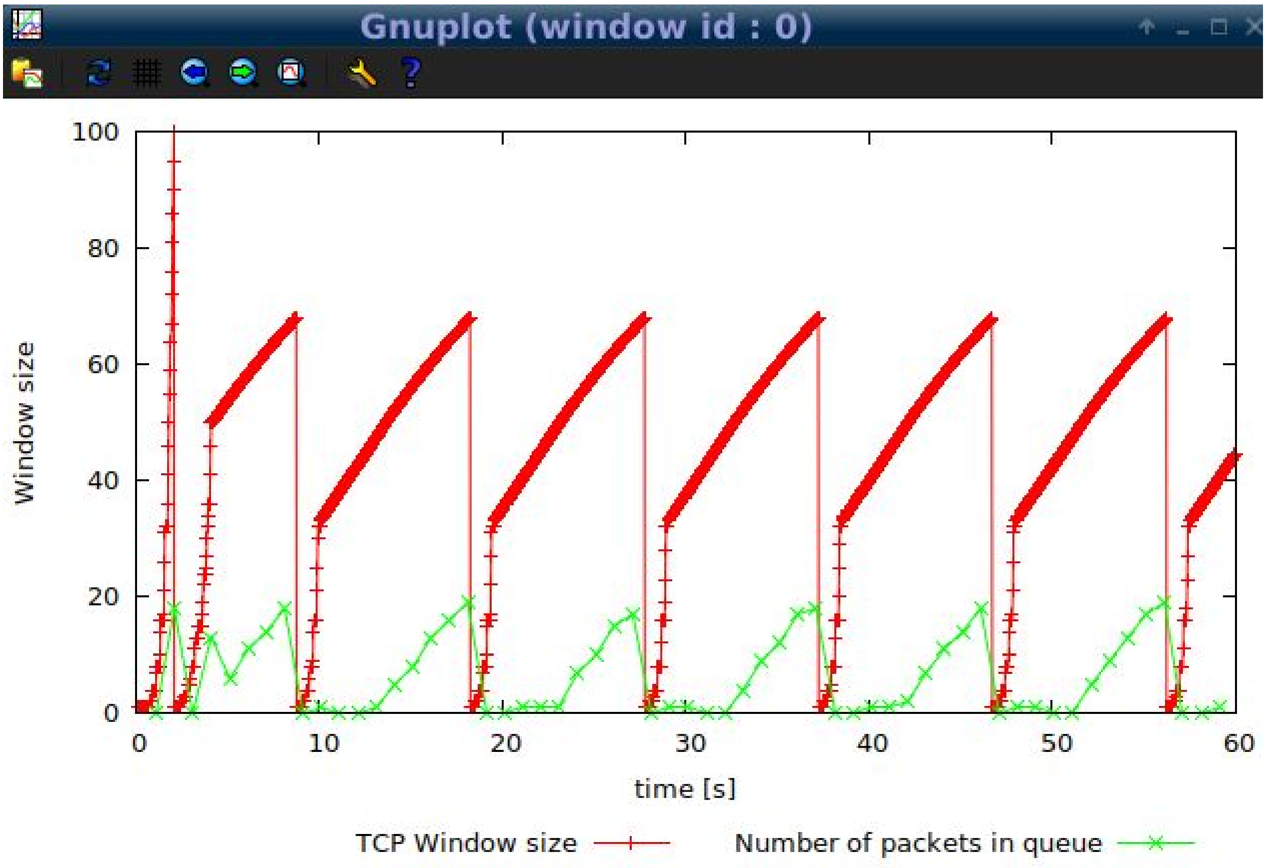
Lab5

Exercise 1: Understanding TCP Congestion Control using ns-2

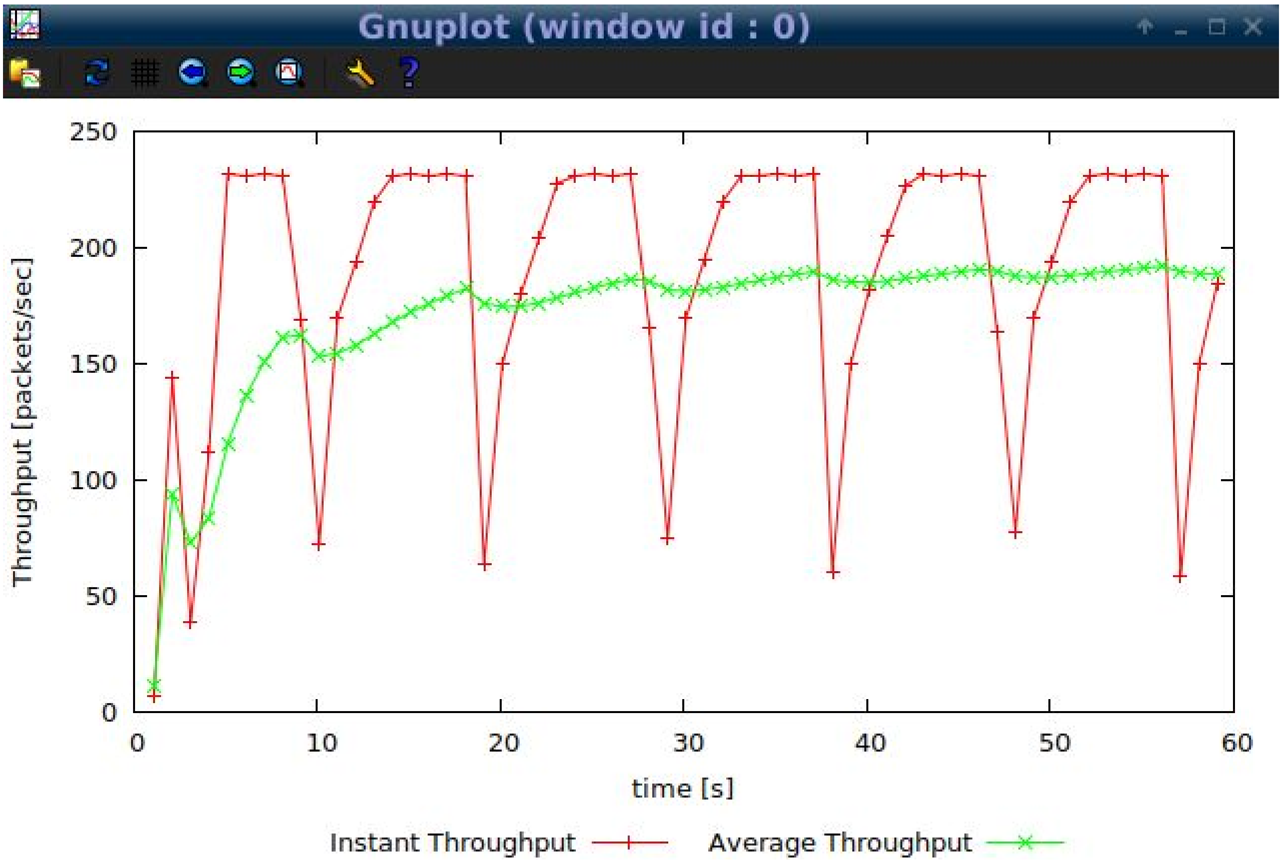
Question 1: What is the maximum size of the congestion window that the TCP flow reaches in this case? What does the TCP flow do when the congestion window reaches this value? Why? What happens next? Include the graph in your submission report.

MAX CWND = 100 packets, in SS (slow start).
The queue is full because the size of queue is 20 packets. When reaching that value, some packets are dropped and the ssthresh set to CWND/2, which is 50 packets.
It will go to the SS (slow start). And then reach to the CA (congestion avoidance) again. It will oscillate between the SS phase and the CA phase.



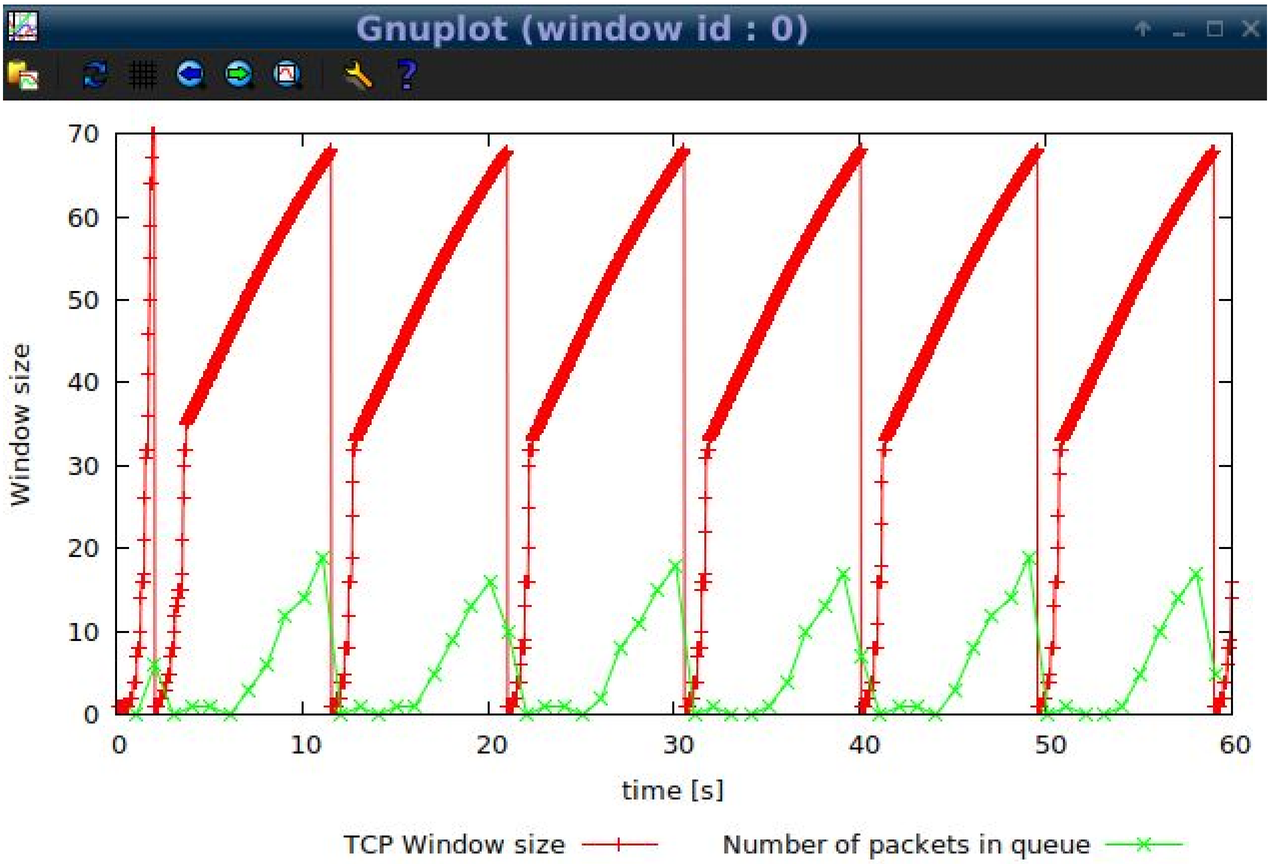
Question 2: From the simulation script we used, we know that the payload of the packet is 500 Bytes. Keep in mind that the size of the IP and TCP headers is 20 Bytes, each. Neglect any other headers. What is the average throughput of TCP in this case? (both in number of packets per second and bps)

The average throughput in packets is around 185 packet/sec.
IP and TCP headers are 20 + 20 = 40 bytes. And the payload is 500 bytes.
Therefore, the average throughput in bps is (40 + 500) *8 * 185 = 799,200 bps.

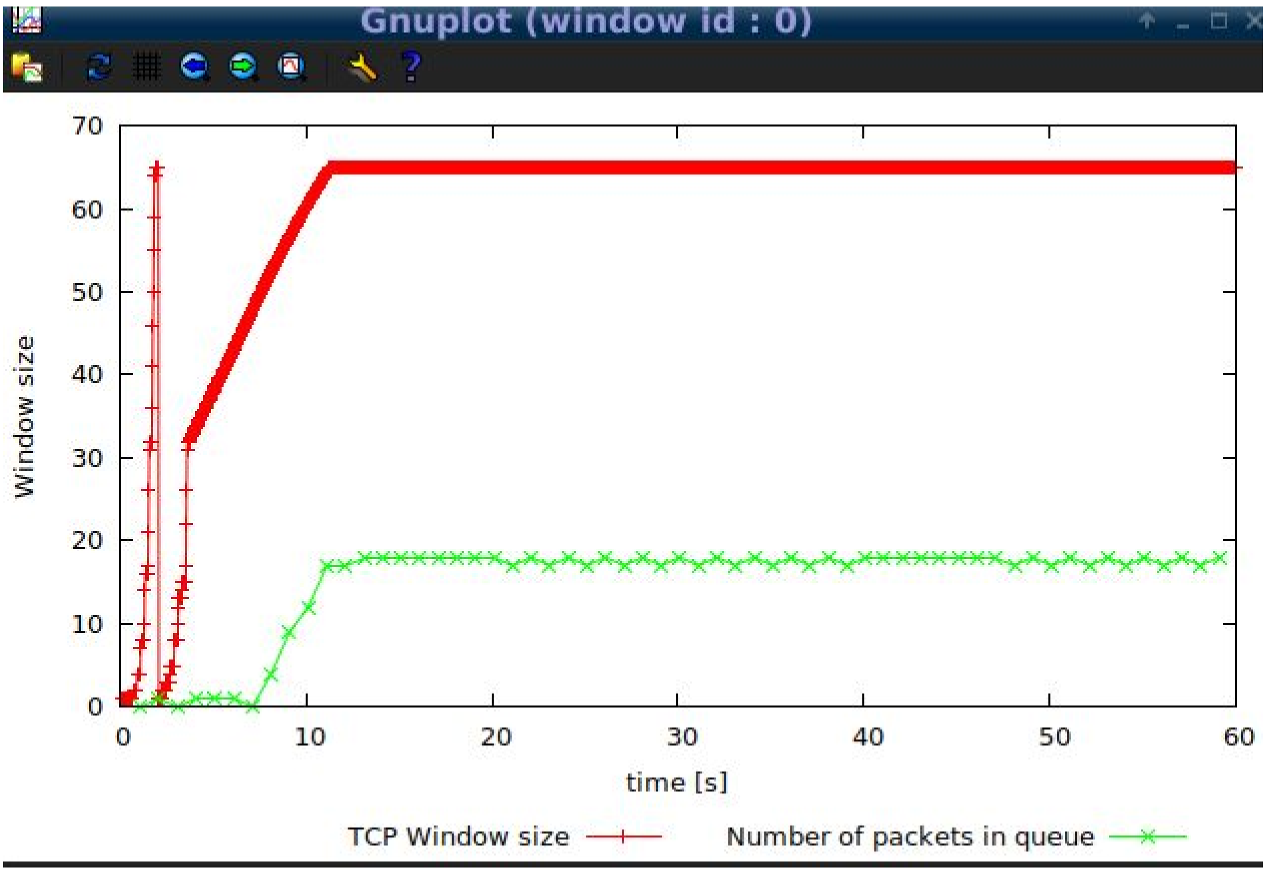


Question 3: Rerun the above script, each time with different values for the max congestion window size but the same RTT (i.e. 100ms). How does TCP respond to the variation of this parameter? Find the value of the maximum congestion window at which TCP stops oscillating (i.e., does not move up and down 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)?

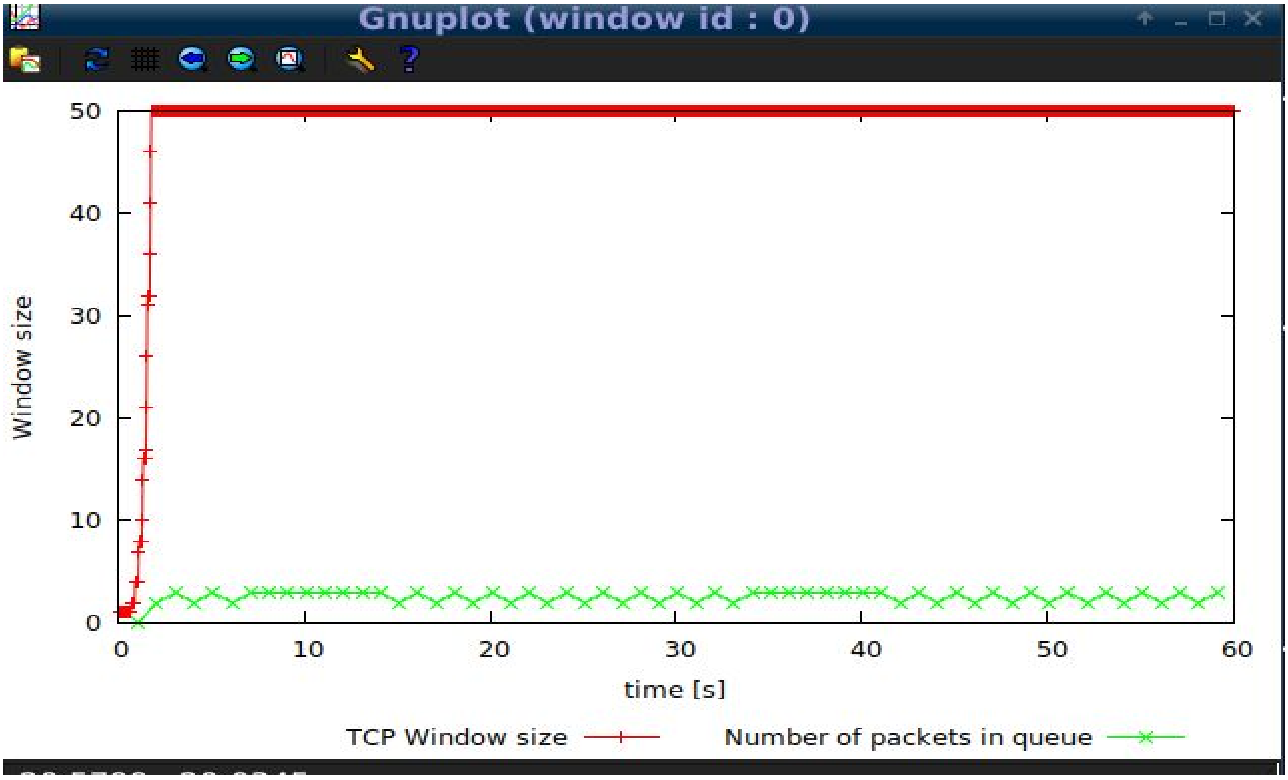
Initial max CWND: 70 (figure 1 below)



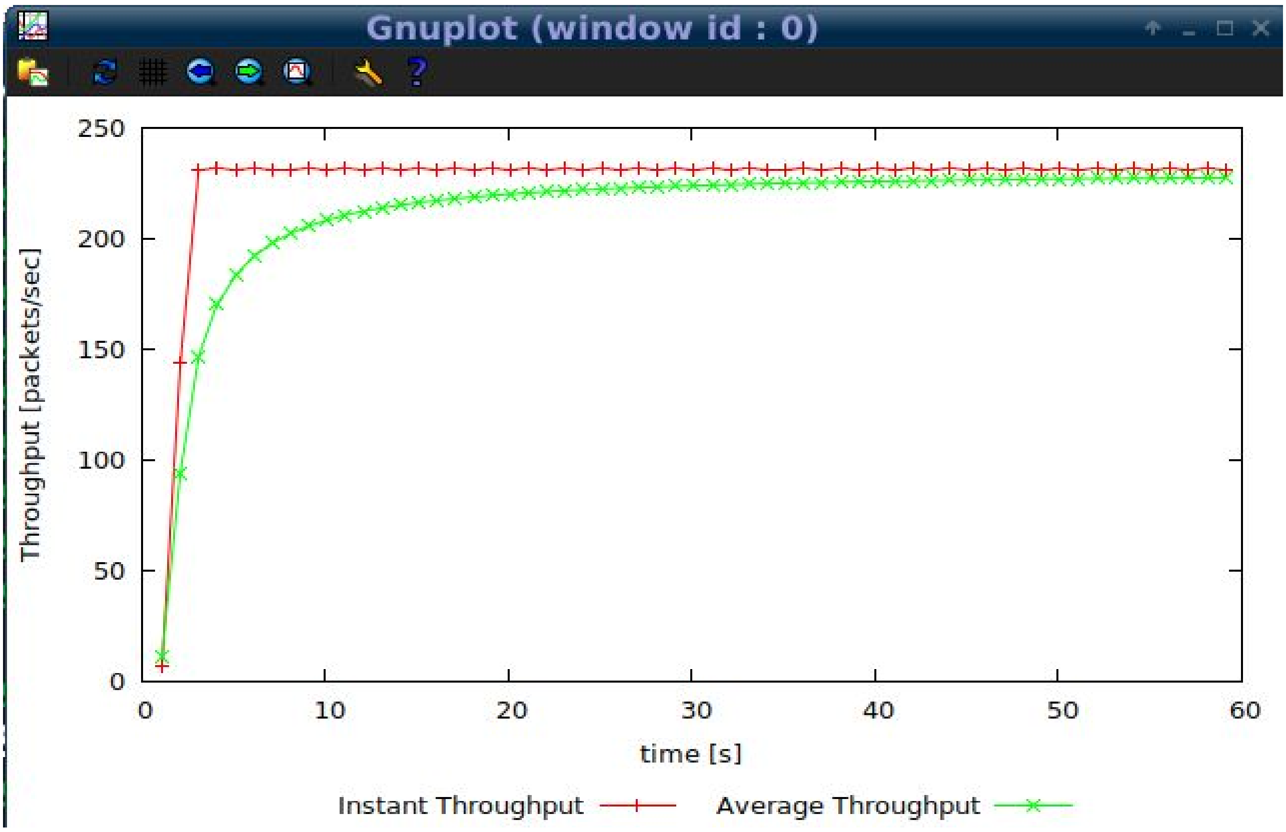
Initial max CWND: 65 (figure 2 below)



Initial max CWND: 50 (figure 3 below)



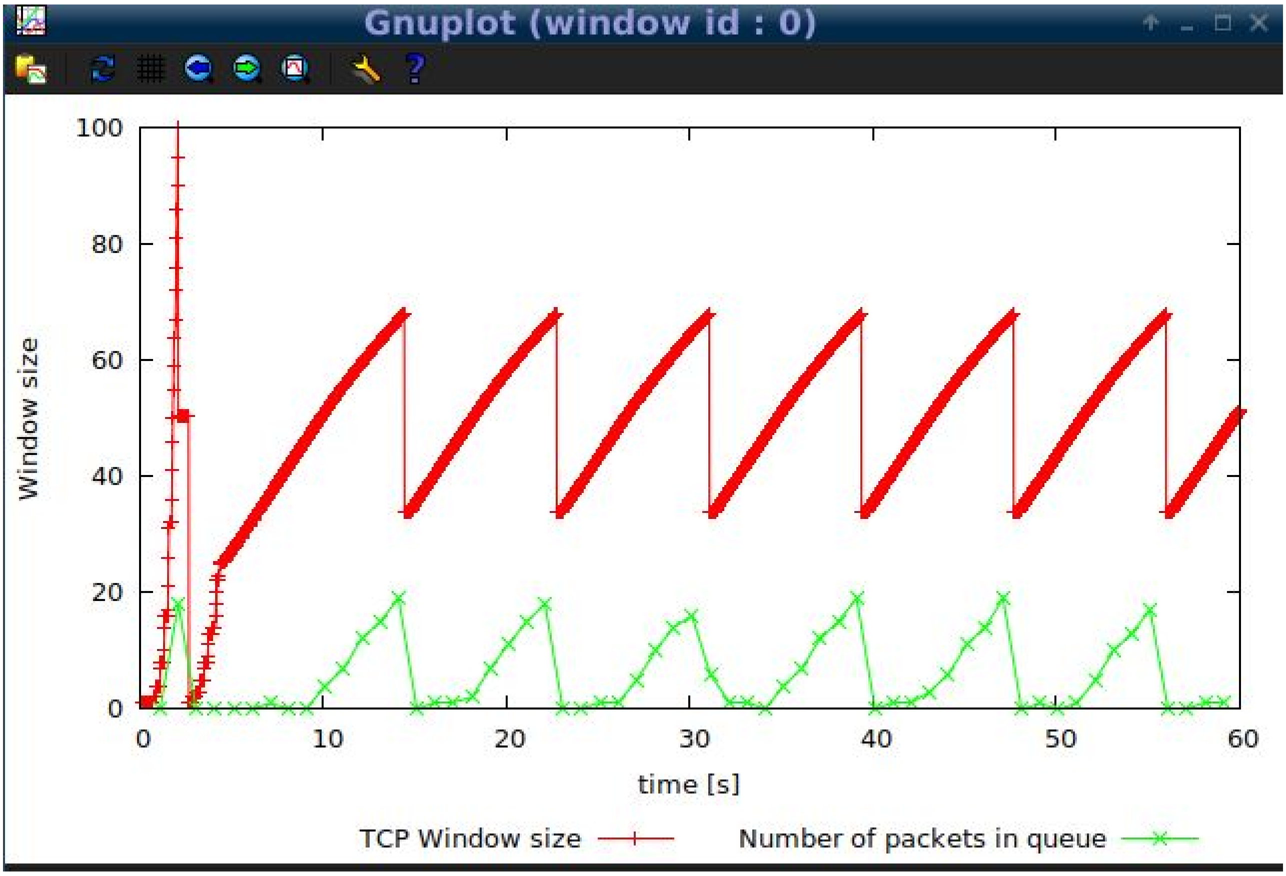
In figure 1 (with 70 packets max CWND), TCP oscillates between the SS (slow-start) phase and the CA (congestion avoidance) phase.
In figure 2 (with 65 packets max CWND), the oscillations stop after the second SS.
In figure 3 (with 50 packets max CWND)., TCP reaches to 50 and stops the oscillating
From figure 4, the average packet throughput is around 225 packets/sec.
Thus, 225 * 8 * 540 = 972,000 bps, which is near the link capacity(1Mbps).
Throughput(figure 4)



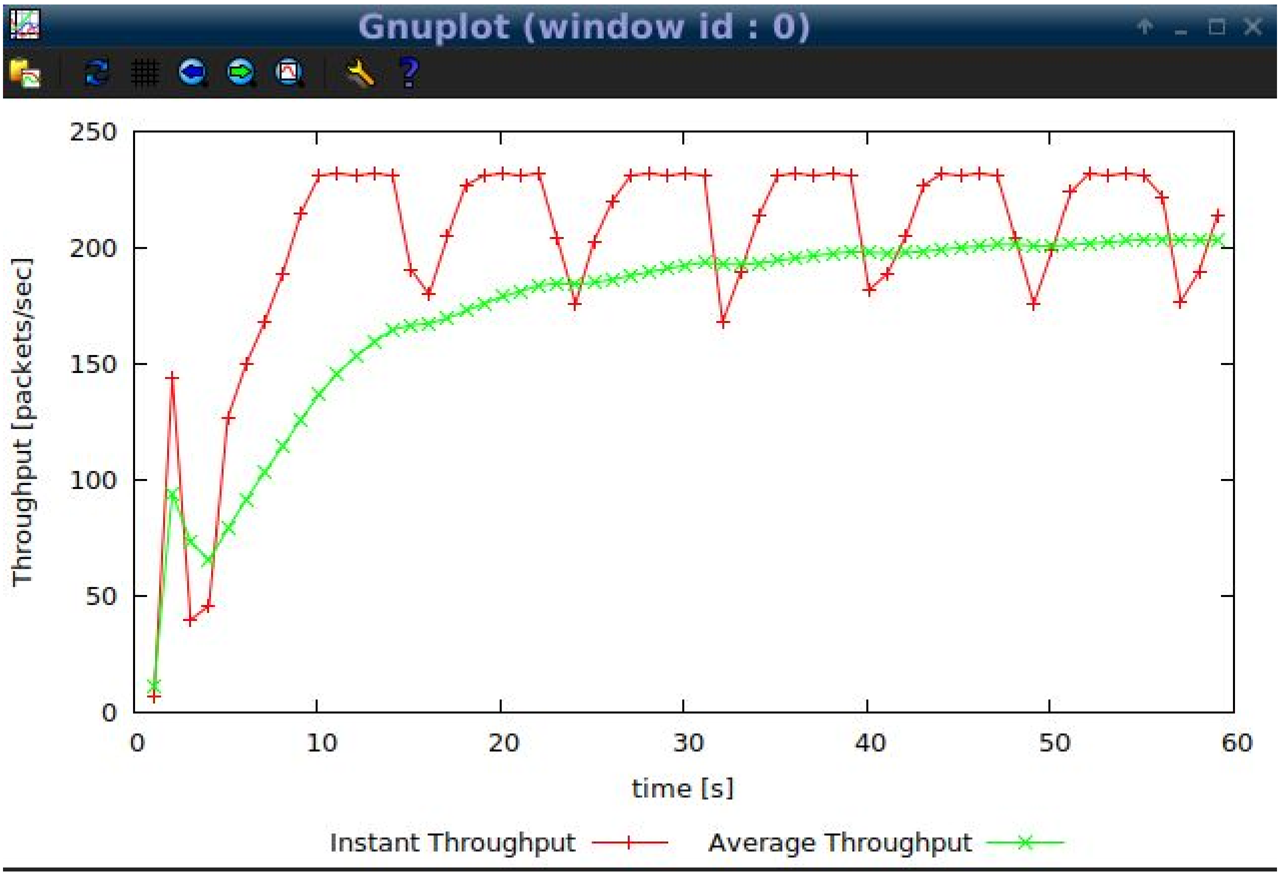
Question 4: Repeat the steps outlined in Question 1 and 2 (NOT Question 3) but for TCP Reno. Compare the graphs for the two implementations and explain the differences. (Hint: compare the number of times the congestion window goes back to zero in each case). How does the average throughput differ in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

The max CWND: 150. When loss happens, the TCP Reno does not go to SS (slow-start) phase (i.e. drop the window size to 1 MSS). It drops to the half and then enters to CA (congestion avoidance). This is called fast recovery and it could prevent 'pipe' from emptying after fast retransmit. The throughput is about 200 packet per second and then there is 200 * 8 * 540 = 864,000 bps.

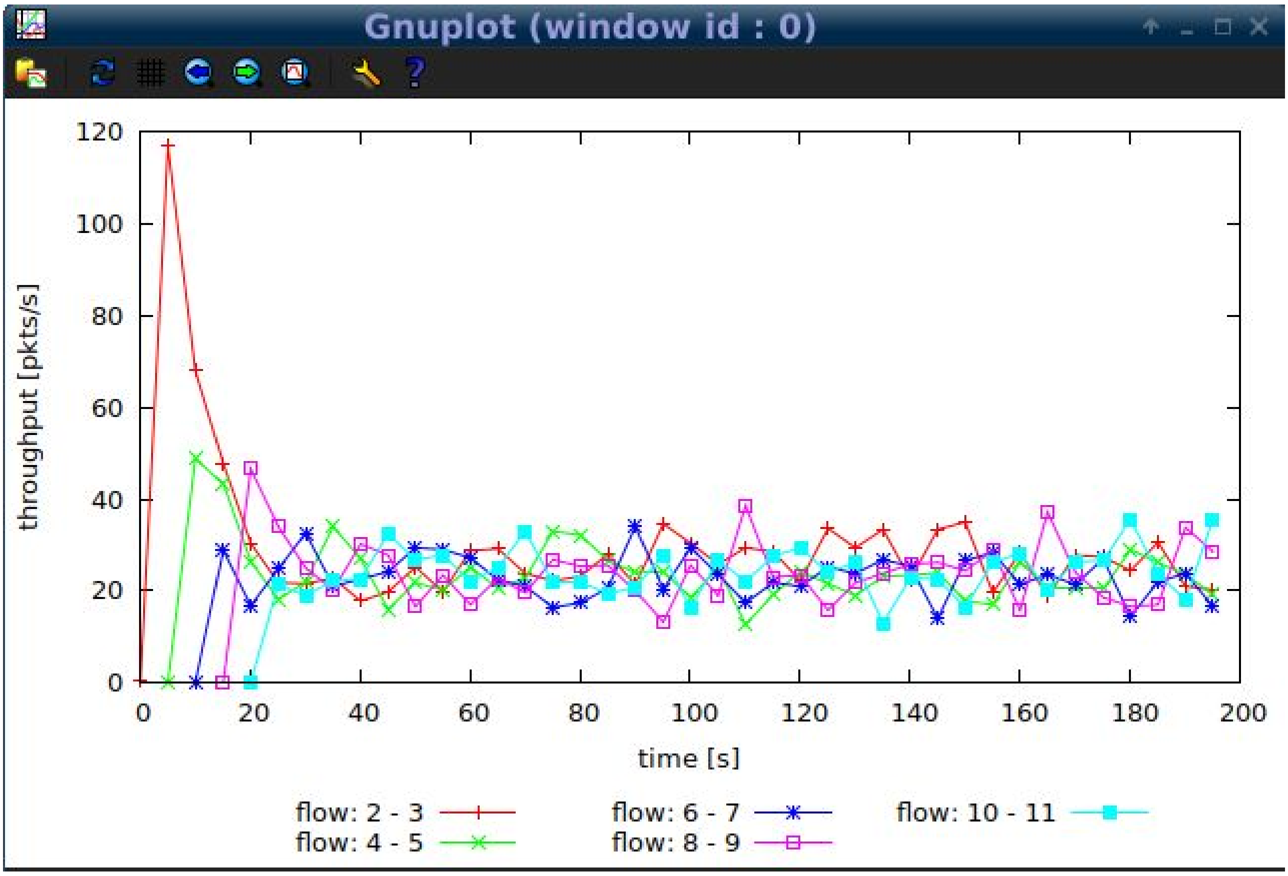
Replace set tcp0 [new Agent/TCP] to set tcp0 [new Agent/TCP/Reno]
$ns tpWindow.tcl 150 100ms $gnuplot Window.plot



$gnuplot WindowTPut.plot



Exercise 2: Flow Fairness with TCP



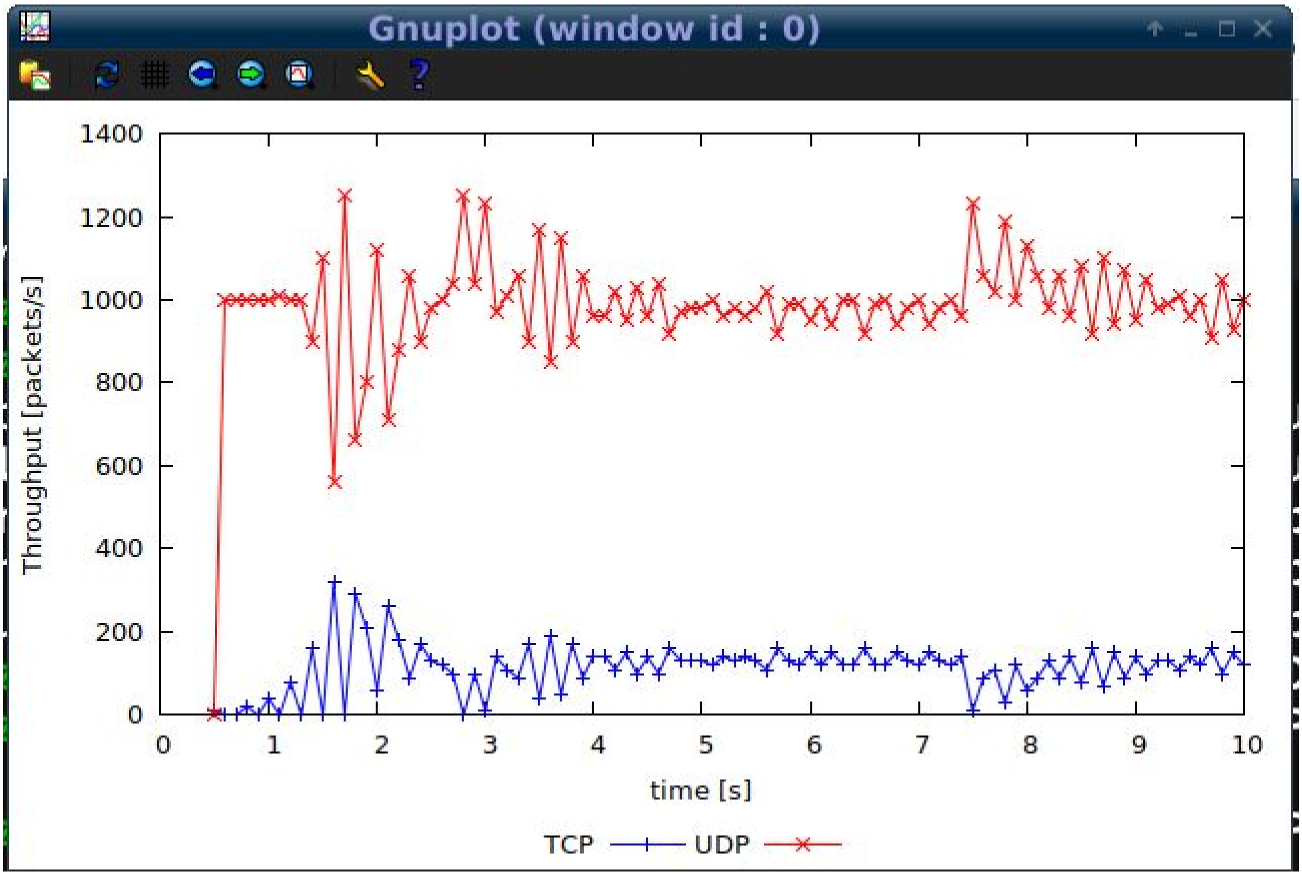
Question 1: Does each flow get an equal share of the capacity of the common link (i.e., is TCP fair) ? Explain which observations lead you to this conclusion.

Yes. The throughput of node 2 is around 120 at the beginning and it decreases whenever a new connection of other nodes start. The first few nodes will do the same behaviour. The throughput of nodes oscillate in a similar range after all connections of nodes start.

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 about whether you consider this behaviour to be fair or unfair.

From the picture, the first few connections have pretty high throughput and it decreases a lot when a new flow is created. After all flows are created, they are average over time.
The congestion control mechanisms will contribute to this behaviour.
I think this is a fair behaviour. Since all flows are under the same network condition, and the same mechanism will be used. In addition, each connection adjusts the size of the connection window when setting up a new connection to allow sharing of the common link.

Exercise 3: TCP competing with UDP



Question 1: How do you expect the TCP flow and the UDP flow to behave if the capacity of the link is 5 Mbps ? Can you guess which colour represents the UDP flow and the TCP flow respectively?

> I expect the UDP throughput is higher than the TCP throughput if the capacity of the link is 5Mbps. The red line represents the UDP and the blue line is the TCP.

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.

> The TCP has congestion control and UDP does not. Thus, the UDP transfers the packet at relatively constant speed regardless of dropping packets in the transmission. And the TCP will detect congestion and reduce the congestion window size.

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?

> Pros: transfer the file fast because UDP could dominate the throughput in a shared link
>
> Cons: it will cause congestion without congestion control If everyone started using UDP, the network will collapse.