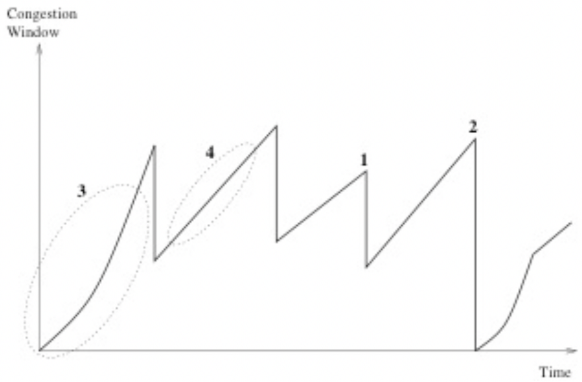
Revision of TCP Congestion Control (NO SUBMISSION REQUIRED)

The following question is included so that you can revise the TCP congestion control algorithm. It is recommended that students have a short discussion with their tutors at the start of the lab to go over this question and TCP congestion control in general. Students are strongly encouraged to participate and ask questions. You are not required to submit the answer to this question in your lab report. I suggest you use the lab's first 15-20 minutes to go over this.

The graph in the figure below shows how the congestion window of a TCP Reno connection changes over time. It is drawn roughly to scale. Certain parts of the graph that are of extra interest have been marked with numbers. With the help of the graph, answer the following questions:



Question 1. Name the loss events that occur at 1 and 2. Explain why the congestion window is changed differently in those two cases.

Question 2. What phase of the TCP congestion control algorithm coincides with the circled segment marked by 3?

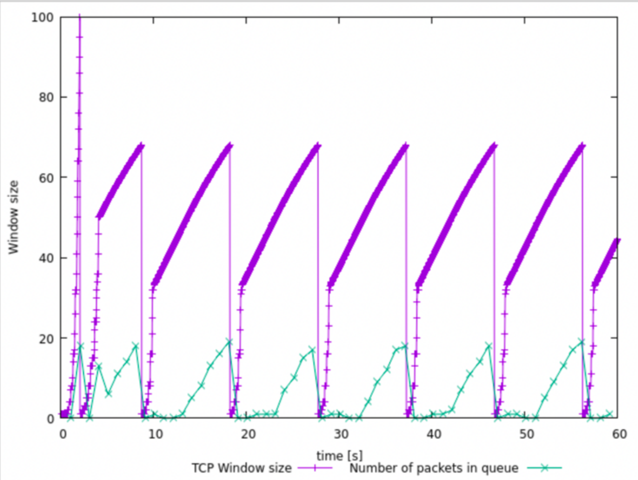
Question 3. What phase of the TCP congestion control algorithm coincides with the circled segment marked by 4?

Question 4: Why is the congestion window increased more rapidly at 3 than at 4?

Question 5: Can you precisely explain what happens to the window after 2?

Q1:
1-> duplicate ACK -> CWND = CWND/2
2-> timeout -> CWND = 1
Q2:
Slow Start (Bandwidth discovery)
Q3:
AIMD
Q4:
As 3 is the slow start phrase, the CWND increase exponentially, and 4 is in AIMD phrase which increase linearly.
Q5.
As it reach the timer, it go back to CWND = IMSS. And it slows start again until it reach the previous CWND/2.

Exercise 1



(a) In this case, what is the maximum size of the congestion window that the TCP flow reaches?

The maximum size of congestion window is 100.

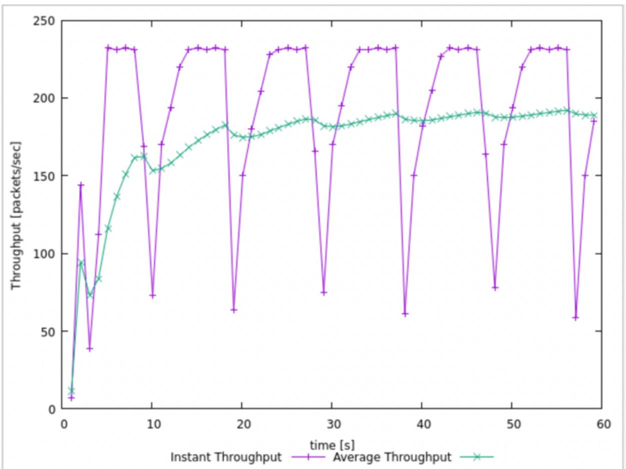
(b) What does the TCP flow do when the congestion window reaches this value? Why?

It reaches the timeout or receives dup ACK which cause the CWND = 1. (TCP-Tahoe) . It reset the window size.

(c) What happens next?

The CWND will drop to 1 and exponentially increase to 50 (the half of the previous window size), then it start the congestion avoidance phrase which increase linearly until loss.

Question 2: From the simulation script we used, we know that the packet's payload is 500 Bytes. Keep in mind that the size of the IP and TCP headers is 20 Bytes each. Neglect any other headers. What is the average throughput of TCP in this case? (both in number of packets per second and bps)



The average throughput is 190.667 packet per second.

Packet length = (500+20+20) bytes => 540*8 (convert into bit)*190 = 820800 bit per second

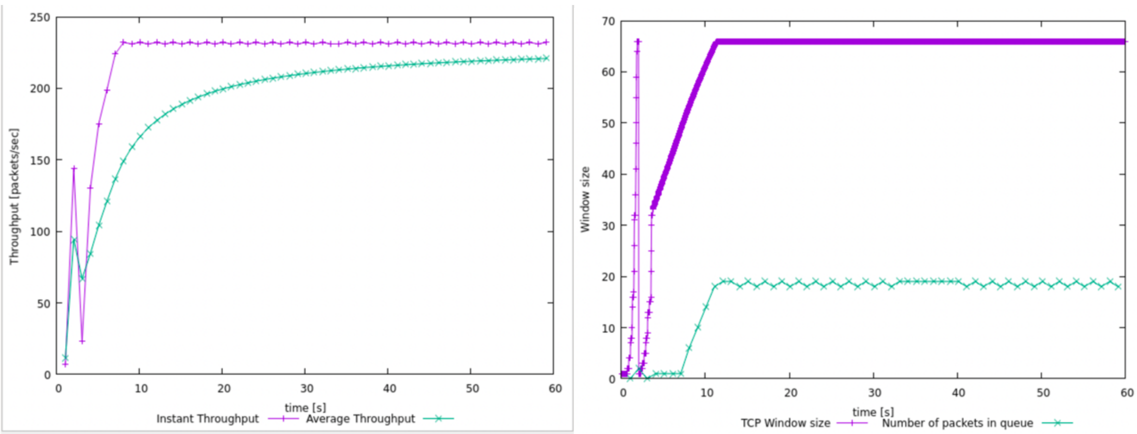
Question 3: Rerun the above script, each time with different values for the max congestion window size but the same RTT (i.e. 100ms). How does TCP respond to the variation of this parameter? Find the value of the maximum congestion window at which TCP stops oscillating (i.e., does not move up and down again) to reach a stable behaviour. What is the average throughput (in packets and bps) at this point? How does the actual average throughput compare to the link capacity (1Mbps)?

We window size decreases to 66, it shows only one oscillating.
If the max initial window size is lower than 51, then there will be no oscillating.

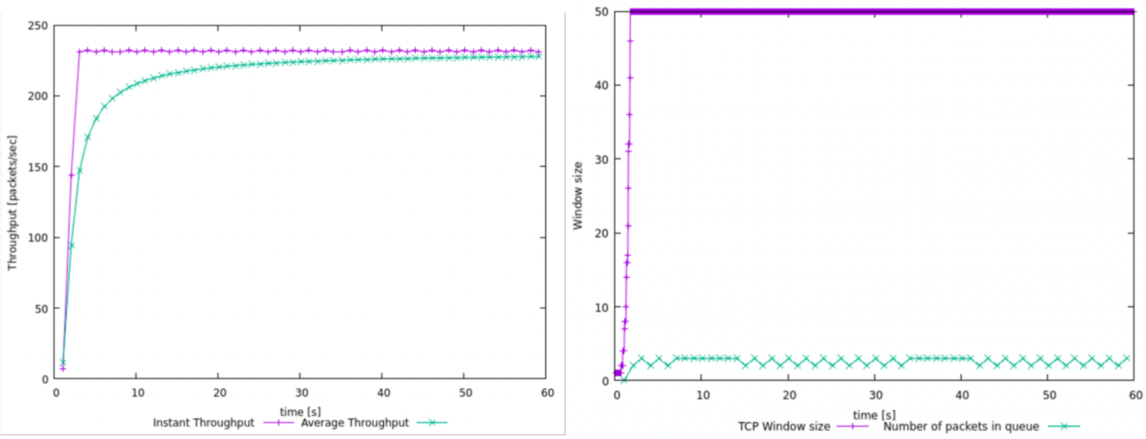
The average throughput is 220 packet per second.
Calculate the throughput: 220*540 *8=950.400 kbps

The link capacity = 950.4 / 1000000 *100 = 0.9539%

ns tpWindow.tcl 66 100ms -> 66 buffer size

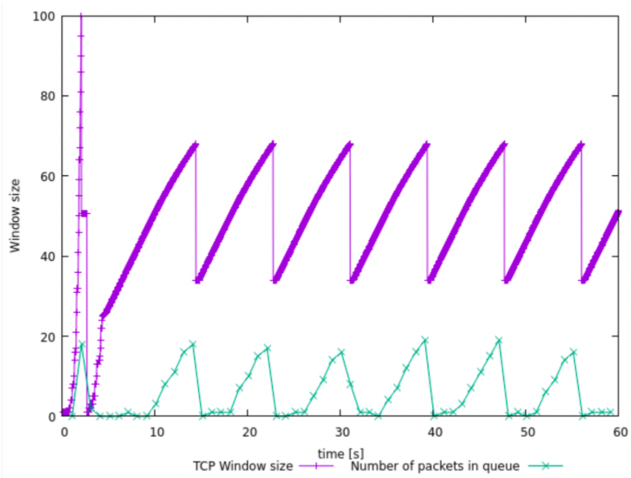


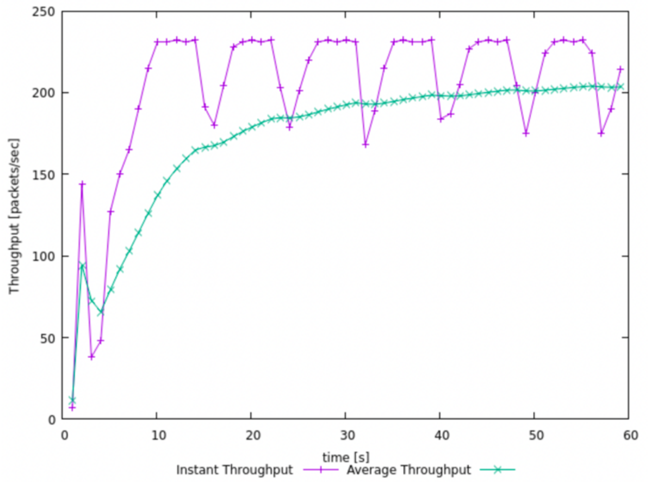
ns tpWindow.tcl 50 100ms



Question 4: Repeat the steps outlined in Questions 1 and 2 (NOT Question 3) but for TCP Reno. Compare the graphs for the two implementations and explain the differences. (Hint: compare the number of times the congestion window returns to zero in each case). How does the average throughput differ in both implementations?

The window size only decreases to zero after slow start phase for TCP Reno.
The average throughput of TCP Reno (203) is higher than the TCP Tahoe (190).

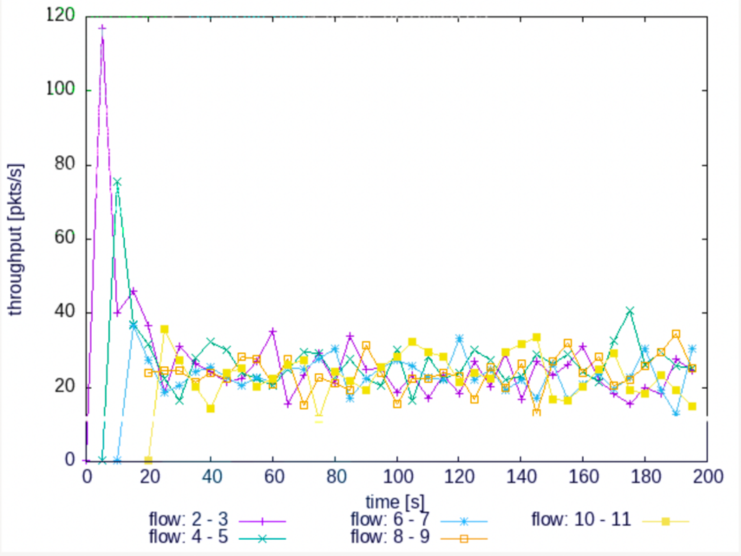




Exercise 2: Flow Fairness with TCP

Question 1: Does each flow get an equal share of the capacity of the common link (i.e., is TCP fair)? Explain which observations lead you to this conclusion.

Yes, each flow gets equal share. As each flow join the common link, all of them shares the throughput between 20-40.

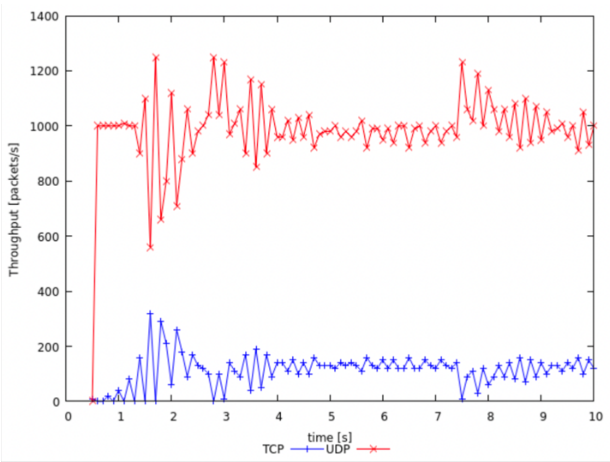


Question 2. What happens to the throughput of the pre-existing TCP flows when a new flow is created? Explain the mechanisms of TCP which contribute to this behaviour. Argue whether you consider this behaviour to be fair or unfair.

The throughput of pre-existing TCP flow drops when a new flow is created because it needs to share with the new one. All flows adapt the network which the behaviour is fair.

Exercise 3: TCP competing with UDP

Question 1: How do you expect the TCP flow and the UDP flow to behave if the link's capacity is 5 Mbps?



TCP: Blue, UDP: Red

Question 2: Why does one flow achieve higher throughput than the other? Try to explain what mechanisms force the two flows to stabilise to the observed throughput.

As TCP applies congestion control but UDP does not, TCP is much more reliable while transmitting the data. And UDP applies best-effort datagram.

Question 3: List the advantages and the disadvantages of using UDP instead of TCP for a file transfer, when our connection has to compete with other flows for the same link. What would happen if everybody started using UDP instead of TCP for that same reason?

The advantage of UDP: higher average throughput/ smaller packet size

The disadvantage of UDP : No congestion controls /Packets arrival out of order

If everyone started to use UDP rather than TCP which may cause many packets loss, performance suffer and network congested, and difficult to detect corrupted packets.