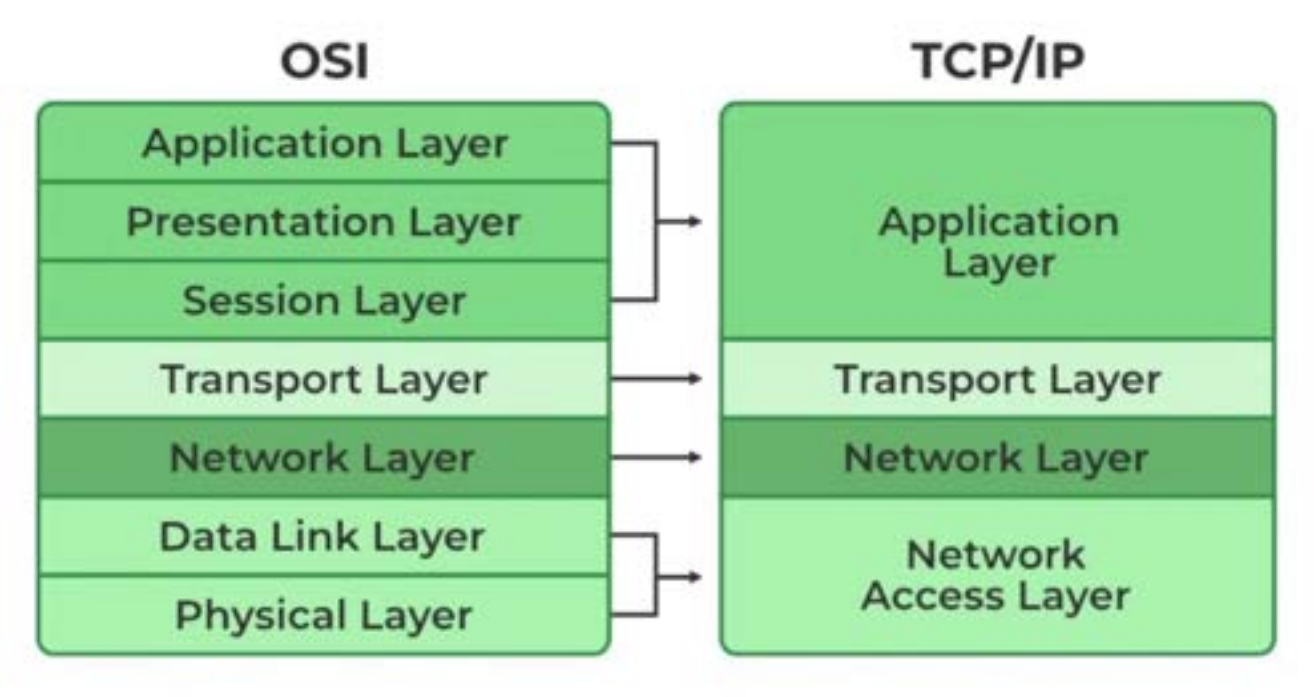
Computer Communications



The Internet (Week 7)

- loose association of thousands of network & millions of computers across the world that all work tgt to share info → network owned by different organisations
- Internet Engineering Task Force (IETF) defines standard for internet
- Request for Comments (RFC) is authored by engineers in form of memorandum describing research relevant to internet & IETF sometimes adopt some RFCs as Internet Standards
- Institute of Electrical & Electronic Engineers (IEEE) standardise most local area network (LAN) & wide-area network technologies

Protocols

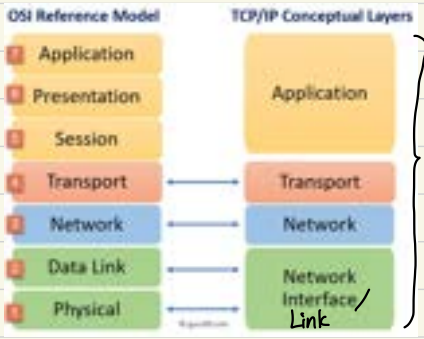
- specific set of communication rules & instruction so computers can communicate with each other
- define format, order of messages sent & received among network entities & action taken on message transmission & reception
- common protocols: transmission control protocol (TCP) → transport layer, internet protocol (IP) → network layer, hypertext transfer protocol (HTTP), file transfer protocol (FTP) → application layer, IEEE 802.3 Ethernet/(CSMA/CD) protocol → link layer

TCP/IP protocol stack (set of communication protocol layers that work tgt)

5. Applications → consists of protocols that focusses on process-to process communication across IP network
4. Transport → responsible for end-to-end communication over network → logical communication bw application process running on different hosts. Also responsible for error correction
3. Network → responsible for routing & forwarding info from source to destination. Also responsible for congestion control, internet-wide addressing
2. Link → governs data transfer bw 2 directly connected network devices/host. Defines medium access protocol that governs how different host can transmit on shared medium & how error could be detected & corrected
1. Physical → defines transmission media, mechanical & electrical aspects of physical layer interfaces

Open System Interconnected (OSI) Reference Model

6. Presentation (allow applications to interpret meaning of data)
   - responsible for transforming data into form application layer can accept, format & encrypt data, providing freedom from compatibility problems
7. Session (synchronisation, checkpointing, recovery of data exchange)
   - sets up, coordinates & terminate conversations & dialogues bw application at each end



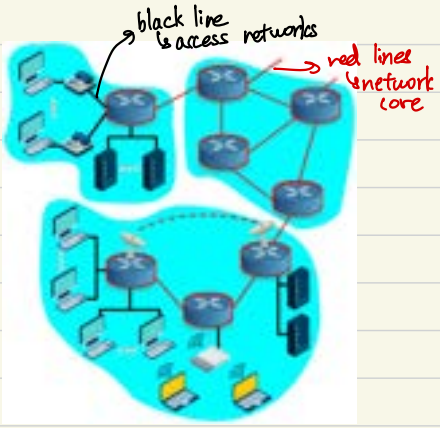
Protocol layering → makes it easier to understand
→ eases maintence, updating of system

Introduction to computer network

- Computer network → collection of network nodes, network links & network protocols to transmit information
  1. Nodes → hosts: communication endpoints (workstations, laptop, tablets)
     - switches/routers: nodes used to interconnect links
  2. Links → carry bits from one place to another/multiple places: fiber (infrared light), copper (electronic voltage/current) (→ coaxial cable), satellite (radio)
  3. Interfaces → attach nodes to links → wired interface: ethernet card, wireless interface → wireless card
- Distinguished services: latency, bandwidth, loss rate, no. of end systems
- Protocols: rules governing communication bw nodes: TCP/IP, ATM, MPLS, SONET, Ethernet, X.25 etc

Network Architecture

1. Network edge [applications & hosts]
   - end systems (hosts): run application programs at "edge of network"
   - client/server model: client host requests, receives service from server host (e.g. WWW, email)
   - peer-peer model: hosts interact symmetrically, working as both server & client (e.g. BitTorrent) (→ security issue & higher bandwidth consumption)
2. Access Networks [connect network edges to network core]
   - residential access networks (e.g. dial-up modem, ADSL, cable modem, FTTH)
   - institutional access networks for school/company (e.g. ethernet access)
   - mobile access networks (e.g. wireless LAN, wide-area wireless access (→ 3G/4G))



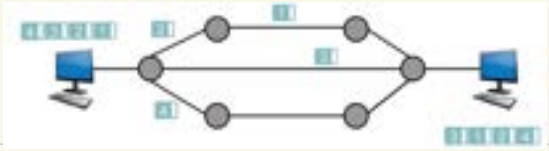
3. Network Core (mesh of interconnected routers)
   - Types of communication:
     a) Circuit switching: Dedicated circuit (e.g. telephone network)
     b) Packet (datagram) switching: data is sent through network in discrete "chunks"

Circuit Switching [telephone network]

- dedicated communication path between end nodes → no sharing of resources → connection setup require switching capacity & channel capacity
- benefits: guaranteed performance
- disadvantages: resources wasted if no data transmission, setup/tearing down takes time

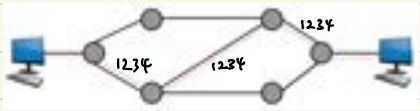
Packet Switching

- data transmitted in small packets & treated independently:
  a) data organised into packets of multiple of bytes
  b) each packet contains a portion of user data (data payload) & some control information (header/overhead)
  c) packets are received, stored briefly (buffered) at each node & passed on to next node
- benefits: allows more users to share the bandwidth
- disadvantages: packets may arrive out of order/go missing → up to receiver to re-order/recover packets
- Packets are handled in two approaches:
  1. Datagram service [IP network]
     - for each packet, each node makes its own decision as to how to forward it so it eventually reaches its destination



→ benefits → more flexible/reliable → routing can be used to avoid congested part of network
→ disadvantages → no call setup phase → better if few packets

2. Virtual circuits



→ path for packet pre-determined for sender-receiver pair & packet always follows this path
→ benefits → network can provide sequencing & error control, packets are forwarded more quickly → no routing decision to make
→ disadvantages → less reliable → loss of a node loses all circuits through that node

Packet Switching vs Circuit Switching

E.g. Each user uses 100Kbps when "active" & they are active 10% of the time. Consider that a 1Mbps link is shared between the users. What if there are 35 active users for packet switching?

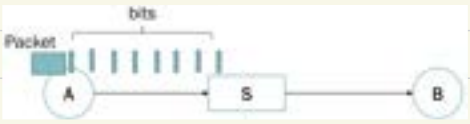
Circuit switching: $\frac{1\text{ Mbps}}{100\text{ Kbps}} = 10$ → support maximum 10 users

Packet-switching: Among 35 users, probability for n users to be active = $\binom{35}{n} 0.1^n (1-0.1)^{35-n}$ ($\frac{35!}{n!(35-n)!}$; $(1-0.1)^{35-n}$ → being inactive)

Probability for more than 10 users to be active at same time = $\sum_{n=11}^{35} \frac{35!}{n!(35-n)!} (0.1)^n \cdot (1-0.1)^{35-n} = 0.0004$ (in g.c → math → math then toggle down)

→ packet-switching network can serve much more users than its capacity actually allows

Forwarding Mode

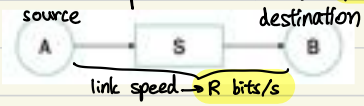


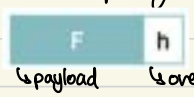
1. Cut-through

→ At network node, frame is forwarded from input to output port without first being received completely
→ Reduced delays but no error checking

2. Store and Forward

→ Entire packet must be received completely by network node before it can be transmitted onto next link



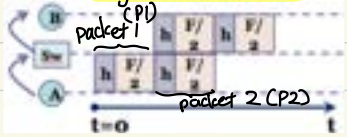


} F+h bits

→ Time for message to transfer from A to S/S+B = $\frac{F+h}{R}$ seconds
→ Total time delay to transfer from A to B = $2\left(\frac{F+h}{R}\right)$ seconds

Pipelining

→ dividing a long message into smaller segments (overhead have to be added to each segment) → can reduce the overall message transfer delay
→ but if segments too small → added overhead of h bits per packet will have detrimental effect. [not gd!] $\left[\frac{F}{n} < h, n \in \mathbb{Z}^+\right]$



→ transmits message of F bits as two segments of $\frac{F}{2}$ bits each
→ after receiving packet 1, switch starts forwarding it to B while it receives packet 2.
→ after packet 2 received, switch forward it to B

P1 from A to S/S to B = $\frac{\frac{F}{2}+h}{R}$ seconds
P1 from S to B & P2 from A to S = $\frac{\frac{F}{2}+h}{R}$ seconds
Total time = $3\left(\frac{\frac{F}{2}+h}{R}\right)$ seconds $< 2\left(\frac{F+h}{R}\right)$ if $\frac{F}{2} > h$

Time saved from this: $\frac{2(F+h)}{R} - 3\frac{\left(\frac{F}{2}+h\right)}{R} = \frac{\frac{F}{2}-h}{R}$ → $\frac{F}{2} - h$

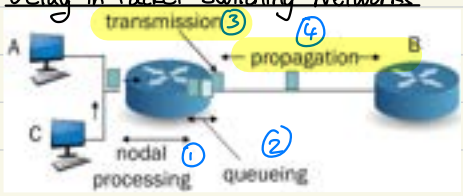
↳ pipelining more effective when there are multiple switches between source & destination nodes

↳ Total time taken = $3(\frac{F+h}{R})$ ↳ Total time taken = $4(\frac{\frac{F}{2}+h}{R})$

Time saved from this: $3(\frac{F+h}{R}) - 4(\frac{\frac{F}{2}+h}{R}) = \frac{F-h}{R} \Rightarrow F-h$

✱ Compared to above: $F-h > \frac{F}{2}-h$ → therefore with more switches there is a bigger gain from pipelining

Delay in Packet Switching Networks



1. Nodal processing delay (process package) } if nodal delay not mentioned in question → ignore it
   ↳ check bit errors & determine output link

2. Queuing delay (normally assume 0)
   ↳ after node processing, packets will join a queue → packet can be transmitted onto outbound link only if no other packet is ahead in queue & no other packet is being transmitted on the link → time wasted

3. Transmission delay (time to send bits into link)
   ✱ ↳ $\frac{\text{packet length}}{\text{link/channel rate}} = \frac{L \text{ (bits)}}{R \text{ (bps)}}$ → bits per second
   (link/channel rate = bandwidth)

4. Propagative delay (time for package to propagate from one end of link to other end of link)
   ✱ ↳ $\frac{d}{s}$ → length of physical link (m)
   → propagation speed in medium (~2×10^8 m/s) ✱

E.g. Packet length is 1K byte (1 byte → 8 bits) & bandwidth is 1 Gbps & fiber is 100 km.

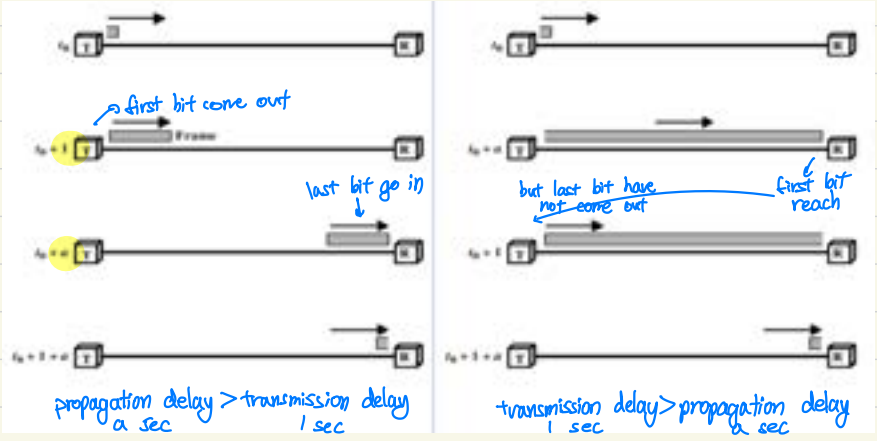
Transmission delay = $\frac{L}{R} = \frac{1 \times 10^3 \times 8}{1 \times 10^9} = 8 \times 10^{-6}$ sec $= 8\,\mu$sec

Propagative delay = $\frac{100 \times 1000 \text{ (m)}}{2 \times 10^8} = 5 \times 10^{-4}$ sec $= 500\,\mu$sec

propagation delay

transmission delay

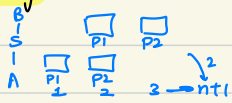
} copper wire



E.g. Will subdividing the message further (to n packets) lead to even lower delay?

$S = \frac{F}{n}$ → Delay for packet over each link = $\frac{\frac{F}{n}+h}{R}$

Total delay $D_n = (n+1)\left(\frac{\frac{F}{n}+h}{R}\right)$ → (n+1) section for packet to arrive

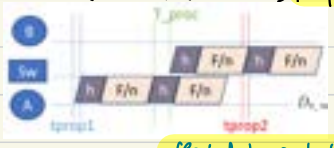


To find minimum delay → $\frac{dD_n}{dn} = 0$

$\frac{dD_n}{dn} = \frac{h}{R} - \frac{F}{n^2 R} = 0 \rightarrow n = \sqrt{\frac{F}{h}}$ (rej. $n = -\sqrt{\frac{F}{h}}$ as no. of packet cannot be -ve)

$\frac{d^2D_n}{dn^2} > 0$ → prove minimum D_n

What if we add in effects of link propagation delay & additional processing delay at switch? Assume distance bw A & switch = d1 & switch & B = d2, propagation speed in link is v. Processing delay at switch fixed at T_proc.

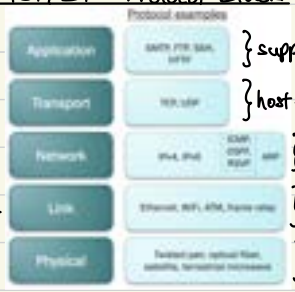


$D_{n\_new} = (n+1)\frac{\frac{F}{n}+h}{R} + t_{prop1} + t_{prop2} + T_{proc}$ (t_{prop1}: d1, t_{prop2}: d2)

not a function of n

★ If transmission delay (affected by packet size) taken into account → will affect n_{min}

TCP/IP Protocol Stack



5 } supporting network applications

4 } host-host data transfer

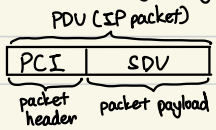
3 } Routing of datagrams from source to destination

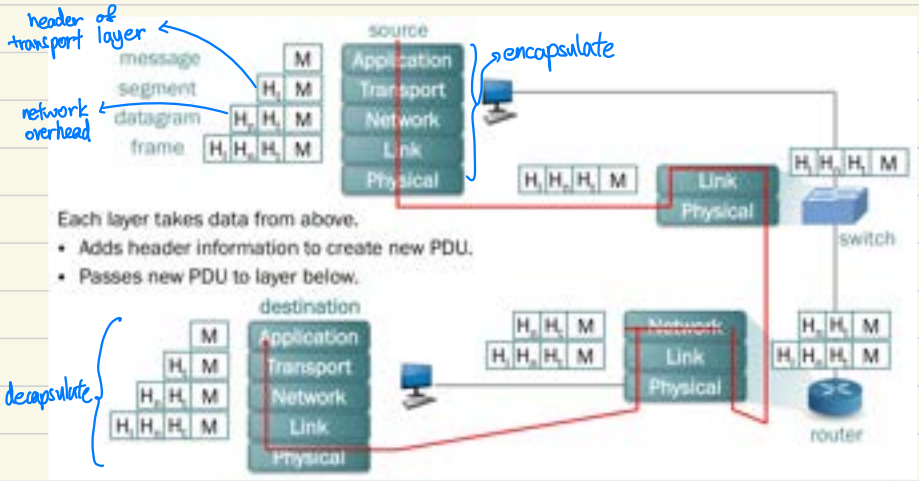
2 } Data transfer bw neighbouring network elements

1 } Bits "on the wire"

Encapsulation

↳ addition of protocol control information (PCI) (overhead) to a service data unit (SDU) (data payload) to form the Protocol Data Unit (IP Packet) (PDU) of a particular layer by a communication protocol





Transmission System

Transmitter ---> Communication Channel (Physical) ---> Receiver

- Transmitter: converts digital data into signal suitable for transmission & sends the signal over this
- Receiver: receives the signal & converts it back into digital data to be delivered to user

→ Depending on transmission medium

- data may be encoded directly & send to medium → data encoding (Digital signal to carry digital data) → mapping of information into sequence of digital signals
  - information discrete states
  - Encoder → Decoder
  - → baseband transmission, line coding
- data may be modulated on to an analog carrier & then transmitted → modulation (analogue signal to carry digital signal) → embedding of information into sinusoidal waveforms
  - Modulation → Demodulation
  - → carrier frequency modulation

Channel Characterisation Parameters

1. Channel Attenuation

Transmitted Signal (P_{in}) → communication channel (→ no active component inside to boost power) → Received Signal (P_{out})

Channel Attenuation (Loss) = $\frac{P_{in}}{P_{out}} > 1 = 10\log_{10}\left(\frac{P_{in}}{P_{out}}\right) dB > 0$

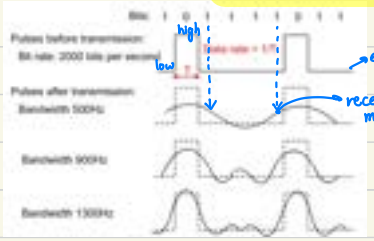
2. Channel Bandwidth

$x(t) = a\cos(2\pi ft) \rightarrow$ channel $\rightarrow y(t) = aA(f)\cos(2\pi ft)$ (→ frequency characterisation of channel)

- if input is sinusoid of frequency f, then output also sinusoid of same frequency, f & attenuated by an amount $A(f)$ that depends on f

3. Date Rate & Bandwidth

- → transmission system can only carry a limited band of frequencies → limit the data rate that it can carry
- → digital signal of R bps → can be carried by signal bandwidth of $2R$ Hz with good accuracy
  - → bandwidth $0.5R$ Hz is sufficient for successful transmission



→ ↑bandwidth → will look more like pulse

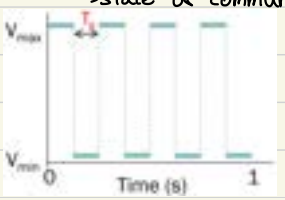
Nyquist Signalling Rate

→ If channel bandwidth = B Hz → Nyquist Signalling Rate (maximum signalling rate achieveable), $N_R = 2B$ pulses/second

Symbol Rate (baud/pulse/modulation rate)

→ number of symbol changes per second, $S = \frac{1}{T_s}$?? → symbol duration, T_s

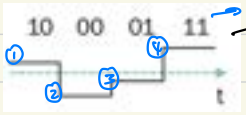
→ state of communication channel that persists for a fixed period of time



Bit Rate (gross bit/raw bit/gross data transfer rate)

↳ R_b → total number of physically transferred bits per second over a communication link, including useful data as well as protocol overhead

$R_b = S \times \log_2 M$ (S: symbol/sec, $\log_2 M$: bits/symbol) → number of modulation levels



→ 2 bit

→ No. of modulation level, $M = 4$

$\log_2 M = \log_2 4 = 2$ → 2 bits/symbol (00, 01, ...)

$R_b = S \times \log_2 4 = 2S$

* $m = 3$ bit/symbol, $M = 2^3 = 8$ → 000, 001, ...

↳ Nyquist Bit Rate, $R_m = 2B \log_2 M$ bps → Nyquist Signalling Rate, if bandwidth = B

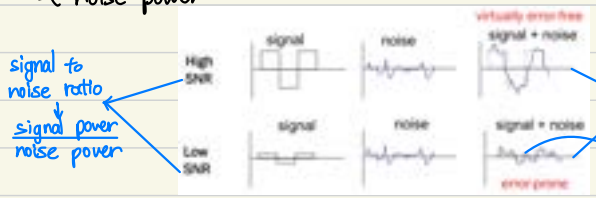
m bits/pulse × $2B$ pulse/sec

↳ in absence of noise bit rate can increased without limit by increasing M

→ Transmission system can only pass a limited band of analog frequencies → bandwidth. Together with noise, this limits data rate.

→ Data rate → no. of bps that is to be carried over transmission system

→ Channel capacity → maximum no. of bps the transmission system can sustain given bandwidth, signal power & noise power

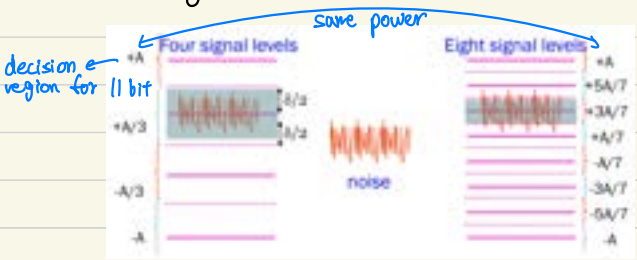


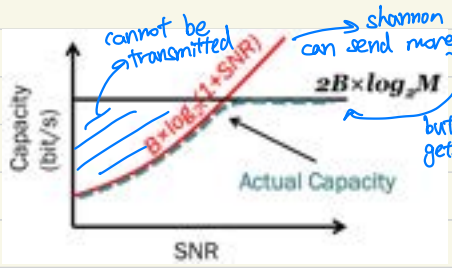
Shannon Channel Capacity (for a noisy channel)

↳ $C = B \log_2 (1+SNR)$ (B: channel bandwidth) → $SNR_{dB} = 10 \log_{10} \left(\frac{\text{signal power}}{\text{noise power}} = \frac{S}{N}\right)$

↳ if channel has a certain B & SNR → can only be used to carry data at rates lower than C bits/sec

↳ error rate depends on relative value of noise amplitude & spacing b/w signal levels → transmitted power limited, $M \uparrow$, spacing b/w levels $\downarrow$ → more errors





↳ Nyquist signal bit rate ≤ Shannon channel capacity

$2B \log_2 (M) \leq B \log_2 (1+SNR) \Rightarrow M^2 \leq 1 + \frac{S}{N} \Rightarrow M_{max} = \sqrt{\left[\frac{S+N}{N}\right]}$

$C_f = \log_2 (1+SNR)$

E.g. Consider a case where signal power is much stronger than noise power in a communication channel. Show that data rate obtained based on Shannon's Channel Capacity theorem can be approximated to be linearly proportional to SNR (in dB) with given channel bandwidth.

Given, $SNR \gg 1$, $C \approx B \log_2 (SNR)$

$= B \frac{\log_{10} SNR}{\log_{10} 2}$

$= B \frac{10 \log_{10} SNR}{10 \log_{10} 2}$

$= B \frac{SNR_{dB}}{10 \log_{10} 2}$

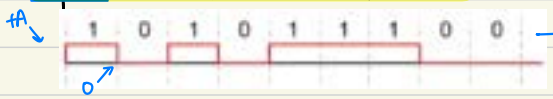
$\approx B \frac{SNR_{dB}}{3\,dB}$

Line Coding (digital signals carry digital data)

↳ digital signal to represent data bits (0 & 1) in baseband transmission

1. Unipolar Non-Return-to-Zero (NRZ)

(one polarity)



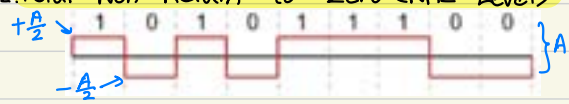
→ 1 is At, 0 is no pulse

→ Pros: easy to engineer, make good use of bandwidth (less transition, less bandwidth) ①

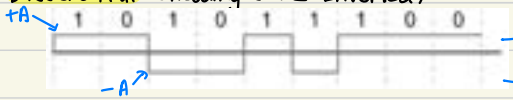
Cons: long strings of 0 & 1s cause poor timing ② & DC component (low frequency content)

↳ no synchronisation bw transmitter & receiver

2. Polar Non-Return-to-Zero (NRZ-Level)



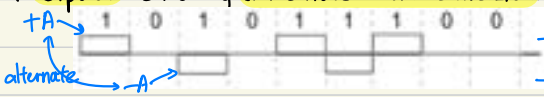
3. Differential Encoding (NRZ-Inverted)



→ "1" maps into transition, Δ in level

→ "0" maps into no transition, no Δ in level

4. Bipolar Encoding (Alternate Mark Inversion)



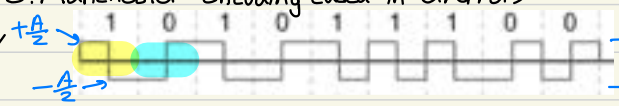
→ "1" maps to +A & -A in alternation

→ "0" maps to no pulse

↳ Pros: easy error detection ① (if have 2 errors will know), every 0ve pulse matched by 0ve pulse ② → little content at low frequencies

↳ Cons: not as efficient as NRZ ① (each element only represent one bit, but 3 level system should represent $\log_2 3 = 1.58$ bits), receiver must distinguish ② bw 3 levels instead of 2, long string of 0's ③

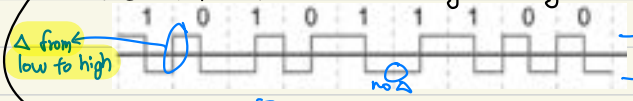
5. Manchester Encoding (used in ethernet)



→ "1" map into high-to-low transition

→ "0" map into low-to-high transition

6. Differential Manchester Encoding (used by IEEE 802.5)



→ Transition at start of bit period → "0"

→ No transition at start of bit → "1"

→ Pros: simple to implement ①, every interval has transition in middle → synchronisation not a issue ② (self-clocking feature, timing recovery easy)

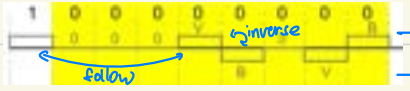
→ Cons: Use double minimum bandwidth ① (pulse rate double bit rate)

Scramble Codes

↳ AMI suitable for long distance transmission if long 0's can be avoided → use scramble code to replace sequences that will produce constant voltage

1. Bipolar with 8 Zeros Substitution (B8ZS)

↳ AMI's "00000000" → "000VB0VB"



→ "V" (violation bit) → follows polarity of previous non-zero bit

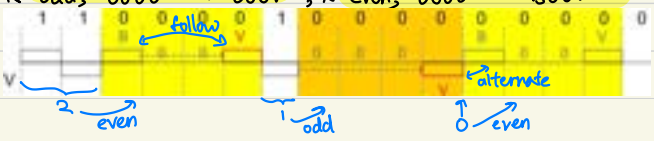
→ "B" (bipolar/balance bit) → inverse the polarity of previous non-zero bit

↳ V & B are unlikely to occur cause of noise → receiver detects & interprets as octet of all zeros

2. High Density Bipolar 3 Zeros (HDB3)

↳ count no. of bipolar bits (no. of "1"s) bw last "V" & "0000"

↳ if odd, "0000" → "000V", if even, "0000" → "B00V"



→ if have 9 → take first 8

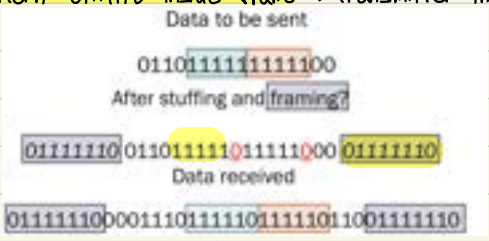
Data Link Layer

- moves a datagram from one node to an adjacent node over a single communication link, datagram transferred by different link protocols from end to end (ethernet on first link, frame relay on intermediate links, WiFi on last link)
- 2 types of link layer channels: point-to-point (e.g. HDLC), broadcast (e.g. LAN segment)

Framing & Bit stuffing

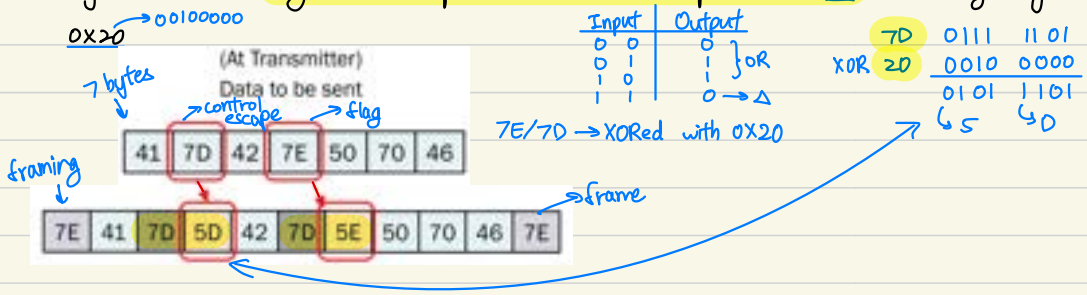
1. in HDLC

- HDLC uses bit stuffing to prevent occurrence of pattern 01111110 inside frame → transmitter inserts extra 0 after each consecutive five 1's inside frame
- Receiver checks for five consecutive 1's:
  - a) if next bit = 0 → '0' is removed
  - b) if next two bits are 10 → flag is detected
  - c) if next two bits are 11 → error indication



2. in Point-to-Point Protocol (PPP)

- PPP is character-oriented version of HDLC & uses similar frame structure as HDLC
- PPP uses same flag (7E), but uses byte stuffing
- Any occurrence of flag/control escape inside of frame is replaced with 7D followed by original octect XORed with 0X20 → 00100000



- At Receiver → remove flag (deframing) & byte destuffing

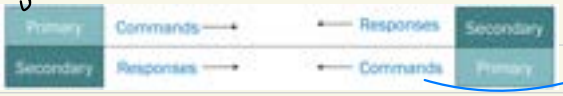
High-level Data Link Control (HDLC)

- HDLC Data Transfer Modes:
  - a) Response Mode → used in polling multidrop lines

Mode selected during connection establishment

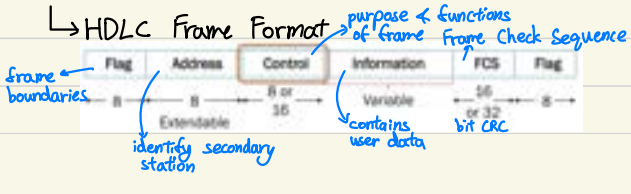
no communication bw them

  - b) Asynchronous Balanced Mode (ABM) → used in full-duplex point-to-point links



2 way & simultaneously

- HDLC Frame Format



Control Field

↳ three types of frames (first 1–2 bits of control field identify frame type):

1. **Information Frames**

1	2-4	5	6-8
0	N(S)	P/F	N(R)

P/F: poll/final bit used in intersection b/w primary & secondary

N(S): send sequence number

N(R): receive sequence no.

(A) —— N(S)=6, ACK(7) ——> (B)

(A) —— N(S)=3, N(S)=3, N(R)=7 ——> (B)

→ 1st type

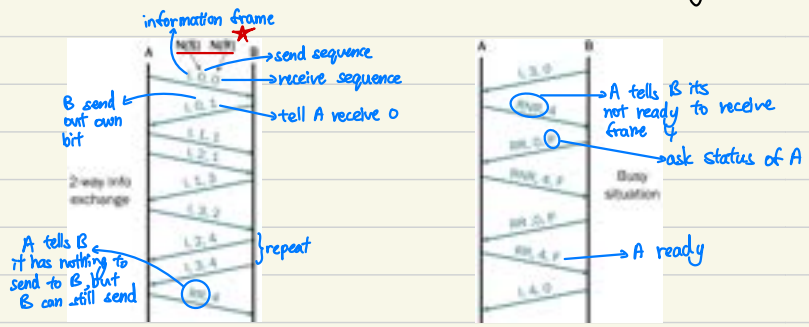
→ 2nd type (positive ACK piggybacked)

2. **Supervisory Frames** (used for error control & flow control)

1	0	S	S	P/F	N(R)

- → **Receive Ready (RR), SS=00**: e.g. RR(7) → ACKs frame up to 6 when piggybacked ACK not available (→ tells transmitter receives up to 6)
- → **Reject (REJ), SS=01**: e.g. REJ(7) → Negative ACK indicating 7 is first frame not received correctly, need resend 7 & later frames
- → **Receive Not Ready (RNR), SS=10**: e.g. RNR(7) → tell sender receive frame 6 but not ready for 7 (tell sender to slow down)
- → **Selective reject (SREJ), SS=11**: e.g. SREJ(7) → transmitter only need to transmit no. 7.

E.g.

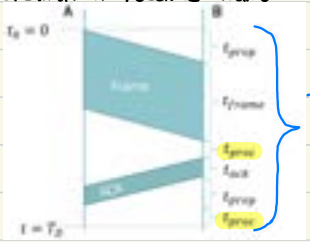


Flow Control (assume all frames received)

↳ technique for preventing the sender from overwhelming the receiver with data

1. **Stop and Wait (S&W) Scheme** (simplest flow control protocol)

↳ destination receives frame → either send an ACK to sender / withhold ACK until ready to accept another frame



T_0 = total time to transmit 1 frame

$T_0 = 2t_{prop} + 2t_{proc} + t_{frame} + t_{ack}$ (process; info frame length; ACK frame length)

$= 2t_{prop} + 2t_{proc} + \frac{L_{frame}}{R} + \frac{L_a}{R}$ (R: channel transmission rate)

- → efficiency / utilisation, $U = \frac{t_{frame}}{T_0}$ (only t_{frame} used to transmit data)
- → effective data rate (throughput), $R_{eff} = \frac{L_{frame}}{T_0}$ → header excluded

$U = \frac{R_{eff}}{R}$ (R: channel rate)

- → define $a = \frac{t_{prop}}{t_{frame}}$, ignore tack & t_{proc}, $U = \frac{1}{1+2a}$ → $a = \frac{d}{\lambda} = \frac{Rd}{LV}$ (R: data rate; L: link length; V: velocity of propagation; L: frame length)

E.g. A link has a data rate of 4Mbps & distance of 1000km. Propagation delay 5μs/km. How long does it take frame of 1000 bytes to get to other end of link? If S&W flow control used, what is link utilisation?

Transmission time, $t_{frame} = \frac{\text{frame length}}{\text{data rate}} = \frac{1000 \times 8}{4 \times 10^6} = 2ms$

Propagation delay, $t_{prop} = 5\mu s/km \times 1000km = 5ms$

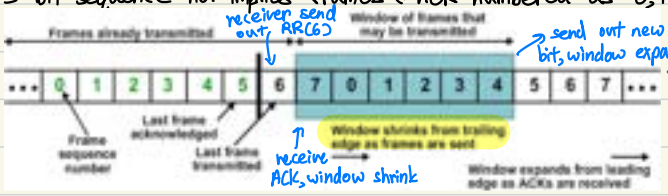
Total delay = transmission time + propagation time = 7ms

$a = \frac{5ms}{2ms} = 2.5$ → $\frac{t_{prop}}{t_{frame}}$ / $t_{transmission}$

Link utilisation, $U = \frac{1}{1+2a} = 0.17$

Sliding Window Scheme

- transmitter can send up W frames without receiving ACK → allow multiple frames to be in transit
- 3-bit sequence no. implies frames & ACK numbered as 0, 1, 2, 3, 4, 5, 6, 7



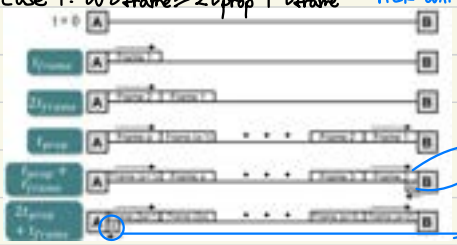
send out new bit, window expand

receive ACK, window shrink

W = 7, up to 7 consecutive frames can be sent without getting ACK

- Link Utilisation:

Case 1: $Wt_{frame} \geq 2t_{prop} + t_{frame}$ (window × t_{frame}) — before window gets to 0, ACK will reach (fully utilised)

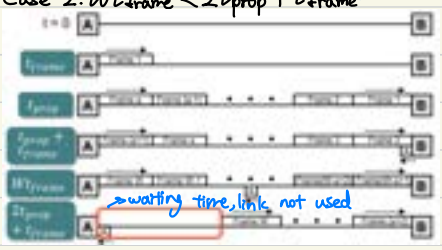


frame 1 received completely

acknowledgement send out

acknowledgement reach

Case 2: $Wt_{frame} < 2t_{prop} + t_{frame}$ → under-utilised



waiting time, link not used

$$U = \begin{cases} 1, & W \geq 2a+1 \text{, frame 1 ACK reaches source before window exhausted} \\ \frac{W}{2a+1} \left/ \frac{Wt_{frame}}{2t_{prop}+t_{frame}}, \right. & W < 2a+1 \text{, window exhausted at } t=W \text{ out of } 2at1 \text{ period} \end{cases}$$

→ almost W times more efficient than $S \& W$

$2t_{prop} + t_{frame}$

E.g. Consider an error free 64kbps satellite channel used to send 512-byte data frames in one direction, with very short ACK in other direction. What is maximum utilisation for window sizes of 1, 7, 15 & 128? Round trip propagation delay is 540ms.

$$t_{frame} = \frac{512 \times 8}{64000} = 64ms$$

$2t_{prop} = 540ms$

$T_1 = t_{frame} + 2t_{prop} = 64 + 540 = 604ms$

$\frac{T_1}{t_{frame}}$

$\frac{604ms}{64ms} = 9.43$ → need window size of at least 10 frames to keep channel busy

$W = 15 / 128 \rightarrow U = 100\%$

$W = 1, U = \frac{t_{frame}}{T_1} = 10.6\%$

$W = 7, U = \frac{7t_{frame}}{T_1} = 7 \times 10.6\% = 74.2\%$

Data Link Layer Error

- data link operates over wire-like, directly-connected system, frames can be corrupted/lost, but arrive in order → data link performs error-checking & retransmission
- detection: lost frame / damaged frame
- automatic repeat request (ARQ) function: error detection, positive acknowledgement, retransmission after timeout, negative acknowledgement & retransmission

- Types of Errors (error occures when a bit is altered bw its transmission & reception → send 0/receive 1)

- single bit errors → randomly selected bits are altered, bit errors are independent events → mostly caused by white noise
- burst errors → contiguous sequence of bits in which first bit, last bit & any no. of bits in between are in error → caused by impulse noise &/ fading in wireless channels (effect is greater at higher data rates)

- Error Detection vs Error Correction

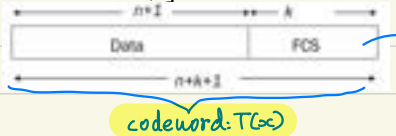
- error detection → check if any error has occurred, no need to know how many bits are in error / where bits are
- error correction → used when error detection & retransmission of corrupted & lost packets not useful (need to know)

Error Detection & Block Coding

- Divide message into blocks with same bit lengths → Dataword / Information, generate codeword by adding redundancy / checkbits to each block
- if an error changes a valid codeword to another valid codeword, then error cannot be detected

Cyclic Redundancy Check (CRC) (widely used in local/wide area network)

- consider not only the value but also the order, polynomial for codewords → powerful error-correction method



codeword: T(x)

→ frame check sequence / checkbits (chosen in such a way T(x) can be exactly divisible by chosen polynomial, C(x))

- receiver divides binary polynomial from incoming frame by the same C(x) → if no remainder, assumes no error in frame, else have error (rlly no error / error cannot be detected)
- How CDC work?
  1. Represent (n+1) bit as n-degree polynomial M(x), e.g. $101 \rightarrow 1x^2+0x+1x^0 = x^2+1$
  2. Choose divisor k-degree polynomial C(x)
  3. Compute remainder R(x) from $\frac{M(x)*x^k}{C(x)}$
  4. Add R(x) to M(x) & this is the transmitted codeword, T(x)
  5. Receiver Side → Received as T'(x). If T(x) divisible by C(x) → no error detected, else have error.

$$C(x) \overline{)\, M(x)\cdot x^k} \quad A(x)$$
$$R(x)$$

(k → shift data to left be k bit)

- Codeword transmitted, T(x), receiver receive T'(x), error pattern → E(x) = T(x) − T'(x) → error not detectable
- if C(x) divides E(x), if C(x) does not divide E(x) → error detectable

→ Binary polynomial addition = subtraction = XOR

e.g. $x^7+x^6+x^6+x^5+1 = x^7+x^5+1$ ($(1+1)x^6 = 0$)

Input		Output
0	0	0
0	1	1
1	0	1
1	1	0

E.g. Generator polynomial: $C(x) = x^3 + x + 1$ (1011)

Information: $i(x) = x^3 + x^2$ (1100) ⟶ Encoding: $x^3 i(x) = x^6 + x^5$

$$x^3+x+1 \overline{)\, x^6+x^5 \,}\quad x^2+x^2+x$$
$$x^6 + \quad x^4+x^3$$
$$x^5+x^4+x^3$$
$$+\{\; x^5 + \quad x^3+x^2$$
$$x^4 \quad +x^2$$
$$x^4 \quad +x^2+x$$
$$x$$

XOR ☆

$$1011 \overline{)\, 1100\,000 \,}\quad 1110$$
$$1011$$
$$1110$$
$$1011$$
$$1010$$
$$1011$$
$$10$$

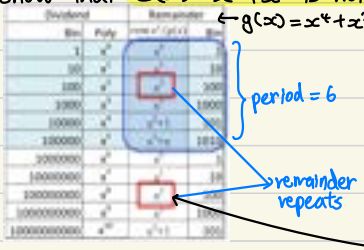
XOR ☆

$=0$

$\therefore T(x) = x^6 + x^5 + x$ (1100010)

E.g. Show that $E(x) = x^8 + x^2$ is not detectable

$g(x) = x^4 + x^2 + 1$



$\text{rem}\left[\frac{x^8}{g(x)}\right] + \text{rem}\left[\frac{x^2}{g(x)}\right] = x^2 + x^2 = 0$

↳ not detectable

$$x^4+x^2+1 \overline{)\, x^8 \,}\quad x^4 - x^2$$
$$\vdots$$
$$x^2$$

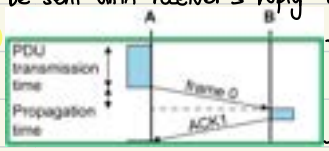
$$x^4+x^2+1 \overline{)\, x^2 \,}\quad 0$$
$$\vdots$$
$$x^2$$

Automatic Repeat Request (ARQ)

↳ to ensure sequence of information packets are delivered in order & without errors or duplications despite transmission error & losses (turn unreliable to reliable)

1. Stop-and-Wait ARQ

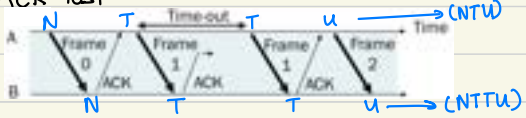
- after transmitting single frame, source sets timer, no other data can be sent until receiver's reply arrives. If no ACK received within timeout period (→ either data frame error/ACK error), source retransmits frame, resets timer
- pro: simplicity, con: inefficient
- Need sequence numbers:



data frame_success & ACK_success

a) Data frame lost

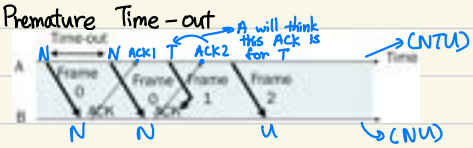
b) ACK lost



No error: $U = \frac{t_{frame}}{t_{total}}$ (→ same as above)

$U = \frac{1}{1+2a}$, $a = \frac{t_{prop}}{t_{frame}}$

c) Premature Time-out



☆
- error occurs: $U = \frac{t_{frame}}{Nr\, t_{total}} = (1-p)\cdot\frac{1}{1+2a}$, $Nr = \frac{1}{1-p}$ → expected no. of attempts for one successful frame transmission → probability frame arrives with errors
- channel with bit error probability (BER), $U = \frac{(1-BER)^L}{1+2a}$ → probability of L-bit frame is error-free

depend on length of frame & channel characteristics

no error, $U = \frac{1}{1+2a} = 0.17$

E.g. $t_{prop} = 5ms$, $t_{frame} = 2ms$. If bit error rate (BER) are $p = 10^{-4}$ / $p = 10^{-5}$, compare link efficiency for frame size of 1000 bytes.

$L = 1000$ bytes $= 8000$ bits, $a = 2.5$

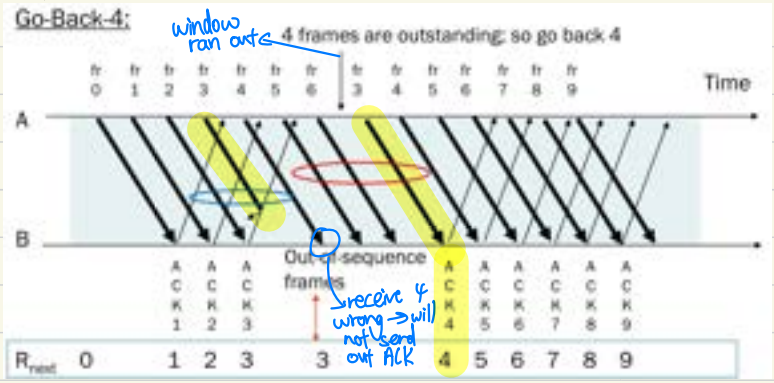
For $p = 10^{-4}$, $(1-10^{-4})^{8000} = 0.45 \longrightarrow U = \frac{0.45}{1+2(2.5)} = 0.075$

For $p = 10^{-5}$, $(1-10^{-5})^{8000} = 0.45 \longrightarrow U = 0.153$ → better → but not as good as 0.17

2. Go-Back-N ARQ (inefficient → multiple frames are resent when error/losses occur)

→ keep channel busy by continuously sending frames based on sliding window → after transmission window is exhausted, retransmit from last acknowledged frame (Max window size = $2^k - 1$) → k-bit sequence no.

→ when out of sequence frame received → reject the frame & discard frame & all subsequent frames, may send negative acknowledgment indicating expected frame → transmitter must go back & retransmit expected & all subsequent frames in window



→ After A exhaust its transmit window, retransmit 3, 4, 5, 6

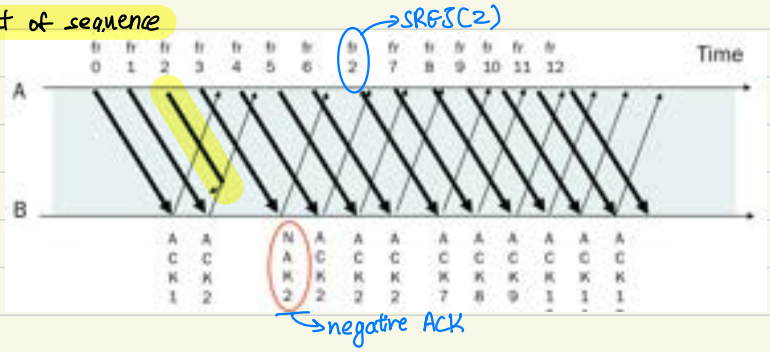
3. Selective Repeat (Reject) ARQ (minimises retransmission)

→ retransmit only individual frame → timeout causes individual corresponding frame to be resent, NAK causes retransmission of oldest un-acked frame (Max window size = 2^{k-1}) → k-bit sequence no.

→ more complex logic in transmitter & receiver:

a) receiver must contain storage to save post-NAK frames until frame in error is retransmitted & must contain logic for reinserting frame in proper sequence

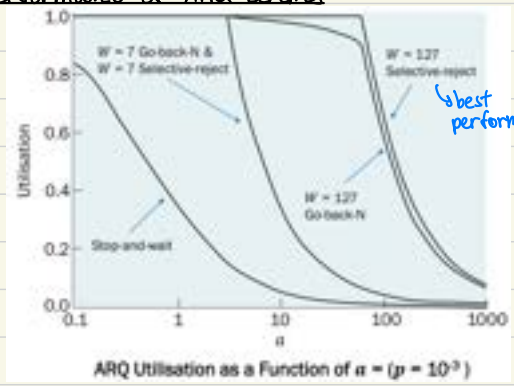
b) transmitter require buffer to store copies of unacknowledged packets & require complex logic to send frames out of sequence



→ retransmit only frame in error

→ store other frame in buffer of receiver, wait for error frame to be received correctly

Performance of ARQ Schemes



Multiple Access Links

→ connects multiple network devices with link being shared medium → one network device transmit on medium, all network device can receive it. More than one transmit at same time → collision

→ Need for **Medium Access Control (MAC) protocol** to govern how nodes can share a medium. And **communication among nodes about medium sharing must be medium itself (in-band signaling)**, else out-of-band signaling. ✓ ✗

→ 3 broad classes of MAC protocols:

a) channel partitioning → divide channel into smaller pieces, allocate piece to node for exclusive use

b) random access → channel not divided, allow collision, need "recover" from collisions

c) "taking turns" → nodes take turns, nodes with more to send can take longer turns

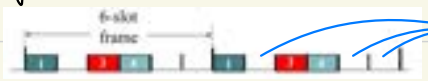
Channel Partitioning

1. Time Division Multiple Access (TDMA)

→ divides time into time frames & divide each time frame into N time slots → each time slot assigned to one of N nodes.

→ node have packet to send → transmit during assigned time slot (transmission rate of $\frac{R}{N}$ bps) → eliminate collision

(R → data rate in shared medium)



2. Frequency Division Multiple Access (FDMA)

→ divides R bps medium into different frequencies, each bandwidth $\frac{R}{N}$ → assigns each frequency to one of N node

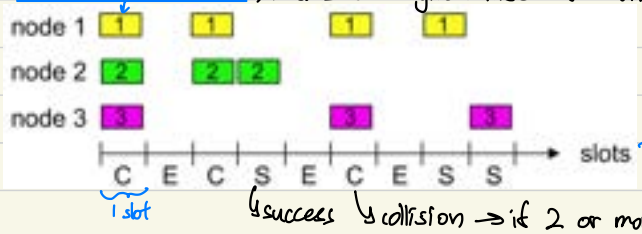
Disadvantages: node limited to average rate of $\frac{R}{N}$ bps even when its the only node to send (1), node must always wait for its turn (2), unused slots go idle (3)

Random Access Protocols

→ transmitting node always transmit at full rate of channel (R bps) → when experience collision → wait for random delay before transmitting frame

1. Slotted ALOHA

→ all frames same size, nodes are synchronised & start to transmit only at beginning of slot



→ each slot (time taken to transmit 1 frame)

→ collision → if 2 or more nodes transmits in slot → sending node will know as it did not receive acknowledgement from receiving station within slot

→ collision → sending node decides whether to retransmit its frame in following slots with probability p

- probability that given node has success in a slot $= p(1-p)^{N-1}$ ($N-1$ → no. of nodes with frames to send)
- probability that any node has success in slot $= Np(1-p)^{N-1}$

(If all nodes same probability)

- $N \to \infty$, max efficiency $= \frac{1}{e} = 0.37$

→ Pros: single node can continuously transmit at full rate of channel, only slots in nodes need to be in sync, simple

→ Cons: collision, idle slots, slot synchronisation

E.g. 2 active nodes A & B, both has infinite no. of packets to send. Node A (probability = a), node B (probability = b).

What is probability that Node A succeed for first time in slot 6? What is the efficiency of system?

A	B	
(1-a)	(1-b)	no node transmits
(1-a)	b	node B succeeds
a	(1-b)	node A succeeds
a	b	collision

→ Node A successful $= a(1-b)$

→ Node A fail $= 1-a(1-b) = 1-a+ab$

P(Node A successful for 1st time in slot 6) = $(1-a+ab)^5 a(1-b)$

Efficiency of system = P(successful transmission in each slot)

$= a(1-b) + b(1-a) = a+b-2ab$

2. ALOHA (unslotted)

→ frame first arrive → transmit immediately → collision probability ↑

→ frame sent at t_0 to collide with frame sent at t_0-F, t_0+F

- probability that given node has success in slot = $p(1-p)^{N-1}(1-p)^{N-1} = p(1-p)^{2(N-1)}$ (node transmit; no other node transmit)
- $N \to \infty$ → Max efficiency = $\frac{1}{2e} = 0.18$

→ Pros: simpler, no synchronisation

E.g. Want to use pure ALOHA for field communication system. Data transmission of system = 10Mbps. Each field station need to transmit 40 frames per second, average frame size of 1000 bytes. Estimate max no. of field stations network can support. If slotted ALOHA used, what is the max no. of field stations network can support?

max efficiency of aloha = $\frac{1}{2e} = 0.18$

max throughput = 10Mbps × 0.18 = 1.8Mbps → actual data rate it can support

traffic generated by each field station = 40 Fps × 1000 × 8 = 0.32Mbps

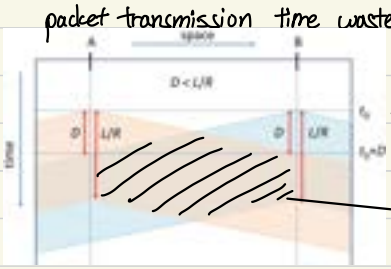
max no. of station supported = $\frac{1.8}{0.32} = 5.6 \approx 5$

Slotted aloha → max no. of station = $\frac{3.7}{0.32} = 11.6 \approx 11$

3. Carrier Sense Multiple Access (CSMA)

→ listen before transmit → if channel sensed idle → transmit entire frame, else wait

→ collision still occur → propagation delay means 2 nodes may not hear each other's transmission → entire packet transmission time wasted



→ At t_0 → both nodes A & B think that channel empty as A transmission will only reach B at t_0+D

→ collision happens

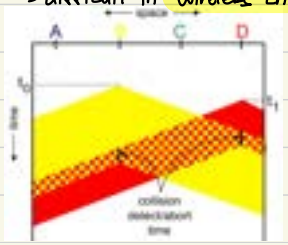
→ distance ↑, propagation delay ↑ → ↑ chance of collision

4. Carrier Sense Multiple Access / Collision Detection (CSMA/CD)

→ collision detected within a short time → colliding transmission aborted, reducing channel wastage

→ easy in wired LAN → measure signal strength / compare transmitted & received signal

→ difficult in wireless LAN → received signal strength overwhelmed by local transmission strength



"Taking Turns" Protocols

1. Polling
   - → one node is master node → polls each node in round-robin fashion. Can inform station being polled up to how many bits it can transmit, determine node finished by observing lack of signal on channel
   - → Pros: eliminates collisions & empty slot → much higher efficiency
   - → Cons: polling overhead, latency → node can only send limited no. of frames & wait to be polled again, master node is single point of failure

2. Token Passing
   - → short token frame is circulated among nodes.
   - → node no frame to transmit → release token immediately, else send up maximum number of frames & release token
   - → Pros: highly decentralised (→ distribute info across multiple devices, instead of a central server), efficient
   - → Cons: failure of one node can crash network, token corrupted/station fail to release token → recovery procedure needed to recover token

E.g. Token Ring has S stations, spaced apart equally, operates at data rate R Mbps. Propagation delay for frame to circulate entire ring network = D seconds. Assume token released when first bit of frame reaches transmitting station after making loop around ring. What is the throughput & efficiency of token ring network if every station will transmit a frame of F bytes whenever it receives a token?

(D > frame transmission delay)

- throughput → rate of message delivery over a communication channel
- efficiency = $\frac{\text{throughput}}{\text{data rate}}$

one transmission cycle $= (D + \frac{F}{S})$ second

no. of bits transmitted in cycle $= SF$ bits

$\text{throughput} = SF \div (D + \frac{F}{S}) = \frac{SFS}{DS+F}$ bps

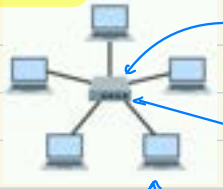
$\text{efficiency} = \frac{SFS}{DS+F} \div (10^6 R) = 8 \times 10^{-6} \frac{FS}{DRS+F}$

MAC Addresses

- → function: get frame from one interface to another physically-connected interface (→ laptop (1 fast ethernet NIC & wifi NIC))
- → every host is connected to LAN via 1/more network interface cards (NICs) → MAC address normally assigned by manufacturer of NIC & stored in hardware (ROM)
- → MAC-48 (48 bits, 6 pairs of 2 hexadecimal no.s): 00-18-77-46-F4-34 (manufacturer)
  - (have other but won't discuss)
  - FF-FF-FF-FF-FF-FF (→ 4 bits (0-F → 16 possible value, 2^4) → $2 \times 4 \times 6 = 48$)
  - MAC broadcast address
- → When NIC_A want to send frame to NIC_B, insert NIC_B's MAC address into frame & send frame into LAN. If LAN is broadcast LAN, frame received & pocessed by all other NICs. Each NIC checks if destination MAC address in frame matches its own. Match → NIC extracts enclosed datagram, else discard

Ethernet

- → 70s to mid 90s → bus topology is popular → all nodes in same collision domain



- hub (layer 1) → computer connected to same hub → one station transmit, all can hear. If more than 1 transmit at same time → collision
- switch (layer 2) → can filter traffic → no collision (separate collision domain)

- → in 80s → hub-based star emerged. Now is switch-based star topology
- → Ethernet Frame Structure (512 bits in total)

- Preamble (use to synchronise receiver, sender clock rates) → seven bytes with pattern 10101010 followed by 1 byte with pattern 10101011
- Addresses (6 bytes) → if adapter receives frame with matching destination address / broadcast address → passes data in frame to network layer protocol, else discards frame
- Type → indicate higher layer protocol (mostly IP)
- CRC → cyclic redundancy check at receiver → error detected, frame dropped

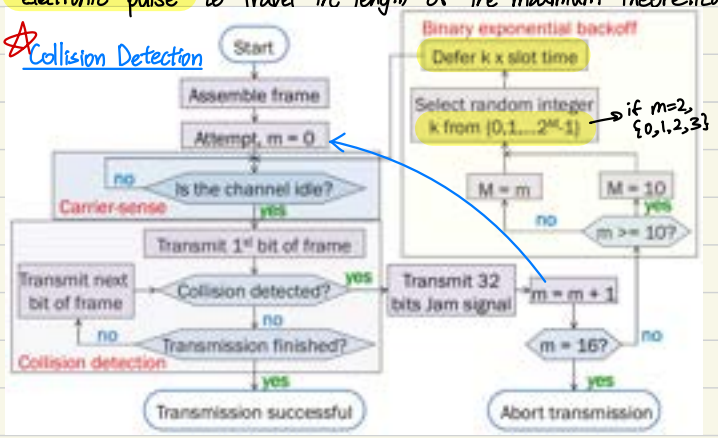
→ Cons: 1. Connectionless → no shaking btw sending & receiving NICs.

2. Unreliable → Receiving NIC doesn't send acks/nacks to sending NIC, stream of datagrams passed to network layer can have gaps (Gaps will be filled if app using TCP, otherwise no)

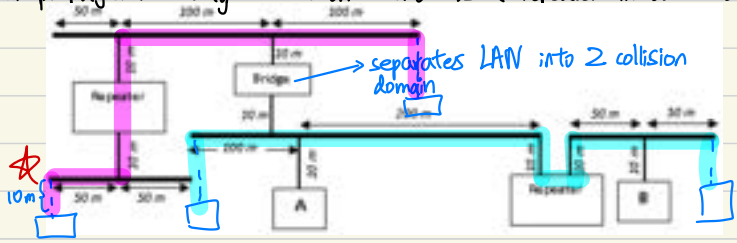
Ethernet CSMA/CD Algorithm

→ default slot time in CSMA/CD Network is 512 bits. Slot time must be twice the time it takes for an electronic pulse to travel the length of the maximum theoretical distance btw 2 nodes.

Collision Detection



E.g. For 10 Mb/s (→ bit) Ethernet shown in diagram, host can connect to any point in network with drop cable. Assume signal propagation velocity of network = 2×10^8 m/s & repeater introduces delay of 8 bits. Find minimum slot time.



Delay for repeater $= \frac{8}{10 \times 10^6} = 0.8\,\mu s$

Slot time for network on LHS $= 2\left(\frac{10+100+100+10+10+50+10}{2 \times 10^8} + 0.8\,\mu s\right) = 4.5\,\mu s$

Slot time for network on RHS $= 2\left(\frac{10+100+200+10+10+50+50+10}{2 \times 10^8} + 0.8\,\mu s\right) = 6.0\,\mu s$

∴ minimum slot time = 6.0 μs (→ worst case)

E.g. How much delay will k=4 correspond to on a 10Mbps Ethernet? Assume slot time = 512 bits (→ random int above)

Slot time $= \frac{512}{10 \times 10^6} = 51.2\,\mu s$

∴ Delay $= 4 \times 51.2\,\mu s = 0.2048\,ms$

E.g. CSMA/CD has 2 stations A & B. Station A needs transmit frame after 4th collision & Station B need to transmit frame after 5th collision, what is the probability they collide again?

A (4th collision) → $\{0, 1, \ldots, 2^4 - 1\} = \{0, 1, 2 \ldots 15\}$

B (5th collision) → $\{0,1,\ldots,2^5-1\} = \{0,1,2,\ldots,31\}$

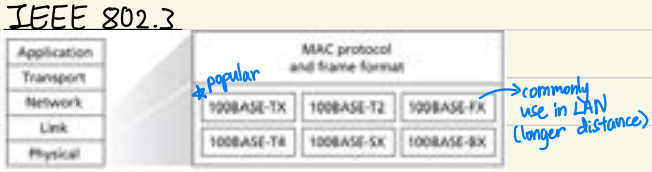
(0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)	…	(0,31)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	…	(1,31)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	…	(2,31)
:							
(15,0)	(15,1)	(15,2)	(15,3)	(15,4)	(15,5)	…	(15,31)

16×32 possible combi & 16 combi that cause collision.

$P(\text{another collision}) = \frac{16}{16\times 32} = 0.03125$

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

IEEE 802.3



E.g. 100 Mbps 100BASE-T Ethernet with all nodes connected to hub. Assume frame length of 64 bytes & signal propagate speed of 2×10^8 m/s. To have efficiency of 0.5, what should max distance bw node & hub be?

$t_{trans} = \frac{64\times 8}{100\times 10^6} = 5.12\,\mu s$

$\text{efficiency} = 0.5 = \frac{1}{1+5a} \rightarrow a = 0.2$

$t_{prop} = a \cdot t_{trans} = 0.2\times 5.12 = 1.024\,\mu s$

End-to-end distance bw 2 nodes = $1.024\times(2\times 10^8) = 204.8$

∴ max distance bw node & hub = 102.4 m → close to 100BASE-T standard of 100m

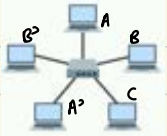
Switch

- → transparent → host are unaware of presence of switches, self-learning → switches do not need to be configured
- → rate at frames arrive to any one of switch's output interfaces may temporarily exceed link capacity → switch output interface have buffers
- → filtering → switch function that determines if frame should be forwarded/dropped
- → forwarding → switch function that determines the interface frame should be directed & move frame to those interface

(filtering & forwarding: done with switch table)

- → Each switch has switch table, each entry contains: MAC address of host, interface to reach host, time stamp
- → When frame received:

1. Record link associated with sending host
2. Index switch table using MAC destination address & set a default time-to-live (TTL) value (reaches 0 → discard frame)
3. If entry found in switch table (& if message meant for destination its from, drop it), else forward the frame on interface indicated, else cannot find on destination, flood (forward on all but not the interface on which the frame arrived)



MAC Address	Interface	TTL
A	1	60
A'	4	60

→ don't have A' stored → flood, when A' receiving it, accepts it & send frame back → switch create A' in table, others will discard frame

E.g. Switch table of an 8-port switch at t0 shown below. Assume aging time for MAC address table entries is 100s.

i) If from A with source address M2, destination address M8 received on port 2 at t+40s. How would switch table look?

MAC address	Port Number	Time-to-Live
M2	2	53
M6	3	15
M7	4	90
M9	4	43
M3	4	77
M5	6	62

→

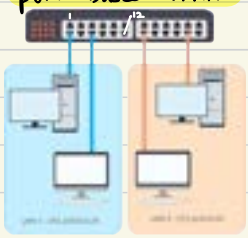
MAC address	Port Number	Time-to-Live (TTL)
M2	2	~~53~~ 100
M7	4	~~90~~ 50
M9	4	~~43~~ 3
M3	4	~~77~~ 37
M5	6	~~62~~ 22

ii) Which port will frame be forwarded to? → Frame will be flooded, every port except port 2

iii) Suggest why there are 3 entries in switch table associated with port 4? → Switch could reach all of them via port 4, what connected to port 4 is another switch

Virtual Local Area Network (VLANS)

- → segregate physical LAN into several smaller broadcast domain → reduce broadcast traffic
- → reduce the need to have routers deployed on network to contain broadcast traffic
- → when user Δ department, no physical re-wiring & other Δs needed → very flexible
- → port-based VLAN:

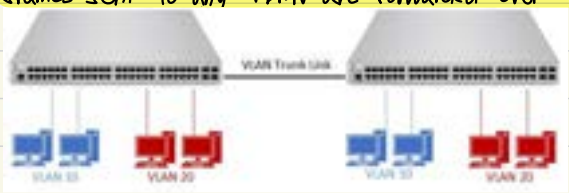


a) Traffic isolation (frames to/from port 1-12 can only reach ports 1-12) → can define VLAN base on MAC address of endpoint & application rather than switch port

b) Dynamic membership → ports can by dynamically assigned among VLANs.

c) Forwarding bw VLANs (done via routing) → in practice, vendors sell switch & router integrated into one piece of network (multi-layer switch)

- → Drawbacks of VLAN: Lack of traffic isolation, Inefficient use of switches, managing users
  - → can be handled by switch that supports VLAN → allows multiple virtual LANs to be defined over a single physical LAN infrastructure.
- → VLAN Trunking (used when multiple VLAN span over multiple switches) → special port on each switch is configured as a trunk port to interconnect 2 VLAN switches, trunk port belongs to all VLAN & frames sent to any VLAN are forwarded over trunk link to other switch



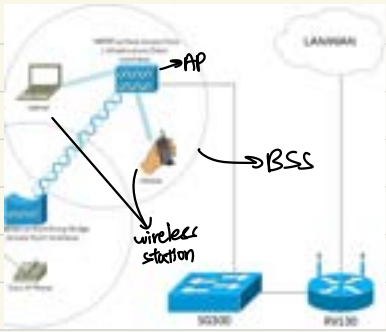
Dest. Address	Source Address	802.1Q VLAN Tag	Type	Data	CRC

↑ carries identity of VLAN which frame belongs (added by switch at sending side of VLAN trunk, parsed & removed by switch at receiving end of trunk

Wireless LAN (WLAN/Wi-Fi)

→ IEEE 802.11 standards

IEEE Standard	802.11a	802.11b	802.11g	802.11n	802.11ac
Year Adopted	1999	1999	2003	2009	2014
Frequency	5 GHz	2.4 GHz	2.4 GHz	2.4/5 GHz	5 GHz
Max. Data Rate	54 Mbps	11 Mbps	54 Mbps	600 Mbps	1 Gbps
Typical Range Indoors	100 ft.	100 ft.	125 ft.	225 ft.	90 ft.
Typical Range Outdoors	400 ft.	450 ft.	450 ft.	825 ft.	1,000 ft.



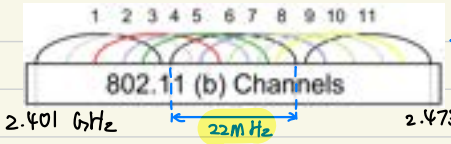
→ Infrastructure Topology → wireless stations may act as clients & WLAN access point (AP) acts as server

→ AP may connect to the internet & AP covers an area with no. of wireless stations → tgt called basic service set (BSS)

→ station sends message to AP → AP forward message to destination based on MAC address of destination (both station & AP uses 48-bit MAC address)

→ Service Set Identifier (SSID/wifi name) & channel no. are configured by administrators

→ 802.11 Channels: 2.4G Hz – 2.485G Hz divided into 11 channels at different frequencies



→ interference possible

→ to gain internet access → wireless station need to associate with exactly one of the AP (→ create virtual wire)

→ 802.11 Passive/Active Scanning

→ beacon frames sent from AP

→ association request frame sent: H1 to selected AP (→ station choose one)

→ association response frame sent: selected AP to H1

→ probes request frame broadcast from H1

→ probes response frame sent from APs

→ association request frame sent: H1 to selected AP

→ association response frame sent: selected AP to H1

Ad Hoc Topology

→ provide communication b/w no. of terminal as free-style connection (no AP needed) → wireless station can directly communicate with each other within transmission range → all these station form independent basic service set (IBSS)

MAC Services

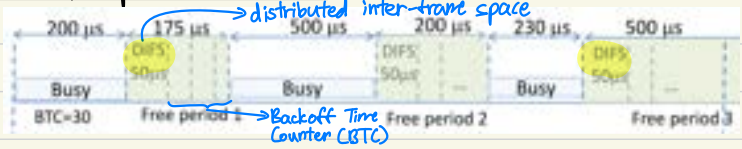
→ MAC can alternate between point coordination function (PCF) & distributed coordination function (DCF)

- PCF is non-contention based & operates over DCF → when PCF used → time on medium divided into contention-free period (CFP) → controlled by PCF & contention period (CP) → controlled by DCF. PCF make use of DCF to secure medium to use polling mechanism to grant access to different wireless station
- DCF is contention-based & is based on carrier-sense multiple access/collision avoidance (CSMA/CA) (→ different from collision)

DCF CSMA with Collision Avoidance (CSMA/CA)

- hidden terminal problem → 2 wireless stations are within range of AP & associated with it but stations are hidden from each other → both transmit to AP at same time → collision
- Collision Avoidance (CA) → allow sender to "reserve" channel rather than random access of data frames:
  1. Sending station first transmit short Request-to-Send (RTS) frame using CSMA (may collide but frame short)
  2. Receiving station broadcast Clear-to-Send (CTS) frame in response, heard by all nodes → reserves channel for sender & notifying possible hidden stations (other station defer transmission)
  3. After data frame sent → receiver broadcast acknowledgement (ACK) to inform sender it receives data frame & signal to others transmission complete

★ Binary Exponential Backoff



- When station senses channel is busy, wait until channel becomes idle for DIFS period & allow BTC to count down (BTC) ↓ by 1 every a period equal to slot time elapses. When BTC = 0, wireless station allowed to start transmission using RTS-CTS process. (above)
- if transmission unsuccessful, BTC regenerated with range of backoff values (contention window/CW) doubling.
- receiving stations of error-free frames send ACK → received before timeout period → successful transmission → CW reset to minimum contention window size (CW_{min}) → depend on physical layer

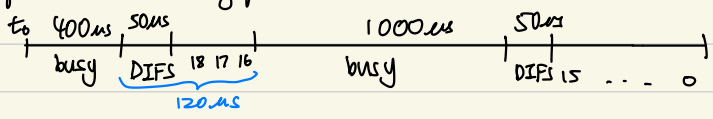
E.g. Assure $CW_{max} = 1023$ & $CW_{min} = 63$. Let f denote the no. of times wireless host has failed in transmitting frame. What is the range of random number gererated for BTC of wireless host for $f = 0, 1, 3, 5$?

$CW_{max} = 1023 \rightarrow \log_2(1023+1) = 10$

$CW_{min} = 63 \rightarrow \log_2(63+1) = 6 \rightarrow m = 6$

	$k = \min\{m+f, M=10\}$	Range for BTC $= \{0, 1, 2 \ldots 2^k - 1\}$
$f=0$	6	$\{0, 1, 2 \ldots 63\}$
$f=1$	7	$\{0, 1, 2 \ldots 127\}$
$f=3$	9	$\{0, 1, 2 \ldots 511\}$
$f=5$	10	$\{0, 1, 2 \ldots 1023\}$

E.g. In wireless network, all stations use IEEE 802.11 MAC protocol, DIFS = 50 µs, slot time = 20 µs. Station A gererate random valve 18 for its BTC at t_0. Carrier sensing mechanism of station A detect medium: busy for 400 µs at t_0, free for 120 µs, busy for 1000 µs, free long enough for BTC to expire at t_1. Compute actual time gap bw t_1 & t_0



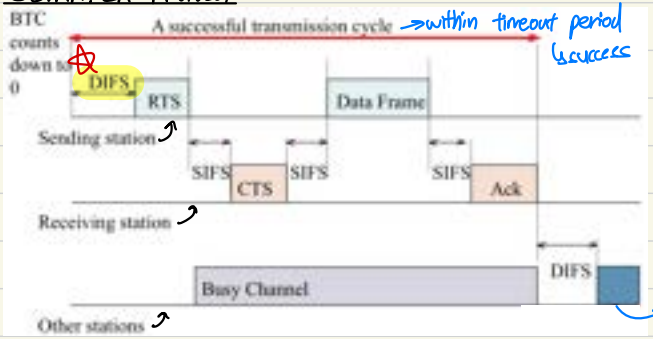
$\frac{120-50}{20} = 3.5 \approx 3$

$15 \times 20 = 300\,\mu s$

$\therefore$ actual time $= 400 + 120 + 1000 + 50 + 300$

$= 1870\,\mu s = 1.87\,ms$

CSMA/CA Protocol



→within timeout period (success)

→after BTC=0, medium idle for duration of DIFS, sending station sends RTS to AP

→Short Inter-frame Space (IFS) required to be maintained b/w RTS, CTS, Data & ACK frames

→available for transmission

E.g. Suppose 11Mbps 802.11b station is configured to always reserve the channel with RTS/CTS sequence. Suppose station suddenly want to transmit 1000 bytes & no bit error & other stations idle, RTS frame = 20 bytes, CTS frame = 14 bytes, ACK frame = 14 bytes, DIFS = 50μs & SIFS = 20μs. Assume header & trailer of IEEE 802.11 frame = 34 bytes, calculate time required to transmit frame & receive acknowledgement.

Time required to trasmit 1000 byte $= \frac{8272}{11\times10^6} = 752.0\mu s$

RTS frame $= \frac{20\times8}{11\times10^6} = 14.5\mu s$

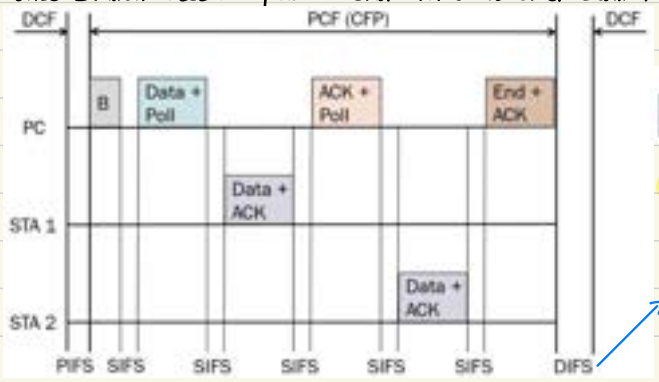
ACK, CTS frame $= \frac{14\times8}{11\times10^6} = 10.2\mu s$

$T = 50+14.5+20+(0.2+20+752.0+20+(0.2)\,ms = 896.90\mu s$

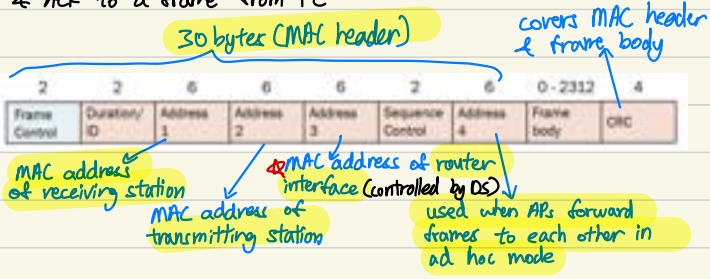
Point Coordination Function (PCF) — Polling

→do not have to wait for BTC to =0 to seize channel for transmission

1. Station need to register that they want to send data using PCF
2. Point Coordinator (PC) within AP senses channel
3. If channel free for PCF Interframe Space (PIFS, 30μs for 802.11b) → PC sends beacon frame that contains the contention-free period (CFP)
4. usual DCF operation prevented for period of CFP
5. After SIFS, PC poll each station on list for data transmission/reception
6. Once station receives poll → send frame to other station & ACK to a frame from PC

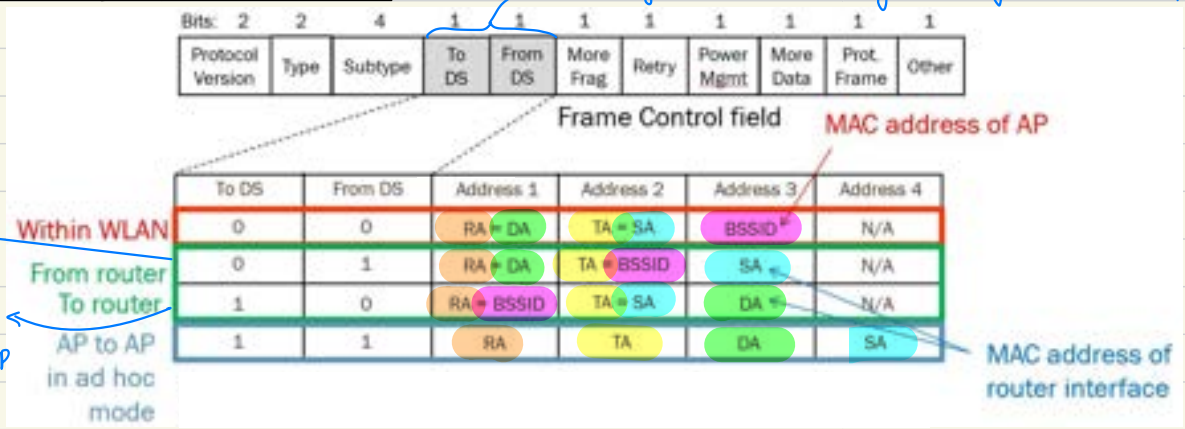


distributed interframe space (DIFS) end of CFP & beginning of CP when DCF takes over control



IEEE 802.11 Frame Structure

→ identify if frame is entering or leaving the environment



when frame is sent from router to H1 via the AP

when frame is sent from H1 to router via the AP

- SA → MAC address of original sender (wired/wireless)
- DA → MAC address of final destination (wired/wireless)
- TA → MAC address of transmitting 802.11 radio
- RA → MAC address of receiving 802.11 radio
- BSSID = L2 identifier of the BSS/MAC address of access point

RTS, CTS & ACK Frames

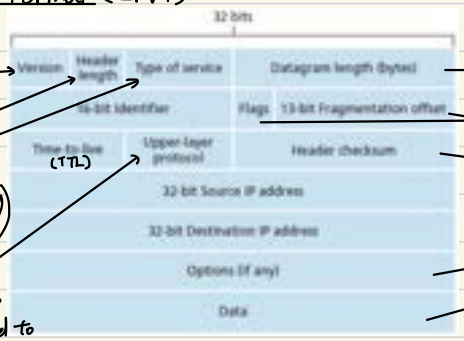
1. RTS →
2. CTS →
3. ACK →

Network Layer

- present in every host & router, responsible for host-to-host communication → transporting segments received from Transport layer of host as IP packets to another host across network
- IP packets forwarded by routers in network. Router receives IP packet, examine IP packet header, look destination IP address → extract subnet address & access routing table to find out which router interface it should forward IP packet to.
  - router exchange routing info with other routers to set up routing table
- routing network layer functions:

Transport Layer: TCP, UDP		
- Routing Protocols → path selection → routing info protocol (RIP), open shortest path first (OSPF), border gateway routing protocol (BGP) → forwarding table	- Internet Control Message Protocol (ICMP) → error reporting → router "signalling"	- IP Protocol → addressing conventions → datagram format → packet handling conventions
Link Layer		
Physical Layer		

IP Datagram Format (IPv4)

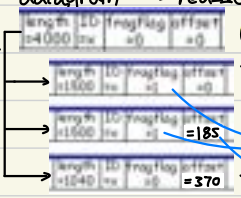


- IP Protocol Version No.
- Header Length (bytes)
- use to differentiate different IP datagrams
  - real-time application → IP telephony
  - non-real-time app → file transfer
  - router can prioritise
- indicate specific transport-layer protocol that data should be passed to
- Total datagram length (bytes) → include both header & data
- when IP datagram needs to be fragmented
- detecting error in IP header
- allow IP header to be extended, rarely used
- contains transport-layer segment & may carry other types of data

IP Fragmentation & Reassembly

- maximum transfer unit (MTU) → maximum amount of data link-layer frame can carry → each IP datagram is encapsulated within link-layer frame to transport from one router to next
- when IP datagram received by router & bigger than MTU of outgoing link → IP datagram fragmented into smaller IP datagram → reassembled at destination host (use IP header bit to identify & order fragments)

E.g.



(3980 byte data + 20 byte header)
÷3
(1480 byte data + 20 byte header each)
indicate not last datagram
1480 / 8 → 8 byte

IP Address

- 32-bit identifier for interface → connection point bw host/router & physical link
- consists of 2 parts: network part (prefix) & host (suffix)

11011111 . 00000001 . 00000001 . 00000001

223 . 1 . 1 . 1

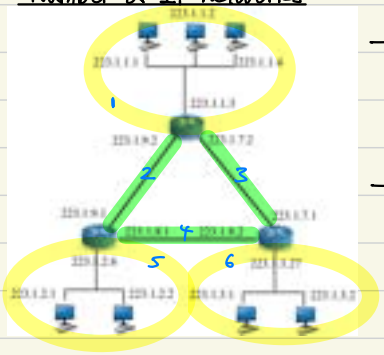
network part | host part

$1\times2^0+1\times2^1+1\times2^2+1\times2^3+1\times2^4+0\times2^5+1\times2^6+1\times2^7=223$

2 | 223	Remainder
2 | 111	— 1
2 | 55	— 1
2 | 27	— 1
2 | 13	— 1
2 | 6	— 1
2 | 3	— 0
2 | 1	— 1
0	— 1

Ans: 11011111

Number of IP networks

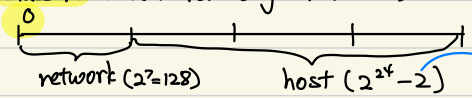


→ 1. Remove the routers
2. Create island of isolated networks → subnet

→ 6 IP network → 6 subnets

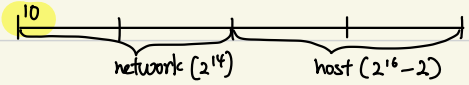
Classful IP Addressing

1. Class A (meant for large network) → 0.0.0.0 to 127.255.255.255



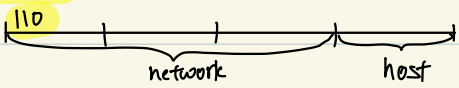
- most significant bit: 0 → 0000 0000 – 0111 1111 (0–127)

2. Class B → 128.0.0.0 to 191.255.255.255



- most significant bit: 10 → 1000 0000 – 1011 1111 (128–191)

3. Class C → 192.0.0.0 to 223.255.255.255

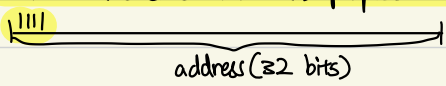


- most significant bit: 110 → 1100 0000 – 1101 1111 (192–223)

4. Class D (reserved for multicasting) → 224.0.0.0 to 239.255.255.255

- most significant bit: 1110 0000 – 1110 1111 (224–239)

5. Class E (reserved for R&D purpose & future) → 240.0.0.0 to 255.255.255.255



- most significant bit: 1111 → 1111 0000 – 1111 1111 (240–255)

E.g. 136.23.45.122 → 136 = 10001000 in binary → first 2 bits "10" → belong to class B

Special IP Address

1. Network Adress (host bits all 0s)
   - refers to network itself & not any interface in the network (e.g. 128.211.0.0/16 denote network with prefix 128.211)
   - Classless InterDomain Routing (CIDR) > Classful Adressing
     - 200.23.16.0/23 → can have 23 network bits.
     - host all 0s → network bit → 16 bits
2. Directed Broadcast Address (host bits all 1s)
   - for sending copy of datagram to all host in an IP network/subnet (e.g. 199.31.0.255/24 broadcast address for 199.31.0.0/24) — host all 1s
3. Limited Broadcast Address (255.255.255.255)
   - broadcast on local physical network during system startup by computer which don't know network address yet
4. This computer address (0.0.0.0) → used as host IP address during system startup as host yet to know its IP address
5. Loopback Address (127/8, host bit irrelevant)
   - sending datagrams from one application to another running on same host during testing (127.0.0.1 is loopback address)

Subnet Addressing (Subnetting)

- technique used by an organisation to divide IP address space into several smaller blocks to serve different subnets/clients
- several most significant bits in host part of IP address used as subnet bits

E.g. Original ISP address space: 200.23.16.0/20 (host)

→ from 11001000 00010111 00010000 00000000 to 11001000 00010111 00011111 11111111 (range of ip address; host)

To split it (borrow 3 bits for subnetting):

- Client 1's address space: 200.23.16.0/23
  - from 11001000 00010111 00010000 00000000 to 11001000 00010001 11111111
  - 4 3 2 1 0
  - borrow 3 bits
  - 0s
  - to 1s
- Client 2's address space: 200.23.18.0/23
  - 00010010 → change to 001
- Client 8's address space: 200.23.30.0/23
  - also 0s (from 000; 8 clients (2^3))
  - 00011110 → change to 111

Subnet Mask

- used by an organisation's router to extract the network part of its subnet

E.g. Subnet mask of previous example: 11111111 11111111 11111110 00000000 (23 bits) and an datagram to be sent to IP address is 11001000 00010111 00010001 01010101. What is network part of client/subnet 1?

ISP router will perform AND operation b/w datagram & subnet:

	11001000	00010111	00010001	01010101
AND	11111111	11111111	11111110	00000000
	11001000	00010111	00010000	00000000

→ 23-bit network part of client (reset network part)

AND operation

0 0	0
0 1	0
1 0	0
1 1	1

- subnet 0, all 0 at end part → both subnet 0s & all-1s subnet is usable but discouraged → normally question will tell you whether or not to use, if not told, assume smt.

E.g. A newly setup company is going to have four functional departments; namely Admin, Engineering, Sales & warehouse, with up to 25 computers/workstations in each department. Suppose this company has obtained a Class C address of 197.65.28.0 from its Internet Service Provider (ISP). Design a subnet addressing scheme & state clearly subnet no. for each subnet & range of assignable IP addresses & broadcast address.

Assume that subnet zero & all 1's subnet cannot be used,

$25 < 2^5 - 2 = 30$ → host part must have at least 5 bits

8 − 5 = 3 bits to be borrowed for subnetting → $2^3 = 8$ subnet, each subnet able to support 30 hosts

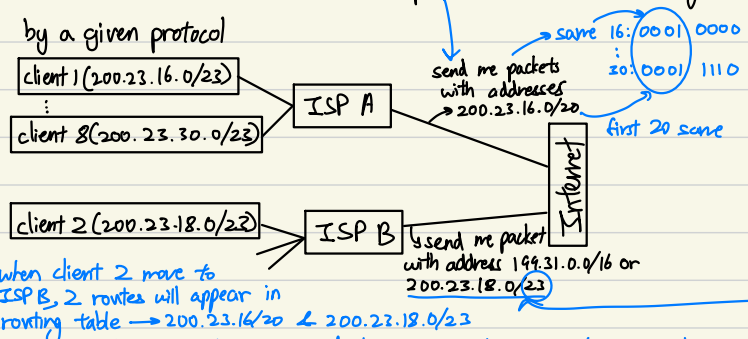
Subnet mask: 11111111 11111111 11111111 11100000 /255.255.255.224

Last 8 bits	Departments	Subnet No.	Broad Addr	Assignable IP Add
000 00000	Not used	x.0	x.31	x.1 to x.30
001 00000	Admin	x.32	x.63	x.33 to x.62
010 00000	Engineering	x.64	x.95	x.65 to x.94
011 00000	Sales	x.96	x.127	x.97 to x.126
100 00000	Warehouse	x.128	x.159	x.129 to x.158
101 00000	Reserved	x.160	x.191	x.161 to x.190
110 00000	Reserved	x.192	x.223	x.193 to x.222
111 00000	Not used	x.224	x.255	x.225 to x.254

, where x denote 197.65.28.

Route Aggregation/Summarisation

↳ used to combine a set of more specific routes into a single more general route → reduce no. of routes advertised by a given protocol



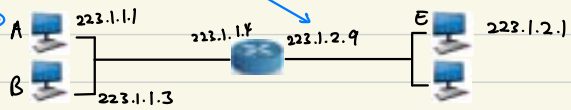
↳ when client 2 move to ISP B, 2 routes will appear in routing table → 200.23.16.0/20 & 200.23.18.0/23

↳ when IP packet received by router & destination address matches more than one route, router will choose route with longest prefix match

Forwarding Table in a Host

table in host A

Destination Net.	Next Router	Nhops (no. of hops)
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



1. Routing datagram same IP network (A to B): look up network address of B in forwarding table, B same network as A (directly connected) → link layer send datagram directly to B

2. Routing datagram from different IP network (A to router): look up network address of E in forwarding table, E on different network (not directly connected) → from routing table, next hop router to E is 223.1.1.4 → link layer send datagram to router 223.1.1.4 inside link-layer frame

Forwarding table in Router

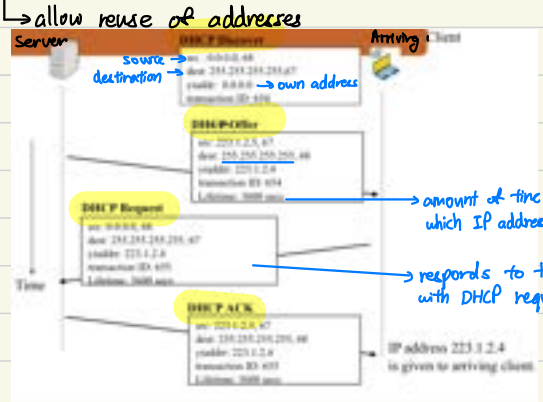
Destination Net.	Next Router	Nhops	Interface
223.1.1	-	1	223.1.1.4
223.1.2	-	1	223.1.2.9
223.1.3	-	1	223.1.3.27

2. (Router to E): datagram arrives via interface 223.1.1.4 → look up network address of E (223.1.2.2) in router's forwarding table, E on same network as interface 223.1.2.9 → router forward datagram from 223.1.1.4 to 223.1.2.9 → link layer sends datagram from interface 223.1.2.9 to 223.1.2.2

Dynamic Host Configuration Protocol (DHCP)

↳ allow host to dynamically obtain its IP address from network server when it joins network, can renew its lease on address in use

↳ allow reuse of addresses



Transport Service & Protocols

network: logical communication b/w hosts

- provide for logical communication between application process running on different host
- transport protocols run in end systems, not in network routers. Sender side: breaks application messages into segments, add transport layer & passes to network layer, becomes datagram. Receiver side: network layer extracts transport-layer segment from datagram, transport layer reassembles segment into messages, passes to application layer
- 2 main transport-layer protocols → UDP & TCP

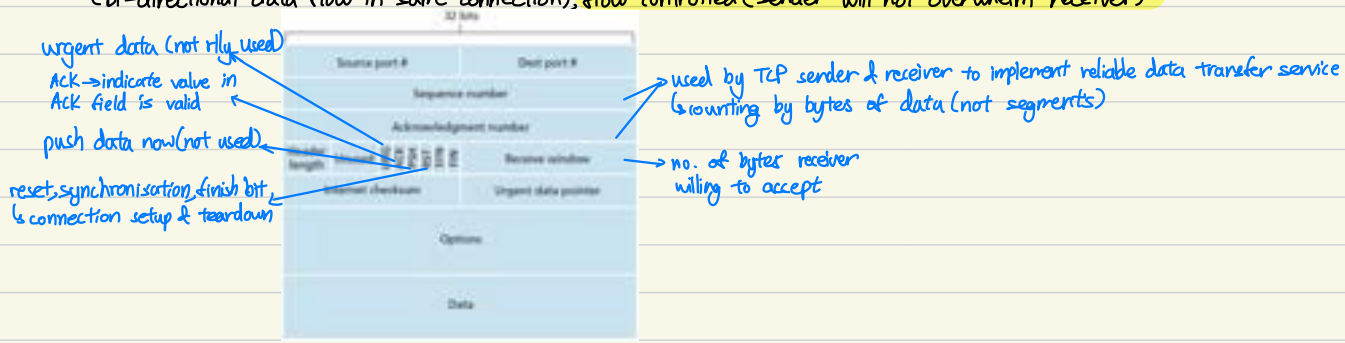
User Datagram Protocol (UDP) (for voice/video call)

- provide unreliable (UDP segments may be lost, delivered out of order to application), connectionless service (no handshaking b/w UDP sender, receiver & each UDP segment handled independently of others)
- pros: no congestion control, simple (no connection state at sender/receiver), small segment header
- if want reliable transfer over UDP → add reliability at application layer (app-specific error recovery)
- checksum field in UDP segment header provides error control over entire UDP segment

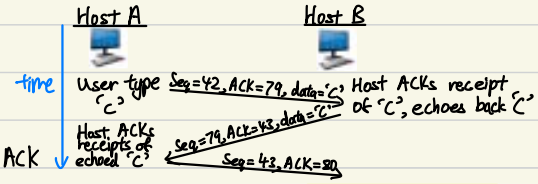
unable to provide bandwidth, delay guarantees

Transmission Control Protocol (TCP)

- provides reliable, connection-oriented service → TCP connection needs to be set up between two processors in two end system before data could be exchanged, connection also need to be torn down after data tranfer completed
- point-to-point (one sender, one receiver), pipelined (TCP congestion & flow control set window size), full duplex data (bi-directional data flow in same connection), flow controlled (sender will not overwhelm receiver)



TCP Sequence No. & ACK



- sequence no. → byte stream "number" of first byte in segment's data
- ACKs → sequence number of next byte expected from other side, cumulative ACK

E.g. Web server needs to send webpage of 5000 bytes to client over TCP connection. Maximum segment size (MSS) = 1460 bytes. total header & trailer size = 66 bytes. Assume initial sequence no. = 2380. How many TCP segments need to be transmitted & what are the sequence no. for each of these segments?

Max no. of data bytes in each segment = 1460 − 66 = 1394

$\frac{5000}{1394} = 3.6$ → no. of segment = 4

Sequence no. are 2380 (1st segment), 2380 + 1394 = 3774 (2nd segment), 5168 (3rd segment), 6562 (4th segment)

Round-trip Time & Timeout

- TCP timeout value need to be longer than RTT, too short → unnecessary retransmission, too long → slow reaction to segment loss
- New estimatedRTT = (1−α) · old estimatedRTT + α · SampleRTT
- New DevRTT = (1−β) · old DevRTT + β · |SampleRTT − new EstimatedRTT|

(absolute)

Timeout Interval = EstimatedRTT + 4 · DevRTT

↳ TCP Sender Events:

- data received from application → create segment with sequence no. → start timer if not already running (TimeOutInterval)
- Timeout out → retransmit segment that caused timeout & restart timer
- ACK received → if acknowledges previously unacked segments → update what is known to be acked, start timer if there are outgoing segment

E.g. Host A sets up TCP connection with Server B with $\alpha = 0.25$ & $\beta = 0.4$. After Host A exchanges some TCP segments with Server B, estimatedRTT = 12.6s & DevRTT = 5.4s. The following TCP segments are sent & acknowledgements are received in chronological order (denotes hour:minute:seconds):

1. Host A sends segment at 5:20:00.34 & receives acknowledgement at 5:20:11.21
2. Host A sends segment at 5:20:23.14 & receives acknowledgement at 5:20:46.42
3. Host A sends another segment at 5:20:56.11 but did not receive ACK. When timeout occurs, host A retransmit & receive acknowledgement at 5:22:10.44.
4. Host A sends another segment at 5:22:19.24 & receive ACK at 5:22:42.20.

Compute the new estimatedRTT & DevRTT for each of the four instances when ACK received by Host A. When did the timeout occur that triggered Host A to retransmit a segment?

First SampleRTT = 5:20:11.21 − 5:20:00.34 = 10.87s

Second SampleRTT = 5:20:46.42 − 5:20:23.14 = 23.28s

Fourth SampleRTT = 5:22:42.20 − 5:22:19.24 = 22.96s → 3rd sample not counted due to retransmission

$\alpha = 0.25$, $\beta = 0.4$
old EstimatedRTT = 12.6s
old DevRTT = 5.4s

	SampleRTT	New estimatedRTT	new DevRTT
1.	10.87s	0.75(12.6)+0.25(10.87)=12.17	0.6(5.4)+0.4|10.87−12.17| = 3.76
2.	23.28s	0.75(12.17)+0.25(23.28)=14.95	0.6(3.76)+0.4|23.28−14.95|=5.59
3.	NA	14.95	5.59
4.	22.96s	0.75(14.95)+0.25(22.96)=16.95	0.6(5.59)+0.4|22.96−16.95|=5.76

(old DevRTT)

TimeoutInterval = EstimatedRTT + 4(DevRTT) = 14.95 + 4(5.59) = 37.31s

∴ Timeout occurs at 5:20:56:11 + 37.31 = 5:21:33.42

Retransmission & Fast Retransmit

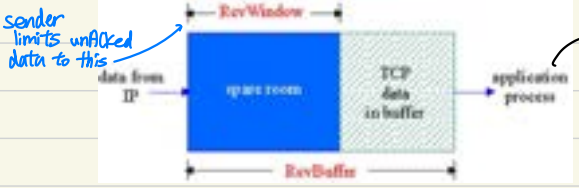
↳ timeout period can be relatively long → when segment lost → long period forces sender to delay resending lost packet, increasing end-to-end delay

↳ sender can detect packet loss well before timeout event occurs by duplicate ACKs → an ACK that acknowledges a segment for which sender has already received on earlier ACK

↳ sender sends a large no. of segments back to back → if one segment lost, likely be many back-back duplicate ACK. If TCP sender receives 3 duplicate ACKs for same data, suppose segment after ACKed data lost → fast retransmit (resend segment before timer expires)

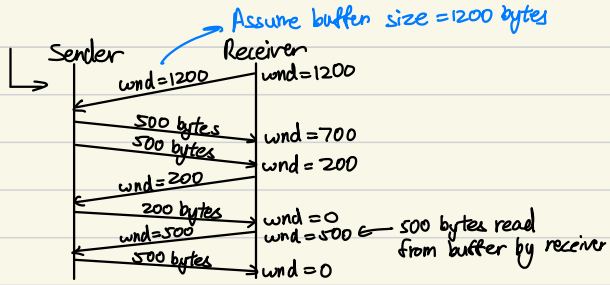
Flow Control

↳ Receive side of TCP connection has a receive buffer:



→ If this does not read data fast enough, buffer fill up
↳ TCP flow control ensure that sender won't overflow receiver's buffer by transmitting too much, too fast

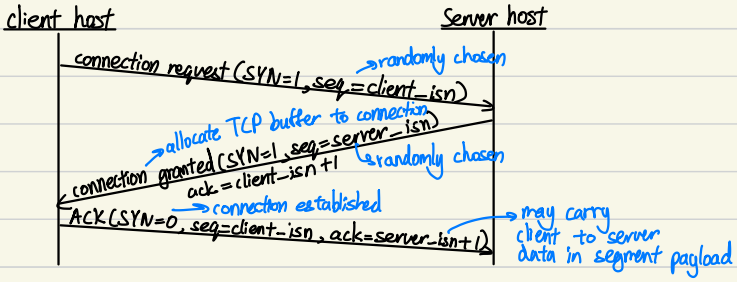
↳ Spare room in buffer = RcvWindow = RcvBuffer − [LastByteRcvd − LastByteRead]



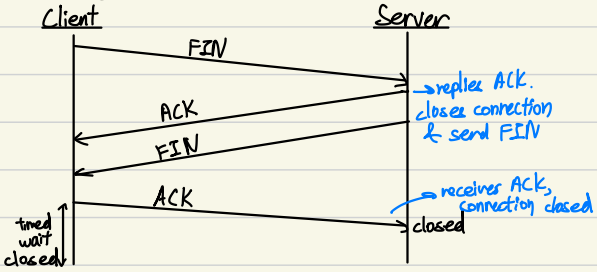
Connection Managements

1. Opening a connection

↳ process in one host wants to initiate connection with another process in another host



2. Closing connection



Checksum

- mostly used at network & transport layer.
- IP header checksum recalculated at every router
- TCP/UDP checksum provides end-to-end error detection for both transport header & transport layer data payload
- Modular Maths → $\frac{13}{5} = 2$ remainder 3 → 13 mod 5 = 3
- Congruence Modulo → $A \equiv B \pmod{C}$ (can use =) → both A & B have same remainder when divided by C
  - Rules for $a \equiv c \pmod{m}$ & $b \equiv d \pmod{m}$:
    - a) $a+b \equiv c+d \pmod{m}$ OR $a-b \equiv c-d \pmod{m}$
    - b) $-a = m \cdot n - a \pmod{m}$, n is an integer

1. Checksum Calculation: Method 1

- step 1: get $2^{16}-1 = 65535$ (16 bits of checksum)
- step 2: change the 2 values to decimal if needed (e.g. $b_0 = 33212$, $b_1 = 18522$)
- step 3: add them tgt (e.g. $b_0 + b_1 = 33212 + 18522$)
- step 4: compute checksum (e.g. $b_2 = -51734 \bmod (65535) = 65535 - 51734 = 13801$ (add into sum for receiver side))

E.g. Assume packet header is: 0100 F203 F4F5 F6F7 0000 → checksum to be calculated. Compute using modulo arithmetic approach.

0100 → 0000 0001 0000 0000 → dec = $2^8 = 256$

F203 → $2^{15}+2^{14}+2^{13}+2^{12}+2^9+2^1+2^0 = 61955$

Hex	Decimal
0100	256
F203	61955
F4F5	62709
F6F7	63223
2DEEF	188143

(11110 + 1111 = 10101; 1111 + 11110; F6F7 → 15)

$2^{16}-1 = 65536$

get the smallest no.

$65535 \times 3 - 188143 = 8462$

checksum = $-188143 \bmod (65535) = 8462$

To check: $256 + 61955 + 62709 + 63223 + 8462 = 196605$

checksum: $-196605 \bmod (65535) = 0$ (no error)

Hex	
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

2. Checksum Calculation: Method 2

- step 1: add 16-bit word using binary addition. If overflow, carryout most significant bit added by wraparou
- step 2: checksum given by one's complement of sum

E.g. Suppose block of 16 bits sent using checksum of 8 bits: 10101001 00111001. What is the final pattern set? Show how receiver determines if there is an error or not. Suppose there is a burst error of length five that affects four bits in message: 10101111 11111001

```
 10101001
+00111001
 11100010   → flip everything
 00011101   → checksum
```

```
 10101001
 00111001
+00011101
 11111111
 00000000   → checksum
∴ no errors
```

Final pattern sent 10101001 00111001 00011101

Pattern received = 10101111 11111001 00011101 (error | checksum)

```
 10101111
 11111001
+00011101
 (1)11000101
+        1
 11000110
 00111001   → checksum (error detected)
```

↳ Internet Checksum (Undetectable errors) ⟶ checksum values cannot be changed by:

a) Reordering the 16-bit word in message block

b) Inserting/deleting zero-valued 16-bit word

c) Multiple errors that cancel that effect