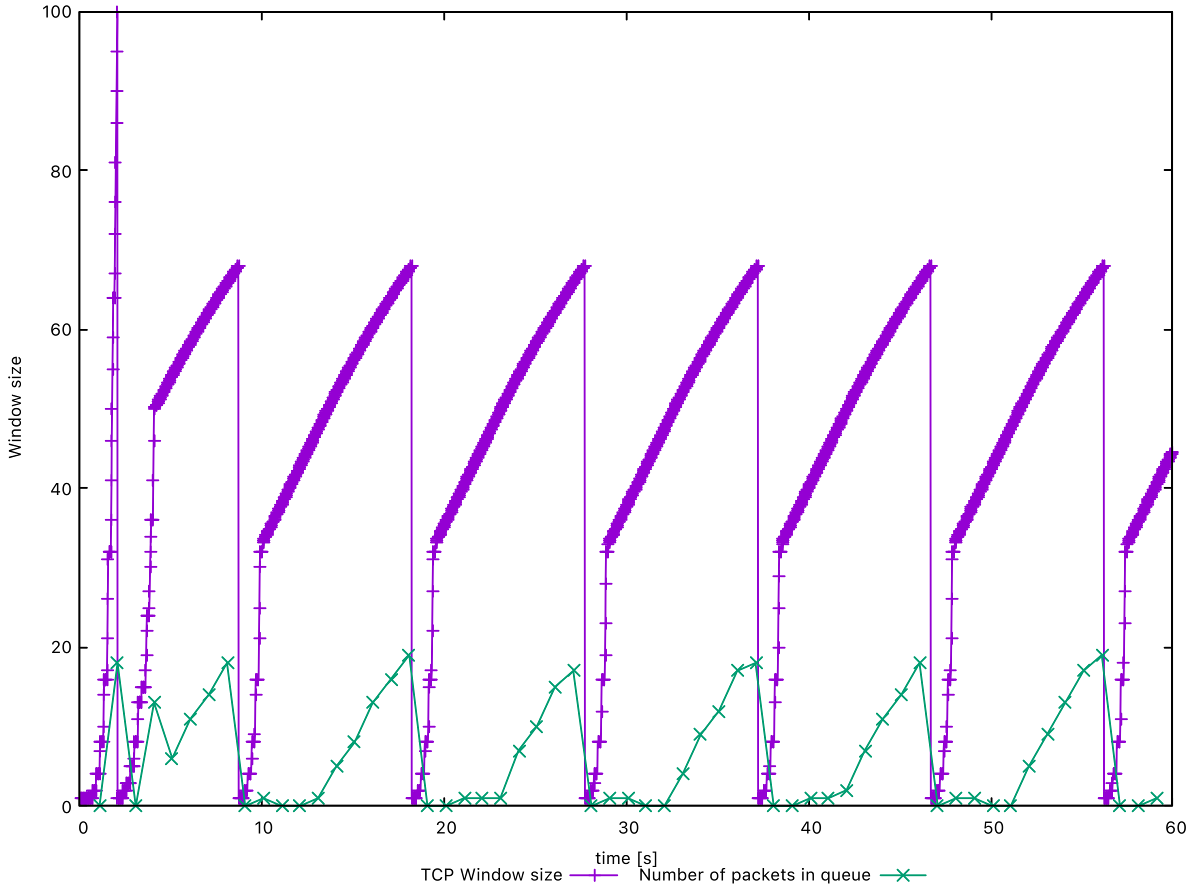
Lab Exercise 5: TCP Congestion Control and Fairness

Exercise 1: Understanding TCP Congestion Control using ns-2

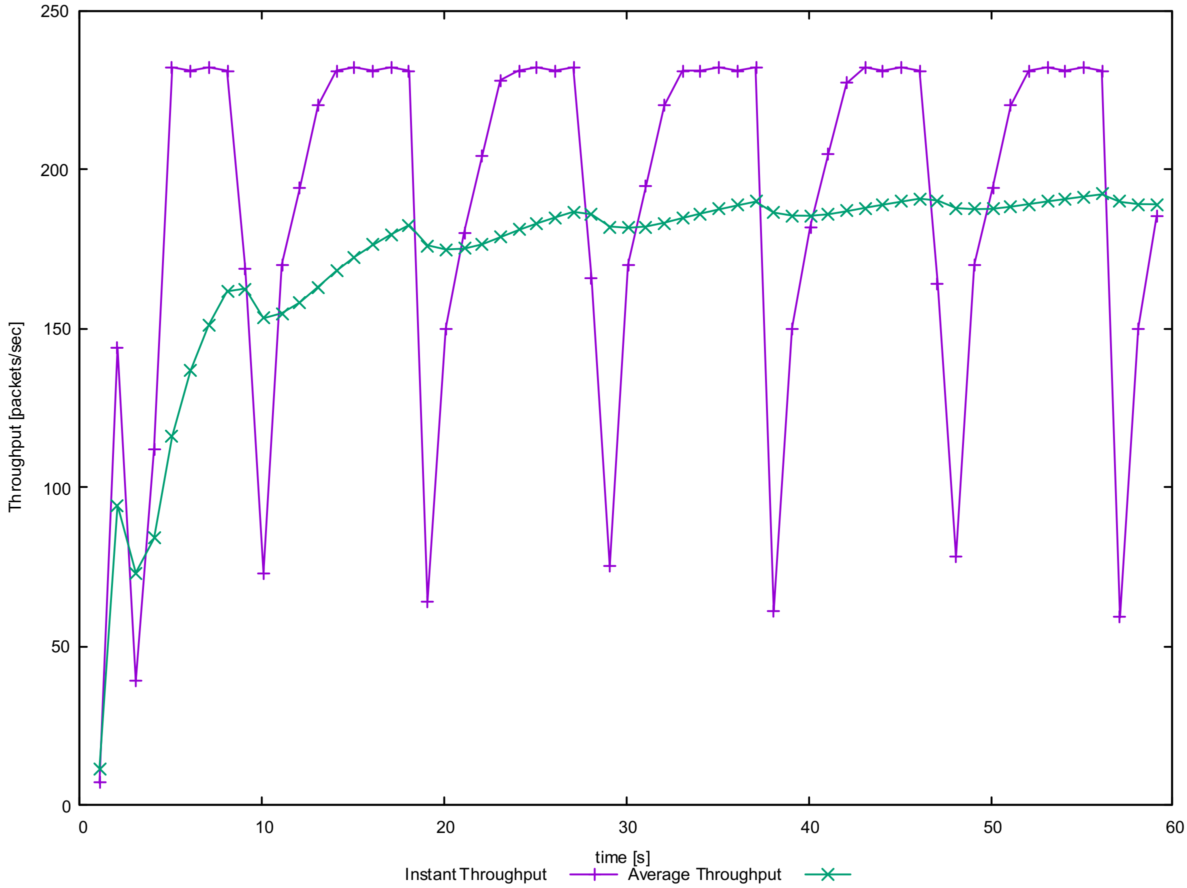
Question 1: Run the script with the max initial window size set to 150 packets and the delay set to 100ms (be sure to type "ms" after 100). In other words, type the following:



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.

As the window size increases, congestion occurs at the sending end. When the queue is full, packets are discarded because the window size increases. Therefore, the sender reduces the congestion window to 1 and the threshold to 1/2 the window size about 50 packages. The connection enters a slow start phase, after which the window size increases rapidly until the threshold is reached and the connection enters the congestion avoidance phase. When the queue is full again, the connection becomes slow start again.

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?



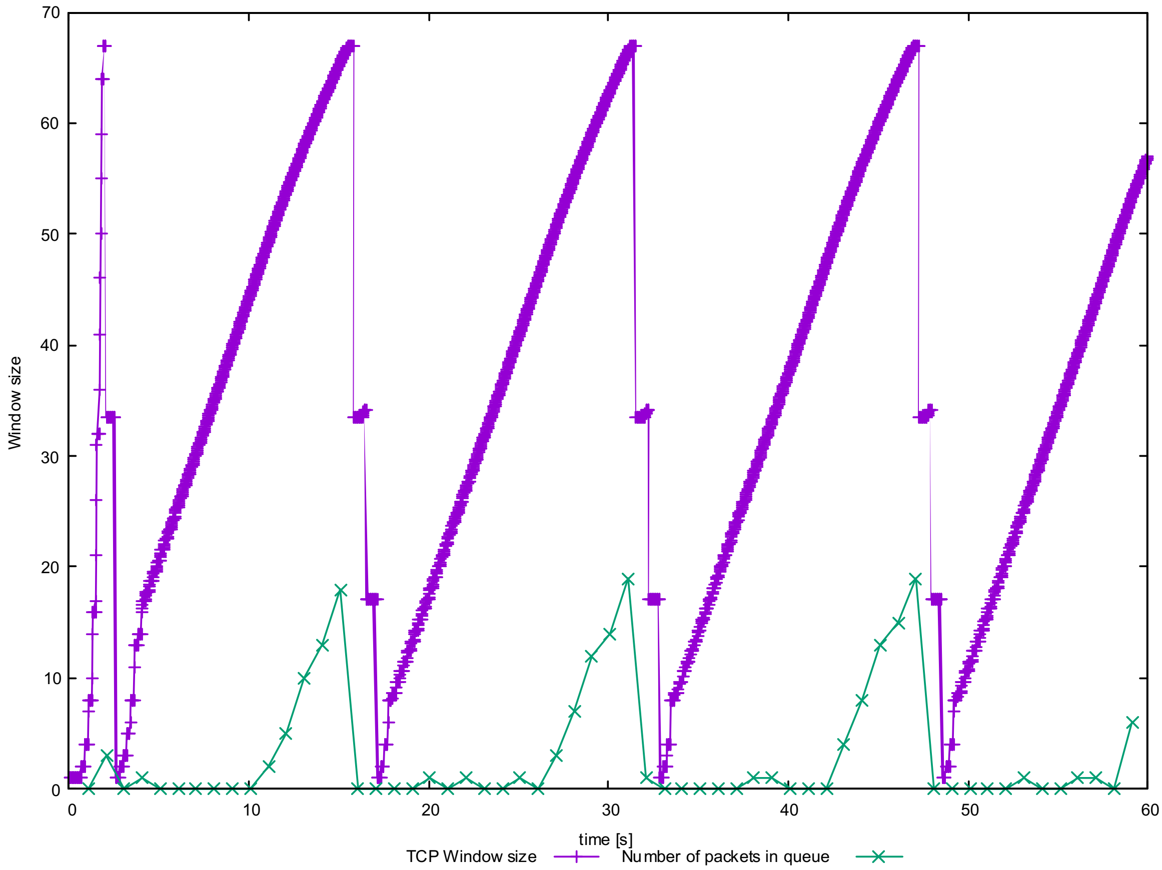
The average packages throughput of TCP is 190pps.

If rate only contains the useful data, the throughput is $190 * 500 * 8 = 760000bps = 760Kbps$.

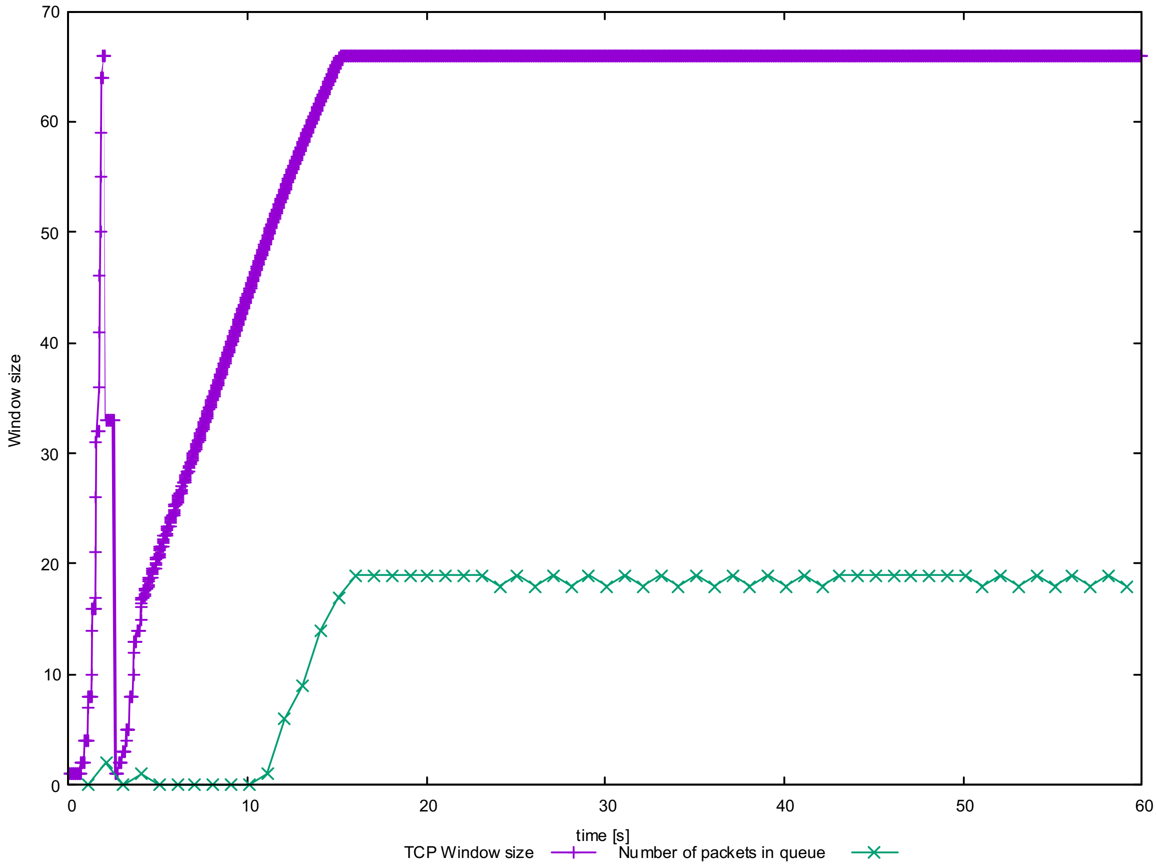
If rate contains the head and payload data, the throughput is $190 * 540 * 8 = 820800bps = 820.8Kbps$.

Question 3: Rerun the above script, each time with different values for the max congestion window size but the same RTT. How does TCP respond to the variation of this parameter? Find the value of the maximum congestion window at which TCP stops oscillating to reach a stable behaviour. What is the average throughput at this point? How does the actual average throughput compare to the link capacity?

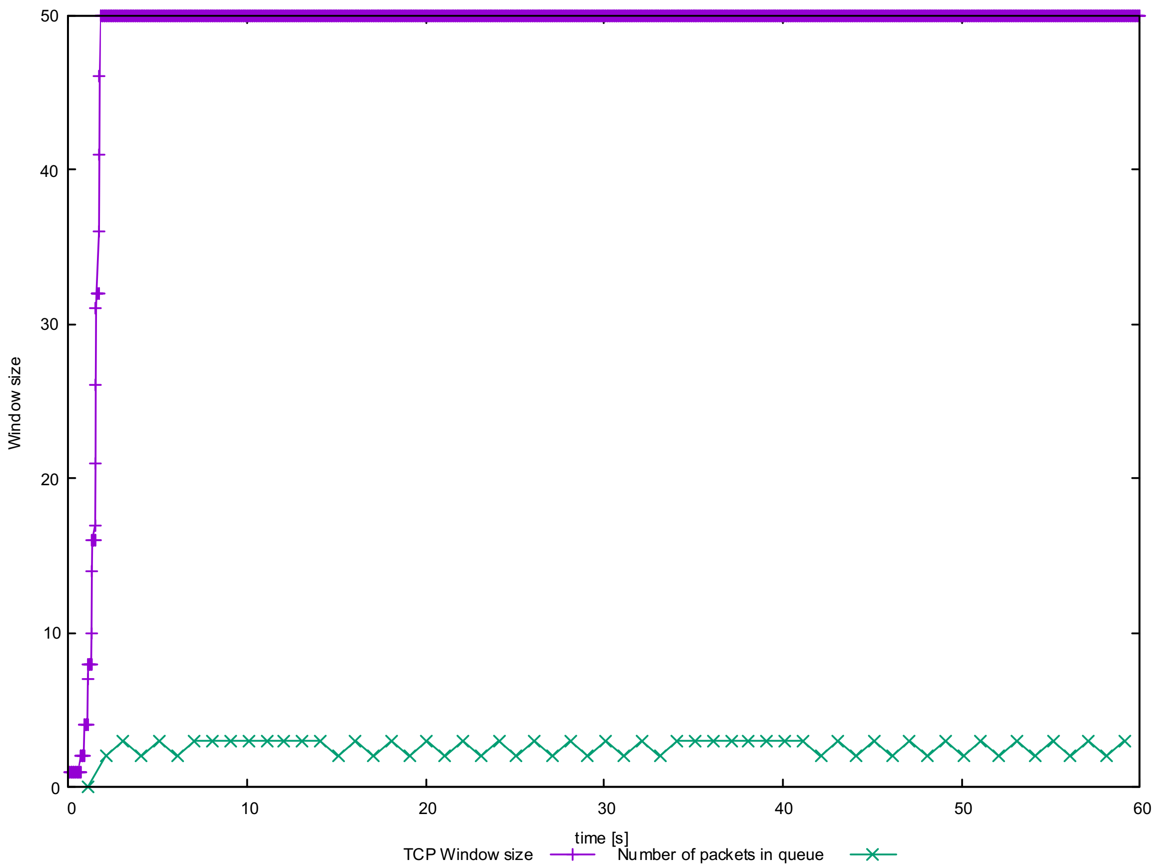
When the maximum congestion window size is 67, it is not stable.



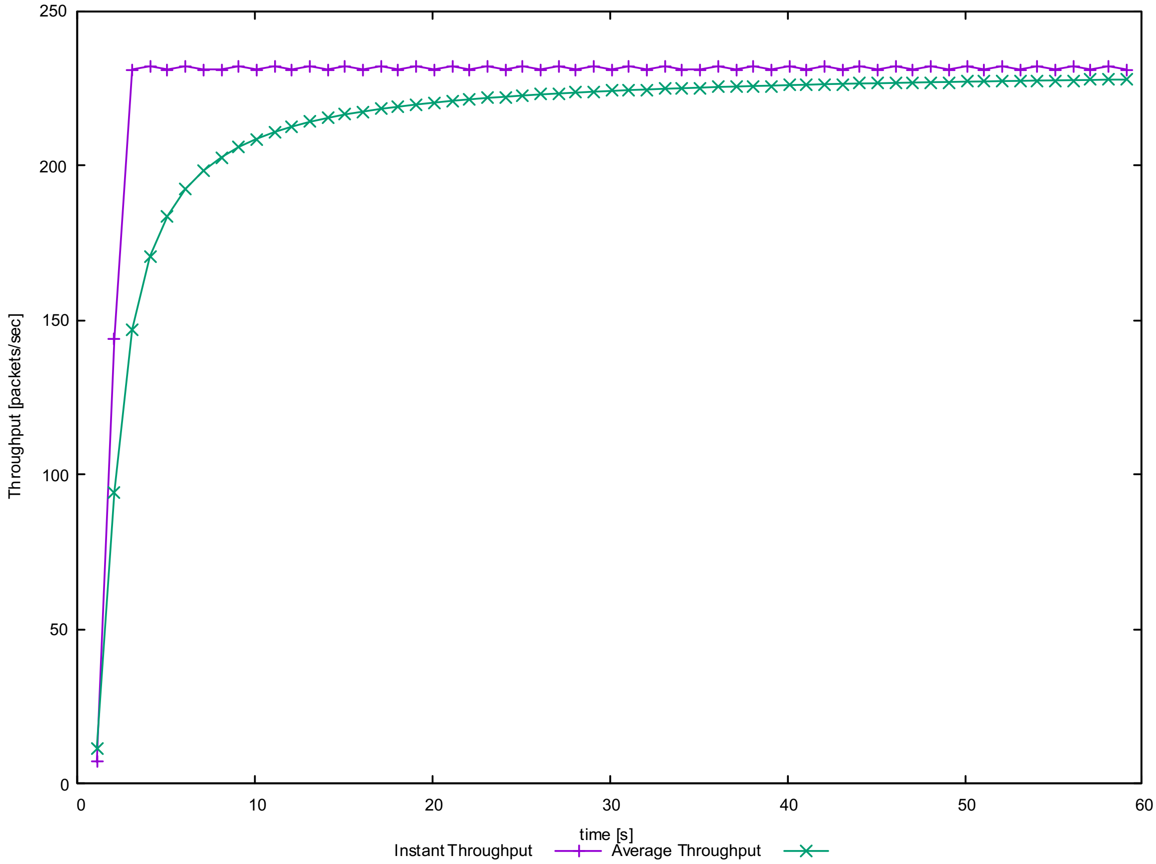
When the max congestion window size equal to 66, TCP stops oscillating after the first return to slow start.



When the maximum congestion window size is 50.



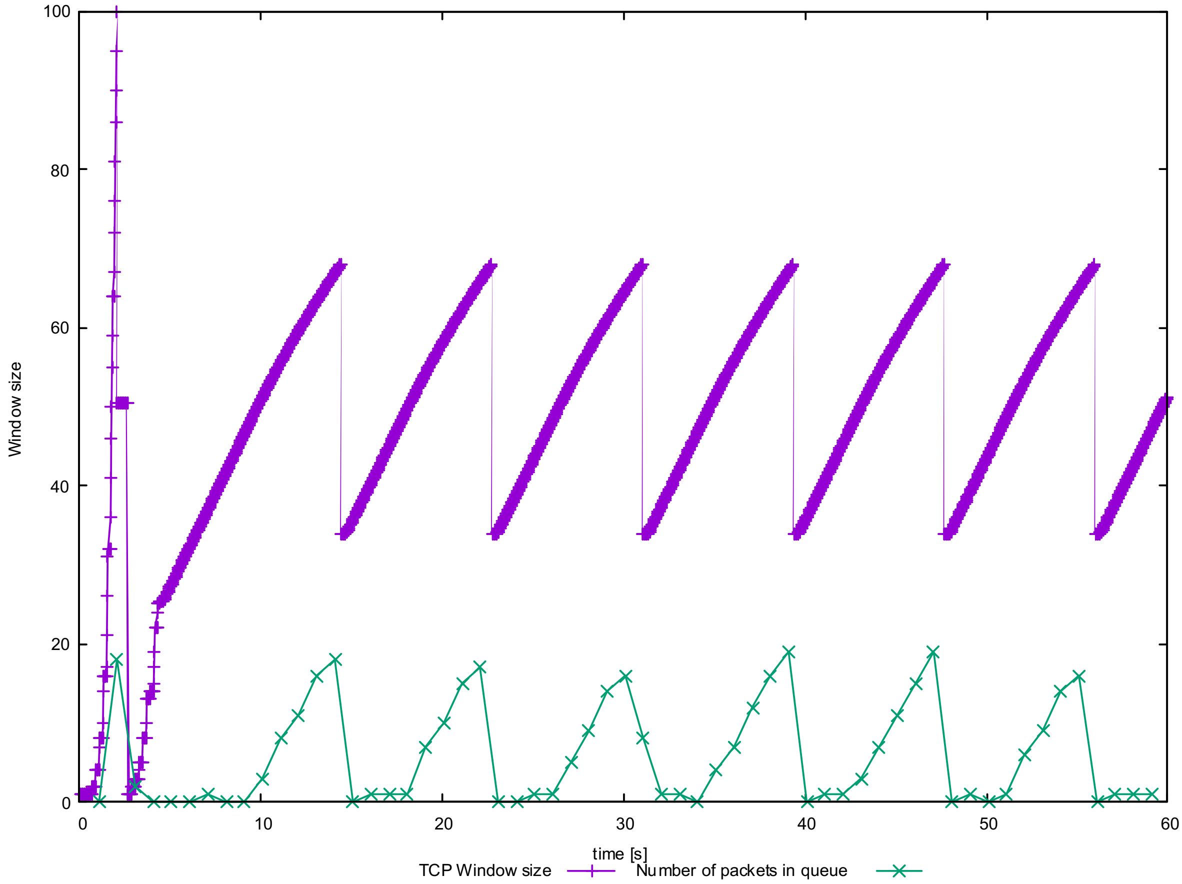
When the max congestion window size is 50, the average throughput is $225 * 500 * 8 = 900\ Kbps$, which is almost equal to the link capacity (1 Mbps).



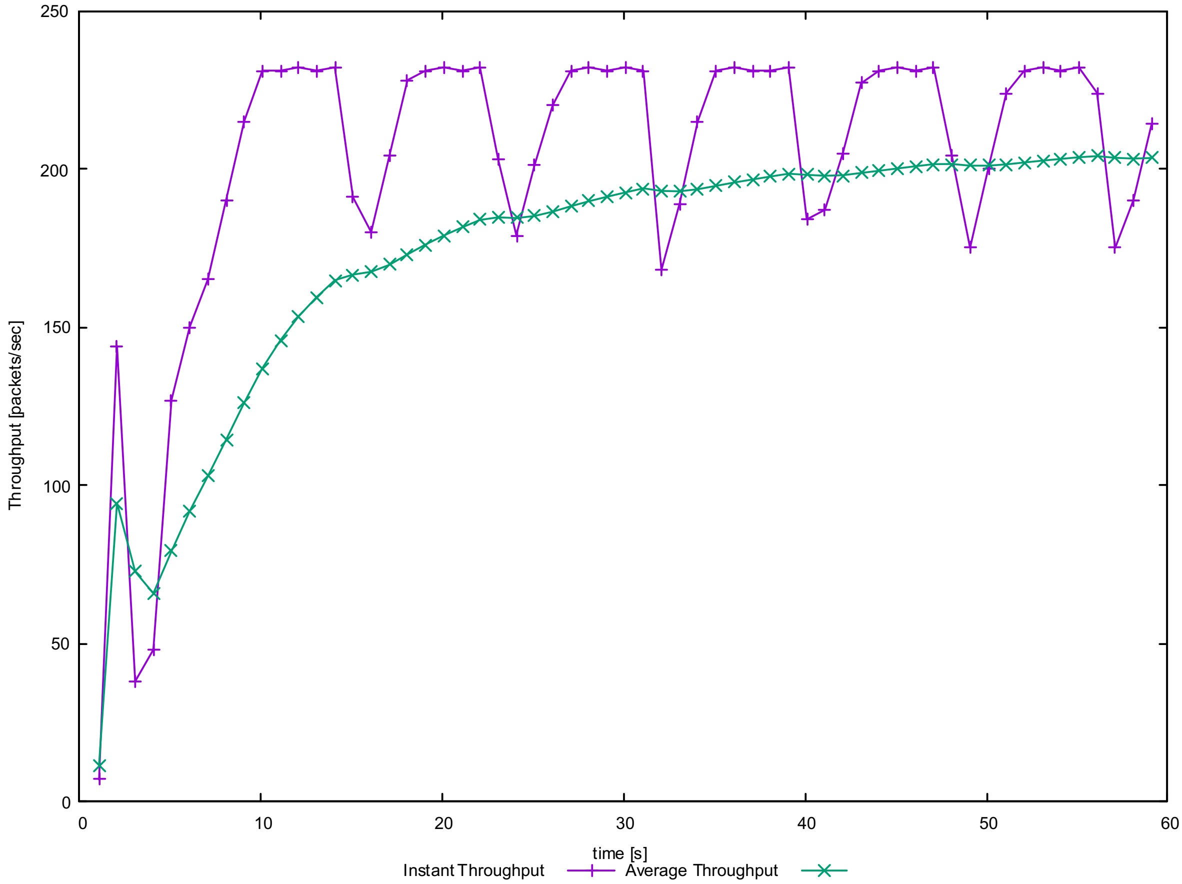
Question 4: Repeat the steps outlined in Question 1 and 2 but for TCP Reno. Compare the graphs for the two implementations and explain the differences. How does the average throughput differ in both implementations?

In TCP Reno, except for the first congestion event, the window size is not reduced to 1 after the congestion event occurs, the TCP connection does not enter slow start. The sender halves its current congestion window and increases until packets are lost again.

The difference between TCP Tahoe and Reno is that TCP Tahoe's throughput is around 190pps, and TCP Reno's throughput is around 200pps, slightly higher than TCP Tahoe's. TCP Reno does not require a slow start after every congestion event, so throughput is higher.

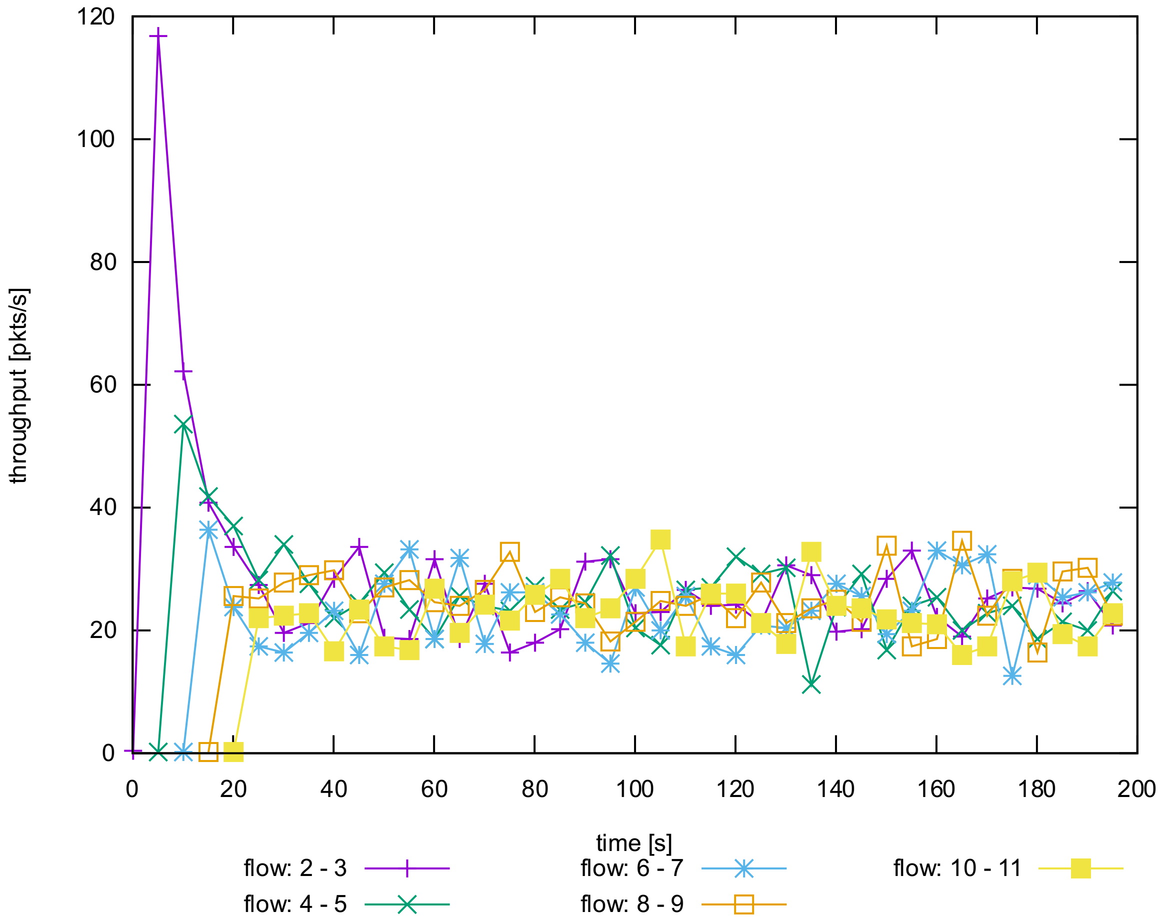


100
80
60
40
20
0
Window size
0
10
20
30
40
50
60
time [s]
TCP Window size
Number of packets in queue



250
200
150
100
50
0
Throughput [packets/sec]
0
10
20
30
40
50
60
time [s]
Instant Throughput
Average Throughput

Exercise 2: Flow Fairness with TCP



Question 1: Does each flow get an equal share of the capacity of the common link? Explain which observations lead you to this conclusion.

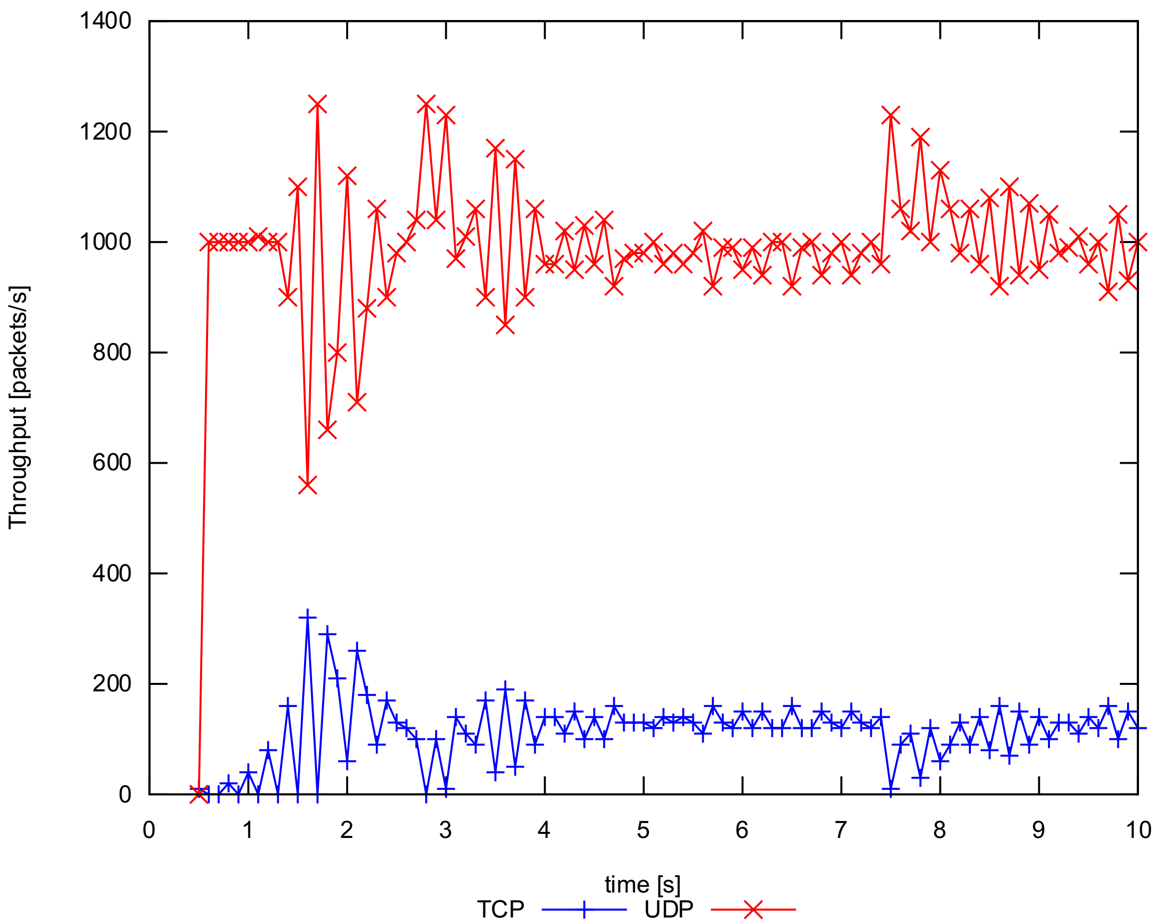
Yes. According to the graph the five flow is not start together, therefore, the first 25 seconds date does not make sense to analyse. After 25 seconds, the throughput for those connections reaches a similar value, each flow gets an equal share of the common link.

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.

When a new flow is created, the throughput of all pre-existing flows is reduced. Because The congestion window of new flow increases quickly and creates congestion during slow start. All flows are automatically resized to accommodate the addition of new flow to avoid overloading. When a new flow was added, the fair share of all existing flows will reduce.

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?



When congestion occurs, UDP flow will not reduce its transmission rate.

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.

UDP achieves higher throughput than TCP. Since UDP need not be restrained by congestion control, it can transmit packets at a high rate, it will not be changed by congestion. When TCP flow and UDP flow are on a same link, UDP flow will force TCP flow to reduce throughput and may starve TCP flow.

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?

Advantage : UDP flow is not limited due to congestion and can greatly reduce delay

Disadvantage: UDP cannot make sure the data be transfer reliably.

If everyone uses UDP, when congestion occurs, the sending rate will be unlimited, the congestion cannot be controlled and are difficult to alleviate. The Internet may encounter congestion collapse.