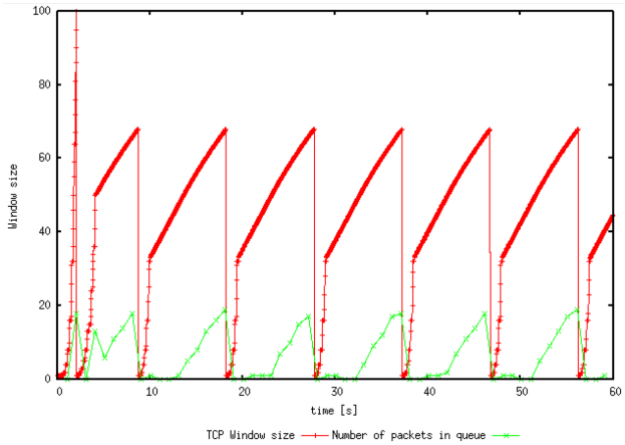
Lab 5: TCP Congestion Control and Fairness

Question 1: 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). 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.

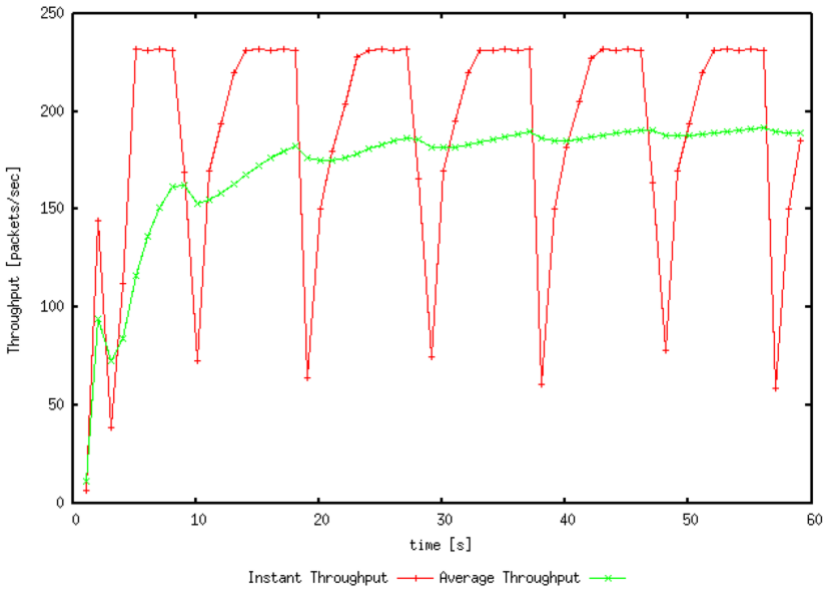
- Maximum size of congestion window: 100 packets
- When the congestion window reaches 100 packets, either a timeout or 3 duplicate ACKs have occurred. This resets the congestion window and the new threshold becomes 50 packets (half of 100).
- The slow start phase will initiate with a window size of 1. The congestion window size will increase up to a maximum of 50. Once the window size reaches 50, congestion avoidance occurs. This is a linear increase until packet loss occurs. This can be seen in the plot below



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)

The average throughput can be seen in the WindowMon.tr file

- Throughput is 188.97610921501706 packets per second.
- There are 540 bytes per packet including the header. Therefore:
    - Throughput is also 540 * 8 * 188.97610921501706 = 816376.7918 bps



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)?

- TCP will set the slow start threshold as the value of max congestion window size. It will also make sure that the window size will be unable to breach the given value. TCP will continue oscillating as the value increases but will stop when it decreases. However, this is not necessarily a good thing because it is not running at full efficiency
- The value of maximum congestion window that stop oscillating and reaches a stable behaviour is 66. Shown in the plot below
- The average throughput at this point is:
    - 220.81911262798636 packets per second
    - 953938.566553 bps
- If the link capacity is 1Mbps, then our total utilisation is 953938.566553bps / 1Mbps =~ 95%. We are almost running at full capacity

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?

<u>Maximum size of Congestion Window:</u>

- The maximum size for TCP Reno is also 100. This is the same as TCP Tahoe

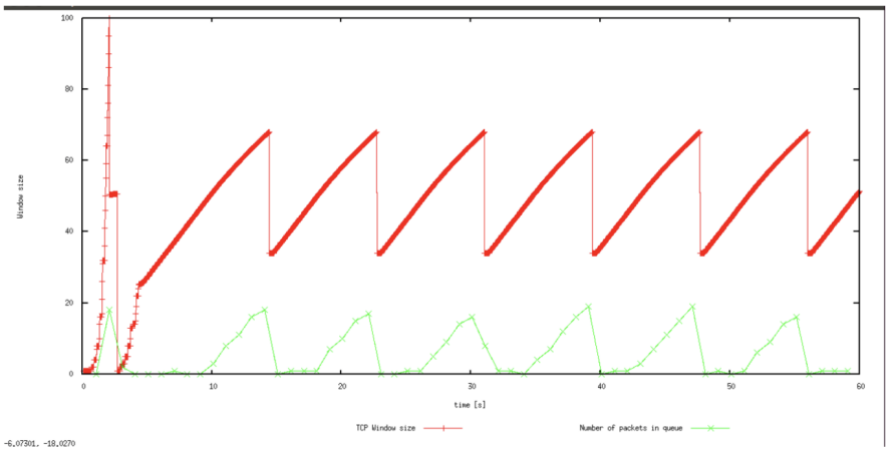
<u>What happens when TCP flow reaches this maximum value:</u>

- When the congestion window reaches 100 packets, it initially drops to half along with the slow start threshold. This indicates the 3 duplicate ACKs have been received. After a short transmission, the window size drops back to 1 and the slow start threshold drops to 25. This indicates a timeout has occurred,

<u>What happens next:</u>

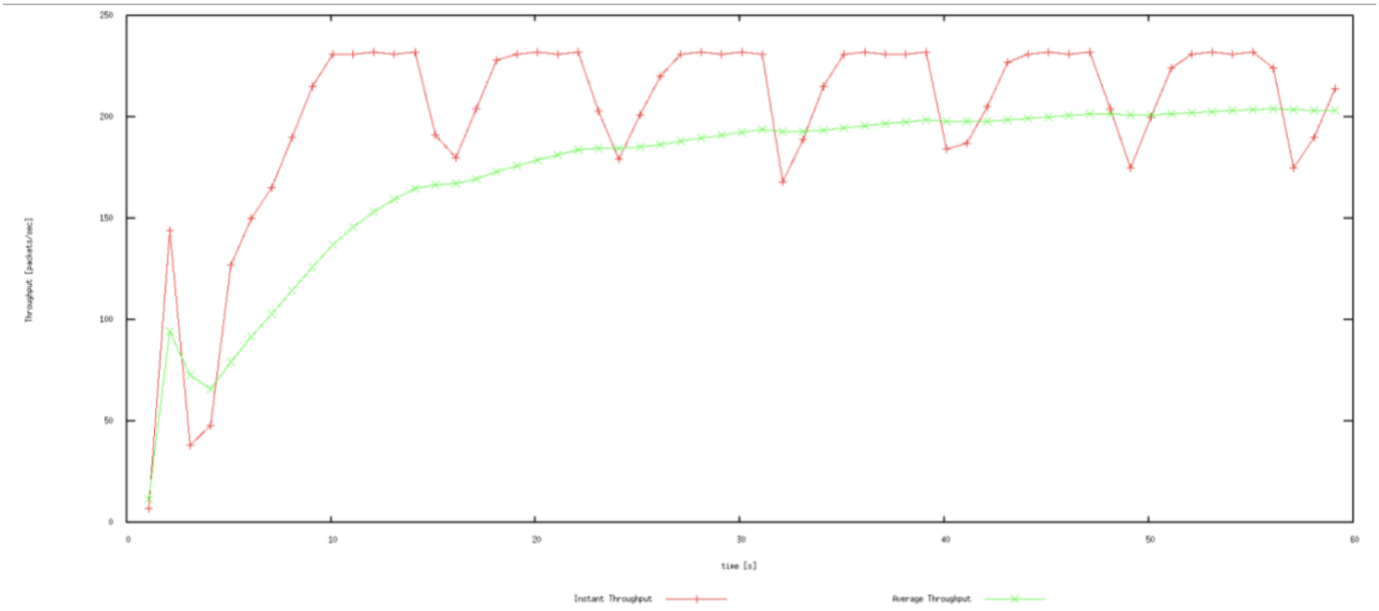
- After initial phase, we enter the congestion avoidance phase. In the case of triple duplicate ACKs, our congestion window halves. The window size will halve again if it is still greater than the slow start threshold.

Below is a plot of TCP Reno window size and packets



Average Throughput for TCP Reno:

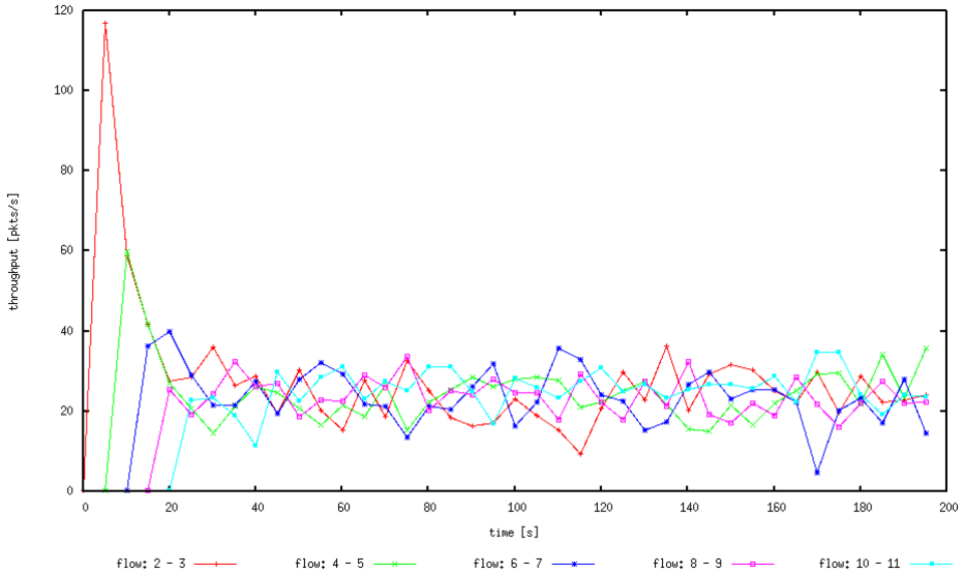
- Throughput is 188.97610921501706 packets per second.
- There are 540 bytes per packet including the header. Therefore:
    - Throughput is also 540 * 8 * 188.97610921501706 = 816376.7918 bps
- This is exactly the same as in TCP Tahoe



Exercise 2

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.

TCP is fair. Each flow gets an equal share of the capacity. The links begin with a different amount of capacity but as time goes on, the all average out to a similar value. This is additive increase and multiplicative decrease. An example of this is shown in the plot below.



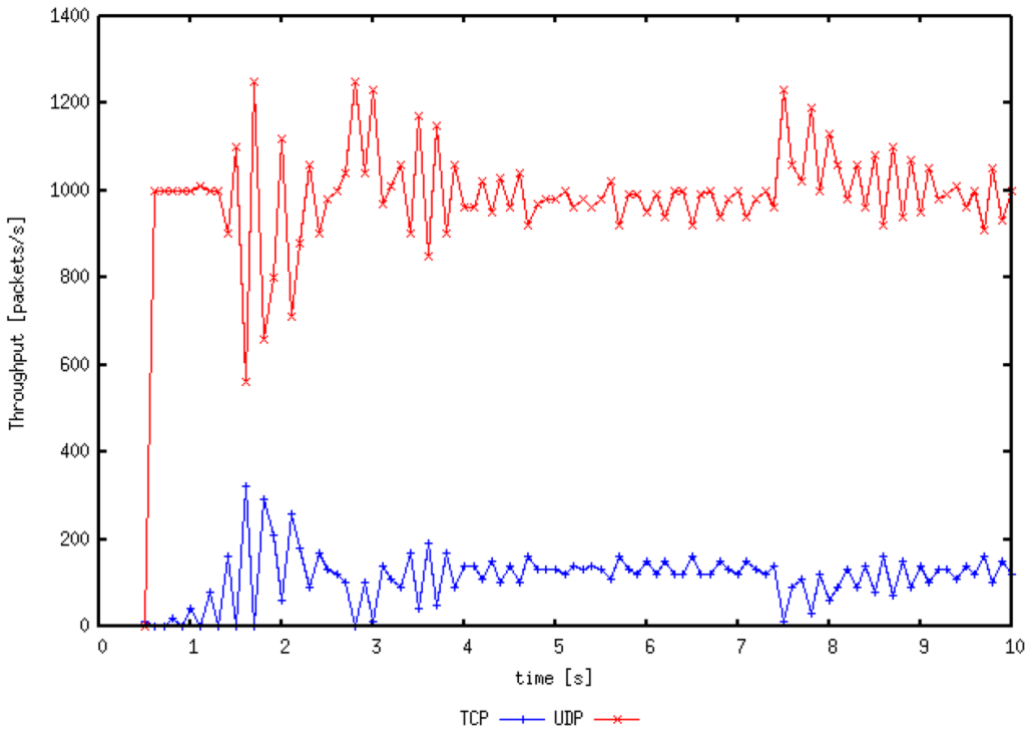
Question 2. What happens to the throughput of the pre-existing TCP flows when a new flow is created? Explain the mechanisms of TCP which contribute to this behaviour. Argue about whether you consider this behaviour to be fair or unfair.

As new flows are created, the throughput of existing flows are decreased so as to give each flow an equal share of capacity. The mechanism that contributes to this is called additive increase, multiplicative decrease (AIMD). I believe that this behaviour is fair as all the flows can receive an equal share of the link.

Exercise 3

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

UDP has a much higher throughput than TCP because it doesn't have congestion control. This can be seen in the plot below



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.

In general terms, the absence of congestion control allows UDP to have a higher throughput than TCP. The congestion control built into TCP automatically changes its window sized based on different conditions. As see in the plot above, sometimes the throughput drops to 0 during a timeout and the window size changed.

Despite the difference in throughput, the lines of UDP and TCP in the plot above are quite similar.

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 of UDP over TCP:

1) UDP has a much higher overall throughput
2) When there are few users and they don't care too much about packet loss, they can get very high data transfer speeds

Disadvantages of UDP over TCP:

1) No form of congestion control means that there is a possibility links get blocked and then collapse
2) There is guarantee that transferred packets aren't lost or corrupted

If everyone started using UDP instead of TCP, it will become a big mess. Networks will most likely collapse and many packets will be dropped due to full queues.