Computer Communications and Networks (COMN)
2019/20, Semester 2

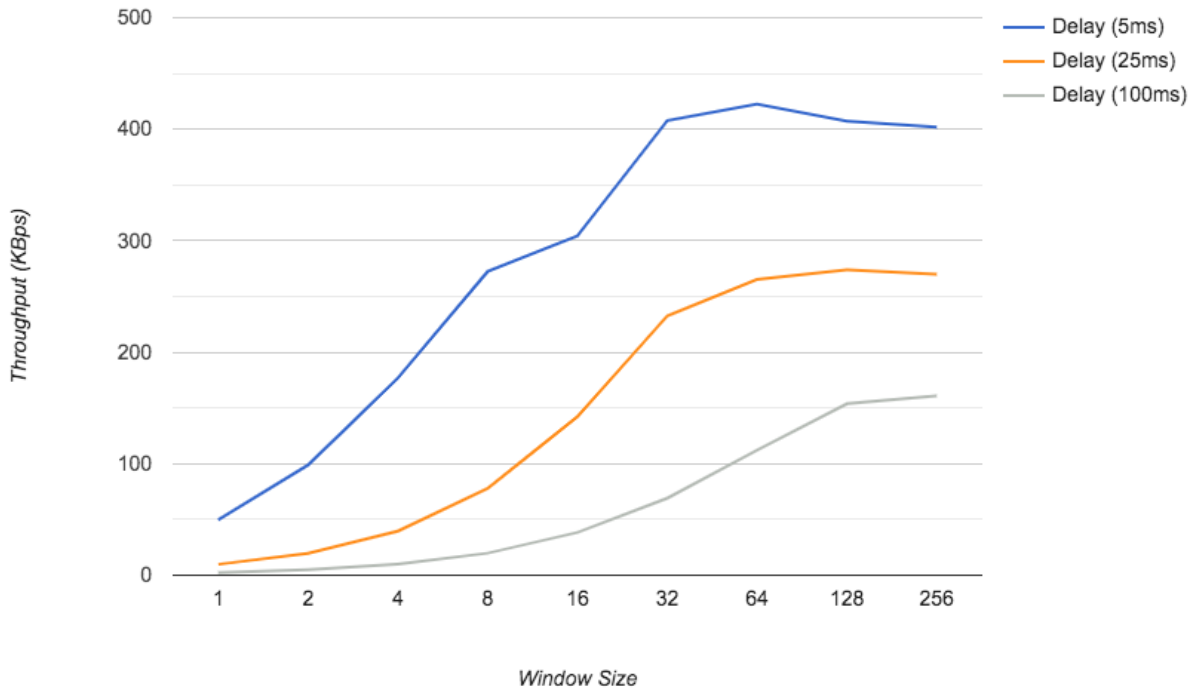
Assignment Part 2 Results Sheet

Forename and Surname:	**BRATIN GHOSH**
Matriculation Number:	**s2032224**

Question 1 – Experimentation with Go-Back-N. For each value of window size, run the experiments for **5 times** and write down **average throughput**.

	Average throughput (Kilobytes per second)		
Window Size	**Delay = 5ms**	**Delay = 25ms**	**Delay = 100ms**
1	49.83	9.81	2.49
2	98.82	19.74	4.98
4	176.90	39.59	9.92
8	272.59	77.89	19.78
16	304.29	142.33	38.41
32	407.78	232.71	69.12
64	422.40	265.40	112.31
128	407.16	273.81	153.84
256	401.99	269.79	160.89

Create a graph as shown below using the results from the above table:



Question 2 – Discuss your results from Question 1.

We observe that the throughput value peaks at window size of 64 with 25ms (from 1b) timeout for 5ms propagation delay. With 0.5% plr, a large window of unAcked packets and timeout greater than the propagation delay all together gives us higher/more throughout. While, having too small window may result in increased number of retransmissions as much fewer unAcked packets are allowed, having a too large window with small propagation delay may negatively affect the throughput as several packets need to be retransmitted in case of a timeout. As expected, on increasing the propagation delay we notice the throughput gets better for larger window sizes (like 128 and 256) as fewer retransmissions occur during timeout since less packets are transferred during that period of time. After experimentation, 20 ms timeout proved to be the best option and the ideal timeout for 25ms and 100ms propagation delay.

Question 3 – Experimentation with Selective Repeat. For each value of window size, run the experiments for **5 times** and write down **average throughput**.

	Average throughput (Kilobytes per second)
Window Size	**Delay = 25ms**
1	10.32
2	19.92
4	44.68
8	85.21
16	149.04
32	238.77

Question 4 - Compare the throughput obtained when using "Selective Repeat" with the corresponding results you got from the "Go Back N" experiment and explain the reasons behind any differences.

We observe that the throughput of Selective Repeat is slightly higher than that of Go Back N. This occurs as, in the Selective Repeat algorithm, the number of retransmitted packets is significantly reduced due to the fact that it only retransmits the unAcked packets instead of sending all the packets in the window as in case of Go Back N. We would expect a major improvement in the throughput but we only get marginal improvements since the propagation delay is very significant meaning the packets are slowly sent and the window of packets to be sent in Go Back N is low. This is why for smaller window sizes Selective Repeat and Go Back N almost have similar throughput values, but as we increase the window size, we start to notice the difference in throughput values.

Question 5 – Experimentation with *iperf*. For each value of window size, run the experiments for **5 times** and write down **average throughput**.

	Average throughput (Kilobytes per second)
Window Size (KB)	**Delay = 25ms**
1	53.01
2	66.87
4	79.16
8	83.40
16	84.33
32	76.32

Question 6 - Compare the throughput obtained when using "Selective Repeat" and "Go Back N" with the corresponding results you got from the *iperf* experiment and explain the reasons behind any differences.

Selective Repeat and Go Back N implementations are based on UDP while *Iperf* is based on TCP. *Iperf*, being connection oriented protocol, performs a handshake to set up the connection before any packets are even sent, unlike UDP based algorithms. We notice that for smaller window sizes, *Iperf* performs better giving us higher throughput than the other UDP algorithms. But at higher window sizes, Go Back N and Selective Repeat start to outperform *Iperf*. The UDP algorithms provide significantly more throughput than *Iperf*. This happens because, for smaller window sizes, *Iperf* makes much better use of the provided bandwidth resulting in lesser retransmissions due to the loss of packet. And for bigger window sizes, UDP can continuously send out packets without caring about 'congestion collapse' unlike TCP, who's congestion control mechanism automatically buffers a lot of the packets for reliable data transfer.