Medium Access Sublayer

So far: We have discussed the Data Link Layer's functionality and some protocols related to point-to-point communication.

A large class of networks is built on top of **broadcast channels**: a number of stations that share the same "wire." If one station sends, all the others get to hear it.

What's the problem: if you're sharing a channel, then two stations may decide to start frame transmission at the same time $\Rightarrow$ **frame collision**, which means rubbish on the wire.

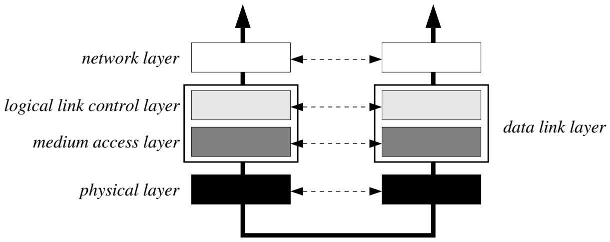
Solution: Allocate the channel to one of the competing stations.

Problem: You'll have to use that same channel to figure out the competition and the allocation.

Medium Access Layer II

Three strategies for channel allocation:

- Exercise **no control** at all: simply let a station try to use the channel, and do something when a collision happens. Applied in **contention systems**.
- Employ a **round-robin** technique: each station in turn is allowed to use the channel. Applied in **token-based** systems – the station that has the token may use the channel.
- Let a station place a **reservation** for the channel. Used in **slotted** systems. The problem is how to make a reservation.



Contention systems: ALOHA

Principal: if you want to send a frame, just do it. If a collision occurs, finish your current transmission and retry later.

Note: it's pretty hard to do any worse than this, but efficiency is not really that bad:

- Let S be the average number of new frames submitted during a *frame time* T_{frame} (time needed to transmit a full frame). Poisson distributed.
- Let G be the number of old and new frame submissions during a frame time ($G \geq S$). Also Poisson distributed:

$$\mathcal{P}[k \text{ frames submitted}] = \frac{G^k \cdot e^{-G}}{k!}$$

- Let P_0 be the probability that frame does not suffer from collision $\Rightarrow S = G \cdot P_0$.

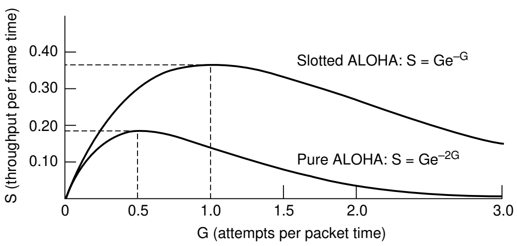
Contention systems: ALOHA II

- When a frame is being sent, it shouldn't bump into its predecessor, and shouldn't be bumped into by a successor. The predecessor requires T_{frame} time, and our frame as well.
- Probability that a frame will not be damaged is

$$P_0 = e^{-2G}$$

- $S = G \cdot e^{-2G}$

We can improve a bit by removing some randomness. in **Slotted ALOHA** frame transmission can only start at fixed times:



CSMA Protocols

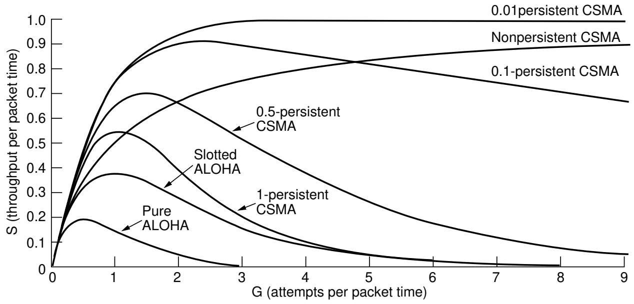
Carrier Sense Multiple Access Protocols do better than ALOHA: you monitor the channel before and/or during transmission.

1-persistent: Listen whether the channel is free before transmitting. If busy, wait until it becomes free and then immediately start your transmission.

Nonpersistent: Less greedy – when the channel is busy, wait a random period of time before trying again. If you wait too long, the channel utilization drops.

p-Persistent: Used with slotted systems. If you find the channel idle during the current slot, you transmit with probability p, and defer until next slot with probability $1-p$. $p=1$ is not really good, $p=0$ makes you *really* polite.

Protocol Comparison

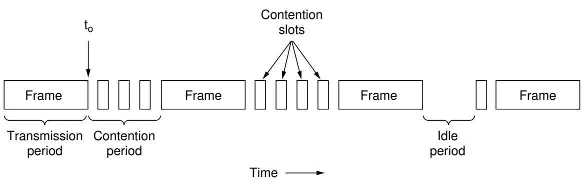


Question: What are we actually displaying here? Should the conclusion be that p-persistent protocols are really good with $p \approx 0$?

CSMA with Collision Detection

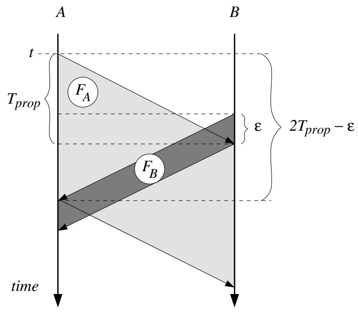
Improvement: sense the channel, but immediately stop transmission when you detect a collision. **Ethernet** works like this.

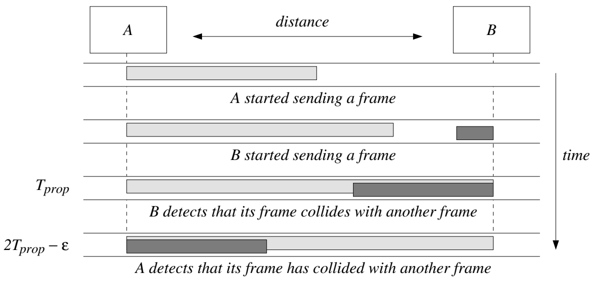
1. Listen to see whether the channel is free. Transmission is delayed until the channel is no longer used.
2. During transmission, keep listening in order to detect a collision. If a collision occurs, transmission immediately stops.
3. If a collision occured, wait a random period of time, and proceed with the first step again.



CSMA with Collision Detection II

Problem: how big should the contention period be?

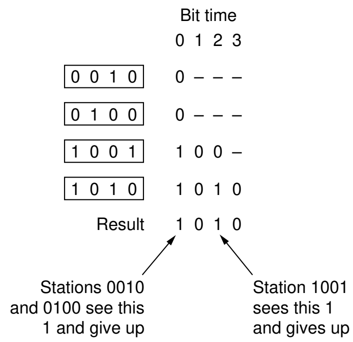




Collision Free Protocols

Bit-map: the contention period contains N slots. If station k wants to transmit a frame, it transmits a 1 during the k^{th} slot. The highest-numbered station goes first.

Binary Countdown: In the contention period a total of $\log_2 N$ bits can be transmitted. Each station transmits its number (bit by bit), and stops as soon as it detects a higher-numbered contender:



Limited-Contention Protocols

Principal: Contention systems are good when there's not much going on – a station can immediately transmit a frame. We do some repairing when things go wrong.

Collision-free systems are good when there's generally a lot of traffic – a station first has to get the channel explicitly before frame transmission. We do a lot of work avoiding collisions.

What we really want is the contention strategy during light loads, and collision-free strategy during rush hours.

A solution: Dynamically regulate the number of competing stations during a contention period. If there's a collision during the k^{th} slot, divide the contenders into two groups.

The first group gets to try it again during the next slot $(k+1)$. If no collisions occur then, the second group gets a try during the slot after that $(k+2)$. Otherwise, the first group is split up again.

Note: if there's not much traffic, the first station will be immediately allowed to transmit a frame. With a lot of traffic, the strategy reduces to the bit-map protocol.

Wavelength Division

Principal: if you have a lot of bandwidth, just divide the channel into sub-channels, and dynamically allocate the sub-channels. Used in fiber optics.

- Each station gets two channels: one control channel for handling incoming requests, one for the actual data transfer.
- Each channel repeatedly carriers a fixed series of slots. The data channel contains a status slot carrying info on free slots in its control channel.
- In order to contact a station A, you first read its status slot from the data channel to see which control slot is unused. You then put a transmission request in a free control slot.
- If the transmission request is accepted, you can send data on your own data channel that will be picked up by A. You put the data in specific slot, and tell A which slot that is.

Note: there's still a lot of competition. If two senders say that there's data in slot #4 for station A, one of the slots will not be read – A can only read from one channel at a time.

Wireless Networks

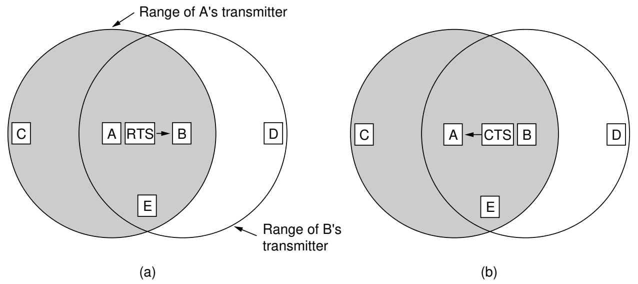
Basic idea: Often, there are a number of **base stations** connected through guided media. A base station can communicate with a mobile computer. The mobile computers use radio/infrared signals for communication.

Problem: There can be subtle interference:

Issue (a): How can C be prevented from trying to transmit something to B? In that case it will ruin any receipt by B (hidden station problem).

Issue (b): How can we tell C that it is allowed to transmit to D, because this will not interfere with the communication from B to A? (exposed station problem).

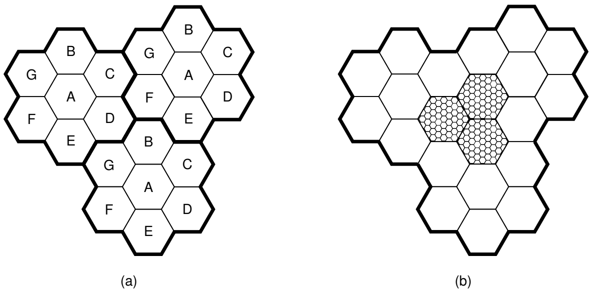
Multiple Access with Collision Avoidance



- *A* first sends a Request To Transmit (RTS).
- *B* answers with a Clear To Transmit (CTS).
- *C* hears only RTS and can freely transmit, knowing it will not interfere with *A*'s transmission.
- *D* hears only the CTS and keeps still for otherwise it would interfere with *B*'s reception.

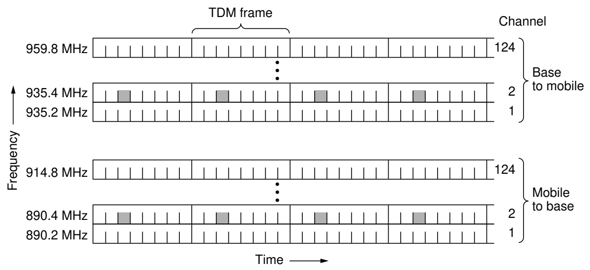
Digital Wireless Transmission

GSM: Global System for Mobile communications, is used in Europe and is a full-blown digital cellular radio transmission system. A cell has one or more base stations, and uses a unique set of frequencies:



GSM

GSM uses 124 **downlink channels**, and 124 **uplink channels** per cell, each channel multiplexed through TDM:



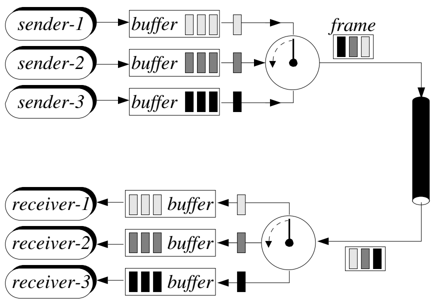
Note: this gives $8 \times 124 = 992$ full duplex channels. A lot of them are not used to avoid interference with neighboring cells. There are also separate channels for:

- broadcasting cell info (so that a mobile station can see whether it has changed cells).
- cell maintenance (the base station has to know who's in its cell).
- call setup (incoming, and outgoing).

Channel Multiplexing

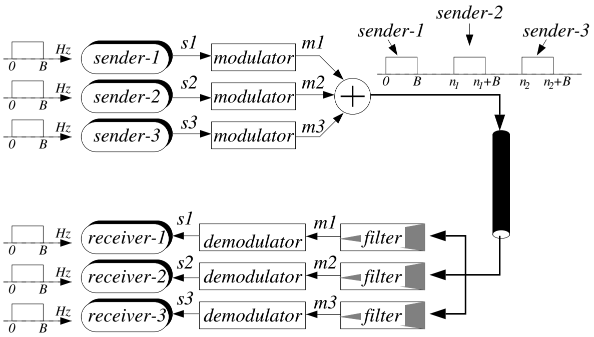
Basic idea: Really simple – divide the available bandwidth into a number of subchannels, either by scanning the different input lines (time division), or by dividing the frequency range (frequency division).

Time Division Multiplexing:



Channel Multiplexing II

Frequency Division Multiplexing:



Code Division Multiple Access

What's new? CDMA allows transmissions to be interleaved, but avoids interference. Note that this means inherently no message collision.

Principal: Assign a **chip sequence** to a station, which is just an m-bit code. Make sure that all chip sequences are pairwise orthogonal:

- rewrite a binary 0 as -1, and a binary 1 as $+1$.
- for every two chip sequences **S** and **T**: $\frac{1}{m}\sum_{i=1}^{m} S_i T_i = 0$.
- send a 1 bit as your chip sequence (**S**), and a 0 bit as the inverse ($\bar{\mathbf{S}}$).
- just transmit your bits when a new bit time slot starts $\Rightarrow$ the (possibly inversed) chip sequences are just added.
- getting the original value means taking the inner product of the orginal chip sequence with the signal sent.

Code Division Multiple Access II

A: 0 0 0 1 1 0 1 1	A: (–1 –1 –1 +1 +1 –1 +1 +1)
B: 0 0 1 0 1 1 1 0	B: (–1 –1 +1 –1 +1 +1 +1 –1)
C: 0 1 0 1 1 1 0 0	C: (–1 +1 –1 +1 +1 +1 –1 –1)
D: 0 1 0 0 0 0 1 0	D: (–1 +1 –1 –1 –1 –1 +1 –1)
(a)	(b)

Six examples:

– – 1 –	**C**	S_1 = (–1 +1 –1 +1 +1 +1 –1 –1)
– 1 1 –	**B** + **C**	S_2 = (–2 0 0 0 +2 +2 0 –2)
1 0 – –	**A** + $\bar{\mathbf{B}}$	S_3 = (0 0 –2 +2 0 –2 0 +2)
1 0 1 –	**A** + $\bar{\mathbf{B}}$ + **C**	S_4 = (–1 +1 –3 +3 +1 –1 –1 +1)
1 1 1 1	**A** + **B** + **C** + **D**	S_5 = (–4 0 –2 0 +2 0 +2 –2)
1 1 0 1	**A** + **B** + $\bar{\mathbf{C}}$ + **D**	S_6 = (–2 –2 0 –2 0 –2 +4 0)

(c)

S_1 • C = (1 +1 +1 +1 +1 +1 +1 +1)/8 = 1
S_2 • C = (2 +0 +0 +0 +2 +2 +0 +2)/8 = 1
S_3 • C = (0 +0 +2 +2 +0 –2 +0 –2)/8 = 0
S_4 • C = (1 +1 +3 +3 +1 –1 +1 –1)/8 = 1
S_5 • C = (4 +0 +2 +0 +2 +0 –2 +2)/8 = 1
S_6 • C = (2 –2 +0 –2 +0 –2 –4 +0)/8 = –1

(d)

Question: doesn't this look a lot like linear algebra?