E1:

Q1: 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:

```
$ns  tpWindow.tcl  150  100ms
```

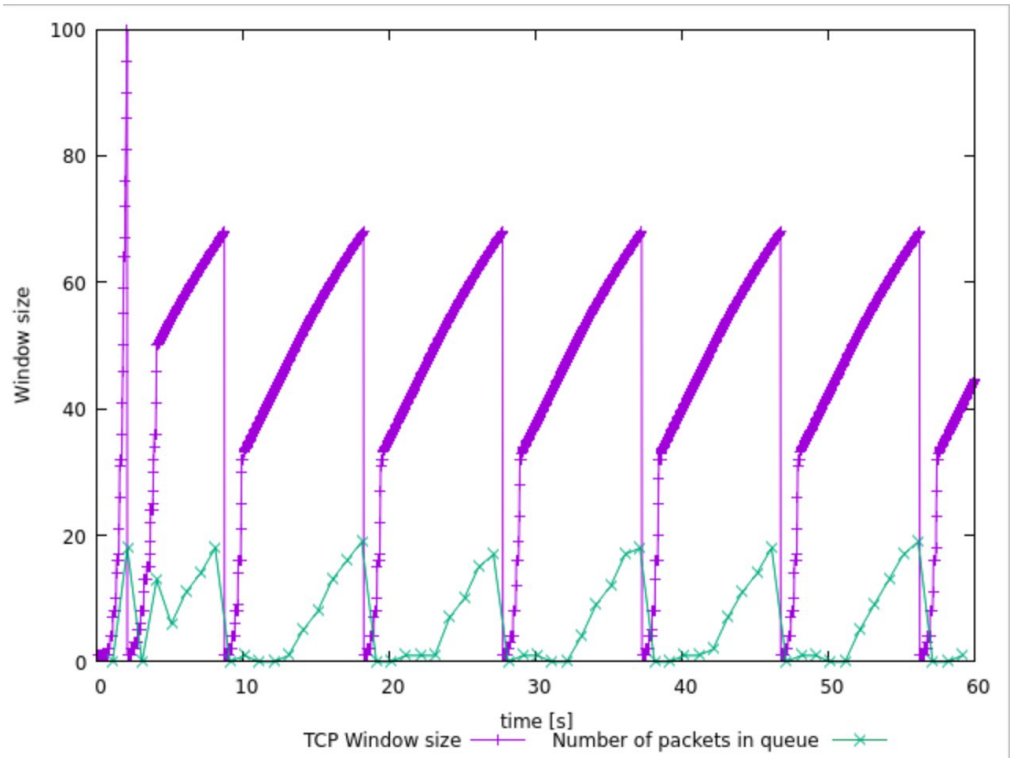
To plot the size of the TCP window and the number of queued packets, we use the provided gnuplot script Window.plot as follows:

```
$gnuplot  Window.plot
```

(a) In this case, what is the maximum size of the congestion window that the TCP flow reaches?

(b) What does the TCP flow do when the congestion window reaches this value? Why?

(c) What happens next?



(a) A: 100

(b) A: timeout or package loss when it reaches 100, then it will drop to one and slow restart.

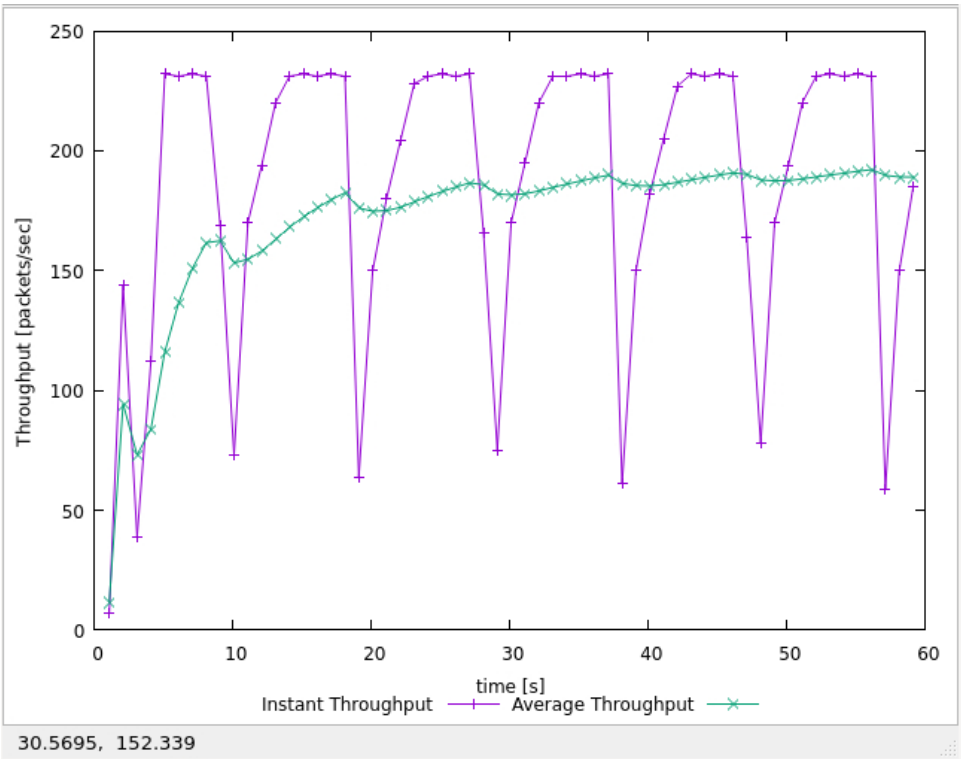
(c) A: aimed the target, and resend the package, keep sending until end

Q2: From the simulation script we used, we know that the packet's payload 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)

You can plot the throughput using the provided gnuplot script WindowTPut.plot as follows:

```
$gnuplot  WindowTPut.plot
```

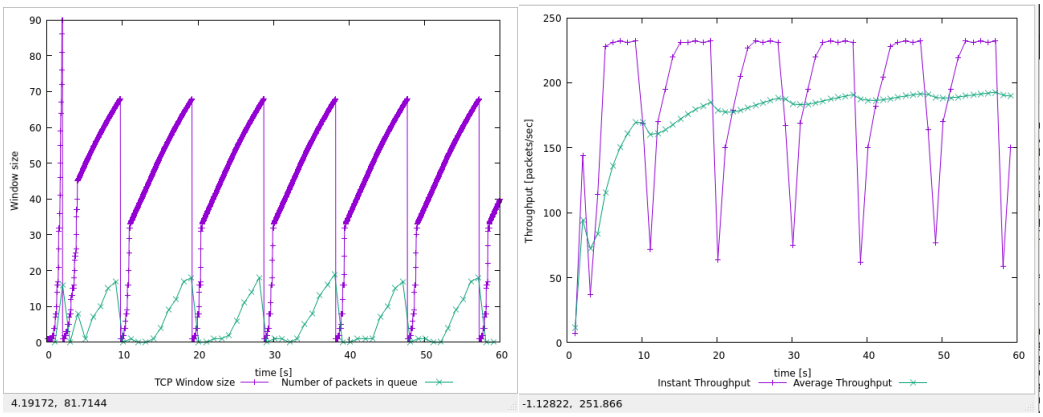
A:



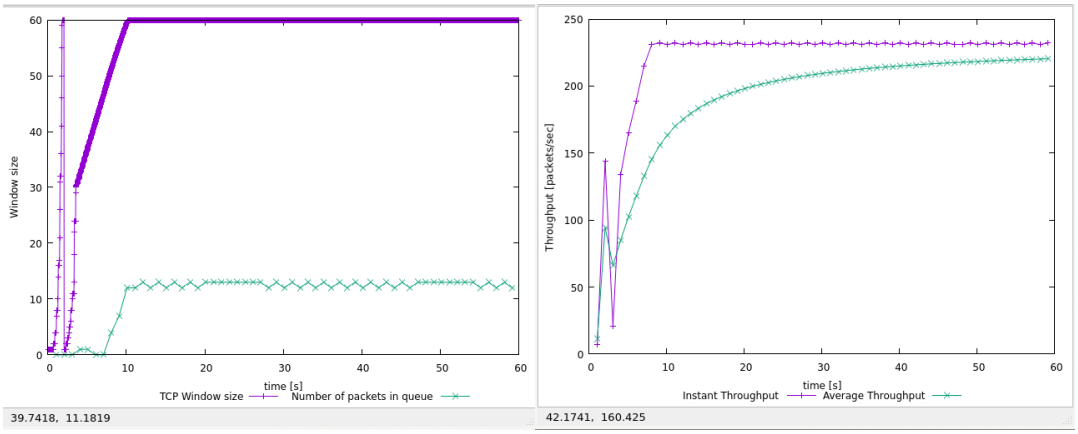
Question 3: Rerun the above script, each time with different values for the max congestion window size but the same RTT (i.e. 100ms). How does TCP respond to the variation of this parameter? Find the value of the maximum congestion window at which TCP stops oscillating (i.e., does not move up and down 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)?

A:

When max congestion window size is 90

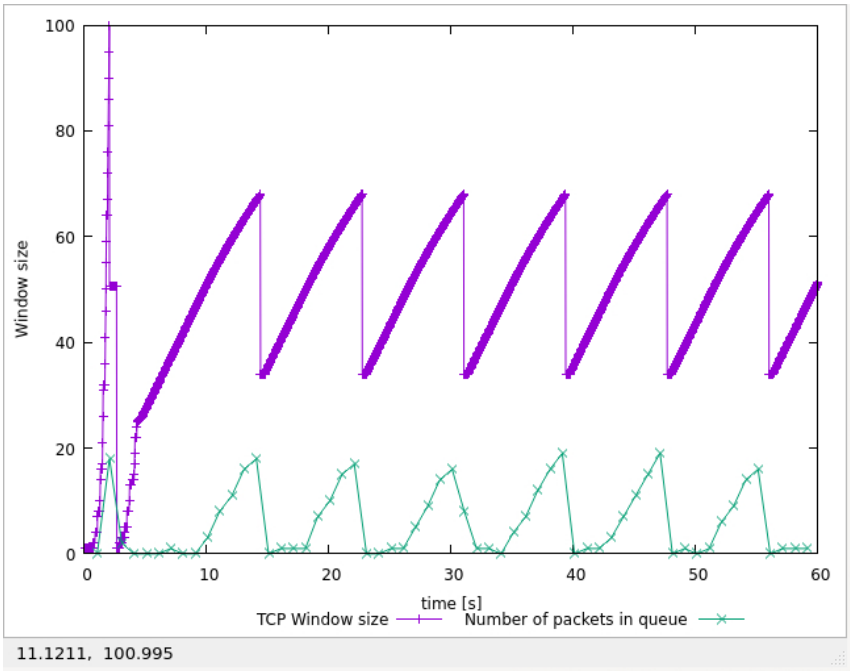


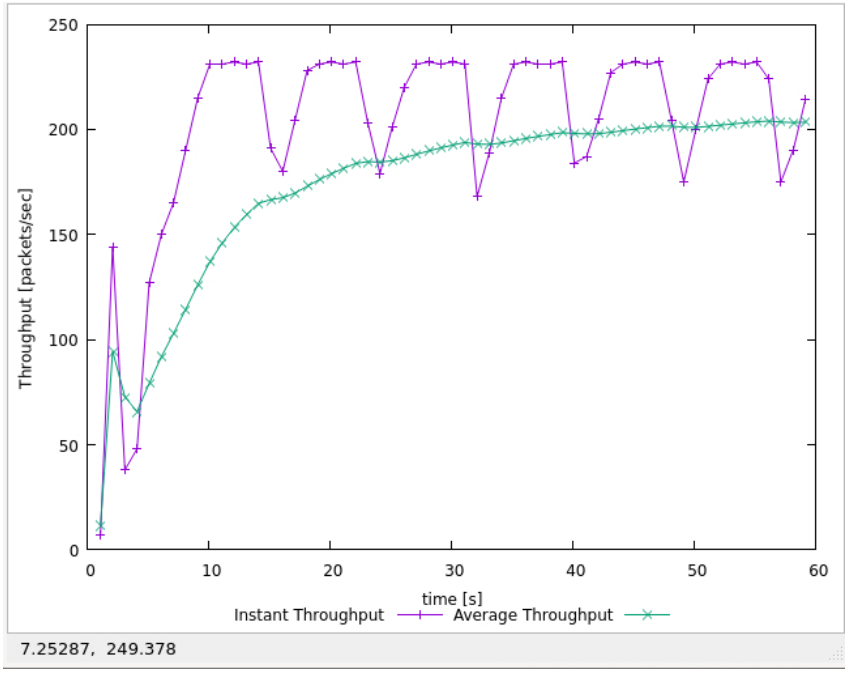
When max congestion window size is 60, there is one timeout event, and then have a slow start phase.



Question 4: Repeat the steps outlined in Questions 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 returns to zero in each case). How does the average throughput differ in both implementations?

A: a slow start at beginning, after timeout, a slow restart, then have a fast free transmit





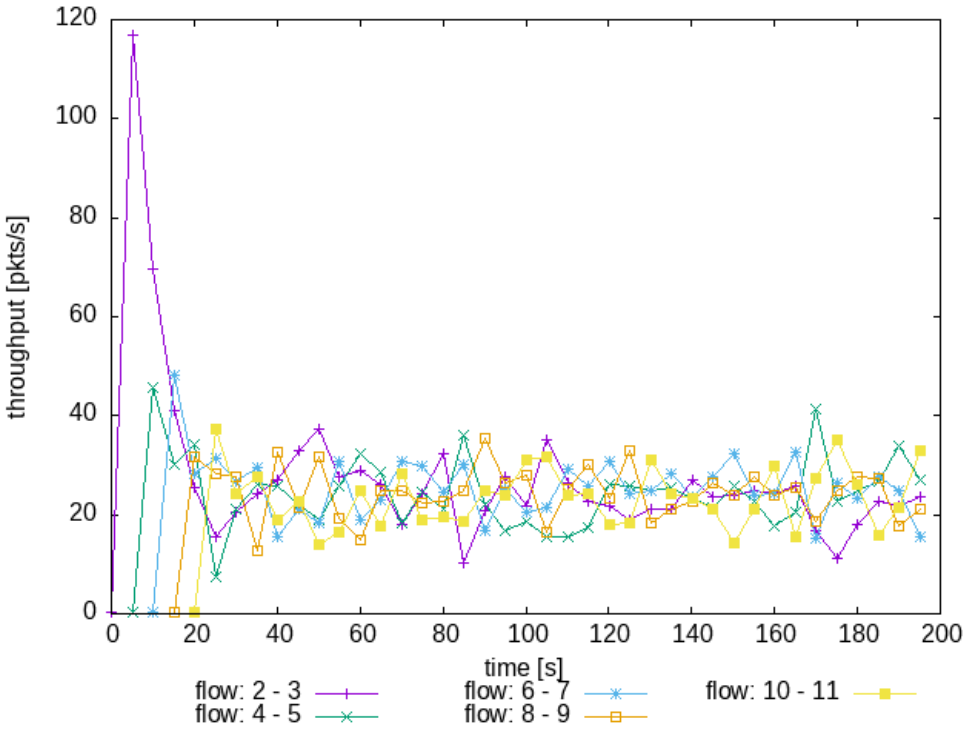
E2:

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.

A: Yes, it is. Because after we establish lots of TCP connection, the speed slows down, all TCP links share the speed.

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 behavior. Argue whether you consider this behavior to be fair or unfair.

A: When a new flow is created, the existing TCP flows will slow down, because the max speed is a definite value, and all TCP links should share this bandwidth.

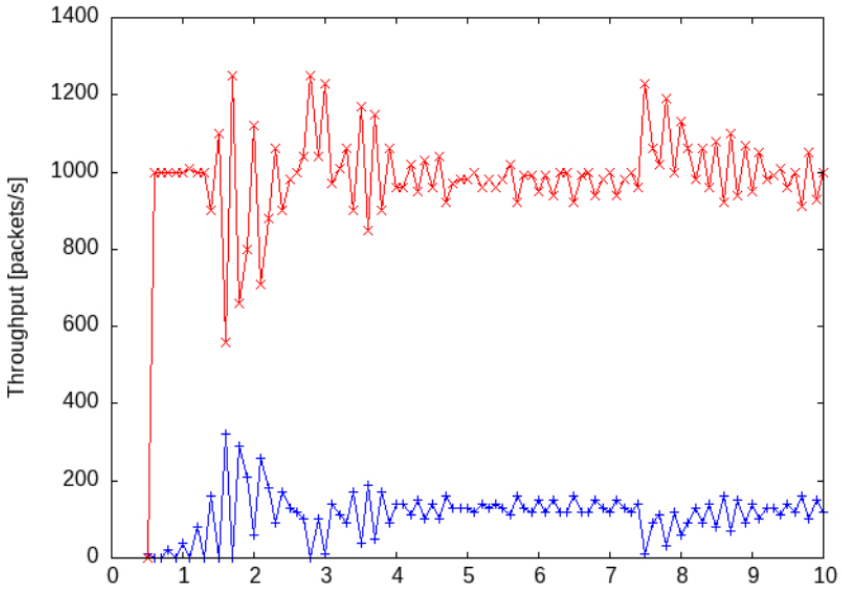


E3:

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

A: It all fluctuate around an average throughput as it uses a share bandwidth.

The orange one is UDP and blue one is TCP.



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.

A: UPD do not care about the package delivered or not, so it will not have any kind of congestion control, but TCP need to check, it needs more time, so throughput is lower the UDP

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?

A:

Advantages:

UDP is a lightweight protocol that has a lower overhead than TCP, which can result in faster transfer speeds. And it is useful for applications where speed is more important than accuracy or reliability, it does not require establishing and maintaining a connection between sender and receiver, which can reduce latency and processing overhead.

Disadvantages:

If packets are lost or corrupted during transmission, UDP will not be retransmitted, which can result in data loss or corruption. And it does not provide any congestion control mechanisms, which means that it can potentially flood the network with packets and negatively impact other flows on the same link.

If everyone started using UDP instead of TCP for file transfer, it could potentially lead to congestion and network instability. UDP does not provide any built-in congestion control mechanisms, so if multiple flows are competing for the same link, it could result in network congestion and packet loss. This could lead to reduced performance and potentially even network downtime if the congestion becomes severe enough.