CS-534: Packet Switch Architecture
Spring 2004

Department of Computer Science

0.2 Central Issues in Packet Switching: an Overview

This section presents an overview of the central issues and problems faced in high-performance networking, and the main alternatives for solving them. In each particular application domain, the parameters of the problem have different values, so different alternatives may be chosen to solve a given problem, leading to different architectures. Yet, because all problems have a few, common sources, these different architectures have several common features, which we will try to identify in this course.

High-performance networking strives to achieve the highest possible throughput, the lowest possible latency, high utilization of the expensive resources, fair allocation of resources to competing users (QoS guarantees), or combinations of these. When performance is pushed to such extreme points, some pieces of equipment are stressed to operate at rates in excess of what conventional processors can achieve, thus requiring the support of *dedicated, special-purpose hardware or processors* . This course deals especially with the architecture of these latter parts of the networking equipment.

0.2.1 Distributed, Multi-Party Communication

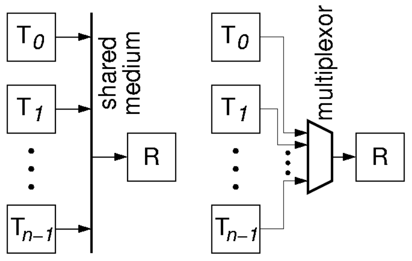
Networking deals with *multi-party* communication, in the sense that there are more parties that can talk to a receiver than the receiver can listen to at once. The system where communication takes place is *distributed*, i.e. it extends physically (geographically) over distances such that it is difficult for the communicating parties to get coordinated with each other using means other than through the network itself. Under these circumstances, the basic problem is *contention* for shared resources, usually in the lack of prior coordination.

Output Contention:

The first central problem to deal with in networking is *output contention*, which is the attempt by multiple sources to "simultaneously" transmit information to a given output (destination) party at an aggregate rate in excess of the capacity of that output. Under these circumstances, the sources (or the streams of information that they injected into the network) *contend* (compete) for access to the desired output port of the network, hence the name "output contention". Usually, the resource under contention is *throughput* (amount of information delivered per unit of time), but contention for other resources (e.g. buffer space) is also possible.

Note that the existence of this problem --the presence of output contention-- is *not* the responsibility of the network: it is the *users* of the network that (a) are numerous, and (b) do not have sufficient capacity, each, to listen to all of the potential transmitters at once. Thus, the network is called upon, basically, to solve a problem of its users; of course, if the users had unbounded receive capacity, this problem would not go away, but would manifest itself as its dual --internal blocking-- as will be discussed shortly.

Elementary Case: Single Resource Contention.



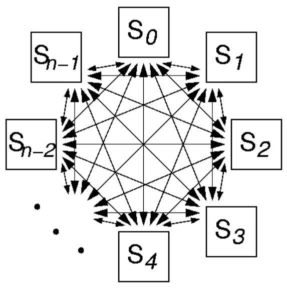
Output contention manifests itself even in the most elementary case: a network consisting of multiple transmitters and one receiver, as shown in the figure, where the aggregate rate of the transmitters exceeds the rate capacity of the receiver. We can study the essence of output contention using this simple case. A network like this can be built using a shared medium (e.g. traditional Ethernet), as illustrated on the left, or by running dedicated (point-to-point) links from each transmitter to the receiver, then using a multiplexor to select which piece of information from which source will be routed to the receiver at each time, as shown in the right. In the former case, assume that the rate of the shared medium is the same as the rate of the receiver; then, to contend for the receiver is the same as to contend for the shared medium. With many shared media, transmitters can directly sense the presence of contention, and can then back off or postpone their transmissions. In larger networks, output contention manifests itself at a later time, and must be handled in a corresponding manner.

Handling Short or Long Term Contention by Buffering, Dropping, Access Control, and Flow Control:

If the network sources want to transmit information at an aggregate rate exceeding the capacity of (a) receiver(s), then there are only three alternatives to handle this problem (note that the problem can be handled by a combination of these methods):

1. **Buffer** the information in excess of the reveiver capacity, in some buffer memory, and transmit it a later time. This only applies to *short-term* contention, because if the rate mismatch persists for an unbounded period of time, an unbounded amount of memory will be needed, which is not feasible. Issues to be resolved in implementing this solution are:
   - *Buffer Memory Architecture:* Where to place the buffer memory(ies) and how to organize them?
   - *Scheduling for Quality of Service (QoS):* Which part of the information to deliver immediately, and which part to buffer, and for how long?
2. **Drop** the information in excess of the reveiver capacity. This is simpler than buffering, but leads to poor or unacceptable QoS, depending on the application. In applications where all information is needed, some protocol is put in place (usually in the end-user stations, but can also be in the network) for retransmitting the droped information; the end result is similar to buffering, but the method, cost, and performance differ a lot.
3. **Coordinate** (notify, control) the sources so that they properly adjust their rates. By contrast to buffering, this applies mostly to *long-term* contention (in some cases, it can also apply to short-term contention). Given the distributed nature of the network, such coordination inevitably involves delays. Coordination may be performed before or *after* the sources start their transmissions:
   - *Access Control* is the case where coordination is required *before* transmission starts; this places less stress on network buffering, a different kind of stress on network control, introduces delays before transmission can start, and raises issues of traffic predictability and resource utilization.
   - *Flow Control* is used to coordinate the rates of the sources *after* transmission starts; flow control is of a more dynamic nature when compared to access control.

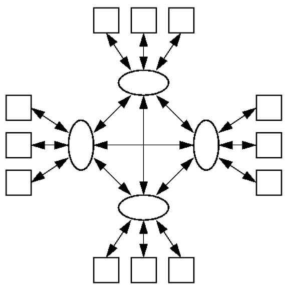
0.2.2 Problems of Scale: Avoiding All-to-All Wiring



all–to–all interconnection

The second central problem in networking is to *scale* the interconnect to *very large* numbers *n* of end-stations, S_i, while avoiding the unacceptably high $O(n^2)$ cost of the all-to-all interconnection pattern illustrated in the top part of the figure. For few and low-rate links, the "all-to-all" pattern can be implemented --as we will see-- by *time-multiplexing* all inputs into one, high-rate link, and then *demultiplexing* this link into all outputs. For higher rate links, time-multiplexing is not viable, due to clock-frequency limitations; the "all-to-all" pattern can still be implemented using the interconnection topology known as the *crossbar* , which is the most elementary space-switching architecture.

Multi-Stage Interconnection:


multi–stage interconnection

The cost of the crossbar switch grows with the square of the number of its I/O ports; for modest fan-in/fan-out --usually up to several tens of ports-- this cost is low enough for the crossbar to be the architecture of choice. For higher numbers of ports, though, *multi-stage* architectures, or "switching fabrics", can offer comparable performance at lower cost. For even higher port counts, we are forced to limit performance to lower levels in order to achieve acceptable cost; multi-stage networks are still the solution. As illustrated in the bottom part of the figure, a multi-stage network uses more than one switches between a source and a destination station; in this way, network links are *shared* among multiple I/O ports, thus reducing the cost of the interconnection. The presence of multiple stages further complicates the issues of coordination among the multiple sources that were discussed above, because contention for resources may now occur further away (in terms of switch stages) from these sources: the network becomes an even more *distributed* system.

Internal Blocking: Output Contention Variant.

Multi-stage interconnection networks may suffer from *internal blocking:* specific flow patterns that would be feasible in an all-to-all interconnection architecture may not be feasible any more in the multi-stage network, due to intermediate, shared link limitations. Internal blocking is a variant of output contention: multiple sources "simultaneously" transmit information at an aggregate rate in excess of the capacity of one or more internal network links (or other resources); if we consider the sub-network up to the first "bottleneck" link, the phenomenon appears as output contention in that sub-network. Depending on the architecture of the multi-stage network, internal blocking may never appear ("non-blocking network"), or it may appear only in few, rare cases, or it may appear in many, frequent cases. Architecting networks for low internal blocking is one of our topics.

Up to the Home Page of CS-534