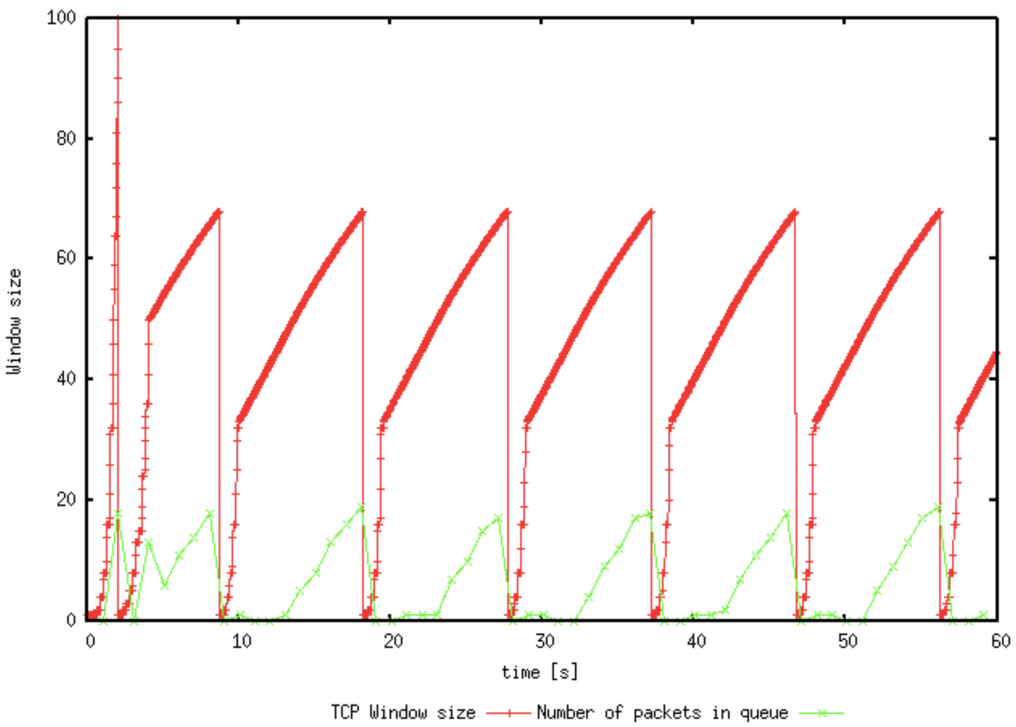
Lab Exercise 5: TCP Congestion Control and Fairness

Mubarra Khan z5161249 - 5th April 2020

Exercise 1: Understanding TCP Congestion Control using ns-2

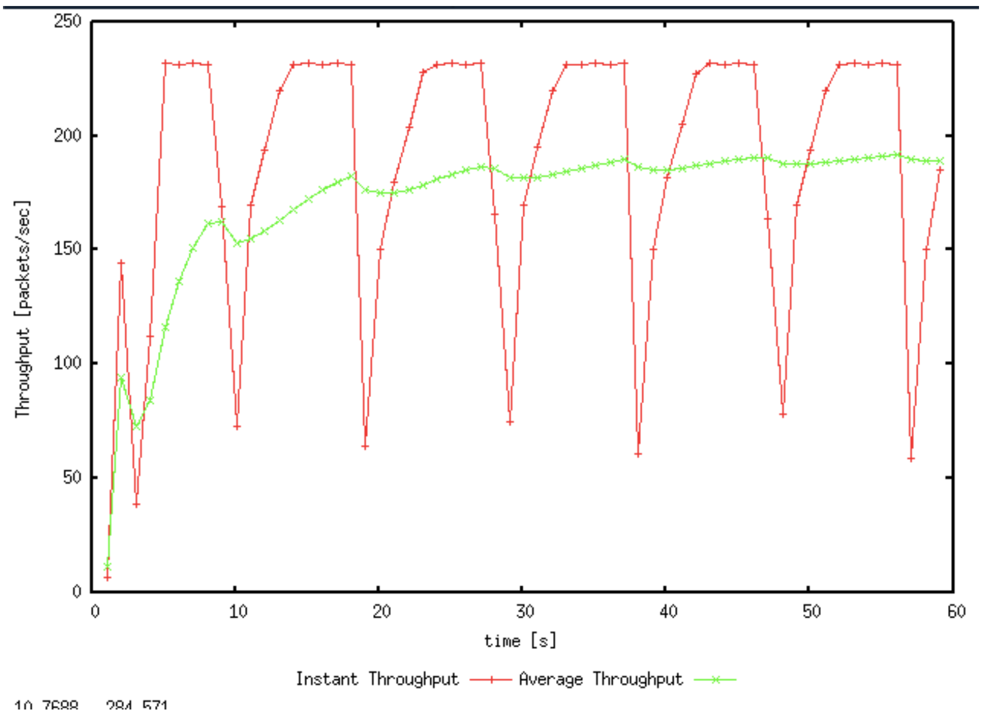
1. What is the maximum size of the congestion window that the TCP flow reaches in this case? What does the TCP flow do when the congestion window reaches this value? Why? What happens next? Include the graph in your submission report.

- The max window size is 100 . After timeout or triple dup ACK, the congestion window size dropped to 1 and the sshthresh is set to half of 100, 50. Slow start phase will start again with window size 1.



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)




```
59.100000000000001 41  0.0036883771140698092 185.0 1 188.97610921501706
```

The average throughput is 188.97610921501706 packets per second (from last line in WindowMon.tr)
Average throughput = (500 +20 + 20) * 188.97610921501706 = 102047.098976 bytes per second.

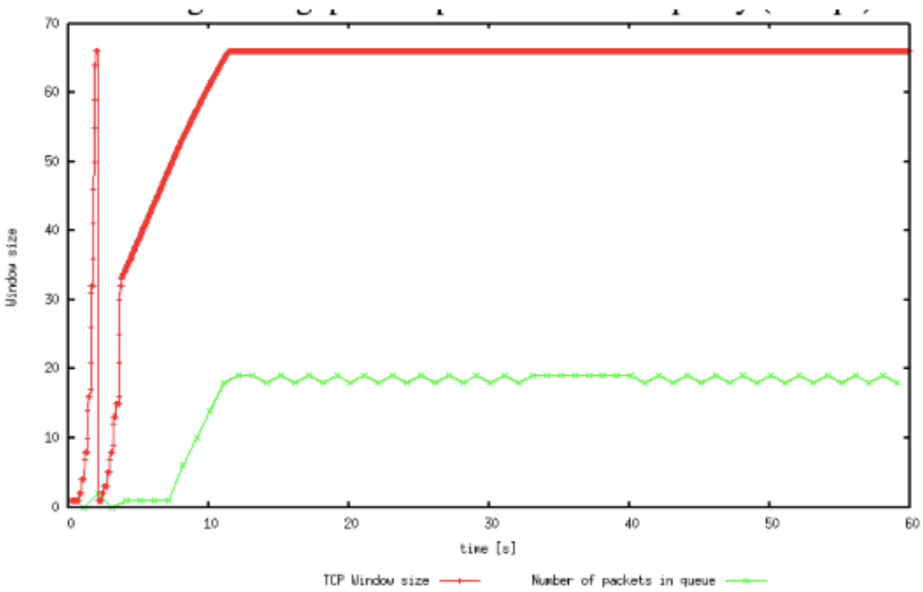
3. How does TCP respond to the variation of this parameter? Find the value of the maximum congestion window at which TCP stops oscillating (i.e., does not move up and down again) to reach a stable behaviour. What is the average throughput (in packets and bps) at this point? How does the actual average throughput compare to the link capacity (1Mbps)?

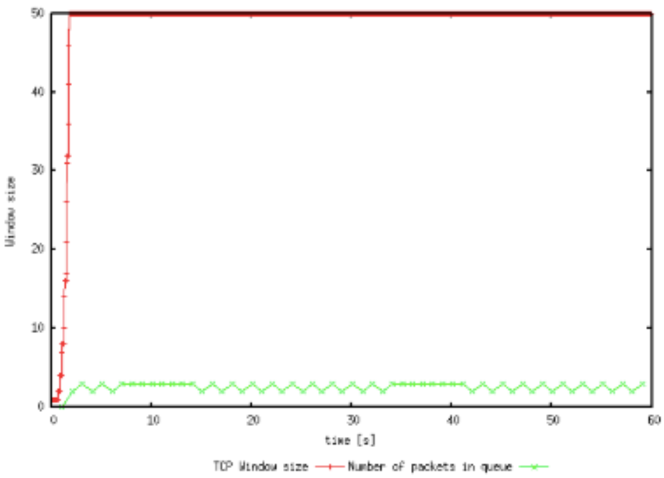
As the window size decreases, the number of oscillations decreases aswell. Once the max window size is down till 666, there there is only one oscillating and if the max window size is lower than 51, there will be no oscillating.
The Max window size to avoid oscillating is 50.

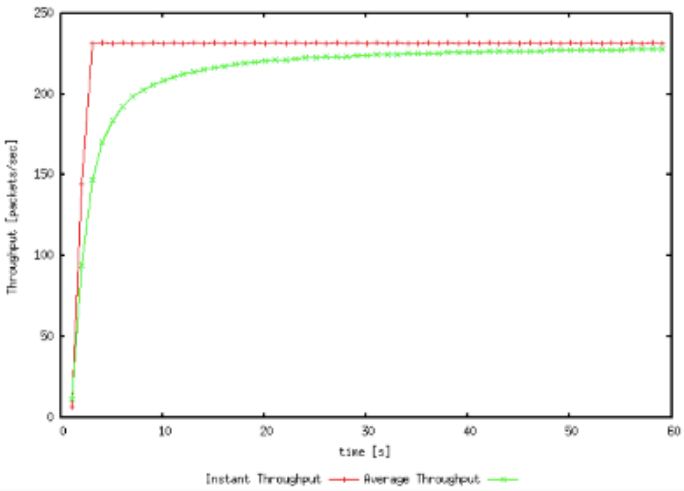
```
59.100000000000001 0  0.0 231.0 3 227.73037542662115
```

The average throughput is 227.73037542662115 packets per second.

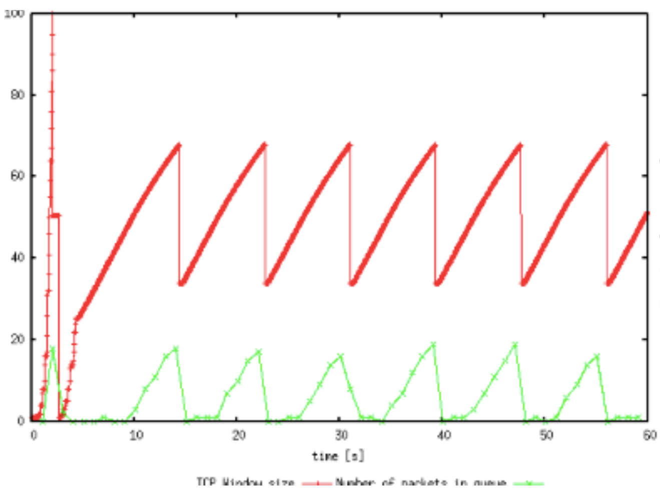
Average throughput = (500 + 20 + 20) * 227.73037542662115 = 122974.40273 bytes per sec
125000 bytes = 1000000
Link capacity = 100 * (122974.40273/125000) = 98.38 %

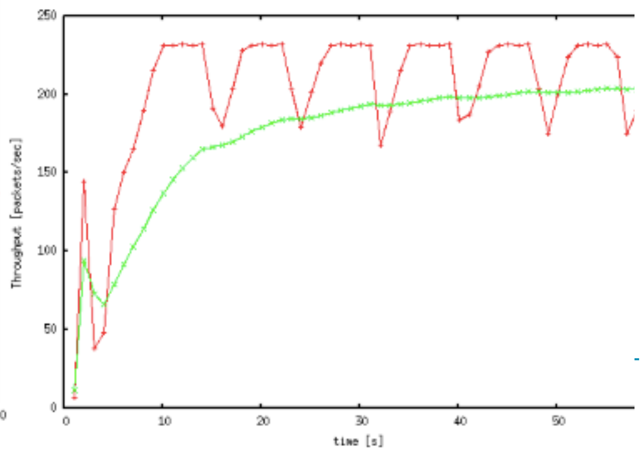






4. Repeat the steps outlined in Question 1 and 2 (NOT Question 3) but for TCP Reno. Compare the graphs for the two implementations and explain the differences. (Hint: compare the number of times the congestion window goes back to zero in each case). How does the average throughput differ in both implementations?

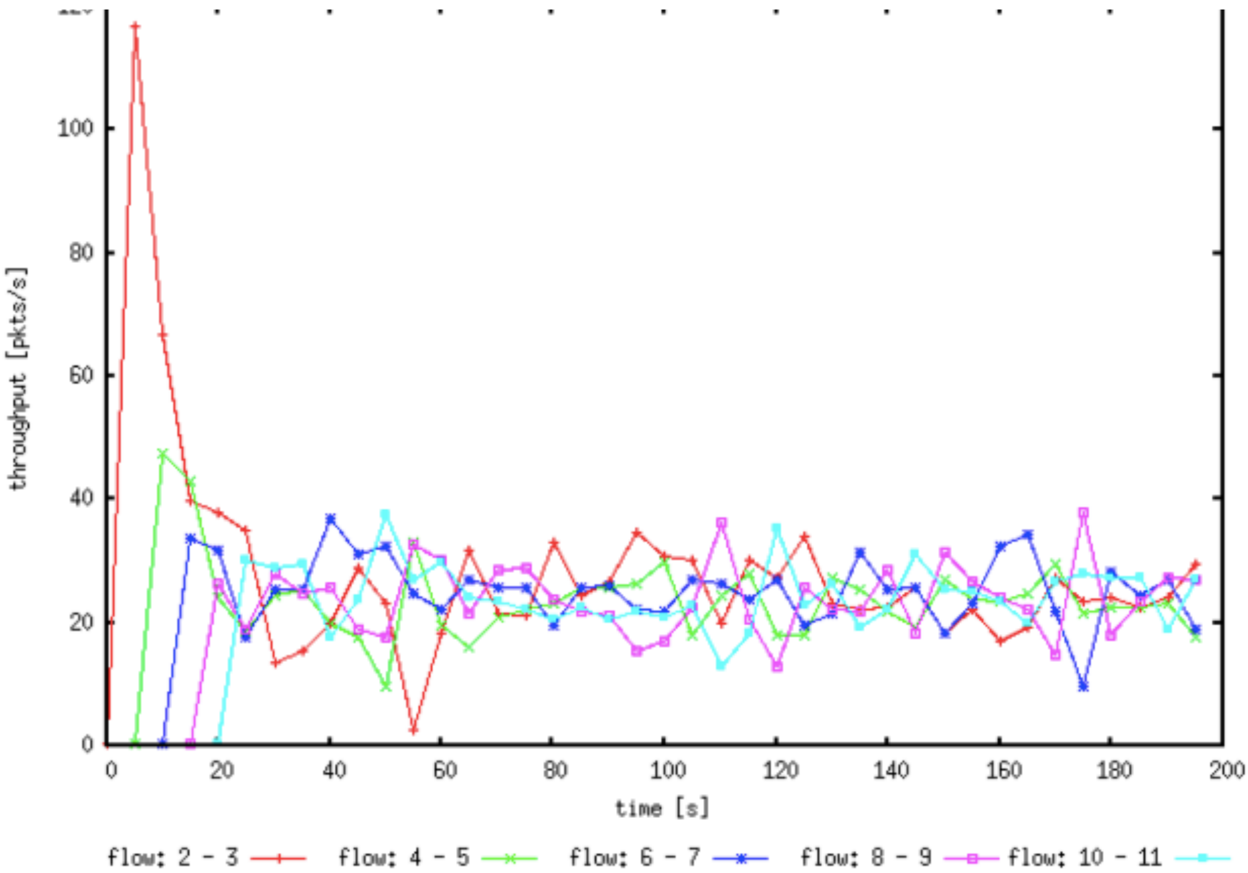




The window size only hit zero once after slow start phase fro Reno. The average throughput is higher for Reno (203.41296928327645) than Tahoe (188.9761092150176).

Exercise 2: Flow Fairness with TCP

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.



Yes, each flow does get an equal share of the capacity of the common link. Although at the start each flow has a different share of the common link , as the time increases, each flow increases and decreases till each flows average is between 20-40 packets per second.

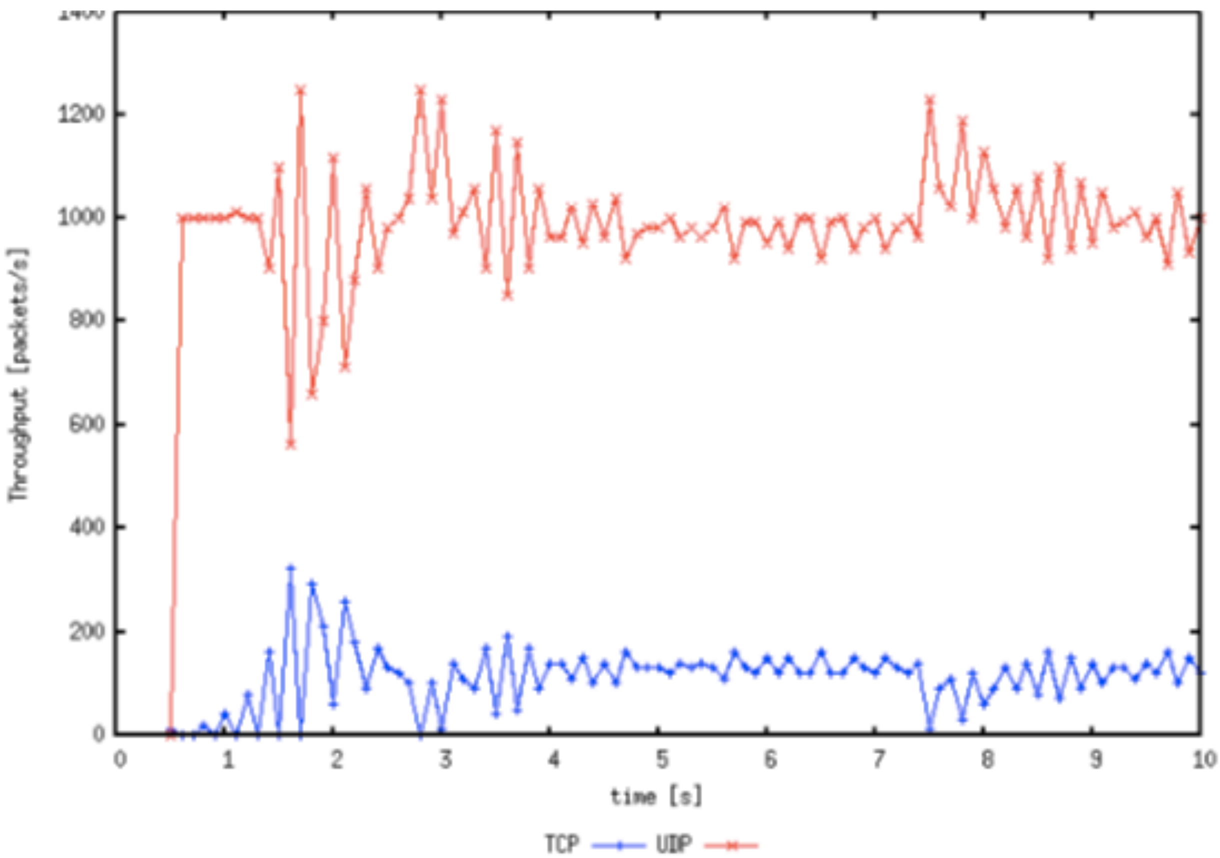
2. What happens to the throughput of the pre-existing TCP flows when a new flow is created? Explain the mechanisms of TCP which contribute to this behaviour. Argue about whether

Throughput of pre existing Tcp flows decreases when a new flow is created to provide equal share. The congestion window size increases rapidly during slow start phase which causes the congestion. Thus, all flows will adjust to adapt the network. The behaviour is considered to be fair.

Exercise 3: TCP competing with UDP

1. How do you expect the TCP flow and the UDP flow to behave if the capacity of the link is 5 Mbps ? You will observe packets with two different colours depicting the UDP and TCP flow. Can you guess which colour represents the UDP flow and the TCP flow respectively

TCP flow has congestion control and UDP flow doesn't. thus, TCP will oscillate while UDP will utilise the capacity which means that UDP will have higher throughput compared to TCP.
In the figure below UDP is red -> with higher throughput
And TCP is blue -> Lower throughput



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 achieves a much higher throughput than TCP since it has no mechanism for congestion control . UDP is simply a best effort protocol, whereas TCP has congestion control mechanisms built in to prevent overloading on the link and provide a stable connection. TCP congestion control will change its window size based on conditions as seen above in figure 15, there are times where the throughput drops to 0, which is when time occurs and the window size changes. There are also drops indicting triple duplicate ACK's. In general, there is fluctuations in both UDP and TCP which seem to come from a network that has changing conditions. The tcp line on the plot has a similar plot to the UDP line, as when the UDP throughput drops the same loss can be observed on the TCP plot.

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 of using UDP instead of TCP
1. Overall UDP has much higher throughput than TCP
2. If only a few people were using UDP network they would get much faster data transfer, as it will transfer based off the link bandwidth.
3. Smaller packet size

Disadvantages of using UDP instead of TCP
1. There is no form of congestion control, so a link could end up collapsing and blocking due to packets being send to a blocked link.
2. There is no form of guarantee that you will receive the file, and receive it in order without any data corruption (UDP is a best effort protocol).
3. Unaware of package loss and corrupted package.

If everyone decides to use UDP instead on TCP for the advantages. The network is expected to be blocked and new packets being sent, would be dropped as the queue would be full.