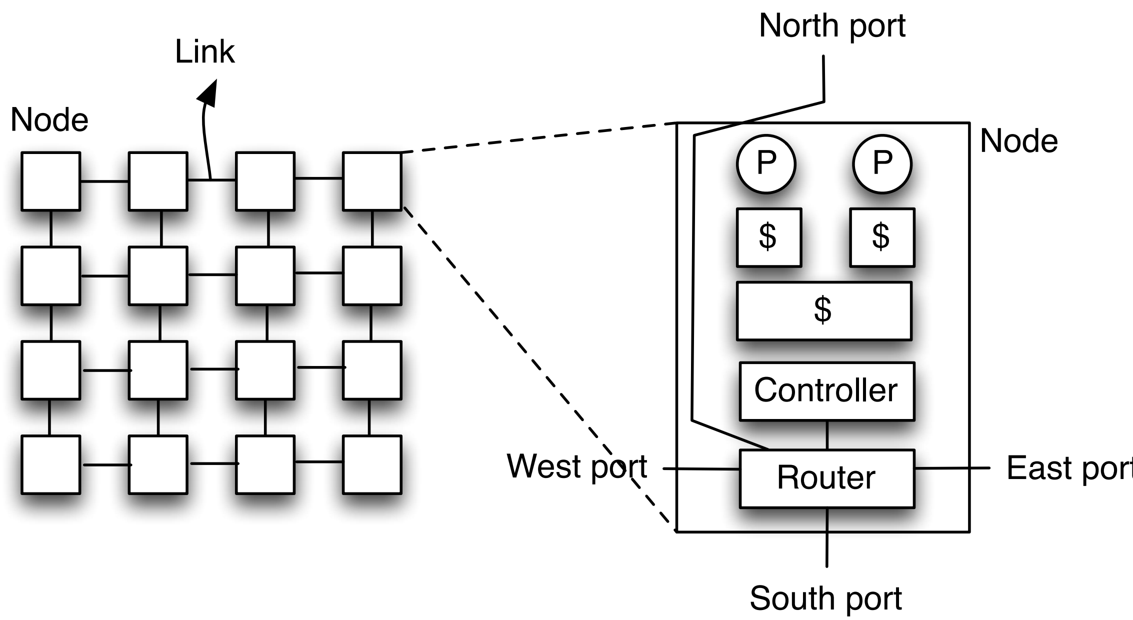
Chapter 12
Interconnection Network

Introduction

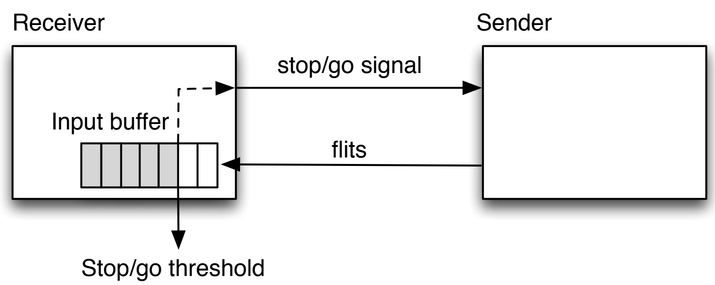
- Interconnection network implements the coherence mechanism that we have discussed so far
  - Sending requests from one node to another
  - Sending data replies from one node to another
- Characteristics of communication in DSM
  - Latency must be low
  - Bandwidth must be high
  - Message characteristics
    - 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)

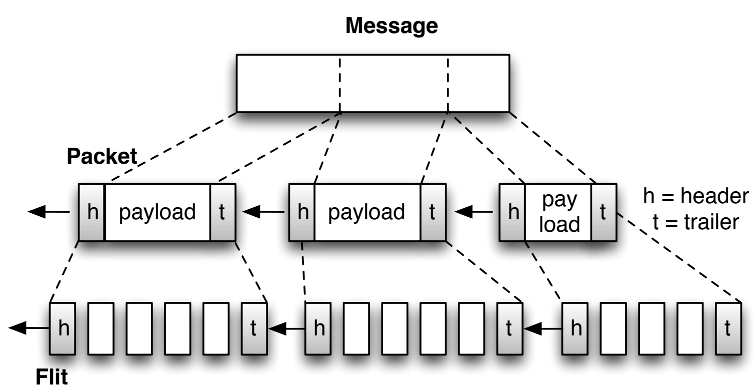
Nodes and Links



- A node encapsulates
  - one or more processors + memory hierarchy
  - communication controller
  - Router, which interconnects a node with other nodes

Link and Channel





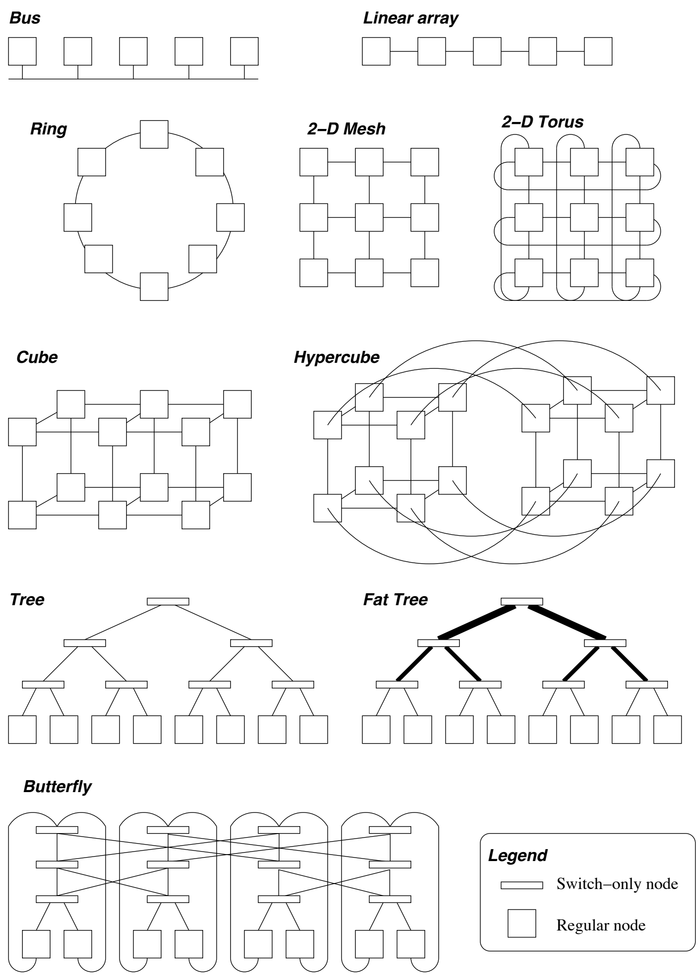
- Link = set of wires interconnecting a pair of nodes
  - May be unidirectional or bidirectional
  - Unit of transfer: flow control unit (flit)
  - Flit size determined by link width, latency, and amount of buffering
  - Channel = link + sender + receiver
- Flow Control
  - Receiver sends stop or go signal depending on available buffer space
  - Sender reacts by stopping/resuming flit transmission
- A message transmitted over a channel is broken into packets
  - A packet has a header, trailer, and payload
  - Packet broken up into flits

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
  - Bisection bandwidth: the minimum number of links cut when the network is partitioned into two equal halves

Popular Network Topologies



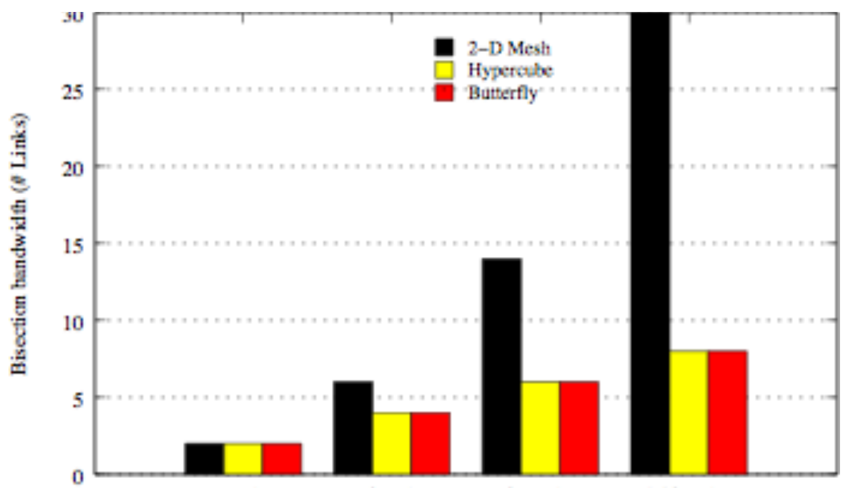
Topology families

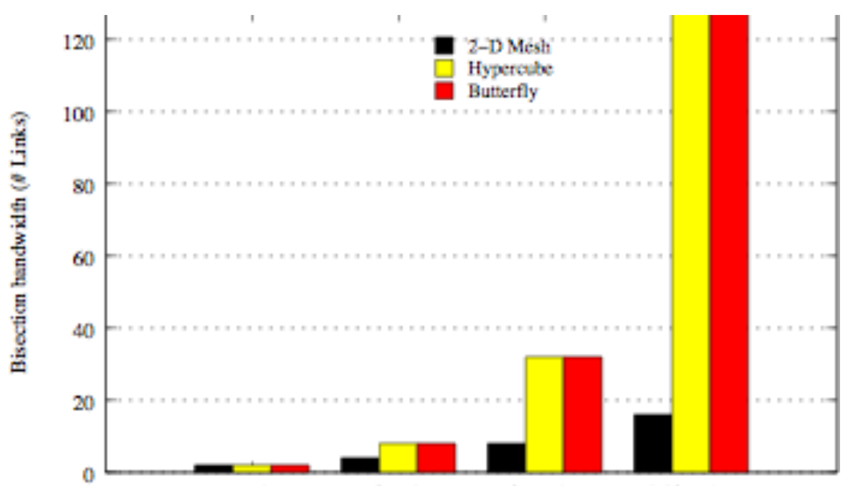
- k-ary d-dimensional mesh
  - d = dimensions
  - k = #nodes/dimension
  - Hypercube = binary d-dimensional mesh
- k-ary tree
- butterfly

Comparison of Topologies

Topology	**Diameter**	**Bisection BW**	**#Links**	**Degree**
Linear array	p-1	1	p-1	2
Ring	$\frac{p}{2}$	2	p	2
2-D Mesh	$2(\sqrt{p}-1)$	$\sqrt{p}$	$2\sqrt{p}(\sqrt{p}-1)$	4
Hypercube	$log_2 p$	$\frac{p}{2}$	$\frac{p}{2} \times log_2 p$	$log_2 p$
k-ary d Mesh	$d(k-1)$	k^{d-1}	$dk^{d-1}(k-1)$	$2d$
k-ary Tree	$2 \times log_k p$	1	$k(p-1)$	$k+1$
k-ary Fat Tree	$2 \times log_k p$	$\frac{p}{2}$	$k(p-1)$	$k+1$
Butterfly	$log_2 p$	$\frac{p}{2}$	$2p \times log_2 p$	4

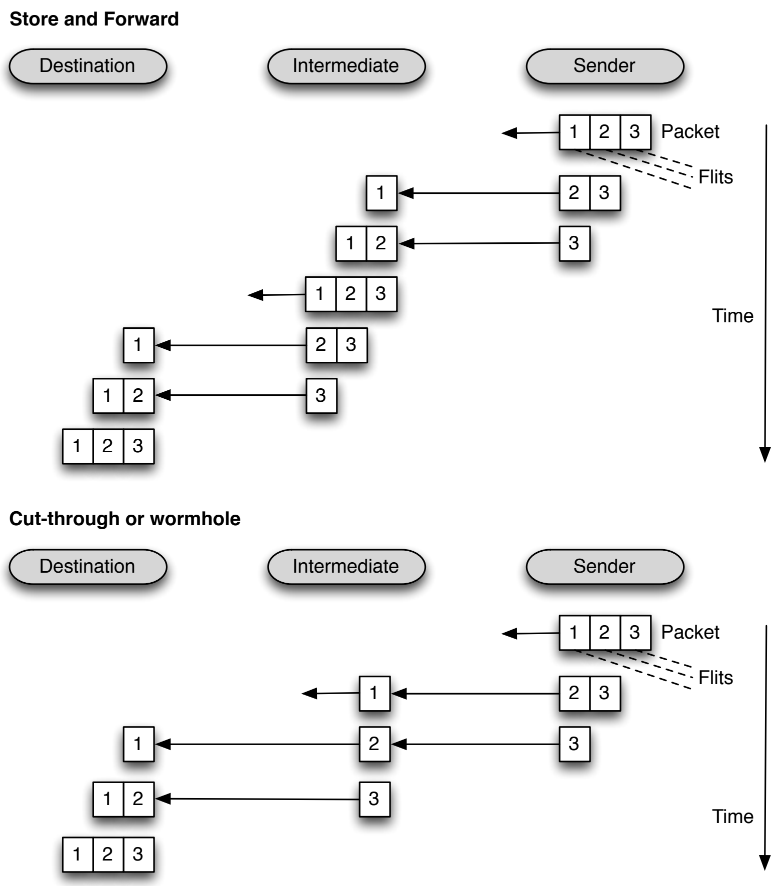
Mesh: Increasing k vs. increasing d





- When more nodes need to be added, is it better to increase the arity (k), or increase the dimension (d)?
  - 2-D mesh vs. hypercube vs. butterfly
- Diameter increases slower in hypercube & butterfly
- Bisection bandwidth increases faster in hypercube and butterfly
- Unfortunately, routers in hypercube are more expensive (higher degrees); while butterfly needs a lot of routers

Routing policy



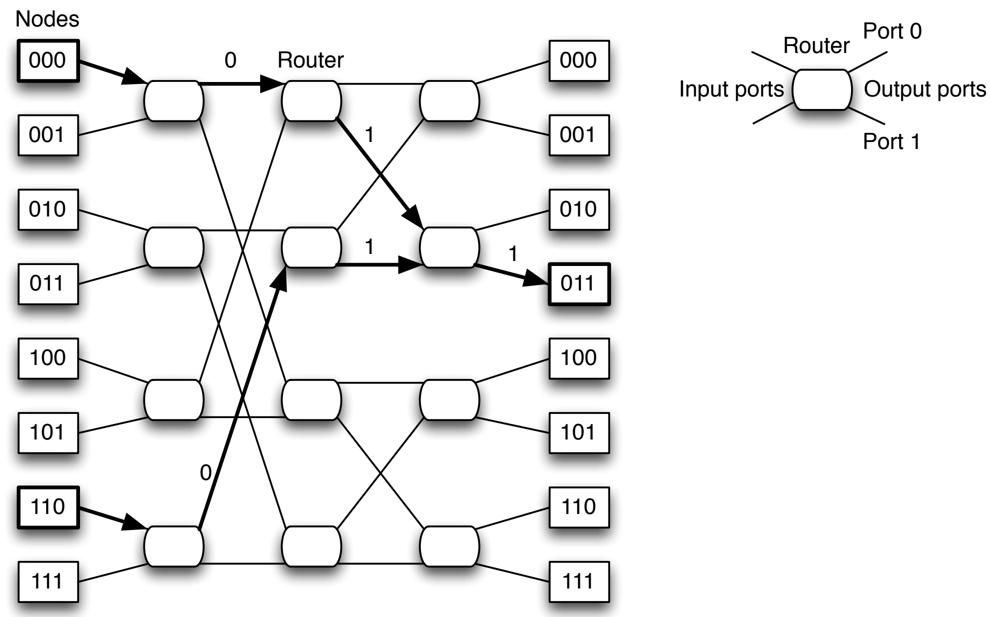
- How to send a packet over the network?
- Store&Forward:
  - Send all flits over a link, then over another, and so on
  - T = H * (Tr+ L/B)
  - H is distance, Tr is router delay, L is packet length, B is channel bandwidth
- Cut-through:
  - Send flits in pipeline over the links
  - T = H * Tr + L/B
- Implication: cut-through routing reduces the importance of topology

Routing Policy

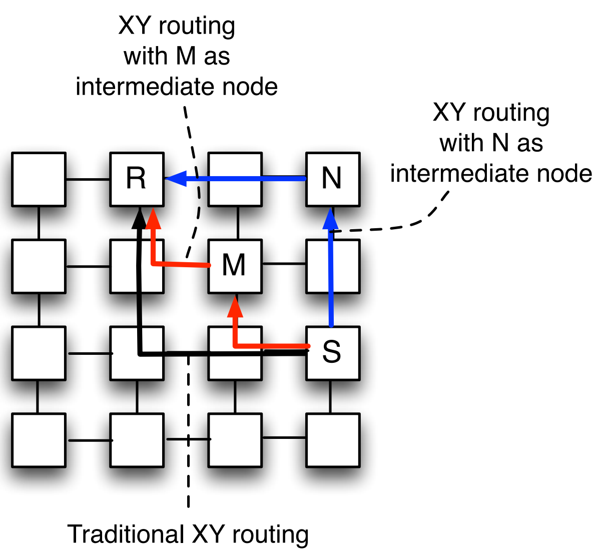
- Path diversity
- Minimal vs. non-minimal
  - Minimal = each packet travels the least number of hops
- Deterministic vs. non-deterministic
  - Deterministic = each sender-dest pair uses a single path for all messages
- Adaptive vs. non-adaptive
  - Adaptive = message can change path as it is being transmitted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

Topology-Dependent Routing

- In some cases, topology imposes routing
- E.g. butterfly: minimal, deterministic, deadlock-free, non-adaptive
  - Lack of path diversity reduces usable bandwidth on adversarial traffic patterns
- Destination ID is directly used to determine output port in each router along the path



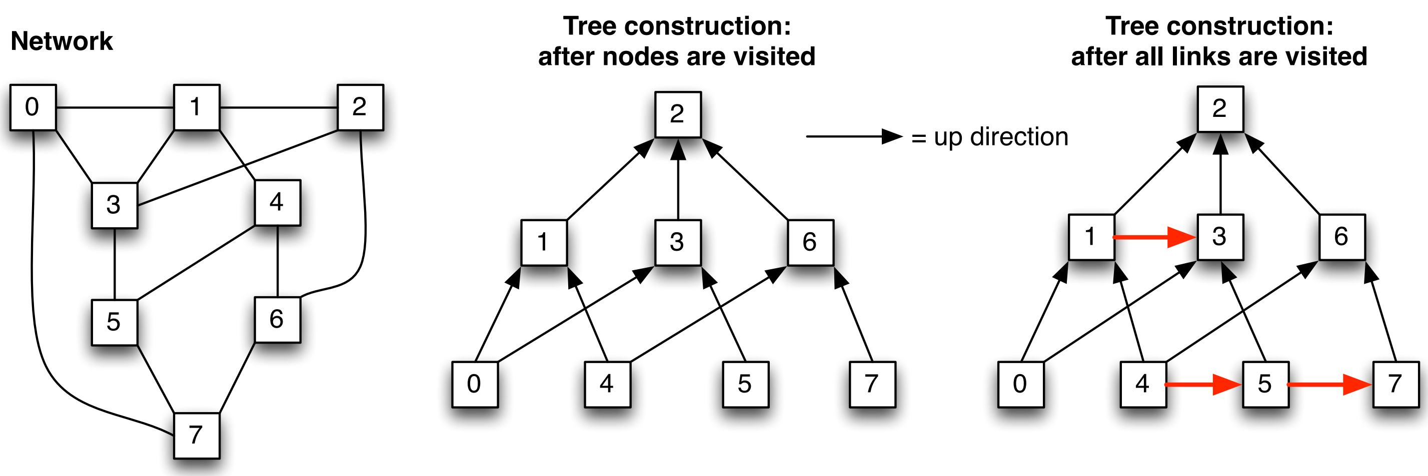
Routing in Mesh/Torus



- Dimension-ordered
  - X-Y: Go in X dimension first, and then Y dimension until destination is reached
  - Y-X is possible as well
  - Simple but lacking path diversity
- Valiant routing
  - Select a random intermediate node
  - Can be minimal or non-minimal
  - Improved path diversity

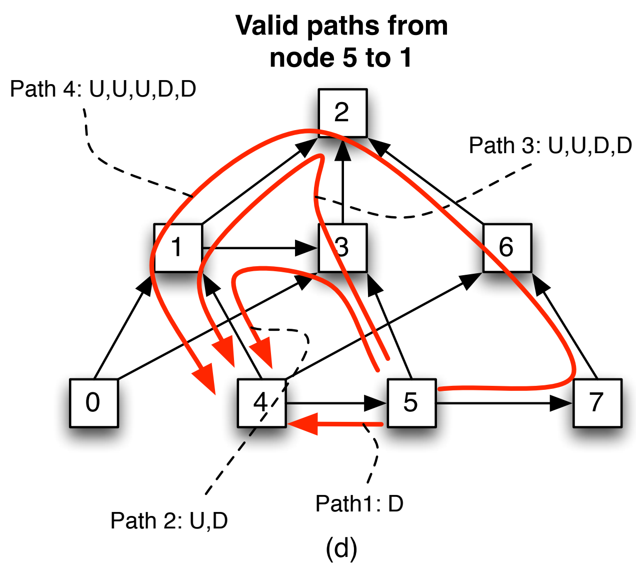
Up*/Down* Routing

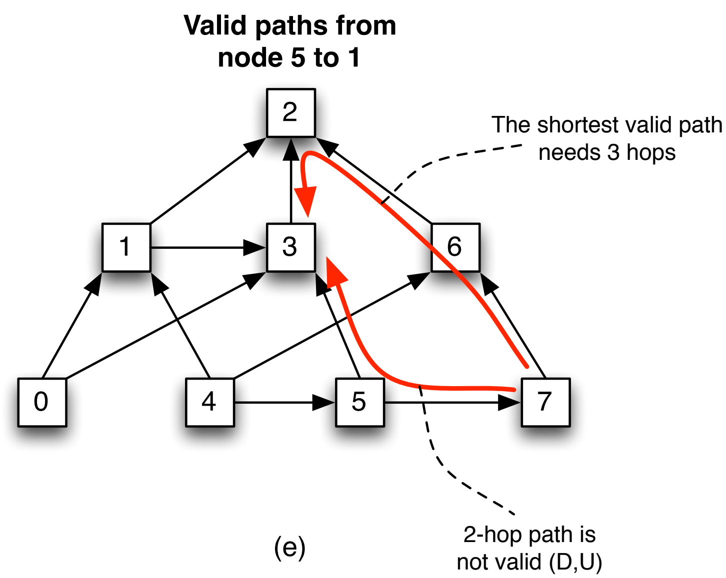
- Enables routing in any topology
  - Step 1: Start by choosing a root
  - Step 2: Perform breadth-first traversal for next hop, all nodes visited become children, repeat until all nodes are visited. Each link connecting child to parent is an "up" link
  - Step 3: For each level, if two nodes are linked, an "up" link is from a lower-numbered node to a higher-numbered node



Up*/Down* Routing

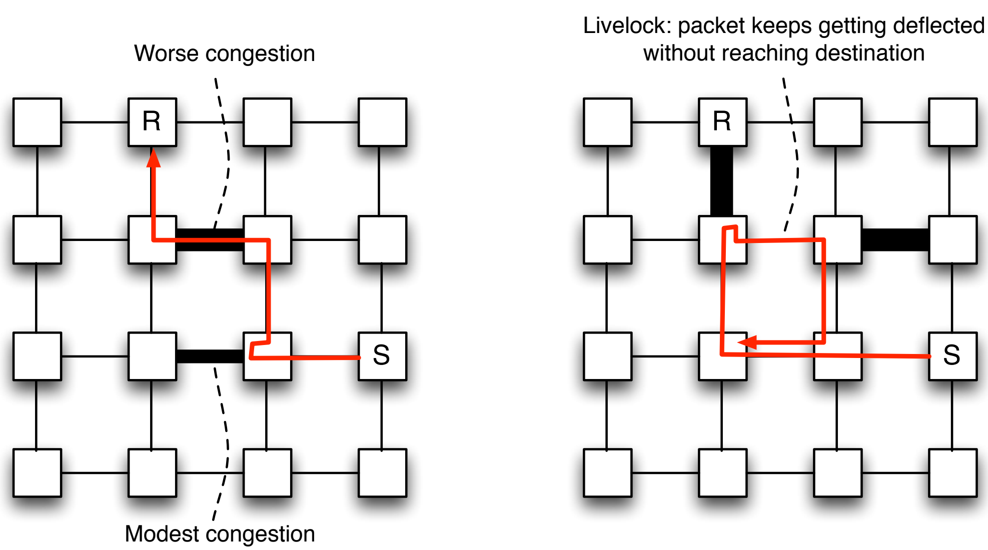
- Valid paths may go up first, then down, until destination
  - Down then up is not allowed
- Path diversity is provided, but paths may not be minimal



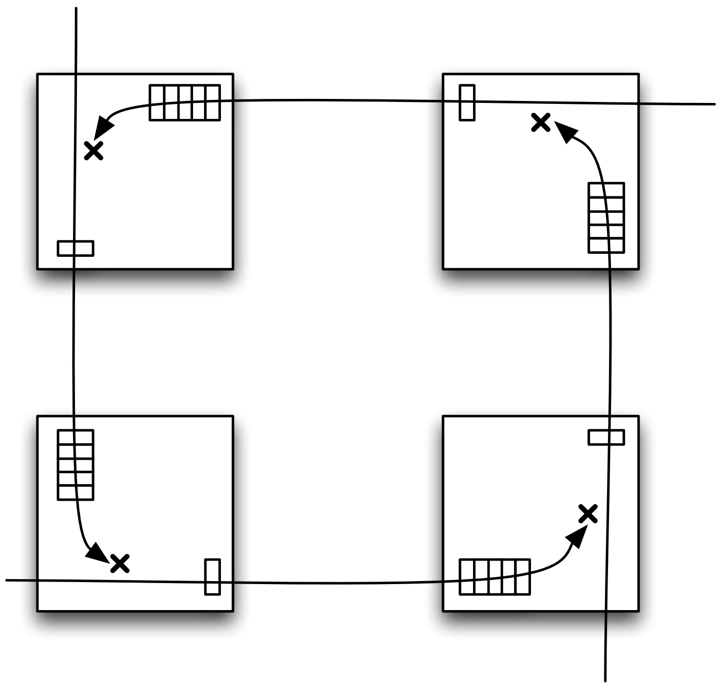


Adaptive Routing

- Adapts to the state of the network
  - Avoids congestion by deflecting a packet to another port
  - May be minimal or not minimal
  - Risks: (1) May face worse congestion after deflection, (2) possibility of livelock – may want to put a limit on number of deflections

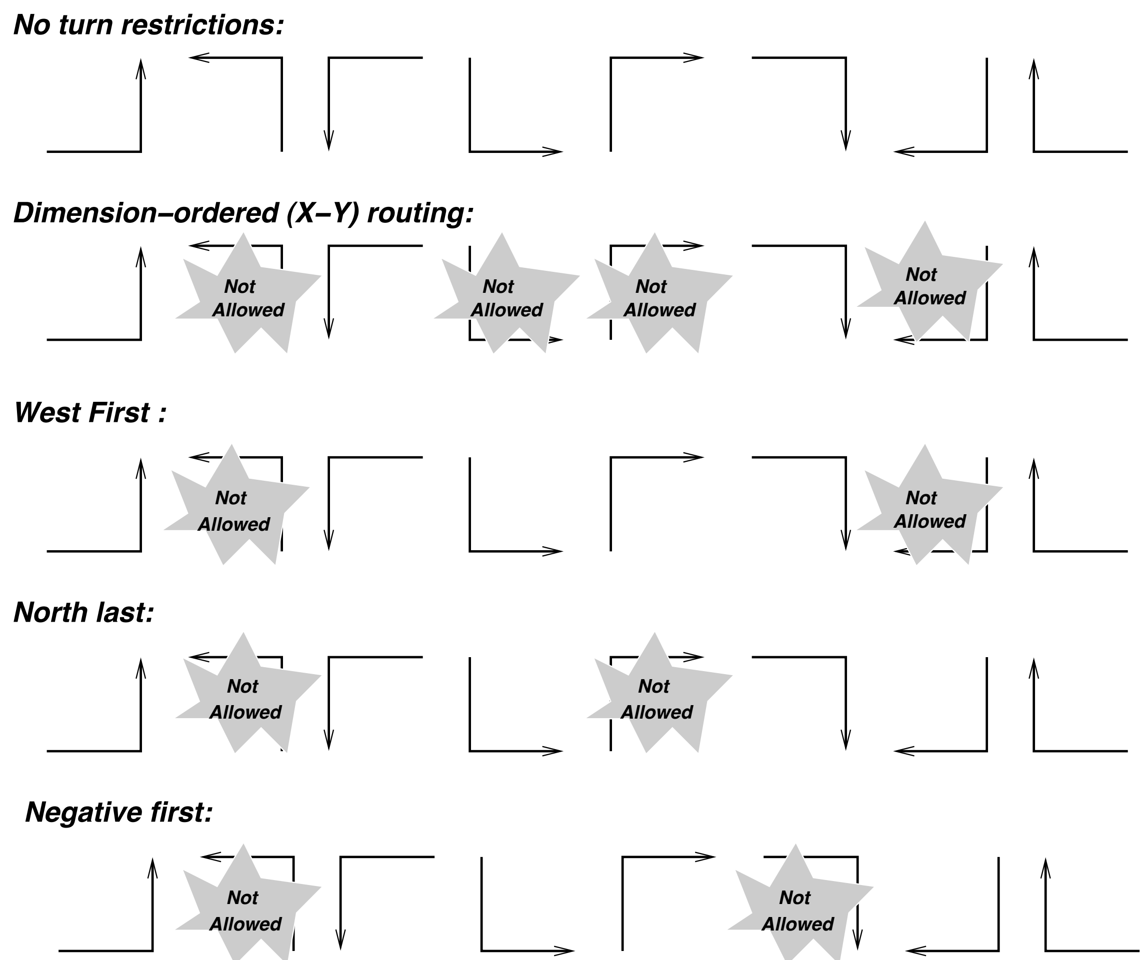


What Deadlock?

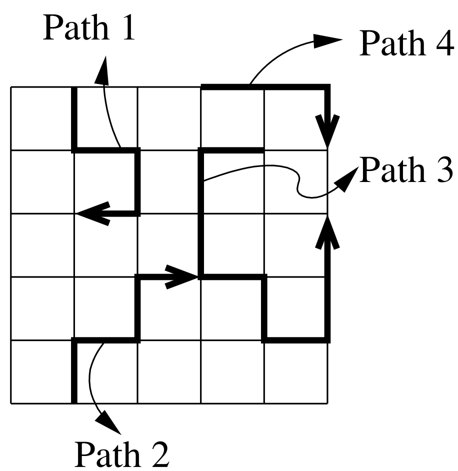


- Inability of the network to forward packets
- Due to limited buffer space and cyclic dependence on buffer acquisition
- The figure shows four packets filling up the input buffer and need the next output buffer, but the output buffer is already full
- Handling a deadlock
  - Detect and break: expensive
  - Drop packets: adverse effect in performance and complicates protocol
  - Avoid: restriction on routing
  - Adapt: Escape channel

Turn Restriction in Dimension-Ordered Routing



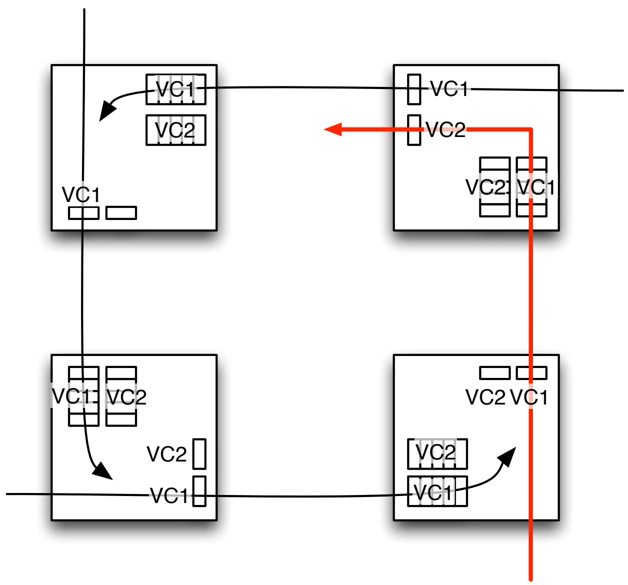
Example



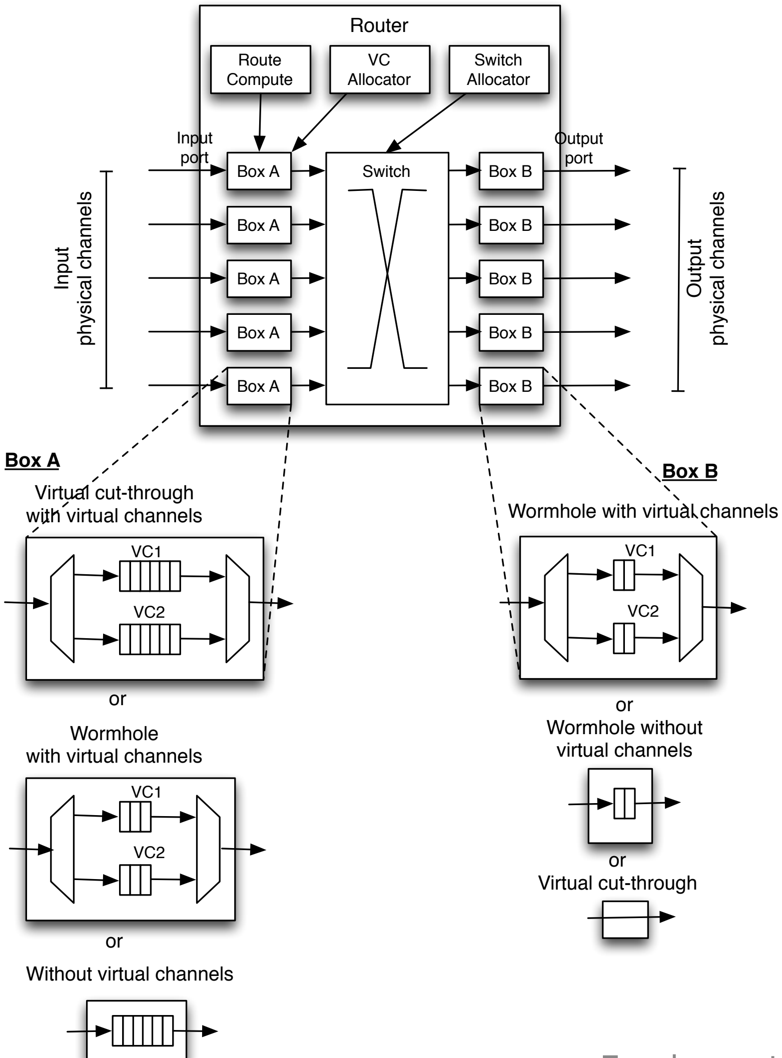
- Allowed in x-y dimension-ordered routing:
  - Path 4
- Allowed in west first routing:
  - Path 2, 3, 4
- Allowed in north-last routing:
  - Path 1, 3, 4

Deadlock

- Avoding down->up turn in up*/down* routing avoids deadlock
- Use virtual channel to escape from possible deadlock
  - Routing in the escape channel must be deadlock-free



Router Architecture



- Complexity depends on
  - Whether virtual channels are used
  - How many virtual channels share a physical channel
  - Whether wormhole routing or virtual cut-through is used
- Head flit (first flit in the packet) must go through
  - Decode & Compute (DC): decode header, compute output channel, buffer at input channel
  - Virtual Channel Allocation (VA): allocate output virtual channel for the packet
  - Switch Allocation (SA): allocate switch to connect input and output virtual channel
  - Switch Traversal (ST): transmit the flit through the switch to the output virtual channel

Cycles

	1	2	3	4	5	6	7
head flit	DC	VA	SA	ST			
body flit		-	-	SA	ST		
body flit			-	-	SA	ST	
tail flit				-	-	SA	ST