University of Dhaka
Department of Computer Science and Engineering

CSE 2213- Data and Telecommunications Lab

2nd Year – 2nd Semester
Session: 2016 - 2017

Experiment No.: 08

Name of the Experiment:

Comparative Performance Analysis of Go-Back-N and Selective-Repeat ARQ Mechanisms

Submitted by:

Tanjid Hasan Tonmoy, Roll: 09
Sakib Hasan, Roll: 63

Experiment Date: 02-11-2017
Submission Date: 09-11-2017

Submitted to:

1. Dr. Md. Rezaul Karim, Professor, Dept. of CSE, DU
2. Ms. Iffat Anjum, Lecturer, Dept. of CSE, DU

Outline

Problem Definition:

In this experiment we had to implement ***Selective-Repeat ARQ*** mechanism. It was said in the problem statement that each of our implementation must have the following features-

Implementation of Selective Repeat ARQ Mechanism: This experiment implements a flow control and error control mechanisms at the data link layer: Selective Repeat ARQ mechanism. Consider,

- In the sender program, data will read from a small text file (1 KB) and produce 8-byte frames.
- Sender window size: 8 (sequence numbers 0, 1, 2, 3, … …, 15).
- Sender program will start a timer for each sent-frame so that it can handle timeout-based retransmissions.
- Receiver window size: 8
- Receiver program will probabilistically (e.g. error<0.5) drop some frames.
- Receiver program should send Acknowledgement to the sender. Receiver will maintain a timer from start time. On time-expiration will send one acknowledgement for all the packets received on that time interval or will send repeat-request for the missing packet only.

There will be five distinct cases:

1. The frame will not be received by the receiver (lost/dropped during Transmission) [implementation on the sender side with send probability (P<0.3)]
2. The frame will be received by the receiver error free.
3. The frame will be received by the receiver with error(s)
    - In this case there will a negative acknowledgement
4. Acknowledgement frame will be received by the sender
5. Acknowledgement frame will not be received by the sender. (lost/dropped during transmission) [Implementation on the receiver side with send probability (P<0.3)]

A source file is taken as user input from which the sender reads and constructs the frames and windows while user output is a destination file where the receiver stores the uncorrupted data received from the sender.

Equipment required

1. JDK latest version
2. IDE or text editor
3. Two computers with proper internet connection
4. Complete knowledge of *Go-Back-N* and *Selective Repeat ARQ*

Theoretical Background:

Go-Back-N ARQ:

Definition:

Go-Back-N ARQ is a protocol where a sender can send a number of frames before receiving acknowledgement from the receiver side and keeps a copy of these frames until the acknowledgement arrives.

Brief History:

In networking and other areas, a task is often begun before the previous task has ended which is known as pipelining. There's no pipelining in Stop-and-Wait ARQ protocol. To improve the efficiency of transmission, mainly filling the pipe, multiple frames must be in transition while waiting for acknowledgement. One protocol that can achieve this goal is Go-Back-N ARQ protocol. In this protocol, we can send a number of frames as a window before receiving acknowledgements.

Operational Principles:

In Go-Back-N ARQ protocol, a sliding window mechanism provides flow control of data between the sender and the receiver. The sender sends packets of data as allowed by the current window size. If there are no errors in the packets, then the receiver accepts the packet, sends acknowledgement for the next frame and the window slides by one frame. In the event of an error, the receiver discards the received packet and the window remains the same as no sliding takes place. In this situation, if the receiver receives the packet with the next sequence number, it may send a negative acknowledgement (NAK). An NAK is usually implemented by sending the ACK asking for the desired packet. So, if the sender is already expecting an ACK for this packet, it will know that the packet in question was never received or was erroneous. It will then go-back by resetting the window position at the discarded packet and resend the packet and all its subsequent ones even if they had been already sent.

Essential mechanisms in Go-Back-N ARQ are-

1. **Sequence Numbers:** In Go-Back-N ARQ protocol, the sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.
2. **Sender Window:** The sender window is an abstract concept defining imaginary box of size 2^m-1, where m is the size of the sequence number field in bits. This window can slide one or more slots when a valid ACK arrives.
3. **Receiver Window:** The receiver window is an abstract concept of defining an imaginary box of size 1. This window slides when a correct frame has arrived, sliding occurs one slot at a time.

4. **Timer:** There can be a timer for each frame sent but in this protocol, we use only one timer. We send all the outstanding frames when the timer for the first outstanding frame expires.
5. **Acknowledgement:** The receiver sends positive ACK if a frame arrives safe and in order. If not, negative ACK or NAK is received by the sender from the receiver and all the frames are resent beginning with the desired one. The receiver can send one cumulative ACK for several frames.

Advantages:

1. The sender can send multiple frames at a time.
2. Send & Receive window size in only 1. So resources needed at receiver is limited.
3. Timer can be set for a group of frames.
4. Only one ACK can acknowledge one or more frames.
5. Easy to find and resend frames that were not received.
6. It avoids large bursts of packet delivery to higher layers.
7. Buffering and protocol processing at sender and receiver is simple and less complex. So waiting time is pretty low.
8. Size of the sender window can be altered.
   So, in less noisy channel, this mechanism serves more efficiently.

<u>Selective-Repeat ARQ:</u>

Definition:
In the Selective-Repeat ARQ protocol, when one frame is damaged, only the damaged frame is resent instead of all the frames. It is more efficient for the noisy links but the processing at the receiver end is very complex.

Brief History:
The Go-Back-N ARQ protocol works well if errors are rare, but if the line is poor, it wastes a lot of bandwidth on the retransmitted frames. An alternative strategy the Selective-Repeat ARQ protocol allows the receiver to accept and buffer the frames following a damaged or lost one.

Operational Principles:
In the Selective-Repeat ARQ protocol, the sender continuously sends frames which are held until acknowledged by the receiver. The NAK sent by the receiver specifies the block of data that is damaged and only that block is sent again. As only the erroneous block of data is retransmitted, performance improves but the receiver becomes very complex. Since data must be delivered in the proper sequence, correct blocks must be stored until all preceding erroneous blocks have been correctly received.

Essential mechanisms in Selective-Repeat ARQ are:

1. **Sender and Receiver Window:** The size of the sender and the receiver window must be at most one half of 2^m, where m is the size of the sequence number field in bits.
2. **ACKs and NAKs:** NAKs are sent to inform the sender to resend a frame. An NAK is sent once for each window position and defines the first slot in the window. ACKS are sent when data are delivered to the network layer. If the data belonging to n frames are delivered in one shot, only one ACK is sent for all of them.

Advantages:

1. Selective repeat protocol retransmits only that frame which is damaged or lost.
2. Selective-Repeat ARQ is efficient for noisy links.
3. Combines NAKs and cumulative ACKs.
4. Specifically indicates which packet is missing.
5. Limited memory required.
6. Better usage of Bandwidth.

Main differences between Go-Back-N and Selective-Repeat ARQ :

1. Go-Back-N protocol is designed to retransmit all the frames that are to arrive after the damaged or a lost frame. On the other hand, Selective Repeat protocol retransmits only the damaged or lost frame.
2. If the error rate is high meaning more frames are being damaged, then retransmitting all the frames that arrived after a damaged frame waste the bandwidth. On the other hand, selective repeat protocol retransmits only damaged frame so minimum bandwidth is wasted.
3. All the frames after the damaged frame are discarded and the retransmitted frames arrive in a sequence from a damaged frame onwards, so, there is less headache of sorting the frames hence it is less complex. On the other hand only damaged or suspected frame is retransmitted so, extra logic has to be applied for sorting hence, it is more complicated.
4. Go-Back-N has a window size of 2^m-1 and selective repeat have a window size of 2^{m-1}, where m is the size of the sequence number field in bits.
5. Neither sender nor receiver need the sorting algorithm in Go-Back-N. Whereas, receiver must be able to sort the frames as it has to maintain the sequence.
6. In Go-Back-N, receiver discards all the frames after the damaged frame so, it doesn't need to store any frames. Selective repeat protocol does not discard the frames arrived after the damaged frame instead it stores those frames till the damaged frame arrives successfully and is sorted in a proper sequence.
7. In selective repeat, NAK frame refers to the damaged frame number. In Go-Back-N, NAK frame refers to the next frame expected.

8. Generally, the Go-Back-N is more used due to its less complex nature instead of Selective Repeat protocol.

Comparison Chart:

COMPARISON TOPIC	GO-BACK-N	SELECTIVE REPEAT
Basic	Retransmits all the frames that is sent after the frame which is suspected to be corrupted or lost.	Retransmits only those frames that are suspected to lost or damaged.
Bandwidth Utilization	For noisy channel, a lot of bandwidth is wasted.	Comparatively less bandwidth is wasted in retransmitting.
Complexity	Less complicated.	More complex as it requires extra logic and sorting and storage.
Window size	2^m - 1	2^(m-1)
Sorting	Sorting is neither required at sender side nor at receiver side.	Receiver must be able to sort as it has to maintain the sequence of the frames.
Storing	Receiver does not store the frames received after the corrupted frame until the that frame is retransmitted.	Receiver stores the frames received after the damaged frame in the buffer until that frame is replaced.
Searching	Is not required	The sender must be able to search and select only the requested frame.
ACK Numbers	NAK number refer to the next expected frame number.	NAK number refer to the frame lost.
Use	It more often used.	Less used in practice because of its complexity.

Working principle:

A certain format to send each frame:

Source IP	Destination IP	Highest Bit Window Number (in Binary)	Frame Number (Binary)	Data Bits	CRC 16

Highest Bit window number is our abstraction methodology to transmit data. Bit size and frame number are both flexible because window size is taken as user input.

In Go-Back-N ARQ, we determined like that if a window size is n, then the highest frame number that can be transmitted is n.

In Selective-Repeat ARQ, if a window size is n, then the highest frame number which can be transmitted is 2*n.

For input 4 we will have

0	1	2	3	4

And for input 4 in Selective-Repeat ARQ, we will have like this

0	1	2	3	4	5	6	7	8

Error Detection:

For each data transmission we tried to implement error detection. This detection method procedure was implemented through CRC-16 (Cyclic Redundancy Check and it was of 17 bits).

Our CRC dividend was 10101010101010101.

It's an odd number. For each transmission, we sent 16 bit CRC remainder as trailer. Through this we wanted to detect if the data was erroneous or not. This was implemented both for data transmission and acknowledgement (ACK) or negative acknowledgement (NAK) signals.

Special Transmission handling:

We handled group acknowledgements, individual acknowledgements, NAKS individually to handle each case.

Timer and Threads:

Our whole implementation was very much dynamic. We used threads for each data frame and timer signals to stop deadlocking and to make the whole procedure much flexible.

Code:

Our whole code was modeled. And each part was designed so that each part can work as a separate entity. In the following we will show the parts of our code.

1) Class Server: This class is used to implement the Selective Repeat ARQ transmission protocol. This class has three main class.
just implements this protocol. Using same named two threads like the previous class definitions.

2) Class Client: This class is just the client class of the whole program. So according to the protocol it handles the program and saves data.

Selective Repeat:

The Transmission Control Protocol uses a variant of Selective Repeat ARQ to ensure reliable transmission of data over the Internet Protocol, which does not provide guaranteed delivery of packets; with Selective Acknowledgement (SACK) extension, it may also use Selective Repeat ARQ.

- **Pseudocode:**

```
Client()
{
      getProtcol();
      getWindowSize();
      startlCorrespondingThreads();
}

Sender extends Thread()
{
        readDataFromInputFile();
        while(all the data are not being sent )
       {
                 while(window is not filled)
                {
                          FormTheHeader();
                          takeDataForFrame();
                          callCRCGenerator();
                          FormTheTrailer();
                          FormTheFrame();
                          sendFrame();
                          startsTheTimer();
                 }

                 if(acknowledgement)
                 {
                           stopTimer();
                           slideWindow();
                 }
                else If(negative acknowledgement)
```

```
        {
                //In case of Go back N
                adjustWindow();
                resendAllTheTrailingFrames();
                //In case of selective
                sendThatParticularFrame();
        }
    }
}

Receive extends Thread()
{
    receiveFromReceiver();
    if(acknowledgement) InformTheSenderThread();
    elseIf (negative acknowledgement) InformTheSenderThread();

}

//Receiver side
Server()
{
    receiveSelectedProtocol();
    receiveWIndowSize();
    startAppropiateThread();
}

Receiver extends Thread()
{
    receiveData();
    If (sequence number is expected)
    {
            callCRC();
            if(dataIsCorrect)
            {
                // Go back N
                sendAcknowledgement();
                // Selective
                sendAcknowledgement();
                slideWindow();
            }
            Else
            {
                sendNegativeAcknowledgement();
            }
    }
}
```

Main Challenges and Their Solutions:

Main problem was the synchronization and inter-thread communication and sliding the windows.
The inter-thread communication was solved by using reference and the sliding window problem was solved using dynamic allocation and controlling.

Our sender side algorithm.

1) First choose which type of transmission protocol is needed.
2) Then read data from file
3) Make the use of calculation of Frame class to proper model the data, make frames, give window number etc.
4) Then start transmission through Go Back or Selective Request
5) In each transmission for each frame two thread starts.
6) They are controlled by their reference from control class
7) Each time window size equal frames are transmitted and thread starts to wait.
8) If there is some kind of acknowledge the total frame shifts. By this every moment the transmission goes one
9) Threads for each frame are continuously waiting and sending and receiving data using timer measurement.
10) After sending and receiving the threads stops. And total data is sent.

Our Receiver Side Algorithm

1) First receive what kind of protocol is being used and necessary information from the server.
2) They through a timer wait for data reception.
3) If valid data arrives (valid data with proper CRC, in go back desired frame and in selective the data from desired window.) acknowledgement signal is generated and sent if it's not valid then necessary information is being sent.
4) By this we save our data.
5) Close our connection
6) End Transmission

Applications:

In the field of data communication and networking, we can see various applications of these ARQ protocols-

Go-Back-N ARQ:

1. This protocol and its variants are widely used when the medium is error prone especially wireless or old POTS. LLC2 or HDLC are examples.

2. It is also used, with some sophistications, in wireless protocols like 802.11 or 802.16. Similar protocols are present in mobile radio protocols.
3. The Transmission Control Protocol uses a variant of Go-Back-N ARQ to ensure reliable transmission of data over the Internet Protocol.

Selective-Repeat ARQ:

1. For guaranteed delivery of packets, TCP uses Selective Repeat ARQ.
2. The ITU-T G.hn standard, which provides a way to create a high-speed (up to 1 Gigabit/s) Local Area Network(LAN) using existing home wiring (power lines, phone lines and coaxial cables), uses Selective Repeat ARQ to ensure reliable transmission over noisy media. G.hn employs packet segmentation to sub-divide messages into smaller units, to increase the probability that each one is received correctly.
3. The STANAG 5066 PROFILE FOR HF RADIO DATA COMMUNICATIONS uses Selective Repeat ARQ, with a maximum window size of 128 protocol-data units (PDUs).

Comparative Analysis:

The reasons that Selective Repeat performs better than Go-Back-N:

Selective-Repeat uses more memory but wastes less bandwidth:

Go-Back-N ARQ has a receiver window of 1, and so it requires less memory for buffers in the receiver. However since the receiver will not store frames that follow a lost frame, using Go-Back-N ARQ may cause retransmissions of properly received frames, wasting bandwidth. Selective-Repeat ARQ requires more buffer space in the receiver, and since it can store frames after a lost frame, it will not request unnecessary retransmissions and will not waste bandwidth.

The reverse situations where Go-Back-N is better than Selective Repeat:

In case of Stop-and-Wait protocol, the sender after sending a frame waits for the acknowledgement from the receiver before sending the next frame. This protocol works efficiently for long frames, where propagation time is small compared to the transmission time of the frame. This ensures an efficient transmission and when the number of frames needed to be sent are small and the link is noiseless, this protocol is more desirable than the Selective-Repeat ARQ protocol.

Throughput:

Simply, we can say throughput is how much work is completed or how much output is produced over specific period of time. Throughput is a measure of how fast we can actually send data through network. It is applied broadly to systems ranging from various aspects of computer and network systems to organizations. Related measures of system productivity include, the speed with which some specific workload can be completed, and response time, the amount of time between a single interactive user request and receipt of the response. Historically, throughput has been a measure of the comparative effectiveness of large commercial computers that run many programs concurrently.

Throughput = $\frac{\text{Number of information bits delivered to destination}}{\text{Total time required to deliver the information bits}}$

In data communication, throughput refers to the average data rate of successful data or message delivery over a specific communications link. It is measured in bits per second (bps).

<u>Example</u>: A communication link that has a maximum transfer rate of 100 Mbps has twice the throughput of a communication link that can only transfer data at 50 Mbps. Similarly, a 54 Mbps wireless connection has roughly 5 times as much throughput as an 11 Mbps connection. However, the actual data transfer speed may be limited by other factors such as the Internet connection speed and other network traffic. Therefore, it is good to remember that the maximum throughput of a device or network may be significantly higher than the actual throughput achieved in everyday use.

<u>Calculating Throughput for Go-Back-N and Selective-Repeat ARQ protocols:</u>

Suppose,

Propagation time, t_{prop} = distance * bit rate

Also, t_{prop} = distance/speed of signal

t_{frame} = bits transmitted / bit rate

Also, t_{frame} = bits transmitted * speed of signal

Let's, a = t_{prop}/t_{frame}

Utilization, U = t_{frame} / (t_{frame} + 2 * t_{prop})

=> U = 1 / (1 + 2 * a)

For **Go-Back-N**,

U =1 / (1 + 2 * a)

Throughput = Utilization * Bandwidth

=> Throughput_{GBN} = 1 / (1 + 2 * a) * Bandwidth

For **Selective Repeat,**

U =W / (1 + 2 * a)

Where,

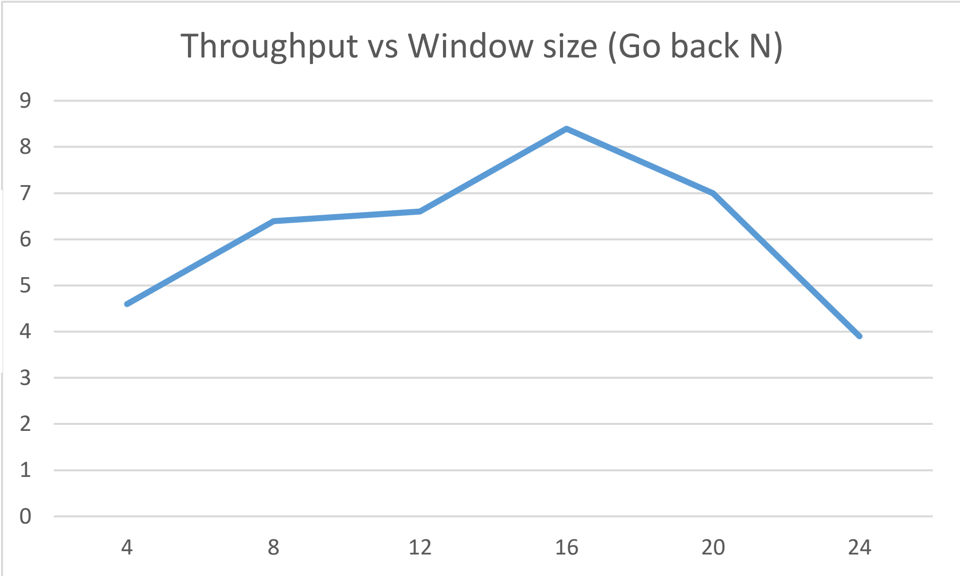
W = the window size

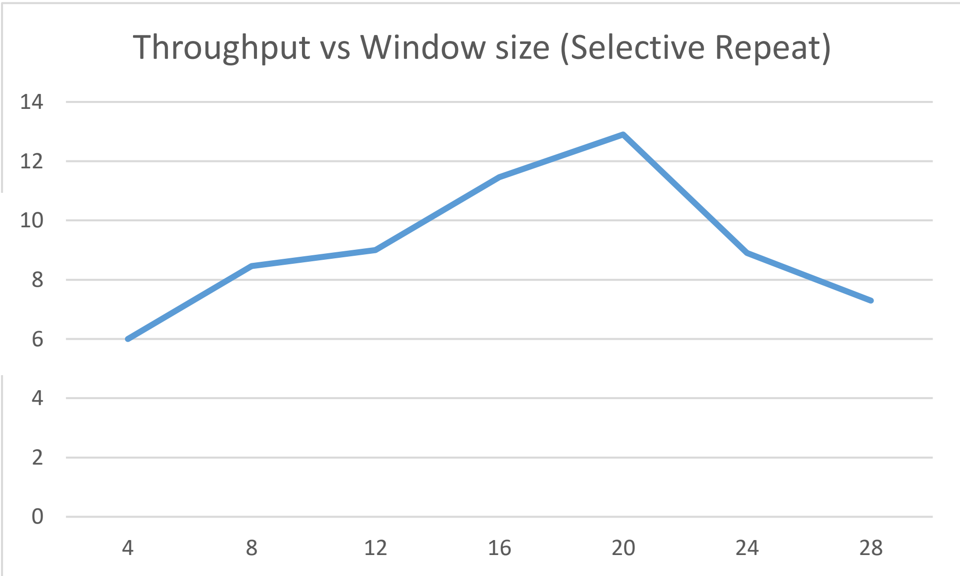
Throughput = Utilization * Bandwidth

=> Throughput_{SR} = W / (1 + 2 * a) * Bandwidth

Graphs:

Go-Back-N ARQ and Selective-Repeat ARQ for 1KB file:





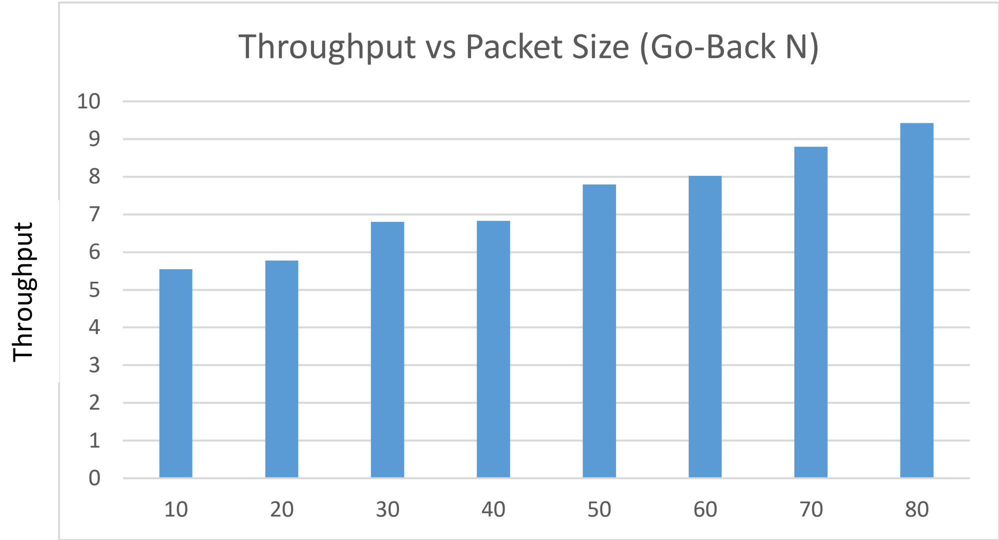
According to our idea, if window size increases, more bits can be transferred, so it can be said if window size increases, throughput increases, this supports equation,

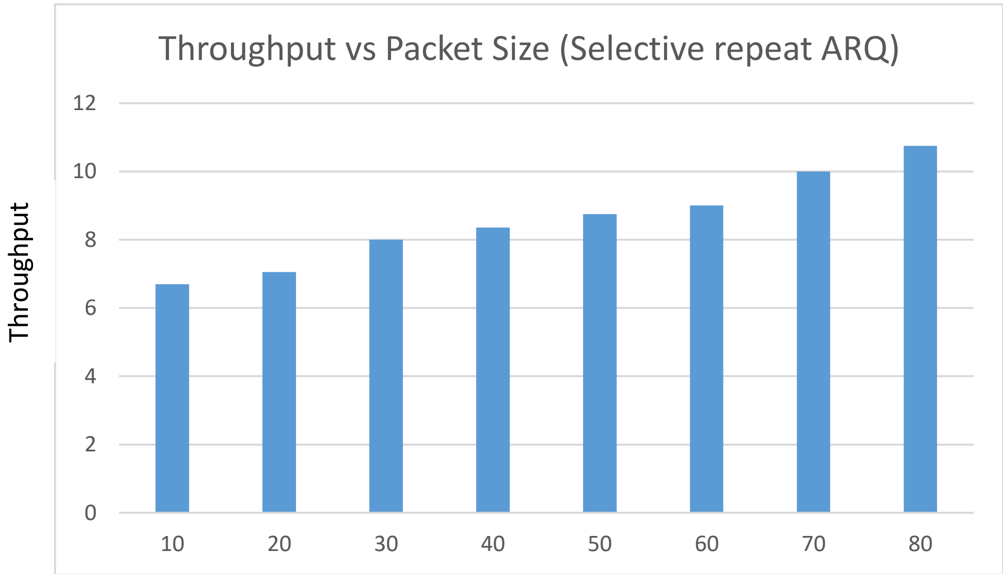
Throughput = (w)/(1+2a) * bandwidth

But too much increasing window size will not be necessary because then without no reason line will be left and for that there will be a downward sloping in curve. And generally throughput increasing is better observed in Selective-Repeat, as it’s comparatively faster than Go-Back-N ARQ.

For simplicity, we multiplied throughput with 10

Packet Size Vs Throughput:

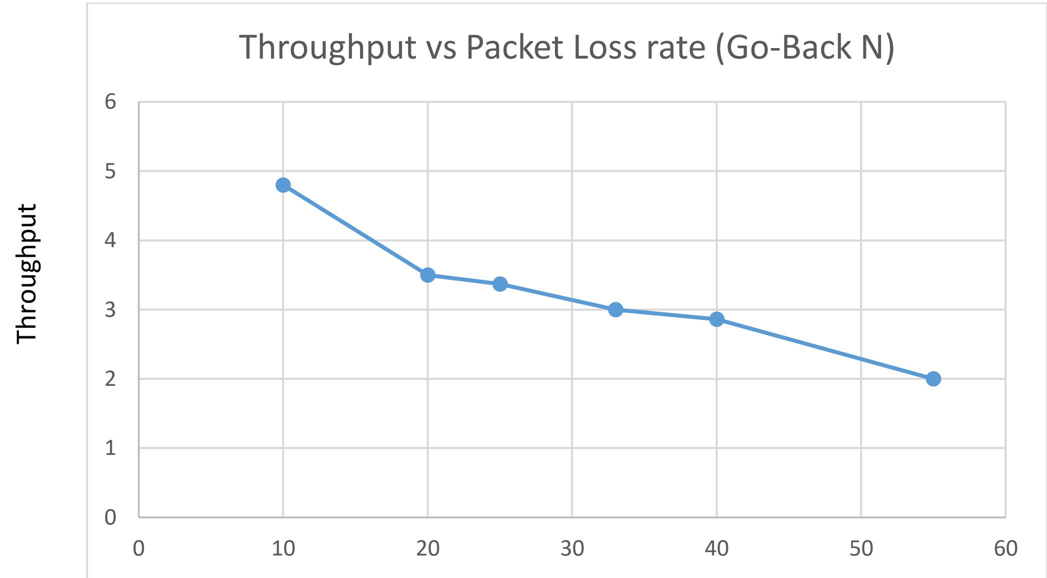


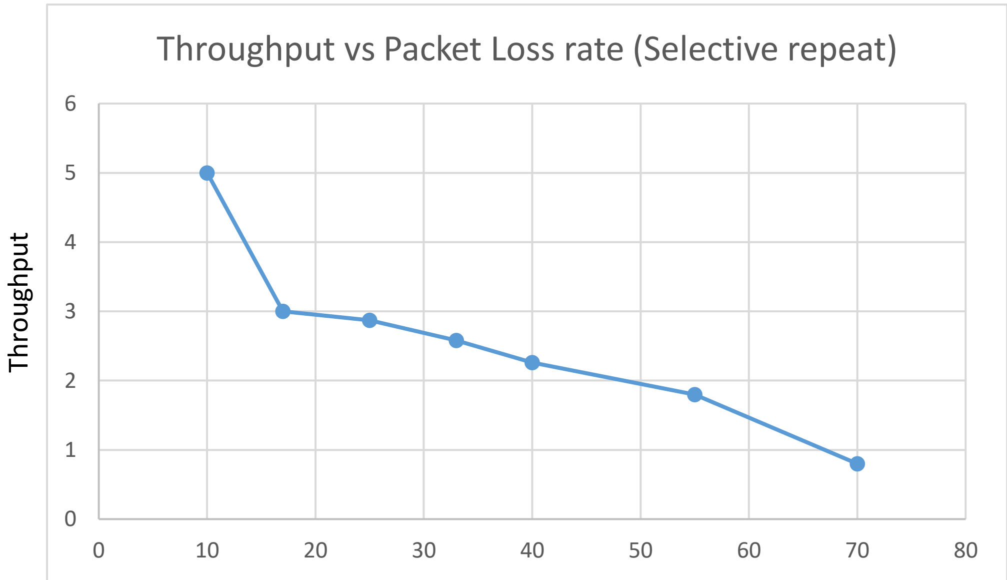


In our observation packet size increase, transfers large amount of data than comparatively less packet size. Time of transfer increases little bit in this case if window number is fixed. We have done it for a 1 KB size file with a window of 8 bits. In data bit part of the transfer we changed the numbers and tested. But it's in the same computer so the whole fact couldn't be properly experimented.

Packet Loss rate vs. Throughput:

For this perspective, what we did in the server side, we chose randomly some positions and then randomly selected whether we will send that packet or not.
Actually, what we saw this time is time increased than normally sending data but couldn't find such specific format or pattern. But it can be said that in selective, it works better. Because as the whole frame can be received in client side so total sequence is not sent, this reduces time but in Go-Back-N, it is needed to send frames from the desired position which increases time.





Observation:

Go Back N ARQ protocol is very much less efficient in extremely noisy channel as it discards all the frames that follow a corrupted. In a noisy channel, error occurs in a higher rate so most of the frames sent will be discarded. On the other hand selective ARQ protocol sends only the corrupted frame, so there is no overhead which makes it way more effective in noisy channel than Go back N. But in noiseless channel or less noisy channel Go Back N performs better than Selective as receiver side of Go Back N is simpler than that of selective. So, in less noisy channel Go Back N does better.

By doing this assignment and report, we were able to gain a clear conception of both the ARQ protocols theoretically and practically.
We got to understand the basic differences between the two protocols. By simulating the graphs, we were able to observe different throughputs in different situations and also different shapes of graphs. We also got to learn about throughput and how they differ in case of both the protocols and also their respective equations.

Conclusion:

We had to go through many challenges while working with this assignment. We had to discuss with each other and overcome those challenges thus it made ourselves more suitable for group projects. Finally, though it was tough enough but really interesting to do.
Both Go-Back-N and Selective-Repeat ARQ protocols are essential in data communication and networking. So making this report has come in pretty handy that will help us in the future as well to solve hard problems and learn more.