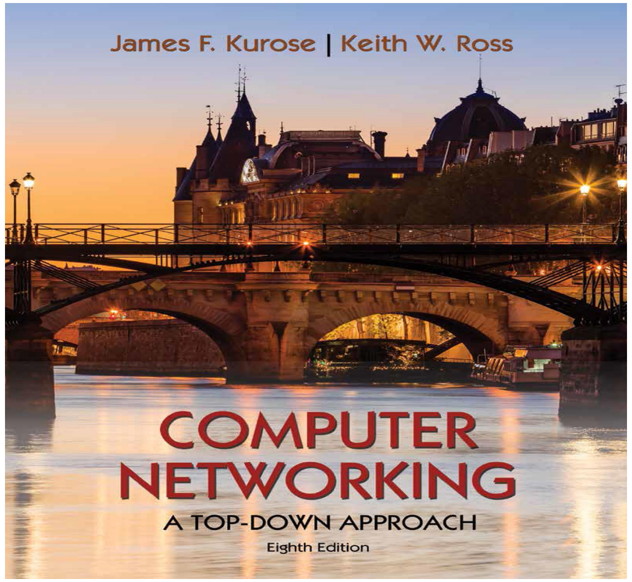
Computer Networks

The Link Layer

- Introduction to the Link Layer
- Error-detection and -correction Techniques
- **Multiple Access Links and Protocols**
- Switched Local Area Networks
- Link Virtualization: a Network as a Link Layer
- Data Center Networking
- Retrospective: A Day in the Life of a Web Page Request



Multiple access links, protocols

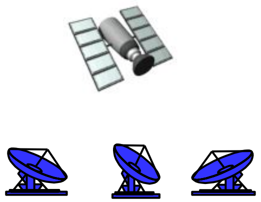
two types of "links":

- point-to-point
  - point-to-point link between Ethernet switch, host
  - PPP for dial-up access
- broadcast (shared wire or medium)
  - old-school Ethernet
  - upstream HFC in cable-based access network
  - 802.11 wireless LAN, 4G/4G. satellite

shared wire (e.g., cabled Ethernet)

shared radio: 4G/5G

shared radio: WiFi



shared radio: satellite

humans at a cocktail party (shared air, acoustical)

Multiple access protocols

- single shared broadcast channel
- two or more simultaneous transmissions by nodes: interference
  - *collision* if node receives two or more signals at the same time

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
  - no out-of-band channel for coordination

An ideal multiple access protocol

given: multiple access channel (MAC) of rate *R* bps

desiderate:

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: taxonomy

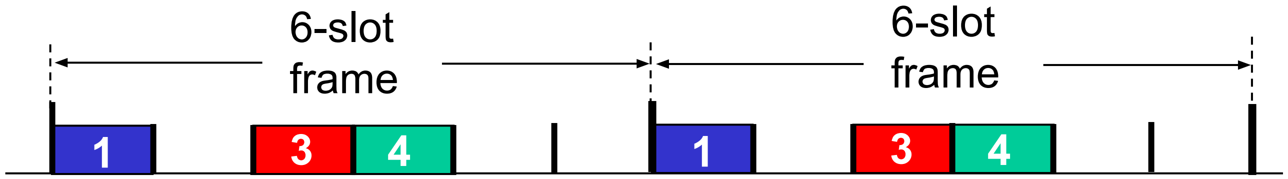
three broad classes:

- channel partitioning
  - divide channel into smaller “pieces” (time slots, frequency, code)
  - allocate piece to node for exclusive use
- random access
  - channel not divided, allow collisions
  - “recover” from collisions
- “taking turns”
  - nodes take turns, but nodes with more to send can take longer turns

Channel partitioning MAC protocols: TDMA

TDMA: time division multiple access

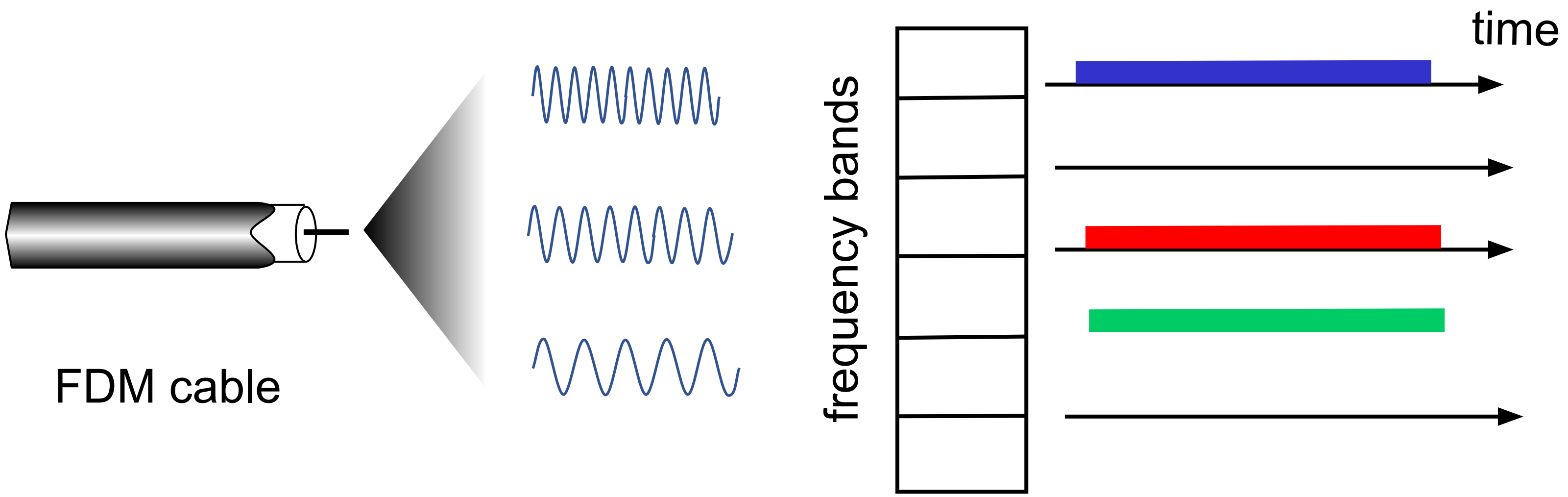
- access to channel in “rounds”
- each station gets fixed length slot (length = packet transmission time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



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
- example: 6-station LAN, 1,3,4 have packet to send, frequency bands 2,5,6 idle



Random access protocols

- when node has packet to send
  - transmit at full channel data rate R
  - no *a priori* coordination among nodes
- two or more transmitting nodes: "collision"
- random access protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)
- examples of random access MAC protocols:
  - ALOHA, slotted ALOHA
  - CSMA, CSMA/CD, CSMA/CA

Slotted ALOHA

assumptions:

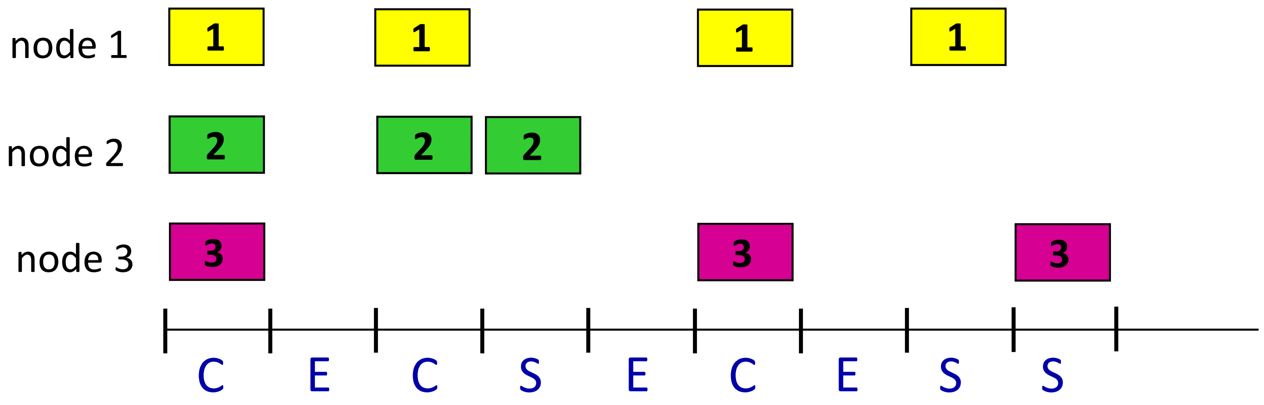
- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

operation:

- when node obtains 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 probability *p* until success

randomization – *why*?

Slotted ALOHA



C: collision

S: success

E: empty

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
- nodes may be able to detect collision in less than time to transmit packet
- clock synchronization

Slotted ALOHA: efficiency

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

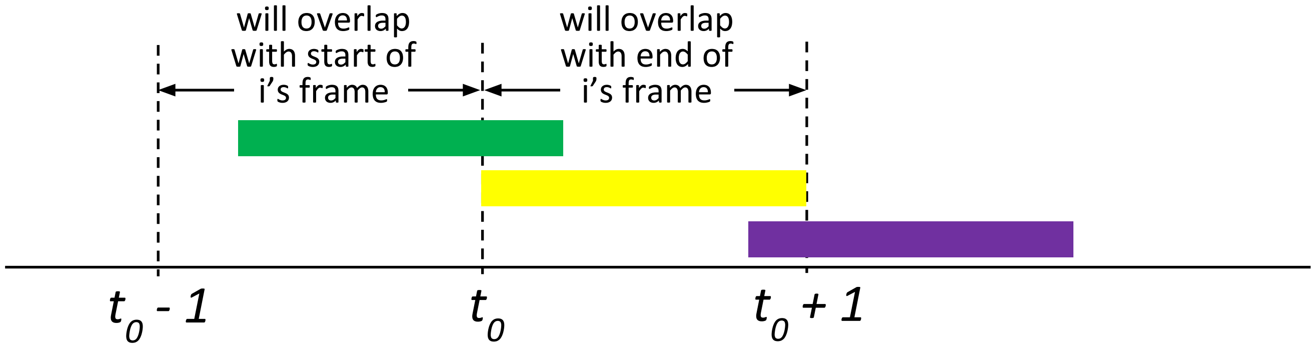
- *suppose: N* nodes with many frames to send, each transmits in slot with probability *p*
  - prob that given node has success in a slot = $p(1-p)^{N-1}$
  - prob that *any* node has a success = $Np(1-p)^{N-1}$
  - max efficiency: find p^* that maximizes $Np(1-p)^{N-1}$
  - for many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives:

  max efficiency = 1/e = .37

- *at best:* channel used for useful transmissions 37% of time!

Pure ALOHA

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


will overlap with start of i's frame
will overlap with end of i's frame
$t_0 - 1$
t_0
$t_0 + 1$

- pure Aloha efficiency: 18% !

CSMA (carrier sense multiple access)

simple CSMA: listen before transmit:

- if channel sensed idle: transmit entire frame
- if channel sensed busy: defer transmission

- human analogy: don't interrupt others!

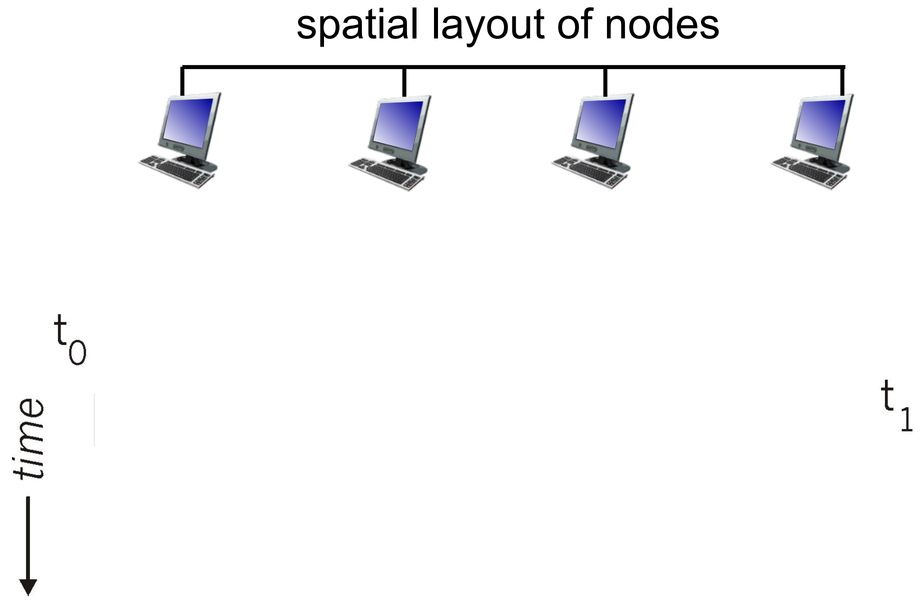
CSMA/CD: CSMA with *collision detection*

- collisions *detected* within short time
- colliding transmissions aborted, reducing channel wastage
- collision detection easy in wired, difficult with wireless

- human analogy: the polite conversationalist

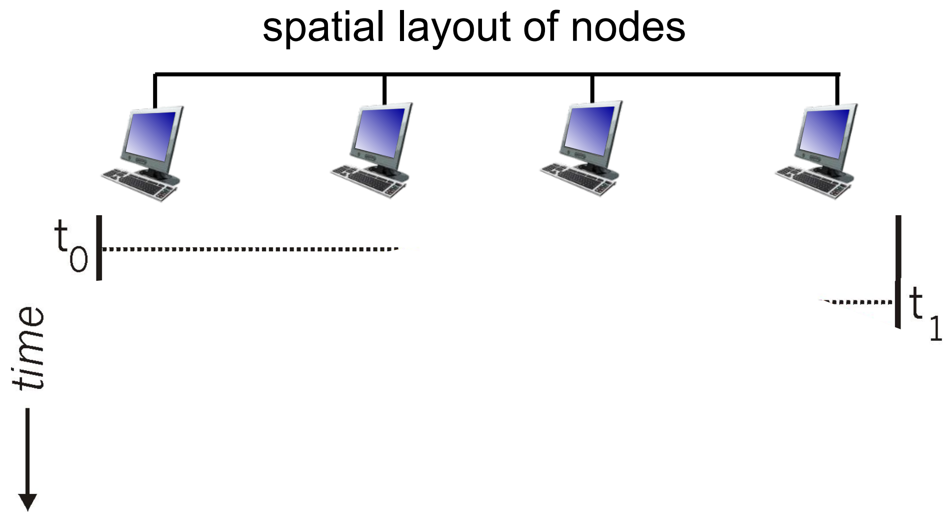
CSMA: collisions

- collisions can *still* occur with carrier sensing:
  - propagation delay means two nodes may not hear each other's just-started transmission
- collision: entire packet transmission time wasted
  - distance & propagation delay play role in determining collision probability



CSMA/CD:

- CSMA/CD reduces the amount of time wasted in collisions
  - transmission aborted on collision detection



Ethernet CSMA/CD algorithm

1. Ethernet receives datagram from network layer, creates frame
2. If Ethernet senses channel:

   if idle: start frame transmission.

   if busy: wait until channel idle, then transmit
3. If entire frame transmitted without collision - done!
4. If another transmission detected while sending: abort, send jam signal
5. After aborting, enter *binary (exponential) backoff:*
   - after *m*th collision, chooses *K* at random from *{0,1,2, …, 2^m-1}*. Ethernet waits *K*·512 bit times, returns to Step 2
   - more collisions: longer backoff interval

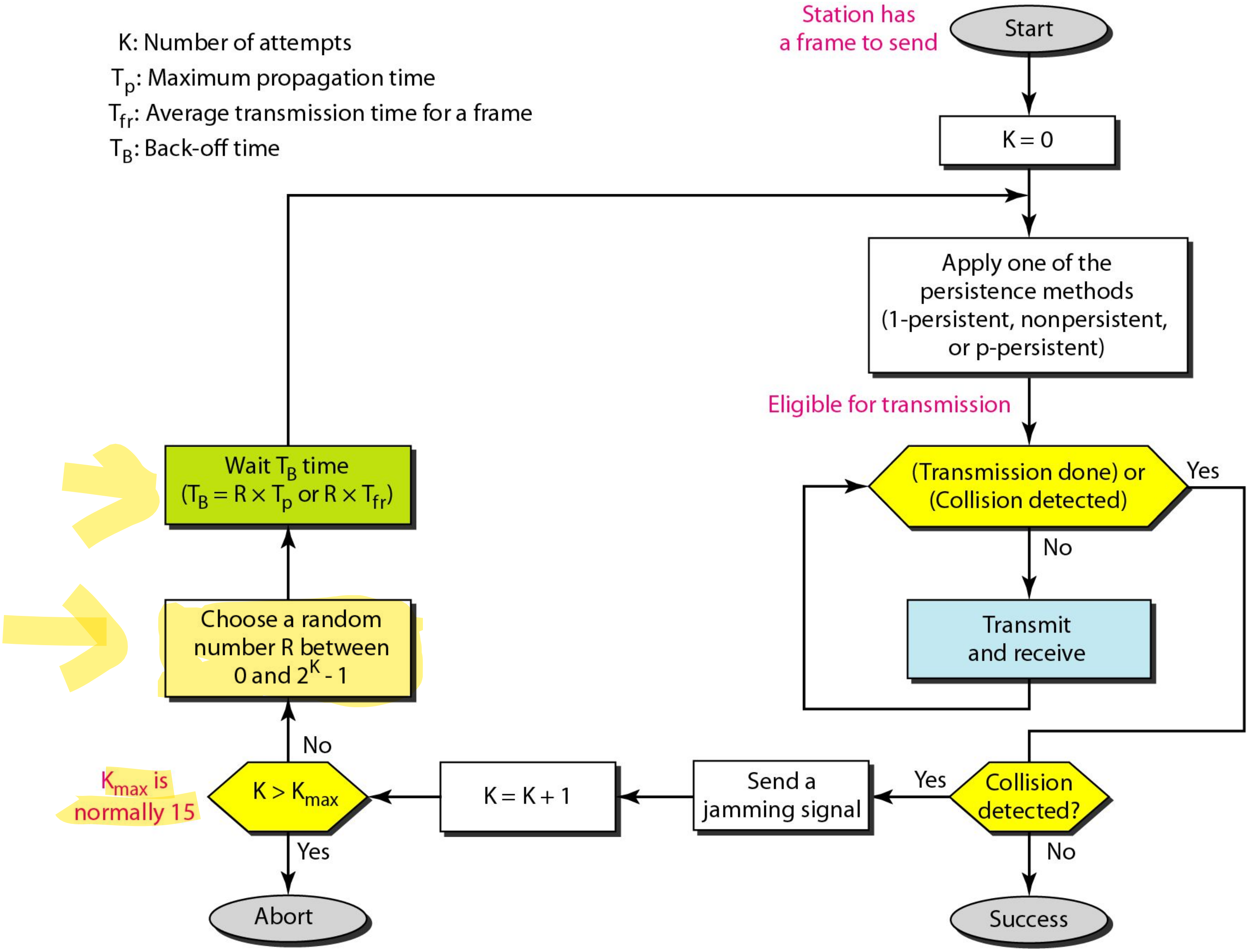
Flow diagram for the CSMA/CD

K: Number of attempts

T_p: Maximum propagation time

T_{fr}: Average transmission time for a frame

T_B: Back-off time

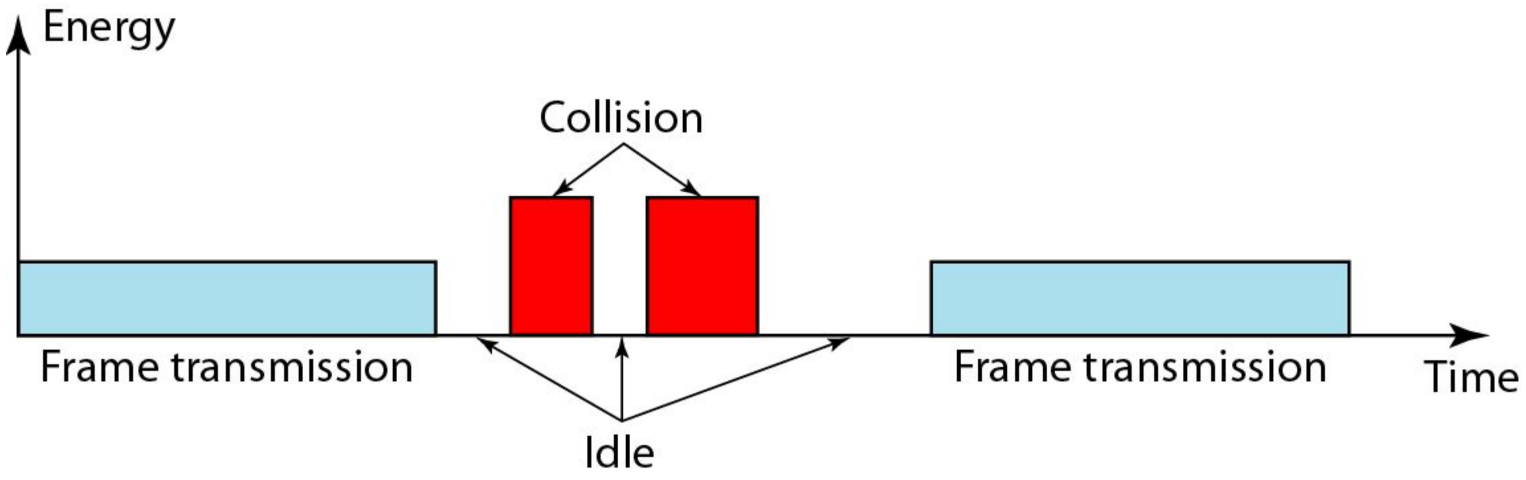


Station has a frame to send
Start
K = 0
Apply one of the persistence methods (1-persistent, nonpersistent, or p-persistent)
Eligible for transmission
(Transmission done) or (Collision detected)
Yes
No
Transmit and receive
Collision detected?
Yes
No
Success
Send a jamming signal
K = K + 1
K > K_max
K_max is normally 15
No
Yes
Abort
Choose a random number R between 0 and 2^K - 1
Wait T_B time (T_B = R × T_p or R × T_fr)

CSMA/CD: Procedure

1. Node has a frame to transmit
2. Sense the medium
   I. If busy (not idle): wait until becomes free
   II. If idle: transmit
3. After transmission, check weather collision has occurred
   I. If collision has detected
      a) Send a jam signal
      b) Increment retransmission counter
      c) Check maximum number of retransmission attempts. If yes, abort transmission
      d) Based on number of collisions, calculate random backoff time and wait that time
      e) Repeat from step 2 again
   II. If no collision detected, transmission is successful
4. Retransmission counters are reset and end frame transmission

Energy level during transmission, idleness, or collision



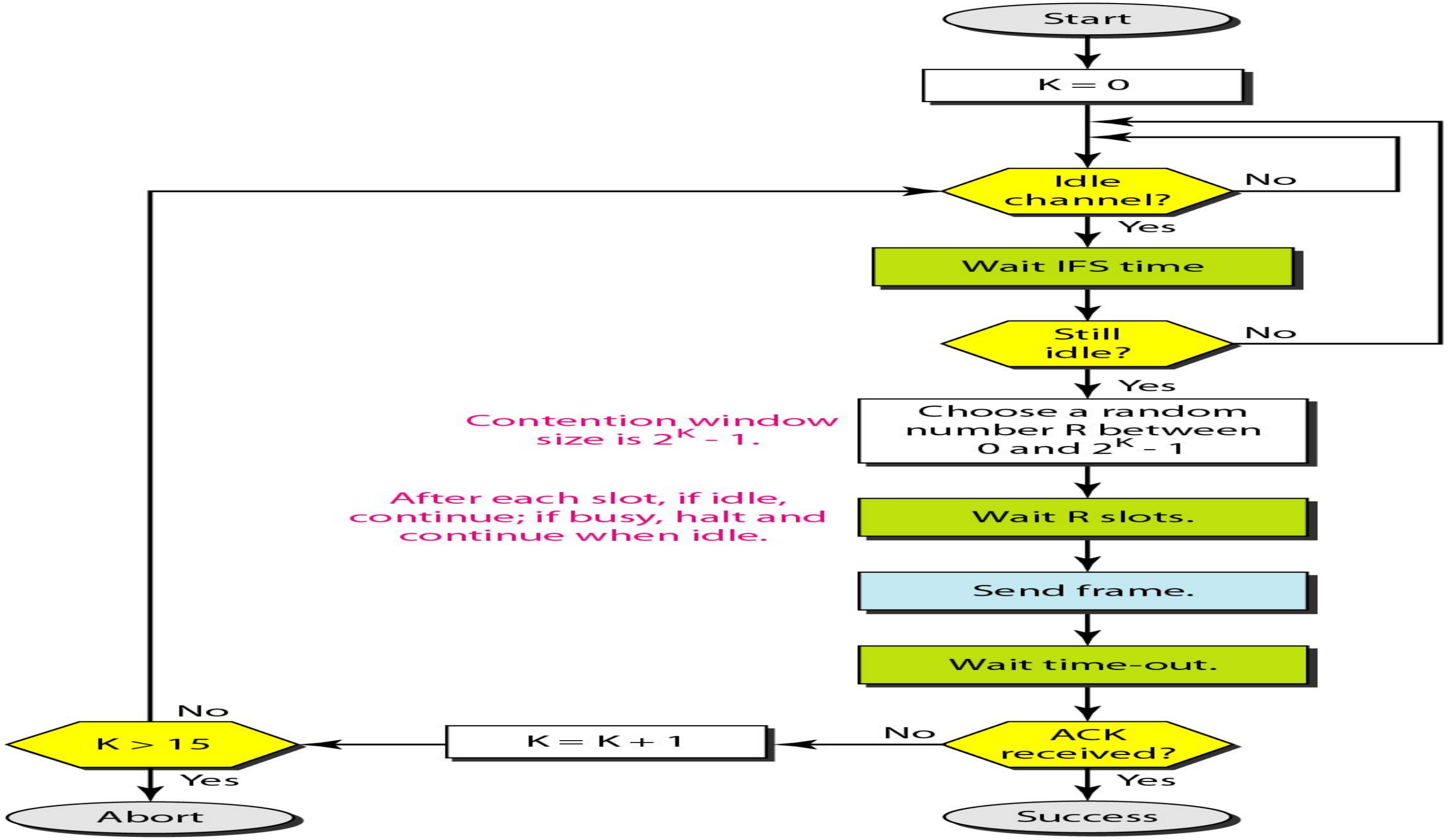
- level of energy in a channel can have three values
  - Zero: the channel is idle
  - Normal: The station has successfully captured the channel and is sending its frame
  - Abnormal: there is a collision and the level of the energy is twice the normal level
- station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy, or in collision mode

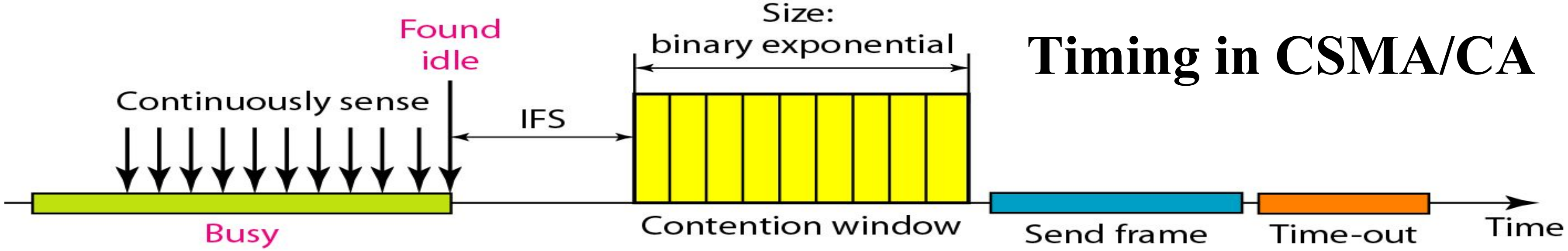
Collision detection: If the signal, the sender reads back, is different from the signal it is putting out, it knows that a collision is occurring

- In fact, a received signal must not be tiny compared to the transmitted signal
- In *CSMA/CD : a station needs to be able to receive while* transmitting to detect a collision
- When there is **no collision**, the station receives **one signal**: its own signal
- When there is **a collision**, the station receives **two signals**: its own signal and the signal transmitted by a second station
- To distinguish between these two cases, the received signals in these two cases must be significantly different

- In a wired network, the received signal has almost the same energy as the sent signal because
- So in a collision, the detected energy almost doubles
- Popular CSMA scheme and its variant CSMA/CD developed for wired networks
  - Can not be used directly in wireless networks

CSMA/Collision Avoidance


Start
K = 0
Idle channel?
No
Yes
Wait IFS time
Still idle?
No
Yes
Contention window size is 2^K - 1.
Choose a random number R between 0 and 2^K - 1
After each slot, if idle, continue; if busy, halt and continue when idle.
Wait R slots.
Send frame.
Wait time-out.
ACK received?
No
Yes
K = K + 1
K > 15
No
Yes
Abort
Success



Timing in CSMA/CA

CSMA/CD efficiency

- T_{prop} = max prop delay between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame

$$efficiency = \frac{1}{1 + 5t_{prop}/t_{trans}}$$

- efficiency goes to 1
  - as t_{prop} goes to 0
  - as t_{trans} goes to infinity
- better performance than ALOHA: and simple, cheap, decentralized!

"Taking turns" MAC protocols

channel partitioning MAC protocols:

- share channel *efficiently* and *fairly* at high load
- inefficient at low load: delay in channel access, 1/N bandwidth allocated even if only 1 active node!

random access MAC protocols

- efficient at low load: single node can fully utilize channel
- high load: collision overhead

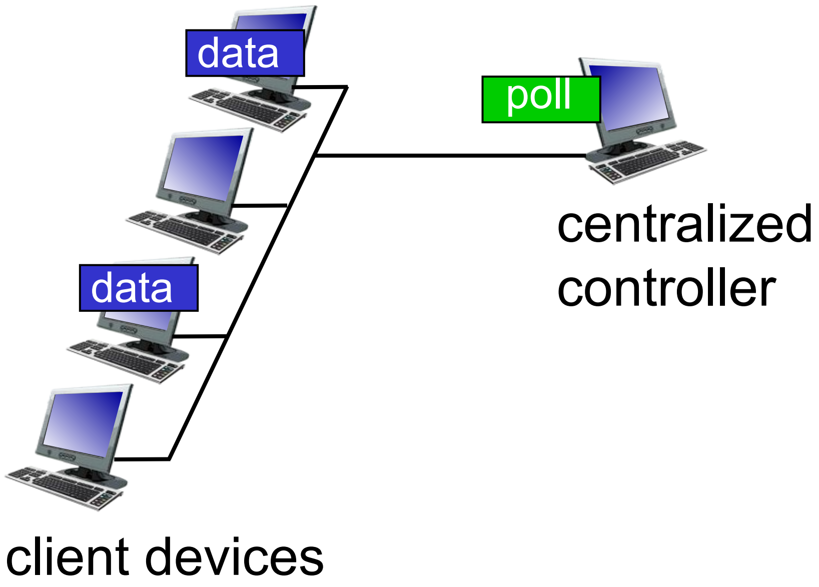
"taking turns" protocols

- look for best of both worlds!

“Taking turns” MAC protocols

polling:

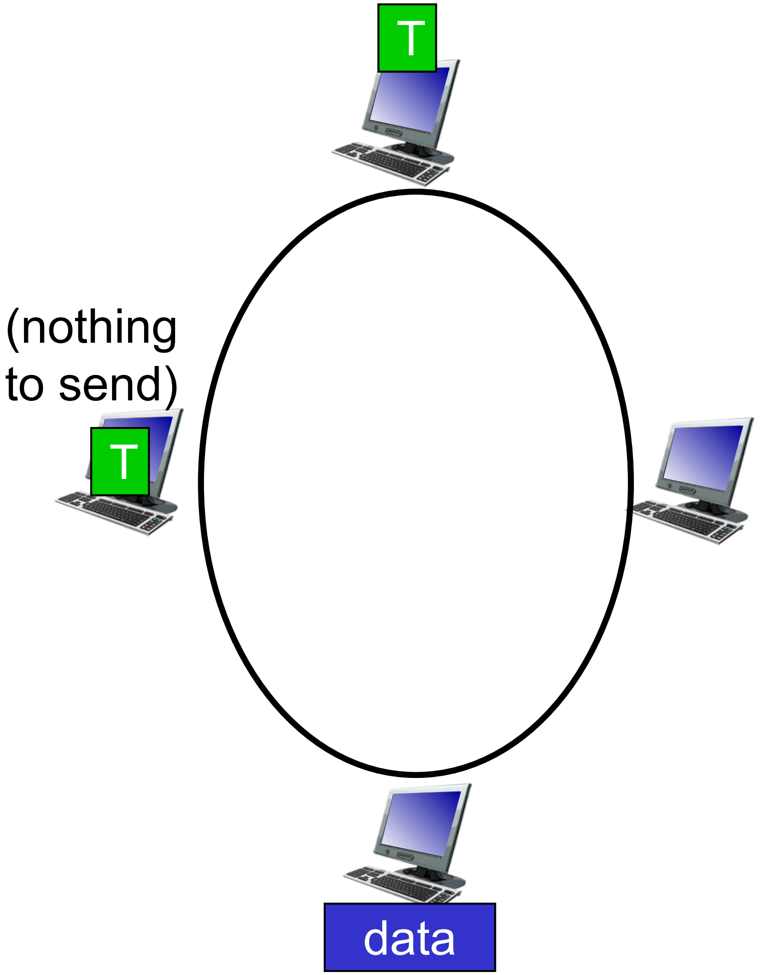
- centralized controller “invites” other nodes to transmit in turn
- typically used with “dumb” devices
- concerns:
  - polling overhead
  - latency
  - single point of failure (master)
- Bluetooth uses polling



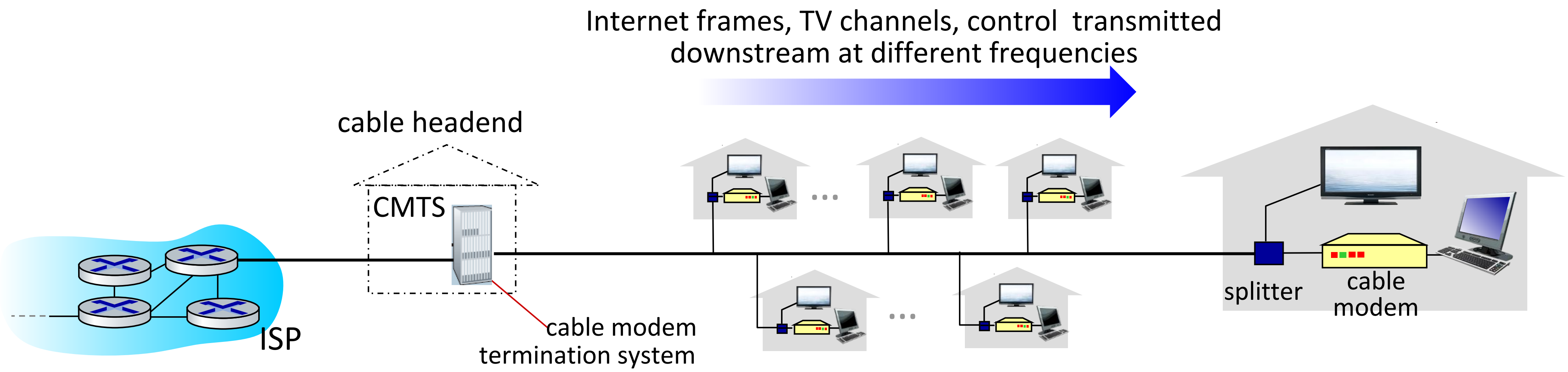
“Taking turns” MAC protocols

token passing:

- control *token* message explicitly passed from one node to next, sequentially
  - transmit while holding token
- FDDI: Fiber Distributed Data Interface, IEEE 802.5 token ring protocol
- concerns:
  - token overhead
  - latency
  - single point of failure (token)

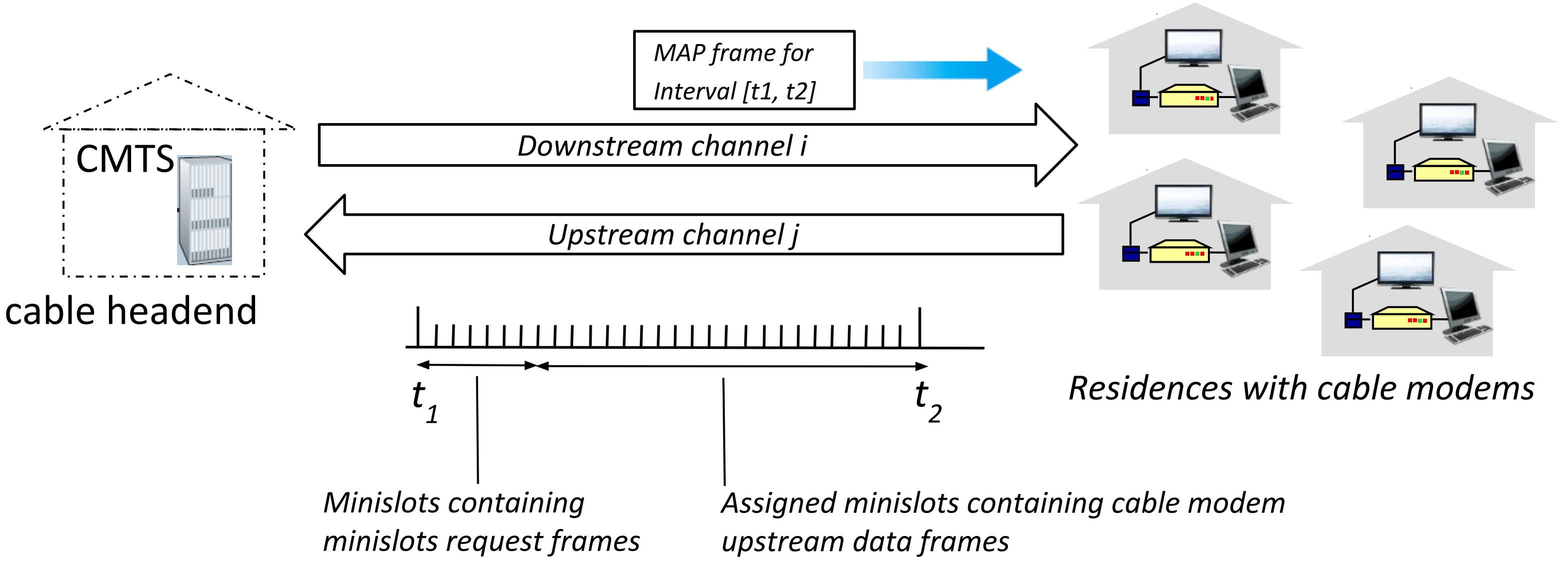


Cable access network: FDM, TDM *and* random access!


Internet frames, TV channels, control transmitted
downstream at different frequencies
cable headend
CMTS
ISP
cable modem
termination system
...
...
splitter
cable
modem

- multiple downstream (broadcast) FDM channels: up to 1.6 Gbps/channel
  - single CMTS transmits into channels
- multiple upstream channels (up to 1 Gbps/channel)
  - multiple access: all users contend (random access) for certain upstream channel time slots; others assigned TDM

Cable access network:



DOCSIS: Data Over Cable Service Interface Specification specifies cable data network architecture and protocols

- FDM over upstream (modem to CMTS), downstream (CMTS to modems) frequency channels (which are broadcast)
- TDM upstream: some slots assigned, some have contention
  - downstream MAP frame: assigns upstream slots
  - request for upstream slots (and data) transmitted random access (binary backoff) in selected slots

Summary of MAC protocols

- channel partitioning, by time, frequency or code
  - TDMA, FDMA
- random access (dynamic),
  - ALOHA, S-ALOHA, CSMA, CSMA/CD
  - carrier sensing: easy in some technologies (wire), hard in others (wireless)
  - used in satellite, Ethernet, WiFi
- taking turns
  - polling from central site, token passing
  - Bluetooth (token ring)