CHAPTER 3

Distributed Medium Access Control in Wireless Networks

3.1 Introduction

Wireless channel is a shared medium, and multiple transmitters can potentially transmit on the same channel at the same time. This raises the possibility of interference. Scheduling or medium access control (MAC) protocols are useful in achieving reliable transmissions despite this potential for interference. Scheduling mechanisms can be broadly classified into two categories: centralized and distributed. In centralized protocols, a designated host (for instance, a base station), is given the responsibility to coordinate the access to the wireless channel. This is analogous to a teacher in a classroom deciding which student, from among those who have raised their hand, may ask the next question. On the other hand, distributed protocols allow the various hosts to coordinate channel access without assigning special responsibility to a single host. The distributed MAC protocols may be further classified into random access and coordinated access protocols [3]. Coordinated access protocols use a priori coordination to try to ensure that the transmissions performed at any given time will all be reliable (that is, achieve high enough SINR). One example of such protocols is *token-passing*; only a host holding a token is allowed to transmit a packet. This is analogous to a meeting wherein each person sitting at the table gets a turn to talk in, say, clockwise order around the table. Another example is a reservation-based scheme, where each host reserves time slots for its own transmissions. Thus, in any given time slot, only the hosts that have that slot reserved may transmit. Random access protocols, in contrast, do not use such a priori coordination. At any given time, each host may potentially attempt to transmit a packet. This is analogous to conversations in a group of people at a party: following some

social norms for politeness, the various individuals wishing to speak ensure that only one person talks at any given time (well, most of the time anyway).

In this chapter, we discuss some basic issues in the design of random access protocols for wireless networks. A goal in the design of such mechanisms is to utilize the wireless channel in the "best" possible manner. The best possible performance is subject to two factors:

- Wireless channel characteristics: With poor channel conditions, best achievable performance is poor as well.
- Physical layer constraints: The physical layer may constrain the performance in many ways. Here we discuss three such possibilities.
  - Consider a wireless link between hosts A and B. If the physical layer only supports transmissions at the rate of 1 Mbps, 2 Mbps and 11 Mbps, then the wireless link is limited to the maximum rate of 11 Mbps, even if the channel conditions may permit reliable transmissions at a higher rate.
  - Consider again a wireless link between nodes A and B, each equipped with one wireless interface (such as one IEEE 802.11g card). Suppose that the available spectrum is divided into several different channels of identical bandwidth, and each wireless interface is constrained to tune to any one channel at any given time. Due to this limitation, the link between A and B can use only a fraction of the available spectrum at any given time.
  - Consider an access point and multiple hosts that want to transmit data to this access point. If the access point is designed such that it can decode only one transmission at any given time, then only one host should transmit data to the access point at any given time. With a more capable access point, it may be possible for the access point to reliably receive multiple transmissions simultaneously.

Design of medium access control protocols is naturally dependent on the physical layer capabilities and constraints. Unless specified otherwise, our discussion in this chapter and later chapters will make the assumption that *each wireless interface may reliably receive at most one transmission at any given time.* Unless otherwise specified, each host is assumed to be equipped with one wireless interface.

3.2 *Basic* MAC Protocol

The discussion in this chapter will focus primarily on random access protocols. We begin our discussion of random access protocols with a *basic* MAC protocol. When designing

the MAC protocols, one important consideration is the propagation delay, which depends on the distance between the hosts and the speed of light. In our discussion on MAC protocols, we will only consider environments wherein the propagation delay is small compared to packet transmission times. When the distance between hosts is 300 meters, the propagation delay will be 1 microsecond (with speed of light 3×10^8 m/s), which is small compared to, for instance, transmission time of a 1500 byte packet at 100 Mbps (specifically, 120 microseconds). On the other hand, if hosts are 9,000 meters apart, then the propagation delay is 30 microseconds, which is not negligible compared to the above 120 microsecond transmissions time. Our discussion of MAC protocols implicitly assumes that the hosts using the protocol are relatively close to each other. The main consequence of the assumption is that the propagation time is a small fraction of the transmission time for a packet.

The *basic* MAC protocol we consider here does not use much intelligence at the hosts (the *basic* protocol is based on the Pure Aloha [1] protocol). In the basic protocol, whenever the MAC layer at a host receives a packet from upper layers of the protocol stack, it immediately transmits the packet on the wireless channel. This simple protocol can be adequate in certain environments, for instance, when the load on the network is low. However, in general, this approach suffers from two important shortcomings:

- The basic MAC protocol makes no provision to detect packet loss, and thus, cannot provide reliability at the MAC layer. However. the basic protocol can be augmented to improve its reliability, as discussed later.
- A packet sent by host S to host R is said to be lost due to *collision* if interference from another simultaneous transmission prevents reliable reception of the packet at host R. When the load on the channel is non-negligible, with the above basic protocol, two hosts may often attempt to transmit simultaneously, leading to a collision. The basic MAC protocol does not make any provision to detect such collisions, or to reduce the frequency of collisions.

Many solutions have been developed to mitigate the above shortcomings of the basic MAC protocol. The rest of this chapter discusses some of these solutions.

3.3 Slotted Access

The basic protocol allows a host to transmit a packet at any point of time. Let us consider an example of the use of this protocol. Assume that propagation delay between all host pairs is negligible. Consider three hosts S, R and P. Suppose that host S will have one packet to be sent to host R, and host P will also have one packet to be sent to host R. Assume that the transmission time for all data packets is L. Using the basic protocol,

each host will transmit its packet as soon as the packet is available. Suppose that host S sends the packet to host R starting at time $t0$. Assume that a collision would occur at host R if transmissions from hosts P and S overlap in time, preventing reliable reception of the packet from S. In general, in wireless networks, this is not always true. Reliability of packet reception depends on the signal-to-interference-and-noise ratio (SINR). Thus, even with transmissions overlapping in time, the transmissions may be received correctly. In the present discussion, however, we assume that a transmission can succeed only in the absence of any overlapping transmissions by other hosts. Figure 3.1(a) shows the *window of vulnerability* for the transmission from S, which is the duration in which a transmission initiated by host P will cause a collision at R. As can be seen from the figure, the length of the window of vulnerability is $2L$. With the basic protocol, if the packet arrives at host P at any time during the $2L$ interval, a collision will occur at host R, since host P will transmit a packet immediately upon receipt.

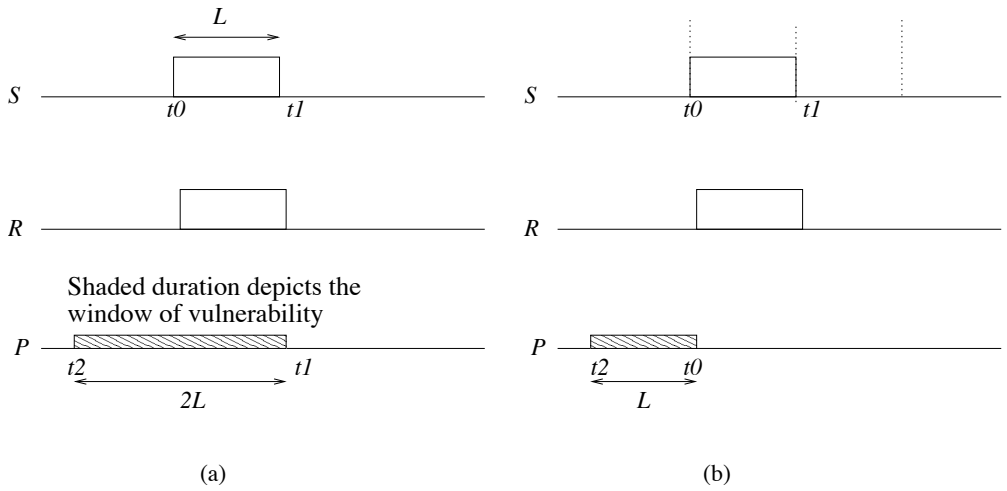


Figure 3.1 *Window of vulnerability: Illustration assumes negligible propagation delays*

The window of vulnerability can be reduced in half by utilizing a slotted access mechanism. This mechanism divides time into equal-length slots, each slot being long enough to transmit one packet. The slot boundaries at all the hosts are synchronized; the slot boundaries are shown by dotted lines in Figure 3.1(b). Each host is allowed to begin transmitting a packet only at the start of a slot. If the packet arrives at host P during a particular slot, it must wait until the start of the next slot. Thus, the transmission from host P will cause collision at host R only if P's packet arrives in the slot preceding the slot in which S transmits its packet, as shown in Figure 3.1(b). Therefore, the window of vulnerability is now 1 slot time or duration L.

Our discussion above assumed that propagation delays are negligible. In general, the propagation delays are non-negligible. Also the propagation delays may vary with time, and may be different between different pairs of hosts. However, we can design a slotted

access protocol for a certain worst-case propagation delay – in other words, the protocol will function as intended so long as the propagation delay between any pair of hosts is smaller than some value, say, τ. Under this assumption, we can achieve a window of vulnerability of 1 slot duration, by choosing a large enough slot size, as a function of maximum packet transmission time L and the propagation delay bound τ. In particular, if the slots begin at all hosts at the same time (that is, slot boundaries are synchronized), then slot size $L + \tau$ can be used while keeping the window of vulnerability equal to 1 slot. With this choice of slot size, regardless of the propagation delay, transmissions by two hosts in a given slot will arrive at the receiver(s) before that slot ends, assuming that the slot boundaries are synchronized. Thus, a transmission by a certain host will overlap in time with at most one transmission from another host, and thus the window of vulnerability is 1 slot.

In the above discussion, we assumed perfect synchronization, so that the slot boundaries occur at all hosts at the same time. Such a perfect clock synchronization is not feasible in practice. When the clocks are not perfectly synchronized. the *clock skew*, or the difference in the clock values at different hosts, can cause the slot boundaries at different hosts to occur at different times even if the hosts begin slots at the same clock value as per their *own* local clocks. This results in a window of vulnerability of 2 slots, if we use a slot size of $L + \tau$. To maintain a window of vulnerability of 1 slot, the slot size should be chosen while taking into account the clock skew. Suppose that the maximum clock skew is μ. Then, compared to host S, the slot boundary at host P may occur early or late by duration μ. By increasing the slot size to $L + \tau + \mu$, we can achieve a window of vulnerability of 1 slot. To achieve this goal, it is important that packets transmissions begin only at occurrence of a slot boundary at the transmitter.

Consider an example where host P transmits a packet to host R in a certain slot and host S also transmits in the same slot. Now, the transmissions from P and S may start at most μ interval apart, since the clock skew is at most μ. Similarly, the propagation delay from P to R can differ from the delay from S to R by at most τ. Thus, the transmissions from P and S can arrive at R at most interval $\tau + \mu$ apart. The "slack" of $\tau + \mu$ in the slot size ensures that the transmission from P to R can overlap with at most 1 transmission from S to R.

To analyze the benefits of slotted access, let us evaluate its performance under some simplifying assumptions. Let us suppose that there are n hosts, which are always backlogged, and each host transmits a packet in each slot with probability p. Let us call p the *access probability*. Let us further assume that if two or more hosts transmit in the same slot, then their transmissions will not be successful. Recall from our discussion of Binary PAM that transmission errors occur with some probability depending on the modulation scheme used, received signal power, etc. Thus, simultaneous transmission of two packets may not always result in both packets being erroneous. Also, depending on the path gains, the error probability for the two simultaneously transmitted packets may not necessarily be identical. However, in the current discussion, we make the simplifying assumption that when two

packets are transmitted simultaneously, they are both corrupted and discarded. Then, given that a host transmits a packet in a certain slot, the probability that the transmission will be successful is given by $(1-p)^{n-1}$: this is the probability with which no other host transmits a packet in the same slot. Thus, the throughput of the system, measured in average number of successful transmissions per slot, will be given by

$$\text{Throughput } = n\,p\,(1-p)^{n-1} \text{ packets/slot}$$

When $n = 1$, clearly $p = 1$ maximizes the throughput. When $n > 1$, it can be easily shown that the throughput is maximized when $p = 1/n$, and the maximum value of throughput is given by

$$\text{Maximum throughput} = \left(1-\frac{1}{n}\right)^{n-1}$$

In the limiting case, for large number of hosts,

$$\begin{aligned}
\lim_{n\to\infty}(\text{Maximum throughput}) &= \lim_{n\to\infty}\left(1-\frac{1}{n}\right)^{n-1} \\
&= \lim_{n\to\infty}\left(1-\frac{1}{n}\right)^{n}\left(1-\frac{1}{n}\right)^{-1} \\
&= e^{-1}\,.\,1 \\
&= \frac{1}{e} \text{ packets/slot}
\end{aligned}$$

Comparison of the optimal performance of the slotted mechanism for $n \to \infty$ with the unslotted mechanism is of interest. Recall that in case of the unslotted mechanism, the window of vulnerability is $2L$, where L is the time required for one packet. To simplify our analysis, we will actually evaluate a slotted scheme but with *unsynchronized* slots. In this case, each host uses the slotted mechanism, but the slot boundaries are not synchronized. The window of vulnerability is *at most* two slots (which is comparable to the window of vulnerability with the unslotted scheme). Let us consider the example in Figure 3.2. It is easy to see that a slot at one host overlaps with *at most* two slots at another host. For simplicity of analysis, let us assume that a slot at each host overlaps with *exactly* two slots at another host. For instance, slot i at host S in Figure 3.2 overlaps with slots i and $i+1$ at host P. If host S transmits a packet in slot i, then a collision will occur if host P sends a packet in slot $(i-1)$ or slot i. Thus, given n hosts, the probability of a successful transmission in slot i by host S will be $((1-p)^2)^{n-1} = (1-p)^{2n-2}$. Thus, we obtain the throughput (in packets/slot) as

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

Figure 3.3 compares the throughput with the two approaches (in both graphs, the lower curve corresponds to unsynchronized slots). Figure 3.3(a) plots throughput for $n = 100$ and different values of p, whereas Figure 3.3(b) plots the throughput for $p = 0.01$ and different

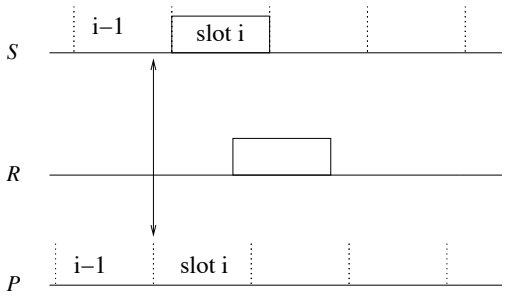
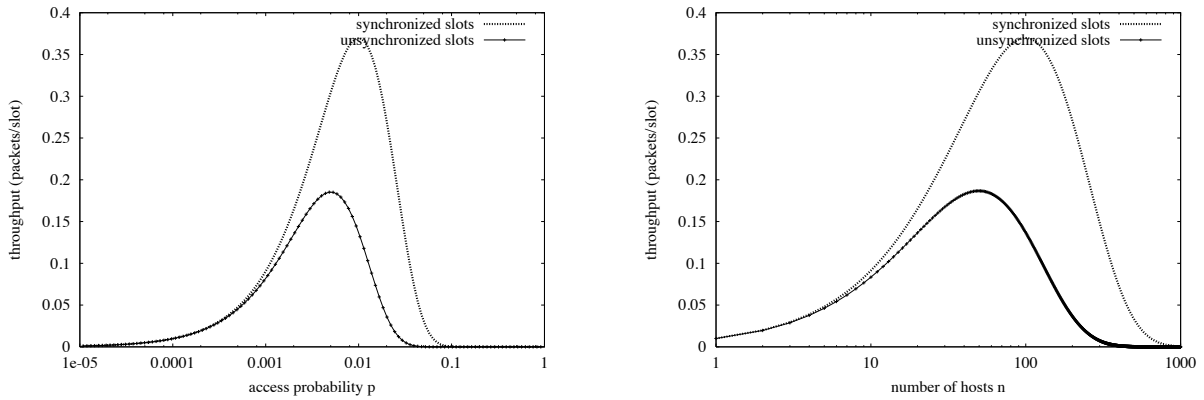


Figure 3.2 *Unsynchronized slots (illustration assumes that propagation delays are negligible)*

values of n. In the case of unsynchronized slots, the maximum throughput can be shown to occur at $p = \frac{1}{2n-1}$. In the limiting case with $n \to \infty$, the throughput is given by $\frac{1}{2e}$ packets/slot. Thus, in the limiting case (when $n \to \infty$), the lack of synchronization results in throughput degradation by a factor of 2 when compared to the case of synchronized slots. However, as a trade-off, the improved performance with synchronized slots requires clock synchronization among the hosts to be able to choose suitable slot boundaries.



(a) $n = 100$ (b) $p = 0.01$

Figure 3.3 *Throughput comparison of synchronized and unsynchronized slots: In both graphs, unsynchronized slots result in lower throughput*

3.4 Carrier Sensing

In the previous section, we improved on the performance of the basic MAC protocol by using a slotted access mechanism, which reduces the window of vulnerability for a transmitted packet. In this section, we consider an alternative approach to reduce collisions,

which may occur with the basic MAC protocol when multiple hosts in the vicinity of each other transmit packets concurrently. For instance, as seen earlier, if host S is transmitting to host R, and at the same time host P transmits its packet, then host R may not receive the transmission from host S reliably. Clearly, this situation is undesirable. Among a group of people standing together at a party, when one person is talking, the rest of the (polite) people keep quiet to avoid interrupting her. Similar mechanisms are useful in wireless networks to allow hosts to be *polite*. Carrier sensing is one such mechanism. Carrier sensing has been used in wired as well as wireless networks, however, the path loss characteristics of wireless networks provide more flexibility in the use of carrier sensing, as we will see later in the context of power control at the MAC layer.

Intuitively, carrier sensing requires that each host listen to the channel, and if the channel seems to be *busy* at a given time, then the host may not transmit at that time. Given that some noise is always present, the channel may never appear to be "idle" in the strictest sense. In other words, each host may always sense non-zero energy on the channel. Thus, mere presence of energy on the channel is not a sufficient mechanism for determining whether a channel is busy. Let us return to our example of a party. Typically, the people at the party will form several groups. While a person will keep quiet when someone in his own group is talking, the same deference is not shown when someone in another group farther away is talking. In other words, if someone close to John is talking at a given time, then John will not start talking himself. However, if no one nearby is talking, then John may talk even if he can hear someone far away talking at the same time. A simple strategy that John may use to achieve this outcome is to talk only when he cannot hear anyone else *loudly enough*.

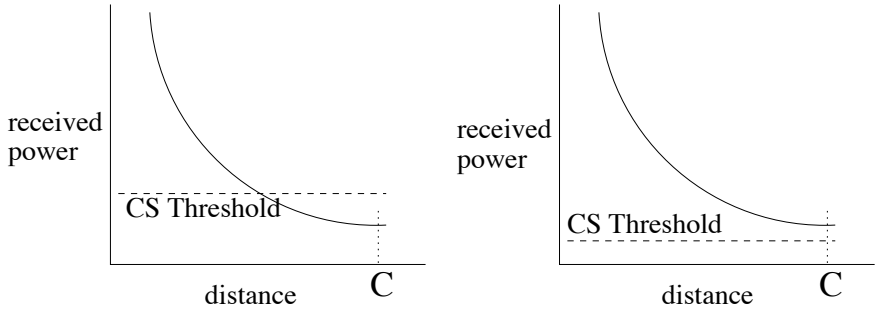
Carrier sensing can be implemented in wireless networks similarly using a *Carrier Sense Threshold* or *CS threshold*. In particular, if the received power level at a host exceeds the CS threshold, then that host may assume that the channel is busy. On the other hand, if the received power does not exceed the CS threshold, then the host may consider the channel as being idle. This approach is referred to as *energy detection*.

Energy detection is just one possible mechanism to detect the presence of ongoing transmissions. The energy detection mechanism ignores the structure of the signal being sensed, and instead, only considers the energy content of the signal. The alternative, referred as *feature detection*, looks for a particular pattern in the signal in order to detect a transmission. For instance, the physical layer transmission format often requires that a well-known a bit sequence, or *preamble*, be transmitted at the start of each packet transmission. Such a preamble can be used by the other hosts to detect that a packet is going be transmitted, without requiring a priori clock synchronization between the hosts (such synchronization is used in the slotted scheme in the previous section). Thus, if a host detects a preamble transmission on the channel, then it "senses" that a transmission is taking place. Energy detection and feature detection provide a trade-off between complexity of implementation

and the accuracy of sensing. In much of our discussion below, we will assume that carrier sensing is being performed using energy detection.

Carrier sensing using a CS threshold is illustrated using Figure 3.4. Consider host A transmitting to host B. In Figure 3.4, the curves show the received power level as a function of distance from host A; the example makes the simplifying assumption that the path loss consists of only the large scale path loss, and that the received power at host C is attributed entirely to the transmission from A. In general, noise and interference from other sources will also affect the outcome of the carrier sensing function. We will mostly ignore this issue when discussing the examples.

Whether host C will sense the channel busy or not depends on the choice of CS threshold at host C. In Figure 3.4(a), host C will perceive the channel as being idle because the received power is smaller than the CS threshold. On the other hand, if C were to use a smaller CS threshold, then as shown in Figure 3.4(b), host C will sense the channel as busy. As noted above, presence of noise and interference will also affect whether the channel is sensed busy or not. Also, since noise is non-deterministic, carrier sensing outcome is also non-deterministic. Thus, even if the received signal power is below the CS threshold, the presence of noise may *sometimes* result in the channel as being detected busy. We will ignore this non-deterministic nature of carrier sensing in our discussion here. However, it should be noted that, when the noise power level is non-trivial compared to the received signal power level, noise will have a significant impact on the efficacy of the energy detection mechanism.



(a) Large CS threshold (b) Small CS threshold

Figure 3.4 *Carrier sensing using CS threshold*

The example in Figure 3.4 shows that whether a host perceives the channel as being idle or busy depends on the choice of the CS threshold. Clearly, while host A is transmitting, host C can also begin transmitting when using the higher CS threshold in Figure 3.4(a), but not with the lower CS threshold in Figure 3.4(b). Which CS threshold is appropriate?

Suppose that the CS threshold is chosen as shown in Figure 3.4(a). Then host C may begin transmitting even after A has started transmitting a packet to B. This transmission

from C will cause interference at host B. Now, the reliability of the transmission from host A to B depends on the SINR at host B. If the transmission from host C causes the SINR at host B to be too small, leading to unreliable reception, then it is preferable that host C does not transmit at the same time. On the other hand, if the SINR at host B is high enough to ensure reliable reception, despite the transmission from host C, then it is better that C transmits as well. Such concurrent (reliable) transmissions can improve the aggregate throughput in the network, by improving the *spatial reuse* of the wireless channel. Thus, in the former case, it would be better to use the lower CS threshold as in Figure 3.4(b), whereas in the latter case, it would be better to use the higher CS threshold in Figure 3.4(a).

To summarize the above discussion, increasing the CS threshold will improve the spatial reuse of the wireless channel, while the SINR at the receivers may decrease. Clearly, there is a trade-off between the spatial reuse and the reliable transmission rate. Higher CS threshold will generally allow more simultaneous transmissions, but at a lower rate, as compared to a lower carrier sense threshold. Since the aggregate throughput in the network depends on spatial reuse as well as transmission rates, the CS threshold should be neither too small, nor too large. The optimal CS threshold depends on many parameters, including network topology, channel conditions, as well as the traffic patterns.

Let us now consider how the choice of the CS threshold affects interference. Consider the example in Figure 3.5(a). Let us assume that all transmitters use the transmit power P_t. Let us denote the path gain from a host X to a host Y as g_{XY}. When host A transmits a packet to host B at power level P_t, this transmission will be received by host C at power level $P_t \ g_{AC}$. Host C can pose interference at the receiver of A's transmission, namely, host B, only if host C actually transmits. For host C to begin transmitting when host A is already transmitting, the received power level $P_t \ g_{AC}$ at host C must be below the carrier sense threshold P_{CS}. As noted earlier, we ignore the noise in our analysis here. Thus,

$$P_t \ g_{AC} \ \leq \ P_{CS} \qquad \Rightarrow \qquad P_t \ \leq \ \frac{P_{CS}}{g_{AC}} \tag{3.1}$$

Now, since host C transmits at power level P_t, the interference I_{CB} posed by host C at host B will be $P_t \ g_{CB}$. The above inequality implies that

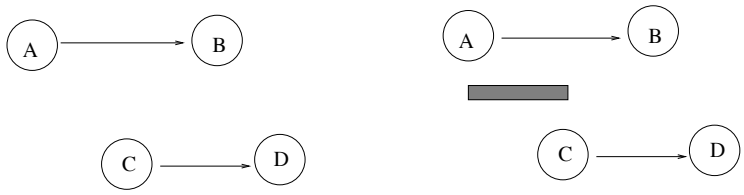
$$\text{Interference } I_{CB} \ = \ P_t \ g_{CB} \ \leq \ \frac{P_{CS}}{g_{AC}} \ g_{CB} = P_{CS} \ \frac{g_{CB}}{g_{AC}} \tag{3.2}$$

The above inequality provides an upper bound on the interference posed by host C at host B. Clearly, a smaller carrier-sense threshold (P_{CS}) used by host C results in a smaller bound on the interference from host C. In fact, the interference bound is linearly proportional to P_{CS}. Thus, the carrier sense threshold provides a mechanism for a host to limit its interference to other ongoing transmissions. In a similar manner, the carrier sense

threshold used by host C also bounds the interference posed by host A at the receiver of C's transmission, say host D. Again, suppose that host A is already transmitting a packet, and then host C transmits to host D despite A's ongoing transmission. Then, we must have $P_t \leq P_{CS}/g_{AC}$, as in Equation 3.1. Thus, the interference I_{AD} posed by host A at host D will be given by

$$I_{AD} \;=\; P_t\, g_{AD} \;\leq\; P_{CS}\, \frac{g_{AD}}{g_{AC}}$$

Thus, the carrier sense threshold used by host C affects the interference C may pose to other ongoing transmissions, as well as the interference at host C's receiver by other ongoing transmissions.



(a) No hidden terminal

(b) Obstacle makes carrier sensing difficult

Figure 3.5 *Example for carrier sensing: The dark box in the figure depicts a wall*

Hidden Terminals:

As discussed earlier, transmissions are said to *collide* at a host, say host R, if the transmission intended for host R is not received reliably by host R due to interference from another transmission or transmissions. While the carrier sensing mechanism discussed above is useful to reduce the possibility of collisions, it does not necessarily eliminate all collisions. To illustrate the limitations of the carrier sensing approach, let us consider the example in Figure 3.5(b). Suppose that while A is transmitting to host B, host C does not sense the channel as busy. In this case, the received power at C is low because the path loss from A to C is higher due to the presence of an obstacle (shown in black in the figure). However, if C begins transmitting, then the interference from C will be high at host B since there is no obstacle between B and C. This may cause a collision at host B, resulting in unreliable reception of A's transmission. Observe in Equation 3.2 that, if g_{AC} is small, the bound on interference becomes large.

Hosts A and C are said to be hidden from each other, since they may not be able to sense each other's transmissions, and yet, their transmissions can collide. To mitigate this limitation of carrier sensing, other mechanisms can be incorporated, as we will discuss later. Alternatively, the hosts can become more "sensitive" by using a smaller CS threshold. However, a smaller CS threshold increases the instances of *exposed terminals* discussed below.

Exposed Terminals:

While the presence of hidden terminals can result in a collision, exposed terminals result in a host unnecessarily deferring transmissions. Consider the example in Figure 3.6. In this case, suppose that concurrent transmissions from B and C, to hosts A and E, respectively, can complete reliably due to high SINR. However, since B and C are sufficiently close to each other, when B is transmitting to A, host C may sense the channel as busy, and delay its transmission to host E. Similarly, when C is transmitting to E, host B may defer transmission. Thus, although it may be possible to perform the two transmissions reliably concurrently, the use of carrier sensing may prevent this possibility. Clearly, by making the carrier sense threshold larger, we can prevent hosts B and C from deferring to each other. However, the larger carrier sense threshold may result in collision in other cases due to an increase in the interference, as seen earlier.

Figure 3.6 *Exposed terminals*

The above discussion on carrier sensing, hidden terminals, and exposed terminals leads to the following two conclusions:

- Carrier sensing alone cannot prevent all the collisions due to the presence of hidden terminals.
- An attempt to reduce the possibility of collisions due to hidden terminals by decreasing the CS thresholds can result in a reduction in spatial reuse due to exposed terminals.

Despite these limitations of carrier sensing, this mechanism is in use in practical protocols such as IEEE 802.11. An attractive feature of carrier sensing is that it is fully distributed. Each host can independently make decisions on when it is acceptable to transmit a packet, based on its local observation of the channel status.

3.5 Collision Detection and Avoidance

The term collision detection refers to the ability of a host to detect that its transmission has collided with another transmission, resulting in unreliable reception of its transmission at the intended receiver. Collision detection is incorporated in the Ethernet protocol. To perform collision detection, when a host using the Ethernet protocol transmits a packet, the host simultaneously receives signal from the Ethernet channel. A discrepancy between the transmitted signal and the received signal indicates the presence of another transmitter on the Ethernet, resulting in collision detection. The collision is detected within a short

time after a colliding packet begins transmission. The delay includes propagation delay and the time required to detect discrepancy between transmitted and received signals. This collision detection mechanism, however, requires the ability to transmit and receive signals simultaneously on the same channel.

In case of wireless networks, when a host transmits on a given channel, attempts to receive on the same channel simultaneously will result in the received signal being dominated by the transmitted signal. Perhaps more importantly, the collision that we want to detect occurs at the receiver, not the transmitter, of a packet. Thus, the transmitter cannot detect a collision accurately without some feedback from the receiver. This makes it difficult to implement Ethernet-style collision detection mechanism in wireless networks. Instead of collision detection, collision *avoidance* mechanisms have been incorporated in practical wireless MAC protocols, with the goal of avoiding or reducing the occurrence of collisions.

Collision with a transmission from a host A to a host B may occur from two types of interfering transmissions, which we will refer to as *simultaneous* and *concurrent* transmissions [24]:

- *Simultaneous* transmissions: Suppose that host A is close enough to host C such that A and C can carrier sense each other's transmissions using the chosen carrier sense thresholds. Now, host C can detect the transmission from host A by means of carrier sensing only if host C performs carrier sensing *after* host A has started its transmission. Let us assume that the maximum propagation delay between hosts A and C is τ, and that the maximum delay required for carrier sensing is σ. Also, let us define $\delta = \tau + \sigma$. As shown in Figure 3.7, if host A starts transmitting a packet at time t, then host C may not sense that transmission before time $t + \delta$. Thus, host C may start its own transmission during the interval $(t, t + \delta)$ despite performing carrier sensing. Now, if host C performs carrier sensing during the interval $(t - \delta, t)$, it will not sense A's transmission, since A is yet to begin transmission. Thus, host C may possibly start transmitting in the interval $(t - \delta, t)$. This transmission from host C will not be sensed by host A prior to t either (due to the propagation and carrier sensing delays). In summary, a transmission from host A that begins at time t can collide with a transmission from C that can start anytime during the interval $(t - \delta, t + \delta)$ despite carrier sensing. We refer to transmissions that occur during this interval as occurring *simultaneously* with the transmission from A at time t. It should be clear that the carrier sensing mechanism cannot help avoid collisions between such simultaneous transmissions. Later in this chapter, parameter δ will be used to define a slotted access scheme that uses carrier sensing.

- *Concurrent* transmissions: Again consider a transmission from host A to host B. In Figure 3.5(b), with the chosen CS threshold, host C is not able to sense the transmission from host A. In this case, even if host C were to sense the channel δ duration after host A's transmission commences, host C will still determine the channel to be idle,

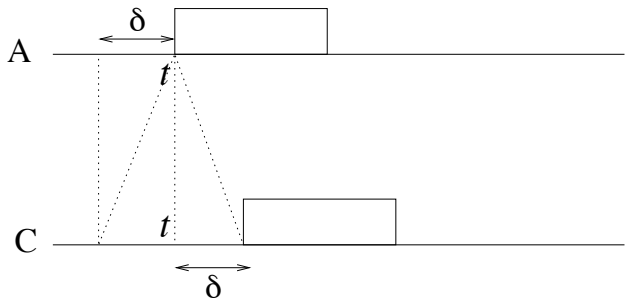


Figure 3.7 *Simultaneous transmission during* $(t - \delta, t + \delta)$ *can lead to a collision.*

and transmit. Such a transmission is said to be a *concurrent* transmission. The concurrency in transmissions is generally a desirable phenomenon, since it improves the spatial reuse of the channel. However, the concurrent transmissions can also lead to excessive interference, resulting in packet loss.

While a suitable choice of the carrier sense threshold may help reduce the interference due to *concurrent* transmissions, it does not help alleviate interference from *simultaneous* transmissions. Other collision *avoidance* mechanisms need to be utilized to reduce such interference. We will discuss some solutions for this purpose later in the chapter.

3.6 Reliability

Each transmission on the wireless channel can potentially be unreliable due to the presence of noise as well as interference from other transmitters. One issue to consider when designing the MAC protocols is whether the MAC layer should attempt to provide high reliability or not. As we will discuss in the context of transport protocol performance over wireless networks, it is often desirable to provide a reasonable level of reliability at the MAC layer. To achieve this, retransmission mechanisms as well as forward error correction mechanisms may be used.

We now discuss a simple retransmission scheme to improve reliability. In this scheme, when a host receives a data packet reliably, it is expected to immediately send an acknowledgement (Ack) to the sender. When the sender receives the Ack, it will know that its data packet was delivered reliably. On the other hand, if the sender does not receive the Ack within a suitable retransmission timeout interval, the sender will assume that the packet was not delivered reliably. While waiting for the timeout to occur, the sender will remain idle. When a timeout occurs, the sender may retransmit the packet to the receiver. The retransmission timeout interval should be long enough to allow time for the data packet to

reach the receiver, processing delay at the receiver in generating the Ack, and the time for the Ack to reach the sender.

This simple "stop-and-wait" protocol, wherein a host waits to receive an Ack before sending the next packet, is inefficient when the *delay-capacity* product of a link is large. Consider an analogy with cars traveling on a street, where we do not allow a new car to enter the street until the previous car has exited the street. If the street length is short so that it takes very little time for a car to travel the street, then this approach will keep the street busy most of the time. However, if the time required to travel on the street is very long, then we will keep the street underutilized. Similarly, the simple retransmission protocol will underutilize the channel if the delay-capacity product is large. However, in our discussion, we implicitly assume small delay-capacity product. Even when delay-capacity product is small, some improvement in performance can be obtained by sending an acknowledgement for a set of packets, as opposed to one acknowledgement for each packet. Note that, *delay-capacity* product is commonly referred to as *delay-bandwidth* product in the networking literature. To avoid confusion with the spectrum used (measured in Hertz), we will avoid using the term bandwidth in the present context.

Let us illustrate the above scheme using the example in Figure 3.8. In the example, the first transmission of a data packet from A to B is successful, but the resulting Ack from B is not received reliably by host A. When host A fails to receive the Ack within the timeout interval, it retransmits the data packet. In this case, the data was, in fact, received by B after the initial transmission, but host A has no mechanism to learn this, since the Ack was lost. The second transmission of the data packet is unreliable, thus, B does not receive the data, and does not send the Ack. After a timeout interval, node A again sends the data packet, followed by the transmission of Ack by B. When A receives the Ack, it knows that the packet is received by B reliably. Thus, for a "dialog" to conclude, the data packet must be received by B reliably, and the subsequent Ack from B must be received reliably by A. This implies that within a single dialog, both A and B act as receivers; B is receiver for the data, and A is receiver for the Ack. Due to this, we must take precautions to protect both data and Ack packets.

The maximum number of retransmission is usually chosen to be a finite number. Allowing for an arbitrarily large number of retransmissions can degrade performance when a link is broken because of host mobility. In such cases, retransmissions cannot deliver packets reliably.

As seen in the example in Figure 3.8, a host may receive the same packet multiple times due to the retransmissions. In this case, to allow the MAC layer at the receiver to detect duplicate packets, a unique sequence number can be associated with each packet. Packets with identical sequence number can be considered duplicates. Of course, since the sequence numbers will usually be assigned a fixed number of bits, the sequence numbers will be reused eventually. However, if the sequence numbers are not reused too close to each

other, then it is possible to avoid ambiguity between two different packets using the same sequence number.

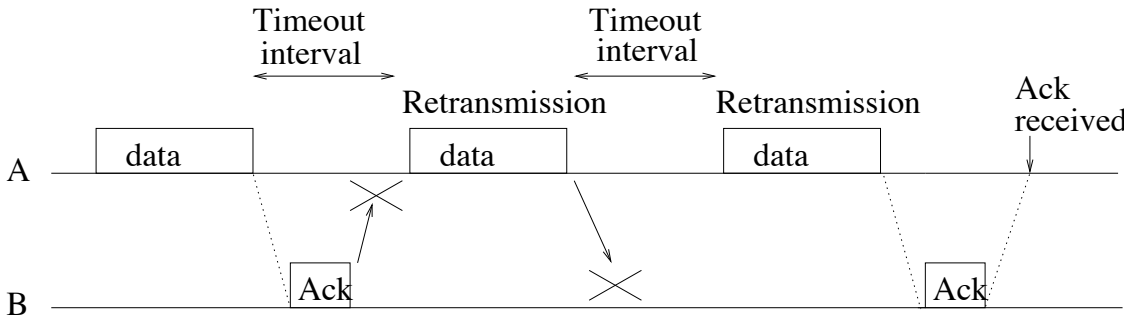


Figure 3.8 *Retransmission mechanism for reliability*

3.7 Overhead of Packet Collisions

When two transmissions collide, the time spent on the lost packets is wasted. Thus, it is important to incorporate mechanisms to reduce the collision overhead. Two factors affect the collision overhead:

- Cost of a collision: Cost of a collision is the channel time that is wasted in unfruitful communication. The time spent in transmitting the lost data packet is clearly a part of this cost. There may also be additional costs. Recall that, in the retransmission protocol discussed above, the retransmission interval includes time for the transmission of the Ack as well. During the retransmission timeout interval, while waiting for the Ack, the transmitter stays idle, thus not utilizing the channel.
- Frequency of collisions: The frequency with which collisions occur affects the total cost of collisions. Clearly, it would be nice to not have any collisions at all. However, eliminating all collisions will require a coordination among the hosts, which itself incurs an overhead. Random access protocols do not eliminate the collisions altogether, but attempt to reduce the frequency.

Clearly, to reduce the overhead due to collisions, it is useful to reduce the cost as well as frequency of collisions. In this section, we present a simple technique to reduce the cost of a collision. In the case of Ethernet, collision cost is reduced by quickly detecting a collision. In the wireless network, as discussed earlier, collision detection is difficult. Thus, even when a collision occurs, the entire data packet is transmitted. When the data packets are long, the

cost of the collision is high. A simple approach for reducing this cost of collision is to reserve the channel before sending the data, by first sending a shorter control packet. In particular, a host, which wants to transmit a data packet, first sends a Request-to-Send (RTS) packet specifying the intended receiver for the packet, and the duration of time required to send the data packet. The RTS may be sent using the basic mechanism discussed earlier. If no host transmits a packet that interferes with the RTS, the intended receiver for the RTS, as well as other nearby hosts in the network will receive the RTS reliably. Thus, the other hosts learn the duration of the impending data transmission, and can elect to stay silent for that duration, avoiding collisions with the subsequent data packet transmission. The intended receiver of the RTS, on receiving the RTS reliably, will respond by transmitting a Clear-to-Send (CTS) packet. Essentially, the CTS serves as an acknowledgement that the receiver has received the RTS reliably. When the sender receives a CTS from its intended receiver, the sender will then send the data. The presumption here is that, since the RTS was delivered reliably, the data will be delivered reliably as well. If a host does not receive a CTS from the intended receiver, the host does not transmit its data packet.

Clearly, the above scheme incurs the overhead of RTS and CTS. However, as a trade-off, the cost of a collision is reduced (unless the time required to send the data packet is smaller than that required for the RTS-CTS exchange). To quantify the relative benefits, let us assume that the data packets sent by any two hosts are identical in size, and that colliding transmissions begin at the same time. When not using RTS-CTS, let us denote the cost of a collision by L_{data}. L_{data} would include the transmission time for the data packet, and time for sending an Ack in response. On the other hand, when using RTS-CTS, let us denote the cost of a collision of the RTS by $L_{rts,cts}$. $L_{rts,cts}$ is the duration required for an RTS and CTS exchange.

With sufficiently large data packets, the cost of a collision when using RTS-CTS will be smaller. However, since RTS-CTS imposes an overhead even when there is no collision, the cost reduction in the presence of collisions needs to be sufficient to justify the increased overhead in absence of collisions. In particular, suppose that each transmission may be lost due to collision with probability p_c, incurring a cost of L_{data} or $L_{rts,cst}$, depending on the scheme used. Now, before a successful transmission occurs, on average, $p_c/(1-p_c)$ transmission attempts will result in collisions. Therefore, for the RTS-CTS mechanism to be beneficial, we must have that

$$\begin{aligned} \frac{p_c}{1-p_c}\, L_{data} + L_{data} &> \frac{p_c}{1-p_c}\, L_{rts,cts} + (L_{rts,cts} + L_{data}) \\ \Rightarrow p_c\, L_{data} &> L_{rts,cts} \end{aligned}$$

The above simplified analysis suggests that, if RTS-CTS impose low overhead, when compared to $p_c\, L_{data}$, then the use of the RTS-CTS mechanism will improve performance. In other words, if collisions are frequent enough and data packets are large enough, then RTS-CTS mechanism is beneficial. On the other hand, if $L_{rts,cts}$ is not small enough, then the RTS-CTS mechanism may not be beneficial.

Recall that collisions can occur due to interference from *simultaneous* transmissions, as well as *concurrent* transmissions. The solutions that might help reduce interference from (or collision with) concurrent transmissions will not always help in avoiding collisions from simultaneous transmissions. In particular, the above RTS-CTS solution will only be effective for collisions due to simultaneous transmissions. In the next section, we begin discussion of solutions for reducing the frequency of collisions.

3.8 Solutions for Hidden Terminals

The example in Figure 3.5(b) illustrates how the hidden terminal problem occurs with carrier sensing. The carrier sensing mechanism relies on the ability of an interferer, host C in Figure 3.5(b), to sense the transmission from host A. However, since the path gain between hosts A host C is low, host C is unable to detect the transmission from A. Yet, if C were to transmit, its transmission collides at host B with the transmission from host A. Since the path gain from A to C is low, an alternative mechanism is needed to *signal* to host C that it is unsafe to transmit. Following this intuition, two mechanisms have been developed to solve the hidden terminal problem: busy-tones, and virtual carrier sensing.

Busy-Tone Mechanism

This mechanism requires the use of an additional channel for the transmission of a busy-tone [21]. The busy-tone channel and the channel used for data occupy non-overlapping bands of spectrum. Conceptually, the busy-tone scheme is quite simple. When a host is receiving a packet, it announces that it is busy by transmitting a tone on the busy-tone channel. This busy-tone would be heard by the nearby hosts. Thus, potential interferers in the vicinity of host B can learn that it is not safe to transmit a packet. Such hosts will remain silent until the busy-tone channel becomes idle. In the example in Figure 3.5(b), when host B begins receiving a packet from host A, host B transmits a busy-tone. When host C senses the busy-tone signal from host B, it will remain silent, thus, preventing a collision at host B.

Despite the use of busy-tones, there is a short interval of time during which a transmission is vulnerable. In particular, it takes some time for host B in our example to detect the transmission from host A, and to begin transmission of the busy-tone; an interfering host may begin transmission before it receives the busy-tone from host B, potentially causing a collision.

The main distinction between the carrier sensing and the busy-tone scheme is in the signal that the potential interferers are required to sense. In particular, in carrier sensing,

the potential interferer would need to sense signal from the transmitter of an ongoing data transmission. On the other hand, the busy-tone is transmitted by the receiver of an ongoing data transmission.

To understand the importance of this relatively small distinction, let us do a simple analysis. Suppose that the transmit power for the busy-tone is P_t, identical to that used for data, and that threshold P_{CS} is used for sensing busy-tone and data channels both. Thus, the received power at host C for data packet transmitted by host A would be $R_{AC} = P_t\ g_{AC}$. On the other hand, the received power at host C for the busy-tone transmitted by host B would be $R_{BC} = P_t\ g_{BC}$. Now, if the path gain g_{AC} is small, then R_{AC} may be below the CS threshold, and yet R_{BC} may be above the CS threshold, if gain g_{BC} is large enough. Thus, it is indeed possible that C may detect the busy-tone sent by B, even if it could not carrier sense the transmission from host A. Similar to the derivation of Equation 3.2, if host C transmits a packet despite the fact that host B is transmitting the busy-tone signal, then it follows that $P_t\ g_{BC} \leq P_{CS}$, or equivalently, $P_t \leq P_{CS}/g_{BC}$. Thus, the interference I_{CB} that might be posed by host C at host B is upper-bounded as shown below:

$$\text{Interference } I_{CB} \quad = \quad P_t\ g_{CB} \ \leq \ \frac{P_{CS}}{g_{BC}}\ g_{CB} = P_{CS} \quad \text{assuming } g_{BC} = g_{CB} \tag{3.3}$$

Observe that the above upper bound on the interference depends only on the carrier sense threshold used by host C. Thus, with the busy-tone, P_{CS} provides an accurate bound on the interference from each interferer. In contrast, recall from Equation 3.2 that, if host C relies on sensing the data transmission from transmitter host A, then the interference posed by host C at host B is bounded by $P_{CS}\ \frac{g_{CB}}{g_{AC}}$. This bound will be large when g_{AC} is small. Thus, carrier sensing a signal sent by the receiver, as opposed to the transmitter, provides a better bound on the interference posed by an interferer, since a host such as C can control its carrier sense threshold P_{CS}, but not the path gain g_{AC}.

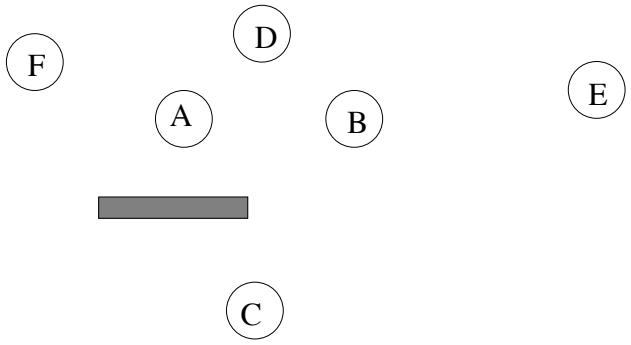


Figure 3.9 *Multiple interferers may be present*

How should we choose the carrier sense threshold P_{CS} for the busy-tone? We know that an interferer will not pose interference greater than P_{CS}. Thus, we could choose the

carrier sense threshold P_{CS} to be equal to the amount of interference we want to be able tolerate at the receiver. Such a choice can allow adequately reliable delivery despite interference from one interferer. But with multiple interferers, it is possible that the total interference may still exceed the tolerable limit. For instance, in Figure 3.9, suppose that host A is transmitting data to B, and that hosts E and F are just far enough that they do not sense the busy-tone from B, or the data transmission from A. Thus, hosts E and F may both potentially transmit packets that will interfere with data reception at host B. To allow for such multiple interferers, P_{CS} should be chosen to be a fraction of the total interference that is tolerable at a receiver. In particular, if total interference that can be tolerated is I, then to allow k interferers, the interferers should choose P_{CS} at most I/k. Now, with $P_{CS} = I/k$, it may actually be possible to tolerate more than k interferers if the interference from each of the interferer turns out to be lower than the P_{CS}. The number of hosts that might interfere with a given transmission is not always known a priori, however. In general, making P_{CS} smaller will result in lower interference, but at the same time, it will also allow fewer concurrent transmissions. Thus, there is a trade-off between spatial reuse and reliability, as we have discussed previously as well.

Although the busy-tone scheme discussed above is relatively simple, several issues affect its efficacy, as discussed next:

- Fading: Busy-tone requires additional spectrum, which increases the overhead of the access mechanism. To keep this overhead small, a narrow-band busy-tone channel may be used. However, narrow-band channels are more prone to fading, which reduces the reliability of busy-tone sensing.

- Gain differences between the busy-tone and data channel: In our derivation of Equations 3.2 and 3.3, we implicitly assumed that the path gains are identical for data and busy-tone channels. However, when the data and busy-tone signals use non-overlapping frequency bands, their propagation characteristics may not be identical. Thus, it is possible that the gain on the busy-tone channel between B and C is low, but the gain on the data channel between C and B is high. In this case, host C may not sense the busy-tone from B, but its transmission on the data channel can cause a collision at B. On the other hand, if busy-tone gain is higher, then host C may unnecessarily defer its transmission, similar to the exposed terminal problem discussed earlier. Thus, significant gain differences will defeat the busy-tone mechanism.

- Impact of the gain between a receiver and an interferer: Consider a transmission from host A to host B in Figure 3.5(b), with host C as the potential interferer. For this discussion, let us assume that the busy-tone and data channel have the same gain. What happens if the path gain between B and C is low, and host C fails to sense the busy-tone transmission from host B?

  Fortunately, a low gain between hosts B and C does not have the same detrimental effect, as a low gain between A and C has for the carrier sensing scheme. If the path

gain from B to C is low, then the gain is also low from C to B. Thus, if C were to transmit, it will cause low level of interference at B due to the low path gain (recall that we are assuming identical gains for busy-tone and data channels). In the analysis of the upper bound on interference when using the busy-tone, notice that the upper bound is independent of the gain g_{CB} (Equation 3.3). Thus, a smaller value for the gain does not affect the worst-case interference from an interferer.

A caveat is in order here. The time interval over which the busy-tone is sensed by host C, and the time interval over which C makes an interfering transmission are disjoint. Gain g_{BC} in the derivation of Equation 3.3 affects sensing of the busy-tone, whereas g_{CB} affects the interference posed by C. In our analysis, we assumed that these two gains are identical. However, if the channel changes rapidly, then the assumption that $g_{CB} = g_{BC}$ will no longer be valid, and the benefit of sensing the busy-tone would be reduced accordingly. The assumption of symmetry (that is, $g_{BC} = g_{CB}$) may also be incorrect in practice due to differences in receiver implementations at different hosts. Also, as noted earlier, since busy-tone and data are sent on different channel, g_{BC} and g_{CB} may be different.

- Reliability of Ack packets: As we discussed previously, a simple mechanism to improve reliability is to require the receiver to send an acknowledgement (Ack) to the sender of the data packet, and to require the sender to retransmit a packet if an Ack is not received within a suitable timeout interval. Consider the use of busy-tone in the example in Figure 3.10 wherein host A sends data to host B. In this case, the busy-tone sent by B may protect data reception at host B by preventing a concurrent transmission from host C. If carrier sensing on data channel is also used in addition to busy-tone sensing, hosts in the vicinity of host A may defer transmission while host A is transmitting the data packet. However, this does not ensure that a collision will not occur when host A receives the Ack from host B. Consider two potential solutions for this problem:

  - The first solution is to require a host to remain silent not only while it is sensing the data channel as busy, but also for a certain interval after the data channel is sensed as changing from busy to idle state. This additional interval should be long enough to allow a host, such as host A in our example in Figure 3.10, to receive the Ack subsequent to transmitting a data packets. With this approach, host D in Figure 3.10 will not cause interference with Ack reception at host A. A disadvantage of this approach is that host D will remain silent for the above interval even if host A does not receive an Ack because its transmission of data to host B is unreliable.
  - The second solution is to also require host A to send a busy-tone when it is receiving the Ack packet. Thus, unlike the above solution, host D will not have to stay silent unnecessarily when host A does not receive an Ack.

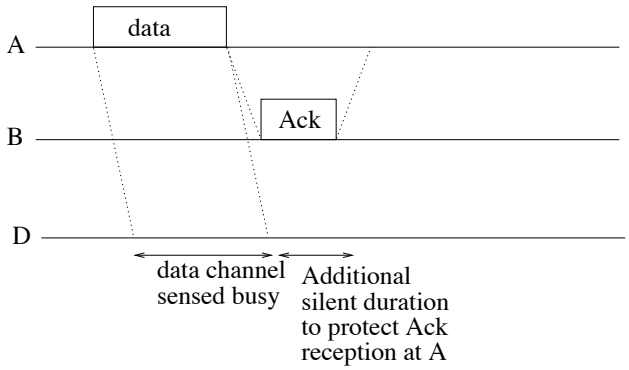


Figure 3.10 *Protecting Ack transmission when using busy-tone and carrier sensing*

Now we discuss an alternative to the busy-tone mechanism. In our subsequent discussion, we will often refer to carrier sensing mechanism discussed so far (using a carrier sense threshold) as *physical* carrier sensing, to distinguish it from the *virtual* carrier sensing mechanism to be introduced below.

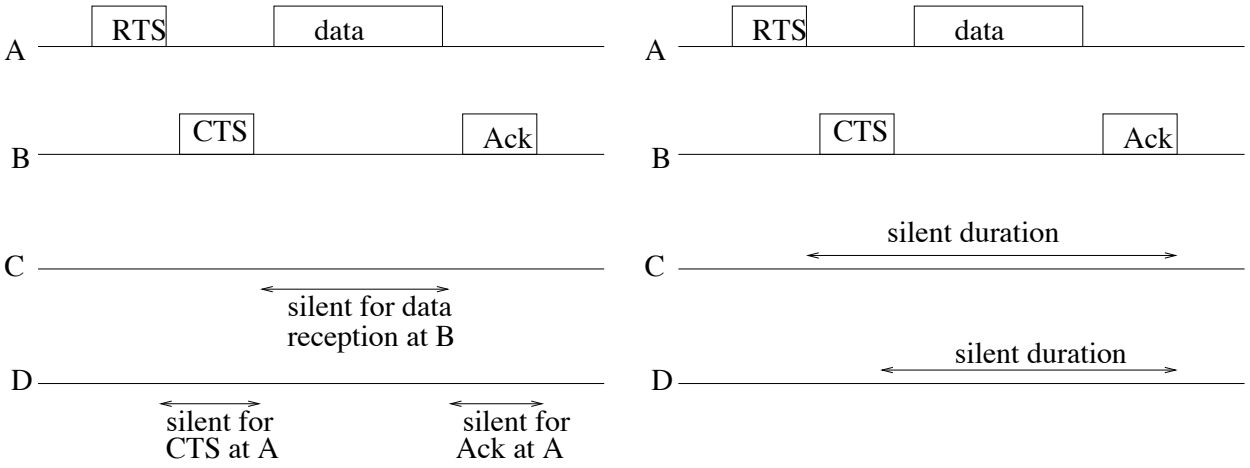
Virtual Carrier Sensing

The use of busy-tone introduces two complexities: First, a separate busy-tone channel is necessary for the busy-tone, requiring the use of additional bandwidth. Second, the hosts are required to transmit a busy-tone while receiving on the data channel, requiring more complex hardware. To overcome these obstacles, an alternative mechanism that relies on additional control packets on the data channel can be used. The alternative mechanism, as described below, also introduces overhead. However, the control packets can be sent on the same channel as the data. This mechanism, called virtual carrier sensing (VCS), is obtained by modifying the RTS-CTS mechanism. We previously saw that the use of RTS-CTS can help reduce the cost of collisions. With a simple modification, RTS-CTS can also help reduce collisions from hidden terminals.

Recall that the RTS-CTS approach we discussed previously requires that, before a host such as host A in Figure 3.9 transmits a data packet, it should send a RTS packet to the receiver, namely, host B in our example. The RTS packet specifies the time duration required for the proposed transmission; essentially, this duration specifies when host A needs the nearby hosts to remain quiet so as to complete the transmission to B successfully. If the hosts that can interfere with the reception at host B are all among the hosts receiving the RTS from A, the RTS may potentially suffice to avoid collision with the data packet from host A to host B. Due to the hidden terminals, hosts such as host C in Figure 3.9 might not receive the RTS from A. However, similar to the busy-tone from host B, host C has a better chance of receiving the CTS from host B. If we augment the CTS to include the duration

of the proposed data transmission from host A to host B, then host C can learn how long it must stay silent in order to avoid interfering with the reception of the data packet at host B. Thus, host C will remain quiet even though it may not be able to physically carrier sense the transmission from A. The mechanism using RTS-CTS is, therefore, called *virtual* carrier sensing, whereas the carrier sensing mechanism described earlier is called *physical* carrier sensing.

The above mechanism may prevent host C from interfering with data reception at host B (in Figure 3.9). What about Ack reception at host A? Protection for Ack reception can be provided by requiring the host receiving the RTS to be silent for the interval when ACK is expected. The resulting silent intervals at hosts C and D are shown in Figure 3.11(a). In this case, host D receives RTS from A, and stays silent when A is expected to receive the CTS and ACK. Similarly, host C receives the CTS from B, and stays silent when B is expected to receive the data. If virtual carrier sensing is used in conjunction with physical carrier sensing, then host D may sense RTS and data transmissions from A, and also stay silent during these transmissions; similarly, C will also be silent when it senses transmission of CTS and Ack from B.



(a) Silent intervals to avoid collision

(b) Conservative silent intervals

Figure 3.11 *Using RTS-CTS handshake to overcome hidden terminal problem*

What happens if host D also hears a RTS from another host, say F? Then host D will have to stay silent long to allow the transmission from F to complete reliably. The silent intervals for F's transmission may be disjoint or overlapping with the silent intervals required for A's transmission. Keeping track of these silent intervals can become somewhat cumbersome. Also, the window of time between the two silent intervals (see Figure 3.11) during which D must be silent might not be long enough for D to communicate with another

host. A simplification of the protocol can be achieved by requiring all hosts that receive an RTS or CTS to stay silent long enough to allow time for the subsequent data and Ack transmissions to complete. With this modification, in our example, host C and D both will stay silent long enough to allow host A to receive the Ack, as shown in Figure 3.11(b). There are two other reasons why this modification is desirable. Refer to the topology in Figure 3.5 for this discussion.

- First, it may be the case that host C is in the transmission range of hosts A and B both, but due to interference during the time when host A transmits the RTS, host C does not receive the RTS. Subsequently, host C does receive the CTS. Now, by staying silent until the Ack reception at host A, host C can avoid interfering with the Ack.

- Second, and perhaps more importantly, even if host C may not receive the RTS from host A reliably, it is still possible for host C to cause enough interference at host A, causing unreliable Ack reception. In Chapter 2, we saw that reliability of the reception of a packet is a function of the SINR. Let us assume the SINR-threshold model from Section 2.7. In particular, assume that a packet will be received reliably if the SINR for the packet exceeds a certain threshold β. Assume that noise power at each host is N. Now, the $SINR_C$ at host C for the RTS transmission from host A is ay most $P_t\ g_{AC}/N$, even without any interference. On the other hand, $SINR_A$ for the Ack transmission from host B to A, while enduring C's interference would be bounded as

$$SINR_A \leq \frac{P_t\ g_{BA}}{P_t\ g_{CA} + N}$$

  Is it possible that $SINR_C < \beta$ and $SINR_A < \beta$ both? If both these conditions hold then C might not receive A's RTS, and yet its interference will cause a collision with Ack reception at host A. These two conditions will hold true, if

$$\frac{P_t\ g_{AC}}{N} < \beta$$

  and

$$\frac{P_t\ g_{BA}}{P_t\ g_{CA} + N} < \beta$$

  By rearranging the terms in the above two inequalities, we obtain the following two inequalities.

$$g_{AC} < \frac{\beta\ N}{P_t}$$

  and

$$g_{CA} > \frac{g_{BA}}{\beta} - \frac{N}{P_t}$$

Then, with $g_{CA} = g_{AC}$, we must have

$$\frac{g_{BA}}{\beta} - \frac{N}{P_t} < \frac{\beta\ N}{P_t}$$

The above inequality can be rewritten as

$$g_{BA} < \frac{\beta(\beta+1)N}{P_t}$$

Clearly, the path gains may turn out to be such that the above condition holds. In essence, the fact that host C does *not* receive an RTS from A is not sufficient to conclude that host C will not cause a collision at host A. Therefore, to be conservative, host C may stay silent long enough to allow host A to receive the Ack from C, as shown in Figure 3.11(b). Similarly, a host that does not receive the CTS may conservatively stay silent during the data transmission to avoid causing a collision with the data packet.

Such a conservative approach may sometimes keep the hosts silent unnecessarily. For instance, what if host A's RTS is not received reliably by host B? In this case, host B will not respond with a CTS, and host A will not transmit data. But host D that receives the RTS will stay silent to allow time for both data and Ack transmission. This is an example of the trade-off between different types overheads that can potentially occur in medium access.

Virtual carrier sensing mechanism can also be extended to data packets. In particular, the data packet can specify the duration of time required to complete the dialog after the data packet has been transmitted. Thus, a host such as D can determine that it needs to be silent during the Ack reception at host A, even if D does not reliably receive the RTS, but receives the data from A reliably.

The RTS-CTS handshake can be viewed as an attempt at reserving some "space" for the proposed data transmission. We illustrate the space reserved by RTS-CTS using the example in Figure 3.12. In this case, host A sends RTS to B, and B responds by sending CTS to A. All hosts that receive the RTS from host A are within the large circle centered at A, and similarly, the that receive the CTS are within the large circle centered at B. Such hosts, which are shaded in the figure, keep silent for the duration specified in the RTS and CTS, respectively. Thus, the shaded hosts do not interfere with the data transmission from A to B, or the Ack from B to A. Thus, the area "occupied" by these silent hosts can be deemed to have been reserved for A's transmission. Note that this is only an intuitive way to interpret the outcome of the RTS-CTS exchange, not a precise description. In this context it is helpful to recall, from our discussion of transmission range in Section 2.6 of Chapter 2, that transmissions are not guaranteed to be received reliably by all hosts within any particular area around a transmitter.

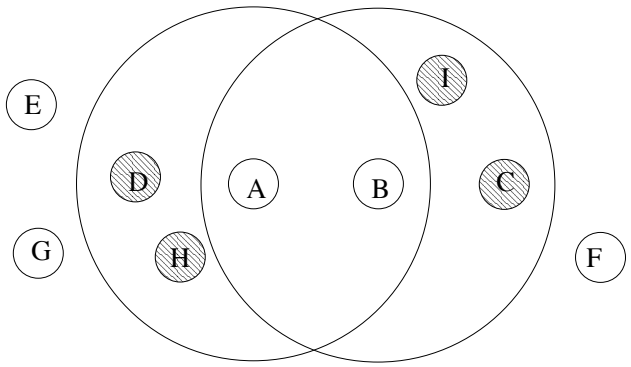


Figure 3.12 *Space reserved by virtual carrier sensing mechanism using RTS-CTS: Although the figure shows circular spaces, in reality, the shape will usually not be circular due to variations in path gains, and differing interference levels.*

As the above example illustrates, the use of RTS-CTS handshake reserves some area around the sender and receiver. Unlike the illustration in the figure, however, the area over which RTS and CTS transmissions are received reliably is not always circular, due to large scale and small scale path loss, and interference.

When only virtual carrier sensing is used (and physical carrier sensing is not used), the hosts that do not receive RTS or CTS can transmit packets. For instance, in our example in Figure 3.12, if only *virtual* carrier sensing is used, but no physical carrier sensing is used, then host E can quite possibly transmit when A is receiving an Ack from B, since E does not receive the RTS or the CTS. However, as the earlier discussion suggests, the transmission from E may still cause collision with the Ack from B to A. Therefore, it is beneficial to utilize physical carrier sensing, even if virtual carrier sensing is also being used. Hosts that do not receive the RTS or the CTS may possibly be able to sense the data and/or Ack transmissions (this is not guaranteed, of course). Thus, physical carrier sensing provides protection that is not provided by virtual carrier sensing. On the other hand, virtual carrier sensing can help avoid collisions in certain environments where physical carrier sensing does not work well. For instance, in Figure 3.9, host C may not be able to physically sense the data transmission from A, but the RTS-CTS exchange may help avoid a collision by host C, since C may receive the CTS from B. Thus, both physical and virtual carrier sensing techniques are of interest.

3.9 Reducing Collision Probability

The mechanisms discussed in Section 3.8 reduce possibility of collisions, however, they do not eliminate collisions altogether. In case of random access protocols, two hosts can po-

tentially start transmitting at approximately the same time leading to a collision. Such transmissions are said to be *simultaneous*, as defined earlier. Carrier sensing mechanism will not allow a host to detect other *simultaneous* transmissions. Thus, alternative mechanisms must be utilized to reduce probability of collisions due to such events. p-persistence, discussed in the next section, is such a mechanism.

3.9.1 p-Persistence

The basic idea here is that, at each *valid* transmission *opportunity*, a host may attempt to transmit a packet only with probability p. We have already considered such a mechanism earlier in Section 3.3, when no carrier sensing is used. Recall that, in that case, each host transmits in a given slot with probability p. Clearly, if p is made very small, rate of attempts to transmit will reduce, reducing collisions. However, a very small p will also degrade throughput by keeping too many slots idle. On the other hand, a large p will cause greater number of collisions. In general, optimal p is neither too large nor too small. In Section 3.3, we saw that for synchronized slot boundaries as well as unsynchronized slot boundaries, the optimal access probability p is a function of the number of hosts n competing for the channel. In that analysis, each slot was implicitly considered to be a *valid* transmission opportunity. That is, a given host may potentially transmit in every slot. With carrier sensing, the notion of a valid transmission opportunity needs to be defined somewhat differently.

Before we define a valid transmission opportunity, let us observe that when a valid transmission opportunity occurs, a transmitter (say, S) can initiate a new *dialog* with a receiver (say, R). The dialog may potentially be defined in different ways:

- The dialog may consist of only a data packet sent by S to R.
- The dialog may consist of a data packet sent by S to R, followed by an Ack sent by R to S, if R receives the data reliably.
- The dialog may consist of an RTS-CTS exchange between S and R, followed by data-Ack transmissions. Recall that CTS is sent only if S receives RTS reliably, data is sent only if S receives CTS reliably, and Ack is sent only if R receives data reliably. With RTS-CTS, we can implement virtual carrier sensing as discussed previously.

Depending on the definition of the dialog, the actual packets exchanged may vary. However, the dialog will be initiated only at a *valid* transmission opportunity, as discussed next.

Let us first consider the case when physical carrier sensing is used, but virtual carrier sensing is not used. We now introduce a slotted access scheme that takes advantage of physical carrier sensing. The time is divided into *slots*. For now, let us assume that the slot boundaries are synchronized at the different hosts. The start of each slot is a potential transmission opportunity. The slot size is chosen such that the following conditions holds:

- When a host begins transmitting at a certain transmission opportunity, each host within its "carrier sensing range" has at most one opportunity to start a simultaneous transmission. A host T is said to be in the carrier sensing range of a host S, if T can detect a transmission from S using carrier sensing.

What is the smallest slot size that can satisfy the above conditions? The discussion of *simultaneous* transmissions in Section 3.5 should provide a hint. In particular, let us define $\delta = \tau + \sigma$, where τ is the worst-case propagation delay, and σ is the physical carrier sensing delay. Thus, if a host, say host S, starts transmitting a packet at time t, then a host T within its carrier sensing range should sense this transmission by time $t + \delta$. It follows that if we make the slot size equal to δ, then host T will have at most one transmission opportunity during $[t, t + \delta)$ to begin transmitting a packet.

Having defined the slot size, let us now summarize the p-persistence mechanism:

- The slots are assumed to be synchronized at each node. The start of each slot is a potential transmission opportunity.
- Each host continually senses the channel to detect any ongoing transmissions. At a potential transmission opportunity, if the carrier sensing mechanism at a host has not detected the channel as being busy, then this transmission opportunity is said to be *valid* for that host.
- At each *valid* transmission opportunity, a host may begin transmitting a packet with probability p (provided, of course, that the host has a packet waiting to be transmitted).

What is the optimal access probability for p-persistence with physical carrier sensing? How does the performance compare with the case without carrier sensing? Consider the simple network wherein only one transmission can succeed reliably at any given time, and all hosts can sense each other's transmissions. Suppose that there are n hosts, which are always backlogged. Suppose that all packets require transmission time L, and that each *dialog* consists only of the data packet (thus, no RTS-CTS or Ack are sent). The analysis below also implicitly assumes that a packet finishes transmission exactly at a slot boundary. Once a transmitter stops transmitting, it takes the other nearby hosts up to 1 slot to detect the idle channel status. Thus, the first valid opportunity occur at these hosts 1 slot after the host stops transmitting. To be fair to the other hosts, after a host completes a transmission, it does not immediately begin transmission of a new packet, considering only the next slot boundary as a potential transmission opportunity. In essence, no host transmits for 1 slot interval after the completion of a transmission, and each transmission or collision consumes $L + \delta$ duration of time.

In this network, when the channel is idle, a new successful transmission will begin at the next slot boundary with probability $P_{success} = np(1-p)^{n-1}$; the probability that

a collision will occur due to multiple transmissions starting in the next slot is given by $P_{collision} = 1 - P_{success} - P_{none}$, where $P_{none} = (1-p)^n$. Since no collision detection is performed, once a collision occurs, duration $L + \delta$ is wasted. A successful transmission also requires $L + \delta$ duration, but results in the channel being fruitfully utilized for duration L, as shown in Figure 3.13. Thus, a successful transmission as well as a collision requires duration $L + \delta$, and the cumulative probability of these two events is $P_{success} + P_{collision} = 1 - P_{none}$. With probability P_{none}, the channel is kept idle for duration of 1 slot (δ duration). The

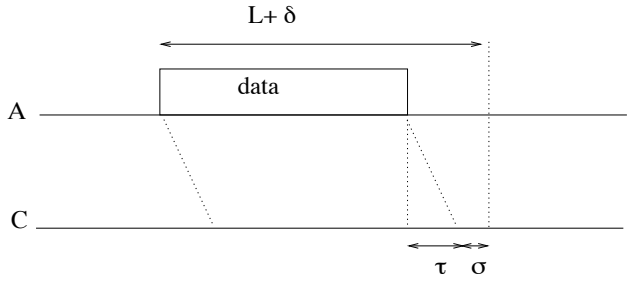


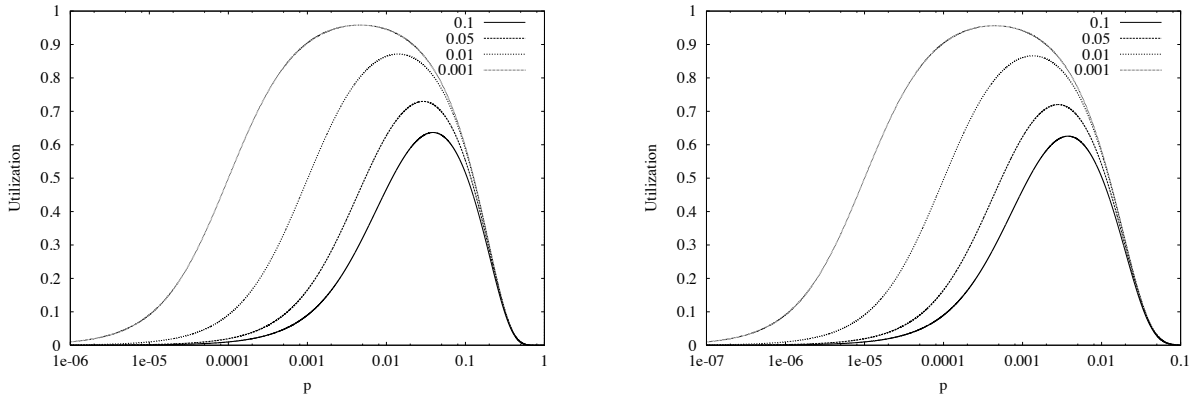
Figure 3.13 Duration required for a transmission

above discussion implies that the channel is used for successful transmissions for a fraction of time given by

$$\frac{P_{success}\ L}{P_{none}\delta + (1 - P_{none})\ (L + \delta)} \quad = \quad \frac{P_{success}\ L}{\delta + (1 - P_{none})\ L} \tag{3.4}$$

The above quantity can be interpreted as the efficiency of channel access, or the fraction of time for which data is transmitted reliably on the channel. Multiplying the above quantity by transmission bit rate yields throughput in bits/second. Observe that smaller δ will result in a greater efficiency for a given value of p. The optimal value of p that maximizes the efficiency is a function of δ as well as the number of competing hosts (n). Figure 3.14 plots the throughput assuming that $L = 1$, that is, the time required to transmit a packet is taken as the unit of time. The graphs plot efficiency as a function of p for $\delta = 0.001$, 0.01, 0.05 and 0.1.

Note that the above analysis made several simplifying assumptions, which may not hold in practice. First, we assumed that all hosts are close to each other such that they can all carrier sense each other's transmissions. In general, as discussed previously, not all hosts can carrier sense each other's transmissions. Also, different hosts will compete for the channel with potentially different sets of hosts. Thus, the optimal value of p may not be identical for all the hosts. Second, the transmissions may not always be reliable even in the absence of collisions (due to channel effects such as fading). Third, the optimal value of p may vary over time, due to time-varying network topology, as well as time-varying traffic patterns. Under such realistic conditions, the above analysis does not hold. However, the above simplified analysis provides intuition that can be useful in the general case.



(a) $n = 10$ (b) $n = 100$

Figure 3.14 Efficiency with physical carrier sensing. The different curves correspond to different values of $\delta = 0.001, 0.01, 0.05, 0.1$, with $L = 1$. The value of δ is measured here relative to the time required to transmit a packet. Thus, $\delta = 0.01$ means that δ is 1% of the time required to transmit a packet. The optimal efficiency decreases as δ increases, as may be expected.

What if the slots are not synchronized when using physical carrier sensing? When the slots are not synchronized, as we have seen previously, the vulnerability of a transmission increases. Suppose that the slot size is δ, but the slot boundaries are not synchronized Instead of analyzing the best possible performance with unsynchronized slots, let us consider a suboptimal scheme in which we only allow each host to consider every alternate slot as a potential transmission opportunity. This scheme is illustrated in Figure 3.15. Observe that when a host begins transmission at a certain transmission opportunity, other hosts have only one transmission opportunity that can cause interference with the ongoing transmission (this, of course, assumes that the hosts can sense the ongoing transmission).

In particular, suppose that each host may begin transmitting a packet at the start of slots that begin at ˆ marks in the figure (thus, each host only uses alternate slots as potential transmission opportunities). Suppose that host B begins transmission at the slot boundary labeled i. Notice that, if host C had started transmitting a packet at its slot boundary $i-1$, clearly host B would have sensed that transmission by its own slot boundary i, and would not start transmitting at that time (this assumes that the packet size is large compared to 2 slots). Also, once host B starts transmitting the packet at boundary i, host C will sense B's transmission by its own $i+1$ slot boundary. Similar argument holds for host A as well. Thus, host A and C only have 1 transmission opportunity (labeled i in the figure) to interfere with a transmission that begins at boundary i at host B. The performance of this scheme can be approximated using Equation 3.4 by replacing δ in that equation by 2δ. Then, the curve labelled as 0.1 in Figure 3.14 will be the approximate performance of asynchronous slots with $\delta = 0.05$. Comparing this curve with the curve labelled 0.05 in Figure 3.14, we

see that lack of synchronization does degrade performance, however, the degradation is less dramatic than the case of asynchronous slots in Section 3.3. Essentially, the use of carrier sensing allows us to use small slots, reducing the detrimental impact of asynchrony.

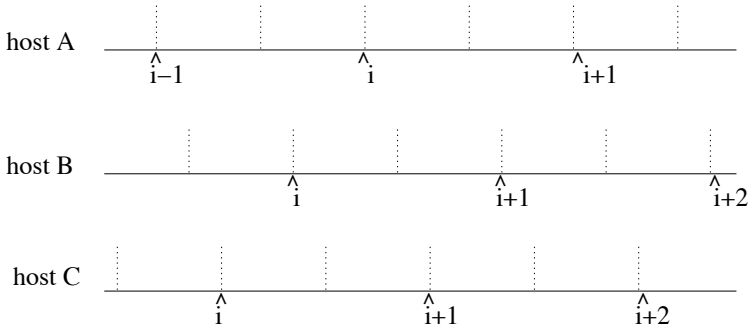


Figure 3.15 *Physical carrier sensing with unsynchronized slots*

In the discussion above, we assumed that physical carrier sensing is used, but virtual carrier sensing is not used. However, the approach can be easily extended to virtual carrier sensing as follows. Suppose that the slot size is δ, with a new slot boundary at a host occurring when it senses the medium as changing state from busy to idle. The busy or idle state of the channel may be detected using physical or virtual carrier sensing. As above, a backlogged host may transmit at a valid transmission opportunity with probability p. The slot boundaries in this case will not be synchronized at the different nodes, since the different nodes may not detect the medium as becoming idle at the same time.

3.9.2 Backoff Intervals

The p-persistence mechanism is memory-less in that the probability that a host transmits in a given slot does not depend on the duration since the last transmission attempt by that host. Thus, in theory, an arbitrarily large number of valid transmission opportunities (slots) may pass before a host attempts to transmit a packet. An alternative approach would be to ensure that the host will attempt to transmit after a bounded number of valid transmission opportunities. The goal of bounding the number of passed valid transmission opportunities can be achieved using a variety of approaches. Let us discuss one such approach, based on the notion of a *backoff interval*. In this case, as also with p-persistence, a host may begin to transmit in a given slot only if the channel is not detected busy (a host may detect busy channel status using physical or virtual carrier sensing). Such a slot presents a valid transmission opportunity. In the p-persistence approach, at each such opportunity, the host decides with probability p whether to transmit or not. With the backoff interval approach, on the other hand, the host picks a number b when a packet arrives at the MAC layer, and transmits at the b-th valid opportunity after the arrival of the packet.

This may be implemented as follows. When the MAC layer at a host receives a packet (from upper layers) to be transmitted, it picks a non-negative integer value for b. Suppose that we initialize a counter with this value. Subsequently, a valid transmission opportunity is considered to arise if the host senses the channel as being free at the start of a slot. In this case, the host may transmit if its backoff counter is 0; else the host decrements its backoff counter by 1. On the other hand, if the channel is sensed busy at the start of a slot, then this slot is not a valid transmission opportunity, and the counter is not decremented. The host may only transmit when the counter reaches 0.

Similar to the choice of suitable p in p-persistent protocol, in the context of backoff intervals, we need to somehow choose the appropriate backoff intervals. If the backoff intervals are chosen too large, then the channel stays idle for too much time, and if the backoff intervals are chosen too small, then there is a greater chance of collisions. The optimal choice of backoff intervals depends on the network topology and traffic patterns.

While the backoff interval b may be chosen to be a finite number, there is a certain risk in choosing b deterministically. For instance, a deterministic approach may be to always choose backoff interval b to be equal to 20. To see the risk of such a deterministic approach, suppose that two different hosts, A and B, happen to both choose $b = 20$, and suppose that the two hosts receive packets to be transmitted at the same time as well. If they are located such that they both observe the channel as busy or idle identically, then they will both attempt to transmit in the same slot, potentially causing a collision. If the choice of b is fixed, then this scenario can also repeat on each retransmission attempt by the two hosts. To avoid such detrimental *synchronization* of transmission attempts, the value of b may be chosen randomly over some range $[b_{min}, b_{max}]$, so that even if the two hosts transmit simultaneously on one attempt, they are likely to transmit at different times on the next attempt.

3.9.3 Responding to Packet Loss

Consider a host A that uses the backoff mechanism described above to determine when to transmit a packet. In particular, host A chooses a backoff interval b uniformly in the range $[b_{min}, b_{max}]$. For this discussion, let us assume that $b_{min} = 0$. Now suppose that MAC layer reliability is implemented by means of an immediate Ack from the receiver, followed by data retransmission if necessary. Thus, when host A transmits a data packet to, say, host B, host A expects an Ack right after sending the data packet. What happens if host A does not receive an acknowledgement? The lack of acknowledgement implies a packet loss. Either the data packet was not received reliably by the receiver, or the Ack sent by the receiver was not reliably received by the sender. The packet loss itself occurs when the SINR is sufficiently low. The absence of an Ack may be due to different causes:

- The data packet transmission may have collided with the transmission by another host that started transmitting simultaneously (i.e., within δ of A's transmission initiation). In this case, host B, not having received the data packet reliably, would not send an acknowledgement. Physical carrier sensing cannot prevent collisions due to simultaneous transmissions.

- Packet loss may also occur due to interference from concurrent transmissions by other hosts that do not sense the ongoing transmission, or interference from a non-cooperative source. The interference may result in the loss of the data packet sent by host A, or in the Ack sent by host B.

In all cases above, host A may choose to retransmit the data packet. How should the backoff value b be chosen for the retransmission? One option, of course, is to choose b again from the range $[0, b_{max}]$ as before, with the same value of b_{max}. However, if the cause of packet loss is collision due to simultaneous transmissions, we may want to reduce the probability that the two hosts will collide again when retransmitting. This can be achieved by increasing the value of b_{max}. One possibility is to increase the value of b_{max} such that the range $[0, b_{max}]$ is doubled before each retransmission. This approach is referred to as an *exponential backoff*. While increasing the value of b_{max} may seem appropriate in response to packet loss due to collision by simultaneous transmissions, this action may degrade throughput unnecessarily when the loss occurs due to other causes, such as interference from non-cooperating sources.

It should be noted that collisions are not a *symmetric* phenomenon. For instance, in Figure 3.9, suppose that hosts D and B want to transmit data to hosts A and C, respectively. Given the proximity of A and B, B causes significant interference at A, leading to a collision. However, D is sufficiently far from C that it does not cause much interference. In this case, simultaneous transmission of data by D and B may cause a collision at A, but not at C. This sort of asymmetry, when combined with exponential backoff, can lead to unfair division of channel time among the hosts, as we will discuss later.

3.10 Examples of Random Access MAC protocols

In this section, we briefly discuss two protocols, namely, Aloha [1] and IEEE 802.11 Distributed Coordination Function (DCF) [10].

The Aloha Protocol

Many issues discussed in this chapter, and their solutions, are motivated by the work on the classic Aloha protocol. The original Aloha protocol, developed in the 1970s, allows a

host to transmit a packet as soon as the packet arrives from the upper layers. The intended recipient of each packet is expected to send an acknowledgement on reliable reception of a data packet. When the sender does not receive an acknowledgement within a suitable timeout interval, it retransmits after waiting for a randomly chosen time interval. Slotted Aloha was later developed to improve performance of the Aloha protocol. Similar to the slotted protocol in Section 3.3, in slotted Aloha, a packet can only begin transmission at a slot boundary.

Since the basic Aloha protocol does not use CSMA or any mechanisms to handle hidden terminals, the performance of Aloha is poor in many wireless environments. The CSMA approach was developed in an attempt to alleviate the shortcomings of Aloha. We have already discussed the notion of CSMA, as well as p-persistent CSMA protocols. In wired networks, CSMA was augmented with collision detection (CD) to obtain the CSMA/CD protocol. As discussed earlier, however, collision detection is difficult in wireless networks, leading to the use of collision avoidance mechanisms (CSMA/CA).

The analysis presented in Section 3.3 to obtain the limiting throughput of $\frac{1}{e}$ packets/slot with synchronized slots is based directly on the analysis of the slotted version of the Aloha protocol. Similarly, for Poisson traffic arrival, the unslotted version of the Aloha protocol has been shown to achieve throughput worse by a factor of 2, as compared to the slotted protocol, similar to the result we derived for the unsynchronized slots in Section 3.3.

IEEE 802.11 Distributed Coordination Function

IEEE 802.11 is an IEEE standard that specifies two component MAC protocols: Point Coordination Function (PCF) is a centralized protocol, whereas Distributed Coordination Function (DCF) is a distributed random access protocol. While DCF is widely used today, PCF has not found much acceptance to date. DCF incorporates many mechanisms that we have discussed in this chapter, including physical and virtual carrier sensing, backoff intervals, exponential backoff, per-packet acknowledgements, and retransmissions for reliability.

Each host maintains a Network Allocation Vector (NAV), which essentially remembers the duration of time that the host must stay silent. RTS, CTS, and data packets for a dialog announce the remaining time that is still needed to complete the dialog. When a host receives one of these packets, it updates its NAV to be the larger of (i) the current NAV value, and (ii) the remaining duration of the overheard dialog as specified in the packet.

DCF specifies several *inter-frame spacings* (IFS), which, as discussed below, are durations of time for which the channel is expected to remain idle prior to various operations performed by a host. In particular, if the channel has already been idle for a duration called *DCF InterFrame Spacing* (DIFS) when a packet arrives at the MAC layer at a host, then the packet is transmitted immediately. Idle channel status is determined using physical as

well as virtual carrier sensing. We will discuss DIFS soon again. If the channel is sensed as busy when a packet arrives, then the host chooses a backoff counter value in the range $[0, cw]$, where cw is called *contention window*. The initial value of cw is set to a certain minimum value, say CW_{min}. Note that cw in IEEE 802.11 is equivalent to b_{max} in our prior discussion. A transmission attempt is said to fail if a CTS is not received in response to an RTS, or if an Ack is not received in response to a data transmission. When a transmission attempt fails, the range $[0, cw]$ is doubled by setting new cw value to be is set to $2cw - 1$. This is how exponential backoff is incorporated in IEEE 802.11. On the other hand, if a transmission attempt is successful, cw is reset to CW_{min} for the next packet transmission. A packet is retransmitted only up to a specified maximum number of transmission attempts.

The use of RTS-CTS is optional in DCF. Thus, a host may optionally choose to transmit RTS before sending a data packet. Why make the RTS-CTS exchange optional? As we discussed previously, the use of RTS-CTS reduces collisions due to hidden terminals, thus, reducing the time wasted on such events. However, RTS-CTS packets themselves consume the channel resource. In some instances, the cost of sending RTS-CTS may not be justified by the reduction in the waste due to collisions (see Equation 3.3). For instance, if the data packet to be sent is small (say, the same size as the RTS), then there is no benefit in sending the RTS. Note that RTS itself may be lost due to collision as well. So the benefit of RTS comes from reducing collisions for the data packets, if the data packets are sufficiently large compared to the time required for RTS-CTS. In IEEE 802.11, RTS-CTS are exchanged prior to a data packet transmission only if the data packet size exceeds a certain threshold called the *RTS threshold*.

DCF also incorporates the notion of priority between different types of packets. Recall that a CTS packet follows successful reception of RTS, and an Ack follows a data packet. Consider host A sending RTS to host B. When host B receives the RTS, it requires B some time to begin sending the CTS packet. This delay includes the receive-to-transmit turnaround time for the wireless device. Now, consider some other host C that is able to physically carrier sense A's RTS transmission, but does not receive the RTS packet reliably due to low SINR for RTS reception at C. In this case, C will defer transmitting while A is transmitting due to physical carrier sensing, however, C may begin transmission soon after host A finishes sending RTS; since C does not reliably receive the RTS, virtual carrier sensing does not succeed in this case. The transmission from C may result in a collision with CTS reception at host A. To reduce the probability of such collisions, 802.11 DCF requires that before a host (such as C) may attempt transmission, or decrement the backoff counter, subsequent to a busy channel state, the channel must be idle for a duration called DIFS, as shown in Figure 3.16. The duration DIFS is long enough for host B to start sending CTS in response to the RTS. In fact, the 802.11 specification requires that the CTS be sent SIFS (Short InterFrame Spacing) duration after receipt of RTS, where SIFS is shorter than DIFS duration. DIFS being sufficiently longer than SIFS allows enough time for physical carrier sensing at host C to detect the CTS. Similarly, Ack is transmitted SIFS duration

after reception of data, reducing the possibility of collision with the Ack. Essentially, the use of SIFS versus DIFS ensures that an on-going dialog has a higher priority over the initiation of a new dialog. Of course, due to the presence of hidden terminals, and channel variations due to fading, this priority mechanism does not always work as intended. While the SIFS and DIFS help provide priority to ongoing dialogs, other priority mechanisms can be incorporated to provide priorities to certain traffic flows over other traffic flows [25]. We will discuss such priority differentiation mechanisms later.

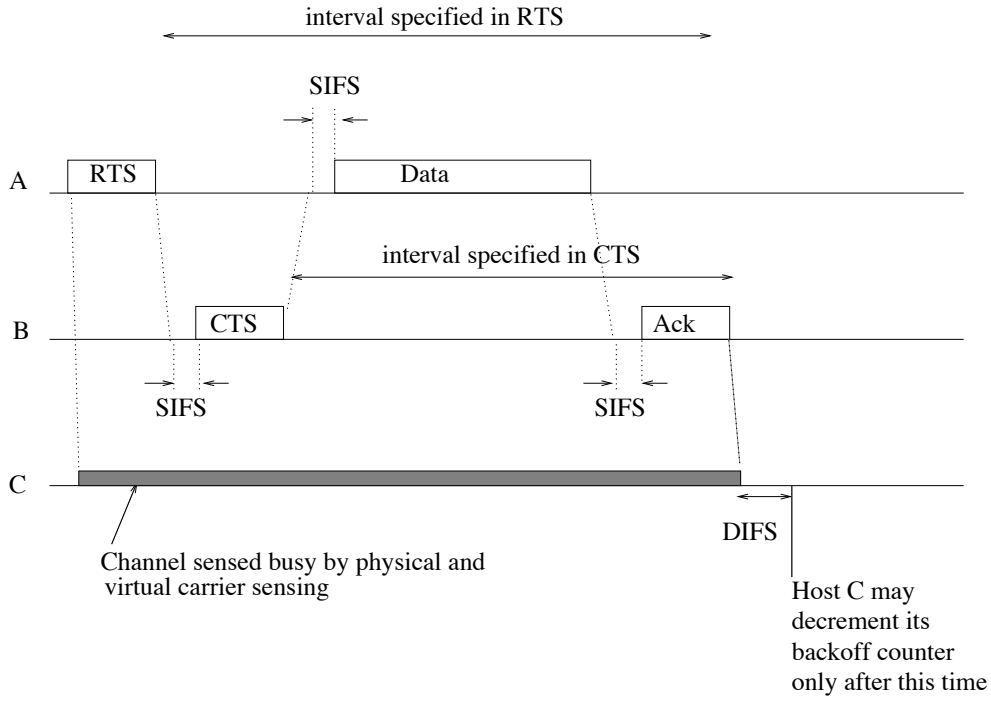


Figure 3.16 Illustration of interframe spacings in IEEE 802.11: The figure is not drawn to scale