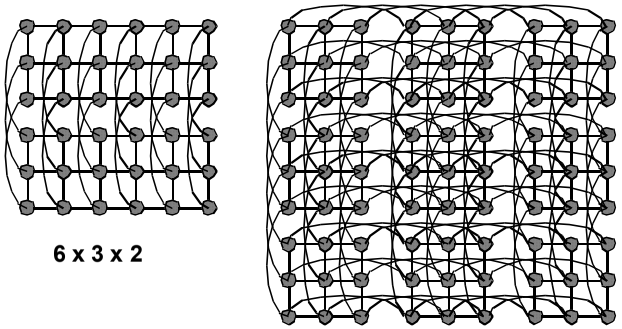
CS 258
Parallel Computer Architecture
Lecture 5

Routing

February 6, 2008
Prof John D. Kubiatowicz
http://www.cs.berkeley.edu/~kubitron/cs258

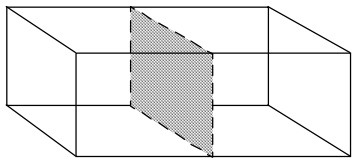
Recall: Embeddings in two dimensions



- When embedding higher-dimension in lower one, either some wires longer than others, or all wires long
- Note: for d>2, wiring density is nonuniform!

Recall: In the 3D world

- For n nodes, bisection area is $O(n^{2/3})$



- For large n, bisection bandwidth is limited to $O(n^{2/3})$
- Paper: Performance Analysis of k-ary n-cube Interconnection Networks (here, n ≡ dimension!)
  - Bill Dally, IEEE Transactions on Computers, June 1990
  - Under assumption of constant wire bisection
    - » Lower n is better because it provides: Lower latency, less contention, and higher throughput

Dally paper (con't)

- Equal Bisection, W=1 for hypercube ⇒ W= ½k
- Three wire models:
  - Constant delay, independent of length
  - Logarithmic delay with length (exponential driver tree)
  - Linear delay (speed of light/optimal repeaters)

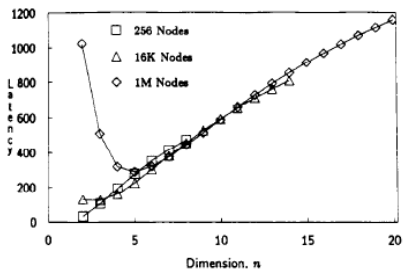


Fig. 9. Latency versus dimension for 256, 16K, and 1M nodes, logarithmic delay.

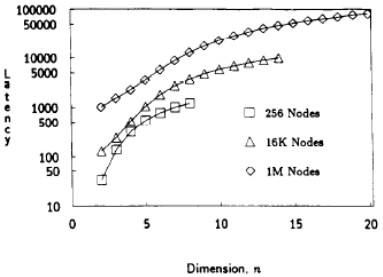


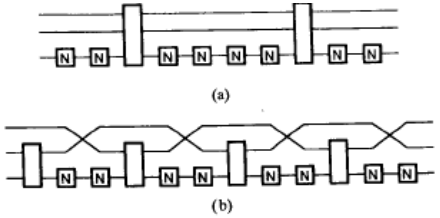
Fig. 10. Latency versus dimension for 256, 16K, and 1M nodes, linear delay.

Logarithmic Delay

Linear Daelay

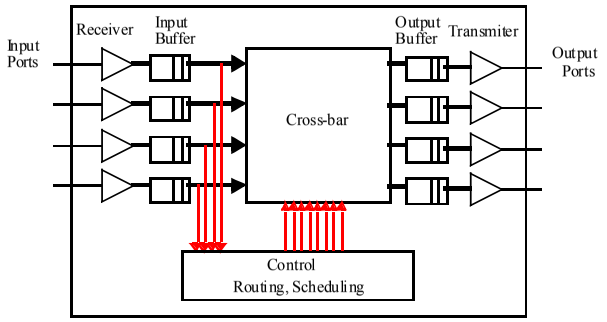
Another Idea: Express Cubes

- Problem: Low-dimensional networks have high k
  - Consequence: may have to travel many hops in single dimension
  - Routing latency can dominate long-distance traffic patterns
- Solution: Provide one or more "express" links



  - Like express trains, express elevators, etc
    - » Delay linear with distance, lower constant
    - » Closer to "speed of light" in medium
    - » Lower power, since no router cost
  - "Express Cubes: Improving performance of k-ary n-cube interconnection networks," Bill Dally 1991
- Another Idea: route with pass transistors through links

The Routing problem: Local decisions



- Routing at each hop: Pick next output port!

Routing

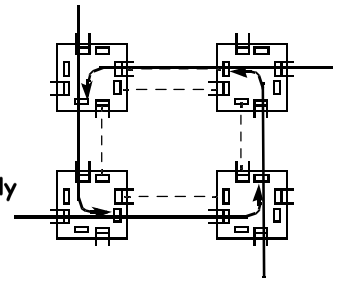
- Recall: routing algorithm determines
  - which of the possible paths are used as routes
  - how the route is determined
  - R: N x N -> C, which at each switch maps the destination node n_d to the next channel on the route
- Issues:
  - Routing mechanism
    - » arithmetic
    - » source-based port select
    - » table driven
    - » general computation
  - Properties of the routes
  - Deadlock free

Routing Mechanism

- need to select output port for each input packet
  - in a few cycles
- Simple arithmetic in regular topologies
  - ex: Δx, Δy routing in a grid
    - » west (-x) $\Delta x < 0$
    - » east (+x) $\Delta x > 0$
    - » south (-y) $\Delta x = 0$, $\Delta y < 0$
    - » north (+y) $\Delta x = 0$, $\Delta y > 0$
    - » processor $\Delta x = 0$, $\Delta y = 0$
- Reduce relative address of each dimension in order
  - Dimension-order routing in k-ary d-cubes
  - e-cube routing in n-cube

Deadlock Freedom

- How can it arise?
  - necessary conditions:
    - » shared resource
    - » incrementally allocated
    - » non-preemptible
  - think of a channel as a shared resource that is acquired incrementally
    - » source buffer then dest. buffer
    - » channels along a route



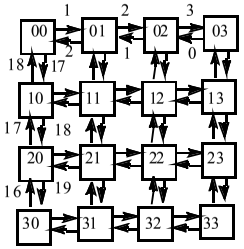
- How do you avoid it?
  - constrain how channel resources are allocated
  - ex: dimension order
- How do you prove that a routing algorithm is deadlock free?
  - Show that channel dependency graph has no cycles!

Proof Technique

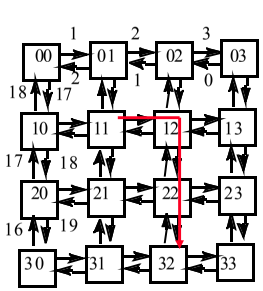
- resources are logically associated with channels
- messages introduce dependences between resources as they move forward
- need to articulate the possible dependences that can arise between channels
- show that there are no cycles in Channel Dependence Graph
  - find a numbering of channel resources such that every legal route follows a monotonic sequence ⇒ no traffic pattern can lead to deadlock
- network need not be acyclic, just channel dependence graph

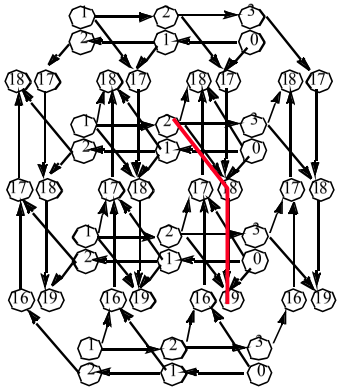
Example: k-ary 2D array

- Thm: Dimension-ordered (x,y) routing is deadlock free
- Numbering
  - +x channel (i,y) -> (i+1,y) gets i
  - similarly for -x with 0 as most positive edge
  - +y channel (x,j) -> (x,j+1) gets N+j
  - similary for -y channels
- any routing sequence: x direction, turn, y direction is increasing

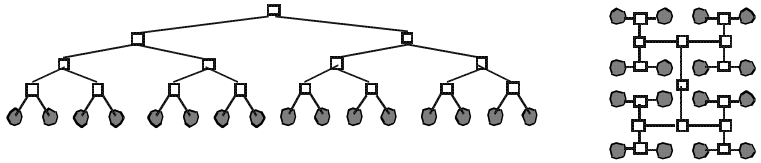


Channel Dependence Graph





Consider Trees

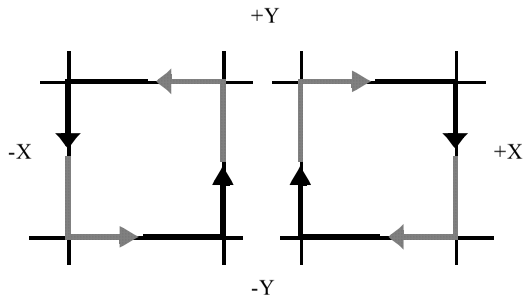


- Why is the obvious routing on X deadlock free?
  - butterfly?
  - tree?
  - fat tree?
- Any assumptions about routing mechanism? amount of buffering?

Up*-Down* routing

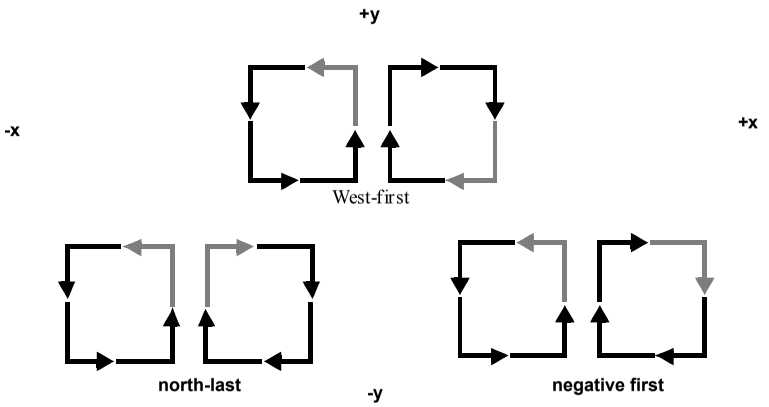
- Given any bidirectional network
- Construct a spanning tree
- Number of the nodes increasing from leaves to roots
- UP increase node numbers
- Any Source -> Dest by UP*-DOWN* route
  - up edges, single turn, down edges
  - Proof of deadlock freedom?
- Performance?
  - Some numberings and routes much better than others
  - interacts with topology in strange ways

Turn Restrictions in X,Y

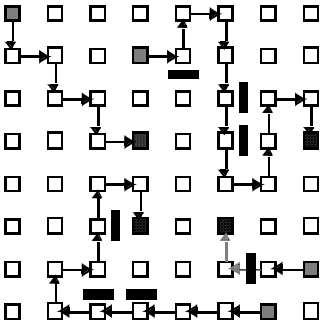


- XY routing forbids 4 of 8 turns and leaves no room for adaptive routing
- Can you allow more turns and still be deadlock free?

Minimal turn restrictions in 2D



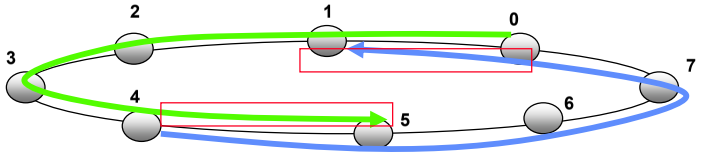
Example legal west-first routes



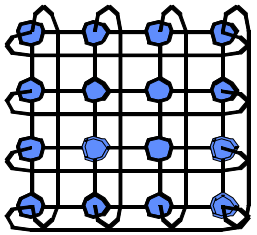
- Can route around failures or congestion
- Can combine turn restrictions with virtual channels

More examples:

- What about wormhole routing on a ring?

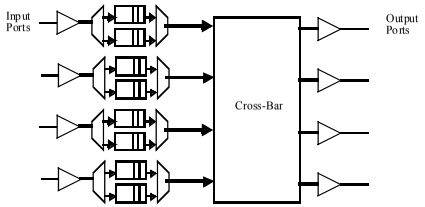


- Or: *Unidirectional* Torus of higher dimension?



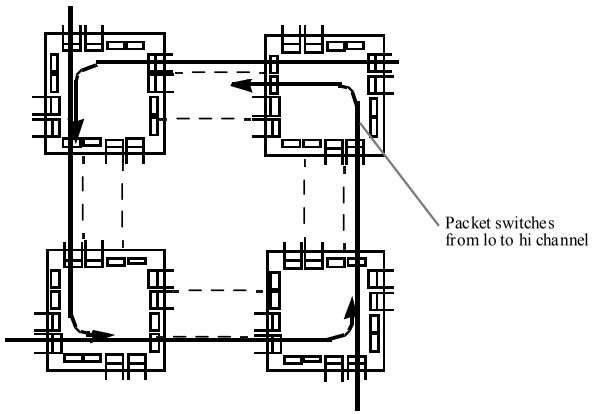
Deadlock free wormhole networks?

- Basic dimension order routing techniques don't work for unidirectional k-ary d-cubes
  - only for k-ary d-arrays (bi-directional)
- Idea: add channels!
  - provide multiple "virtual channels" to break the dependence cycle
  - good for BW too!



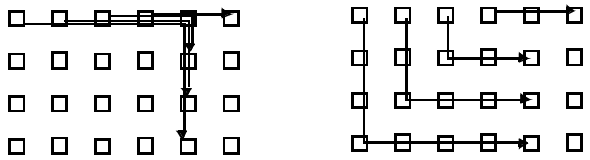
  - Do not need to add links, or xbar, only buffer resources
- This adds nodes to the CDG, remove edges?

Breaking deadlock with virtual channels



Adaptive Routing

- R: C x N x Σ -> C
- Essential for fault tolerance
  - at least multipath
- Can improve utilization of the network
- Simple deterministic algorithms easily run into bad permutations



- fully/partially adaptive, minimal/non-minimal
- can introduce complexity or anomolies
- little adaptation goes a long way!

Use of virtual channels for adaptation

- Want to route around hotspots/faults while avoiding deadlock
- "An adaptive and Fault Tolerant Wormhole Routing Strategy for k-ary n-cubes,"
  - Linder and Harden, 1991
  - General technique for k-ary n-cubes
    - » Requires: 2^{n-1} virtual channels/lane!!!
- Alternative: Planar adaptive routing
  - Chien and Kim, 1995
  - Divide dimensions into "planes",
    - » i.e. in 3-cube, use X-Y and Y-Z
  - Route planes adaptively in order: first X-Y, then Y-Z
    - » Never go back to plane once have left it
    - » Can't leave plane until have routed lowest coordinate
  - Use Linder-Harden technique for series of 2-dim planes
    - » Now, need only 3 × number of planes virtual channels
- Alternative: two phase routing
  - Provide set of virtual channels that can be used arbitrarily for routing
  - When blocked, use unrelated virtual channels for dimension-order (deterministic) routing
  - Never progress from deterministic routing back to adaptive routing

Summary

- Routing Algorithms restrict the set of routes within the topology
  - simple mechanism selects turn at each hop
  - arithmetic, selection, lookup
- Deadlock-free if channel dependence graph is acyclic
  - limit turns to eliminate dependences
  - add separate channel resources to break dependences
  - combination of topology, algorithm, and switch design
- Virtual Channels
  - Adds complexity to router
  - Can be used for performance
  - Can be used for deadlock avoidance
- Deterministic vs adaptive routing