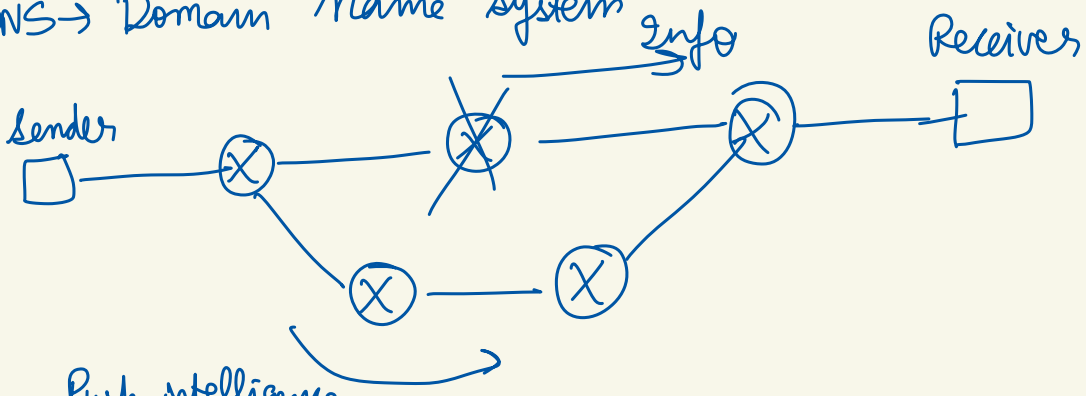
CS 348

US- ARPANET - DEFENCE

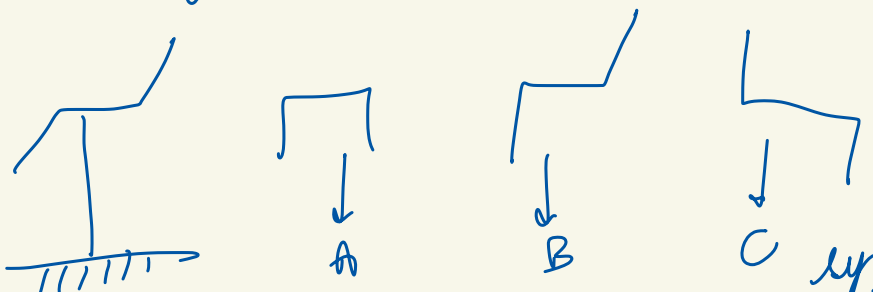
↓

Advanced Research Projects Agency NET
(forerunner to the contemporary internet)

DNS → Domain Name System

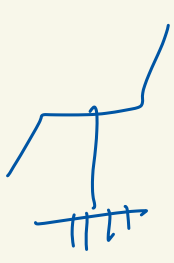


Push intelligence to edge



Semaphore

symbols
↓
bits (internet)



Repeat ... General

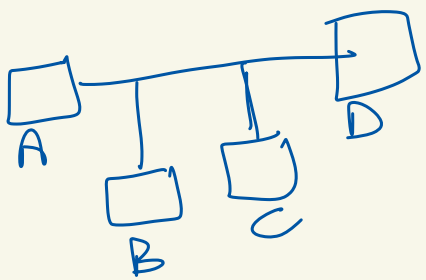
Napolean's semaphore was the world's first telegraph network.

(le systeme Chappe)

Physical Layer: Communicating bits from one node to next using signals.

Protocol: Rules formatting and transmission of data

Bus topology



Collisions because of shared medium

Medium Access Control (MAC)

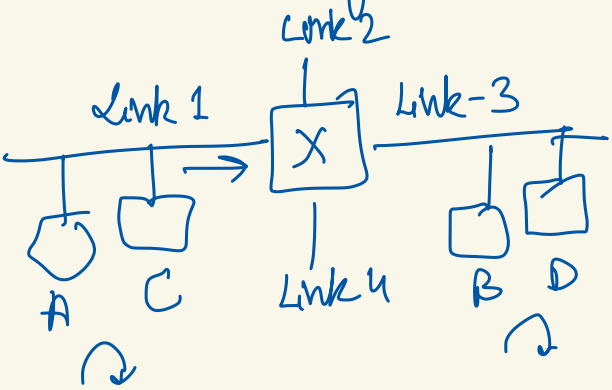
CSMA
- CD → Collision Detection
- CA → Collision Avoidance

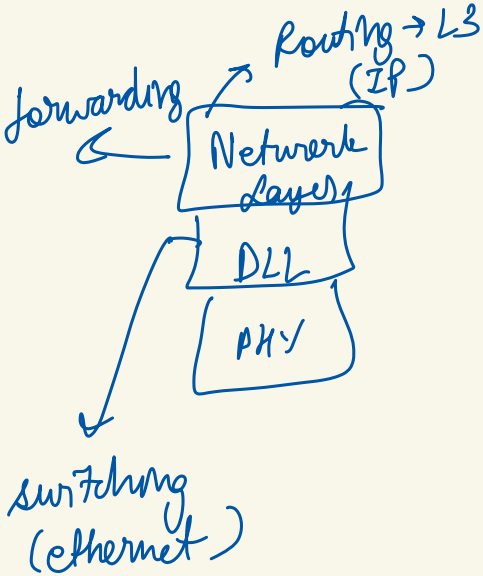
Carrier Sense multiple Access

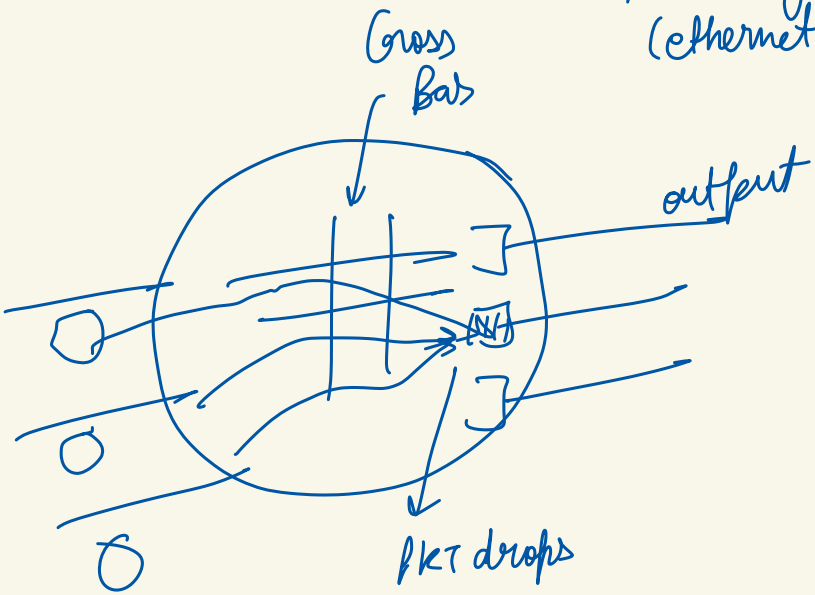
MAC → part of data link layer

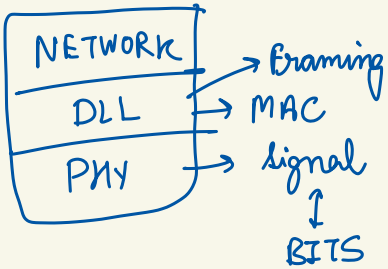
→ framing

Topology = graph of network







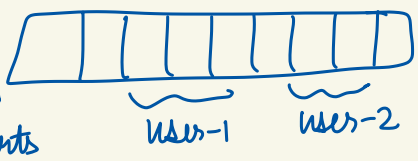


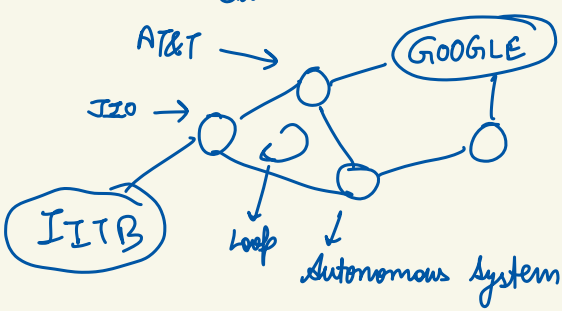
BIT STRING

Simplex : One-way communication

Full-duplex : simultaneous communication in both directions

Half-duplex : Communication in both directions but not simultaneous





BGP: Border Gateway Protocol

Message : L2 → frame (1 unit of data)

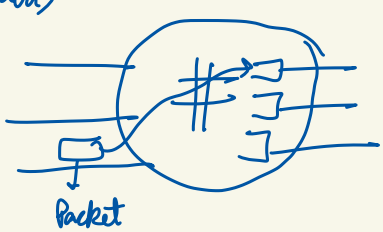
L1 → symbol

L3 → packet

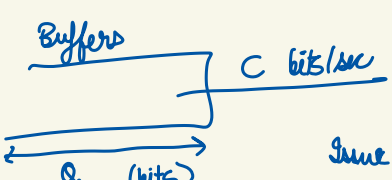
L4 → (TCP) (UDP)

↓ ↳ Datagram

Segment



Latency → Delay



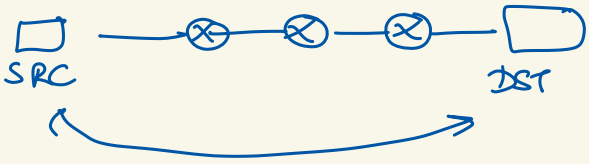
Issue : (1) cost

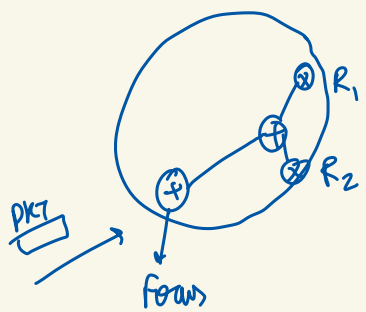
(i) → i^{th} router

$\frac{Q_{max}}{C}$

End-to-end delay of 1 pkt (SRC to DST) $= \sum_{i=1}^{N} \frac{Q_{max}^{(i)}}{C^{(i)}} + \sum_{i} d_i$

$\sum_{i} d_i$: speed of light delay + transmission delay





L4: Transport Layer

TCP: Transmission Control Protocol
- Congestion control → Video Call
- Reliability → File transfer

UDP:

Layer 5: Application Layer
- Web (HTTP)
- Email (SMTP)
- VoIP
- Text Messaging
- P2P

L4: TCP UDP

L3: IP (Internet Protocol)

L2: WIFI, 4G, Ethernet ...

L1:

Design Protocols in Modules

Each sub-problem handled by some protocol.

OSI-S

APPL

TRANS

Netw

DLL

Phy

Advantages of Layering / Modularity

(1) Ease of Development → only certain problems handled by particular layer

(2) Debugging

(3) Many applications and many physical layer technologies.

(4) Ease of modifying → only change 1 layer to address a problem

Protocol Layering :

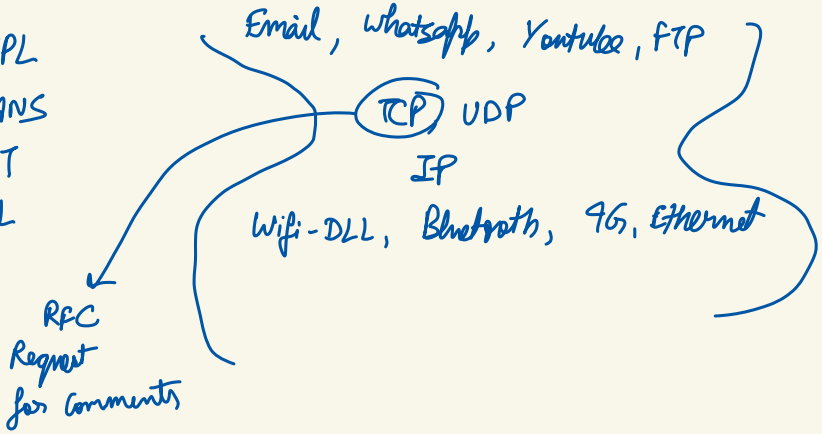
APPL

TRANS

NET

DLL

PHY



Advantages:

(i) Development & debugging (Ease)

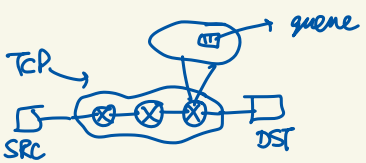
(ii) Easy to modify one layer without breaking entire system

(iii) Can have different choices for each layer (Compatibility)

Disadvantages:

(i) Opaqueness about other layers

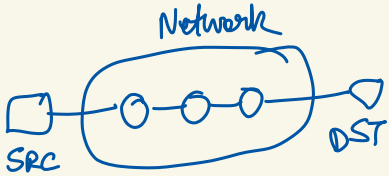
may not be a PKT-Drop (guess)



(2) Redundancy of Tasks

(3) Suboptimality

Ex:



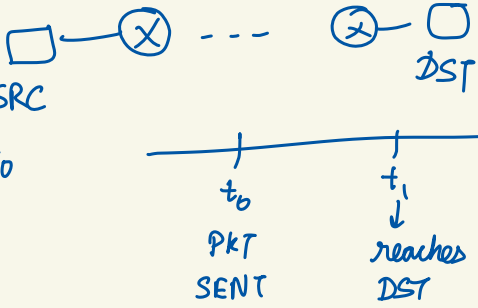
Best-effort: No guarantees on QoS

Quality of service: Amt of packet drops

Latency (delays)

Latency Delay:

(i) One-way delay



$OWD = t_1 - t_0$

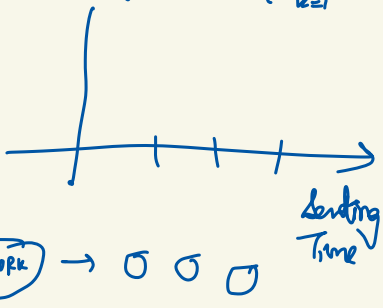
Jitter: Variability in owd latencies

PKTS : 1, 2, 3, ...

OWD $d_1, d_2, \ldots$

$e_k = |d_{k+1} - d_k|$

Average Jitter $= \frac{1}{n-1} \sum_{k=1}^{n} e_k$

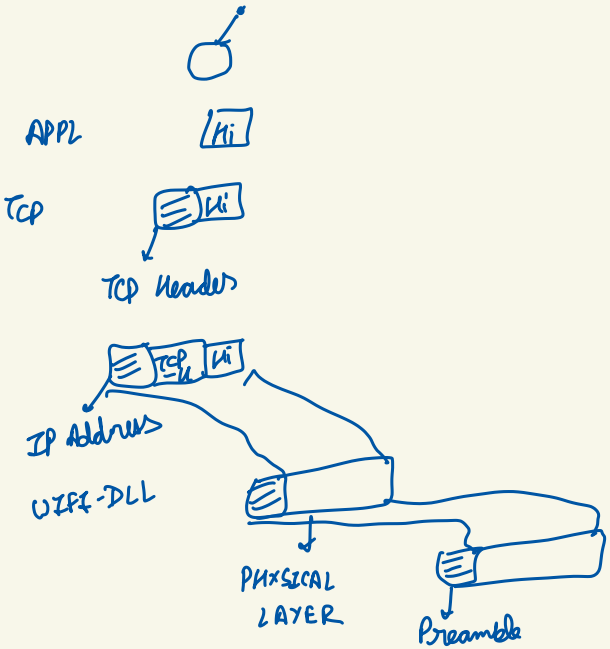


Telephone Network

First one (by Graham Bell) → voice

Low jitter
RTC, OWD

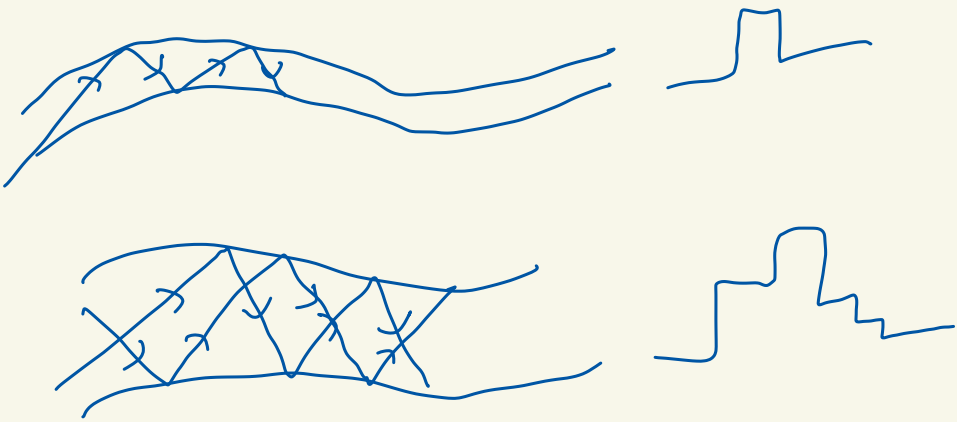
No data loss



PHY LAYER

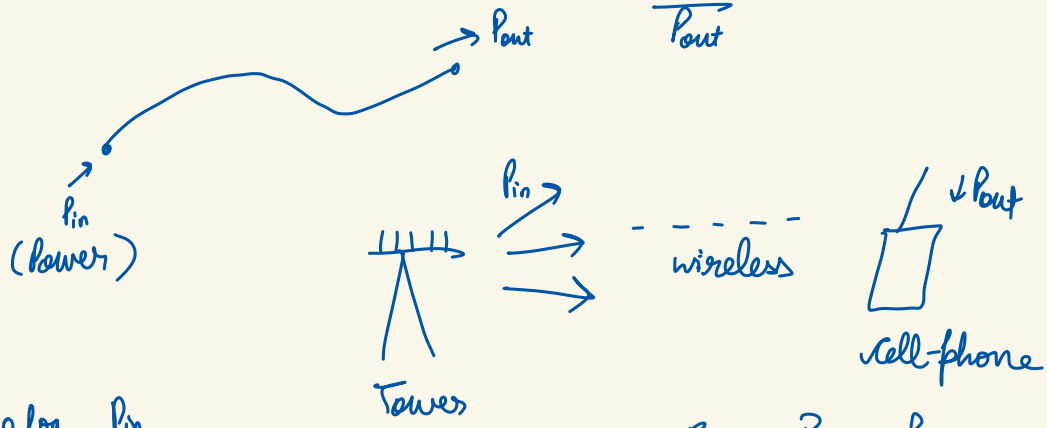
Optic fibre → submarine cables

Single-mode better than multi-mode



Attenuation

$\frac{P_{in}}{P_{out}}$



$10 \log_{10} \frac{P_{in}}{P_{out}}$

→ Attenuation in decibels (dB)

$P_{out} = \frac{P_{in}}{10}$ Atten = 10 dB

Ex: $P_{out} = \frac{P_{in}}{2}$

Atten $10 \log_{10} \frac{P_{in}}{P_{out}}$

$= 10 \log_{10} 2$

≈ 3 dB

→ Connecting together
Total atten = 6 dB

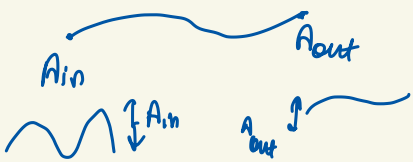
$10\log_{10}\frac{P_{in}}{P_1} = 3 \qquad 10\log_{10}\frac{P_1}{P_{out}} = 3$

$6 = 10\log_{10}\left(\frac{P_{in}}{P_1} \times \frac{P_1}{P_{out}}\right) = 10\log_{10}\frac{P_{in}}{P_{out}}$

$\frac{P_{in}}{P_{out}}$	dB
2	3
4	6
10	10
100	20
1000	30
10000	40

Power $\propto$ (Ampl.)^2

Attenuation



$= 10\log_{10}\left(\frac{A_{in}}{A_{out}}\right)^2 = 20\log_{10}\frac{A_{in}}{A_{out}}$

Absolute Power in decibel scale

1 mW as a reference

Power P (watts) in dBm → not unit less

$10\log_{10}\frac{P}{10^{-3}}$

Ex.: (1) P = 1 mW $= 10^{-3}$ W

$10\log_{10}\frac{10^{-3}}{10^{-3}} = 0$ dBm

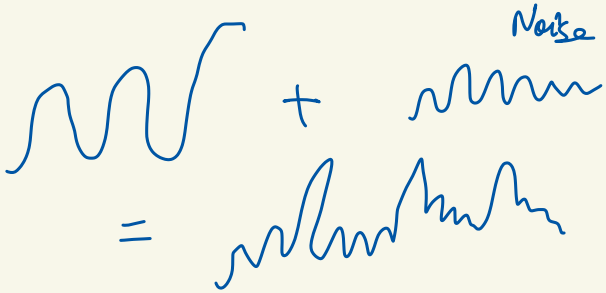
(2) P = 2 mW $\quad 10\log_{10} 2 = 3$ dBm

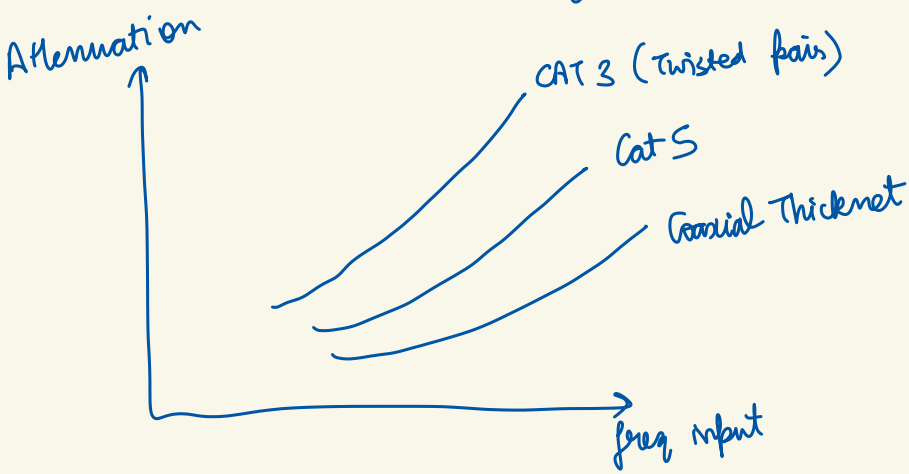
(3) P = 10 mW $\quad 10\log_{10} 10 = 10$ dBm

(4) $P = 1\mu W = 10^{-6}\ W$

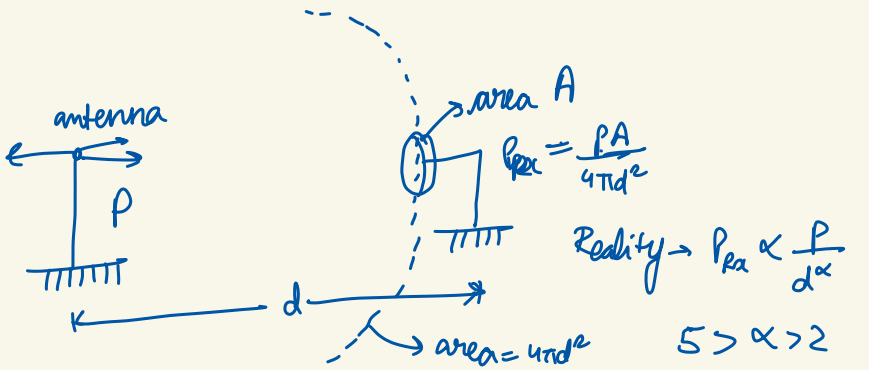
$$10\log_{10}\frac{10^{-6}}{10^{-3}} = -30\ dBm$$

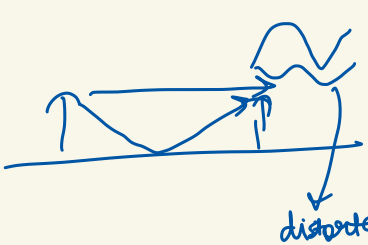
Received Power / Noise power

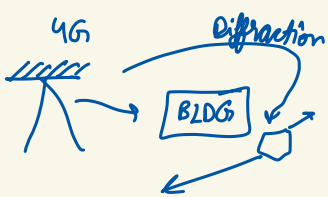


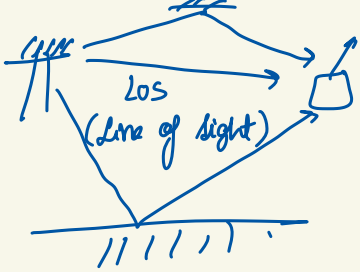


Wireless







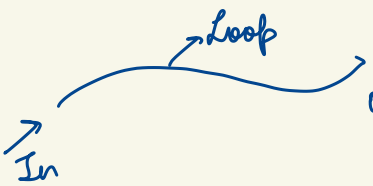


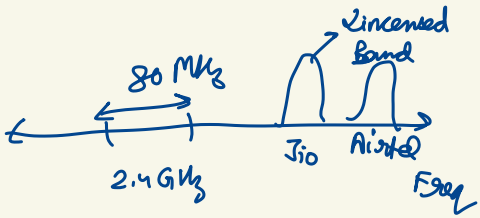
Shannon → Entropy

→ Capacity

Max data rate for communication over a channel

Wireless Channels

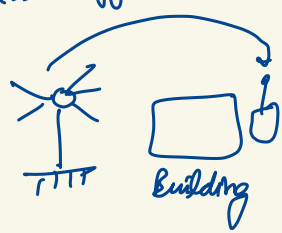




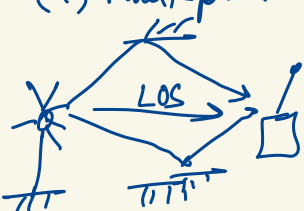
(1) Attenuation much higher than wired

(2) Interference

(3) Diffraction

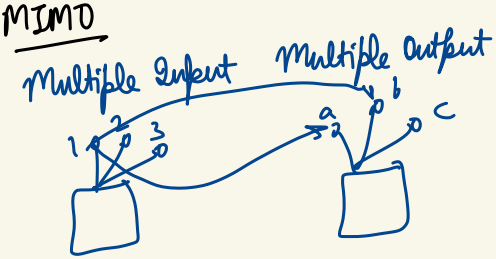


(4) Multi-path



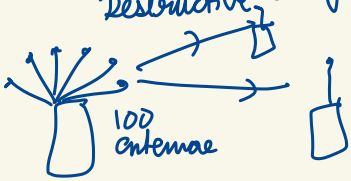
LOS = Line of sight

MIMO



Massive MIMO

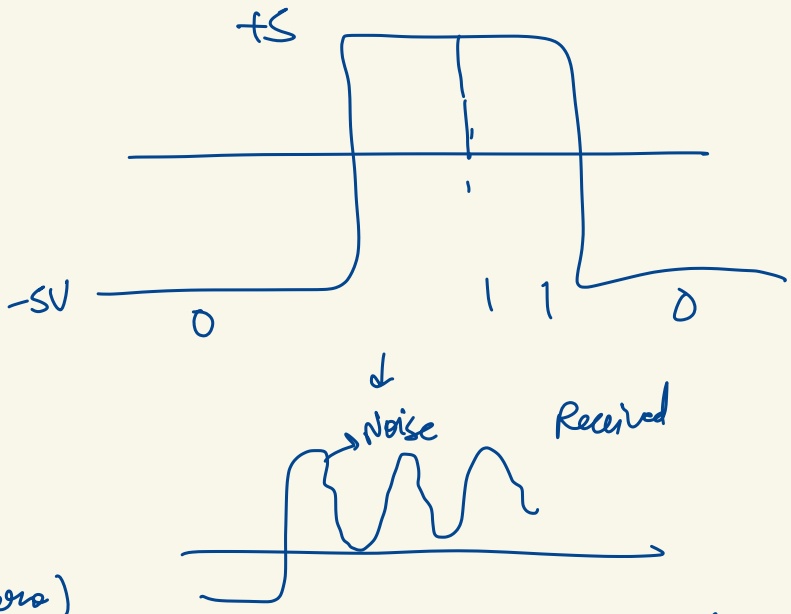
Constructive and Destructive Interference



Signalling

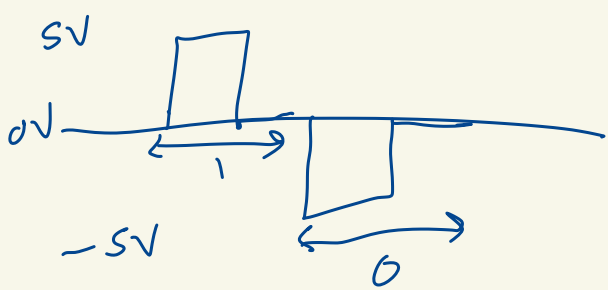
Wired

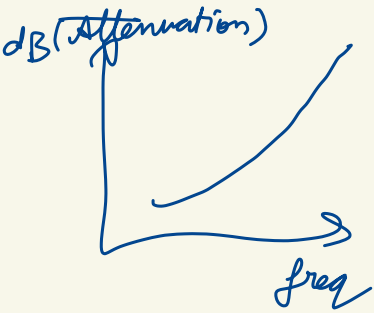
Non-return to zero (NRZ)



RZ (return to zero)

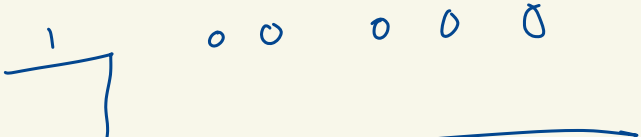
transmit the signal, then return to zero in each pulse





Issues with NRZ

(1) clocks not in sync (frequencies can be different) at Tx/Rx

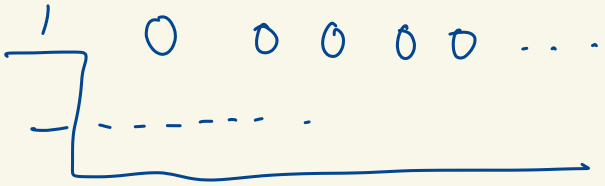


Rx faster clock, counts more '0' bits

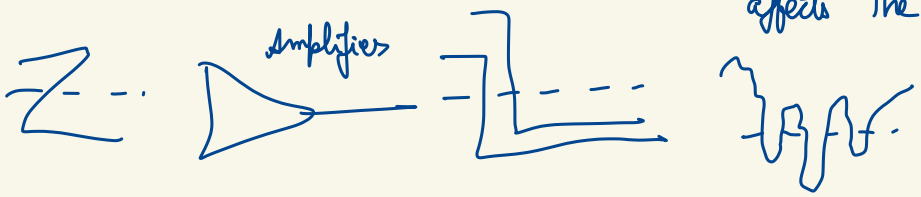
Preamble

10101...1011

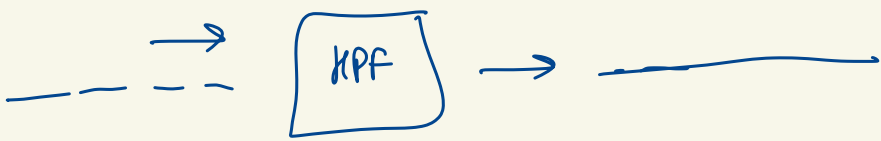
(2) Baseline wander

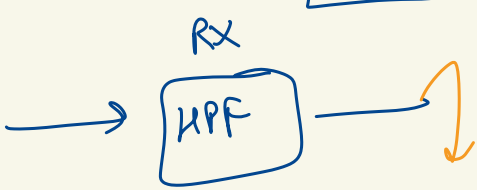


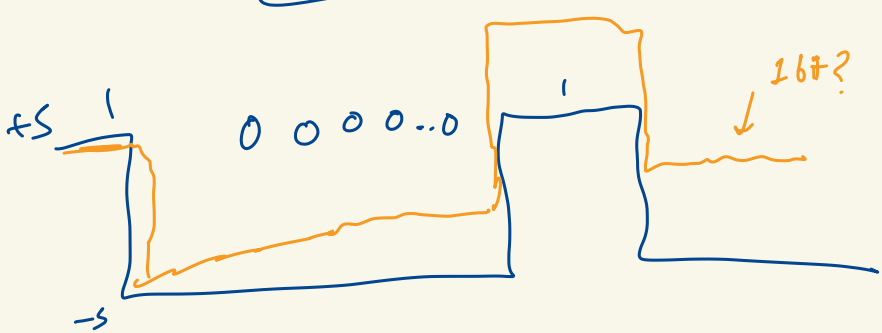
Average signal power at the receiver is taken as baseline.
If it is high (because of more 1s) then noise affects the further 1s



High-pass filter → Removes DC (offset) and low freq. signals



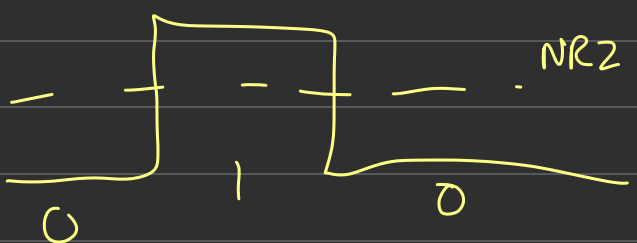




A high pass filter is an equalisation tool that removes all frequencies below a set point

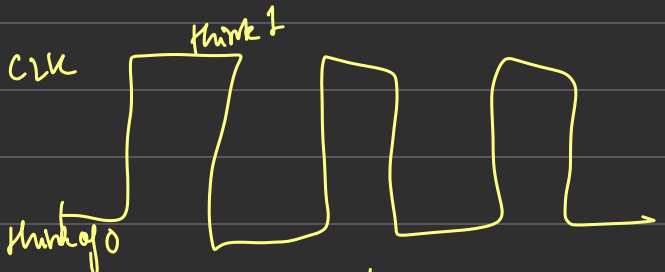
Band pass filter allows signals within a selected range of frequencies to be heard/ decoded.

Manchester Coding



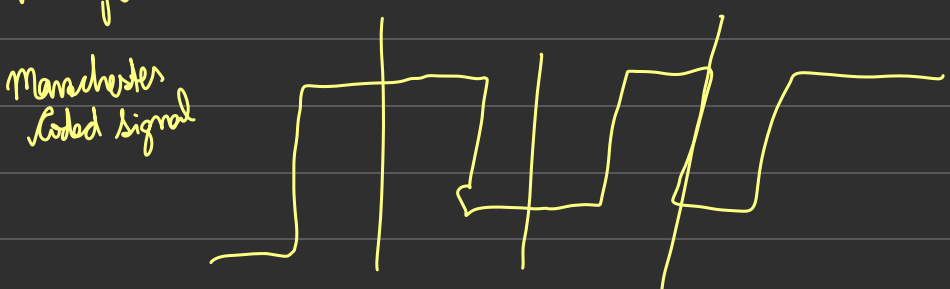
XOR

0 0	1



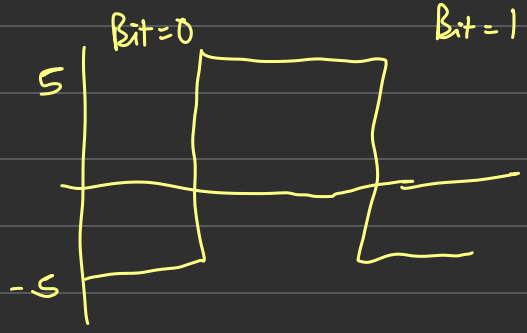
(a) signal transition every bit period

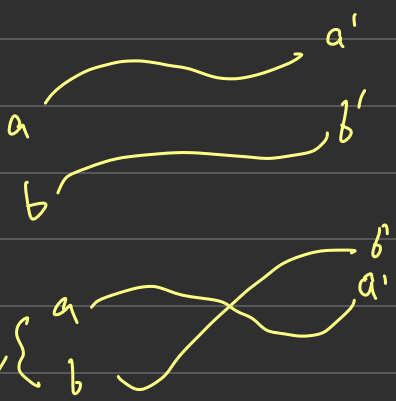
(b) Avg signal per bit period is 0



Line Coding → Wired media Bits → signals

Manchester Coding





If flipping happens, then receiver thinks:

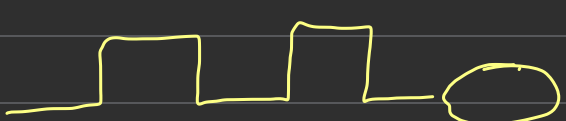
Need to know polarity

↓

Soln: Idea → Default first bit at 0.

Tx

Rx



↓

End of preamble can be used to convey about polarity

Ethernet

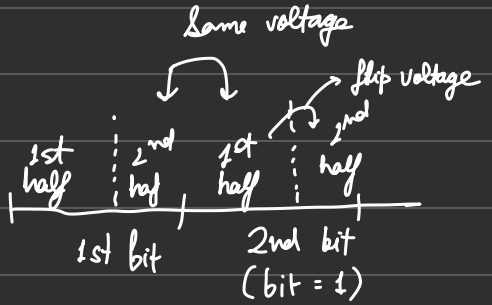
Consecutive 1s

10 10 10 ... 11

↔ Receiver might start receiving later

Differential Manchester Coding

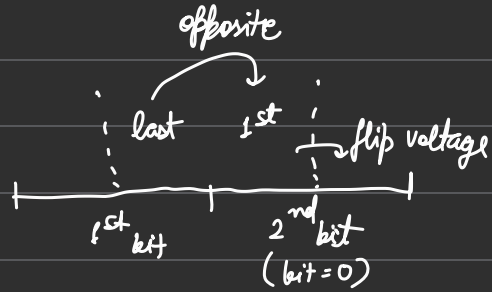
Rule: bit = 1 →

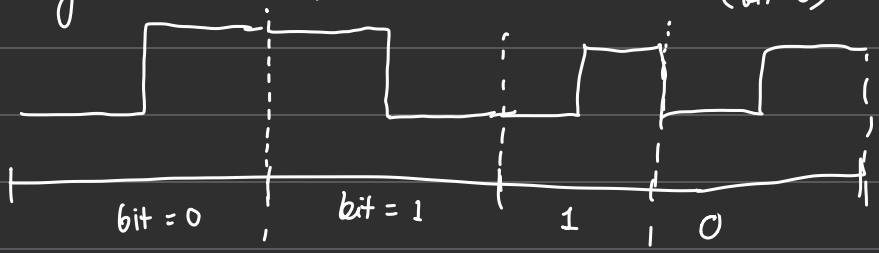


Voltage in 1st half of bit period is same as last half of prev. bit.

If bit = 0 →

Voltage in 1st half of bit is opp. to that of prev. bit period

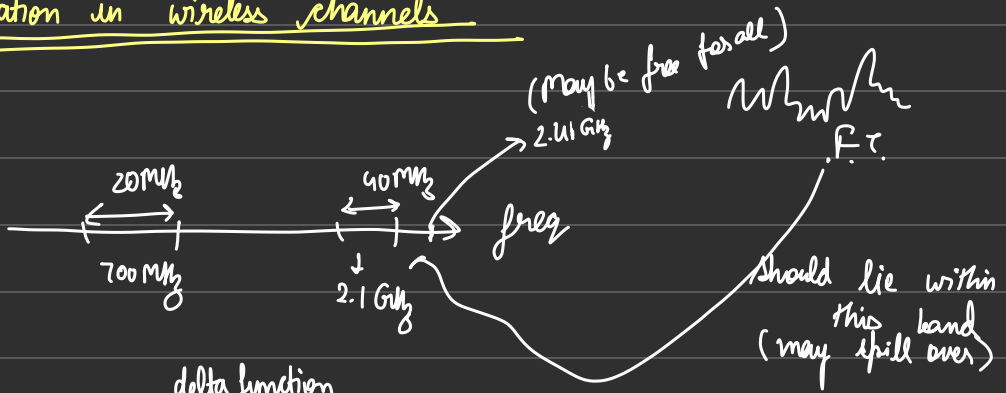


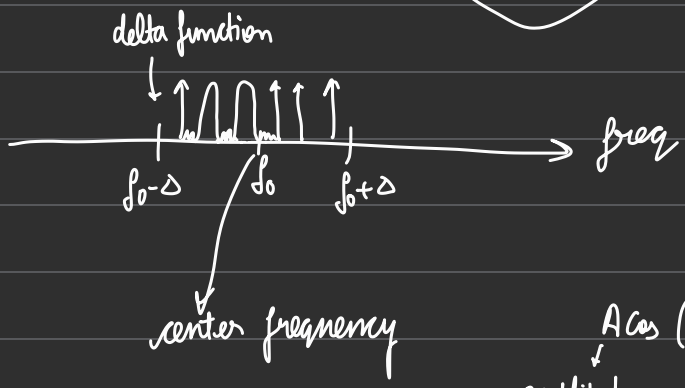


Even if this is flipped over, the message transmitted remains the same.

Default signal for the first bit.

Modulation in wireless channels

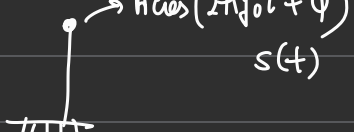




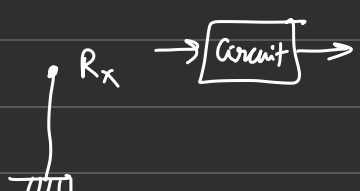
$A\cos(2\pi f_0 t + \phi)$

amplitude, freq, phase

Tx



Rx

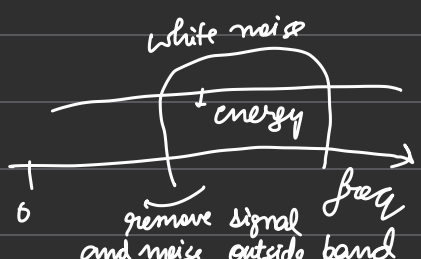


$\alpha\, s(t-\delta) + n(t)$

α: distortion (attenuation) ($\alpha < 1$)

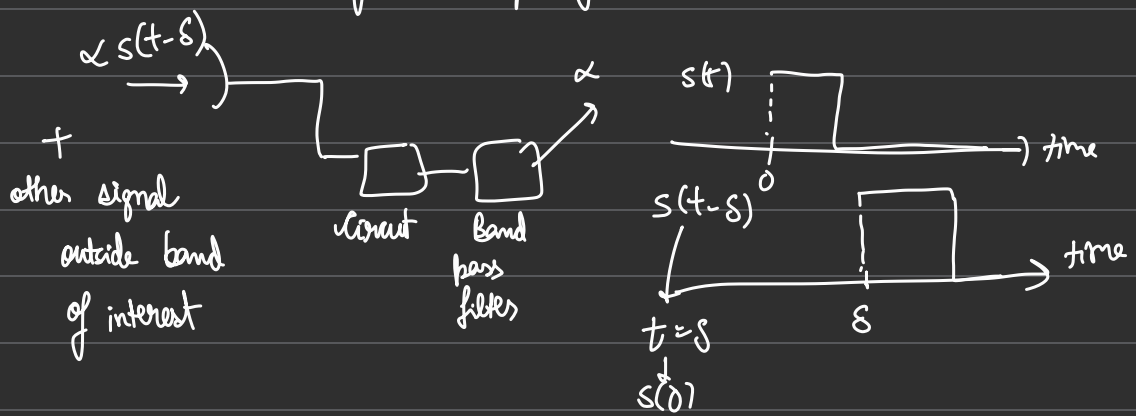
δ: delay

$n(t)$: noise → thermal noise (white noise)

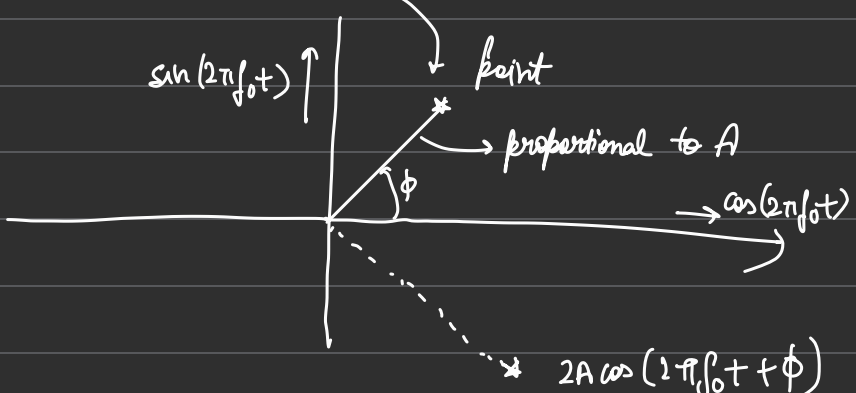


$\alpha s(t-\delta) + \underbrace{n(t)}$

noise after band pass filter



$A\cos(2\pi f_0 t - \phi)$ ⟶ vector space



In this vector space

$a(t) = \ldots \quad 0 \le t \le \tau$

$b(t) = \ldots$

Inner product

$\langle a, b \rangle = \int_0^\tau a(t) b(t)\, dt$

3d space

$\underline{a} = a_x \hat{e}_x + a_y \hat{e}_y + a_z \hat{e}_z$

$\underline{b} = b_x \hat{e}_x + b_y \hat{e}_y + b_z \hat{e}_z$

Dot product

$\langle \underline{a}, \underline{b} \rangle = a_x b_x + a_y b_y + a_z b_z$

↳ Information about the angle

Unit Vectors

$$\sqrt{\frac{2}{T}}\cos(2\pi f_0 t) \qquad T = \frac{1}{f_0}$$

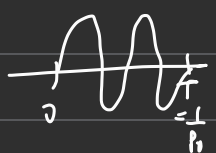
$$\sqrt{\frac{2}{T}}\sin(2\pi f_0 t)$$

Q: Inner product of these?

$$\langle s_1(t), s_2(t) \rangle = \frac{2}{T}\int_0^T \underbrace{\sin 2\pi f_0 t \cos 2\pi f_0 t}_{\frac{1}{2}\sin 4\pi f_0 t}\, dt$$

$$= 0$$

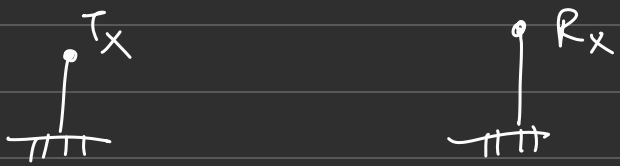
2 time periods in 0 to $\frac{1}{f_0}$

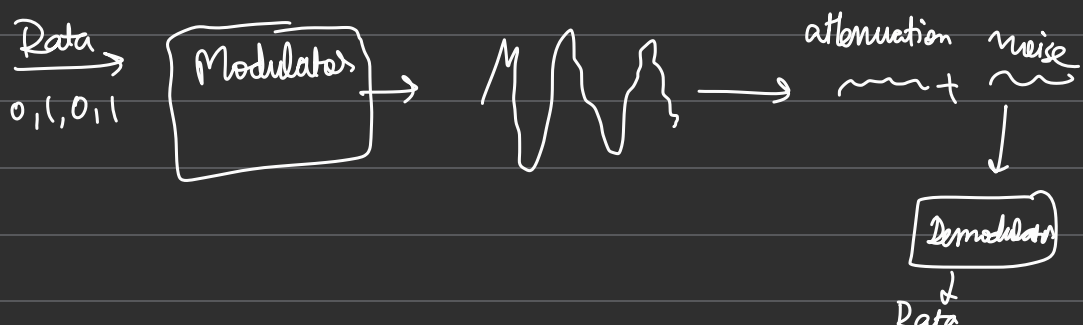


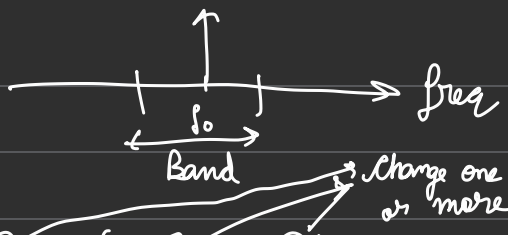
$$\langle s_1(t), s_1(t) \rangle = \frac{2}{T}\int_0^T \cos^2(2\pi f_0 t)\, dt$$

$$= \frac{1}{T}\int_0^T 1 + \underbrace{\cos(4\pi f_0 t)}_{\text{integration} = 0}\, dt$$

$$= 1$$

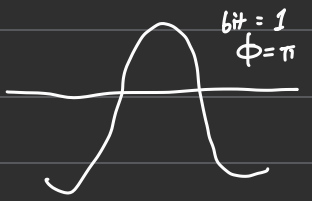


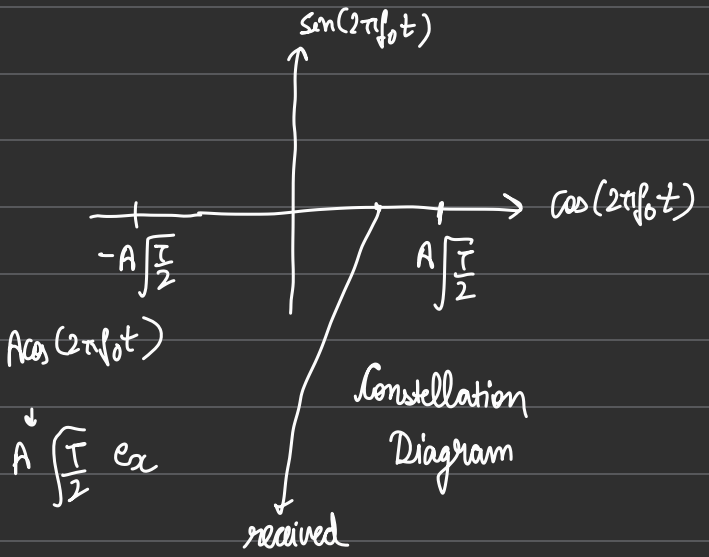




Modulate the carrier signal $\rightarrow \cos(2\pi f_0 t)$

Ex: $\phi = 0$ bit = 0

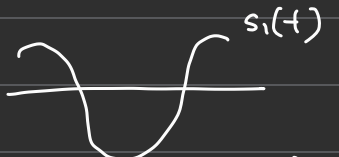




Unit vectors =

$\sqrt{\frac{2}{T}}\cos(2\pi f_0 t)$

$\sqrt{\frac{2}{T}}\sin(2\pi f_0 t)$



$r(t) = \alpha s_1(t-\delta) + n(t)$

Noise : White, Gaussian, Additive (AWGN)

$\alpha s_1(t-\delta) + n(t)$

noise power / energy

signal

$s_{1,x} = \langle s_1(t), e_x \rangle$ $\quad s_{1,y} = \langle s_1(t), e_y \rangle$

$$\langle a(t), b(t) \rangle = \int_0^T a(t)\, b(t)\, dt$$

$$s_1(t) = A\cos(2\pi f_0 t)$$

$$s_{1,x} = \langle s_1(t), e_x \rangle = \int_0^T A\cos(2\pi f_0 t) \left(\sqrt{\frac{2}{T}}\cos(2\pi f_0 t)\right) dt$$

$$= A\sqrt{\frac{T}{2}}$$

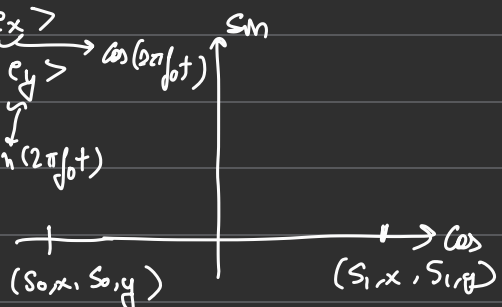
$$s_{1,y} = \langle s_1(t), e_y \rangle = \int_0^T \sqrt{\frac{2}{T}} \sin 2\pi f_0 t \, dt \quad (A\cos 2\pi f_0 t)$$

$\sin(4\pi f_0 t)$; integral over T is 0

$r(t) \rightarrow r_x, r_y$?

$r_x = \langle r(t), e_x \rangle$

$r_y = \langle r(t), e_y \rangle$



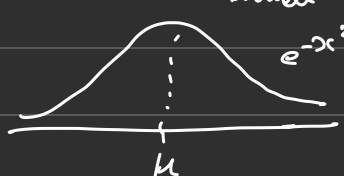
Suppose $r(t) = \alpha s_1(t) + n(t)$

$n_x = \langle n(t), e_x \rangle$

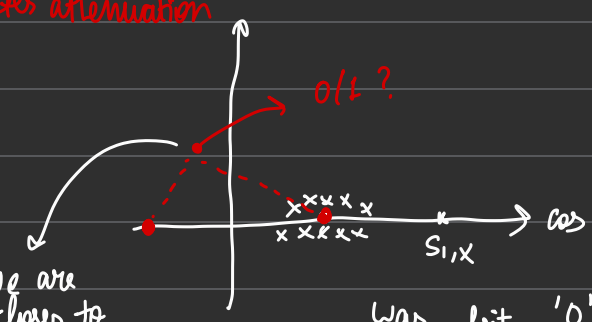
$n_y = \langle n(t), e_y \rangle$

Each is independent gaussian random variable

identically distributed,



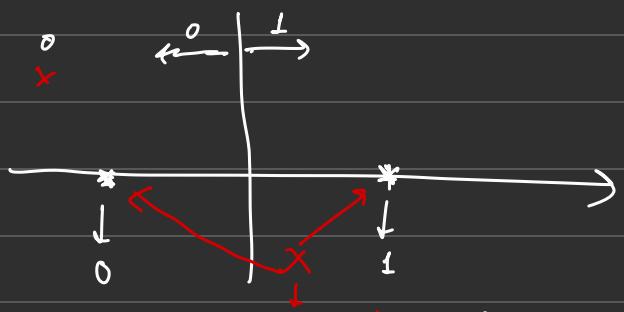
• = After attenuation



We are closer to which point?

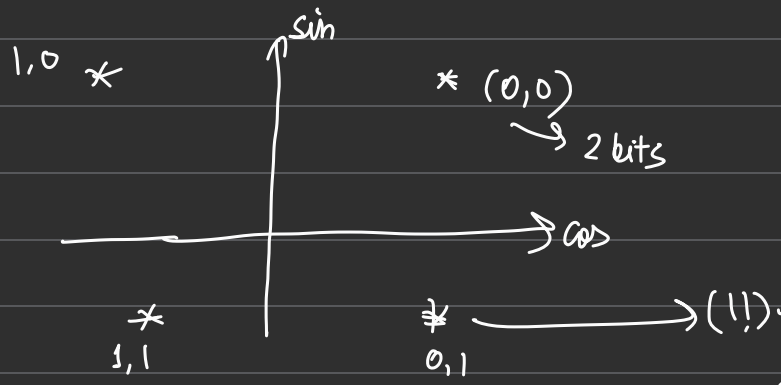
Was bit '0' sent or '1' ?

Suppose we know constellation after attenuation and without noise



Anything on left side → 0
right side → 1

QPSK Quadrature PSK

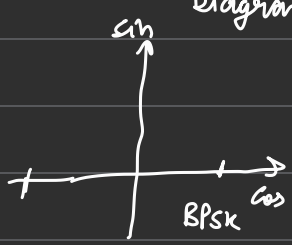


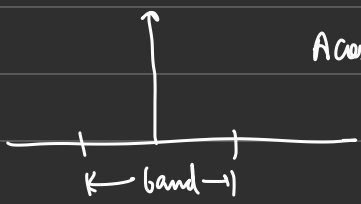
PHY
↓
DLL

$s(t)$ → $r(t)$
T_x

$r(t)$ = signal + noise

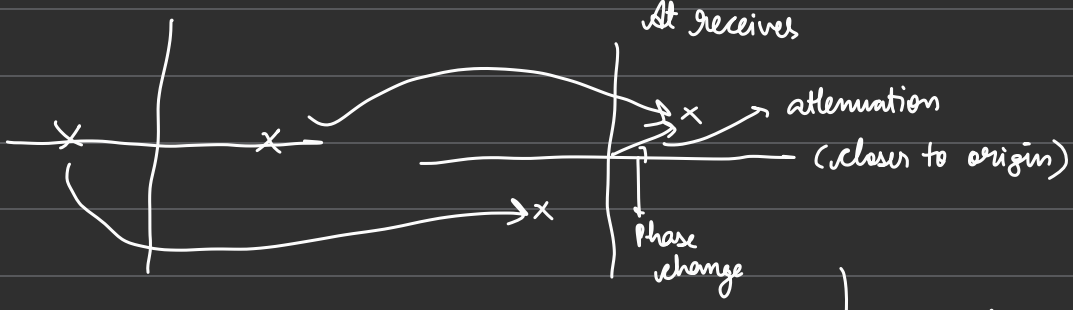
Constellation Diagram





$A\cos(2\pi f_0 t + \theta)$

signal → channel → R_x
T_x

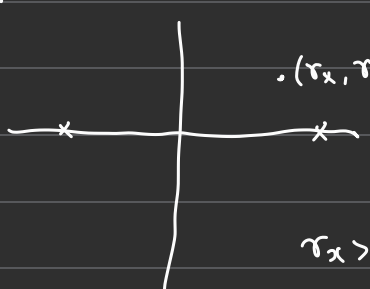


correct for phase change,
(rotate back by ϕ)

$r(t)$

$r_x = \langle r(t), e_x(t) \rangle$

$r_y = \langle r(t), e_y(t) \rangle$

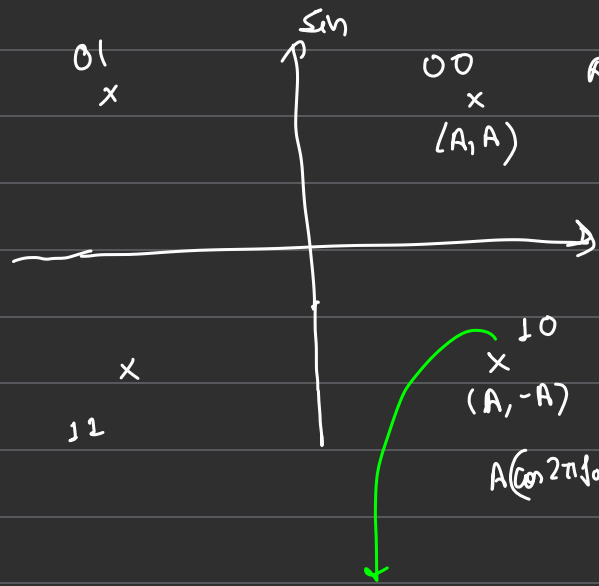


AWGN → Additive white Gaussian noise

BPSK → Binary Phase Shift Keying

$r_x > 0$ $r_{x1} \leq 0$

QPSK

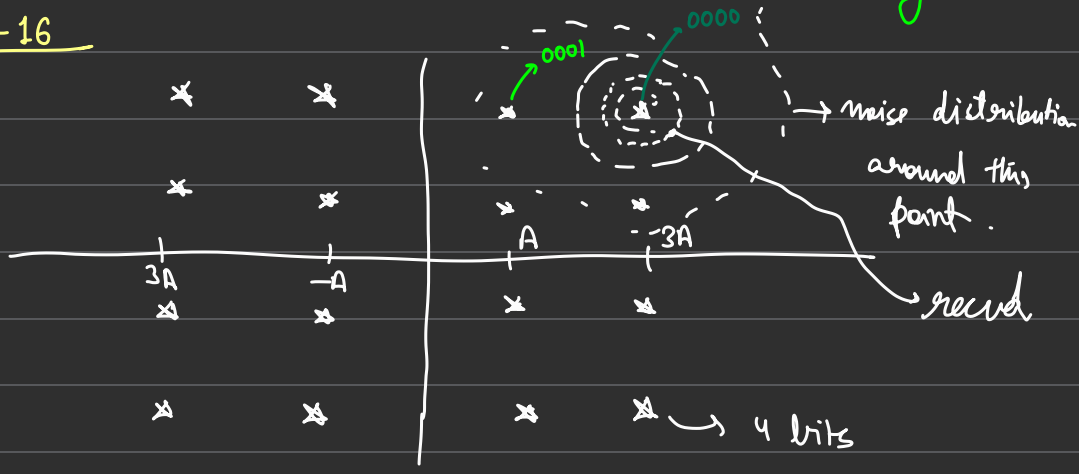


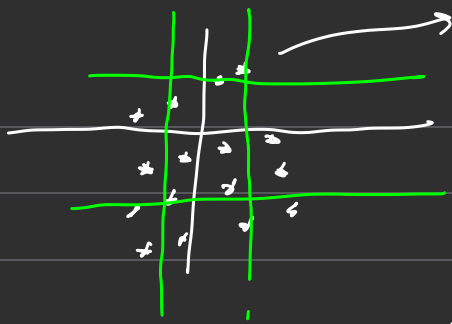
$A(\cos(2\pi f_0 t) + \sin(2\pi f_0 t))$. const

$A(\cos 2\pi f_0 t - \sin 2\pi f_0 t)$. const

11? Not a good idea because 00 & 11 are near hence, we might have 2 bit error instead of one

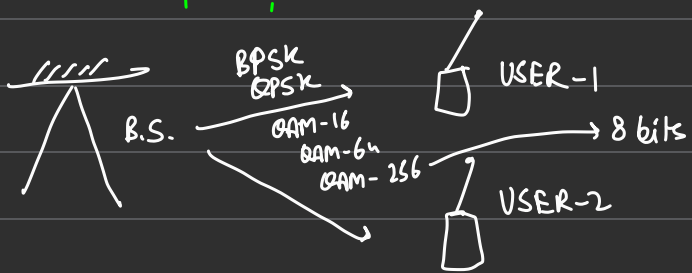
QAM-16



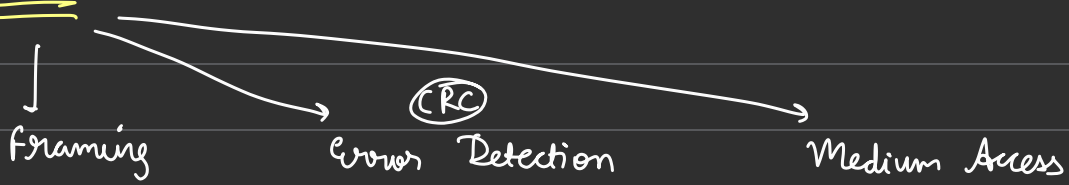


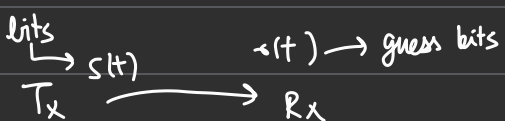
Given P_e (bit errors) $< \varepsilon$

Attenuation?



DLL





data 1011 01011 011010

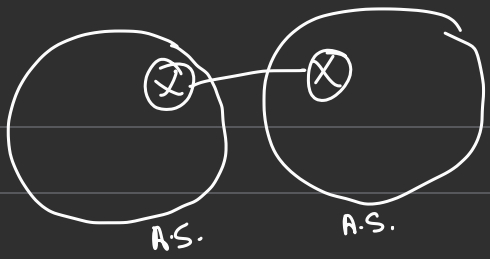
IP addr?

Frame (1 chunk of meaningful data)

HDLC

High level Data Link Control

WAN → Wide Area Network

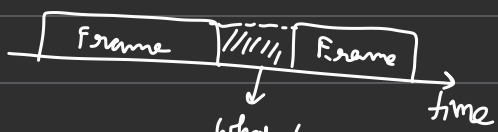


Synchronous Mode (HDLC)

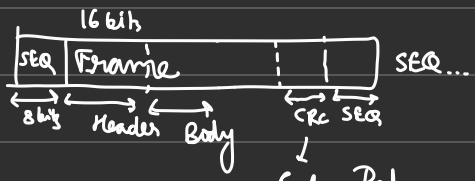
Always → transmit

HDLC → Wired (NRZ, Manchester...)

Default (Seq): 01111110



Seq, Seq, SEQ, ...

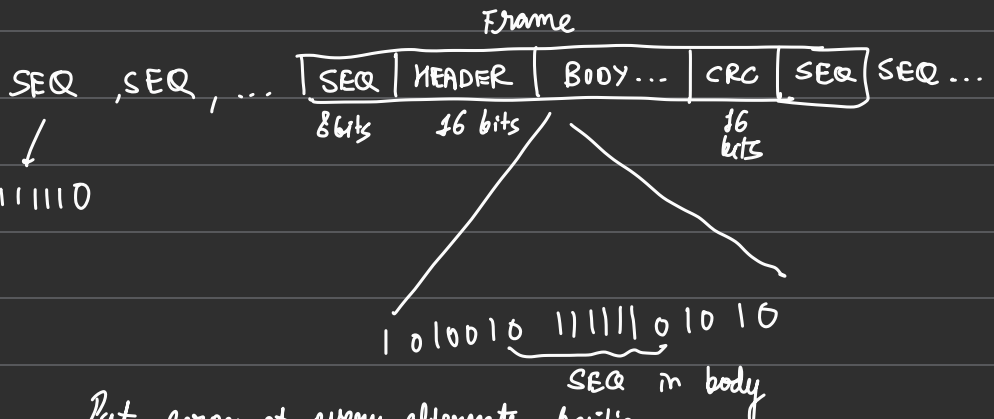


Bit stuffing

Cyclic Red. Check

DLL

HDLC



Sol^n – Put zeroes at every alternate position → lot of redundant bits.

Sol^n: Every time after 5 ones → add a 0.

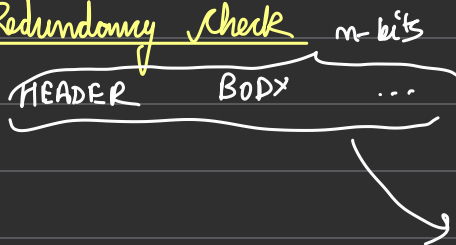
At transmitter: 00 11111 0 1010 11111 0 1010 1111 1111 010

0 0 0

At receiver:

If seq 0 11111 | 0 ⇒ Assume bit stuffing and remove
10 → SEQ at end of frame
11 → Error, so discard frame

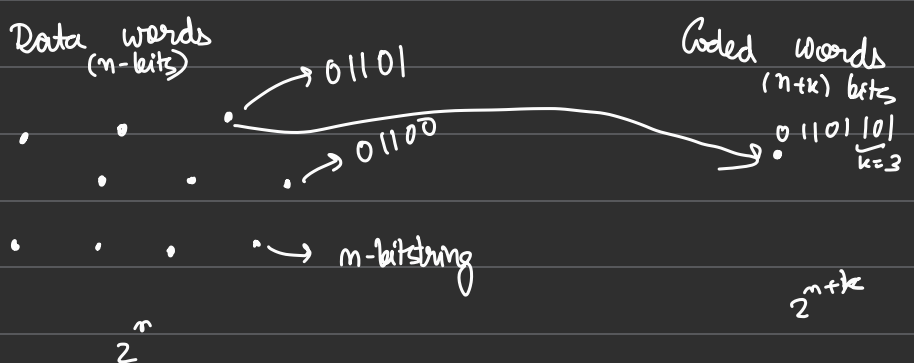
Cyclic Redundancy Check



(1) want to be able to easily detect a large type of bit errors (single bit, burst of consec. errors etc)

(2) Creation of CRC and verification of CRC should be computationally efficient

(3) For given 'k', CRC should be computable for any 'n'



Hamming Distance: Given two codewords $a_1 a_2 \dots a_n$ and $b_1 b_2 \dots b_n$

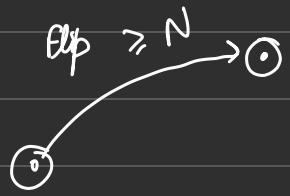
bits in which they differ

Min. Hamming Distance of a code.

pairs of codewords

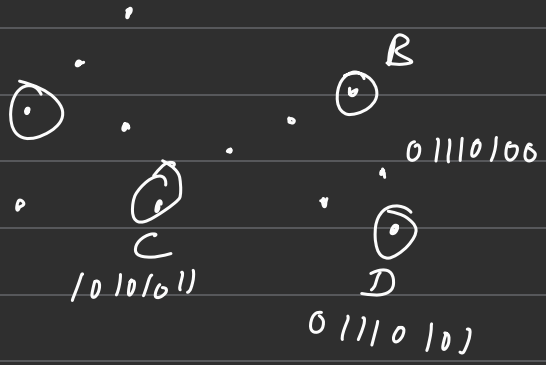
Error Detection: If min. HD is N (of a coding scheme) then we can always detect (N-1) or less bit errors

* We detect that there were errors but may not know position of bit errors.



Error Correction

Suppose min. HD is (2t+1) and # bit errors is 't' or less. then by mapping recd code word to the nearest valid code word corrects all errors.

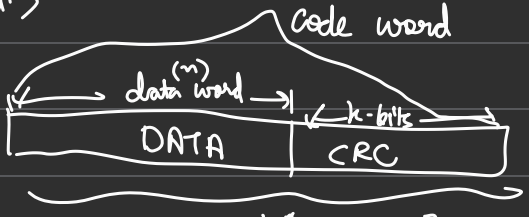


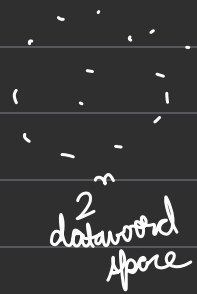
Galois Fields

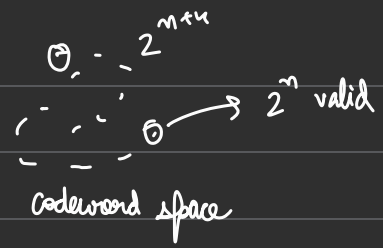
1101

Divisor (k+1) bits

CRC







Galois Fields

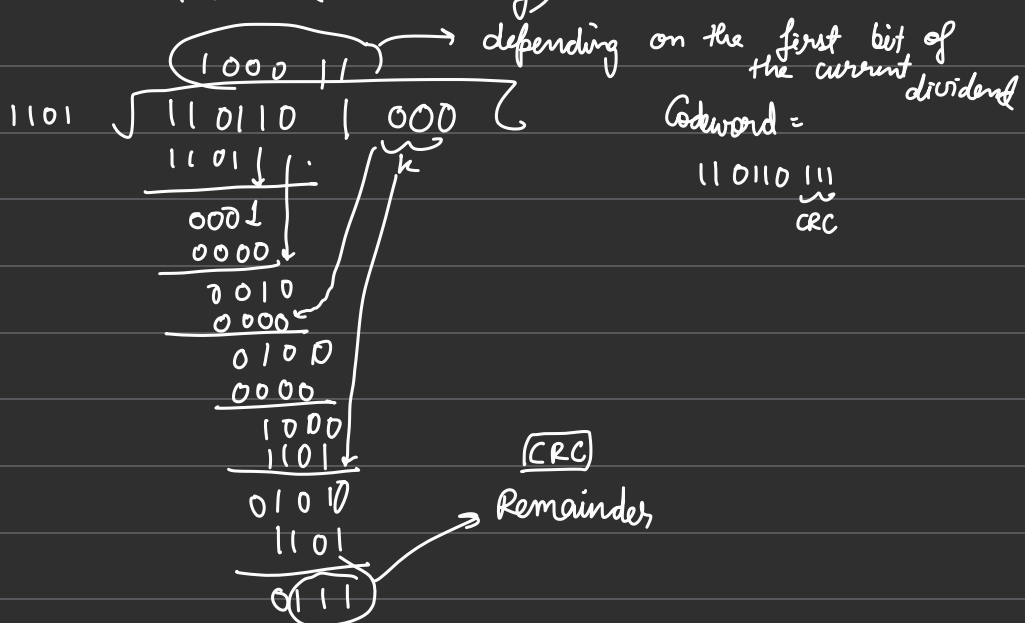
X	0	1
0	0	0
1	0	1

XOR

+	0	1
0	0	1
1	1	0

Data: 110110 K=3

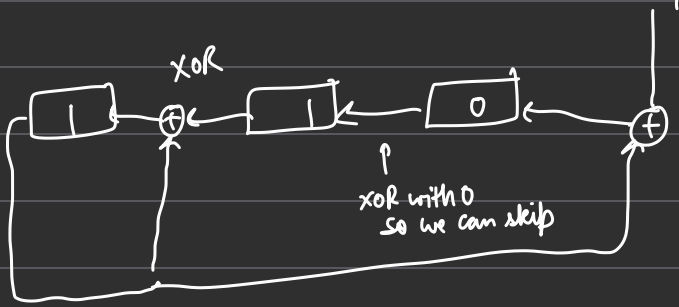
Divisor / Generator: 1101 (K+1 bits long)



Initializing with all zeros

1101

Data
110110 | 000



At receiver:

codeword
Data' | CRC'
Concatenation

Polynomial Representation of Bitstrings

1101 → $1.x^3 + 1.x^2 + 0.x + 1.x^0$

$x^3 + x^2 + 1$ → C(x)

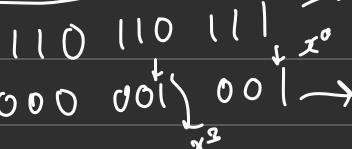
(Divisor / Gen. Poly)

$C(x)(1+x) = (x^3+x^2+1)(x+1)$

$= x^4 + x^2 + x + 1$

```
1101
 11
-----
1101
1101
-----
10111
```

Errors?



110 110 111 → P(x)

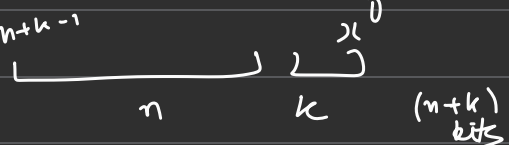
000 001 001 → E(x) Error bitstring

Q) $\frac{P(x) + E(x)}{C(x)} = 0$?

If not zero then detected errors

Types of Errors

(1) Single bit errors



$E(x) = x^i$; for some $i \in \{0, 1, \ldots, n-k+1\}$

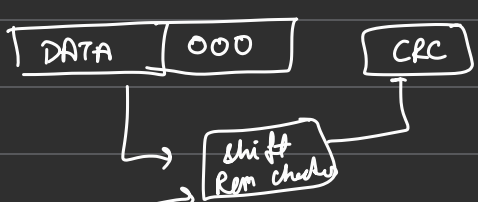
$$\frac{P(x) + E(x)}{C(x)} = \frac{P(x)}{C(x)} + \frac{E(x)}{C(x)}$$

$\frac{P(x)}{C(x)} = 0$

If $C(x) = x^h + \underbrace{\ldots}_{\text{anything}} + 1$

$C(x) = [x^m + \ldots + x^q]$

x^{n+m} ← highest, x^q ← lowest



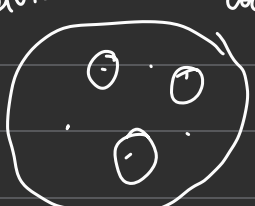
Min HD = d

⇓

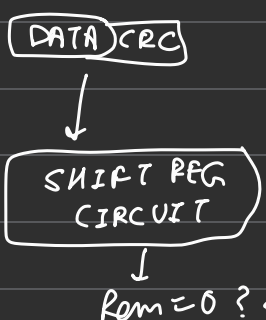
fewer than d errors always detected

* May detect error of ≥ d bits

→ No guarantee



TRANSMIT



Polynomial Arithmetic

$1101 \rightarrow 1.x^3 + 1.x^2 + 0.x + 1 = x^3 + x^2 + 1$

GF(2) : Addition / Subtr (XOR)

Data: $110110 \rightarrow x^5 + x^4 + x^2 + x$

$1101 \overline{)\,110110\,000}$ → $x^3 \; \dfrac{(x^5 + x^4 + x^2 + x)}{x^3 + x^2 + 1}$

Remainder $= x^2 + x + 1 = (111)$

If A(x) is divisibly B(x)

$A(x) = B(x) \cdot D(x)$

we will send $110110 \mid 111 \rightarrow x^3 (x^5 + x^4 + x^2 + x) + x^2 + x + 1$

P(x) = Codeword

C(x) → divisor (generator)

E(x) → error polygon

P(x) + E(x) is received

$E(x) = x^6 + x^3$

$110 \; 110 \; 111$

↓ ↓

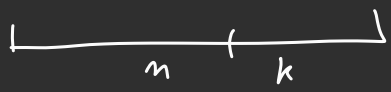
error bits

$$\frac{P(x) + E(x)}{C(x)} = \frac{P(x)}{C(x)} + \frac{E(x)}{C(x)}$$

P(x) chosen so that C(x) divides it

Q) Does C(x) divide E(x) ?

Types of errors



(1) Single bit errors

$E(x) = x^i$ for some $i \in [0, 1, \ldots, n+k-1]$

Suppose $C(x) = \underbrace{x^k + \ldots + 1}_{\text{anything}}$

then $C(x)$ does not divide $E(x)$

$$\frac{x^i}{x^k + \ldots + 1} \quad (k>0)$$

$C(x)\,P(x) \stackrel{?}{=} x^i$

$C(x)(x^m + \ldots + x^q)$ $\quad m > q$

$\downarrow$

$x^{k+m} + \ldots + x^q$

$k+m > q$

(2) Two bit errors

$E(x) = x^j + x^i \quad (j>i)$

$= x^i (x^{j-i} + 1)$

Suppose $C(x)$ is of form $x^k + \ldots + 1$

$$\frac{E(x)}{C(x)} = \frac{x^i (x^{j-i} + 1)}{x^k + \ldots + 1}$$

Know x^i terms not cancelled out

Q: Does $C(x)$ divide $x^{j-i} + 1$

Def: (order of a Polynomial):

The smallest r such that $C(x)$ divides $x^r + 1$ is called its order.

Known $C(x) = x^k + \ldots + 1$ s.t. order is $2^k - 1$

Ex. $k=16$; $C(x) = x^{16} + \dots + 1$ can find $C(x)$ s.t.

it will not divide

any x^p+1 for $p < 2^{16}-1$

(3) Odd number of errors

$$E(x) = \underbrace{x^i + x^i + \dots}_{\text{odd number of terms}}$$

If $C(x) = (1+x)(\cdot)(\dots)$ then all odd errors detected

If $C(x)$ has even # terms

$$CRC = \underbrace{x^{16} + x^{15}}_{x^{15}(x+1)} + \underbrace{x^2+1}_{(x+1)(x+1)}$$

$$= x^{15}(x^2+x+1)$$

want to show $E(x) = C(x)D(x)$ ↓ not possible

$$x^i + x^i + \dots +$$

odd # terms

$$F(1) = 1 + \dots - 1 \quad 1$$

Case (1) $C(x) = (1+x)(\dots)$

$C(1) = \underbrace{(1+1)}_{0} (\dots)$

RHS is 0

Case (1) $C(x)$ has even # terms

$C(1) = \underbrace{1+\dots+1}_{\text{even}}$

$= 0$

(4) Burst of errors :

Data / CRC : 101 .. 1101 ... 11

↳ interference → consecutive errors

$$E(x) = x^{i+l-1} + x^{i+l-2} + \dots + x^i = x^i \left[x^{l-1} + x^{l-2} + \dots + 1 \right]$$

$$\frac{E(x)}{C(x)} = \frac{x^i \left[x^{l-1} + x^{l-2} + \dots + 1 \right]}{x^k + \dots + 1}$$

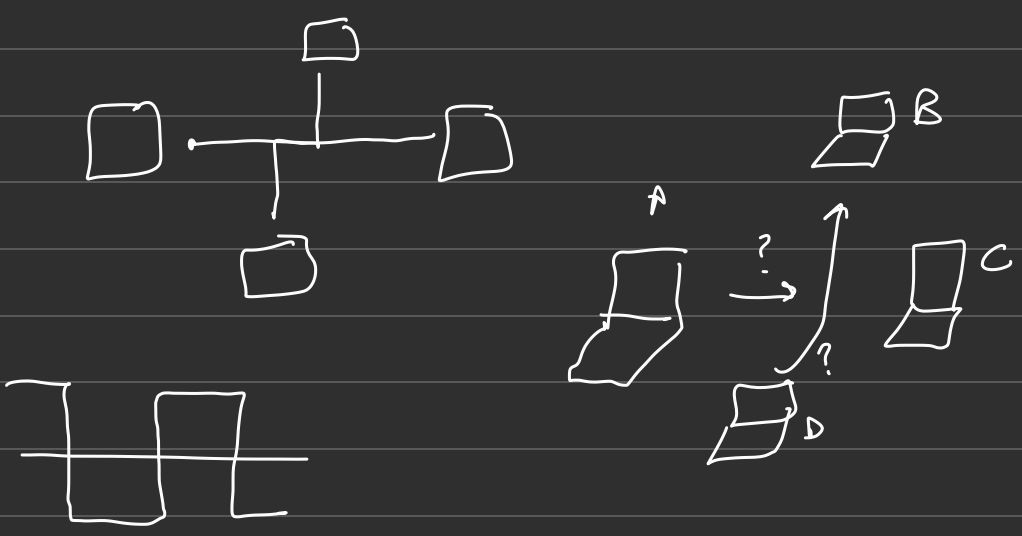
x^i does not cancel any factor in the denominator

If $l-1 < k$ then all factors of $C(x)$ cannot be canceled.

So $C(x)$ of form $x^k + \dots + 1$ detects all bits of error of length 'l' such that $l-1 < k$

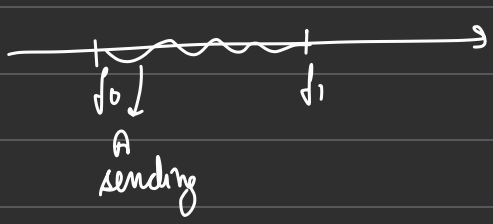
Ethernet: CRC-32: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

Medium Access



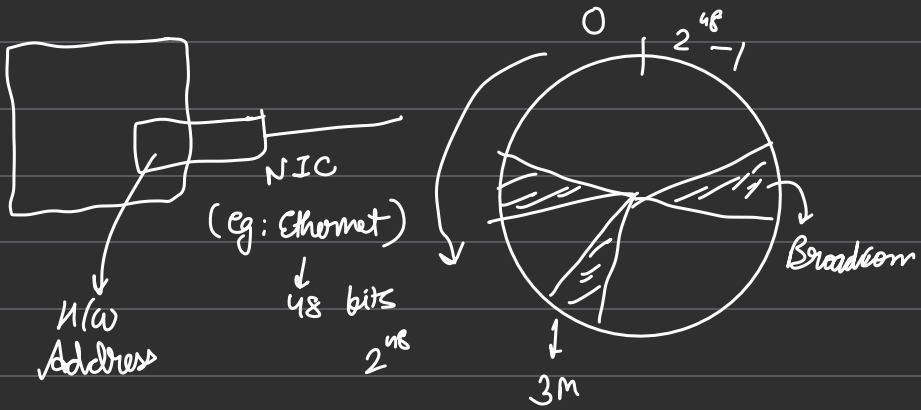
At C

Q) what is being sent, by whom, for whom?
↳ Identifiers

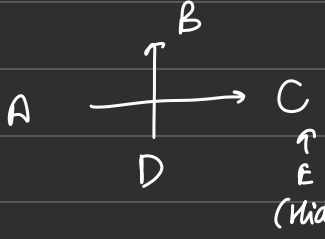


Drawbacks: (1) what if E, F want to join
(2) Throughput of A is $\frac{1}{4}$ compared to if A uses the full band.

MAC Layer (DLL) → Use hardware address



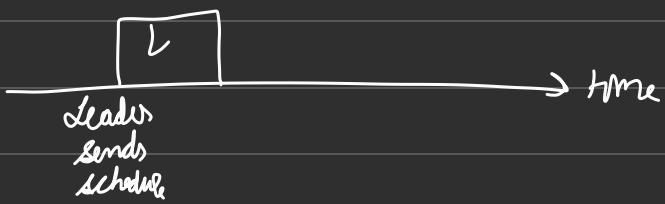
Interference?



Diff. Freq Bands?
Who uses what bands?

(Hidden terminals A and E are hidden)

Idea: One node is leader

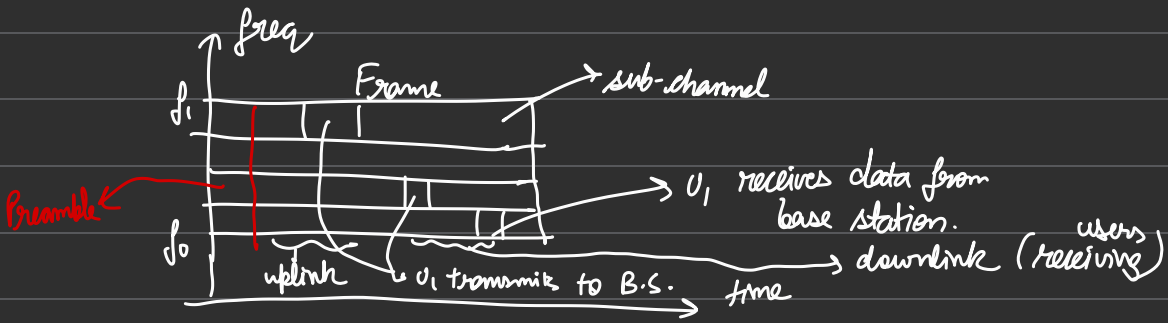


Token based

Token release message → say who giving to

Medium Access

Options: (1) Central coordinator
(2) Decentralised



Ex. BS → U_1 (less attenuation) } f_1 reverse for f_2
BS → U_2 (more attenuation)

Same column divided into different slots

Issues with central coordinator

→ single pt. of failure
→ may not be possible in certain situations → wireless — unlicensed bands

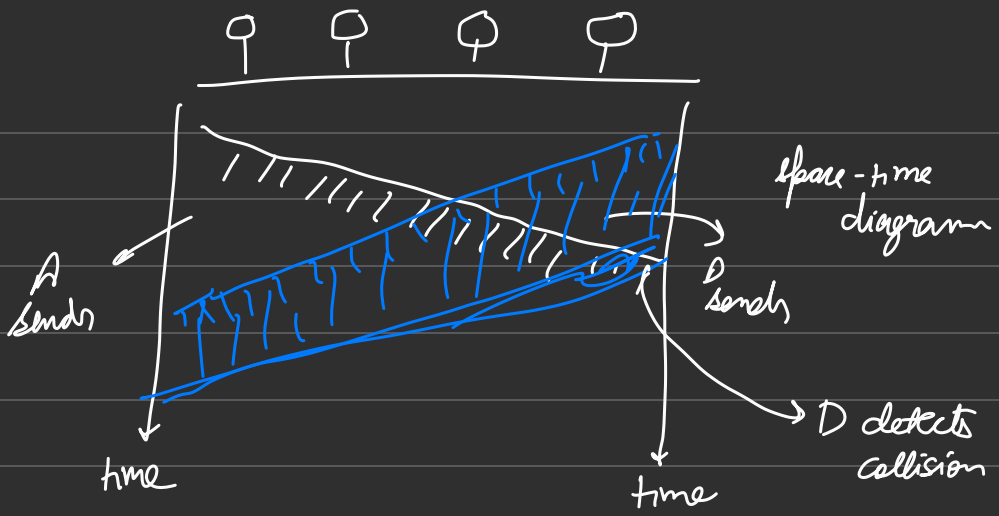
No central coordinator

Want
(1) Plug a play
- Connect and disconnect
(2) No central coordinator

Idea: Suppose most of time 1 node transmits, only rarely we have collisions

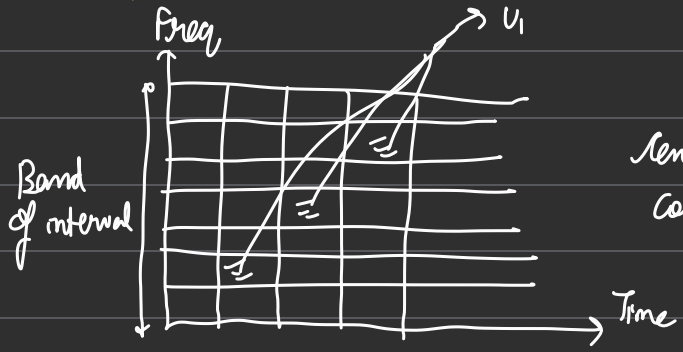
Collisions: multiple nodes transmit simultaneously
→ signals add up at receiver.
↳ neither can be deciphered.

If collision occurs → stop transmitting.



Ethernet allows large frame size so that (A) senders can detect collisions.

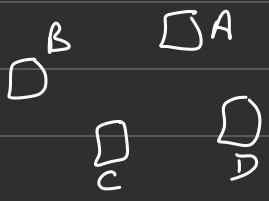
Combination (TDMA + FDMA)



4G → OFDMA → orthogonal

Centralised coordinator

Token Passing :

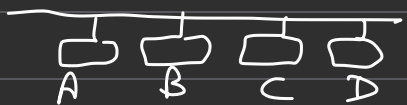


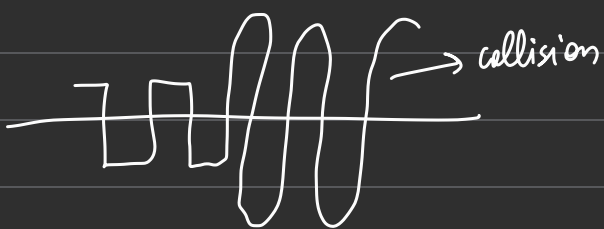
- Only token holder can transmit
- Pass on token every so often

Ethernet : CSMA - CD

- Carrier Sense
- multiple access
- Collision Detection

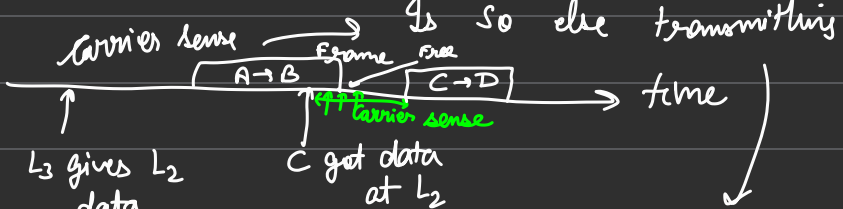
CD: Energy level



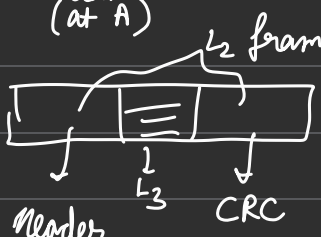


IEEE 802.3 → ethernet

Wifi : 802.11



If yes : Remain silent
else : transmit



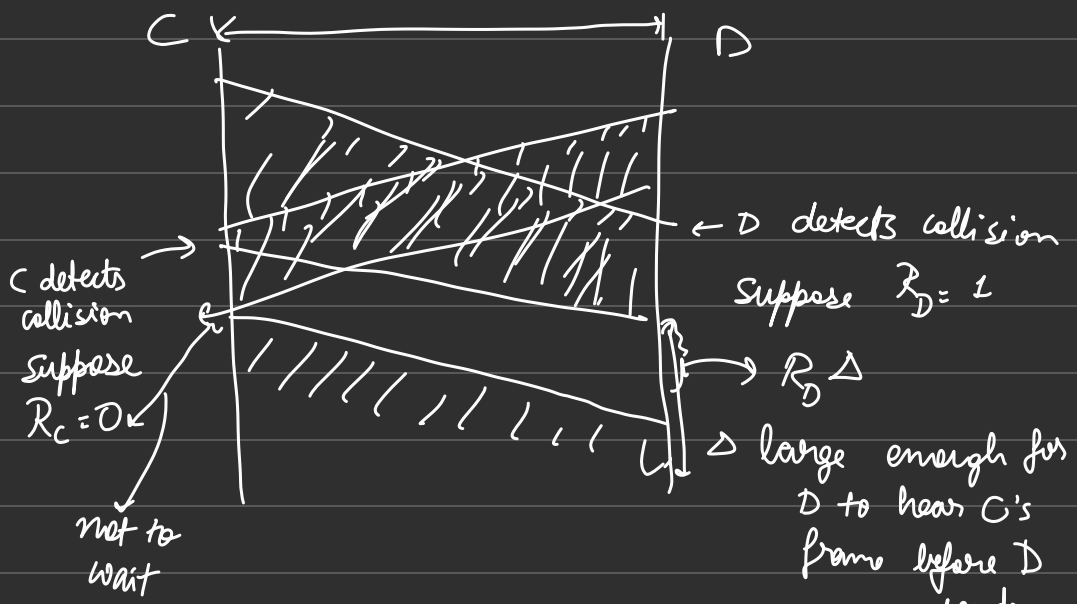
Manchester coding

Jamming signal → long enough for all to hear collision.

Random wait time

Each node throws random number R, unif. from some range

wait for $R\Delta$ before trying to send again.



Now show if $\Delta > RTT$, then D will not transmit before C.

$RTT = 2 \times$ (time for signal to go from one end of network to the other)

Max cable length = 2500m

Speed of signal = 2×10^8 m/s

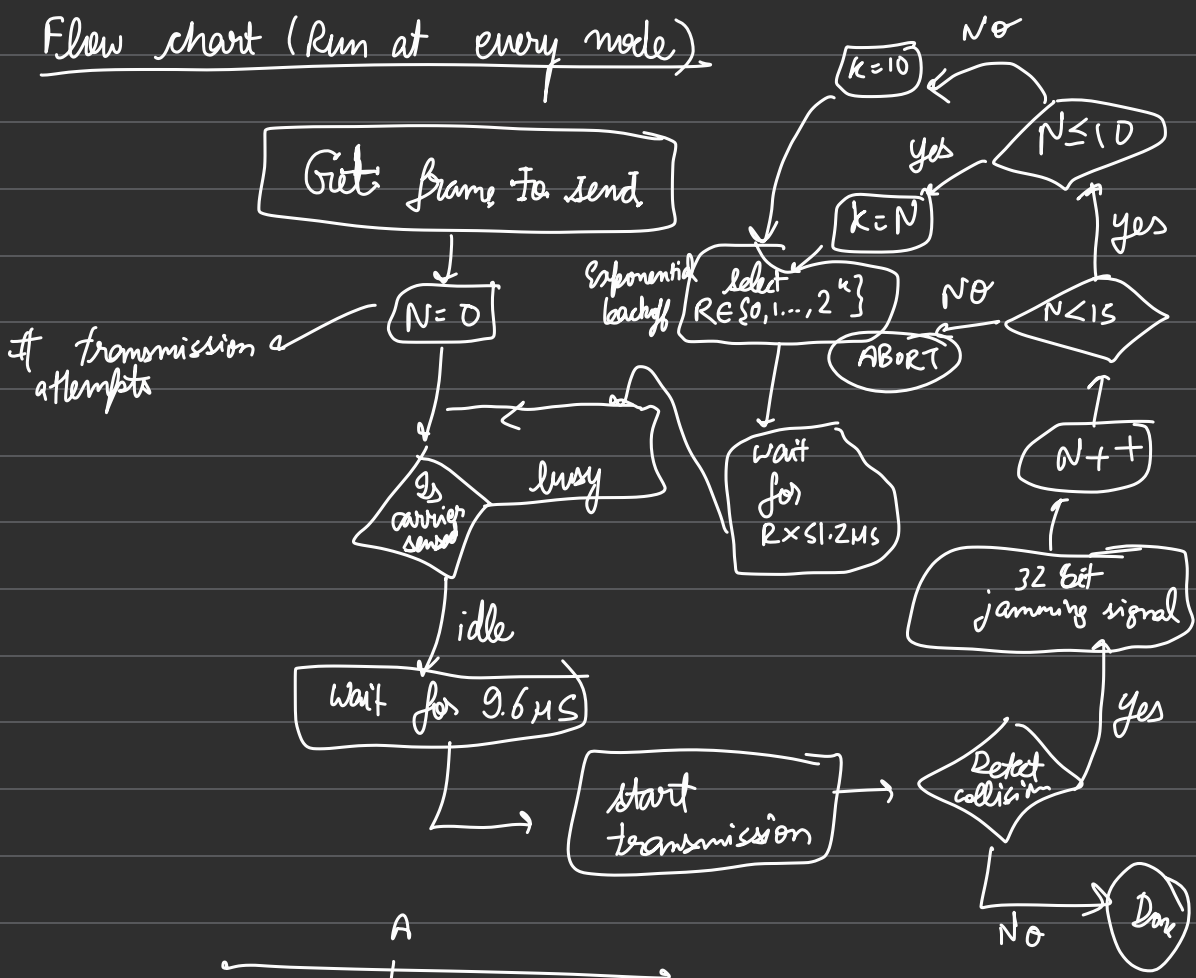
$OWD = \frac{2500}{2 \times 10^8} = 12.5\ \mu s$

$RTT = 2 \times OWD = 25\ \mu s$

Including these worst case $RTT = 51.2\ \mu s$

△ chosen by ethernet = 51.2 μs.

Flow chart (Run at every node)



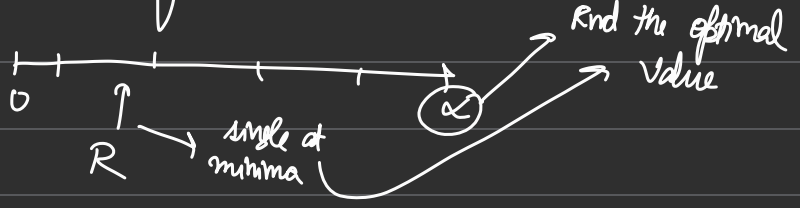
(Interference gap, for receiver to finish processing prev. frame)

Ex: CRC check

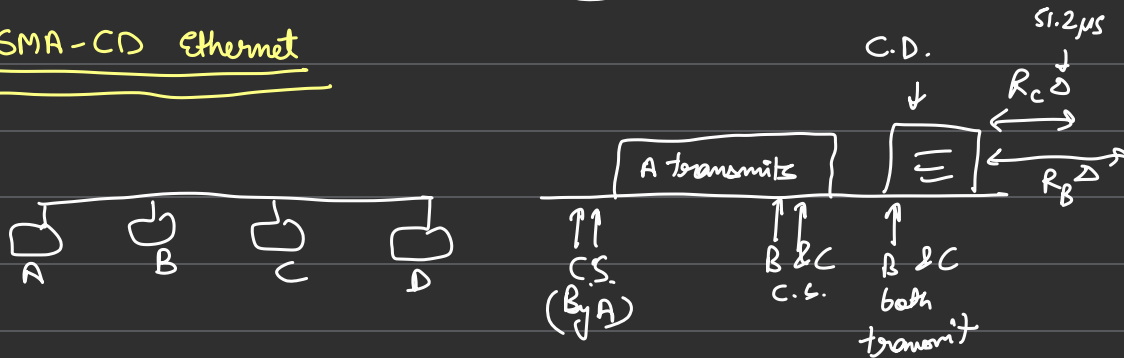
Exponential Backoff

Don't know how many colliding

Suppose M colliding



CSMA-CD Ethernet



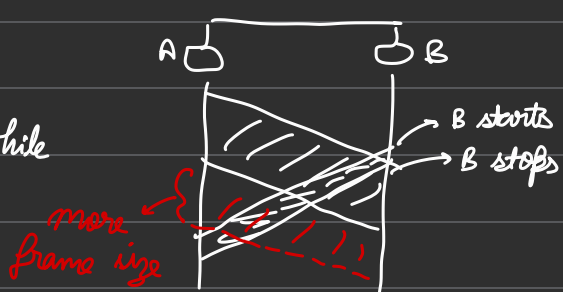
$R \in [0, 1, 2, \ldots, 2^k]$

k increments up to 10 for every collision (for same frame)

Limits on frame size (Ethernet 802.3)

(1) Min. frame size

Want A to detect collision while transmitting



Δ = Worst case R.T.T

$\Delta = 51.2\,\mu s$

original standard 10 Mbps

Frame size 64 bytes

= 512 bits

Time to transmit @ 10 Mbps

$= \frac{512}{10} = 51.2\,\mu s$

(2) Max frame size

(a) bit errors

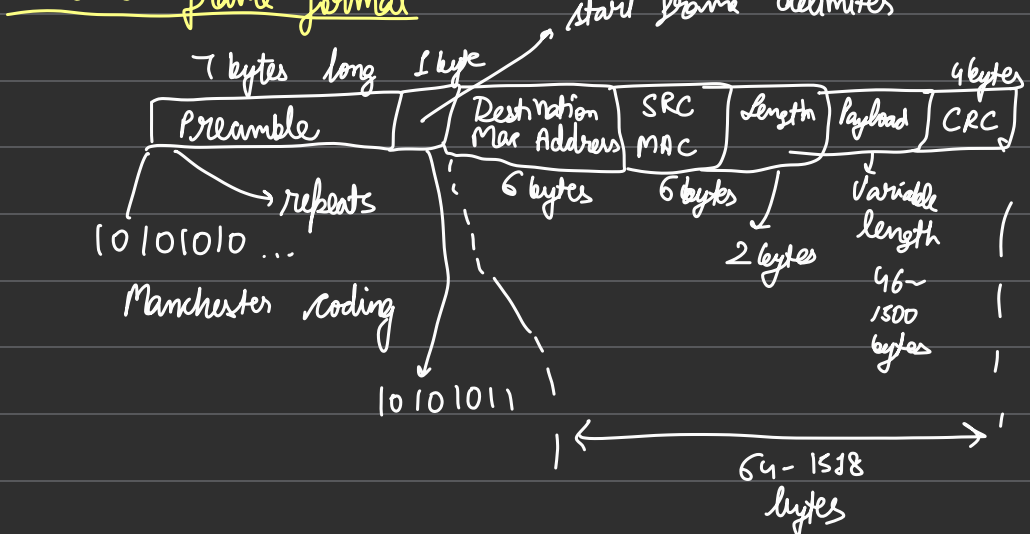
$\sim 10^8$ bits

High prob. of at least one bit errors

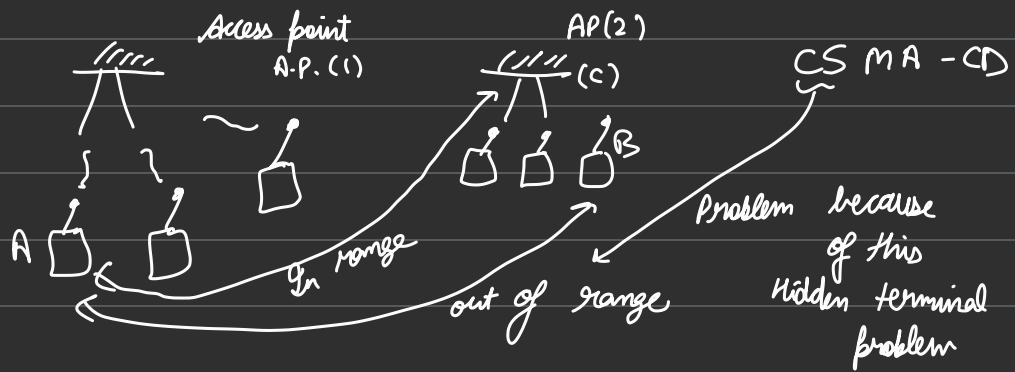
(b) Don't want single user to monopolise the channel for very long

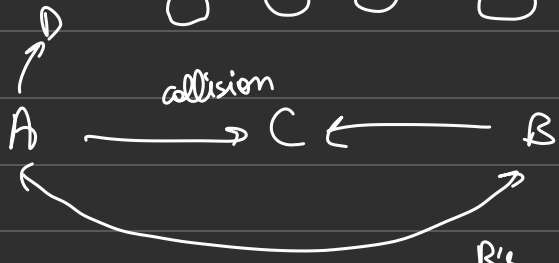
Max frame size = 1518 bytes.

Ethernet frame format

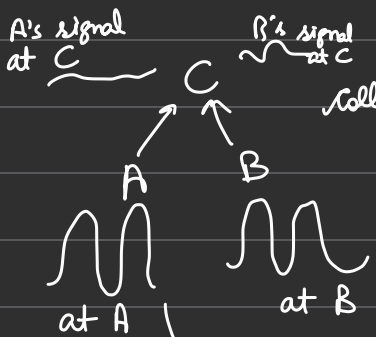


WiFi (IEEE 802.11) 802.11 b/a/g/n/ac





A cannot sense B's signal (Hidden from each other)



Collision at C

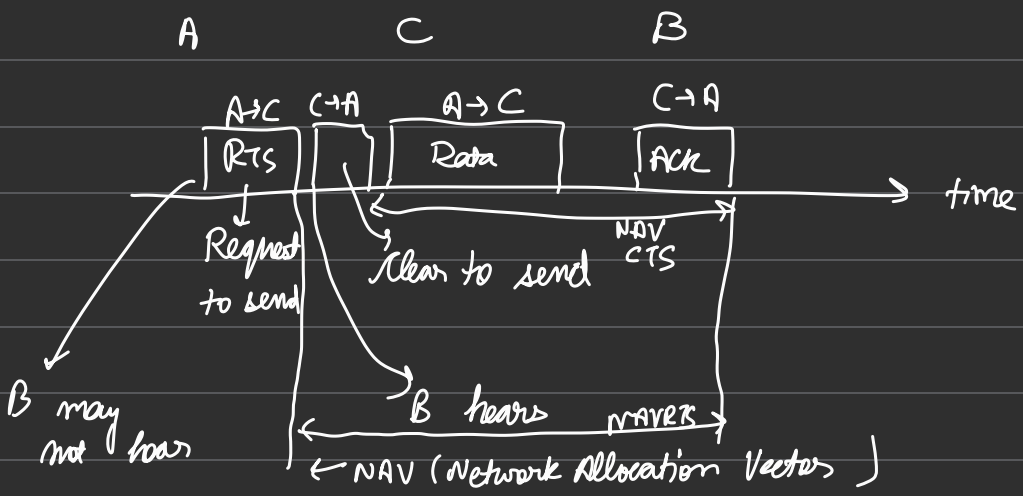
Signal decay with distance d $\sim \frac{1}{d^2}$

While transmitting A cannot hear anyone else

Collision detection does not work.

Sol^n: Receiver sends ACK

How to handle Hidden Terminal?



Symmetry is assumed (if B can hear C then C can hear B too)

Rule: All hearing RTS & CTS (other than A & C) must remain silent for NAV duration.

↓

Virtual carrier sensing

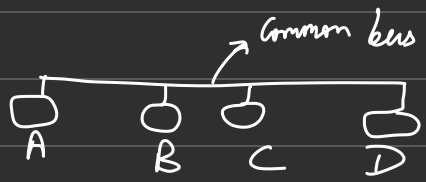
hearing CTS → B remains silent.

* What if A sends RTS but does not get CTS?
  → A assumes collision, tries to retransmit
* What if RTS, CTS, DATA are sent but ACK didn't reach A?

→ A assumes collision, tries to retransmit (starting with Data)

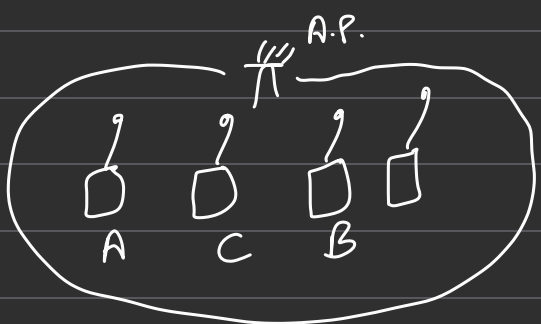
WiFi MAC (IEEE 802.11)

LAN



CSMA-CD

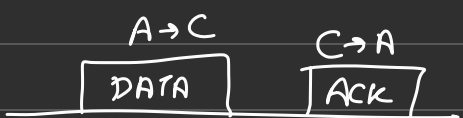
Coll. Det

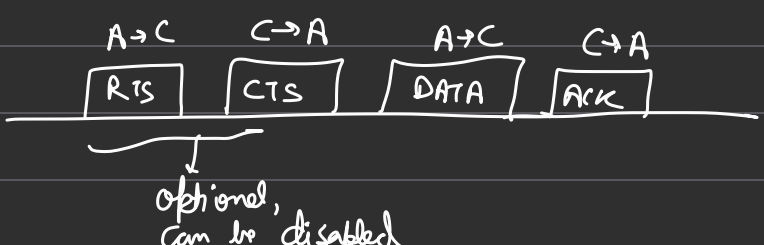


X

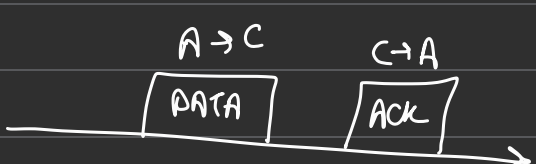
CSMA-CA

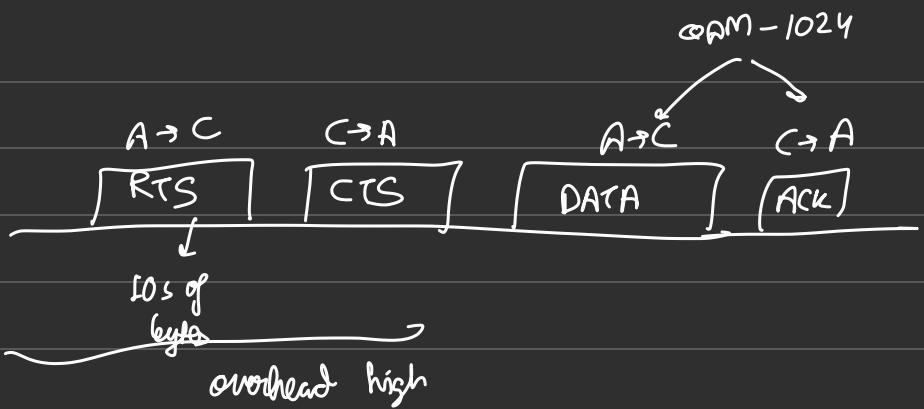
Collision Avoidance



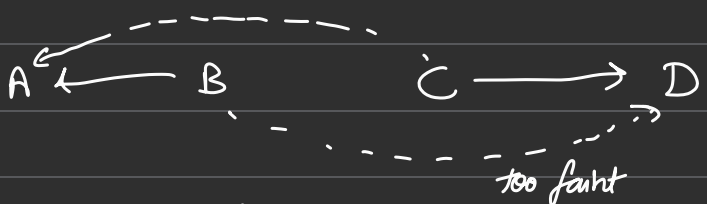


QAM – 1024





Exposed Terminal Problem :

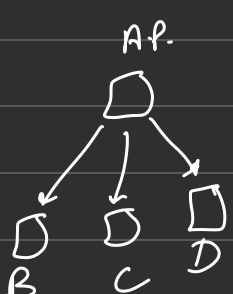


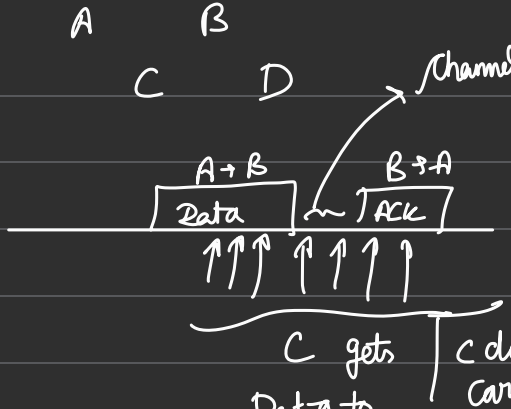
A can hear only B.
B hears A, C
D hears only C

Case (i) RTS/CTS enabled
If B starts first, C remains silent (due to RTS)

Case -(ii) RTS/CTS disabled B starts, C does carrier sense so C remains silent.

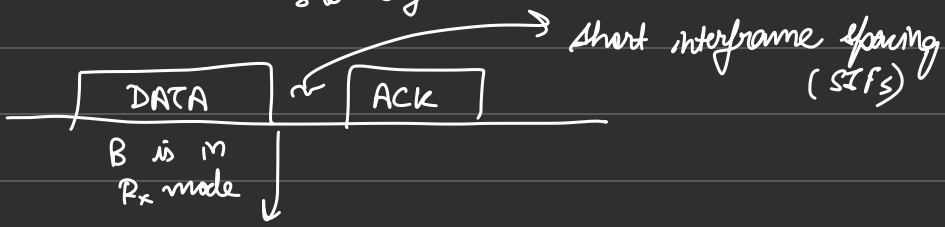
Usually :





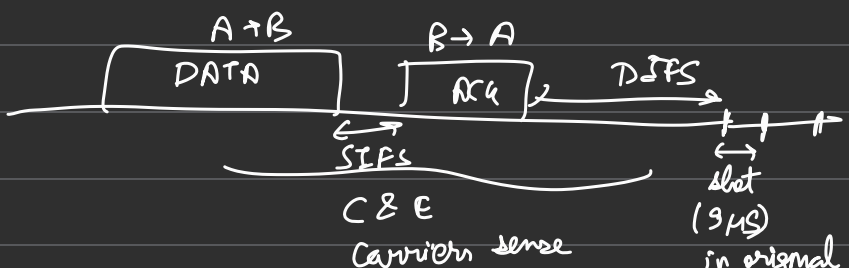
(1) Don't want C & E to start together.

(2) Don't want C & E to send between the data and ACK.



B has to receive Data frame, check for errors, read frame, switch from Rx to Tx

Rule: C has to wait for atleast DIFS (SIFS + 2Xt)



C & E choose contention window random variable

$r_C \in \{0, 1 \ldots, CW_C\}$ $r_E \in \{0, 1, \ldots, CW_E\}$

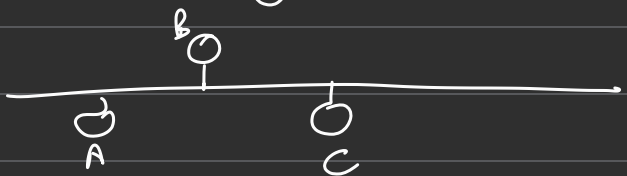
Initially $CW_{max} = 15$ upper limit $= 1023$

For collision $CW_{max} = 2 \times CW_{max} + 1$

Wireless Laws (WiFi)

A.P. (Access point)

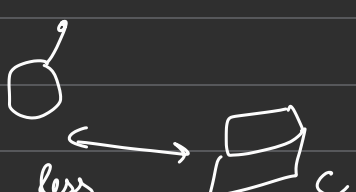
almost like noise in amplitude



decay in amplitude $\frac{1}{d^{\alpha}}$ $2 < \alpha < 5$

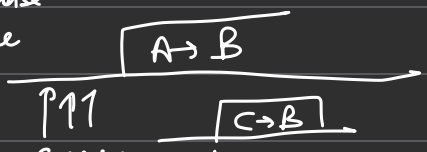
theoretically d^2

practically decay is faster than d^2



↳ so C.S. work

CSMA-CD will it work?

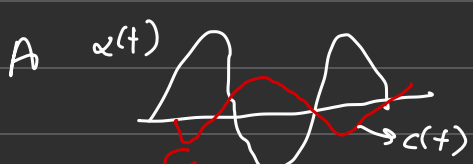


C cannot hear A in carrier sense

Collision detection will also not work

↓

because difference in amplitudes is of orders of magnitude



A+C → hearing

$-A$

↓

to get C

A receives $\beta \cdot x(t) + \theta \cdot c(t)$

$\beta \neq 1$ $\quad$ $\theta \ll 1$

$\beta a(t) + \theta c(t) - x(t)$

$= \underbrace{(\beta - 1) x(t)}_{\text{still large than } \theta \cdot c(t)} + \theta c(t)$

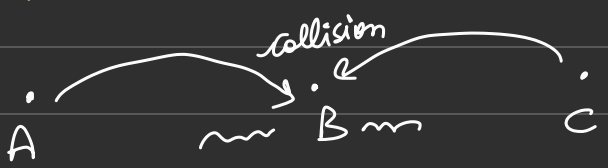
C.S. ✓ How to detect collisions ?

No Ack

↓

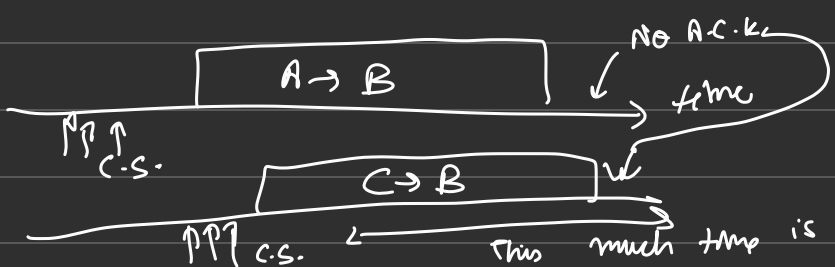
Assume collision has taken place

Hidden Terminal problem :



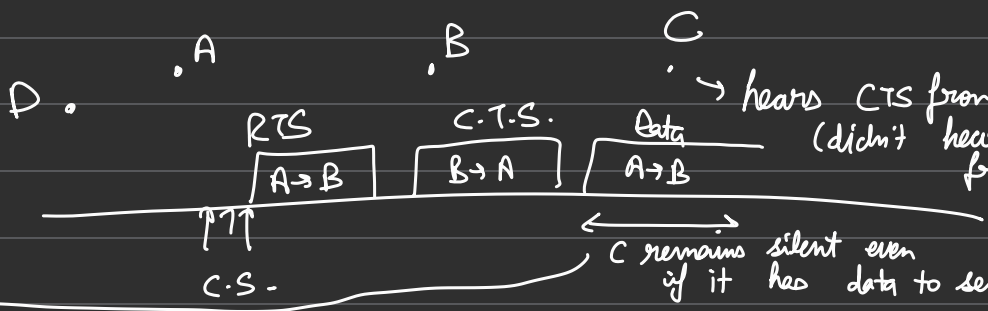
B can hear A and C

A and C cannot hear each other (hidden)



Solution to save the wasted time → Virtual Carrier Sensing

RTS: Req. to send (short frame to tell the receiver it intends to send a data frame



CTS → clear to send (short frame, saying that sender can transmit DATA)

One solution: wait till B→A acknowledge

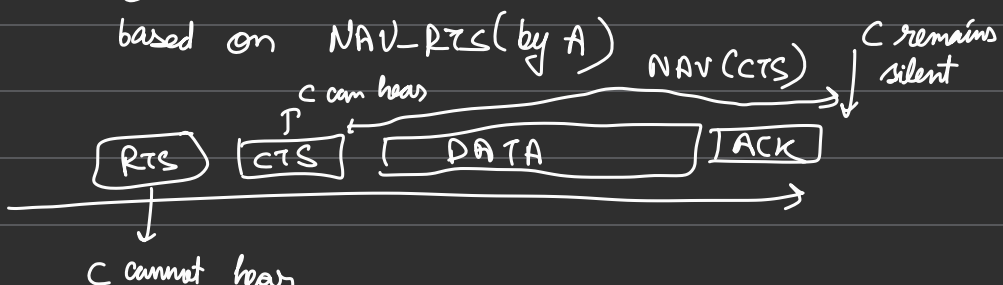
Another solution: CTS has the length of waiting time

NAV: Network Alloc Vector

NAV.CTS (by B)

↓

based on NAV_RTS (by A)



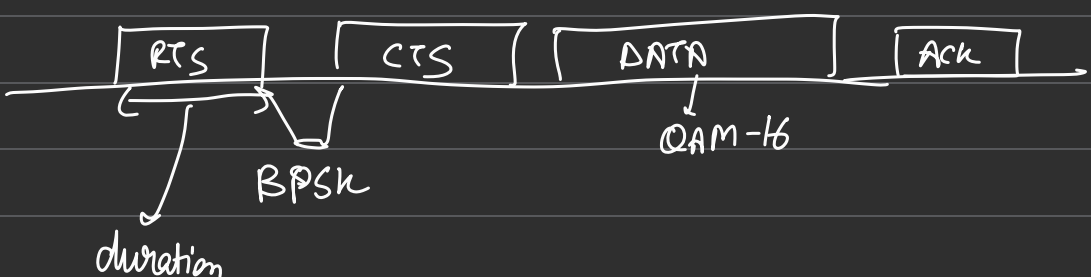
Rule: Anyone hearing an RTS or CTS should remain silent for NAV mentioned in it.

Option: (i) Don't use RTS/CTS : C.S., DATA, Get ACK

(ii) Use RTS/CTS : C.S., RTS, CTS, DATA, ACK

Get ACK → try to retransmit if No ACK

RTS/CTS → Usually sent using lower modulation (eg. BPSK) and hence their duration is large.



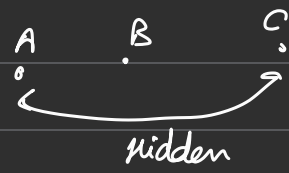
No, CTS, assume collision, try to retransmit later

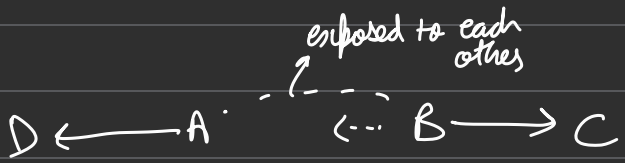
Wifi

CSMA - CA

Collision avoidance

Expose Terminal Problem:

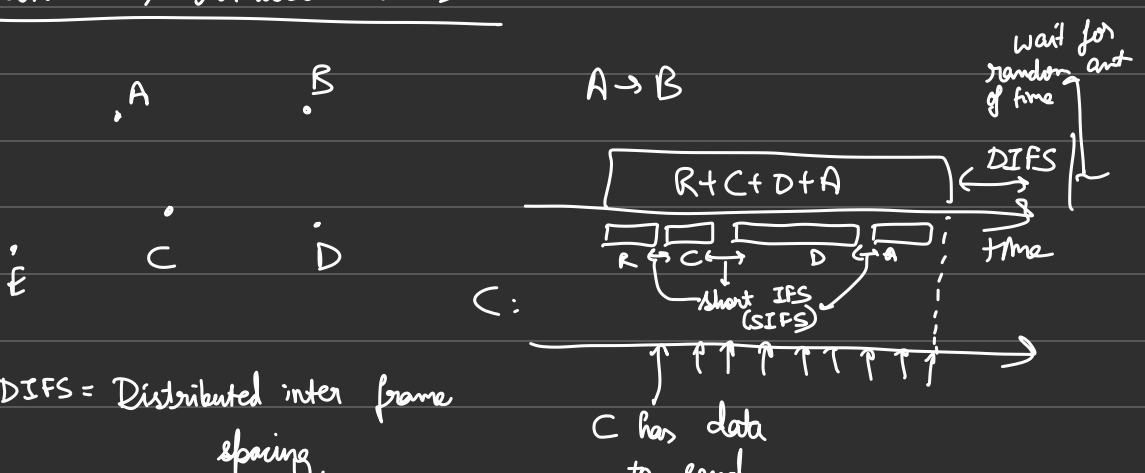




D can receive from A and C can receive from B simultaneously (in theory)

However, if A→D then B remains silent due to C.S.

Contention Window (CW)



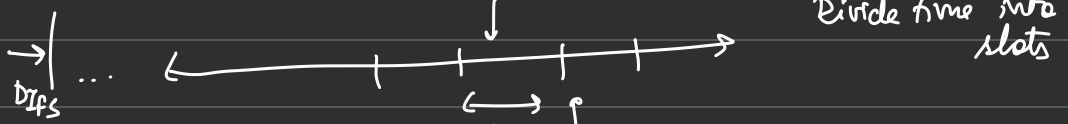
DIFS = Distributed inter frame spacing.

SIFS? : B hears RTS, A sent RTS, Tx → Rx / Rx → Tx circuit shift starts listening

decodes RTS
(PHY/MAC processing)

DIFS > SIFS

A starts transmitting



Divide time into slots

D; wait time was one more slot than A

should allow D to hear A and not transmit

prop. delay + possible offset (starting time for measuring slots) + time to C.S.

C must hear A, carries sense it → If A was transmitting

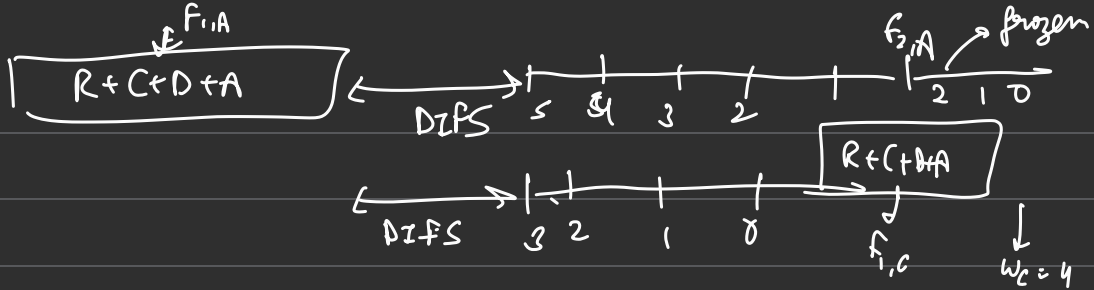
prop. delay + C.S. time + RX. → TX → No one else was transmitting

How long to wait?

Remaining waiting time W

Initialize $W \in Unif(0, CW_{max})$

For every idle slot, decrement W by 1.



Collision: sent RTS, no CTS

sent DATA, no ACK

$CW_{max} = CW_{max} \times 2$ → on collision

$CW_{max} = \max(CW_{max}, \text{max allowed value})$

repeat procedure:

$w \in Unif(0, CW_{max})$

wait for DIFS free period

decrement w every free slot

if channel get busy

freeze w

Transmit when $w = 0$

IEEE 802.11 b, g, n, ac, ax

↓ latest

Frame:

- MAC addr of sender and receiver
- Framing ↓ check sum (CRC)

rate 3/4 : send N bits, $\frac{3}{4}$ N information bits

5/6 : " $\frac{5}{6}$ N information bits

802.11 g → 64 QAM, rate $(\frac{3}{4})$ (54 Mbps) ↓ bit rate — Chl width 20 MHz

- 11n → 64 QAM, rate $(\frac{5}{6})$ (150 Mbps) 40 MHz
- 11ac → 256 QAM, rate $(\frac{3}{4}, \frac{5}{6})$ (866 Mbps) 160 MHz
- 11ax → 1024-QAM, rate ... $\frac{5}{6}$ (1.2 Gbps / 160 MHz

MIMO
multiple input
multiple output

SISO
single input
single output

DIFS > SIFS : To give priority to someone already transmitting

DIFS = SIFS + 2x slot time

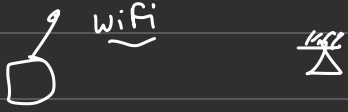
PIFS = SIFS + slot_time

↓

point coordination function. (PCF)

D → distributed CF

QoS



Higher priority possible ?

$W \in Unif(0, CW_{max})$

802.11e

	CWmax	Initial	Max value
Voice		3	7
Video		7	15
other		15	1023

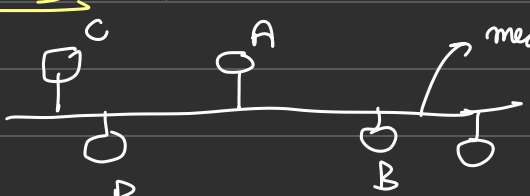
Need cross-layer interaction

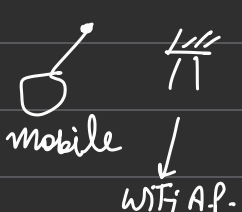
SIFS = 10 μs 11g 11n

higher for others

Max. retries if collisions and then give up.

Medium Access (PLL)





Ethernet LANs

CSMA : Carrier sense multiple access.
TDMA : Time division multiple access

→ Random access protocols : schedule for transmissions is not decided in advance.

Idea: If a node has data to send, create a frame and sends it.

Broadcast : all nodes on network are destination of frame
unicast → single node is destination of frame
multicast → a subset of nodes is destination of a frame (packet)

What if there is a collision?

A → B at same time
C → D

(1) Collision Detection → so that A and C know that a collision occurred.

(2) Back-off for random time

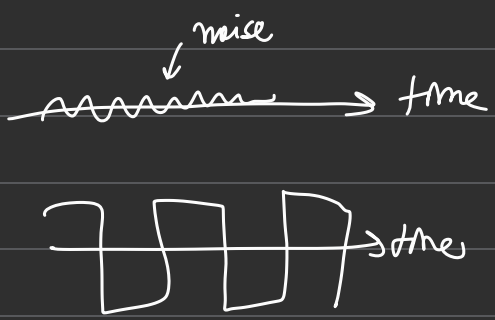
(3) Carrier-sense : If a node has data to send, first sense the medium to see if a transmission is ongoing Don't transmit if medium is busy.

CSMA-CD — IEEE 802.3

↓

collision detection

Carrier sense

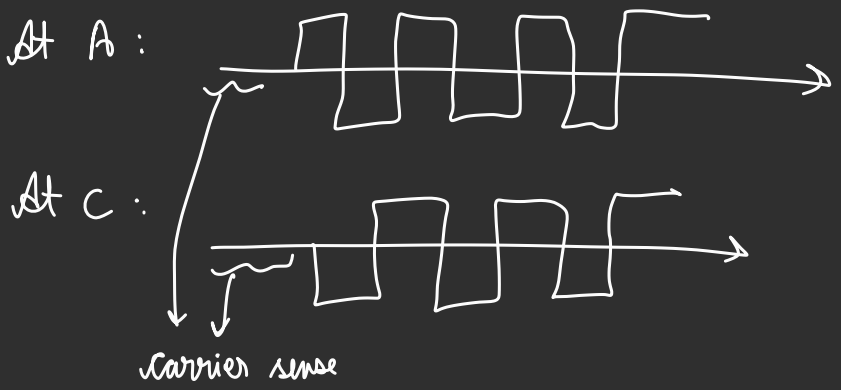


If energy of signal > threshold → conclude that chanel is busy

↓

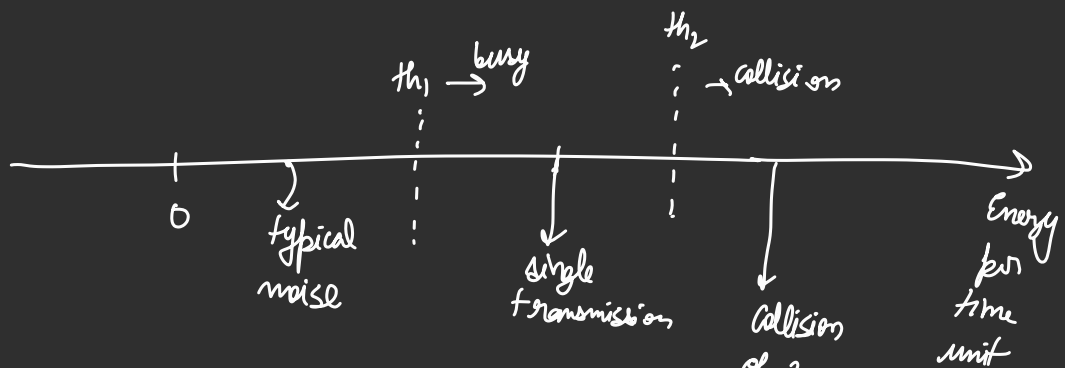
larger than noisy energy

Collision Detection



idea : Another threshold for collision detection (th_2)

energy > $Threshold_2$ then collision



Jamming signal (for short duration)

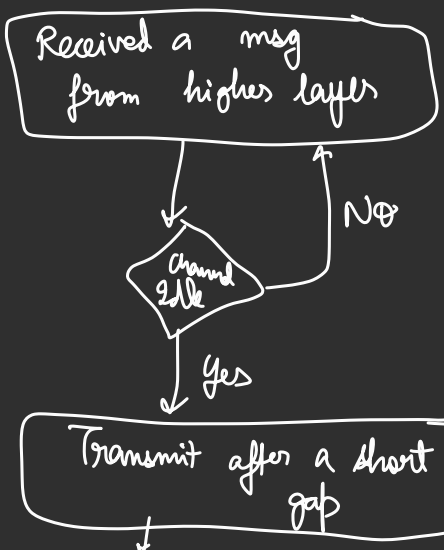
A sends

↓

for C to detect

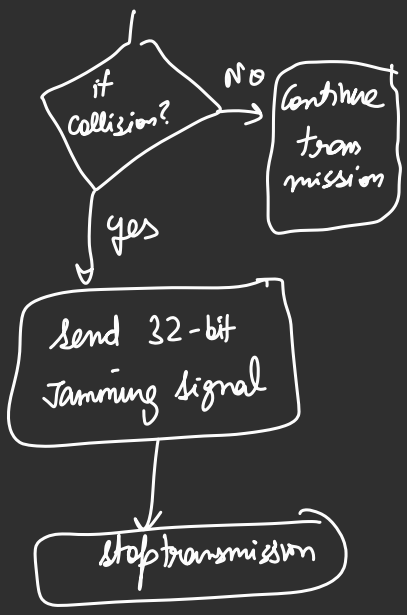
C also sends a jamming signal after collision detection

Flow chart



set to be 9.6 μs (if the receiver was the same for prev. frame, he should be given some time for processing)

Collision - Detection



Frame Details in 802.3

↙ to signify that preamble has ended

start frame delimiter

Preamble	SFD	Dest. MAC ADDR	SRC MAC ADDR	LEN	Payload	CRC
7 bytes	1 byte	6 bytes	6 bytes	2 bytes		4 bytes

64 - 1518 bytes

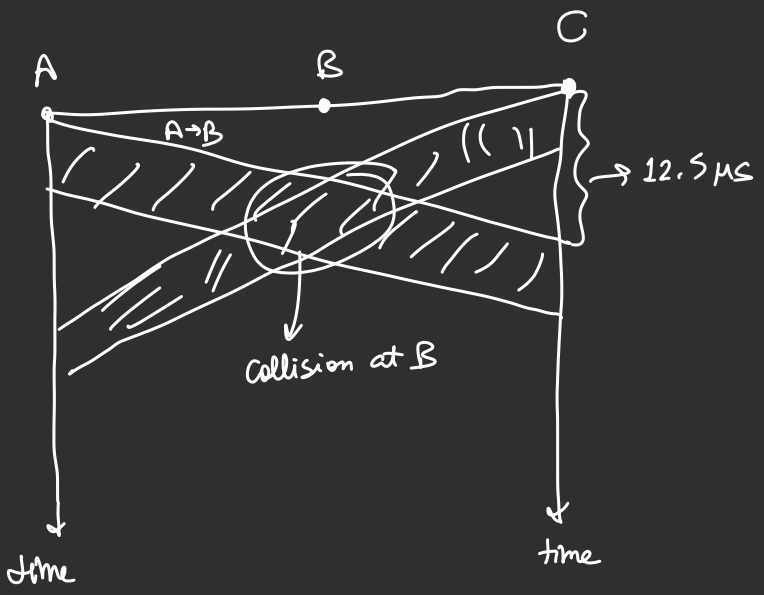
2^{48} → Mac addresses

2^{32} → IPv4

Why min frame size?

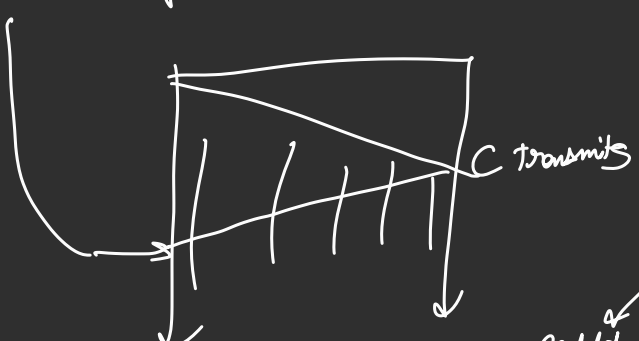
2500m (approx)

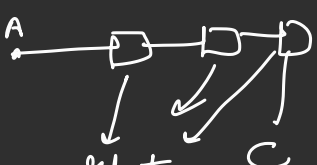
$$RTT = \frac{5000}{2 \times 10^8} = 25\,\mu s$$



A must detect collisions before his frame ends, if a collision occurs at the receiver.

Want A:s frame to continue till the end





repeates (reproduces signal on one link onto other link)

amplify and reproduce

≈ 50 μs RTT in the worst case.

64 bytes, 10 Mbps transmission rate → 51.2 μs

Max frame size

Large frames

(1) Higher prob of getting a bit in error in frame

'p' → suppose prob. of bit error

N → frame length

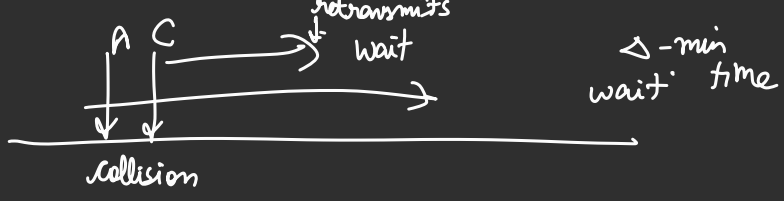
Prob of no error $(1-p)^N$ → assuming bit errors are independent

Entire frame will have to be discarded.

(2) Others need to wait longer to get a chance to transmit.

(3) Memory requirements at NIC card go up.

Back-off



wait ∈ Δ·V → {0, 1, ..., V} → larger if more are colliding

Exponential backoff

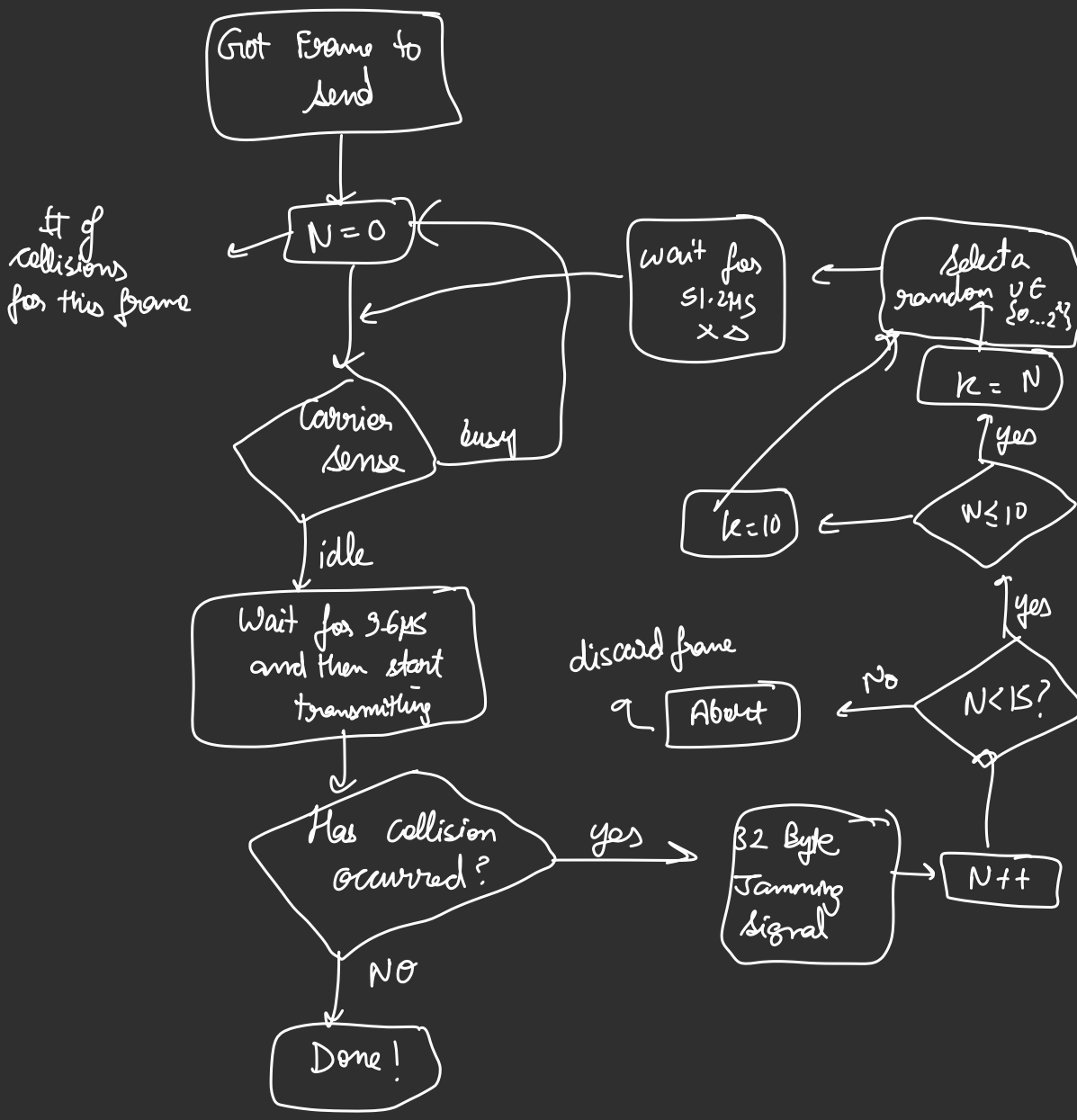
1st collision : k = 1 ;

2nd collision : k = 2 ;

⋮

10th collision k = 10 ; 11th → k = 10 ... 15th

Flow Chart


Got Frame to send
N=0
# of collisions for this frame
Carrier sense
busy
idle
Wait for 9.6μs and then start transmitting
Has collision occurred?
yes
NO
Done!
32 Byte Jamming Signal
N++
N<15?
No
discard frame & Abort
yes
N≤10
yes
k=10
k=N
Select a random integer from {0...2^k - 1}
wait for 51.2μs x Δ

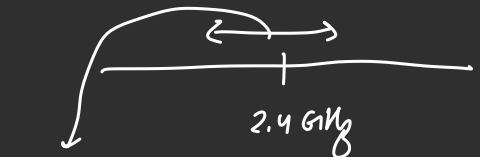
MAC

CSMA – CD / CA → Collision Avoidance

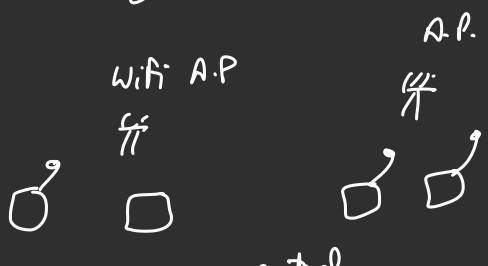
↓ ~ wifi

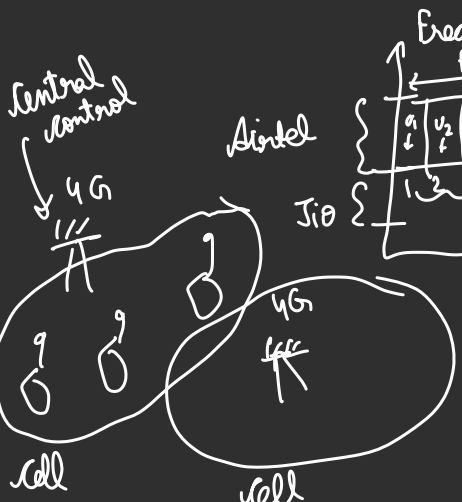
Collision Detection

Decentralised



80 MHz → unlicensed





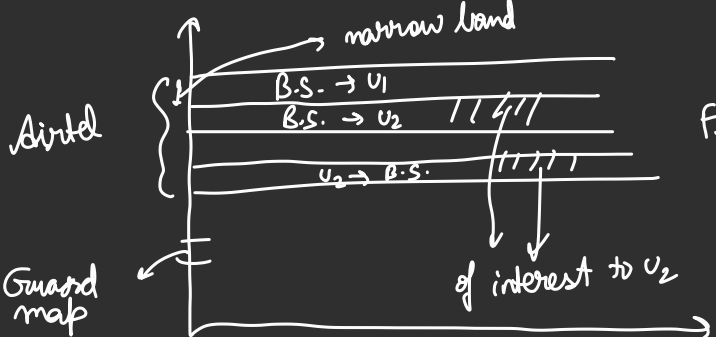
Uplink : User → B.S.

Downlink : B.S → Users

schedule

UL-MAP

DL-MAP



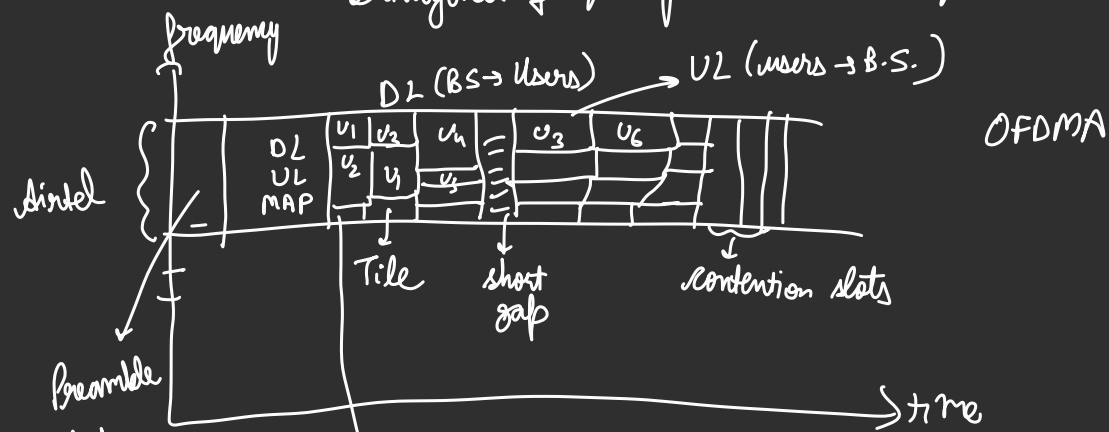
FDMA

Frequency division multiple access

4G-LTE uses OFDMA

↓

orthogonal frequency division multiple access



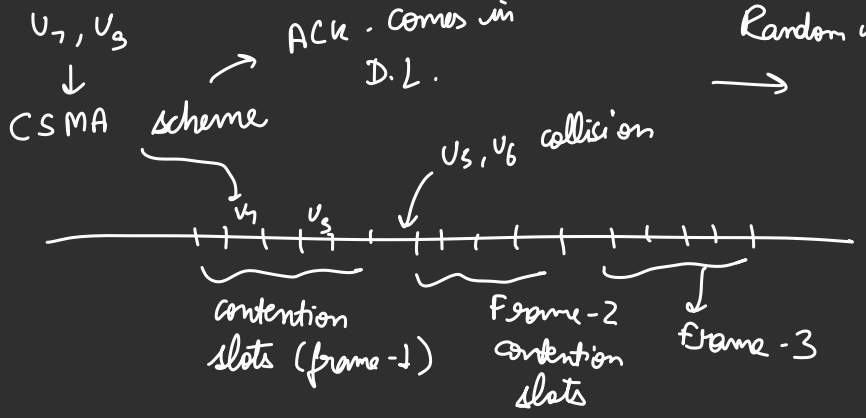
(1) clock synchronisation (to B.S.)

(2) frame start

(3) attenuation & phase changes (delay)

good for u_1

bad for u_2



OFDMA

OFDM → in wifi

multiplexing

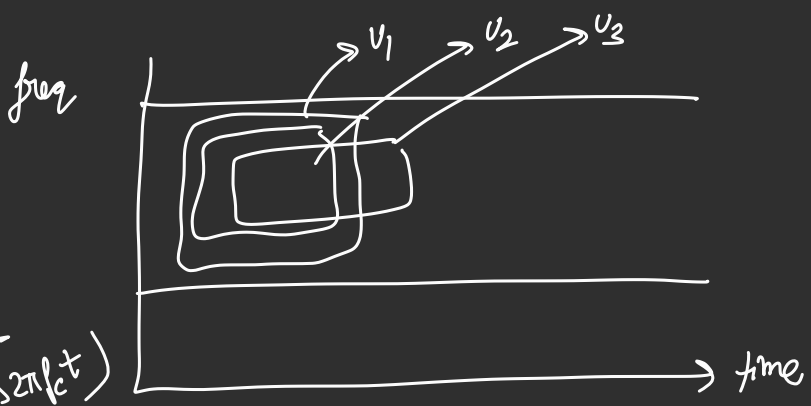
f_c'' → cos / sin $2\pi f_c'' t$ → any modulation

f_c

f_c'

for this narrow band conten-freq = f_c' → cos $2\pi f_c' t$, sin $2\pi f_c' t$ → BPSK, QPSK, QAM − 16

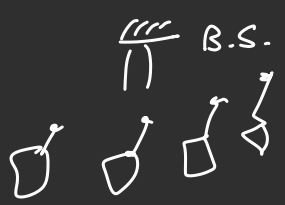
CDMA : Code Division Multiple Access



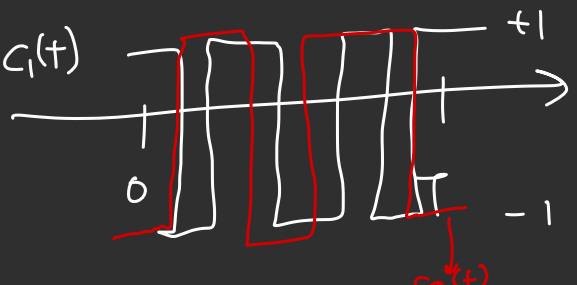
Spreading Code

v_1: $(A\cos 2\pi f_c t)$ → $c_1(t)$, $0 < t < \frac{N}{f_c}$

v_2: $(-A\cos 2\pi f_c t)$ → $c_2(t)$, $0 < t < \frac{N}{f_c}$



$v_i = c_i(t) \times$ (signal)



$c_1^2(t) = 1$

$c_1(t) \perp c_2(t)$

$\int_0^T c_1(t) c_2(t) = 0$

All codes are mutually orthogonal

B.S $r(t) = \sum_{i=1}^{n} s_i(t)$ (n: # users)

$r(t) \times c_1(t) \cos 2\pi f_c t$

$s_1(t) \times c_1(t) \cos 2\pi f_c t$

$= (c_1(t) A \cdot \cos 2\pi f_c t)(c_1(t) \cos 2\pi f_c t)$

$= A \cos^2 2\pi f_c t$

$= \frac{A}{2}\left(1 + \cos 4\pi f_c t\right)$

$\cos 4\pi f_c t$: removed by integration (also by low pass filter since $2f_c$ is a high frequency)

$s_2(t)\, c_1(t) \cos 2\pi f_c t$

$= -c_2(t) c_1(t) A \cos^2 2\pi f_c t$

$= \frac{-A}{2} c_1(t) c_2(t)\left(1 + \cos 4\pi f_c t\right)$

$\cos 4\pi f_c t$: low pass filter remove

After low pass filter

$$\int_0^T \frac{A}{2} + \left(\frac{-A}{2} \underbrace{c_1(t) c_2(t)}_{\text{orthogonal}}\right) dt = \frac{AT}{2}$$

CFDM, OFDM robust to multi-path

CFDM → 3G

OFDM → Wifi, 4G

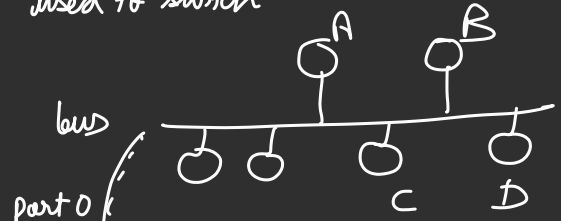
Switching → L2- switches (Ethernet)

↓

MAC Address used to switch

L3- switches (Routers)

↳ use IP address to switch

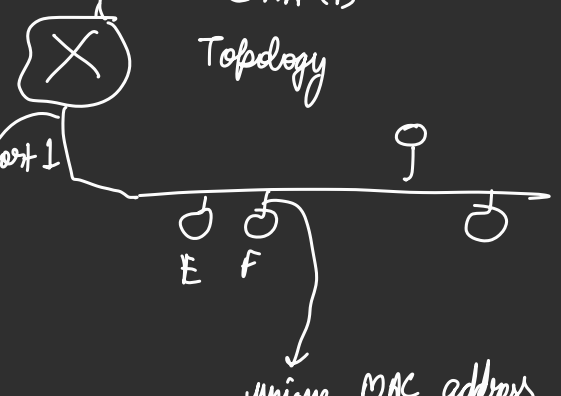


CSMA-CD Topology

Switch

Intelligent isolation (for scalability)

Isolating different parts of LAN



port also have MAC addresses

unique MAC address 6 bytes (48 bits)

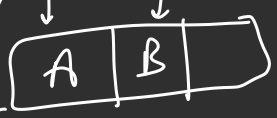
FORWARDING Table

Dest	Port #
A	0
B	0
C	0
D	1
E	1
F	1

Init Table

Dest	Port #
A	0
B	0
F	1

populates the table with info



Frame

expiry time

delete after expiry time

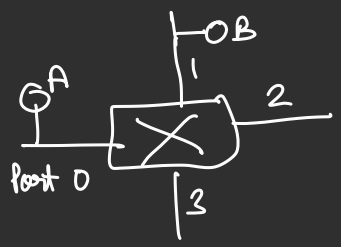
forwards to port 1

forwards to port 1

→ received on port 0

Don't forward.

Ethernet switches = bridges



(MAC)

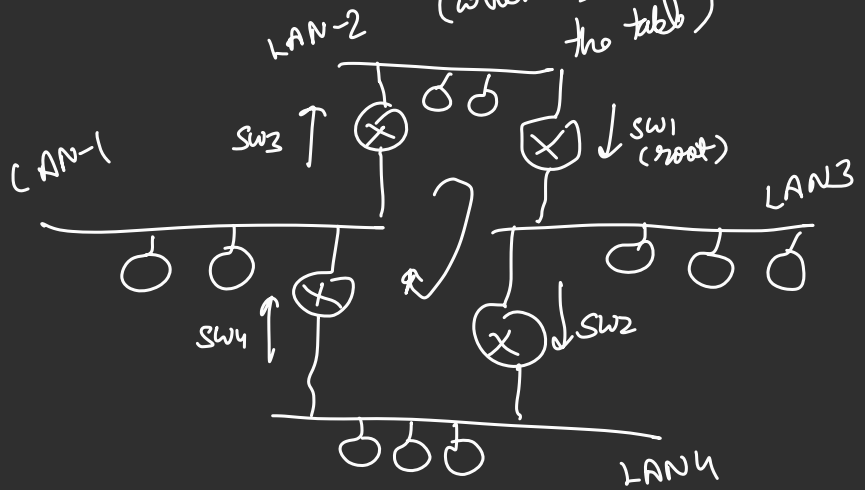
Dest	Port
A	0

If dest is not in the table, then forward on all ports

populates A in the table

A	B

send it to all ports (when B is not present in the table)



Frame is going around forever (using resources)

We need to avoid loops.

Spanning Tree Protocol
(Radia Perlman)

SPT Protocol

(1) Elect root bridge
(2) Each bridge finds which port is closest to root, assigns this port as root port. (Tie breaking rule) → Designated port.
(3) Any port which is not a root port or a designated port is disabled

Details:

(1) Elect root

Bridge ID

Lowest becomes root

Configurable part (2 bytes) | 6 bytes MAC Address (smallest MAC of all ports)

default value: 32768

0 – 61440 (multiples of 4096)

Each bridge tells its neighbours

(Y, d, X)

- Y: the smallest ID heard till now
- d: distance to Y
- X: my ID

SW_1 : (1, 0, 1)

⋮

SW_4 : (4, 0, 4)

after hearing SW_2, SW_3

SW_2 : (1, 1, 2)

⋮

SW_4 : (2, 1, 4)

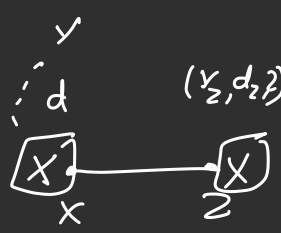
⋮ converge

SW_1 : (1, 0, 1)

SW_2 : (1, 1, 2)

SW_3 : (1, 1, 3)

SW_4 : (1, 2, 4)



(Y, d, X)

If $Y < Y_2$

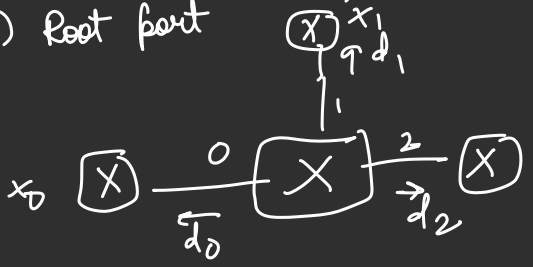
then $Y_2 = Y$

$d_2 = d + dist(X, 2)$

If $Y = Y_2$ but $d + dist(X, 2) < d_2$

then $d_2 = d + dist(X, 2)$

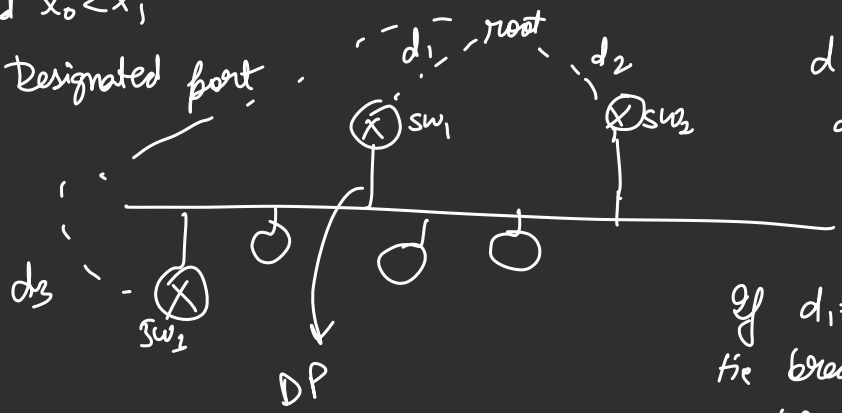
(2) Root port



if more than one port has the same smallest distance to root, then tie-break based on ID (smallest)

If $d_0 = d_1 < d_2$ and $x_0 < x_1$ → 0 = root port

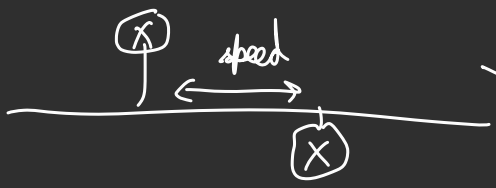
(3) Designated port.



$d_1 < d_2$
$d_1 < d_3$

If $d_1 = d_2 < d_3$ then tie break based on ID of SW_1, SW_2, SW_3

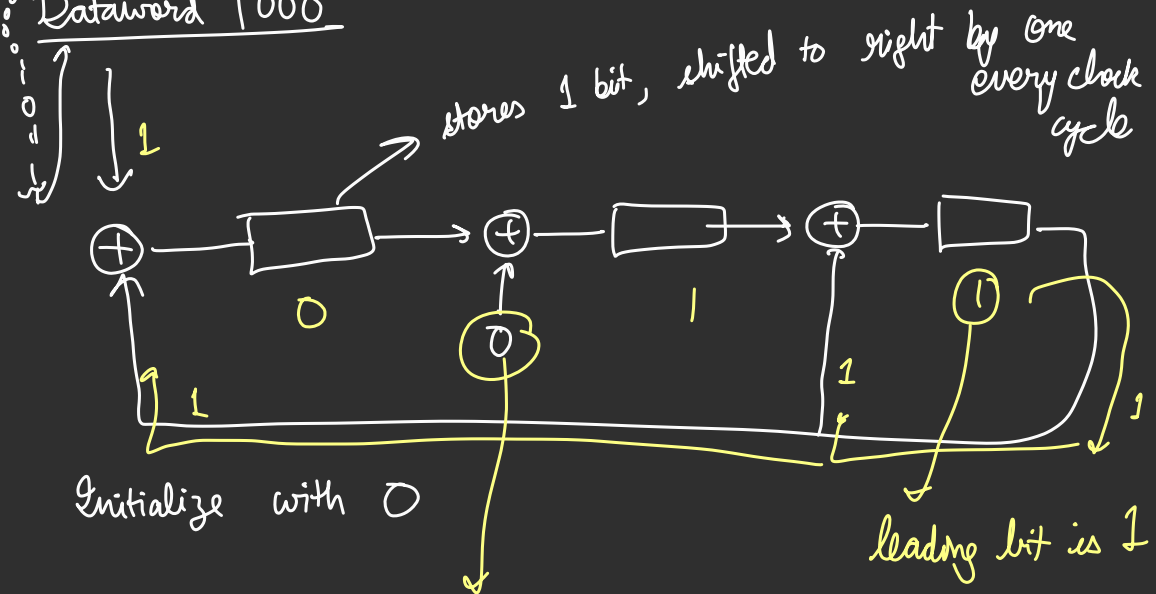
Ports which are not R.P. or D.P. are disabled.



speed	cost
10 Gbps	2
1 Gbps	4
100 Mbps	19
10 Mbps	100

distance function based on speed

Dataword 1000



XOR with 101
or 000

$C(x) = (101$
leading bit

Last bits in the circuit will be CRC

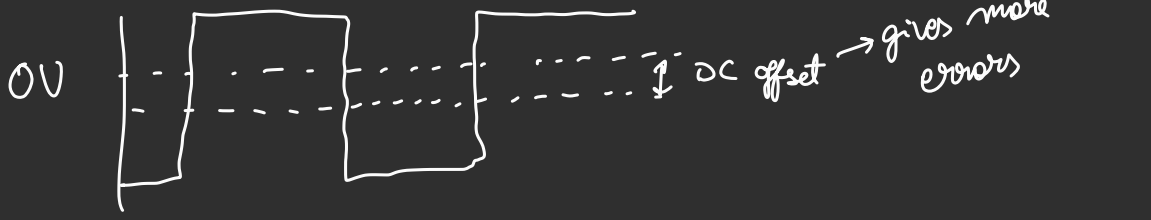
Baseline Wander (Problem with NRZ)

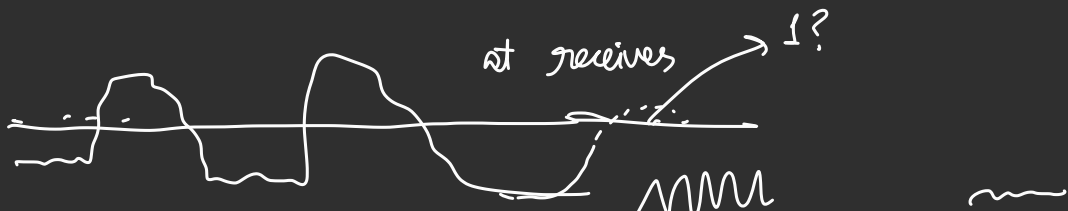
Receiver

High-pass-filter

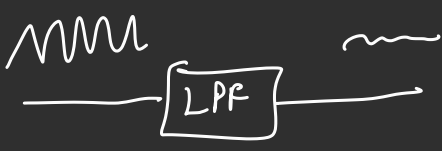
Sender

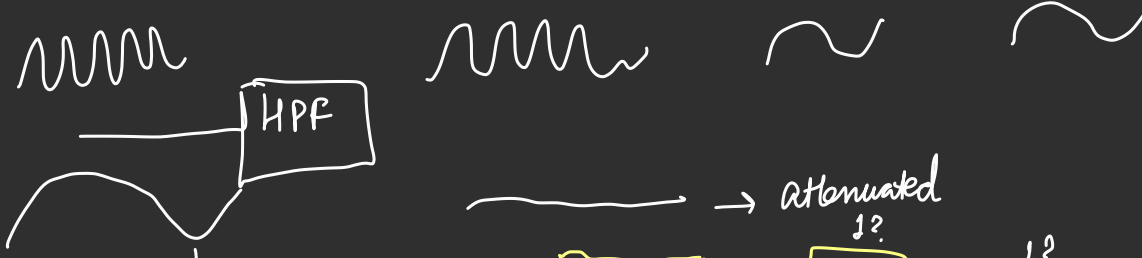
Amplifier



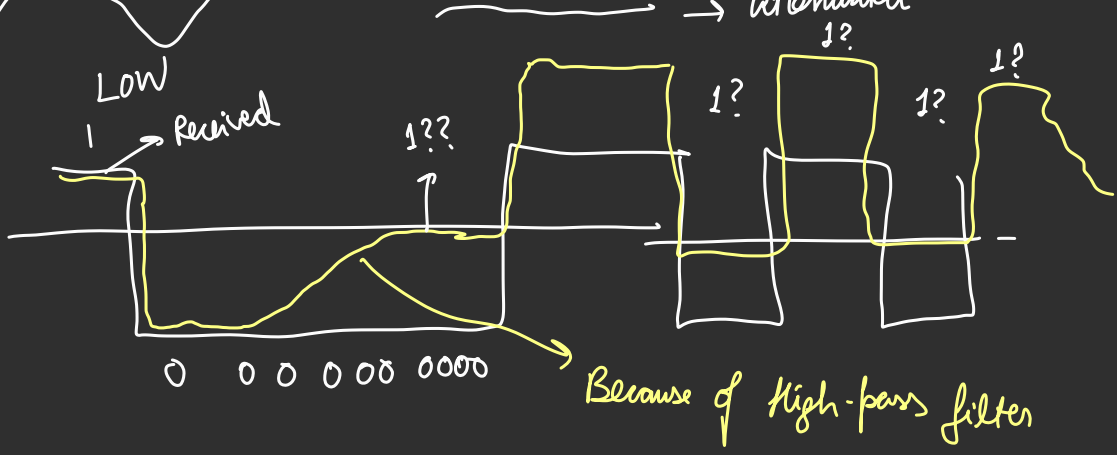


High-freq



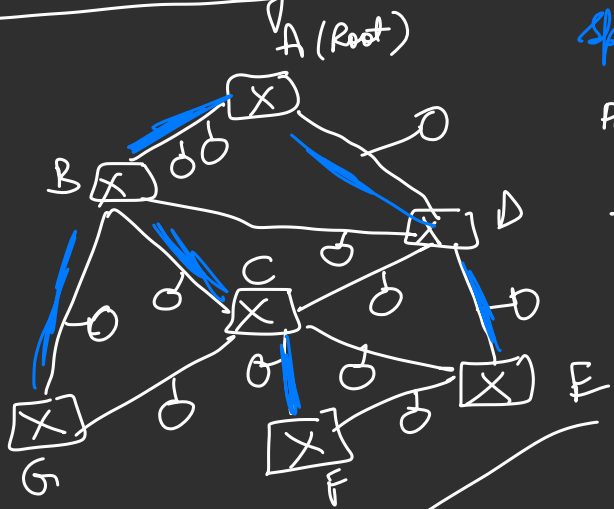


→ attenuated



Layer-3 Switching

Shortcomings of Ethernet switching:



Spanning tree

C → E:
C - B - A - D - E

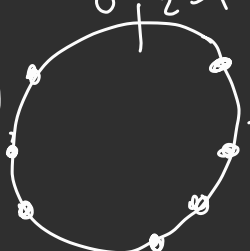
- path is not optimal (shortest path)
- not using some links - resource utilization poor
- O(N) table size
- Stability

Fwd table

Dest	Port
host-1	0
host-2	1
⋮	⋮
all hosts	1

N hosts
O(N)

Flat addressing (Layer-2)



→ MAC addresses

Layer-3 (IP): Hierarchical addressing

If root fails: SPT is reconstructed

↳ roots sends a frequent msg that other nodes should hear

↳ Hello messages from root. don't come

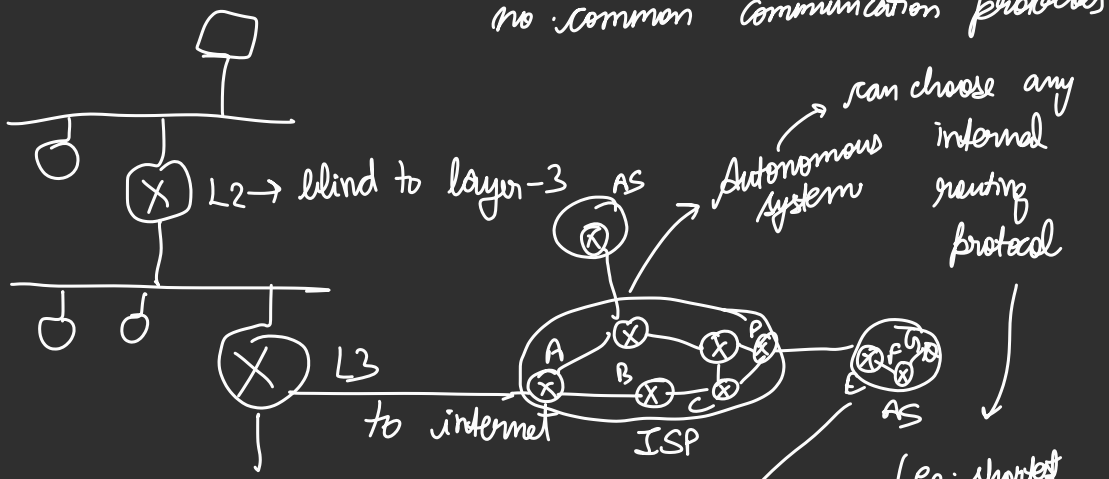
or other switch in spanning tree fails

Larger LANs → more freq. SPT construction

Other issues : No common addressing scheme across globe.

↓

no common communication protocols



A → G

A - B - C - D - E - F - G

not same optimum criteria

Intra-domain routing: Routing protocols within AS

Inter-domain routing : Routing protocols between AS

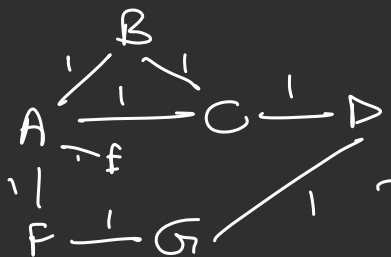
↳ Borders Gateway Protocol (BGP)

Intra domain routing

- Distance vector
  - RIP routing information protocol
- Link-state routing
  - OSPF : open shortest path first
  - IS-IS : Intermediate system to IS

Use shortest path avoid loops.

Distance Vector



A sends out:

(A, 0)

↓ distance to myself is 0

A hears (B,0) (F,0) (C,0)

A sends out to OWN neighbours :

(A,0), (B,1) (C,1) (F,1))

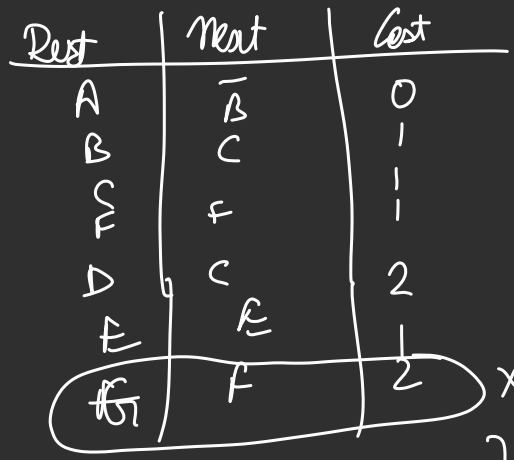
hears from (B) (C,1) (A,1)

from (C) (D,1) (B,1) (A,1)

from (F) (A,1) (G,1)

Fwd (Routing) Table at A

Dest	Next hop	Cost
A	–	0
B	B	1
C	C	1
F	F	1
E	E	)



If F ⊗ G fails

F tells A that its distance to G = ∞

× Invalid

replace → G : — ∞ } Announce this to neighbours

G : C ∞

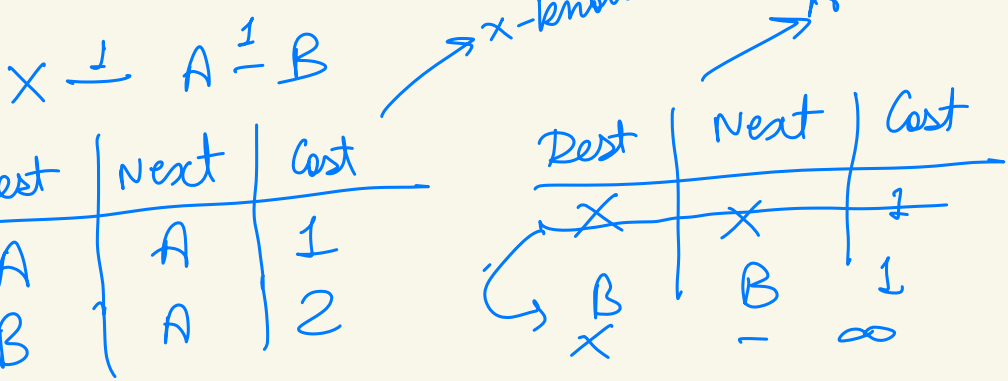
C tells A that I have distance 2 to G.

- Triggered update : Event triggers a routing update to neighbours

  Eg: F ⊗ G fails which triggers an update (G, ∞) to A

- Periodic updates : Tell neighbours information in routing table about (dest, dist) to various destinations

Count to Infinity Problem



Dest	next	Cost
A	A	1
X	A	2

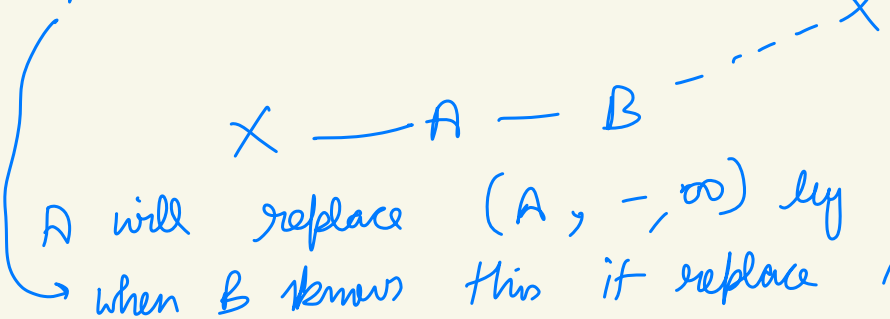
→ B knows

If X-A link fails, A needs to send to B

B sends (X, ∞), (X, 2) → These are sent at the same time

B tells A: (A, 1), (X, 2)

A tells B: (X, ∞), (B, ')



A will replace (A, -, ∞) by (X, B, 3)

when B knows this it replace (X, A, 2) by (X, -, ∞)

A tells B : (x, 3), (B, 1) → B receives and updates

B tells A : (x, 4), (A, 1) ← Replaces (x, -, ∞) by (x, A, 4)

In A's table (x, B, 3) replaced by (x, B, 5)

A tells B : (x, 5), (B, 1)

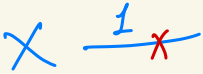
B tells A : (x, 6) (A, 1)

* This is the routing loop and goes to infinity.

* Routing Information Protocol (RIP) : max distance of 16
  Cost = 16 ⇒ cannot reach destination.

Split - Horizon

Do not advertize information about a destination to a neighbour if that neighbour is the next hop to the destination.



Dist	next	Cost
B	B	1
X	-	∞

Dist	next	Cost
A	A	1
X	A	2

Not told to A under split horizon.

A tells B: (X, ∞)

B tells A: nothing about dist to X.

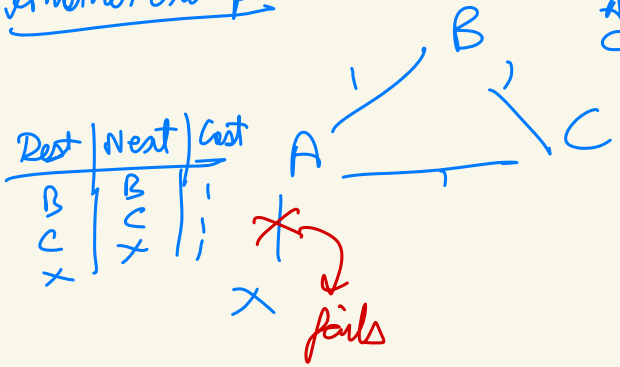
B updates $(X, A, 2)$ to $(X, -, \infty)$

Split Horizon with Poison Reverse:

A node tells its next hop to a destination that its distance to the destination is ∞.

B sends advertisements (X, ∞) to A.

Another example:



Dest	next	Cost
A	A	1
C	C	1
X	A	2

A sends (X, ∞) to B & C

Dest	Next	Cost
B	B	1
C	C	1
X	X	1

Dest	Next	Cost
A	A	1
B	B	1
X	A	2

Suppose the message to C gets lost.

C tells B, $(X, 2)$

C uses split horizon, so it does not give distance about X to A.

B put $(X, C, 3)$ in its table

Again count-to-infinity problem!

B sends (X, ∞) to C

B tells A: $(X, 3)$

A will update $(X, B, 4)$

A will tell C: $(X, 4)$

RIP: Routing Implementation Protocol — D.V. based

Cost of all links are 1

distance vector

max. cost allowed to a destination is 16 (16 = ∞)

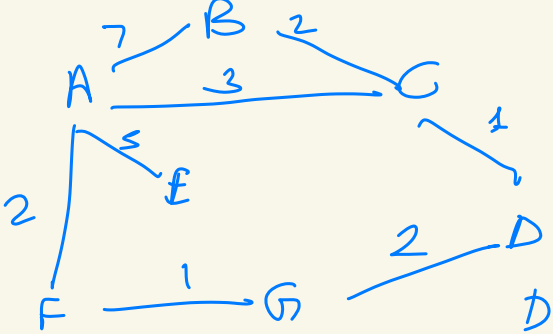
Advantages of distance vector:

Simple and easy to implement.

Disadvantages — Cost-to-infinity & routing loops

— convergence of routing tables takes some time.

Link state Routing:

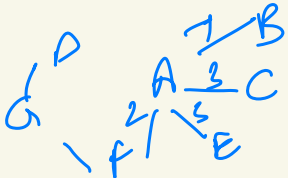


Each node sends to all others (broadcast) information about cost to immediate neighbours

D sends D 1 C, D 2 G to All.

Dijkstra Algorithm:

Each nodes finds shortest path tree to all other nodes in network.



$T = \{A\}$

choose closest neighbour

$T = \{A, F\}$ → lowest distance from A

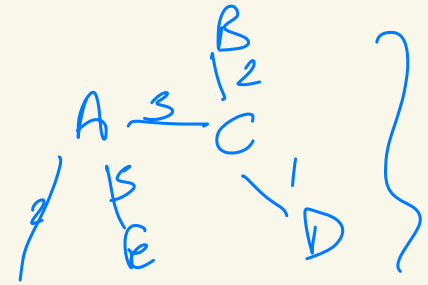
$T = \{A, F, G\}$

$T = \{A, F, G, C\}$

$T = \{A, F, G, C, D\}$

$T = \{\ldots, E\}$

$T = \{\ldots, B\}$ Finally



our tree

A's routing table:

Dest	Next	Cost
B	C	5
C	C	3
D	C	4
E	E	5
F	F	2
G	F	3

On link failure

A ✗ F (2)

A and F broadcast to All, that their link is failed.

All are going to rerun Dijkstra's algo.

Advantages – No routing loops, count-to-infinity problems, convergence to routing tables is fast.

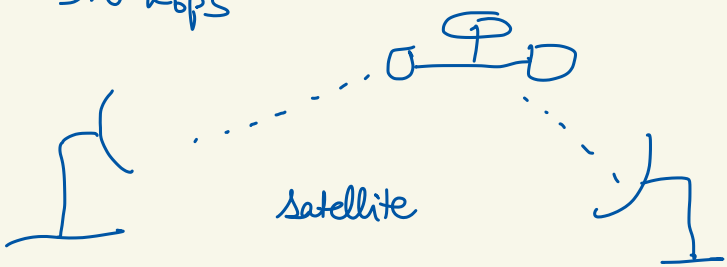
Disadvantages – Algo is more complex than D.V.

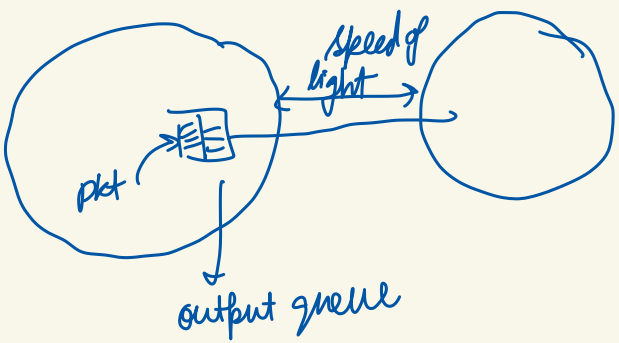
Q) what to use as cost?

How to choose link weights?

ARPANET = 56 kbps
9.6 kbps

terrestrial
satellite link

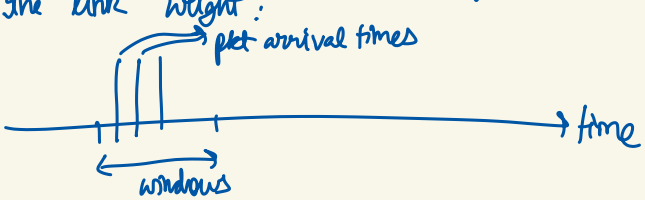




Idea-1
Use latency

latency of pkt on this link = queuing delay
+ speed of light delay
+ transmission delay (P/C)

Take time window, avg latency of all pkts in window is the link weight!

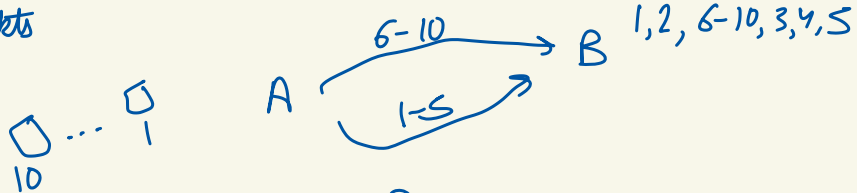


Issues :

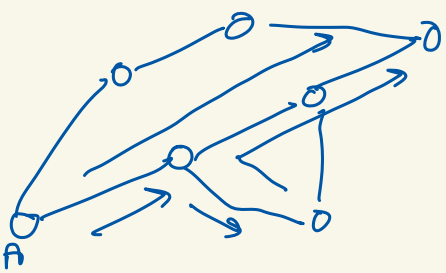
1. Under heavy load, routing path oscillates

Queuing delays go up → increased link weights → new shortest paths → can paths used heavily

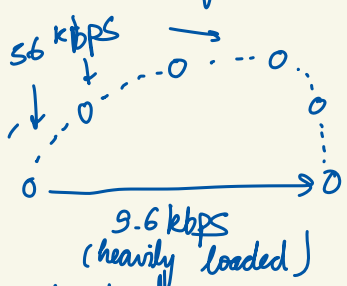
- end-to-end latency (eg: A to B) keeps varying
  ↳ may affect application layer performance.
- reordering of pkts



- routing loops possible

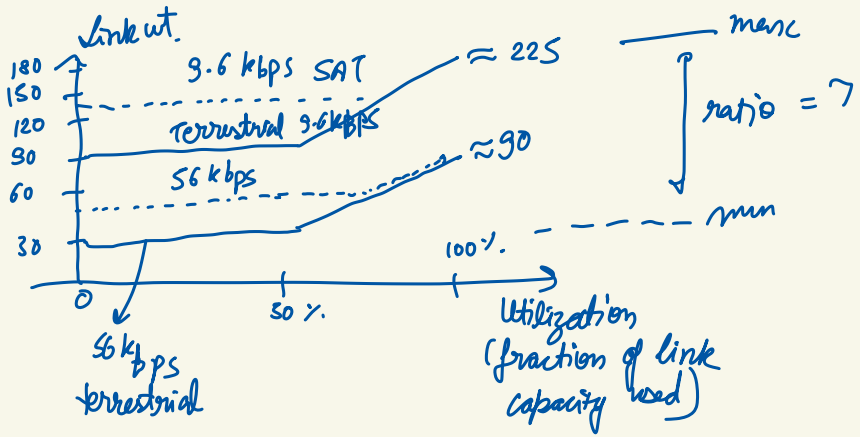


2. Range of link weights was large due to which some links were penalized (due to very high weights) too much.



Found that 126 links of 56 kbps had same wt as single 9.6 kbps link

3. Satellite links penalized too much



uts change infrequently

Today : OSPF : ut of link max. $\left(\frac{10^9}{\text{link speed (bps)}} \cdot 1\right)$

NOC : network operations centers (AT&T)

IP Addressing

MAC address (layer 2)

IP address (layer 3)

IPv4 : 32 bits → 2^{32}

IPv6 → 128 bits

NAT Network Addr Translation (to reuse IP addresses)

8 bits | 8 bits | 8 bits | 8 bits

8 bits → write as decimal

Ex: 255. 255. 255. 255 → all 1s reserved for broadcast

10. *. *. * → anything

{ 192. * . * . *

private IP

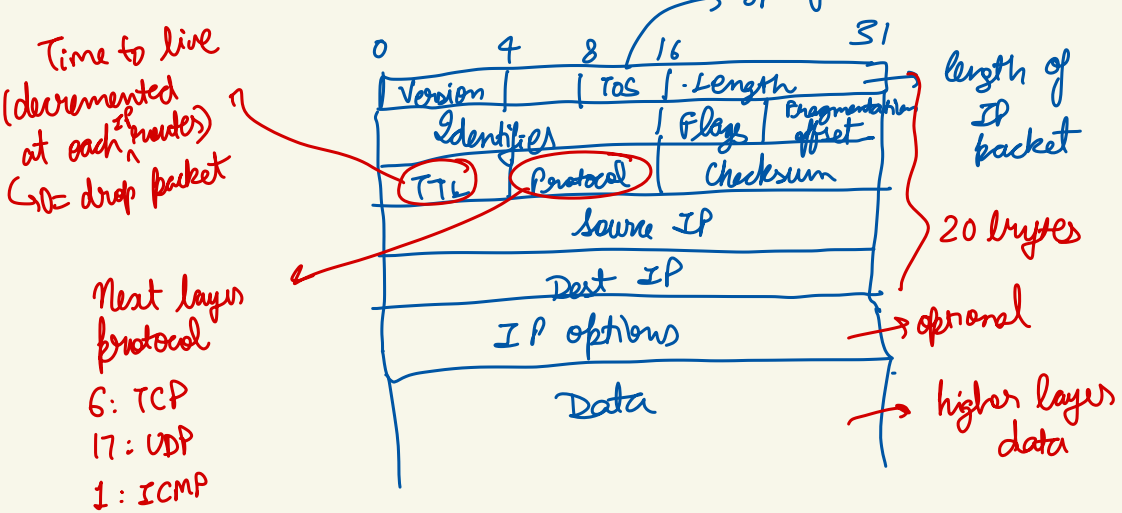
public IP ⟶ should be unique in the internet

↓

only one host must use that IP address

IP header

IPv4 header



Routing Table

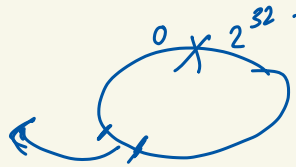
Destination	Next hop
130.52.30.*	R_2

small in number

IP prefix

IIT Bombay

IPv4 ≃ 2^{32}



slice

a.b.c.d

↓

octet

730.52.30.* → vary from machine to machine

assign to IIT

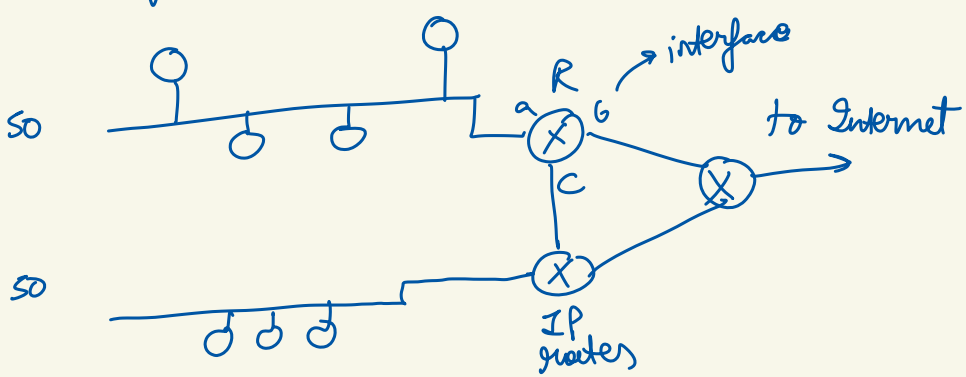
prefix of all machines

Class A : 8 bits network | 24 bits host. → 2^{24} hosts

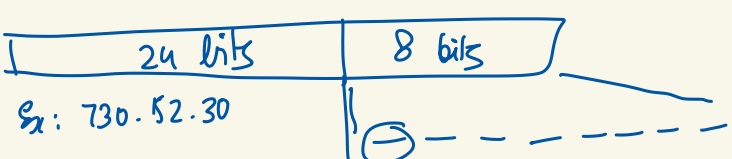
B : 16 bits network | 16 bits host

C : 24 bits network | 8 bits host

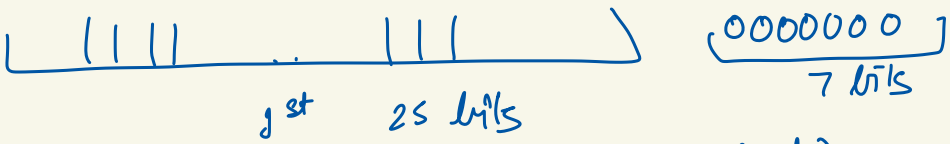
Subnetting: Given a slice of IP addresses; how to divide among LANs, setup config, internal router.



Class C:



Subnet mask : says which bits in IP address to use to decide which LAN to route to



subnet address : S_1 for LAN-1 (M_1 is Mask)
S_2 for LAN-2 (M_2 is mask for LAN$_2$)

Router-R : suppose dest. IP is 'D'

Is (D and M_1) == S_1 ?

S_1 = 730.52.30.0

Yes → Do nothing.

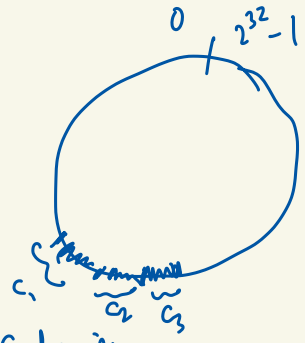
No → Is (D and M_2) == S_2 ?

(S_2 = 730.52.30.128)

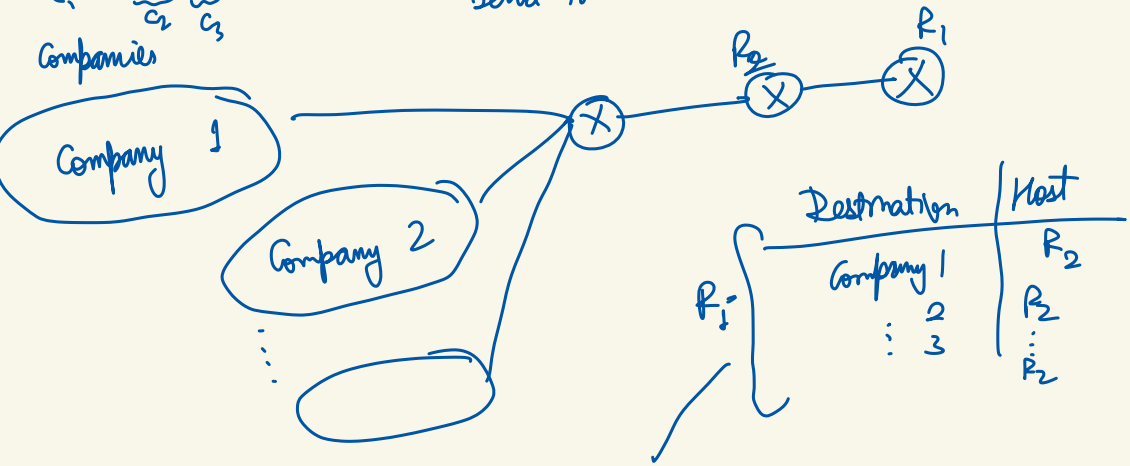
1 is the leading bit

yes → FWD to interface C

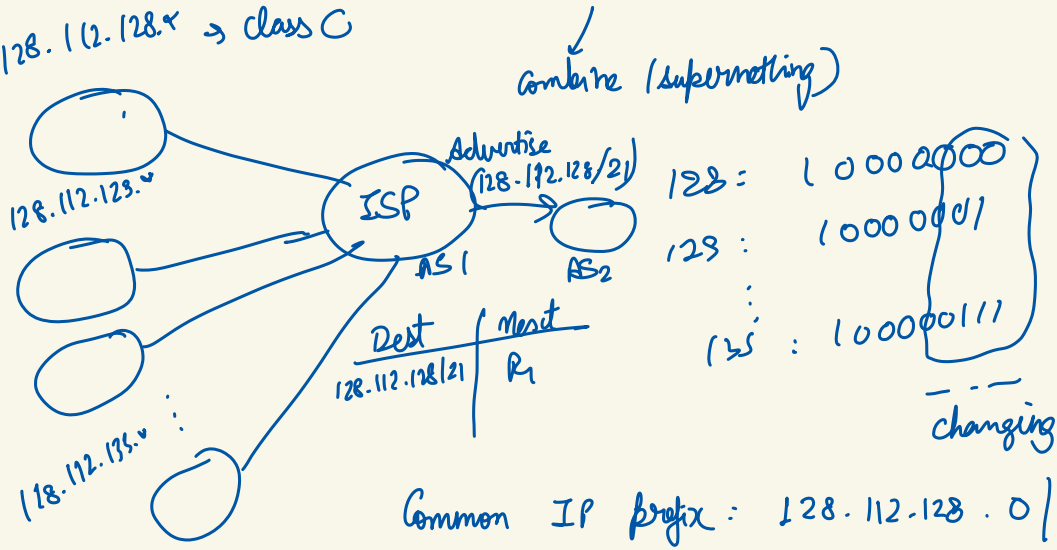
NO → Send to 'b'



Companies



128.112.128.* → class C



Common IP prefix : 128.112.128.0/21

a.b.c.d/N

Consider N leading bits to get the IP prefix.

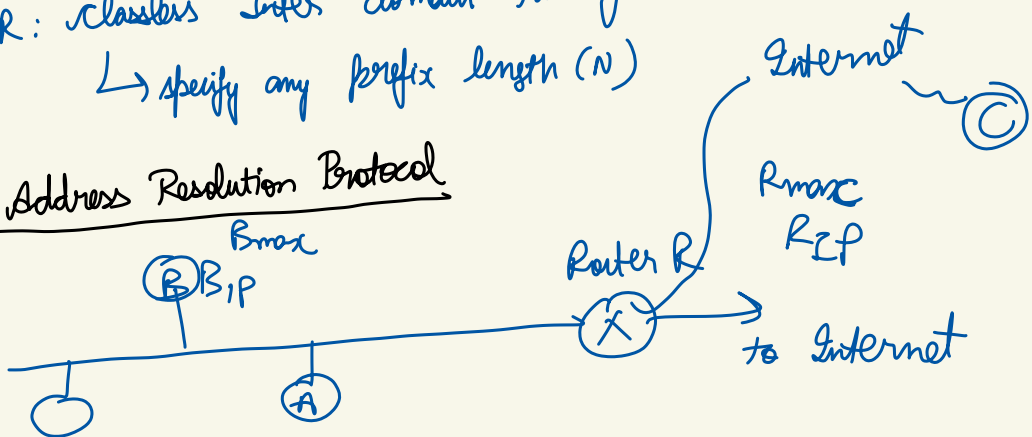
If given dest. IP address D

If first N bits of D match with the first N bits of a.b.c.d then D belongs to that prefix

CIDR: classless Inter domain routing

↳ specify any prefix length (N)

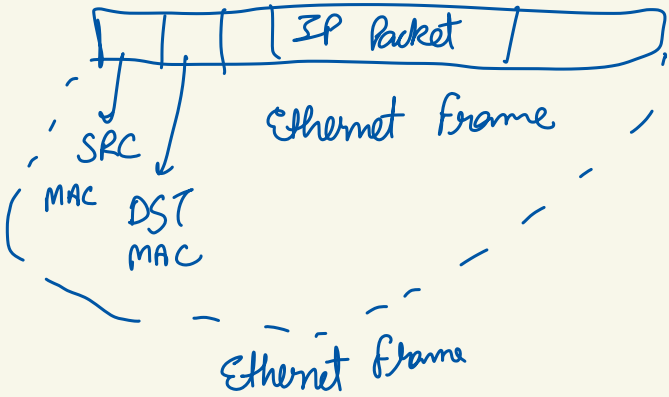
ARP: Address Resolution Protocol

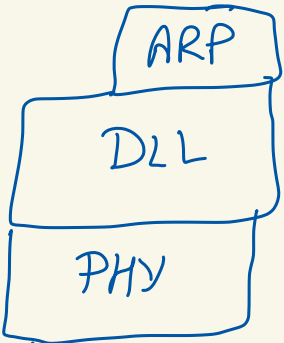


A wants to send an IP packet to B

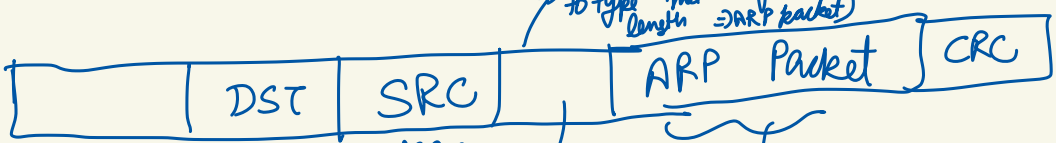
A knows B's IP; A does not know Bmac

ARP to the rescue





Ethernet Frame :



all 1s → broadcast (MAC)

ARP REQ (Request) · Sender MAC (A_{MAC}) ; Sender IP (A_{IP})

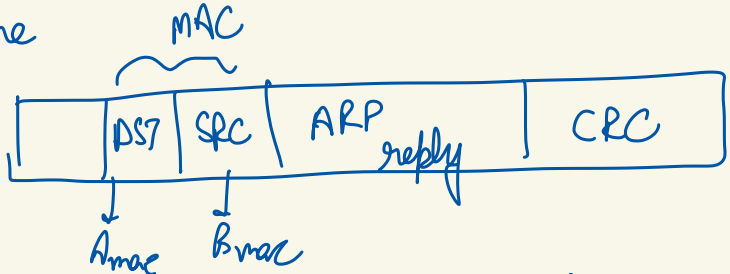
Target MAC ; Target IP (B_{IP})

↳ All zeros

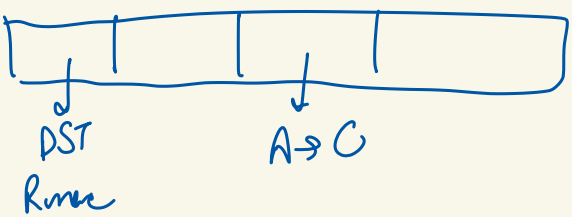
ARP Reply : (from B → A) sender MAC (B_{MAC}) ; sender IP (B_{IP})

Target MAC ; Target IP
(A_{MAC}) (A_{IP})

Unicast frame



Information (B_{mac}) stored in ARP cache at A.
with a timeout (orders of minutes)



(1) How does A know if DST IP belongs own network or not?

(2) If DST IP is not in own network, how to known R_{IP}, R_{MAC}.

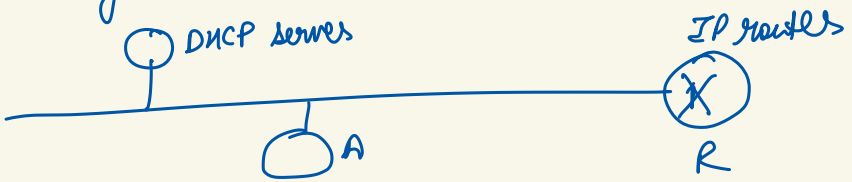
(1) $A_{IP} = a_1.a_2.a_3.a_4$
subnet MASK: Ex 255.255.255.0
all 1s

If DST-IP AND MASK = A_{IP} AND MASK
Network address
then DST-IP is in my network.

(2) Suppose know R_{IP}, how to find R_{MAC}

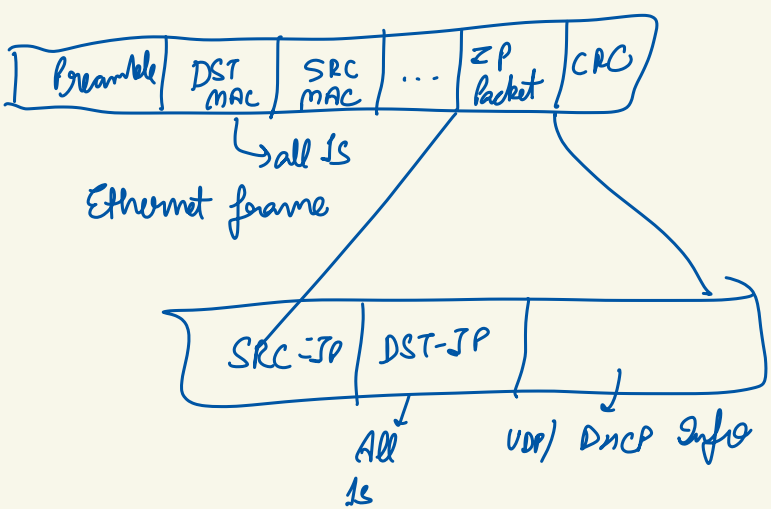
Use ARP

DHCP: Dynamic Host Configuration Portal

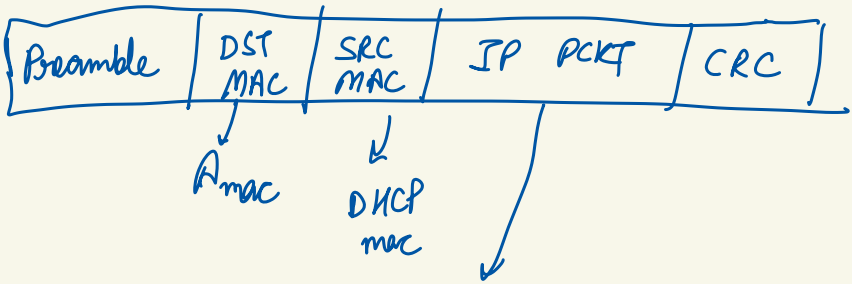


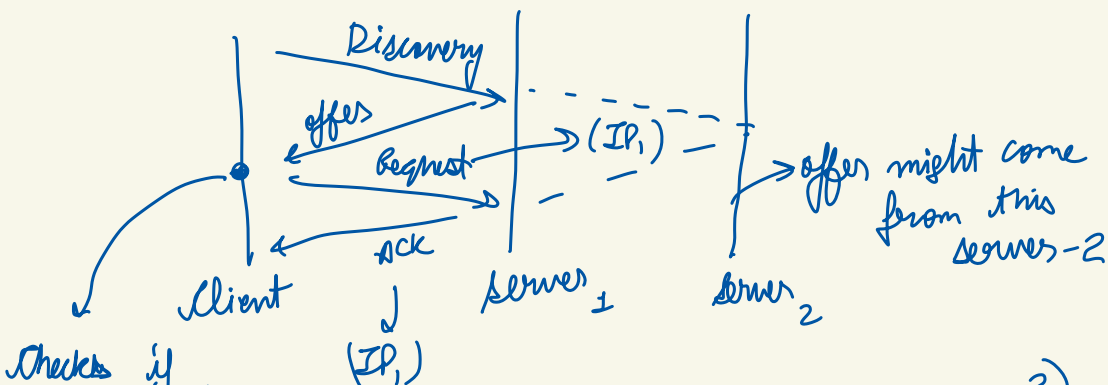
DHCP
↓
UDP ⋯ cross-layer → Port #s : 68 → DHCP server
↓ 67 → DHCP client
IP
↓
DLL → PHY
→ Broadcast
→ Broadcast

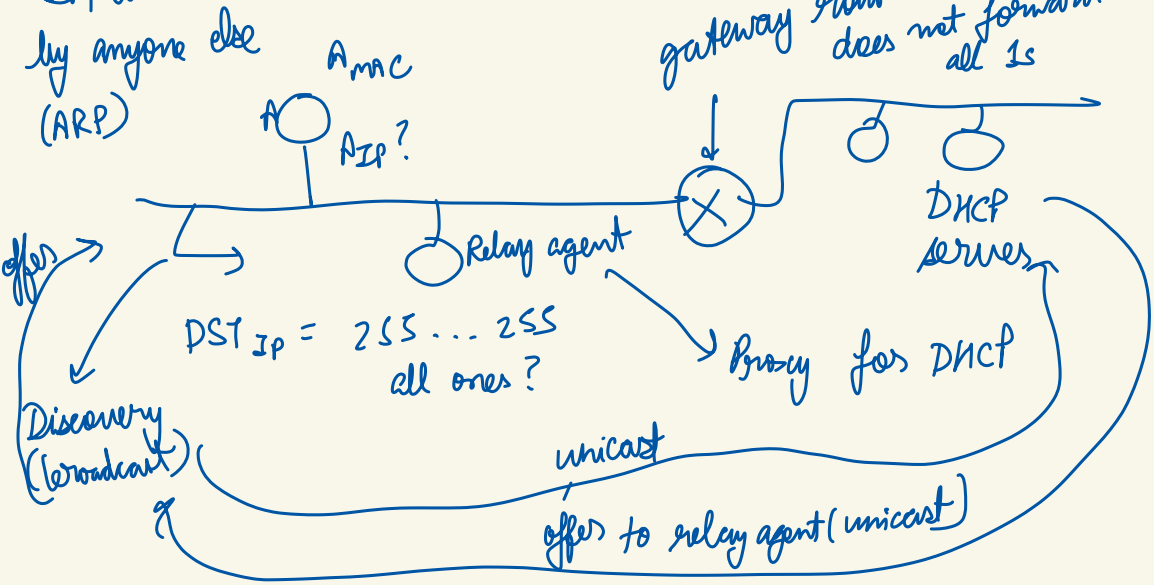
'A' sends out DHCP 'discovery' packet



DHCP server replies (offer) → Contains a potential IP address for A.


Preamble
DST MAC
SRC MAC
IP PCKT
CRC
A mac
DHCP mac
SRC IP: DHCP IP
DST IP: All ones (broadcast)
we are yet to configure A IP


Discovery
offer
Request
(IP1)
offer might come from this server-2
ack
Client
Server 1
Server 2
(IP1)
Checks if IP1 is used by anyone else (ARP)

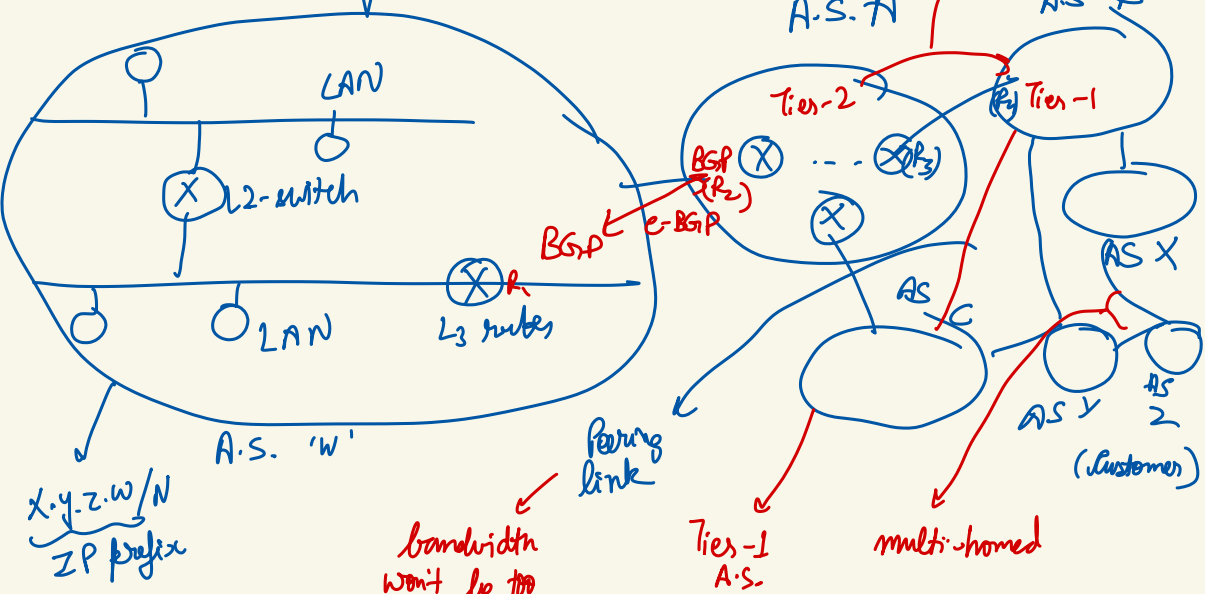

A mac
A IP?
gateway routes (layer 3) does not forward all 1s
DHCP server
offer
Relay agent
DST IP = 255 ... 255 all ones?
Proxy for DHCP
Discovery (broadcast)
unicast
offer to relay agent (unicast)

BGP : Border Gateway Protocol

SLA = service level agreement

take internet service from Tier-1 ISP

Inter domain routing : between ASes



bandwidth won't be too high

usually under-provisioned because both don't get paid for delivering traffic on that link

Tier-1 A.S.

usually widely connected across country/globe, not customers for any other A.S.

multi-homed

e-BGP
$R_1 \rightarrow R_2$: x.y.z.w/N — Prefix

i-BGP
$R_3 \rightarrow R_4$: x.y.z.w/N — Prefix

e-BGP
$R_5 - R_6$: ↓

W — AS. path

A-W — Path

B-A-W

attributes

attributes

attributes

BGP advertisement

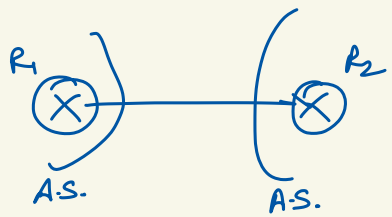
means if R_6 gives R_5 an IP pkt with DST IP

matching prefix, then it will definitely route it along AS path B-A-W

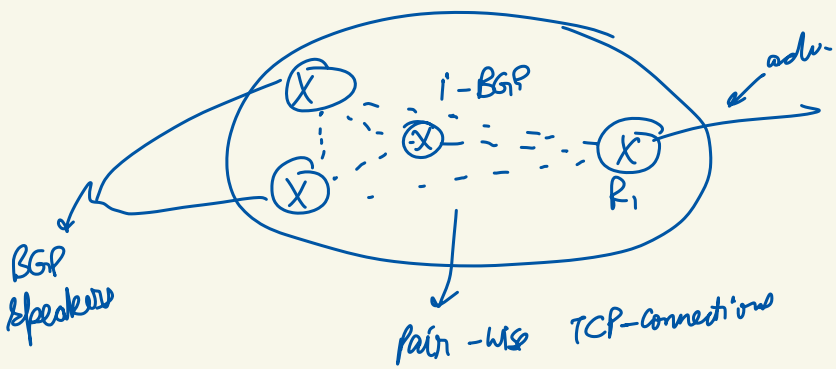
B does not have to tell C: that there is a path B-A or B-A—

e-BGP: Protocol between BGP speakers in different (neighbouring) ASes

TCP port 179.



i-BGP : Protocol between BGP speakers within same A.S.



IGP : Interior Gateway Protocol
(Intra-domain routing, LSR, DV)

Steps to decide BGP paths

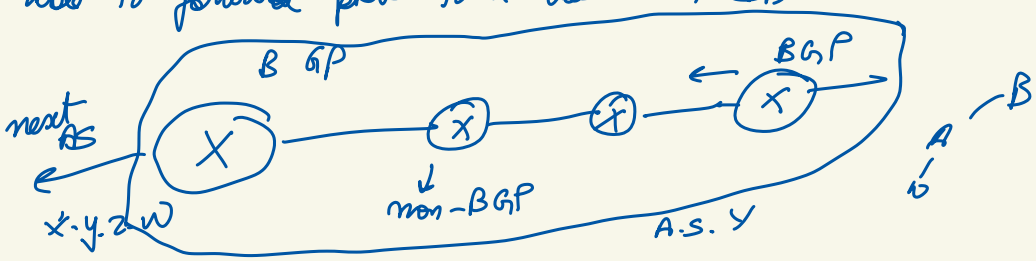
(i) e-BGP speakers learn AS-paths from neighbouring routers in other ASes

(ii) e-BGP nodes share learned information via i-BGP with other BGP speakers in own A.S.

(iii) BGP speakers select routes to various IP prefixes.

(iv) Insert chosen routes into IGP (since all routers must be

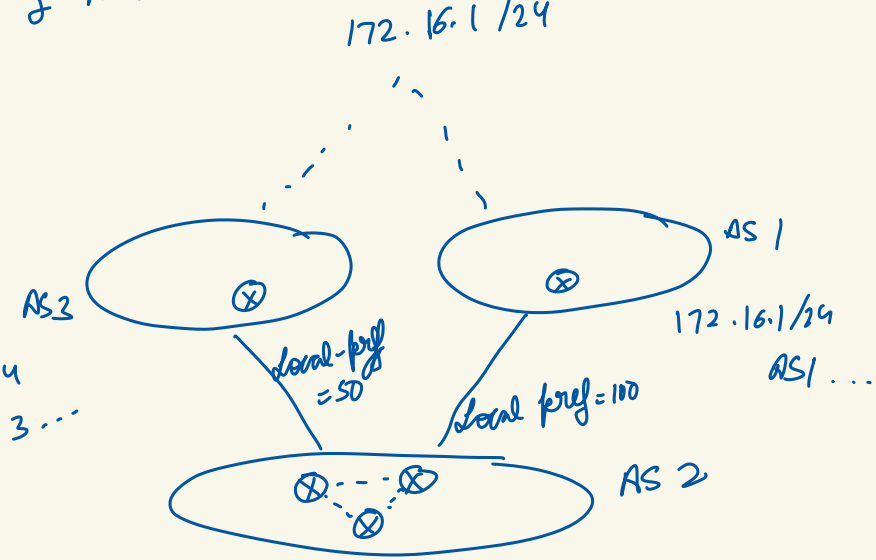
able to forward pkts to a destination IP)



(v) eBGP speakers can advertise newly created routes to neighbouring ASes

BGP attributes

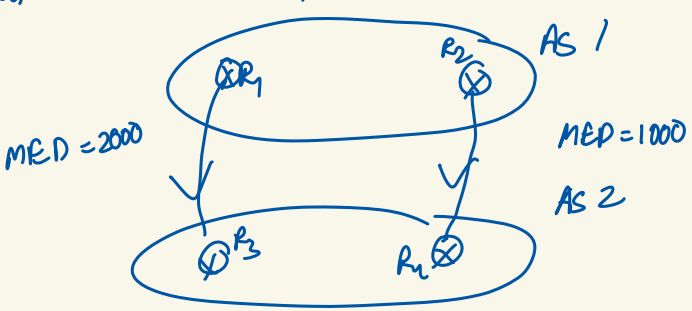
(i) Local pref



Admin of AS2 adds local pref himself

Higher local pref ⇒ more preferrable path

(ii) multi-exit discriminator (med)

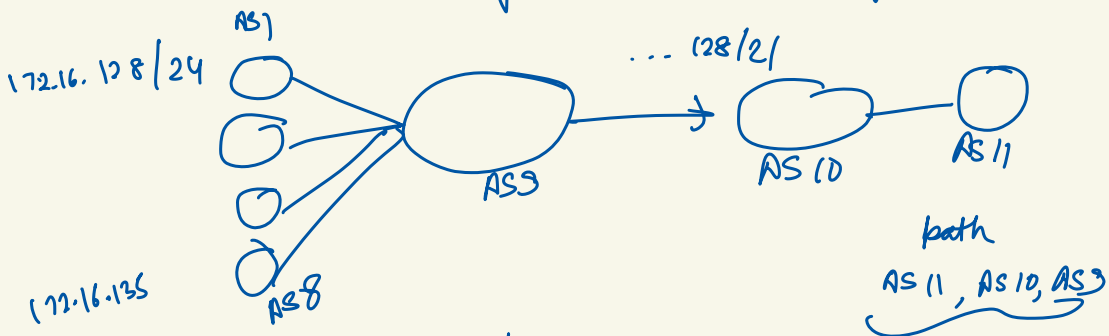


ADV: 172.16.1/24
AS1...
Path

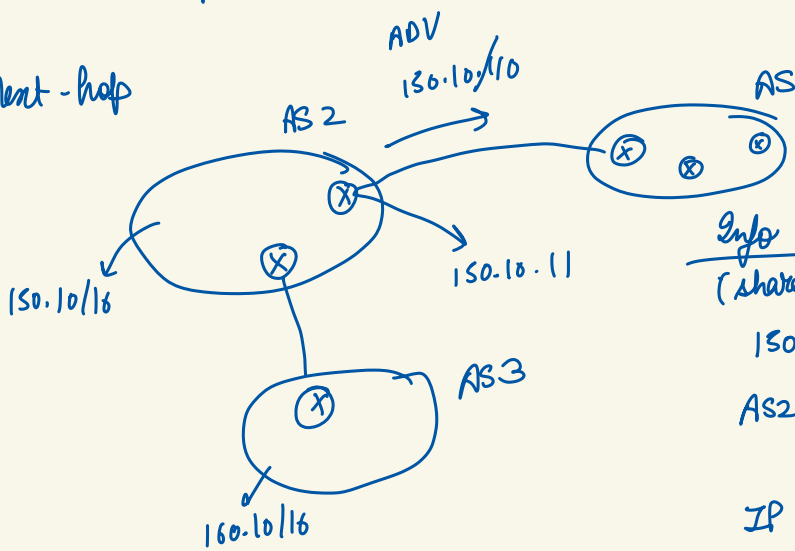
means: AS1 is telling AS2: I prefer that you send me pkts for 172.16.1/24 on link R_4-R_2 (lower MED)

(iii) AS-path: List of AS numbers all the way to the destination with that IP prefix

↳ whoever first adv that IP prefix



(iv) Next-hop



Info AS1 has (shared by i-BGP)

150.10/16

AS2 150.10.1.1 next-hop

IP address of 1st router in next A.S.

160.10/16 AS2, AS3 (path) 150.10.1.1 (next hop)

Rules to choose routes:

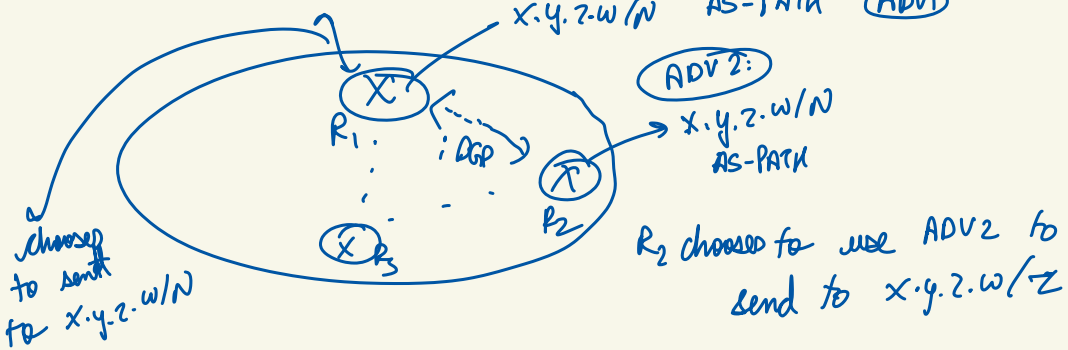
Each BGP speaker decides which AS route to use among many available for the same prefix

(a) Use route with largest local pref

(b) Choose path with shortest AS-path

(c) Choose path with lowest MED

(d) Choose path learned over e-BGP over path learned over i-BGP



(e) Choose path with lowest IGP metric to next Hop

dist (R_3, R_2) dist (R_3, R_1)

R_3 compares and chooses the smaller -1

Suppose dist (R_3, R_2) > dist (R_3, R_1)

$\Rightarrow$ R_3 chooses ADV1

Hot-potato routing

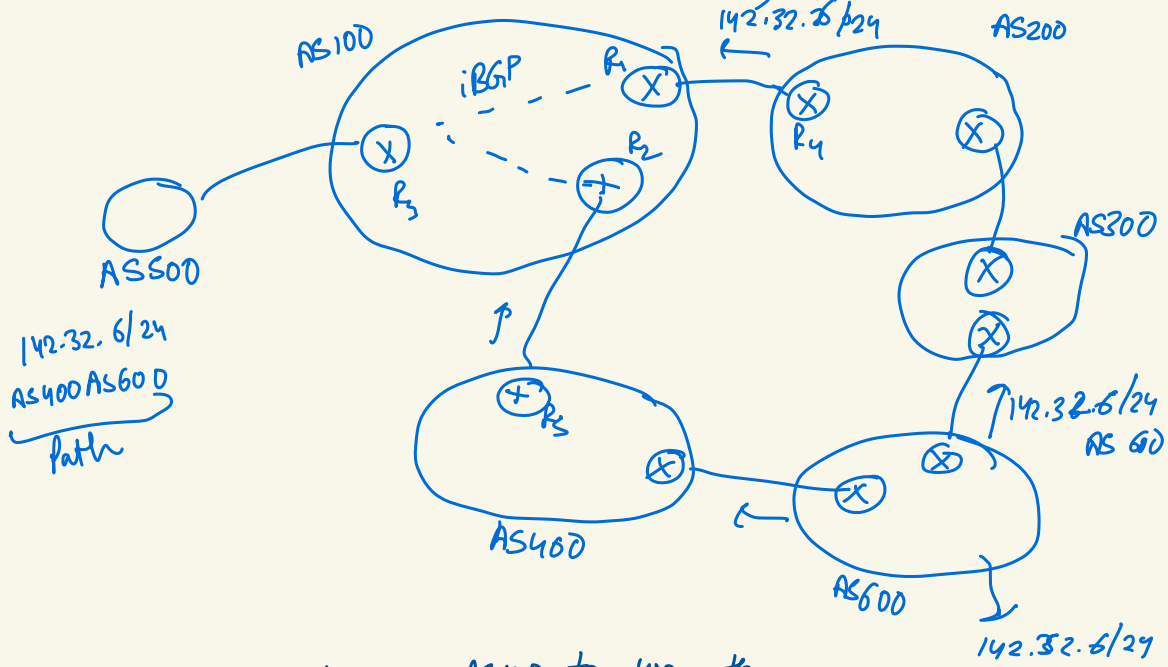
(f) Use router id (lowest router id among all BGP speakers who have sent these advertisements)

Router-id = highest IP address on route

BGP

1. Local Pref → Largest
2. As-Path → shortest
3. Med → lowest
4. eBGP learned route over iBGP learned one
5. Hot potato routing
6. Router ID (lowest)

AS200 AS300 AS600

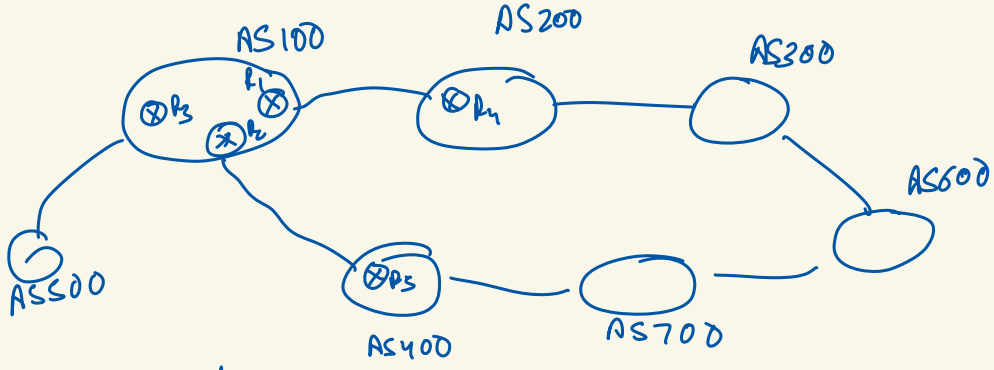


1) If want all routers in AS100 to use the top route, admin of AS100 can set local-pref highest for it.

2) Suppose we want all BGP routers to use the lower route.
AS 400 - AS 600

(i) Use local-pref

(ii) By default all will use the shortest AS-path (no need to set local-pref highest for this path).



Suppose local pref and MED are same for both paths

Path 1 : AS200 – 200-600
Path 2 : AS400 – 700-600

R_1 : learned path 1 over eBGP path 2 over iBGP
R_2 : learned Path 2 over eBGP, path 1 over iBGP
R_3 : both over iBGP

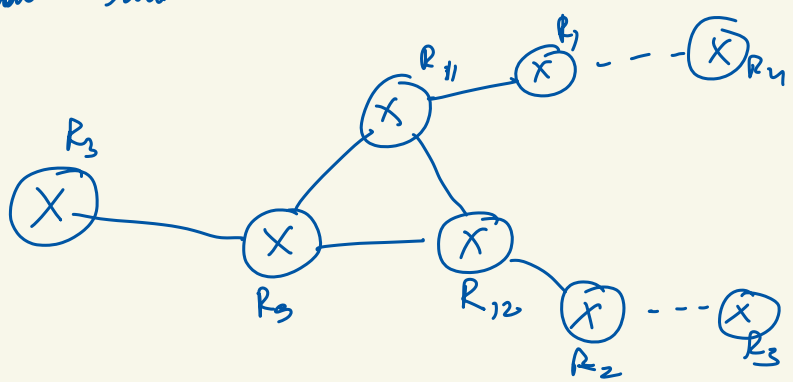
R_1 uses Path 1
R_2 uses Path 2

iGP metric

if dist (R_3, R_1) < dist (R_3, R_2) then
R_3 uses path 1 (hot potato

If equal then lowest router id.

AS100



BGP interact with IGP?

Solutions:

(1) Encapsulation

Suppose R_3, R_{11}, R_{12} have no BGP information

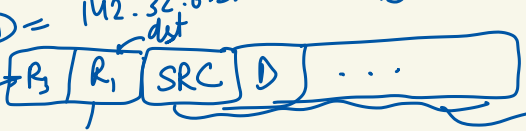
Ex: in R_3's table

142.32.6/24 → ??] external prefix not in the table

R_3 encapsulates

DST	next
R_1	R_{11}
R_2	R_{12}
R_3	R_3
R_{11}	R_{11}
R_{12}	R_{12}

IP address are → D = 142.32.6.21



another IP header

R_1 receives this packet, strips off outer IP layer, finds the internal pkt to R_4

R_1's table

encapsulated

DST	Next
R_2	
R_3	
⋮	
142.326/24	R_4

(2) Pervasive BGP

All routers are BGP speakers in AS100

* Suppose there is a unique exit for 142.32.6/24

BGP table

Prefix	Gateway/Exit
142.32.6/24	R_1

true for all

IGP

DST IP Addr	Next
R_2	
R_3	
R_1	⋮
R_{11}	R_{11}
R_{12}	

On receiving

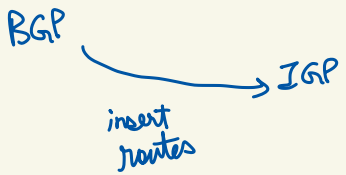
SRC	D	...

from R_3

recursive lookup.

forward to R_{11}

(3) Tagged IGP



Internal routers may not be BGP speakers

But IGP allows addition of tags

R_1 can insert into its own IGP

142.32.6 /24 — prefix

R_1 — tag → gateway router

Propagated to all routers using IGP (say LSP)

At R_n IGP Table

DST	Next	Tag	Cost
R_1	R_{11}		5
R_2	R_{12}		12
⋮			
142.32.6/24		R_1	
142.32.6/24		R_2	

your R_n

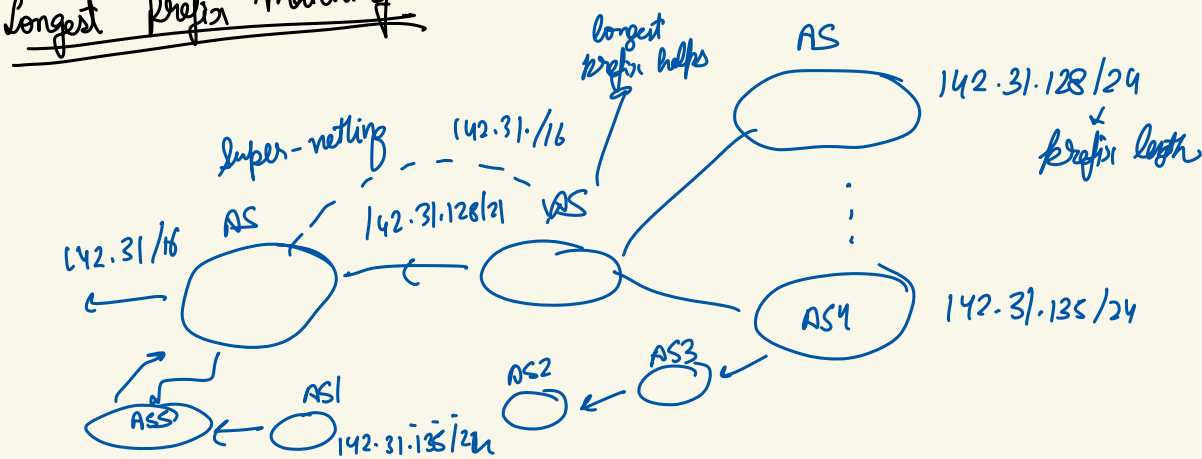
142.32.6/24 announced by R_2

$R_3 \to R_5$: SRC | D

Lookup, find matches and forwards to closest "tag"

→ hot potato routing

Longest Prefix Matching



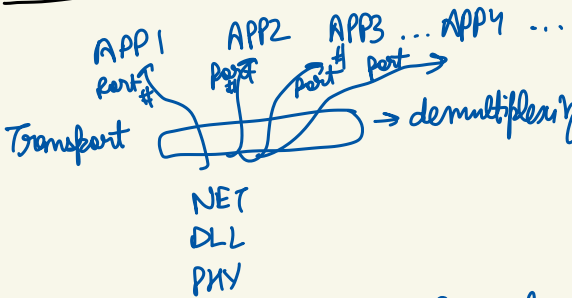
ASS gets a packet with dst IP = 142.31.135.20

Prefix	Next hop
142.31 /16	-----
142.31.135 /24	....

Rule = choose the longest prefix to route

CAM: Content addressable memory for easy lookup.

Layer-4 Transport Layer

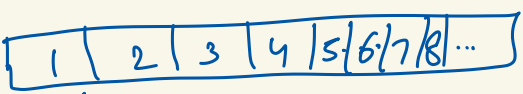


→ demultiplexing → basic task

web = port # 80 (eg)

UDP: User Datagram Protocol

File Transfer:

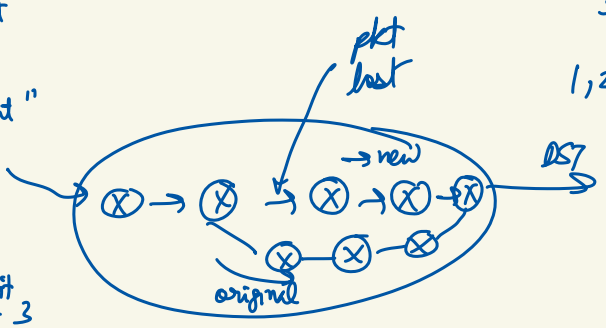


1500B pkt

↓

"segment"

SRC



retransmit segment 3

3 lost

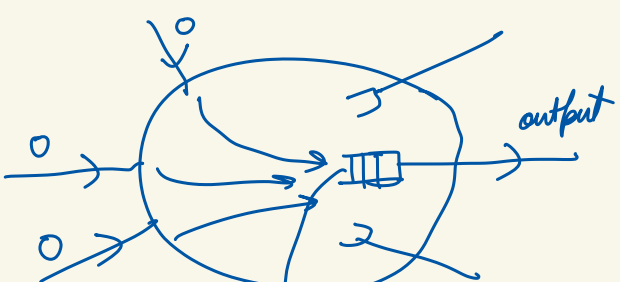
1, 2, 4, 5, 7, 8, 6,

reordering due to ≠ path change

Transport (TCP): Reorder segments

Reliable data transfer: retransmitting lost segments → Congestion and flow control

↳ infer packet loss

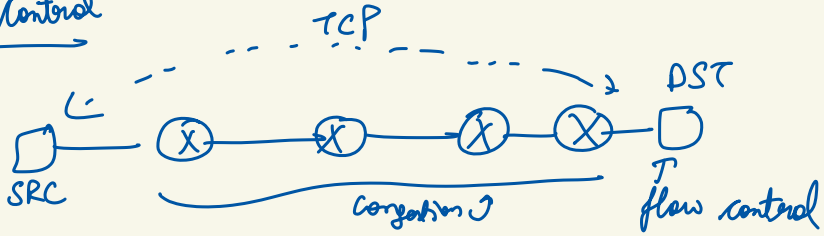


Congestion

↳ Queue full: packets dropped

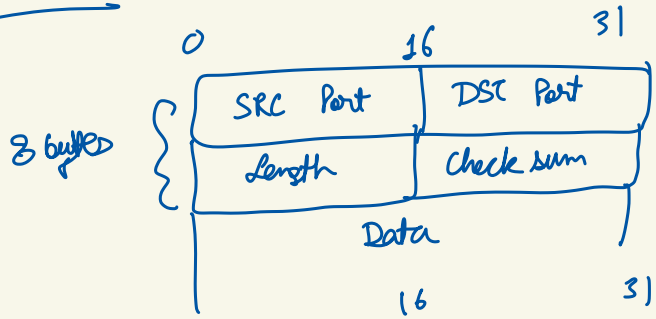
TCP infers congestion does congestion control reduce segment sending rate

Flow Control

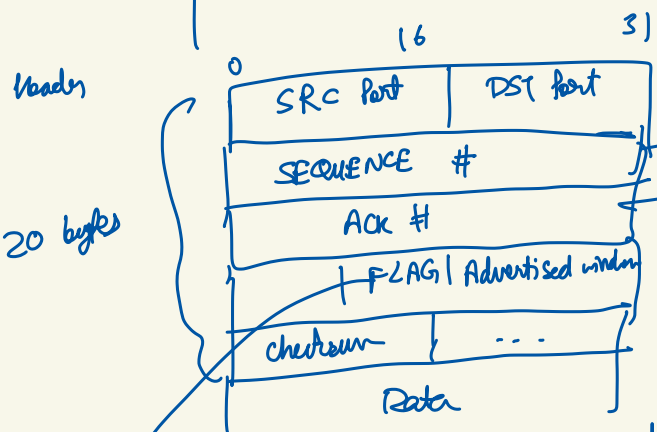


Van Jacobson — 1980s

UDP Header



TCP Header



byte # of 1st byte in segment

says what byte # expecting next (got everything till the previous byte)

ack is present in segment

SYN | FIN | RESET | PUSH | URG | ACK

SYN PKT (start)

FIN PKT (end of connection)

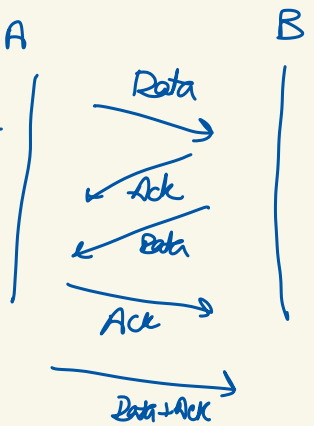
reset connection

Protocol field at IP layer → 6 TCP, 17 UDP

TCP

SEQ # ACK #

Connection Establishment

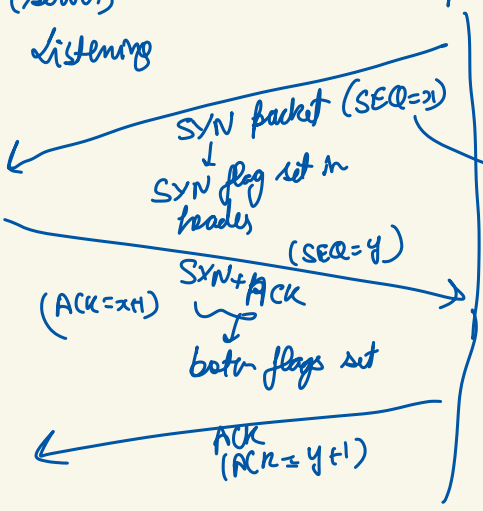


we can have data and ack in the same packet too

Passive Participant (Server)

Listening

Active open (client)



starting seq #

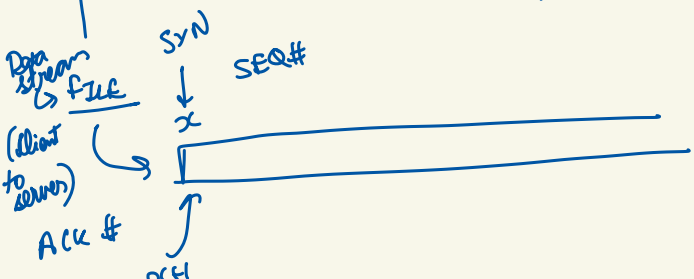
SYN = synchronization

3-way handshake

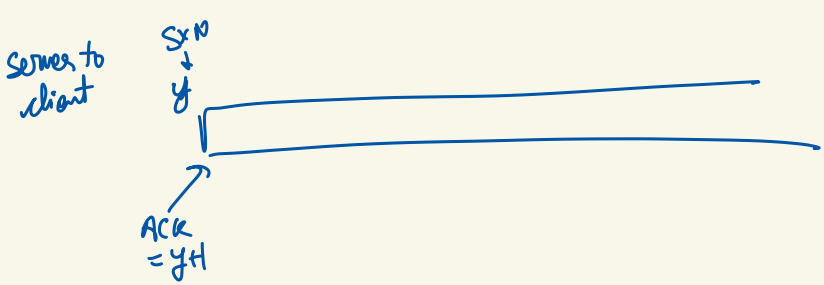
SYN, FIN segments don't have data

But considered to communicate 1 byte.

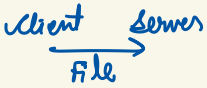
x → randomly selected

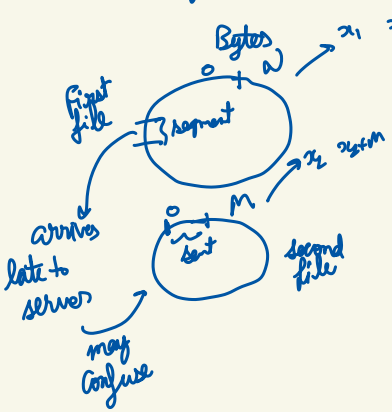


y got everything from start till one less than the ACK#

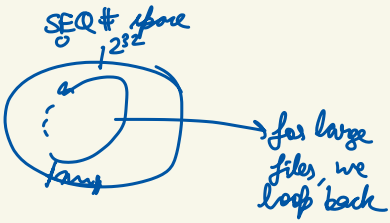


Ex: Transmitting Files

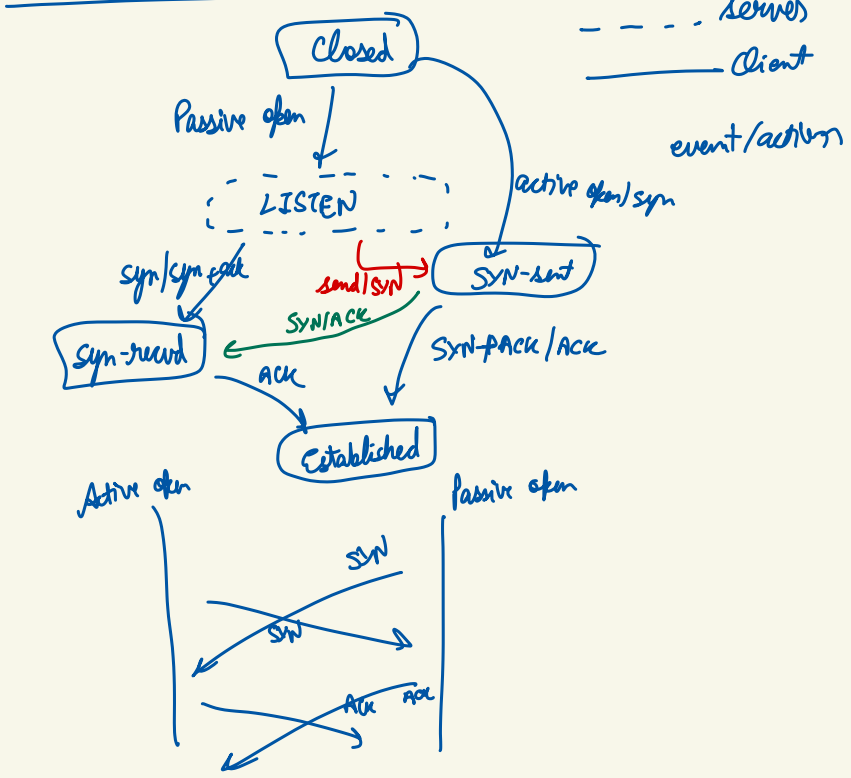




Assume port # are same

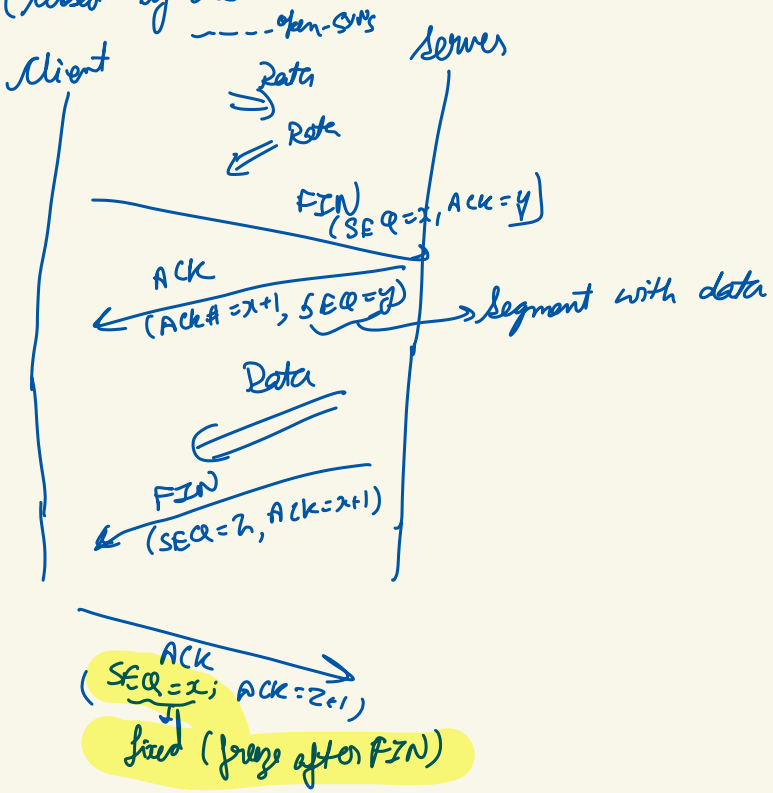


State Diagram

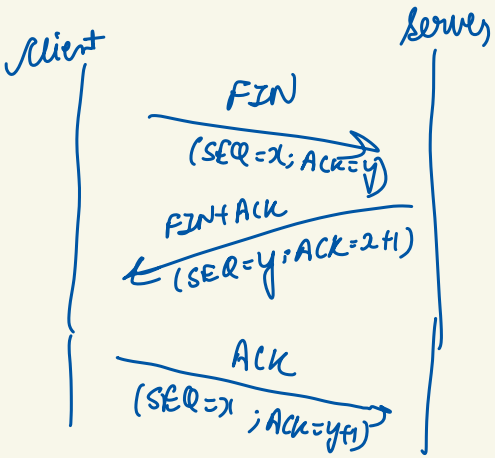


Connection Termination : (Use FIN)

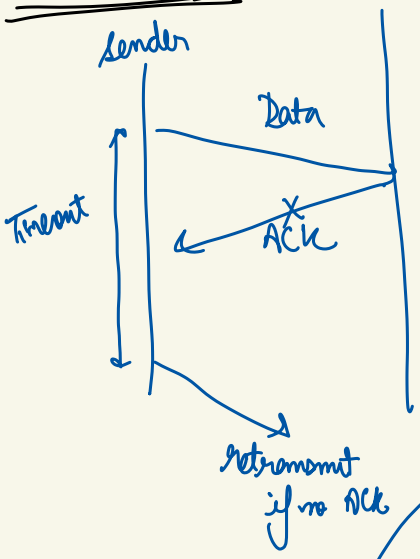
(1) Half-close (closed by one end at a time)

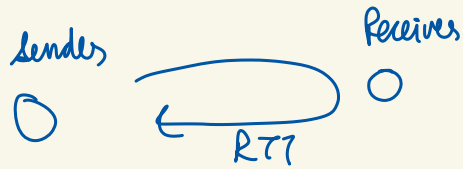


(2) 3-way handshake termination (both closed at the same time)



TCP Timeout



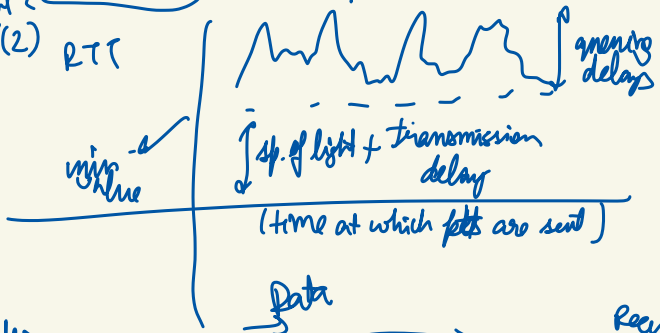


Issue:

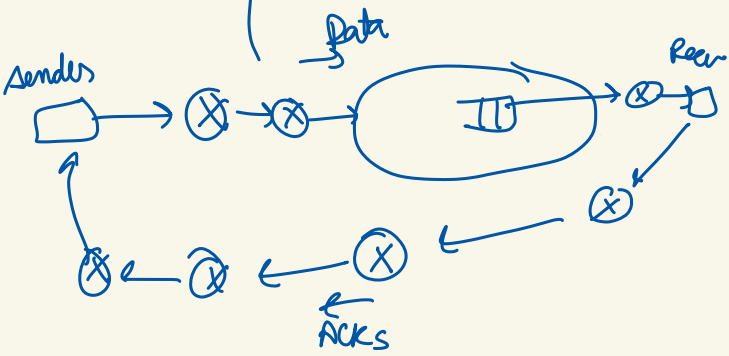
(1) RTT varies by orders of magnitude across network paths

different

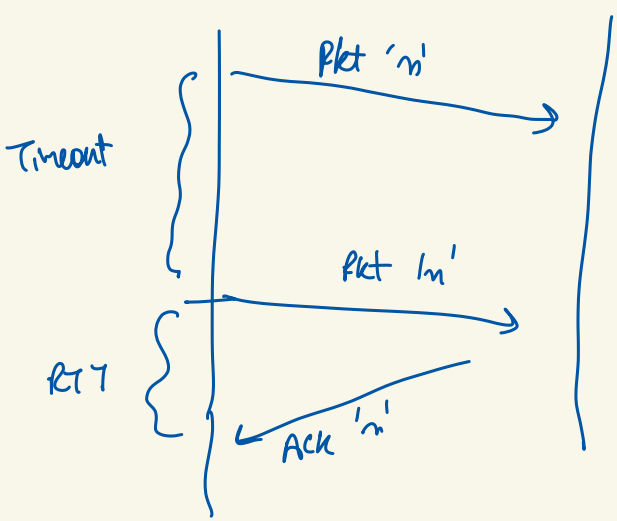
(2) RTT



same network path



3. Packet Loss



Old Algorithm

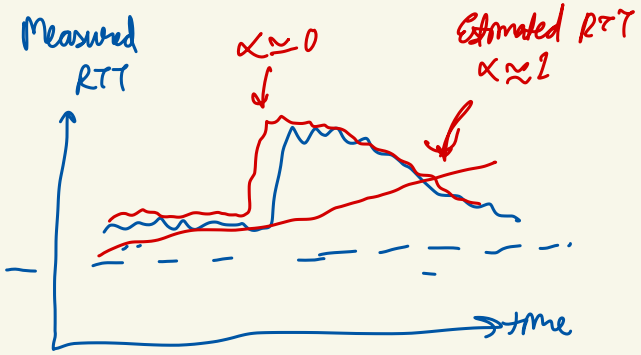
Sample RTT = most recent RTT measurement

Estimated RTT

= α Estimated RTT + $(1-\alpha)$ Sample RTT

- like an average
- moving average

Timeout = 2 × Estim. RTT



$\alpha \in (0,1)$

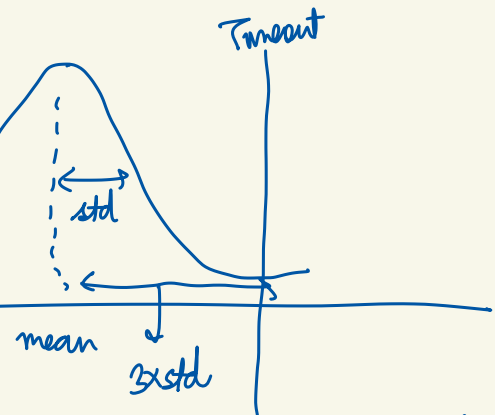
$\alpha \simeq 1 \Rightarrow$ less importance to the current measurement

$\alpha \simeq 0 \Rightarrow$ lot of importance to recent measurement

- not good for spikes

New Algorithm

Gaussian Distribution:



Idea: set Timeout = mean + m×std

measure: Estim RTT as before

Diff = Sample RTT − Estim. RTT

- assume mean

suppose $x_1, x_2, \ldots, x_n$ are some values from random variable X. (i.i.d)

Estim mean $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$

Mean. std. $= \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2}$

Mean deviation = $\frac{1}{N}\sum_{i=1}^{N}|x_i - \bar{x}|$

$\beta = \frac{1}{4}$

Dev = Dev $(1-\beta) + \beta|\text{Diff}|$

Deviation

most recent deviation value

Timeout = $\mu \times$ Estimated RTT $+ \phi \times$ Dev

$\mu = 1$

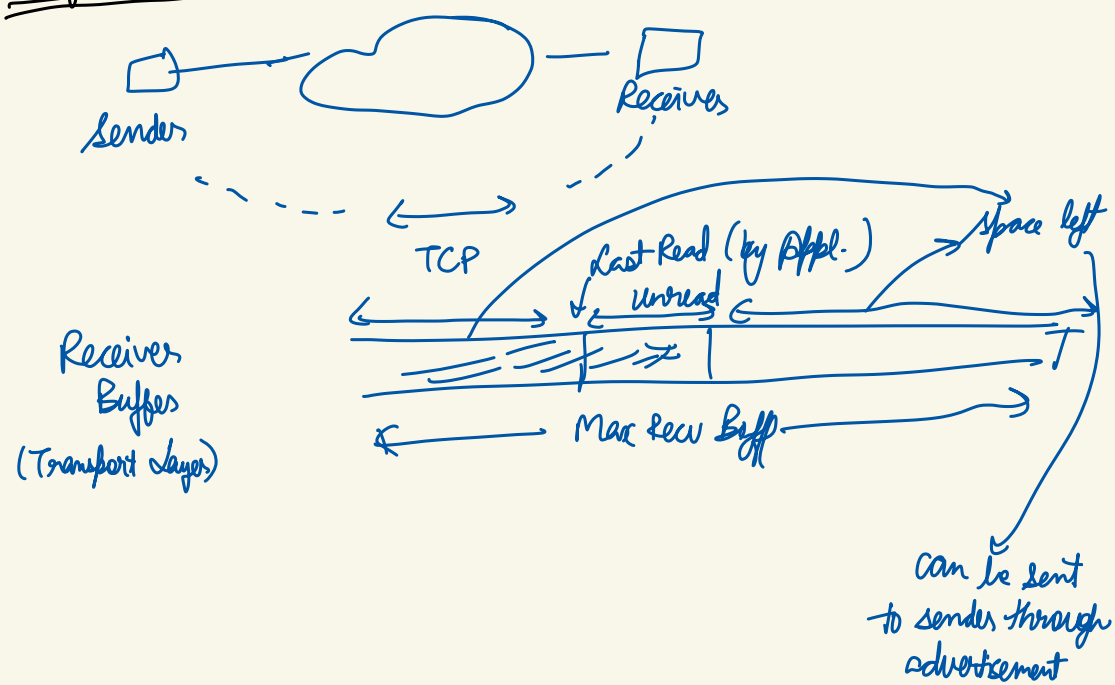
$\phi = 4$

Recall Estimated RTT = α Estimated + $(1-\alpha)$ Sample RTT

$\alpha = \frac{7}{8}$

Sol^n = Do not use RTT measurement for retransmitted pkt.

Congestion Control



Flow-control :⇒ Deals with congestion at receiver

Adv. window (field in TCP header)

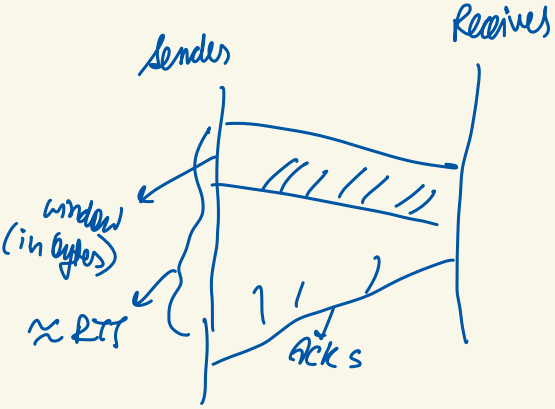
⇒ space left in the Recv Buffer

Sender has a "window"

window = max amount of data (in bytes) which is outstanding (sent into network but not ACKed)

Suppose RTT is const.

Data rate = $\frac{\text{Window}}{\text{RTT}}$



Want window < Adv. window

Ensured to take care of Flow control

How to deal with congestion at routers ?

Calculate Cong. window

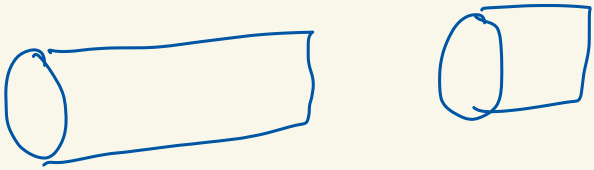
window = min (Adv. window, Cong. window)

window = Datarate × RTT

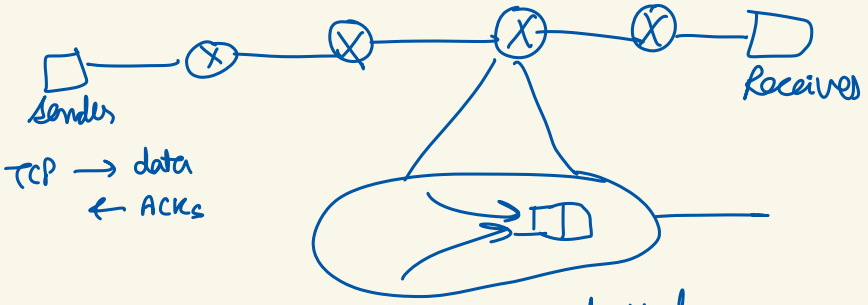
Delay - Bandwidth product

Pipes

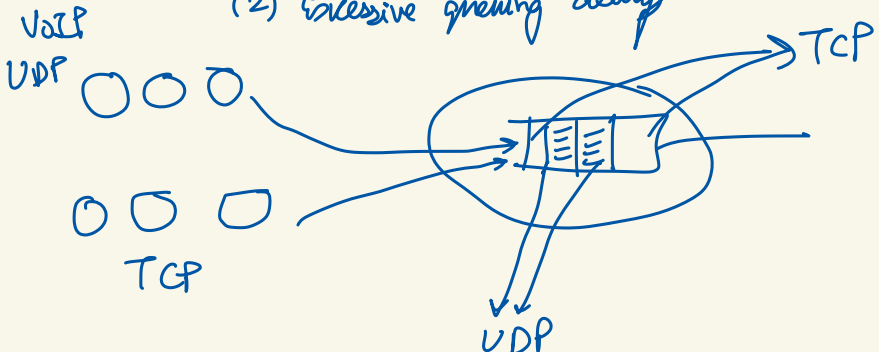
volume → analogous to window size.



TCP Congestion Control



Congestion Problem: (1) Packets get dropped
(2) Excessive queuing delay



Window = min (Adv. window, Cong. window)

↓ Flow control

↓ Congestion control

Detect packet loss

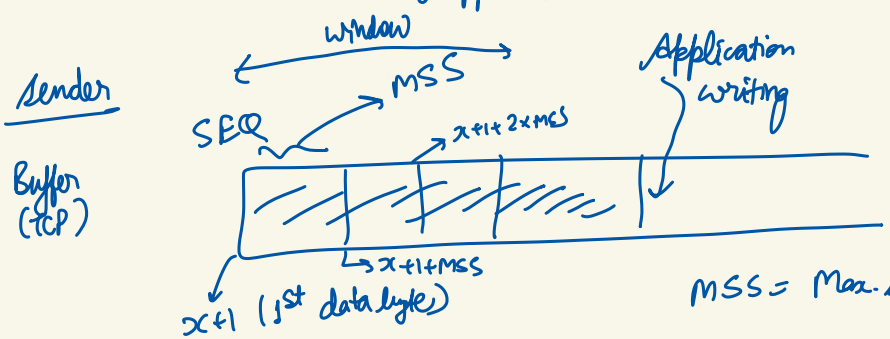
One idea: Timeout

Second idea: Use ACK feedback to infer losses

Pkt # 1, 2, 3, 4, 5, 6 ...

ACKs ↓ ↓ ↓ ↓ ↓ ↓

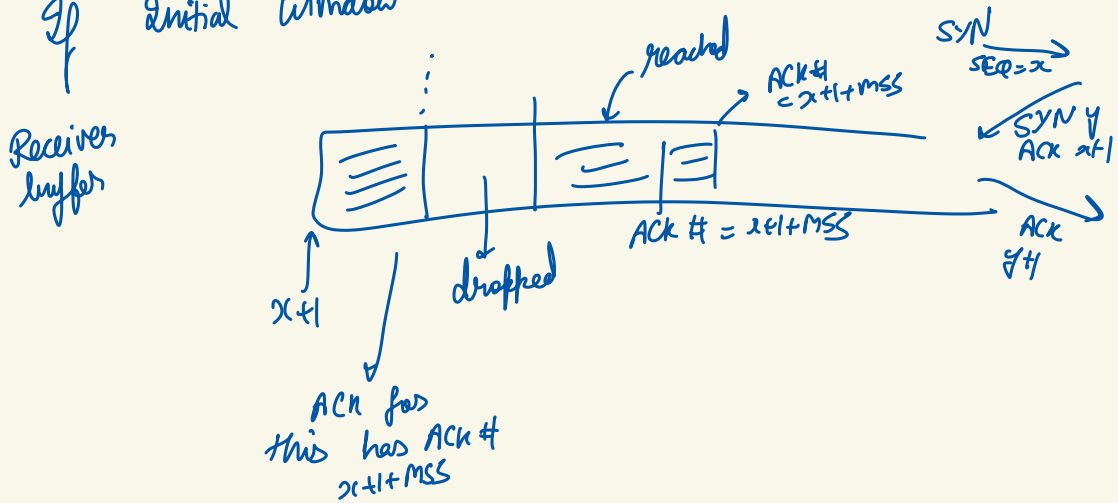
dropped ??



MSS = Max. segment size

x → SYN SEQ

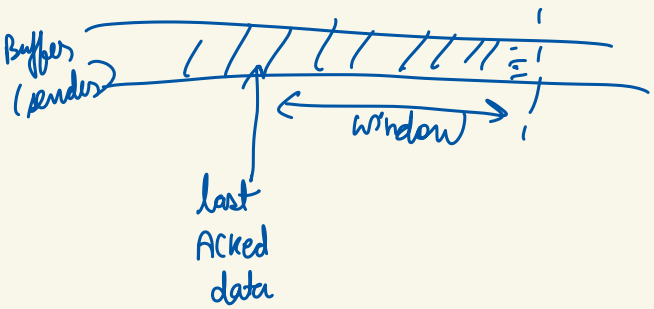
If Initial window = 3 MSS, we send out 3 segments



Cumulative ACK → Everything from start till byte (z-1) has been received

ACK#z (sent)

Duplicate ACKs → ACK # of some previous ACK sent out
↳ Each dup ACK says one more segment received.



Principles of Congestion Control

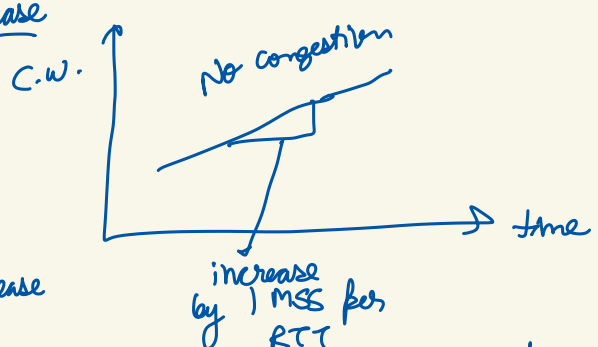
1. If no congestion, increase CW (congestion window) conservatively.

$$\text{Datarate} \approx \frac{\text{window}}{RTT}$$

[window = min (Adv. window, CW)] ← congestion window

For now assume window = CW

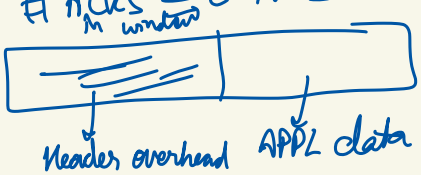
Additive increase

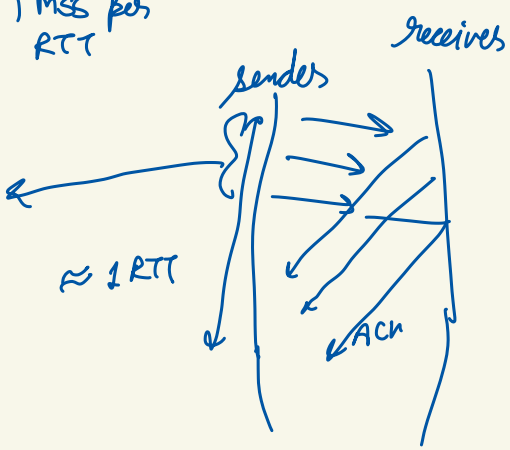


TCP uses "self-clocking": every time an ACK arrives, we increase window.

$$\#\text{segments in window} = \frac{CW}{MSS}$$

$$\#\text{ACKs in window} \rightarrow CW/MSS$$





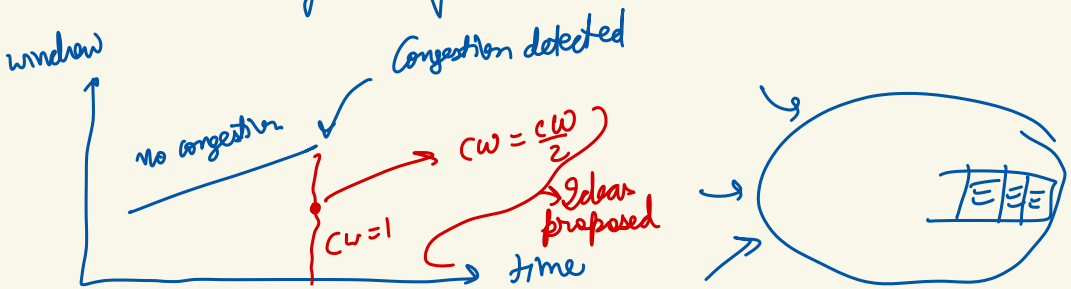
Per ACKs window increase $= \frac{MSS}{\#ACKs} = \frac{MSS}{CW/MSS} = \frac{(MSS)^2}{CW}$

Additive increase

On receiving an ACK (Assuming no congestion)

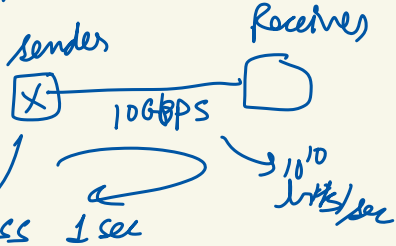
$$CW \mathrel{+}= \frac{(MSS)^2}{CW}$$

(2) If congestion is detected, decrease window size agressively.



TCP Tahoe → CW = 1 on detecting congestion
TCP Reno → CW = CW/2 → Multiplicative decrease
TCP Vegas → study later
TCP Africa → not study

AIMD: Add Incr Multi. Decrease
Increase



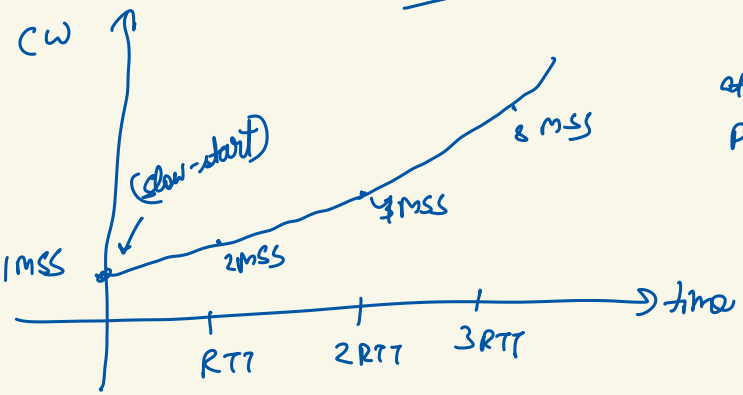
// Initial window $\approx$ 10,000 bits
data rate $\approx \frac{10^4}{1} = 10^4$ bits/sec

(3) Initially
set CW small size
we do not know the appropriate datarate
AI may be too slow

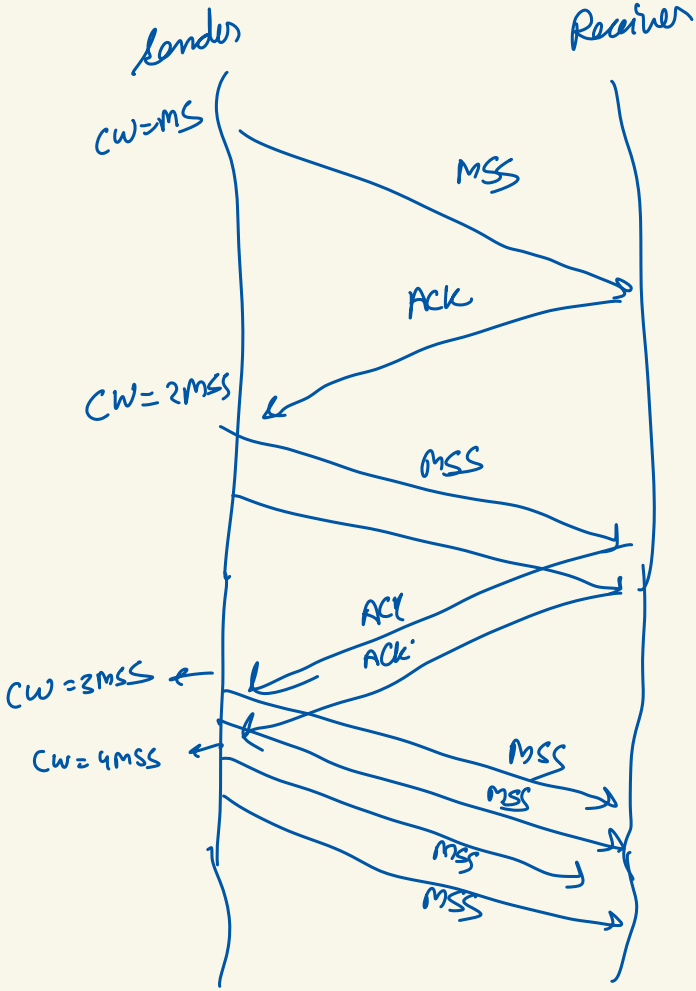
Be aggressive to increase window (initially)

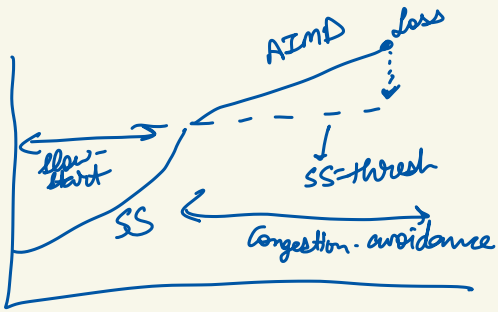
Exponential Increase

Implementation: # ACKs (in window) = $\frac{CW}{MSS}$



at sender
PER ACK
CW += 1MS
(total increase for $\frac{CW}{MSS}$ ACKs
$\approx \frac{CW}{MSS} \times MSS \approx CW$)





RFC

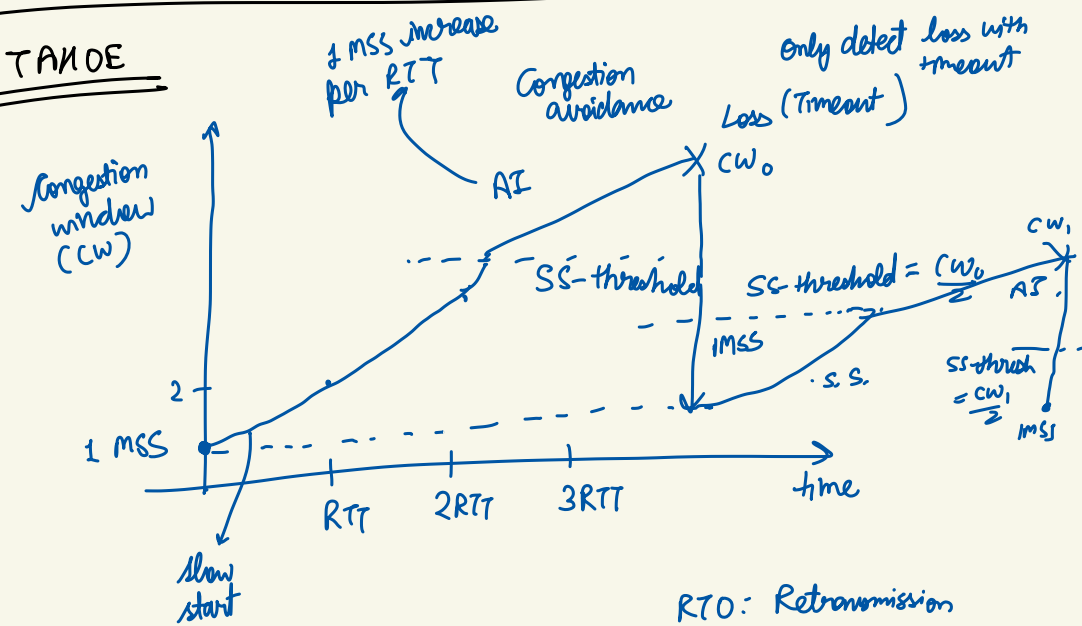
TCP cubic
more agressive

Request for Comments

TCP TAHOE
↓
RENO
↓
VEGAS

CUBIC → latest

TCP TAHOE



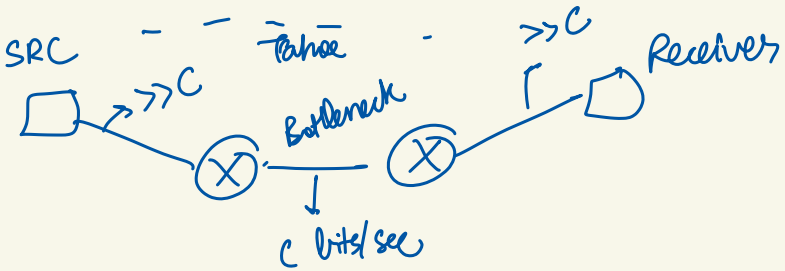
RTO: Retransmission Timeout

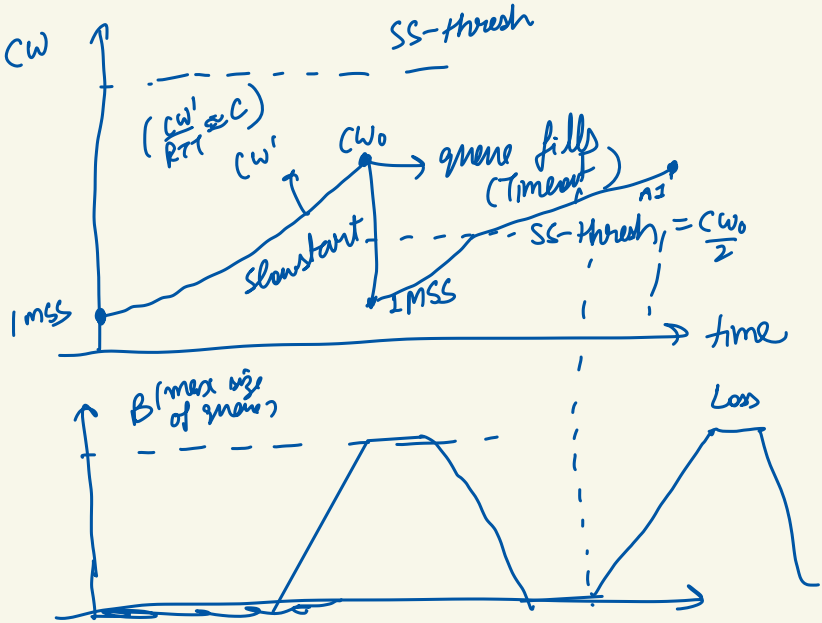
Tahoe Rules

- CW = 1 MSS (Initialization)
- CW += 1 MSS for each received ACK (slow start)
  - ↓ if CW ≥ SS-thresh
- Loss → SS-thresh = CW/2 (MD)

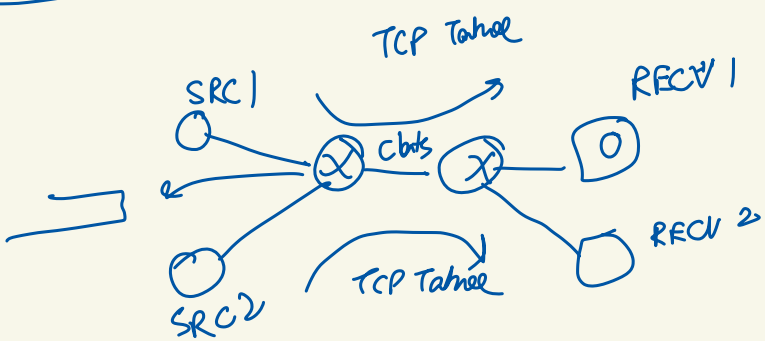
CW += $\frac{(MSS)^2}{CW}$ per ACK received (Congestion avoidance)

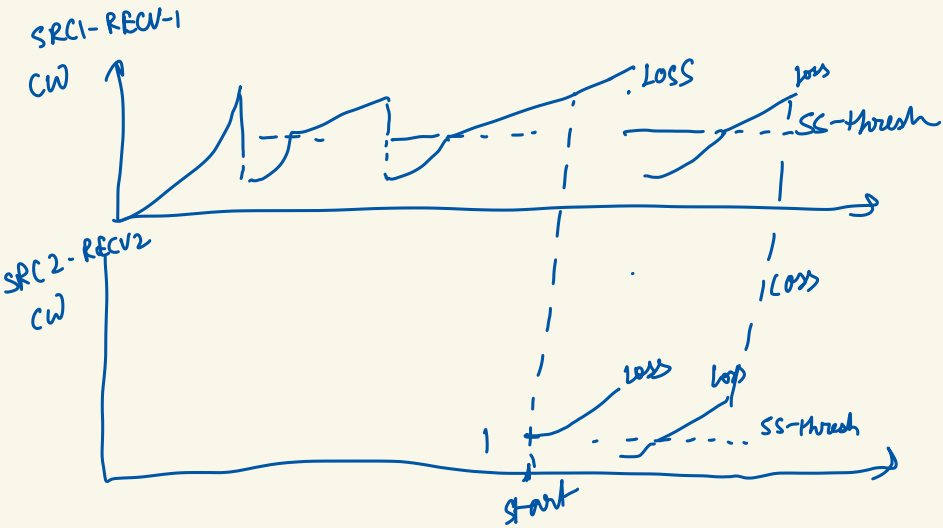
↳ AI





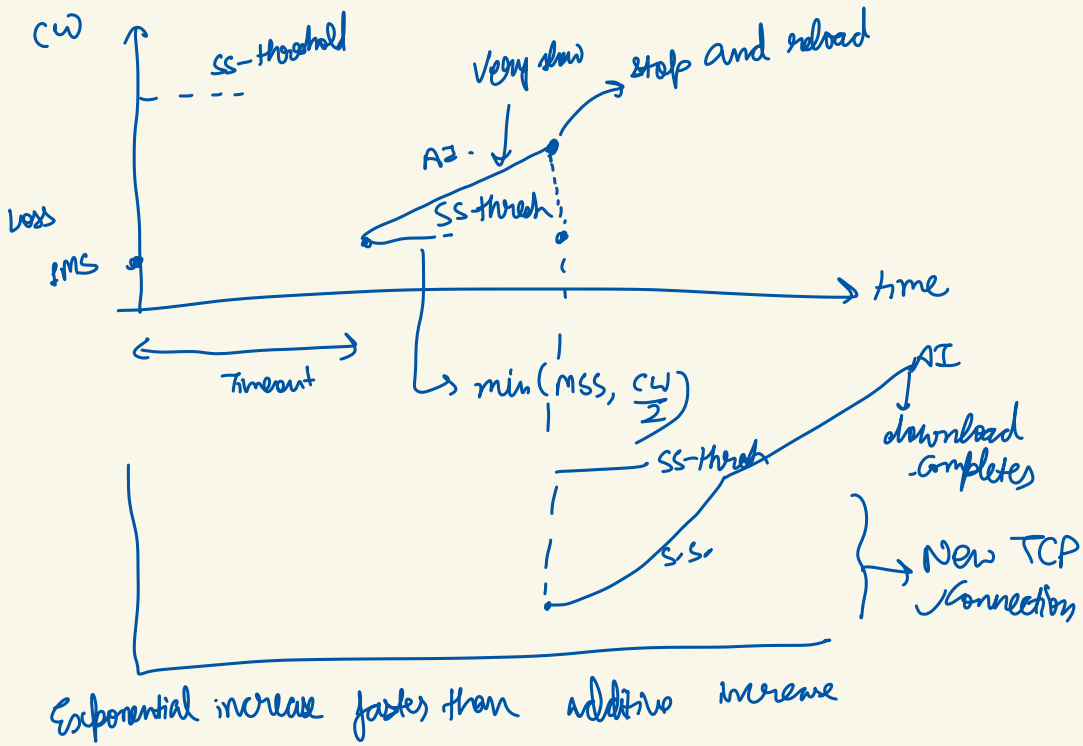
Two TCP Tahoe flows





Stop and Reload:

webpage download slow, hit stop and reload, downloads fast



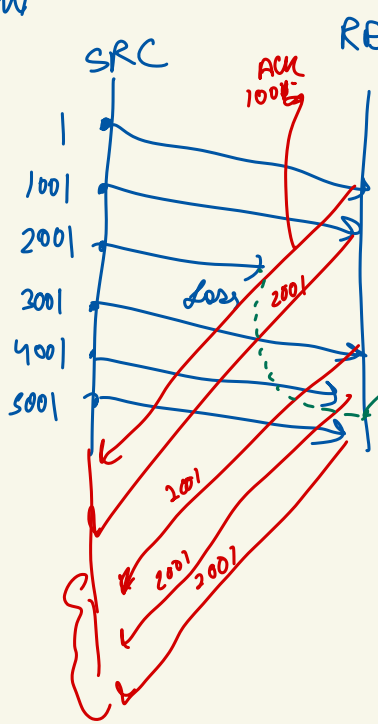
Exponential increase faster than additive increase

Question: (1) should CW = 1MSS after loss

(2) Can we detect loss in other ways?

Enhancements

→ (1) Fast retransmit



that is why wait till 3 duplicate acks waited for

Duplicate acknowledgements DUP Acks

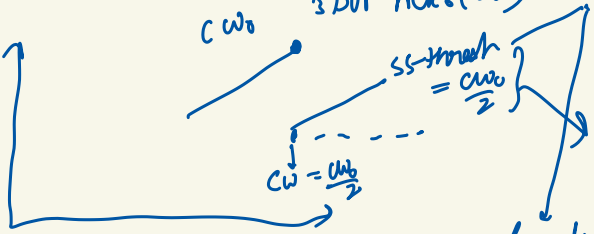
Fast Retransmit: Retransmit segments with SEQ # 2001 if 3 DUP ACKs received with SEQ # 2001 and disable old timers and have new timers

new way to detect Packet loss (Triple dup Acks)

(2) Fast recovery

when loss detected due to 3 DUP ACKs

3 DUP ACKs (Loss)



Loss due to timeout → assumed worse than 3 DUP ACKs

TCP Reno : Includes Fast Retransmit, Fast Recovery.

Miscellaneous

RFC 5681 → Congestion Control

RFC 6298 → timer

→ In slow start, suppose we receive an ACK which acknowledges N bytes of new data

then $cw += \min(N, MSS)$

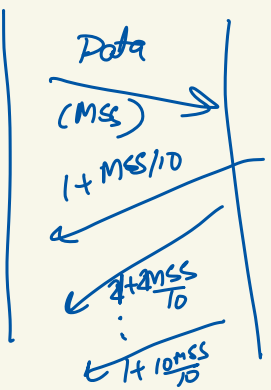
In congestion-avoidance ($CW \geq SS\text{-thresh}$)

For every ACK which acknowledges New data

$CW += \frac{(MSS)^2}{CW}$

So this is done to prevent attack

If we increase by MSS each time, then we can explode CW



RFC 6298 : Initial RTO = 1 sec or higher

Loss detected by timeout ⇒ SS-thresh = max (window/2, 2MSS)

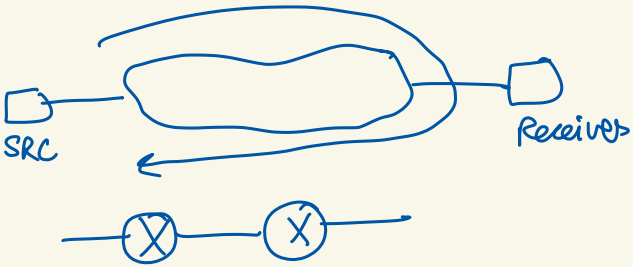
max (cw, Advertised window)

⇒ RTO ← min (2 * RTO, max-RTO)

(To avoid high data rate in case of congestion)

TCP Vegas → 1994-95

RTT detect congestion

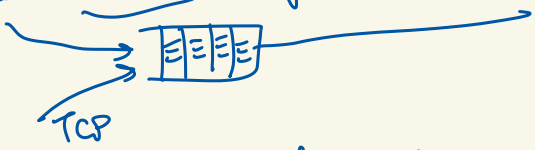


Detect loss by T.O and 3 DUP ACKs and modify cw and SS-thresh as in TCP-Reno.

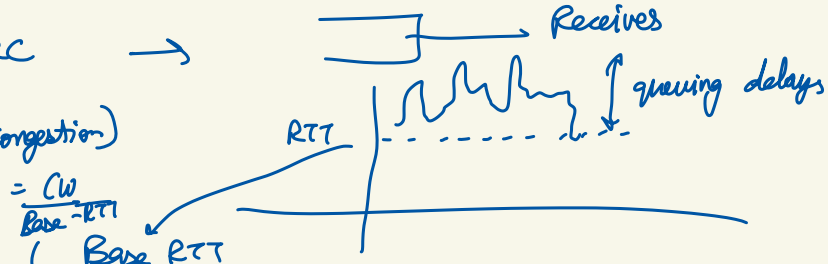
Slow-start rules are the same as Reno

Does RTT for congestion help?

VoIP (over UDP) → latency sensitive



Model path as a single queue



If queue empty (no congestion) then expt rate = $\frac{CW}{\text{Base-RTT}}$

(RTT = Base RTT)

Actual rate = $\frac{CW}{RTT} \leq$ Expt rate (since RTT ≥ Base-RTT)

↳ smooth out the measured RTTs in the recent past

Diff = Expt. rate − Actual rate

In congestion avoidance (CW ≥ SS thresh)

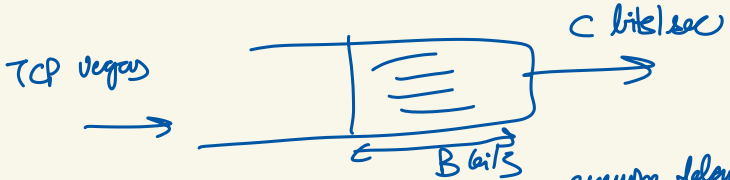
Diff < α (parameter) ⇒ CW is increased by 1MSS per RTT (eg: $CW += \frac{(MSS)^2}{CW}$) at every ACK for new data

$\beta < Diff \Rightarrow$ CW decreased by 1 MSS per RTT

$\alpha < Diff < \beta \Rightarrow$ CW is not modified.

$$Diff = \frac{CW}{Base\text{-}RTT} - \frac{CW}{RTT} = CW\left(\frac{RTT - Base\ RTT}{Base\ RTT \times RTT}\right)$$

(RTT − Base RTT: queuing delay)



$$Diff = \frac{CW\,(B/C)}{RTT \times Base\text{-}RTT}$$ (B/C: queuing delay)

$$\frac{CW}{RTT} \approx C$$ (CW/RTT: Actual rate)

$$\approx \frac{B}{Base\text{-}RTT}$$

Suppose $\alpha = 30$ kBps

$\beta = 60$ kBps

$Base\text{-}RTT = 100$ ms

$\alpha < Diff < \beta$ (targeting)

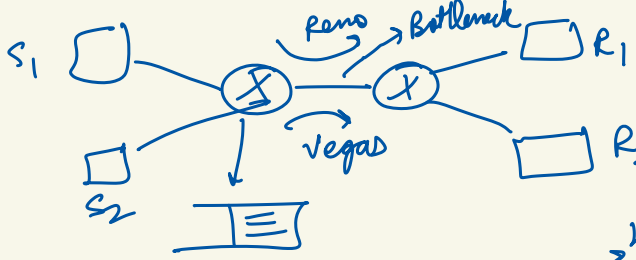
$$\alpha < \frac{B}{Base\text{-}RTT} < \beta$$

$\alpha \cdot Base\text{-}RTT = 3$ kB $< B <$ $\beta \cdot Base\text{-}RTT = 6$ kB

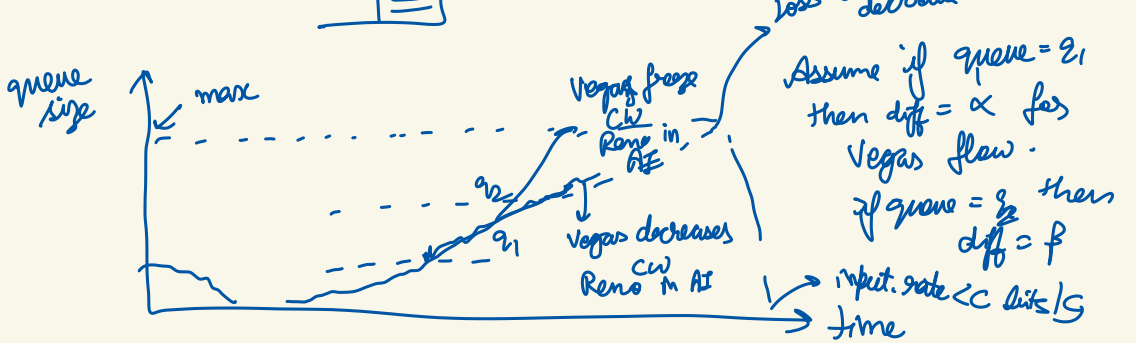
1) What if TCP Vegas flows compete with Reno flows?

2) What if we replace all Reno flows in Internet with Vegas?

1) Reno vs Vegas



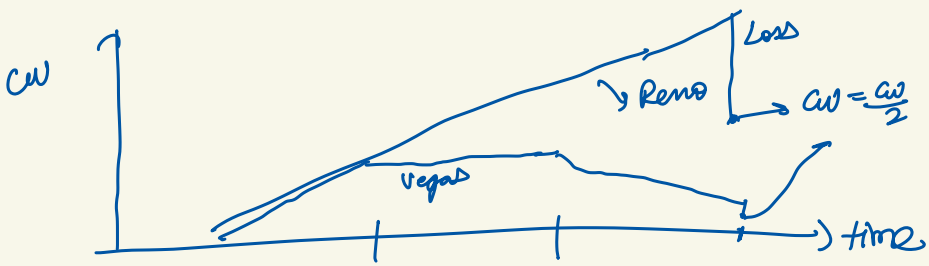
Ignore slow start



Assume if queue = q_1 then diff = α for Vegas flow.

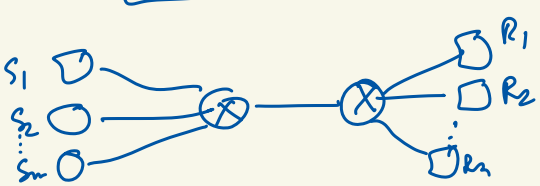
If queue = q_2 then diff = β

input. rate < C link rate



Reno is aggressive compared to Vegas

2) Only Vegas flows



$S_i \rightarrow R_i$ is a Vegas flow

TCP – 5 tuple
(SRC & RST, IP, port, protocol)

Assume all have same RTT

↓ It header (says TCP is transport)

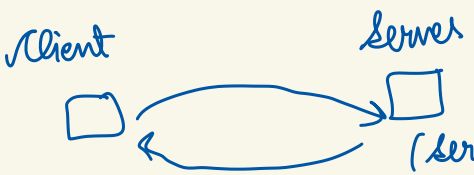
(a) No packet-loss

(b) queuing delays between $\frac{q_1}{C}$ and $\frac{q_2}{C}$ where C = output rate of the queue

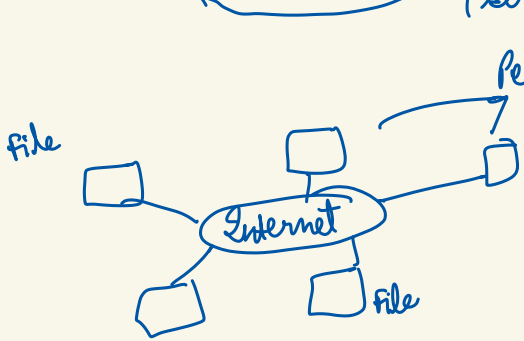
(c) Throughput can be higher than for the all-Reno case, because here the queues never empty unlike the all-reno case

Claim: all Vegas case give 50% higher throughput than all Reno case

P2P Peer-to-Peer Networks

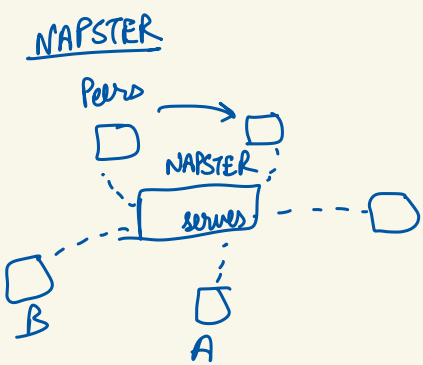


- Bootstrapping
- What service?



1990's

share digital content



- A logs in
- A gives details of files it has

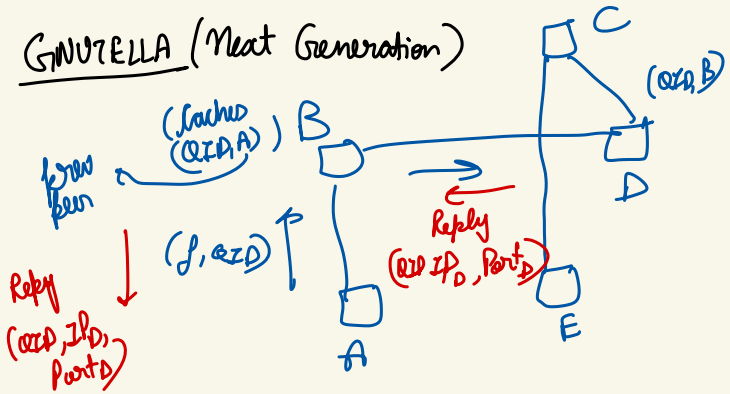
Servers

File 1	IP_A
File 2	IP_B

- C logs in
- Search for song
- Match with file 1, (File 1 $\leftrightarrow$ IP_A) given to C
- C can connect to A and get File

Drawbacks: Centralised → single point of failure (not fault tolerant)
→ legally easy to take down

GNUTELLA (Next Generation)



Bootstrapping
(1) Application may have IPs of some other peers
(2) Lookup IPs from one or more websites

Suppose A wants to search for "f"

- Limited Broadcast (TTL = n, TTL-- at every hop, TTL == 0 ⇒ don't forward query)

Suppose n = 2

First method: Query had "f", QID (Unique ID for query)
Reply over the path which the query traveled.

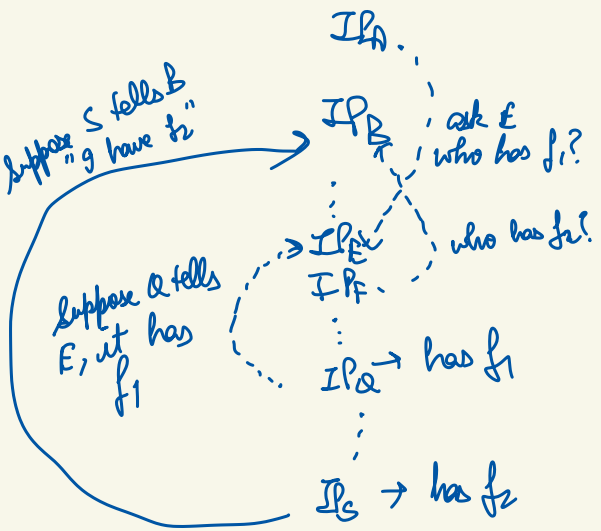
Second method: Query has IP_A as well
Reply directly to A

V0.4 → max TTL = 7

V0.6 → max TTL = 4

Question- If N nodes in network, can I query only $O(\log N)$ nodes to find where data is stored?

IP Address



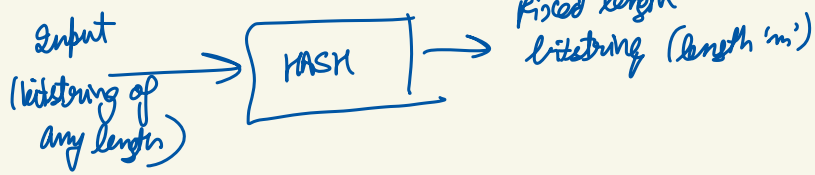
Files

f_1 → mapped to IP_E

f_2 → mapped to IP_B

⋮

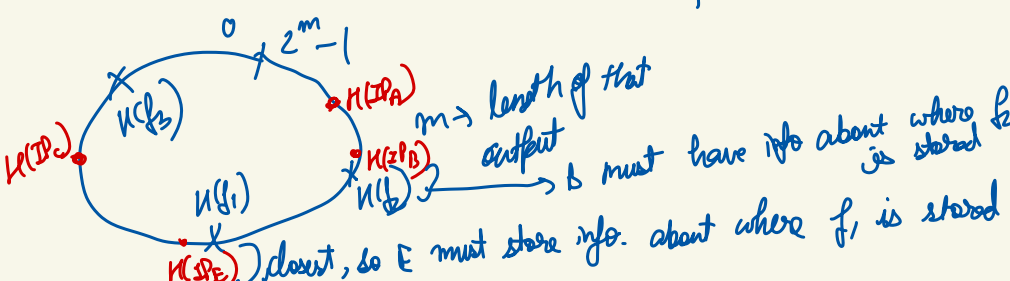
Use hash function

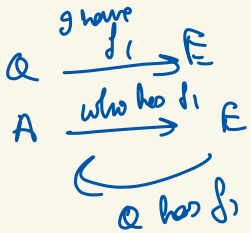


$f_1 \rightarrow [H] \rightarrow H(f_1)$

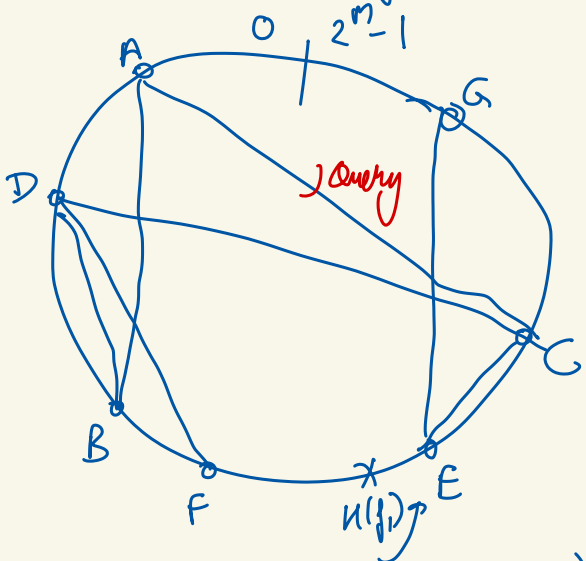
$f_2 \rightarrow [H] \rightarrow H(f_2)$

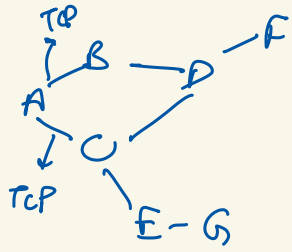
Some IPs might get loaded because of this





Want to store info about where an object is stored at node whose node ID is closest to objID of the object.





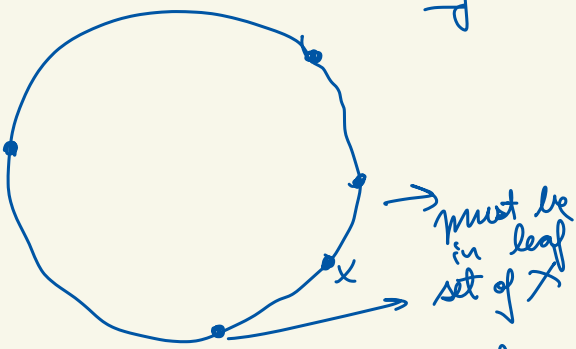
Idea: Route query messages close to $h(f_1)$ in this virtual space till we reach 'E'.

- Suppose A wants to find out where f_1 is stored
- A finds which of its connected peers (leaf set of A) is closest to E
- This repeats till the query reaches a node who has no connected peer closer to $h(f_1)$

PASTRY

In leaf node of X, include $\frac{L}{2}$ closest nodes on either side of X in the virtual space.

Eg $L=2$

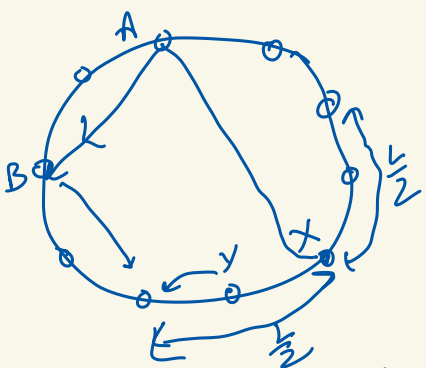


This ensures that the routing protocol gets to node closest to $H(f_i)$.

PASTRY ensures that leaf set of any node is of size $L + O(\log N)$ (N = # peers in network)

Suppose P2P network already has the property that leaf set of all contains $\frac{L}{2}$ closest nodes on either side of the node.

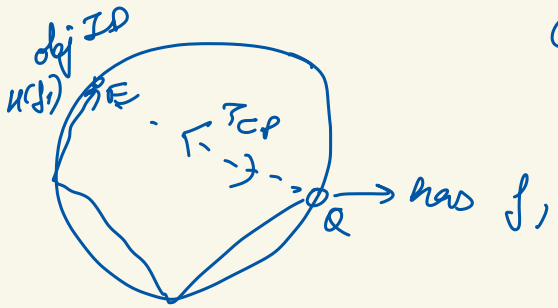
(Chord, Tapestry, Pastry)



($L=4$)

New node X routes a message to find node whose node ID is closest to own ID

From Y, X determines $\frac{L}{2}$ nodes who are closest to it on left and right.

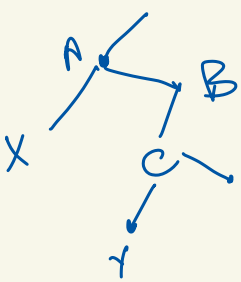


Query has Q's IP address

- E contacts Q
- Q tells E that it has f_1

Similar procedure to find where f_1 is stored.

DHT → Distributed Hash Table

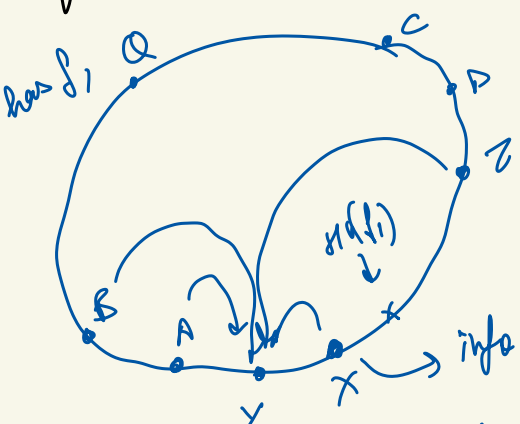


X can add A, B, C, Y to its leaf set

$O(\log n)$

this ensures that leaf set is of size $O(\log n) + L$

Q. what if a node fails?



Y stores information which is present in 2 nodes on either side

All peers send keep alive messages to peers they are connected to (leaf set).

- If any node 2 nodes to right or left fails then it finds out about nodes further away and ensures

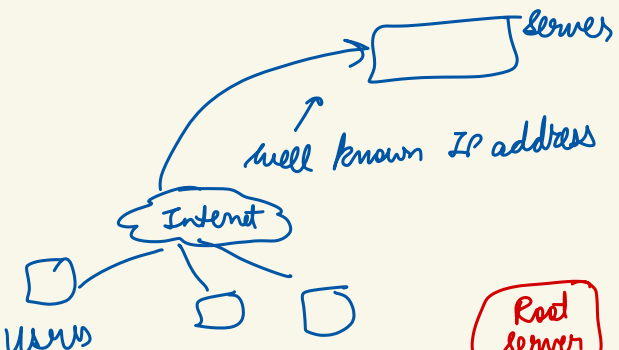
Domain Name System

Originally → IP Addresses 121.5.7.32

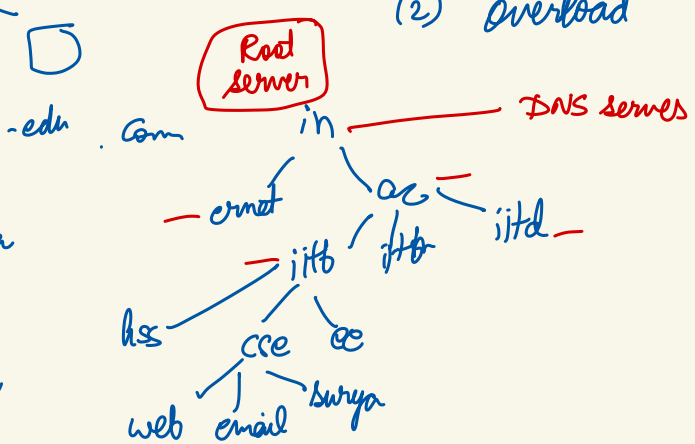
www.google.com
www.cse.iitb.ac.in
} User friendly → DNS → IP addresses

has to be robust

Idea:



(1) Single point of failure
(2) overload



www.cse.iitb.ac.in
in
india

Root Servers
- well provisioned - bandwidths, CPU

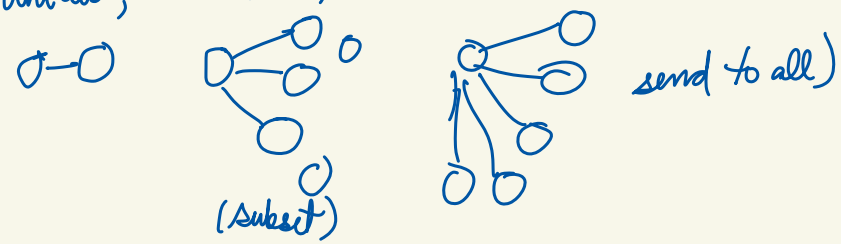
DOS - Denial of service
- Redundancy
  13 Root Servers

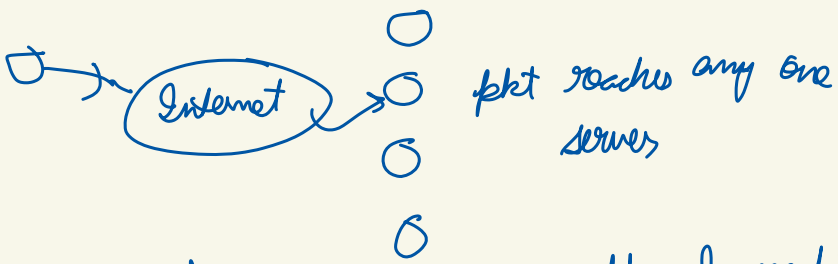
A.root-servers.net
⋮
M.root-servers.net

DNS packets size of 512 bytes
has space for 13 addresses at most

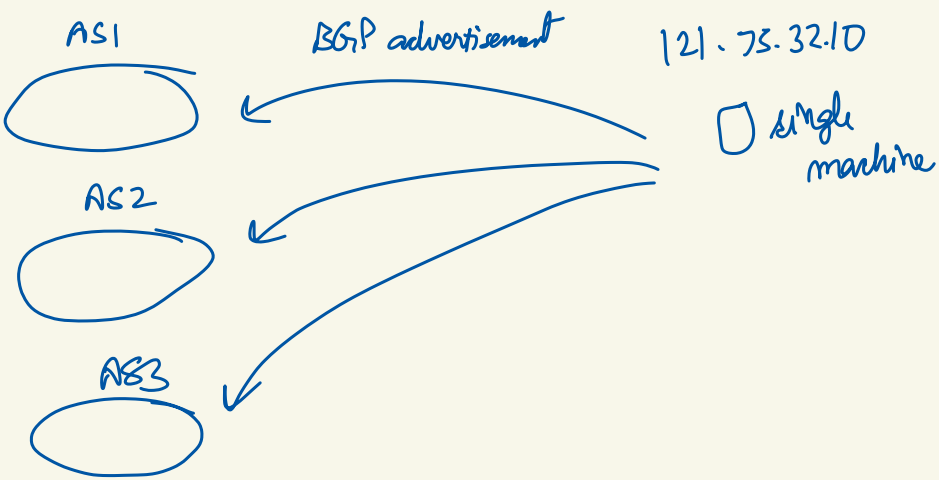
- Uses anycast for more redundancy

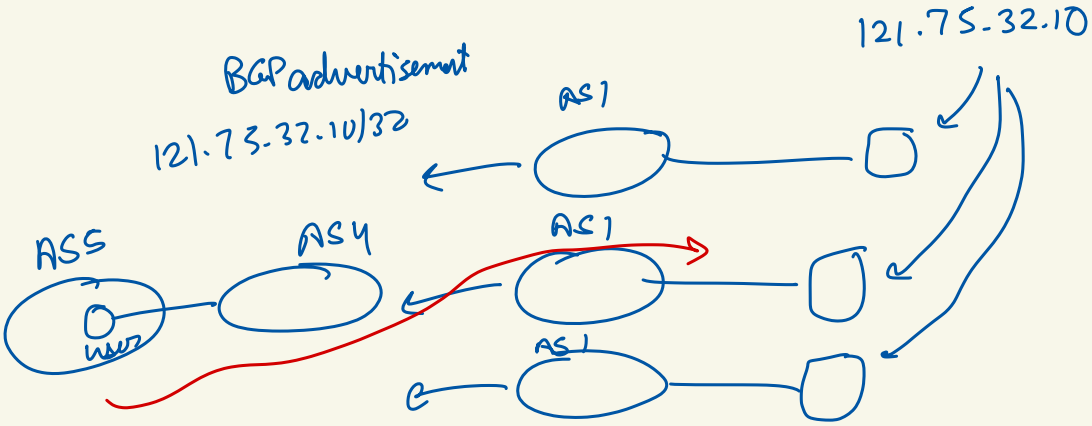
Unicast, multicast, broadcast





A.root-servers.net → mapped to many physical machines geographically distributed





Resource Record

<Name, Value, Type, Class, TTL>

- Name → mapping
- Type → says how value should be interpreted
- Class → IN always for the internet

Type	Value
A	IP address
NS	Name Server (host running the DNS service corresponding to "Name")
CNAME	(alias of "Name"; canonical name of the host specified in "Name")
MX	(name of host running mail server in domain specified in "name")

Linux : dig www.cs.google.com
↳ number decreasing

Examples

Root server has

<edu, a3.nstld.com, NS>
↓
Name of DNS servers of "edu" domain

<a3.nstld.com, 192.5.6.32, A>
↓
IP of "Name"

Server a3.nstld.com has

<princeton.edu, dns.princeton.edu, NS>

<dns.princeton.edu, 128.112.183.5, A>

server dns.princeton.edu

<dns.cs.princeton.edu ...>

server dns.cs.princeton.edu

<penguin.cs.princeton.edu, 128.112.15.6, A>

<www.cs.princeton.edu, cornrods.cs.princeton.edu, CNAME>

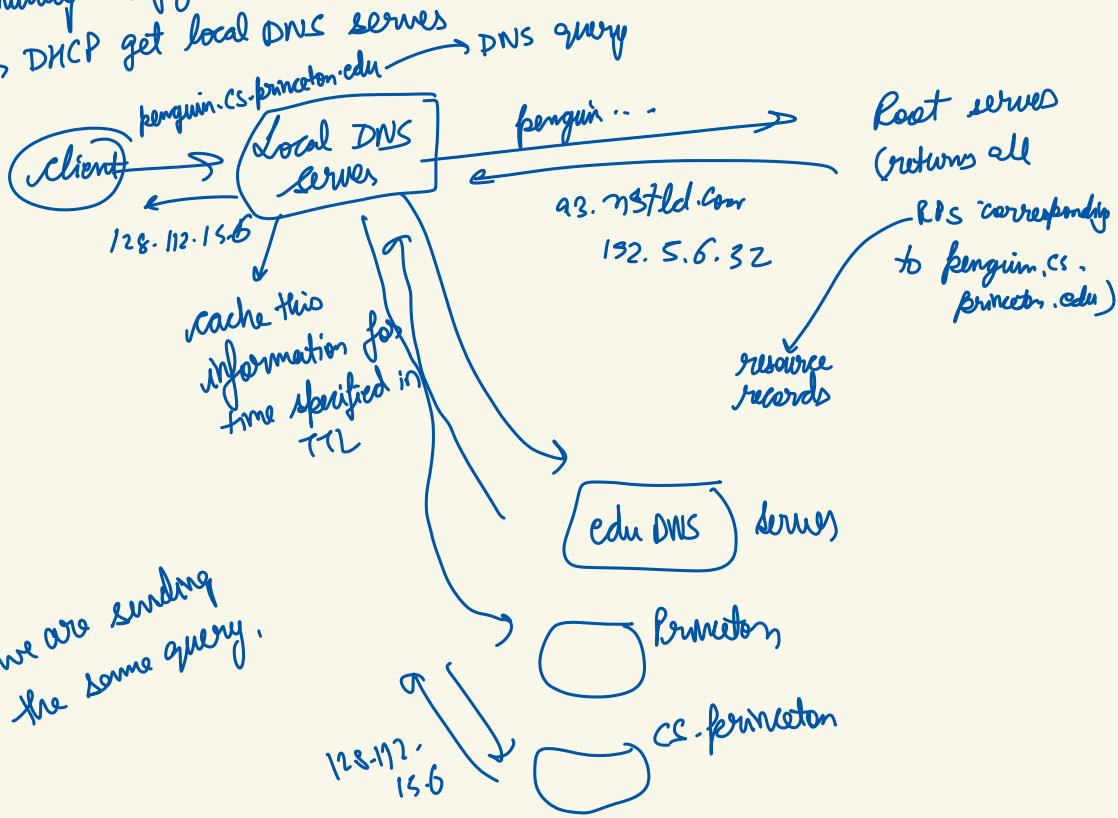
<cs.princeton.edu, mail.cs.princeton.edu, MX>

<mail.cs. ..., 128.12.1.5, A>

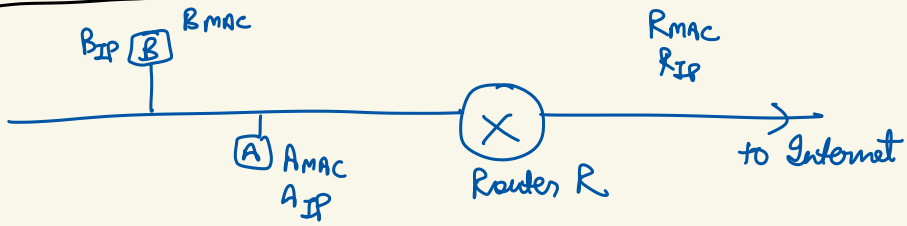
Look up:

DNS runs on UDP (port 53 for servers)

Manually config
or DHCP get local DNS servers

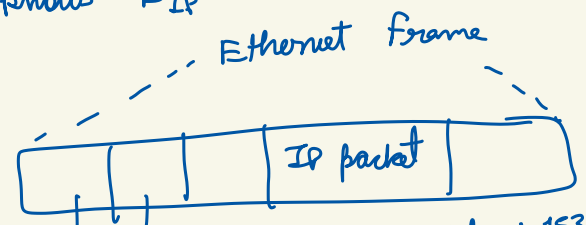


ARP: Address Resolution Protocol

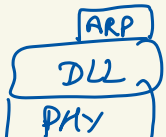


A wants to send an IP packet to B

A knows B_{IP}; A does not know B_{MAC}



ARP to the rescue



Ethernet Frame

Type/Length

Value > 1536 ⇒ this corresponds to TYPE not Length (eg: 0x0806 ⇒ ARP Packet)

all 1's ⇒ broadcast (MAC)

ARP Req: Request

SENDER MAC ; SENDER IP
(A_{MAC}) (A_{IP})

TARGET MAC ; TARGET IP
(B_{IP})

other people see this and ignore

↳ All zeroes

ARP Reply: (from B→A)

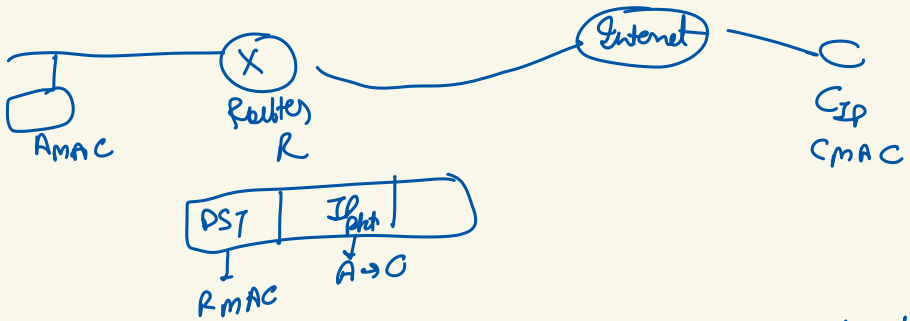
↓

Unicast Ethernet frame

SENDER MAC ; SENDER IP
(B_{MAC}) (B_{IP})

TARGET MAC ; TARGET IP
(A_{MAC}) (A_{IP})

Information about R_{MAC} stored in ARP cache at A with a timeout.



(1) How does A know if DST-IP belongs to own network or not?

(2) If DST-IP is not in our own network, has to know R_{IP}, R_{MAC}

(1) A_{IP} : a_1 a_2 . a_3 . a_4

subnet mask : Ex: 255.255.255.0 (all 1s)

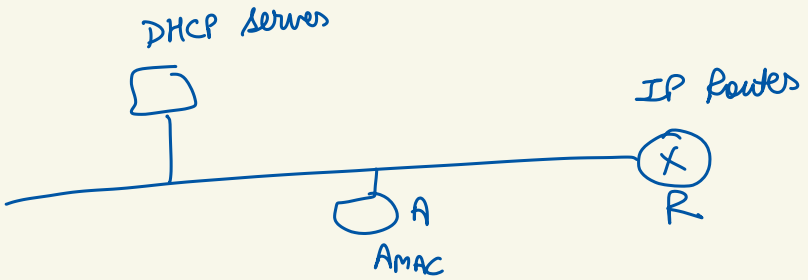
if (DST IP) AND (MASK) = (A_{IP}) AND (MASK)

then DST IP is in my network

(2) Suppose know R_{IP}, how to find R_{MAC}

Use ARP

DHCP : Dynamic Host Configuration Protocol

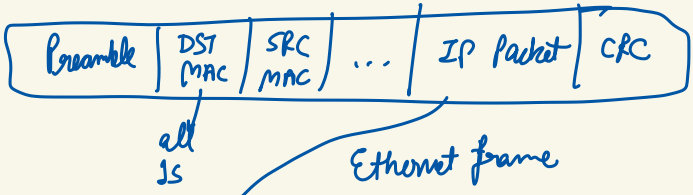


DHCP
↓
UDP
↓
IP → Broadcast
↓
DLL → Broadcast
↓
PHY

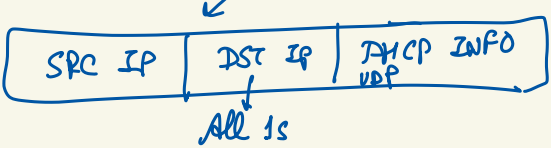
IP ?

Port numbers : 68 → DHCP server
67 → DHCP client

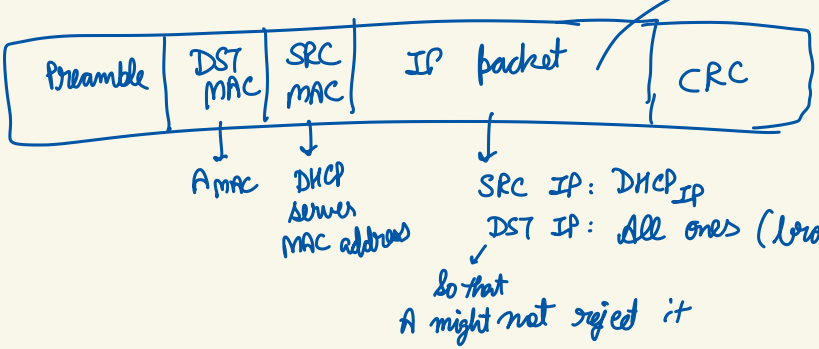
A sends out DHCP 'Discover' packet

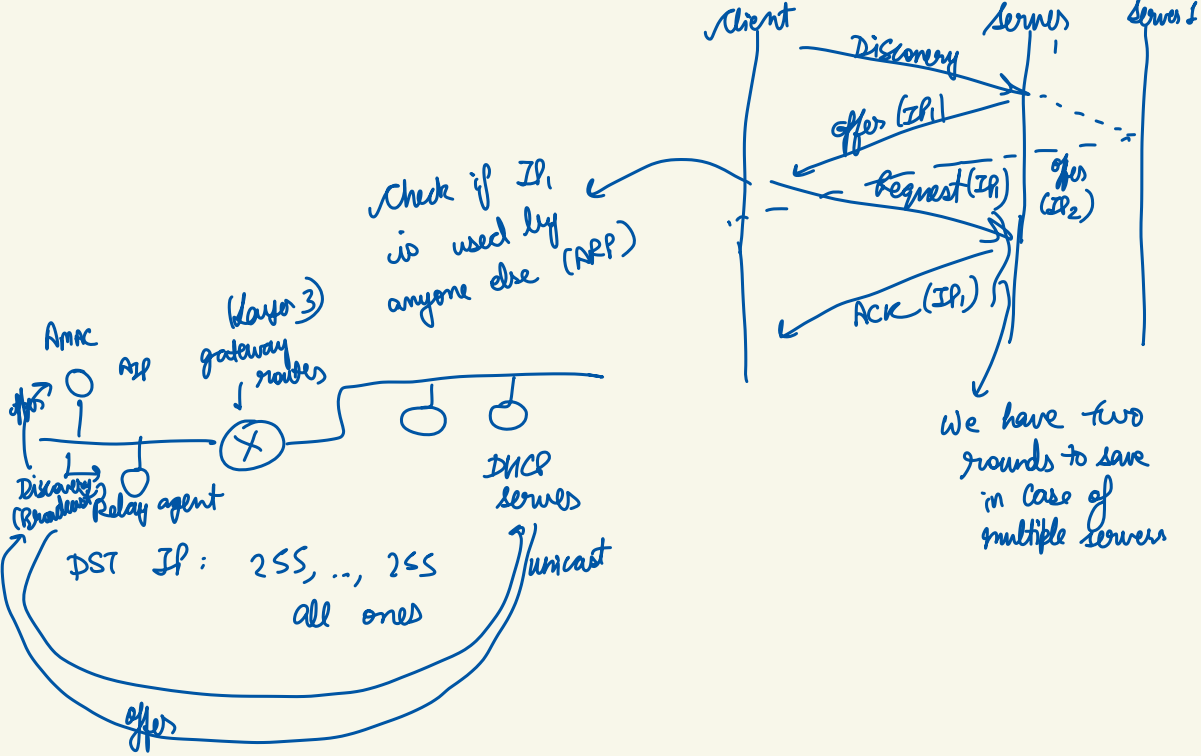


IP packet :

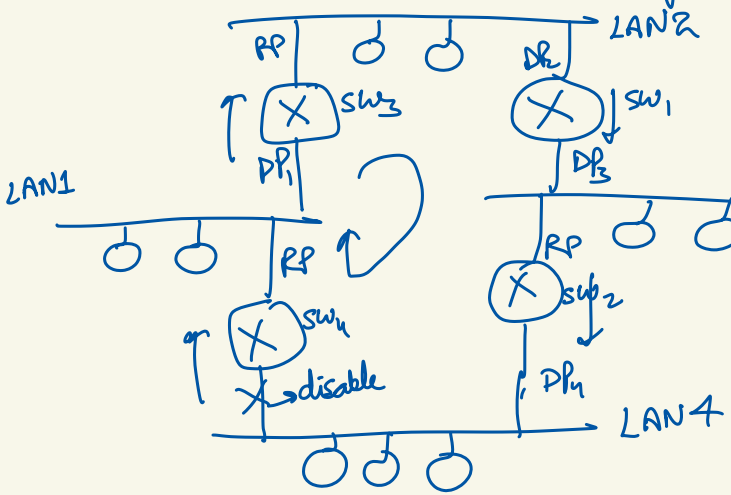


DHCP server Replies (offer) → Contains a potential IP address for A




Client
Server 1
Server 2
Discovery
offer (IP1)
offer (IP2)
Request (IP1)
ACK (IP1)
Check if IP1 is used by anyone else (ARP)
We have two rounds to save in case of multiple servers
Addr
ASP
(layer 3) gateway router
offer
Discovery (Broadcast)
Relay agent
DHCP servers
DST IP: 255, .., 255
all ones
unicast
offer

* Ethernet switches are sometimes called bridges.



Spanning Tree Protocol (Radia Perlman)

Any port not a RP or a DP is disabled.

SPT Protocol

1. Elect root bridge
2. Each bridge finds which port is closest to root and assigns this port as root port.
   (Tie breaking rule)
3. All bridges connected to a LAN, elect one among them to forward frames on that LAN (Designated Port)

Details:

1) Elect root

Bridge ID:
↓
Lowest becomes root.

def: 32768
0 – 61440 (multiples of 4096)
Configurable port (2 bytes)

MAC Address
6 bytes
(smallest MAC of all ports)

Each bridge tells its neighbours
(y, d, X)
X → my ID
d → distance to y
y → the smallest ID heard till now

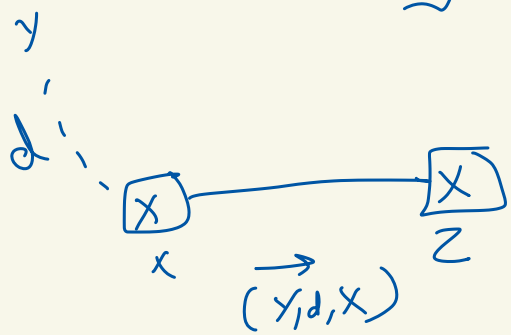
SW$_i$ → ID$_i$
SW1 → (1, 0, 1)
SW4 → (4, 0, 4)

SW2 : (1,1,2)

:

SW4 : (2,1,4)

after hearing SW2, SW3

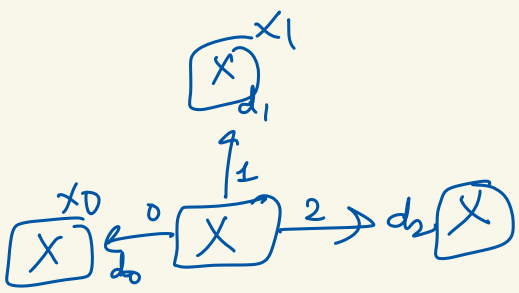


(Y_2, d_2, Z)

If $y < y_2$ then $y_2 = x$ and $d_2 = d + dist(X,Z)$

If $y = y_2$ but $d + dist(X,Z) < d_2$ then $d_2 = d + dist(X,Z)$

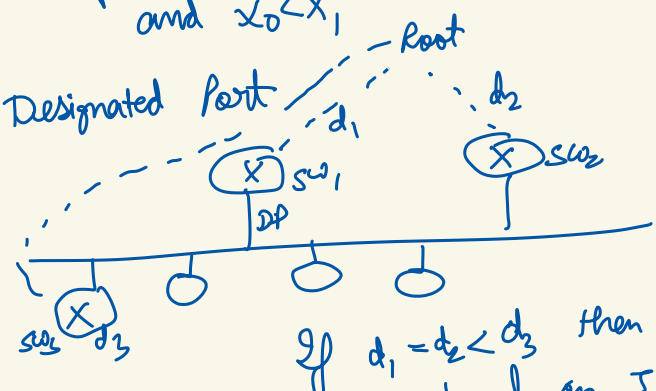
2) Root Port



if more than one port has the smallest distance to root, then tie-break based on ID (smallest) of neighbours on ports

If $d_0 = d_1 < d_2$ and $x_0 < X_1$

3) Designated Port



$d_1 < d_2$

$d_1 < d_3$

If $d_1 = d_2 < d_3$ then tie break based on ID of SW_1, SW_2

Ports not RP or DP are disabled