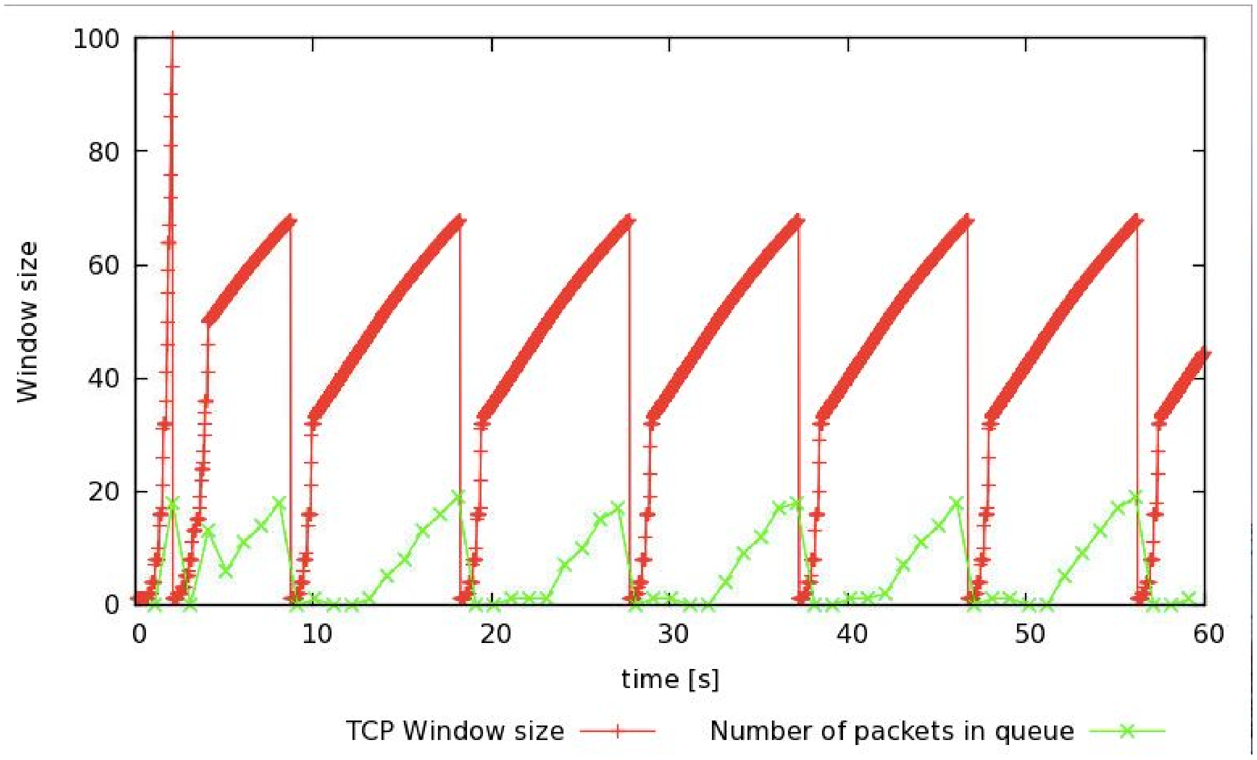
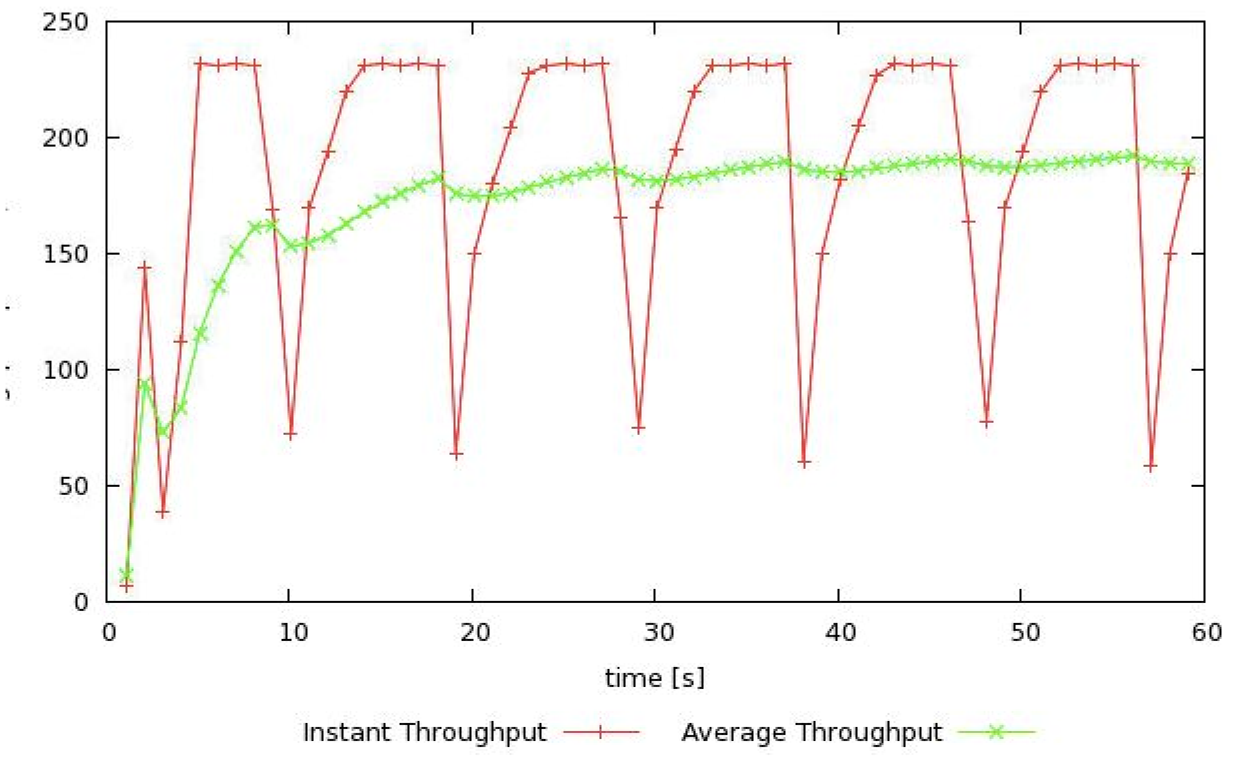
COMP3331 Lab5 Report
Ziming Zheng
z5052592

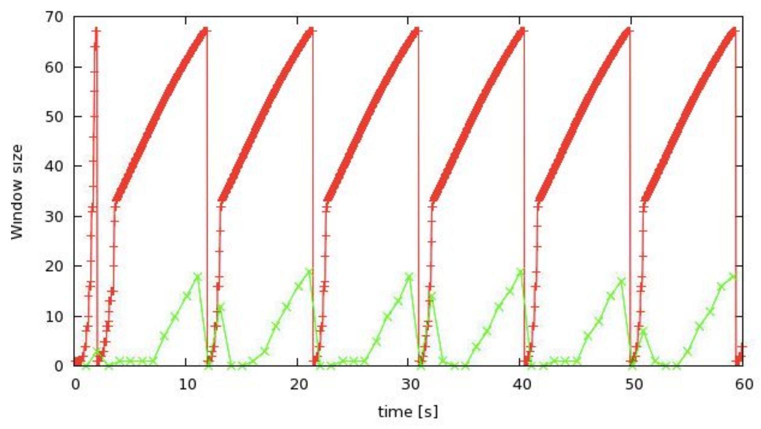
Q1:



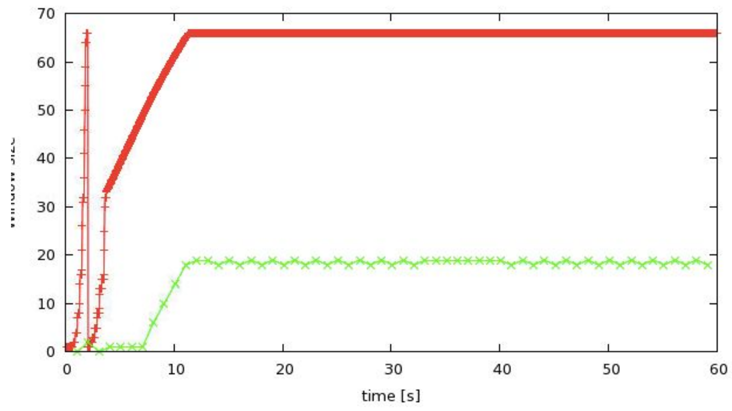
1. The congestion window goes up to 100 packets by slow start. This is observed by the graph where at the beginning there was an apparent exponential increase. As the size of queue is only 20 packets, while sending 150 packets, the queue must be full. When reaching that value, a loss event cause TCP to reduce the current congestion window size to 1 MSS and use the half of the previous value as its slow start threshold, which is 50 packets in this example.
   TCP will be in congestion avoidance after reaching 50 packets, and oscillate between a slow start phase and a congestion avoidance phase.



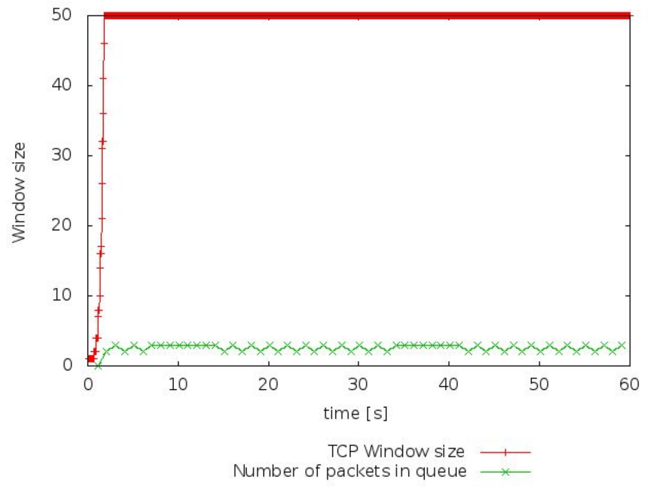
2. The average throughput is 189 packets per second. So, the average throughput in bps is calculated in 189 * (500 + 20 + 20) * 8 = 816480 bps, where (50+20+20)*8 is the bits in a packet including payload and headers.



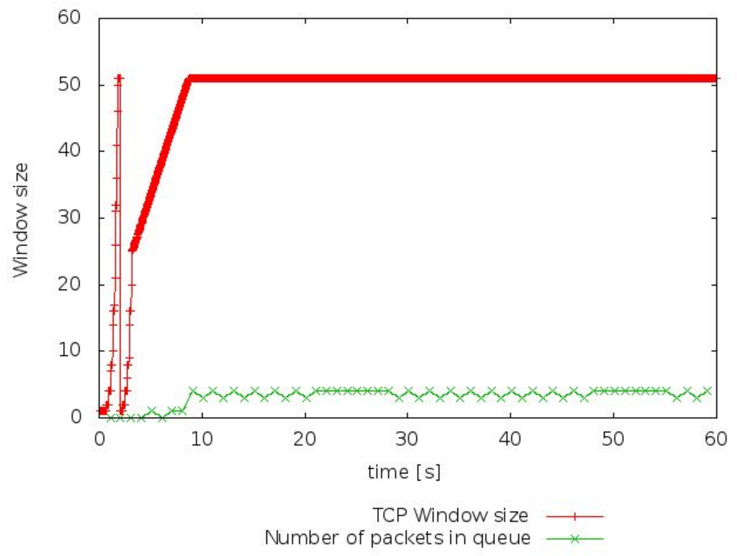
Graph A



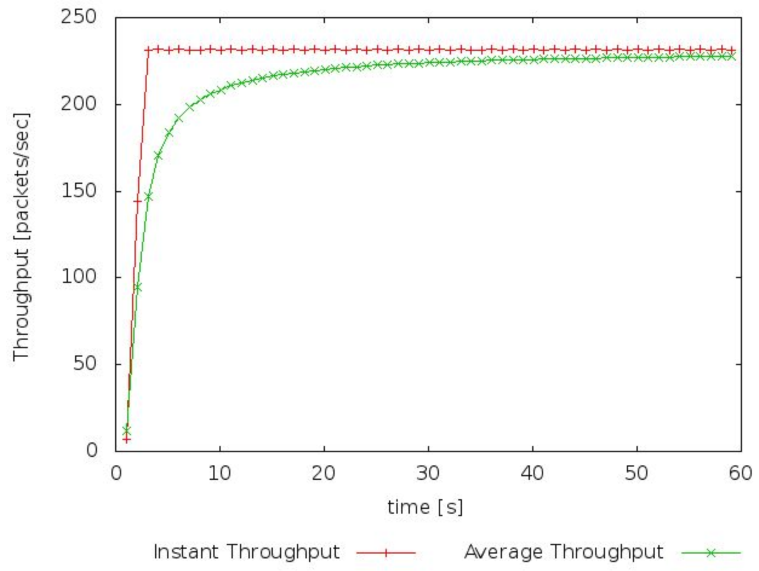
Graph B



Graph C

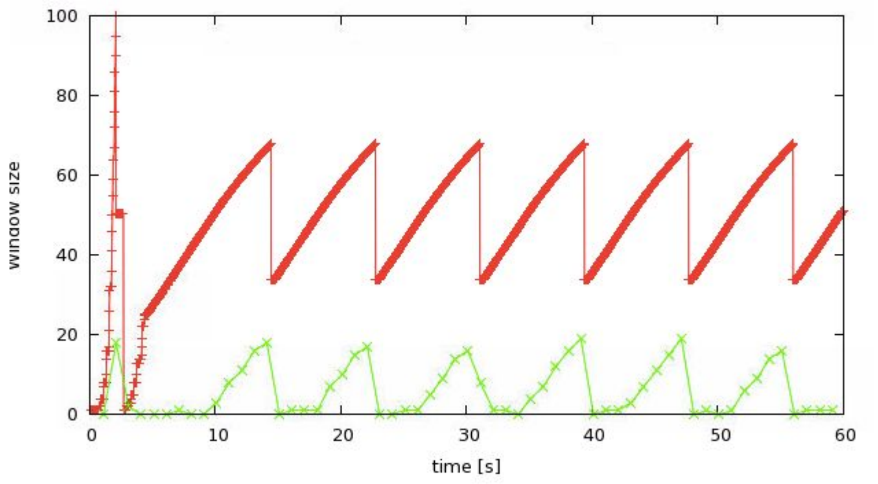


Graph D

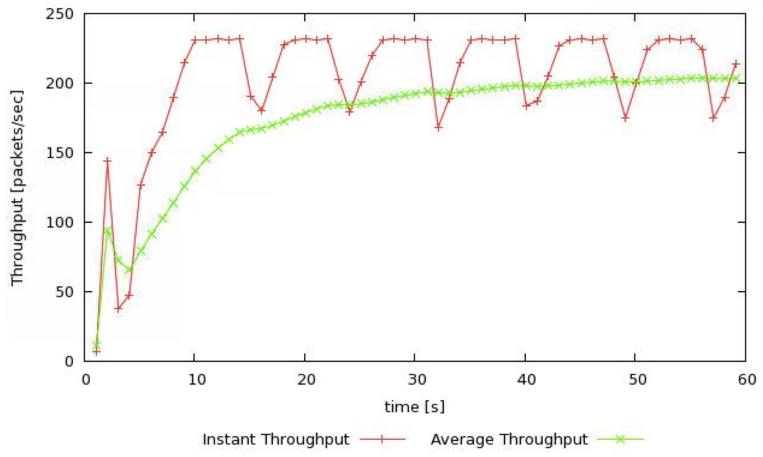


Graph E

3. We can see that when the initial max window size is 66 packets(Graph B), the oscillations stop after the second slow start. (Graph A is with 67 packets of initial max window size)
   Furthermore, when the initial max window size is 50 packets(Graph C), TCP stop oscillating when it reaches 50. (Graph D is with 51 packets of initial max window size)
   Therefore, from the observation in Graph E, the average packet throughput is close to 225 packets per second and in bps it is 225*8*540 = 97200 bps. Also it is almost near the link capacity(1 Mbps)



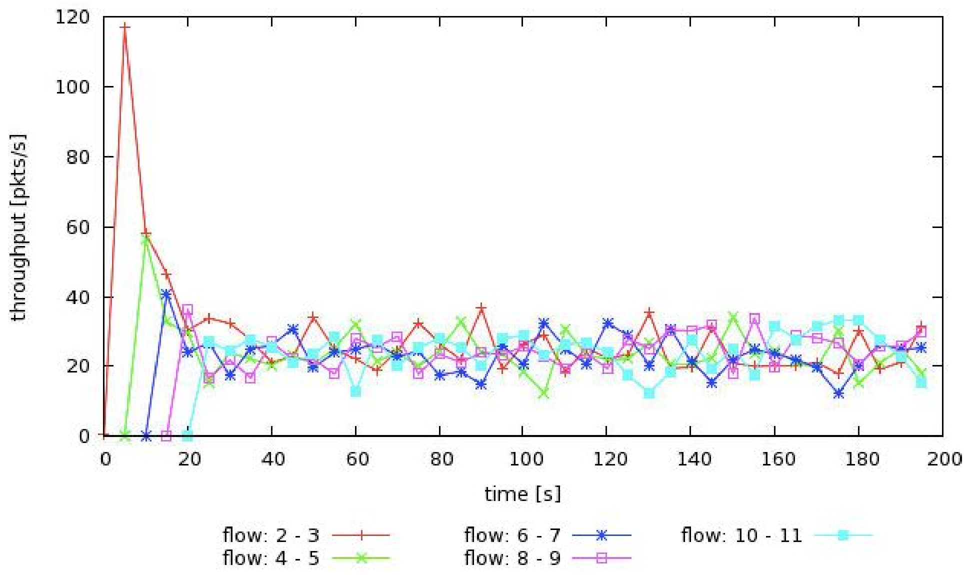
Graph A



Graph B

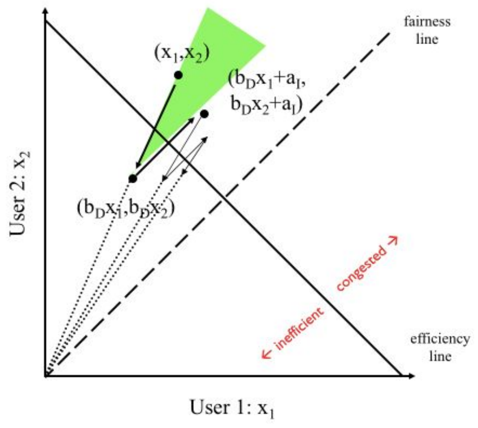
4. Graph A & B are run with 150 initial max window size. TCP Reno does not drop the window size to 1 MSS when a loss occurred, particularly when it received a triple duplicate ACK. Instead, it dropped to the half and entered into congestion avoidance. This is called fast recovery and TCP will deal with it differently from time-out loss event. In a word, TCP Tahoe always entered into slow start phase whenever the loss event occurred, while TCP Reno does not.
   In terms of throughput, TCP Reno might be better. The throughput is approximately 200 packets per second, which is more than 189 packets per second.

Q2:



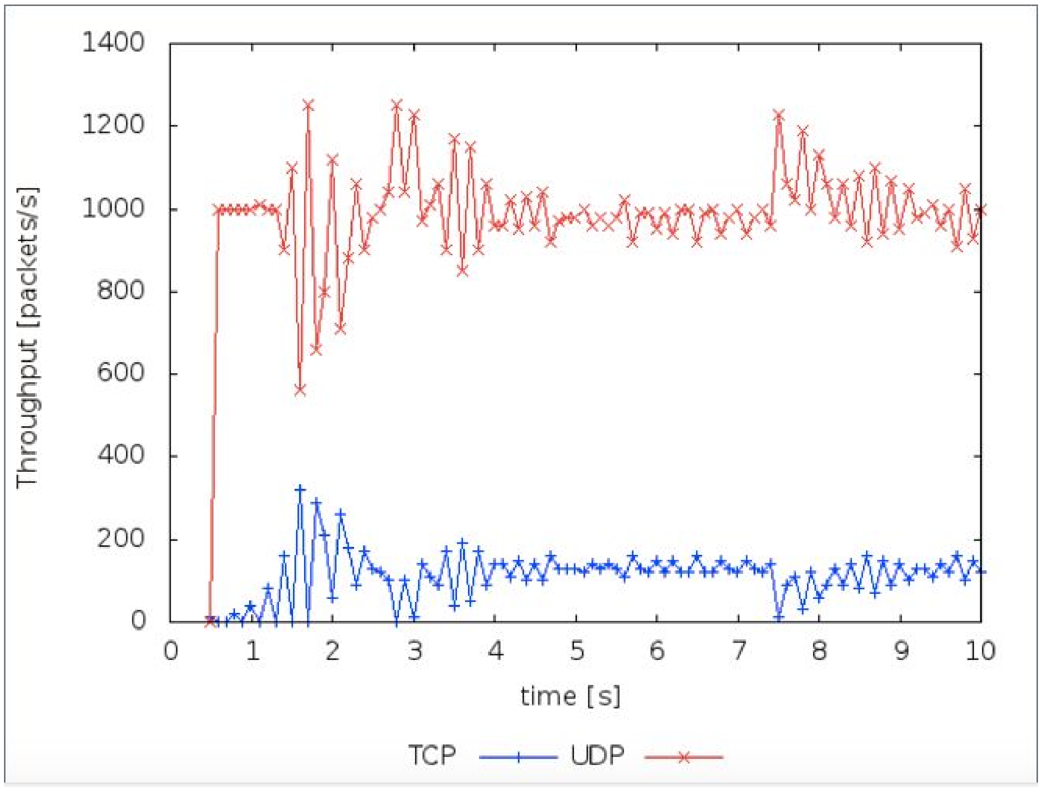
1. I believe that it is reasonably fair among all connections. This is because the throughput of nodes decrease a lot whenever a new connection starts. Eventually they all fluctuate among the similar range.

2. The new connection will drag the throughput of pre-existing TCP flows down the level where they could nearly equally share the capacity. The mechanism is congestion control(AIDM). The reason is illustrated below:



From this model, although it simplifies the example, we can still see that by using congestion control(AIMD), it will converge to fairness. In this example, as all the flows are under the same network condition, they would use the same mechanism.

Q3:



1. I would say UDP will have higher throughput than TCP. In the amination, the red represents UDP and the blue represents TCP.

2. This is because UDP does not have congestion control manner. It transfers the packet at relatively constant speed regardless of dropping packets during transmission. However, TCP has congestion control, and therefore it will detect congestion in the network and reduce the congestion window size (& the sending rate). Additionally, 'fast recovery' can make the TCP throughput relatively stabilized, and UDP is always sending at relatively constant speed and we can say it is stabilized per se.

3. Normally, using UDP could dominate the throughput in a shared link. Compared with TCP, the more aggressive UDP flows can suppress the TCP flows more. The downside is that using UDP can easily cause congestion without congestion control mechanism. If everyone started using UDP instead of TCP, it will lead to congestive collapse in which there is large amount of packet loss and the network would be congested.