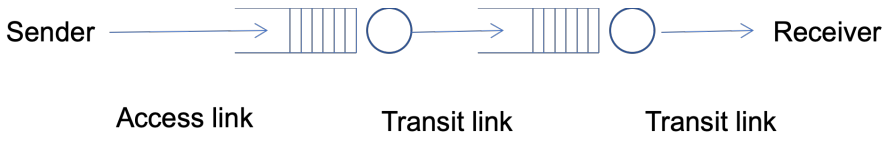
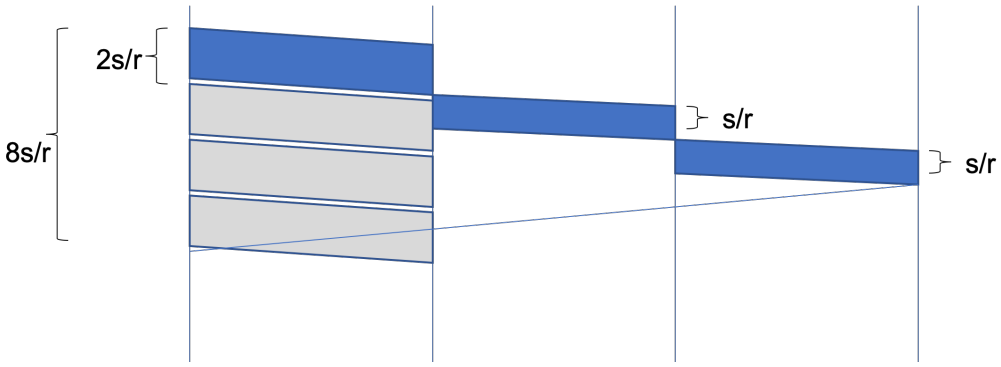
COL334/672: Assignment 4

Dipen Kumar
2018CS50098

1. We have the following topology: a sender communicating to a receiver via a series of two routers. Packets are of size s, the transit links have a transmission rate of r, while the access link operates at half the rate of the transit links. The round trip propagation delay is 4s/r, hence within an RTT of 8s/r the access link can just about support a window size of 4 packets. Ignore acknowledgement sizes.



Access link Transit link Transit link



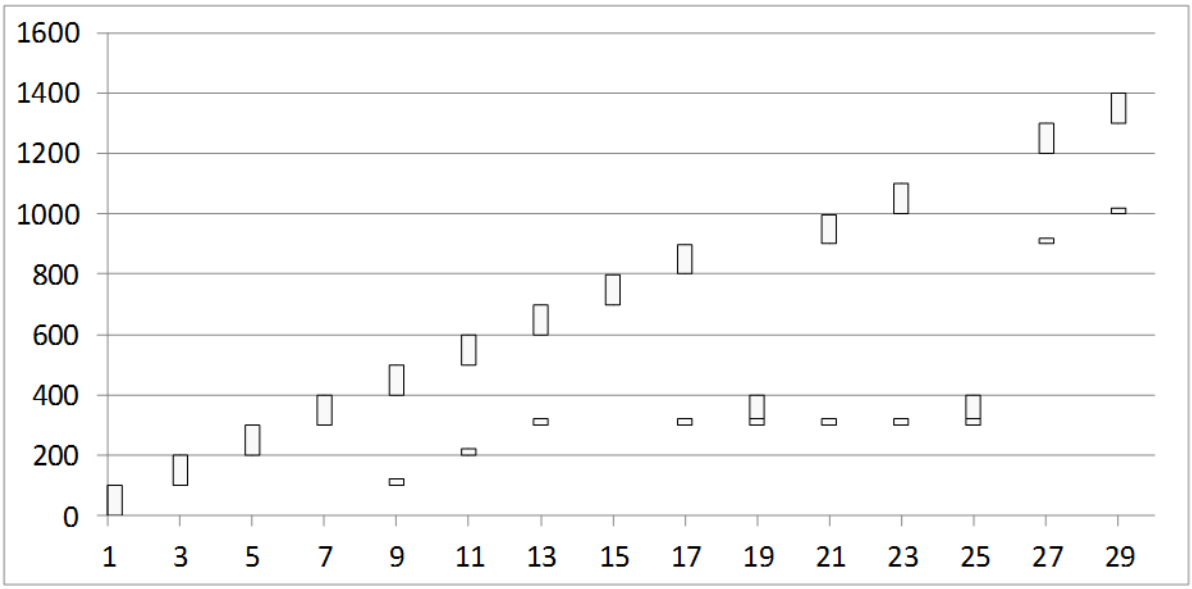
Consider the following packet trace at the sender using a transport protocol similar to TCP. The y-axis indicates sequence numbers in bytes – long vertical rectangles are packets and each packet is 100 bytes long, thus the first packet contains data from sequence number 0 to 99, the second packet from sequence number 100 to 199, etc. The x-axis indicates time in units of s/r. The small stubs are acknowledgement numbers – thus, the stub at time 9 (after one RTT) is the cumulative acknowledgement for the first packet with sequence number 0 to 99, the stub at time unit 11 is the cumulative acknowledgement for the second packet, etc.

The congestion window indicates the maximum number of unacked packets that can be sent. Assume this window size to be fixed and much greater than 4 packets – this implies that the sender will try to push out a packet every 2 time units, which is the maximum transmission rate its access link allows.

Also assume for simplicity that no acknowledgements are lost and no reordering occurs.

Fast retransmission is triggered upon getting triple duplicate acks, taking into account the very first ack as well.

Now explain the packet trace below.



i. Packet 300-399 seems to have been lost. What triggers the retransmission of the lost packet?

⇨ triple duplicate acks referring 300 triggers the retransmission of the lost packet.

ii. The acknowledgement received at time 21 was generated at the receipt of which packet at the receiver?

⇨ 600-699 because at time t = 21-8 = 13, packet 600-699was transmitted.

iii. Why is the acknowledgement at time 21 still referring to the lost packet even though it has been retransmitted?

⇨ Although the lost packet i.e. 300-399 was retransmitted at t-19 it will reach the server at t = 19+6 = 25 and ack will be received to client at t = 19+8 = 27 where we can see it is no longer referring to the lost packet.

iv. Why is the lost packet again retransmitted at time 25?

⇨ Because according to our retransmission triggering criteria – triple duplicate acks referring the lost packet triggers the retransmission of that packet.

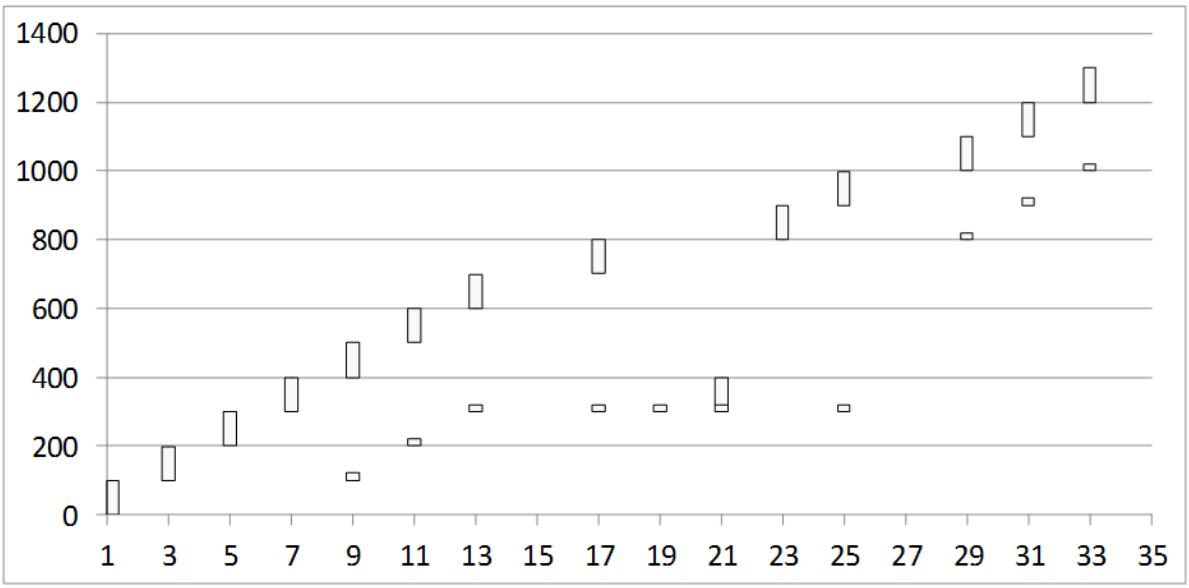
v. Why is there a sudden jump in the acknowledgement number at time 27? The acknowledgement was generated at the receipt of which packet?

⇨ Earlier the acknowledgement was referring to the lost packet but once it was retransmitted at t = 19 and received by the server at t = 25, the client finally gets back its updated acknowledgment at t = 27 referring to the next immediate lost packet that in this case is 900-999 since every packet from 0 to 899 was received by server at t = 25

2. In another variant, the initial congestion window size is started at 4 packets, and the window is incremented fractionally by (1 / int(current window size)) upon receiving an acknowledgement. Thus, a window size of 4 will increment to 5 after having received 4 acknowledgements (4.25 after the first ack, 4.5 after the second ack, 4.75 after the third, and 5 after the fourth ack). Note that packets are dispatched only if they can be accommodated fully within the window, ie. even with a window of 4.75 only four outstanding packets will be allowed.

   The window size also reduces to half upon receiving triple duplicate acknowledgements which is seen as an evidence of packet loss. Note that receipt of the third dup ack will not increment the window, ie. if the window is 5 when the third dup ack arrives, it will just be reduced to 2.5, and not add another 1 / int(2.5) increment for this ack as is done for other acks. Also note that the event of a triple dup ack is also interpreted as a loss, and hence the number of outstanding packets will be assumed to be one less than what was it estimated to be earlier. This is almost identical to TCP operations in the congestion avoidance phase.

   Answer the following questions. Hint: Maintain two variables for congestion window and the outstanding data to understand what is happening.



i. What is the window size at time 17?

⇨ Window size till t < 9 remains 4 (as per question it starts with 4)
At t = 9, window size += 1/int(4.00) => window size = 4.25
At t = 11, window size += 1/int(4.25) => window size = 4.50
At t = 13, window size += 1/int(4.50) => window size = 4.75
At t = 17, window size += 1/int(4.75) => window size = 5.00

ii. What is the window size at time 19? Why is no packet pushed out at time 19?

⇨ window size reduces to half upon receiving triple duplicate acknowledgements which is seen as an evidence of packet loss which means at t = 19, window size = window size/2 = 2.5
At t = 19 window should have 3 packets but since this is a case of a triple dup ack which is interpreted as a loss, and hence the number of outstanding packets will be assumed to be one less than what was it estimated to be earlier which means the number of outstanding packets in the window is 2. We can't push any packet at t = 19 because 2 + 1 > 2.5

iii. What is the window size at time 21? What is the outstanding data estimated by the sender at time 21?

⇨ At t = 21, window size += 1/int(2.5) => window size = 3
the outstanding data estimated by the sender at time 21 is size of a packet because at t = 19 we had 2 packets in window and now at t = 21 we received a ack that implies number of unacked packed in the window should decrease by one. At t = 19 we didn't pushed any packed in window. At t = 21 we can push since we have only one packet in the window size of 3 which will finally make it have 2 outstanding packets in the window.

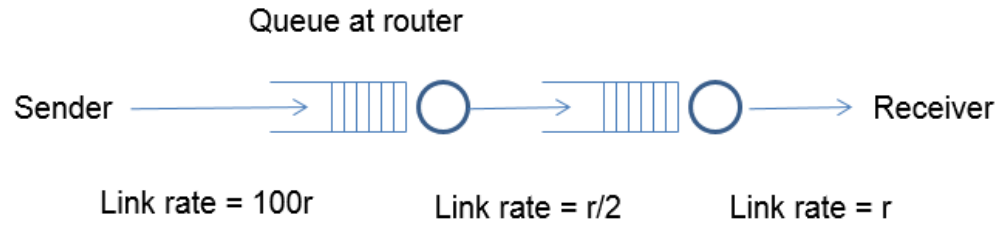
iv. Why is a packet pushed out at time 23 even though no ack is received at that time?

⇨ Because as stated in the previous question we resulted with 2 outstanding packets in the window size of 3 giving room for one more packet which at t = 23 was used to push a packet even though no ack is received at that time.

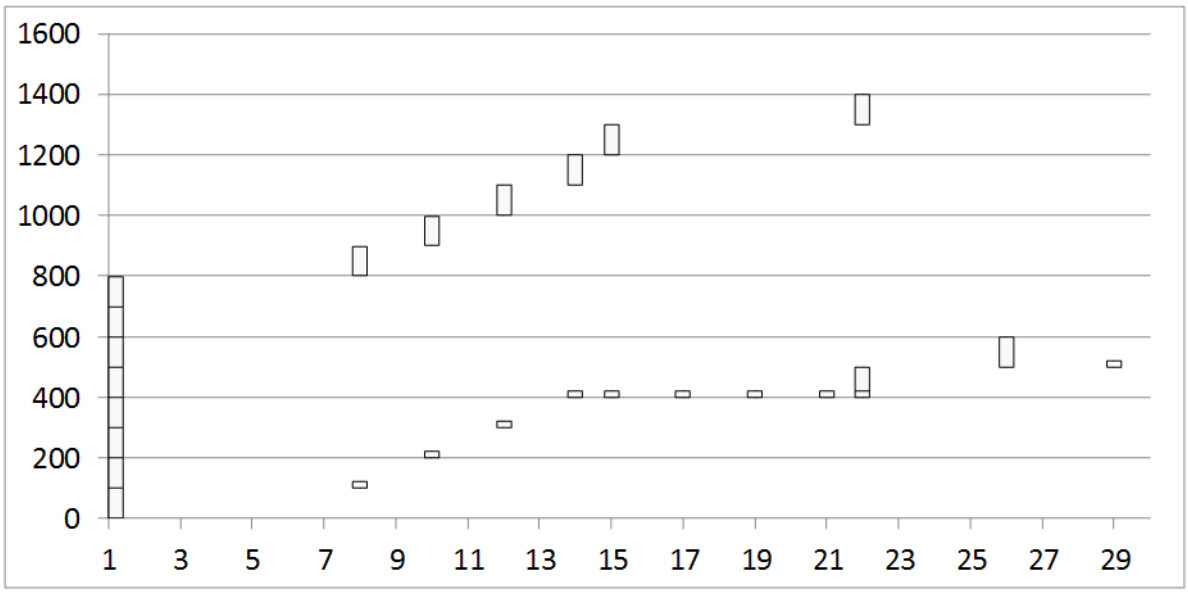
v. What is the window size at time 31?

⇨ As discussed at t = 21, window size = 3
At t = 25, window size += 1/int(3.00) => window size = 3.33
At t = 29, window size += 1/int(3.33) => window size = 3.66
At t = 31, window size += 1/int(3.66) => window size = 4.00

3. Now consider a different scenario where the access link is very fast but the next link is slower.



Assume an initial window size of 8 this time. As in the previous question, the window size is reduced to half upon receiving triple duplicate acknowledgements, and it is incremented by 1 / int(congestion window) upon receiving an acknowledgment. A timeout occurs if the last unacked packet goes unacknowledged for more than 25 time units. The buffer size at the first router is limited, and it follows a drop-tail policy. Assume that all packet losses happen due to buffer overflow at the first router. Answer the following questions:



i. What is the RTT in this case?

⇨ Transmission delay + propagation delay
= s/100r + 2s/r + s/r + 4s/r
= 7.01s/r

ii. Can you infer the buffer size at the first router? How?

⇨ Yes, I can infer the buffer size of first router by observing the triggered retransmitted lost packet which is 400-499. Hence the buffer size is 4 i.e. 5th packet was lost due to buffer overflow at the first router.

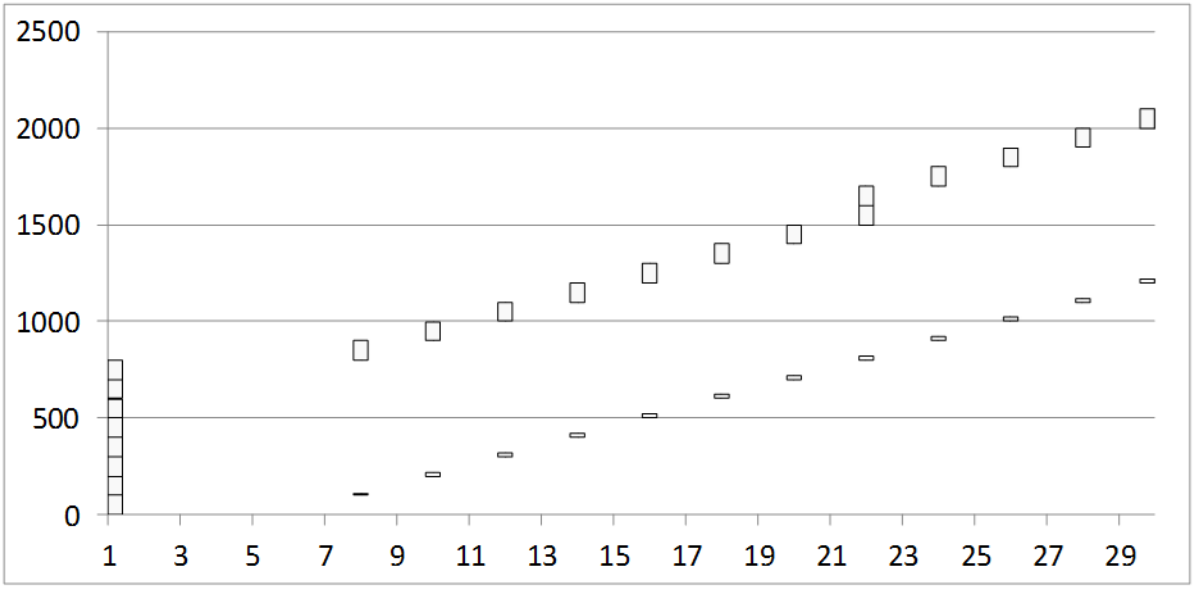
iii. Why is there no retransmission at time 17 despite a triple dup ack? Why does this retransmission happen later at time 22? Why do two packets get fired off at time 22?

⇨ Because the window is full w.r.t the size at t = 17 that is –
At t = 15 window size = 8 + 5/8 = 8.625 => at t = 17, window size = window size/2 = 4.3125 and 7 outstanding packets but wait, this is case of packet loss hence outstanding packets = 6
Moving further,
at t = 19, window size is 4.5625 and outstanding packets = 5
at t = 21, window size is 4.8125 and outstanding packets = 4
at t = 22, window size is 5.0625 and outstanding packets = 3
hence this retransmission happen later at t = 22 and two packets get fired off since 3 + 2 <= 5.0625

iv. When did a timeout occur? Which packet gets timed out?

⇨ Time out occur at t = 26 because and packet 500-599 gets timed out because it was retransmitted after 25 time units without being triggered by triple dup ack which means it was timed out and needed to be retransmitted.

4. Consider now a scenario where the buffer size at the first router is very large so that no drops occur. Answer the following questions:



i. What is the round trip time clocked for the first packet? For the fifth packet? For the eighth packet?

⇨ 7, 15 and 21 time units for the first, fifth and eighth packet respectively from the above graph. packets were received in order and acks were send in order.

ii. When is the window size increased to 9?

⇨ At time t = 22, window size = 8 + 8*(1/8) = 9 when given initial window size of 8

iii. Since the buffers are assumed to be large enough, there will be no drops and the window size will keep increasing each time a complete round of acknowledgments for the current window are received. What is the problem with such a network?

⇨ Window size will be ever increasing. Every time it is increased to the next integer it will allow 2 packets to be pushed in the window and hence the second packet and all the other future packets to send will experience extra delay for acknowledgments compared when window size was previous integer. This is because buffer size will be ever increasing which is because access link is faster than transit link.

iv. Suggest a method to trigger a window size reduction in such a scenario, without witnessing any loss events.

⇨ If we want to avoid any loss events which in this case will be loss due to time out, we need to set the upper limit on the window size which it can increase to max and stop increasing further.
Let's find the upper limit – I claim that when I receive an ack, the next three acks will be after 2s/r, 4s/r and 6s/r respectively and 3rd one will be in router1 buffer and half transmitted because not it will take s/r + s/r + 4s/r = 6s/r. now we can at keep buffer length at max 10 because s/r + (2s/r)*8 + 7s/r = 24s/r < 25s/r and hence maximum window length = 3 + 9 = 12. But initially it should not start more than 9 because (2s/r)*8 + (7.01s/r) = 23.01s/r < 25s/r
So, whenever window size becomes equal to 13 than bring it back to 12, in this question it is given that it starts with 8 that is less than equal to 9. Hence good.