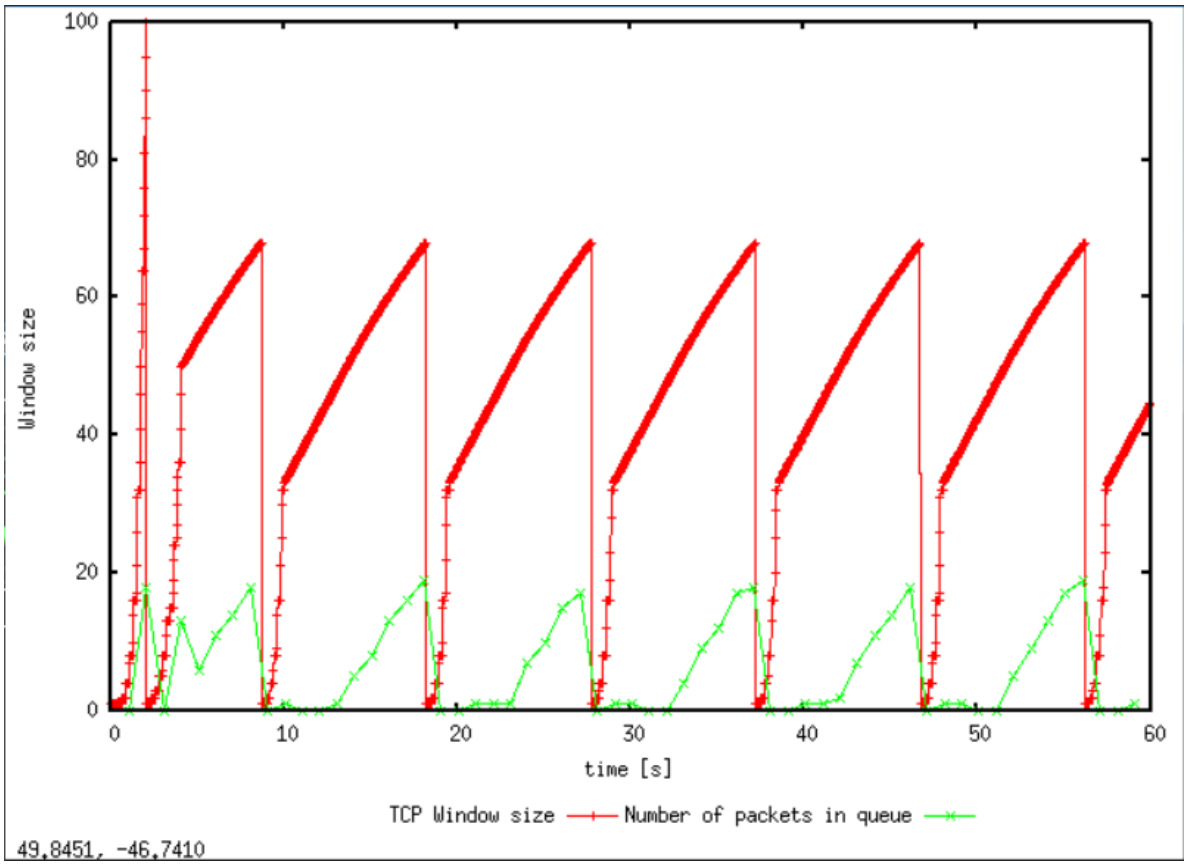
CS3331 Lab05

z5265106 Aniket Chavan

Exercise 1: Understanding TCP Congestion Control using ns-2

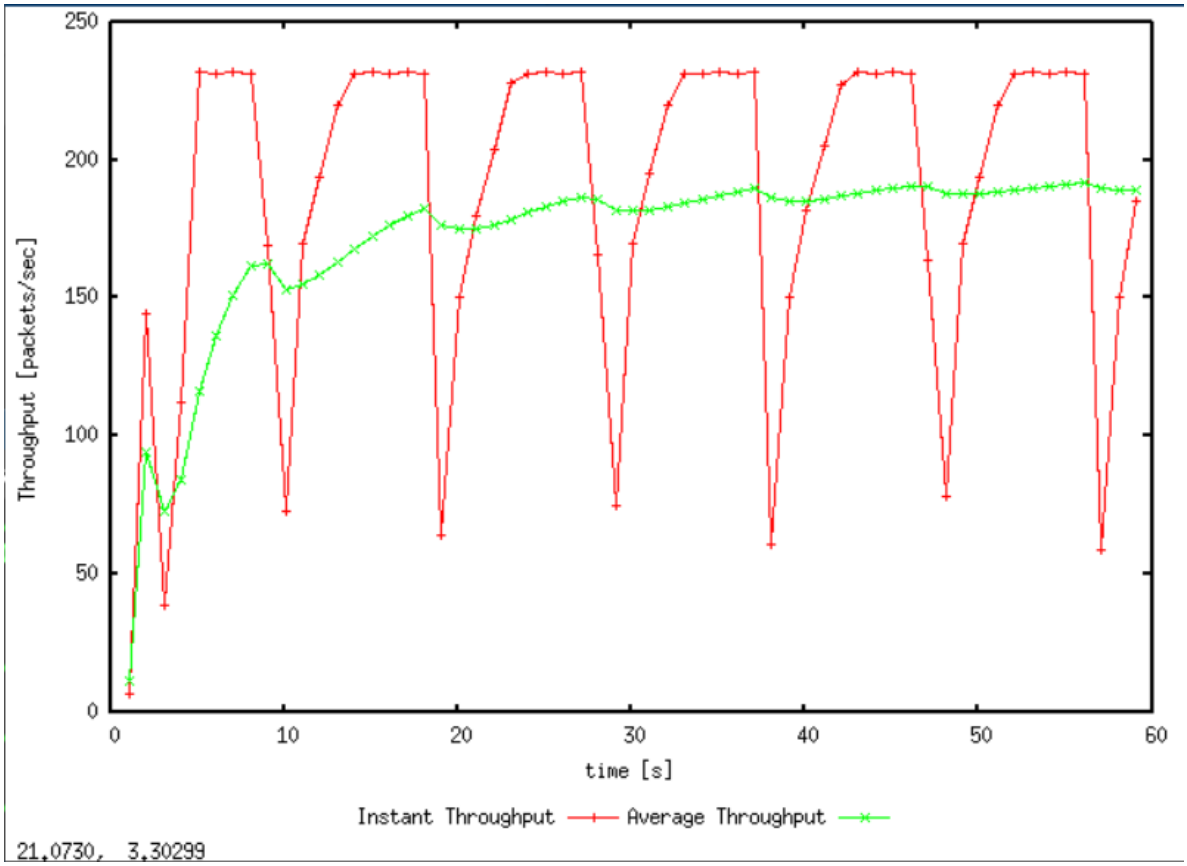
Question 1



- *What is the maximum size of the congestion window that the TCP flow reaches in this case?*
  - The max size of the congestion window that the TXP flow reaches in this case is 68-69 packets.
- *What does the TCP flow do when the congestion window reaches this value? Why? What happens next? Include the graph in your submission report.*
  - Initially uses a slow start and then linearly increases until another congestion is observed.
  - When it reaches this value, it drops all the window sizes to 0 in order to control the congestion of packet flow. Afterward it builds it up from 0 until reaching the maximum value which is considered the threshold. This process repeats again as needed.

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)

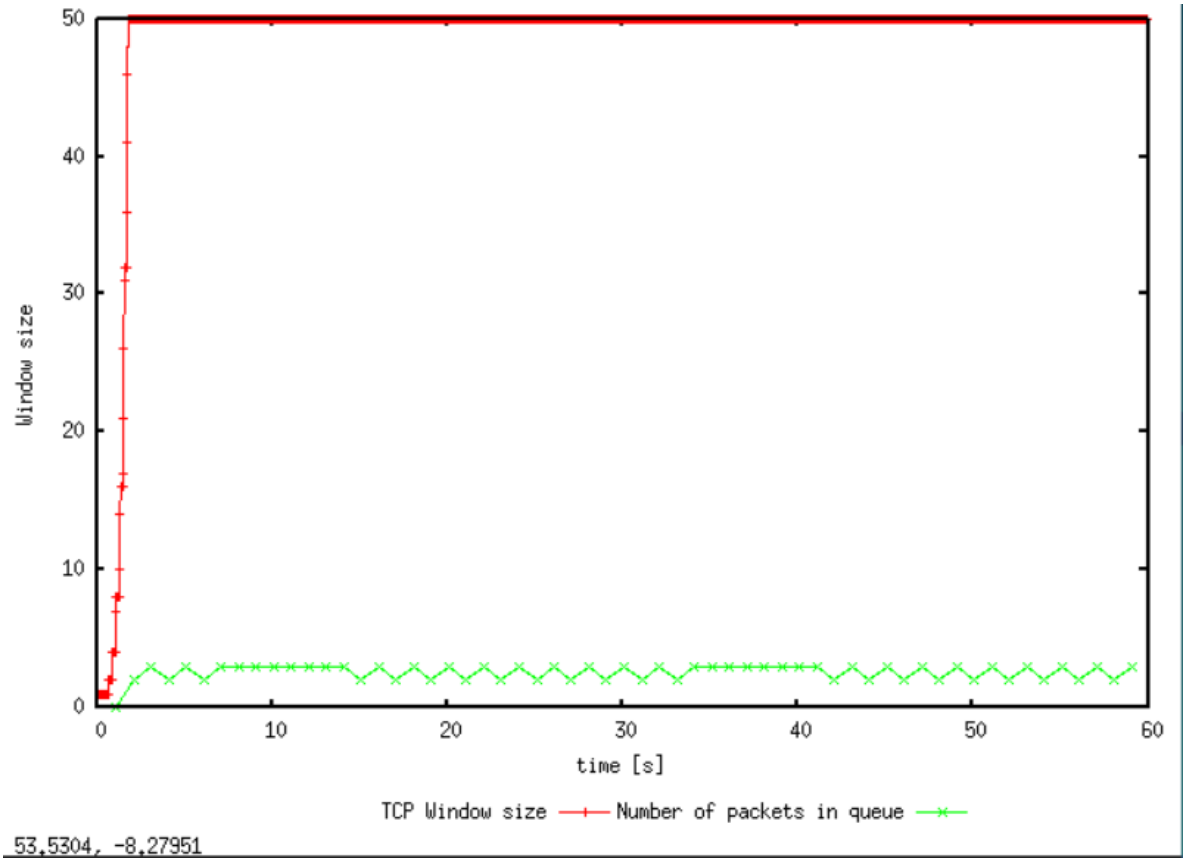


- Average throughput is 190 packets/sec
- since one packet have payload + IP header + TCP header
- $190(500 + 20 + 20) = 102600$ Bytes/sec
- 100.195 Kilobytes/sec = 801.562 Kilobits/sec

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)?



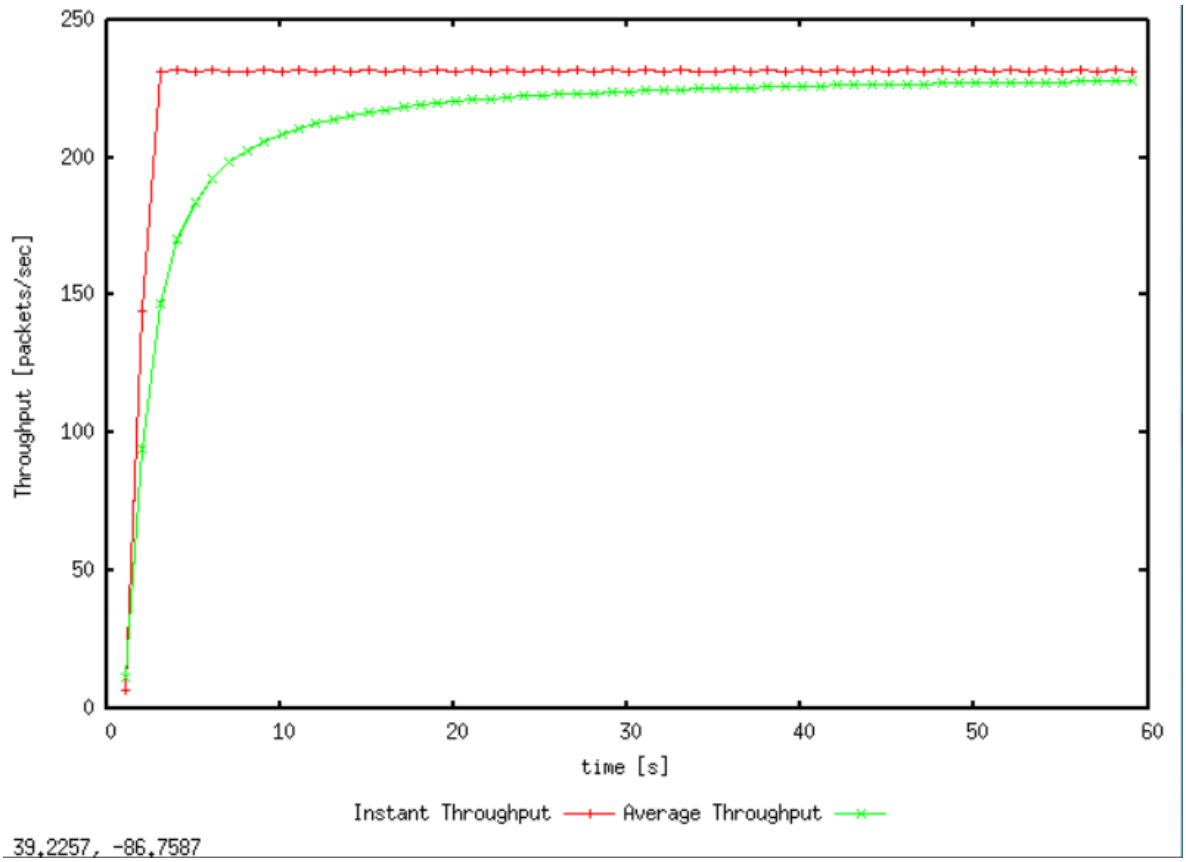
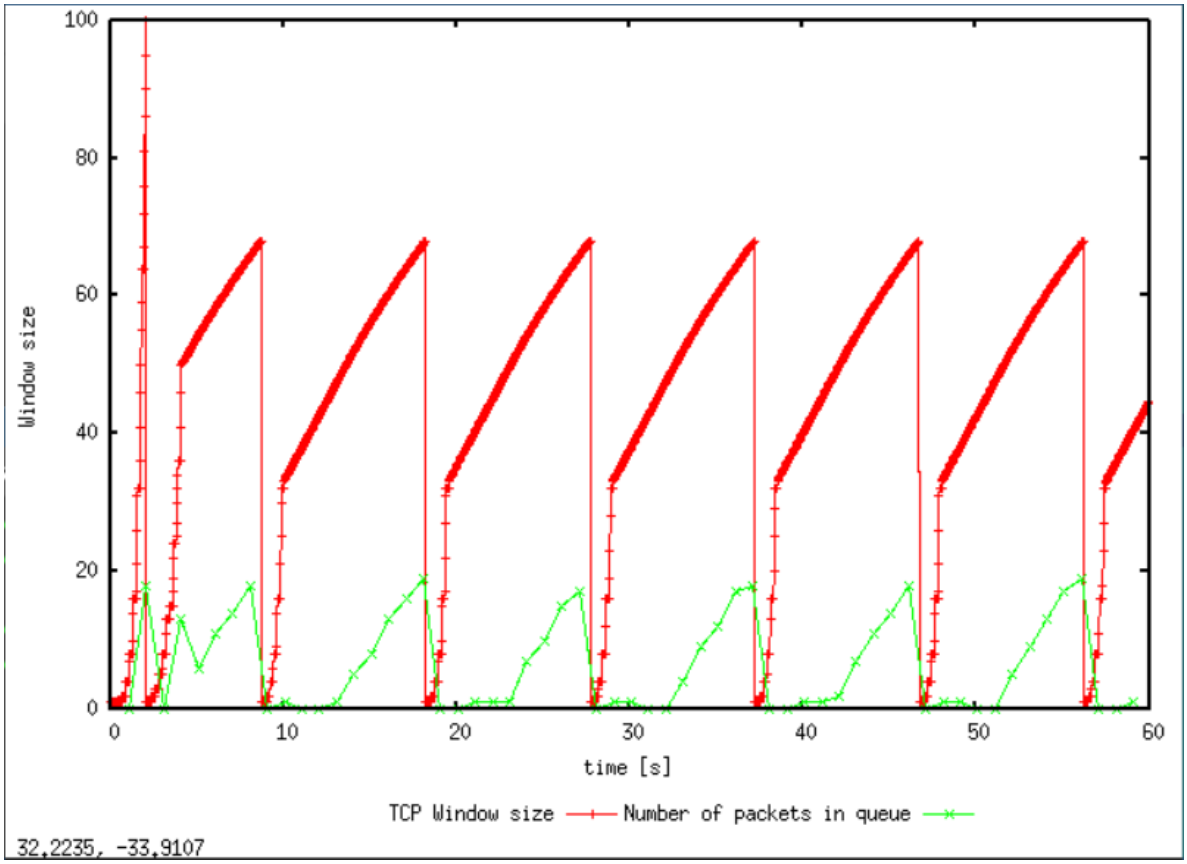


Image 1-2: max_cwd = 50



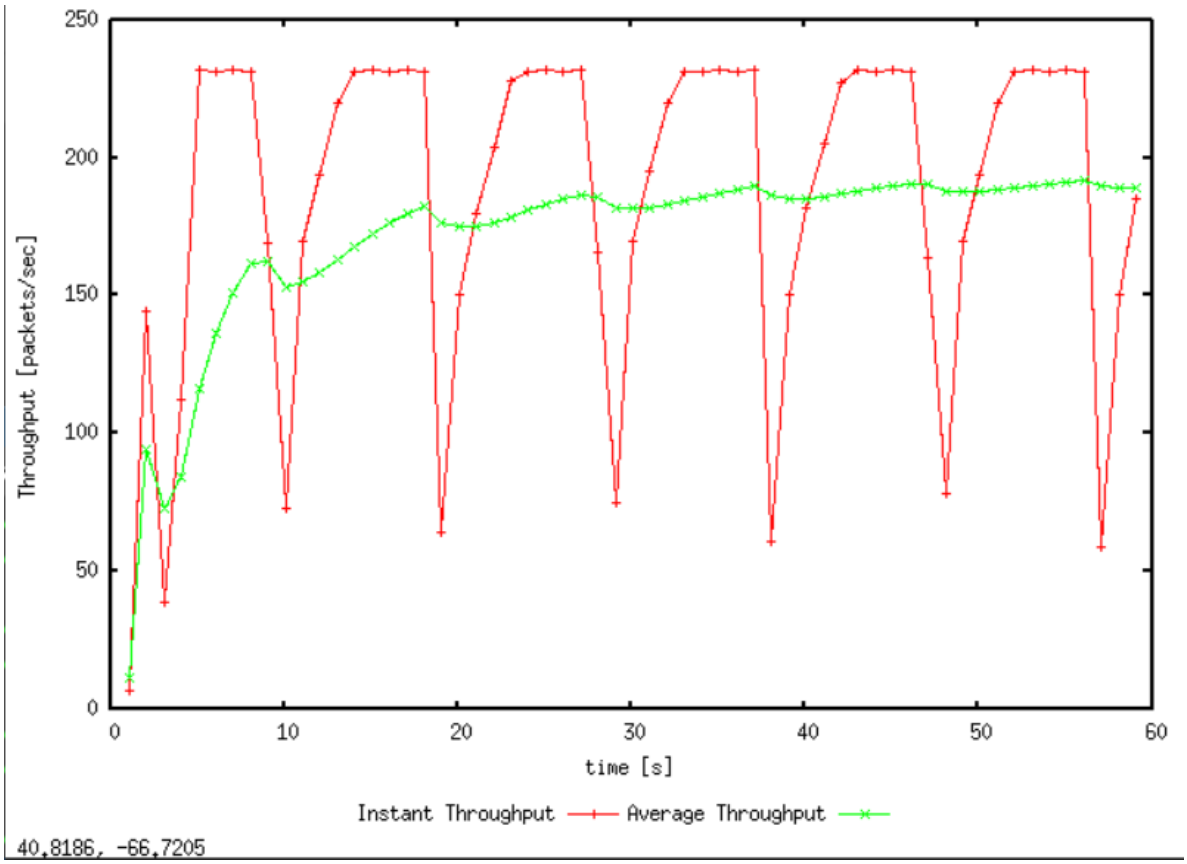


Image 3-4: max_cwd = 300

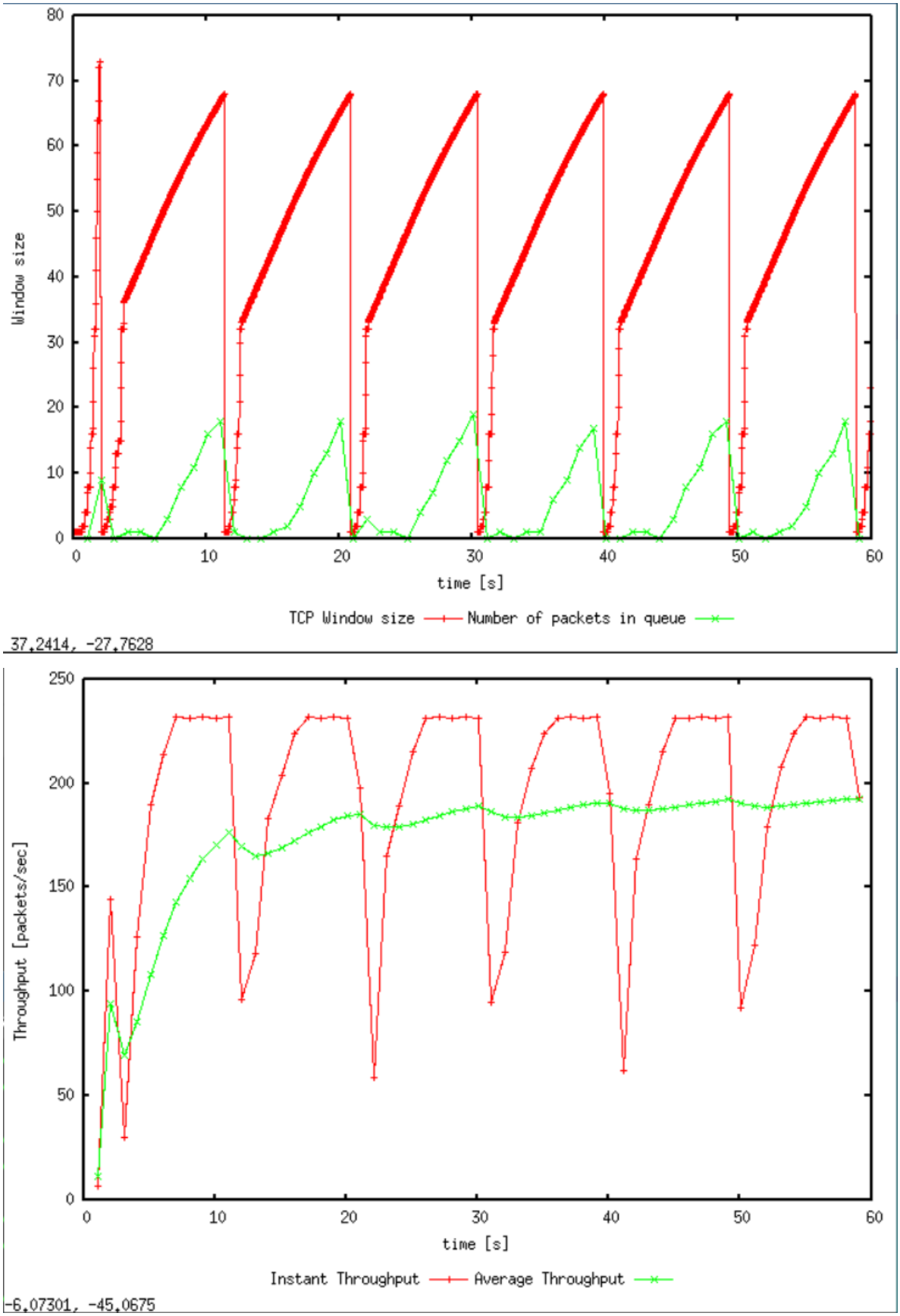
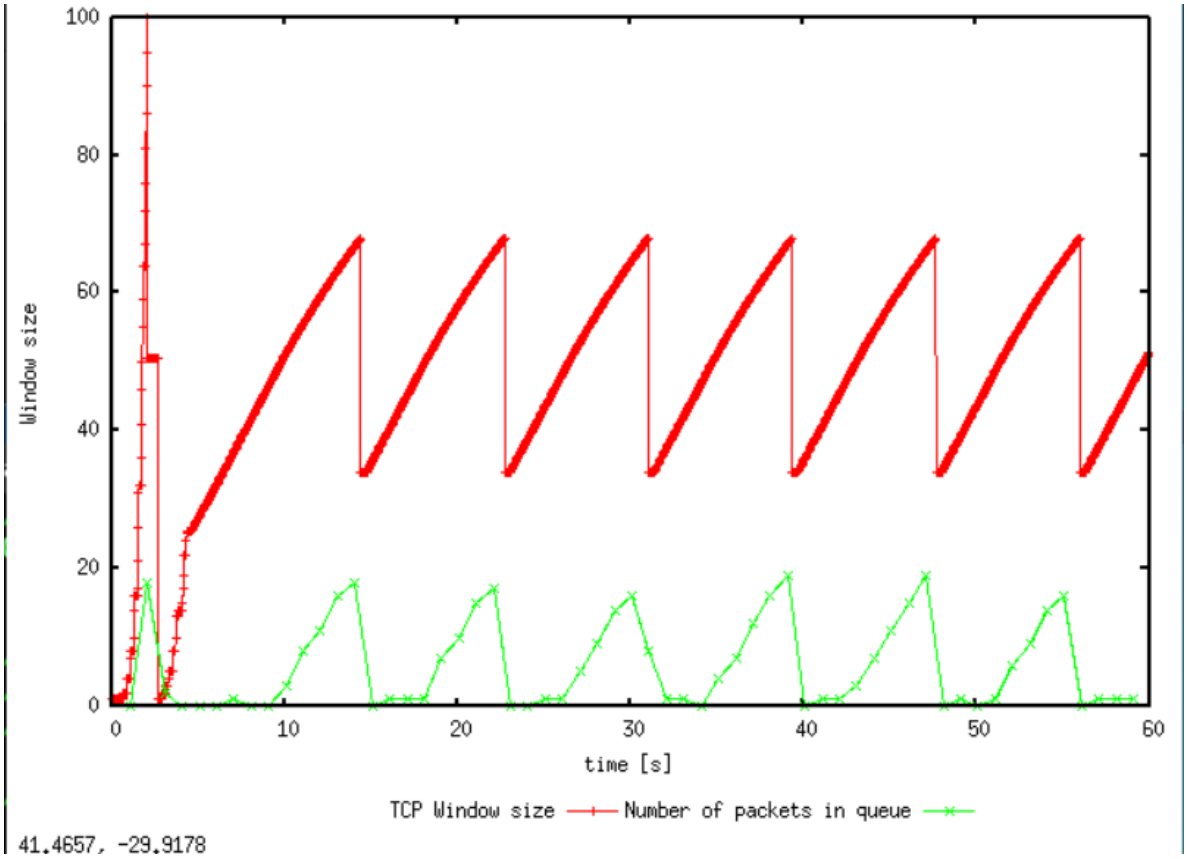


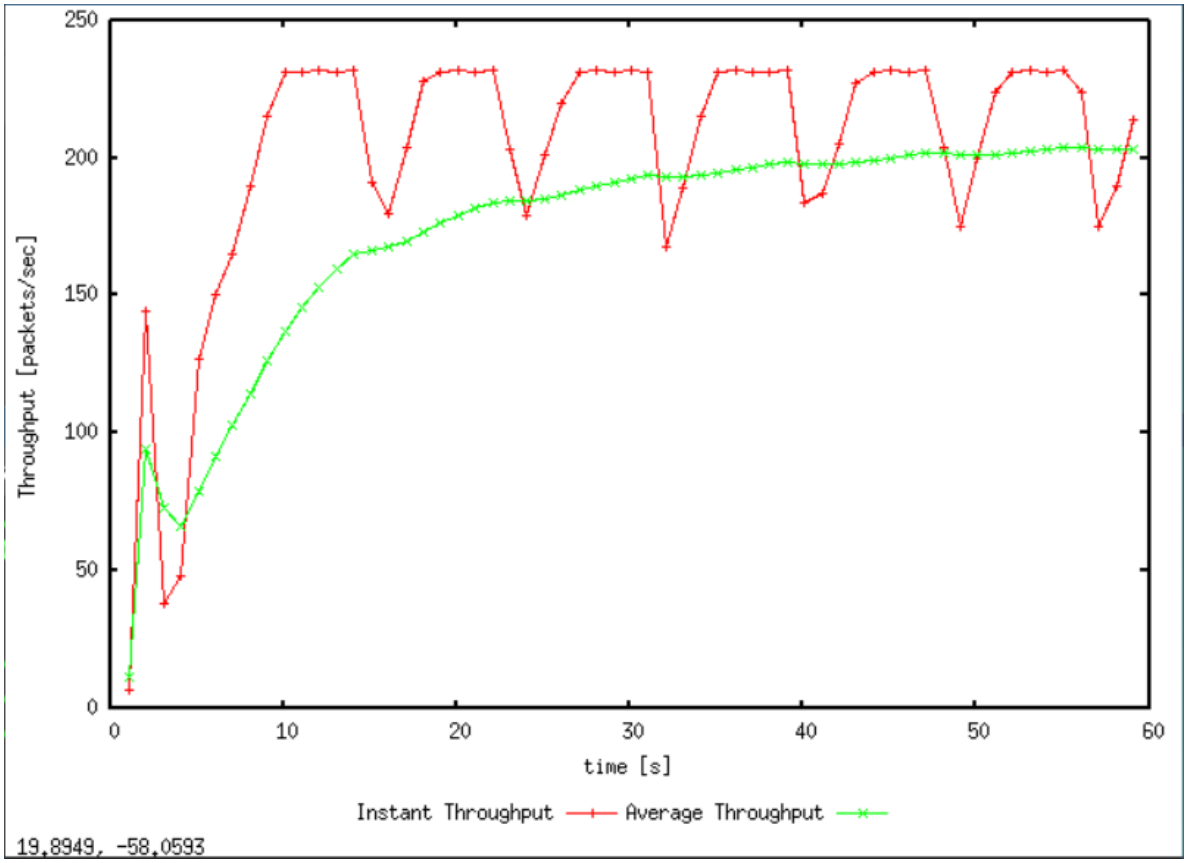
Image 5-6: max_cwd = 73

- As seen with max_cwd = 50, the maximum window size reaches 50 and is capped. With values higher than 65 however, it can be seen that the window size stabilises around 65-67.
- Average throughput is around 225 packets/sec
- Throughput is a bit lower but still reasonable comparing to the link capacity of 1Mbps

Question 4

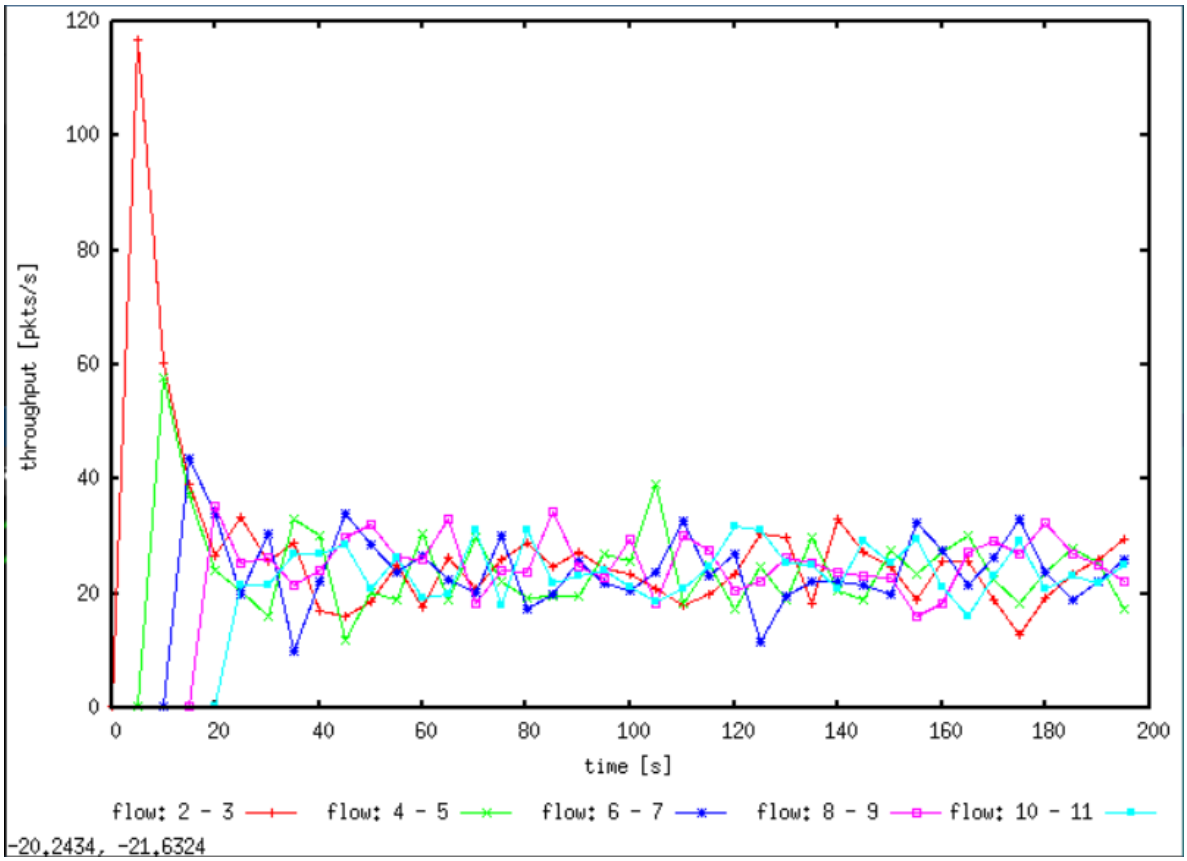
Repeat the steps outlined in Question 1 and 2 (NOT Question 3) but for TCP Reno. Compare the graphs for the two implementations and explain the differences. (Hint: compare the number of times the congestion window goes back to zero in each case).
How does the average throughput differ in both implementations?





- Instead of the window sizes dropping to zero, they halve and begin linearly increasing again.
- Throughput is a lot more stable here compared to TCP Tahoe, allowing the average trendline to reach 200 while for Tahoe the average stays below.

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.

When new users join the link, all existing TCP flows drop down to similar ranges of throughput to the new users which shows that TCP is a fair protocol.

Question 2

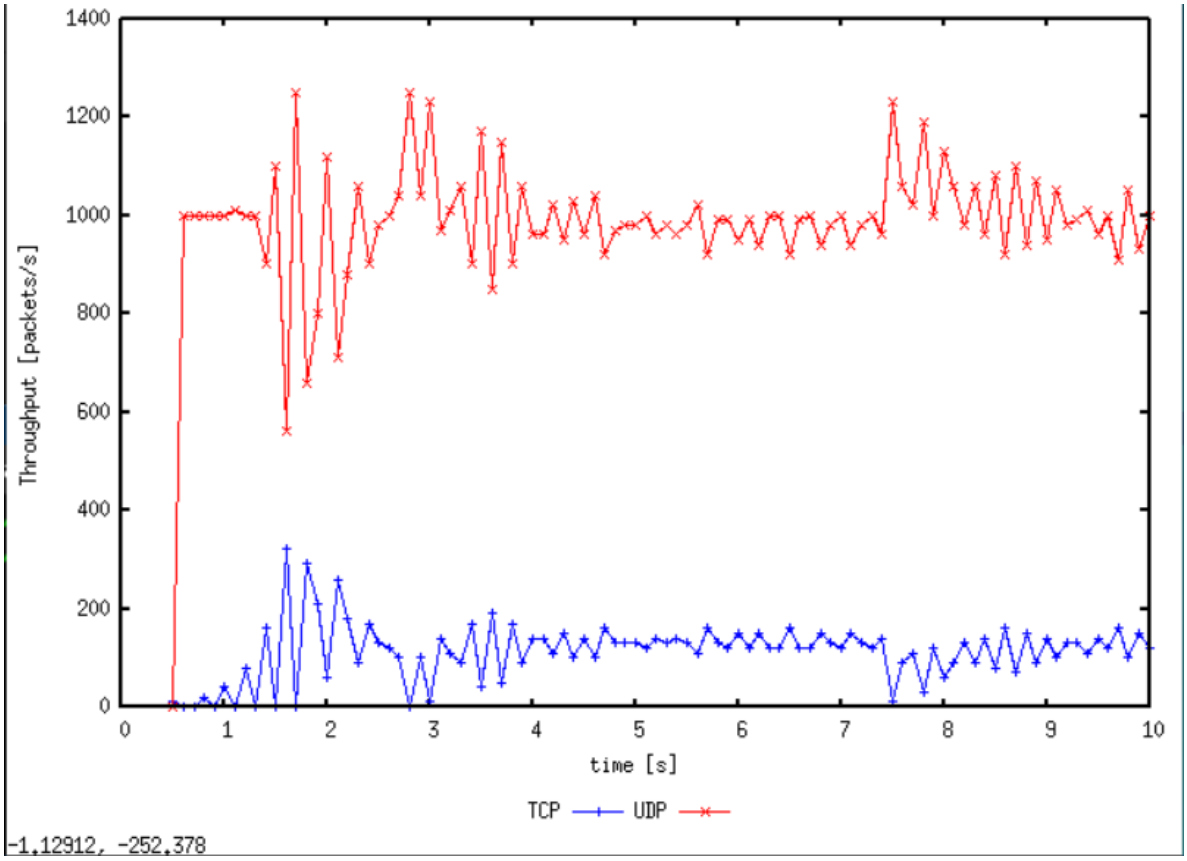
What happens to the throughput of the pre-existing TCP flows when a new flow is created? Explain the mechanisms of TCP which contribute to this behaviour.
Argue about whether you consider this behaviour to be fair or unfair.

TCP congestion control uses an AIMD protocol, when a congestion event occurs the window reduces by a multiple which is usually 2 to halve the window. When no congestion event occurs, the window increases linearly, usually by 1.

When a new flow enters the link, the pre-existing will experience a congestion event which will cause this flow to halve its congestion window and reduce its throughput. At this time both flows will linearly increase their speed until another congestion event occurs at which point whoever experiences the congestion event will halve their windows again.

Because the window increases linearly, flows with a larger congestion window experience a greater reduction in window size, however both flows will see reductions in window size (linearly) by their windows halving and over time and both will tend towards an equilibrium. The effect extends to n flows with each flow receiving 1/n of the link capacity.

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?

Although throughput of TCP is more stable than UDP, an overall throughput of UDP flow is significantly higher than TCP with approximately 1100 packets/sec. This is close to the maximum capacity of the link.

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 is expected to have higher throughput as it has no congestion control and has a constant transferring rate regardless of packet drops. Because we're using the same link, the much larger throughput from UDP will starve TCP packet 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?

- **Advantages**
  - Smaller headers means a larger payload can be carried per packet
  - No delay in establishing a connection
  - No congestion control limiting throughout
- **Disadvantages**
  - Additional application level headers have to be added to the payload decreasing space for data
  - Reliable delivery has to be implemented at the application level

    - Lack of congestion control inhibits QoS (Other users can broadcast at will)
    - Increased network congestion leads to lots of dropped packets, requiring more retransmits, further increasing congestion

Therefore,

- UDP is practically faster than TCP as it has higher throughput and transfer rate, but UDP does not ensure that the receiver will get all packets and all complete packets. TCP on the other hand is reliable data transfer and so will ensure that every packet reaches the destination.
- If everyone uses UDP instead of TCP, internet would be considerably faster but much more unreliable. The quality of service across networks would decrease dramatically as routers would become overwhelmed with the increased amount of unregulated throughput coming at them from every link