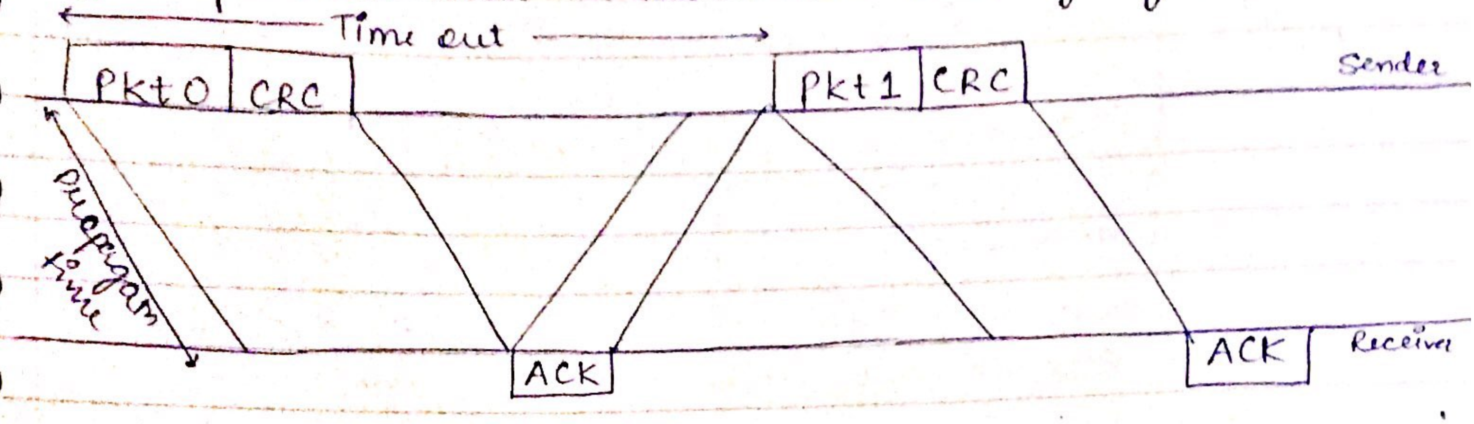
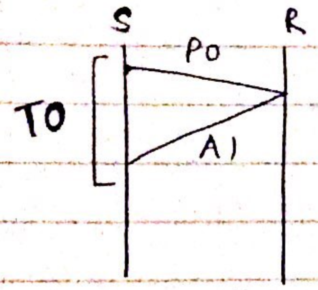
① Stop and Wait → used in Binary sync.



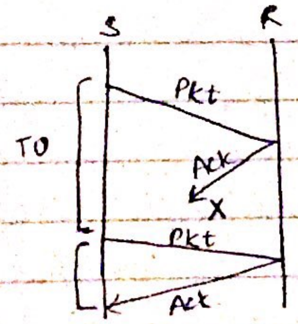
Pkt0 → data packet + control packet
↳ data actual (Payload) ↳ info about packet i.e, headers.

CRC is also control info.

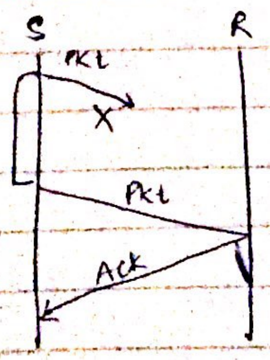
Once sender receives ACK, then only next Pkt is sent.



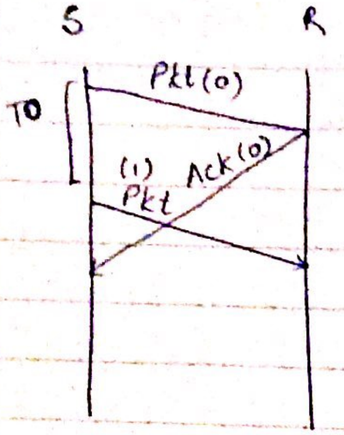
Flow control takes care of error in packets.



Sender waits till TO and resends same pkt



In S&W, one bit sequence no. is used i.e, 0 and 1.
↳ resend pkt.

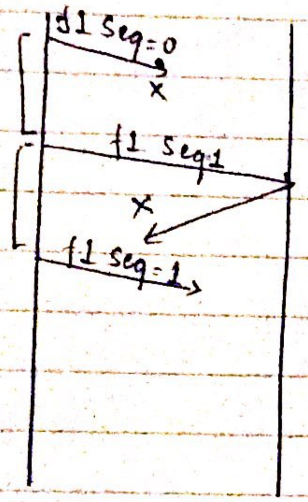


delay in Ack.
Sequence nos. helps here.

Sender now expects Ack(1)
→ discards Ack(0).

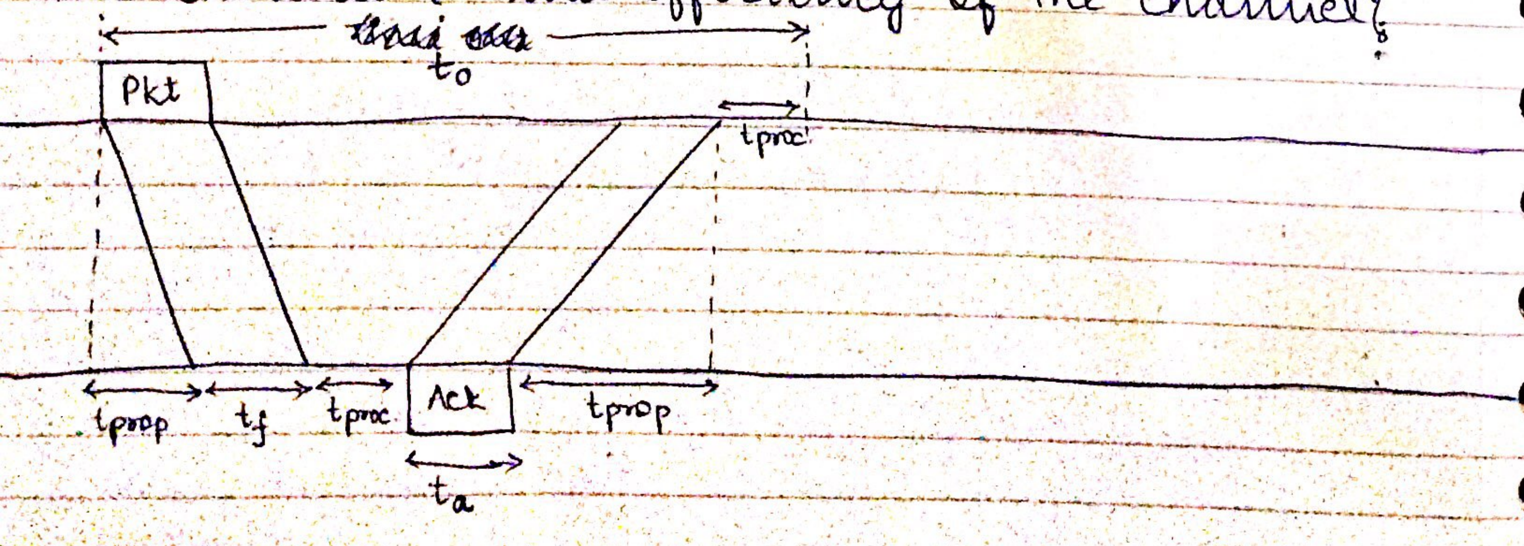
Receiver knows that Pkt(1) has arrived, it means either Ack(0) was lost or sent after TO
→ resends Ack with Ack(1)'es name.

Without seq no, server(R) will consider pkt as duplicate.



Drawbacks ⇒ huge TO's.

Q.) Transmitted segment size = 1000 bits long.
Link speed = 1.5 Mbps
The segment transmissⁿ and reception of ack is 40ms.
How many no. of bits can be transmitted about this channel ? Find efficiency of the channel?



Bandwidth delay product $= 40\times10^{-2}\times1.5\times10^{+6}$
$= 60,000$ bits

Efficiency $= \dfrac{1000}{60,000} = \dfrac{1}{60}$

$$t_0 = 2\times t_{prop} + \underbrace{t_f}_{\text{frame transmiss}^n\text{ time}} + \underbrace{t_a}_{\text{ack transmiss}^n\text{ time}} + 2\times t_{proc}.$$

$$= 2(t_{prop} + t_{proc}) + \frac{n_a}{R} + \frac{n_f}{R}$$

$n_a \rightarrow$ no. of bits in ack.
$n_f \rightarrow$ no of bits in the info. data
$R \rightarrow$ bit rate of link/channel.

$$R^{o}_{efficiency} = \frac{\text{no of bits delivered to destination}}{\text{Total time req. to deliver the info bits.}}$$

$$= \frac{(\overset{\text{Total bits}}{n_f} - \overset{\text{control/overhead bits}}{n_o}) \rightarrow \text{data bits}}{t_0}$$

The transmission efficiency of stop and wait ARQ is ratio of R^{o}_{eff} to R.

$$\eta_{SW} = \frac{R^{o}_{eff}}{R} = \frac{(n_f - n_o)}{t_0} \div R = \frac{n_f - n_o}{R\left(\frac{2R(t_{prop}+t_{proc})+n_f+n_a}{R}\right)}$$

$$\boxed{\eta_{SW} = \frac{1 - \frac{n_o}{n_f}}{1 + \frac{n_a}{n_f} + \frac{2(t_{prop}+t_{proc})R}{n_f}}}$$

B → Bytes

Q) a) segment sizes are 1250 bytes and 25 byte overhead. and 25 byte ack. Rate R = 1 Mbps (bits).

~~Tproc = 1~~

3 delays → 1 ms, 10 ms, 100 ms.

↳ 2(tproc + tprop). = (i) 1 ms, (ii) 10 ms, (iii) 100 ms

b) Repeat same problem for R = 1 Gbps.

Find out efficiencies of your S&W protocol.

(a) $n_f = 1250$ bytes $= 1250 \times 8$ bits $= 10000$ bits

$n_o = 25$ bytes $= 25 \times 8$ bits $= 200$ bits

$n_a = 25$ bytes $= 200$ bits

$R = 1 \times 10^6$ bits/sec

(i) $2(t_{proc} + t_{prop}) = 10^{-3}$ sec.

$$\eta_{SW} = \frac{1 - \frac{200}{10000}}{1 + \frac{200}{10000} + \frac{10^{-3} \times 10^{6}}{10000}}$$

$$= \frac{9800}{10000 + 200 + 1000} = \frac{9800}{11200} \approx 0.875$$

$= \underline{87.5\%}$

(ii) $2(t_{proc} + t_{prop}) = 10^{-2}$ sec

$$\eta_{SW} = \frac{9800}{10000 + 200 + 10000} = \frac{9800}{20200} \approx 0.485$$

$= 48.5\%$

(iii) $2(t_{proc} + t_{prop}) = 10^{-1}$ sec

$$\eta_{SW} = \frac{9800}{10000 + 200 + 100000} = \frac{9800}{110200} \approx 0.0889$$

$= 8.89\%$

The affect of transmiss^n errors on the efficiency of stop and wait ARQ —

1) If a frame/segment incurred errors during transmiss^n Time out mechanism will cause retransmiss^n of the frame/segment. In that case, t_o is transmiss^n or retransmiss^n time in secs.

$t_o \rightarrow$ Transmiss^n/ Retransmiss^n time (sec).

$P_f \rightarrow$ probability of segment transmiss^n has errors and it needs retransmission.

Average successful retransmissions $= \frac{1}{1-P_f}$ (1 in 10 segments)

$$t_{sw} = \frac{t_o}{1-P_f} = t_o\left(\frac{1}{1-P_f}\right)$$

$$\eta_{sw} = \frac{\left(\frac{n_f - n_o}{t_{sw}}\right)}{R} = \eta_{sw}\Big|_{\text{without error}} \times (1-P_f)$$

Q) Prev. Que.

$P_f = 10^{-6}, 10^{-5}, 10^{-4}$ (bit error rates).

To calculate the transmission eff. for stop and wait ARQ,

1) we need to calc. an avg. total time req. to deliver a correct frame.
2) ~~n_t~~ i transmissions are req. to deliver a correct pkt / segment.

$$n_t \leftarrow i$$

i e, (i-1) retransmissions are req.

Derivation

$$P[n_t = i] = (1-p_f)\, p_f^{i-1} \quad \text{for } i = 1, 2, 3 \ldots$$

$$E[t_{sw}] = t_0 + \sum_{i=1}^{\infty} (i-1)\, t_{out}\, P[n_t = i]$$

$$= t_0 + \sum_{i=1}^{\infty} (i-1)\, t_{out}\, (1-p_f)\, p_f^{i-1}$$

$$\cancel{= t_0 \left[1 + (1-p_f) \sum_{i=1}^{\infty} (i-1)\, p_f^{(i-1)} \right]}$$

$$\cancel{S = 1 + 2p_f + 3p_f^2 + \ldots}$$

$$= t_0 + t_{out} \frac{p_f}{1-p_f}$$

if $t_0 = t_{out}$;

$$E[t_{sw}] = t_0 + t_0 \frac{p_f}{1-p_f} = t_0 \left[\frac{1}{1-p_f} \right]$$

Q.) Bandwidth = 50 Kbps.
one way transmissn time = 250 ms.

i) Find Bandwidth delay product.

ii) window-size, W = 2 × bandwidth delay + 1.
Find link utilization.

CN by Peterson & Davgid (A system approach)

RTT

Ans) Bandwidth delay product = $50 \times 10^3 \times 2 \times 250 \times 10^{-3}$

= 25000 bits.

IP Address
Subnet mask
default gateway
DNS server

Server

Go to Email → Services → SMTP & POP3 → ON it.
Domain name → mnit.com

Go to DNS now. → give Name as mnit.com.
Address → 192.168.1.1. ↳ local DNS

Open Laptop → email → compose → configure mail.
make email ids.
Username & pswd must be same as in server.
incoming mail server → mnit.com
outgoing ——————— → mnit.com
or add IP address.

Filter apply → DNS, SMTP, POP3.
go to edit filter on right side.

Laptop DNS	switch1 SMTP
Laptop DNS	email SMTP
Switch1 DNS	switch1 SMTP
Email DNS	laptop SMTP.

Switch1 DNS
Laptop DNS
Laptop SMTP
Laptop SMTP

Go Back N → X-modem.
Selective Repeat → TCP.

Primary objective of transport layer or TCP is <u>end-to-end delivery</u>.

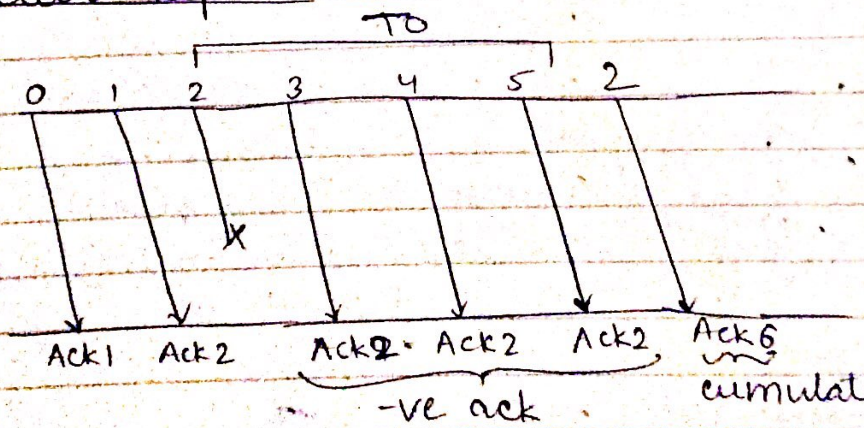
reliability, congestion control and flow control are secondary.

delays → processing, propagation, <u>queuing</u>, transmissⁿ
↳ comes under congestⁿ control.

Simplex → radio transmissⁿ (one way)
(unidirectional)
Half-duplex → walkie-talkie (one at a time)
(Bidirectional)
✓ Full-duplex → transmit & receive at same time.
(Bidirectional)

<u>Sliding window (Pipelined) protocol.</u>
Go Back N
window size = $2^n - 1$ seq. bits → n.
seq nos → 0 to $2^n - 1$.
Retransmit all frames in the window starting the window from the frame where error occured.

<u>Selective Repeat</u>



$\boxed{W = 2 \times BD + 1}$
↳ Bandwidth-delay product

eg:- Bandwidth = 1Mbps , delay = 1ms.

BD product = $10^6 \times 10^{-3}$ = 1000 bits.

segment size = 1024 ~~bits~~ x 8 = 8192 bits.

segment size is 8 times larger than BDP.
Sliding window protocols do not improve performance.
Stop & wait is better. - less complexity.
Large window size ⇒ large timeouts.

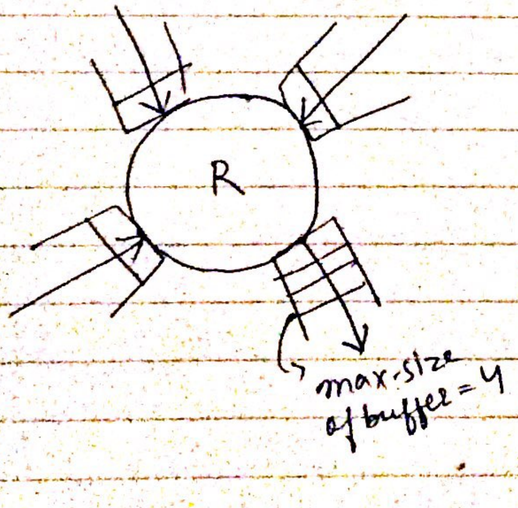
Retransmission ⊗ and ARQ ⊘

Congestion control

Apply max Flow Min cut Theorem.

Congestn control and congestn avoidance.

Changing bandwidth allocatn overtime.



First 3 packets arrive & get stored in buffer.
2 packets again arrive
one of the two gets stored, other gets dropped.
congestion takes place.
we are ~~st~~ using store & forward method.

Sending Rate = min (n/w rate, Receiver rate)

N/w modelling simulator (NS2)

↓ Req.
N/W components very expensive, lets utilize use simulator.

NS2:) discrete event — discrete time
(simulate moreeasy event by event)

when to use?
→ N/w planning
→ N/w evaluatⁿ.

Prog. lang :- OTCL and C++
OTCL → interpreted (scripting lang) defined on top of prog. lang.
C++ ↳ compiler
OTCL → object tool command language.

not used C++ completely → writing model in C++ is difficult.

ID   TYPE   TIME   HANDLER
TIME ↳ simulatⁿ time
HANDLER ↳ call one by one event

TYPE → Time based: ARQ (TO).
TYPE → Packet Based: (drop, send, receive send).

→ scheduler manages simulatⁿ event list.

Running an OTCL script → ns script-file.tcl.[parameters]

case1) Modelling a wired n/w.

```
set n0 [$ns node]
set n1 [$ns node].
```

node → fⁿ in c++

```
$ns duplex-link $n0 $n1 10Mb 100ms DropTail.
```

DropTail → when queue fills, drop packets from tail.

(1)
Install NS2 all in one 2.35.
(2) Go to NS all in one folder. Go to NS 2.35. Go to TCL folder. Go to ex (example) folder.
Plenty of .tcl scripting files are there (wired and wireless). Just run them.

[ns filename.tcl]

We get 2 files related to that:-

- nam file → nam filename.nam (Graphical Repres)
- .tr file → observe it.

new trace   old trace.

Q) Suppose 2 hosts A and B are separated by 10,000 kms and are connected by a direct link of R=1Mbps.
Suppose the propagation speed over link is 2.5×10^8 m/sec

a) Calc. the bandwidth delay product.

b) Consider sending a file of 400,000 bits from A to B. Suppose the file is sent continuously as one big message. What is the maximum no. of bits that will be in the link at any given time.

c) How long does it take to send the above file if sent continuously.

d) Suppose now that the file is broken up into 10 packets, with each pkt. containing 40,000 bits. Suppose that each pkt is acknowledged by the receiver and the transmissi^n time of an acknowledgement pkt is negligible. Finally, assume that the sender cannot send a pkt until the preceeding pkt is acknowledged. How long does it take to send the file?

Ans) propagation delay $= \frac{10000 \times 10^3}{2.5 \times 10^8} = \frac{100 \times 10^5}{25 \times 10^7} = \frac{4}{100} = \frac{1}{25}$

propagation delay = 0.04 sec.

a) Bandwidth delay product = 0.04×10^6 bits

= 40000 bits.

b) 40,000 bits

c) $\frac{40,00,000}{40,000} = 10$

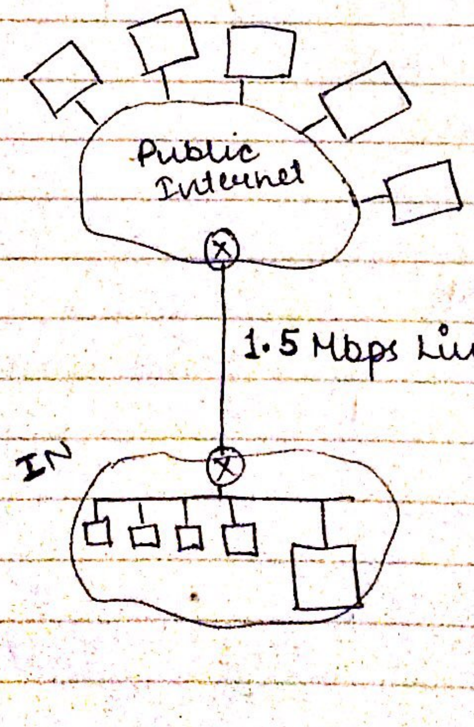
Total time = $10 \times 0.04 = 0.4 + 0.04 = 0.44$ sec

$+ \frac{40,000}{10^6}$ ↳ trasmit

d) Total time = time taken to send 10 pkts + time taken to receive 10 acks

= 0.44 + 0.40 = 0.84 sec

Q) Consider the figure for which there is an institutional n/w connected to the internet. Suppose that the avg. pkt. size is 900,000 bits and the avg.



request rate from the institution's browser to the origin server is 1.5 requests/sec. Also, the amount of time it takes from when the router on the internet side of the axis link forwards a HTTP Request until it receives the response in 2 secs on an avg. Model the total avg. response time as the sum of the avg. axis delay and the avg. internet delay. For the avg axis delay, use $\frac{A}{(1-AB)}$ where A is the avg. time required to send an object over the axis link and B is the arrival rate of objects to the axis link. You can assume that the HTTP request messages are negligibly small & thus create no traffic on the n/w or the axis link.

a) Find the total average response time.
b) Now, suppose a cache is installed at the institutional n/w and hit ratio is 0.4. Find total response time.

Ans) pkt size = 900,000 bits = 0.9 Mb.
request rate = 1.5 req./sec.
request process time = 2 sec.

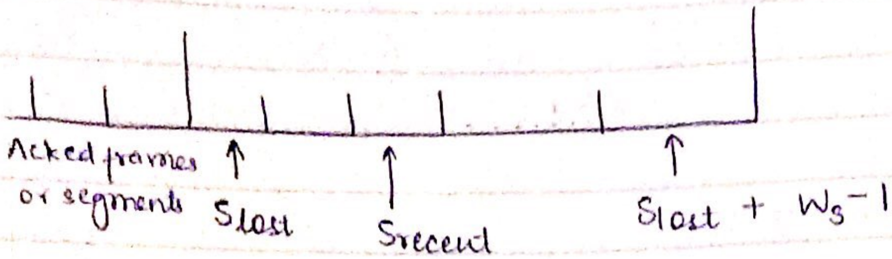
$$\text{avg. axis delay} = \frac{A}{1-AB}$$

$$A = \frac{900,000}{1.5 \times 10^6} = \frac{9}{15}\ \text{sec} = 0.6\ \text{sec}.$$

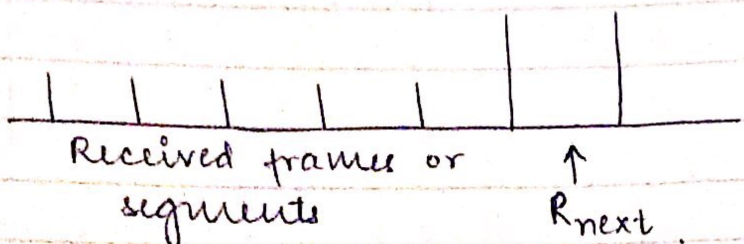
$$\text{avg. axis delay} = \frac{0.6}{1-0.6\times 1.5} = \cancel{\frac{0.6}{1-0.36}} = \cancel{\frac{0.6}{0.64}} = \cancel{\frac{60}{64}}$$

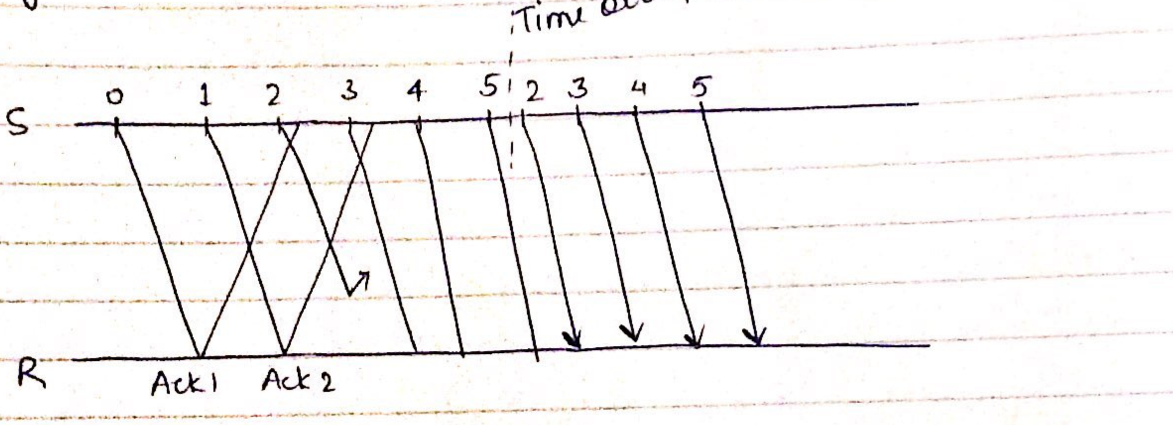
$$= \cancel{\frac{60}{64} = 0.9375\ \text{sec}}$$

SENDER SIDE



RECEIVER SIDE





$$t_{out} = \underbrace{2t_{prop}}_{RTT} + 2T_f^{max} + T_{proc}$$

↳ assuming full-duplex channel.

$$\eta_{GBN} = \frac{\frac{(n_f - n_0)}{t_{GBN}}}{R}$$

$$\boxed{\eta_{GBN} = (1-P_f)\frac{\left(1 - \frac{n_0}{n_f}\right)}{1 + (W_s - 1)P_f}}$$

$$\boxed{\eta_{SR} = (1-P_f)\left(1 - \frac{n_0}{n_f}\right)}$$

if errors are neglected,

$$\eta_{SR} = \left(1 - \frac{n_0}{n_f}\right)$$

(a)

Total response time = $2 + \frac{0.6}{1 - 1.5 \times 0.6}$ = 8 sec

(b) miss ratio = $1 - 0.4 = 0.6$.

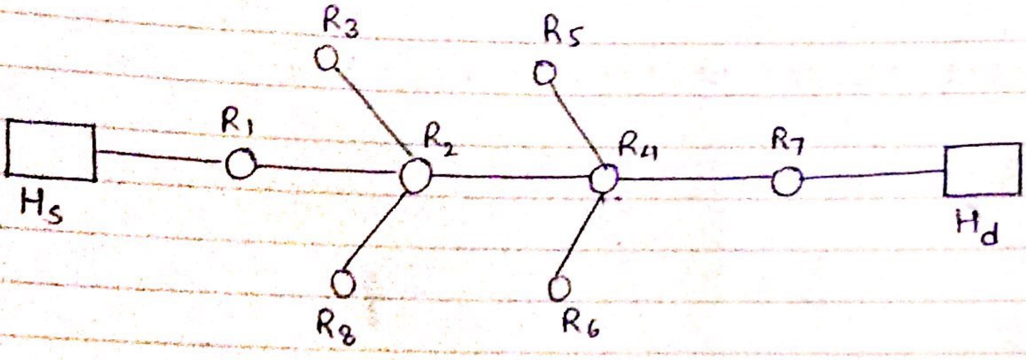
$B = 1.5 \times 0.6 = 0.9$.

$\therefore$ avg access delay = $\frac{0.6}{1 - 0.6 \times 0.9}$ = 1.3 s.

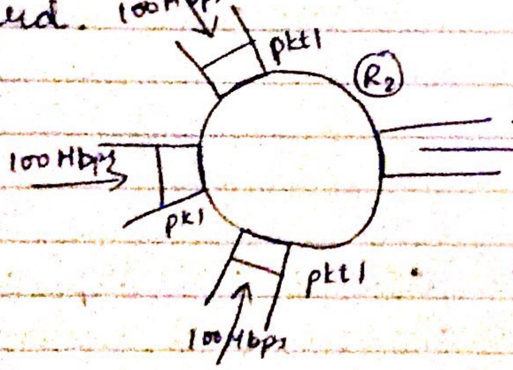
Total time = $2 + 1.3 = 3.3$ sec.

End-to-end communicatⁿ makes it reliable → implemented using ARQs. (acks)

Congestion control



R_2 has 4 different buffers.
Sequence nos from diff links may be identical.
ARQs won't help us here we dont know which pkt to forward.

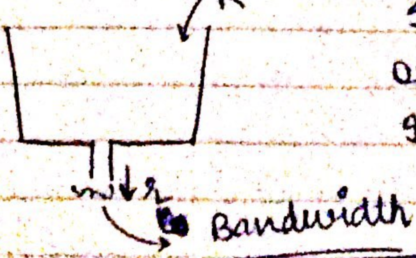


→ Congestion
R_2 need feedback mechanism and tell R_1; R_3, R_8 to control the rate.

Now, if forwarding link is busy, storage buffers may get filled, now how we drop packets.

Traffic Shaping

a) Leaky Bucket



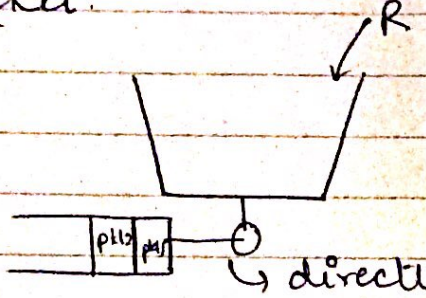
If R=r, no congestⁿ or buffer delay.
If R<r, no buffer delay
If R>r, buffer filled with pkts.

If bucket size is B, and bucket filled & R>r ⇒ packet drop. (drop-tail concept).
↳ newer pkts get dropped.

Pkts have priorities. ~~Let~~ Less priority pkt is removed from bucket when overflow occurs.

Leaky bucket causes problem when R>r.

b) Token Bucket.



↳ directly connected to link.

Bucket fills with token, only if token present, forward token. If pkt rate is high, increase the token rate. If bucket fills, we don't lose pkts, we lose tokens.

Not so good.

We can use additive increase, multiplicative decrease, slow start, fast retransmit fast recovery.
↳ whenever error occurs, slow down rate.
These mechanisms make TCP reliable.