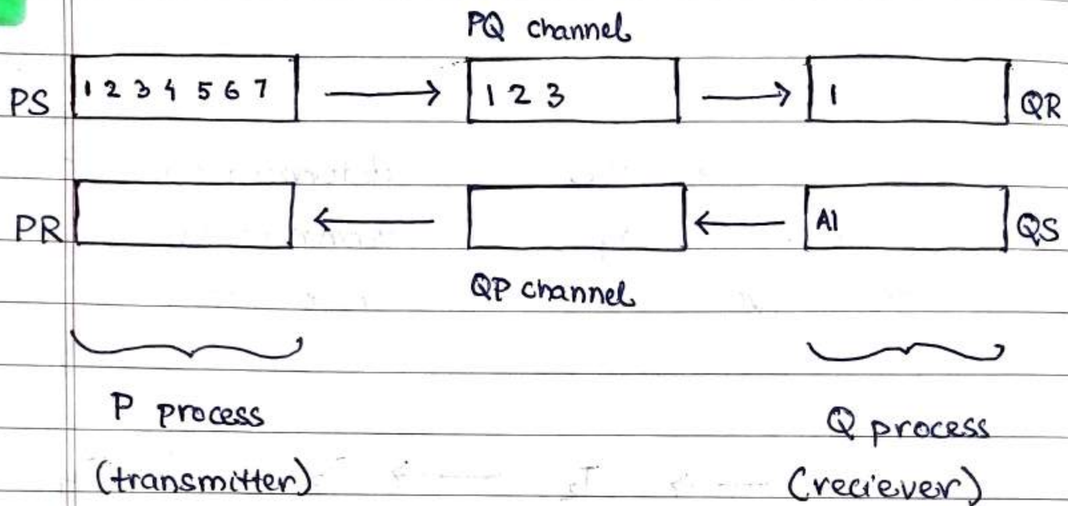
BALANCED SLIDING WINDOW PROTOCOL

its a packet transmission method used for reliable in-order delivery. ex- TCP



in think of P trying to send a file to Q in parts (1-7). Q acknowledges the reciept of a part with an acknowledgement packet.(A1).

THE CHANNELS

2 separate channels are PQ and QP.

- they can drop packets
- they can reorder "
- they cant corrupt packets (even if they could we could use checksum)

THE PROCESSES

2 separate processes P and Q.

- each sends packets through a channel.
- each recieves packets through another channel.
- each only sends packet within a "window".
- position of "window" is determined by the packets recieved by each.

here, P = transmitter process
Q = reciever process

REQUIRED PROPERTIES

(1) safe delivery

packets recieved at Q are the ones sent by P.
packets recieved at P are the ones sent by Q.

(2) eventual delivery

eventually all packets sent by P are recieved at Q

eventually all packets sent by Q are recieved ~~at~~ P.

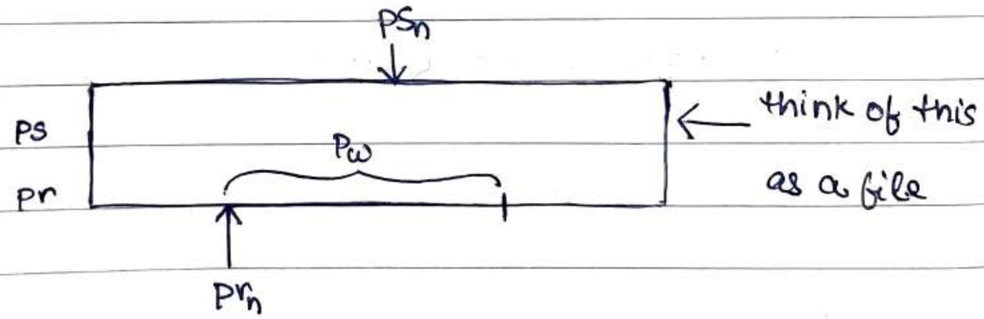
MODELLING AS SYSTEM

consider process P :

state
- ps - packets to send to Q
- pr - packets recieved from Q (ack.)

functions
- ps_n - no. of contiguous packets sent by P
- pr_n - no. of contiguous packets recieved by P

constant
- P_w - window size of P



window of P = $(pr_n, \; pr_n + P_w]$

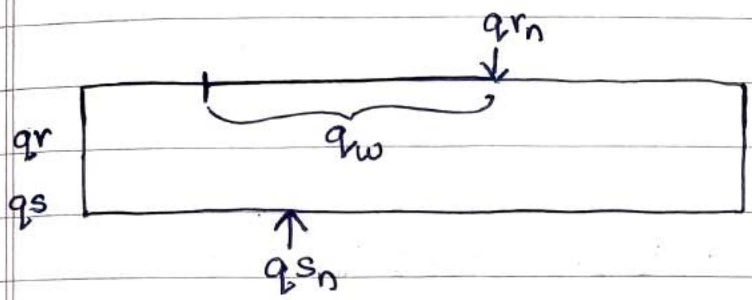
trivial conditions :

$pr_n \leq ps_n$

note: P can send any packet in its window, but for determinism, we can add a state variable P_t (turn) indicating which one to send.

similarly for process Q, we have:

qs, qr, qs_n, qr_n, q_ω. (q_t)



remember, we are acknow. reciept here

window of $Q = [qr_n - q_\omega, qr_n]$

trivial conditions:

$qs_n \leq qr_n$

note: in general,

$$\text{window of } P = \begin{cases} [pr_n - p_\omega, pr_n] & \text{if } ps_n \leq pr_n \\ [pr_n, pr_n + p_\omega] & \text{otherwise} \end{cases}$$

note: with trivial conditions for P and Q are 4 cases:

X 1. $ps_n \leq pr_n$, $qs_n \leq qr_n$ — both acknowledgers (recievers). not valid.

✓ 2. $ps_n \leq pr_n$, $qs_n \geq qr_n$ — p = reciever, q = transmitter.

✓ 3. $ps_n \geq pr_n$, $qs_n \leq qr_n$ — p = transmitter, q = reciever

? 4. $ps_n \geq pr_n$, $qs_n \geq qr_n$ — both transmitters
Cant guarentee eventual delivery for both without acknowledgement).

SYSTEM DEFINITION

$S = \langle X_s, X_s^0, U_s, \xrightarrow{s}, \rangle$

states
initial state
actions
transitions

$X_s = \{ ps[n], pq[n], qr[n],$
$pr[n], qp[n], qs[n] \}$

array of size n (packets)

$X_s^0 = \{ ps = [1, 2, 3, 4, 5, 6, 7]$

$pq(i) = \phi \quad \forall i,$

$qr(i) = \phi \quad \forall i,$

acknowledge-ments

$qs = [-1, -2, -3, -4, -5, -6, -7]$

or it can be just ϕ too

$qp(i) = \phi \quad \forall i,$

$pr(i) = \phi \quad \forall i \}$

$U_s = \{ psend(i), \ precv(i),$

$qsend(i), \ qrecv(i) \}$

$\{ps, pq, \ldots\} \xrightarrow[s]{psend(i)} \{ps', pq', \ldots\}$

iff $pq' = \{i \to ps(i)\} pq$, and

$i \in (pr_n, pr_n + p_\omega]$, and

$i \in [0, n)$.

$\{qs, qp, \ldots\} \xrightarrow[s]{qsend(i)} \{qs, qp', \ldots\}$

iff $qp' = \{i \to qs(i)\} qp$, and

$i \in (qr_n - q_\omega, qr_n]$, and

$i \in [0, n)$, and

$qr(i) \neq \phi$

if you set
$qs(i) = \phi \quad \forall i$
initially

$$\{pq, qr, \cancel{qs} \ldots\} \xrightarrow[s]{qrecv(i)} \{pq, qr', \ldots\}$$

iff $qr' = \{i \rightarrow pq(i)\}\, qr$, and

$pq(i) \neq \phi$, and

$i \in [0, n)$

$$\{qp, pr, \ldots\} \xrightarrow[s]{precv(i)} \{qp, pr', \ldots\}$$

iff $pr' = \{i \rightarrow qp(i)\}\, pr$, and

$qp(i) \neq \phi$, and

$i \in [0, n)$

SAFE DELIVERY

$$\left.\begin{array}{ll} 1. & qr(i) = ps(i) \text{ or } \phi \quad \forall i \\ 2. & pr(i) = qs(i) \text{ or } \phi \quad \forall i \end{array}\right\} \text{required to prove}$$

we can prove this in 4 steps:

$$\left.\begin{array}{ll} 1.1 & pq(i) = ps(i) \text{ or } \phi \quad \forall i \\ 1.2 & qr(i) = pq(i) \text{ or } \phi \quad \forall i \end{array}\right\} \Rightarrow (1).$$

2.1 $qp(i) = qs(i)$ or ϕ $\forall i$
2.2 $pr(i) = qp(i)$ or ϕ $\forall i$ $\Big\} \Rightarrow$ 2

1.1 BASE CASE

$pq(i) = \phi \quad \forall i$ (trivially true)

INDUCTIVE CASE

assume pq satisfies 1.1
the only action U_s that changes pq is $psend(i)$.

$$\{ps, pq, \ldots\} \xrightarrow[s]{psend(i)} \{ps, pq', \ldots\}$$

$$\text{iff } pq' = \{i \to ps(i)\}\, pq, \text{ and}$$

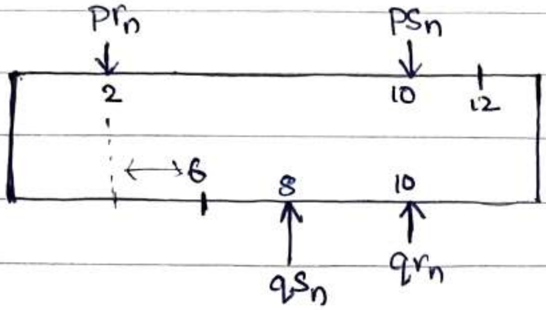
$$i \in (pr_n, pr_n + P_\omega], \text{ and}$$

$$i \in [0, n)$$

$\Rightarrow pq'(i) = ps(i)$ for some i
so 1.1 still holds.

Similarly 1.2, 2.1, 2.2 can be proved.
$\Rightarrow$ our system guarentees safe delivery.
(which means no packet is modified in between or reordered).

EVENTUAL DELIVERY



← think of this as a file

here an example file delivery from P to Q.

P: $pr_n = 2$ $ps_n = 10$ $P_\omega = 10$ window $= (2, 12]$

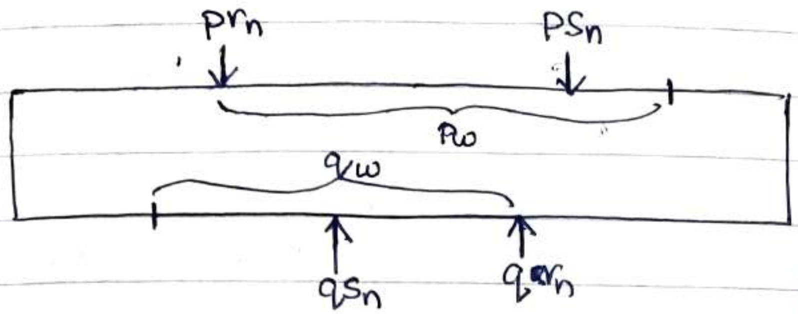
Q: $qs_n = 8$ $qr_n = 10$ $q_\omega = 4$ window $= (6, 10]$

so, P has sent 10 packets
Q has recieved all 10 packets
Q has sent ACK for 8 packets
but P has recieved ACK for only 2 packets

now, what if channel QP drops any packet 2-6. since Q's window is (6, 10] it wont resend those ACK packets.

but, P cant slide its window forward without recieving ACK for packet 2.

so, when $q_\omega < P_\omega$
P and Q can stay stuck on channel loss. (deadlock).



in order to guarentee eventual delivery despite channel loss, we need to ensure that window of Q atleast includes the start of window of P.

P: window $= [pr_n, \; pr_n + p_\omega]$

Q: window $= [qr_n - q_\omega, \; qr_n]$

RTP: $qr_n - q_\omega \leq pr_n$

$pr_n + p_\omega \geq ps_n$ by window def.

$ps_n \geq qr_n$ by trivial cond.

$\Rightarrow pr_n \geq qr_n - p_\omega$

$\Rightarrow pr_n \geq qr_n - q_\omega$ as $q_\omega \geq p_\omega$

$\Rightarrow qr_n - q_\omega \leq pr_n$ (proved) now, for P's window to slide forward by 1 step, P has to recieve packet $qs(pr_n)$ from QP.

since we proved that pr_n lies in window of Q, we can say when Q sends $qs(pr_n)$ P eventually recieves it.

similar argument can be made of sliding of Q's window by 1 step.

since we can guarentee sliding of window by 1 step at each step, and that is equivalent to packet reception and it's acknowledgement, we can thus say that eventual delivery is guarenteed.