Message-Passing Parallel Computers

Mark Greenstreet

CpSc 418 – Oct. 8, 2013

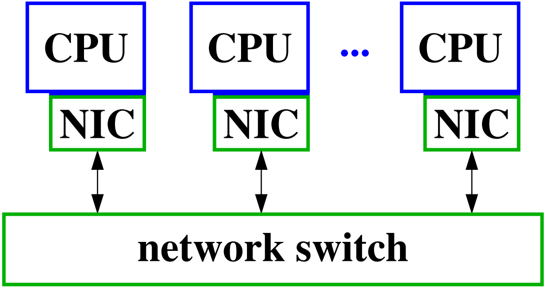
Outline:

- Message Passing Computers
- Examples

Objectives

- Familiar with typical network topologies:
  - rings, meshes, crossbars, tori, hypercubes, trees, fat-trees.
  - why some topologies become “all wires”.
- Implications for programming
  - bandwidth bottlenecks
  - latency considerations
  - location matters
  - heterogeneous computes.

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.

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 Tianhe 2 Machine

- The world's fastest (Linpack) super-computer (as of June 17, 2013)
- 16,000 nodes each with
  - Two Intel Xeon E5-2692 processors (12 cores per processor)
  - Three Intel Xeon Phi 31S1P processors (57 cores per processor)
    - Each processor has 61 cores.
    - 57 are exposed to the programmer, the other 4 are disabled.
    - This allows the chip to work with less than perfect fabrication.
- Total of 3,120,000 cores
- LINPACK performance: 33 PFlops
- Power consumption 18MW (computer) + 24MW (cooling) = 42MW.
  - Roughly 42,000 homes.
  - At 0.05/KWH, that's $18M/year, just for the power bill.
- Interconnect: Fat-tree
- Programming model: A version linux with MPI tuned for this machine.

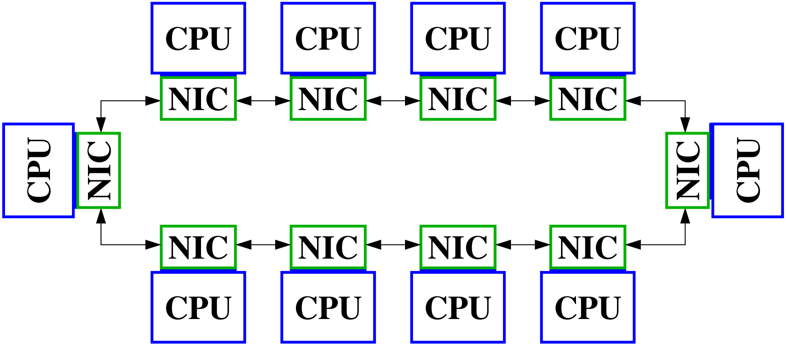
The Westgrid Clusters

- Clusters at various Western Canadian Universities (including UBC).
- Up to 9600 cores.
- Available for research use.

Network Topologies

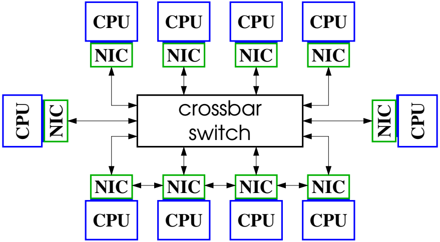
- Network topologies are to the message-passing community what cache-coherence protocols are to the shared-memory people:
  - **Lots** of papers have been published.
  - Machine designers are always looking for better networks.
  - Network topology has a strong impact on performance, the programming 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



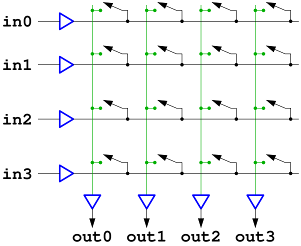
- Advantages: simple.
- Disadvantages:
  - Worst-case latency grows as $O(P)$ where P is the number of processors.
  - Easily congested – limited bandwidth.

Star Networks

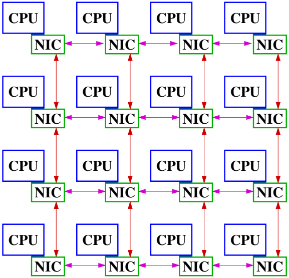


- Advantages:
  - Low-latency – single hop between any two nodes
  - High-bandwidth – no contention for connections with different sources and destinations.
- Disadvantages:
  - Amount of routing hardware grows as $O(P^2)$.
  - Requires lots of wires, to and from switch –
    Imagine trying to build 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

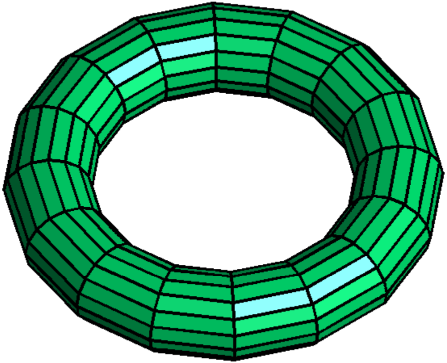


Meshes



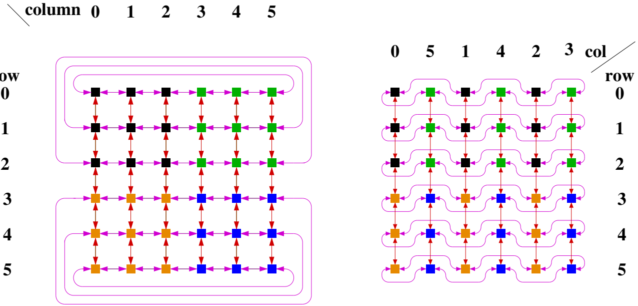
- Advantages:
  - Easy to implement: chips and circuit boards are effectively two-dimensional.
  - Cross-section bandwidth grow with number of processors – more specifically, bandwidth grows as $\sqrt{P}$.
- Disadvantages:
  - Worst-case latency grows as $\sqrt{P}$.
  - Edges of mesh are “special cases.”

Tori



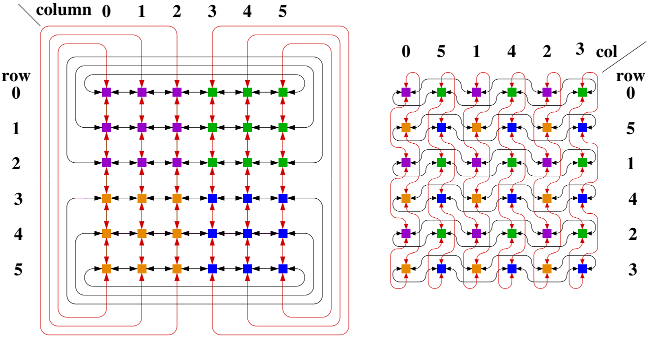
- Advantages:
  - Has the good features of a mesh, and
  - No special cases at the edges.
- Disadvantages:
  - Worst-case latency grows as $\sqrt{P}$.

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)



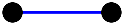
- Fold top-to-bottom, and make connections where the top and bottom edges meet.
- Now, we've got a torus.
- Again there are no "long" wires.

Hypercubes

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

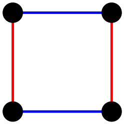
Hypercubes

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



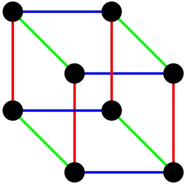
Hypercubes

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



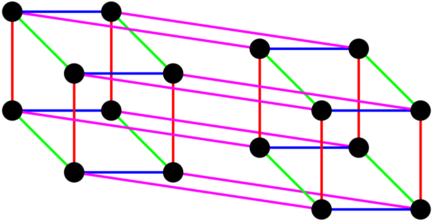
Hypercubes

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



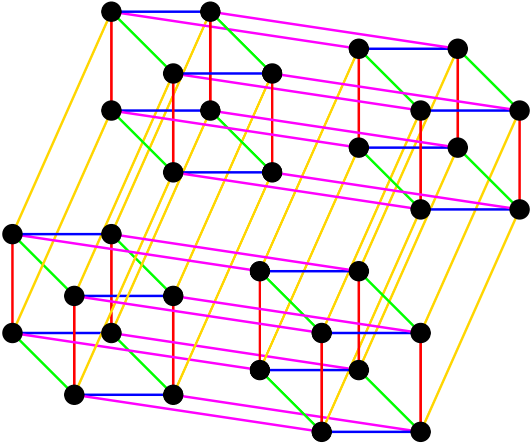
Hypercubes

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



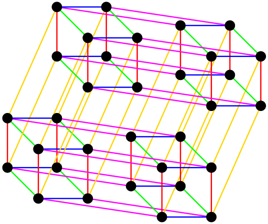
Hypercubes

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



Hypercubes

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



- Advantages
  - Small diameter (log *N*)
  - Lots of bandwidth
  - Easy to partition.
  - Simple model for algorithm design.
- Disadvantages
  - Needs to be squeezed into a three-dimensional universe.
  - Lots of long wires to connect nodes.
  - Design of a node depends on the size of the machine.

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
```

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 $\left(\begin{array}{c} N \\ 2 \end{array}\right)$ 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,

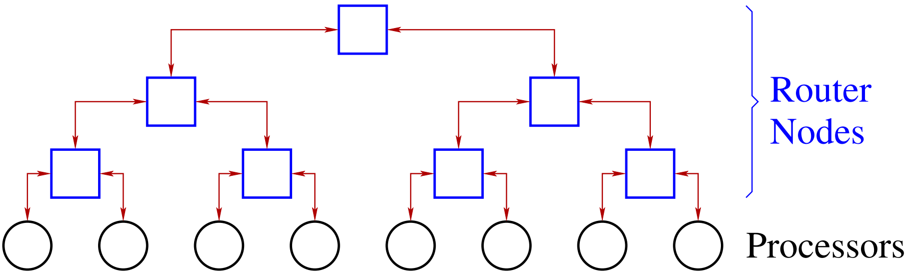
  we can route with a throughput of $\left(\begin{array}{c} N \\ 2 \end{array}\right)$ messages per $N/2$ time.
- Consider any plane such that $N/2$ nodes are on each side of the plane.
  - $\frac{1}{2}\left(\begin{array}{c} N \\ 2 \end{array}\right)$ messages must cross this plane in $N/2$ time.
  - This 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,

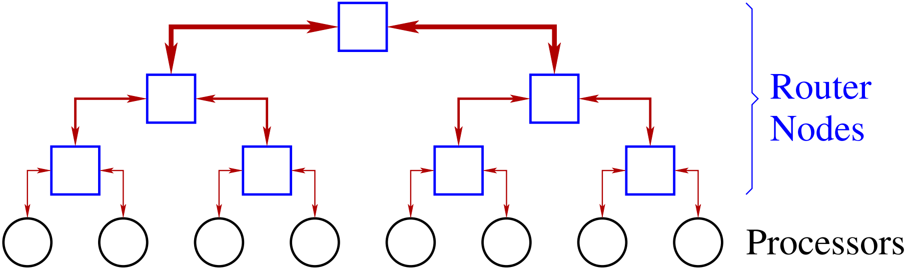
  we can route with a throughput of $\binom{N}{2}$ messages per $N/2$ time.
- Consider any plane such that $N/2$ nodes are on each side of the plane.
  - The plane has area $O(N)$.
- Because the argument applies for *any* plane, we conclude that the hypercube has diameter $O(\sqrt{N})$ and thus volume $O(N^{\frac{3}{2}})$.
- Asymptotically, the hypercube is all wire.

Trees



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

Real-life networks

- 3D Tori: Titan (Oakridge National Labs, USA)
- Trees and fat-trees: Tianhe-2 (China)
- 5 and 6D tori: K-machine (Japan), Sequoia (Lawrence Livermore Labs, USA).

What this means for programmers

- Location matters.
  - The meaning of location depends on the machine.
  - Getting a good programming model is hard.
  - Challenges of heterogeneous machines.
- What it means for different kinds of computers
  - Supercomputers
  - Clouds
  - PCs of the future(?)

Our favorite problems

- Reduce: an obvious fit for a tree.
  - Works well on fat-tree or hypercube – the tree is a subnetwork.
  - Easily adapted to mesh or torus
- Matrix-multiply
  - Tree: the root is a bottleneck
  - Mesh or torus: there are good algorithms – I'm looking for a simple one.
  - Fat-tree: works well for sufficiently large α ($\alpha > \frac{1}{2}$?)

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:
  - Easy to have the machine become "all wire".
  - Performance and cost are often dominated by network bandwidth and latency.
  - 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

October 10: Models of Parallel Computation

Reading:	Lin & Snyder, chapter 2, pp. 43–59.
Homework:	Homework 2 due.

October 15: Peril-L, Reduce, and Scan

Reading:	Lin & Snyder, chapter 3, pp. 112–125.

October 17: Work allocation

Reading:	Lin & Snyder, chapter 3, pp. 125–142.

October 22: Midterm

October 24: PReach: a parallel model checker, and an example of a large-scale Erlang application

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?
- Why are crossbars and hypercubes only used for machines with a “small” number of processors?