# Message-Passing Parallel Computers

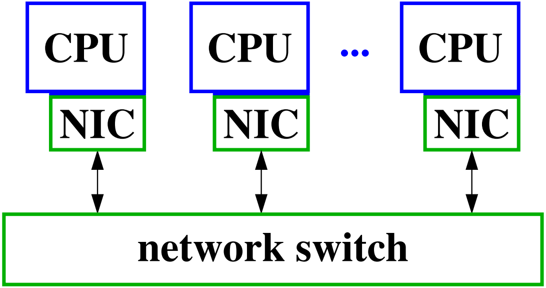
Mark Greenstreet

CpSc 448B – Oct. 6, 2011

Outline:

- Message Passing Computers
- Examples

# 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 exampe:
  - 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 K Machine

- Currently, the world's fastest super-computer
- 68,544 8-core SPARC processors.
- LINPACK performance: 8.162 PFlops
- Power consumption 9.89 MW
  - Roughly 10,000 homes.
  - Operating costs estimated at $10M/year.
  - But, it's one of the best supercomputers for PFlops/Watt
- Still under construction.
  - Final version will have 88,128 8-core processors
  - Projected performance $>$ 10 PetaFlops.
- Interconnect:
  - "6D" torus network (called "Tofu").
  - 10 Gbytes/sec, bi-directional, for each link.
  - Hardware support for reduce operations and synchronization.
- Programming model:
  - A version of MPI tuned for this machine.
  - Supports topology-aware programs.
  - The interconnect is designed to make it easy to partition the machine so different jobs can run on different partitions.

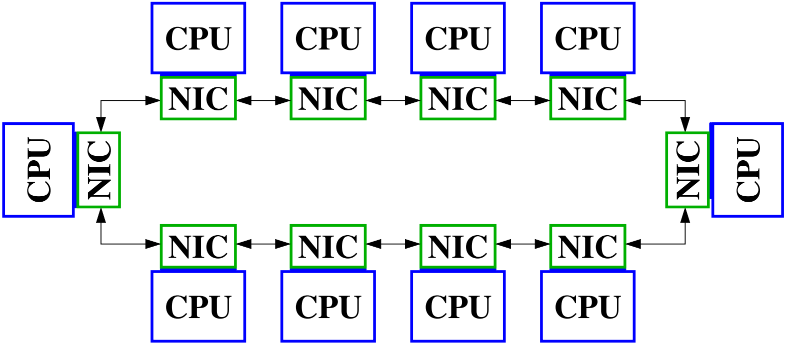
# The Westgrid Clusters

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

# Network Topologies

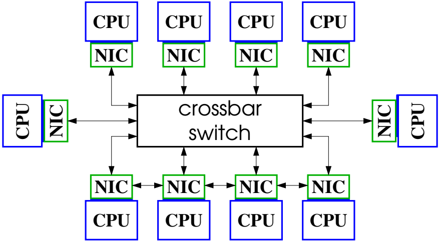
- Network topologies are to the message-passing community what cache-coherence protocols are to the shared-memory people:
  - **Lots** of papers have been published.
  - Machine designers are always looking for better networks.
  - Network topology has a strong impact on performance, the programming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



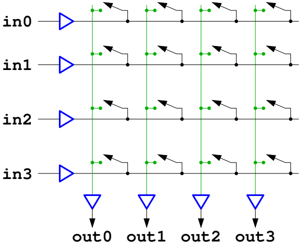

- Advantages: simple.
- Disadvantages:
  - Worst-case latency grows as $O(P)$ where $P$ is the number of processors.
  - Easily congested – limited bandwidth.

# Star Networks

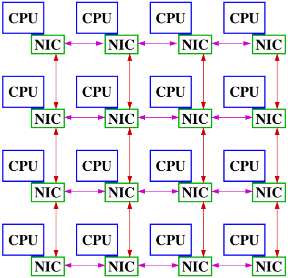



- Advantages:
  - Low-latency – single hop betweeen any two nodes
  - High-bandwidth – no contention for connections with different sources and destinations.
- Disadvantages:
  - Amount of routing hardware grows as $O(P^2)$.
  - Requires lots of wires, to and from switch –
    Imagaine trying to build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

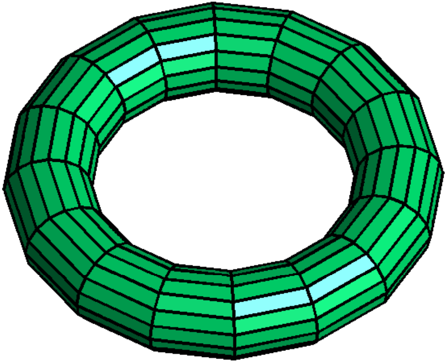


# Meshes



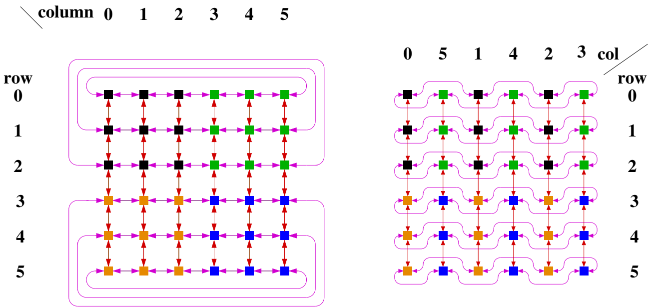

- Advantages:
  - Easy to implement: chips and circuit boards are effectively two-dimensional.
  - Cross-section bandwidth grow with number of processors – more specifically, bandwidth grows as $\sqrt{P}$.
- Disadvantages:
  - Worst-case latency grows as $\sqrt{P}$.
  - Edges of mesh are "special cases."

# Tori



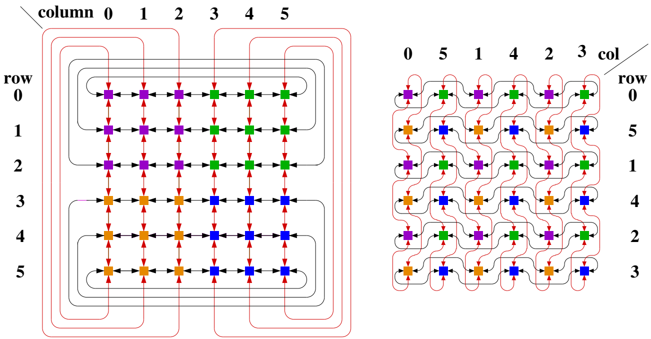
- Advantages:
  - Has the good features of a mesh, and
  - No special cases at the edges.
- Disadvantages:
  - Worst-case latency grows as $\sqrt{P}$.

# 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

# Real-life networks

- 3D Tori.
- Trees and fat-trees.
- 5 and 6D tori.

# What this means for programmers

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