Computer Communications and Networks (COMN)
2020/21, Semester 2

Assignment 2 Results Sheet

Forename and Surname:	SIYU WANG
Matriculation Number:	s1703367

Question 1 – Number of retransmissions and throughput with different retransmission timeout values with stop-and-wait protocol. For each value of retransmission timeout, run the experiments for **5 times** and write down **average number of retransmissions** and **average throughput**.

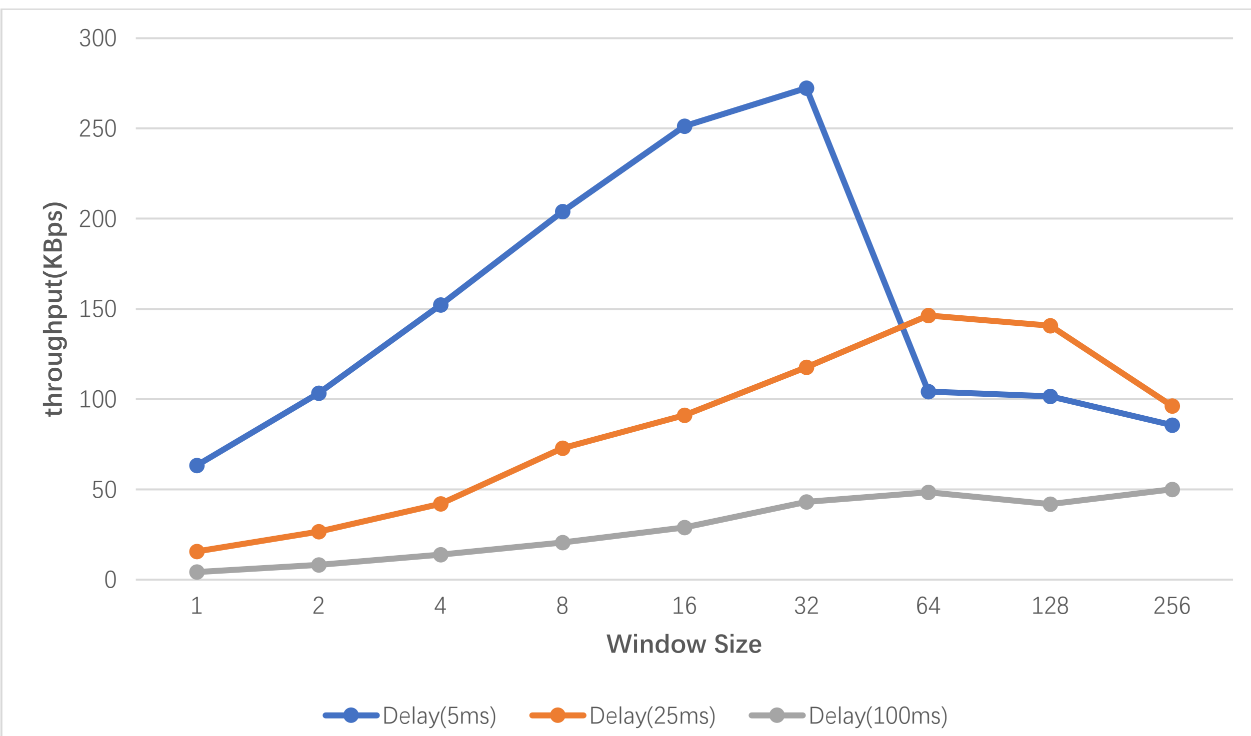
Retransmission timeout (ms)	Average number of re-transmissions	Average throughput (Kilobytes per second)
5	1845.8	73.778
10	975.8	67.566
15	135.6	65.382
20	107.8	62.42
25	108.2	58.378
30	106.6	57.202
40	110	51.186
50	104.6	50.062
75	115	40.142
100	101.2	36.98

Question 2 – Discuss the impact of retransmission timeout value on the number of retransmissions and throughput. Indicate the optimal timeout value from a communication efficiency viewpoint (i.e., the timeout that minimizes the number of retransmissions while ensuring a high throughput).

We can see that when the retransmission timeout values are 5 and 10, we have a high number of retransmissions. This is because the average RTT is 10ms and this means receiver should expect to receive the ACK 10ms after the package is sent. If the timeout value is too small, sender will resend a packet which is already received by the receiver too many times. When timeout is 15ms the retransmissions is also relatively high since it is still close to 10ms RTT. As the timeout values goes larger, the average throughput becomes smaller. This is because for some packet, they will lose during the transmission so no mater how long sender waits, there will no ACK return from receiver. So long timeout is just wasting of time and will reduce the average throughput. In fact, the (retransmissions / total packets) value at 100ms timeout is close to 0.1 which is the packet loss rate of the TC. The optimal timeout value is **20ms** as it has a relatively low retransmissions and high throughput.

Question 3 – 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	63.308	15.638	4.218
2	103.418	26.692	8.282
4	152.242	41.972	13.926
8	203.98	72.812	20.632
16	251.36	91.146	28.872
32	272.384	117.722	43.078
64	104.172	146.364	48.458
128	101.562	140.646	41.852
256	85.54	96.268	50.064



Question 4 – Discuss your results from Question 3.

The time out value at Delay(25ms) is set to 70ms and for Delay(100ms) it is set to 220ms.

Firstly the maximum value of throughput will decrease due to the combination of the delay and the retransmission timeout. The window size can also affect throughput significantly. For Delay(5ms), at first the throughput increase significantly when window size increases. At window size = 32 it has the max throughput of 272KBps. After that the throughput suffers a significant dive and remains at a relatively low value. This is because GBN protocol is forced to retransmit all the packets in the window when the oldest packet in the window is timed out. With a large window size, the chance of packet loss during a transmission is higher and the number of packets need to retransmission is large. More over, receiver may be flooded with the arrival of large amount of data at once so it cannot process and send ACKs on time, causing a timeout at sender. When the delay is longer, the timeout value is set longer and the optimal value of window size is also larger. But the throughput is also lower because of the delay and longer timeout. So when the network delay is long, the window size can be set larger to increase the throughput.
Another observation is that when the window size is large, the value of throughputs of each run is not stable especially when the delay is 5ms. This could also because of GBN protocol is forced to retransmit all the packets in the window and large window size will suffer frequent retransmit.

Question 5 – 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	16.41
2	29.714
4	56.634
8	94.376
16	164.086
32	284.476

Question 6 - 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.

The time out is set to 70ms.
On small window sizes (1,2 and 4), the throughputs have no big difference to GBN which is expected since when window is small, we still don’t need much time to retransmit the whole window. But as window size grows larger, the throughput of selective repeat increases significantly and much larger than the GBN when window size is 32. This is because Selective repeat does not need to retransmit the whole window when

packet timeouts. It only need to retransmit the packets that timed out. So overall selective repeat is more efficient than GBN when widow size is large.
But for Selective repeat, the number of timer needed is equal to the window size. If the window size is too large, there will be many times running at the same time and if sender's machine cannot handle such more session, the efficiency may be affected.

Question 7 – 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	10.6575
2	23.125
4	29.575
8	62.3
16	81.225
32	90.375

Question 8 - 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.

We can see that the overall throughput of iperf is smaller than the GBN and Selective repeat. The average throughput increases as the window size increases. This is as expected because iperf uses TCP to transmitting files and TCP sacrifices speed for reliability. TCP transfers data as a stream, does error checking and handshakes so TCP will take more time to process a packet. The size of the header of a TCP segment(20Bytes) is slightly bigger than the header of the UDP packet(3Bytes). This means we have less message in each TCP segment than the UDP packet and may slow down the transmission speed. Also, TCP has flow control to slow down the transmission and make sure that the sender is not overwhelming the receiver. UDP just push as many packet as it can down the connection and handle the data in the application layer so usually has a higher throughput.