Computer Networks

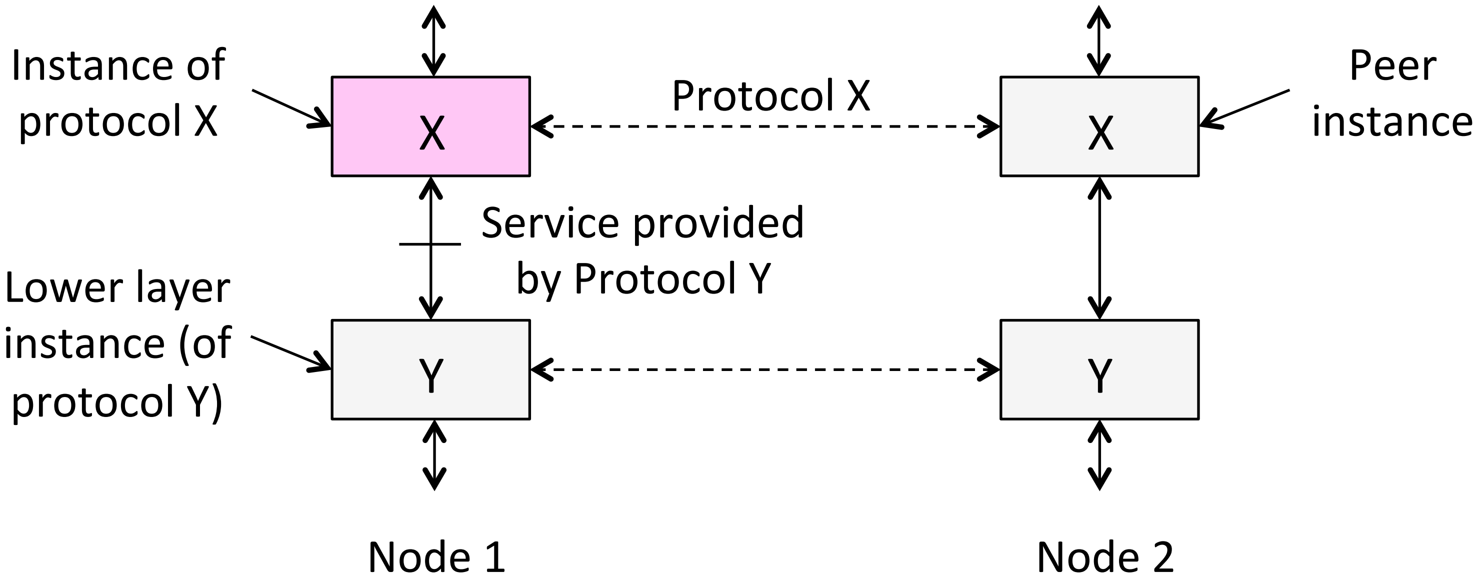
Shyam Gollakota

Protocols and Layers

- Protocols and layering is the main structuring method used to divide up network functionality
  - Each instance of a protocol talks virtually to its peer using the protocol
  - Each instance of a protocol uses only the services of the lower layer

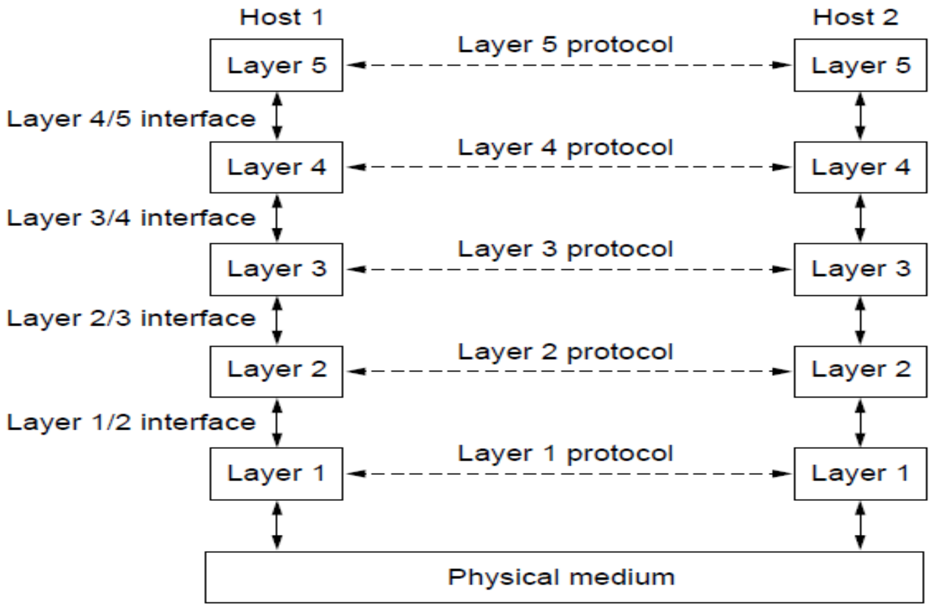
Protocols and Layers (3)

- Protocols are horizontal, layers are vertical



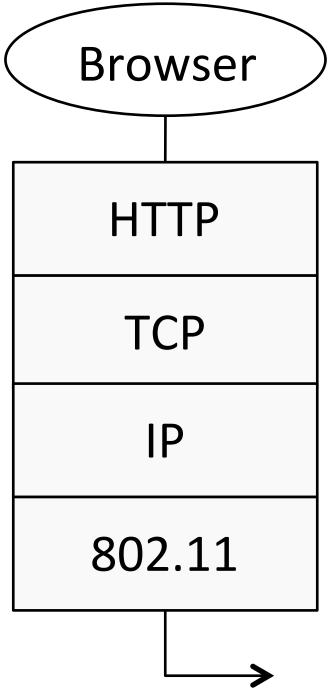
Protocols and Layers (4)

- Set of protocols in use is called a <u>protocol stack</u>



Protocols and Layers (6)

- Protocols you've probably heard of:
  - TCP, IP, 802.11, Ethernet, HTTP, SSL, DNS, … and many more
- An example protocol stack
  - Used by a web browser on a host that is wirelessly connected to the Internet

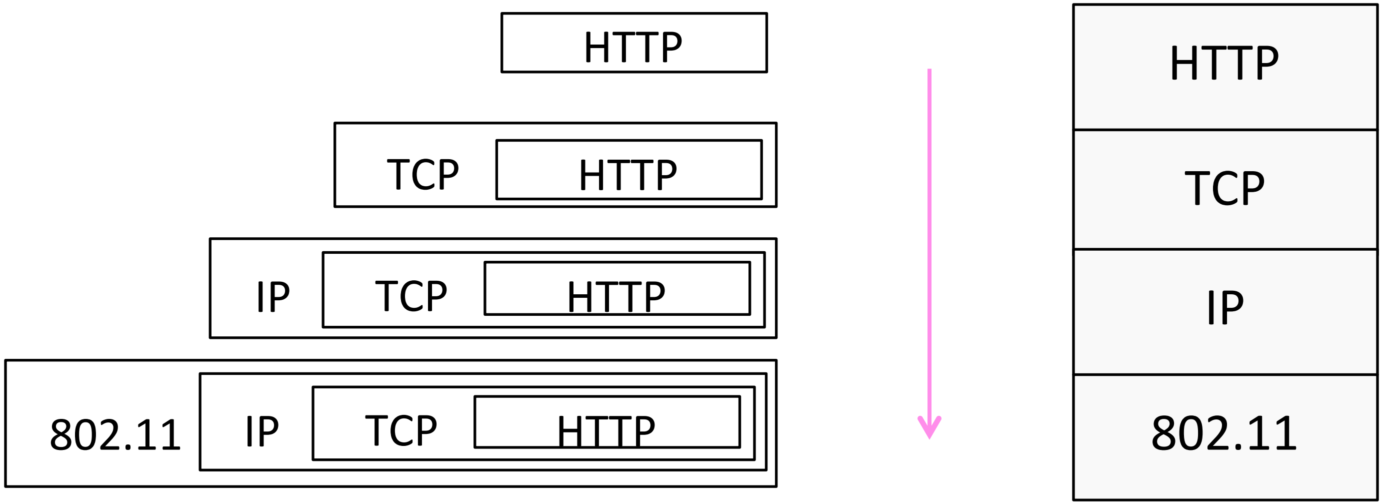


Encapsulation

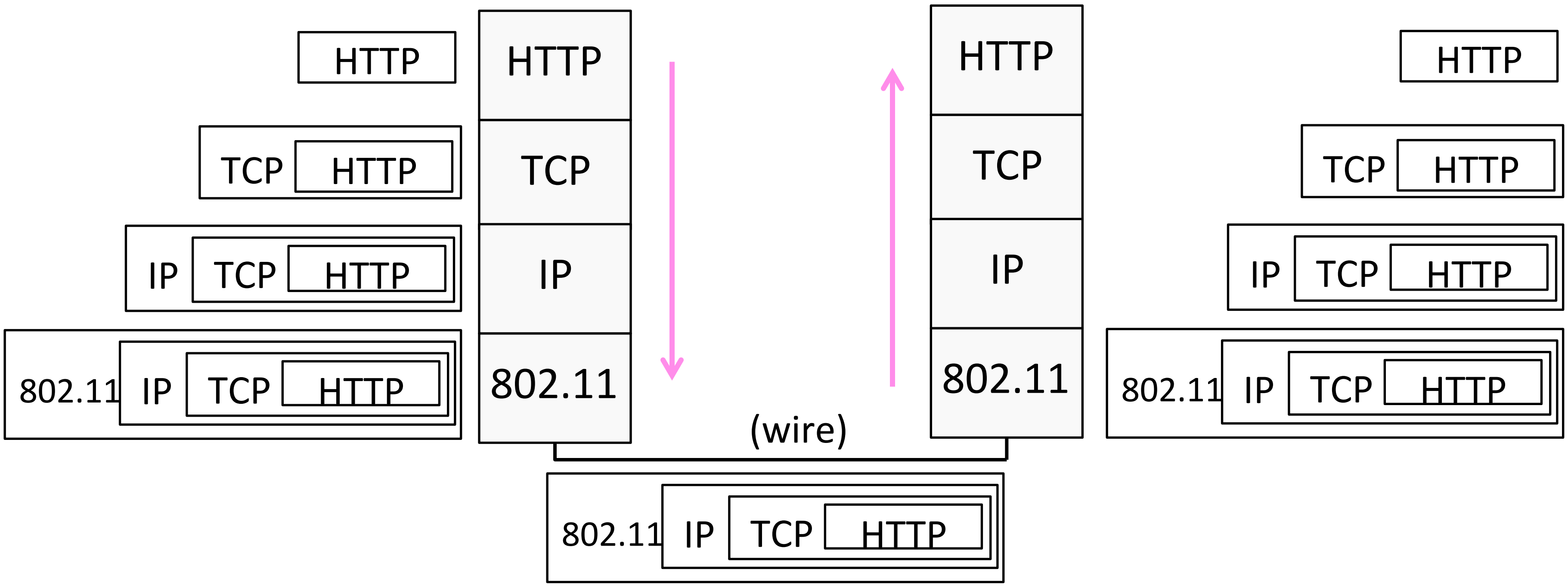
- Encapsulation is the mechanism used to effect protocol layering
  - Lower layer wraps higher layer content, adding its own information to make a new message for delivery
  - Like sending a letter in an envelope; postal service doesn't look inside

Encapsulation (3)

- Message "on the wire" begins to look like an onion
  - Lower layers are outermost

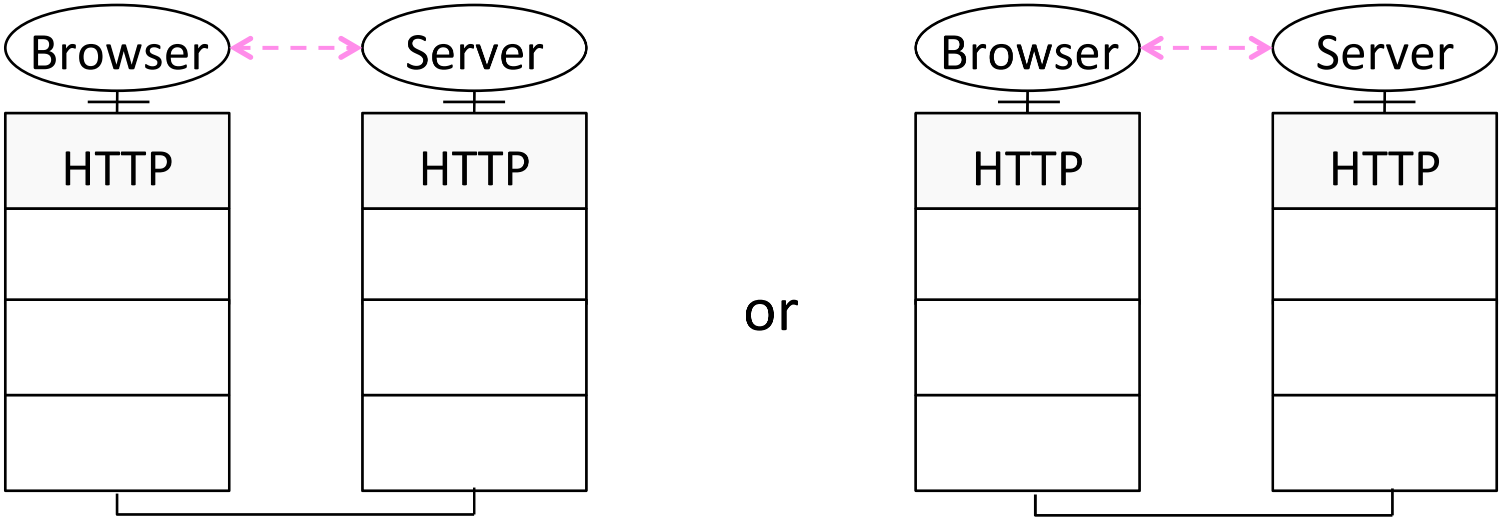


Encapsulation (4)



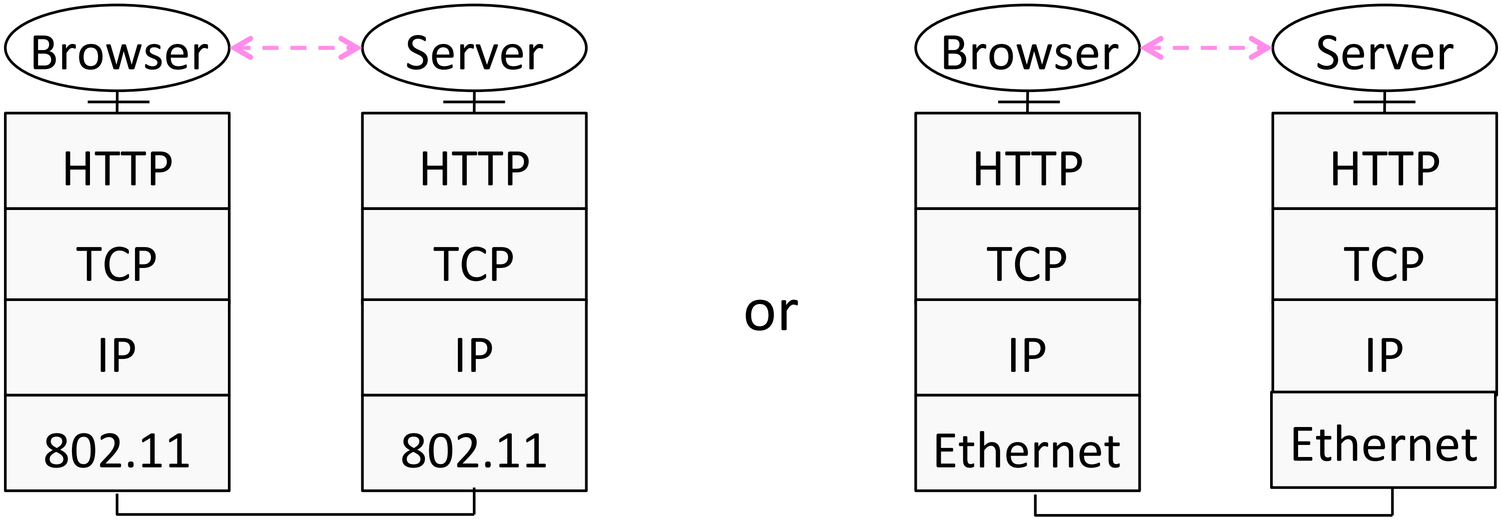
Advantage of Layering

- Information hiding and reuse



Advantage of Layering (2)

- Information hiding and reuse

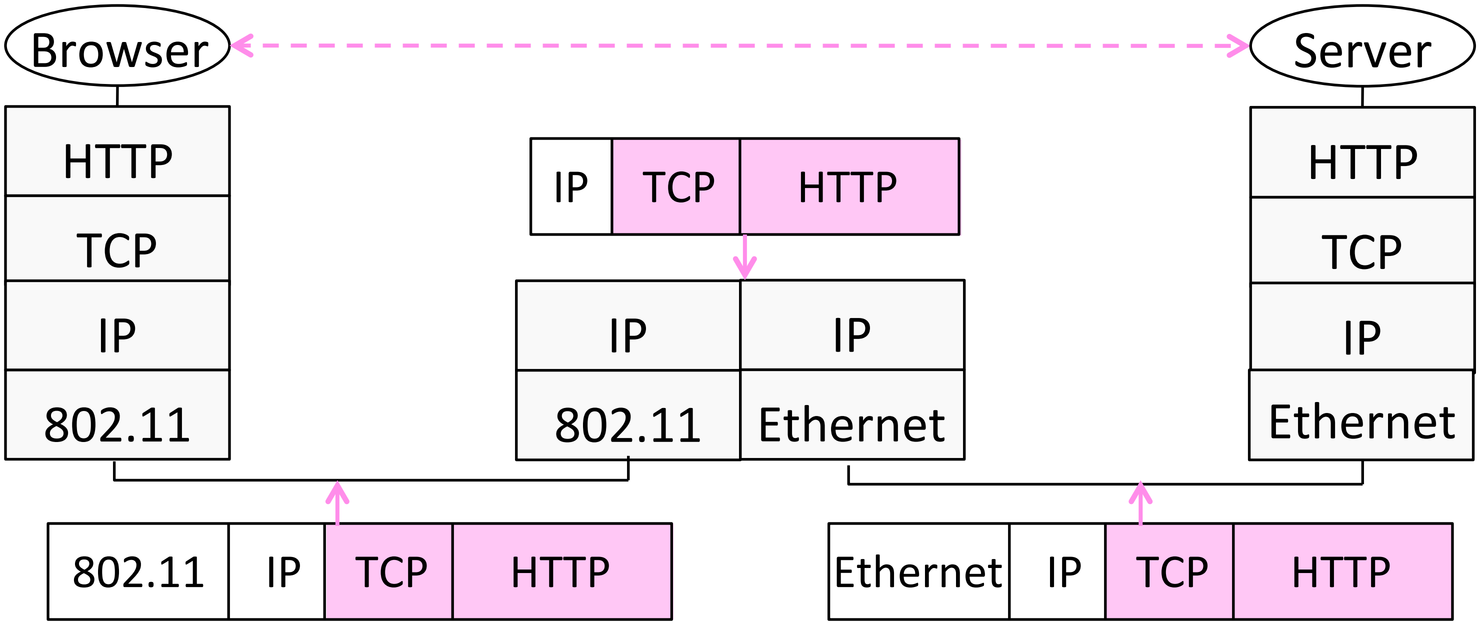


Advantage of Layering (3)

- Using information hiding to connect different systems

Advantage of Layering (4)

- Using information hiding to connect different systems



Disadvantage of Layering

- ??

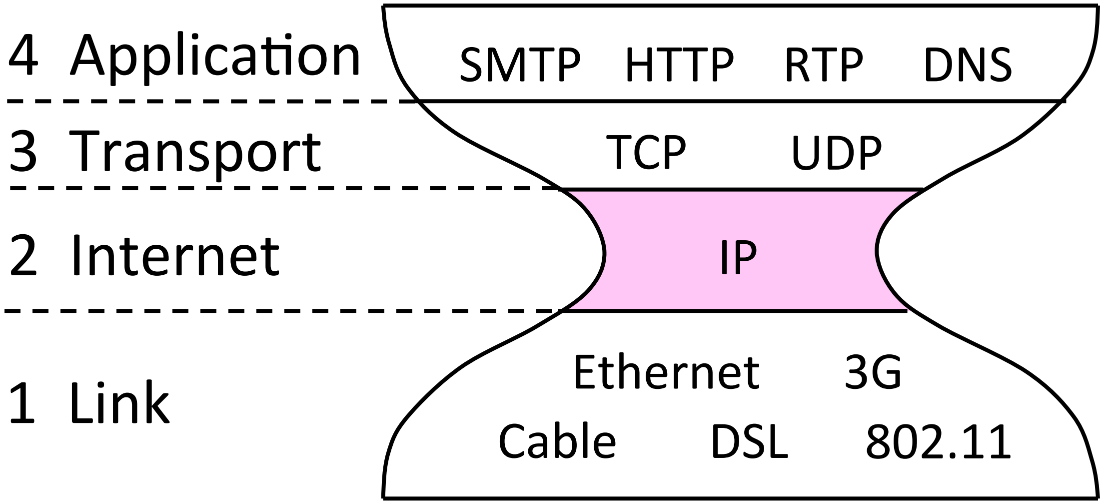
Internet Reference Model

- A four layer model based on experience; omits some OSI layers and uses IP as the network layer.

4	Application	– Programs that use network service
3	Transport	– Provides end-to-end data delivery
2	Internet	– Send packets over multiple networks
1	Link	– Send frames over a link

Internet Reference Model (3)

- IP is the "narrow waist" of the Internet
  - Supports many different links below and apps above



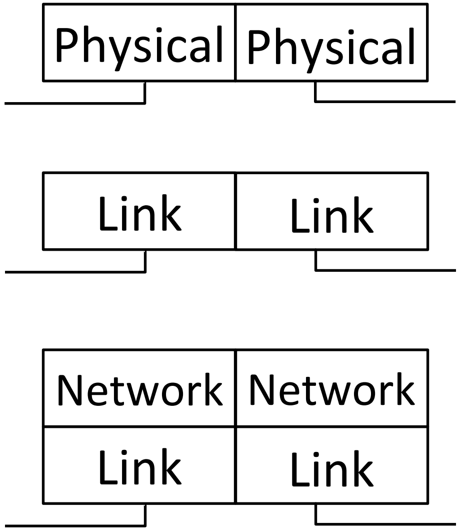
Layer-based Names (2)

- For devices in the network:

Repeater (or hub)

Switch (or bridge)

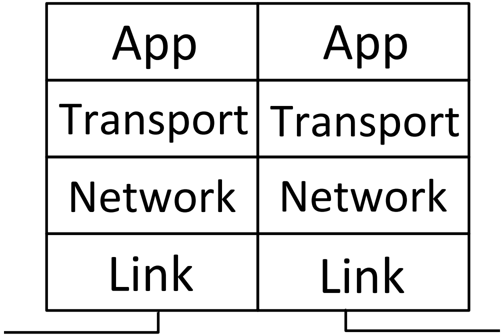
Router



Layer-based Names (3)

- For devices in the network:

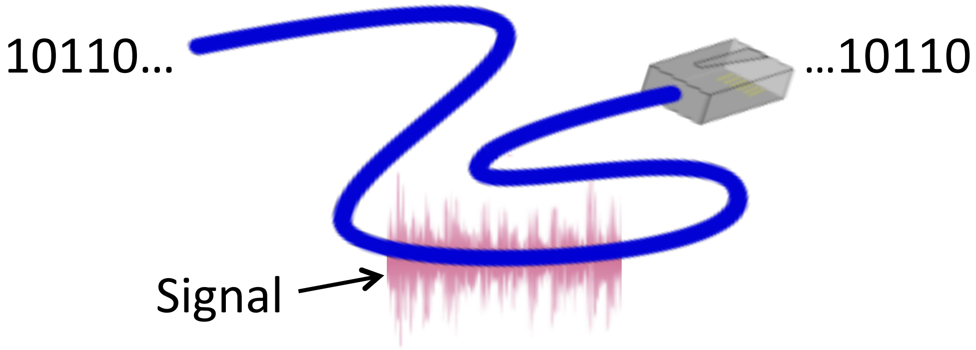
Proxy or middlebox or gateway



But they all look like this!

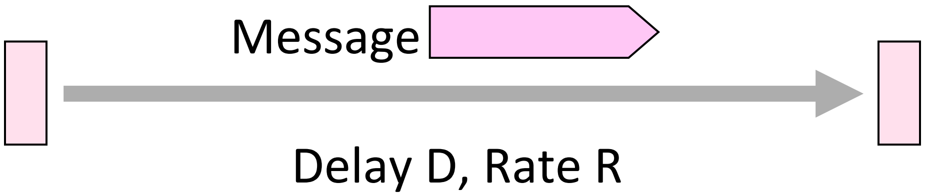
Scope of the Physical Layer

- Concerns how signals are used to transfer message bits over a link
  - Wires etc. carry analog signals
  - We want to send digital bits



Simple Link Model

- We'll end with an abstraction of a physical channel
  - Rate (or bandwidth, capacity, speed) in bits/second
  - Delay in seconds, related to length



- Other important properties:
  - Whether the channel is broadcast, and its error rate

Message Latency

- Latency is the delay to send a message over a link
  - Transmission delay: time to put M-bit message "on the wire"
  - Propagation delay: time for bits to propagate across the wire
  - Combining the two terms we have:

Message Latency (2)

- <u>Latency</u> is the delay to send a message over a link
  - <u>Transmission delay</u>: time to put M-bit message "on the wire"

    T-delay = M (bits) / Rate (bits/sec) = M/R seconds

  - <u>Propagation delay</u>: time for bits to propagate across the wire

    P-delay = Length / speed of signals = Length / ⅔c = D seconds

  - Combining the two terms we have: L = M/R + D

Metric Units

- The main prefixes we use:

Prefix	Exp.	prefix	exp.
K(ilo)	10^3	m(illi)	10^{-3}
M(ega)	10^6	μ(micro)	10^{-6}
G(iga)	10^9	n(ano)	10^{-9}

- Use powers of 10 for rates, 2 for storage
  - 1 Mbps = 1,000,000 bps, 1 KB = 2^{10} bytes
- "B" is for bytes, "b" is for bits

Latency Examples (2)

- "Dialup" with a telephone modem:

  D = 5 ms, R = 56 kbps, M = 1250 bytes

  L = 5 ms + (1250x8)/(56 x 10^3) sec = 184 ms!

- Broadband cross-country link:

  D = 50 ms, R = 10 Mbps, M = 1250 bytes

  L = 50 ms + (1250x8) / (10 x 10^6) sec = 51 ms

- A long link or a slow rate means high latency
  - Often, one delay component dominates

Bandwidth-Delay Product

- Messages take space on the wire!

- The amount of data in flight is the <u>bandwidth-delay (BD) product</u>

$$BD = R \times D$$

  - Measure in bits, or in messages
  - Small for LANs, big for “long fat” pipes

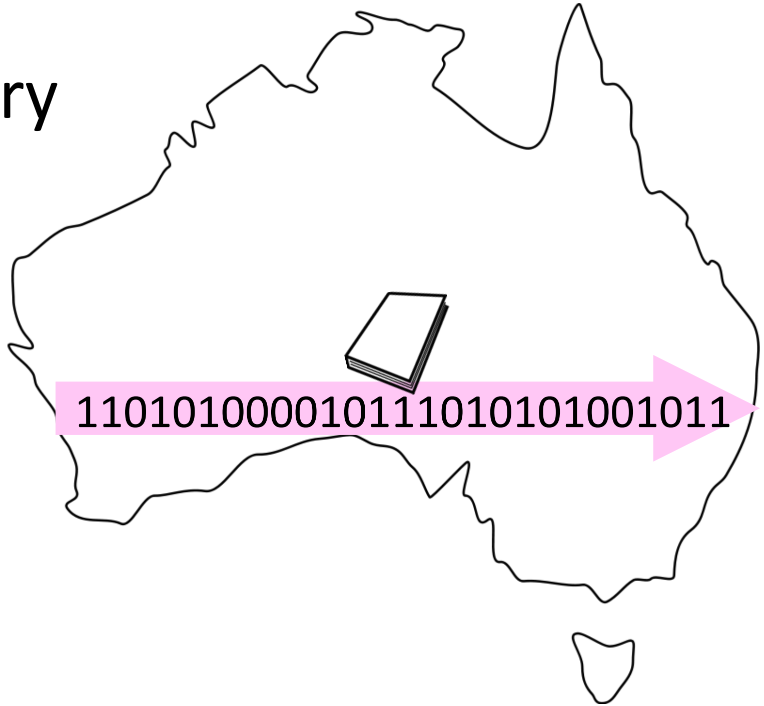
Bandwidth-Delay Example (2)

- Fiber at home, cross-country

  R=40 Mbps, D=50 ms

  BD $= 40 \times 10^{6} \times 50 \times 10^{-3}$ bits
  $= 2000$ Kbit
  $= 250$ KB

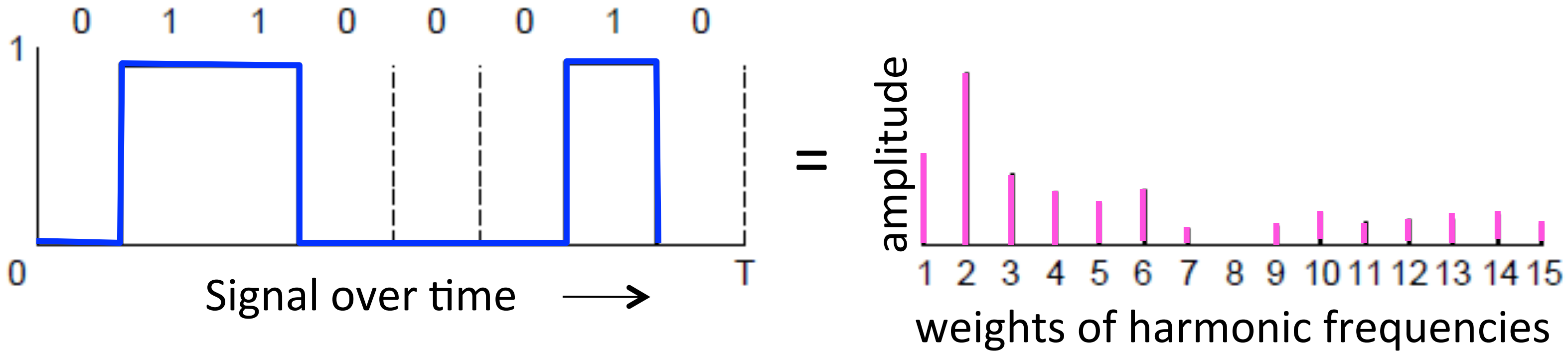
- That's quite a lot of data "in the network"!



Frequency Representation

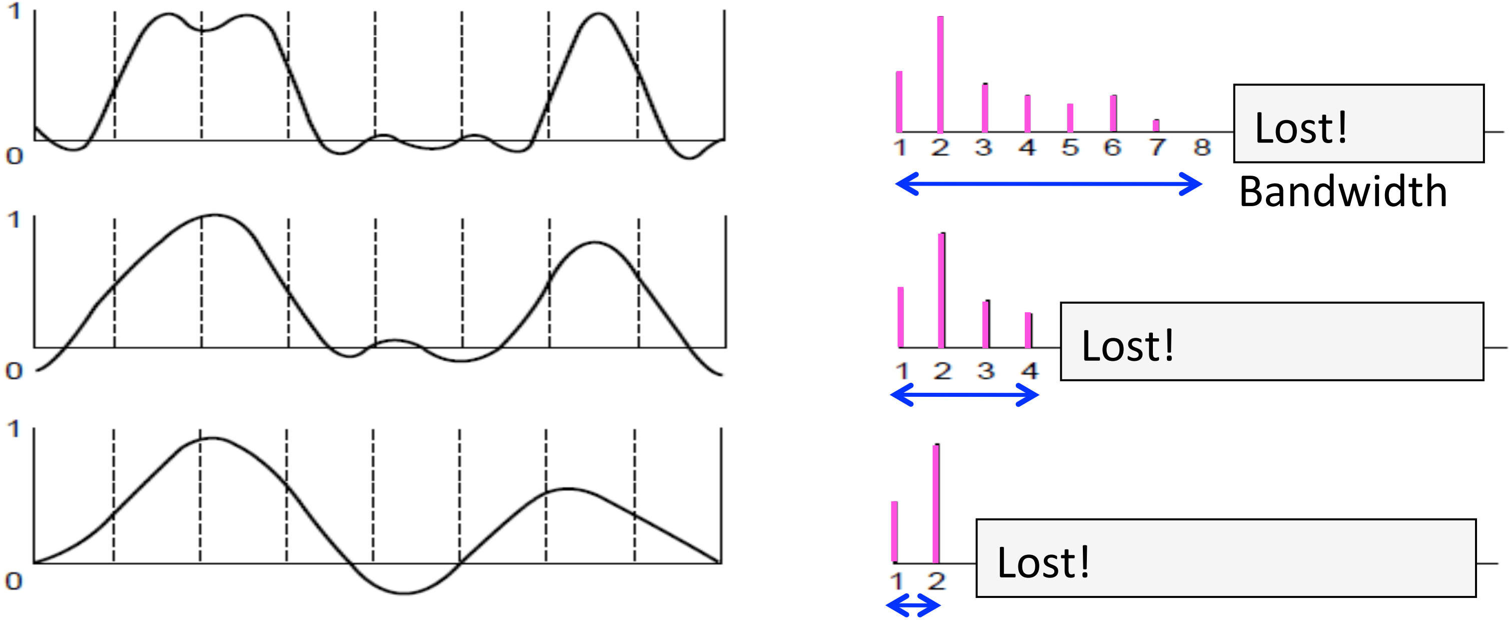
- A signal over time can be represented by its frequency components (called Fourier analysis)

$$g(t) = \frac{1}{2}c + \sum_{n=1}^{\infty} a_n \sin(2\pi n f t) + \sum_{n=1}^{\infty} b_n \cos(2\pi n f t)$$



Effect of Less Bandwidth

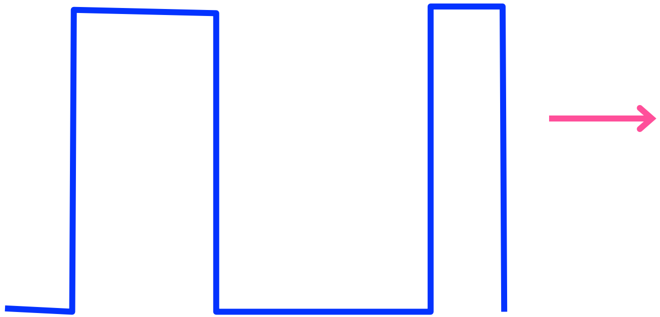
- Fewer frequencies (=less bandwidth) degrades signal



Signals over a Wire (2)

- Example:

Sent signal



2: Attenuation:

3: Bandwidth:

4: Noise:

Signals over Wireless

- Signals transmitted on a carrier frequency, like fiber
- Travel at speed of light, spread out and attenuate faster than $1/dist^2$
- Multiple signals on the same frequency interfere at a receiver

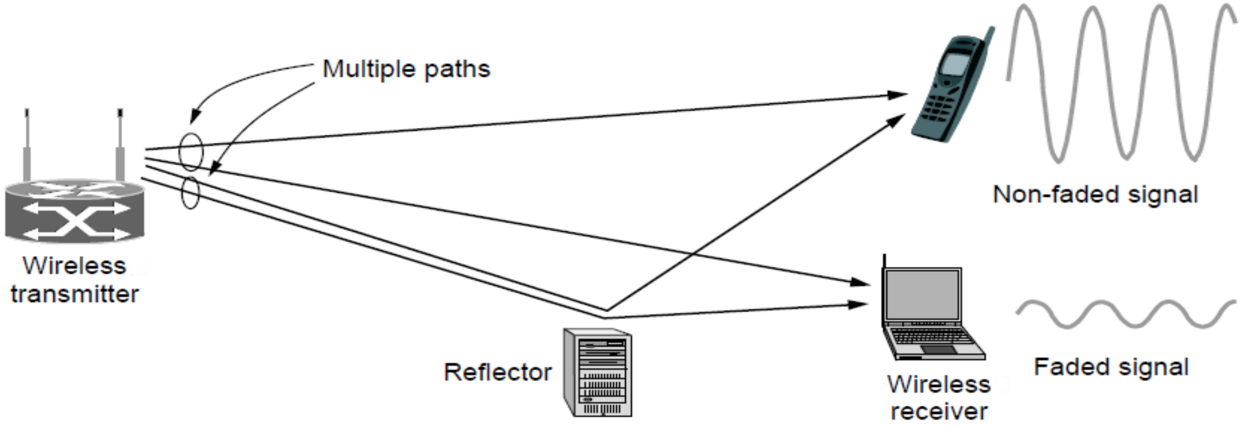
Signals over Wireless (5)

- Various other effects too!
  - Wireless propagation is complex, depends on environment

- Some key effects are highly frequency dependent,
  - E.g., multipath at microwave frequencies

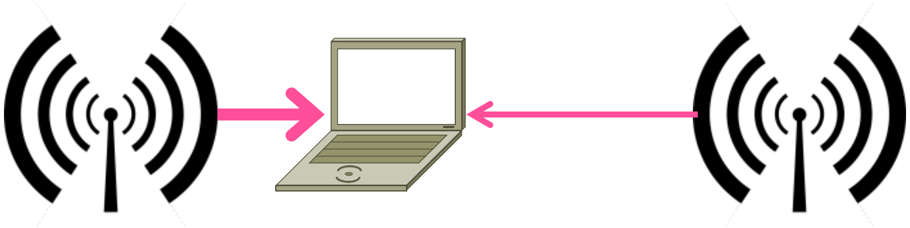
Wireless Multipath

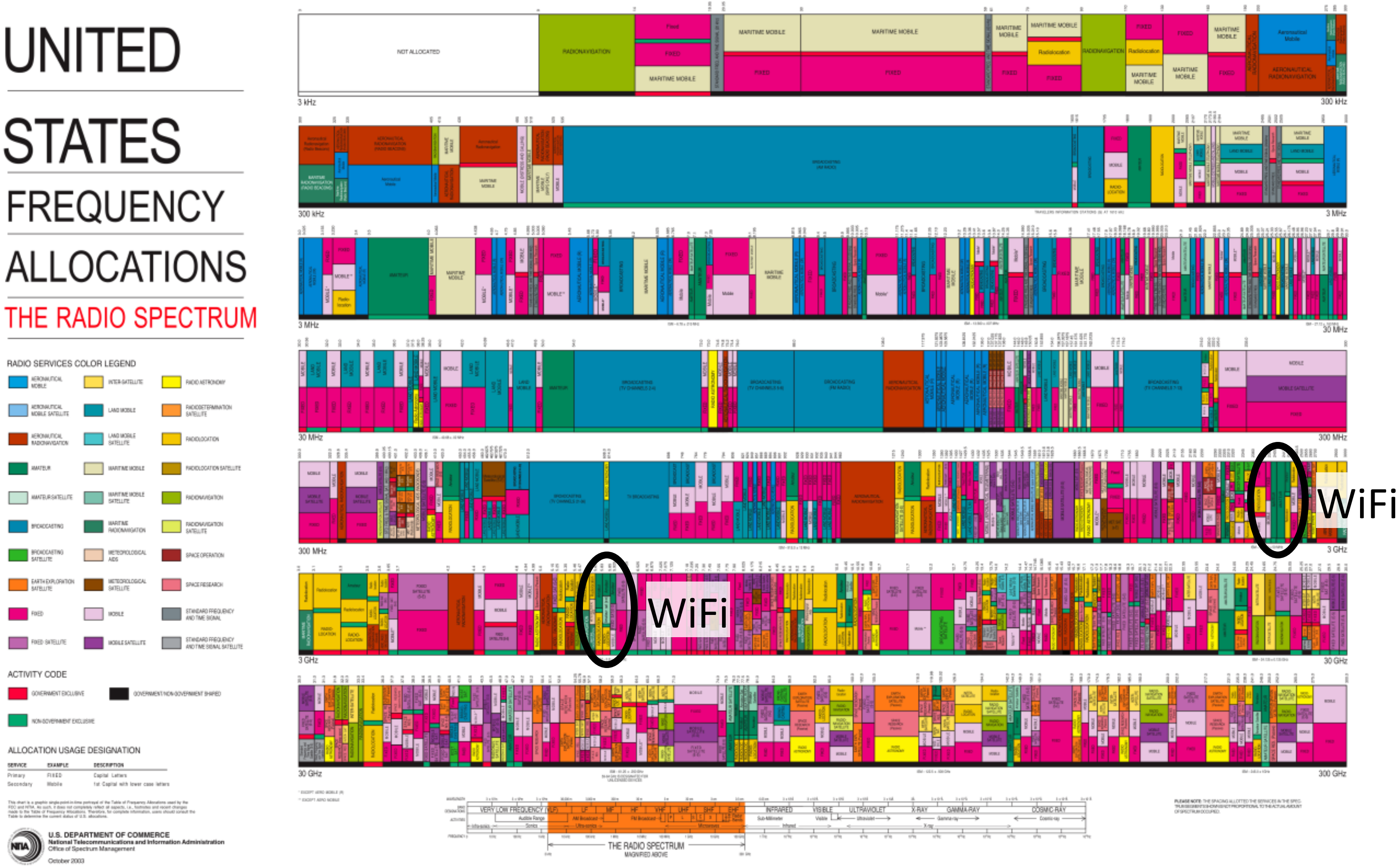
- Signals bounce off objects and take multiple paths
  - Some frequencies attenuated at receiver, varies with location
  - Messes up signal; handled with sophisticated methods



Wireless

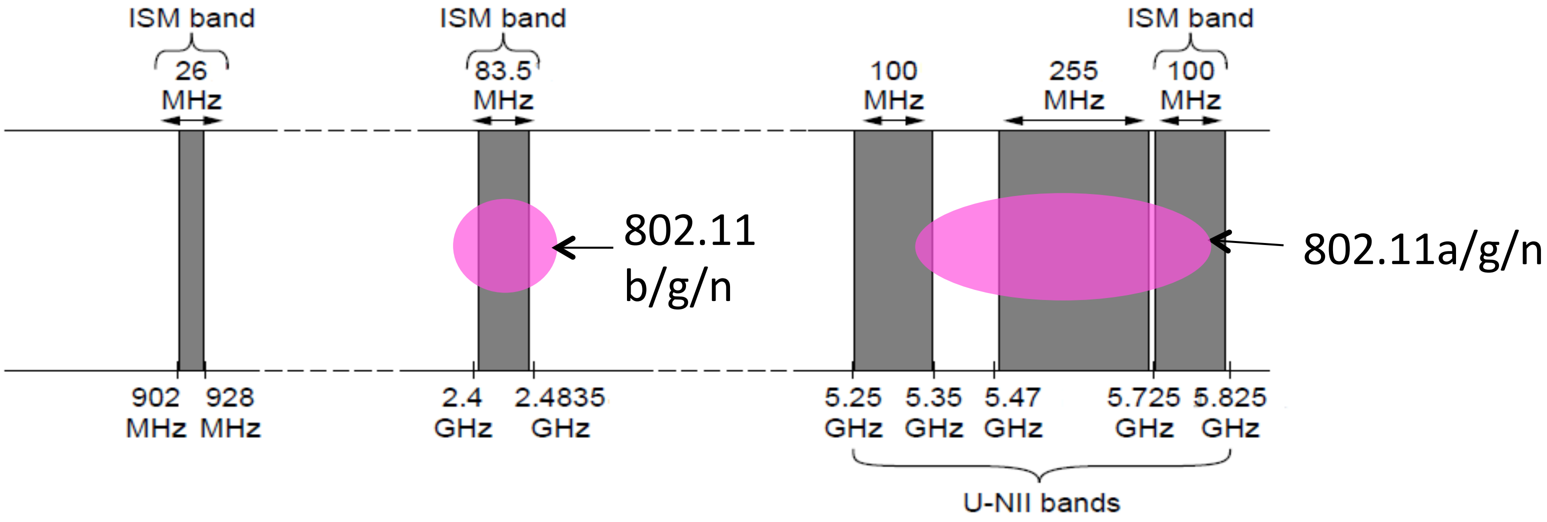
- Sender radiates signal over a region
  - In many directions, unlike a wire, to potentially many receivers
  - Nearby signals (same freq.) interfere at a receiver; need to coordinate use




UNITED
STATES
FREQUENCY
ALLOCATIONS
THE RADIO SPECTRUM
RADIO SERVICES COLOR LEGEND
ACTIVITY CODE
ALLOCATION USAGE DESIGNATION
U.S. DEPARTMENT OF COMMERCE
National Telecommunications and Information Administration
Office of Spectrum Management
October 2003
3 kHz
300 kHz
3 MHz
30 MHz
300 MHz
3 GHz
30 GHz
300 GHz
WiFi
WiFi
THE RADIO SPECTRUM

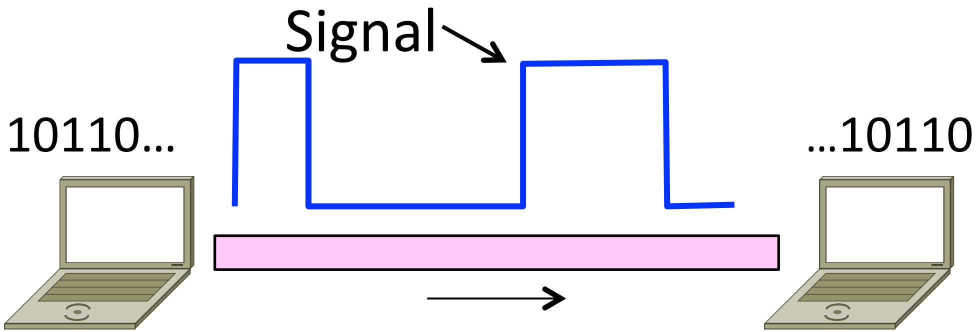
Wireless (2)

- Microwave, e.g., 3G, and unlicensed (ISM) frequencies, e.g., WiFi, are widely used for computer networking



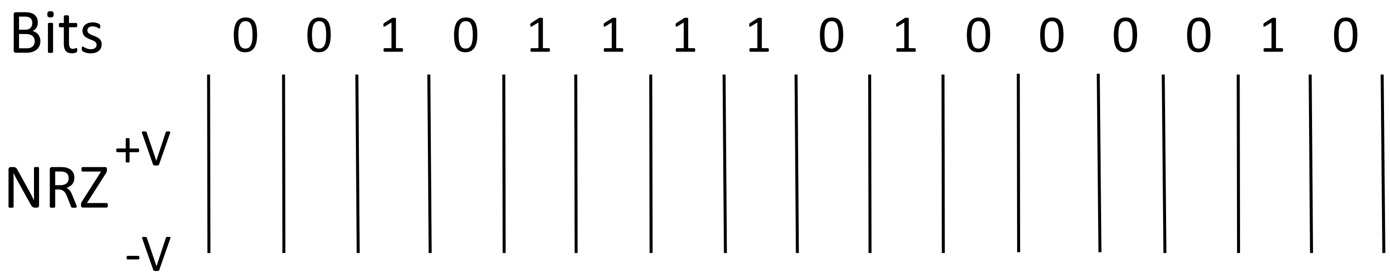
Topic

- We've talked about signals representing bits. How, exactly?
  - This is the topic of modulation



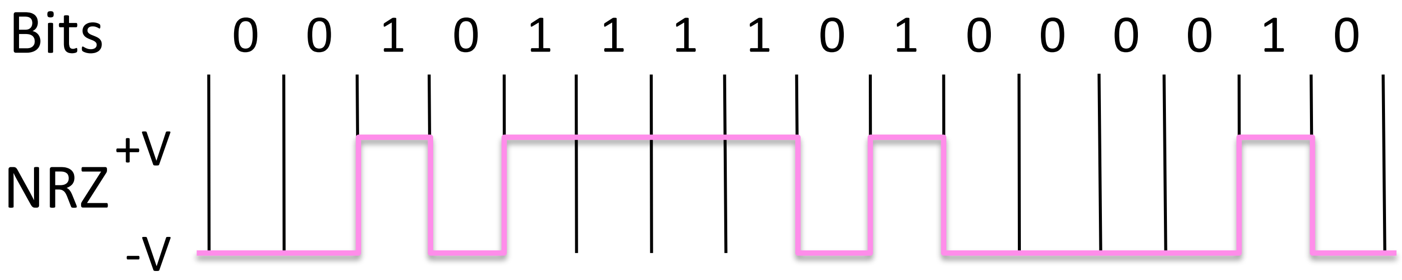
A Simple Modulation

- Let a high voltage (+V) represent a 1, and low voltage (-V) represent a 0
  - This is called NRZ (Non-Return to Zero)



A Simple Modulation (2)

- Let a high voltage (+V) represent a 1, and low voltage (-V) represent a 0
  - This is called NRZ (Non-Return to Zero)



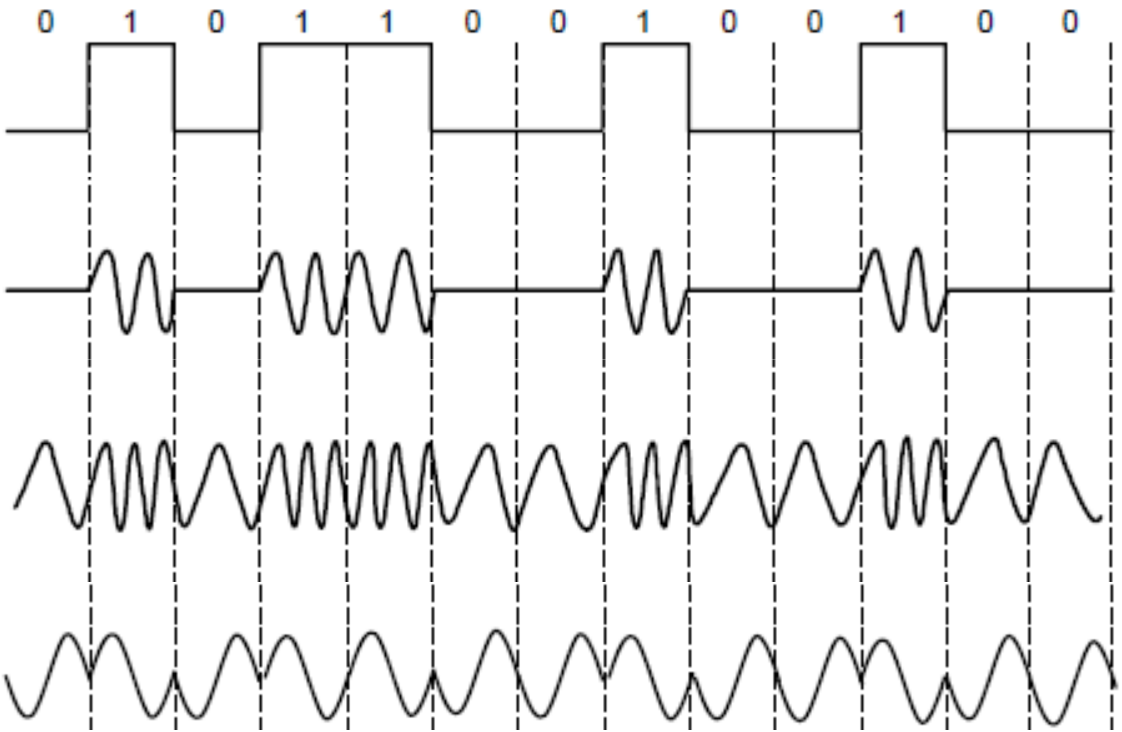
Modulation

NRZ signal of bits

Amplitude shift keying

Frequency shift keying

Phase shift keying



Topic

- How rapidly can we send information over a link?
  - Nyquist limit (~1924) »
  - Shannon capacity (1948) »

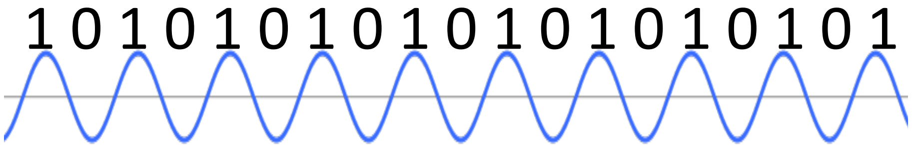
- Practical systems are devised to approach these limits

Key Channel Properties

- The bandwidth (B), signal strength (S), and noise strength (N)
  - B limits the rate of transitions
  - S and N limit how many signal levels we can distinguish

Nyquist Limit

- The maximum symbol rate is 2B

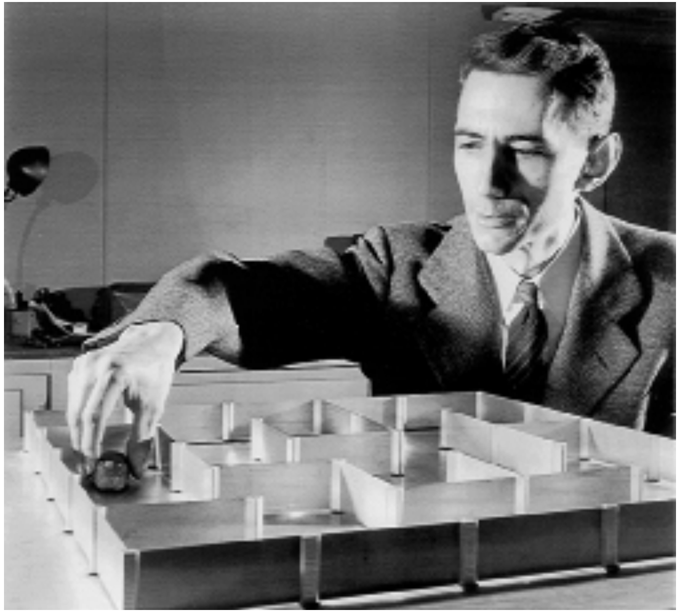


- Thus if there are V signal levels, ignoring noise, the maximum bit rate is: $R = 2B \log_2 V$ bits/sec

Claude Shannon (1916-2001)

- Father of information theory
  - "A Mathematical Theory of Communication", 1948
- Fundamental contributions to digital computers, security, and communications

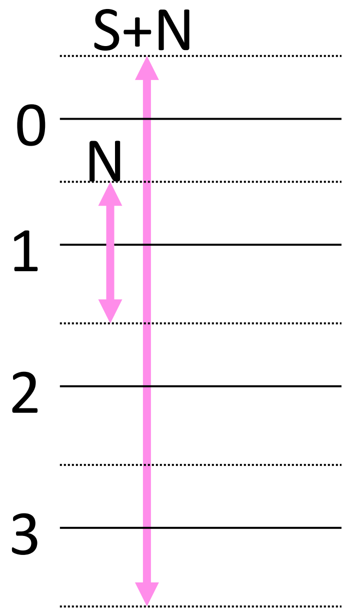
Electromechanical mouse
that "solves" mazes!



Credit: Courtesy MIT Museum

Shannon Capacity

- How many levels we can distinguish depends on S/N
  - Or SNR, the Signal-to-Noise Ratio
  - Note noise is random, hence some errors
- SNR given on a log-scale in deciBels:
  - $SNR_{dB} = 10\log_{10}(S/N)$



Shannon Capacity (2)

- Shannon limit is for capacity (C), the maximum information carrying rate of the channel:

$$C = B \log_2(1 + S/(BN)) \text{ bits/sec}$$

Wired/Wireless Perspective

- Wires, and Fiber
  - Engineer link to have requisite SNR and B
  - → Can fix data rate
- Wireless
  - Given B, but SNR varies greatly, e.g., up to 60 dB!
  - → Can't design for worst case, must adapt data rate

Wired/Wireless Perspective (2)

- Wires, and Fiber
  - Engineer link to have requisite SNR and B

  →Can fix data rate

Engineer SNR for data rate

- Wireless
  - Given B, but SNR varies greatly, e.g., up to 60 dB!

  →Can't design for worst case, must adapt data rate

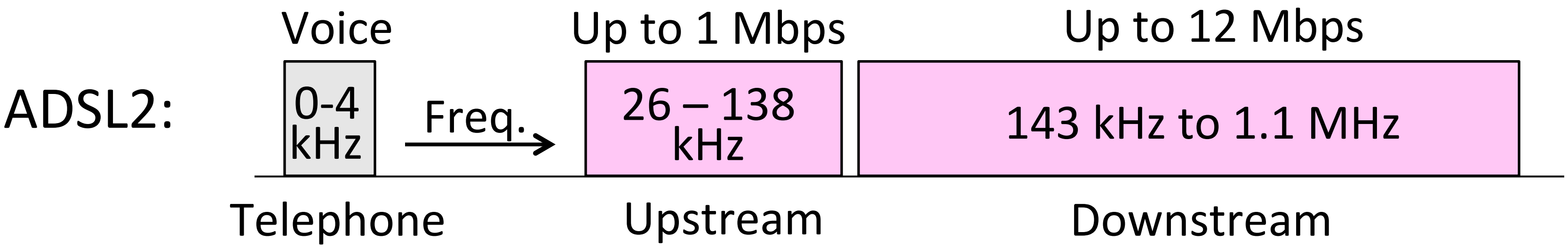
Adapt data rate to SNR

Putting it all together – DSL

- DSL (Digital Subscriber Line) is widely used for broadband; many variants offer 10s of Mbps
  - Reuses twisted pair telephone line to the home; it has up to ~2 MHz of bandwidth but uses only the lowest ~4 kHz

DSL (2)

- DSL uses passband modulation (called OFDM)
  - Separate bands for upstream and downstream (larger)
  - Modulation varies both amplitude and phase (called QAM)
  - High SNR, up to 15 bits/symbol, low SNR only 1 bit/symbol

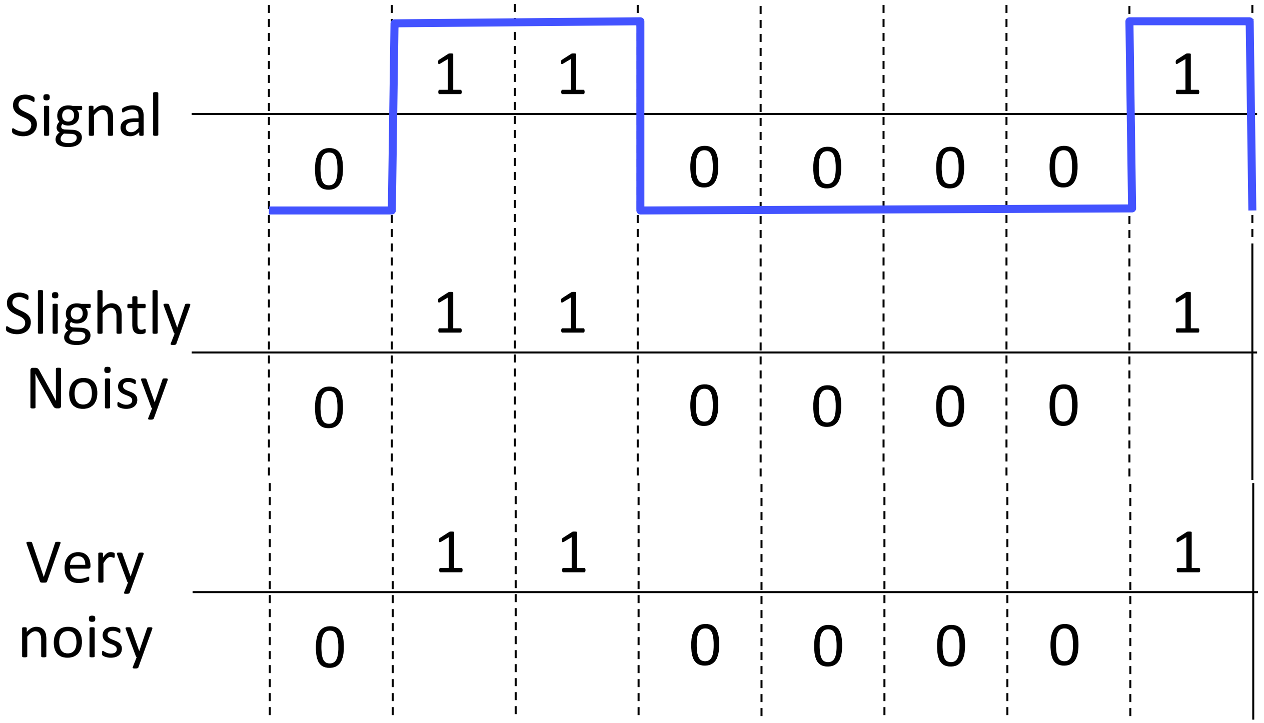


Topic

- Some bits will be received in error due to noise. What can we do?
  - Detect errors with codes »
  - Correct errors with codes »
  - Retransmit lost frames ← Later

- Reliability is a concern that cuts across the layers – we'll see it again

Problem – Noise may flip received bits



Approach – Add Redundancy

- Error detection codes
  - Add <u>check bits</u> to the message bits to let some errors be detected
- Error correction codes
  - Add more <u>check bits</u> to let some errors be corrected

- Key issue is now to structure the code to detect many errors with few check bits and modest computation

Motivating Example

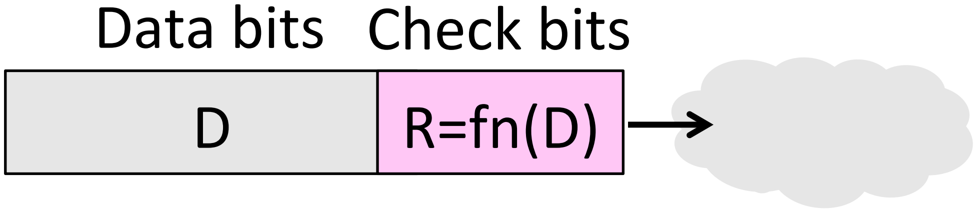
- A simple code to handle errors:
  - Send two copies! Error if different.
- How good is this code?
  - How many errors can it detect/correct?
  - How many errors will make it fail?

Motivating Example (2)

- We want to handle more errors with less overhead
  - Will look at better codes; they are applied mathematics
  - But, they can't handle all errors
  - And they focus on accidental errors (will look at secure hashes later)

Using Error Codes

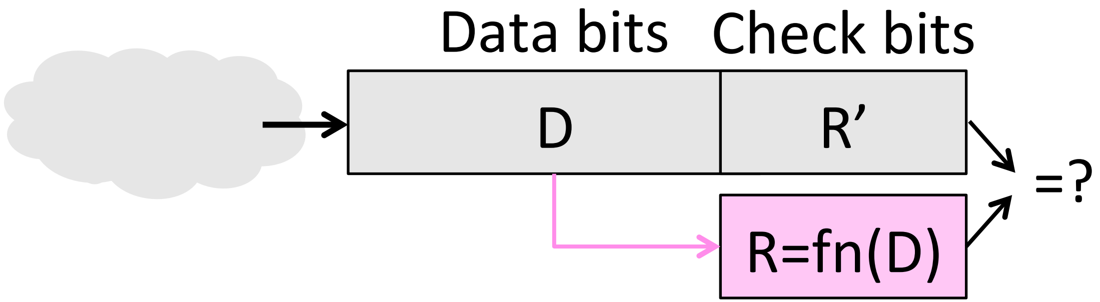
- Codeword consists of D data plus R check bits (=systematic block code)



- Sender:
  - Compute R check bits based on the D data bits; send the codeword of D+R bits

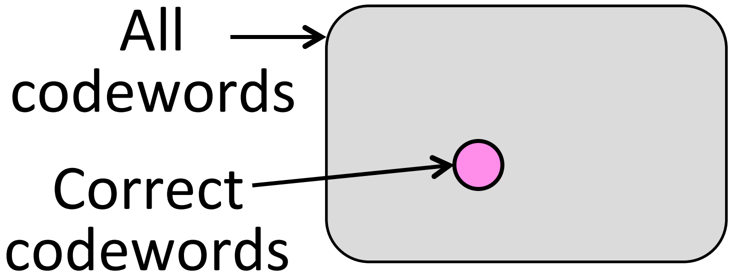
Using Error Codes (2)

- Receiver:
  - Receive D+R bits with unknown errors
  - Recompute R check bits based on the D data bits; error if R doesn't match R'



Intuition for Error Codes

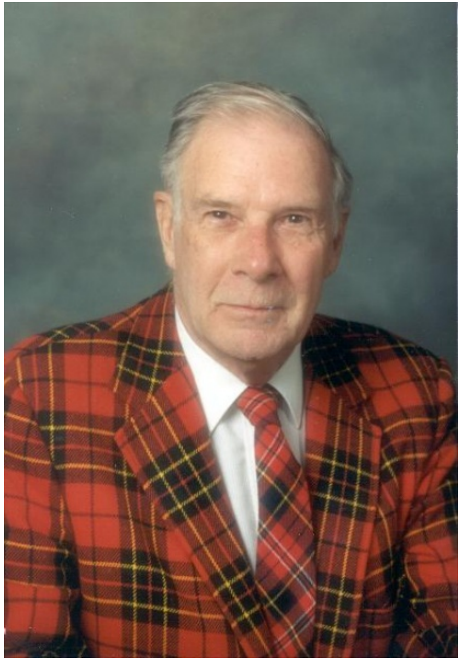
- For D data bits, R check bits:



- Randomly chosen codeword is unlikely to be correct; overhead is low

R.W. Hamming (1915-1998)

- Much early work on codes:
  - "Error Detecting and Error Correcting Codes", BSTJ, 1950
- See also:
  - "You and Your Research", 1986



Source: IEEE GHN, © 2009 IEEE

Hamming Distance

- Distance is the number of bit flips needed to change D_1 to D_2
- Hamming distance of a code is the minimum distance between any pair of codewords

Hamming Distance (2)

- Error detection:
  - For a code of distance d+1, up to d errors will always be detected

Hamming Distance (3)

- Error correction:
  - For a code of distance 2d+1, up to d errors can always be corrected by mapping to the closest codeword

Topic

- Some bits may be received in error due to noise. How do we detect this?
  - Parity »
  - Checksums »
  - CRCs »

- Detection will let us fix the error, for example, by retransmission (later).

Simple Error Detection – Parity Bit

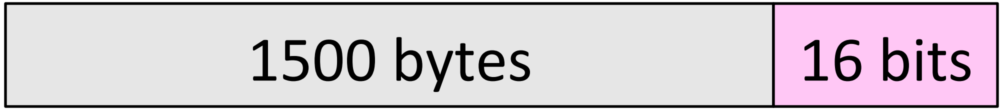
- Take D data bits, add 1 check bit that is the sum of the D bits
  - Sum is modulo 2 or XOR

Parity Bit (2)

- How well does parity work?
  - What is the distance of the code?
  - How many errors will it detect/ correct?
- What about larger errors?

Checksums

- Idea: sum up data in N-bit words
  - Widely used in, e.g., TCP/IP/UDP



- Stronger protection than parity

Internet Checksum

- Sum is defined in 1s complement arithmetic (must add back carries)
  - And it’s the negative sum
- *“The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words …”* – RFC 791

Internet Checksum (2)

Sending:

1. Arrange data in 16-bit words
2. Put zero in checksum position, add
3. Add any carryover back to get 16 bits
4. Negate (complement) to get sum

```
0001
f203
f4f5
f6f7
```

Internet Checksum (3)

Sending:

1. Arrange data in 16-bit words
2. Put zero in checksum position, add
3. Add any carryover back to get 16 bits
4. Negate (complement) to get sum

```
   0001
   f203
   f4f5
   f6f7
+(0000)
------
  2ddf0
    ↓
   ddf0
+     2
------
   ddf2
    ↓
   220d
```

Internet Checksum (4)

Receiving:

1. Arrange data in 16-bit words
2. Checksum will be non-zero, add

3. Add any carryover back to get 16 bits

4. Negate the result and check it is 0

```
    0001
    f203
    f4f5
    f6f7
+   220d
  ------
```

Internet Checksum (5)

Receiving:

1. Arrange data in 16-bit words
2. Checksum will be non-zero, add
3. Add any carryover back to get 16 bits
4. Negate the result and check it is 0

```
   0001
   f203
   f4f5
   f6f7
+  220d
------
  2fffd

   fffd
+     2
------
   ffff

   0000
```

Internet Checksum (6)

- How well does the checksum work?
  - What is the distance of the code?
  - How many errors will it detect/ correct?

- What about larger errors?

Cyclic Redundancy Check (CRC)

- Even stronger protection
  - Given n data bits, generate k check bits such that the n+k bits are evenly divisible by a generator C
- Example with numbers:
  - n = 302, k = one digit, C = 3

CRCs (2)

- The catch:
  - It's based on mathematics of finite fields, in which "numbers" represent polynomials
  - e.g, 10011010 is $x^7 + x^4 + x^3 + x^1$
- What this means:
  - We work with binary values and operate using modulo 2 arithmetic

CRCs (3)

- Send Procedure:

1. Extend the n data bits with k zeros
2. Divide by the generator value C
3. Keep remainder, ignore quotient
4. Adjust k check bits by remainder

- Receive Procedure:

1. Divide and check for zero remainder

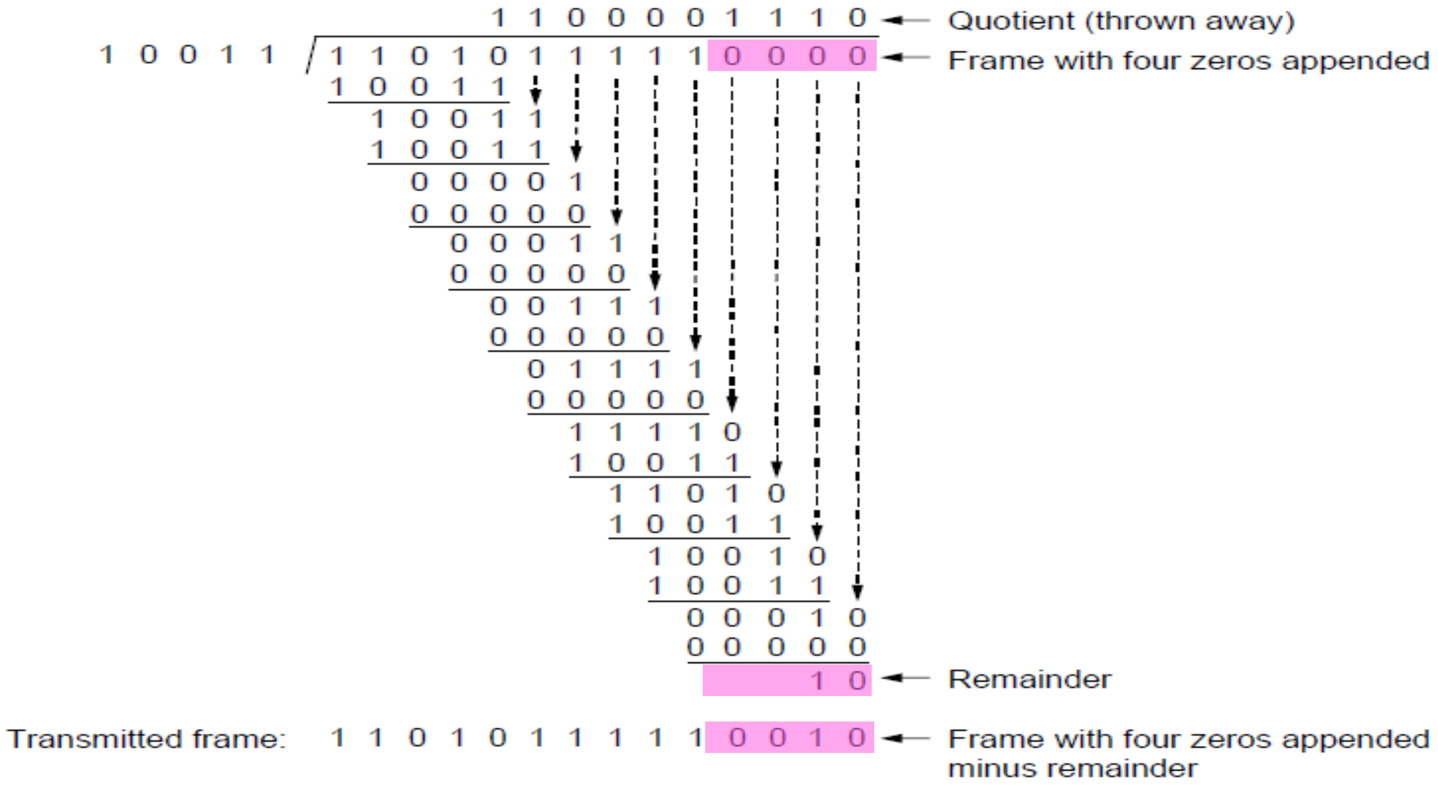
CRCs (4)

Data bits:
1101011111

Check bits:
$C(x)=x^4+x^1+1$
C = 10011
k = 4

$$10011 \overline{)\,1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 1}$$

CRCs (5)


1 1 0 0 0 0 1 1 1 0
Quotient (thrown away)
1 0 0 1 1
1 1 0 1 0 1 1 1 1 1 0 0 0 0
Frame with four zeros appended
Remainder
1 0
Transmitted frame: 1 1 0 1 0 1 1 1 1 1 0 0 1 0
Frame with four zeros appended minus remainder

CRCs (6)

- Protection depend on generator
  - Standard CRC-32 is 10000010 01100000 10001110 110110111
- Properties:
  - HD=4, detects up to triple bit errors
  - Also odd number of errors
  - And bursts of up to k bits in error
  - Not vulnerable to systematic errors like checksums

Error Detection in Practice

- CRCs are widely used on links
  - Ethernet, 802.11, ADSL, Cable …
- Checksum used in Internet
  - IP, TCP, UDP … but it is weak
- Parity
  - Is little used

Topic

- Two strategies to handle errors:

1. Detect errors and retransmit frame (Automatic Repeat reQuest, ARQ)

2. Correct errors with an error correcting code

Done this

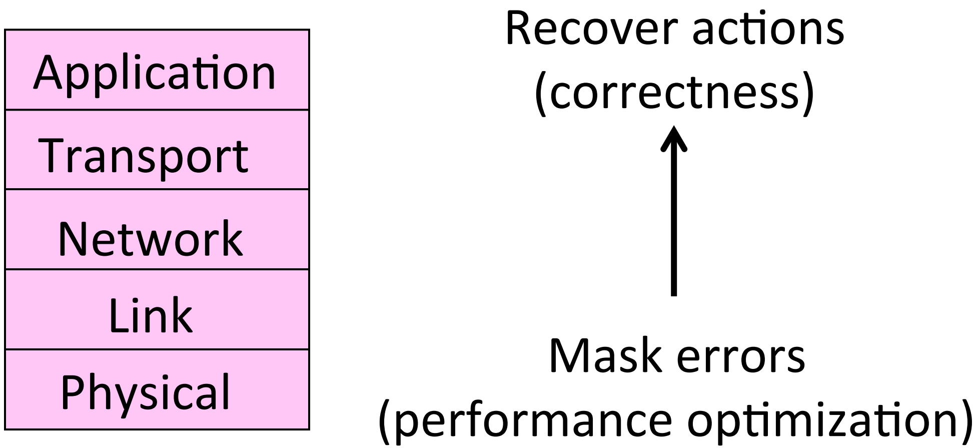
Context on Reliability

- Where in the stack should we place reliability functions?

Application
Transport
Network
Link
Physical

Context on Reliability (2)

- Everywhere! It is a key issue
  - Different layers contribute differently



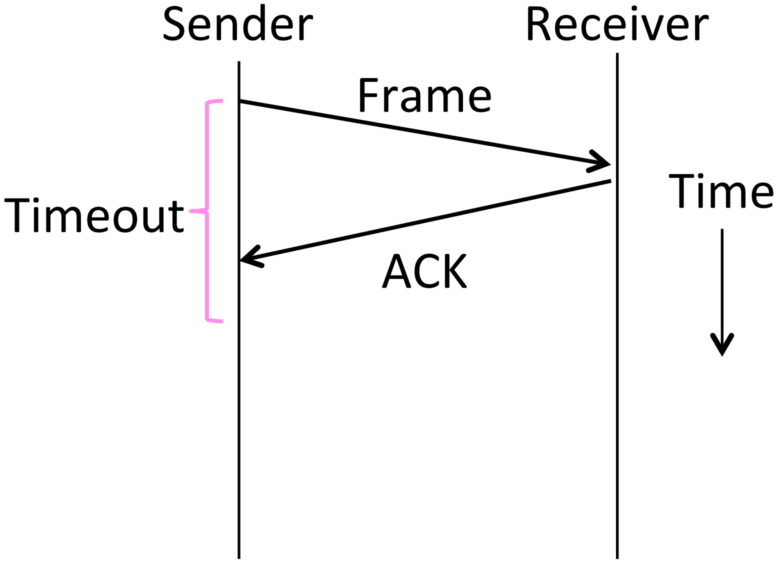
ARQ

- ARQ often used when errors are common or must be corrected
  - E.g., WiFi, and TCP (later)

- Rules at sender and receiver:
  - Receiver automatically acknowledges correct frames with an ACK
  - Sender automatically resends after a timeout, until an ACK is received

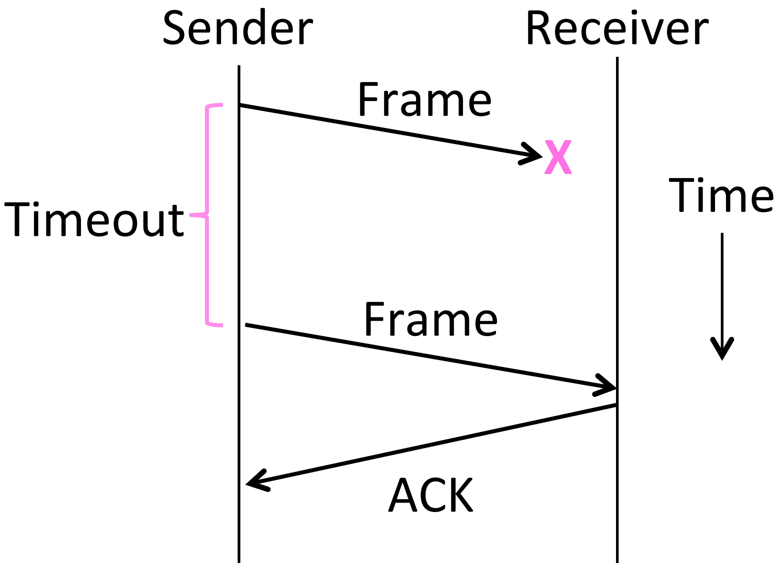
ARQ (2)

- Normal operation (no loss)



ARQ (3)

- Loss and retransmission



So What's Tricky About ARQ?

- Two non-trivial issues:
  - How long to set the timeout? »
  - How to avoid accepting duplicate frames as new frames »
- Want performance in the common case and correctness always

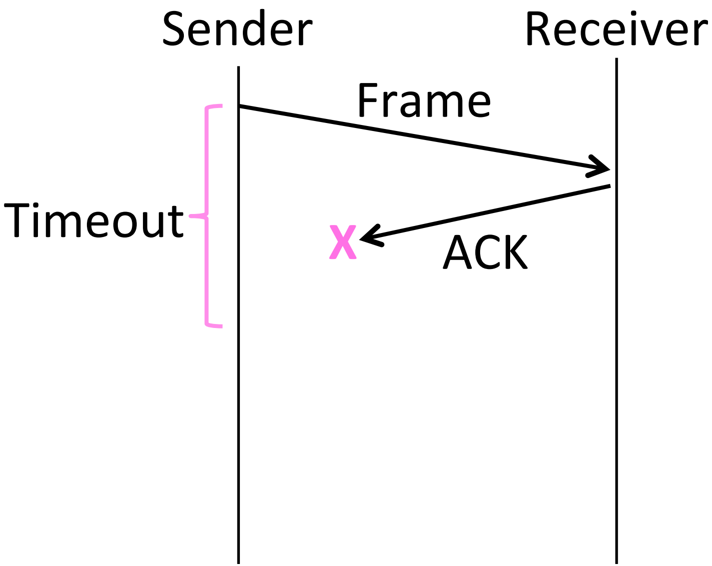
Timeouts

- Timeout should be:
  - Not too big (link goes idle)
  - Not too small (spurious resend)

- Fairly easy on a LAN
  - Clear worst case, little variation
- Fairly difficult over the Internet
  - Much variation, no obvious bound
  - We’ll revisit this with TCP (later)

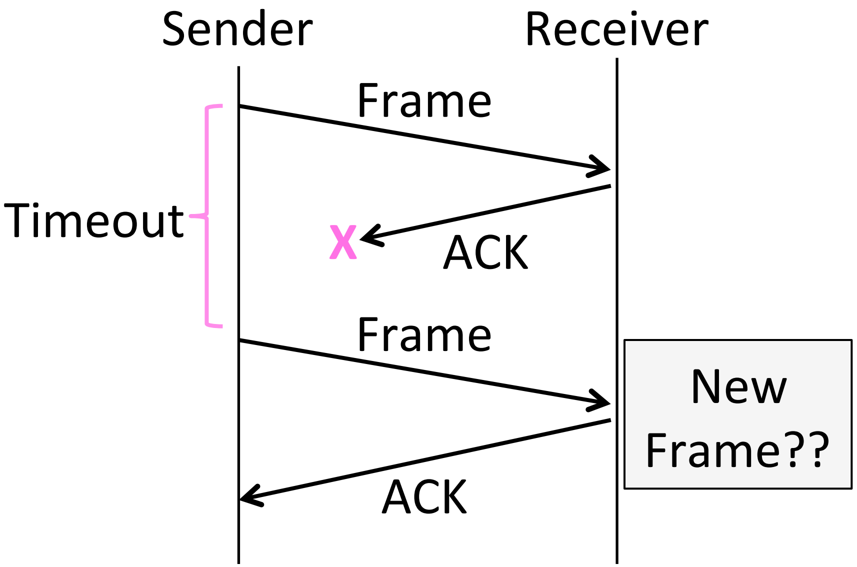
Duplicates

- What happens if an ACK is lost?



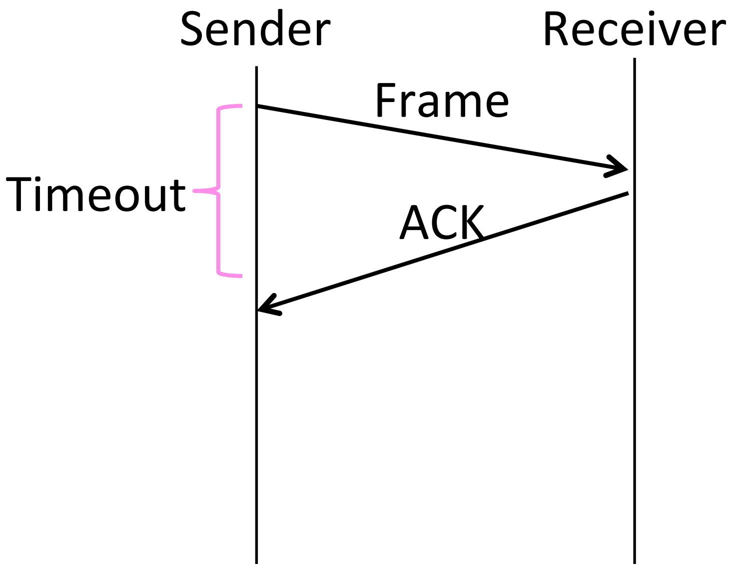
Duplicates (2)

- What happens if an ACK is lost?



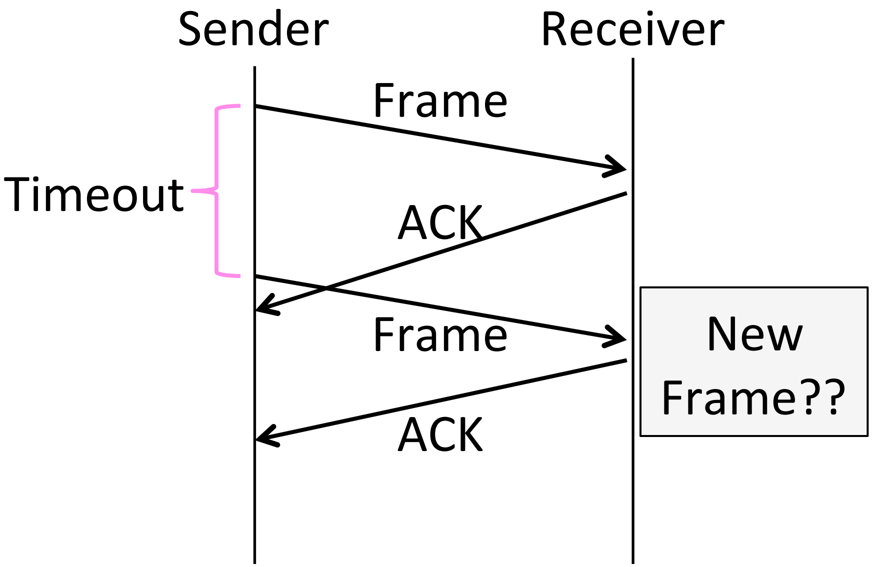
Duplicates (3)

- Or the timeout is early?



Duplicates (4)

- Or the timeout is early?

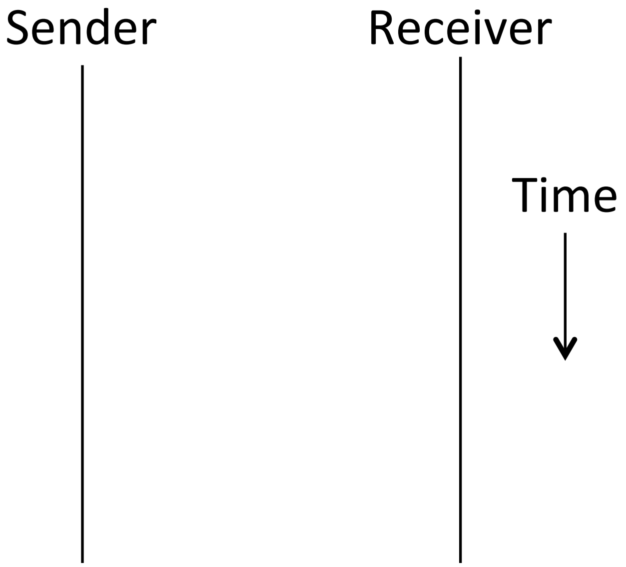


Sequence Numbers

- Frames and ACKs must both carry sequence numbers for correctness
- To distinguish the current frame from the next one, a single bit (two numbers) is sufficient
  - Called Stop-and-Wait

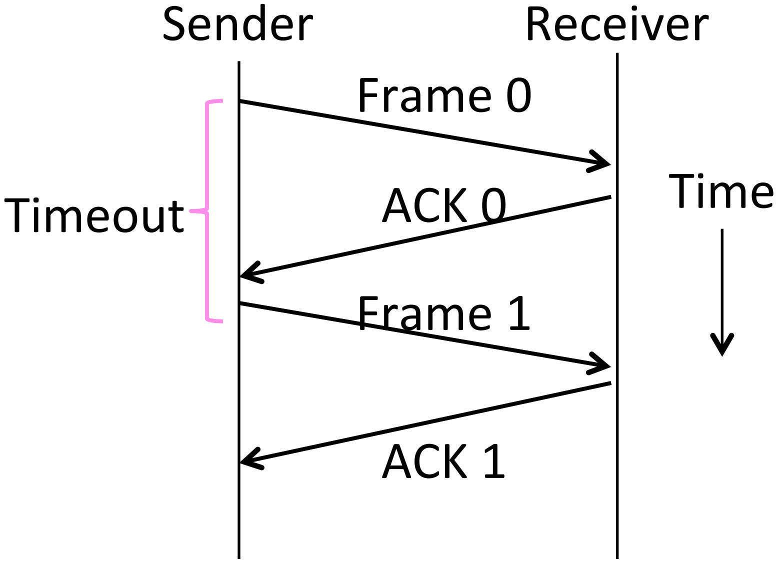
Stop-and-Wait

- In the normal case:



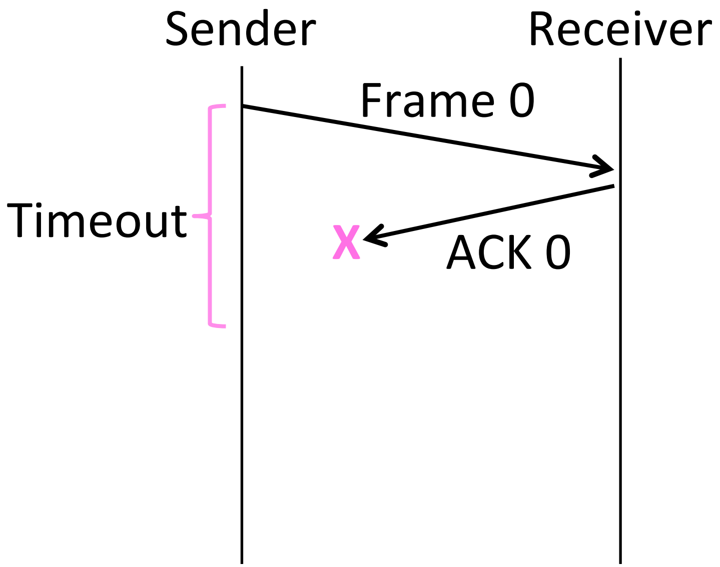
Stop-and-Wait (2)

- In the normal case:



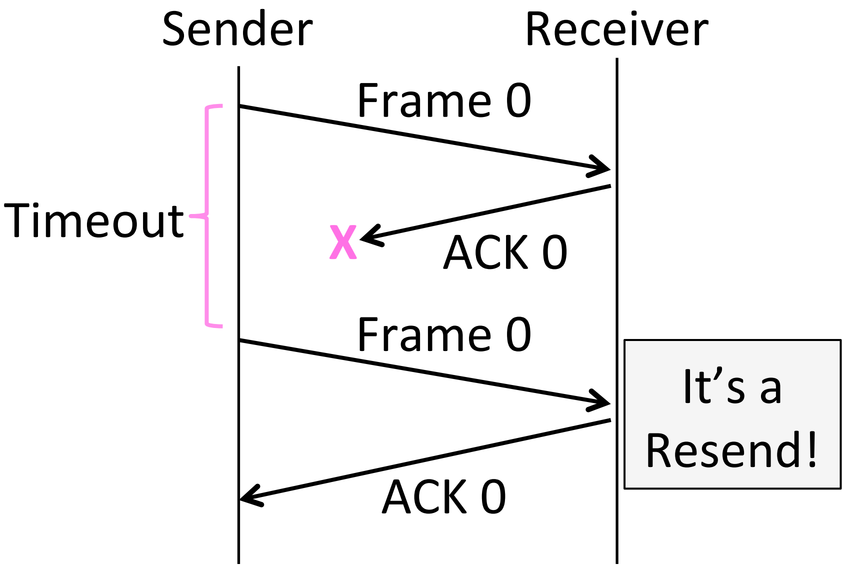
Stop-and-Wait (3)

- With ACK loss:



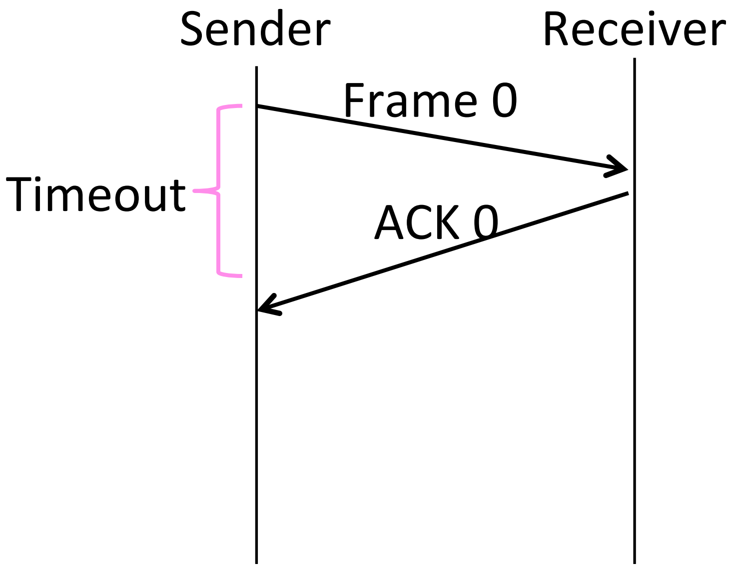
Stop-and-Wait (4)

- With ACK loss:



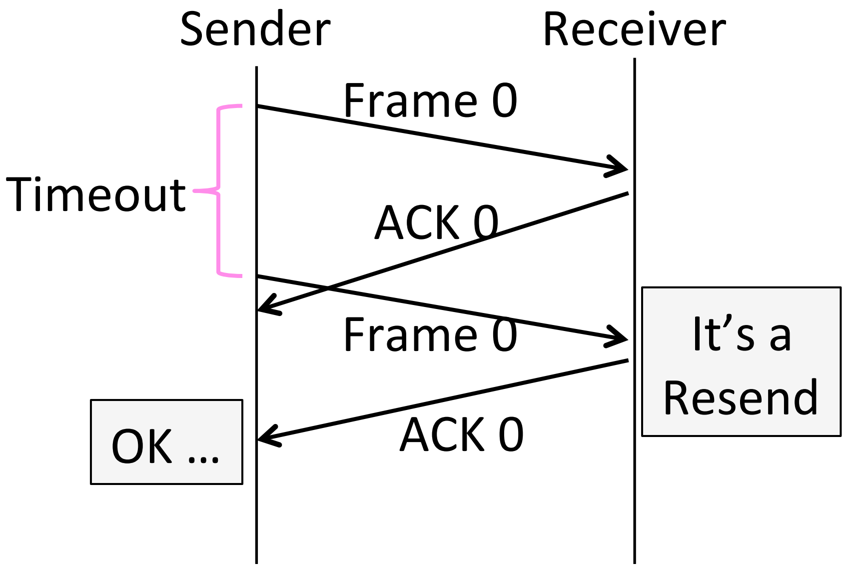
Stop-and-Wait (5)

- With early timeout:



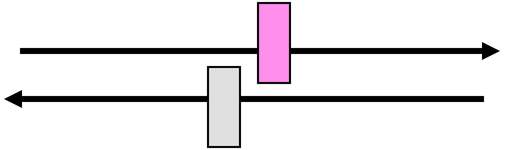
Stop-and-Wait (6)

- With early timeout:



Limitation of Stop-and-Wait

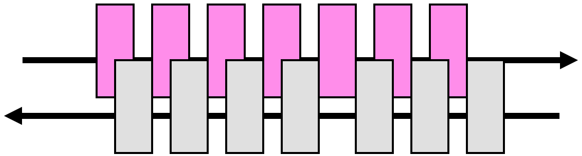
- It allows only a single frame to be outstanding from the sender:
  - Good for LAN, not efficient for high BD



- Ex: R=1 Mbps, D = 50 ms
  - How many frames/sec? If R=10 Mbps?

Sliding Window

- Generalization of stop-and-wait
  - Allows W frames to be outstanding
  - Can send W frames per <u>RTT</u> (=2D)



  - Various options for numbering frames/ACKs and handling loss
    - Will look at along with TCP (later)