6.826 Principles of Computer Systems	**Fall Semester, 1993**

Lecture 15: November 3, 1993

Lecturer: Butler Lampson *Scribe: Cheuk-san Edward Wang*[1]

1 Overview

This lecture is mostly about the *network* layer of the ISO Reference Model. This layer builds arbitrary network from point-to-point messages. There are many kinds of existing networks with different mechanisms for transmitting messages, but these mechanisms all belong to a few categories.

Last lecture we described ways of converting electrical signals on wires to bit sequences (physical layer) and then to messages or packets with reasonable size (data link layer).

2 Specification of Network Layer

The network layer is about routing addressed messages given point-to-point links. A network as a whole can be view as providing point-to-point services among network hosts. A network provides only *unreliable communication*. Once a sender puts a message on the network, the message may be lost or duplicated. If several messages are sent by a sender, the messages may not be received in order. The only guarantee that the network provides is that if a message is received, it was actually sent by a sender. In other words, a message is delivered only if it was sent. This guarantee is achieved by computing checksums on messages. Figure 1 contains the specification of the network. `Get` models the fact that messages are not always received in order. If `m` is always the head of `q`, messages would be received in order. People have attempted to build networks that are more reliable than this specification. However, the stronger specifications are much harder to implement. Our specification is a good tradeoff between reliability and performance.

With this specification, we can build arbitrarily large networks. Logically, a network is a big multiplexor and demultiplexor. Physically, a network looks like a web with hosts (end nodes) connecting to routers (gateways) and routers interconnect with each other. Figure 2 gives the Spec code that models the physical topology of a network. In a real network, links are usually full duplex, but our program models them as only half duplex. With the topology `t` of a network, we can find the links going in and out of a router `r`.

```
FUNC In/Out(t,r) -> SET L
   = RET {l | t(l).src = r \/ t(l).sink = r}
```

We can also find all nodes on a path, the end nodes of a path, or all possible paths in a network.

[1] These notes are based on notes prepared by Nikhil Gautam and Thomas Lee in Fall '92, the latter being based on notes prepared in Fall '91 by Clifford Tse, Uri Blank and Rainer Gawlick.

```
TYPE
   M                   % message
   A                   % source or destination address
   SR = [s:A, r:A]     % sender receiver pair

VAR
   q : SR -> Multiset M    % represented by a sequence

% External actions
   PROC Send(sr,m) =
      q(sr) := q(sr) ++ m

   PROC Get(sr) -> M =
      VAR m | m IN q(sr) =>
                q(sr) := q(sr) -- m;
              RET m

% Threads that cause unreliability
   THREAD Lose(sr) = Get(sr)

   THREAD Dup(sr) =
      VAR m | m IN q(sr) =>
                q(sr) := q(sr) ++ m
```

Figure 1: Specification of a network.

```
TYPE
   L     % Link
   R     % Routers that performs switching
   H     % Hosts
   N = (R + H)    % Nodes
   SS = [src:N, sink:N]
   TOPOLOGY = L -> SS
   P = SEQ L      % path
```

Figure 2: Modeling the topology of a network.

3 Broadcast Networks

There are basically two strategies in implementing networks—broadcast or switching. We shall talk about only broadcast networks in this lecture.

The fundamental characteristics of a broadcast network is that abstractly, every message that is sent appears at all nodes. This characteristic is the interface specification of a broadcast network. Note that the interface specification is separate from the broadcast implementation. For example, a large Ethernet (extended LAN) can have many segments of bus structure each connected by bridges (switches). Abstractly, it still behaves as if all hosts are connected to one common wire. An extreme implementation of the broadcast network is the "personal Ethernet" where only two nodes share a physical connection.

There are three basic issues concerning a broadcast network.

1. **Addressing** Addressing is about of getting the message from the sender to the receiver. Because all messages go to every node, addressing becomes quite simple in a broadcast network. Every packet in the network has an address field and a message body. `P = (A,M)` To receive a host just compares its own addresses with the address field of each packet. The host does not need any global information.

2. **Arbitration** Because the medium is shared in a broadcast network, receiving is simple but arbitration is hard. All hosts share the bandwidth of the broadcast medium, but only one node can send at a time. There must be a scheduling mechanism. There are several types of multiplexing possible for a broadcast network.

   (a) **Spatial** We can have several physical copies of the medium. For examples, we can add more wires between the senders and receivers.

   (b) **Frequency division** Different channels uses different frequency of the spectrum.

   (c) **Time** We can have either synchronous or asynchronous time multiplexing.

   - **Synchronous** The timing and length of packets are fixed for synchronous multiplexing. For example, the T1 channel divides the channel into repeating 24 byte blocks. Addressing is implicit in the order of the byte sequence. Each host has exactly the same bandwidth.
   - **Asynchronous** A broadcast LAN (*e.g.*, Ethernet) uses asynchronous multiplexing. The timing and length of packets are not fixed.

   (d) **Code division** (Spread spectrum) The channel is divided into many chunks. Each sender has a random subset of the chunks. Receivers use correlation to demultiplex.

3. **Availability** The network should be available even when some of the nodes fail. This is extremely important in a ring network.

A physical broadcast network may look like a graph of routers and hosts. To implement the broadcast specification, sometimes a message has to be sent to all routers. Usually a spanning tree is built to control broadcast and prevent cycles. First, a node is selected as the root of the spanning tree. its neighbors in the network become its children. This goes on recursively until all nodes belong to the spanning tree. The algorithm is run every time there is a change in the topology.

4 Ethernet

Ethernet is a widely used 10 Mbit/sec broadcast network. It was invented by researchers at Xerox PARC in the late '70's. Conceptually, it has a bus structure, and originally it used coaxial cable as its transmission medium, with nodes tapping directly onto the cable. Modern Ethernets use mostly a hub configuration.

This is the process of sending a packet in Ethernet:

1. Listen to the network.
2. If the network is busy, wait.
3. If it is idle, send the packet.
4. Monitor the network for collision. If two or more nodes register idle at the same time, multiple messages may be broadcast simultaneously. A node detects collision by sensing whether the signal on the bus is the one sent by itself or it has been garbled by other senders.
5. If there is a collision, backoff and retry.

Ethernet uses *exponential backoff* to overcome collisions. The longest possible time it takes to detect a collision is the round trip time between two farthest nodes on the network. In Ethernet, this is specified to be 50μs which allows time for 2km of cables and several repeaters.

To compute the efficiency of Ethernet, imagine dividing time into slots. Let k be the number of senders and p be the probability that a transmission will succeed. Then the average number of time slots, A, required for one successful transmission (by any station) is given by:

$$A = kp(1-p)^{k-1}$$

which is maximized when $k = 1/p$. As $k \to \infty$, $A \to 1/e \approx 0.37$. The probability that a sender needs j trials to send is $A(1-A)^{j-1}$. The expected number of trials, $E(\textit{trials})$, is given by:

$$E(\textit{trials}) = \sum_{0}^{\infty} jA(1-A)^{j-1} = \frac{1}{A}$$

As $k \to \infty$, $E(\textit{trials}) \to e$. Suppose the average packet takes $t\mu$s to transmit and the size of the slot is 50μs. The efficiency of the network is

$$\frac{t}{t + (e-1)50}$$

The smallest possible packet has to take one slot time to transmit. If all packets are minimally sized, the efficiency approaches $1/e$ in the worst scenario. In reality, most packets are much larger and the efficiency of Ethernet approaches 97%.

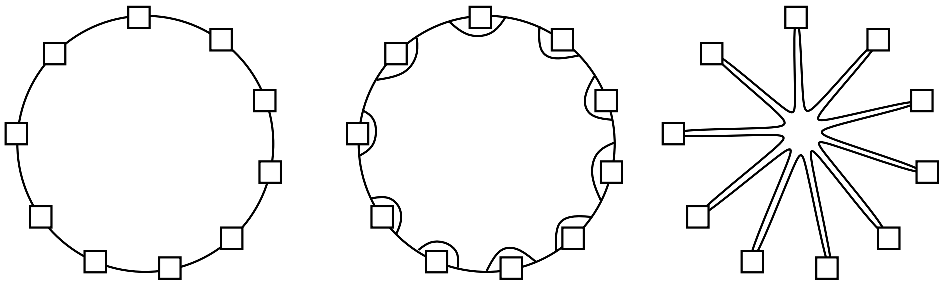

Figure 3: A ring, a ring with bypasses, and a ring in a star configuration. Note that the star configuration allows an arbitrary number of nodes to be cut out of the ring without noticably increasing the distance between consecutive nodes.

5 Token Ring

If we use Ethernet's mechanism in a 100 Mbits network such as FDDI, the minimum sized packet has to be ten times larger. Since there are many short packets for ack, request, *etc*., the effective utilization of the 100 Mbit network could drop by as much as a half. FDDI actually uses a ring structure similar to a token ring. Token rings have been proposed by IBM as an alternative to Ethernet. In a token ring, all nodes are connected via a ring interface. The ring is actually a collection of point-to-point links configured in a circle. A special bit pattern called the *token* is circulated around the ring when the system is idle. To send a message, a node must wait until the token reaches it. The node removes the token from the ring, sends its message and then appends the token. By keeping ahead of the token, a message is sent to every node in the ring.

There are a couple of requirements for a token ring to work:

- There must be exactly one token when the network is idle.
- All links must transfer packets at the same rate.

These requirements are nontrivial. How can we initialize and maintain the invariant that exactly one token exists? Also, note that if a link fails, the entire network would halt. This availability problem might be solved by bypass relays, but if too many relays are used in a row, then the distance between consecutive nodes can increase to the point where attenuation reduces the quality of the signals. Another possible solution is to physically configure the ring like a star and let a central node perform the bypassing (see Figure 3). Yet another solution is Dual Attachment Ring where two physical rings are used.