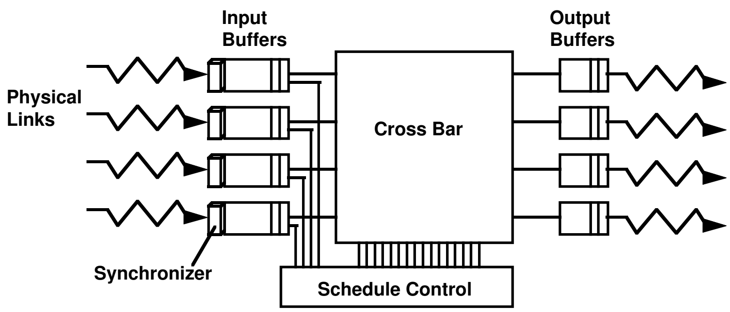
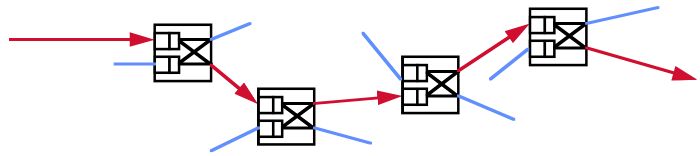
Routing in Scalable Networks

What's in a switch?

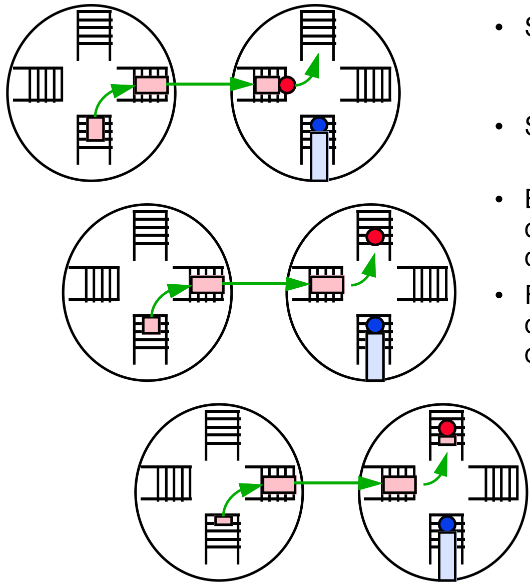


Basic Switching Strategies



- Circuit Switching
  - **establish a circuit from source to destination**
    - » **retract and retry on block**
  - **transmit message along circuit**
- Packet Switching
  - **Store and Forward**
    - » **move entire message one hop toward destination**
    - » **buffer till next hop permitted**
  - **Virtual Cut-through**
    - » **pipeline the hops, but buffer on blockage**
  - **Wormhole**
    - » **leave mesage spread through network on blockage**

Wormhole Routing



- Small Buffers
  - **independent of packet size**
- Small switch delay
  - **low latency**
- Behavior under contention similar to circuit switching
- FLITS must remain contiguous in channels

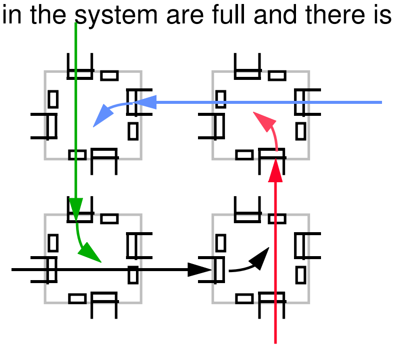
Unloaded Latency

- $T_{sf}(M, h) = T_{snd} + h(M/w + T_s) + T_{rcv}$
- $T_{wh}(M, h) = T_{snd} + M/w + hT_s \quad + T_{rcv}$
-
- Consider P=1024, M = 128 Bytes, $T_{snd} + T_{rcv} = 50$ cycles

Topology	h(ave)	w	Ts	Latency(ave)
2D Mesh	**21**	**16**	**4**	**50 + 64 + 84 = 198**
3D Cube	**8**	**8**	**4**	**50 + 128 + 32 = 210**
n-cube	**5**	**1**	**10**	**50 + 1024 + 50 = 1124**

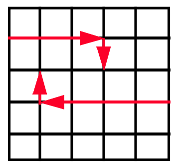
Deterministic Routing

- Routing taken by a message is determined only by its destination
  - **not by traffic in the network**
  - **non-adaptive**
- A route is minimal if each step leads closer to the destination
- Deadlock in the interconnection network occurs when no message can advance toward its destination because the queues in the system are full and there is a circular wait.

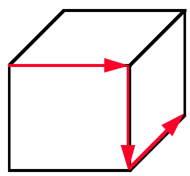


Deterministic Deadlock Free Routing

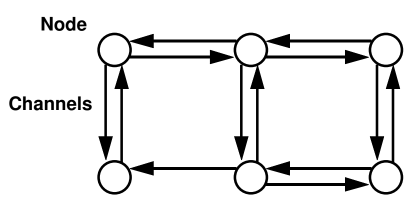
- Assuming a message arriving at its destination is eventually consumed
- Mesh: “XY” – route first in X then in Y

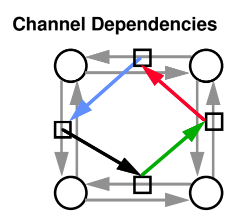


- Hypercube: “e-cube” route along lowest dimension, then along higher and higher dimensions

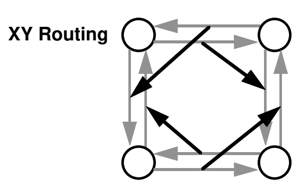


Deadlock Conditions





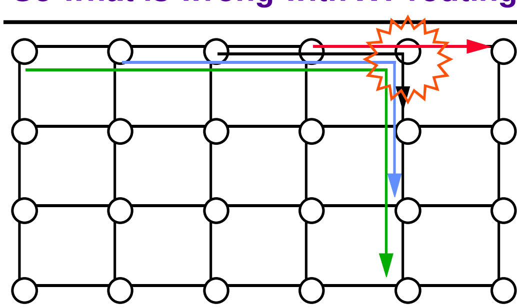
- A routing function R : Channel X Dest –> Channel for an interconnection network is deadlock free iff there are no cycles in the channel dependency graph

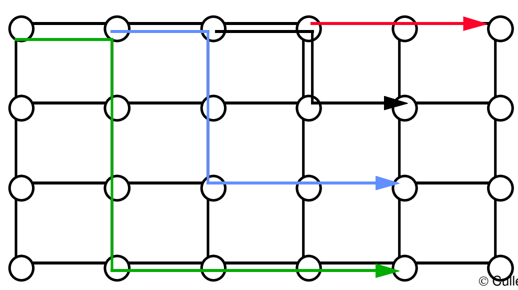


E-cube routing is deadlock free

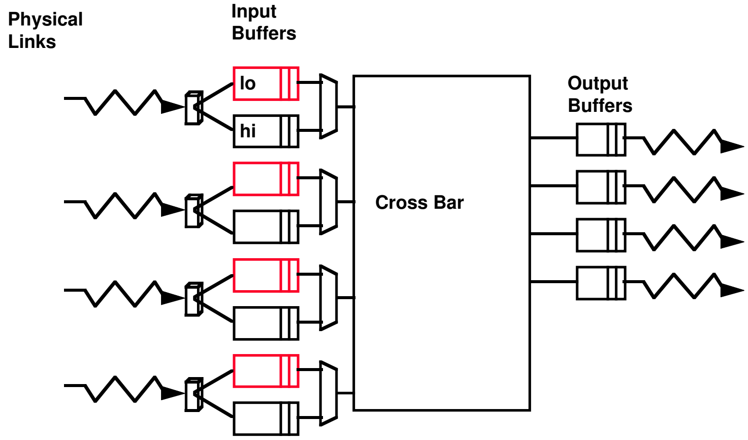
- node n_k has n output channels $C_{0k}, \ldots, C_{(n-1)k}$
- A msg arriving at node n_k destined for node n_l is routed on channel C_{ik}, where *i* is the most significant position where *l* and k differ.
- Since msgs are routed in order of decreasing dimension, and hence decreasing channel subscript, there are no cycles in the channel dependency graph.

So what is wrong with XY routing?



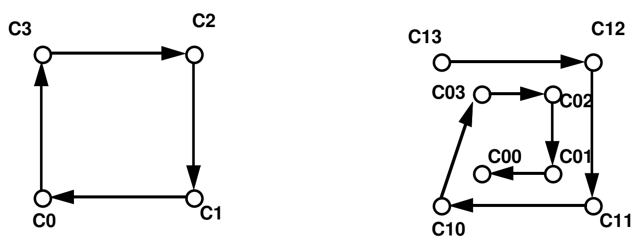


Virtual Channels



cs258 24.11

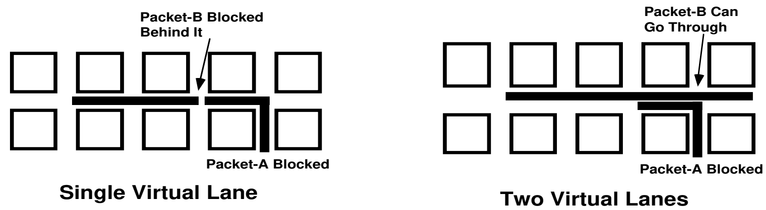
Use virtual channels to break deadlock cycles

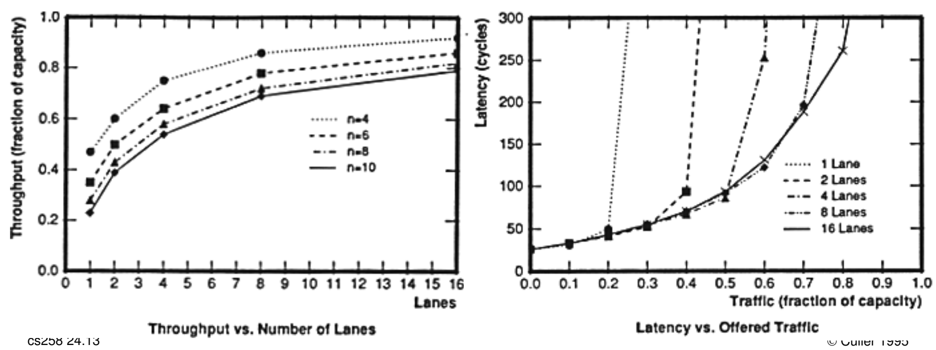


Old Channel Dependency graph

cs258 24.12

Virtual Channels also help performance





cs258 24.13 © Culler 1995

Deadlock Free Routing on k-ary n-cube

- Virtual channel subscripts of the form dvn
  - **d = dimension**
  - **v = channel (lo, hi)**
  - **n = source node**
- Route in dimension order, from highest decreasing
  - **like e-cube**
- In each dimension *i*, a msf is routed in that dimension until it reaches a node whose subscript matches the destination address in the *i* th position.
- Route on hi channel is dest address is greater than *i* th digit of current node address
- otherwise, route on lo channel

=> descending subscripts

- Approach works for arbitrary graphs, but number of virtual channels is an issue

cs258 24.14 © Culler 1995

Adaptive Routing

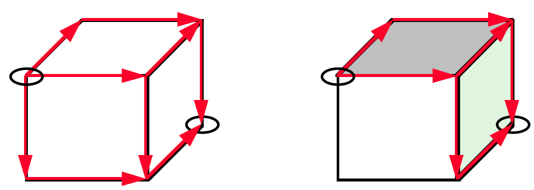
- Route around faults
- Route around congestion
- Better load balance

Important terms:

- Minimal adaptive routing
  - **only routes along shortest paths**
- Fully-adaptive routing
  - **can route paths along any shortest path**
- Partially adaptive routing

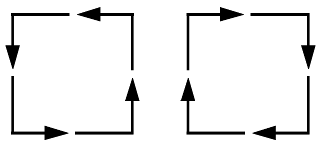
Planar Adaptive Routing (Chen ISCA92)

- Avoid exponential increase in virtual channels by restricting degrees of freedom at every step
- With a single plane at a time, only need 2n channels

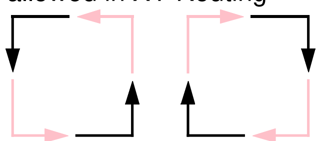


Turn Model for Adaptive Routing (Glass, ISCA92)

- Idea: Restrict the set of turns to eliminate dependence cycles
- Possible abstract cycles and turns in a 2D Mesh

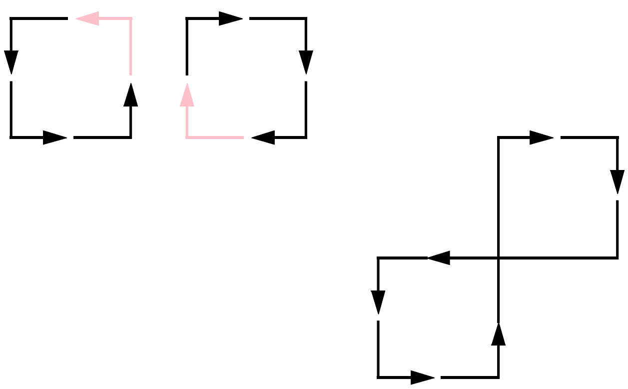


- Turns allowed in XY Routing



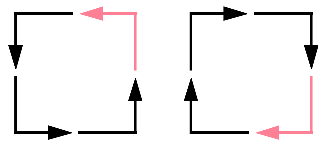
- Should be enough to eliminate 2 turns

Not Just Any two turns

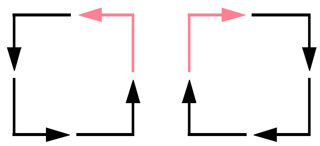


Example Deadlock Free Routes

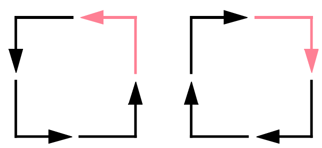
- West-first: route west first, then adaptively N,E, and S



- North-Last



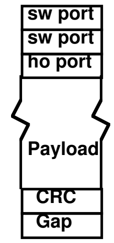
- Negative-First



Performance Analysis

- No advantage for uniform traffic
  - **adaptive routing of any sort on a mesh is bad in this case because everybody turns to the center**
- Big improvement in “bad cases”
  - **transpose**
  -
- Many interesting papers on minimal turn restrictions X minimal virtual channels for minimal, fully adpative routing
  - **see Schwiebert & Jayasimha SPAA95**

Myrinet: recent interesting switch



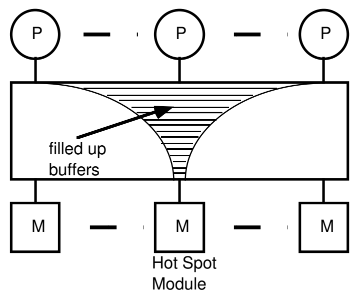
- 8x8 non-blocking cross-bar
  - **wormhole**
  - **source-based routing**
- 80 MB/s per port
  - **synchronous at sending port**
- 550 ns max delay through switch (44 characters)
- 23 characters on a 25 M cable (~ 300 ns)
- 9-bit electrical cable links
  - **2V differential signalling**
  - **256 data + 5 control symbols**
  - **Non-return-to-zero encoding**
    - » **transition encodes 1 and non encodes zero**
    - » **all zeros is idle character (at least one bit provides edge)**
- Stop/Go link by link flow contol
  - **Go @ 32, Stop @ 48, Block @ 80**

Hot Spots: prolonged Contention

- Location Hot Spot: Higher accesses to a single location imposed on a uniform background traffic.
  - **May arise due to synch accesses or other heavily shared data**
  - **Not only are accesses to hot-spot location delayed, they found all other accesses were delayed too.**
  - **Tree Saturation effect.)**

Saturation Model

- Parameters: p = # of PEs; r = # of refs / PE / cycle; h = % refs from PE to hot spot
  - **Total traffic to hot-spot memory module = rhp + r(1-h)**
  - **"rhp" is hot-spot refs and "r(1-h)" is due to uniform traffic**
- Latencies for all refs rise suddenly when [rhp + r(1-h)] = 1, assuming memory handles one request per cycle.
- <u>Tree Saturation Effect:</u> Buffers at all switches in the shaded area fill up, and even non-hot-spot requests have to pass through there.
- Network combining ???



Location Hot Spot ???

- F&A(V,e) returns old value of V and atomically sets V = V + e;
- If V = k, and X = F&A(V, a) and Y = F&A(V, b) done at same time
- One possible result: X = k, Y = k+a, and V = k+a+b.
- Another possible result: Y = k, X = k+b, and V = k+a+b.
- Example use: Implementation of task queues.

```
Insert:  myI = F&A(qi, 1);
         Q[myI] = data;
         full[myI] = 1;

Delete:  myI = F&A(qd, 1);

         while (!full[myI]) ;
               data = Q[myI];
         full[myI] = 0;
```