ELEC3500

Tutorial – 6 (Solution)

6.0 This is the GBN protocol where the sender has a finite size window. ~~It ca~~ A sender can continuously transmit until its number of un Acked frame becomes N-1 where N is the window size.

ⓐ Suppose that the receiver has received k-1 frames/packets and has Acked all packets up to k-1. If all of these ACK's have been received by the sender, then the sender's window is $[k, k+N-1]$

Current Tx — End of the window

This is a sliding window.

Numerical Example:

$N = 8$, $k = 4$

Allowed seq no: 0, 1, 2, 3, 4, 5, 6, 7

At time t, $k = 4$

Sender's window: [4, 4+8-1]=[4,11]

Suppose that none of the ACK's have been received by the sender. Then all unAcked packets will still be sitting in the window. Assume M packets transmitted in the current window and all packets/frames of the previous window have been Acked.

7 0 1 2 3 4 5 6 7

Acked — Current Tx

Allowed window: $[k-M, N-1]$

Where M is no. of currently unacked frames/packets.

(b) If the receiver is waiting for k packets then it has received and Acked packet k-1 and the N-1 packets before that (previous window). If none of these N ACKs have yet been received by the sender, then senders window is [k, k+N-1], ACK's may still be propagating.

Because the sender has sent packets [k-N, k-1], it must be the case that the sender has received Ack for k-N-1. Once the receiver has sent an ACK for k-N-1 it will not send ACK that is less than k-N-1. Thus the range of inflight ACK values can range from k-N-1 to k-1

6-1 TCP use 32 bit sequence number

So, the window size is $2^{32} = 4,294,967,296$

(a) The sequence no. does not increment by one with each segment. Rather it increments by the number of bytes of data sent. So the size of the MSS is irrelevant. The maximum size of file that can be sent from A to B is simply represented by $2^{32} \approx 4.19$ GB.

Also, the sequen number uses the circular addressing i.e. as the ACK's are coming back on time it reuse the sequence no. as shown below.

0, 1, 2, 3 x

Received ACKs

(b) MSS = 536 bytes

No. of segments $\left\lceil \frac{2^{32}}{536} \right\rceil = 8,012,999$

66 bytes of header is added to each each segment, thus generating a total of 528,857,934 bytes of header.

So, the total no. bytes transmitted to transmit ~~the file~~ all segments $= 2^{32} + 528,857,934$

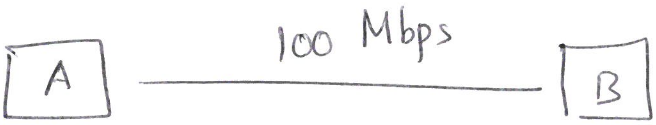
$= 4.824 \times 10^{9}$ bytes.

Assuming a simple transmission mechanism

Time to transmit the file $= \frac{4.824 \times 10^{9} \times 8}{155 \times 10^{6}} = 248.98$ sec

≈ 249 sec

6.2



The TCP sends at a rate of 120 Mbps but the link capacity is only 100 Mbps.

Also, the receiver can only empty its buffer at a maximum rate of 50 Mbps.

In this case the maximum sending rate will be 100 Mbps, and the receiver clearence rate of 50 Mbps.

When the buffer is full, the host B signals host A to stop sending by setting RcvWindow = 0.

The host A ~~send~~ stop sending any more data until it receives a RcvWindow > 0.

Thus the host A send data in the stop and start mode. On average the host A can send at a data rate < 50 Mbps. Rate is less than 50 Mbps due to the signalling requirement.

6-3 RTT measurement

$$DevRTT = (1-\beta) * DevRTT + \beta * |SampleRTT - EstimateRTT|$$

$$EstimateRTT = (1-\alpha) * EstimateRTT + \alpha * SampleRTT$$

$$TimeoutInterval = EstimateRTT + 4 * DevRTT$$

1st SampleRTT = 106 ms, prior EstimateRTT = 100 ms

$\beta = 0.25$, $\alpha = 0.125$ DevRTT = 5 ms

$$DevRTT = (1-0.25) * 5 + 0.25 |106 - 100|$$

$$= 0.75 * 5 + 0.25 * 6 = 3.75 + 1.5 = 5.25 \text{ ms}$$

$$EstimateRTT = (1-0.125) * 100 + 0.125 * 106 = 100.75 \text{ ms}$$

After obtaining second SampleRTT of 120 ms:

DevRTT = 8.75 ms

EstimateRTT = 103.16 ms

TimeoutInterval = 138.16 ms

Similarly calculate others

Last value:

DevRTT = 14.57 ms

EstimateRTT = 106.72 ms

Timeout Int = 165 ms.

5

6-4 (a) TCP ~~is~~ slow start is operating between [1,6] and [23,26]

(b) TCP congestion avoidance is operating in the in the interval
[6,16] and [17,22]

(c) After the 16th transmission round, packet loss is recognised by a triple duplicate ACK. if there was a timeout, the congestion window size would have dropped to 1.

(d) After the 22nd transmission round, segment loss is detected and due to timeout, hence, the congestion window size is set to 1.

(e) The threshold is initially 32, since it is at this window size the slow start stops and congestion avoidance begins.

(f) The threshold is set to half the value of the congestion window when the packet loss is detected. When the loss is detected during transmission round 16, the congestion window is 42. In the ~~congestion~~ 18th round the congestion window is dropped to $21+3=24$ ~~(figure is not very accurate)~~.

(g) The ssthresh (threshold) is set to half the value of the congestion window when the packet loss is detected. When loss is detected in round 22, the cwnd size is 29. Hence, the threshold is 14 (lower floor of 14.5) during the 24th transmission round.

(h)

Tx Round		Packet No.(s) Transmitted		
1	–	1	(1)	→ No. of Packets sent
2	–	2–3,	(2)	
3	–	~~8–15~~ 4–7,	(4)	
4	–	8–15 ~~16–31~~	~~(16)~~ (8)	
5	–	16 – 31	(16)	
6	–	32 – 63	(32)	
7	–	64 – 96	(33)	

6

Packet 70 is sent in the 7th transmission round

(i) The threshold will set to half of the current value of 8 to 4. When the loss occured and the congestion window will set to the new threshold value +3 MSS. Thus the new values of the threshold ~~will be~~ and window will be 4 and 7 respectively.

6-5

(a) It takes 1 RTT to increase ~~Cw~~ cwnd to 6; we use the assumption.

RTT	Cwnd
	6
1	7
2	8
3	9
4	10
5	11
6	12
7	

(b)

$$\text{throughput} = \frac{\sum P_i}{\sum T_i}$$

P = Packets sent in ith ~~[illegible]~~ RTT

T = RTT Values

RTT	No. of ~~Packets~~ MSS Sent
1	5
2	6
3	7
4	8
5	9
6	10

$$\sum P_i = 5+6+7+8+9+10 = 45$$

$$\sum T_i = 6$$

$$\text{throughput} = \frac{45}{6} = 7.5 \text{ MSS/RTT}$$

6-6

(a) Let assume W denote the maximum window size measured in segments.

Thus $$\frac{W * MSS}{RTT} = 10 \text{ Mbps}$$

Packets will be dropped if the maximum sending rate exceeds link capacity.

$$\Rightarrow \frac{W * 1500 * 8}{0.15} = 10 \times 10^6$$

$$\Rightarrow W = \frac{0.15 \times 10 \times 10^6}{1500 \times 8} = 125 \text{ segments}$$

(b) As the cwnd varies from $W/2$ to W, the average window size ~~throughput~~ will be $\lfloor 0.75 W \rfloor = \lfloor 0.75 \times 125 \rfloor$
$= 94$

Thus the average throughput $= \frac{cwnd * MSS}{T}$

$$\Rightarrow \frac{94 * 1500 * 8}{0.15} = 7.52 \text{ Mbps.}$$

(c) Time to reach maxm. window size $= \frac{cwnd}{2} * T$

$\Rightarrow$ $\frac{Cwnd}{2} * T = \frac{94}{2} * 0.15 = 7.05$ sec.

To recover the TCP connection to increase from $\frac{W}{2}$ to W. Consider that window size increases by one in each RTT.

—x—