Guarentee of service, quality of service in fix sized frame

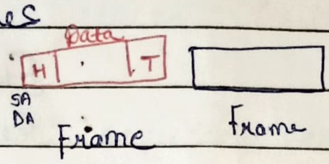
9/March/2022

Unit - 3

Datalink Control (DLL)

Datalink control

i → Framing :- composing data from upper layer into frames



H = Header

T = Trailer → Redundancy check & error detec^n

→ Size should be optimal

→ Framing can be formed as -

(i) fixed size frame

Size of frame is already fixed.

Advantage :- • need not to tell reciever when frame started/ended

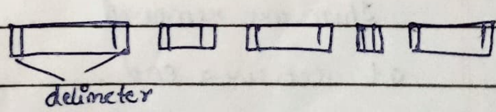
~~(ii)~~

• Every frame takes equivalent time to pass from any switch etc.

(ii) Variable sized frame :-

frame size is not fixed

At Start & end bit = delimeter/ preamble/flag.



Two types :—

Error Control
~ Flow control

(a) Character Oriented (byte oriented) framing

→ Data - in form of byte by byte
Header & trailer - in 8 bit (byte) multiples'

Start frame delimeter }
End " " } 1 byte.

flag	H	Data	T	flag
1 byte	multiple of 8	Each in byte form	multiple of 8	→ 1 byte

→ Only supports english characters/ numbers.

Disadvantage -
The flag's combination of 8 bits could be same as in data.

i.e. -

fdag	H	Da	flag	a	T	flag

→ By this reciever could only consider data till midde data combinat^n (= flag)

• To prevent this, an escape character is added ~~after~~ before the flag's sequence (that came in b/w).
This is called byte stuffing
--

flag	H			Esc	flag			T	flag

→ when reciever collects data, it removes escape character.
→ If the escape character's combin^n is also in b/w data as a part, the another escape character is added.

Imp For each flag or escape appearing in data as part, 1 escape character is added.
Additional escapes are removed at reciever end.

flag	H			Esc	flag			Esc	Esc			T	flag

These are removed at reciever end.

Stuffing is used when flag appears in data

(b) **Bit Oriented Framing**

↳ Any no. of bits in flag, Header, trailer or data

↳ Now in general protocols flag sequence = 01111110 is used.

↳ This sequence can appear in data also, for this Bit Stuffing is used.

Rule → if in data 011111 appears, after this add 0. Receiver will remove this. Even if data is 0111110 → 0 is added.

Eg:

01111110	H	0101011111101101111101010	T	01111110

Flag ⇓ 011111010

⇓ Already like rule, add 0 01111100

{ Flag is made with rarest combin.^{n} }

Data Link

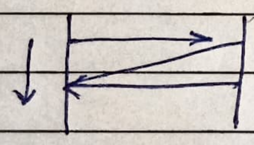
- Framing
  - Fixed Sized framing
    - Character Oriented
    - Bit Oriented
  - Variable sized framing
    - Character Oriented
    - Bit Oriented
- Error Control
- Flow control

{complete chart}

11/March/2022

Error Control
- Packet Loss
- Packet Error

timer



{ Packet retransmission if we get lost in way }

Receiver silently discard the error and give no acknowledgement due to which the timeout at sender side and then it retransmit the packet

Acknowledgment के आने से पहले कितना Data भेजना है, यह flow control का काम है।

If packet has error or less, "what should be done" is work of error control.

Flow Control & Error Control Protocol

Noiseless Channel
- → Simplest protocol
- → Stop and wait

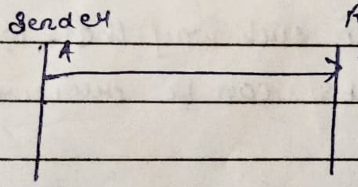
Noise Channel
- → Stop & wait ARQ } Simplex
- Duplex { → Go-Back NARQ
- → Select repeat ARQ

Noise Free channel :-

(1) Simplest protocol / unrestricted Protocol / Utopia Protocol :-

Assumptions of simplest protocol

- → Data traffic is simplex
- → Channel is noise/error free
- → Both sender & receiver has infinite no. of buffer
- → " " " " " " processing speed



(simplex communication)

Note :- There is no need of acknowledgement in this type.

14/March/2022

(2) Stop and wait ARQ (Automatic Repeat Request)

Assumptions :-

i → Data channel is simplex

14/March/2021

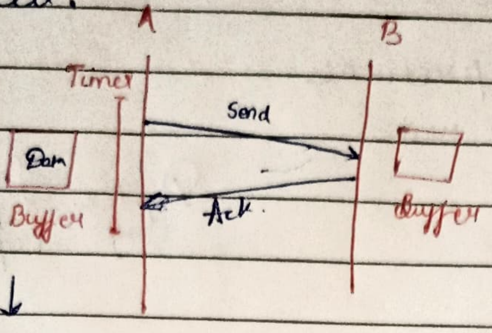
~~Unit — III~~

ii → Channel is noisy.

iii → Both sender and reciever has 1 buffer each

iv → Reciever has limited & finite processing speed.

Buffer = 1 i.e size of 1 frame.



↓ Have a copy of data sent

i) A sends data to B

ii) In B's buffer, error detectⁿ /control is done.

iii) from B's buffer, data sent to upper Layer (Network layer)

iv) When network layer confirms data to B, B acknowledges this to A

* If acknowledgement is recieved within time limit, then everything will go as flow,
But if acknowlegement is not recieved →

A will know data is lost, ∴ it sends copy of data kept in buffer, & timer restarts.

* Timer is decided in such way that last bit is sent & acknowledgements bits can be recieved by A.

* When at reciever's end, data has error, acknow-legment is not sent and data recieved is discarded. This is ~~known~~ called as silently discarding

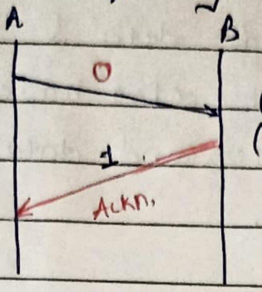
* When acknowledgement is lost -
Data is again sent, reciever recieving same data - it will discard it & will send acknowledgement

Propogation delay / time / latency - time taken by 1 bit to start from sender & reach

Sequence no. →

→ In form of 0101 → alternate manner. (sent)

→ acknowledgment has ~~bit~~ sequence that reciever would be expecting from sender next.



(Recieved data of sequence 0)
(want to recieve data of sequence 1)

Transmission Time

R or C = bandwidth of channel (can be varying depending on channel)

F = frame per bit

$$\text{Trasmission time} = \frac{F}{R}$$

• Sender when finishes insertⁿ of bits, then it waits for data to reach at reciever end & the acknowlegment is recieved

यह channel इतनी देर के लिए खाली रहेगा

Total time to insert, send, recieving acknowledgement

$$= \frac{F}{C} + \frac{F}{C} + 2i$$

{ i = propog.ⁿ time.

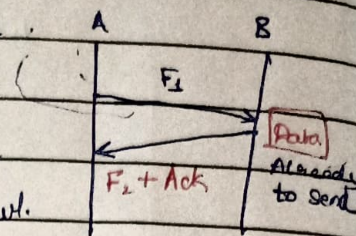
#i Cumulative Acknowledgement

If acknowledgement of previous message is lost but " of next messages are recieved, sender will know that previous message is recieved.

This is known as cumulative acknowledgement

Timer should be set in such a way that cumulative ack. can be recieved.

* ii **Piggybacking** - used in duplex.

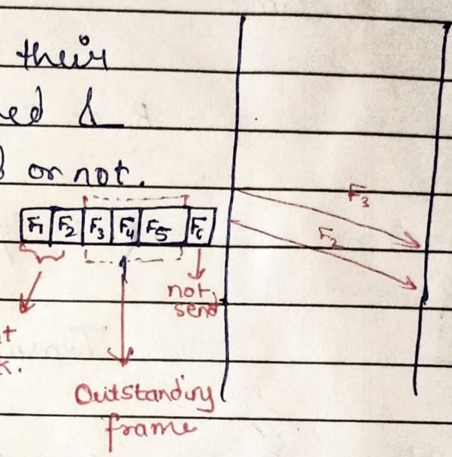


when A sends data & B received & it also have data to send, it will send data + acknowl. of previous data.

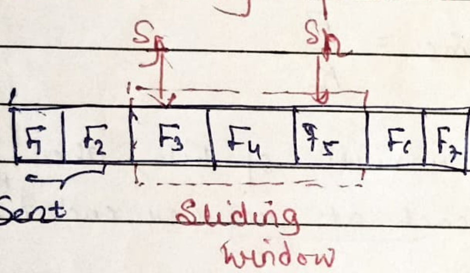
* iii **Sliding Window**

Packets are sent but their acknowledgement is not recieved & could be received or not.

These frames are called



Outstanding frames.



→ As ack. of one packet from window is received, window slides to next

S_n → Pointer pointing to packet that has to be sent next

S_f → " " " first packet in sliding window

(3) Go Back N ARQ

Assumptions

i → Channel is duplex hance concept of piggybacking is used.

ii → Channel is Noisy.

iii → Sender has large no. of buffers

iv → Receiver " only 1 buffer

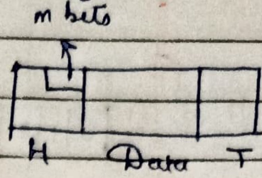
v → Reciever has finite but sufficient processing speed

Sliding window slides when ackno. is recived. vice versa.

Book → Data Communic.ⁿ {Easy} – Forouzan (3rd or 4th)
→ – William Starlin
→ – TeninOne

Sequence No. :-

→ In frame's header
→ m bits
→ packet numbering range
$= 0$ to $2^m - 1$ {modulo 2^m}
(Repeats after $2^m - 1$)



Sliding Window Size

↳ At sender's end = $2^m - 1$
{ size ~~can~~ $\leq 2^m - 1$ }

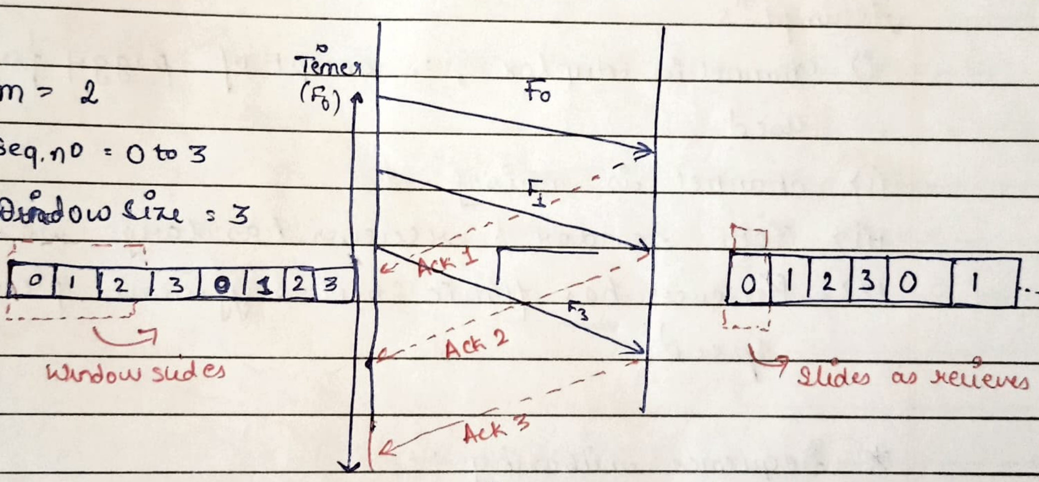
{If m bit represents seq. numbering size in header.}

Eg :-

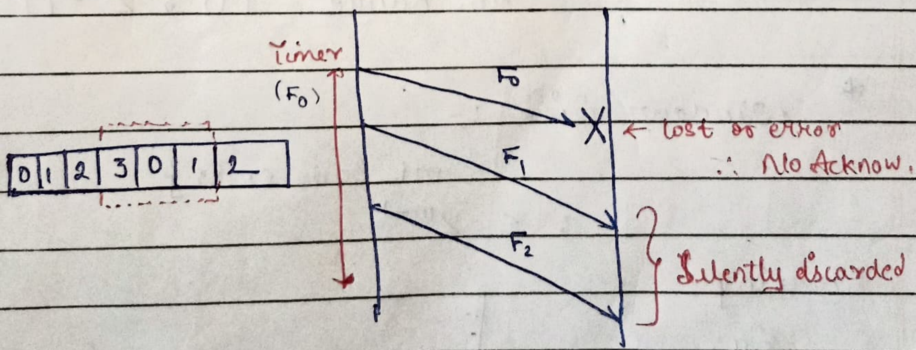
$m = 2$

Seq.no = 0 to 3

Window size = 3



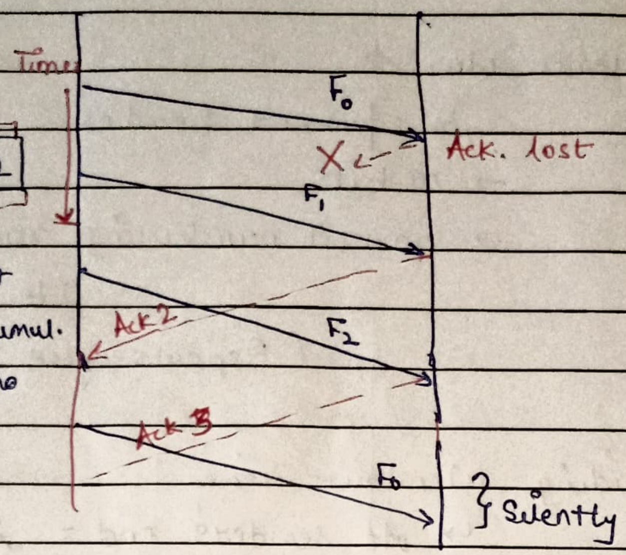
Case 1. – Data & Ack ✓



Time out, ⇒ Resends all from where ack. is not recieved {packets from sliding window}

Case 2 :- Packet lost / error.

0	1	2	3	0	1	2



Time out for F0, no cumul. recieved in time

∴ Resends F0

23/March/22

(4) Selective Repeat

Assumptⁿs —

i) Channel is duplex, ∴ concept of piggy backing is used

ii) channel is noisy

iii) Both sender & reciever has large no. of buffers

iv) Reciever has finite but sufficient processing speed

* Sequence numbering :-

In header : m bits

No. Range $[0 ; 2^m - 1]$

* Window Size :-

{ At Both ends }

$\leq 2^{m-1}$

Working

Eg:

m : 3

Seq. no : 0-7

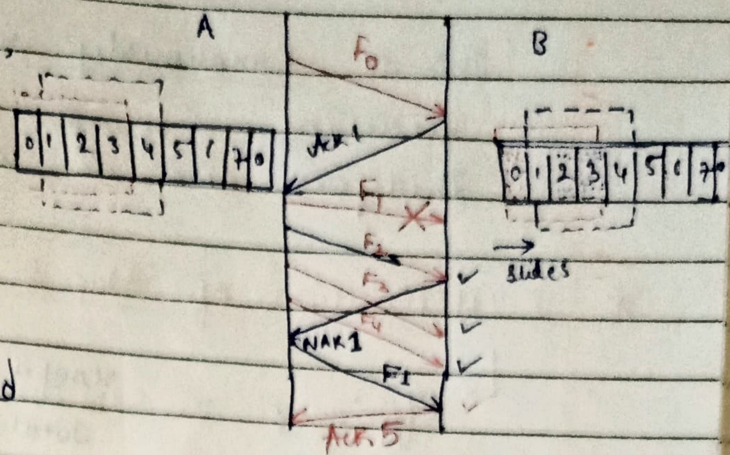
Window size = 4

In sliding window, packets are recieved out of order but is sent to network (upper) layer in order & hence the ackno. is sent. { Ack. is <u>not</u> sent out of order }

→ Sends packet from A, recievers → Sends Ack.

→ Sends packet from A, not recieved → ~~negative Ack.~~

→ Sends next packet, ~~recievers~~, but not recieved prev. → ~~no Ack.~~ negative ack. = NAK



→ NAK is sent only ONCE in a window.

→ NAK is not sent repeatatively bcz sender would be flooded.

→ NAK ~~would~~ is be sent just after recieving frame after lost frame.

→ NAK is sent for speedy re-transmission

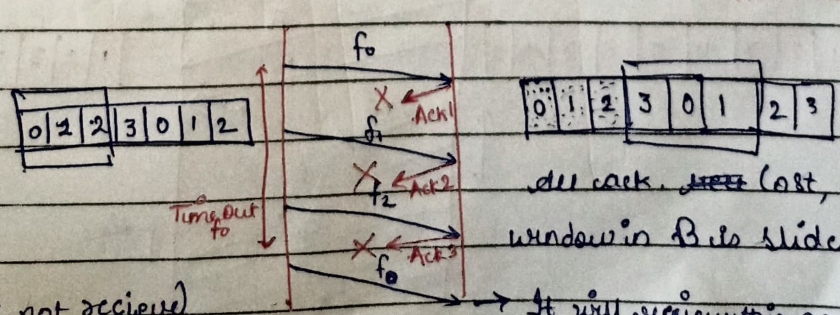
→ Even if NAK is lost, timer is still there. timer will do the work.

Q Why window size has to be 2^{m-1} ?

Let $m = 2$; Window size : $2^{2-1} = 2$

⇒ { check when Ack. of both frames lost at first, ⇒ still it works fine }

Now <u>let</u> Windowsize : 3. { Case: frame recieved, Ack. lost }



all ack. ~~lost~~ lost, sliding window in B is slided

As Ack. not recieved, f0 again sent

→ It will recieve this as current window's 0 !!!

This is erroneously recieved. & duplicate recieving.

Therefore size of window is always $\leq 2^{m-1}$

* Utilization of Stop & Wait protocol

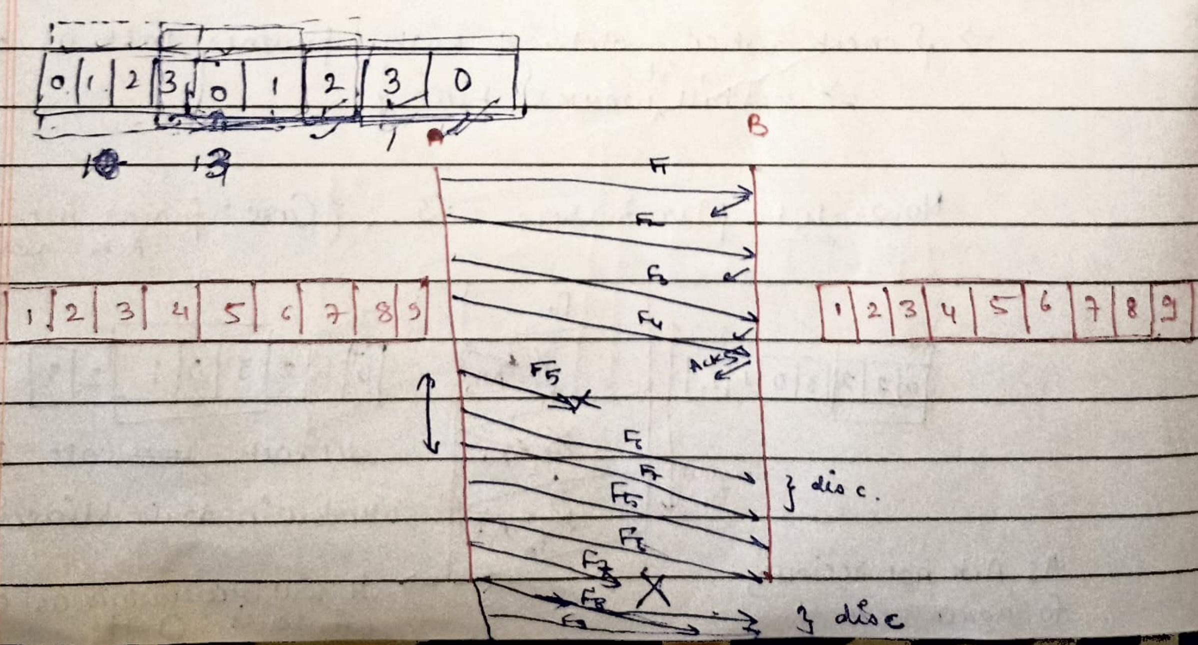
↳ Efficiency = $\frac{\text{Useful time}}{\text{Total time}}$

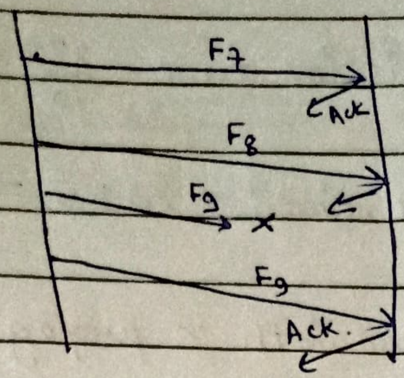
- Bandwidth (Transmission Rate)
- Propogation delay time (Delay / Latency)
- Transmission time

25/Mar/thurs

Q Station A needs to send a message consisting of 9 packets to station B using a sliding window (window size 3) & Go Back-n error control strategy. All packets are ready & immediately available for transmission. If every 5th packet that A transmits gets lost (but no ack. from B ever get lost), then what is the no. of packets that A will transmit for sending the msg to B

Packet = 9 ; Window Size = 3





∴ Total Packets sent = 16

Q) A bit stuffing based framing protocol uses an 8 bit delimeter pattern 01111110. Output is 01111100101 after bit stuffing. What is original input?

(CSE GATE 2014)

Flag: 01111110

Input Data ? $\xrightarrow[\text{Stuffing}]{\text{bit}}$ 01111100101

↓

$\boxed{0111110101}$ Input

28/Mar/2022

Note!

Transmission time depends on bandwidth, packet size

$$\text{Transmission time} = \frac{\text{Packet Size } (f)}{\text{Bandwidth } (R)} = \frac{f}{R}$$

Bandwidth (Transmission Rate)

Eg
Bandwidth
channel = 10 Kbps

frame = 200 bit

Propogation delay = 5 sec

Total time = ?

transmission time = $\frac{200}{10}$ = 20

Total time = 20 + 5 = 25

Bit length = Bandwidth × propagation delay

Bandwidth = 1 bps

Propagation delay = 5 s

Bit length = 5

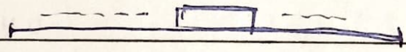
* Bit length - also called bandwidth delay product

Now, Let Bit length = B

Frame size = L

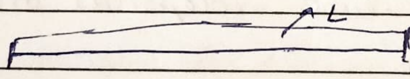
$\left\{ a = \frac{B}{L} \right\}$

(i) $L < B$.

situation 

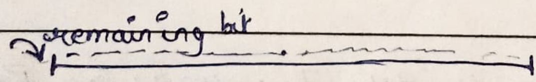
⇒ $a > 1$, i.e. poor utiliz.n (under utilized)

(ii) $L = B$



⇒ $a = 1$

(iii) $L > B$

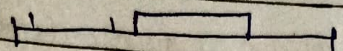


⇒ $a < 1$ i.e good utilization (sufficient utilized)

Now, in terms of propagation delay and transmission time

$$\boxed{a = \frac{B}{L} = \frac{\text{propog.n delay}}{\text{transmission time}}}$$

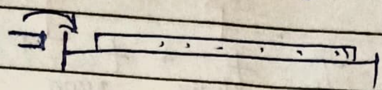
(i) L < B



(under utilized)

(ii) L ≥ B

Insertion of remaining bit

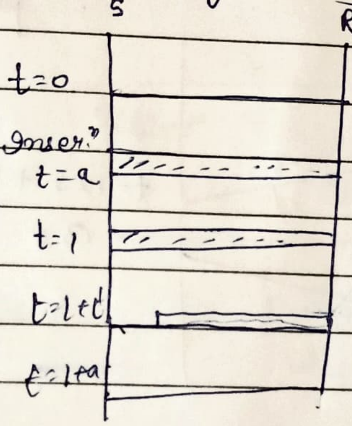


Sufficient utilized.

30/March/2022

Assume Transmission time = 1

then a = propogaⁿ delay



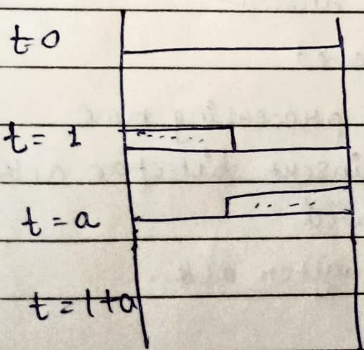
At t=0 first bit insert

At t = a first bit reach R & still inserⁿ is going on at S (a ≤ 1)

At t=1, insertion stops

recived all

At a > 1, propogation delay > transmission time



first bit inserⁿ

last bit insert

a > 1 (channel is underutilized)

Note

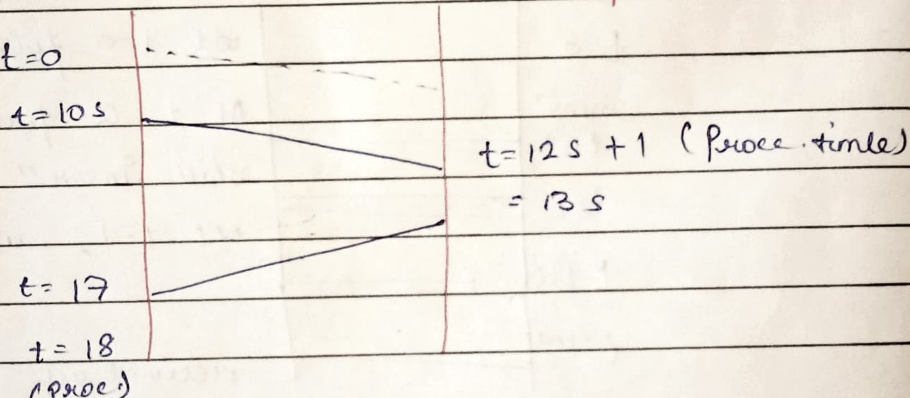
Transmission time preferred to be greater than propogaⁿ delay so that channel is sufficiently utilized.

useful time = transmission time (bit insertion time)

to transmit — prop. time from 1 staⁿ to another — to transmit ack

$$\boxed{\text{Total time} = T_f = t_{frame} + t_{prop} + t_{process} + t_{ack} + t_{prop} + t_{process}}$$

processing at each staⁿ



0	First bit insert
10	last bit insert
12 s	data recieved
13 s	Add 1 s processing time
15 s	time to insert bit for ack
17 s	ack recieved
18	processing after ack.

$$nT_f = n\left(t_{frame} + t_{prop} + t_{proc} + t_{ack} + t_{prop} + t_{proc}\right)$$

As t_{proc} <<< , it can be neglected.
Similarly t_{ack} is negligible

$$nT_f = n(t_{frame} + 2t_{prop})$$

$$\text{Utiliz}^n = \frac{\text{Useful time}}{\text{total time}}$$

$$\text{Utilizat}^n = \frac{n\,t_{frame}}{n(t_{frame} + 2t_{prop})}$$

$$= \frac{t_{frame}}{t_{frame}\left(1 + \frac{2t_{prop}}{t_{frame}}\right)}$$

$$\boxed{\text{Utiliz}^n = \frac{1}{1+2a}}$$

Now, $a = \frac{\text{Propogation delay}}{\text{transmission time}} = \frac{d/V}{L/R}$

where, d = distance
V = velocity
L = frame size
R = bandwidth

$$\boxed{a = \frac{dR}{VL}}$$

1 KB — 2^{10} bit
→ 10^3

/April/2022

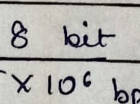
→ Given

Bandwidth = 1.5 Mbps
RTT = 45 msec
Packet size = 1 KB
Protocol – Stop & wait
Utiliz.ⁿ = ?

=> Utiliz.ⁿ $= \frac{1}{1+2a}$

$a = \frac{\text{Propaga.}^n \text{ delay}}{\text{Transmiss.}^n \text{ time}}$

Propagatⁿ delay = $\frac{45 \text{ msec}}{2}$ = 22.5 msc



Transmission time = $\frac{\text{Frame size}}{\text{Bandwidth}} = \frac{1 \text{ KB}}{1.5 \text{ Mbps}}$

$= \frac{2^{10} \times 8 \text{ bit}}{1.5 \times 10^6 \text{ bps}}$

= 5.461 msec

$a = \frac{22.5}{5.461}$ = 4.12 msec

Utilization = $\frac{1}{1 + 2 \times 4.12} \times 100\%$ = 10.8%

Q. A channel has a bit rate of 4 Kbps and oneway propagaⁿ delay of 20 msec. The channel uses stop & wait protocol. The transmission time of ack frame is negligible. To get a channel efficiency of at least 50%, what will be the minimum no. of frames size?

=> Given

Bandwidth = 4 Kbps

Propogation delay = 20 msec

Utilizⁿ = $\frac{1}{2}$

Frame size = ?

Utiliz.ⁿ = $\frac{1}{1+2a}$ => $\frac{1}{2} = \frac{1}{1+2a}$

$\left\{ a = \frac{\text{Propog.}^n}{\text{Transmission}} \right\}$

=> $\frac{1}{\cancel{1}} + 2a = 2$

$2a = 1$

$a = 1/2$

=> $\frac{1}{2} = \frac{20\text{ msec}}{(\text{Frame Size / Bandwidth})}$

$\frac{1}{2} = \frac{20 \times 10^{-3}}{\text{F. S.}} \times 4 \times 10^{3}$

F.S = 160 =

Q A link has a transmission speed of 10^6 bit/sec. It uses data packets of size 1000 bytes each. Assume that the acknowledgement has negligible transmission delay, & that its p. delay is same as for data prop. delay. Also, assume that proc. delay at nodes = negli. Efficiency of Stop & wait in this is exactly 25%. The value of the one way prop. delay? (in msec)

Gate CSE 2015 Set 2

Given

Bandwidth = 10^6 bps

Frame Size = 1000 bytes

P.D. = ?

Utiliz.^{n} = $\frac{1}{4}$

$$\text{Util.}^n \Rightarrow \frac{1}{1+2a} = \frac{1}{4}$$

$$\Rightarrow 4 = 1+2a$$

$$a = \frac{3}{2}$$

$$a = \frac{\text{Pr. D}}{\text{T.t.}} = \frac{\text{P.D.}}{1000 \times 8 / 10^6}$$

$$\frac{8 \times 1000 \times 3}{10^6 \times 2} = \text{P.D}$$

$$\text{P.D.} = \frac{\cancel{187.5}}{\cancel{10^6}} = \cancel{1.5 \text{ msec}} \quad \frac{\cancel{1500}}{\cancel{100} \times 10^4}$$

$= \underline{12 \text{ msec}}$