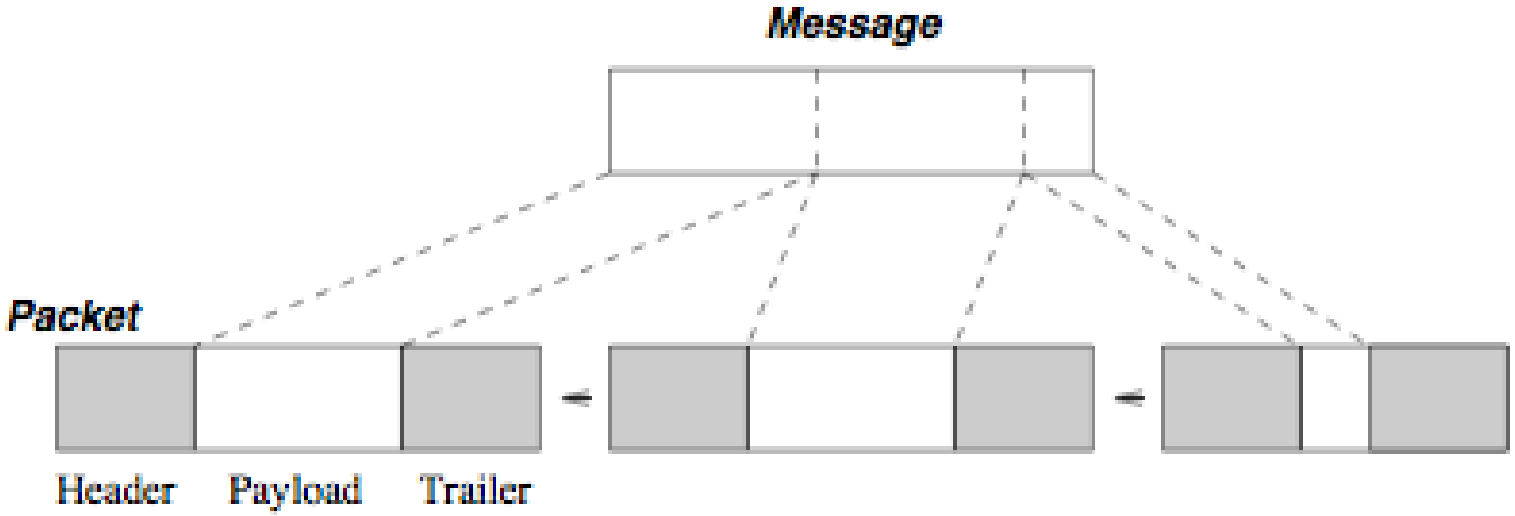
Nodes and Links



- **A node encapsulates**
  - one or more processors
  - communication controller/assist (CA)
  - router
  - Routers are connected by links
  - Links may be unidirectional or bidirectional

Link and Channel

- **Link = set of wires interconnecting a pair of nodes**
  - Link can be unidirectional or bidirectional
  - Unit of transfer determined by flow control protocol (flit)
  - Flit size determined by link latency and amount of buffering available
- **Channel = link + sender + receiver**
- A message transmitted over a channel is broken into **packets**
  - A packet has a header, trailer, and payload



Design Considerations

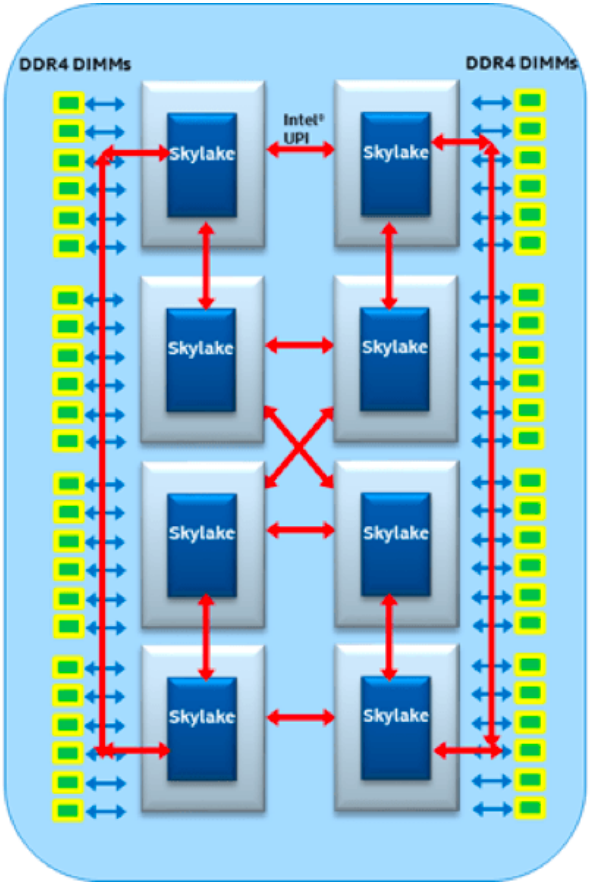
- **Application requirements**
  - Number of terminals or ports to support
  - Peak bandwidth of each terminal
  - Average bandwidth of each terminal
  - Latency requirements
  - Message size distribution
  - Expected traffic patterns
  - Required quality of service
  - Required reliability and availability

Design Considerations

- **Job of an interconnection network:**
  - to transfer information from source node to dest. node
  - support network transactions that realize application
- **Latency as small as possible**
  - As many concurrent transfers as possible
- **Cost as low as possible**

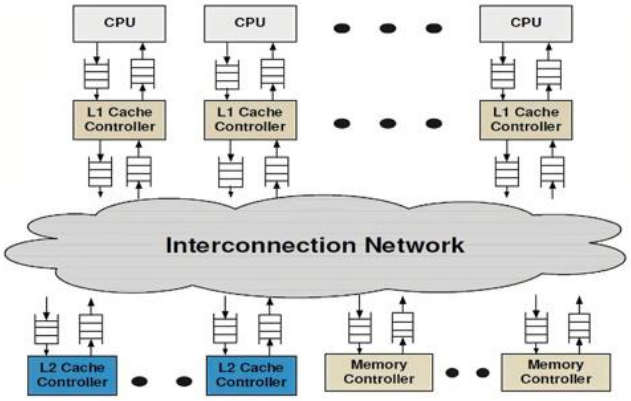
Example Requirements

- **Example requirements for a coherent processor-memory interconnect**
  - Processor ports 1-2048
  - Memory ports 1-4096
  - Peak BW 8 GB/s
  - Average BW 400 MB/s
  - Message Latency 100 ns
  - Message size 64 or 576 bits
  - Traffic pattern arbitrary
  - Quality of service none
  - Reliability no message loss
  - Availability 0.999 to 0.99999



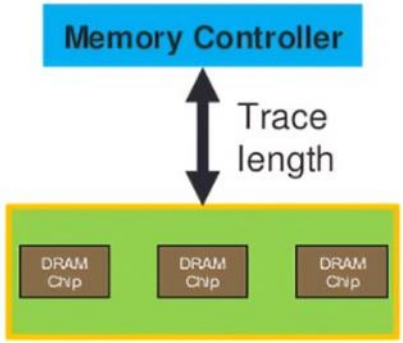
Technology constraints

- **Technology constraints**
  - Signaling rate
  - Chip pin count (if off-chip networking)
  - Area constraints (typically for on-chip networking)
  - Chip cost
  - Circuit board cost (if backplane boards needed)
  - Signals per circuit board
  - Signals per cable
  - Cable cost
  - Cable length
  - Channel and switch power constraints

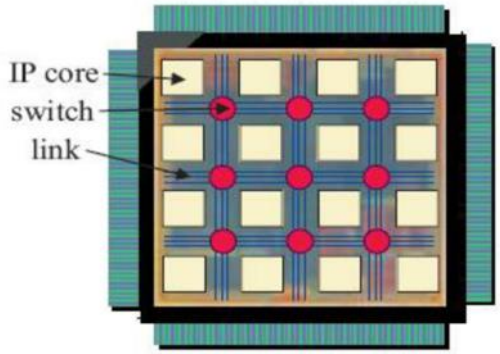


Off-chip vs. On-chip ICNs

- **Off-chip: I/O bottlenecks**
  - Pin-limited bandwidth
  - Inherent overheads of off-chip I/O transmission



- **On-chip**
  - Wiring constraints
    - Horizontal and vertical layout
    - Short, fixed length
  - Power
    - Consume 10-15% or more of die power budget
  - Latency
    - Different order of magnitude



Main Aspects of an ICN

- **Topology**
  - Static arrangement of channels and nodes in a network
- **Routing**
  - Determines the set of paths a packet can follow
- **Flow control[流控]**
  - Allocating network resources (channels, buffers, etc.) to packets and managing contention

Main Aspects of an ICN

- **Switch microarchitecture**
  - Internal architecture of a network switch
- **Network interface**
  - How to interface a terminal with a switch
- **Link architecture**
  - Signaling technology and data representation on the channel

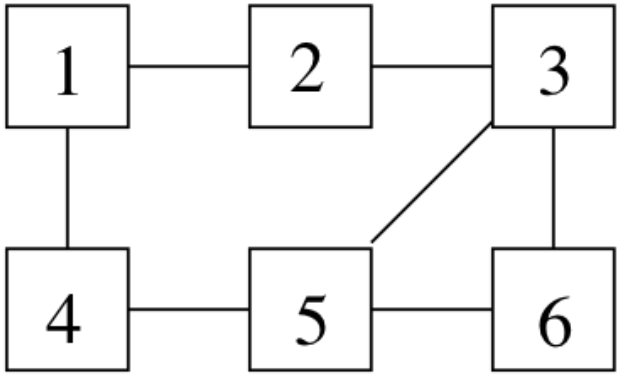
1. Network Topology

Network Topology

- **Network topology = shape of the network**
  - Determines the distance that a message needs to travel (in # links or network hops) from one node to another
- **Important characteristics**
  - **Diameter**: largest distance
  - **Average distance**
  - **Degree**: how many links are connected to a switch

Example

- **For the shown network, compute diameter, avg distance, bisection bandwidth (assuming each link is bidirectional)**



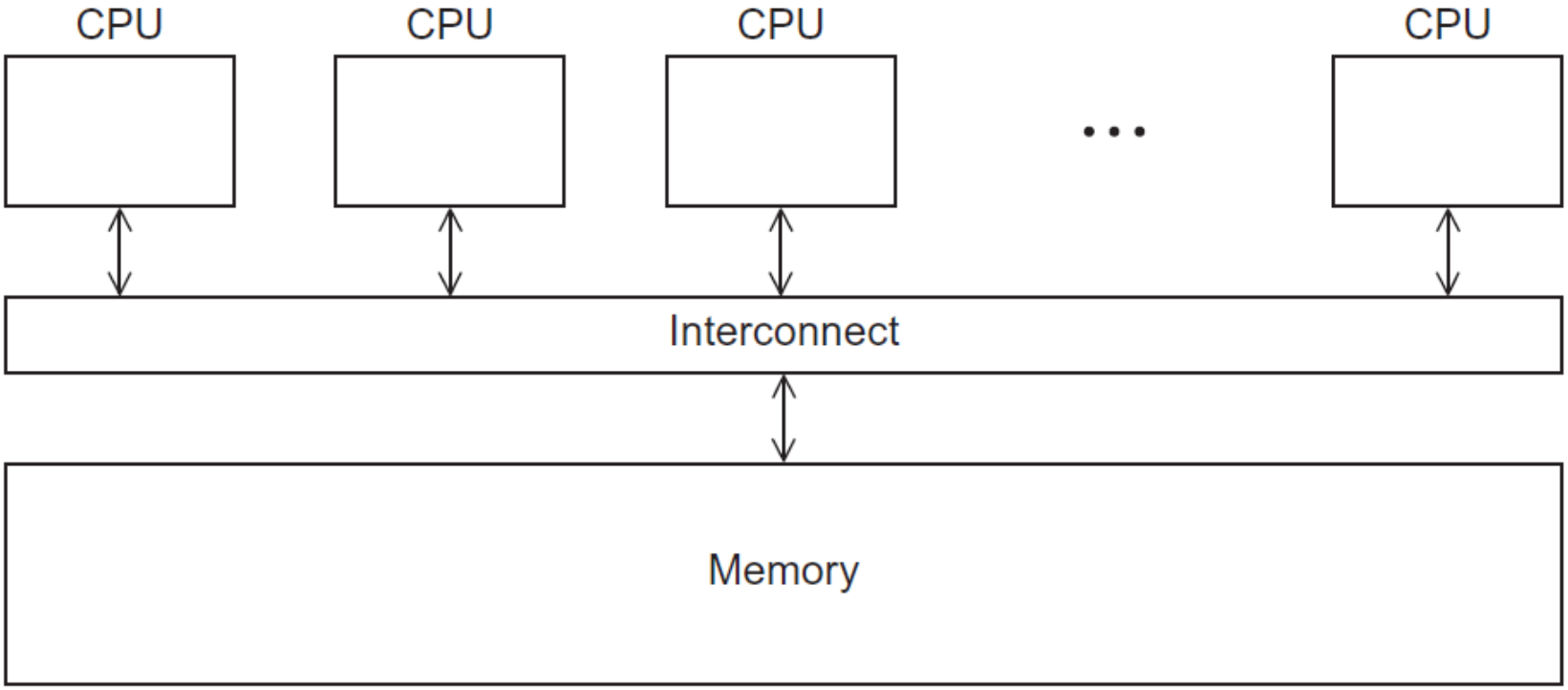
Diameter = 3

e.g. the distance between node 1 and 6

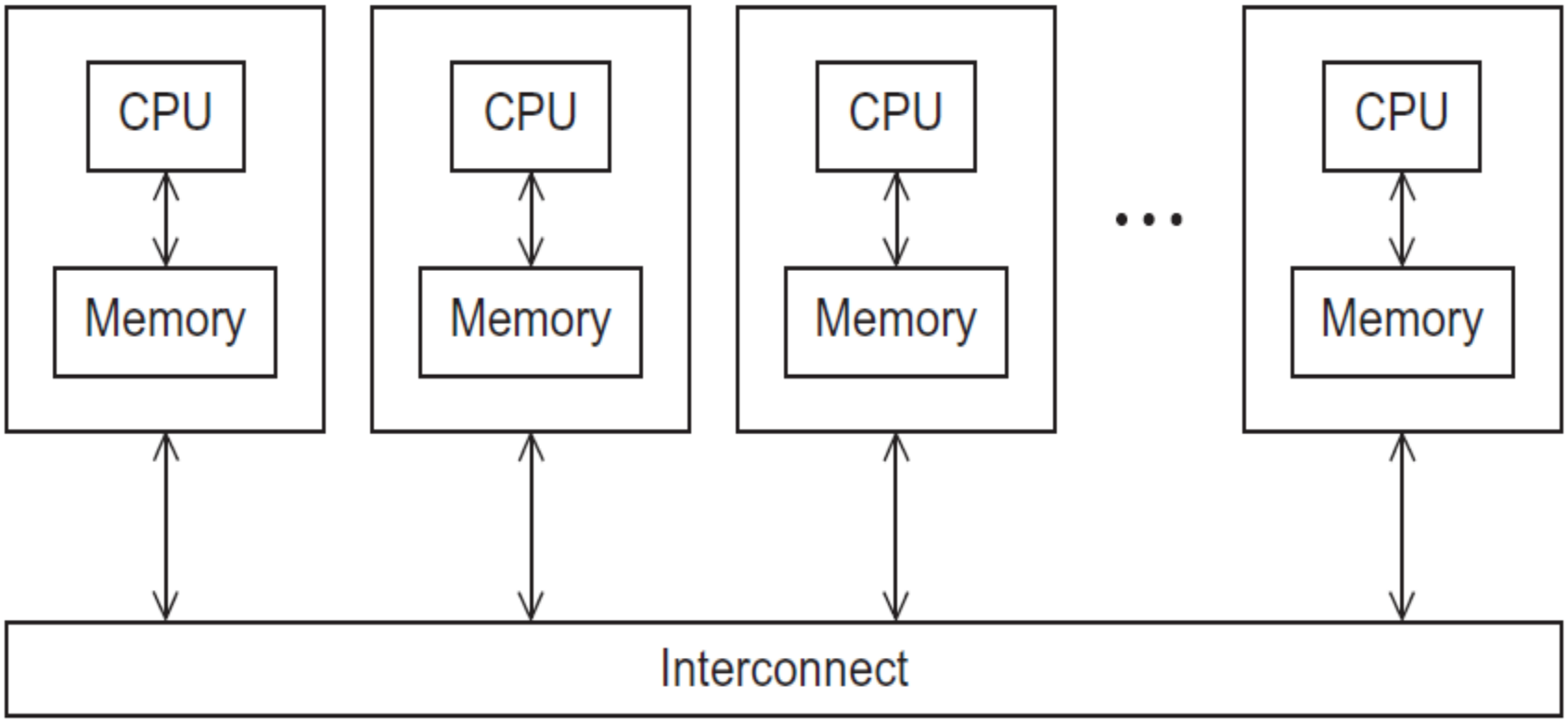
Avg distance:

d(1,2)=1, d(1,3)=2, d(1,4)=1, d(1,5)=2, d(1,6)=3
d(2,3)=1, d(2,4)=2, d(2,5)=2, d(2,6)=2
d(3,4)=2, d(3,5)=1, d(3,6)=1
d(4,5)=1, d(4,6)=2
d(5,6)=1
Sum of all distances = 24, number of node pairs = 15
Average distance = 24/15

Recall: Shared Memory System



Recall: Distributed Memory System



Interconnection networks

- Affects performance of both distributed and shared memory systems.

- Two categories:
  - Shared memory interconnects
  - Distributed memory interconnects

1.1 Shared memory interconnects

Shared memory interconnects

- **Bus interconnect**
  - A collection of parallel communication wires together with some hardware that controls access to the bus.
  - Communication wires are shared by the devices that are connected to it.
  - As the number of devices connected to the bus increases, contention for use of the bus increases, and performance decreases.

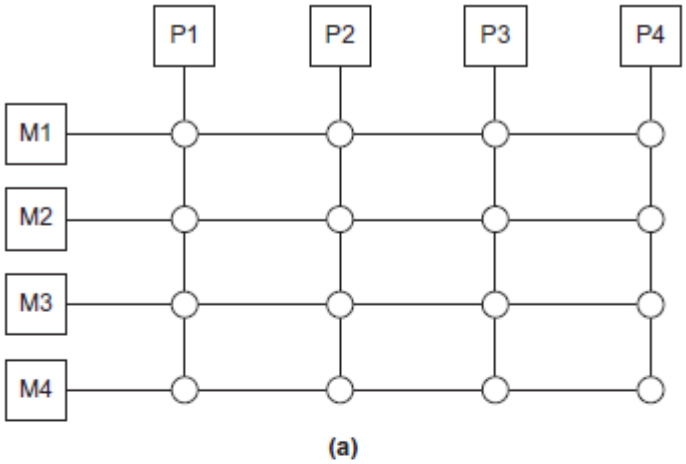
Shared memory interconnects

- **Switched interconnect**
  - Uses switches to control the routing of data among the connected devices.

  - **Crossbar** –
    - Allows simultaneous communication among different devices.
    - Faster than buses.
    - But the cost of the switches and links is relatively high.

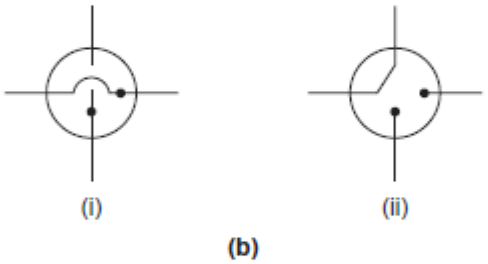
(a)

A crossbar switch connecting 4 processors (P_i) and 4 memory modules (M_j)

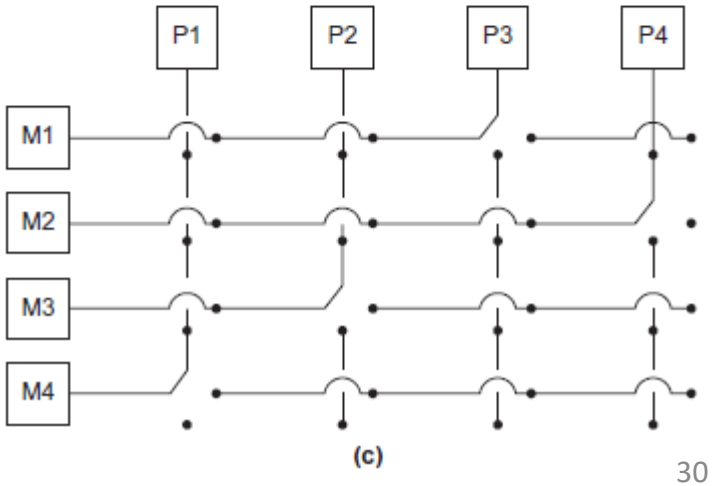


(b)

Configuration of internal switches in a crossbar



(c) Simultaneous memory accesses by the processors



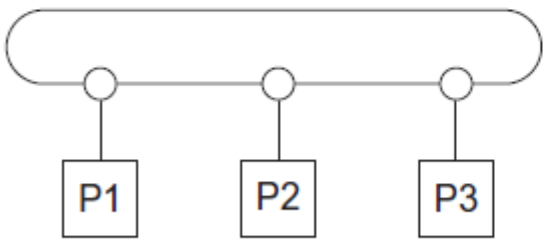
1.2 Distributed memory interconnects

Distributed memory interconnects

- **Two groups**
  - **Direct interconnect**
    - Each switch is directly connected to a processor memory pair, and the switches are connected to each other.
  - **Indirect interconnect**
    - Switches may not be directly connected to a processor.

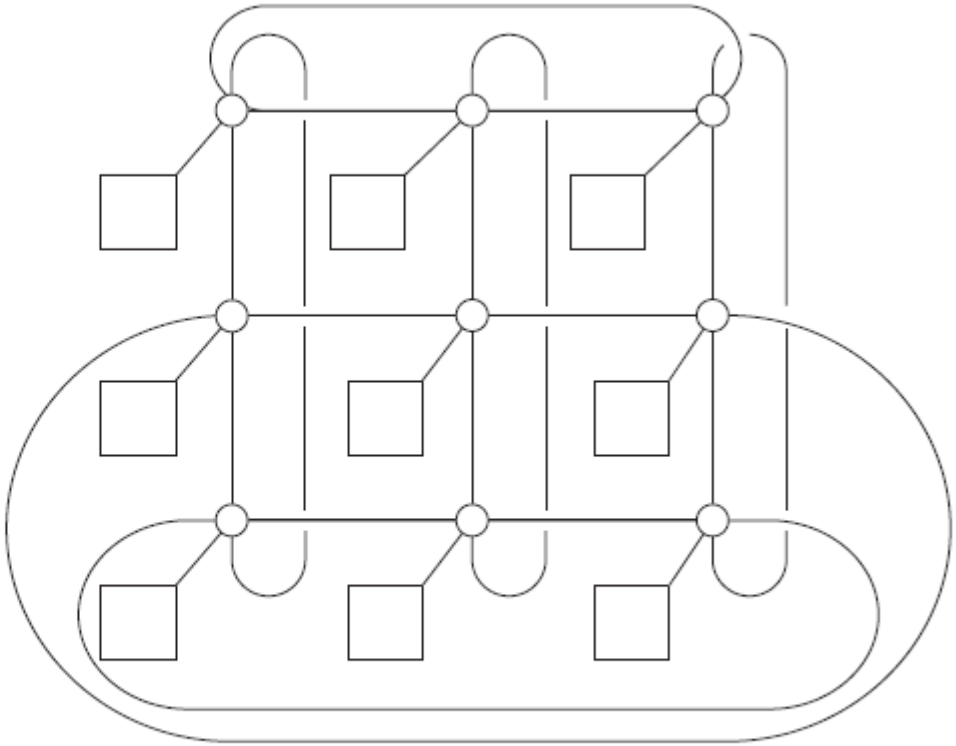
(1) Direct interconnect

Direct interconnect



(a)

ring



(b)

toroidal mesh

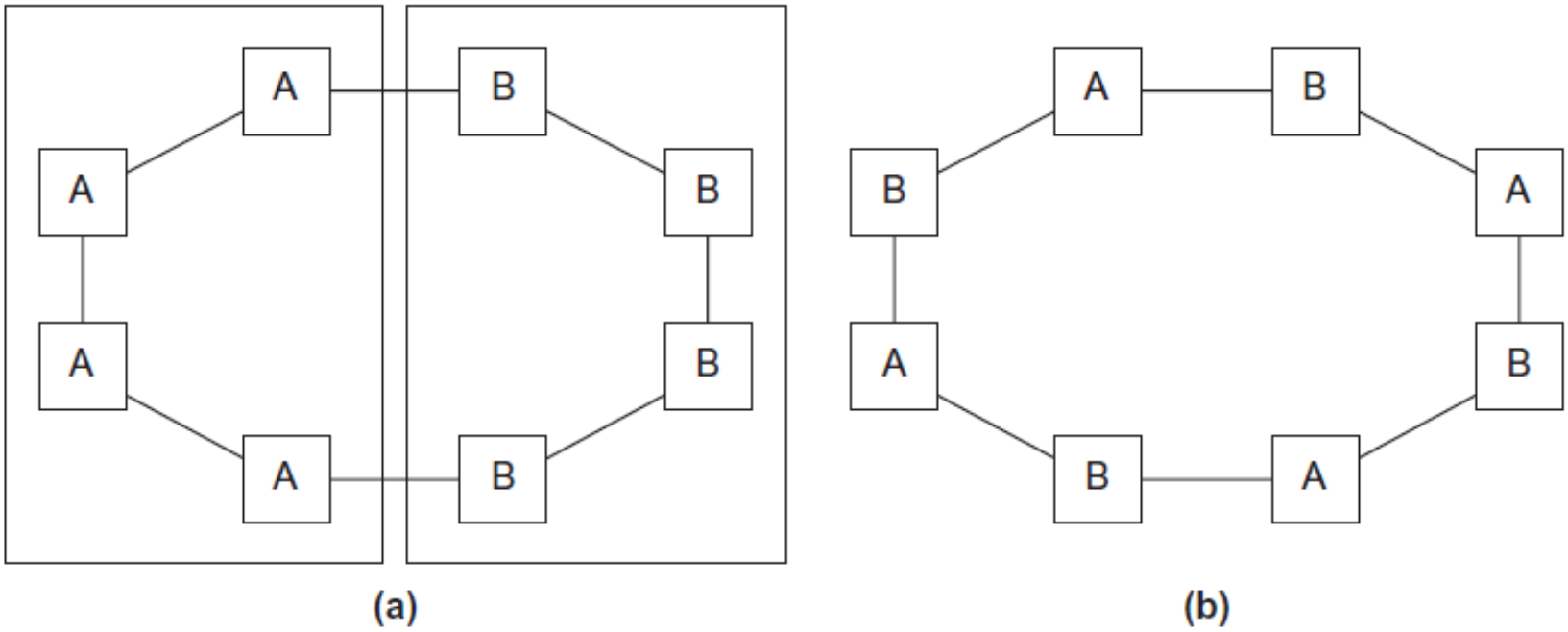
（二维环面网格）

Bisection width（等分宽度）

- A measure of “number of simultaneous communications” or “connectivity”.

- How many simultaneous communications can take place “across the divide” between the halves?

Two bisections of a ring



For toroidal mesh

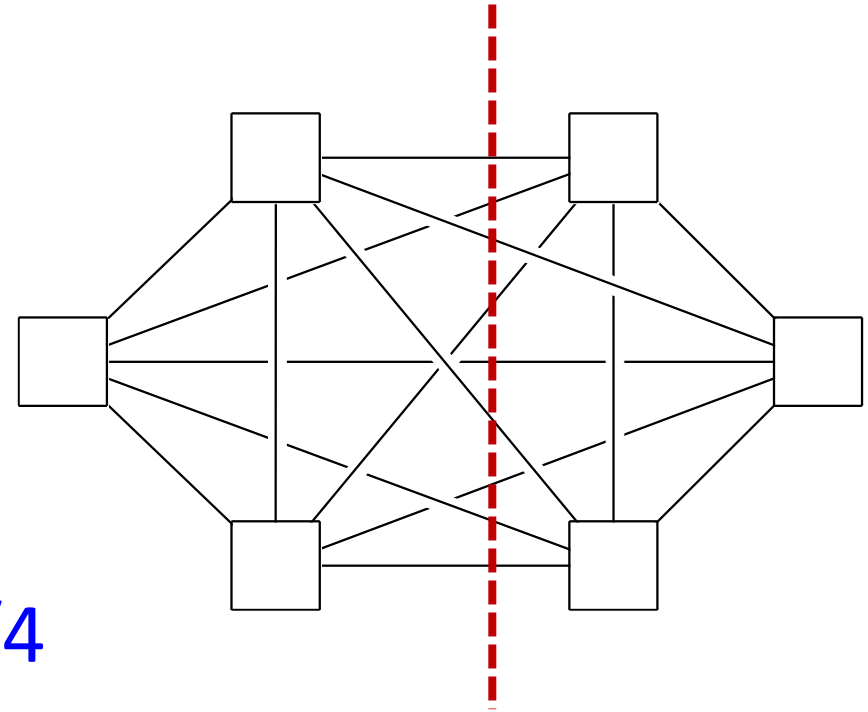
Definitions

- Bandwidth
  - The rate at which a link can transmit data.
    - Usually given in megabits or megabytes per second.
- Bisection bandwidth
  - A measure of network quality.
  - Instead of counting the number of links joining the halves, it sums the bandwidth of the links.

Fully connected network

- Each switch is directly connected to every other switch.

impractical



bisection width = $p^2/4$

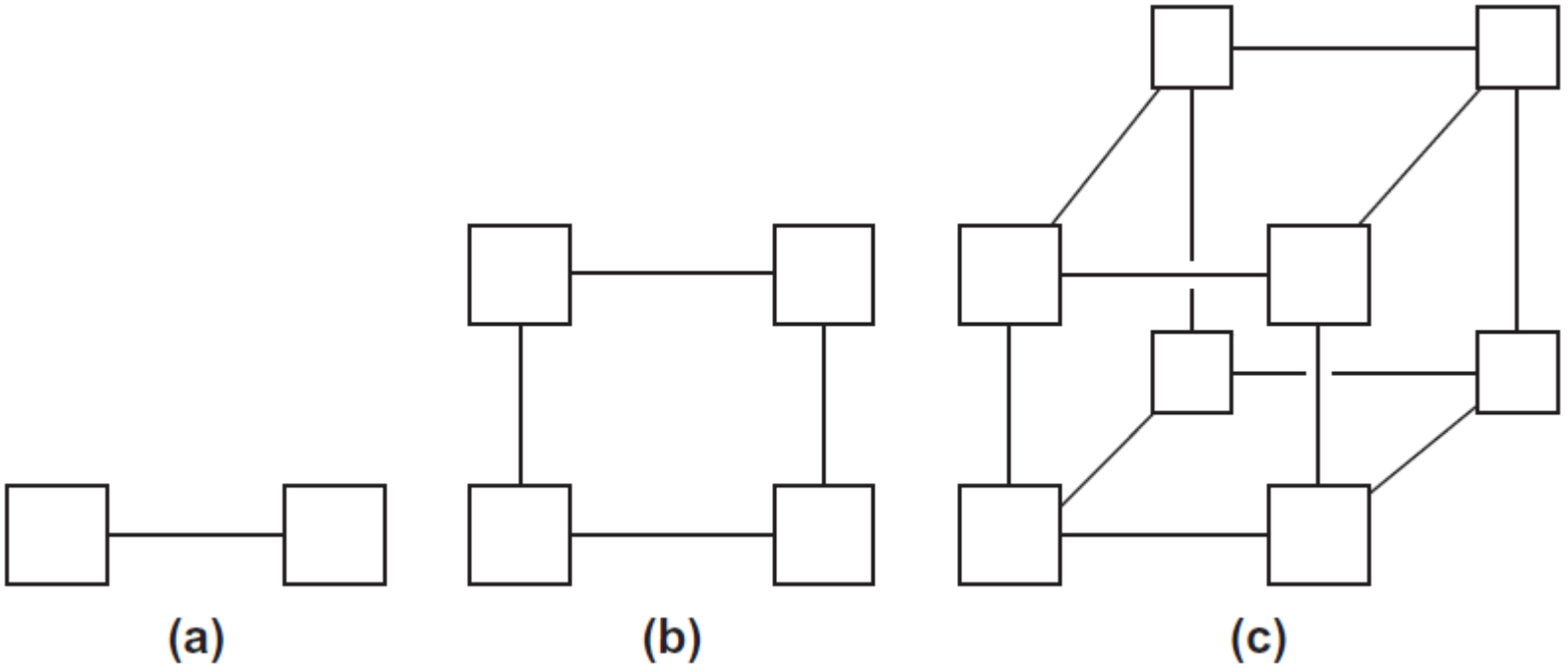
(p: # of nodes)

Hypercube

- Highly connected direct interconnect.

- **Built inductively:**
  - A one-dimensional hypercube is a fully-connected system with two processors.
  - A two-dimensional hypercube is built from two one-dimensional hypercubes by joining "corresponding" switches.
  - Similarly a three-dimensional hypercube is built from two two-dimensional hypercubes.

Hypercubes



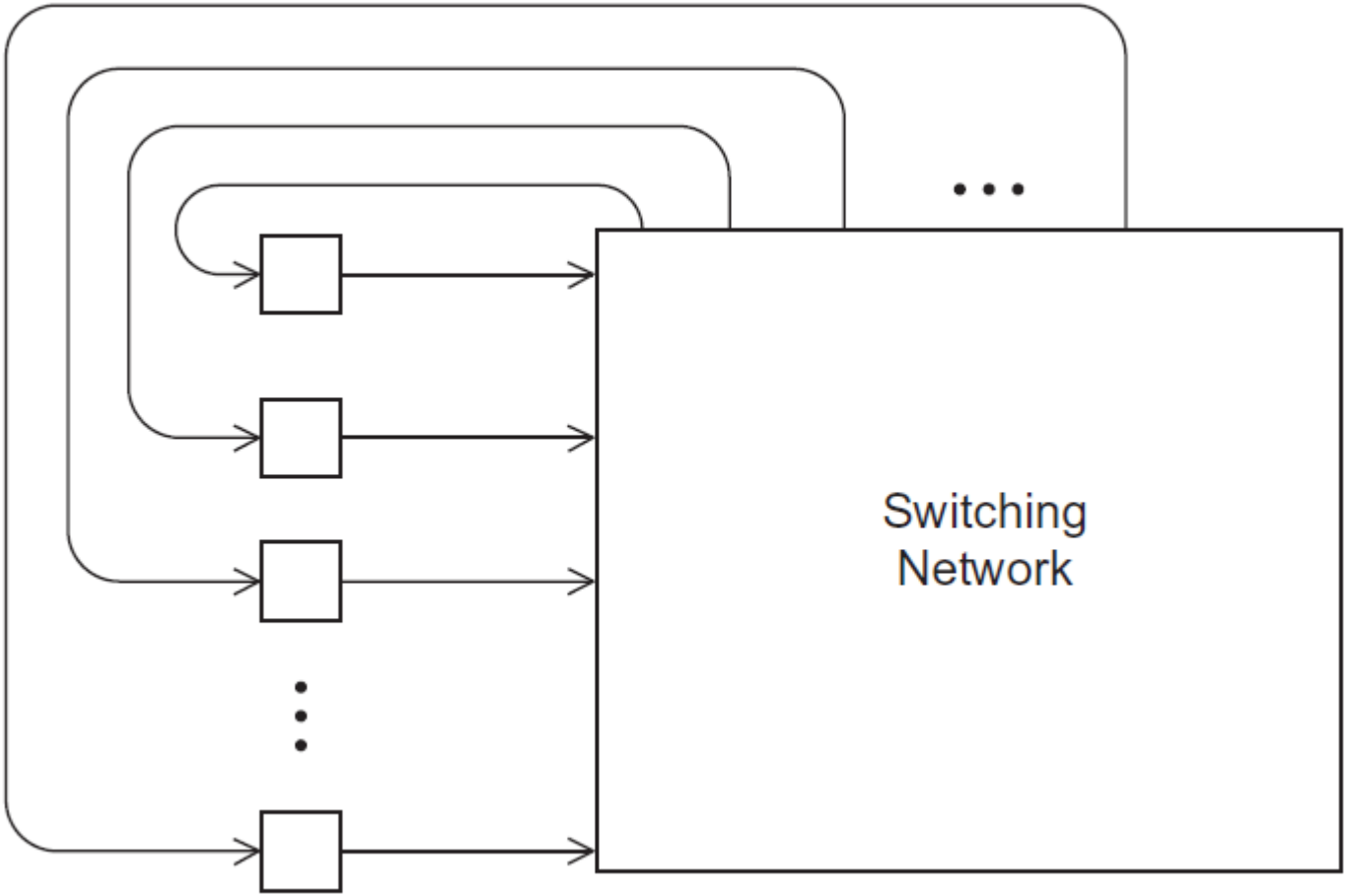
one- two- three-dimensional

(2) Indirect interconnect

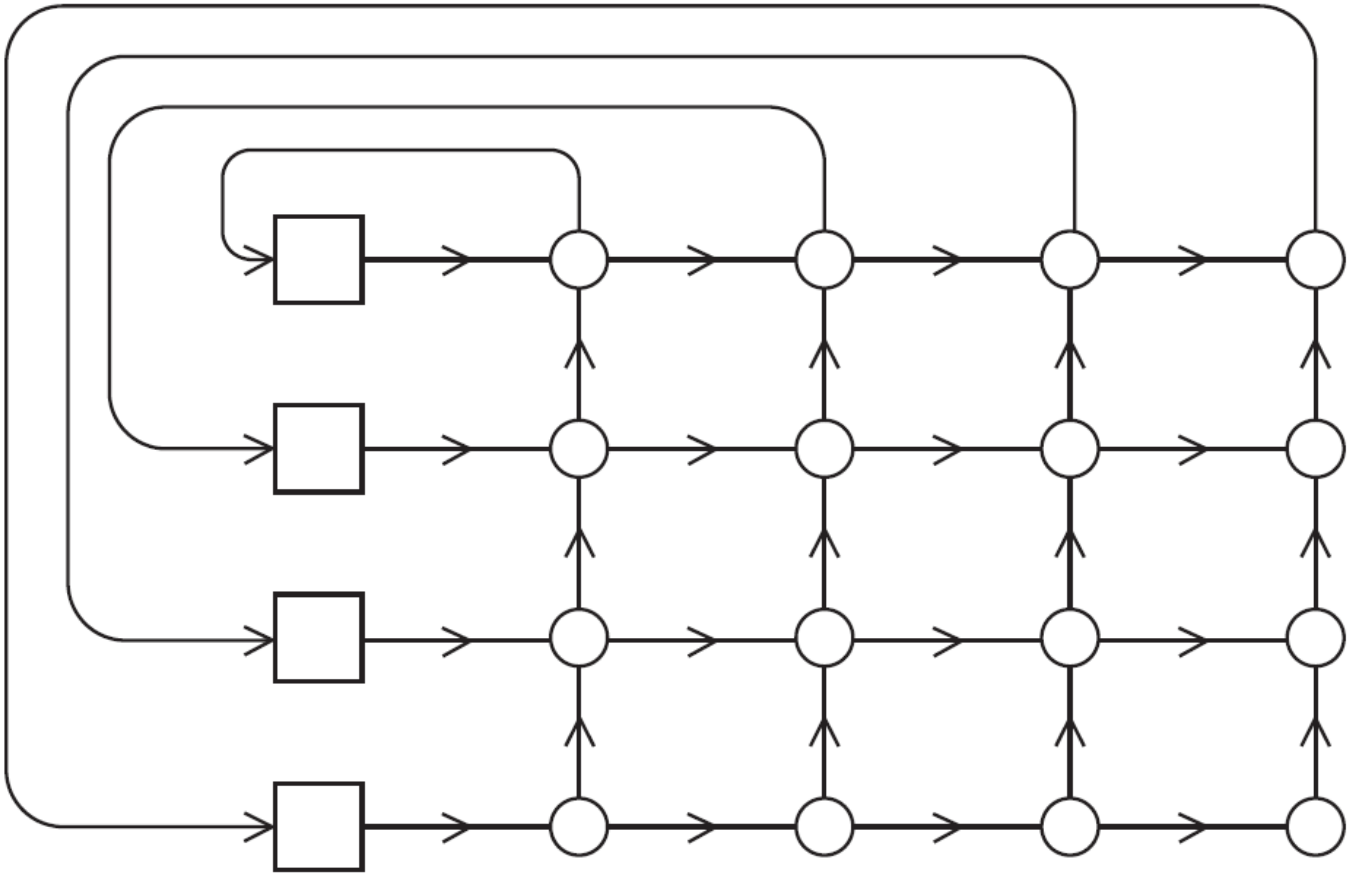
Indirect interconnects

- Simple examples of indirect networks:
  - Crossbar
  - Omega network
- Often shown with unidirectional links and a collection of processors
- Each of which has an outgoing and an incoming link, and a switching network.

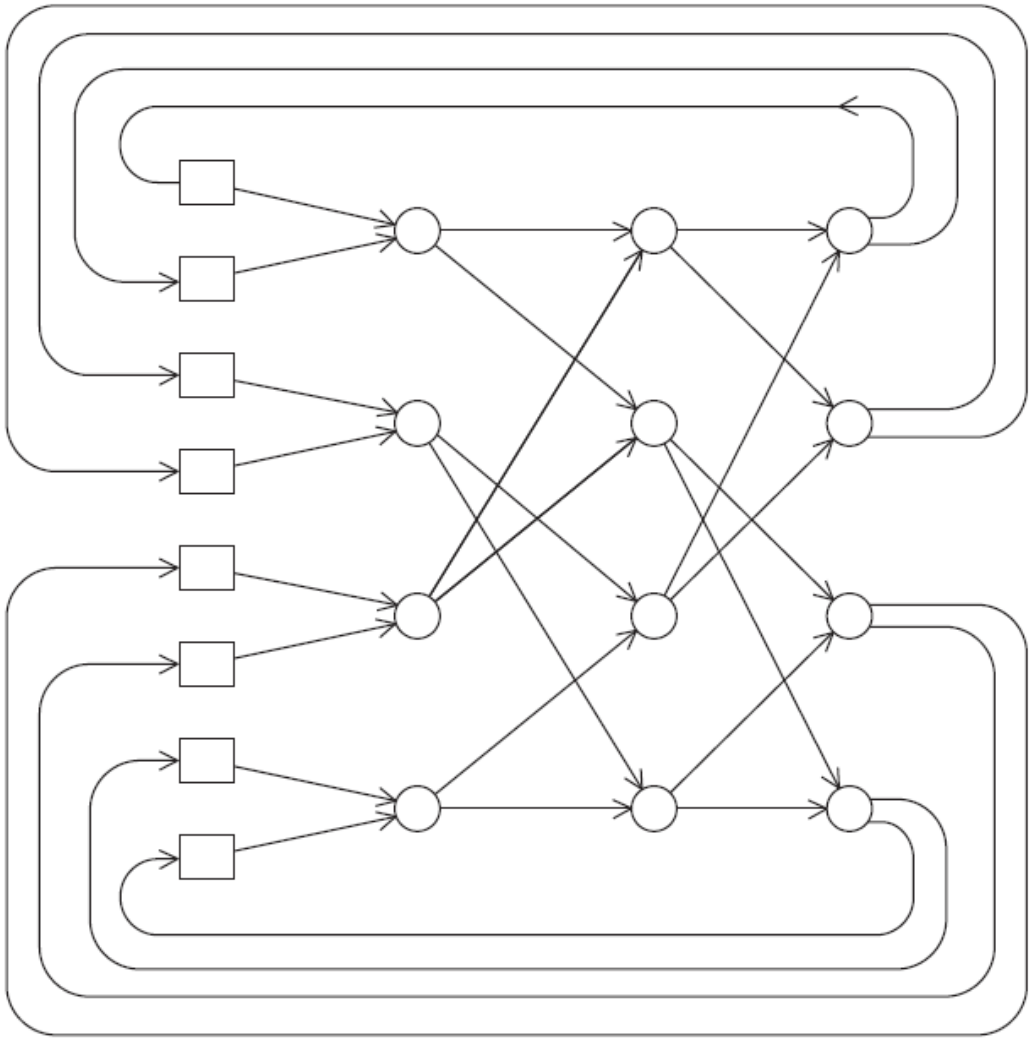
A generic indirect network



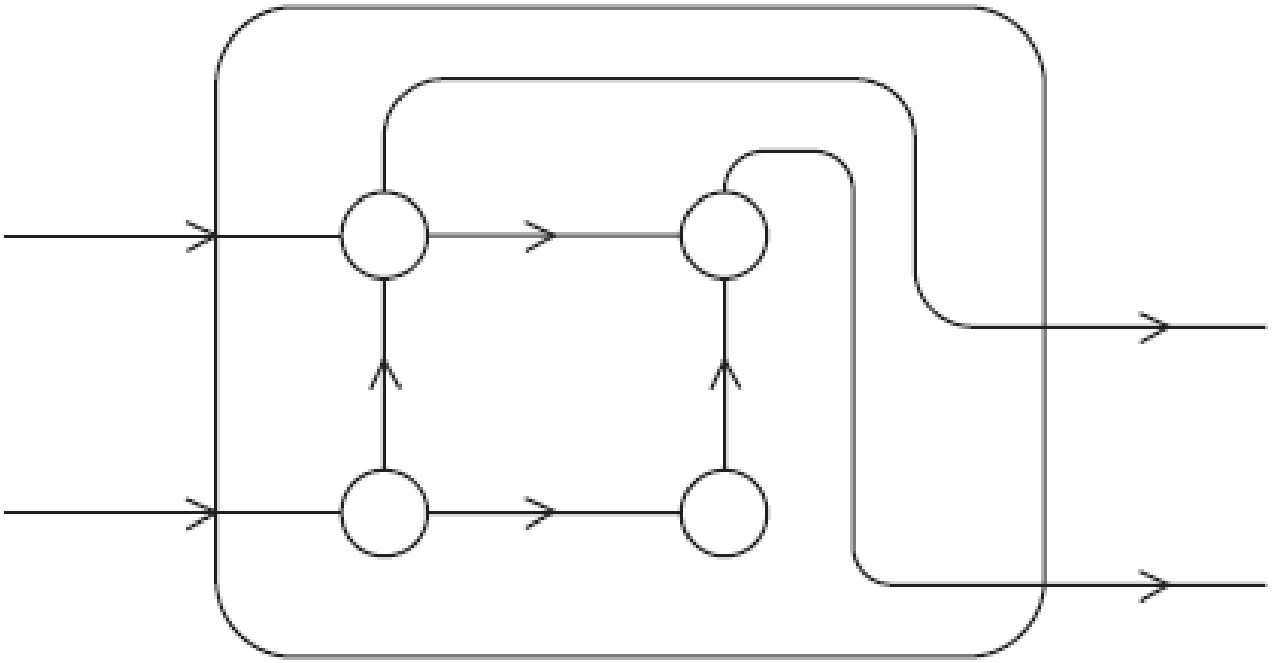
Crossbar interconnect for distributed memory



An omega network



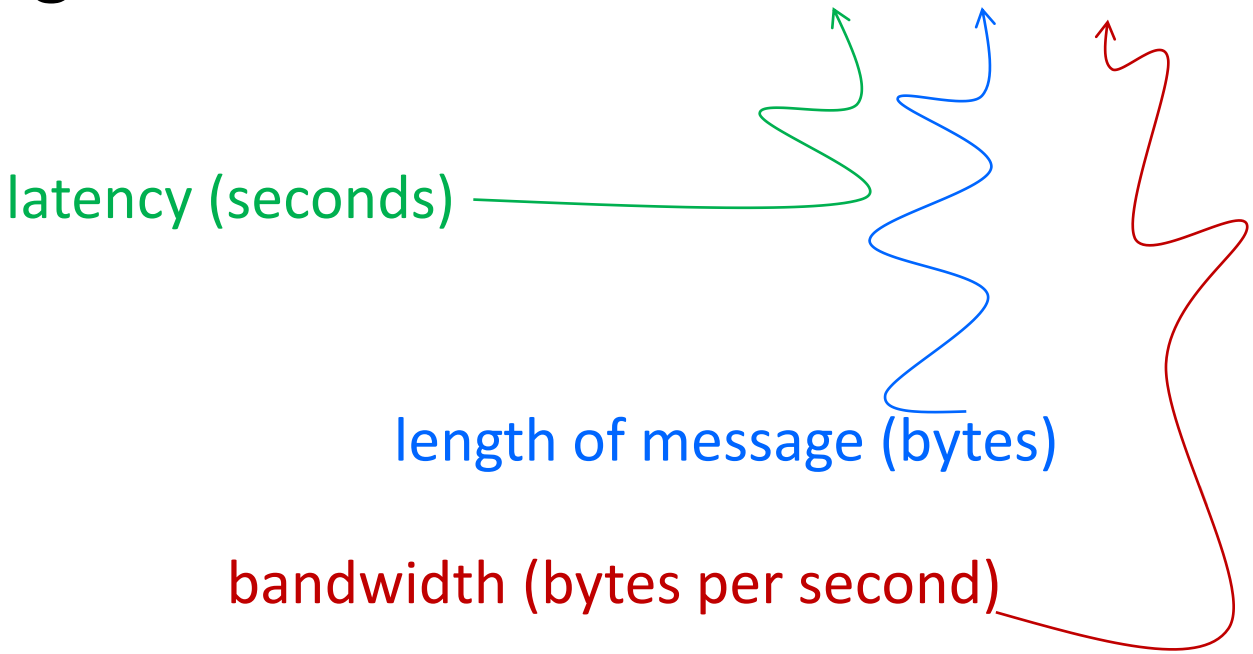
A switch in an omega network



More definitions

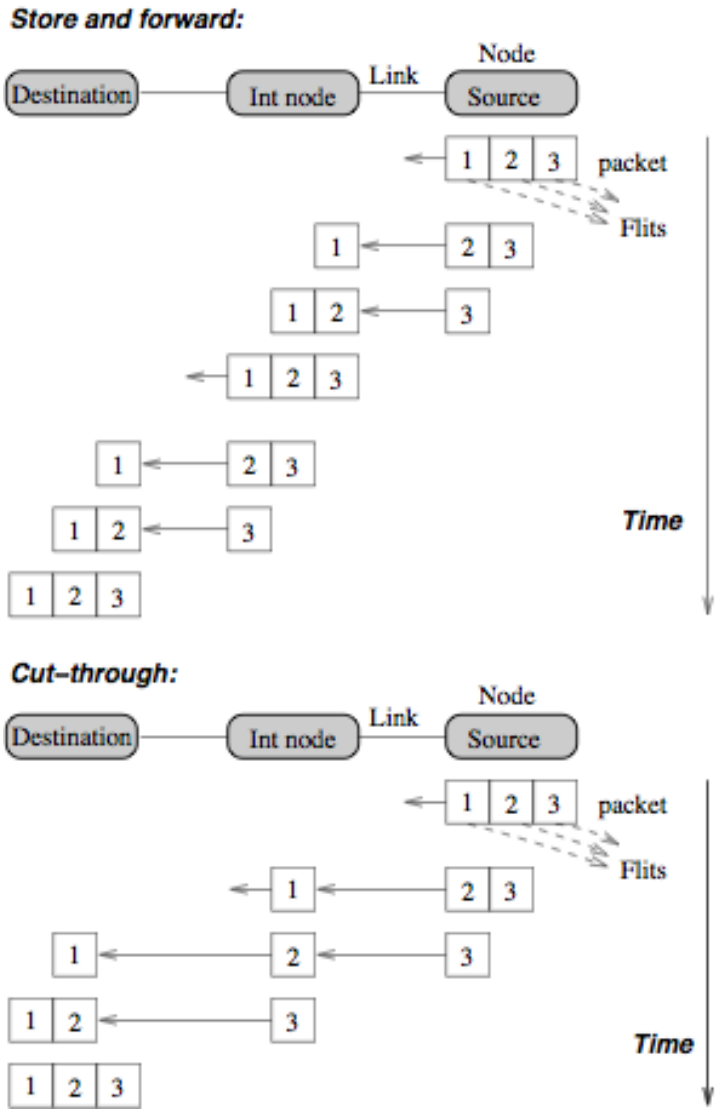
- Any time data is transmitted, we're interested in how long it will take for the data to reach its destination.
- Latency
  - The time that elapses between the source's beginning to transmit the data and the destination's starting to receive the first byte.
- Bandwidth
  - The rate at which the destination receives data after it has started to receive the first byte.

Message transmission time = `l + n / b`



2. Routing Policy

Routing policy



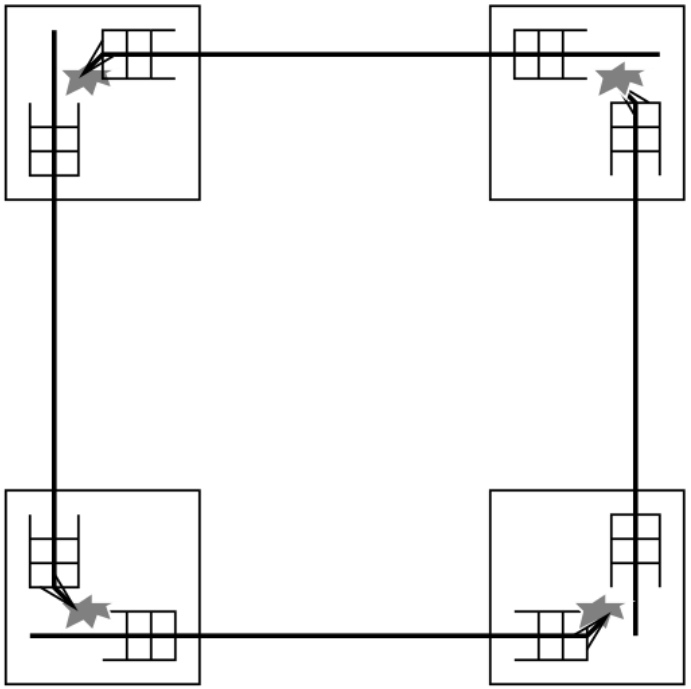
How to send a packet over the network?

- **Store & Forward:**
  - Send all flits over a link, then over another, and so on
  - T = h * (Txmit + Tswitch)
- **Cut-through:**
  - Send flits in pipeline over the links
  - T = h * Tswitch + Txmit
- **Implication**:
  - cut-through routing reduces the importance of topology

Routing Policy

- **Minimal vs. non-minimal**
  - **Minimal** = each packet travels the least number of hops
- **Deterministic vs. non-deterministic**
  - **Deterministic** = each sender-dest pair uses a single path for all messages
- **Adaptive vs. non-adaptive**
  - **Adaptive** = message can change path as it is being transmitted
- **Deadlock-free vs. Deadlock possible**
  - Deadlock due to several messages mutually waiting for buffer space

What is Deadlock?

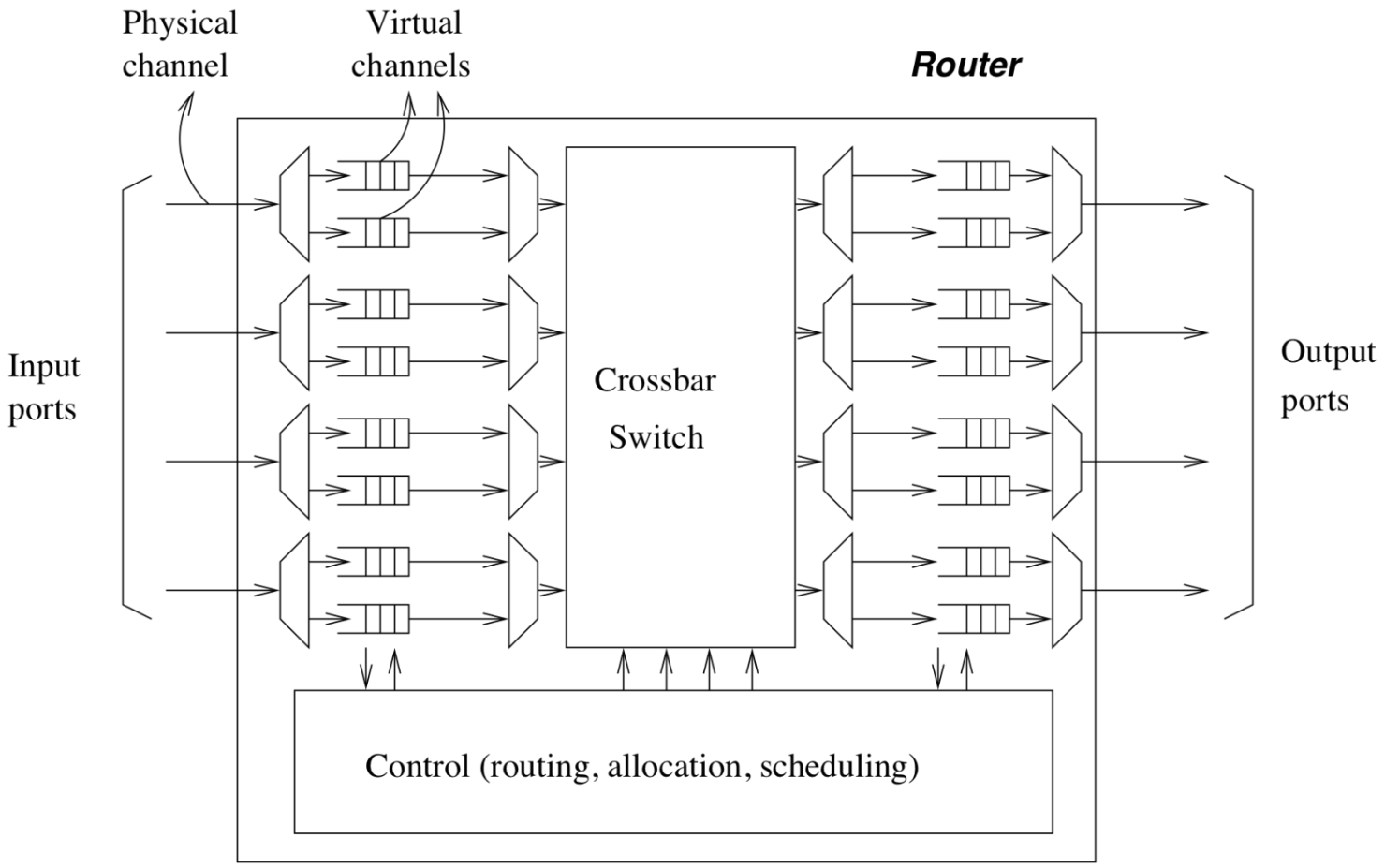


The figure shows four packets filling up the input buffer and need the next output buffer, but the output buffer is already full

- **Inability of the network to forward packets**
  - Due to limited buffer space and cyclic dependence on buffer acquisition
- **Handling a deadlock**
  - **Detect and break**: expensive
  - **Drop packets**: adverse effect in performance and complicates protocol
  - **Avoid**: restriction on routing

3. Router Architecture

Router Architecture



- Router has four input ports and four output ports
- Each channel has two virtual channels (VCs), with separate buffers

Router Architecture

- **Arriving packet sent to the appropriate VC (buffer) based on its header**

- **Control logic arbitrates among multiple input channels and allocates them to the input of the crossbar switch**
  - It also allocates output VC to the output of the switch
  - It arbitrates multiple input ports that want to send packet to the same output port

Router Architecture

- **Switch typically implemented using a crossbar (to minimize latency and collision)**
  - Multiple flits from different VCs can go on the switch as long as they go to different output VCs
- **Steps can also be pipelined to improve router throughput**

Summary

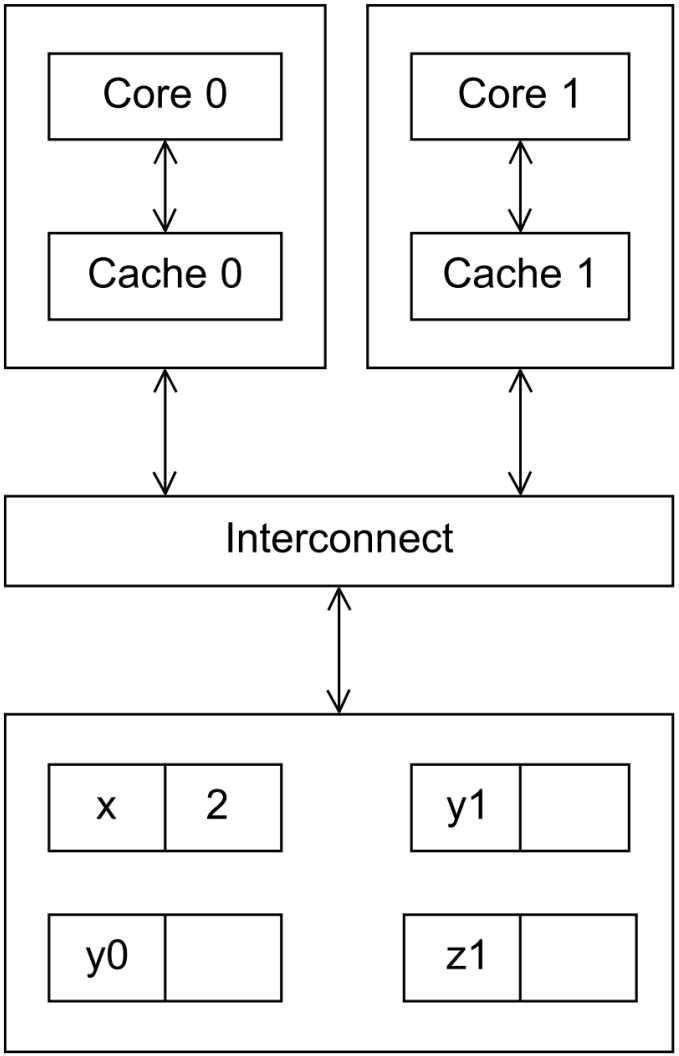
- **Interconnection network**
  - Basic concepts
  - Design consideration
  - Main aspects
- **Network Topology**
- **Routing Policy**
- **Router Architecture**

Cache coherence

Recall that…

- **Caches are managed by system hardware**
- **Out of a programmer's direct control!**
- **Any undesired consequence?**

What would happen?



What would happen?

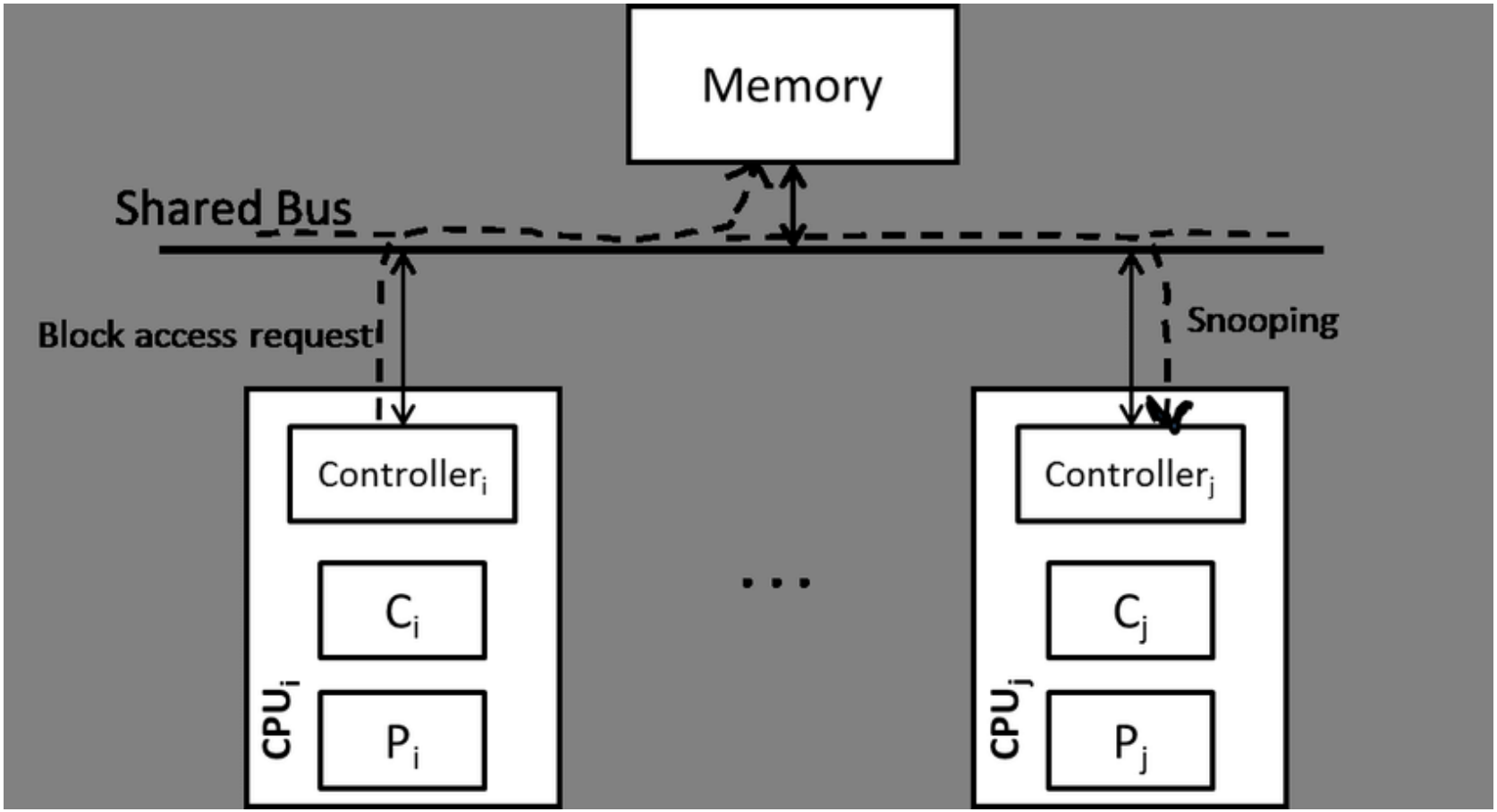
Time	Core 0	Core 1
0	`y0 = x;`	`y1 = 3*x;`
1	`x = 7;`	Statement(s) not involving `x`
2	Statement(s) not involving `x`	`z1 = 4*x;`

- **Suppose executed at the indicated times**
- **What value z1 will get, 4*7=28 or 4*2=8?**
- **Unpredictable! Whether write-through or write-back…**

Cache Coherence Problem

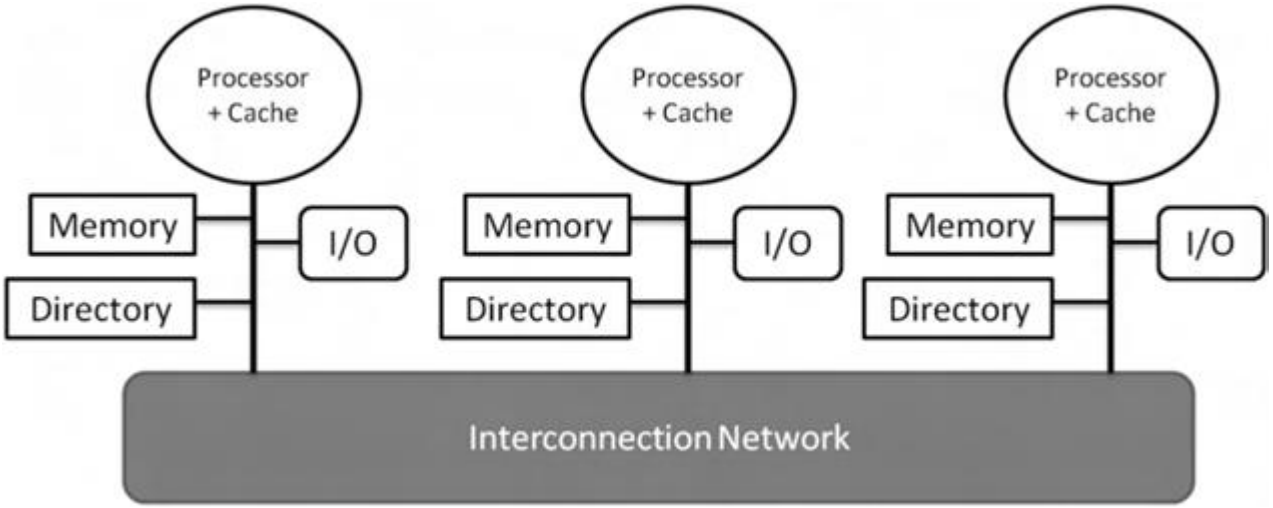
- **When the caches of multiple processors store the same variable, an update by one processor to the cached variable, how to ensure the cached value stored by the other processors is also updated.**
  - Snooping cache coherence
  - Directory-based cache coherence

监听（Snooping）Cache Coherence

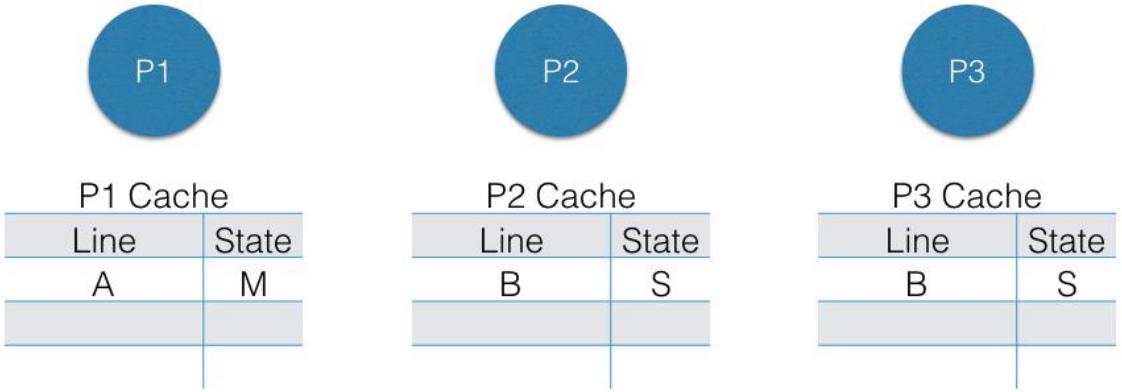


基于目录的(Directory-based) Cache Coherence

- **Directory (a data structure) stores the status of each cache line**
  - Also support distributed-memory
  - Directory is maintained in a distributed way



基于目录的(Directory-based) Cache Coherence



Location	Nodes	State
A	100	EM
B	011	S
C	000	U
D	000	U
…		

Diagram of full bit vector directory format, where E=Exclusive, S=Shared, M=Modified, and U=Uncached

基于目录的(Directory-based) Cache Coherence

- **Cons: Substantial additional storage required for the directory**

- **Pros: When a cache variable is updated, only the cores storing that variable need to be contacted**

伪共享 (False Sharing)

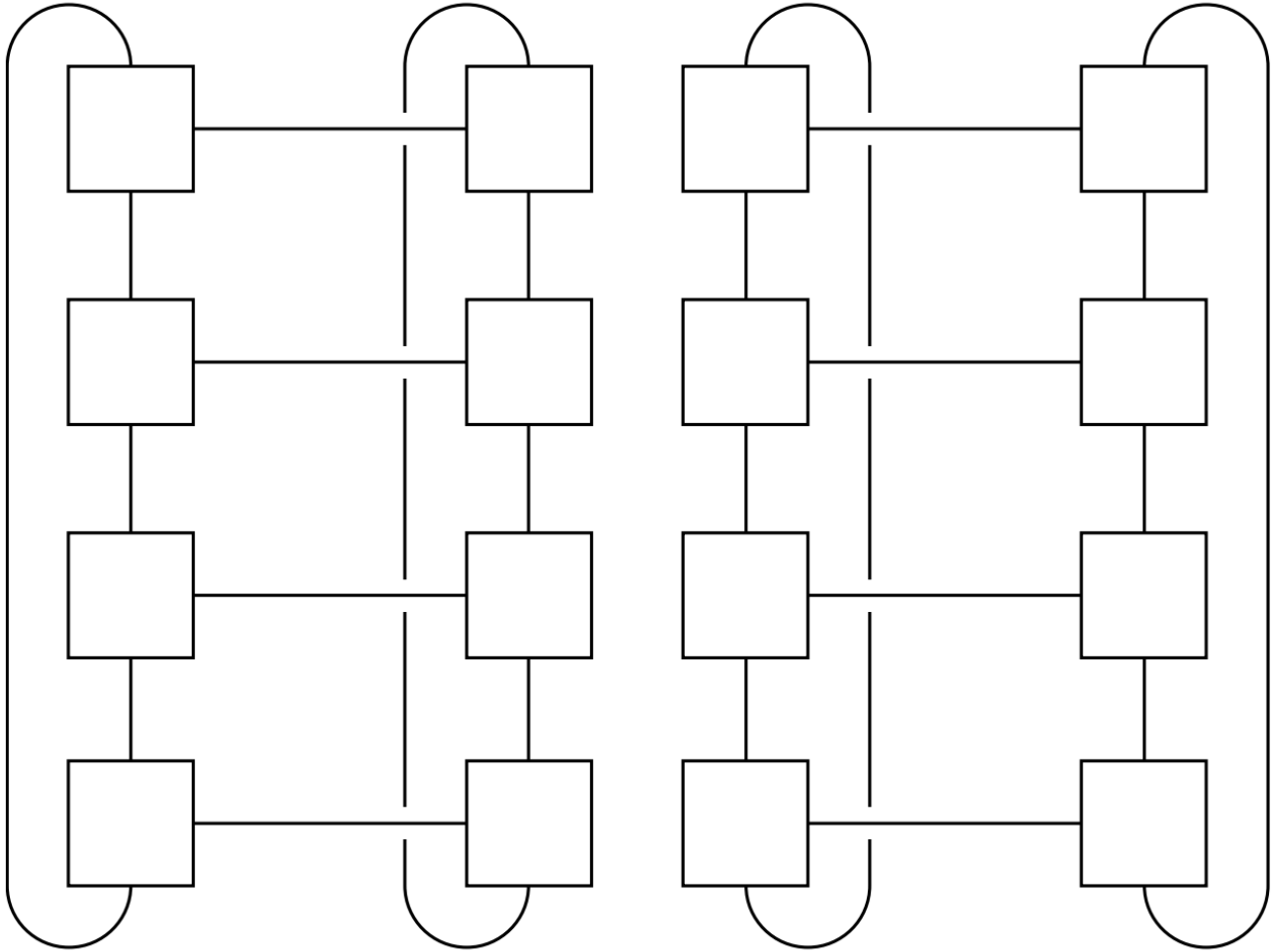
```
/* Private variables */
int i, j, iter_count;

/* Shared variables initialized by one core */
int m, n, core_count
double y[m];

iter_count = m/core_count

/* Core 0 does this */
for (i = 0; i < iter_count; i++)
    for (j = 0; j < n; j++)
        y[i] += f(i,j);

/* Core 1 does this */
for (i = iter_count; i < 2*iter_count; i++)
    for (j = 0; j < n; j++)
        y[i] += f(i,j);
```

Suppose $m = 8$, core_count = 2, **each cache line is 64 bytes!**

The system is behaving *as if* the elements of y were being shared by the cores !

Supplementary materials

Computing bisection width



back