I have read and understood the course academic integrity policy.

Bharadwaj Parthasarathy
50246591

Timeout Scheme:

ABT - 10
GBN - 30
SR - 30

The timer for ABT, GBN and SR was chosen based on experiments analyzing the throughput and time taken.

ABT: considering timer values of **10**, 20 and 30, timer value of 10 produced better throughput when analyzed with loss probabilities of 0.1, 0.2 and 0.6, the time taken being almost the same.

GBN and SR: Timer value of 15 produced less throughput when compared to timer value of 25 and 30. While 25's and 30's timer value had almost the same throughput, but when corruption values increased (say 0.6 and 0.8) the time taken for timer value 25 was more than that of **30**.

Multiple Software timers :

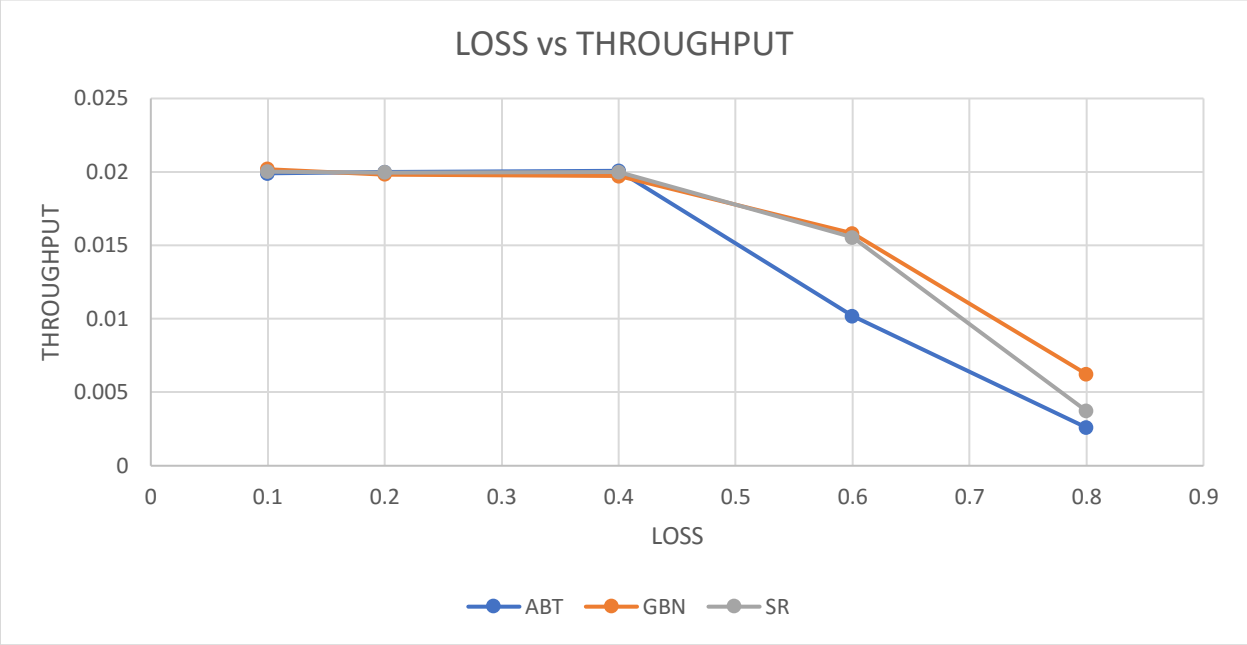
Initial Implementation: Initial implementation was to start the timer for the base packet, and store the timeout value (current time + timer value) and store it in an array. Each time timer interrupt occurs the array is checked for expired packets and then the packets are resent. This implementation passed all test cases, but the packets are not sent exactly at the moment its timeout happens but later depending on timer value. Hence this is not the correct implementation.

The next thought was to have timer value as 1, thereby causing the timer interrupt to happen every single time unit. This would be the right implementation but will be very inefficient and the simulation may happen for a long time.

Final Implementation: The idea was to maintain the timer values in the array relative to each other. When a packet is sent the value of (current time + timer value) is stored in the array. Each time a timer interrupt occurs, the timer value in the array which got expired is incremented by original timer value units and the next timer value to start the timer is calculated. The next timer value is calculated by finding the minimum value in the array and subtracting the current time from minimum value. This way we can ensure that each packet is retransmitted at exactly that moment its timer expires.

Experiment 1:

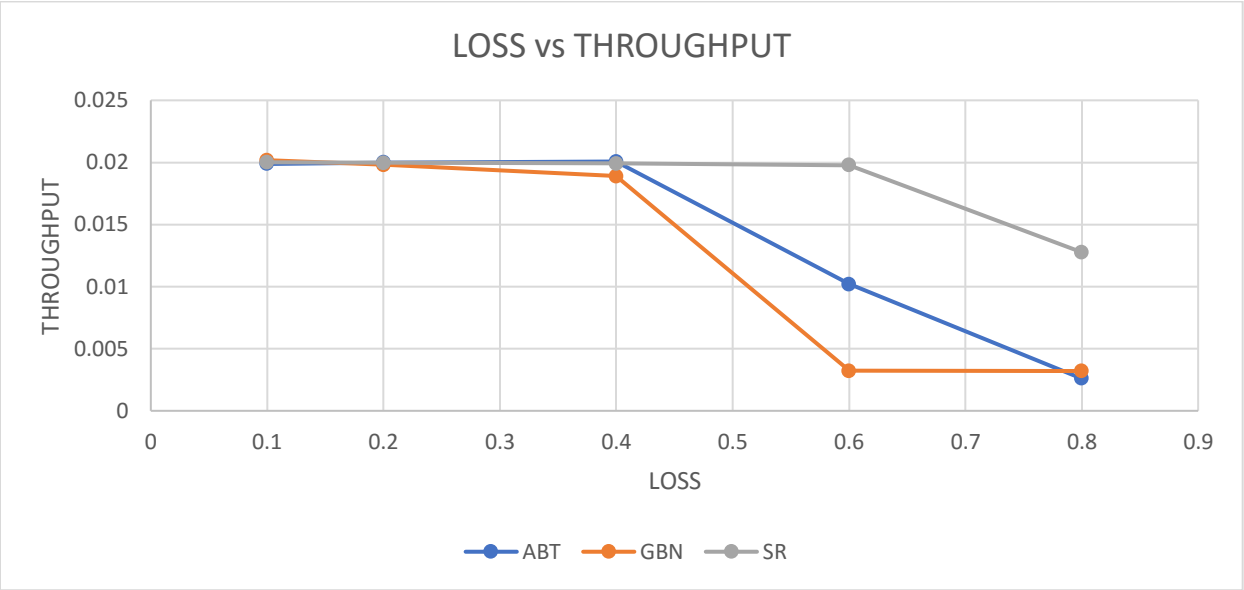
Window size: 10



Observation:

As loss increases throughput must decrease, as more and more packets will be lost and retransmission happens. This pattern is followed properly in the graph for all the three protocols. When throughputs are compared between the protocols, ideally SR > GBN > ABT. But we observe that SR < GBN. This may be due to the small window size as results become better in the next case.

Window size: 50



Observation: Here GBN's performance is less than ABT's. This may be due to the fact that the timer implemented is not an adaptive timer. Hence when the window size is more, retransmissions happen and hence may have affected GBN's throughput. Maybe better implementation of the timer would have given the expected result.

Experiment 2:

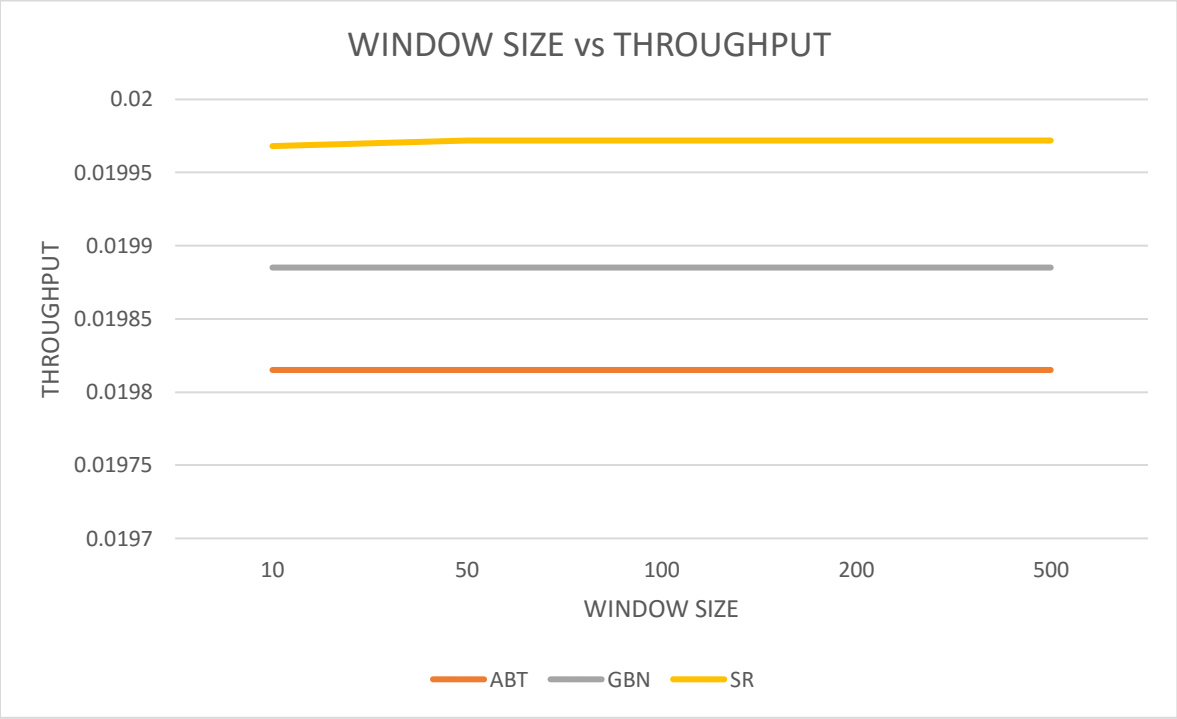
Observation:

Here we are comparing the throughput values with window size with different loss. ABT does not depend on window size and hence should remain constant **(in graph a and graph b)** for different window sizes given a loss value. The behavior of GBN and SR can been seen in the graphs below. Since retransmission depends on window size, the graph for GBN must reduce as the window size increases, because we retransmit all packets until next sequence at base packet timeout.

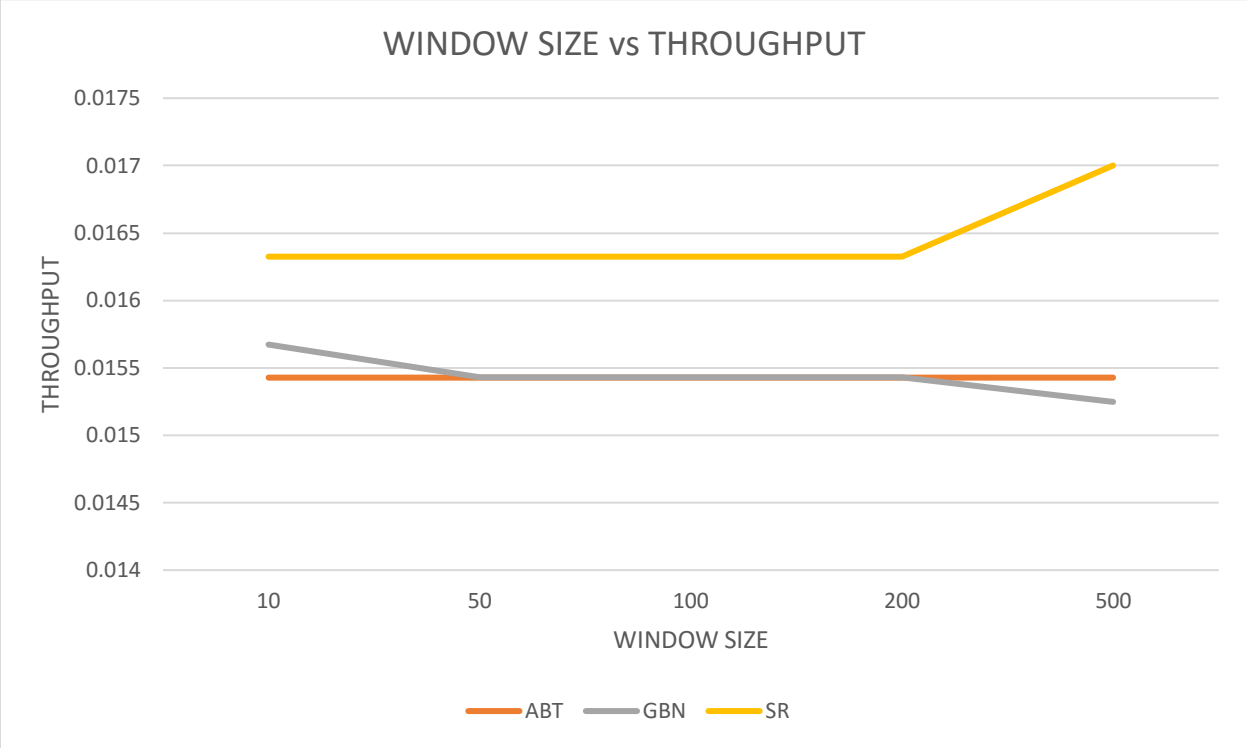
1. This means there will be lot of retransmission as window size is more resulting in throughput decrease.
2. Whereas in SR the retransmission happens only for the timeout packet and hence there is a chance for the throughput to increase momentarily as loss increases.

Points 1 and 2 can be observed clearly in graph b whereas in graph a throughput is not clearly increasing or decreasing it remains almost constant.

a. **Loss - 0.2**



b. Loss: 0.5


WINDOW SIZE vs THROUGHPUT
0.0175
0.017
0.0165
0.016
0.0155
0.015
0.0145
0.014
THROUGHPUT
10
50
100
200
500
WINDOW SIZE
ABT
GBN
SR