VII. Cache Coherence. Interconnection Networks (1)

March 16th, 2009

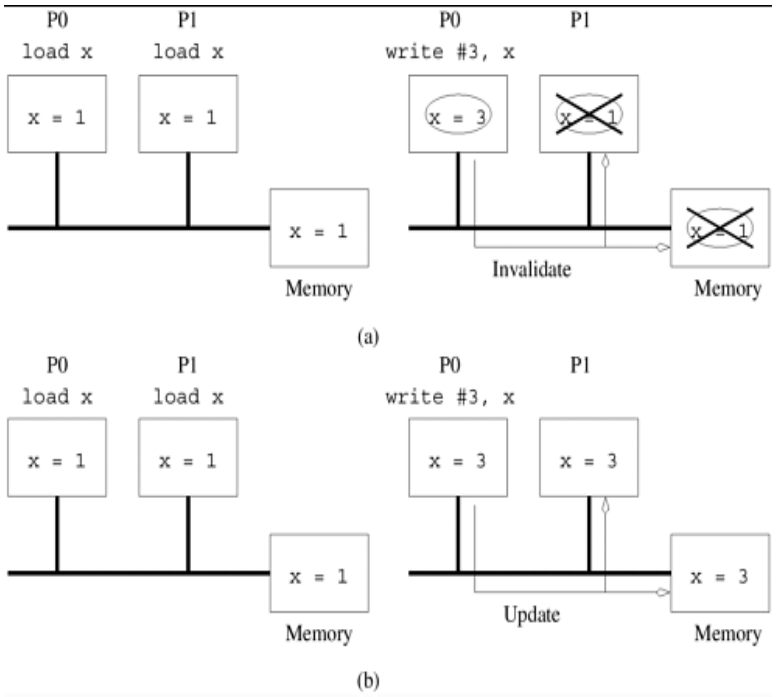
Content

- cache coherence in multiprocessor systems
- Interconnection Networks (1)
  - classification
  - topologies
  - evaluating static and dynamic interconnection networks

Cache Coherence in Multiprocessor Systems

The problem

- Interconnection nets provide basic mechanisms for communicating mess.
- In the case of shared-address-space computers additional hardware is required to keep multiple copies of data consistent with each other
- If there exist two copies of the data (in different caches/memory elements), how do we ensure that different processors operate on these in a manner that follows predefined semantics?
- Example:
  - Two procs P0 & P1 are connected over a shared bus to a globally accessible mem.
  - Both processors load the same variable.
  - There are now three copies of the variable
  - The coherence mechanism must now ensure that all operationsperformed on these copies are serializable



Protocols

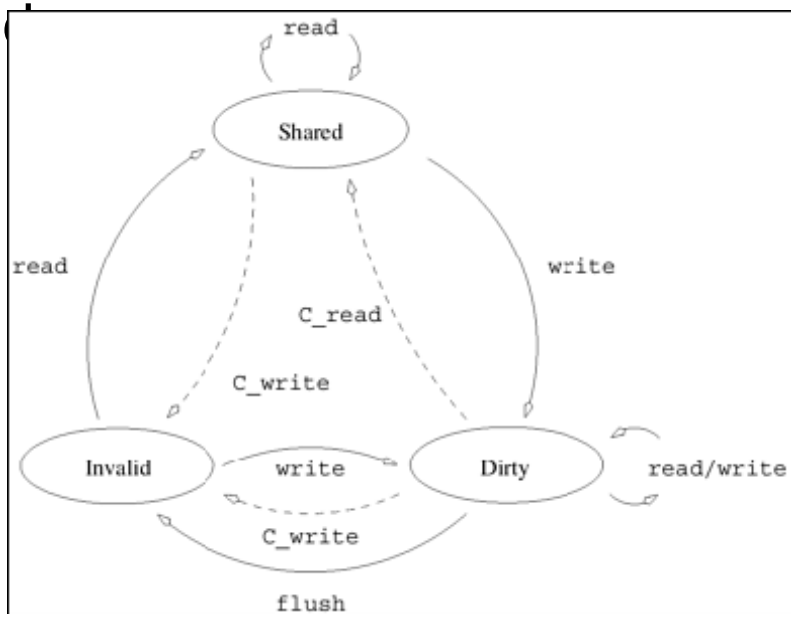
- Update protocol (b)
  - whenever a data item is written, all of its copies in the system are updated.
  - If a processor simply reads a data item once and never uses it, subsequent updates to this item at other processors cause excess overhead in terms of latency at source and bandwidth on the network.
- Invalidate protocol (a)
  - invalidates the data item on the first update at a remote processor
  - subsequent updates need not be performed on this copy.

False sharing

- Refers to the situation in which different processors update different parts of of the same cache-line.
  - although the updates are not performed on shared variables, the system does not detect this.
- In an invalidate protocol
  - when a processor updates its part of the cache-line, the other copies of this line are invalidated.
  - When other processors try to update their parts of the cache-line, the line must actually be fetched from the remote processor.
  - False sharing can cause a cache-line to be ping-ponged between various processors.
- In an update protocol,
  - this situation is slightly better: all reads can be performed locally and the writes must be updated.
  - This saves an invalidate operation that is otherwise wasted.

Three-state coherence protocol

- States labeled invalid, dirty, and share
- Solid lines depict processor actions
- Dashed lines coherence actions.
- For example,
  - When a processor executes a read on an invalid block,
    - the block is fetched and
    - a transition is made from invalid to shared.
  - If a processor does a write on a shared block,
    - the coherence protocol propagates a C_write (a coherence write) on the block.
    - This triggers a transition from shared to invalid at all the other blocks.



Implementation of coherence protocols

- can be carried out using a variety of hardware mechanisms:
  - snoopy systems,
  - directory based systems, or
  - combinations thereof.

Snoopy Cache Systems

- Snoopy caches are typically associated with multiprocessor systems based on broadcast interconnection networks such as a bus or a ring.
- All processors snoop on (monitor) the bus for transactions.
  - This allows the processor to make state transitions for its cache-blocks.
- Extensively studied and used in commercial systems.

1. If different procs operate on different data items, these can be cached.
   1. Once these items are tagged dirty, all subsequent operations can be performed locally on the cache without generating external traffic.
2. If a data item is read by a no. processors, it transitions to the shared state in the cache and all subsequent read operations become local.

- In both cases, the coherence protocol does not add any overhead.
- Disadvantages:
  - If multiple processors read and update the same data item, they generate coherence functions across processors.
  - Since a shared bus has a finite bandwidth, only a constant number of such coherence operations can execute in unit time.

Directory based systems

- the global memory is augmented with a directory that maintains a bitmap representing cache-blocks and the processors at which they are cached.
  - These bitmap entries are sometimes referred to as the presence bits.
- Only processors that hold a particular block (or are reading it) participate in the state transitions due to coherence operations.
  - Note that there may be other state transitions triggered by processor read, write, or flush (retiring a line from cache) but these transitions can be handled locally with the operation reflected in the presence bits and state in the directory.
- If different processors operate on distinct data blocks,
  - these blocks become dirty in the respective caches and
  - all operations after the first one can be performed locally.
- If multiple processors read (but do not update) a single data block,
  - the data block gets replicated in the caches in the shared state and
  - subsequent reads can happen without triggering any coherence overheads.

Distributed directory schemes

- In scalable architectures, memory is physically distributed across processors.
- The corresponding presence bits of the blocks are also distributed.
- Each processor is responsible for maintaining the coherence of its own memory blocks.
  - Since each memory block has an owner its directory location is implicitly known to all processors.
  - When a processor attempts to read a block for the first time, it requests the owner for the block.
  - The owner suitably directs this request based on presence and state information locally available.
  - When a processor writes into a memory block, it propagates an invalidate to the owner, which in turn forwards the invalidate to all procs that have a cached copy of the block.
- Note that the communication overhead associated with state update messages is not reduced.
- Distributed directories permit O(p) simultaneous coherence operations, provided the underlying network can sustain the associated state update messages.
  - From this point of view, distributed directories are inherently more scalable than snoopy systems or centralized directory systems.
  - The latency and bandwidth of the network become fundamental performance bottlenecks for such systems.

Interconnection networks

Interconnection networks

- Provide mechanisms for data transfer between
  - processing nodes or
  - between processors and memory modules.
- A black-box view of an interconnection network consists of n inputs and m outputs.
  - The outputs may or may not be distinct from the inputs.
- Typical interconnection networks are built using links & switches.
- Links:
  - A link corresponds to physical media such as a set of wires or fibers capable of carrying information.
  - For links based on conducting media, the capacitive coupling between wires limits the speed of signal propagation.
  - This capacitive coupling and attenuation of signal strength are functions of the length of the link.

Switch

- An intercon. network consisting of two set of input and output ports.
- Degree: total no. of ports on a switch
- Switches provide a range of functionality
  - Minimal functionality: a mapping from the input to the output ports.
  - Switches may also provide support for
    - internal buffering (when the requested output port is busy),
    - routing (to alleviate congestion on the network), and
    - multicast (same output on multiple ports).
- The mapping provided using a variety of mechanisms based on
  - physical crossbars,
  - multi-ported memories,
  - multiplexor-demultiplexors, and
  - multiplexed buses.
- The cost of a switch is influenced by the cost of
  - the mapping hardware: grows as the square of the degree
  - the peripheral hardware: linearly as the degree
  - the packaging costs: linearly as the number of pins.

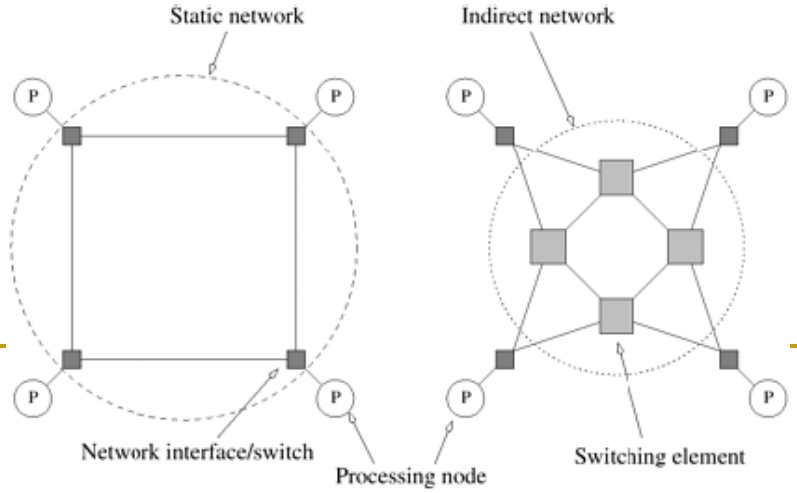
Network interface

- Provides the connectivity between the nodes and the network
- The network interface has input and output ports that pipe data into and out of the network
- Responsibility of
  - packetizing data,
  - computing routing information,
  - buffering incoming and outgoing data for matching speeds of network and processing elements
  - error checking.
- Conventional network interfaces hang off the I/O buses
- Interfaces in tightly coupled parallel machines hang off the memory bus

Since I/O buses are typically slower than memory buses => the latter can support higher bandwidth.

Static vs. dynamic networks

- Static network
  - consist of point-to-point communication links among processing nodes
  - are also referred to as *direct networks*.
- Dynamic networks
  - are built using switches and communication links.
  - communication links are connected dynamically by the switches to establish paths among processing nodes and memory banks.
  - Are also referred to as *indirect networks*.
- Figure:
  - (a) a simple static network of four processing elements or nodes.
    - Each proc.node connected via a netw. interface to 2 nodes in a mesh config
  - (b) a dynamic netw.of 4 nodes connected via a netw.of switches to other nodes

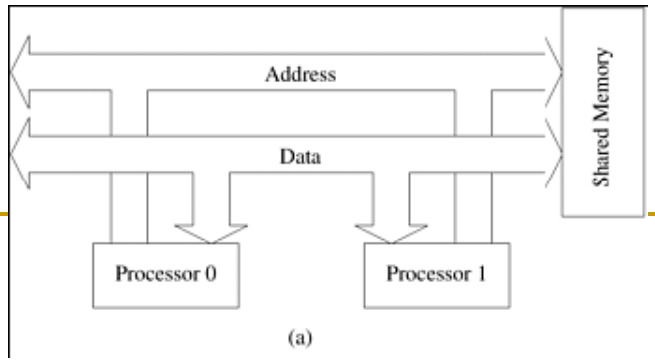


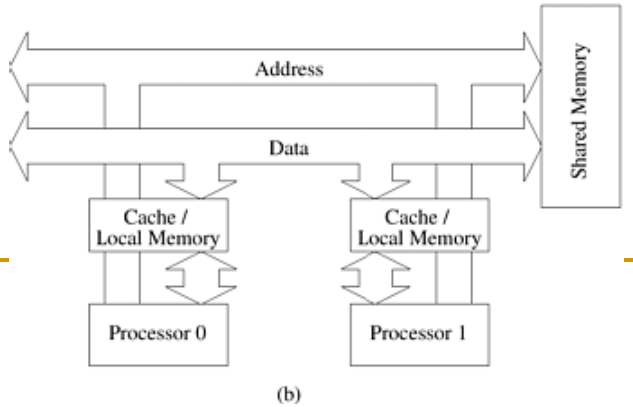
Network Topologies: Bus-based Networks

- A wide variety of network topologies have been used in interconnection networks.
  - These topologies try to trade off cost and scalability with performance
- Simplest network: A bus-based network
  - consisting of a shared medium that is common to all the nodes.
  - Advantages:
    - cost of the network scales linearly as the number of nodes, *p*.
    - the distance between any two nodes in the network is constant
    - ideal for broadcasting information among nodes.
  - Disadvantages:
    - the bounded bandwidth of a bus places limitations on the overall performance of the network as the number of nodes increases.
    - Typical bus based machines are limited to dozens of nodes.
  - Sun Enterprise servers and Intel Pentium based shared-bus multiprocessors are examples of such architectures.

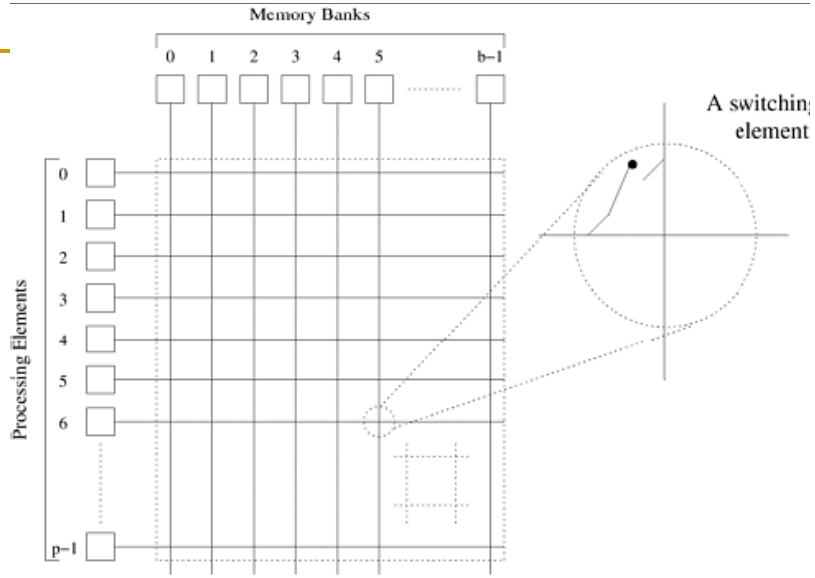
Bus-based interconnects, with caches

- Typical programs: a majority of the data accessed is local to the node
  - For such programs, it is possible to provide a cache for each node.
  - Private data is cached at the node&only remote data is accessed through bus
- Figure (a) :
  - *p* processors sharing a bus to the memory.
  - Assume that each processor accesses *k* data items,
  - Assume each data access takes time *tcycle*,
  - The execution time is lower bounded by *tcycle* x *k p* seconds.
- Figure (b):
  - Assume 50% of the memory accesses (0.5k) are made to local data
  - Assume access time to the private mem identical to the global mem, *tcycle*.
  - Total execution time is lower bounded by 0.5 x *tcycle* x *k* + 0.5 x *tcycle* x *k p*.
  - *p* large, the organization of Figure (b) results in a lower bound
  - This time is a 50% improvement in lower bound on execution time compared to the organization of Figure (a).





Crossbar Networks



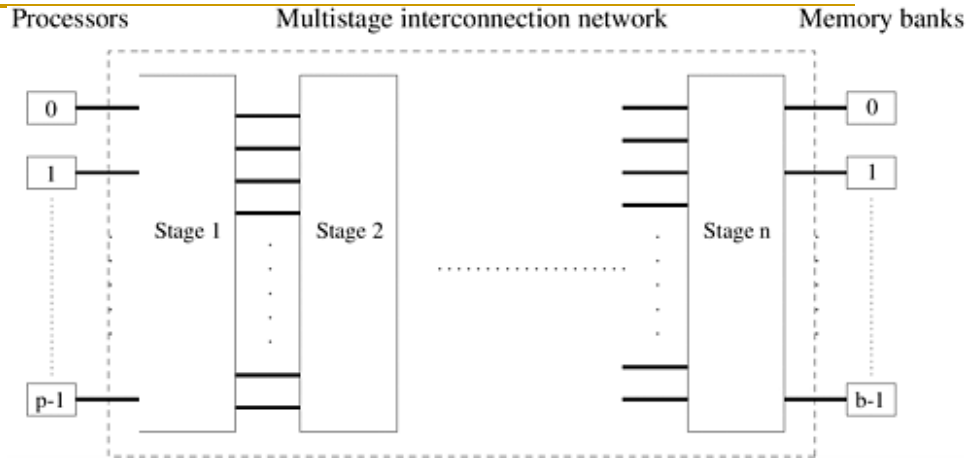
- A simple way to connect p processors to b memory banks
- Employs a grid of switches or switching nodes.
- Non-blocking network
  - in the sense that the connection of a processing node to a memory bank does not block the connection of any other processing nodes to other memory banks.
- Total number of switching nodes required to implement such a network is $O(pb)$.
- Usualy $b < p$; otherwise, at any given time, there will be some processing nodes that will be unable to access any memory banks.

⇒ As the value of p is increased, the complexity (component count) of the switching network grows as $O(p^2)$.

- As the no.processing nodes becomes large, this switch complexity is difficult to realize at high data rates.

⇒ crossbar networks are not very scalable in terms of cost.

Multistage netws



- Remarks:
  - The crossbar interconnect.netw
    - scalable in terms of performance
    - unscalable in terms of cost.
  - The shared bus network is
    - scalable in terms of cost
    - unscalable in terms of performance.
- An intermediate class of networks (multistage interconnection netws) lies between these two extremes.
  - More scalable than the bus in terms of performance and
  - More scalable than the crossbar in terms of cost.
  - The general schematic of a multistage network consisting of p processing nodes and b memory banks is shown

Omega network

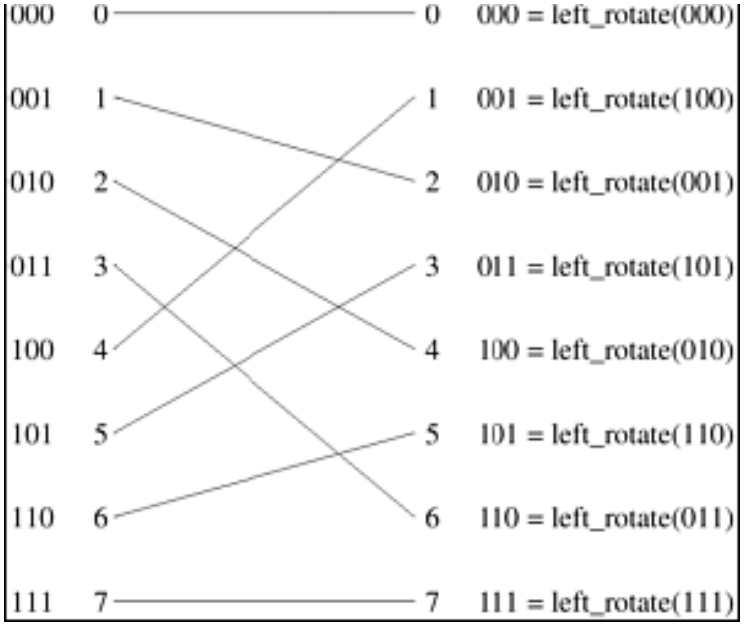
- A commonly used multistage connection network
- Consists of log p stages, where p is the number of inputs (processing nodes) and also the number of outputs (memory banks).
- Each stage consists of an interconnection pattern that connects *p* inputs and *p* outputs
- A link exists between input *i* and output *j* if the following is true:

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

  - This equation represents a left-rotation operation on the binary representation of *i* to obtain *j*.
  - This interconnection pattern is called a perfect shuffle.
- At each stage of an omega network, a perfect shuffle interconnection pattern feeds into a set of *p*/2 switches

Perfect shuffle example and switch mode

- Perfect shuffle for p=8



- Each switch of Omega networks is in one mode.

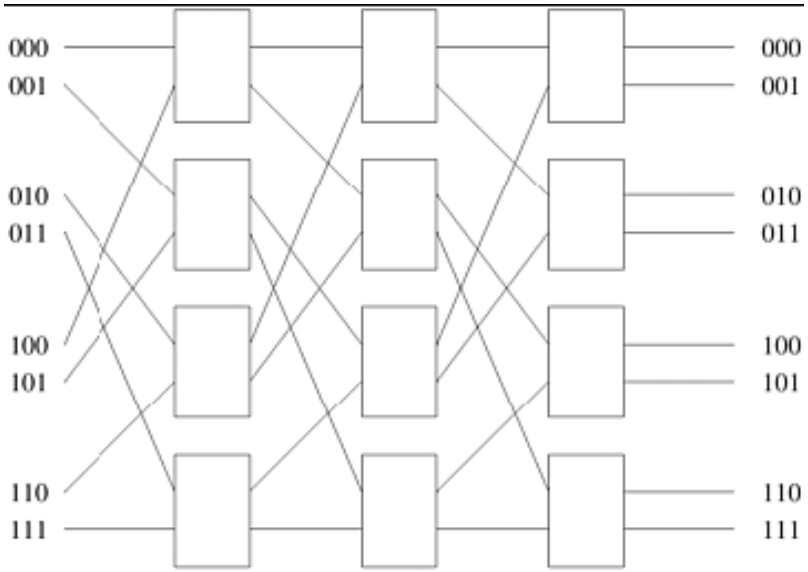
1. the inputs are sent straight through to the outputs - pass-through connection.
2. the inputs to the switching node are crossed over and then sent out - cross-over connection.

Costs and routing in Omega netw

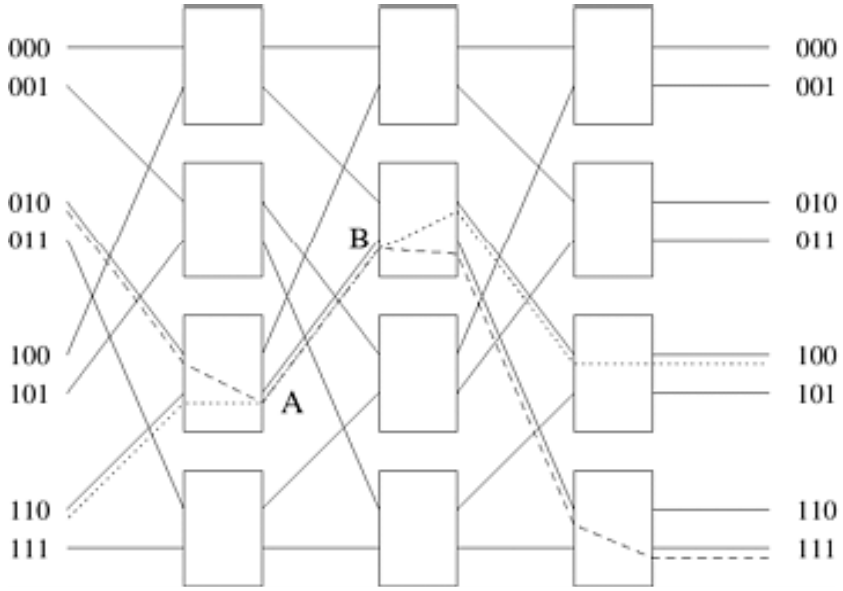
- An omega network has $p/2$ x log p switching nodes, and the cost of such a network grows as $O(p \log p)$.
  - this cost is less than the $\Theta(p^2)$ cost of a complete crossbar network.
- Routing data in an omega network is accomplished as follows:
  - Let *s* be the binary representation of a processor that needs to write some data into memory bank *t.*
  - The data traverses the link to the first switching node.
  - If the most significant bits of *s* and *t* are the same, then the data is routed in pass-through mode by the switch.
  - If these bits are different, the data is routed through in crossover mode.
  - This scheme is repeated at the next switching stage using the next most significant bit.
  - Traversing log p stages uses all log p bits in the binary representations of s and t.

Examples

- A complete omega network connecting 8 inputs and 8 outputs.

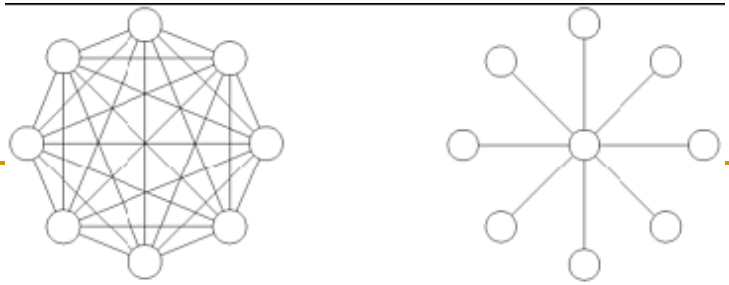


- Access to a memory bank by a processor may disallow access to another memory bank by another processor.
  - Networks with this property are referred to as *blocking networks*
- Example of blocking in omega netw: one of the messages (010 to 111 or 110 to 100) is blocked at link AB.



Completely- & Star – Connected Network

- Completely-connected net:
  - each node has a direct communication link to every other node in the network.
  - ideal in the sense that a node can send a message to another node in a single step
  - static counterparts of crossbar switching networks,
    - the communication between any I/O pair does not block communication between any other pair.
- Star-connected netw,
  - one processor acts as the central processor.
  - Every other proc. has a communication link connecting it to this proc.
  - Similar to bus-based net
    - Communication between any pair of processors is routed through the central processor
  - Central processor: the bottleneck



Linear arrays

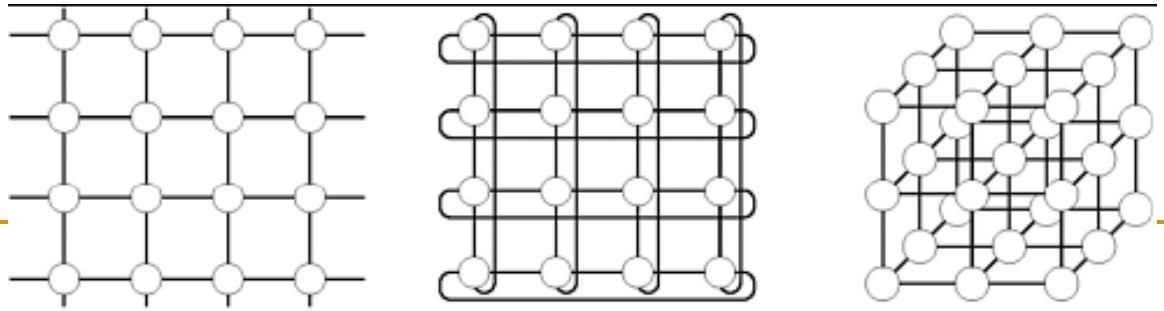
- Due to the large number of links in completely connected networks, sparser networks are typically used to build parallel computers.
- A linear array is a static network in which each node (except the two nodes at the ends) has two neighbors, one each to its left and right.
- A simple extension of the linear array is the ring or a 1- D torus :
  - The ring has a wraparound connection between the extremities of the linear array.
  - In this case, each node has two neighbors.

(a) (b)

2D mesh, 2D tor and 3D mesh

- A two-dimensional mesh:
  - is an extension of the linear array to two-dimensions.
  - Each dimension has sqrt(*p*) nodes with a node identified by a two-tuple (*i, j*).
  - Every node (except periphery) connected to 4 nodes (indices differ in any dim by 1)
  - It can be laid out in 2-D space, making it attractive from a wiring standpoint.
  - A variety of regularly structured computations map very naturally to a 2-D mesh.
  - ⇒ 2-D meshes were often used as interconnects in parallel machines.
- 2D meshes can be augmented with wraparound links to form 2D tori (b).
- The three-dimensional cube
  - Is a generalization of the 2-D mesh to three dimensions, as illustrated in (c).
  - Each node element in a 3-D cube (exception on periphery), is connected to six other nodes, two along each of the three dimensions.
  - For example, in the Cray T3E
  - A variety of physical simulations (for example, 3-D weather modeling, structural modeling, etc.) can be mapped naturally to 3-D network topologies.

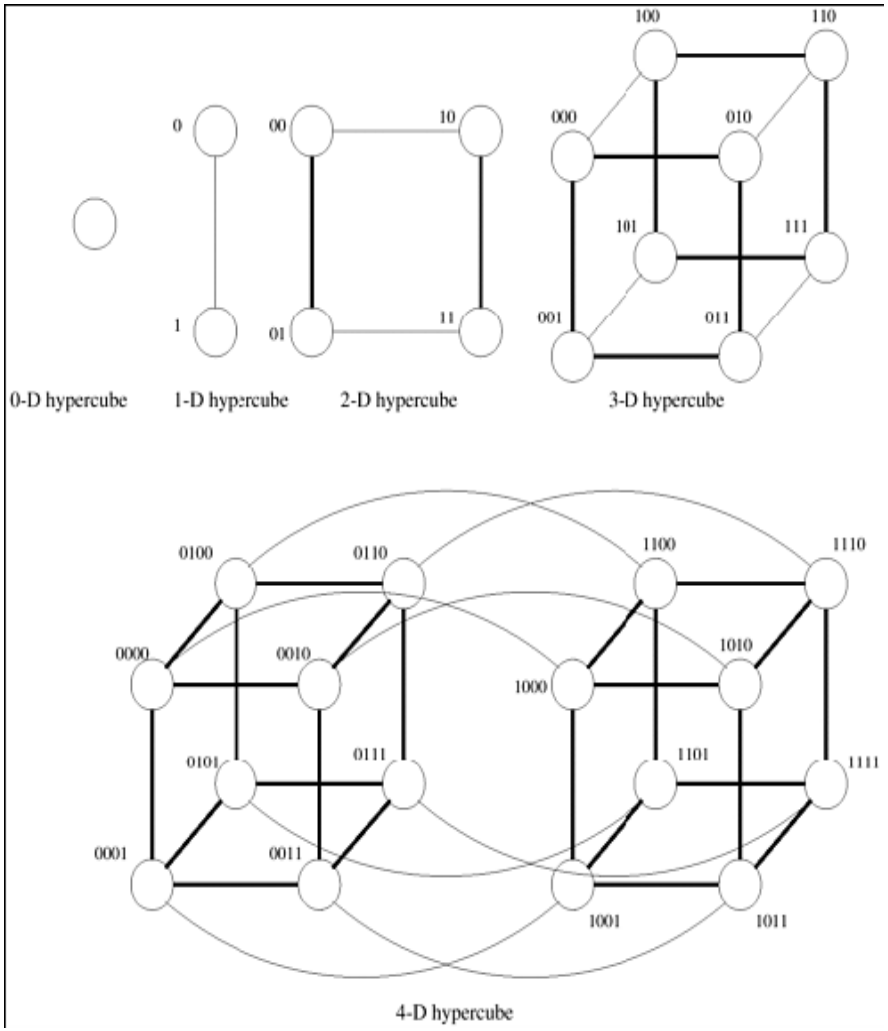
=> 3-D cubes are used commonly in interconnection networks for parallel computers



k-d mesh & hypercube

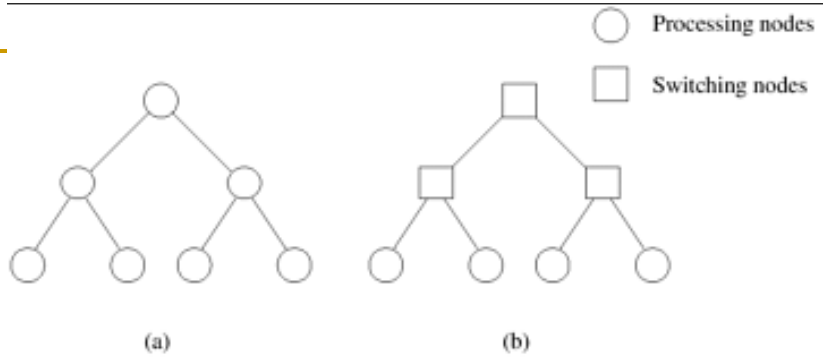
- *k-d* mesh: topology consisting of *d* dimensions with *k* nodes along each dimension.
  - *d*=1: linear case
  - *d*=2: mesh
  - *d*=3: 3D cube
  - *k*=2: hypercube
- Hypercube topology: two nodes along each dimension and log p dimensions.
- Construction:
  - A zero-dimensional hypercube consists of 2^0, i.e., one node.
  - A one-dimensional hypercube is constructed from two zero-dimensional hypercubes by connecting them.
  - A two-dimensional hypercube of four nodes is constructed from two one-dimensional hypercubes by connecting corresponding nodes.
  - In general a *d*-dimensional hypercube is constructed by connecting corresponding nodes of two (*d* - 1) dimensional hypercubes.

Construction of hypercube and numbering scheme



- Numbering scheme:
  - A numbering of two subcubes of p/2 nodes => derive a numbering scheme for the cube of p nodes by prefixing the labels of one of the subcubes with "0" and the labels of the other with a "1".
- Useful property that the minimum distance between two nodes is given by the no. of bits that are different in the two labels.
  - Example: nodes labeled 0110 and 0101 are 2 links apart: they differ at two bit positions.
  - Property useful for deriving a no. parallel algs for the hypercube architecture.

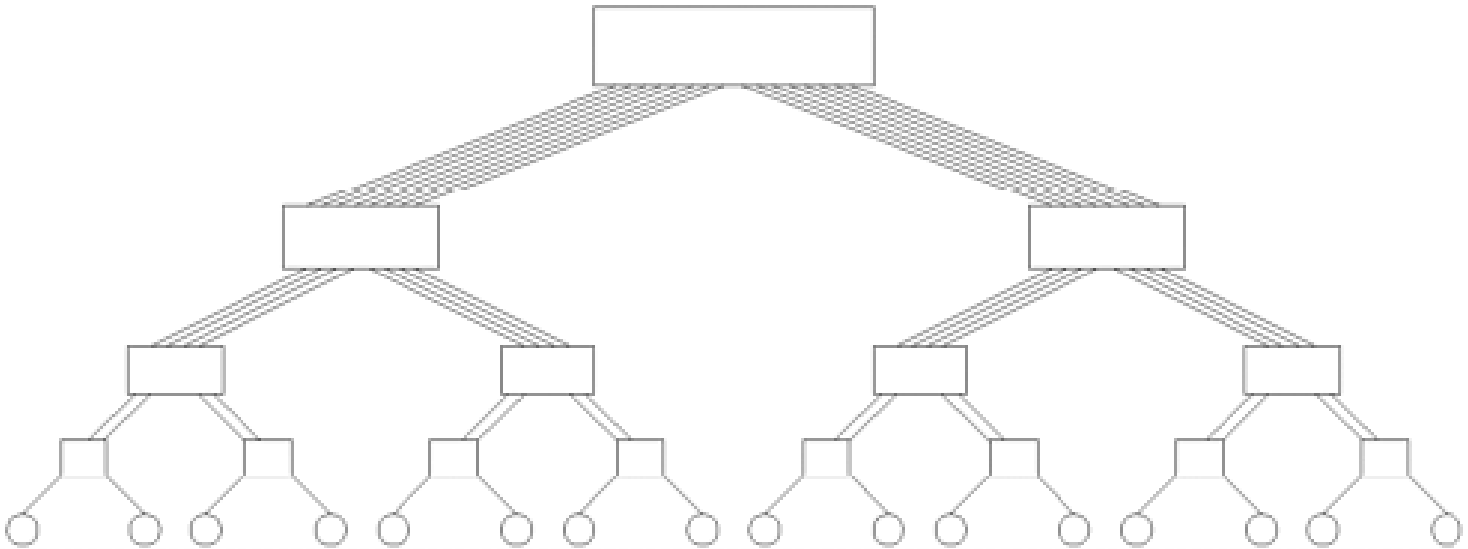
Tree-based networks



- Tree n.: one in which there is only one path between any pair of nodes
- Linear arrays & star-connected net. are special cases of tree networks.
- Figure: networks based on complete binary trees.
- Static tree networks have a processing element at each node of the tree (in (a)).
- In a dynamic tree network, nodes at intermediate levels are switching nodes and the leaf nodes are processing elements (in (b)).
- To route a message in a tree:
  - the source node sends the message up the tree until it reaches the node at the root of the smallest subtree containing both the source and destination nodes.
  - Then the mess is routed down the tree towards the destination node

Fat tree

- Tree networks suffer from a communication bottleneck at higher levels of the tree.
  - For example, when many nodes in the left subtree of a node communicate with nodes in the right subtree, the root node must handle all the messages.
- This problem can be alleviated in dynamic tree networks by increasing the no. communication links and switching nodes closer to the root (fat tree)



Criteria in evaluating static interconnection networks

1. Diameter
2. Connectivity
3. Bisection width
4. Channel capacity
5. Bisection bandwidth
6. Cost

1. Diameter

- The diameter of a network is the maximum distance between any two processing nodes in the network.
- The distance between two processing nodes is defined as the shortest path (in terms of number of links) between them.
- Examples:
  - Completely-connected network is one,
  - Star-connected network is two.
  - Ring network is floor(p/2).
  - Two-dimensional mesh without wraparound connections: 2(sqrt(p)-1)
  - Wraparound mesh is 2floor(sqrt(p)/2).
  - Hypercube-connected network is log p
  - Complete binary tree is 2 log((p + 1)/2)

2. Connectivity

- The connectivity of a network is a measure of the multiplicity of paths between any two processing nodes.
- A network with high connectivity is desirable, because it lowers contention for communication resources.
- One measure of connectivity:
  - The minimum no. of arcs that must be removed from the network to break it into two disconnected networks.
  - This is called the *arc connectivity* of the network.
  - Examples:
    - one for linear arrays, as well as tree and star networks.
    - two for rings and 2-D meshes without wraparound,
    - four for 2-D wraparound meshes, and
    - *d* for *d*-dimensional hypercubes.

3. Bisection width

- Defined as the minimum number of communication links that must be removed to partition the network into two *equal halves*.
- Examples:
  - ring is two,
  - two-dimensional *p*-node mesh without wraparound connections is sqrt(*p*) and with wraparound connections is 2sqrt(p).
  - tree and a star is one,
  - completely-connected network of *p* nodes is $p^2/4$.
  - hypercube is *p*/2 (from its construction)

4. Channel width, rate & bandwidth

- Channel width:
  - Defines as the number of bits that can be communicated simultaneously over a link connecting two nodes
  - Equal to the no. of physical wires in each communication link.
- Channel rate:
  - The peak rate at which a single physical wire can deliver bits.
- Channel bandwidth:
  - The peak rate at which data can be communicated between the ends of a communication link
  - The product of channel rate and channel width.

5. Bisection bandwidth

- defined as the minimum volume of communication allowed between any two halves of the network.
- the product of the bisection width and the channel bandwidth.
- sometimes referred to as cross-section bandwidth.
- can be used as a measure of cost
  - it provides a lower bound on the area in a 2D packaging or the volume in a 3D packaging.
  - If is *w*, the lower bound on the area
    - in a 2D packaging is $\Theta(w^2)$,
    - in a 3D packaging is $\Theta(w^{3/2})$.

6. Cost

- can be defined in terms of the no. of communication links or the no. of wires required by the network.
- Examples:
  - Linear arrays and trees use only p - 1 links to connect p nodes.
  - A d-dimensional wraparound mesh has dp links.
  - A hypercube-connected network has (p log p)/2 links.

A summary of static network topologies

Network	Diameter	Bisection Width	Arc Connectivity	Cost (No. of links)
Completely-connected	1	$p^2/4$	$p - 1$	$p(p - 1)/2$
Star	2	1	1	$p - 1$
Complete binary tree	$2 \log((p + 1)/2)$	1	1	$p - 1$
Linear array	$p - 1$	1	1	$p - 1$
2-D mesh, no wraparound	$2(\text{sqrt}(p)-1)$	$\text{sqrt}(p)$	2	$2(p-\text{sqrt}(p))$
2-D wraparound mesh	$2\text{floor}(\text{sqrt}(p)/2)$	$2\text{sqrt}(p)$	4	$2p$
Hypercube	$\log p$	$p/2$	$\log p$	$(p \log p)/2$
Wraparound k-ary d-cube	d floor(k/2)	$2k^{d-1}$	$2d$	dp

Evaluating Dynamic Interconnection Networks

- A no. of evaluation metrics for dynamic networks follow from the corresponding metrics for static networks.
- Since a message traversing a switch must pay an overhead, it is logical to think of each switch as a node in the network, in addition to the processing nodes.

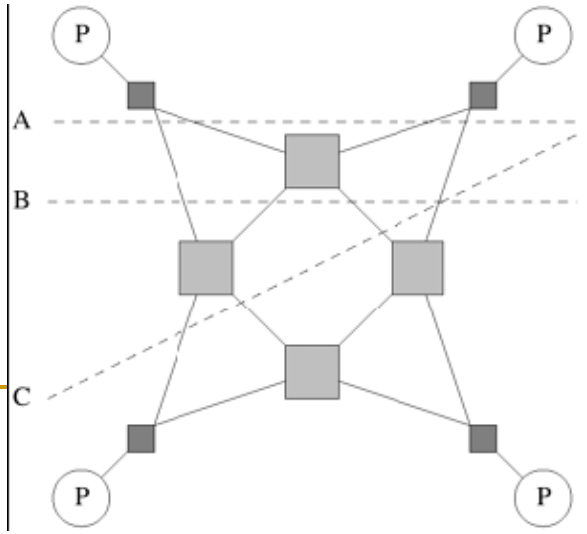
1. Diameter of the network
   - can now be defined as the maximum distance between any two nodes in the network (processing or switching)
2. Connectivity of a dynamic network can be defined in terms of node or edge connectivity.
   - The node connectivity is the min.no.nodes that must fail (be removed from the network) to fragment the network into two parts.
   - We should consider only switching nodes (as opposed to all nodes).
   - The arc connectivity of the network can be similarly defined as the minimum number of edges that must fail (be removed from the network) to fragment the network into two unreachable parts.

Evaluating Dynamic Interconnection Networks – bisection width

- We consider any possible partitioning of the *p* processing nodes into two equal parts.
  - For each such partition, we select an induced partitioning of the switching nodes s.t. the no. edges crossing this partition is minimized.
- The min.no. of edges for any such partition is the bisection width of the dynamic network.
- Another intuitive way of thinking of bisection width:
  - in terms of the min. no. edges that must be removed from the network so as to partition the network into two halves with identical no.of processing nodes.

- Example:
  - three bisections, A, B, and C, each of which partitions the network into 2 groups of 2 processing nodes each.
  - Each partition results in an edge cut of four.

=> Conclude that the bisection width of this graph is four.



Cost of dynamic network topologies

- Determined by the link cost & switch cost.
- Typical dynamic netws: the degree of a switch is const.
- The no.of links and switches is asymptotically identical.
- Switch cost exceeds link cost => the cost of dynamic networks is often determined by the number of switching nodes in the network.

Network	Diameter	Bisection Width	Arc Connectivity	Cost (No. of links)
Crossbar	1	p	1	p^2
Omega Network	log p	p/2	2	p/2
Dynamic Tree	2 log p	1	2	p - 1