Lecture 7 - multiple access protocols

Multiple Access Problem:

Broadcast channel: multiple nodes transmit and receive on the same shared medium
- Coaxial cable segment
- WiFi radio channel
- Satellite uplink used by many ground stations

Problem: if 2 nodes transmit simultaneously, signals interfere
→ collision = both frames are garbled and lost

Multiple access (MAC) protocol is a distributed algorithm that coordinates who transmits when

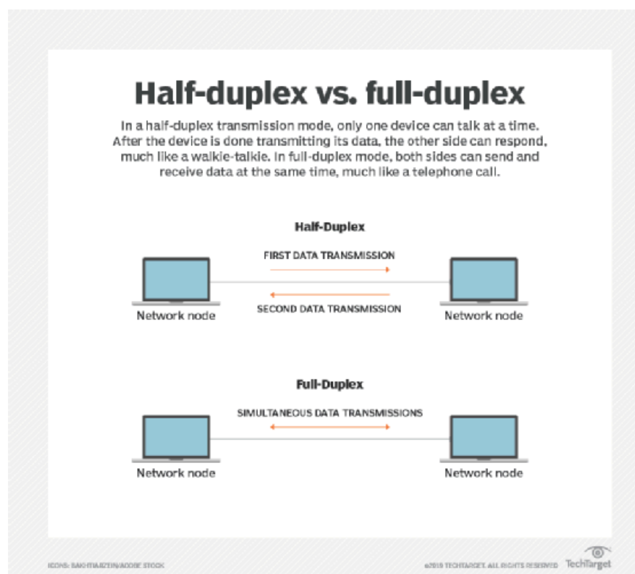
Channel Directionality
Three modes of communication on a link

Mode	Description	Examples
Simplex	One direction only	Broadcast radio, cable TV
Half-duplex	Both directions, **not simultaneously**	Classic Ethernet, WiFi, walkie-talkie
Full-duplex	Both directions **simultaneously**	Telephone, switched Ethernet

Key point → the MAC problem is a half duplex problem

→ on a shared half duplex channel, only one node can transmit at a time → hence the need for coordination

On a full duplex, point to point link each direction is independent: no collisions, no MAC protocol



What we need from a MAC protocol
ideal MAC protocol for channel of rate R:
1. When one node has data, it gets full rate R
2. When M nodes have data, each gets roughly R/M on average
3. Fully decentralized (no central coordinator, no single point of failure)
4. Simple to implement

3 families of solutions:

1. Channel partitioning → divide channel and give each node a piece
2. Taking turns → nodes take turns explicitly
3. Random access → transmit freely and detect/ recover from collisions

TDMA→ time division multiple access

Idea: divide time into frames → each of N nodes gets one fixed slot per frame

N1	N2	N3	N4	N1	N2	...

Properties:

- Each node gets rate R/N → rate/number of nodes
- No collisions
- Efficiently problem → if only one node has data, channel is 1/N utilized
- Idle slots are wasted → no other node can use them

FDMA and CDMA

FDMA → frequency division multiple access

- Divide bandwidth W into N frequency bands and each node gets one band permanently
- Same efficiency problem as TDMA→ wasted bands are idle

CDMA→ code division multiple access

- Each node uses an orthogonal spreading code; all nodes share full bandwidth simultaneously
- Receiver separates signals using the code

Channel partitioning works well under heady, symmetric load, most network traffic is bursty → partitioning wastes capacity

Taking Turns: Polling → instead of dividing by time, frequency, code, it takes turns

Idea: a coordinator node pools each node in round robin order, granting permission to transmit

Properties:

- No collisions
- Master can enforce fairness
- Polling overhead: one pool message per node per round even if idle
- Master is a single point of failure
- Latency even when most nodes have nothing to send

Used in bluetooth

Taking Turns: Token Passing

Idea: a special token frame circulates around a logical ring. A node may transmit only while holding the token and then passes it on

Node a → node b → node c → node d…

Advantage:
- No collisions
- Predictable worst case latency

Problems:
- Token loss requires recovery protocol
- One failed node can break the ring
- Overheard when few nodes have data

<u>Random Access: the ALOHA idea</u>
Key insight: don't coordinate in advance → just transmit → handles collisions after the fact

Pure aloha rule:
- If you have a frame, transmit it immediately
- If a collision occurs, wait a random time and retransmit

<u>Pure ALOHA: collision Window</u>
When can 2 frames collide?
- Let frame transmission time=T (time to send one frame)
- A frame set at time t occupies the channel during [t,t+T] → time to time + frame
- It can collide with any frame whose transmission begins during [t-T, t+T] → current time - last transmit frame
    - Collision window = 2T

Intuition: a late starting frame can collide at the tail of our frame, an early starting frames tail can collide with our start

→ pure ALOHA has the largest possible collision window

<u>Pure ALOHA: throughput analysis</u>
G= offered load
Assume poisson arrivals. A transmission succeeds if no other node transmits during the collision window 2T:

$$P(\text{success}) = e^{-2G}$$

throughput(successful transmissions per frame time):

$$S=Ge^{-2G}$$

Maximize: differentiate and set to 0 → G=½

$$S_{\max} = \frac{1}{2} \cdot e^{-1} = \frac{1}{2e} \approx 0.184$$

Pure ALOHA maximum efficiency: $\approx 18\%$

<u>Slotted ALOHA: having the collision window</u>
Key idea: synchronize all nodes to a common slot clock

Rule: transmissions begin only at slot boundaries → if collision: retransmit in a future slot with probability p

Effect on collision window:

- Two frames can only collide if they begin in the same slot
- Collusion window shrinks from 2T to T → having a slotted aloha halves the collision window

Throughput analysis with offered load G attempts per slot:

$P(\text{success})=e^{-G} \rightarrow S=Ge^{-G}$

Maximize G= 1→ $S_{max}=e^{-1}=1/e = 0.368$

Slotted ALOHA: what happens at the peak load

At $G = 1$ (the efficiency-maximizing load):

Slot outcome	Probability	Note
Successful frame	$1/e \approx 37\%$	Exactly one transmission
Empty slot	$1/e \approx 37\%$	No transmissions
Collision	$1 - 2/e \approx 26\%$	Two or more transmissions

Practical issue: requires slot synchronization across all nodes→ non trivial in a distributed system → can do better with carrier sensing

ALOHA summary and motivation for CSMA

Protocol	Max Efficiency	Collision Window	Sync?
Pure ALOHA	$\approx 18\%$	$2T$	No
Slotted ALOHA	$\approx 37\%$	T	Yes
CSMA (next)	Much better	$\tau \ll T$	No

ALOHA weakness: nodes transmit without knowing what others are doing → before transmitting, we can listen to see if the channel is in use

CSMA - Carrier Sense Multiple Access

→ before transmitting, listen to the channel

Channel idle → transit

Channel busy → defer

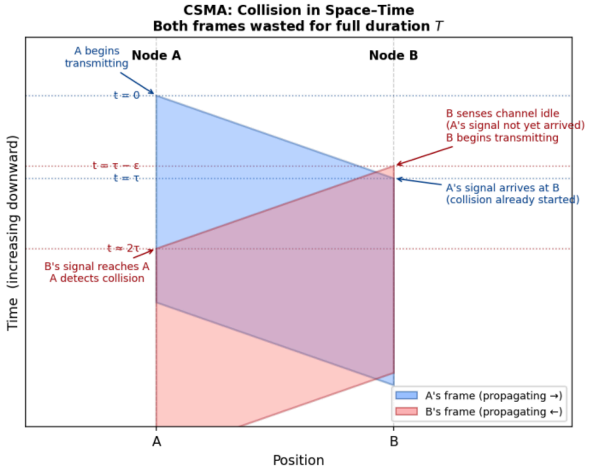
→ collisions only occur when 2 nodes both sense idle within one propagation delay t of each other , efficiency is much higher than ALOHA

CSMA variants:

- 1-persistent: if idle, transmit immediately; if busy, wait then transmit immediately when idle
    - Problem when all waiting nodes transmit at once when channel clears → guaranteed collision
- Non-persistent: is busy wait a random time then sense again
- P-persistent: if idle, transmit with probability p, defer with probability 1-p

CSMA: collisions still happen

Problem → 2 nodes can both sense the channel idle and start transmitting simultaneously



CSMA/CD- collision detection

Collision detection (CD): while transmitting, listen to the channel simultaneously

→ if what you hear is not what you're sensing → collision is occurring → abort immediately

Key improvement: instead of wasting T on a collided frame, abort after at most 2t since t < T

→ this is the basis of classic ethernet

CSMA/CD Algorithm

1. if channel idle → transmit frame
2. If channel busy → wait until idle then transmit
3. While transmitting, monitor the channel
4. If collision detected
    a. Send jam signal to ensure all nodes detect the collision
    b. Abort transmission
    c. Enter binary exponential backoff
5. After backoff delay, return to step 1

Jam signal: a short burst that reinforces the collision signal so every node on the signal is certain a collision occured

Binary Exponential Backoff

After the k-th collision on a frame, choose a random wait from:

$$\{0,\ 1,\ 2,\ \ldots,\ 2^{\min(k,10)}-1\} \quad \text{slot times}$$

Collision #	Backoff range	Max wait
1	$\{0,1\}$	2 slots
2	$\{0,1,2,3\}$	4 slots
3	$\{0,\ldots,7\}$	8 slots
$\vdots$	$\vdots$	$\vdots$
10+	$\{0,\ldots,1023\}$	1024 slots
16	Give up; report error to upper layer	

Minimum Frame Size: the Constraint

Problem: a node must still be transmitting when it hears a collision → is a node finishes transmitting before the collision signal returns, it has no way to know a collision occurred

Required condition: Transmission time T>= round trip propagation delay 2T

$$T = \frac{L}{R} \geq 2\tau \qquad \Rightarrow \qquad \boxed{L \geq 2\tau R}$$

Classic Ethernet example

$R = 10$ Mb/s, max segment $= 2500$ m

$$\tau \approx 25\ \mu\text{s} \quad (\text{at } 2 \times 10^8\ \text{m/s})$$

$$L_{\min} = 2 \times 25\ \mu\text{s} \times 10^7\ \text{bps} = 500\ \text{bits} = \mathbf{64}\ \text{bytes}$$

CSMA/CD Efficiency and the end of shared ethernet

$$\text{Efficiency} = \frac{1}{1+5a}$$

→ as a→ 0 = efficiency → 1

End of shared ethernet:

Modern ethernet uses switches with full duplex point to point links

- Eachc device connects to exactly one switch port
- Only 2 nodes share a link: a device and the switch
- No contention → **collisions are impossible**

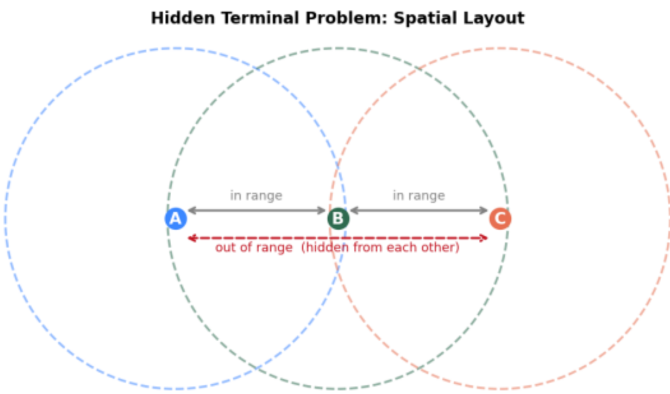
Why collision detection fails in wireless

CSMA/ CD requires: transmitting node can simultaneously hear what's on the channel

In wireless it fails because:

1. Self interference → the transmitters own signal is order of magnitude stronger than any received signal → radio cannot hear others while transmitting

2. Hidden terminal problem



MAC Protocols: summary

Family	Protocol	Used In	Key Mechanism
Partitioning	TDMA	GSM, satellite	Fixed time slots
Partitioning	FDMA	FM radio, cable	Fixed frequency bands
Taking turns	Token Ring	Legacy LANs	Circulating token
Random access	Pure ALOHA	Packet radio	Transmit freely
Random access	Slotted ALOHA	Satellite, LTE	Slot-synchronized
Random access	CSMA/CD	Classic Ethernet	Sense + detect + backoff
Random access	CSMA/CA	WiFi (802.11)	Sense + avoid + ACK

Key Numbers to remember

Result	Value
Pure ALOHA max efficiency	$S_{\max} = 1/(2e) \approx 18\%$ at $G = 1/2$
Slotted ALOHA max efficiency	$S_{\max} = 1/e \approx 37\%$ at $G = 1$
Slotted vs. pure	Slotted is exactly $2\times$ better
Ethernet min frame	64 bytes (from $L \geq 2\tau R$)
Classic Ethernet round-trip delay	$2\tau \approx 50\ \mu$s (2500m segment)

The big picture:

- Channel partitioning: wastes capacity under bursty load
- ALOHA: simple but low efficiency
- CSMA + collision detection/avoidance: near-ideal under typical load
- Modern Ethernet: switched, full-duplex — MAC problem mostly goes away