

THE MEDIUM ACCESS CONTROL SUBLAYER

- In any broadcast network, the key issue is how to determine who gets to use the channel when there is competition for it. To make this point, consider a conference call in which six people, on six different telephones, are all connected so that each one can hear and talk to all the others.
- When only a single channel is available, it is much harder to determine who should go next. Many protocols for solving the problem are known. They form the contents of this chapter. In the literature, broadcast channels are sometimes referred to as **multiaccess channels** or **random access channels**.
- The protocols used to determine who goes next on a multiaccess channel belong to a sublayer of the data link layer called the **MAC (Medium Access Control)** sublayer. The MAC sublayer is especially important in LANs, particularly wireless ones because wireless is naturally a broadcast channel.
- WANs, in contrast, use point-to-point links, except for satellite networks. Because multiaccess channels and LANs are so closely related, in this chapter we will discuss LANs in general, including a few issues that are not strictly part of the MAC sublayer, but the main subject here will be control of the channel.
- Technically, the MAC sublayer is the bottom part of the data link layer, so logically we should have studied it before examining all the point-to-point protocols in Chap. 3. Nevertheless, for most people, it is easier to understand protocols involving multiple parties after two-party protocols are well understood.

❖ THE CHANNEL ALLOCATION PROBLEM

- The central theme of this chapter is how to allocate a single broadcast channel among competing users. The channel might be a portion of the wireless spectrum in a geographic region, or a single wire or optical fiber to which multiple nodes are connected.
- In both cases, the channel connects each user to all other users and any user who makes full use of the channel interferes with other users who also wish to use the channel.
- We will first look at the shortcomings of static allocation schemes for bursty traffic. Then, we will lay out the key assumptions used to model the dynamic schemes that we examine in the following sections.

❖ Static Channel Allocation

- The traditional way of allocating a single channel, such as a telephone trunk, among multiple competing users is to chop up its capacity by using one of the multiplexing schemes we described in Sec. 2.5, such as FDM (Frequency Division Multiplexing).
- When there is only a small and constant number of users, each of which has a steady stream or a heavy load of traffic, this division is a simple and efficient allocation mechanism. A wireless example is FM radio stations. Each station gets a portion of the FM band and uses it most of the time to broadcast its signal.
- when the number of senders is large and varying or the traffic is bursty, FDM presents some problems. If the spectrum is cut up into N regions and fewer than N users are currently interested in communicating, a large piece of valuable spectrum will be wasted.
- The poor performance of static FDM can easily be seen with a simple queueing theory calculation. Let us start by finding the mean time delay, T , to send a frame onto a channel of capacity C bps. We assume that the frames arrive randomly with an average arrival rate of λ frames/sec, and that the frames vary in length with an average length of $1/\mu$ bits. With these parameters, the service rate of the channel is μ/C frames/sec.

A standard queueing theory result is: $T = 1/(C - \lambda)$

- The mean delay for the divided channel is N times worse than if all the frames were somehow magically arranged orderly in a big central queue. Since none of the traditional static channel allocation methods work well at all with bursty traffic, we will now explore dynamic methods.

❖ Assumptions for Dynamic Channel Allocation

Before we get to the first of the many channel allocation methods in this chapter, it is worthwhile to carefully formulate the allocation problem. Underlying all the work done in this area are the following five key assumptions:

0. **Independent Traffic.** The model consists of N independent **stations** (e.g., computers, telephones), each with a program or user that generates frames for transmission. The expected number of frames generated in an interval of length t is λt , where λ is a constant (the arrival rate of new frames). Once a frame has been generated, the station is blocked and does nothing until the frame has been successfully transmitted.
1. **Single Channel.** A single channel is available for all communication. All stations can transmit on it and all can receive from it. The stations are assumed to be equally capable, though protocols may assign them different roles (e.g., priorities).
2. **Observable Collisions.** If two frames are transmitted simultaneously, they overlap in time and the resulting signal is garbled. This event is called a **collision**. All stations can detect that a collision has occurred. A collided frame must be transmitted again later. No errors other than those generated by collisions occur.
3. **Continuous or Slotted Time.** Time may be assumed continuous, in which case frame transmission can begin at any instant. Alternatively, time may be slotted or divided into discrete intervals (called slots). Frame transmissions must then begin at the start of a slot. A slot may contain 0, 1, or more frames, corresponding to an idle slot, a successful transmission, or a collision, respectively.
4. **Carrier Sense or No Carrier Sense.** With the carrier sense assumption, stations can tell if the channel is in use before trying to use it. No station will attempt to use the channel while it is sensed as busy. If there is no carrier sense, stations cannot sense the channel before trying to use it. They just go ahead and transmit. Only later can they determine whether the transmission was successful.
 - The first one says that frame arrivals are independent, both across stations and at a particular station, and that frames are generated unpredictably but at a constant rate.
 - **Poisson models**, as they are frequently called, are useful because they are mathematically tractable. They help us analyze protocols to understand roughly how performance changes over an operating range and how it compares with other designs.
 - The single-channel assumption is the heart of the model. No external ways to communicate exist. Stations cannot raise their hands to request that the teacher call on them, so we will have to come up with better solutions.
 - The collision assumption is basic. Stations need some way to detect collisions if they are to retransmit frames rather than let them be lost. For wired channels, node hardware can be designed to detect collisions when they occur. The stations can then terminate their transmissions prematurely to avoid wasting capacity.

- The reason for the two alternative assumptions about time is that slotted time can be used to improve performance. However, it requires the stations to follow a master clock or synchronize their actions with each other to divide time into discrete intervals. Hence, it is not always available.
- similarly, a network may have carrier sensing or not have it. Wired networks will generally have carrier sense. Wireless networks cannot always use it effectively because not every station may be within radio range of every other station. Similarly, carrier sense will not be available in other settings in which a station cannot communicate directly with other stations, for example a cable modem in which stations must communicate via the cable headend.
- To avoid any misunderstanding, it is worth noting that no multiaccess protocol guarantees reliable delivery. Even in the absence of collisions, the receiver may have copied some of the frame incorrectly for various reasons. Other parts of the link layer or higher layers provide reliability.

❖ MULTIPLE ACCESS PROTOCOLS

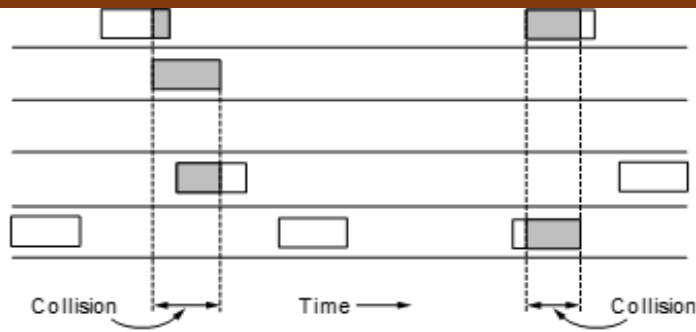
- Any algorithms for allocating a multiple access channel are known. In the following sections, we will study a small sample of the more interesting ones and give some examples of how they are commonly used in practice.

□ ALOHA

- The story of our first MAC starts out in pristine Hawaii in the early 1970s. In this case, “pristine” can be interpreted as “not having a working telephone system.” This did not make life more pleasant for researcher Norman Abramson and his colleagues at the University of Hawaii who were trying to connect users on remote islands to the main computer in Honolulu.
- The one they found used short-range radios, with each user terminal sharing the same upstream frequency to send frames to the central computer. It included a simple and elegant method to solve the channel allocation problem.
- Their work has been extended by many researchers since then (Schwartz and Abramson, 2009). Although Abramson’s work, called the ALOHA system, used ground-based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users are competing for the use of a single shared channel.
- We will discuss two versions of ALOHA here: pure and slotted. They differ with respect to whether time is continuous, as in the pure version, or divided into discrete slots into which all frames must fit.

□ Pure ALOHA

- The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent. There will be collisions, of course, and the colliding frames will be damaged. Senders need some way to find out if this is the case. In the ALOHA system, after each station has sent its frame to the central computer, this computer rebroadcasts the frame to all of the stations
- If the frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over, in lockstep. Systems in which multiple users share a common channel in a way that can lead to conflicts are known as **contention** systems.



- A user is always in one of two states: typing or waiting. Initially, all users are in the typing state. When a line is finished, the user stops typing, waiting for a response. The station then transmits a frame containing the line over the shared channel to the central computer and checks the channel to see if it was successful.
- If $N > 1$, the user community is generating frames at a higher rate than the channel can handle, and nearly every frame will suffer a collision. For reasonable throughput, we would expect $0 < N < 1$.
- Let us further assume that the old and new frames combined are well modeled by a Poisson distribution, with mean of G frames per frame time. Clearly, $G \leq N$. At low load (i.e., $N \rightarrow 0$), there will be few collisions, hence few retransmissions, so $G \approx N$. At high load, there will be many collisions, so $G > N$.

the end of that frame will

- If any other user has generated a frame between time t_0 and $t_0 + t_f$, it will collide with the beginning of the shaded one.

□ t_f

□ Slotted ALOHA

- Soon after ALOHA came onto the scene, Roberts (1972) published a method for doubling the capacity of an ALOHA system. His proposal was to divide time into discrete intervals called **slots**, each interval corresponding to one frame.
- In Roberts' method, which has come to be known as **slotted ALOHA**—in contrast to Abramson's **pure ALOHA**—a station is not permitted to send whenever the user types a line.

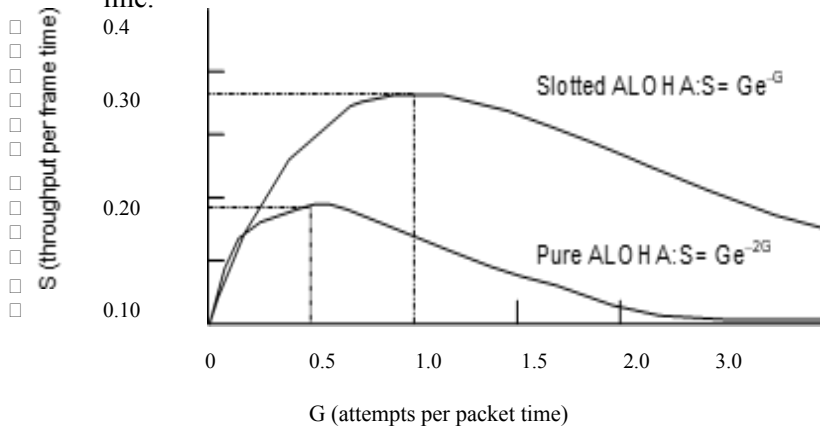


Figure 4-3. Throughput versus offered traffic for ALOHA systems.

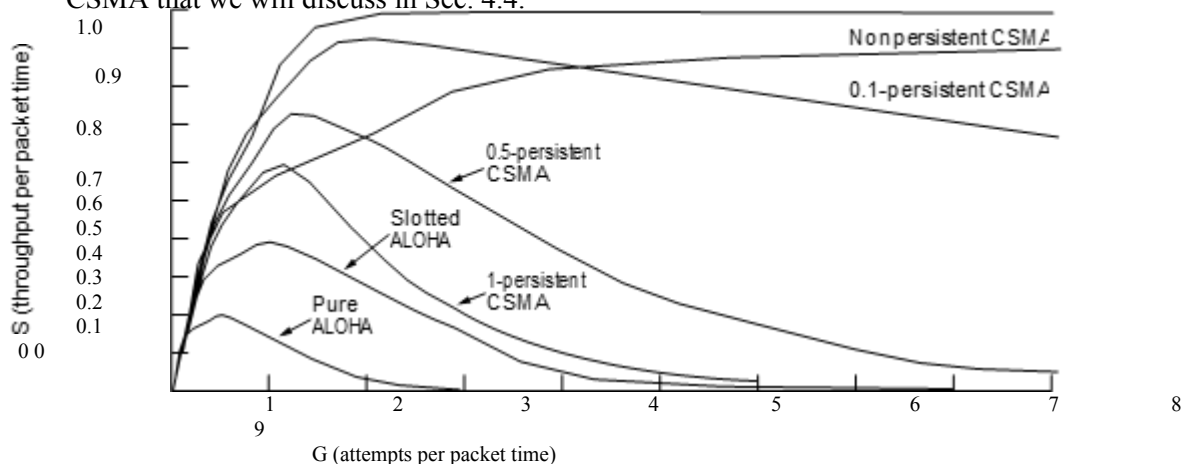
- Instead, it is required to wait for the beginning of the next slot. Thus, the continuous time ALOHA is turned into a discrete time one. This halves the vulnerable period. To see this, look at Fig. 4-3 and imagine the collisions that are now possible. The probability of no other traffic during the same slot as our test frame is then e^{-G} , which leads to
- $S = Ge^{-G}$
- As you can see from Fig. 4-3, slotted ALOHA peaks at $G = 1$, with a throughput of $S = 1/e$ or about 0.368, twice that of pure ALOHA. If the system is operating at $G = 1$, the probability of an empty slot is 0.368 (from Eq. 4-2). The best we can hope for using slotted ALOHA is 37% of the slots empty, 37% successes, and 26% collisions.
- Operating at higher values of G reduces the number of empties but increases the number of collisions exponentially. To see how this rapid growth of collisions with G comes about, consider the transmission of a test frame. The probability that it will avoid a collision is e^{-G} , which is the probability that all the other stations are silent in that slot.
- The probability of a collision is then just $1 - e^{-G}$. The probability of a transmission requiring exactly k attempts (i.e., $k - 1$ collisions followed by one success) is
- $P_k = e^{-G}(1 - e^{-G})^{k-1}$

◆ Carrier Sense Multiple Access Protocols

- This low result is hardly surprising, since with stations transmitting at will, without knowing what the other stations are doing there are bound to be many collisions.
- In LANs, however, it is often possible for stations to detect what other stations are doing, and thus adapt their behavior accordingly. These networks can achieve a much better utilization than $1/e$. In this section, we will discuss some protocols for improving performance.
- Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called **carrier sense protocols**.
- A number of them have been proposed, and they were long ago analyzed in detail. For example, see Kleinrock and Tobagi (1975). Below we will look at several versions of carrier sense protocols.

◆ Persistent and Nonpersistent CSMA

- The first carrier sense protocol that we will study here is called **1-persistent CSMA (Carrier Sense Multiple Access)**. That is a bit of a mouthful for the simplest CSMA scheme. When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment.
- If the first station's signal has not yet reached the second one, the latter will sense an idle channel and will also begin sending, resulting in a collision. This chance depends on the number of frames that fit on the channel, or the **bandwidth-delay product** of the channel.
- This protocol has better performance than pure ALOHA because both stations have the decency to desist from interfering with the third station's frame. Exactly the same holds for slotted ALOHA.
- A second carrier sense protocol is **nonpersistent CSMA**. In this protocol, a conscious attempt is made to be less greedy than in the previous one. As before, a station senses the channel when it wants to send a frame, and if no one else is sending, the station begins doing so itself.
- The last protocol is **p-persistent CSMA**. It applies to slotted channels and works as follows. When a station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability p . With a probability $q = 1 - p$, it defers until the next slot. If that slot is also idle, it either transmits or defers again, with probabilities p and q .
- This process is repeated until either the frame has been transmitted or another station has begun transmitting. In the latter case, the unlucky station acts as if there had been a collision (i.e., it waits a random time and starts again). If the station initially senses that the channel is busy, it waits until the next slot and applies the above algorithm. IEEE 802.11 uses a refinement of p-persistent CSMA that we will discuss in Sec. 4.4.



G (attempts per packet time)

Figure 4-4. Comparison of the channel utilization versus load for various random access

protocols.

Figure 4-4 shows the computed throughput versus offered traffic for all three protocols, as well as for pure and slotted ALOHA.

❖ CSMA with Collision Detection

- Persistent and nonpersistent CSMA protocols are definitely an improvement over ALOHA because they ensure that no station begins to transmit while the channel is busy.
- Another improvement is for the stations to quickly detect the collision and abruptly stop transmitting, (rather than finishing them) since they are irretrievably garbled anyway. This strategy saves time and bandwidth.
- This protocol, known as **CSMA/CD (CSMA with Collision Detection)**, is the basis of the classic Ethernet LAN, so it is worth devoting some time to looking at it in detail. It is important to realize that collision detection is an analog process. The station's hardware must listen to the channel while it is transmitting.
- CSMA/CD, as well as many other LAN protocols, uses the conceptual model of Fig. 4-5. At the point marked t_0 , a station has finished transmitting its frame. Any other station having a frame to send may now attempt to do so. If two or more stations decide to transmit simultaneously, there will be a collision.

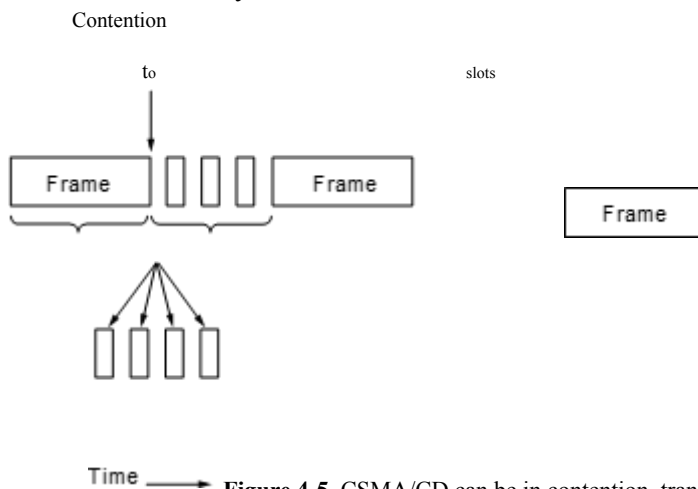


Figure 4-5. CSMA/CD can be in contention, transmission, or idle state.

- The minimum time to detect the collision is just the time it takes the signal to propagate from one station to the other. Based on this information, you might think that a station that has not heard a collision for a time equal to the full cable propagation time after starting its transmission can be sure it has seized the cable.
- Consider the following worst-case scenario. Let the time for a signal to propagate between the two farthest stations be τ . At t_0 , one station begins transmitting. At $t_0 + \tau$, an instant before the signal arrives at the most distant station, that station also begins transmitting.
- It detects the collision almost instantly and stops, but the little noise burst caused by the collision does not get back to the original station until time 2τ . In other words, in the worst case a station cannot be sure that it has seized the channel until it has transmitted for 2τ without hearing a collision.
- The difference for CSMA/CD compared to slotted ALOHA is that slots in which only one station transmits (i.e., in which the channel is seized) are followed by the rest of a frame. This difference will greatly improve performance if the frame time is much longer than the

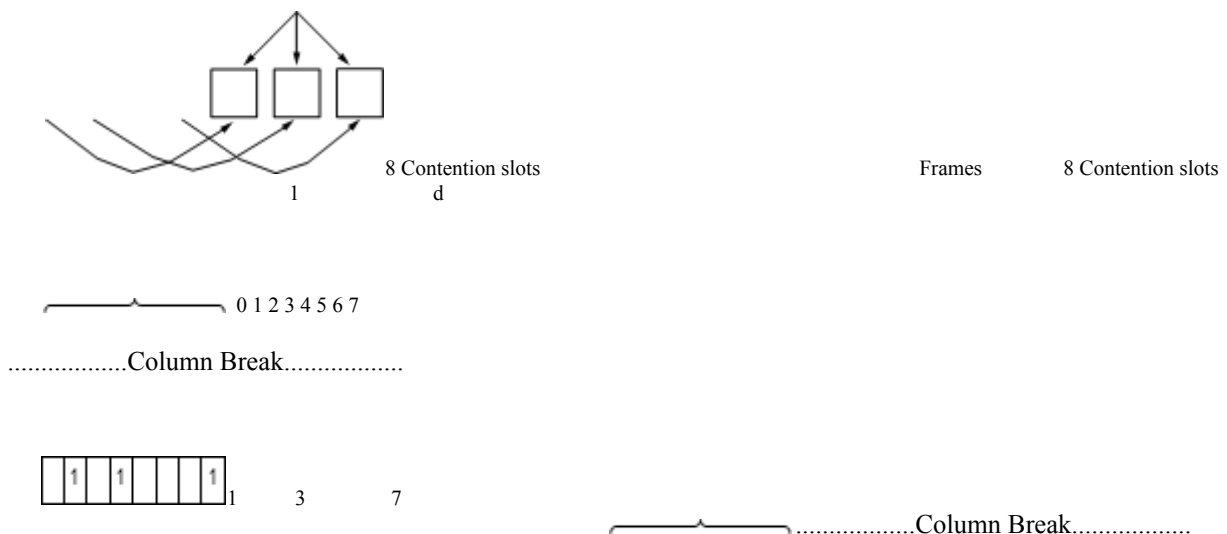
propagation time.

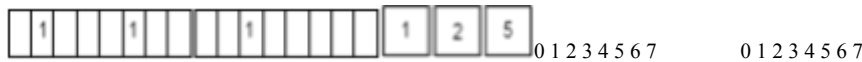
◆ Collision-Free Protocols

- Although collisions do not occur with CSMA/CD once a station has unambiguously captured the channel, they can still occur during the contention period. These collisions adversely affect the system performance, especially when the bandwidth-delay product is large, such as when the cable is long (i.e., large τ) and the frames are short.
- Most of these protocols are not currently used in major systems, but in a rapidly changing field, having some protocols with excellent properties available for future systems is often a good thing.
- In the protocols to be described, we assume that there are exactly N stations, each programmed with a unique address from 0 to $N-1$. It does not matter that some stations may be inactive part of the time.

◆ A Bit-Map Protocol

- In our first collision-free protocol, the **basic bit-map method**, each contention period consists of exactly N slots. If station 0 has a frame to send, it transmits a 1 bit during the slot 0.
- In general, station j may announce that it has a frame to send by inserting a 1 bit into slot j . After all N slots have passed by, each station has complete knowledge of which stations wish to transmit.





- Since everyone agrees on who goes next, there will never be any collisions. After the last ready station has transmitted its frame, an event all stations can easily monitor, another N -bit contention period is begun.
- Protocols like this in which the desire to transmit is broadcast before the actual transmission are called **reservation protocols** because they reserve channel ownership in advance and prevent collisions.
- Under conditions of low load, the bit map will simply be repeated over and over, for lack of data frames. Consider the situation from the point of view of a low-numbered station, such as 0 or 1. Typically, when it becomes ready to send, the “current” slot will be somewhere in the middle of the bit map.
- The prospects for high-numbered stations are brighter. Generally, these will only have to wait half a scan ($N/2$ bit slots) before starting to transmit. High-numbered stations rarely have to wait for the next scan.
- The channel efficiency at low load is easy to compute. The overhead per frame is N bits and the amount of data is d bits, for an efficiency of $d/(d + N)$.
- At high load, when all the stations have something to send all the time, the N -bit contention period is prorated over N frames, yielding an overhead of only 1 bit per frame, or an efficiency of $d/(d + 1)$.
- The mean delay for a frame is equal to the sum of the time it queues inside its station, plus an additional $(N - 1)d/N$ once it gets to the head of its internal queue. This interval is how long it takes to wait for all other stations to have their turn sending a frame and another bitmap.

▪ Token Passing

- The essence of the bit-map protocol is that it lets every station transmit a frame in turn in a predefined order. Another way to accomplish the same thing is to pass a small message called a **token**, from one station to the next in the same predefined order. The token represents permission to send.
- In a **token ring** protocol, the topology of the network is used to define the order in which stations send. The stations are connected one to the next in a single ring. Passing the token to the next station then simply consists of receiving the token in from one direction and transmitting it out in the other direction.

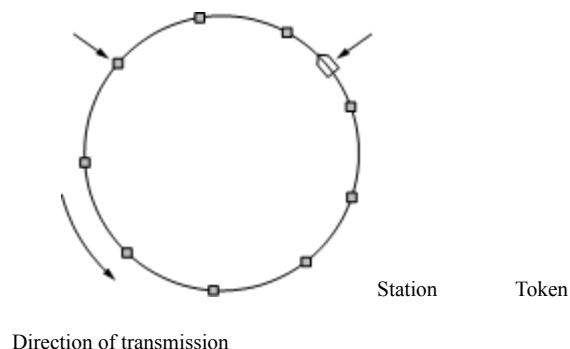


Figure 4-7. Token ring.

- The channel connecting the stations might instead be a single long bus. Each station then uses the bus to send the token to the next station in the predefined sequence. Possession of the token allows a station to use the bus to send one frame, as before. This protocol is called **token bus**.
- In the 1990s, a much faster token ring called **FDDI (Fiber Distributed Data Interface)** was beaten out by switched Ethernet. In the 2000s, a token ring called **RPR (Resilient Packet Ring)** was defined as IEEE 802.17 to standardize the mix of metro-politan area rings in use by ISPs. We wonder what the 2010s will have to offer.

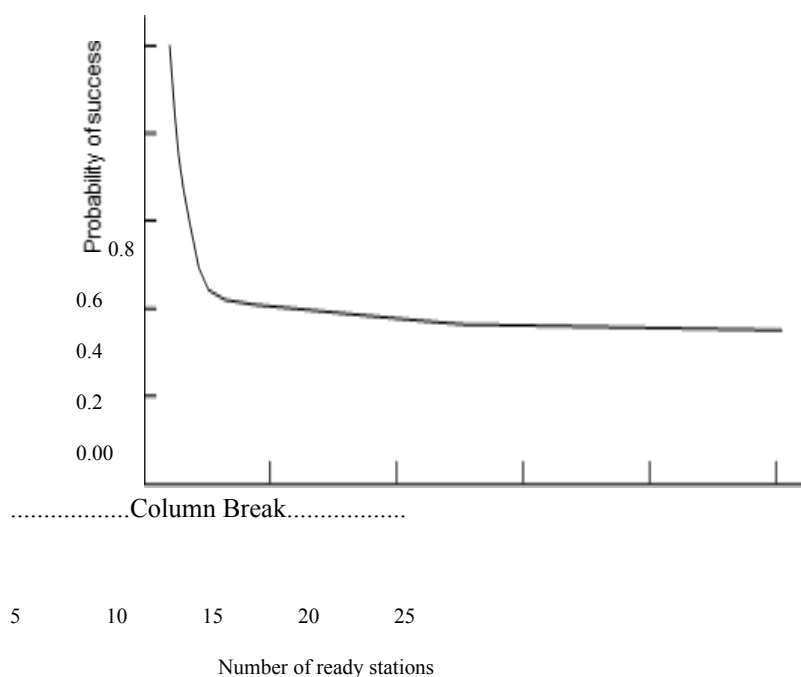
Binary Countdown

◆ Limited-Contention Protocols

- ◇ The contention and collision-free protocols, arriving at a new protocol that used contention at low load to provide low delay, but used a collision-free technique at high load to provide good channel efficiency. Such protocols, which we will call **limited-contention protocols**, do in fact exist, and will conclude our study of carrier sense networks.
- ◇ The only contention protocols we have studied have been symmetric. That is, each station attempts to acquire the channel with some probability, p , with all stations using the same p .
- ◇ Each has a probability p of transmitting during each slot. The probability that some station successfully acquires the channel during a given slot is the probability that any one station transmits, with stations defer, each probability p , and all other k with probability $1 - p$. This value is kp .

$$(1 - p)^k$$

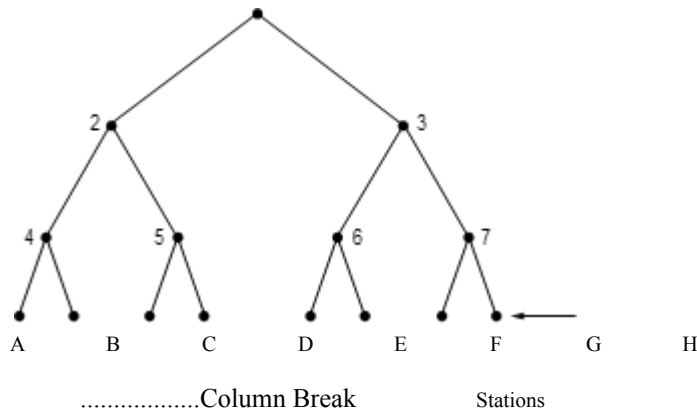
1.0



- They first divide the stations into (not necessarily disjoint) groups. Only the members of group 0 are permitted to compete for slot 0. If one of them succeeds, it acquires the channel and transmits its frame
- The limiting case is a single group containing all stations (i.e., slotted ALOHA). What we need is a way to assign stations to slots dynamically, with many stations per slot when the load is low and few (or even just one) station per slot when the load is high.

▪ The Adaptive Tree Walk Protocol

- A portion of each sample was poured into a single test tube. This mixed sample was then tested for antibodies. If none were found, all the soldiers in the group were declared healthy. If antibodies were present, two new mixed samples were prepared, one from soldiers 1 through $N/2$ and one from the rest. The process was repeated recursively until the infected soldiers were determined.



- In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node's left and right children.
- When the load on the system is heavy, it is hardly worth the effort to dedicate slot 0 to node 1 because that makes sense only in the unlikely event that precisely one station has a frame to send. Similarly, one could argue that nodes 2 and 3 should be skipped as well for the same reason.
- the expected number of them below a specific node at level i is just $2^i q$. Intuitively, we would expect the optimal level to begin searching the tree to be the one at which the mean number of contending stations per slot is 1, that is, the level at which $2^i q \approx 1$. Solving this equation, we find that $i \approx \log_2 q$.
- Numerous improvements to the basic algorithm have been discovered and are discussed in some detail by Bertsekas and Gallager (1992). For example, consider the case of stations G and H being the only ones wanting to transmit.

◆ Wireless LAN Protocols

- ◇ A system of laptop computers that communicate by radio can be regarded as a wireless LAN, as we discussed in Sec. 1.5.3. Such a LAN is an example of a broadcast channel. It also has somewhat different properties than a wired LAN, which leads to different MAC protocols.
- ◇ There is an even more important difference between wireless LANs and wired LANs. A station on a wireless LAN may not be able to transmit frames to or receive frames from all other stations because of the limited radio range of the stations.
- ◇ The naive approach to using a wireless LAN might be to try CSMA: just listen for other transmissions and only transmit if no one else is doing so. The trouble is, this protocol is not really a good way to think about wireless because what matters for reception is interference at the receiver, not at the sender.

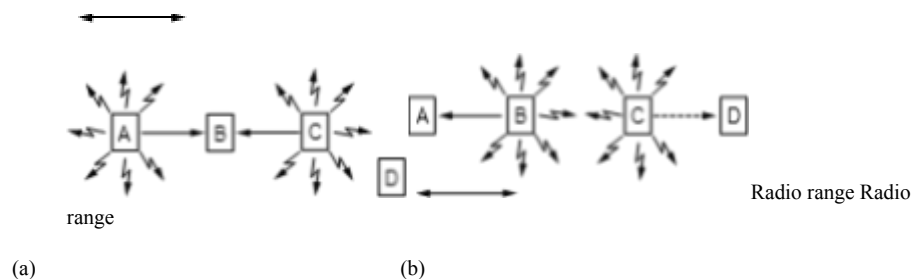


Figure 4-11. A wireless LAN. (a) *A* and *C* are hidden terminals when transmitting to *B*. (b) *B* and *C* are exposed terminals when transmitting to *A* and *D*.

- First consider what happens when *A* and *C* transmit to *B*, as depicted in Fig. 4-11(a). If *A* sends and then *C* immediately senses the medium, it will not hear *A* because *A* is out of range. Thus *C* will falsely conclude that it can transmit to *B*. If *C* does start transmitting, it will interfere at *B*, wiping out the frame from *A*.
- We want a MAC protocol that will prevent this kind of collision from happening because it wastes bandwidth. The problem of a station not being able to detect a potential competitor for the medium because the competitor is too far away is called the **hidden terminal problem**.
- In fact, such a transmission would cause bad reception only in the zone between *B* and *C*, where neither of the intended receivers is located. We want a MAC protocol that prevents this kind of deferral from happening because it wastes bandwidth. The problem is called the **exposed terminal problem**.
- We want this concurrency to happen as the cell gets larger and larger, in the same way that people at a party should not wait for everyone in the room to go silent before they talk; multiple conversations can take place at once in a large room as long as they are not directed to the same location.
- An early and influential protocol that tackles these problems for wireless LANs is **MACA (Multiple Access with Collision Avoidance)** (Karn, 1990). The basic idea behind it is for the sender to stimulate the receiver into outputting a short frame, so stations nearby can detect this transmission and avoid transmitting for the duration of the upcoming (large) data frame.
- MACA is illustrated in Fig. 4-12. Let us see how *A* sends a frame to *B*. *A* starts by sending an **RTS (Request To Send)** frame to *B*, as shown in Fig. 4-12(a). This short frame (30 bytes) contains the length of the data frame that will eventually follow.
- Then *B* replies with a **CTS (Clear To Send)** frame, as shown in Fig. 4-12(b). The CTS frame contains the data length (copied from the RTS frame). Upon receipt of the CTS frame, *A* begins transmission.

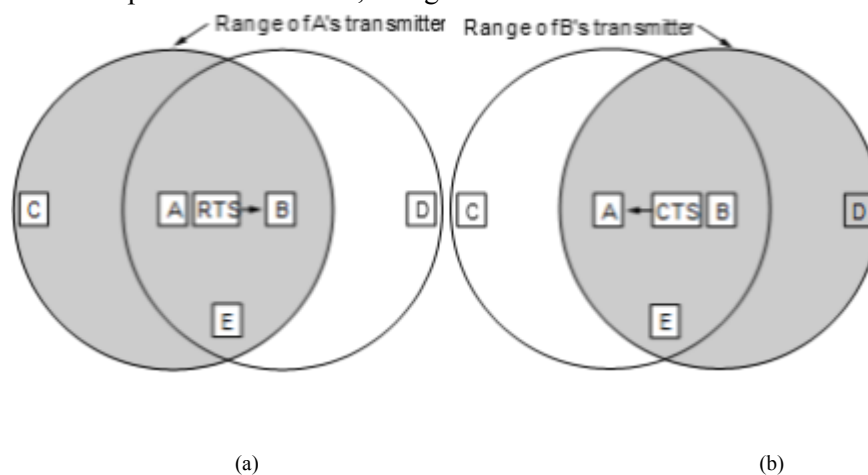


Figure 4-12. The MACA protocol. (a) *A* sending an RTS to *B*. (b) *B* responding with a CTS to *A*.

- Therefore, it hears the RTS from *A* but not the CTS from *B*. As long as it does not interfere with the CTS, it is free to transmit while the data frame is being sent. In contrast, *D* is within range of *B* but not *A*. Station *E* hears both control messages and, like *D*, must be silent until the data frame is complete.
- Despite these precautions, collisions can still occur. For example, *B* and *C* could both send RTS frames to *A* at the same time. These will collide and be lost. In the event of a collision, an unsuccessful transmitter (i.e., one that does not hear a CTS within the expected time interval) waits a random amount of time and tries again later.