Computer Communications and Networks (COMN)
2022/23, Semester 1

Assignment 2 Results Sheet

Forename and Surname:	Yubo Shao
Matriculation Number:	S2084333

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 the **average number of retransmissions** and the **average throughput**.

Retransmission timeout (ms)	Average number of retransmissions	Average throughput (Kilobytes per second)
5	1153	60.31
10	657	58.15
15	155	56.20
20	120	54.57
25	114	52.08
30	111	48.56
40	109	48.32
50	108	45.82
75	108	41.08
100	108	36.63

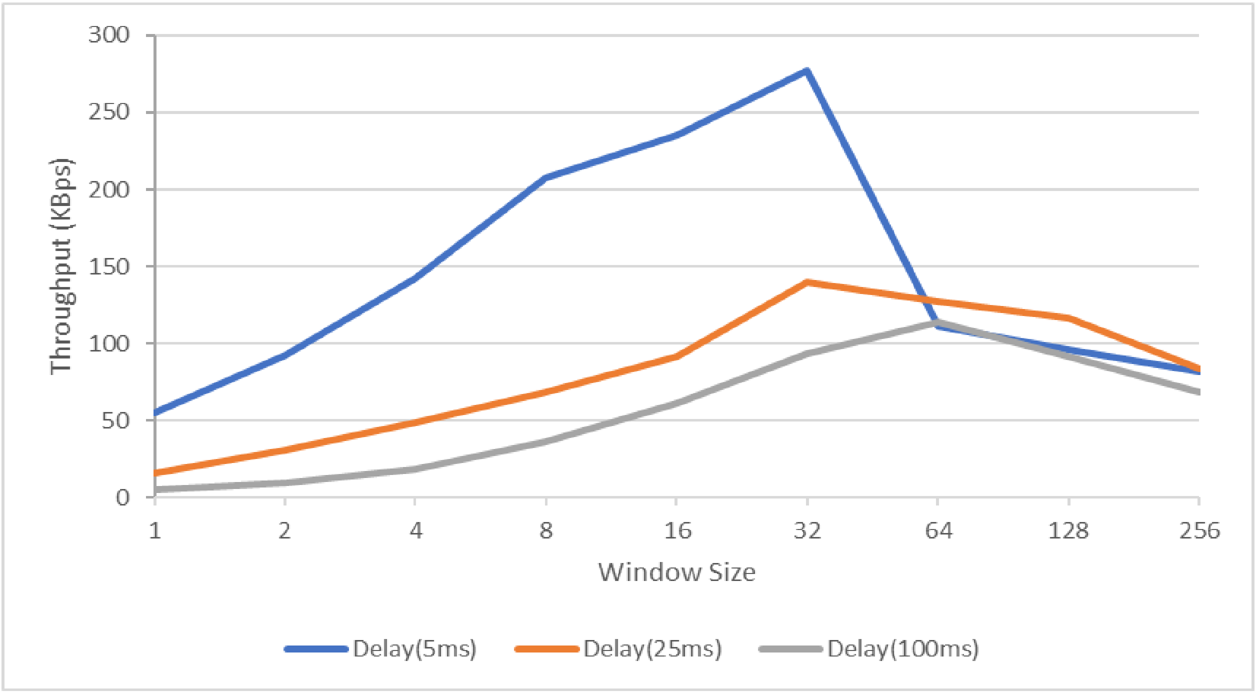
Question 2 – Discuss the impact of retransmission timeout value on the number of retransmissions and throughput. Indicate the optimal timeout value from a communication efficiency viewpoint (i.e., the timeout that minimizes the number of retransmissions while ensuring a high throughput).

As retransmission timeout increases, the number of retransmissions decreases exponentially and the throughput decreases linearly. This makes sense as the retransmission timeout increases, there is a bigger time gap for the ACK message to arrive at the Sender before timeout and retransmission of the packet, however, the average transmission delay of the ACK message stays about the same, then there would be less impact on the number of retransmission as retransmission timeout increases, hence the exponential decrease. For the throughput, due to a longer timeout, the time taken to fully transfer the file linearly increases, hence a linearly decreasing throughput. The optimal timeout value is determined by throughput/retransmissions for each timeout value, and 25ms is the highest hence the optimal.

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

	Average throughput (Kilobytes per second)		
Window Size	**Delay = 5ms**	**Delay = 25ms**	**Delay = 100ms**
1	54.76	15.54	4.73
2	92.52	30.75	9.45
4	142.1	48.92	18.36
8	207.8	67.93	36.37
16	234.9	91.51	61.53
32	277.5	139.6	93.42
64	110.8	127.5	113.9
128	95.68	116.4	91.27
256	81.84	83.11	68.25

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



Question 4 – Discuss your results from Question 3.

25ms retransmission timeout for 5ms one-way delay is used, which is 15ms longer than the RTT (2 * 5ms), for 25ms one-way delay, 65ms retransmission timeout is selected (2 * 25ms + 15ms). For 100ms, 65ms is also used because 215ms (2 * 100ms + 15ms) would result in excruciating low throughput, rendering it impractical. From the experiment data, the window sizes increase causes the initial increase in throughput as more packets can be sent without waiting for their ACK messages, and the throughput decline after window size 32 (or 64 for 100ms one-way delay) shows that the massive time loss caused by the timed-out packet resends for the entire window outweighs the time gained from being able to send more packets without waiting for ACK thanks to a larger window. The higher throughput at 5ms delay is because of both low RTT as well as a lower likelihood that a timeout resend would happen as the RTT is below the retransmission timeout. This can also be seen between 25ms and 100ms, as the RTT for 25ms one-way delay (50ms) is below the retransmission timeout (65ms), it performed marginally better than 100ms one-way delay as it would timeout 3 times before the first ACK message arrives. However, if a lower retransmission timeout is selected for higher one-way delays, the throughput doesn't necessarily decrease due to time-out resends. It increases most of the time because of the lower RTT and causes the time taken to be much shorter and resulting in much higher throughput. It is another trade-off between the number of retransmissions and the throughput.

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

	Average throughput (Kilobytes per second)
Window Size	**Delay = 25ms**
1	16.57
2	32.57
4	60.63
8	109.2
16	182.5
32	335.6

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.

For Selective Repeat (SR), since there is a buffer for both sender and receiver, it significantly increases the throughput by only requiring a time-out resend for the unacknowledged packets, hence drastically reducing the number of retransmissions and ensuring a more consistent "sliding" window by buffering incoming packets within the window. (Won't require full retransmission for the whole window then wait and wasting time.) The experiment results at 65ms retransmission timeout for SR demonstrated these advantages clearly with higher throughput across the whole range of window sizes. The throughput improvement isn't very obvious with low receiver window sizes from 1 to 2 as there isn't much increase as SR with both window sizes at 1 is essentially the same as Go Back N ARQ (GBN) with sender window size at 1, and SR with both window sizes at 2 only provided one additional buffer at the receiver's side compared to GBN with window size 2. However, the throughput quickly scales and surpasses GBN as the window size exponentially increases. The throughput for SR only starts declining after window size 64, indicating the massive improvement receiver's packet buffer provides.

Question 7 – Experimentation with *iperf*. For each value of window size, run the experiments for **5 times** and write down the **average throughput**.

	Average throughput (Kilobytes per second)
Window Size (KB)	**Delay = 25ms**
1	12.8
2	25.7
4	31.1
8	64.7
16	85.2
32	101

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.

One would expect TCP to have similar throughput compared to Selective Repeat (SR) and Go Back N (GBN), as conceptually TCP is a hybrid of both.
It is similar to GBN because both protocols have a maximum size for the number of unacknowledged packets the sender can send, but GBN retransmits every unacknowledged packet when timed out while TCP only retransmits the oldest unacknowledged one.
TCP is similar to SR because both protocols don't require retransmitting every unacknowledged packet and only retransmit the ones that are lost, but SR uses individual ACK and requires the ACK for each packet while TCP uses cumulative ACK and only requires the highest consecutive ACK to show every packet before this ACK is received.
However, as demonstrated by the data, TCP has a marginally lower throughput compared to GBN and SR across the different window sizes. This is because of the additional work TCP has done besides just packet transmission, like error detection (checksum) and congestion control (The flow control in this case is pre-specified as window sizes are set before the transmission happens). The processes mentioned above create lots of overhead and hence result in slower throughput when using TCP.
These additional processes also cause the transmitted packets more likely to time out when there is a larger window, this delay increases as window size increases and reduce the improvement brought by being able to send more packets without ACK. As shown by the data, although the average throughput increases as window size increases, the rate of throughput incline decreases accordingly. This can also be seen with the initially similar throughput performance between all 3 protocols and the drastic difference in the rate of throughput incline as the window size increases.