College of Computer Science
Northeastern University
COM 3510

Lecture 7
8 November 2001
Computer Networks: Theory, Modeling, and Analysis

Multiple-Access communication

Handouts: Sample solution for midterm problems; Problem Set 3 out.

Lecture Outline:

- Introduction.
- Random-access protocols.
- The Ethernet protocol.

1 Introduction to multiple-access communication

In many data network environments, the transmission links are broadcast media which are shared by multiple nodes. Such environments, referred to as multiple-access environments, require protocols for resolving contention for the resource and/or scheduling the resource among the multiple users. These protocols form the *medium access sublayer* of the data link layer. Examples of multiple-access environments include satellite networks, cellular networks, Ethernet and other local-area networks, packet radio networks, and mobile ad-hoc networks.

- **Cellular networks** divide a geographical region into disjoint cells. Each cell has a base station that acts as a control tower for communicating with mobile stations in the cell. The communication channel is divided into two subchannels, one for *uplink* communication and the other for *downlink* communication. There is no contention in the downlink channel, since only the base station transmits on this channel. Access to the uplink channel needs to be controlled, however, since all mobile stations can transmit on this channel.
- **Satellite networks** allow communication among ground stations via oribiting satellites. Thus, a satellite acts as a control tower managing communication among the different ground stations. Satellite communication can be modeled in the same way as communication within a single cell of a cellular network. Of course, the associated paramters – the frequency bands, latencies, power of nodes – are very different and hence different mutiple-access protocols are needed.
- Owing to the low cost and simplicity of broadcast networks (no addressing, switching, or routing), **local-area networks** are based on the broadcast approach. *Ethernet*, a widely deployed LAN protocol, is a like a shared bus accessible to multiple users simultaneously. Another example of a multiple-access local-area network is the *token ring*.
- **Packet radio networks** and **mobile ad-hoc networks** are multihop networks that employ a limited form of broadcast. In these networks, a node can directly communicate with geographically nearby nodes by broadcasting on a common frequency band. While there is little inteference between the transmissions of two nodes that are distant, nodes in the same

"neighborhood" need to manage the access to the communication medium using an appropriate multiple-access protocol. Communication among geographically distant nodes is done by sending the message along multiple hops, transmission along a hop being implemented by a local broadcast. In traditional packet radio networks, there is no fixed infrastructure like the base station in a cellular network; so it is quite a challenge to provide reliable communication in this distributed mobile environment.

In a queueing-theoretic sense, a multiple-access environment has a single server, namely, the broadcast medium, and multiple queues. The packets in these queues contend for the shared server. Ideally, we would like to view the multiple queues as one queue, but this is difficult to do since this requires coordination among nodes that are physically distributed. One natural mechanism to share the medium among multiple nodes is to divide it equally, either across the time dimension or across the channel capacity dimension. This leads to the time-division multiple-access (TDMA) and frequency-division multiple-access (FDMA) schemes. While both TDMA and CDMA are commonly used in certain multiple-access environments (e.g., cellular networks), they are not useful in general data networks for the same reason as time- and frequency-division multiplexing are not. In this lecture, we will study protocols for controlling access to multiple-access channels. Such protocols are referred to as *multiple-access control* protocols or MAC protocols.

MAC protocols can be classified into two categories: *random-access*, in which the multiple stations sharing the common medium attempt to transmit on the channel randomly and independently, and *scheduling* protocols, in which the individual stations are provided access slots according to a schedule determined by a central controller. Ethernet, packet radio, and mobile ad-hoc networks employ random-access MAC protocols, while cellular networks and satellite networks employ schedule-based MAC protocols. Token rings may be viewed as a hybrid of random-access and schedule-based protocols.

2 Random access protocols

Random access protocols involve distributed control; each station decides independently whether to transmit on the channel. We begin with the classic *Aloha* protocol, which was designed in the 1970s for radio communication among the various campuses of the University of Hawaii system. We next discuss a backoff strategy commonly used to resolve contention. Then, we discuss the *CSMA* (Carrier-sense multiple access), and *CSMA-CD* (Carrier-sense multiple access with collision detection) technologies which provide a better utilization of the channel by avoiding collisions under certain conditions.

2.1 The Aloha protocol

The Aloha protocol was designed for a scenario in which multiple stations are communicating with a control tower using the same transmission medium. These stations compete for the channel and use a simple random-access mechanism to decide when to transmit. For simplicity, we consider an idealized model for the system, which we refer to as the *slotted model*. This model consists of the following assumptions:

1. Time is divided into slots. Every transmission takes unit time, and all the nodes are synchronized across slot boundaries.

2. If exactly one packet is transmitted in any slot, then it is received without error by the receiver. If more than one packet is transmitted in any slot, then there is a collision and none of the transmitted packets is successfully received.

3. Every node gets feedback as to whether 0, 1, or greater than 1 packets were transmitted in a slot.

4. When a packet collides, it needs to be retransmitted.

5. Buffering: Typically, two cases are considered. One in which there is no buffering, which means that any new arrival at a node is discarded when there is a packet waiting at the node. The second is infinite buffering, which is usually modeled by setting m to infinity. The first model provides an upper bound on the throughput of a given protocol while the second model provides a lower bound.

Slotted Aloha. In the slotted Aloha protocol, when a new packet arrives, a transmission is attempted immediately in order to avoid any delay. If a collision occurs, then the packet is considered *backlogged*. Each node with a backlogged packet transmits the packet with probability p in each slot until the packet is successfully transmitted.

Suppose n packets are backlogged in a system with no buffering. That is, these packets reside in n different nodes. Furthermore, assume that no new packet arrives into the system at this time. Then, there is a successful transmission if exactly one of the backlogged packets is transmitted. The probability of this event is

$$\binom{n}{1} p(1-p)^{n-1} = np(1-p)^{n-1}.$$

If p is chosen to be $1/2$, we find that the success probability equals $n/2^n$, which is extremely small for large n. In fact, if p is chosen to be a constant (independent of n), it can be easily seen that the probability of success will grow as an inverse exponential of n, thus implying that for large n the protocol would have a low success probability that would ultimately lead to instability. One can see that if the arrival rate is larger than this success probability, then once a large number of packets are backlogged, the number of packets in the system will tend to grow without bound.

Let us calculate the value of p that maximizes the success probability. By differentiation, we get that this value of p is such that

$$n(1-p)^{n-1} - n(n-1)p(1-p)^{n-2} = 0$$

thus yielding $p = 1/n$. For this value of p, the success probability equals $(1-1/n)^{n-1}$ which is at least $1/e$. This suggests that we can maximize success if we vary the transmission probability dynamically. In order to do so, however, we need to know the number of nodes in the system with backlogged packets. The stabilized Aloha protocol maintains an estimate of n and uses this estimate to set the probability of transmission.

Stabilized slotted Aloha. In the stabilized Aloha protocol, we set p to $\min\{1, 1/\widehat{n}\}$, where $\widehat{n}$ is an estimate of the number of backlogged packets that is maintained by each node in the system. Furthermore, every packet is assumed to be backlogged upon arrival. Thus, the stabilized protocol does not transmit a packet immediately upon arrival, unlike the standard Aloha protocol. This leads to higher delay for some packets, but ultimately results in a stable protocol.

The estimate $\widehat{n}$ is updated dynamically as the nodes receive feedback after each slot. In each slot, the expected number of new arrivals is λ. If the node receives feedback of a success, then the estimate needs to be decreased by $1 - \lambda$. If there is a collision, then the estimate needs to be increased by λ. These changes to the estimate are appropriate if we are able to accurately estimate the number in each step. However, the estimates may drift away from a correct value and this could lead, for example, to a sequence of collisions when a large number of nodes with backlogged packets estimate the number of backlogged packets to be small. Similarly, if there are a small number of backlogged packets and the nodes estimate the backlog number to be large, then there may be more idle slots than necessary. It turns out that the following modification rule leads to a stable protocol. let $\widehat{n}_k$ denote the estimate of the number of backlogged packets in the system at the start of slot k. Then, after slot k, each node updates the estimate as follows.

$$\widehat{n}_{k+1} = \begin{cases} \max\{\lambda, \widehat{n}_k + \lambda - 1\} & \text{idle or success,} \\ \widehat{n}_k + \lambda + \frac{1}{e-2} & \text{collision} \end{cases}$$

Note that the ratio of the correction factors $1/(e-2)$ (in case of collision) and 1 (in the case of idle or success) is the same as the ratio of their respective probabilities. (When the estimate is accurate, the probability of a successful or idle slot is approximately $1/e + 1/e = 2/e$, while the probability of collision is approximately $(e-2)/e$.)

2.2 The exponential backoff protocol

One drawback of the stabilized Aloha protocol is that every node requires 3-way feedback; that is, every node should be able to differentiate between idle, successful, and collision slots. In some situations, only nodes that are transmitting receive feedback about the success or failure of their transmission. A protocol that is often used in these scenarios is *the exponential backoff protocol*. In a backoff protocol, the probability of transmission of a packet in a slot is a function of how many times the packet has collided. The more number of collisions the packet has had, smaller is the probability of transmission. An alternative, more practical implementation is that after any collision, the transmitting node waits for a random period of time, which is a function of the number of collisions incurred by the packet, and then retransmits the packet.

In the exponential backoff protocol, if the packet at the head of the queue has had i collisions, then the node waits for a random number x of slots chosen uniformly from $[1, 2^i]$ and attempts to transmit the packet after x slots. It has been shown that in the infinite user case, the exponential backoff protocol is unstable for any $\lambda > 0$. For the finite user case, the stability is still open. We will have a closer look at exponential backoff when we study the Ethernet protocol (see Section 3).

2.3 CSMA

In the slotted model, we assumed that time is divided into slots and all nodes are synchronized across slot boundaries. This is not true in practice. It is very difficult to synchronize the nodes exactly. Consequently, collisions can occur when packets are transmitted outside slot boundaries. For example, suppose it takes 1 unit time to transmit a packet and suppose a station transmits a packet x at time 1. A transmission by a different station any time during the interval $[1, 2)$ will collide with x. Similarly, any packet whose transmission begins in the period $(0, 1]$ will also collide with packet x. Thus, the *vulnerable period*, the interval during which no packet other than x should begin transmission is 2 units long. If the packets are long (and hence, the above time unit is long), the vulnerability period is long, resulting in inefficient use of the multiple-access channel.

The vulnerable period can be shortened (and, hence the efficiency increased) if whenever a node is transmitting, other nodes can listen to it and not start transmitting to avoid collision. Protocols that make use of this "sensing" are referred to as Carrier Sense Multiple-Access (CSMA) protocols. Now the sensing is not instantaneous; it depends on the bandwidth of the medium, the distance between the transmitting node and the sensing node, the speed of light in the medium. These characteristics of the system are captured by a parameter, simply referred to as a in the networking literature. We define a as follows.

$$a = \frac{t_{prop}}{T},$$

where t_{prop} is the maximum propagation delay between two nodes in the system, and T is the time taken to transmit a packet of average packet size. The parameter a is small for wired LANs, cellular and packet radio (< 0.01), larger (about 1) for faster LANs, and large for satellites (could be > 100). Note that as the bandwidth increases, a increases. And as the physical distance between nodes increases, so does a.

The vulnerable period in CSMA schemes decreases to $2t_{prop}$. As far as transmission strategies are concerned, the standard CSMA protocols adopt the same strategy as the Aloha protocol when the channel is idle; that is, if a packet arrives at a station and the station senses that the channel is idle, the station immediately transmits the packet. The protocols differ according to what the stations do when they have a packet to transmit and they find that the channel is busy. In the *1-persistent CSMA* protocol, the sender waits until the channel is idle and immediately sends the packet. This strategy works well under very light loads, but results in sure collisions when more than one sender receives packets during a busy period, since they attempt to transmit immediately after the current busy period ends. Collisions are then resolved by using a backoff algorithm similar to the one described in Section 2.2.

In order to avoid collisions immediately after a busy period, the *non-persistent CSMA* employs a backoff protocol immediately; that is, when a packet arrives at a station that finds the channel busy, this is treated as a collision and the station immediately runs the backoff algorithm and reschedules the transmission time to later. A hybrid protocol is the *p-persistent CSMA* protocol in which the sender persist sensing the channel until the channel is idle and then transmit with probability p. If the station does not transmit, then it waits for t_{prop}, resenses the channel, and repeats. If the station transmits and a collision occurs, then they employ a backoff protocol.

2.4 CSMA-CD

In CSMA protocols, while the vulnerable period is reduced to t_{prop}, the time wasted due to a collision equals the packet transmission time, which could be large when packets are large. If a station can determine that a collision has taken place soon after the occurrence, then this wastage can be reduced. The carrier sense mutiple-access with collision detection protocols use this approach.

Consider a 1-persistent CSMA protocol with collision detection. Assume that the time is divided into alternating periods of contention and packet transmission. For convenience, we divide the contention period into "minislots". In order to ensure that stations can always detect a collision within a minislot, we assume that each minislot is of length $2t_{prop}$.

We now perform a simplified analysis of the throughput of the 1-persistent CSMA-CD protocol. Consider the contention period right after a busy period. Suppose that n nodes have a packet each and each is attempting to transmit with probability p. The success probability then equals

$$np(1-p)^{n-1}.$$

Standard calculus yields that the success probability is maximized when $p = 1/n$, and the resultant value is $1/e$. Using algebraic calculations as we have done before, we ca show that the expected number of contention periods it takes for a successful transmission is e. Since each contention period is of size $2t_{prop}$, we obtain that the fraction of time that the channel is used successfully is given by

$$\frac{T}{T + t_{prop} + 2et_{prop}},$$

where T is the transmission time for an average packet, the first t_{prop} time units is the period just after the busy period, during which stations find out that the packet transmission is successfully completed, and $2et_{prop}$ is the expected time spent in the contention period. Plugging in the parameter a for t_{prop}/T, we get the throughput of a 1-persistent CSMA-CD protocol is at most

$$\frac{1}{1 + (2e + 1)a}.$$

3 The Ethernet LAN protocol

A classic example of CSMA-CD protocols is the Ethernet LAN protocol which uses a 1-persistent CSMA protocol with collision detection and exponential backoff. In an Ethernet LAN, when a new packet is at the head of the queue of a node, the node waits for the channel to be free (it is listening in) and immediately attempts to transmit (with probability 1). If there is a collision, then the node recognizes this quickly (in time much smaller than the typical length of the packet). It then employs an exponential backoff protocol to determine how much time it should wait until it will attempt its next transmission.

We now discuss some details of the Ethernet protocol. As discussed in Section 2.4, time may be divided into alternating periods of contention and packet transmission. A contention minislot should be long enough that when a collision occurs, all the nodes hear it. To enable this, when a transmitting node senses a collision, it jams the channel for a small period of time so that every other node is able to listen to the collision. After that, there is a small waiting time, before the next contention minislot begins. We now ask the question: how big should a slot be? In other words, what is the minimum packet-size allowed? This question is related to the parameter a of the medium.

Let D denote the maximum propagation delay between any two nodes in the system. Consider 2 nodes A and B at distance D. Let A begin a transmission at time 0. The node B receives the first bit of this transmission at time D. It is possible, however, that B initiates its transmission just before time D. If the length of A's transmission is less than $2D$, then A will not sense B's transmission before completing its own transmission; this may cause A to wrongly believe that it had a successful transmission. We thus see that the length of a slot, in time, should be at least $2D$. Expressed in bits, the length of a slot should be at least $2D$ times the bandwidth of the medium. To this, we also need to add the bits corresponding to the jam time.

For a 10-BaseT Ethernet cable, the calculation of a slot leads to a packet size of 512 bits, which equals 64 bytes. This implies that the minimum packet-size is 64 bytes.

Figure 1: The Ethernet capture effect: A sends its packets one after another and incurs small backoffs, while the backoff counter for B grows. X denotes a collision and I an idle slot.

In our analysis of the 1-persistent CSMA-CD protocol, we placed an upper bound on the throughput by calculating the maximum success probability in a contention minislot. Since the stations are contending for the channel in a distributed manner, they cannot always set their transmission probabilities to the optimally chosen one. In the Ethernet protocol, the transmission probabilities (in a minislot) are determined by an exponential backoff protocol. The analysis of backoff protocols is quite complex. Here, we limit ourselves to an interesting phenomenon that may occur due to the exponential backoff protocol. This phenomenon is called the *Ethernet capture effect*. Suppose two nodes A and B have a large queue of packets waiting to be transmitted. Say they both transmit at slot 0. A collision occurs and both of them backoff. Each selects a random slot within the next 2 slots to attempts again. Suppose one of them (say A) selects slot 2 while the other (say B) selects slot 3 (the probability of this event is $1/2$). Then, A will succeed and B will hold off on its transmission until the packet sent by A is completely transmitted. Soon after the completion of that transmission, B will attempt its transmission. Since A has long queue of packets, however, A will also attempt to transmit a packet. So there will be a collision. A will now backoff for a number of slots chosen uniformly at random from $[1, 2]$ since this packet has only incurred one collision. On the other hand, B will backoff for a period of time chosen from $[1, 4]$ since its packet has incurred two collisions. Thus, it is more likely that A will pick a slot earlier than B and succeed, thus delaying B further. This example is illustrated in Figure 1.

Continuing the above line of argument, we see that there is a possibility that A "captures" the Ethernet while B is delayed. This delay at B could cause a huge backlog at B and result in several packets getting delayed. In the Ethernet protocol, this capture phenomena is addressed by discarding any packet that has suffered 16 collisions. This means that in the above example, after 16 failures, B will discard the packet at the head of its queue and move to the next packet. Now, B will start will a fresh backoff counter of 0 and thus resumes from a state in which it is at least as likely to succeed as A.