Link Layer

(continued)

Topics

1. Framing
   - Delimiting start/end of frames
2. Error detection and correction
   - Handling errors
3. Multiple Access
   - 802.11, classic Ethernet
4. Switching
   - Modern Ethernet

Detection vs. Correction

- Which is better will depend on the pattern of errors. For example:
  - 1000 bit messages with a bit error rate (BER) of 1 in 10000
- Which has less overhead?

Detection vs. Correction

- Which is better will depend on the pattern of errors. For example:
  - 1000 bit messages with a bit error rate (BER) of 1 in 10000
- Which has less overhead?
  - It still depends! We need to know more about the errors

Detection vs. Correction (2)

Assume bit errors are random

- Messages have 0 or maybe 1 error (1/10 of the time)

Error correction:

- Need ~10 check bits per message
- Overhead:

Error detection:

- Need ~1 check bits per message plus 1000 bit retransmission
- Overhead:

Detection vs. Correction (3)

Assume errors come in bursts of 100

- Only 1 or 2 messages in 1000 have significant (multi-bit) errors

Error correction:

- Need >>100 check bits per message
- Overhead:

Error detection:

- Need 32 check bits per message plus 1000 bit resend 2/1000 of the time
- Overhead:

Detection vs. Correction (4)

- Error correction:
  - Needed when errors are expected
  - Or when no time for retransmission
- Error detection:
  - More efficient when errors are not expected
  - And when errors are large when they do occur

Error Correction in Practice

- Heavily used in physical layer
  - Convolutional codes widely used in practice
- Error detection (w/ retransmission) is used in the link layer and above for residual errors
- Correction also used in the application layer
  - Called Forward Error Correction (FEC)
  - Normally with an erasure error model
  - E.g., Reed-Solomon (CDs, DVDs, etc.)

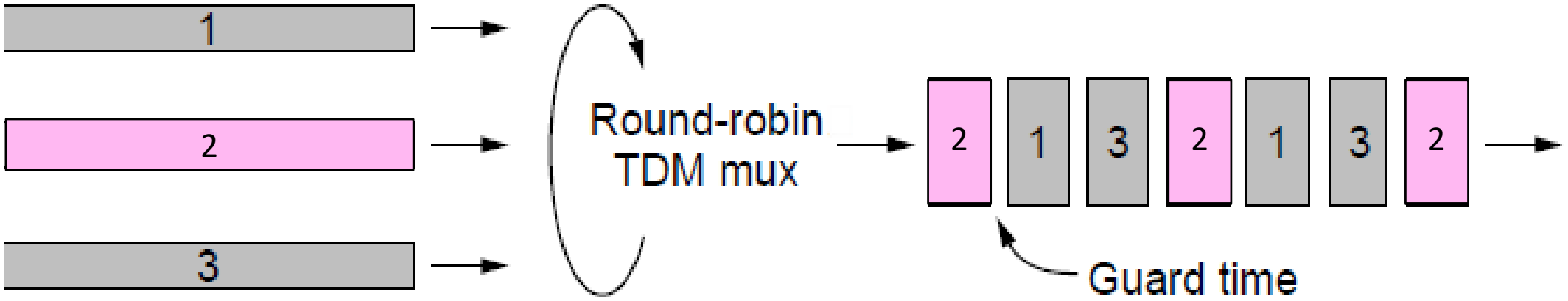
Multiple Access

Topic

- Multiplexing is the network word for the sharing of a resource
- Classic scenario is sharing a link among different users
  - Time Division Multiplexing (TDM)
  - Frequency Division Multiplexing (FDM)

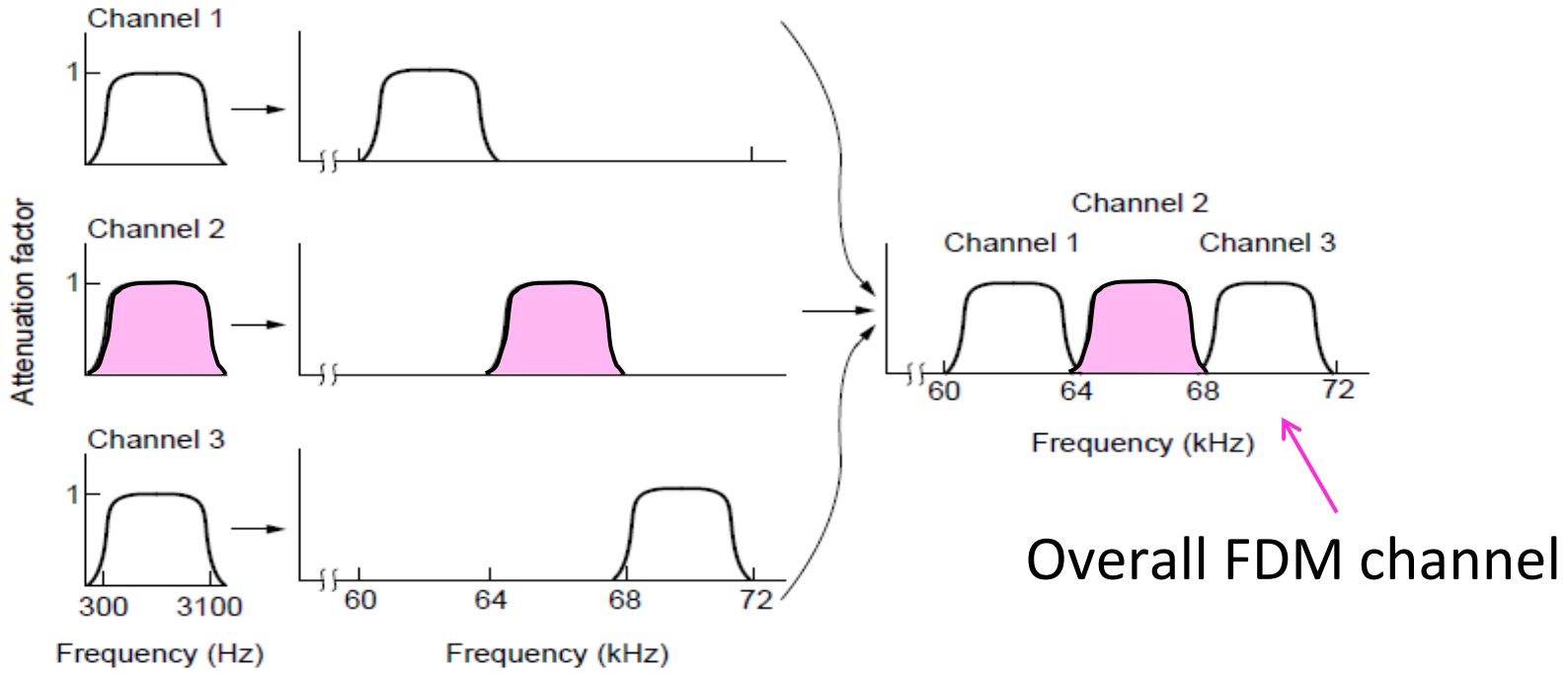
Time Division Multiplexing (TDM)

- Users take turns on a fixed schedule



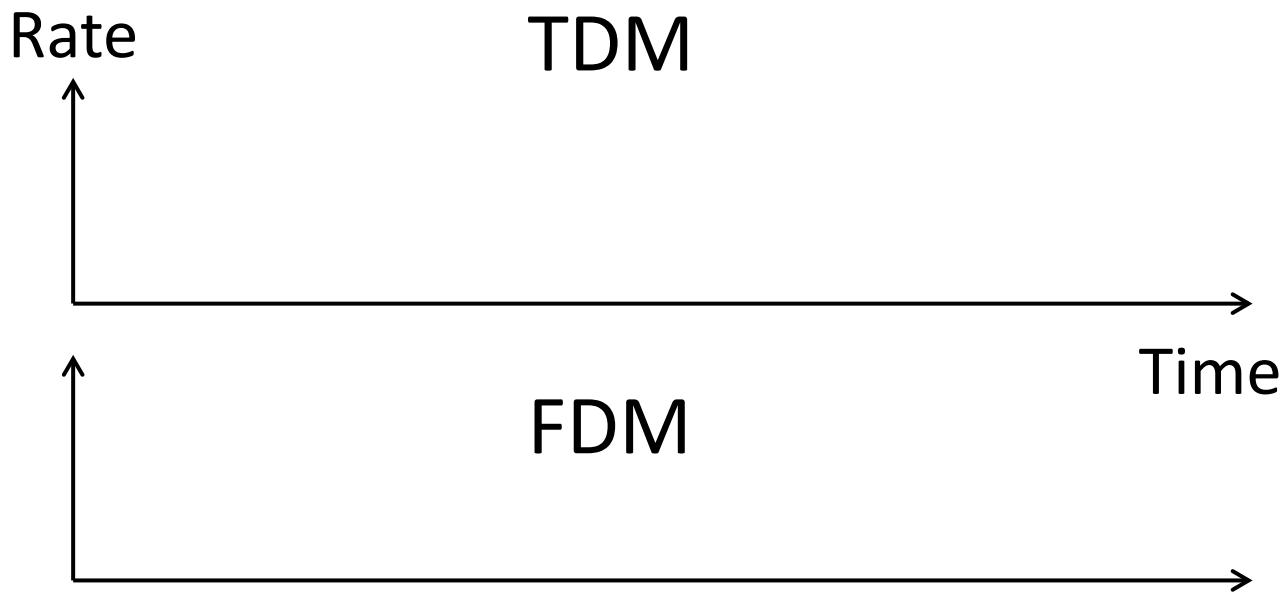
Frequency Division Multiplexing (FDM)

- Put different users on different frequency bands



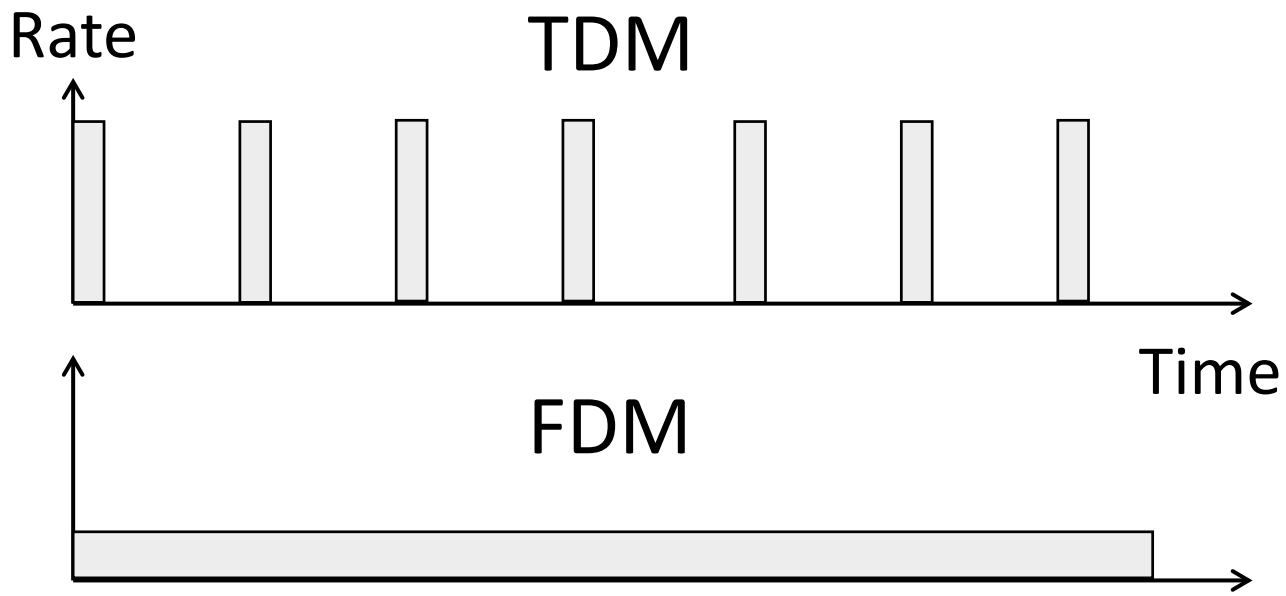
TDM versus FDM

- In TDM a user sends at a high rate a fraction of the time; in FDM, a user sends at a low rate all the time



TDM versus FDM (2)

- In TDM a user sends at a high rate a fraction of the time; in FDM, a user sends at a low rate all the time

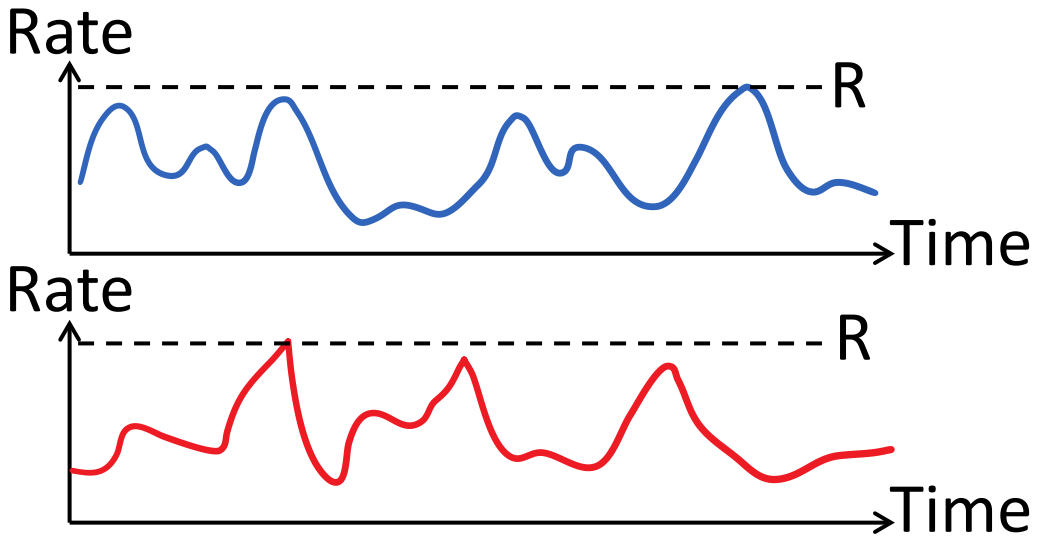


TDM/FDM Usage

- Statically divide a resource
  - Suited for continuous traffic, fixed number of users
- Widely used in telecommunications
  - TV and radio stations (FDM)
  - GSM (2G cellular) allocates calls using TDM within FDM

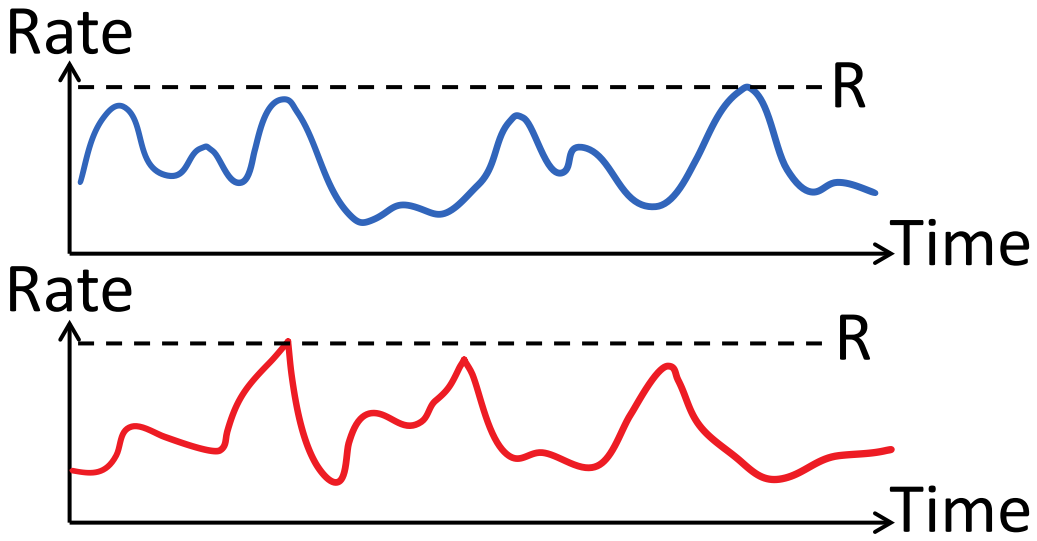
Multiplexing Network Traffic

- Network traffic is bursty
  - ON/OFF sources
  - Load varies greatly over time



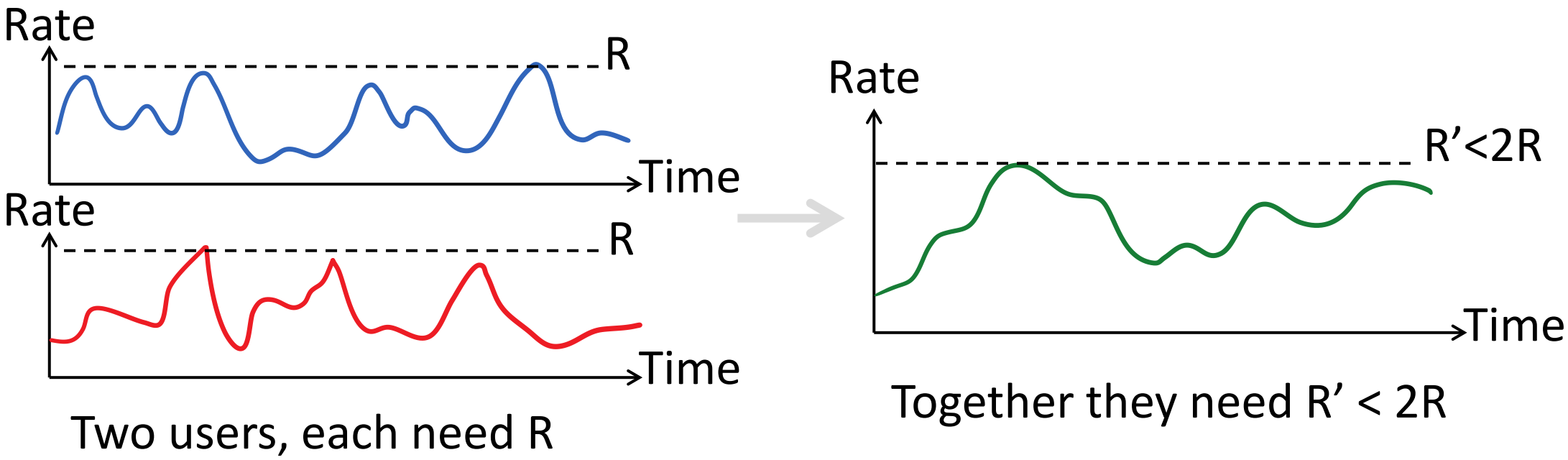
Multiplexing Network Traffic (2)

- Network traffic is bursty
  - Inefficient to always allocate user their ON needs with TDM/FDM



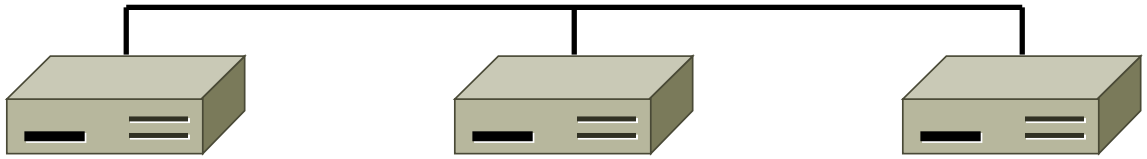
Multiplexing Network Traffic (3)

- Multiple access schemes multiplex users according to demands – for gains of statistical multiplexing



Random Access

- How do nodes share a single link? Who sends when, e.g., in WiFI?
  - Explore with a simple model



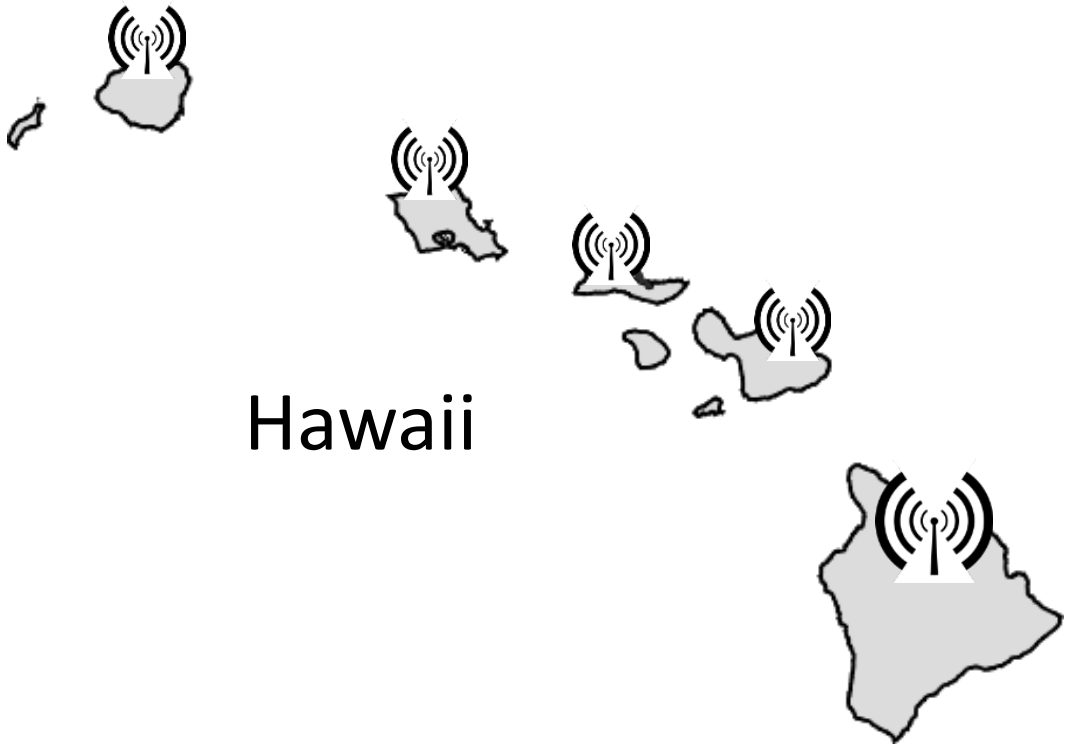
- Assume no-one is in charge
  - Distributed system

Random Access (2)

- We will explore random multiple access control (MAC) protocols
  - This is the basis for classic Ethernet
  - Remember: data traffic is bursty

ALOHA Network

- Seminal computer network connecting the Hawaiian islands in the late 1960s
  - When should nodes send?
  - A new protocol was devised by Norm Abramson …

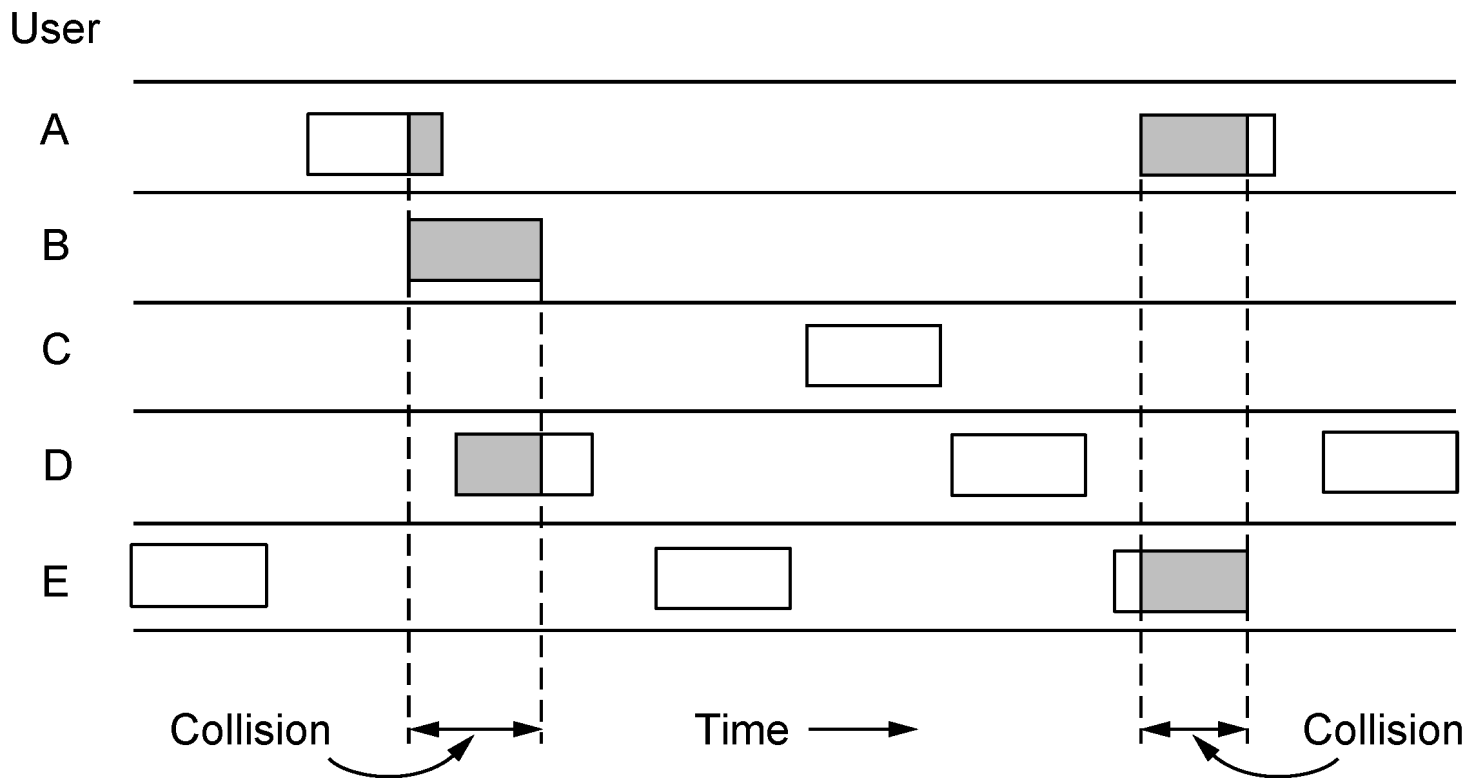


ALOHA Protocol

- Simple idea:
  - Node just sends when it has traffic.
  - If there was a collision (no ACK received) then wait a random time and resend
- That's it!

ALOHA Protocol (2)

- Some frames will be lost, but many may get through…
- Good idea?

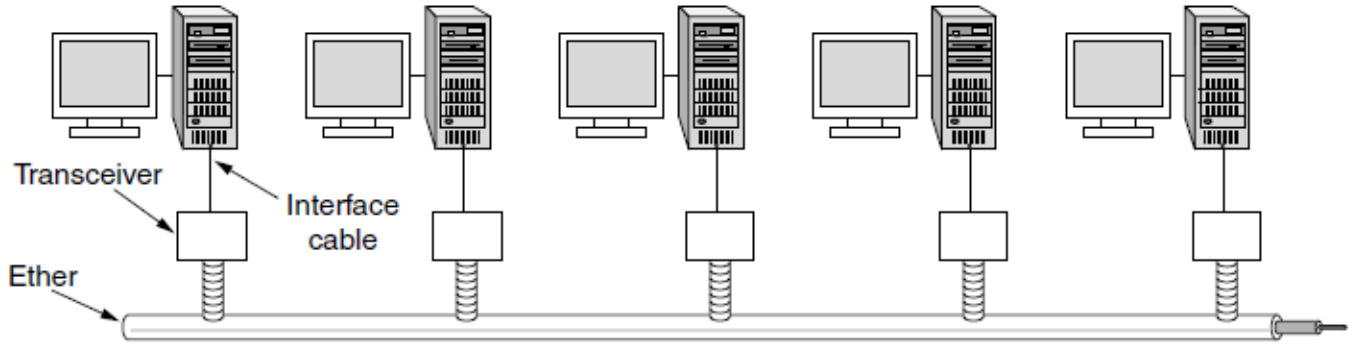


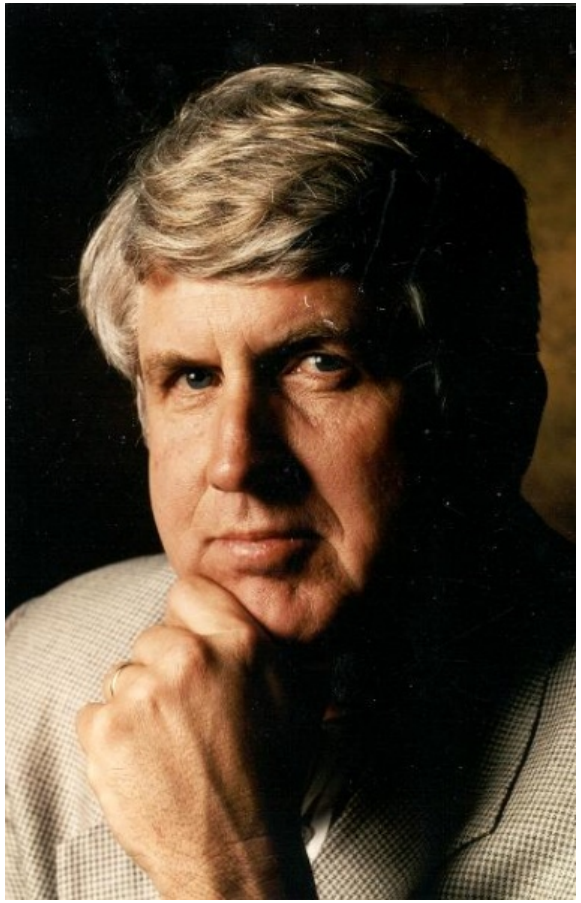
ALOHA Protocol (3)

- Simple, decentralized protocol that works well under low load!
- Not efficient under high load
  - Analysis shows at most 18% efficiency
  - Improvement: divide time into slots and efficiency goes up to 36%
- We'll look at other improvements

Classic Ethernet

- ALOHA inspired Bob Metcalfe to invent Ethernet for LANs in 1973
  - Nodes share 10 Mbps coaxial cable
  - Hugely popular in 1980s, 1990s





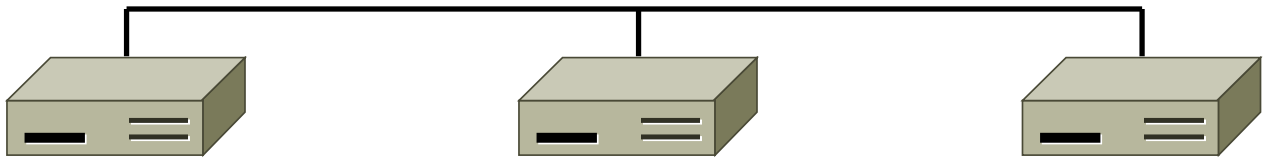
: © 2009 IEEE

CSMA (Carrier Sense Multiple Access)

- Improve ALOHA by listening for activity before we send (Doh!)
  - Can do easily with wires, not wireless
    - Why not with wireless?
- So does this eliminate collisions?
  - Why or why not?

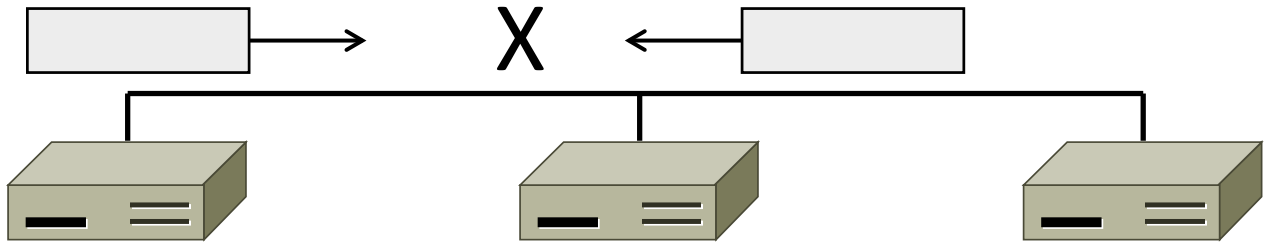
CSMA (2)

- Still possible to listen and hear nothing when another node is sending because of delay



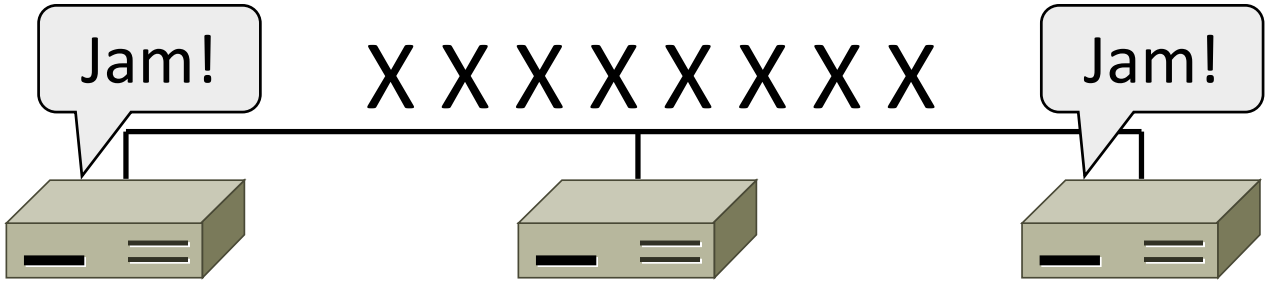
CSMA (3)

- CSMA is a good defense against collisions only when transmitting for periods much longer than lantency



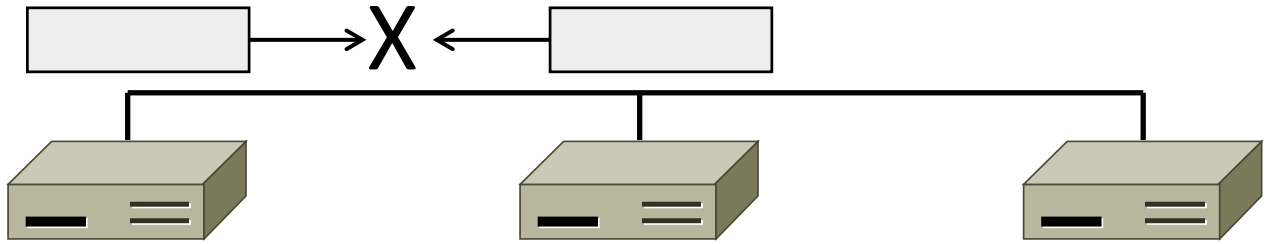
CSMA/CD (with Collision Detection)

- Can reduce the cost of collisions by detecting them and aborting (Jam) the rest of the frame time
  - Again, we can do this with wires, not so easy with wireless



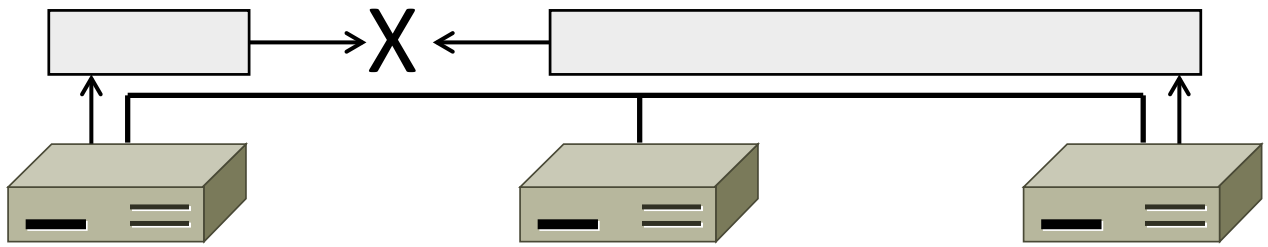
CSMA/CD Complications

- Everyone who collides needs to know it happened
  - Retransmissions occur only if sender knows there was a collision
    - There are no ACKs
- Time window in which a node may hear of a collision is 2D seconds



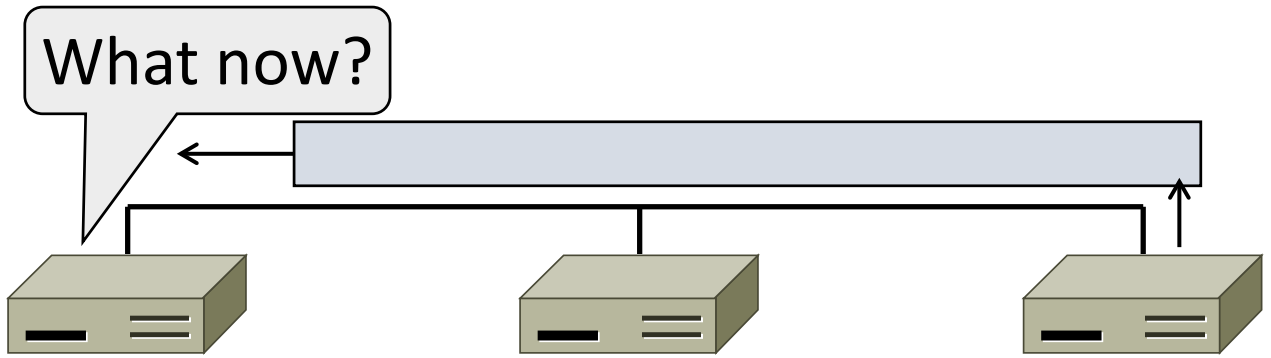
CSMA/CD Complications (2)

- Impose a minimum frame length corresponding to 2D seconds
  - So node can't finish before collision
  - Ethernet minimum frame is 64 bytes



Persistence of CSMA

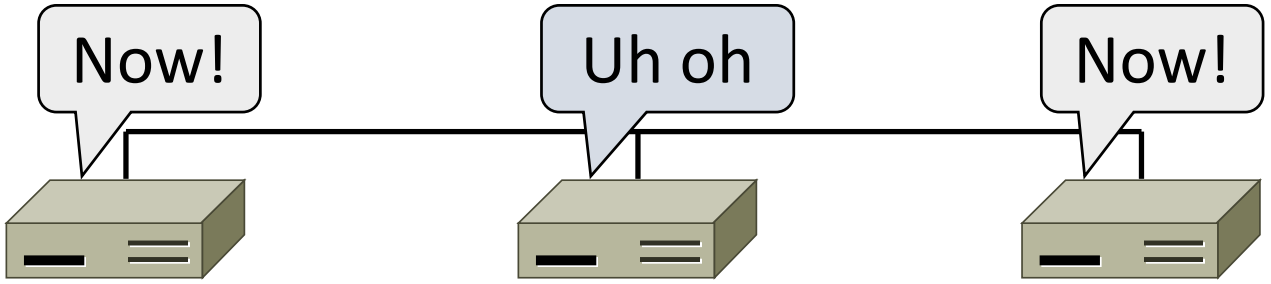
- What should a node do if another node is sending?



- Idea: Wait until it is done, and send

Persistence of CSMA (2)

- Problem is that deferring due to carrier sense synchronizes the deferring nodes
  - multiple waiting nodes will queue up then collide
  - More load, more of a problem



Persistence of CSMA(3)

- Intuition for a better solution
  - We want just one of the N waiting senders to decide to send when the current transmission ends
  - If there are N queued senders, we want each to send next with probability 1/N