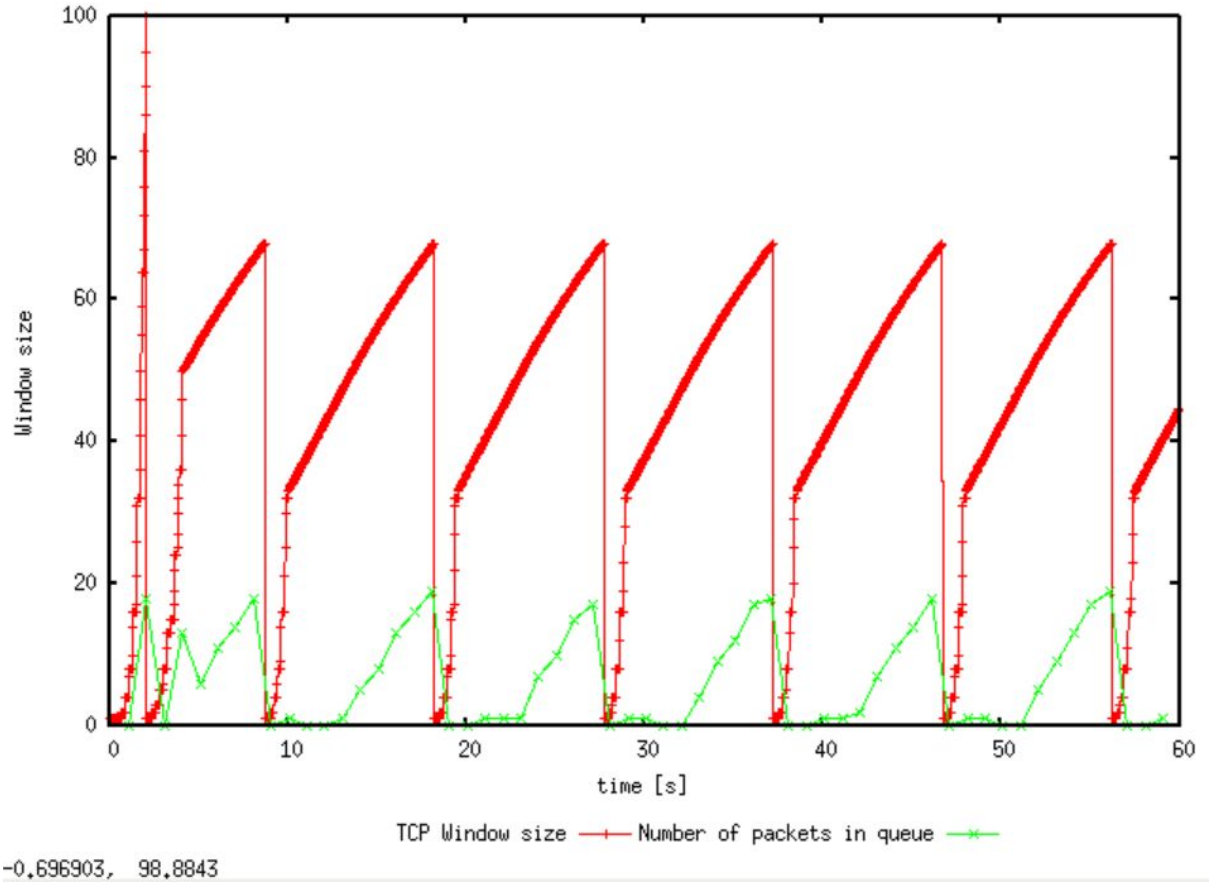
Ex1:

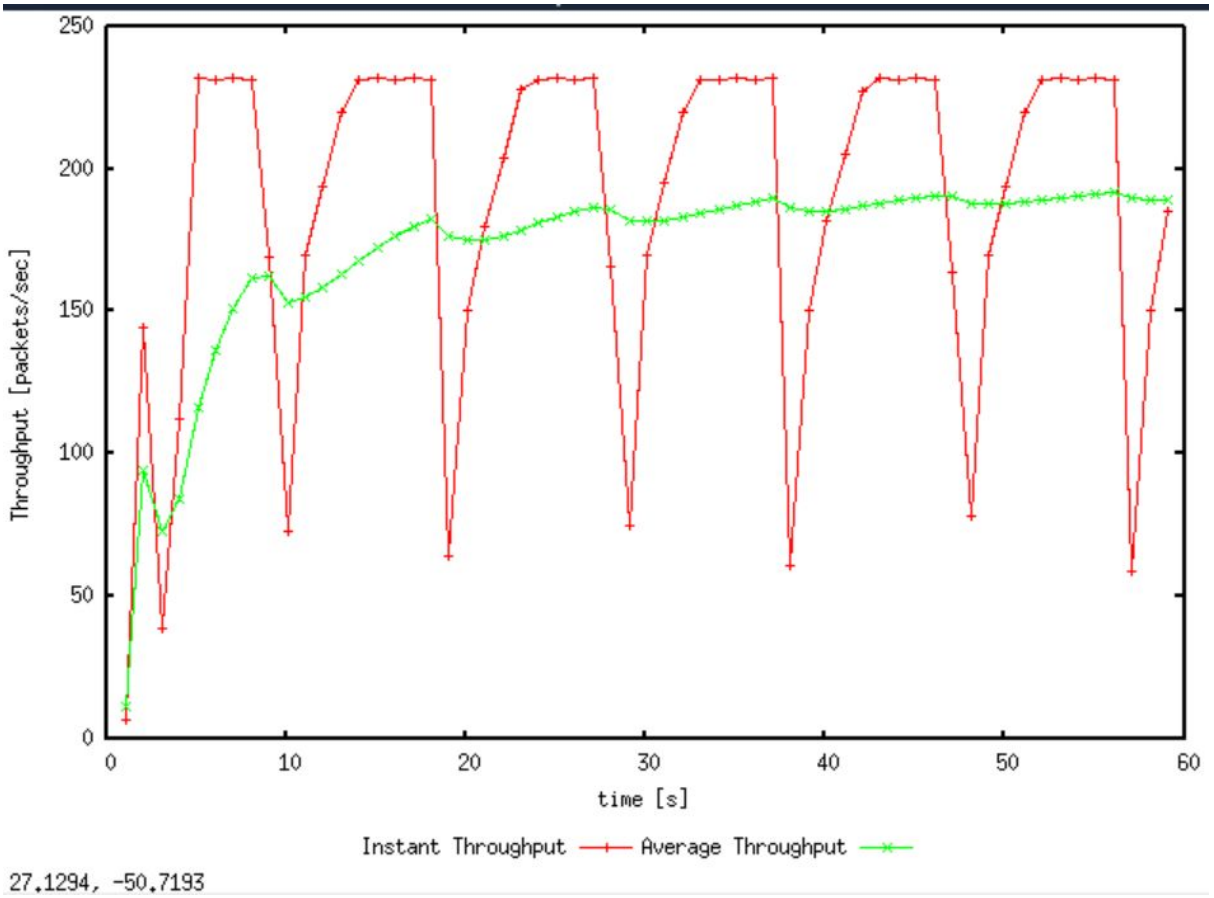
Question 1:What is the maximum size of the congestion window that the TCP flow reaches in this case? What does the TCP flow do when the congestion window reaches this value? Why? What happens next? Include the graph in your submission report.

The maximum size of the congestion window that the TCP flow reaches is 100. It is observed in the time 2.02 ms. Because slow start is stopping since a loss is occurs and fast-retransmit is sent and it is call TCP Tahoe also slow start begins again from its initial CWND.



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)

By the graph we observed average throughput of TCP is 188.675 packet/sec. Also a packet size is 500 + 20 = 520 Bytes. Hence the average throughput of TCP is 520* 188.675 = 98111 bytes/sec.



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)?

Set RTT 100 ms

In congestion_window size = 100:

Maximum congestion window size: 100

Average throughput = 188.570 packets/sec

In congestion_window size = 80:

Maximum congestion window size: 80

Average throughput = 189.894

In congestion_window size = 66:

Maximum congestion window size: 66

Average throughput = 220.394

In congestion_window size = 60:

Maximum congestion window size: 60

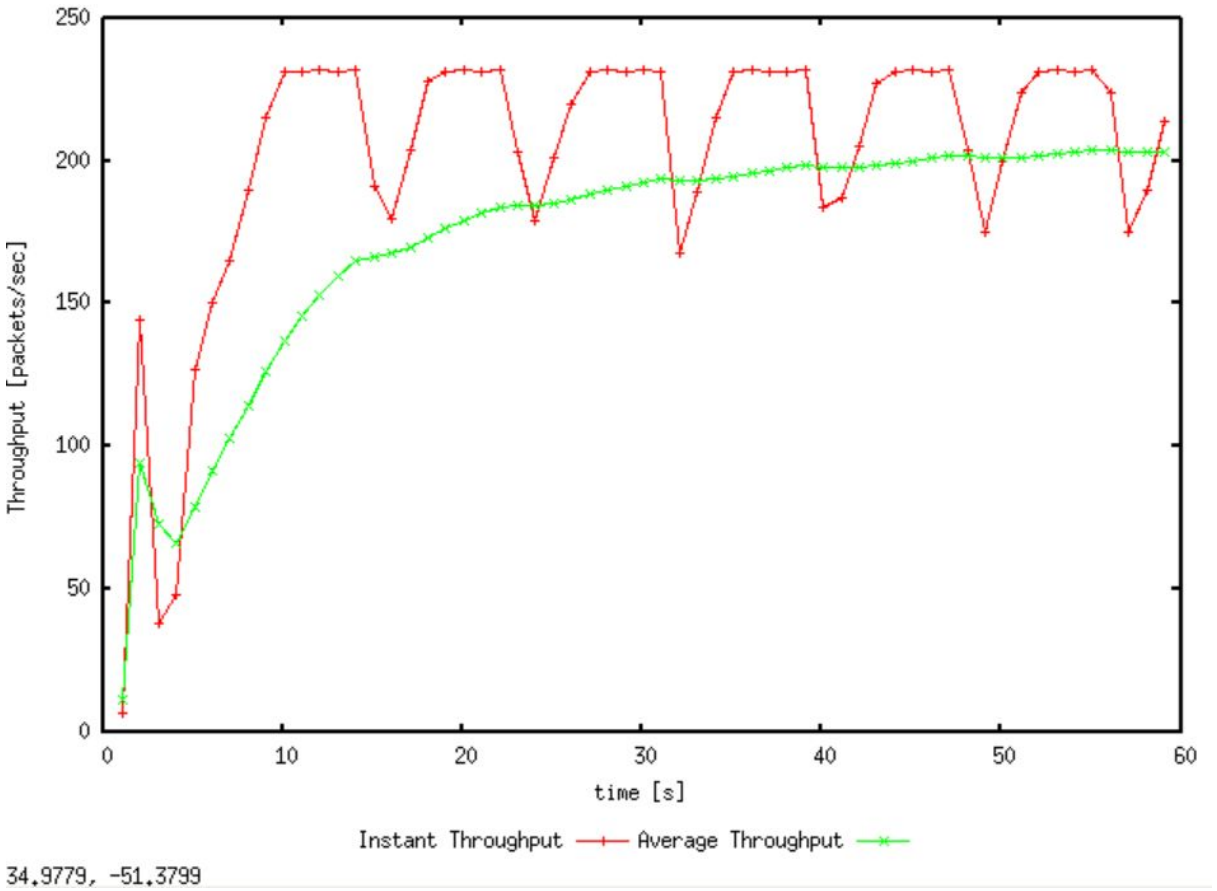
Average throughput = 219.684

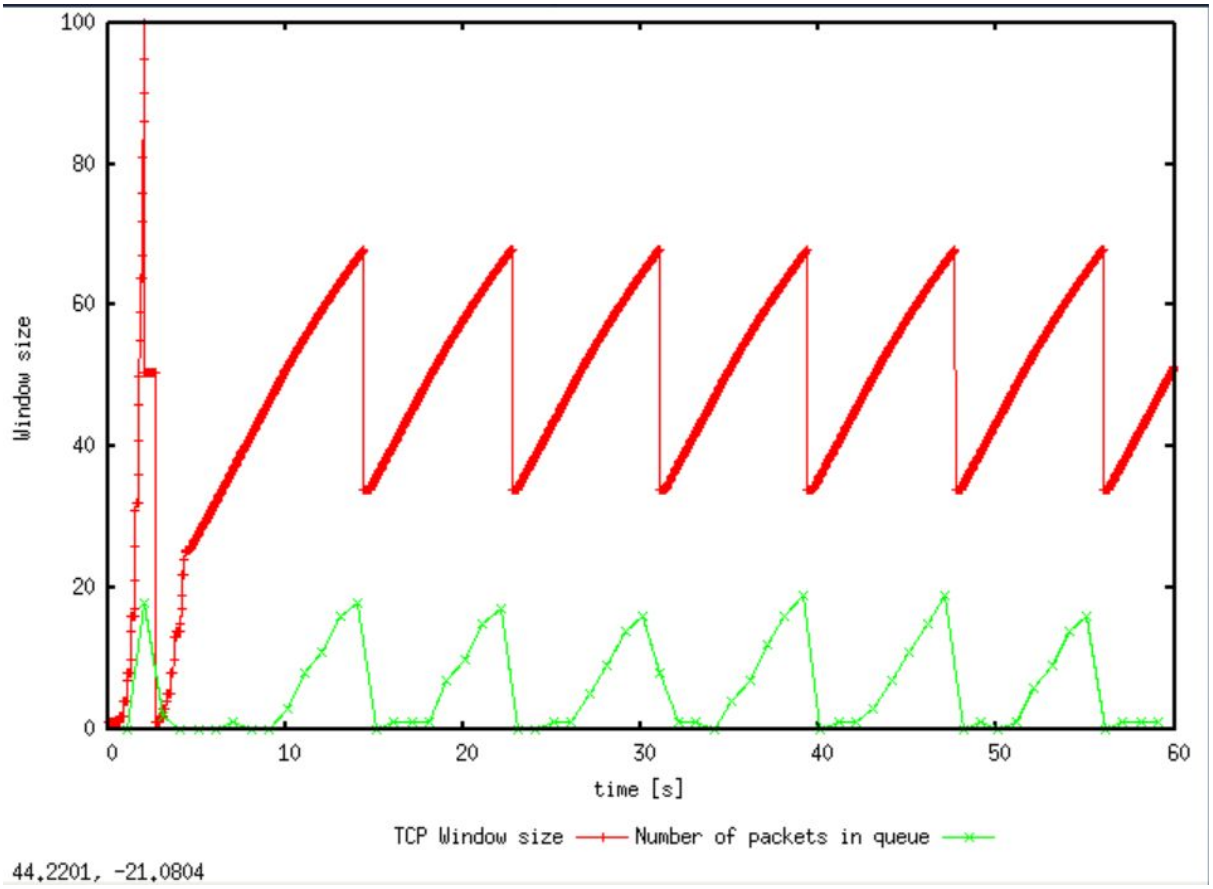
In congestion_window size = 50:

Maximum congestion window size: 50
Average throughput = 227.213

By the test cases above, we observe below 67 congestion window size, the maximum congestion window size is the input congestion_window_size. Also the average of throughput is nearly close to the instant average and become higher. The maximum congestion window at which TCP stops oscillating is 66. The average throughput in that point is 220.394 packets/sec and 114604.88 bytes/sec. The actual average throughput is 0.91683904 Mbps is really close to 1 Mbps when the maximum congestion window is equal to the provide congestion window size.

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?





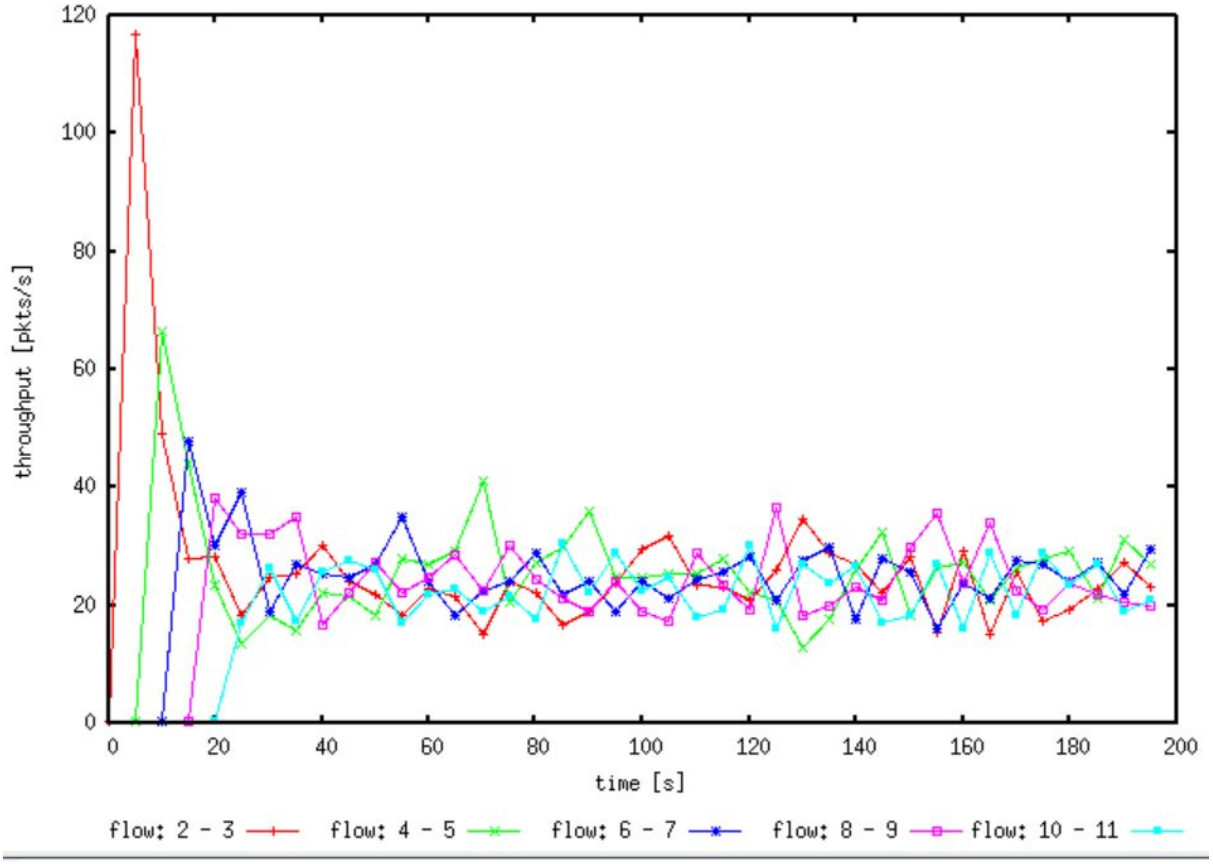
	Tahoe	Reno
Number of times that congestion window size go back to zero	7	1
Average throughput	188.675 packet/sec	203.158 packet/sec

Since Tahoe congestion window size always reduce to zero when packet loss and the throughput will also be significantly dropped at that time then the average throughput will be much lower than Reno. Because Reno only reduced to half away when packet loss so the throughput will be stable and higher than Tahoe.

Ex2:

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.

It is not TCP fair. If is equal share of the capacity of the common link, all the flow will be in one line after the final flow's slow start and joint the link. By observation the line of each flow are not joint together hence there was not TCP fair.



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.

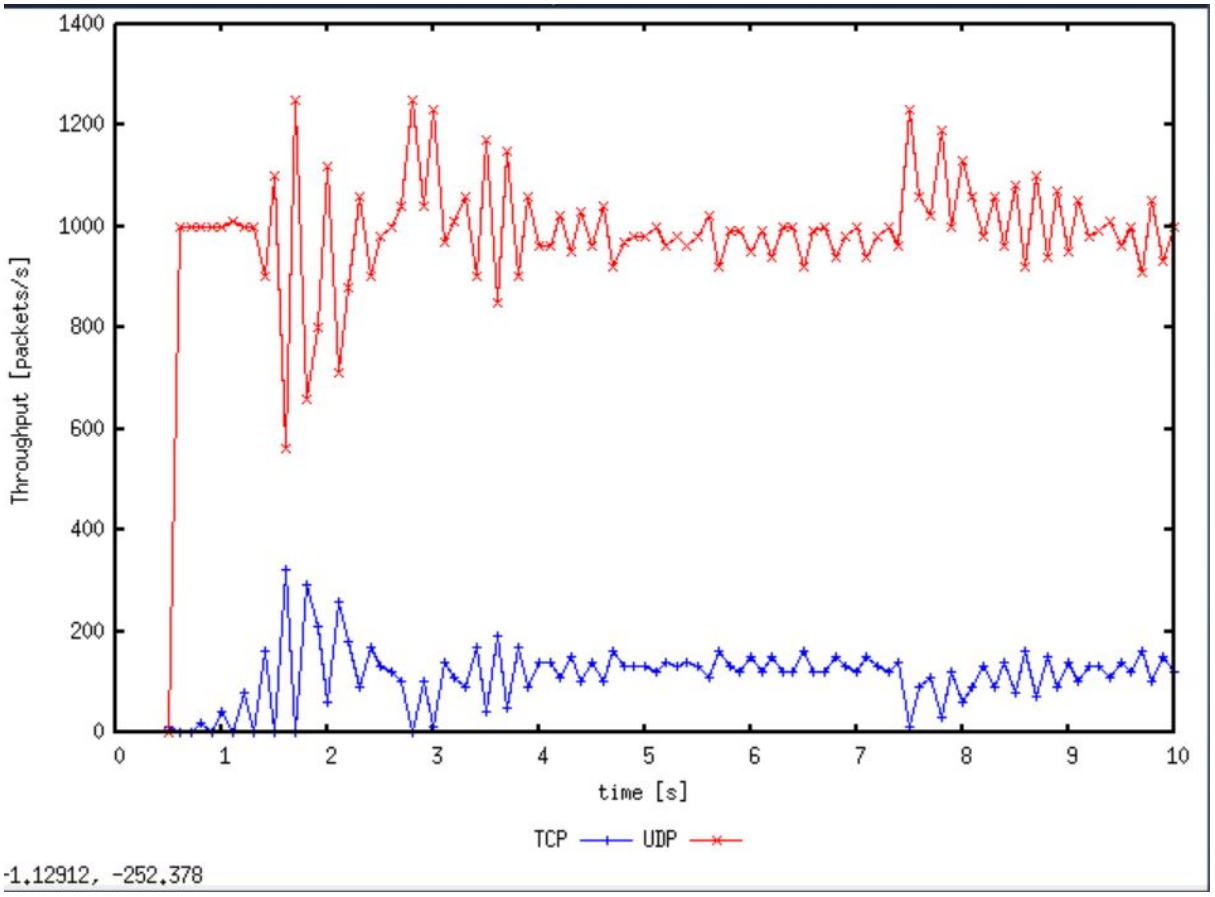
The throughput of the per-existing TCP flows increased a lit bit but not much. This is TCP flow control. This behaviour is unfair, since the new flow must share capacity when joint in the link, to be fair all the per-existing flows need to slow down there throughput and let the new flow catch up the throughput then create a balance.

Ex3:

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

Since TCP is stable and with slow start and fast transmit, TCP will not have the most capacity of the link but it will increase the link capacity and share the capacity with UDP flow. On the other hand UDP is unstable but without slow start and flow control, it will have the most of the capacity at first then share the capacity with TCP flow.

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.



Firstly TCP is stable so it uses ACK to check the packet loss, however UDP do not consider the packet loss. Hence UDP is faster than TCP. Also TCP throughput will be drop when packet loss(flow control) occurs so the throughput of TCP could not reach a very high speed.

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?

	UDP
Disadvantages	1. Unstable, packet loss easily 2. No flow control and slow start the throughput maybe by extreme high or low 3. Transmit might disconnect without reason

Advantages	1. Faster than TCP 2. Quick start 3. Use most of the capacity of the link

If all the people using UDP instead of TCP for quick transmit, the connection will be more unstable and more packet loss will occur also the throughput will be unstable as well.