Lecture 14: Interconnection Networks

- Topics: dimension vs. arity, deadlock

Interconnection Networks

- Recall: fully connected network, arrays/rings, meshes/tori, trees, butterflies, hypercubes

- 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?

Interconnection Networks

- Recall: fully connected network, arrays/rings, meshes/tori, trees, butterflies, hypercubes

- 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 $N = k^d$

(with no wraparound)

Number of switches	N	Avg. routing distance	$d(k-1)/2$
Switch degree	$2d + 1$	Diameter	$d(k-1)$
Number of links	Nd	Bisection bandwidth	$2wk^{d-1}$
Pins per node	2wd	Switch complexity	$(2d + 1)^2$

Should we minimize or maximize dimension?

Bisection Bandwidth

Break the k^d nodes into two groups such that all elements
in group-1 are of the form: [0 - k/2-1] [*][*]...[*]
in group-2 are of the form: [k/2 – k] [*][*]...[*]

- Each node has an edge to other nodes that differ in only one dimension by one
- Any node in group-1 differs from any node in group-2 in at least the first dimension – hence, any edge from group-1 to group-2 is an edge that connects nodes that are identical in d-1 dimensions and differ in the first dimension by 1
- If we fix the co-ordinates of the d-1 dimensions, we can identify two edges: $[0, i_1,\ldots,i_{d-1}]$ – $[k-1, i_1,\ldots,i_{d-1}]$ and $[k/2-1, i_1,\ldots,i_{d-1}]$ – $[k/2, i_1,\ldots,i_{d-1}]$: there are totally $2k^{d-1}$ edges

Dimension

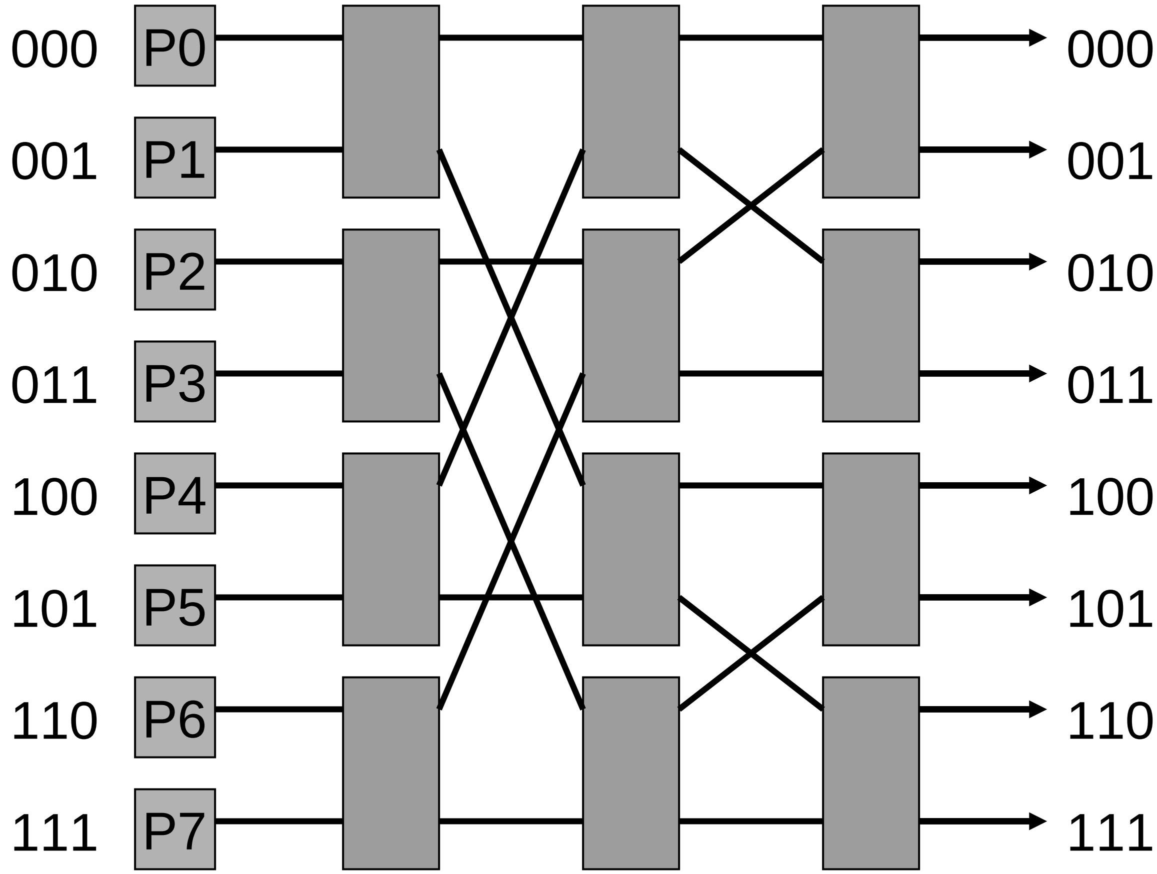
- For a fixed machine size N, low-dimension networks have significantly higher latencies for a packet – scalable machines should employ high dimensionality (high cost!)
- For a fixed number of pins, message latency decreases at first, then increases (as we increase dimensionality)
- What if we keep constant bisection bandwidth?

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	:	$(2d + 1)^2$

$N = k^d$

Butterfly Network



Routing

- Deterministic routing: given the source and destination, there exists a unique route

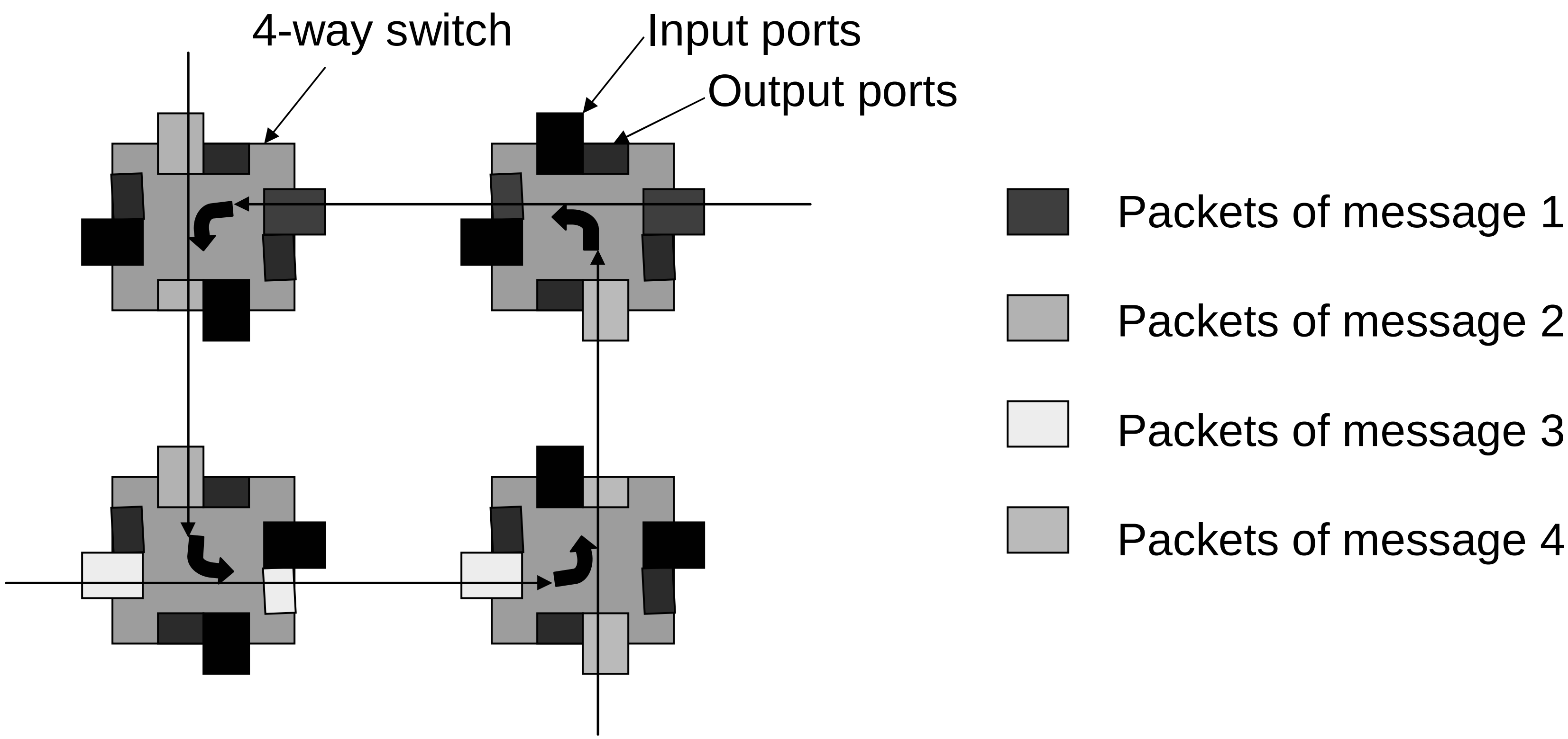
- Adaptive routing: a switch may alter the route in order to deal with unexpected events (faults, congestion) – more complexity in the router vs. potentially better performance

- Example of deterministic routing: dimension order routing: send packet along first dimension until destination co-ord (in that dimension) is reached, then next dimension, etc.

Deadlock

- Deadlock happens when there is a cycle of resource dependencies – a process holds on to a resource (A) and attempts to acquire another resource (B) – A is not relinquished until B is acquired

Deadlock Example

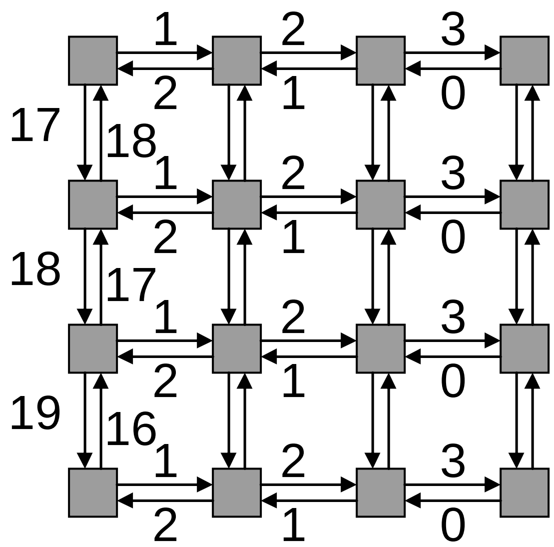


Each message is attempting to make a left turn – it must acquire an output port, while still holding on to a series of input and output ports

Deadlock-Free Proofs

- Number edges and show that all routes will traverse edges in increasing (or decreasing) order – therefore, it will be impossible to have cyclic dependencies

- Example: k-ary 2-d array with dimension routing: first route along x-dimension, then along y

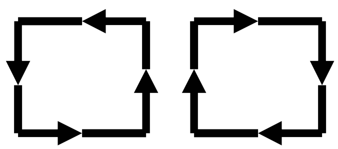


Breaking Deadlock I

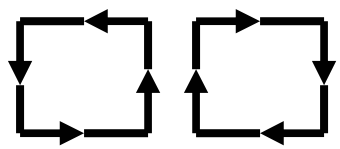
- The earlier proof does not apply to tori because of wraparound edges
- Partition resources across multiple virtual channels
- If a wraparound edge must be used in a torus, travel on virtual channel 1, else travel on virtual channel 0

Breaking Deadlock II

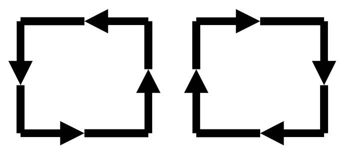
- Consider the eight possible turns in a 2-d array (note that turns lead to cycles)
- By preventing just two turns, cycles can be eliminated
- Dimension-order routing disallows four turns
- Helps avoid deadlock even in adaptive routing



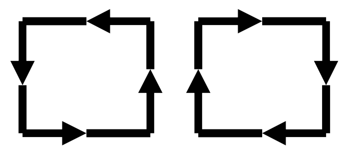
West-First



North-Last



Negative-First



Can allow deadlocks

Title

- Bullet