Interconnection Network Topology Design Trade-offs

CS 258, Spring 99
David E. Culler
Computer Science Division
U.C. Berkeley

Real Machines

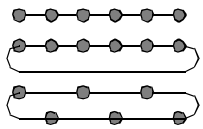
Machine	Topology	Cycle Time (ns)	Channel Width (bits)	Routing Delay (cycles)	Flit (data bits)
nCUBE/2	Hypercube	25	1	40	32
TMC CM-5	Fat-Tree	25	4	10	4
IBM SP-2	Banyan	25	8	5	16
Intel Paragon	2D Mesh	11.5	16	2	16
Meiko CS-2	Fat-Tree	20	8	7	8
CRAY T3D	3D Torus	6.67	16	2	16
DASH	Torus	30	16	2	16
J-Machine	3D Mesh	31	8	2	8
Monsoon	Butterfly	20	16	2	16
SGI Origin	Hypercube	2.5	20	16	160
Myricom	Arbitrary	6.25	16	50	16

- Wide links, smaller routing delay
- Tremendous variation

Interconnection Topologies

- Class networks scaling with N
- Logical Properties:
  - distance, degree
- Physcial properties
  - length, width
- Fully connected network
  - diameter = 1
  - degree = N
  - cost?
    - » bus => O(N), but BW is O(1) - actually worse
    - » crossbar => $O(N^2)$ for BW O(N)
- VLSI technology determines switch degree

Linear Arrays and Rings



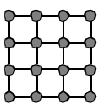
Linear Array

Torus

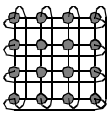
Torus arranged to use short wires

- Linear Array
  - Diameter?
  - Average Distance?
  - Bisection bandwidth?
  - Route A -> B given by relative address R = B-A
- Torus?
- Examples: FDDI, SCI, FiberChannel Arbitrated Loop, KSR1

Multidimensional Meshes and Tori



2D Grid



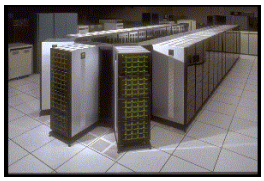
3D Cube

- *d*-dimensional array
  - $n = k_{d-1} \times \ldots \times k_0$ nodes
  - described by *d*-vector of coordinates $(i_{d-1}, \ldots, i_0)$
- *d*-dimensional *k*-ary mesh: $N = k^d$
  - $k = \sqrt[d]{N}$
  - described by *d*-vector of radix k coordinate
- *d*-dimensional *k*-ary torus (or *k*-ary *d*-cube)?

Properties

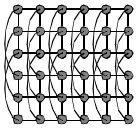
- Routing
  - relative distance: $R = (b_{d-1} - a_{d-1}, \ldots, b_0 - a_0)$
  - traverse $r_i = b_i - a_i$ hops in each dimension
  - *dimension-order routing*
- Average Distance — Wire Length?
  - d x 2k/3 for mesh
  - dk/2 for cube
- Degree?
- Bisection bandwidth? — Partitioning?
  - k^{d-1} bidirectional links
- Physical layout?
  - 2D in O(N) space — Short wires
  - higher dimension?

Real World 2D mesh

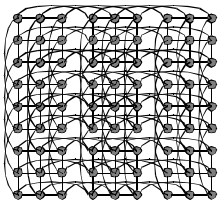


- 1824 node Paragon: 16 x 114 array

Embeddings in two dimensions

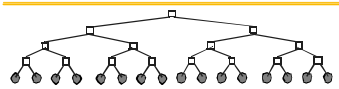


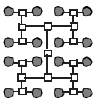
6 x 3 x 2



- Embed multiple logical dimension in one physical dimension using long wires

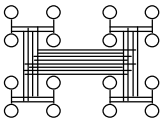
Trees





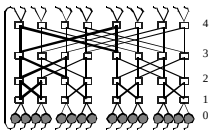
- Diameter and ave distance logarithmic
  - k-ary tree, height d = $\log_k N$
  - address specified d-vector of radix k coordinates describing path down from root
- Fixed degree
- Route up to common ancestor and down
  - R = B xor A
  - let i be position of most significant 1 in R, route up i+1 levels
  - down in direction given by low i+1 bits of B
- H-tree space is O(N) with O(ÖN) long wires
- Bisection BW?

Fat-Trees

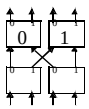


- Fatter links (really more of them) as you go up, so bisection BW scales with N

Butterflies



16 node butterfly



building block

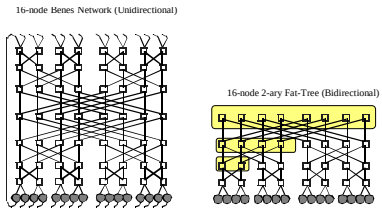
- Tree with lots of roots!
- N log N (actually N/2 x logN)
- Exactly one route from any source to any dest
- R = A xor B, at level i use 'straight' edge if r_i=0, otherwise cross edge
- Bisection N/2 vs $n^{(d-1)/d}$

k-ary d-cubes vs d-ary k-flies

- degree d
- N switches vs N log N switches
- diminishing BW per node vs constant
- requires locality vs little benefit to locality

- Can you route all permutations?

Benes network and Fat Tree

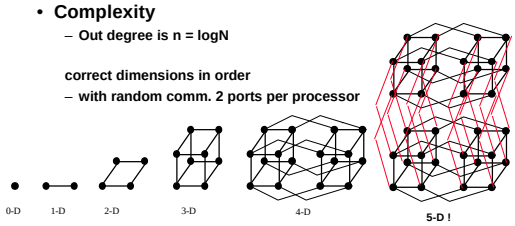


- Back-to-back butterfly can route all permutations
  - off line
- What if you just pick a random mid point?

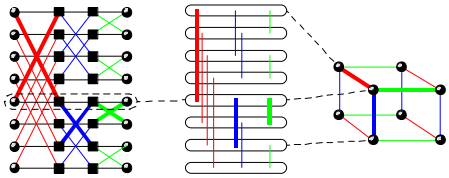
Hypercubes

- Also called binary n-cubes. # of nodes = N = 2^n.
- O(logN) Hops
- Good bisection BW
- Complexity
  - Out degree is n = logN

    correct dimensions in order
  - with random comm. 2 ports per processor



Relationship BttrFlies to Hypercubes



- Wiring is isomorphic
- Except that Butterfly always takes log n steps

Toplology Summary

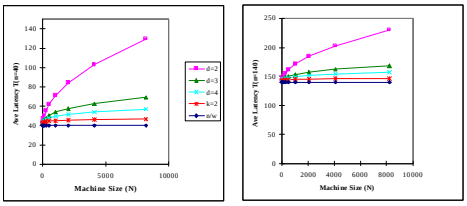
Topology	Degree	Diameter	Ave Dist	Bisection	D (D ave) @ P=1024
1D Array	2	N-1	N / 3	1	huge
1D Ring	2	N/2	N/4	2	
2D Mesh	4	2 ($N^{1/2}$ - 1)	2/3 $N^{1/2}$	$N^{1/2}$	63 (21)
2D Torus	4	$N^{1/2}$	1/2 $N^{1/2}$	$2N^{1/2}$	32 (16)
k-ary n-cube	2n	nk/2	nk/4	nk/4	15 (7.5) @n=3
Hypercube	n =log N		n	n/2	N/2 10 (5)

- All have some "bad permutations"
  - many popular permutations are very bad for meshs (transpose)
  - ramdomness in wiring or routing makes it hard to find a bad one!

How Many Dimensions?

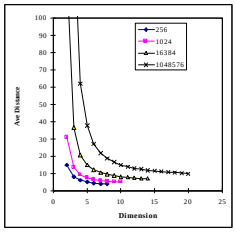
- n = 2 or n = 3
  - Short wires, easy to build
  - Many hops, low bisection bandwidth
  - Requires traffic locality
- n >= 4
  - Harder to build, more wires, longer average length
  - Fewer hops, better bisection bandwidth
  - Can handle non-local traffic
- k-ary d-cubes provide a consistent framework for comparison
  - N = k^d
  - scale dimension (d) or nodes per dimension (k)
  - assume cut-through

Traditional Scaling: Latency(P)



- Assumes equal channel width
  - independent of node count or dimension
  - dominated by average distance

Average Distance

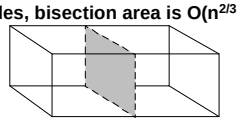


ave dist = d (k-1)/2

- but, equal channel width is not equal cost!
- Higher dimension => more channels

In the 3D world

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



- For large n, bisection bandwidth is limited to $O(n^{2/3})$
  - Bill Dally, IEEE TPDS, [Dal90a]
  - For fixed bisection bandwidth, low-dimensional k-ary n-cubes are better (otherwise higher is better)
  - i.e., a few short fat wires are better than many long thin wires
  - What about many long fat wires?

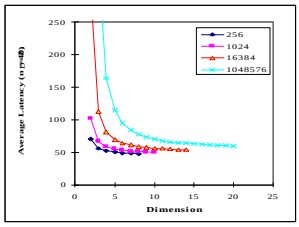
Equal cost in k-ary n-cubes

- Equal number of nodes?
- Equal number of pins/wires?
- Equal bisection bandwidth?
- Equal area? Equal wire length?

What do we know?

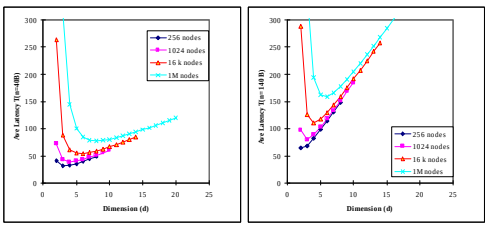
- switch degree: d diameter = d(k-1)
- total links = Nd
- pins per node = 2wd
- bisection = k^{d-1} = N/k links in each directions
- 2Nw/k wires cross the middle

Latency(d) for P with Equal Width



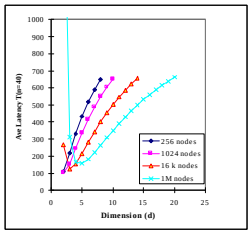
- total links(N) = Nd

Latency with Equal Pin Count



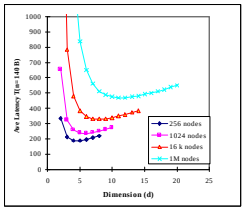
- Baseline d=2, has w = 32 (128 wires per node)
- fix 2dw pins => w(d) = 64/d
- distance up with d, but channel time down

Latency with Equal Bisection Width



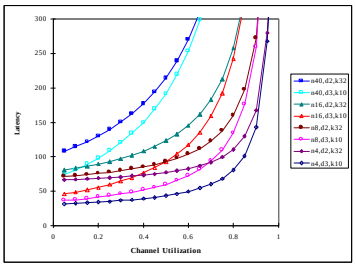
- N-node hypercube has N bisection links
- 2d torus has $2N^{1/2}$
- Fixed bisection => $w(d) = N^{1/d} / 2 = k/2$
- 1 M nodes, d=2 has w=512!

Larger Routing Delay (w/ equal pin)



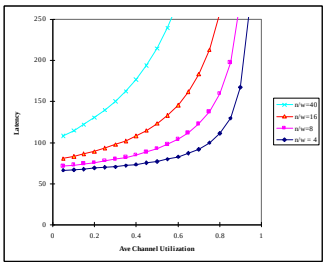
- Dally's conclusions strongly influenced by assumption of small routing delay

Latency under Contention



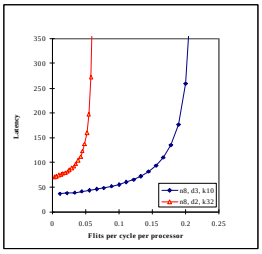
- Optimal packet size? Channel utilization?

Saturation



- Fatter links shorten queuing delays

Phits per cycle



- higher degree network has larger available bandwidth
- cost?

Discussion

- Rich set of topological alternatives with deep relationships
- Design point depends heavily on cost model
  - nodes, pins, area, ...
  - Wire length or wire delay metrics favor small dimension
  - Long (pipelined) links increase optimal dimension
- Need a consistent framework and analysis to separate opinion from design
- Optimal point changes with technology