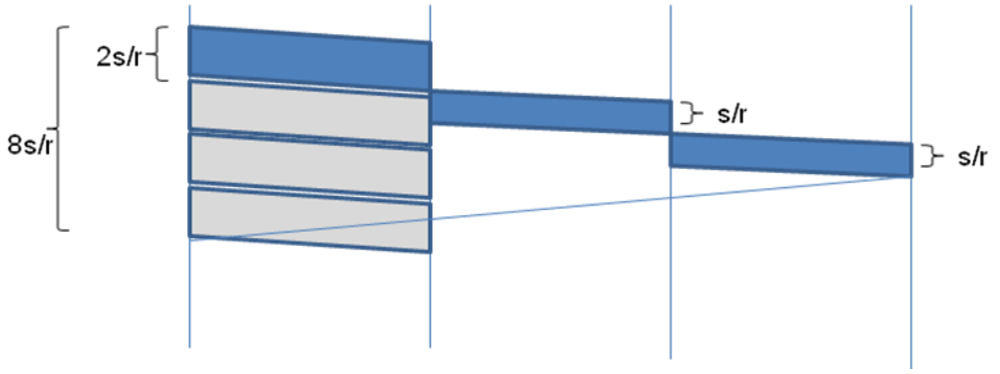
COL334/672: Assignment 4

Jai Javeria 2018CS10340

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.



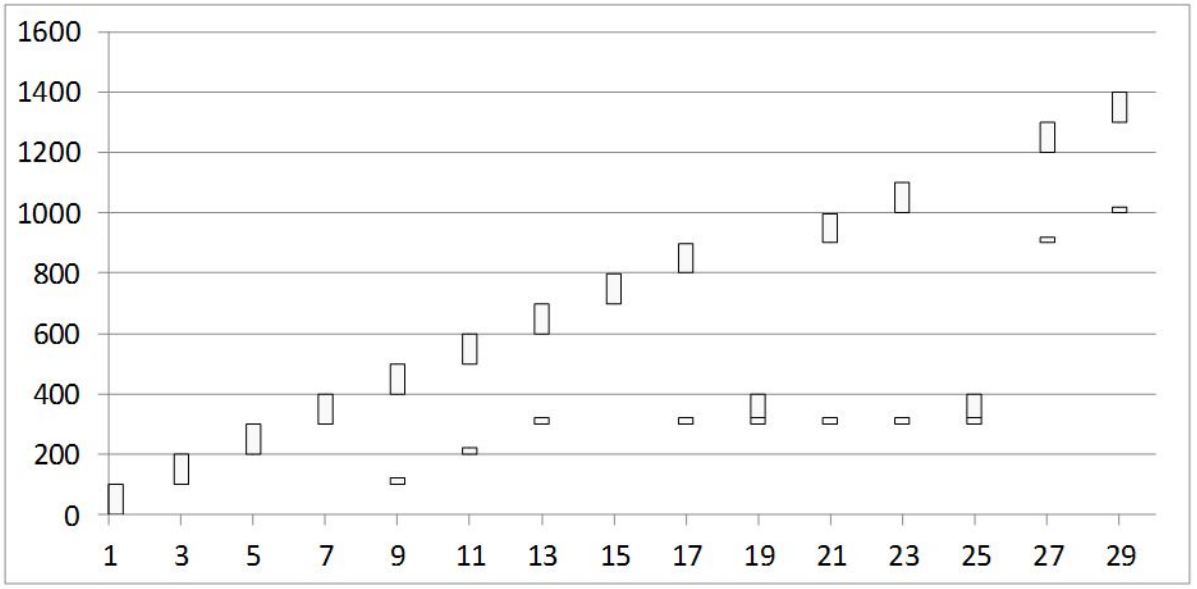
Consider the following packet trace at the sender using a transport protocol similar to TCP. The y-axis indicates sequence numbers in bytes – long vertical rectangles are packets and each packet is 100 bytes long, thus the first packet contains data from sequence number 0 to 99, the second packet from 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?
A. The triple dup Ack at time 19 triggers the retransmission of packet 300-399.

ii. The acknowledgement received at time 21 was generated at the receipt of which packet at the receiver?
A. It was generated at the receipt of of the packet sent at time 13 i.e. 600-699

iii. Why is the acknowledgement at time 21 still referring to the lost packet even though it has been retransmitted?
A. The acknowledgement received at 21 was generated by the receiver around time 17 due to the latency of receiving packet 13. The re transmission did not happen at that time, plus even if it would have just happened the receiver might not know because of the latency of 4 time units.

iv. Why is the lost packet again retransmitted at time 25?
A. The retransmission happens due to the fact that the sender has received 3 dup acks.

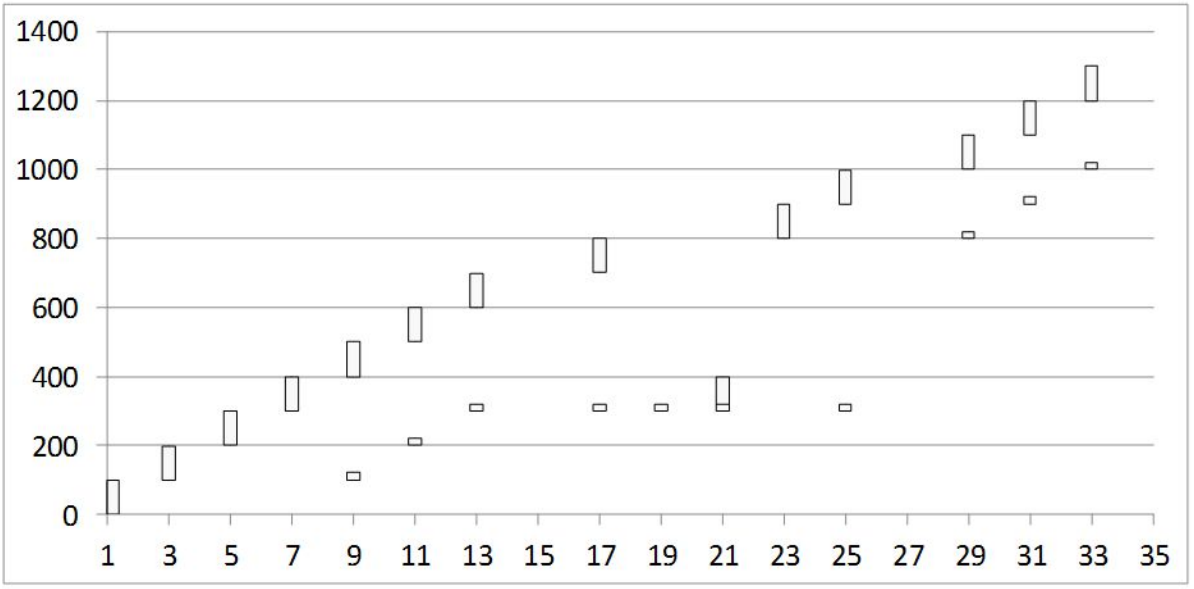
v. Why is there a sudden jump in the acknowledgement number at time 27? The acknowledgement was generated at the receipt of which packet?
A. The acknowledgement was generated on the receipt of the packet 300-399 transmitted at time 19. There is a sudden jump in the ack number because all the bytes 0-899 are now received successfully, because the lost packet 300-399 is also received.

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 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?
A. The window size at 17 has been incremented to 5.

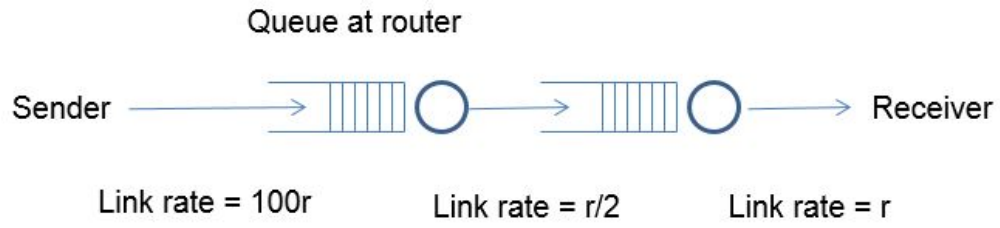
ii. What is the window size at time 19? Why is no packet pushed out at time 19?
A. Window size at time 19 is 2.5. no packet is pushed at 19 is because packet 600-699 and 700-799 are unacknowledged meaning the window size is already full and no other packet can be accomodated.

iii. What is the window size at time 21? What is the outstanding data estimated by the sender at time 21?
A. The window size at time 21 is 3. Outstanding data consists of packets 300-399 and 700-799.

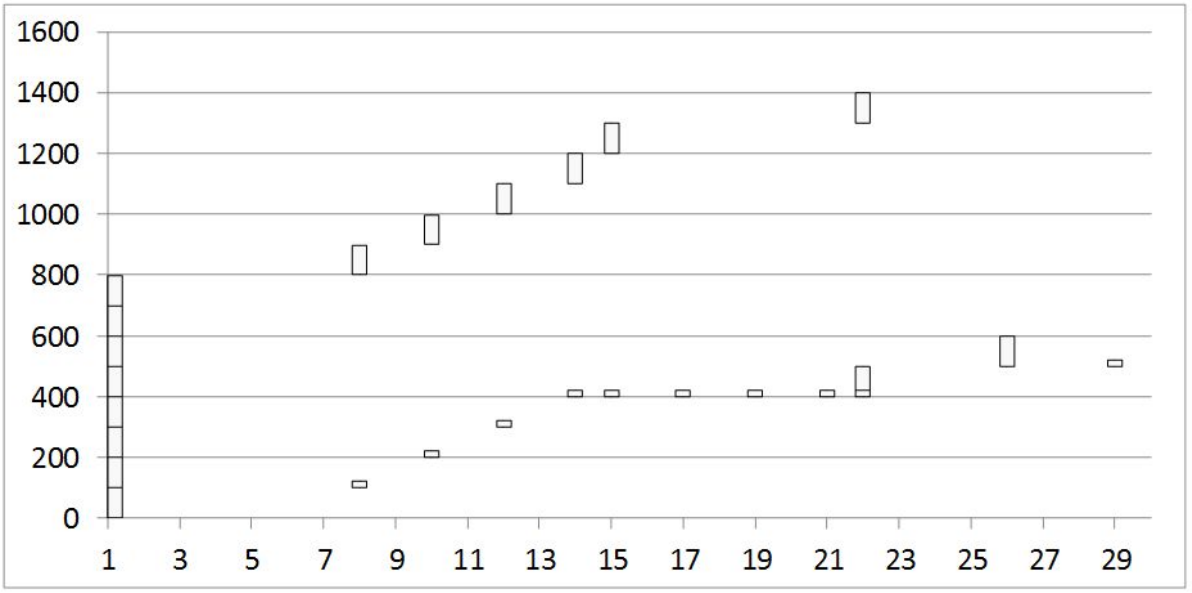
iv. Why is a packet pushed out at time 23 even though no ack is received at that time?
A. Because unack packets are only 2 (300-399 and 700-799), while the window size is 3, so one more packet can be transmitted.

v. What is the window size at time 31?
A. The window size is 4.

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?
A. The RTT in this case is 2*(1/100+2+1)*s/r=6.02*s/r without any queuing delay to consider. That is variable for the packets.

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

A. Given that all the packet losses happen due to the buffer overflow at router 1, we can calculate the size of the queue to be of 400 bytes, since we can see packet 400-499 is lost.

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?

A. Window size at time 17 becomes equal to 4+5/16. There are already 6 unacknowledged packets(since 1 got lost,6 acks, total 13 sent), so there is no transmission.

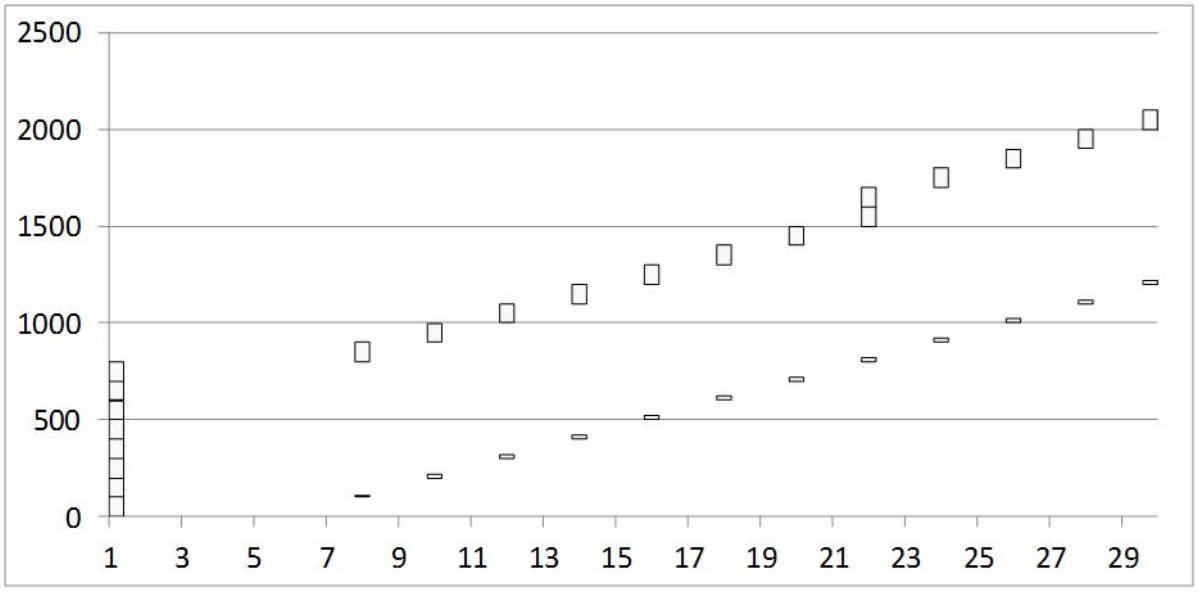
At 22, unacknowledged packets go down to 3 and window size increases to 4.75+5/16>5. The window size is not reduced because the congestion window is already halved from the triple dup ack received at time 17. Thus the retransmission happens there.

Since the difference between the window size and un ack packets is 2, thus 2 packets were sent.

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

A. Timeout occurs at time 26. Packet 500-599 gets timed out.

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?

A.

1. first packet: around 7 time units.
2. Fifth packet: around 15 time units.
3. Eight packet: around 21 time units.

ii. When is the window size increased to 9?
A. window size increases to 9 at time 22.

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?
A. Since the buffers are assumed to be large enough, to make them practically would lead to huge costs in setting up these resources.
Also as the queues start increasing, the RTT would start increasing meaning that acks would take a long time to be received and can be out of the timeout window. Thus this would lead to redundant restramissions of those packets whose acks are received after the timeout.

iv. Suggest a method to trigger a window size reduction in such a scenario, without witnessing any loss events.
A. As said in the previous answer, we would start getting a lot of timeouts as the queue lengths start to increase. Thus we could look into the number of timeouts and try to control the window size accordingly.
One way could be to make “triple dup timeouts”, to reduce the window size by half when we receive timeouts thrice for the same packet even after transmissions.
Another way could be to see the total timeouts across various packets and reduce the window size by half if the number of timeouts increases to some threshold. What the threshold would be would have to be experimented with.