Announcements:

- Lab 3 is due now
- Lab 4 and Assignment 4 are posted
- Assignment 4 is due next Thursday
- Lab 4 is due the following Tuesday
Link Layer

- 1 Error detection and correction
- 2 Link-layer Addressing
- 3 Multiple access protocols
- 4 Ethernet
Multiple Access Links and Protocols

Two types of “links”:

- **point-to-point**
  - PPP for dial-up access
  - point-to-point link between Ethernet switch and host

- **broadcast** (shared wire or medium)
  - Ethernet
  - 802.11 wireless LAN
Multiple Access protocols

**need for sharing of medium/channel**
- single channel
  - needs be used by all nodes
- interference/collision
  - two or more simultaneous transmissions lead to collided signals

**multiple access protocol**
- allows multiple, concurrent access
  - algorithm that nodes use to share channel, i.e., determines when a node can transmit
- no coordination, no out-of-band channel
  - agreeing about channel sharing must use channel itself!
Ideal Multiple Access Protocol

Broadcast channel of rate $R$ bps
1. when one node wants to transmit, it can send at rate $R$.
2. when $M$ nodes want to transmit, each can send at average rate $R/M$.
3. fully decentralized:
   - no special node to coordinate transmissions
   - no synchronization of clocks, slots
4. simple
MAC Protocols: a taxonomy

Three broad classes:

- **Channel Partitioning**
  - divide channel into smaller “pieces” (time slots, frequency, code)
  - allocate piece to node for exclusive use

- “Taking turns”
  - nodes take turns, but nodes with more to send can take longer turns

- **Random Access**
  - channel not divided, allow collisions
  - need to know how to “recover” from collisions
Channel Partitioning MAC protocols: TDMA

TDMA: time division multiple access
- access to channel in "rounds"
- each station gets fixed length slot (length = pkt trans time) in each round
- unused slots go idle

- E.g.: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle
**Channel Partitioning MAC protocols: FDMA**

**FDMA: frequency division multiple access**
- channel spectrum divided into frequency bands
- each station assigned fixed frequency band
- unused transmission time in frequency bands go idle
- E.g.: 6-station LAN, 1,3,4 have frames, but frequency bands 2,5,6 idle

![Diagram of frequency division multiple access (FDMA)](image)
“Taking Turns” MAC protocols

Polling:

- master node “invites” slave nodes to transmit in turn

- concerns:
  - polling overhead
  - latency
  - single point of failure (master)
“Taking Turns” MAC protocols

Token passing:
- control token passed from one node to next sequentially.
- If token, then send message

Concerns:
- token overhead
- latency
- single point of failure (token)
Random Access Protocols

- distributed
  - unlike TDMA or FDMA
  - no coordination among nodes

- one node at a time
  - when a node transmits, it does so at full data rate R.

- collisions can occur
  - two or more transmitting nodes ➜ “collision”,

random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions

examples of random access MAC protocols:
  - ALOHA, CSMA/CD, CSMA/CA

(CSMA: Carrier Sense Multiple Access; CD: Collision Detection; CA: Collision Avoidance)
Slotted ALOHA

Assumptions:
- all frames same size
- time divided into slots
  - slot = time to transmit 1 frame
- start transmit at beginning of slot only
- nodes are synchronized
- if multiple nodes transmit in slot, all can detect collision

Operation:
- when node gets a fresh frame, transmits in next slot
  - if no collision: node can send new frame in next slot
  - if collision: node retransmits frame in each subsequent slot with prob. $p$ until success
Slotted ALOHA

Pros
- single active node can continuously transmit at full rate of channel
- highly decentralized: only slots in nodes need to be in sync
- simple

Cons
- collisions, wasting slots
- idle slots, wasting slots
- clock synchronization

Legend:
- C = collision
- E = empty/idle
- S = success
**Slotted Aloha efficiency**

**Efficiency**: long-run fraction of successful slots (many nodes, all with many frames to send)

- **suppose**: $N$ nodes with many frames to send, each transmits in slot with probability $p$

- prob that a given node has success in a slot?
  \[ p(1-p)^{N-1} \]

- prob that any node has a success?
  \[ Np(1-p)^{N-1} \]

- Efficiency = $Np(1-p)^{N-1}$
Slotted Aloha efficiency

- Efficiency = $Np(1-p)^{N-1}$

- Max efficiency: find $p^*$ that maximizes $Np(1-p)^{N-1}$
  
  $p^* = 1/N$

  $\text{Max Eff} = (1-1/N)^{N-1}$

- When $N$ increases to $\infty$, $\text{max eff} = (1-1/N)^{N-1}$ goes to $1/e = 0.37$

**At best:** channel used for 37% useful transmission time!
Pure (unslotted) ALOHA

- unslotted Aloha: simpler, no synchronization
- when frame first arrives
  - transmit immediately
- collision probability increases:
  - frame sent at $t_0$ collides with other frames sent in $[t_0-1, t_0+1]$
Pure Aloha efficiency

P(success by given node) = P(node transmits) ×

P(no other node transmits in \([t_0-1, t_0]\)) ×
P(no other node transmits in \([t_0, t_0+1]\])

= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1}

= p \cdot (1-p)^{2(N-1)}

... choosing optimum p and then letting N \to infinity ...

= \frac{1}{2e} = .18

(to be derived in homework problem)

even worse than slotted Aloha!
CSMA (Carrier Sense Multiple Access)

**CSMA:** listen before transmit:
- If channel sensed idle: transmit entire frame
- If channel sensed busy, defer transmission

- Human analogy: don’t interrupt others!
CSMA collisions

collisions can still occur:
Node B’s transmission collides with Node D’s transmission

collision is a waste!
A cannot hear B’s signal
C cannot hear D’s signal

note:
role of distance & propagation delay in collision likelihood

- Longer distances => weaker signals
  => collision may be recovered
- Longer delays  => collision may be avoided
CSMA/CD (Collision Detection)

CD (collision detection):
- easy in wired LANs: measure signal strengths, compare transmitted, received signals
- difficult in wireless LANs: received signal strength overwhelmed by local transmission strength

CSMA/CD: (CSMA w/ Collision Detection)
- collisions detectable
- colliding transmissions aborted
- reducing channel wastage
CSMA/CD collision detection
Link Layer

- 1 Error detection and correction
- 2 Link-layer addressing
- 3 Multiple access protocols
- 4 Ethernet
Ethernet topology

- bus topology popular through mid 90s
  - all nodes in same collision domain (can collide with each other)

- today: star topology prevails
  - active switch in center
  - each “spoke” runs a (separate) Ethernet protocol (nodes do not collide with each other)

bus: coaxial cable

switch

star
**Ethernet: Unreliable, connectionless**

- **connectionless**: No handshaking between sending and receiving NICs

- **unreliable**: receiving NIC doesn’t send acks or nacks to sending NIC
  - stream of datagrams passed to network layer can have gaps (missing datagrams)
  - gaps will be filled if app is using TCP
  - otherwise, app will see gaps

- Ethernet’s MAC protocol: **CSMA/CD**
  (more on this next...)

Chapter 5, slide: 44
CSMA/CD vs. slotted ALOHA

**CSMA/CD**

1. **Unsynchronized:**
   - NIC (adapter) may transmit at anytime; no notion of timeslots

2. **Carrier-sense:**
   - Never transmit if others are transmitting

3. **Collision detection:**
   - Stop transmitting as soon as collision is detected

4. **Random backoff:**
   - Wait a random time before retransmitting (more later)

**Slotted ALOHA**

1. **Synchronized:**
   - Transmit at beginning of a timeslot only

2. **No carrier-sense:**
   - No check for whether others transmit or not

3. **No collision detection:**
   - No stop during collision

4. **No random backoff**
Notion of bit time

Before describing CSMA/CD, let's introduce:

\[
\text{bit time} = \text{time to transmit one bit on a Ethernet link}
\]

Example: consider a 10 Mbps Ethernet link

Q1: what is a “bit time”
A1: \( \frac{1}{(10 \times 10^6)} \) second = 0.1 microsecond

Q2: how much time is “96 bit time”
A2: 96 x 0.1 = 9.6 microsecond
Ethernet CSMA/CD algorithm

1. Adapter receives datagram from network layer, creates frame

2. If adapter senses channel idle for 96 bit time, starts frame transmission (gap to allow interface recovery)

3. If adapter senses channel busy, waits until channel idle (plus 96 bit time), then transmits

4. If adapter transmits entire frame without detecting another transmission, adapter is done with frame!

5. If adapter detects another transmission while transmitting, aborts and sends a 48-bit jam signal (to make sure all nodes are aware of collision)

6. After aborting (after sending jam signal), adapter enters **exponential backoff**:
   - adapter chooses $K$ at random (next slide is explained how)
   - adapter waits $K \cdot 512$ bit times,
   - returns to Step 2
Ethernet’s CSMA/CD (more)

Jam Signal:
make sure all other transmitters are aware of collision; 48 bits

Exponential Backoff:

- **Goal**: adapt retrans. attempts to estimate current load
  - heavy load: random wait will be longer
  - Light load: random wait will be shorter
- **first collision**: choose $K$ from \{0,1\}...
- **after 2\(^{nd}\) collision**: choose $K$ from \{0,1,2,3\}...
- **after 3\(^{rd}\) collision**: choose $K$ from \{0,1,2,3,4,5,6,7\}...
- **after 10\(^{th}\) collision**: choose $K$ from \{0,1,2,3,4,...,1023\}...
- **after $m$\(^{th}\) collision**: choose $K$ from \{0,1,2,...,2^{m-1}\}
- Then, delay transmission until $K \times 512$ bit times
Example

- A and B are connected via an Ethernet link of 10 Mbps
- Propagation delay between them = 224 bit time
- At time $t=0$, both transmit which results in collision
- After collision, A chooses $K=0$ and B chooses $K=1$

Q1: how much is “bit time”: $\frac{1}{10^7} = 0.1$ microsecond

Q2: at what time does collision occur?
$=\frac{224}{2} \times 0.1 = 11.2$ microsecond

Q3: at what time does bus become idle?
$=224$ (both A & B detect collision) + 48 (A & B finish sending jam signals) +
   224 (last bit of B’s jam signal arrives at A) = 496 bit time

$=496 \times 0.1$ (bit time) = 49.6 microseconds

Q4: at what time does A begin retransmission?
$[(496 \text{ (bus becomes idle)} + 96) \times 0.1]$ (bit time) = 59.2 microsecond
Distance

- A and B are connected via an Ethernet link of R bps
- A and B are separated by distance d meters
- Speed of propagation is s meter/second
- Length of each frame is L bits

- Suppose A starts transmitting a frame, and before it finishes, B starts also
- Q1: can A finish transmitting before it detects that B starts transmitting? Basically, is it possible for A not to detect collision?
  Answer: yes- if frame is not long enough, A can't detect collision

- Q1: find relationship between L, R, s, and d, so A can detect collision?
  Hints:
  1. B can't send if medium is busy: listen-before-talk policy
  2. Think of worst case, when just before the first bit coming from A arrives to B, B starts sending
  3. Solution is in hw4 Solution ;)

Chapter 5, slide: 50