MEDIA ACCESS CONTROL
Random Access & Token Rings

The Data Link Layer

- Simplest way to create a network of nodes
  - Connect the nodes directly with a physical medium

- Point-to-point links:
  - Choice of Media
  - Encoding
  - Framing
  - Error detection
  - Error recovery

- Shared Media:
  - Media access control
Shared Access Networks

- One way of scaling up point-to-point links:
  - Connect multiple hosts to the same “link”

- Media access control protocols:
  - Arbitrate access if multiple hosts wish to transmit at the same time

- Protocol objectives (for a shared channel of capacity $R$):
  - Efficiency:
    - When only one node has data to send, it gets throughput of $R$
  - Fairness:
    - When $M$ nodes want to send, each gets throughput of $R/M$
  - Support for low-latency applications
  - Decentralized protocol (no single point of failure)
  - Simple and inexpensive to implement

Types of Media Access Protocols

- Channel Partitioning protocols
  - e.g.: TDM, FDM, CDMA

- Random access protocols
  - e.g.: Ethernet, 802.11

- Taking turns protocols
  - e.g.: Token ring
Channel Partitioning Protocols

- Channel is “statically” partitioned among nodes
  - Transmission divided into “slots”
  - Slots allocated to nodes
    - Frequency division multiplexing (FDM)
    - Time division multiplexing (TDM)

- Slots are “idle” if not used by owning node
  - No sharing of slots!
  - Fair and decentralized, but not efficient

Examples of Types of Partitioning

- Partitioning frequency:
  - Different radio stations use different frequency bands

- Partitioning space:
  - FM channels in different cities use same frequency band
  - “Sectors” in cell networks

- Partitioning time:
  - Used by DECT (cordless phones)

- Partitioning frequency and time:
  - Used by GSM cellular systems

- Code division multiple access:
  - Used by CDMA cellular systems
Random Access Protocols

- No fixed pre-determined schedule of transmission
  - Transmit, if data available
  - Detect if “collision” occurs

- Typically, CSMA/CD
  - Carrier Sense (CS): listen before transmitting
  - Collision Detection (CD): listen while transmitting and stop on detecting collision
  - Multiple Access (MA): Multiple nodes plugged into common “bus” medium

- Example protocols:
  - Slotted Aloha
  - Ethernet (multiple-access bus version)
  - 802.11 (wireless)

Ethernet: Shared Bus Media Access

- If line is idle
  - Send immediately (efficiency)
  - Upper bound message size: 1500 B (latency)

- If line is busy
  - Wait until idle and transmit immediately (efficiency)
  - Is this really efficient when multiple nodes waiting?
    - Aloha: wait until idle and transmit with probability $p$

- How to guarantee that collision is detected (before frame is completely transmitted)?
  - Impose max propagation delay (51.2 µs), minimum frame size (64 B)
  - If collision, then what?
Ethernet: If Collision, Then What?

- **Exponential back-off**
  - 1<sup>st</sup> collision: randomly transmit after: 0 or 51.2 µs
  - 2<sup>nd</sup> collision: randomly transmit after: 0, 51.5, 102.4, 153.6 µs
  - n<sup>th</sup> collision: transmit after k x 51.2 µs, for randomly selected k = 0, ..., 2^n – 1
    - n capped at 10
  - Give up after 16 tries

- How does randomization help here?
- Why increase back-off range with time?
- What fairness properties can you expect?
- What is the efficiency of this protocol?

## Token Ring: Media Access Protocol

- **Structure:**
  - Set of nodes connected in a ring structure
    - Data always flows in one direction
  - Each node receives frames and forwards it
    - Destination saves a copy as frame passes by

- **Basic Idea for Media Access:**
  - A special “token” frame denotes right to transmit
    - Node that currently holds the token can transmit
  - When a nodes with data to transmit sees a token, it forwards data instead of token
    - When data frame reaches back to sender, it removes it and reinserts token
Token Ring: Properties

- **Basic Idea:**
  - A special “token” frame denotes right to transmit
    - Node that currently holds the token can transmit
  - When a node with data to transmit sees a token, it forwards data instead of token
    - When data frame reaches back to sender, it removes it and reinserts token
- How to guarantee fairness?
- How to improve efficiency?
- How to guarantee low latency?
- How to provide failure resistance?

Token Ring: Failure Recovery

- **On time expiry, a node that doesn’t see token**
  - Transmits a “claim” token with a token-holding-time bid
  - When others receive claim frame
    - Replace with a lower bid (if needed) and forward
  - If your claim frame makes it back to you,
    - Your bid was lowest (and everyone has agreed to it by now)
    - Create and insert new token
Challenges in a Wireless Setting

- Basic framework for Media Access:
  - Carrier-sense: like Ethernet, listen to medium & wait if busy
  - Collision avoidance: unlike Ethernet, no collision detection

- Complication with collision detection:
  - Expensive
  - Hidden node problem
    - Both A and C transmit, unaware of collision

- Complication with carrier-sense:
  - Exposed node problem
    - If B is transmitting to A, C will refrain from transmitting to D
      - Even though it can!

802.11 Media Access Protocol: MACAW

- MACAW Protocol:
  - Multiple Access With Collision Avoidance for Wireless LANs
  - Basic philosophy: don’t do collision detection, but collision avoidance

- Sender, receiver exchange control frames before a data frame
  - Sender sends RTS (Request to Send), with length of data frame
  - Receiver replies CTS (Clear to Send), with echo of frame length
  - If sender gets CTS frame, it sends data frame
    - If not, does exponential back-off before trying again
  - Receiver sends ACK after receiving data frame

- What should other nodes do?
  - If they see a CTS frame?
  - If they see RTS, but not CTS?

- Properties: Fairness? Efficiency? Latency?
  - Zig-zag decoding: using collisions
Media Access Control: Summary

- Protocol objectives (for a shared channel of capacity $R$):
  - Efficiency: when only one node has data to send, it gets throughput of $R$
  - Fairness: when $M$ nodes want to send, each gets throughput of $R/M$
  - Support for low latency applications
  - Decentralized protocol (no single point of failure)
  - Simple and inexpensive to implement

- Ideas used:
  - Virtualization: helps guarantee fairness
    - Partitioning (TDM/FDM), round-robin access opportunity (Token Ring)
  - Fine-grained multiplexing: helps achieve low latency
    - Fixed slot size (TDM/FDM), Limited frame size (Ethernet, 802.11, Token Ring)
  - Randomization: helps achieve fairness in many cases
    - Exponential back-off on collision (Ethernet, 802.11)
  - Signaling: helps achieve efficiency in multi-node settings
    - Channel-grabbing protocol (MACAW)