Introduction

In this assignment, I implemented 2 reliable data transfer protocols called Selective Repeat (SR) and Go-Back-N with Selective Ack (GBN+SACK). The goal was to correctly handle various states of error such as packet loss and corruption while handling the transfer of data from a sender (A) to a receiver (B).

Protocol Design

I implemented SR using a configurable sliding window in which the receiver buffers the out of order packets and then sends an ACK for each individual packet. Thus, by storing the out of order packets, the receiver delivers them in sequential order to layer 5 after any of the missing packets arrive.

I implemented GBN+SACK with the sender maintaining a sliding window and transmitting packets within the window. I included SACK to help the sender avoid retransmitting any packets the receiver already has and arrived out of order. And I also incorporated cumulative ACKS for the sliding window which is standard in GBN without SACK, to make sure that recovery is available in cases of packet loss or corruption.

Results and Analysis

I conducted an experiment with SR using retransmission timeout values of 20, 30, 50, 100, 200, and 500, and from my observations, I noticed a tradeoff between retransmission frequency and communication efficiency. The lower timeout values of 20 and 30 had more frequent retransmissions, which resulted in a higher throughput. While a higher throughput isn't necessarily bad, it could potentially result in network congestion and wasted bandwidth. As I predicted, with the slightly larger timeout values of 50 and 100, retransmissions decreased while still allowing for a reasonably timed recovery from loss. However, when testing the higher timeout values of 200 and 500, the number of retransmissions decreased by a lot. From my observations, the tradeoff with this is that there is a higher average communication time and RTT. This is because the sender is waiting much longer before retransmitting lost packets which consequently lead to a delayed recovery and a slower throughput.

Next I conducted the same experiment with GBN+SACK. At lower timeout values like 20 and 30, I noticed that the sender detected the losses quickly which led to a faster recovery and lower average communication times of 15.27 ms and 17.74ms respectively. However it did have more frequent retransmissions. Overall, this approach kept communication times efficient but also probably increased the network overhead. Then with timeout values of 50 and 100, I noticed that retransmission remained similar to each other, but communication time increased by a lot. At 100, the communication time was 36.87 which could imply that as a result of the larger delay, the sender's ability to recover from loss suffers. And like SR, at high timeouts like 200 and 500, the retransmissoin dropped by a lot, but the sender most likely waited too long before detecting the lost packets which resulted in much higher communication times. And from the results, I observed that the trade off in GBN+SACK is that at the cost of more network congestion there are smaller timeouts.

Figure 1: Selective Repeat Metrics with Confidence Intervals

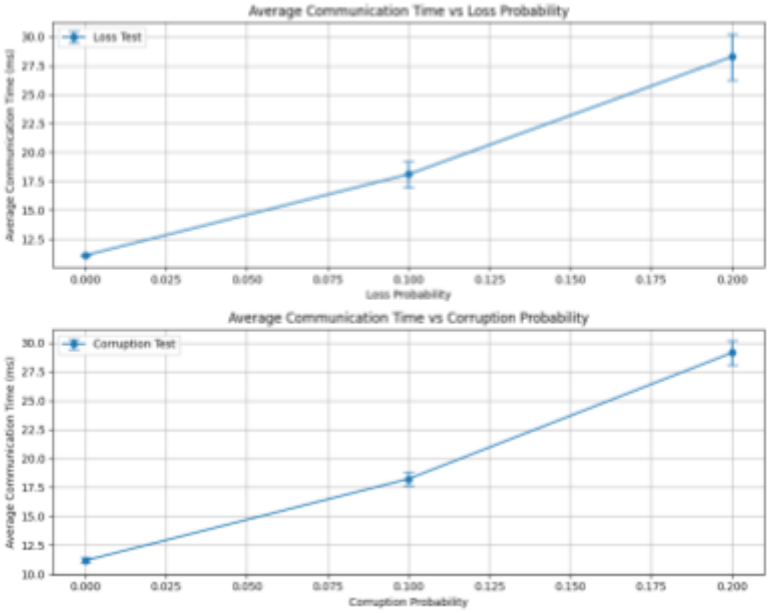
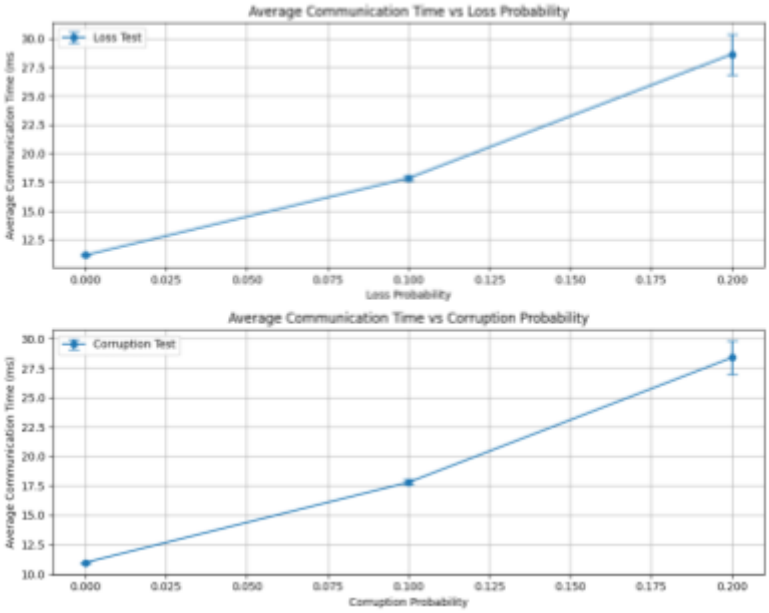


Figure 2: Go-Back-N Metrics with Confidence Intervals



For the above figures, I evaluated the performance in terms of loss and corruption with SR and GBN+SACK. Each data point in the figures were generated by performing 3 tests, each with a unique seed to bring it variability in network conditions and variables. I felt like this approach guaranteed that the results weren't outliers. And after collecting the average communication time for each configuration, I computed 90% confidence intervals. The intervals can be seen as the error bars above and under the data points.

When measuring throughput (TPUT), goodput (GPUT) and average packet delay, I performed the experiment under the loss probability of 0.1 and corruption probability of 0.1. Using this setup I observed that SR and GBN+SACK yield similar TPUT of 0.013 packet/ms and GPUT of 0.005 packets/ms. However, GBN+SACK resulted in a slightly higher TPUT with 0.0137 packets/ms in one test while SR resulted in 0.0136. The GPUT was about the same for both protocols which shows that the final number of packets that were successfully deliver are the same.

Also both protocols experienced a similar number of retransmissions, ratio of lost packets and ratio of corruption. I think the results remain consistent because each protocol eventually recovers and delivers all 1000 packets. The average RTT is lower for GBN+SACK is also lower for GBN+SACK which came at about 11.2 ms, as compared to SR's 12.4ms. I think this could be due to the differences in how the two protocols handle their acknowledgements and the differences in how their windows behave. Similarly, I saw that both protocols had similar APD for recovered and successful deliveries which came to about 25.9 ms for GBN+SACK and 27.0 ms for SR.

Nonetheless, despite some differences in the metrics, SR and GBN+SACK reliably delivered all the packets within a similar range of each other.

Formulas Used

```
	// Throughput
	double totalTime = endTime - startTime;
	int totalPacketsSent = totalOriginalPackets + totalRetransmissions +
totalACKsSent;
	double throughput = totalPacketsSent / totalTime;
	System.out.println("Throughput: " + throughput + " p/ms");

	// Goodput
	double goodput = totalDeliveredPackets / totalTime;
	System.out.println("Goodput: " + goodput + " p/ms");

	// Average Packet Delay
	double averagePacketDelay = avgCommunicationTime;
	System.out.println("Average Packet Delay: " + averagePacketDelay + " ms");
```

```
	# Confidence Intervals
	def compute_confidenceInterval(data, confidence=0.90):
		n = len(data)
		mean = np.mean(data)
		s = np.std(data, ddof=1)
		t_value = np.sqrt(n) * s / np.sqrt((n-1) * (1 - confidence))
		h = t_value / np.sqrt(n)
		return mean, h
```

Trace Cases

Selective Repeat

C1: No Loss + No Corruption

```
EVENT time: 43.4339426862392  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 43.4339426862392
generateNextArrival(): future time for event 1 at entity 0 will be 67.36045411542099
toLayer3: seqnum: 0  acknum: -1  checksum: 65535  payload: aaaaaaaaaaaaaaaaaaaa  retransmission: false  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 43.4339426862392
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 43.4339426862392

EVENT time: 45.66060365611743  type: 2  entity: 1
bInput() called
bInput(): expecting packet 0 getting packet 0
toLayer3: seqnum: -1  acknum: 0  checksum: 65535  payload:   retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 47.569475386822106  type: 2  entity: 0
aInput() called
stopTimer: stopping timer at 47.569475386822106
```

C2: ACK is Lost/Corrupted, Later ACK slides window by >1

```
EVENT time: 4808.081301309244  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 4808.081301309244
generateNextArrival(): future time for event 1 at entity 0 will be 4868.385876033814
toLayer3: seqnum: 12  acknum: -1  checksum: 65523  payload: ccccccccccccccccccc  retransmission: false  sack: null
toLayer3: packet being lost
stopTimer: stopping timer at 4808.081301309244
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 4808.081301309244

EVENT time: 4838.081301309244  type: 0  entity: 0
aTimerInterrupt() called
toLayer3: seqnum: 12  acknum: -1  checksum: 65523  payload: ccccccccccccccccccc  retransmission: true  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 4838.081301309244
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 4838.081301309244

EVENT time: 4846.016357921078  type: 2  entity: 1
bInput() called
bInput(): expecting packet 12 getting packet 12
toLayer3: seqnum: -1  acknum: 12  checksum: 65523  payload:   retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: packet being lost

EVENT time: 4868.081301309244  type: 0  entity: 0
aTimerInterrupt() called
toLayer3: seqnum: 12  acknum: -1  checksum: 65523  payload: ccccccccccccccccccc  retransmission: true  sack: null
toLayer3: packet being corrupted
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 4868.081301309244
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 4868.081301309244

EVENT time: 4868.385876033814  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 4868.385876033814
generateNextArrival(): future time for event 1 at entity 0 will be 4922.171697134312
toLayer3: seqnum: 13  acknum: -1  checksum: 65522  payload: ddddddddddddddddddd  retransmission: false  sack: null
toLayer3: packet being corrupted
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 4868.385876033814
startTimer: starting timer at 4868.385876033814

EVENT time: 4871.758476516006  type: 2  entity: 1
bInput() called

EVENT time: 4873.55088961455  type: 2  entity: 1
bInput() called

EVENT time: 4898.385876033814  type: 0  entity: 0
aTimerInterrupt() called
toLayer3: seqnum: 12  acknum: -1  checksum: 65523  payload: ccccccccccccccccccc  retransmission: true  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 4898.385876033814
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 4898.385876033814

EVENT time: 4907.840113528855  type: 2  entity: 1
bInput() called
bInput(): expecting packet 13 getting packet 12
toLayer3: seqnum: -1  acknum: 12  checksum: 65523  payload:   retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 4909.747337260048  type: 2  entity: 0
aInput() called
stopTimer: stopping timer at 4909.747337260048
startTimer: starting timer at 4909.747337260048
```

C3: Data Packet is Lost/Corrupted, and Retransmitted After Timeout

```
EVENT time: 5341.752372814112  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 5341.752372814112
generateNextArrival(): future time for event 1 at entity 0 will be 5631.238746299631
toLayer3: seqnum: 0  acknum: -1  checksum: 65535  payload: gggggggggggggggggg  retransmission: false  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 5341.752372814112
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 5341.752372814112

EVENT time: 5343.521439252908  type: 2  entity: 1
bInput() called
bInput(): expecting packet 0 getting packet 0
toLayer3: seqnum: -1  acknum: 0  checksum: 65535  payload:   retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: packet being lost

EVENT time: 5371.752372814112  type: 0  entity: 0
aTimerInterrupt() called
toLayer3: seqnum: 0  acknum: -1  checksum: 65535  payload: gggggggggggggggggg  retransmission: true  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 5371.752372814112
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 5371.752372814112

EVENT time: 5375.81600390637  type: 2  entity: 1
bInput() called
bInput(): expecting packet 1 getting packet 0
toLayer3: seqnum: -1  acknum: 0  checksum: 65535  payload:   retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 5385.146641438033  type: 2  entity: 0
aInput() called
stopTimer: stopping timer at 5385.146641438033
```

C4: Data Packet is Lost/Corrupted, and Retransmitted After Receiving Duplicate ACK

```
EVENT time: 30701.095614490183  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 30701.095614490183
generateNextArrival(): future time for event 1 at entity 0 will be 30712.082277901543
toLayer3: seqnum: 6  acknum: -1  checksum: 65529  payload: uuuuuuuuuuuuuuuuuu  retransmission: false  sack: null
toLayer3: packet being lost
stopTimer: stopping timer at 30701.095614490183
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 30701.095614490183

EVENT time: 30712.082277901543  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 30712.082277901543
generateNextArrival(): future time for event 1 at entity 0 will be 30718.72310556352
toLayer3: seqnum: 7  acknum: -1  checksum: 65528  payload: vvvvvvvvvvvvvvvvvv  retransmission: false  sack: null
toLayer3: packet being lost
stopTimer: stopping timer at 30712.082277901543
startTimer: starting timer at 30712.082277901543

EVENT time: 30718.72310556352  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 30718.72310556352
generateNextArrival(): future time for event 1 at entity 0 will be 31038.78902554399
toLayer3: seqnum: 8  acknum: -1  checksum: 65527  payload: wwwwwwwwwwwwwwwwww  retransmission: false  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 30718.72310556352
startTimer: starting timer at 30718.72310556352

EVENT time: 30728.488540299688  type: 2  entity: 1
bInput() called
bInput(): expecting packet 6 getting packet 8
toLayer3: seqnum: -1  acknum: 5  checksum: 65530  payload:   retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 30731.151646473583  type: 2  entity: 0
aInput() called
toLayer3: seqnum: 6  acknum: -1  checksum: 65529  payload: uuuuuuuuuuuuuuuuuu  retransmission: true  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 30731.151646473583
startTimer: starting timer at 30731.151646473583

EVENT time: 30737.93804742576  type: 2  entity: 1
bInput() called
bInput(): expecting packet 6 getting packet 6
toLayer3: seqnum: -1  acknum: 6  checksum: 65529  payload:   retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 30742.636158330024  type: 2  entity: 0
aInput() called
stopTimer: stopping timer at 30742.636158330024
startTimer: starting timer at 30742.636158330024

EVENT time: 30772.636158330024  type: 0  entity: 0
aTimerInterrupt() called
toLayer3: seqnum: 7  acknum: -1  checksum: 65528  payload: vvvvvvvvvvvvvvvvvv  retransmission: true  sack: null
toLayer3: packet being corrupted
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 30772.636158330024
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 30772.636158330024
```

C5: Retransmitted Data is Delivered, and Cumulative ACK Moves Window by >1

```
EVENT time: 130195.9654530473  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 130195.9654530473
generateNextArrival(): future time for event 1 at entity 0 will be 130197.40592669457
toLayer3: seqnum: 2  acknum: -1  checksum: 65533  payload: yyyyyyyyyyyyyyyyyy  retransmission: false  sack: null
toLayer3: packet being lost
stopTimer: stopping timer at 130195.9654530473
startTimer: starting timer at 130195.9654530473

EVENT time: 130197.40592669457  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 130197.40592669457
generateNextArrival(): future time for event 1 at entity 0 will be 130545.82964568013
toLayer3: seqnum: 3  acknum: -1  checksum: 65532  payload: zzzzzzzzzzzzzzzzzz  retransmission: false  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 130197.40592669457
startTimer: starting timer at 130197.40592669457

EVENT time: 130197.95113986636  type: 2  entity: 1
bInput() called
bInput(): expecting packet 1 getting packet 1
toLayer3: seqnum: -1  acknum: 1  checksum: 65534  payload:    retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 130204.9473860897  type: 2  entity: 0
aInput() called
stopTimer: stopping timer at 130204.9473860897
startTimer: starting timer at 130204.9473860897

EVENT time: 130207.06231879638  type: 2  entity: 1
bInput() called
bInput(): expecting packet 2 getting packet 3
toLayer3: seqnum: -1  acknum: 1  checksum: 65534  payload:    retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 130208.34165471361  type: 2  entity: 0
aInput() called
toLayer3: seqnum: 2  acknum: -1  checksum: 65533  payload: yyyyyyyyyyyyyyyyyy  retransmission: true  sack: null
toLayer3: packet being lost
stopTimer: stopping timer at 130208.34165471361
startTimer: starting timer at 130208.34165471361

EVENT time: 130238.34165471361  type: 0  entity: 0
aTimerInterrupt() called
toLayer3: seqnum: 2  acknum: -1  checksum: 65533  payload: yyyyyyyyyyyyyyyyyy  retransmission: true  sack: null
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 130238.34165471361
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 130238.34165471361

EVENT time: 130240.36423841062  type: 2  entity: 1
bInput() called
bInput(): expecting packet 2 getting packet 2
toLayer3: seqnum: -1  acknum: 3  checksum: 65532  payload:    retransmission: false  sack: [0, 0, 0, 0, 0]
toLayer3: scheduling arrival on other side

EVENT time: 130242.87902462846  type: 2  entity: 0
aInput() called
stopTimer: stopping timer at 130242.87902462846
```

Go Back N

C1: No Loss + No Corruption

```
EVENT time: 43.4339426862392  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 43.4339426862392
generateNextArrival(): future time for event 1 at entity 0 will be 67.36045411542099
toLayer3: seqnum: 0  acknum: -1  checksum: 65535  payload: aaaaaaaaaaaaaaaaaaa  retransmission: false  sack: [I@8297b3a]
toLayer3: scheduling arrival on other side
stopTimer: stopping timer at 43.4339426862392
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 43.4339426862392

EVENT time: 45.66060365611743  type: 2  entity: 1
bInput() called
B receive a correct packet from A, the sequence number is: 0
receiver buffer: {}
the next packet expected is 1
B sent SACK: [-1, -1, -1, -1, -1]
toLayer3: seqnum: -1  acknum: 0  checksum: 65535  payload:   retransmission: false  sack: [I@22fcf7ab]
toLayer3: scheduling arrival on other side

EVENT time: 47.569475386822106  type: 2  entity: 0
aInput() called
print send buffer: [seqnum: 0  acknum: -1  checksum: 65535  payload: aaaaaaaaaaaaaaaaaaa  retransmission: false  sack: [I@8297b3a]]
A gets a correct ACK from B, the ack number is: 0
print send buffer:[]
stopTimer: stopping timer at 47.569475386822106
```

C2: ACK is Lost/Corrupted, Later ACK slides window by >1

```
EVENT time: 246.99136326181826  type: 2  entity: 1
bInput() called
B receive a correct packet from A, the sequence number is: 3
receiver buffer: {seqnum: 3  acknum: -1  checksum: 65532  payload: ddddddddddddddddddd  retransmission: false  sack: [I@17f7cd29]}
the next packet expected is 2
B sent SACK: [3, -1, -1, -1, -1]
toLayer3: seqnum: -1  acknum: 1  checksum: 65534  payload:   retransmission: false  sack: [I@7d8704ef]
toLayer3: packet being lost

EVENT time: 268.654744102298  type: 0  entity: 0
aTimerInterrupt() called
stopTimer: stopping timer at 268.654744102298
stopTimer: Warning: Unable to cancel your timer
sendFromAToB() called
toLayer3: seqnum: 2  acknum: -1  checksum: 65533  payload: ccccccccccccccccccc  retransmission: true  sack: [I@13b6aecc]
toLayer3: packet being lost
startTimer: starting timer at 268.654744102298

EVENT time: 298.654744102298  type: 0  entity: 0
aTimerInterrupt() called
stopTimer: stopping timer at 298.654744102298
stopTimer: Warning: Unable to cancel your timer
sendFromAToB() called
toLayer3: seqnum: 2  acknum: -1  checksum: 65533  payload: ccccccccccccccccccc  retransmission: true  sack: [I@158a8276]
toLayer3: scheduling arrival on other side
startTimer: starting timer at 298.654744102298

EVENT time: 305.46751304666276  type: 2  entity: 1
bInput() called
B receive a correct packet from A, the sequence number is: 2
receiver buffer: {}
the next packet expected is 4
B sent SACK: [-1, -1, -1, -1, -1]
toLayer3: seqnum: -1  acknum: 3  checksum: 65532  payload:   retransmission: false  sack: [I@3c3d9b6b]
toLayer3: packet being corrupted
toLayer3: scheduling arrival on other side

EVENT time: 308.00180059205906  type: 2  entity: 0
aInput() called
A gets a corrupt ACK from B, the ack number is: 3
```

C3: Data Packet is Lost/Corrupted, and Retransmitted After Timeout

```
EVENT time: 67.36045411542099  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 67.36045411542099
generateNextArrival(): future time for event 1 at entity 0 will be 83.31553086947233
toLayer3: seqnum: 1  acknum: -1  checksum: 65534  payload: bbbbbbbbbbbbbbbbbbb  retransmission: false  sack: [I@2de23121]
toLayer3: packet being lost
stopTimer: stopping timer at 67.36045411542099
stopTimer: Warning: Unable to cancel your timer
startTimer: starting timer at 67.36045411542099

EVENT time: 83.31553086947233  type: 1  entity: 0
generateNextArrival(): called
generateNextArrival(): time is 83.31553086947233
generateNextArrival(): future time for event 1 at entity 0 will be 238.65474410229803
toLayer3: seqnum: 2  acknum: -1  checksum: 65533  payload: cccccccccccccccccc  retransmission: false  sack: [I@63475ace]
toLayer3: packet being lost
stopTimer: stopping timer at 83.31553086947233
startTimer: starting timer at 83.31553086947233

EVENT time: 113.31553086947233  type: 0  entity: 0
aTimerInterrupt() called
stopTimer: stopping timer at 113.31553086947233
stopTimer: Warning: Unable to cancel your timer
sendFromAToB() called
toLayer3: seqnum: 1  acknum: -1  checksum: 65534  payload: bbbbbbbbbbbbbbbbbbb  retransmission: true  sack: [I@4988d8b8]
toLayer3: packet being corrupted
toLayer3: scheduling arrival on other side
startTimer: starting timer at 113.31553086947233

EVENT time: 114.74373607593004  type: 2  entity: 1
bInput() called
B receive a packet, the packet is corrupt!
```

C4: Data Packet is Lost/Corrupted, and Retransmitted After Receiving Duplicate ACK

```
EVENT time: 7096.437147129734  type: 2  entity: 0
aInput() called
print send buffer: [seqnum: 13  acknum: -1  checksum: 65522  payload: lllllllllllllllll  retransmission: true  sack: [I@4da4253], seqnum: 14  acknum
A get a duplicate ACK, the ack number is: 12
stopTimer: stopping timer at 7096.437147129734
sendFromAToB() called
toLayer3: seqnum: 13  acknum: -1  checksum: 65522  payload: lllllllllllllllll  retransmission: true  sack: [I@503d687a]
toLayer3: packet being corrupted
toLayer3: scheduling arrival on other side
startTimer: starting timer at 7096.437147129734

EVENT time: 7105.134678182317  type: 2  entity: 1
bInput() called
B receive a packet, the packet is corrupt!

EVENT time: 7126.437147129734  type: 0  entity: 0
aTimerInterrupt() called
stopTimer: stopping timer at 7126.437147129734
stopTimer: Warning: Unable to cancel your timer
sendFromAToB() called
toLayer3: seqnum: 13  acknum: -1  checksum: 65522  payload: lllllllllllllllll  retransmission: true  sack: [I@6a370f4]
toLayer3: scheduling arrival on other side
startTimer: starting timer at 7126.437147129734

EVENT time: 7135.3198034607985  type: 2  entity: 1
bInput() called
B receive a correct packet from A, the sequence number is: 13
receiver buffer: {}
the next packet expected is 15
B sent SACK: [-1, -1, -1, -1, -1]
toLayer3: seqnum: -1  acknum: 14  checksum: 65521  payload:   retransmission: false  sack: [I@2abf4075]
toLayer3: packet being corrupted
toLayer3: scheduling arrival on other side
```

C5: Retransmitted Data is Delivered, and Cumulative ACK Moves Window by >1

```
215  EVENT time: 363.79006317331465  type: 2  entity: 0
216  aInput() called
217  print send buffer: [seqnum: 2  acknum: -1  checksum: 65533  payload: ccccccccccccccccc  retransmission: true  sack: [I@5fbe4146], seqnum: 3  acknum:
218  A gets a correct ACK from B, the ack number is: 3
219  print send buffer:[]
220  stopTimer: stopping timer at 363.79006317331465
```

Tradeoffs and Discussion

Throughout my analysis, I came to a few interesting conclusions. After running tests with shorter timeout values, I observed that it reduced the time it took to detect and resent lost packets, which led to higher throughput. However, I noticed that this also led to more retransmissions which weren't necessary. While the longer timeout values reduced the unnecessary retransmissions, it consequently increased the average packet delay when errors of loss and corruption occurred.

Both SR and GBN+SACK handled the packet corruption by relying on timeouts for the recovery. When an acknowledgment or data packet was corrupted, the sender or receiver would eventually timeout and retransmit the missing data packet. What I noticed is that the two protocols are different in how many retransmissions they require in different error cases. SR is more conservative in the approach, while GBN+SACK resents more packets than would be needed depending on the amount of sequence numbers stored in the sack.

And if I were to expand on this project further, I would want to create a more elegant solution in handling the sequence number wrapping. For example, if the sequence number space only ranges from 0 to 65,535, and more than 65,535 packets are transmitted, the implementation of the protocols would need to be able to distinguish between the old and new packets in terms of when they wrapped around. In the TCP implementation, the handshake is necessary to initalize the sequence number, congestion windows, and other variables needed for finalizing a connection. Hypothetically, I think it could be useful to implement a flag or more logic to indicate when wrapping is occurring for a segment, so that both the sender and receiver are on the same page.