Today: (1) Medium Access Intro, (2) Packet Radio, (3) 802.11 MAC

- Reading: Today – Abramson paper, Bianchi paper. Thu: Rappaport 7.11, 5.3.
- HW 7 is due Friday (Nov. 4) at 5pm. 6325-only assignment 2 is due today by midnight.

1 Medium Access Control

In this lecture, we describe some of the ways in which wireless networks can accommodate many users at the same time, even on the same channel. This is called medium access control (MAC). It is also called multiple access control, because we are designing a protocol for multiple users to share one channel (the “medium”). We have talked about multiple access throughout this semester already

- Frequency-division multiple access (FDMA): we have used distinct frequency channels to allow many users in a cellular communication system.
- Time-division multiple access (TDMA): users of the same frequency channel are assigned different “slots” in time.
- Space-division multiple access (SDMA): because of the cellular concept, we can assign users separated in space (in different cells or sectors) the same channel and still ensure that they do not interfere with each other.
- Code-division multiple access (CDMA): We can assign users distinct code channels, that is, have them use orthogonal pulse shapes that allow them, at the receiver, to be separated from any other user’s signal on a different code channel.

In this lecture, we will talk specifically about a variation of TDMA called “packet radio”. Packet radio allows multiple users to use a shared channel sometimes to send a “packet” of data. Compared to cellular TDMA, we are now talking about data communications systems which only sometimes need to send data. If we were to reserve each user a regular slot (like TDMA does), most of the time this slot would go unused. By having users send data
only when they need to, we hope to have a more efficient use of this shared channel.

This lecture is about control of medium access, i.e., MAC. What can control of the channel do for the efficiency of packet radio? What are some of the problems experienced in MAC protocols?

We will in this lecture:

- Learn how aloha and slotted aloha packet radio networks work, and their capacities.
- Learn what the "hidden terminal problem" is.
- Learn how 802.11 networks schedule traffic, and their capacities.

The first topic motivates the problem with packet radio, which then motivates why it can be more efficient to employ scheduling in a MAC protocol.

1. Some control: An 802.11 (WiFi) access point also exercises control over the users that communicate with it. But it has no control over where in space the next access point is that is also using the same channel. So problems ensue.

2. No control: Some wireless network protocols do not attempt to exercise control over when users offer traffic to the channel. In these cases, when two users transmit at the same time, their signals collide and can both be lost (i.e., not recoverable at the intended receiver). Actually, depending on the signal to interference ratio (SIR), a signal may be able to be received despite the fact that another interfering signal overlaps with it. Further, as we saw in the spread spectrum lecture, some modulation methods make it easier to recover the desired signal when it overlaps with other interfering signals.

2 Packet Radio

Let’s first discuss the reading [1]. I assign this reading because Dr. Abramson and the University of Hawai‘i addressed the design of a data communication system from scratch, like you are assigned to do for the semester design project. They did this 40 years ago, but we still use the analysis that they did in wireless networking systems today, because they did such a good job of presenting the fundamental tradeoffs in the design of packet radio systems.

The problem that Dr. Abramson had was that the University of Hawaii wanted to connect its campuses on different islands to a central server (called the “menelune” in the paper), which itself was connected to the ARPANET. However using telephone line
connections just was such an inefficient and expensive method to do so. Essentially, it was expensive to reserve a dedicated telephone channel for each computer terminal in the Hawaii system. It ended up being much cheaper to use radio communication, on a single channel with a higher bit rate (24 kbps!).

2.1 Aloha

This protocol is called “aloha” after the Hawaiian word. In this single channel, each terminal, when it had data, would just transmit a fixed duration (τ-length in time) “packet” of data. If two terminals happened to send packets that overlapped in time, you might have them collide and get neither packet, but that if this happened, both terminals would just retransmit later. There is a reverse channel on which the receiver acknowledged any packet from any sender that was (correctly) received. This positive acknowledgement is abbreviated as “ACK”.

Let the average number of data packets per second (from all senders) be \( r \). Again, \( \tau \) is the duration of a packet. Then the average total utilization of the channel is \( r\tau \). A utilization of \( r\tau = 1 \) would be perfect utilization. The result of [1] is that the utilization of the channel is, at most, \( 18.4\% \). At a higher \( r \) (and thus utilization), the number of collisions increases faster in a way that reduces the rate of successfully received packets. This maximum utilization is also referred to as the capacity of an aloha packet radio system.

Note that in this protocol, the receiver exercises very little control. It is the senders who decide when to transmit.

We also have a formula for the maximum number of active terminals (or users) sending packets to the server. This maximum is \( k_{\text{max}} \),

\[
k_{\text{max}} = \frac{1}{2e\lambda\tau}
\]

where \( e \) is the base of the natural logarithm, \( \lambda \) is the average rate at which each terminal sends packets to the server (in packets per second), and \( \tau \) is the packet duration. Note that we use “active terminals” to describe terminals that send packets right now at the average rate of \( \lambda \) per second. There might be other “inactive terminals” which are not sending packets at all at this time.

2.2 Slotted Aloha

Abramson also studied a variation of the aloha system in which terminals were synchronized and could agree on non-overlapping “slots”, that is, times during which packets could be transmitted [2]. That is, rather than transmitting at just \( \text{any} \) time, a terminal would wait for the next slot’s start time. By having all packets
transmitted starting at the slot start time, there will be fewer collisions. Abramson showed that this simple modification allows the maximum utilization to double, to 36.8%. The maximum number of active terminals becomes $k_{max} = \frac{1}{e\lambda \tau}$, also double the result for regular aloha.

The only problem is that, now, each terminal must be synchronized to a clock. This may be easy or difficult depending on the synchronization requirements.

Slotted aloha is slightly more controlled, in the sense that each terminal must be synced to the same clock, and must start transmitting only at the start of a slot.

**Example:** How many active users could you support?

How many active users could you support on a 1 MHz channel using a packet radio system? Take, in this example, a QPSK system with SRRC pulse shaping with $\alpha = 0.5$. For this example, use the same packet rate and duration used in the original aloha system – 704 bits per packet, and a packet generation rate of 1 per minute. Calculate both for aloha and slotted aloha.

**Solution:** The bit rate is 1 MHz $(\log_2 4)/(1 + 0.5) = 1.333$ MHz. For 704 bits, this would take a duration of 0.53 ms. Using aloha,

$$k_{max} = \frac{1}{2e\lambda \tau} = \frac{60}{2e(0.0021)} \approx 20.9 \times 10^3$$

For slotted aloha, the max number of users would double to $41.8 \times 10^3$.

## 3 CSMA-CA

### 3.1 Carrier Sensing

One more simple idea in packet radio is *carrier sensing*. Carrier sensing refers to a transceiver’s use of its receiver to sense whether or not there is currently another packet being transmitted on the channel, *i.e.*, “if the channel is busy”, before transmitting its own packet. If it detects transmission, it will wait until the transmission is completed, until starting its own transmission. This is called “carrier sense multiple access” (CSMA).

A minor modification of CSMA is called CSMA-collision avoidance, or CSMA-CA. This means that, when a CSMA transceiver detects the channel to be busy, rather than just waiting for the other transmission to be completed, it also waits a random time interval before trying to retransmit.
3.2 Hidden Terminal Problem

What is a problem of relying on one terminal to know when the channel is busy? Since signal power is spatially varying, one terminal may be transmitting to the server successfully, even while a second terminal at a third location is not able to hear that the channel is busy, as shown in Figure 1.

![Figure 1](TX_RX_CR.png)

Figure 1: Terminal TX is transmitting to server RX. Terminal CR cannot hear TX’s signal, so even if it uses carrier sensing, when it has data to send, it would transmit and cause interference at terminal CR.

The hidden terminal problem is ubiquitous in packet radio networks. This makes carrier sensing systems less useful than you would otherwise imagine.

3.3 802.11 DCF

Our 802.11 networks operate using a basic CSMA-CA protocol which is named the “distributed control function” (DCF). They use CSMA-CA, and when a collision occurs, for collision avoidance, a terminal waits a random amount of time in a procedure called exponential backoff.

This material supplements and draws heavily from [3].

The job of a terminal’s transmitter is to send its packets. When it has a packet to send, it:

1. Listens to the channel for a duration of DIFS (a constant time window called the distributed interframe space).

2. After the channel is sensed to be free for a duration of DIFS (by the receiver), the transmitter picks a random backoff delay $X$, which is an integer, picked as follows. $X$ is chosen randomly and uniformly from the range 0 to $2^i CW_{min} - 1$ for some integer $i$ called the backoff stage and some minimum window length $CW_{min}$. This backoff stage $i$ is initially set to zero. So initially, when a collision occurs, the transmitter picks this random integer $X$ between 0 and $CW_{min} - 1$.

3. If $X = 0$, the terminal immediately transmits.
4. If not, it continues sensing. After $\sigma$ period of time (called the slot time) with the channel unoccupied, the receiver decrements $X$ by one. Note that if the receiver hears another terminal transmit, it waits a period DIFS after the end of the transmission, and then waits $\sigma$ and decrements $X$ by one.

5. The terminal goes back to step 3.

After the transmitter sends its packet, its receiver checks for the ACK from the access point within ACK_Timeout (another constant period of time):

1. *If it does not receive the ACK within the specified time window*, it assumes that the packet collided and was lost. In this case, it increments the backoff stage $i$ (up to a maximum of $m$), and starts the procedure to transmit the packet again.

2. *If it does receive the ACK*, it resets the backoff stage $i$ to zero, and then starts the procedure to transmit the next packet (assuming there is another one).

Because there are so many acronyms, here is a table of the most important:

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>positive acknowledgement</td>
</tr>
<tr>
<td>CSMA/CA</td>
<td>carrier sense multiple access with collision avoidance</td>
</tr>
<tr>
<td>DCF</td>
<td>distributed coordination function: the algorithm in 802.11 which is the main subject of this paper</td>
</tr>
<tr>
<td>DIFS</td>
<td>distributed interframe space: how long a terminal measures the channel before determining it is idle</td>
</tr>
<tr>
<td>MAC</td>
<td>medium access control</td>
</tr>
<tr>
<td>SIFS</td>
<td>short inter-frame space: delay between end of reception and transmission of ACK</td>
</tr>
</tbody>
</table>

Notation:

- $\sigma$: slot time size (time needed for any terminal to detect a transmission)
- $CW_{\text{min}}$: minimum size of the contention window
- $n$: number of “contending terminals”, i.e., those offering packet traffic to the network.
- $i$: backoff stage, in the range $\{0, \ldots, m\}$
- $m$: maximum backoff stage ($CW_{\text{max}} = 2^m W$)
3.4 In-Class DCF Demo

This activity recreates a few milliseconds in the life of a 802.11 network running the DCF. For our exercise, let:

- $W = 4$, $m = 2$. Thus the maximum contention window is length 16.
- Let $\sigma = 1$, DIFS = 3, SIFS = 1.
- packet duration $P = 20$, ACK duration = 3.
- $n = 3$: terminals A, B, and C. All terminals can hear the access point. First assume that all terminals can hear each other, then assume that terminal A cannot hear terminal C, and vice versa.
- $ACK_{Timeout} = 24$.

Each person will “act out” a terminal TX or RX, or access point TX or RX. Other “actors” include the random number generator (the person who selects random numbers from 0 to $2^{iCW_{min}} - 1$), and the “time counter” who moves time to the next multiple of $\sigma$ when the actors are ready.

The exercise starts by the random number generator presenting a random number in $\{0, \ldots, CW_{min}\}$ to each of the terminal transmitters to use as their backoff counter. Terminals are all in backoff stage 0 at the start.

Each terminal comprises a transmitter and receiver. The transmitter’s job is to decrement the backoff counter whenever the receiver allows it to do so, and then transmit a packet whenever the backoff counter hits zero. The receiver’s job is to sense the channel (and thus stop the backoff counter from whenever a packet is transmitted until DIFS after the ACK is finished). After transmitting a packet, or a collision, the transmitter requests a random number (ask the random number generator to pick a number out of a hat according to the backoff stage). The receiver’s job is also to make sure that an ACK is received within $ACK_{Timeout}$ of the start of the transmitter sending a packet. If it is not, increment the backoff stage (up to the maximum stage $m$) and tell the transmitter to request a new random number for the backoff counter.

The access point receiver’s job is to listen for each packet; then SIFS after the end of a packet, the access point transmitter sends an ACK. Unless, of course, two packets were transmitted at the same time, in which case, neither packet is received and an ACK is not transmitted.

Whenever your terminal or access point transmits a packet, the transmitter actor will hold up card letting the other players know that he or she is occupying the medium. For the hidden terminal simulation, terminal A should ignore terminal C, and vice versa.
Important questions:

1. How efficient is the 802.11 DCF when there is no hidden terminal problem? The paper eventually shows that a utilization rate of just above 80% is possible in this case.

2. How significant is the hidden node problem in the 802.11 DCF?

3.5 RTS/CTS

To help deal with the hidden terminal problem, a MAC protocol named request-to-send/clear-to-send or RTS/CTS is used. In short, a terminal doesn’t just send its data — instead, it sends a very short packet with its id and an indication that it wants to send a data packet of a given duration. The destination schedules a reserved time for that transmission, and other terminals in range of the destination terminal must not transmit during that reserved time. After successful reception, the destination also sends a final ACK to acknowledge receipt.

References

