Christopher Osborn
6.911 - Networks
Presentation Writeup

A Networking Overview

As individual microprocessors in high-performance computing systems reach ever greater speeds, performance begins to depend less on the speed of processors, and more on the system that supplies them with data. This system is the network, an often ignored but very important part of any complex computer. Networking is an extremely broad topic in information science, second perhaps to computation itself in terms of depth and importance. Taken simply, however, a network is no more than a system by which one processing element can send information to another. As such, the critical parameters of a network measure information flow. One important metric is the bisection bandwidth, or the maximum rate at which a network can move information across a line that divides the nodes into two equal groups. Just as important for tightly-coupled multiprocessing is the time required to transfer a message between nodes, called the latency. A large and continuous research effort dedicates itself to improving these two numbers, resulting in a huge range of approaches to network design.

In addition to bandwidth, latency, and other raw statitistics describing capacity, the topology of a network is an important characteristic. Traditionally, many people have found it useful to divide all topologies into two large groups, saying that a network is either static or dynamic. This is somewhat misleading, because very few real topologies submit to such easy classification, instead falling somewhere on a broad spectrum between static and dynamic. A static network topology is so called because it seeks to provide unchanging data paths that fit the communication needs of the application. In a perfectly static topology, the network itself is a logically passive device that simply provides a collection of pipes with known and dependable endpoints. The ideal static network is the completely connected graph, where a dedicated channel exists between each and every pair of nodes. In this case, the network does no routing of any sort, and the sending node specifies a receiver simply by placing the message on the correct channel. Occupying the opposite end of the spectrum are dynamic networks, which make very different assumptions about the role of the network in handling messages. Dynamic networks attempt to use switching and arbitration to share a smaller number of physical links among more nodes. The logical extreme would be a simple bus

topology, which has exactly one physical channel that is shared by all attached nodes. In actual practice, high performance networking systems rarely look like either logical extreme. Given that both static and dynamic strategies have definite tradeoffs, most actual systems choose some well-balanced middle ground.

Although the ideal static network is a completely connected graph, this topology is rarely used for networks of more than a few nodes because the node degree (the number of links entering or exiting a node) is too large. Each node in such a network needs a separate physical connection to every other node, making complete connection attractive in theory and nightmarish in practice. For this reason, real computers that use static interconnection settle for some subset of a complete graph, with the choice of subsets tailored to the particular needs of the application. Common topologies of this type include hypercubes (Connection Machine 2), meshes (Illiac IV), and trees (Connection Machine 5). Each of these topologies has specific communication patterns for which it is especially well suited, and if a computing application can be parallelized to take advantage of these patterns, then it may enjoy very low latency (node-to-node message travel time). This low latency is enhanced by the fact that static networks generally require only very simple routing, meaning that message processing time is kept to a minimum.

While it is true that the routing necessary to move messages in a static network is often quite simple as a result of convenient geometry, some sort of routing is still necessary. Actual networks are subsets of a complete graph, and in general, the smaller the subset, the more dynamic a static network has to be. Even static networks share physical links to some extent, and thus they encounter the same contention problems that plague dynamic networks. Because a single physical link belongs to many possible data paths, packets may collide at network junctures, in which case the congested switch must either block one of the packets, misdirect it to an available link, drop it completely, or accept it into a buffer for later transmission. Each of these policies has drawbacks. Blocking of messages introduces the potential for deadlock, in which multiple routing switches wait on each other in an unbreakable cycle. Such a situation may seem like an unlikely coincidence, but with a fast network, small probabilities are multiplied by huge numbers of trials. In practice, a network in which deadlock is possible will actually deadlock quite frequently. Packet misdirection prevents deadlock but allows for livelock, a phenomenon in which packets are eternally shuffled around among a small group of switches. Buffering packets at a congested switch is a good idea, but does not completely eliminate the problem. Buffers can always overflow, in which case the

switch must deal with further packets in some other way. Dropping packets completely is not desirable, but if the higher-level software is properly designed, it will deal gracefully with dropped packets and the network will function. Most networks that must deal with switching contention use some combination of these methods.

Whereas static networks are specially designed to fit the needs of a particular type of application, dynamic networks take a more general approach. Dynamic networks seek to simplify the design and physical construction of a network by sharing communication channels. Whereas static networks are almost always custom solutions, dynamic networks are built under much less controlled circumstances. For this reason, their routing and arbitration schemes make fewer assumptions about precise topology. A side effect of generality is that inexpensive off-the-shelf implementations are practical. Whereas the networking architecture for a custom-built supercomputer may cost millions of dollars, a Beowulf cluster uses interconnect that costs only a few hundred dollars per node. As always, there are tradeoffs. The more elaborate processing required for routing and arbitration in a dynamic network leads to relatively long message latencies, making highly dynamic networks awkward for shared memory applications. Also, such general networking systems tend to place the responsibity for packet acknowledgment and deadlock avoidance with higher-level software, adding to the computational work performed by each node. Nonetheless, the popularity of Beowulf clusters demonstrates that dynamic networks have a place in high-performance computing.

As with most engineering problems, almost every part of network design presents tradeoffs. It is difficult, therefore, to determine the primary direction of current network research, because it is attacking many fronts simultaneously. In any case, there is ample room for improvement on modern network systems. Even the fastest networks are substantially slower than the fundamental limit imposed by the speed of light, and improvements in this latency would have a great effect on multiprocessing performance. There is also the possibility that new methods of parallelizing applications will lead networking in previously unexplored directions. In any case, networking technology is by no means an exhausted science, and much work remains to be done.