1 Introduction

There are two methods to model a network:

- create a database of port connections
- create autonomous switches which direct packets to other components

For this model I will be using a database of connections.

1.1 objectives

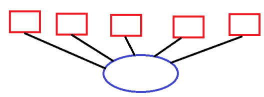
If I can show a given topology is better than some other, that is useful.

If I can show how one switch versus another (differentiated by number of ports) affects performance, that is useful.

1.2 Does Topology matter? A toy model example

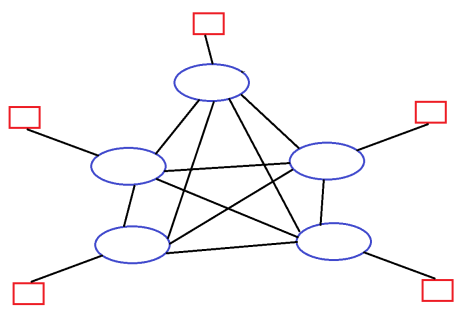
An optimal all-to-all design would have each compute node with $N-1$ ports and no switches (0 hops). For N¿1000 this becomes difficult. Thus we introduce switches.

Suppose now we have $N=5$ compute nodes with 1 port each. Then the optimal network design (fewest hops) is to have a 5 port switch:



Here number of hops is 1 for each compute node, with 10 pairs (=5*4/2).

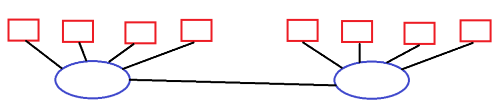
The other extreme would be to use five of these same switches with only one compute node per switch:



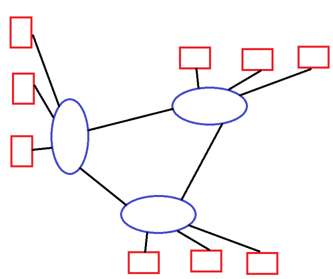
Here number of hops is 2 for each compute node.

Clearly we are spending too much money on switches for the same number of compute nodes. However, this increased hop count (2) also lowers conjestion.

As a compromise, we can use two switches and increase the number of compute nodes:



8 nodes and 2 switches: 12 pairs with 1 hop, 16 pairs with 2 hops.



1 or 2 hops, 9 nodes and 3 switches.

Notice a few constraints were followed:

- each switch has same number of ports
- each compute node has one port
- each switch is fully occupied
- each node can reach every other node
- each switch has at least one computer connected to it

The number of permutations increases when we have more than 9 compute nodes and only 5 ports. Even worse, consider when there are multiple ports per computer (but much less than the number of computers).

The parameter space includes

- number of computers
- number of ports per computer
- number of switches
- number of ports per switch

Metrics:

- hop count for each pair
  - average hop count
  - maximum hop count
- bisection bandwidth

1.3 Permutations

The number of unique pairs on a network swith N computers is $N(N-1)/2$.

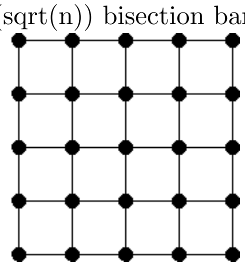
When $N = 4$ then there are 6 pairs. $N = 100$ is 4950 pairs. $N = 10,000$ is 49,995,000 pairs.

2 standard networks

See http://www.cs.nmsu.edu/ pfeiffer/classes/573/notes/topology.html

2.1 Mesh

Mesh (and the related torus) can be of n dimensions, commonly n =2, 3, 6. Useful for physical sciences due to local communication (nearest neighbors). Mesh networks are well-characterized. "Meshes have O(n) cost, O(sqrt(n)) bisection bandwidth, O(n) aggregate bandwidth, and O(sqrt(n)) latency."



2.2 Hypercube

"the latency is is O(log N). There are N processors, each with log2N interfaces, so the cost is O(N log N). and all the processors can use their links simultaneously, so our aggregate bandwidth is O(N). The bisection bandwidth is O(log N)."

2.3 Fat Tree

http://en.wikipedia.org/wiki/Fat_tree

2.4 Flattened Butterfly

2.5 Dragonfly

http://research.google.com/pubs/pub34926.html

Fractal

2.6 Clos Network

http://en.wikipedia.org/wiki/Clos_network

2.7 randomly-connected networks

For networks supporting physical models (i.e., mesh), it makes sense to think about dimension, perimeter/surface area, area/volume. This may not apply to scale-free topologies.

If the topology turns out to be scale free, then we wouldn't need to model 1E6 endpoints (that is desirable).

3 more than one port per endpoint

If each endpoint has only one network connection, then we can model a switch-only network. A switch with 4 ports not connected to other switches would have 4 endpoints.

4 random network creation

For a given {(number of computers), (number of ports per computer), (number of ports per switch)}, should random computers be plugged into random port switches, or should random switches be connected first?

Whether local symmetry (same number of computers plugged into each switch) is a hinderance, benefit, or irrelevant is not clear to me.

Random connections lead to unexpected paths. This could be good, bad, or inconsequential.

5 route enumeration

1. For each computer, see what other computers are available on the same switch (1 hop)
2. For each computer, see what other computers are two switches away (2 hops)
3. ...

When a switch has had all of its computers touched for a given iteration, then we should mark that switch as "touched" (doesn't need to be queried again for current iteration). That is, mark a switch to indicate "all locally-attached computers have number of hops known." This should reduce search time.

6 future task list

Once the hop counter is implemented, it would be useful to validate it against analytic values for mesh, torus, fat tree topologies.