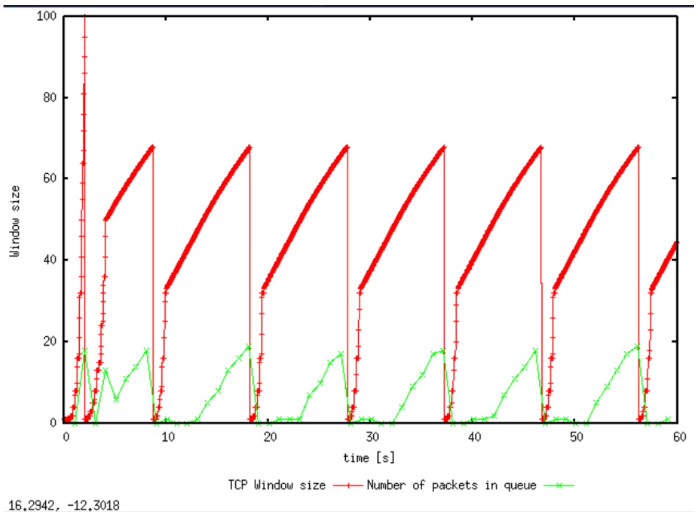
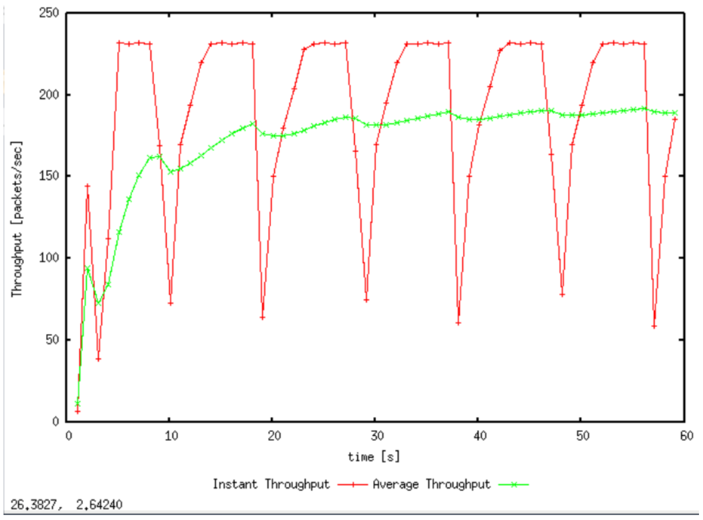
Exercise 1:

Q1

- With a window size of 150 packet at 100ms delay, the max congestion seems to go to 100 packets with the slow-start mechanism.
- Some packets are dropped because the 100 packet congestion > size of the queue (20 packets). As the window size is ramping up the queue becomes full, then the packets get dropped as a result of congestion at the sender.
- Next, the connection enters slow start and ramps up the window fast until it hits the threshold. Eventually, the queue becomes full again, resulting in packet loss as a loop.It will keep alternating back to the slow start phase from packet congestion / dropping of packets.



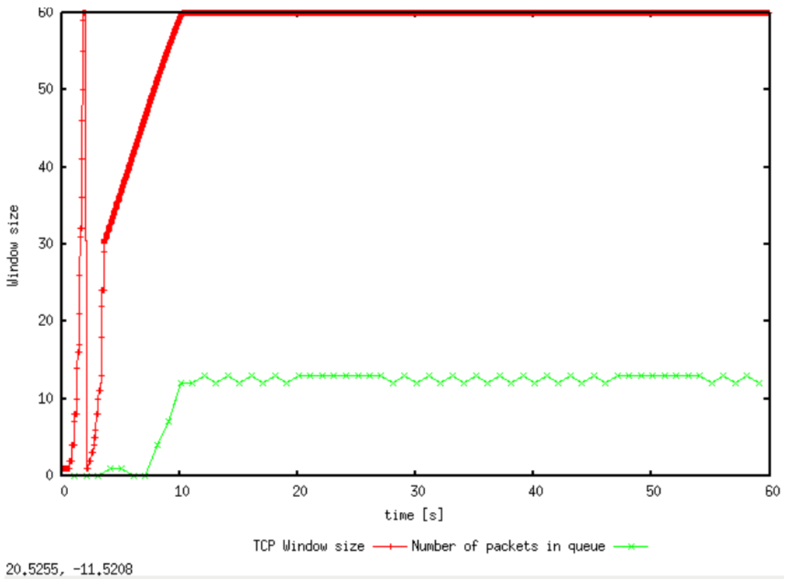
Q2.



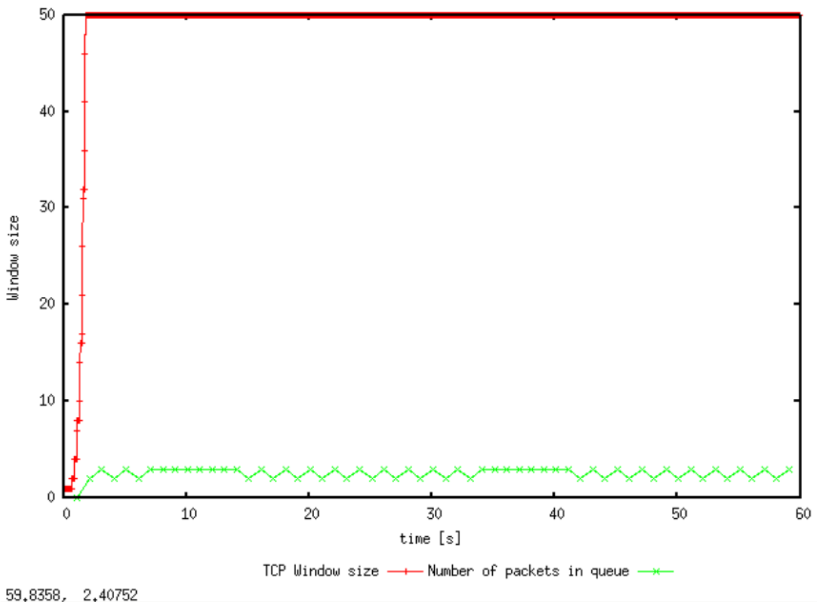
The average throughput of TCP in the graph is about 190 pps.
Bytes per second throughout: IP + TCP Headers = 20 + 20 = 40 bytes
= (500 + 40) * 8 bits per byte * 190 packets= 820,800 bps throughput
Without headers:500*8*190 =760000 bps throughput

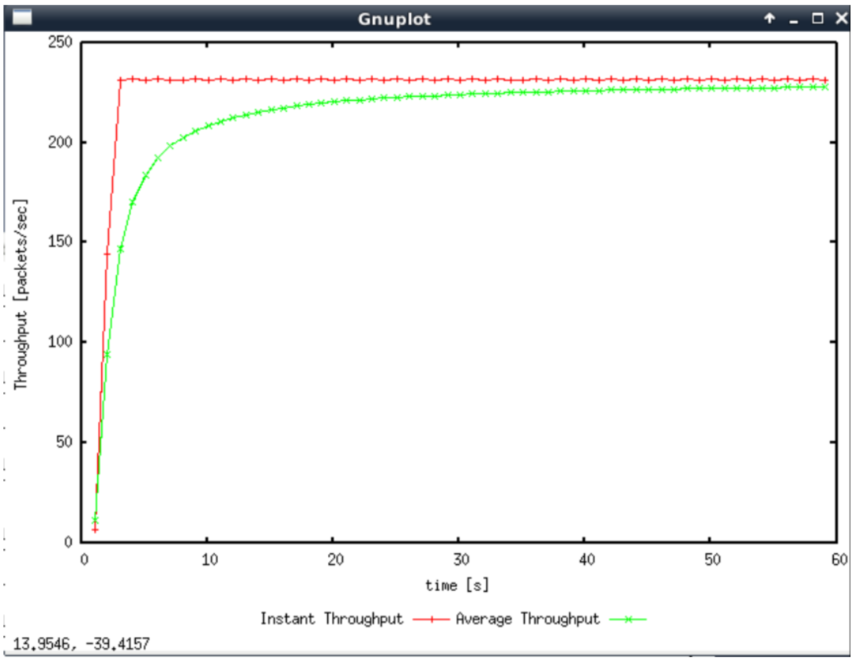
Q3

The maximum congestion window size of this graph we have set is 66 packets.



The maximum congestion window size of this graph we have set is 50 packets.



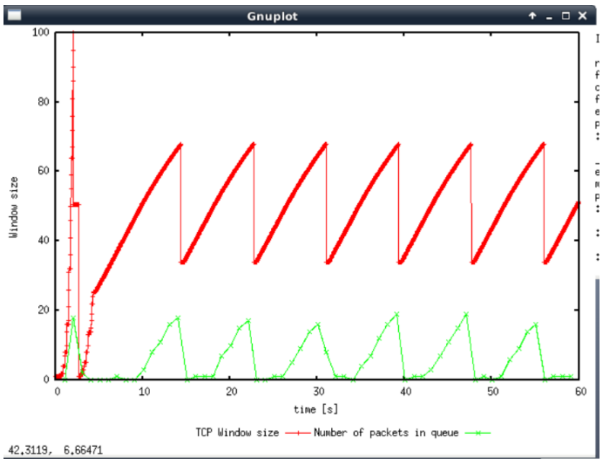


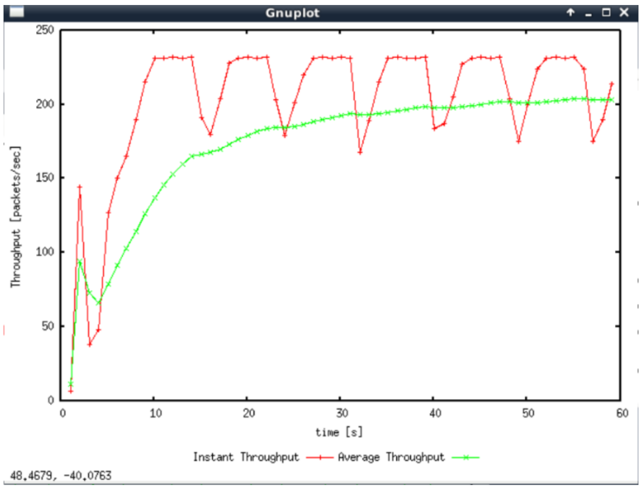
We find that some packets will be dropped and TCP will go back slow start when the maximum congestion window size becomes greater than maximum queue size.

When we set the initial value is 66 packets, the oscillations stop after the first return to slow start. And it is enough to stabilizes the number of packets in the sending queue. There does not exist packet loss.

When we the initial value is 50 packets, TCP stabilizes immediately. At this point, we can find that the average packet throughput is about 225 packets/s. The average throughput is 225*500*8=900 Kbps, which is approximately equal to the link capacity.(1 Mbps)

Q4





With TCP Reno, it does not re-enter a slow-start phase when loss occurs

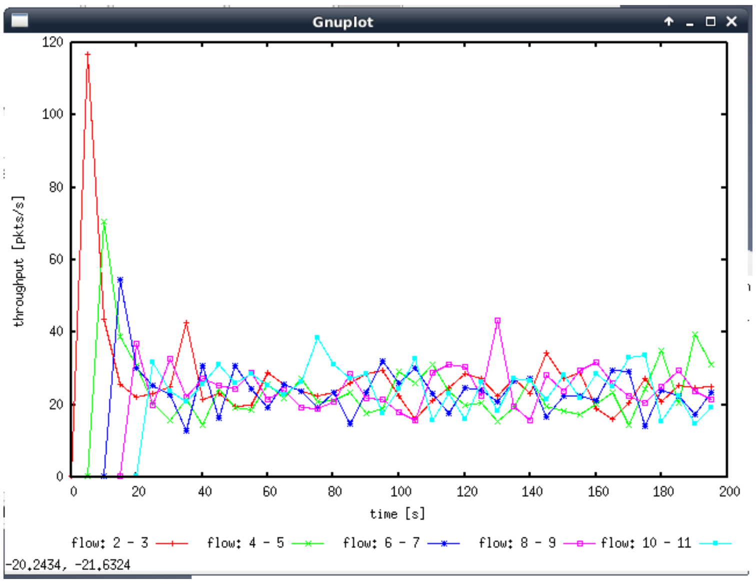
Instead, the sender halves its current congestion window and increases it linearly until losses occur, which implying that most of the losses are detected due to triple duplicate ACKs.

TCP Reno throughput = 200 pps

(500 + 40) * 8 bits * 200 = 864,000 bps > TCP Tahoe was 820,800 bps

Therefore TCP Reno performs better.

Exercise 2:

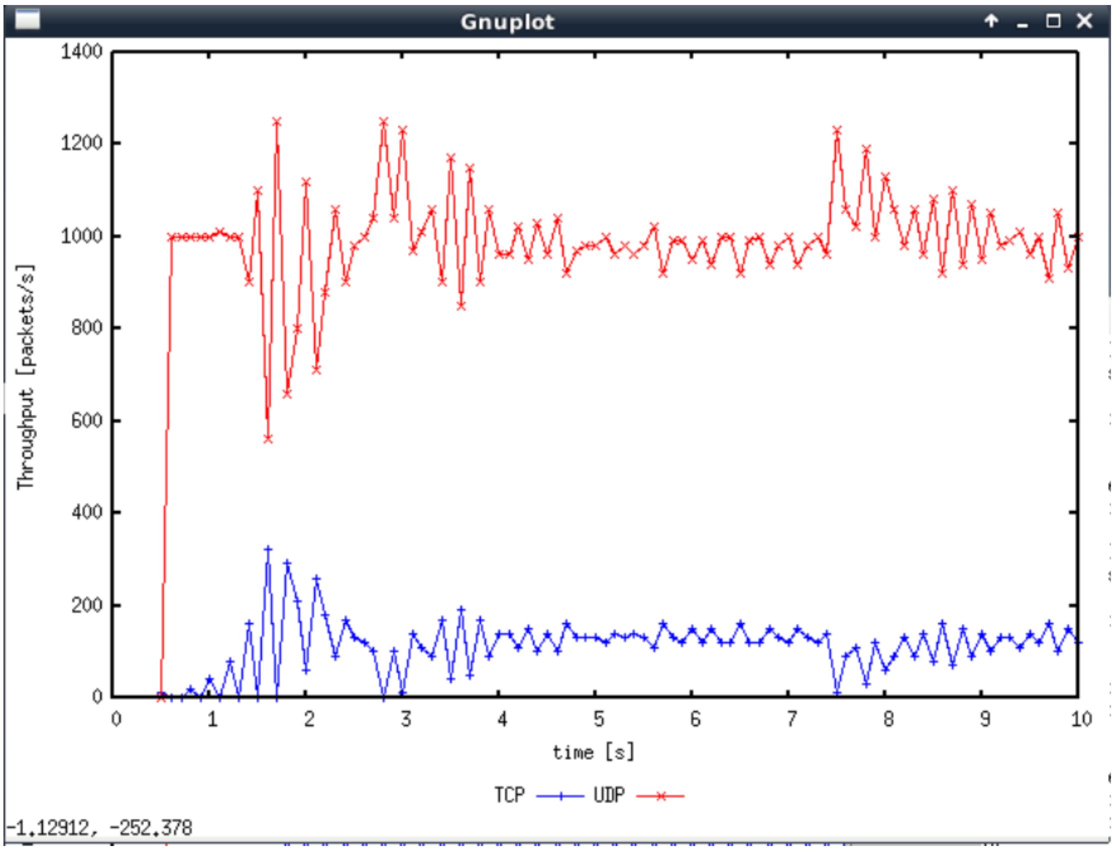


1. We can conclude from the graph above that each flow gets an equal share of the capacity of the common link. Once all the connections have commenced, the throughput of them is a bit similar.(although there may be some fluctuations).

2. This is because of the congestion control mechanisms which I consider this to be fair behavior. Each connection will adjusts the size of the connection window when a new

connection comes in. For example, as we can see throughput for the first few connections start off higher and then decreases dramatically when flow that is created, then they are all averaged out over time.

Exercise 3:



1.

I expect the UDP throughput to be higher than the TCP throughput.

2.

UDP. UDP will not result in any packet loss which is different from TCP. This means UDP will not reduce its transmission rate in response to congestion(regardless of any lost packets) and transmit packets at a stable rate. The UDP will continue to transmit while the TCP will handle the congestion.

3.

The advantages of UDP is that the sender will keep transmitting regardless of packet loss. This may reduce the delay of file transmission. However, UDP will not provide reliable data because it will not check whether has package loss happen
If everyone used UDP, although there existed congestion, none of these flows would slow down their sending rate. Eventually, it is very likely that most of the internet world will occur

congestion collapse.