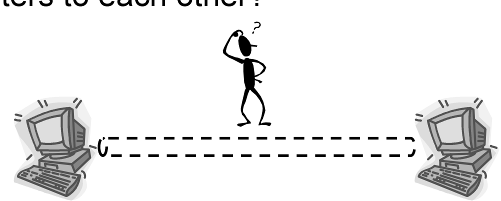
# Link Layer review and LAN technologies



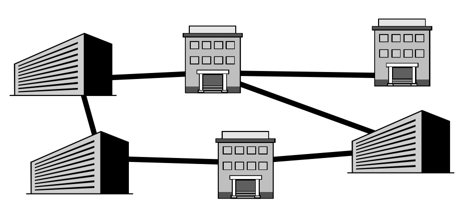


## Problem

- How to connect two (or more) computers to each other?
  - physical cable, bit encoding, framing, error detection?
  - reliable transfer mechanisms?
  - multiple access? (LANs)



- How to build larger networks?
  - single LANs have limited number of stations that can be connected and limited geographical coverage



## Overview

- Physical link technologies
  - Electrical, optical and wireless media; speeds from 2 Mbps (E1) to 2.4 Gbps (SDH-16)
  - "Last mile link": modem, cable modem, ADSL, VDSL
- Encoding
  - Rules to convert bits into electrical/optical signals, e.g., 4B/5B
- Framing
  - Frames separated by Start of Frame/End of frame bit sequences
- Error detection/correction
  - Extra information in frames to allow detection/correction of bit errors
- Reliable transmission
  - Basic mechanism at L2
- Multiple access techniques
  - Ethernet, Token ring and FDDI, Wireless
  - Limited by "collision domain"
- Extending a single LAN by using LAN switching techniques

Topics of this lecture





## Outline

- Error detection/correction
- Reliable transmission
- Multiple access techniques
  - Ethernet
  - Token ring and FDDI
  - Wireless
- Extended LANs

## Error detection vs. correction

- Method: add *k* redundant extra bits computed from the original message to detect (and to correct) bit error(s)
- Error detection
  - idea: $k << n$ ($n$ = length of the original message)
  - In Ethernet $k = 32$ and $n = 1500*8 = 12\,000$
  - bit error patterns can be detected but not corrected
  - reasonable if packets (frames) are corrupted relatively seldom
    - in modern optical networks bit error rates ~ $10^{-12}$
    - corrupted frames are retransmitted (and with high probability correctly)
- Error correction
  - now: $k \sim n$
  - what errors can be corrected depends on the used algorithm (codes) and *k*
  - useful in an environment where packets are frequently corrupted
    - in wireless, bit error rates are often ~ $10^{-6}$ - $10^{-4}$
    - $\Rightarrow$ retransmitted frames encounter errors with high probability!





## CRC (Cyclic Redundancy Check)

- One of the most common error detection techniques
  - used in almost all L2 protocols (HDLC, Ethernet, Token Ring, ATM)
- The method:
  - Let $C(x)$ = divisor polynomial of degree *k*, e.g., $x^3+x+1$
  - Let $T(x)$ = original message with *k* zeros appended
  - Divide T(x) with C(x) (modulo 2 logic), subtract the remainder from T(x)
    - the result is now exactly divisible with $C(x)$
  - Thus, if e.g. original message and the remainder are transmitted in a frame, the receiver can determine if the message is corrupted
  - Advantage: can be implemented efficiently in hardware using shift registers
- General error detecting properties of $C(x)$ with degree *k*
  - All single bit errors if $x^k = x^0 = 1$
  - All double bit errors if $C(x)$ has a factor with at least three terms
  - Any odd number of errors if $C(x)$ contains the term $(x+1)$
  - Any burst error for which the length of the burst is less than *k* bits (most burst errors of larger length can also be detected)

## Internet checksum

- CRC provides strong protection and is used in (almost) all L2 protocols
- Thus, in Internet (L3 and L4) protocols strong protection not necessary
- Simple checksum based method is used in Internet
- Method:
  - A message is viewed as a sequence of 16 bit integers
  - Add all these up using ones complement arithmetic
  - Take ones complement of the result ⇒ checksum
  - Ones complement with 4 bits
    - A negative integer -x is represented by the complement of x
    - {+5 = 0101, -5 = 1010}, {+3=0011, -3 = 1100}
    - Carry bit: -5 + -3 = 1010 + 1100 = 0110 + “carry bit” = 0111





## Outline

- Error detection/correction
- Reliable transmission
- Multiple access techniques
  - Ethernet
  - Token ring and FDDI
  - Wireless
- Extended LANs

# Reliable transmission

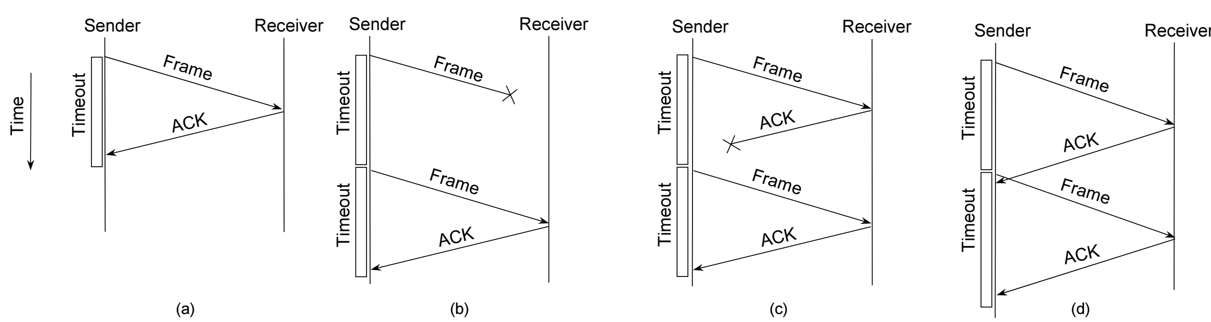
- Packets may be corrupted or simply dropped (due to congestion)
- In packet networks lost packets are retransmitted
- How is this achieved?
  - Acknowledgements (ACKs):
    - short control packet from the receiver
    - acknowledges a successfully received packet
  - Time outs:
    - sender waits for a “reasonable” time for an ACK
    - if an ACK is not received, a time out occurs ⇒ packet is retransmitted
- Two ARQ (Automatic Repeat Request) algorithms
  - Stop-and-wait
  - Sliding window





# Stop-and-wait

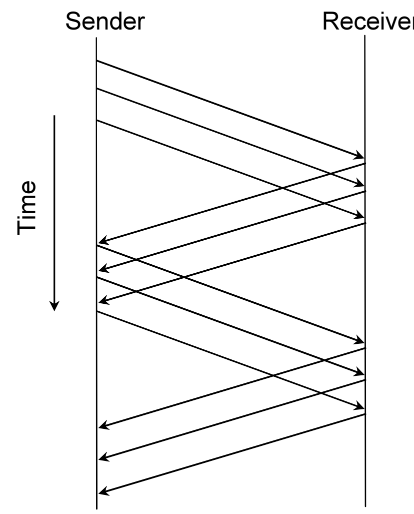
- The sender stops after each packet and waits for an ACK (a)
  - different error situations in b,c,d




- Problem: keeping the pipe full
  - C = 10 Mbps, RTT = 30 ms ⇒ RTT*C = 38 KB
  - sending only one 1500 B packet at a time gives utilization of 1.5/38 ≈ 4 %!
  - simple solution: to send for example 3 frames, use 3 stop-and-wait processes in parallel (concurrent logical channels, used in ARPANET)

## Sliding window

- Algorithm for handling multiple outstanding ACKs
  - the method used in Internet retransmission schemes
- An upper bound is set on the number of outstanding ACKs
  - called a *window*
  - protocol ensures that the number of unacknowledged packets is less than the window size
- In practice, bandwidth-delay product varies over time
  - need to manage the window size dynamically
  - TCP with congestion control

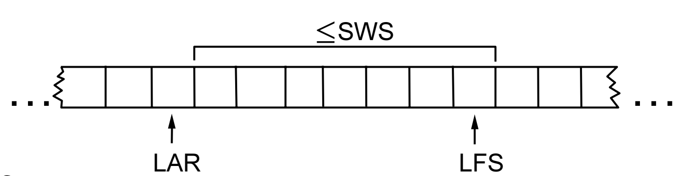







## Sliding window - sender side operations

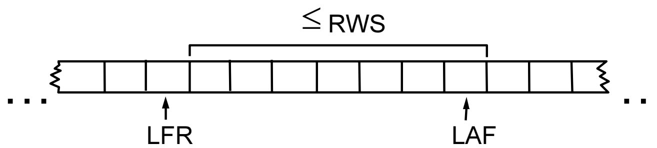
- A sequence number (SeqNum) is assigned to each frame
  - assume for the time being that SeqNum can grow infinitely large
- Three state variables
  - Send Window Size (SWS) (upper bound on nof outstanding ACKs)
  - Last Acknowledgement Received (LAR)
  - Last Frame Sent (LFS)
- Sender maintains: LFS - LAR ≤ SWS




- Sender's actions
  - For each ACK, LAR is incremented and a new packet can be transmitted
  - For each packet, a timeout timer is associated (retransmission if time expires)
  - Thus, sender must buffer upto SWS amount of packets

## Sliding window - receiver side operations

- Three state variables
  - Receive Window Size (RWS) (bound on out of order frames)
  - Largest Acceptable Frame (LAF)
  - Last Frame Received (LFR)
- Receiver maintains: LAF - LFR ≤ RWS




- Receiver's actions:
  - If SeqNum ≤ LFR or SeqNum > LAF, the frame is discarded
  - If LFR < SeqNum ≤ LAF, the frame is accepted
  - If SeqNum = LFR + 1, LFR is incremented and frame SeqNum is ACKed
  - If SeqNum > LFR + 1, no ACK is generated until the missing frames arrive (sends cumulative ACKs)





## Sliding window operation

- If packets arrive in order and none are lost
  - receiver keeps increasing its LFR and acknowledges each packet
  - sender receives a flow of ACKs and sends a new packet for each ACK
    - pipe stays full
- If packets arrive out of order at the receiver
  - the receiver does not generate ACKs
    - the sender is throttled (cannot send new packets)
  - if missing packets are lost, sender's timeout mechanism takes care of retransmissions
    - sending NAKs not useful
    - how quickly the timeout mechanism detects the missing packet becomes important
  - sending selective ACKs (ACKs for each received packet even if they are out of order) is possible but increases protocol complexity

## Finite sequence numbers

- In practice, SeqNum is a field in the protocol header $\Rightarrow$ SeqNum finite
  - sequence numbers bound to wrap around during operation
- Problem:
  - How big must MaxSeqNum be in order to guarantee that receiver never mistakes a received SeqNum to the previous “round’s” same SeqNum?
- Answer: if SWS = RWS, then SWS $\leq$ (MaxSeqNum + 1) / 2
  - SWS must be smaller than half of the nof values of MaxSeqNum
  - Interpretation: assume MaxSeqNum = SWS + 1 and SWS = RWS = 7
    - in the worst case sender sends frames 0, ... , 6
    - assume that all ACKs for the packets are lost
    - sender retransmits again all frames 0, ... , 6
    - receiver is expecting frames 7, 8, 0, 1, ..., but receives 0, ... , 6 interpreting them as belonging to the “next round”!
    - MaxSeqNum must be big enough that all retransmitted frames still “fit” in the same round, i.e., in this case SeqNo range = {0, ..., 13}



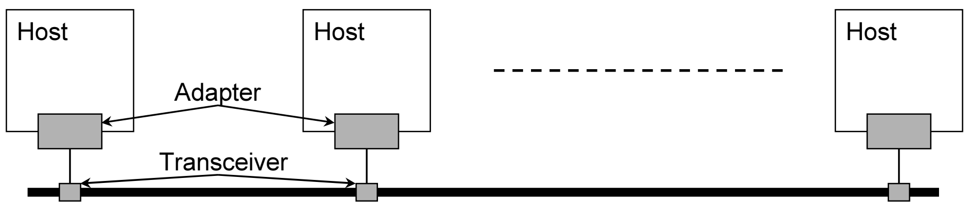


## Outline

- Error detection/correction
- Reliable transmission
- Multiple access techniques
  - Ethernet
  - Token ring and FDDI
  - Wireless
- Extended LANs

# Ethernet overview

- History
  - developed by Xerox in mid 70s, roots in Aloha packet-radio network
  - standardized by Xerox, DEC, and Intel in 1978, (later IEEE 802.3 standard)
- CSMA/CD
  - carrier sense: nodes detect if line is idle or busy
  - multiple access: multiple stations share the bandwidth
  - collision detection: stations listen to their transmission and detect collisions
- Bandwidth: 10Mbps, 100Mbps, 1Gbps
- Ethernet segment (different coaxial cables, max 500 m):
  - transceiver: detects if line is idle, sends the electrical signals
  - adapter: implements the Ethernet MAC protocol (in hardware)

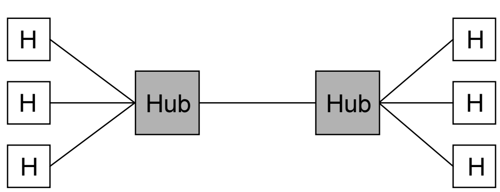







# Collision domain

- Using max 4 repeaters at most 5 segments can be connected
  - max 2500 m distance between any two nodes
- Hubs can be used to create a star (hierarchical) topology
  - used in 10BaseT networks with twisted pair cabling
  - 10 = 10 Mbit/s, Base = Baseband system, T=twisted pair (< 100 m)




- Repeaters and hubs are layer 1 devices connecting Ethernet segments
  - data transmitted by any host on that Ethernet is received by all hosts
  - all compete for the same resource
  - all hosts are in the same *collision domain*

## Ethernet frame format and addresses

- Frame Format (field lengths in bits)
  - max body length 1500 bytes
  - min body length 46 bytes (long enough to detect a collision)

| 64 | 48 | 48 | 16 | | 32 |
|---|---|---|---|---|---|
| Preamble | Dest addr | Src addr | Type | Body | CRC |

- Addresses
  - unique, 48-bit unicast address assigned to each adapter
  - example: 8:0:e4:b1:2
  - broadcast: all 1s
  - multicast: first bit is 1
- Receiver functionality simple:
  - adapter forwards to the host all unicast traffic directed to it, all broadcast traffic and the multicast traffic it has subscribed to
- Problem: Distributed algorithm that provides fair access
  - Media Access Protocol (MAC)





## Transmit algorithm (1)

- No centralized access control
  - collisions occur and they are detected
- If line is idle…
  - send immediately
  - upper bound message size of 1500 bytes
  - must wait 9.6 $\mu$s between back-to-back frames
- If line is busy…
  - wait until idle and transmit immediately
  - called 1-persistent (special case of p-persistent)
    - p-persistent: if line is idle, transmit with probability p
    - idea: many hosts may be waiting for the line to become idle and we do not want them all to start transmitting (minimizes prob. of collisions)

## Transmit algorithm (2)

- If collision…
  - jam for 32 bits, then stop transmitting frame
  - minimum frame is 64 bytes (header + 46 bytes of data)
    - long enough to fill a 2500 m Ethernet operating at 10 Mbps
  - delay and try again
    - 1st time: 0 or 51.2us
    - 2nd time: 0, 51.2, 102.4, or 153.6us
    - 3rd time: choose k=0,…,$2^3$-1 randomly, wait k x 51.2us
    - nth time: for randomly selected k=0,…, $2^n$-1, wait k x 51.2us
    - give up after several tries (usually 16)
    - exponential backoff





## Ethernets in practice

- Performance
  - Ethernet works efficiently in light load
  - 30% load is considered a heavy load and too much of Ethernet's capacity is wasted on collisions
  - no flow control in Ethernet (flow control implemented in IP protocols)
- Nof hosts
  - theoretical maximum 1024 hosts
  - in reality most have < 200 hosts
- Length
  - theoretical maximum 2500 m with round-trip delay 51.2 $\mu$s
  - in practice, delay is closer to 5 $\mu$s
- Ethernet advantages:
  - easy to manage and administer (add/remove hosts, no route configuration)
  - cheap

## Outline

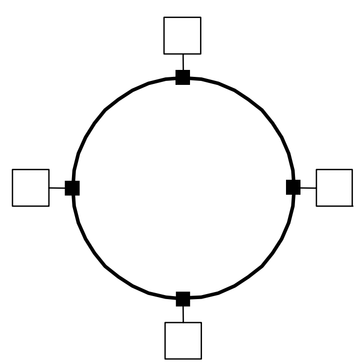
- Error detection/correction
- Reliable transmission
- Multiple access techniques
  - Ethernet
  - Token ring and FDDI
  - Wireless
- Extended LANs





## Token ring overview

- Examples
  - 16Mbps IEEE 802.5 (based on earlier IBM ring)
  - 100Mbps Fiber Distributed Data Interface (FDDI)
- Similarities with Ethernet
  - shared medium with a distributed access algorithm
  - all nodes see all frames
- Differences to Ethernet
  - ring topology
  - access to the ring is tightly controlled (tokens)
    - NOT random access



## Token ring operation

- Idea
  - frames flow in one direction: upstream to downstream
  - special bit pattern (token, length 24 bits) rotates around ring
  - if a node has frame(s) to transmit and sees the token
    - node inserts its frame into the ring
    - each node forwards the frame, receiver copies it
    - node can transmit for upto THT (Token Holding Time, default 10 ms)
  - release token after done transmitting
    - immediate release (token is inserted before last frame has been sent)
    - delayed release (token is inserted after last frame has been sent)
  - remove your frame when it comes back around
  - stations get round-robin service
- Additional features
  - supports unicast, broadcast and multicast addresses
  - reliable frame delivery: receiver sets A and C bits if frame OK
  - supports priorities





## Token ring maintenance

- Designated monitor station guards operation of the ring
  - any station can be the monitor
  - healthy monitor issues regular control messages
  - if control msgs are not received for some period of time
    - any station can try to become new monitor by issuing “claim token” msg
    - new monitor is elected based on “highest address” rule
- Monitor functions
  - adds extra bits of delay if necessary (ring must be at least 24 “bits long”)
  - makes sure that token is not lost (host crash, bit errors, ...)
    - maximum rotation time timer (NumStations x THT + RingLatency)
  - check for corrupted or orphaned frames
  - detection of dead stations

## Physical properties and frame format of token ring

- Properties:
  - Robustness:
    - in a ring, if any station fails the ring is inoperable
    - solution: an electromechanical relay in the network adapter is open as long as the station is operating
  - data rate: 4 Mbps (old version) or 16 Mbps
  - bit encoding: Manchester
  - upto 260 stations (250 in IEEE 802.5)
  - physical medium: twisted pair
- Frame Format

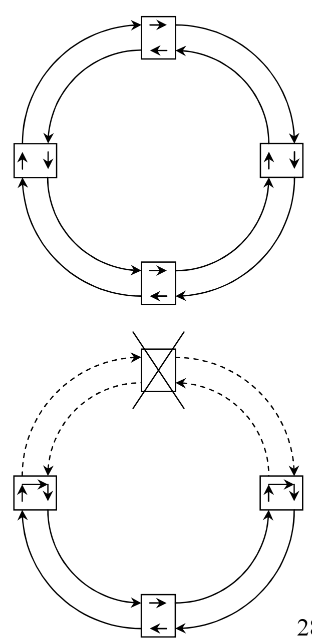
| 8 | 8 | 8 | 48 | 48 | | 32 | 8 | 24 |
|---|---|---|---|---|---|---|---|---|
| Start of frame | Access control | Frame control | Dest addr | Src addr | Body | CRC | End of frame | Status |

  - 48 bit addresses (as in Ethernet)
  - frame status contains the A and C bits





## FDDI overview



- Dual ring
  - second ring not used in normal operation
  - if a node fails or a single link fails, ring loops back to form a complete ring
- Dual ring expensive
  - SAS (single attachment station) possible
  - multiple SAS connected to a DAS (dual attachment station) (=concentrator)
  - in case of SAS failure, DAS bypasses the faulty SAS
- Physical properties
  - at most 500 stations
  - max cable length 200 km (100 km ring length)
  - physical media: coax, twisted pair, fiber
  - encoding: 4B/5B
- Frame format (almost) the same as in Token ring

## Timed Token Algorithm (1)

- Algorithm for controlling token holding time: Timed Token Algorithm
  - ensures that every station gets to transmit within a defined period of time
- Token Holding Time (THT)
  - upper limit on how long a station can hold the token, configured value
  - same as in Token ring
- Token Rotation Time (TRT)
  - how long it takes the token to traverse the ring
  - TRT <= ActiveNodes x THT + RingLatency
- Target Token Rotation Time (TTRT)
  - agreed-upon upper bound on TRT
  - decided during token generation (ring initialization)





## Timed Token Algorithm (2)

- Each node measures TRT between successive tokens
  - if measured-TRT > TTRT: token is late so don't send
  - if measured-TRT < TTRT: token is early so OK to send
- Problem:
  - an upstream node with lot of data to send hogs all bw before the token reaches downstream nodes which might have delay critical data
- Two classes of traffic
  - synchronous: can always send
  - asynchronous: can send only if token is early
- Problem: synchronous traffic can take all bw
- Solution: max TTRT amount of synch data can be sent during token round
  - nodes first send TTRT worth of asynchronous data and then other nodes send TTRT worth of synchronous data
  - worst case: measured-TRT can be 2xTTRT between seeing token
  - in the next token round, token is already late and asynchronous data cannot be sent ⇒ back-to-back 2xTTRT rotations not possible

## Token maintenance

- Lost Token
  - no token when initializing ring
  - bit error corrupts token pattern
  - node holding token crashes
  - token loss monitored by all stations
    - stations should see valid frames or tokens every now and then
    - if nothing is seen for 2.5 ms, node issues a “claim” msg
- Generating a Token (and agreeing on TTRT)
  - execute when joining ring or suspect a failure
  - send a claim frame that includes the node’s TTRT bid
  - when receive claim frame, update the bid and forward
  - if your claim frame makes it all the way around the ring:
    - your bid was the lowest
    - everyone knows TTRT
    - you insert new token





## Outline

- Error detection/correction
- Reliable transmission
- Multiple access techniques
  - Ethernet
  - Token ring and FDDI
  - Wireless
- Extended LANs

## Wireless LANs

- (Original) Wireless LAN standard: IEEE 802.11
  - limited geographical coverage
  - defines MAC protocol suitable for wireless environment
  - additional features: real time support, power mgmt, security
- Physical properties
  - bandwidth: 1 or 2 Mbps
  - physical media
    - 2 media based on spread spectrum radio operating in 2.4GHz frequency range
    - diffused infrared (sender and receiver do not need to have line of sight contact), distance limitation approx. 10 m
- New standards
  - IEEE 802.11a and IEEE 802.11b
  - Higher data rates: 10 Mbit/s upto 54 Mbit/s
  - New frequency range: 5 GHz



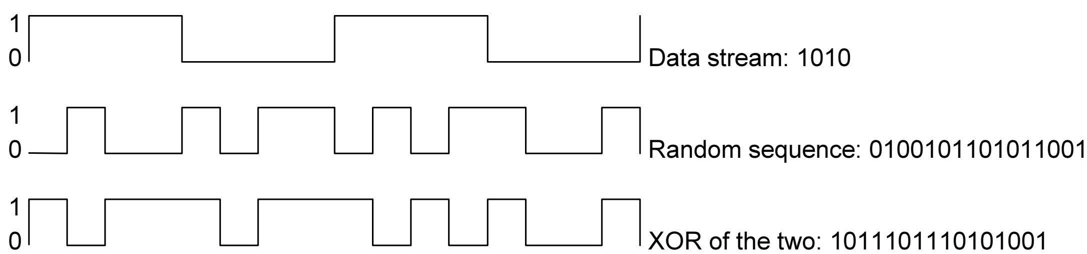


## Spread spectrum techniques (1)

- General principles
  - signal spread over wider frequency band than required
  - minimizes impact of interference from other devices
  - originally military technology, deigned to thwart jamming
  - transmission “coded” such that the signal appears as noise to an observer not knowing the “key”
    - possible to trade off capacity and amount of noise
- Frequency hopping
  - signal transmitted over random sequence of frequencies
  - sender and receiver share…
    - pseudorandom number generator
    - seed
  - $\Rightarrow$ receiver can hop frequencies in sync
  - 802.11 uses 79 x 1MHz-wide frequency bands

# Spread spectrum techniques (2)

- Direct sequence
  - for each bit, send XOR of that bit and n random bits
  - random sequence known to both sender and receiver
  - called n-bit chipping code
  - 802.11 defines an 11-bit chipping code

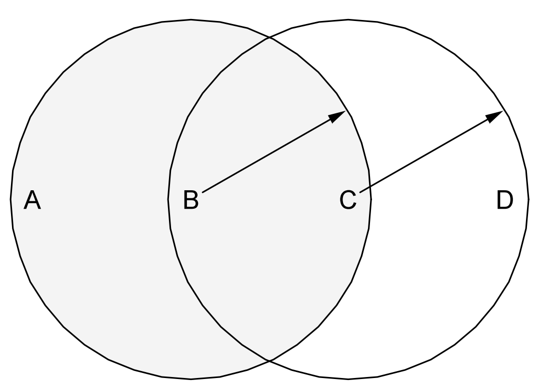







# MAC for wireless

- Idea to provide similar random access as in Ethernet, but ...
  - in wireless environment not all nodes are always within reach of each other
- Problem 1: hidden nodes
  - Assume node A and C want to transmit to B
  - A and C are unaware of each other
  - transmissions collide at B, but A and C do not know about that
- Problem 2: exposed nodes
  - suppose B is sending to A
  - C hears this
  - however, C can still transmit to D




- Wireless MAC addresses the problems by collision avoidance strategy

# MACAW

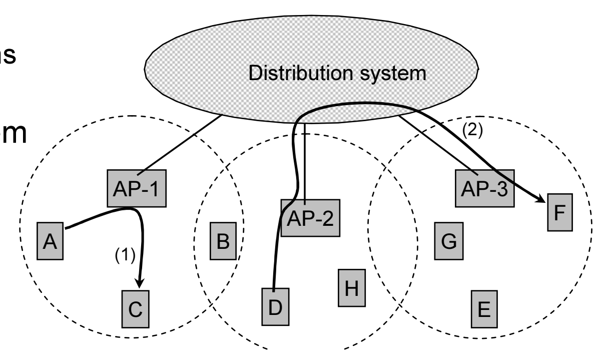
- MACAW (MACA for Wireless LANs)
  - MACA = Multiple Access with Collision Avoidance
  - idea: nodes ask for permission to send
- MACAW operation:
  - sender transmits RequestToSend (RTS) frame
  - receiver replies with ClearToSend (CTS) frame
  - neighbors…
    - that see CTS: keep quiet (they are too close to sender)
    - that see RTS but not CTS: ok to transmit
  - receiver sends ACK when it has received the frame
    - neighbors silent until see ACK
  - Collisions (= multiple RTS frames sent at the same time)
    - no collision detection
    - known when senders do not receive CTS
    - exponential backoff





# Supporting mobility: Access Points (AP)

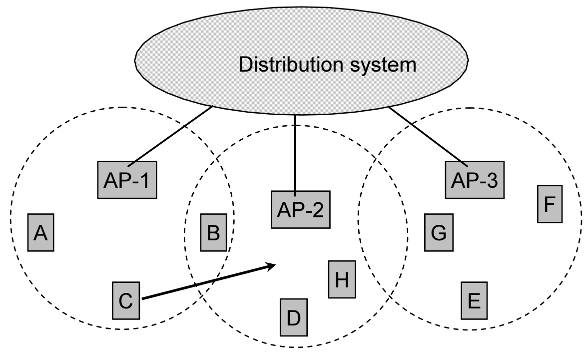
- Each AP serves hosts within a cell
  - cf. base stations in cellular systems
- APs connected to distribution system
  - 802.11 does not specify what (can be e.g. Ethernet)




- Each mobile node associates with an AP
  - hierarchical network
  - process of making associations called scanning
- Routing
  - within a cell transmissions through AP (1)
  - transmitting to a node in neighboring AP done via distribution network (2)

## Associating with an Access Point

- Active scanning
  - node C sends Probe frame
  - all APs within reach reply with ProbeResponse
  - at some point node C selects AP-2 and sends a new AssociationRequest
  - AP-2 replies with AssociationResponse and notifies AP-1 that host C has moved



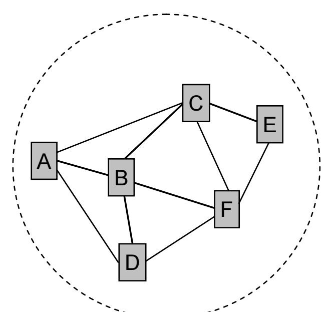

- Passive scanning
  - APs periodically send Beacon frames
  - host can decide to join at will by replying with AssociationRequest



S-38.188 - Computer Networks - Spring 2005

## Supporting mobility: ad hoc networks

- Ad hoc network
  - IEEE 802.11 stations can dynamically form a network without APs
  - each host acts as a “switch” that relays packets based on information about neighboring host location
  - mesh type network topology
  - ad hoc routing a very active research field




- Applications
  - laptop meetings in a conference
  - interconnection of personal devices (e.g. in a house)
  - battlefield

- IETF MANET (Mobile Ad hoc Networks) working group

## 802.11 frame format

| 8 | 8 | 48 | 48 | 48 | 16 | 48 | | 32 |
|---|---|---|---|---|---|---|---|---|
| Control | Duration | Addr1 | Addr2 | Addr3 | SeqCtrl | Addr4 | Body | CRC |

- Control field
  - indicates if frame is a data frame; an RTS or CTS frame; or is used by the scanning algorithm
  - ToDS and FromDS bits (used with 4 address fields)
- 4 address fields
  - if sender and receiver in same cell
    - ToDS = FromDS = 0
    - Addr1 = target, Addr2 = source
  - if sender and receiver in different cells
    - ToDS = FromDS = 1
    - Addr1 = target, Addr4 = source
    - Addr2 = AP that sent frame to target
    - Addr3 = AP that received frame from source





## Outline

- Error detection/correction
- Reliable transmission
- Multiple access techniques
  - Ethernet
  - Token ring and FDDI
  - Wireless
- Extended LANs

## Extending LANs

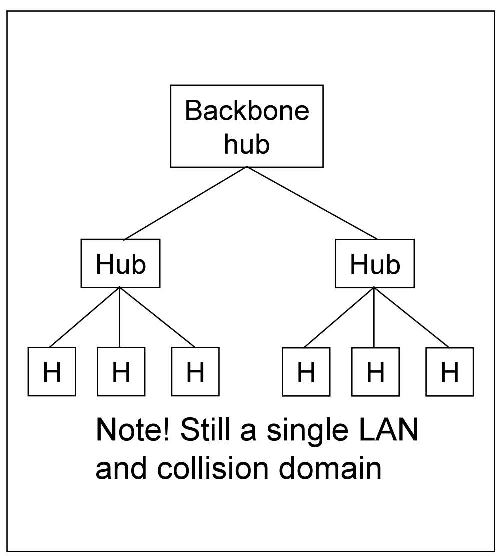
- A single LAN
  - all stations on the LAN share bandwidth
  - limited geographical coverage (Ethernet 2500 m)
  - = single collision domain
  - nof stations limited
- Techniques to extend LANs beoynd a single collision domain
  - hubs
  - bridges
  - Ethernet switches





## Hubs

- Layer 1 device
  - repeats received bits on one interface to all other interfaces (repeaters)
- Hubs can be arranged in hierarchy
  - provides star topology
- Each connected LAN referred to as LAN segment
- Hubs do not isolate collision domains
  - collisions may happen with any node on any segment



## Hubs: advantages and limitations

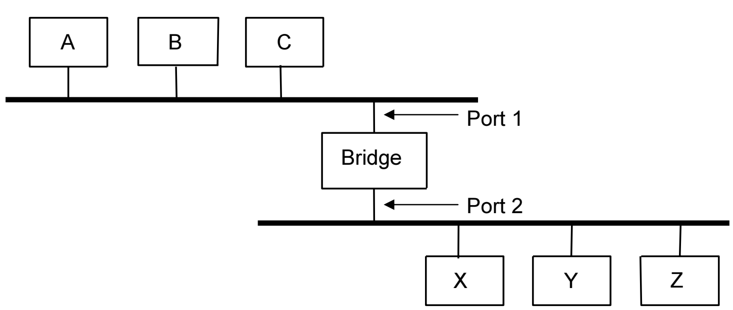
- Hub advantages:
  - simple, inexpensive device
  - hierarchy provides graceful degradation: portions of the LAN continue to operate if one hub malfunctions
  - extends maximum distance between node pairs (100m per Hub)
- Hub limitations:
  - single collision domain results in no increase in max throughput
  - multi-tier throughput same as single segment throughput
  - individual LAN restrictions pose limits on number of nodes in same collision domain and on total allowed geographical coverage
  - cannot connect different Ethernet types (e.g., 10BaseT and 100baseT)





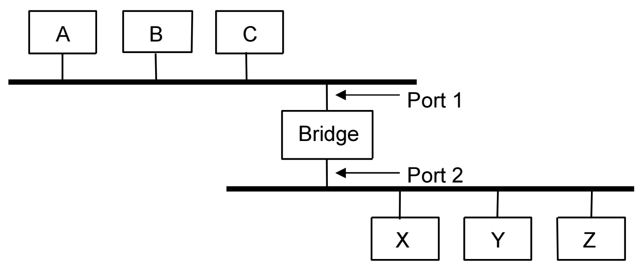
## Simple bridges

- Connect two or more LANs with a bridge
- Simple bridge
  - all packets from a particular port forwarded on all other ports
  - level 2 forwarding/switching (does not add packet header)
  - isolates collision domains
  - Ethernet bridge uses CSMA/CD to transmit packets onto connected LANs



## Learning bridges

- Idea: forward only when necessary
  - bridges maintain forwarding tables
  - datagram switching on layer 2
- Dynamic algorithm
  - bridge examines source address of each packet seen on a port
  - addresses saved in a table
  - table entries have time outs (if host moves from one segment to another)
  - broadcast frames always forwarded
- Table is an optimization; need not be complete



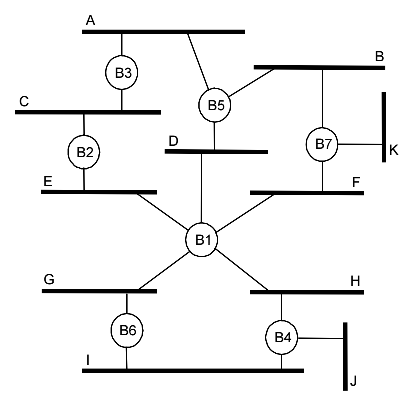

| Host | Port |
|---|---|
| A | 1 |
| B | 1 |
| C | 1 |
| X | 2 |
| Y | 2 |
| Z | 2 |

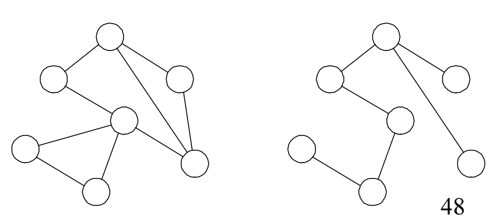




## Spanning tree algorithm

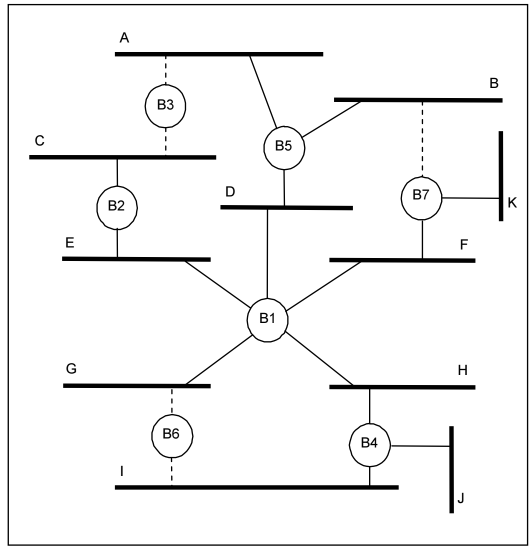
- Problem: loops in topology
  - loops used to provide redundancy in case of link failure
- Bridges run a distributed spanning tree algorithm
  - selects which bridges actively forward traffic
  - subset of network graph representing a tree that spans all nodes
  - dynamic algorithm: tree reconfigures when topology changes
  - developed by Radia Perlman
  - now IEEE 802.1 specification



# Algorithm overview

- Bridges have unique ids (B1, B2, etc.)
- Bridge with smallest id is root
  - root forwards all traffic onto all ports
- Select designated bridge on each LAN
  - LAN may be connected to many bridges
  - bridge “closest” (=min nof hops) to root selected as designated bridge
  - id used to break ties
- Each bridge forwards frames over each LAN for which it is the designated bridge
- Dynamic (survives link failures), but not able to utilize multiple paths during congestion







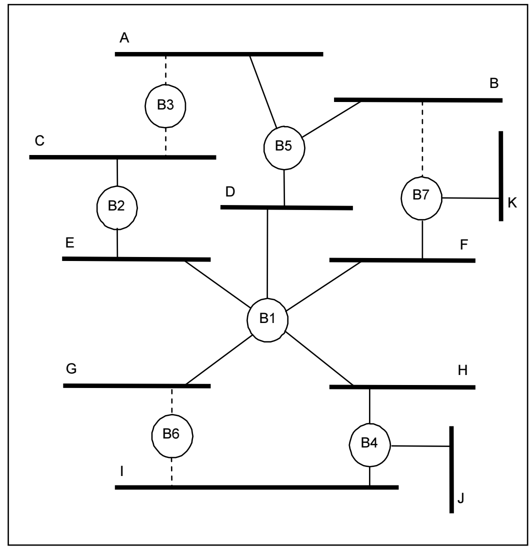

# Algorithm details

- Bridges exchange configuration messages containing
  - id for bridge sending the message
  - id for what the sending bridge believes to be root bridge
  - distance (hops) from sending bridge to root bridge
- Each bridge records current best configuration message for each port
  - identifies root with smaller id
  - identifies root with same id but with shorter distance
  - root id and distance are same but sending bridge has smaller id
- Initially, each bridge believes it is the root
- When learn not root, stop generating config messages
  - in steady state, only root generates configuration messages
- When learn not designated bridge, stop forwarding config messages
  - in steady state, only designated bridges forward config messages
- Root continues to periodically send config messages
  - If any bridge does not receive config message after a period of time, it starts generating config messages claiming to be the root

## Algorithm example

- (Y, *d*, X): Message from X, at distance *d* from root Y
- Consider node B3:

1 B3 receives (B2, 0, B2)
2 2 < 3, B3 accepts B2 as root
3 B3 increments d and sends (B2, 1, B3) towards B5
4 B2 accepts B1 as root (lower id) and sends (B1, 1, B2) towards B3
5 B5 accepts B1 as root and sends (B1, 1, B5) towards B3
6 B3 accepts B1 as root, notes that B2 and B5 are closer to root ⇒ stop

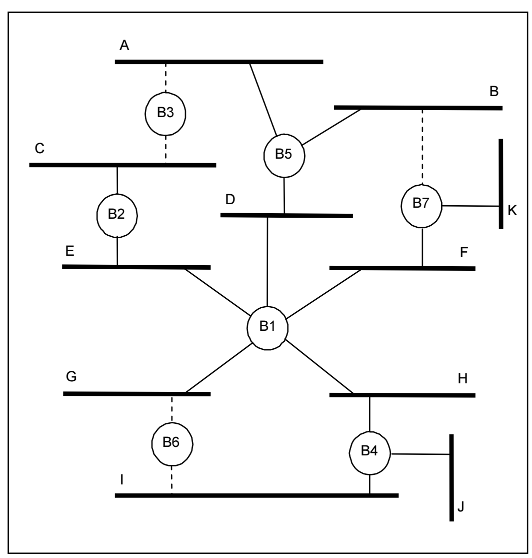







## Broadcast and Multicast

- Forward all broadcast/multicast frames
  - current practice
- Possible optimization for multicast:
  - learn when no group members downstream (similarly as in learning bridges)
  - typically group members are not sending any traffic
  - accomplished by having each member of group G send a frame to bridge multicast address with G in source field
  - not widely deployed



## Bridges: advantages and limitations

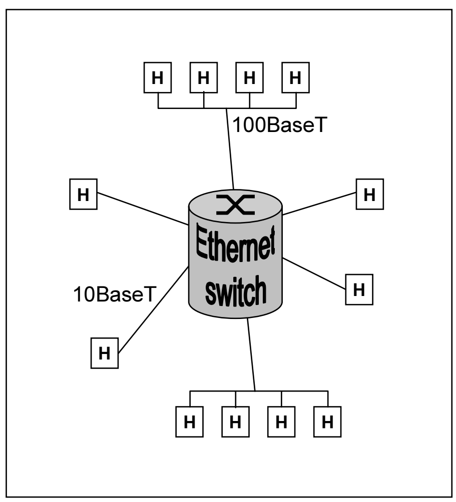
- Advantages:
  - isolates collision domains (higher throughput than when using hubs)
  - can connect multiple LAN types (different Ethernets, Token Rings)
  - transparent: no need for any changes in end hosts
  - used to build network of “tens” of LAN segments within e.g. a campus area
- Limitations:
  - scalability:
    - spanning tree algorithm does not scale
    - broadcast does not scale (VLANs can be used to alleviate)
  - heterogeneity: not all network technologies use 48 bit addresses
- Caution: beware of transparency
  - in an extended LAN, there may be congestion, larger delays, ...
  - end host applications should not assume that all is behind a single LAN





## Ethernet switches

- Layer 2 forwarding and filtering using LAN addresses
- Uses switching (frames can be sent in parallel between multiple ports)
- Can accommodate large number of interfaces
  - mix of 10/100/1000 Mbit Ethernets
  - shared (multiple hosts) or dedicated (single host) Ethernets
- Common configuration
  - star topology: hosts connected to switch
  - Ethernet, but no collisions!



# Building a campus area network