Computer Communications and Networks (COMN) 2016/17, Semester 2

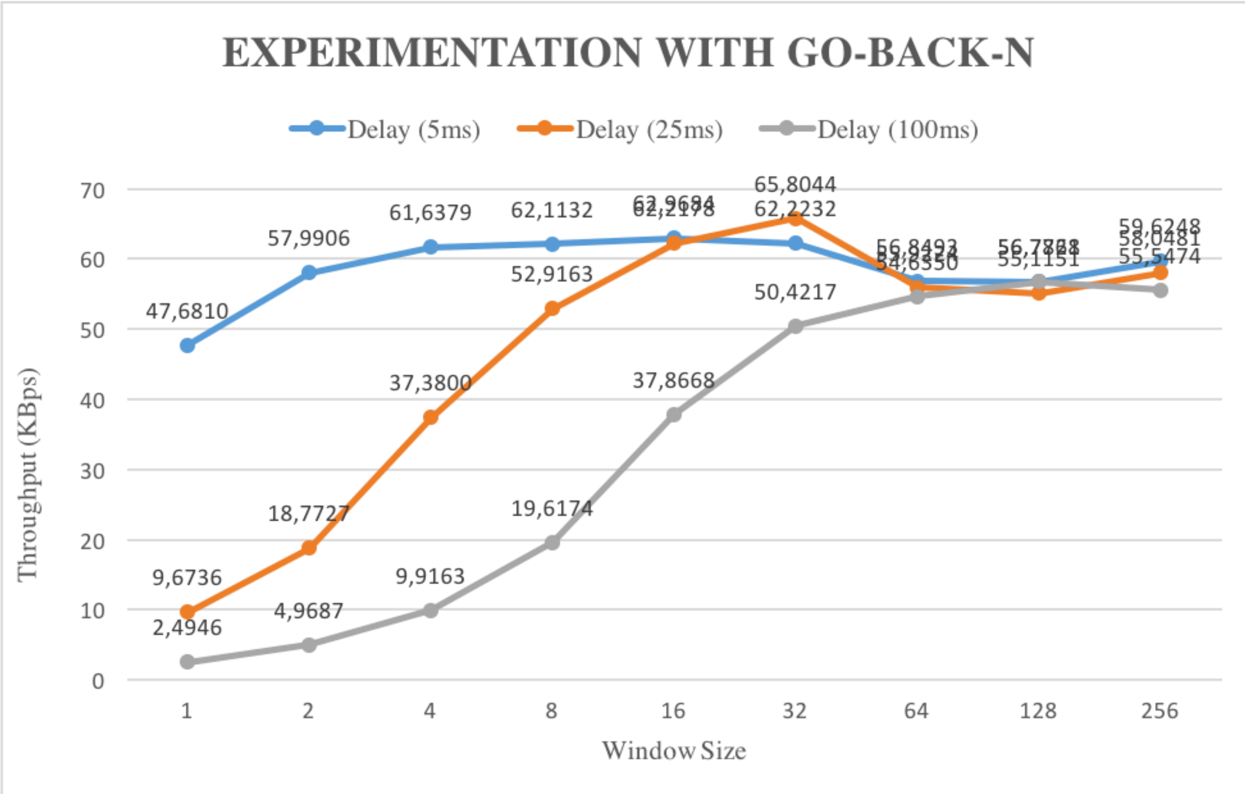
Assignment Part 2 Results Sheet

Forename and Surname:	Lorena Strechie
Matriculation Number:	S1419115

Question 1 – Experimentation with Go-Back-N:

	Throughput (Kilobytes per second)		
Window Size	**Delay = 5ms**	**Delay = 25ms**	**Delay = 100ms**
1	47.6810	9.6736	2,4946
2	57.9906	18.7727	4,9687
4	61.6379	37,3800	9,9163
8	62.1132	52,9163	19,6164
16	62.9684	62,2178	37,8668
32	62.2232	65,8044	50,4217
64	56.8493	55,9224	54,6350
128	56.7868	55,1151	56,7721
256	59.6248	58,0481	55,5474

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



Question 2 – Discuss your results from Question 1.

For this experiment, I have used the optimal retransmission timeout identified from the previous part – 30ms, delay of 5ms, and adjusted it to 120 and 420 for delays of 25ms and 100ms, respectively.

In this experiment, the sender is allowed to transmit multiple packets (when available) without waiting for an acknowledgment, but is constrained to have no more than some maximum allowable number, N, of unacknowledged packets in the pipeline. Therefore, the sender to 'fill the pipeline'. However, if a packet is lost or if a timeout occurs, the sender has to retransmit all N packets within its window which can be very expensive.
Ideally, the sender must not transmit too fast. If the window size is too large, a very large number of packets will be retransmitted when only a single packet is lost. This increase of time brings down the throughput.

The experiment shows the raising returns of increasing the window size. However, the throughput slightly decreases from a window size of 32 which seems to be optimal, as it allows us to fill the pipeline, and yet retransmit significantly fewer packets on loss. On the other hand, window size of 1 is equivalent to Stop-and-Wait ARQ, whilst a window size of 256 and a large delay may mean that many packets are in the pipeline and a single error can cause Go-Back-N ARQ to retransmit a substantial number of packets, many unnecessarily.

Question 3 – Experimentation with Selective Repeat

	Throughput (Kilobytes per second)
Window Size	**Delay = 25ms**
1	9.7276
2	19.8690
4	39.1185
8	74.9099
16	104.1383
32	124.1825

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.

Selective Repeat ARQ is different from Go-Back-N ARQ in the sense that it avoids the problem described above by retransmitting only packets that it believes were lost at the receiver’s end. Thus, it avoids filling the pipeline with unnecessary retransmissions, making a better use of the channel.

The results obtained in the table above show a dramatic increase in the throughput compared to when using the Go-Back-N ARQ, which is also further increased by the choice of the window size (e.g. the greater the window size the larger the throughput). We have to bear in mind that the window size must be less than or equal to half the size of the sequence number space for SR protocols.

Selective Repeat is also bounded by the receiver’s ability to process packets and the efficiency of implementation which can significantly affect the performance of the system. Having hundreds or thousands of independent timers (one per packet) and running the Selective Repeat ARQ on a virtual machine with limited resources and memory is one example of how Selective Repeat might not give the best results at all times.

In conclusion, the mechanisms behind Selective Repeat are indeed more complex than the ones behind Go-Back-N and the table above is nothing but proof that retransmitting only the packets that were received in error contributes to a great difference in the throughput on our channel.

Question 5 – Experimentation with *iperf*

Window Size (KB)	Throughput (Kilobytes per second) Delay = 25ms
1	13.9
2	22.2
4	46.5
8	60.0
16	75.4
32	85.4

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.

The results obtained with iperf in the table above are an average of all the values I obtained after conducting the experiment three times for each window size.

As we have previously seen while conducting the other two experiments, the value of the throughput increases along with the increase in the window size.

The values obtained with iperf are very comparable with the ones obtained using Go Back N, however lower than the ones obtained using Selective Repeat.

Iperf uses TCP to measure the maximum throughput of a network connection.
Since in this experiment, we specified maximum segment size of 1024 bytes which can cause IP fragmentation and lead to excessive retransmissions if there is a packet lost.

Depending on the options, each packet header has a size of a minimum of 20 bytes and a maximum of 60 bytes which is significantly more than the 3 bytes we use for header in Selective Repeat and Go-Back-N. This results in lower throughput. TCP manages retransmissions significantly more efficiently than Go-Back-N, however the general trend suggests that it is slower than Selective Repeat.