Lecture 17: Medium Access Control V

- Perfectly scheduled approaches

- Token ring networks

We have categorized channel allocation techniques as either static or dynamic, and we further sub-divided dynamic approaches as either contention-based or perfectly scheduled approaches. So far, we have mainly discussed contention-based dynamic approaches such as Aloha and CSMA/CD. The Ethernet protocol and wireless LANs were given as examples of where contention-based approaches are used. In all contention-based protocols, the possibility exists for two or more nodes to transmit at the same time resulting in a collision. The protocols must then provide a means for arbitrating between colliding nodes, so eventually each can successfully transmit. Collisions limit the throughput of these protocols at high offered loads.

In this lecture, we take a closer look at the second type of dynamic channel allocation approaches - *perfectly scheduled approaches* (In Tannenbaum these are also referred to as *collision-free protocols*). These protocols attempt to dynamically schedule transmissions while ensuring that collisions do **not** occur. First, we look at some general examples of this type of approach. We then look at **token ring** LANs as specific example of a perfectly scheduled approach.

Basic Reservation system:

The basic idea behind scheduled channel allocation is to allow nodes to somehow **reserve** the channel when they have data to send. Note in several previous approaches, such as CSMA/CD, nodes effectively reserve the channel, however they do this via a contention-based method. Now we look at techniques that avoid even this contention.

Consider a broadcast network with N stations. A straightforward approach for scheduling transmissions is to periodically have a reservation period with N reservation time-slots. Assume each station is assigned a unique number between 1 and N. During the ith reservation-slot the ith station sends a signal to indicate that it has a packet to send. The reservation period is followed by a sequence of data slots, one for each station that made a reservation. This cycle then repeats. This is shown below for N=4 stations. Here an r indicates that a reservation is made.

	r		r		2		4		r					1	

reservations   data frames   reservations   data frames

3
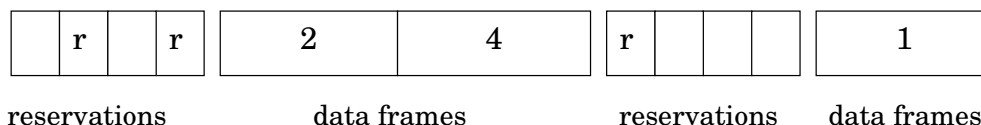
The performance of the above algorithm will depend on the time it takes for nodes to make reservations. At low loads, this will determine the delay experienced and at high loads this will limit the throughput. The time depends on the amount of data needed to be sent to make a reservation and any additional delay needed to make sure that all nodes know about the reservations. If all nodes are initially synchronized and all data packets are the same length, then only one bit is needed to make a reservation. Since all nodes can receive every transmission, each node will know which nodes are transmitting and can stay synchronized. In practice some additional overhead is required, for example to ensure synchronization and to aid in recovery when errors occur.

Once a nodes makes a reservation, the time until all nodes know about it again depends on β, the normalized maximum propagation delay in the network (see Lectures 21). In a hubbed network with a star topology, it takes β packet-times after each node transmits until all other nodes are aware of this, thus the total reservation interval takes $N(\beta +r/d)$ packet times, where d is the number of bits in each data packet and each reservation consists of r bits.

4

Next we look at an example for a shared bus.

Assume nodes are connected to a bus and that nodes make reservations in the order they are connected as shown below.
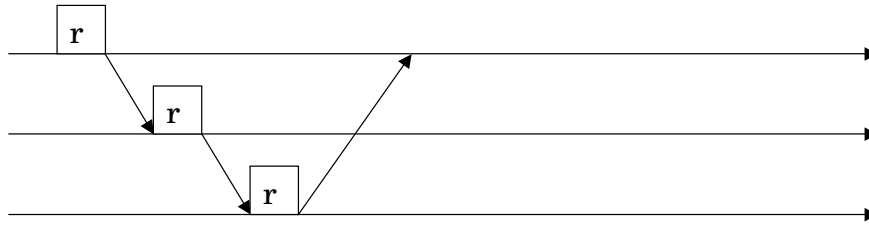
Again assuming reservations contain r bits then the reservation interval takes $(Nr)/d + 2\beta$ packet-times.

At low loads, the above protocol can be unfair: Assume that no stations are active and that a packet arrives for a node to send. It is reasonable to assume that it is equally likely that the system will be at any place in the current reservation sequence. In this case, low number stations have to wait on average $\sim (1.5)(Nr/d+2\beta)$ packet times transmit, while high number nodes only have to wait an average of $\sim (0.5)(Nr/d+2\beta)$. (This can be somewhat remedied by varying the transmission order according to some rule.) Thus at low loads, the average wait for all stations is $Nr/d + 2\beta$ packet times.

If all stations are transmitting, then it takes $N + 2\beta$ packet times for the stations to all transmit (after making reservations) and for the last frame to be heard by the first station. Thus at high loads, the channel efficiency is approximately:

$$\frac{N}{Nr/d + 4\beta + N}$$

(Note these quantities where calculated assuming a shared bus as on the previous page.)

Comments:

- If $(4\beta/N + r/d) \ll 1$ and the system is heavily loaded then the efficiency will be near 1. (This is much better than in a corresponding contention-based approach.)

- In lightly loaded case, stations incur extra delay (compared to contention-based approaches) depending on length of reservation interval. For small N and small β this may not be a big concern.

- In satellite networks, β can be larger than time to transmit data frames. One technique to improve efficiency at high loads is to make current reservation period apply to the data frames sent after the following reservation period. However this further increases the delay at low loads.
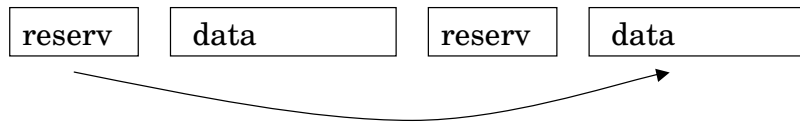

- We have assumed fixed size packets. We could allow for variable sized packets but at a cost of more overhead and more difficult timing.

- An issue we have not addressed is how stations can become resynchronized after a fault occurs. One approach is to use special framing characters to indicate the reservation intervals; this increases the overhead required for reservations. Another approaches is to use a centralized controller to regulate the reservation intervals, as in a polling system.

- The above model assumes a fixed number of stations. If the number of stations varies, then an additional protocol is needed to allow users to join the reservation interval.

As opposed to contention-based approaches, reservation approaches can offer a guaranteed worse case delay (if fixed size packets are used).

Token ring networks

Suppose that instead of having a reservation interval followed by an interval for data packets, each node is allowed to transmit directly after its reservation interval (if it has a packet to send). This is illustrated below.

| R	1	R	2	R	R
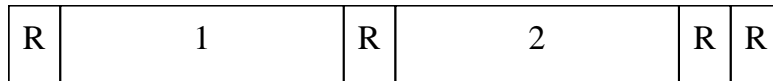
In this case we think of each reservation interval as being synonymous to a node having a token that gives it permission to transmit. This token is then passed from node-to-node.

There are several protocols that are based on variations of the above idea. These networks are referred to as *token ring networks*. These protocols include:

1. *IEEE 802.5 token ring LAN* - this standard is based on a protocol developed by IBM in the 1970's (aptly called the IBM token ring).
2. *IEEE 802.4 token bus LAN* - based on a protocol developed by GM for manufacturing and industrial applications.
3. *Fiber Distributed Data Interface (FDDI)* this is a high-speed LAN/MAN technology standardized by ANSI in the mid-1980's.

9

IEEE 802.5 Token Ring

As a representative of token ring networks, we describe the operation of the IEEE 802.5 token ring LAN in the following. As suggested by the name, these networks are based on a ring topology. Data flows in one direction around the ring. The standard defines transmission rates of 4 Mbps and 16Mbps. The physical medium used to connect nodes is a twisted pair cable and up to 250 nodes can be supported. Physically the network is often wired in a star topology with a wiring center that transmits the incoming signals so that logically the nodes are connected in a ring.

10
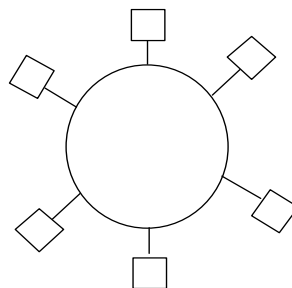
Ring Interface

Each node is connected to the ring via a *ring interface*. Each interface consists of two ports, one on which data is received and one on which data is transmitted. When a station is idle (not transmitting new data) it simply repeats on the output port what it receives on the input port after approximately a 1 bit delay, during which time each received bit is stored into memory. While a station is transmitting, the received bits are discarded.

Listen   Transmit

1 bit delay

To station   From Station   To station   From Station
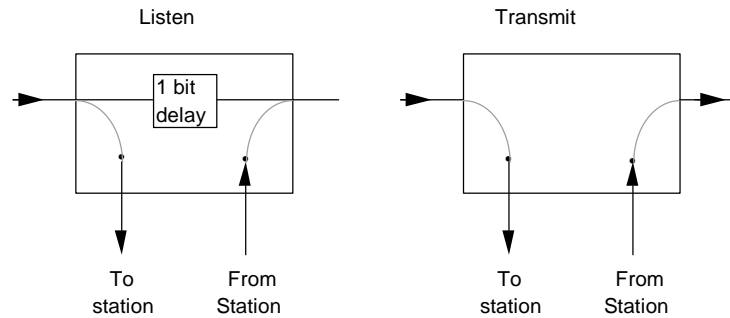
Medium Access

To access the channel in a token ring the station must first receive a specific bit pattern called the token. If a station has no data to send it simply forwards what ever it receives and checks for any data packets that are addressed to it. If the station has data to send, then once the station receives the token, it removes the token from the ring and proceeds to send the data. In 802.5 the token is removed by flipping a bit that changes the token into part of the header for a data frame. The transmitted data frame will propagate all the way around the ring and is removed by the originating station. Once it is finished transmitting, the station must release a new token, in the original 802.5 standard this is done only after the last part of the message has been transmitted and the leading edge of the transmitted frame has returned around the ring. (We will see several variations of this in the following.)

Performance

A basic performance parameter in a token ring is the delay around the ring, this includes both the propagation delay and the delay added by each ring interface. Suppose that this time is T seconds, we will again normalize this by the time to send a message. We denote this normalized time by a, thus
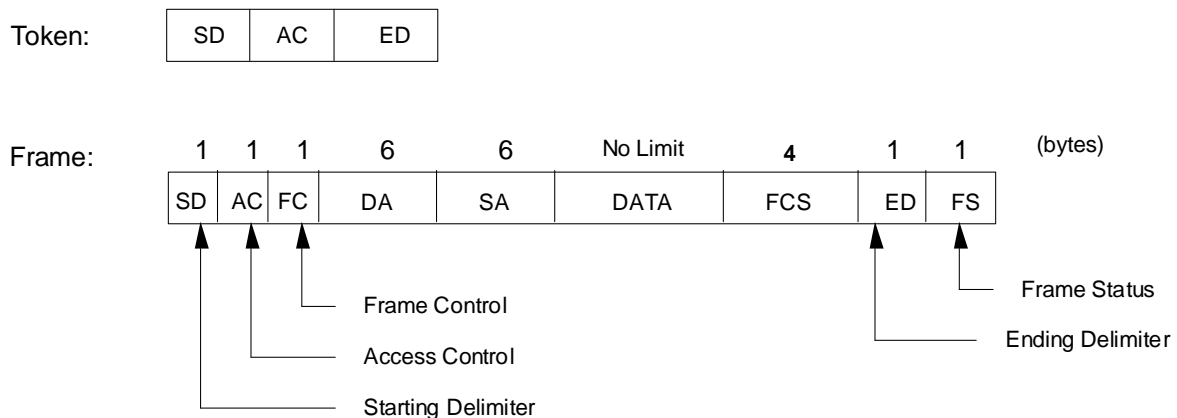
$$a = \frac{T}{F/C}$$

where F is the number of bits/message and C is the transmission rate. Two different cases can be identified:

- If $a > 1$, then each message does not fill the ring and the maximum throughput $\cong \frac{1}{a}$.
- If $a < 1$, then each message fills the ring and the only idle time is during token passing. Thus the throughput ≈ 1 (100% efficient at high loads)

Thus as the data rate increases or as the ring gets physically longer or stations are added, a gets larger, resulting in less efficiency.

13

802.5 Frame Format

Token:

SD	AC	ED

Frame:

1	1	1	6	6	No Limit	4	1	1	(bytes)
SD	AC	FC	DA	SA	DATA	FCS	ED	FS	

Frame Control
Access Control
Starting Delimiter

Frame Status
Ending Delimiter

Note: while there is no explicit limit on the data field size, there is a maximum token holding time for any node, which implicitly limits the data field size.

14

802.5 Frame Format

Many of the fields in the 802.5 frame are the same as those we have seen in other protocols. For example, the starting and ending delimiters are illegal character codes used for framing. The DA and SA fields are the source and destination addresses, and the FCS field contains a 32 bit CRC.

Two new fields of interest are the access control (AC) and frame status (FS) fields. The access control field has the format shown below. The third bit in this field is set to 0 to indicate a token and to 1 to indicate a data frame.

AC - Access Control

PPP	T	M	RRR

Reservation bits

Monitor bit

Token bit: O = token 1 = frame
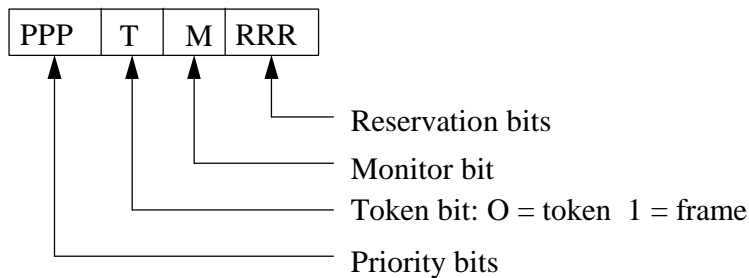
Priority bits

The PPP and RRR bits in the access control field are used for implementing a priority scheme. Under this scheme packets and tokens are assigned different levels of priority, a node can only transmit a packet with a given priority using a token with the same level or lower priority. The priority is indicated in the priority bits (PPP) of the access control field. Stations can try and reserve the token by setting the reservation bits (RRR) in the access control field. This allows stations to raise the priority of the next token generated. After transmitting a frame a station must lower the priority to the previous value.

The frame status byte (FS) is used to provide an acknowledgement to the transmitter. Several bits in this byte are set to zero by the transmitter, when the intended receiver successfully copies a frame into its buffer, these bits are changed to ones. If a packet returns around the ring with these bits still set to zero then the transmitter will know that the packet was not received.

Ring Maintenance

Perhaps the biggest shortcoming of token ring networks is that elaborate procedures are needed for ring maintenance, i.e. to ensure that the protocol is operating correctly. For example procedures are needed to detect and remove duplicate tokens, to replace lost tokens, and to detect breaks of the links in the ring. To accomplish the 802.5 standard makes one station on the ring the *active monitor*; this station is responsible for ring maintenance. Procedures are also provided to detect when the monitor fails and to choose another node to replace it.

17