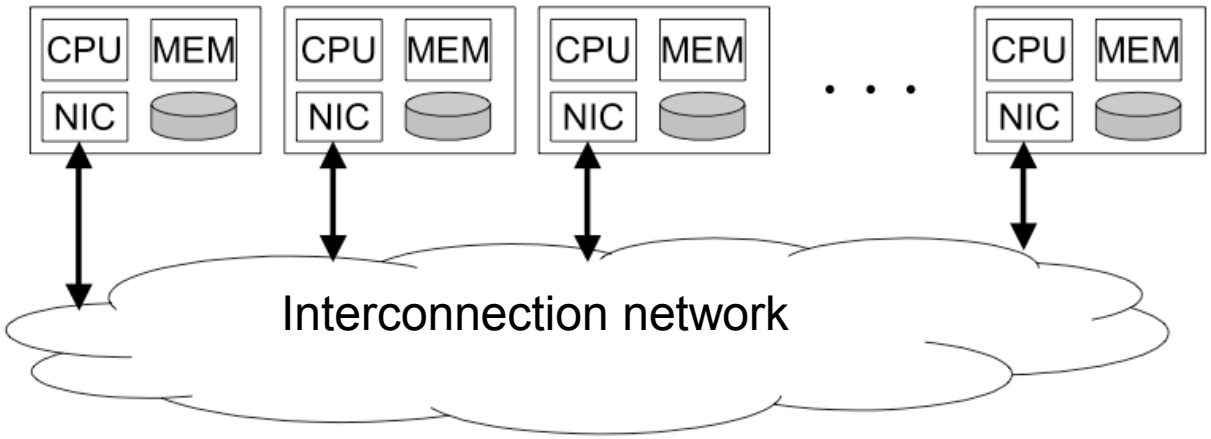
Interconnect Basics

Where Is Interconnect Used?

- To connect components

- Many examples
  - Processors and processors
  - Processors and memories (banks)
  - Processors and caches (banks)
  - Caches and caches
  - I/O devices



Introduction

- The interconnect plays a decisive role in the performance of both distributed and shared memory systems:
  - Even if the processors and memory have virtually unlimited performance,
  - A slow interconnect will seriously degrade the overall performance of all but the simplest parallel program.

Why Is It Important?

- Affects the scalability of the system
  - How large of a system can you build?
  - How easily can you add more processors?

- Affects performance and energy efficiency
  - How fast can processors, caches, and memory communicate?
  - How long are the latencies to memory?
  - How much energy is spent on communication?

Interconnection Network Basics

- Topology
  - Specifies the way switches are wired
  - Affects routing, reliability, throughput, latency, building ease
- Routing (algorithm)
  - How does a message get from source to destination
  - Static or adaptive
- Buffering and Flow Control
  - What do we store within the network?
    - Entire packets, parts of packets, etc?
  - Tightly coupled with routing strategy

Topology

- Bus (simplest)
- Point-to-point connections (ideal and most costly)
- Crossbar (less costly)
- Ring
- Tree
- Omega
- Hypercube
- Mesh
- Torus
- Butterfly
- …

Shared-memory interconnects

Shared-memory interconnects

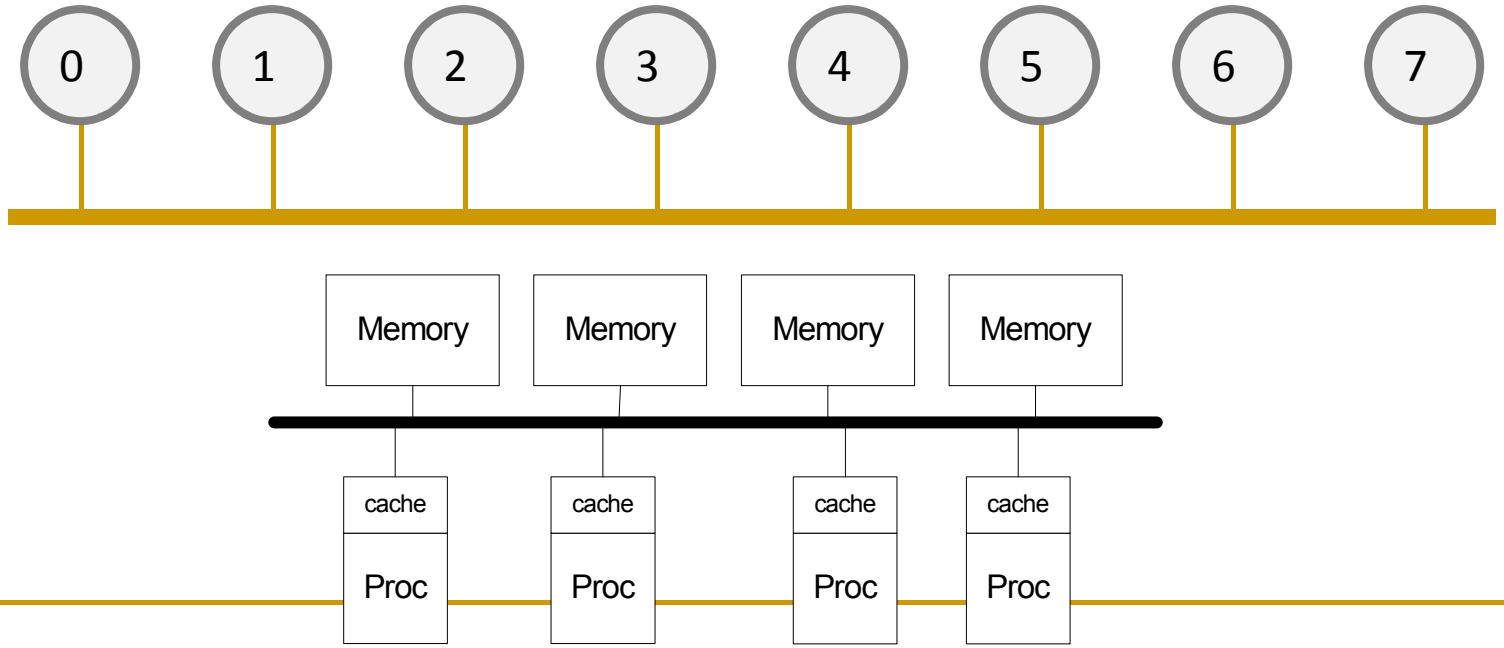
- Two most widely used interconnects on shared-memory systems are
  - buses and crossbars

Bus

- A bus is a collection of parallel communication wires together with some hardware that controls access to the bus.
- The key characteristic of a bus is that the communication wires are shared by the **devices** that are connected to it.

Bus

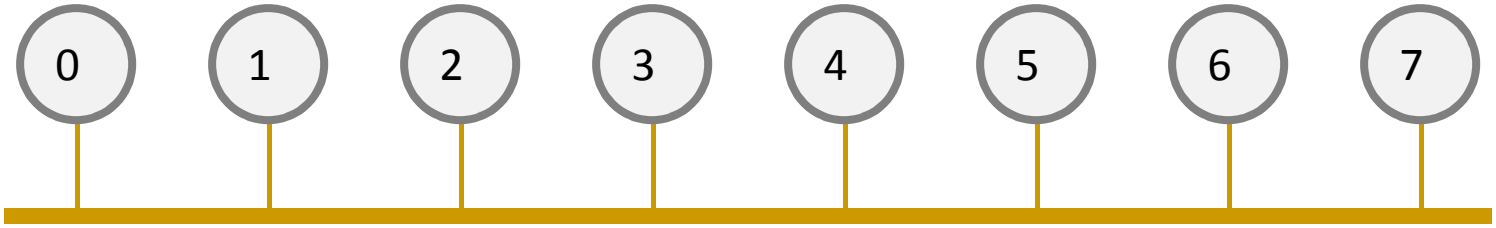
+ Simple
+ Cost effective for a small number of nodes (lost cost)
+ Easy to implement coherence (snooping and serialization)

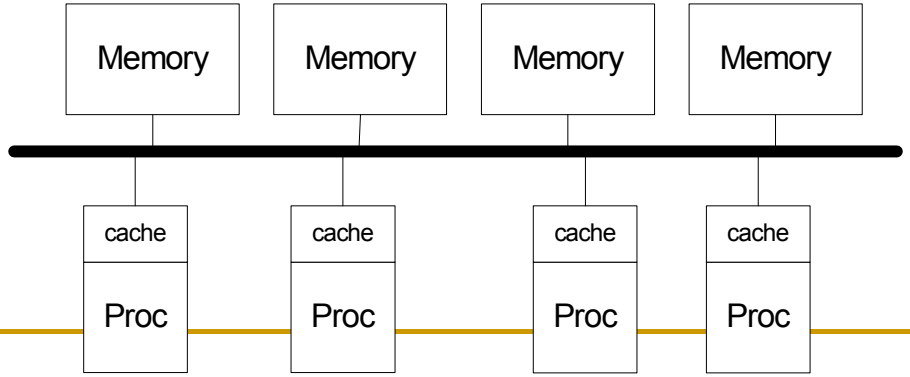


Bus

- Not scalable to large number of nodes (limited bandwidth, electrical loading → reduced frequency)
- High contention → fast saturation (Since the communication wires are shared, as the number of devices connected to the bus increases)

Since the communication wires are shared, as the number of devices connected to the bus increases, the likelihood that there will be contention for use of the bus increases, and the expected performance of bus decreases.



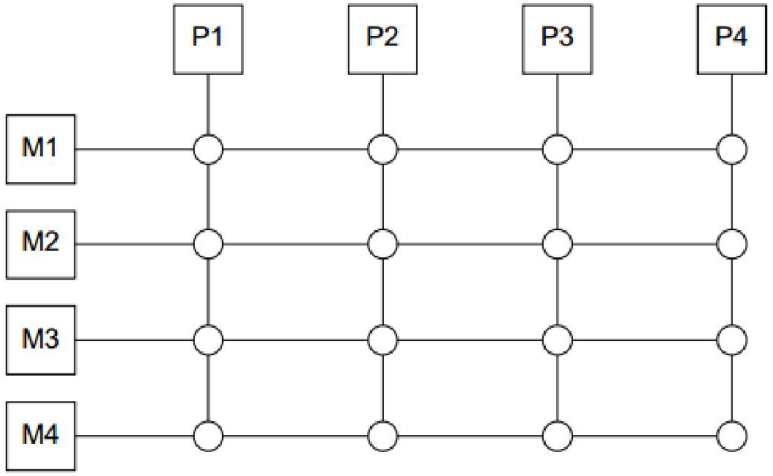


Why Switches ?

- Therefore, if we connect a large number of processors to a bus, we would expect that the processors would frequently have to wait for access to main memory.
- Thus, as the size of shared-memory systems increases, buses are rapidly being replaced by switched interconnects.

Switches

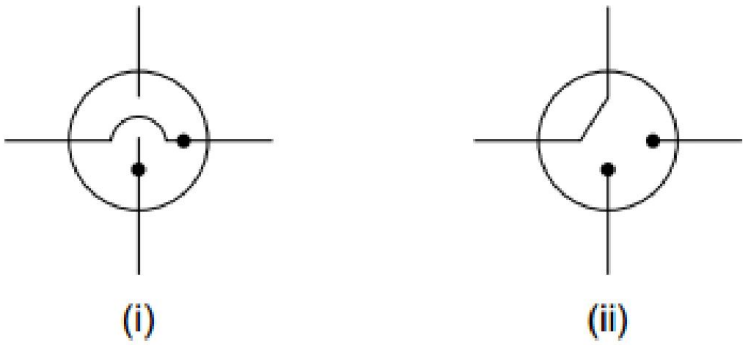
- **Switched** interconnects use switches to control the routing of data among the connected devices. A crossbar is illustrated in Figure 2.7(a).



(a)

- The lines are bidirectional communication links, the squares are cores or memory modules, and the circles are switches.

Switches

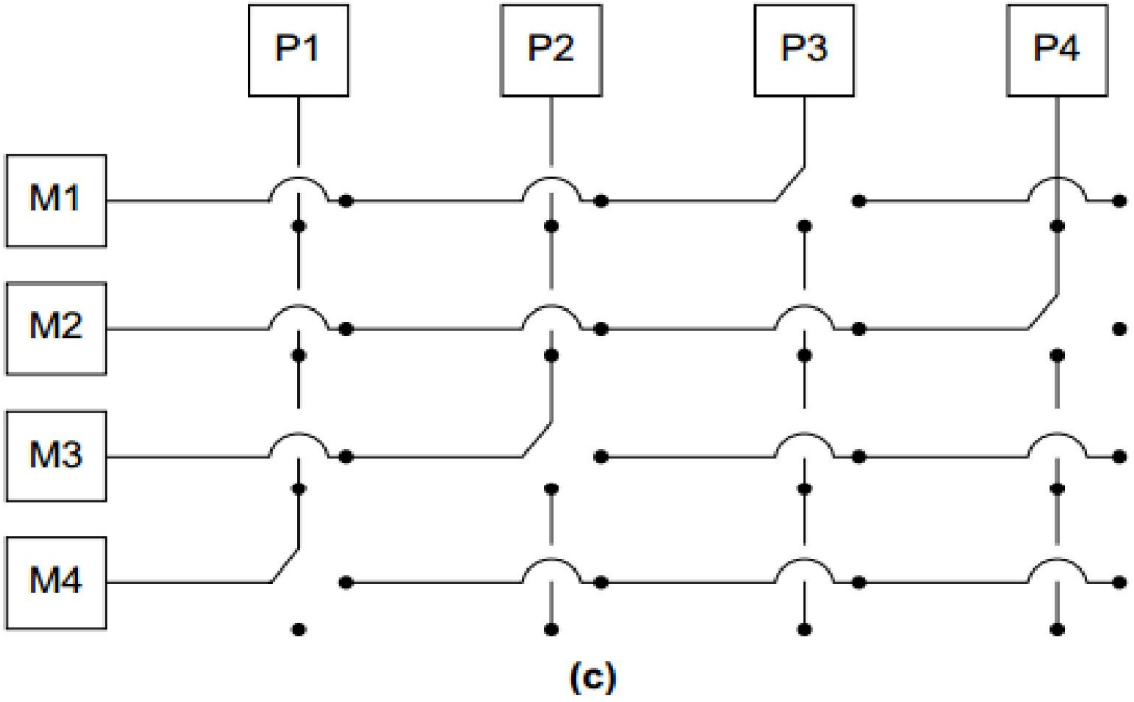


(b) configuration of internal switches in a crossbar;

Switches

- The individual switches can assume one of the 2 configurations shown in (b).
- With these switches and at least as many memory modules as processors, there will only be a conflict between 2 cores attempting to access memory if the 2 cores attempt to simultaneously access the same memory module.

Switches



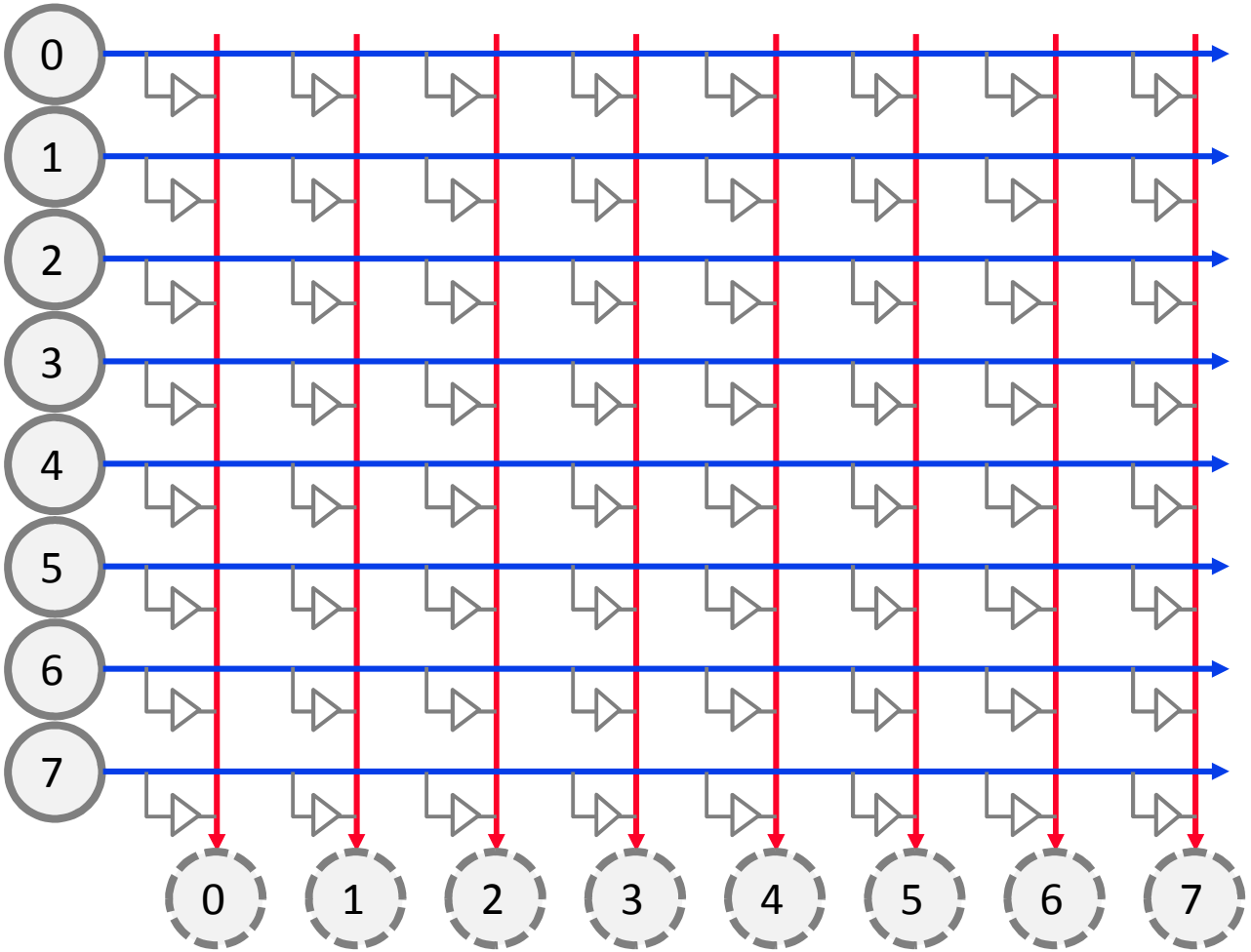
c) simultaneous memory accesses by the processors

Figure 2.7(c) shows the configuration of the switches if P1 writes to M4, P2 reads from M3, P3 reads from M1 , and P4 writes to M2.

Switches

- Crossbars allow simultaneous communication among different devices, so they are much faster than buses. However, the cost of the switches and links is relatively high.
- A small bus-based system will be much less expensive than a crossbar-based system of the same size.

Another Crossbar Design



Distributed-memory interconnects

Distributed-memory interconnects

- Two most widely used interconnects on Distributed-memory systems are
  - direct interconnect and indirect interconnects

Direct Interconnects

Direct interconnects

- In a direct interconnect each switch is directly connected to a processor-memory pair, and the switches are connected to each other.

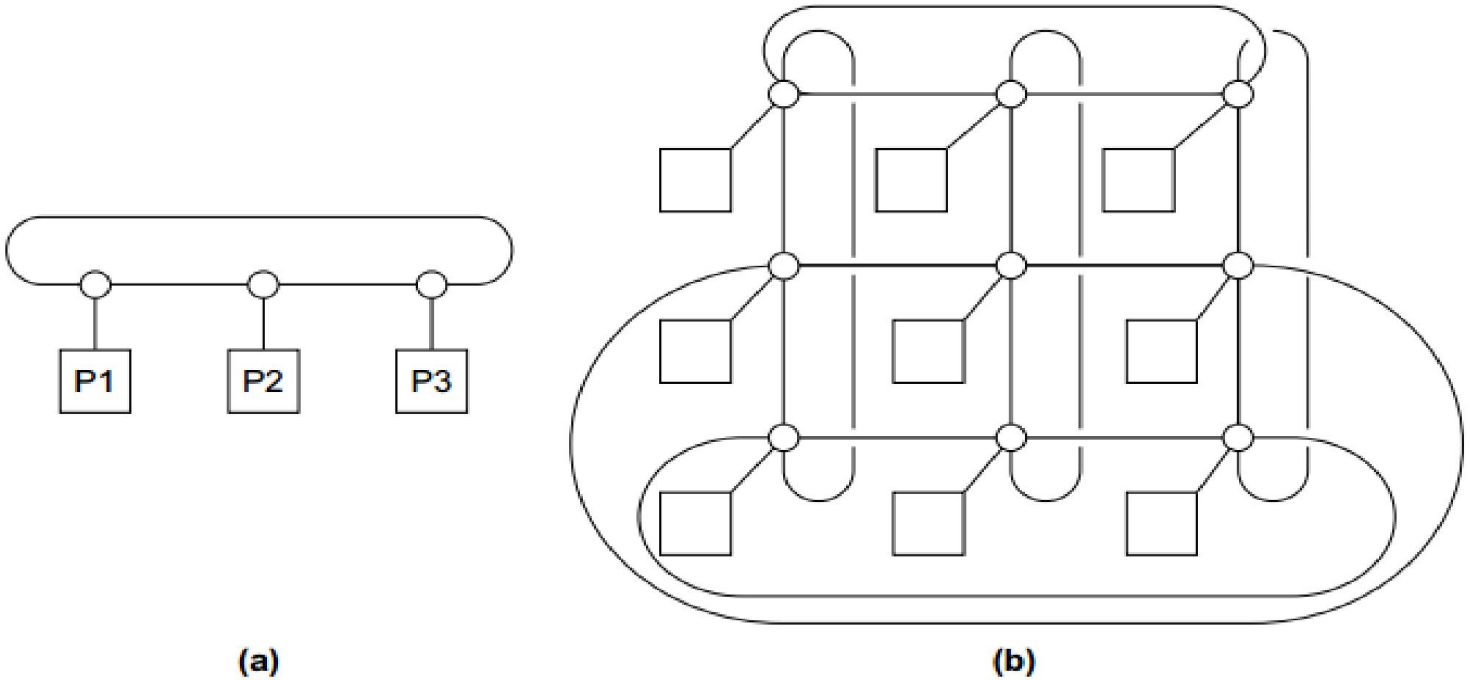


FIGURE 2.8

(a) A ring and (b) a toroidal mesh

Direct interconnects

- the circles are switches, the squares are processors, and the lines are bidirectional links.
- A ring is superior to a simple bus since it allows multiple simultaneous communications.
  - it's easy to devise communication schemes in which some of the processors must wait for other processors to complete their communications.

Direct interconnects

- The toroidal mesh will be more expensive than the ring, because the switches are more complex
- They must support five links(4 switch connections + 1 PE connection) instead of three—and if there are p processors, the number of links is 3p in a toroidal mesh, while it's only 2p in a ring.
- Number of possible simultaneous communications patterns is greater with a mesh than with a ring.

bisection width.

- One measure of "number of simultaneous communications" or "connectivity" is bisection width.
- To understand this measure, imagine that the
  - Parallel system is divided into two halves, and each half contains half of the processors or nodes. How many simultaneous communications can take place "across the divide" between the halves?

Bisection Width (Simple words)

- Bisection width

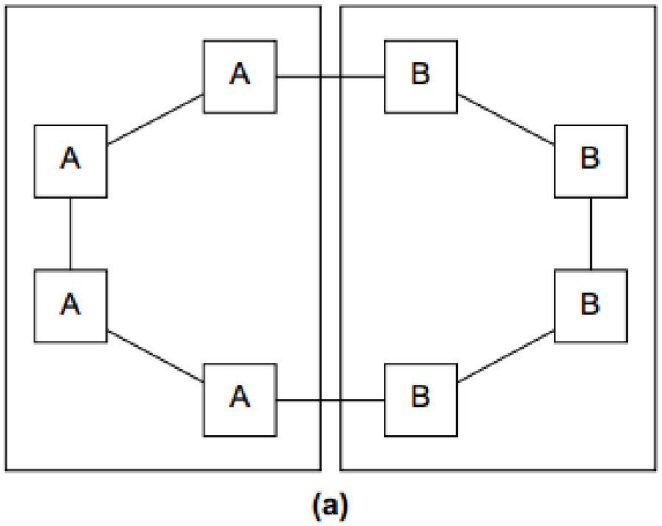
  The # of edges to be removed to separate the graph into two equal parts.

- Cost

  1) # of nodes & # of edges

  2) Bisection width.

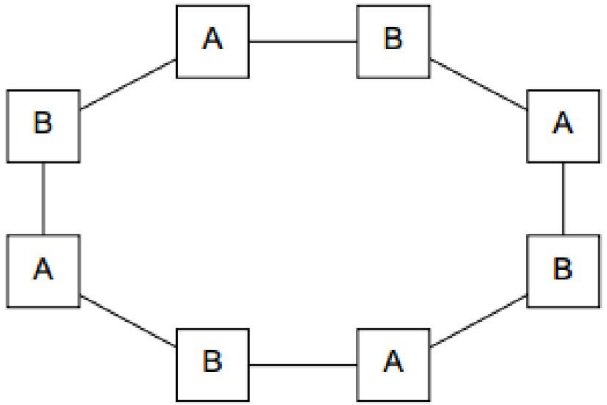
bisection width.



Two bisections of a ring: (a) only two communications can take place between the halves

- we've divided a ring with eight nodes into two groups of four nodes, and we can see that only two communications can take place between the halves.

bisection width.



(b)

Two bisections of a ring: (a) only four communications can take place between the halves

However, in this figure,

- We've divided the nodes into two parts so that four simultaneous communications can take place.
- so what's the bisection width? The bisection width is supposed to give a "worst-case" estimate, so the bisection width is two—not four.

Alternative way of computing the bisection width

- to remove the minimum number of links needed to split the set of nodes into two equal halves. The number of links removed is the bisection width.

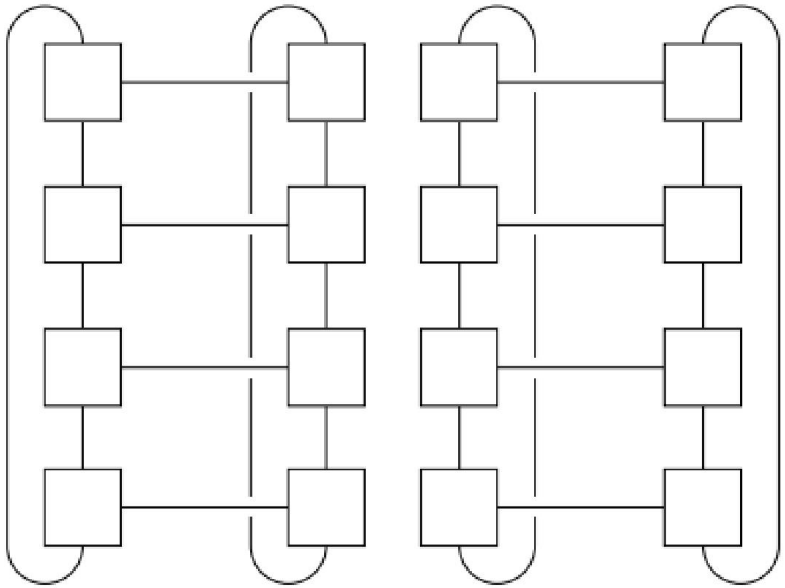


Figure 2.10: A bisection of toroidal mesh

If we have a square two-dimensional toroidal mesh with $p = q^2$ nodes (where q is even), then we can split the nodes into two halves by removing the "middle" horizontal links and the "wraparound" horizontal links. See Figure 2.10. This suggests that the bisection width is at most $2q = 2\sqrt{p}$. In fact, this is the smallest possible number of links and the bisection width of a square two-dimensional toroidal mesh is $2\sqrt{p}$.

What is bisection bandwidth? How to compute?

- *Split network so that both sides have equal number of nodes.*
- *Count the number of links crossing the dividing lines. Multiply by bandwidth of each line to obtain bisection bandwidth (or simply count links if individual bandwidth is not known).*
- The bandwidth of a link is the rate at which it can transmit data. It's usually given in megabits or megabytes per second.

What is bisection bandwidth? How to compute?

- Bisection bandwidth is often used as a measure of network quality. It's similar to bisection width.
- However, instead of counting the number of links joining the halves, it sums the bandwidth of the links.
- For example, if the links in a ring have a bandwidth of one billion bits per second, then the bisection bandwidth of the ring will be two billion bits per second or 2000 megabits per second.

What is bisection bandwidth? How to compute?

- Fully interconnected network
  - each switch is directly connected to every other switch.
  - Take into account all the nodes on side A of the bisection. Each node must connect with n/2 nodes on the other side of the bisection (side B) (since the number of nodes is equally distributed). There are n/2 nodes on side A. Therefore # of connections crossing the bisection is (n/2) * (n/2) = $(n/2)^2$.
  - Therefore, its bisection width = $(n/2)^2$.

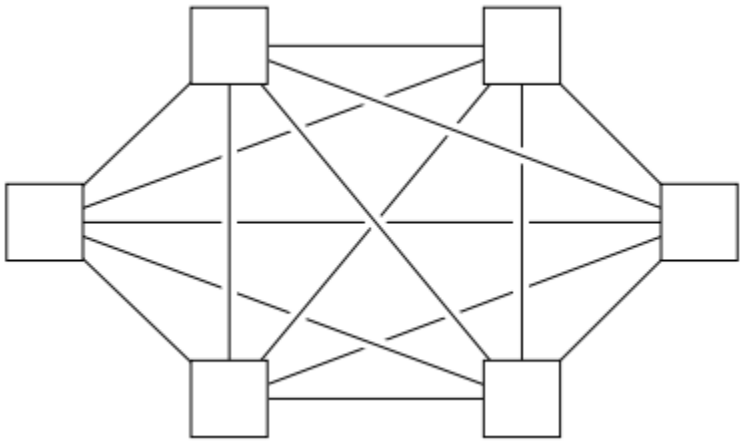


Figure 2.11 : A fully connected network

What is bisection bandwidth? How to compute?

- Fully interconnected network
  - it's impractical to construct such an interconnect for systems with more than a few nodes and each switch must be capable of connecting to p links.
  - It is therefore more a "theoretical best possible" interconnect than a practical one, and it is used as a basis for evaluating other interconnects.

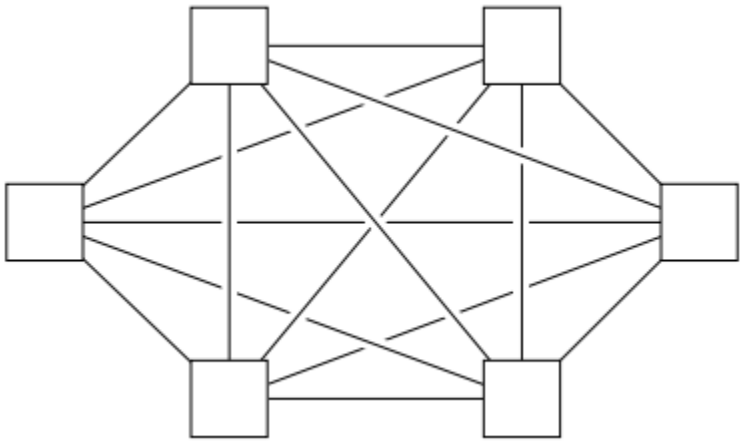


Figure 2.11 : A fully connected network

Hypercube

- The hypercube is a highly connected direct interconnect that has been used in actual systems.
- 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.

Hypercube

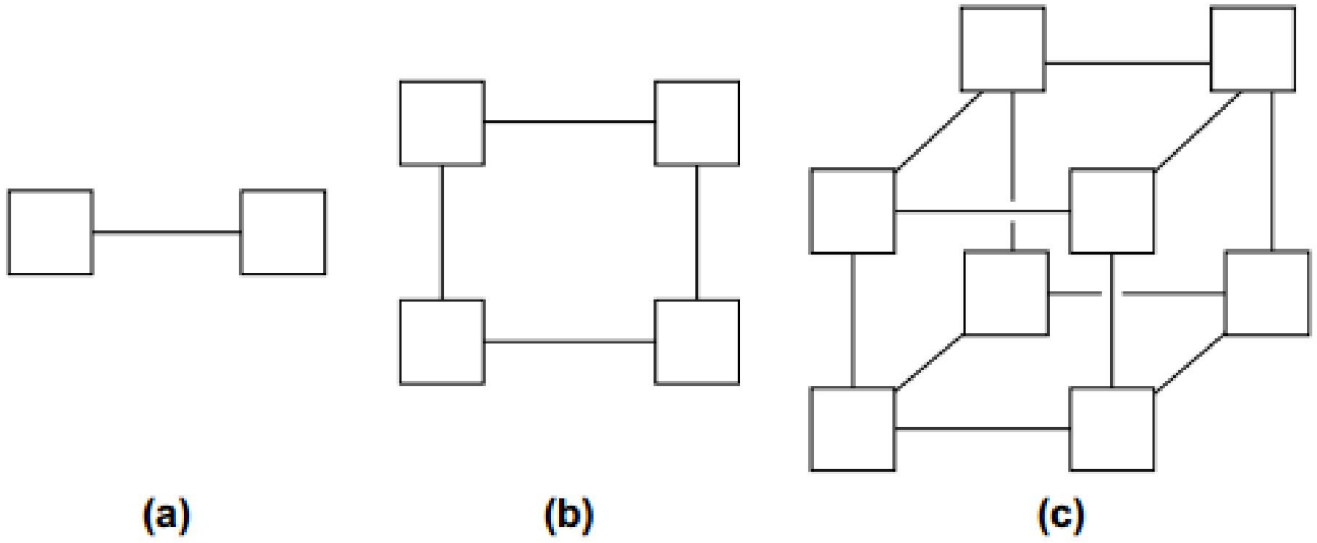


Figure 2.12 (a) One-, (b) two-, and (c) three-dimensional hypercubes

Hypercube

- Thus, a hypercube of dimension d has p = 2^d nodes, and a switch in a d-dimensional hypercube is directly connected to a processor and d switches.
- The bisection width of a hypercube is p/2, so it has more connectivity than a ring or toroidal mesh, but the switches must be more powerful, since they must support 1 + d = 1 + $\log_2$ (p) wires, while the mesh switches only require five wires.

What is bisection bandwidth? How to compute?

- Buses:

What is bisection bandwidth? How to compute?

- Buses:
  - It is simpler for buses. Since there is one connection for all nodes, if we divided all the nodes into two equal groups, the number of links crossing the bisection will still remain as 1.

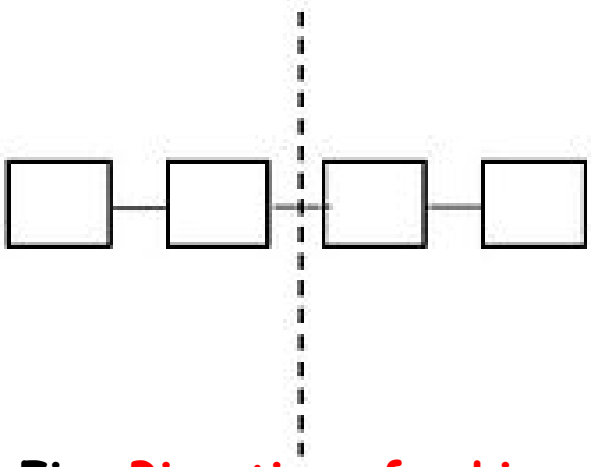


Fig: Bisection of a Linear array Network

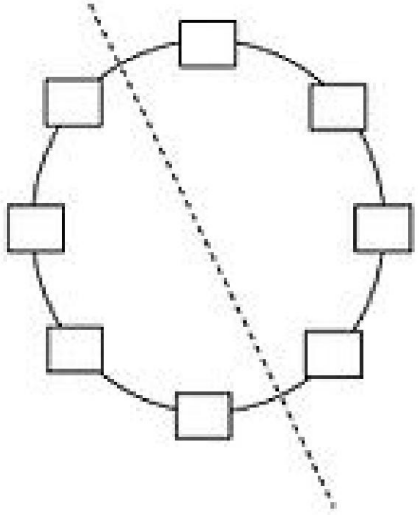
For a linear array with n nodes bisection bandwidth is one link bandwidth. For linear array only one link needs to be broken to bisect the network into two partitions.

What is bisection bandwidth? How to compute?

- Ring:

What is bisection bandwidth? How to compute?

- Ring:
  - Again this is trivial. Splitting a ring into two will always result in two links crossing the bisection. Bisection bandwidth is therefore 2.



For ring topology with n nodes two links should be broken to bisect the network, so bisection bandwidth becomes bandwidth of two links.

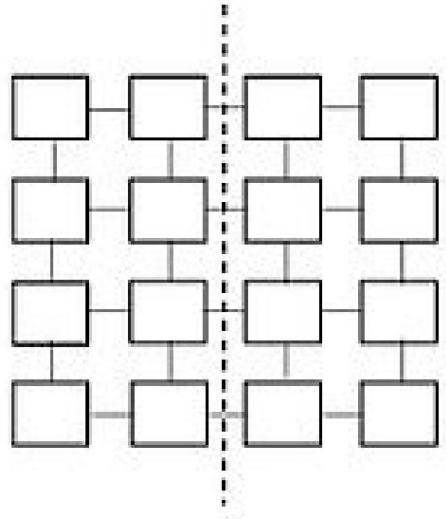
Fig: Bisection of a Ring Network

What is bisection bandwidth? How to compute?

- **Mesh:**

What is bisection bandwidth? How to compute?

- **Mesh:**



For Mesh topology with n nodes, √(n) links should be broken to bisect the network, so bisection bandwidth is bandwidth of √(n) links.

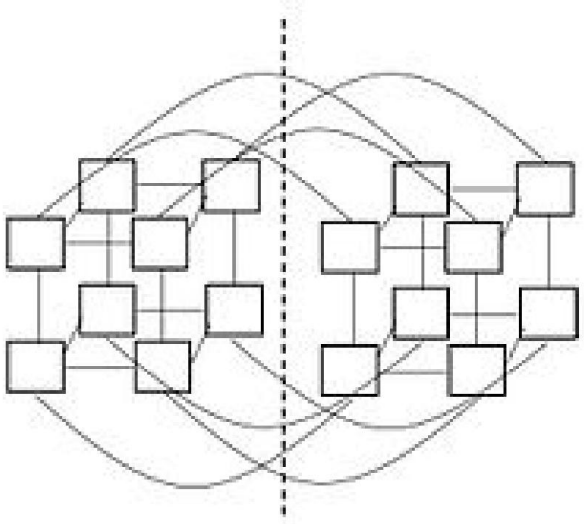
Fig: Bisection of a 2d Mesh Network

What is bisection bandwidth? How to compute?

- Hypercube:

What is bisection bandwidth? How to compute?

- Hypercube:



For Hyper-cube topology with n nodes, n/2 links should be broken to bisect the network, so bisection bandwidth is bandwidth of n/2 links.

Fig: Bisection of a Hypercube Network

Indirect Interconnects

Indirect interconnects

- Indirect interconnects provide an alternative to direct interconnects.
  - In an indirect interconnect, the switches may not be directly connected to a processor.
  - They're often shown with unidirectional links and a collection of processors, each of which has an outgoing and an incoming link, and a switching network.

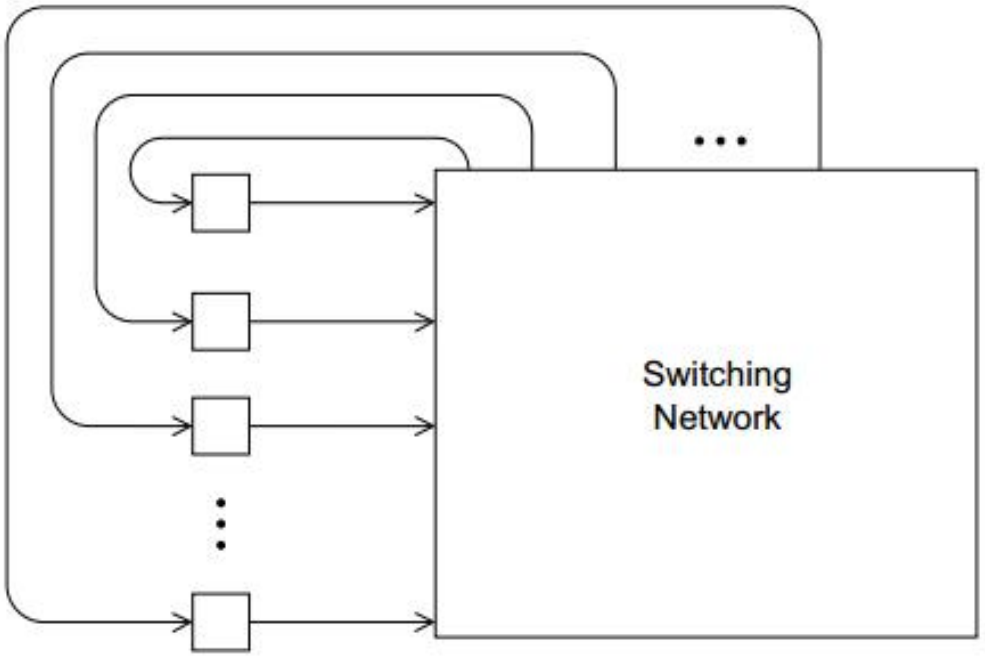


Figure 2.13 : A generic indirect network

Topology

- Internet topologies are not very regular – they grew incrementally.
- Supercomputers have regular interconnect topologies and trade off cost for high bandwidth.
- Nodes can be connected with
  - centralized switch: all nodes have input and output wires going to a centralized chip that internally handles all routing
  - decentralized switch: each node is connected to a switch that routes data to one of a few neighbors

Topology

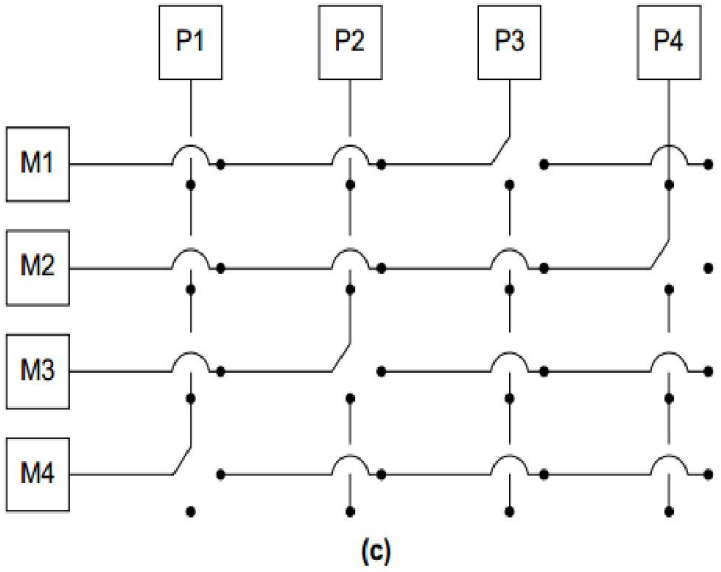


Fig: 2.7 shared-memory crossbar with bidirectional links

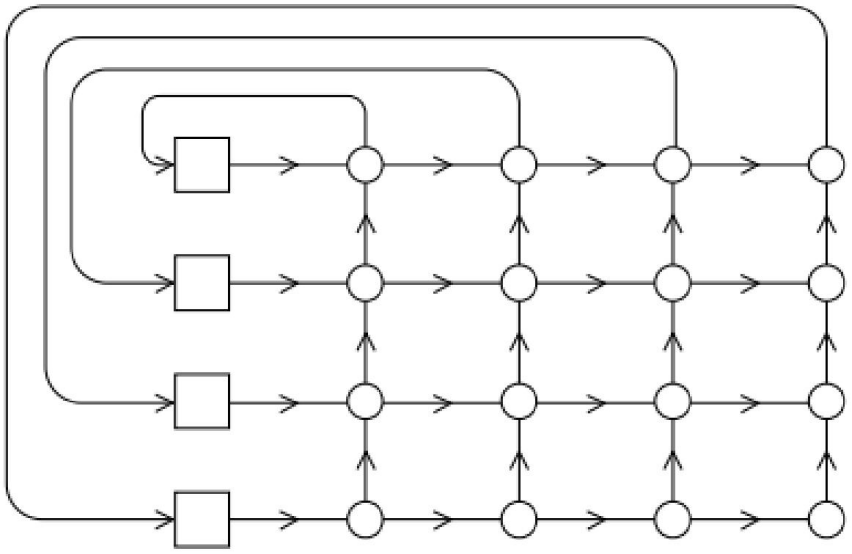


Fig: 2.14 Distributed-memory crossbar with unidirectional links

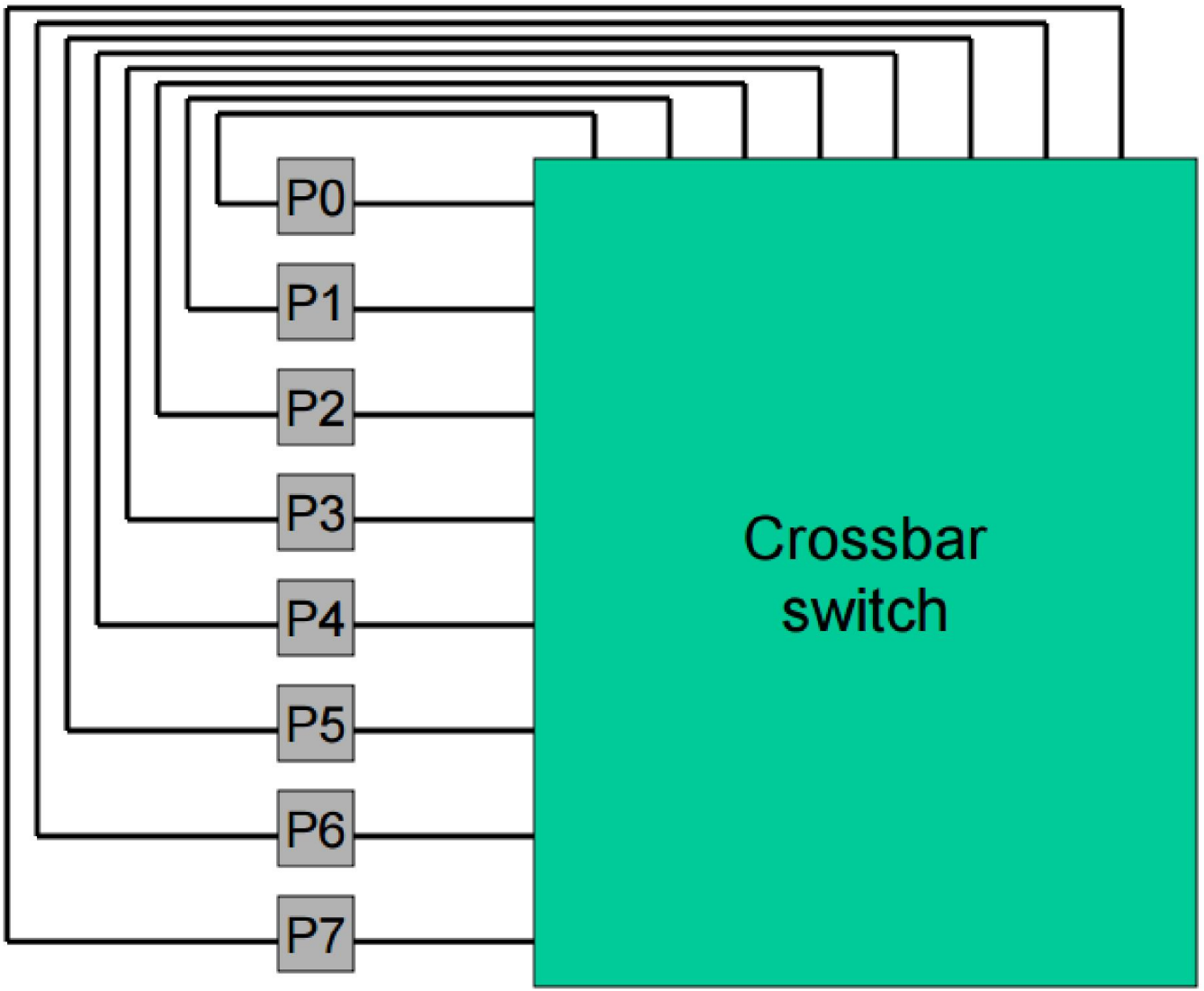
For <u>Distributed-memory crossbar</u> : Notice that as long as two processors don't attempt to communicate with the same processor, all the processors can simultaneously communicate withanother processor.

Indirect Interconnects

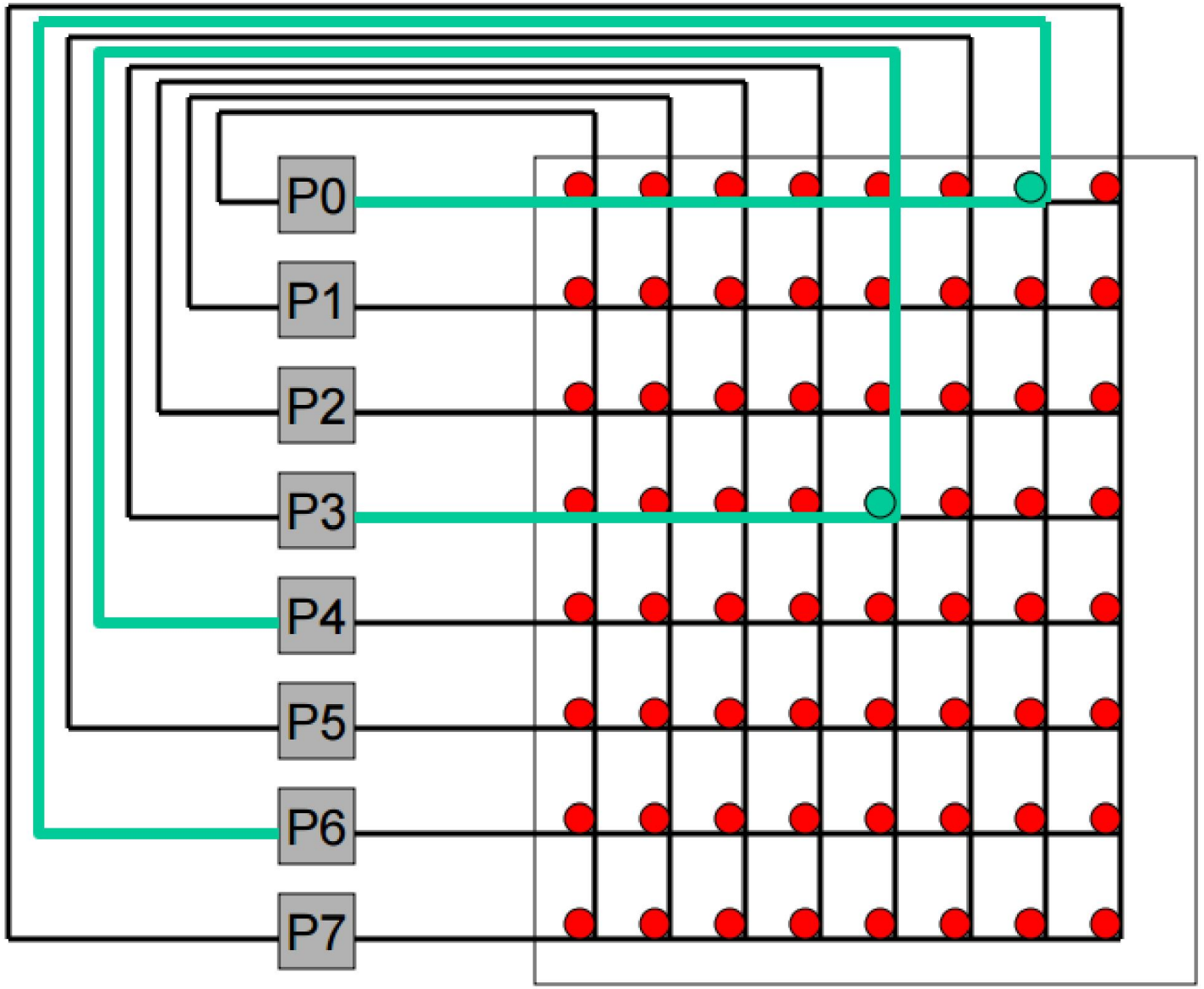
The 2 examples of indirect networks are

- crossbar and
- omega network

Centralised crossbar Switch



Centralised crossbar Switch



Crossbar Properties

- Assuming each node has one input and one output, a crossbar can provide maximum bandwidth: N messages can be sent as long as there are N unique sources and N unique destinations
- Maximum overhead: WN^2 internal switches, where W is data width and N is number of nodes •
- To reduce overhead, use smaller switches as building blocks – trade off overhead for lower effective bandwidth

Omega Network

- Here the switches are 2 X 2 crossbars. Observe that unlike the crossbar, there are communications that cannot occur simultaneously

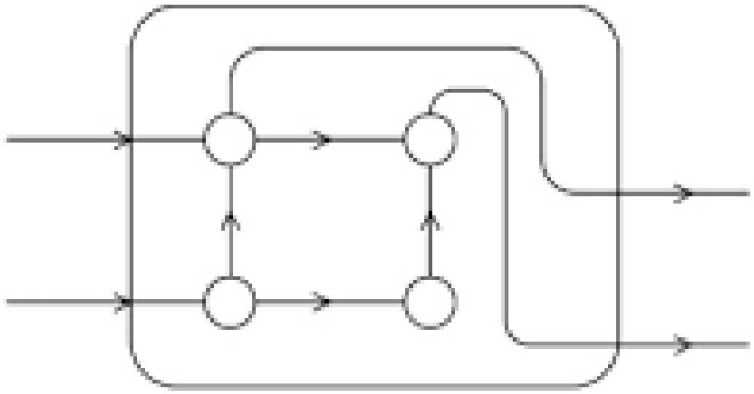
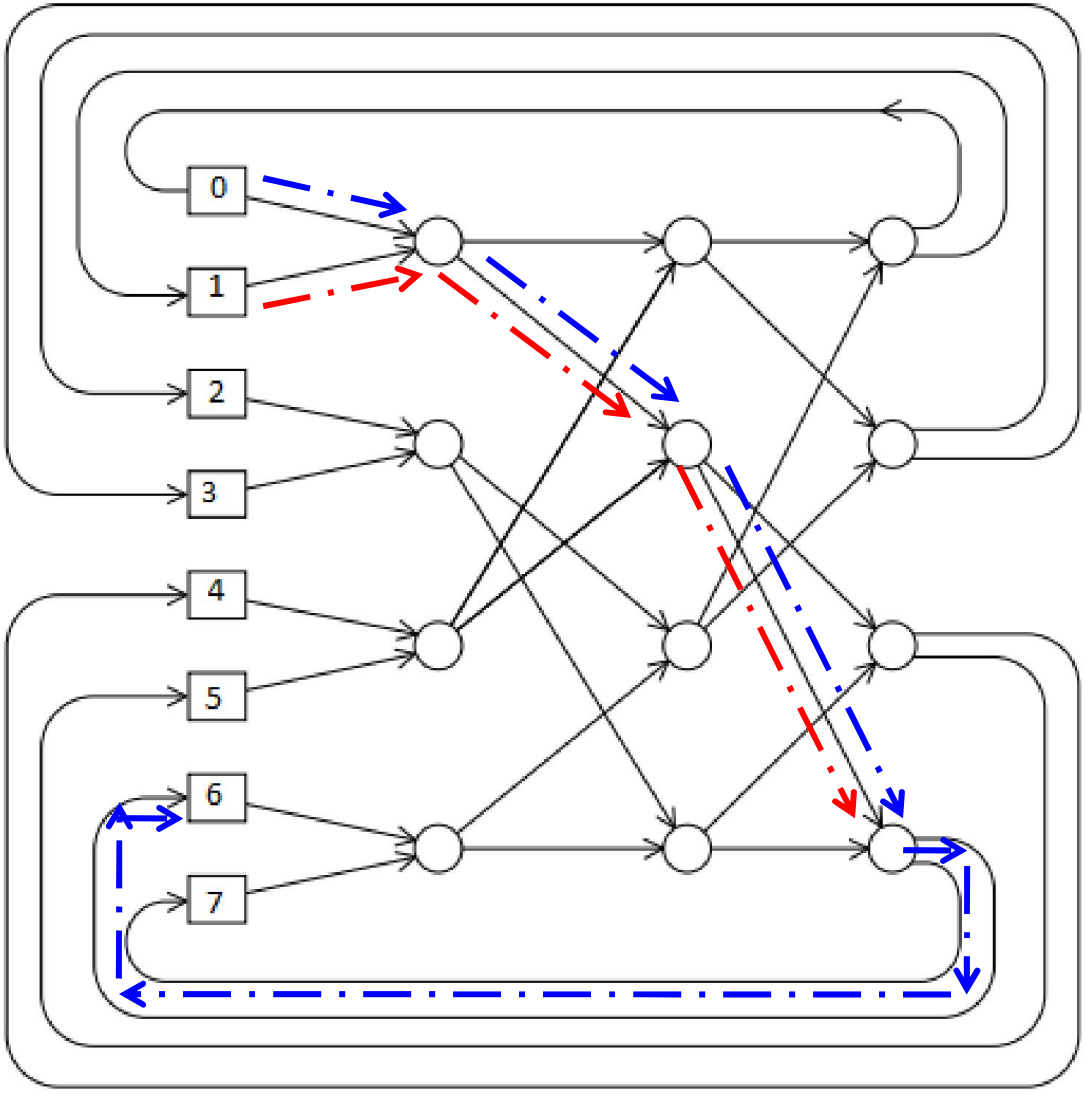


FIGURE 2.16

A switch in an omega network

Switch with Omega Network



Observe that unlike the crossbar, there are **communications that cannot occur simultaneously**. For example, in Figure 2.15 if processor 0 sends a message to processor 6, then processor 1 cannot simultaneously send a message to processor 7. On the other hand, the omega network is less expensive than the crossbar.

The omega network uses $\frac{1}{2}$ p $\log_2(p)$ of the 2 X 2 crossbar switches, so it uses a total of 2p $\log_2(p)$ switches, while the crossbar uses p^2.

Fig. 2.15 An Omega Network

Omega Network Properties

- The switch complexity is now O(N log N) •
- Contention increases: P0 -> P6 and P1 -> P7 cannot happen concurrently (this was possible in a crossbar) •
- To deal with contention, can increase the number of levels (redundant paths) – by mirroring the network, we can route from P0 to P6 via N intermediate nodes, while increasing complexity by a factor of 2.

Bisection Bandwidth

- Split N nodes into two groups of N/2 nodes such that the bandwidth between these two groups is minimum: that is the **bisection bandwidth** •
- Why is it relevant: if traffic is completely random, the probability of a message going across the two halves is $\frac{1}{2}$ – if all nodes send a message, the bisection bandwidth will have to be N/2 •
- The concept of bisection bandwidth confirms that ***the tree network is not suited for random traffic patterns***, but for localized traffic patterns

Latency and Bandwidth

- two figures that are often used to describe the performance of an interconnect :
  1. **Latency**: the time that elapses between the source's beginning to transmit the data and the destination's starting to receive the first byte.
  2. **Bandwidth** : the rate at which the destination receives data after it has started to receive the first byte.

  So if the latency of an interconnect is p seconds and the bandwidth is b bytes per second, then the time it takes to transmit a message of n bytes is

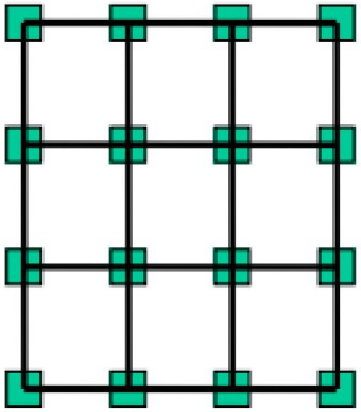
  message transmission time = p + n/b .

Latency and Bandwidth

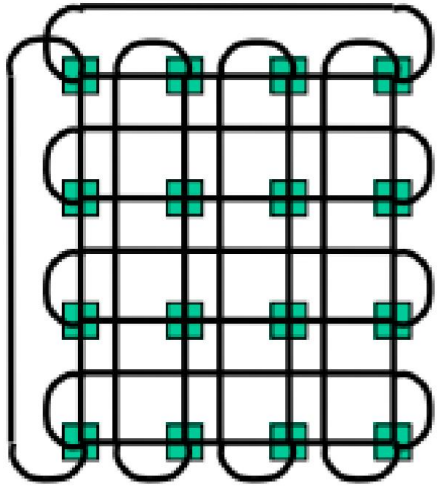
- For example, **latency** is sometimes used to describe total message transmission time. It's also often used to describe the time required for *any fixed overhead involved in transmitting data*.
- For example, if we're sending a message between two nodes in a distributedmemory system, a message is not just raw data. It might include the data to be transmitted, a destination address, some information specifying the size of the message, some information for error correction, and so on.
- So in this setting, latency might be the time it takes to assemble the message on the sending side—the time needed to combine the various parts—and the time to disassemble the message on the receiving side—the time needed to extract the raw data from the message and store it in its destination.

Excercises

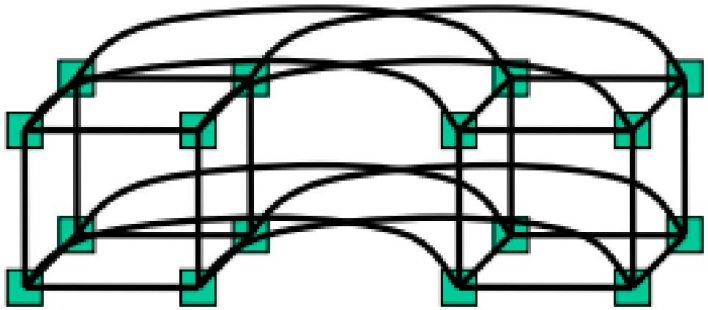
Topology Examples



Grid



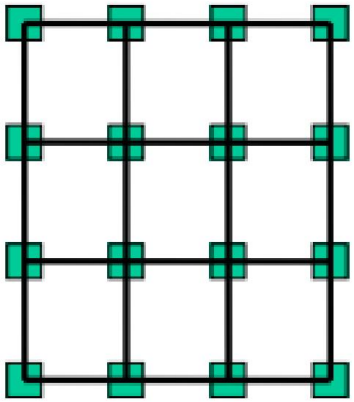
Torus



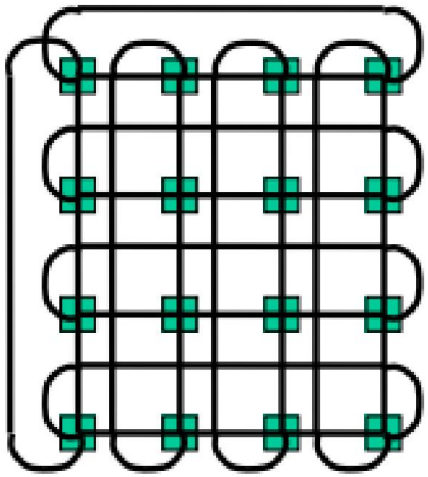
Hypercube

Criteria	Bus	Ring	2Dtorus	6-cube	Fully connected
Performance Bisection bandwidth					
Cost Ports/switch Total links					

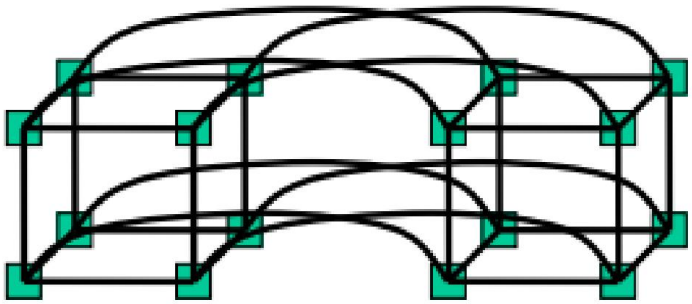
Topology Examples (Solutions)



Grid



Torus



Hypercube

Criteria	Bus	Ring	2Dtorus	6-cube	Fully connected
Performance					
Bisection bandwidth	1	2	16	32	1024
Cost					
Ports/switch		3	5	7	64
Total links	1	128	192	256	2080

k - ary d - cube

- Consider a k-ary d-cube: a d-dimension array with k elements in each dimension, there are links between elements that differ in one dimension by 1 (mod k)
- Number of nodes N = k^d
  - Number of switches :
  - Switch degree :
  - Number of links :
  - Pins per node :
  - Avg. routing distance:
  - Diameter :
  - Bisection bandwidth :
  - Switch complexity :
- Should we minimize or maximize dimension?

k - ary d – cube (Soultions)

- Consider a k-ary d-cube: a d-dimension array with k elements in each dimension, there are links between elements that differ in one dimension by 1 (mod k)
- Number of nodes N = k^d
  - Number of switches : N
  - Switch degree : 2d + 1
  - Number of links : Nd
  - Pins per node : 2wd
  - Avg. routing distance: d(k-1)/2
  - Diameter : d(k-1)
  - Bisection bandwidth : $2wk^{d-1}$
  - Switch complexity : $(2d + 1)^2$

(with no wraparound)

- Should we minimize or maximize dimension?

References

1. Text Book
2. http://www.cs.utah.edu/~rajeev/cs6810/pres/07-6810-23.pdf