

Message Passing Computers

Mark Greenstreet

CpSc 418 – September 26, 2018

Outline:

- [Network Topologies](#)
- [Performance Considerations](#)
- [Examples](#)

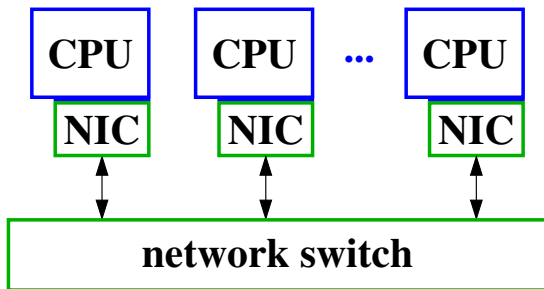


Unless otherwise noted or cited, these slides are copyright 2018 by Mark Greenstreet and are made available under the terms of the Creative Commons Attribution 4.0 International license <http://creativecommons.org/licenses/by/4.0/>

Objectives

- Familiar with typical network topologies: rings, meshes, crossbars, tori, hypercubes, trees, fat-trees.
- Understand implications for programming
 - ▶ bandwidth bottlenecks
 - ▶ latency considerations
 - ▶ location matters
 - ▶ heterogeneous computers.

Message Passing Computers



- Multiple CPU's
- Communication through a network:
 - ▶ Commodity networks for small clusters.
 - ▶ Special high-performance networks for super-computers
- Programming model:
 - ▶ Explicit message passing between processes (like Erlang)
 - ▶ No shared memory or variables.

What programmers Want

- **Low latency:** messages take very little time from sender to receiver.
- **High bandwidth:**
 - ▶ We can send large amounts of data between processors.
 - ▶ Everyone can communicate at the same time.
- **Uniformity:**
 - ▶ All processors and connections are equivalent.
 - ▶ The programmer shouldn't need to worry about **where** each processor is.
- *"You can't always get what you want."*

Some simple message-passing clusters

- 25 linux workstations (e.g. lin01 . . . lin25.ugrad.cs.ubc.ca) and standard network routers.
 - ▶ A good platform for learning to use a message-passing cluster.
 - ▶ But, we'll figure out that network bandwidth and latency are key bottlenecks.
- A “blade” based cluster, for example:
 - ▶ 16 “blades” each with 4 6-core CPU chips, and 32G of DRAM.
 - ▶ An “infiniband” or similar router for about 10-100 times the bandwidth of typical ethernet.
 - ▶ The price tag is ~\$300K.
 - ★ Great if you need the compute power.
 - ★ But, we won't be using one in this class.

The Sunway TaihuLight

- The world's fastest (Linpack) super-computer (as of June 2016)
- 40,960 multicore CPUs
 - ▶ 256 cores per CPU chip.
 - ▶ 1.45GHz clock frequency, 8 flops/core/cycle.
- Total of 10,485,760 cores
- LINPACK performance: 93 PFlops
 - ▶ 1 Petaflop = 10^3 Teraflops = 10^6 Gigaflops = 10^{15} flops
 - ▶ 1 flop = one floating point operation per second
- Power consumption 15MW (computer) + cooling (unspecified)
- Tree-like
 - ▶ Five levels of hierarchy.
 - ▶ Each level has a high-bandwidth switch.
 - ▶ Some levels (all?) are fully-connected for that level.
- Programming model: A version linux with MPI tuned for this machine.
- For more information, see [*Report on the Sunway TaihuLight System*](#), J. Dongarra, June 2016.

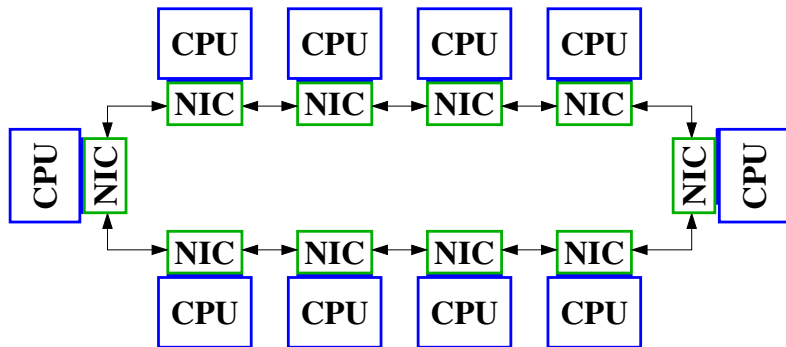


- Clusters at various Western Canadian Universities (including UBC).
- Up to 27,000 cores.
- Available for research use.

Network Topologies

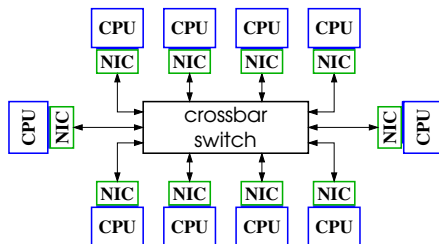
- Network topologies are to the message-passing community what cache-coherence protocols are to the shared-memory people:
 - ▶ **Lots** of papers have been published.
 - ▶ Machine designers are always looking for better networks.
 - ▶ Network topology has a strong impact on performance, the programming model, and the cost of building the machine.
- A message-passing machine may have multiple networks:
 - ▶ A general purpose network for sending messages between machines.
 - ▶ Dedicated networks for reduce, scan, and synchronization:
 - ★ The reduce and scan networks can include ALUs (integer and/or floating point) to perform common operations such as sums, max, product, all, any, etc. in the networking hardware.
 - ★ A synchronization network only needs to carry a few bits and can be designed to minimize latency.

Ring-Networks



- Advantages: simple.
- Disadvantages:
 - ▶ Worst-case latency grows as $O(P)$ where P is the number of processors.
 - ▶ Easily congested – limited bandwidth.

Star Networks



- Advantages:

- ▶ Low-latency – single hop between any two nodes
- ▶ High-bandwidth – no contention for connections with different sources and destinations.

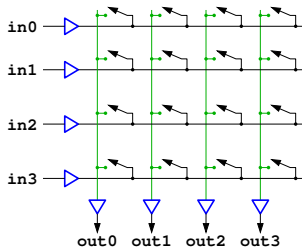
- Disadvantages:

- ▶ Amount of routing hardware grows as $O(P^2)$.
- ▶ Requires lots of wires, to and from switch –
Imagine trying to build a switch that connects to 1000 nodes!

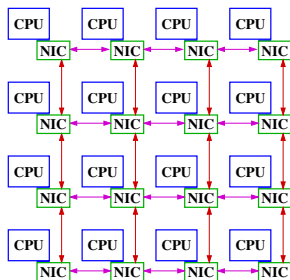
- Summary

- ▶ Surprisingly practical for 10-50 ports.
- ▶ Hierarchies of cross-bars are often used for larger networks.

A crossbar switch



Meshes



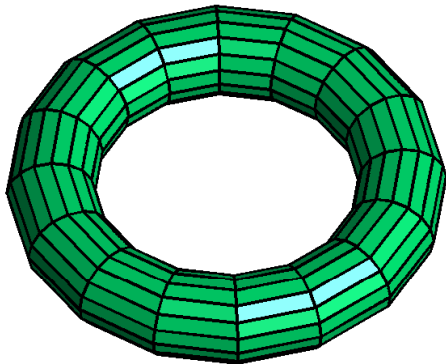
- Advantages:

- ▶ Easy to implement: chips and circuit boards are effectively two-dimensional.
- ▶ Cross-section bandwidth grow with number of processors – more specifically, bandwidth grows as \sqrt{P} .

- Disadvantages:

- ▶ Worst-case latency grows as \sqrt{P} .
- ▶ Edges of mesh are “special cases.”

Tori



- Advantages:
 - ▶ Has the good features of a mesh, and
 - ▶ No special cases at the edges.
- Disadvantages:
 - ▶ Worst-case latency grows as \sqrt{P} .

Mesh & Torus coordinates

- Consider an $N \times M$ grid of processors.
- Each processor can be identified by its coordinates, (i, j) , with $0 \leq i < N$ and $0 \leq j < M$.
- For a **mesh**:
 - ▶ Processor (i, j) has a connection to $(i + 1, j)$ **if** $i < N - 1$.
 - ▶ Processor (i, j) has a connection to $(i - 1, j)$ **if** $i > 0$.
 - ▶ Processor (i, j) has a connection to $(i, j + 1)$ **if** $j < M - 1$.
 - ▶ Processor (i, j) has a connection to $(i, j - 1)$ **if** $j > 0$.
- For a **torus**:
 - ▶ Processor (i, j) has a bi-directional connection to $((i + 1) \bmod N, j)$.
 - ▶ Processor (i, j) has a bi-directional connection to $(i, (j + 1) \bmod M)$.
 - ▶ No special cases for the edges.
- Both definitions generalize to higher dimensions “in the obvious way”.

Hypercubes

A 0-dimensional (1 node), radix-2 hypercube



A 0-dimensional (1 node), radix-2 hypercube

Hypercubes

A 1-dimensional (2 node), radix-2 hypercube

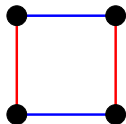


A 5-dimensional (32 node), radix-2 hypercube



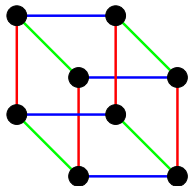
Hypercubes

A 2-dimensional (4 node), radix-2 hypercube



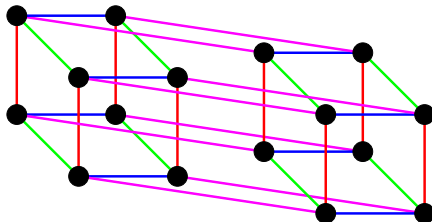
Hypercubes

A 3-dimensional (8 node), radix-2 hypercube



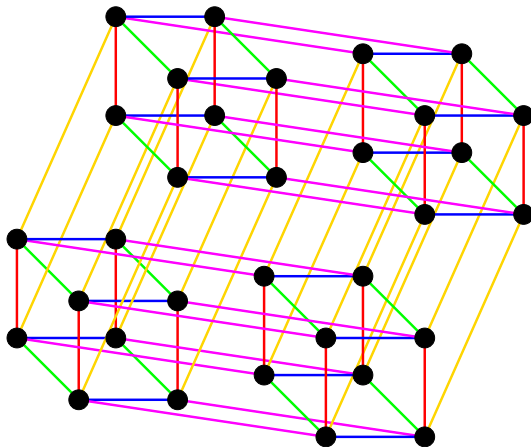
Hypercubes

A 4-dimensional (16 node), radix-2 hypercube



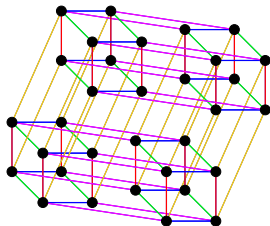
Hypercubes

A 5-dimensional (32 node), radix-2 hypercube



Hypercubes

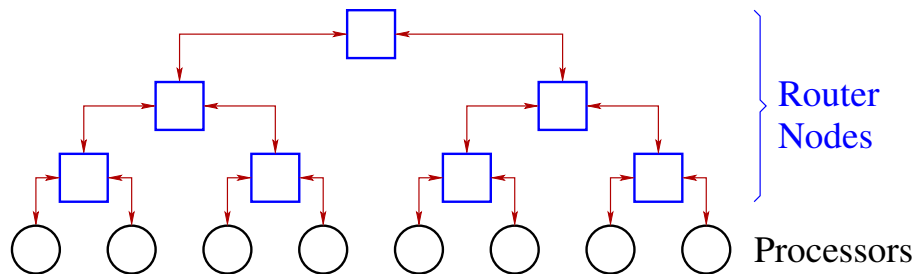
A 5-dimensional (32 node), radix-2 hypercube



Dimension Routing

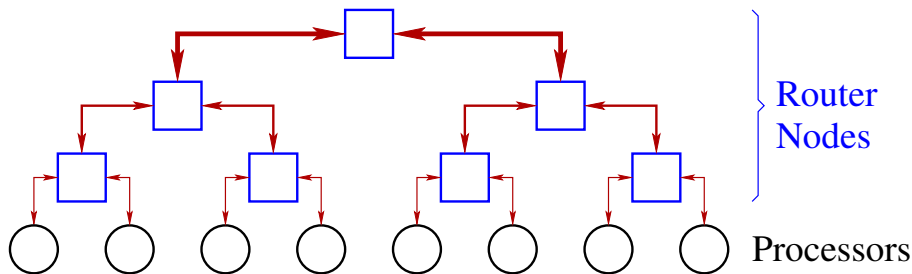
```
% Send a message, msg, from node src to node dst
for i = 1:d                                % d is dimension of the hypercube
    if(bit(i, src) != bit(i, dst))        % if different for dimension i
        send(msg, link[i]);              % then send msg to our i-neighbour
```

Trees



- Simple network: number of routing nodes = number of processors – 1.
- Wiring: $O(\log N)$ extra height ($O(N \log N)$) extra area.
 - ▶ Wiring: $O(\sqrt{N} \log N)$ extra area for H-tree.
- Low-latency: $O(\log N)$ + wire delay.
- Low-bandwidth: bottleneck at root.

Fat-Trees



- Use M^α parallel links to connect subtrees with M leaves.
- $0 \leq \alpha \leq 1$
 - ▶ $\alpha = 0$: simple tree
 - ▶ $\alpha = 1$: strange crossbar
- Fat-trees are “universal”
 - ▶ For $\frac{2}{3} < \alpha < 1$ a fat-tree interconnect with volume V can simulate **any** interconnect that occupies the same volume with a time overhead that is poly-log factor of N .

It's all about wires

Consider a network with P processors and cross-section bandwidth of B .

- For simplicity, assume a two-dimensional implementation (chip or circuit board)
- Wires (or optical fibers, etc.) have $\Omega(1)$ diameter.
- This gives a lower bound for the **diameter** of the machine of $\Omega(B)$.
- The bound for the area of the machine is $\Omega(B^2)$.
- P processors take $\Theta(P)$ area not counting the network.
- If B is asymptotically bigger than \sqrt{P} , then the machine becomes “all wire” as $P \rightarrow \infty$.
- Similar reasoning applies for a three-dimensional machine:
 - ▶ Cross section area must be $\Omega(B)$.
 - ▶ Diameter must be $\Omega(\sqrt{B})$.
 - ▶ Volume must be $\Omega(B^{3/2})$.
 - ▶ If $B > O(P^{2/3})$, then the machine is asymptotically all wire.

Performance Considerations: Bandwidth

- How many bytes per-second can we send between two processors?
 - ▶ May depend on which two processors: neighbours may have faster links than spanning the whole machine.
- Bisection **width**: find the **worst** way to divide the processors into to sets of $\sim P/2$ processors each.
 - ▶ How many directed edges cross from the first set to the second.
 - ▶ The bisection width is an integer – a property of the interconnect graph.
 - ▶ If we multiply this by the bandwidth (bytes/second) we get the physical quantity of the bisection **bandwidth**.
- Bisection bandwidth is often much less than peak bandwidth.
 - ▶ Example: a mesh with $\sqrt{N} \times \sqrt{N}$ processors.
 - ★ Peak bandwidth: $\mathcal{O}(N)$ – every processor has links to 4 neighbours. A “checkerboard” partition realizes this bandwidth.
 - ★ Bisection bandwidth: $\mathcal{O}(\sqrt{N})$ – divide the mesh in half with a vertical line down the middle.
- Location matters.
 - ▶ It's 1:40pm – do you know where your data is?

Performance Considerations: Latency & Cost

- Latency

- ▶ How long does it take to send a message from one processor to another?
 - ★ Typically matters the most for short messages.
 - ★ Round-trip time is often a good way to measure latency.

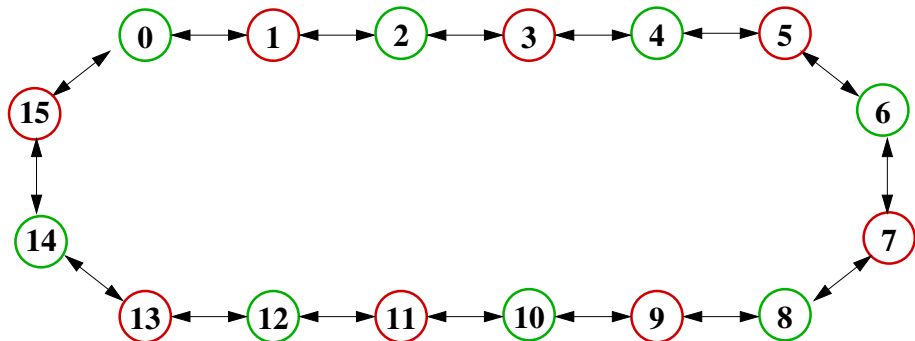
- Cost

- ▶ How expensive is the interconnect – it may dominate the total machine cost.
 - ★ Cost of the network interface hardware.
 - ★ Cost of the cables.

Real-life networks

- InfiniBand is becoming increasingly prevalent
- Peak bandwidths $\geq 6\text{GBytes/sec}$.
 - ▶ achieved bandwidths of 2–3GB/s.
- Support for RDMA and “one-sided” communication
 - ▶ CPU A can read or write a block of memory residing with CPU B.
- Often, networks include trees for synchronization (e.g. barriers), and common reduce and scan operations.
- The MPI (message-passing interface) evolves to track the capabilities of the hardware.

Bandwidth Matters



- Assume each link has a bandwidth in each direction of 1Gbyte/sec.
- Each node, i , sends an 8Kbyte message to node $(i + 1) \bmod P$, where P is the number of processors?
- How long does this take?
- What if each node, i , sends an 8Kbyte message to node $(i + P/2) \bmod P$?

What this means for programmers

- Location matters.
 - ▶ The meaning of location depends on the machine.
 - ▶ Challenges of heterogeneous machines.
 - ★ Communication on chip is much higher bandwidth and lower latency than communication across a circuit board or backplane (e.g. between blades).
 - ★ Communication between processors on the same circuit board or backplane is much faster than communication between such clusters.
- Message passing makes it possible to account for heterogeneity.
 - ▶ We can adapt simple algorithms to work on a heterogeneous architecture.
 - ▶ But we need to have the right models.

Summary

- Message passing machines have an architecture that corresponds to the message-passing programming paradigm.
- Message passing machines can range from
 - ▶ Clusters of PC's with a commodity switch.
 - ▶ Clouds: lots of computers with a general purpose network.
 - ▶ Super-computers: lots of compute nodes tightly connected with high-performance interconnect.
- Many network topologies have been proposed:
 - ▶ Performance and cost are often dominated by network bandwidth and latency.
 - ▶ The network can be more expensive than the CPUs.
 - ▶ Peta-flops or other instruction counting measures are an indirect measure of performance.
- Implications for programmers
 - ▶ Location matters
 - ▶ Communication costs of algorithms is very important
 - ▶ Heterogeneous computing is likely in your future.

Preview

September 28: Superscalar Architectures

Oct. 1 – Oct. 10: Performance Analysis

October 12 – 17: Sorting

October 19: Midterm review

October 22: Midterm

Oct. 24 – Nov. 30: Data Parallel Computing, GPUs, and CUDA

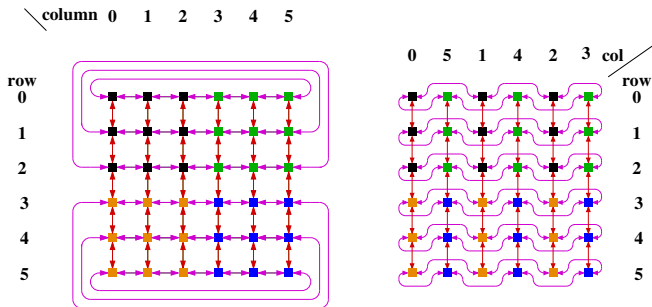
Review

- Consider a machine with 4096 processors.
- What is the maximum latency for sending a message between two processors (measured in network hops) if the network is
 - ▶ A ring?
 - ▶ A crossbar?
 - ▶ A 2-D mesh?
 - ▶ A 3-D mesh?
 - ▶ A hypercube?
 - ▶ A binary tree?
 - ▶ A radix-4 tree?

Supplementary Material

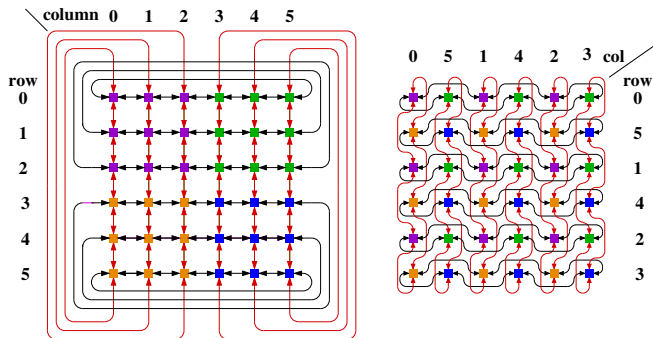
- [Message-passing origami](#): how to fold a mesh into a torus.
- [How big is a hypercube](#): it's all about the wires.

From a mesh to a torus (1/2)



- Fold left-to-right, and make connections where the left and right edges meet.
- Now, we've got a cylinder.
- Note that there are no “long” horizontal wires: the longest wires jump across one processor.

From a mesh to a torus (2/2)



- Fold top-to-bottom, and make connections where the top and bottom edges meet.
- Now, we've got a torus.
- Again there are no "long" wires.

How big is a hypercube?

- Consider a hypercube with $N = 2^d$ nodes.
- Assume each link can transfer one message in each direction in one time unit. The analysis here easily generalizes for links of higher or lower bandwidths.
- Let each node send a message to each of the other nodes.
- Using dimension routing,
 - ▶ Each node will send $N/2$ messages for each of the d dimensions.
 - ▶ This takes time $N/2$.
 - ▶ As soon as one batch of messages finishes the dimension-0 route, that batch can continue with the dimension-1 route, and the next batch can start the dimension 0 route.
 - ▶ So, we can route with a throughput of $\binom{N}{2}$ messages per $N/2$ time.

How big is a hypercube?

- Consider a hypercube with $N = 2^d$ nodes.
- Assume each link can transfer one message in each direction in one time unit. The analysis here easily generalizes for links of higher or lower bandwidths.
- Let each node send a message to each of the other nodes.
- Using dimension routing,
we can route with a throughput of $\binom{N}{2}$ messages per $N/2$ time.
- Consider any plane such that $N/2$ nodes are on each side of the plane.
 - ▶ $\frac{1}{2} \binom{N}{2}$ messages must cross this plane in $N/2$ time.
 - ▶ This means that at least $N - 1$ links must cross the plane.
 - ▶ The plane has area $O(N)$.

How big is a hypercube?

- Consider a hypercube with $N = 2^d$ nodes.
- Assume each link can transfer one message in each direction in one time unit. The analysis here easily generalizes for links of higher or lower bandwidths.
- Let each node send a message to each of the other nodes.
- Using dimension routing,
we can route with a throughput of $\binom{N}{2}$ messages per $N/2$ time.
- Consider any plane such that $N/2$ nodes are on each side of the plane.
 - ▶ The plane has area $O(N)$.
- Because the argument applies for *any* plane, we conclude that the hypercube has diameter $O(\sqrt{N})$ and thus volume $O(N^{\frac{3}{2}})$.
- Asymptotically, the hypercube is all wire.