Summary of last lecture

- Key requirements of HPC networks
  - extremely low latency, high bandwidth, scalable
  - low network diameter, high bisection bandwidth

- Torus networks (less common now)
  - Network diameter grows as $O(\sqrt[3]{N})$ where $N$ is the number of nodes

- Different types of routing algorithms:
  - Shortest path vs. non-minimal
  - Static vs. dynamic
Fat-tree network

- Most popular network topology
  - Low network diameter, high bandwidth
Fat-tree network

• Most popular network topology

• Low network diameter, high bandwidth
Fat-tree network

- Most popular network topology
  - Low network diameter, high bandwidth

Router/switch radix = number of ports = k
Fat-tree network

- Most popular network topology
  - Low network diameter, high bandwidth

Router/switch radix = number of ports = k
**Fat-tree network**

- Most popular network topology
  - Low network diameter, high bandwidth

Router/switch radix = number of ports = \( k \)
Fat-tree network

- Most popular network topology
  - Low network diameter, high bandwidth

Router/switch radix = number of ports = k
Fat-tree network

- Most popular network topology
  - Low network diameter, high bandwidth

Router/switch radix = number of ports = k
Pod = group of switches = k/2 switches
Fat-tree network

- Most popular network topology
  - Low network diameter, high bandwidth

Router/switch radix = number of ports = k
Pod = group of switches = k/2 switches
Fat-tree network

- Most popular network topology

  - Low network diameter, high bandwidth

Router/switch radix = number of ports = k
Pod = group of switches = k/2 switches
Fat-tree network

- Most popular network topology
  - Low network diameter, high bandwidth

Router/switch radix = number of ports = k
Pod = group of switches = k/2 switches
Max. number of pods = k
Fat-tree networks on the top500 list

- Infiniband EDR/FDR/HDR, Intel Omni-Path

Interconnect System Share

Interconnect Performance Share

https://www.top500.org/statistics/list, November 2020
Routing on a fat-tree

• Until recently, most fat-tree installations used static routing
  • Destination-mod-k (D-mod-k) routing
• Adaptive routing is now starting to be used
Variations on a full bandwidth fat-tree

Single-rail single-plane fat-tree
Variations on a full bandwidth fat-tree

Single-rail single-plane fat-tree (tapered)
Variations on a full bandwidth fat-tree

Dual-rail single-plane fat-tree
Variations on a full bandwidth fat-tree

Single-rail single-plane fat-tree
Variations on a full bandwidth fat-tree

Dual-rail dual-plane fat-tree
Dragonfly network
IBM PERCS network

- All-to-all connections within each group

One supernode in the PERCS topology
IBM PERCS network

- All-to-all connections within each group
Cray Aries network

• Row and column all-to-all connections within each group
Cray Aries network

- Row and column all-to-all connections within each group

A group with 96 Aries routers

- Column all-to-all (black) links
- Row all-to-all (green) links

Abhinav Bhatle (CMSC714)
Cray Aries network

- Row and column all-to-all connections within each group

![Diagram of Cray Aries network]

- Traffic in current fat-tree networks is usually forwarded using a static routing algorithm, meaning that all messages are configured to let jobs share the interconnect. For example, on the IBM Blue Gene machines, jobs are always placed so that communication patterns, they can conflict on row and column links. Outside the group, and even if both have localized (e.g., nearest-neighbor) applications can be assigned to the same routers within a group, and even if both have localized (e.g., nearest-neighbor) applications can be assigned to the same routers within a group. By contrast, the dragonfly topology is becoming another popular choice outside the group, and even if both have localized (e.g., nearest-neighbor) applications can be assigned to the same routers within a group. By contrast, the dragonfly topology is becoming another popular choice for interconnection networks in post-petascale supercomputing. The dragonfly topology is becoming another popular choice for interconnection networks in post-petascale supercomputing.
Network comparisons

<table>
<thead>
<tr>
<th>Network topology</th>
<th>#nodes/router</th>
<th>#links/router</th>
<th>Maximum system size (#nodes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>All-to-all (A2A) dragonfly</td>
<td>k/4</td>
<td>k/2 (L), k/4 (G)</td>
<td>((k/2 + 1)^2 \times (k/4 + 1) \times k/4)</td>
</tr>
<tr>
<td>Row-column (RC) dragonfly</td>
<td>k/6</td>
<td>2k/3 (L), k/6 (G)</td>
<td>((k/3 + 1)^4 \times (k/6 + 1) \times k/6)</td>
</tr>
<tr>
<td>Express mesh (3D, gap=1)</td>
<td>k/4</td>
<td>3k/4</td>
<td>((k/4 + 1)^3 \times k/4)</td>
</tr>
<tr>
<td>Fat-tree (three-level)</td>
<td>k/2</td>
<td>k/4</td>
<td>(k/2 \times k/2 \times k)</td>
</tr>
</tbody>
</table>

Figure 1: Plot showing the largest systems that can be built.
Questions?

Abhinav Bhavele
5218 Brendan Iribe Center (IRB) / College Park, MD 20742
phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu