CS3205 - Computer Networks
Assignment 4: Go Back N and Selective Repeat Protocol
Prof. Siva Ram Murthy

May 11, 2021

Name:	Bharath Dakarapu
Roll No:	CS18B009

1 Objective

The objective of this project is to implement the Selective Repeat reliable transmission protocol and GobackN reliable transmission protocol and measure the roundtrip delays.
The project requires two separate programs, running at the same time, on two different hosts: a sender program that generates and transmits packets; and a receiver, that accepts the packets, and transmits the acknowledgments to the sender. Note that the receiver does not send any data packet; it only sends acknowledgments. Communication between the sender and receiver is through UDP sockets.

1.1 Introduction

Go Back N Protocol :

Go-Back-N ARQ is a specific instance of the automatic repeat request (ARQ) protocol, in which the sending process continues to send a number of frames specified by a window size even without receiving an acknowledgement (ACK) packet from the receiver. It is a special case of the general sliding window protocol with the transmit window size of N and receive window size of 1. It can transmit N frames to the peer before requiring an ACK.It uses frame pipelining so as to send multiple frames without waiting for ACks.It helps in achieving better link utilization, throughput so as to use bandwidth effectively. The number of frames that can be sent at a time totally depends on the size of the sender's window Go Back N

Protocol uses the principle of protocol pipelining in which the multiple frames can be sent before receiving the acknowledgment of the first frame. In Go-Back-N protocol the Sender Window Size (SWS) is N and Receiver Window Size (RWS) is always 1.Cumulative acknowledgements are used in GBN.Receiver receives in-order frame only and cannot accept (discards) frames out-of-sequence. Sender must re-send entire window in the event of an erroneous/lost frame.

However, this method also results in sending frames multiple times – if any frame was lost or damaged, or the ACK acknowledging them was lost or damaged, then that frame and all following frames in the send window (even if they were received without error) will be re-sent. To avoid this, Selective Repeat ARQ can be used.

Selective Repeat Protocol :

Selective repeat protocol, also called Selective Repeat ARQ (Automatic Repeat reQuest), is a data link layer protocol that uses sliding window method for reliable delivery of data frames. Here, only the erroneous or lost frames are retransmitted, while the good frames are received and buffered. Selective Repeat protocol provides for sending multiple frames depending upon the availability of frames in the sending window, even if it does not receive acknowledgement for any frame in the interim. The maximum number of frames that can be sent depends upon the size of the sending window. Induvidual acknowledgements are used in Selective Repeat Protocol

SR Repeat Protocol is used because we need the receiver to be able to accept packets out-of-order using buffer space, for a superior protocol to combine advantages of both Stop-Wait and GBN.Selective Repeat attempts to retransmit only those packets that are actually lost (due to errors). Independent acknowledgements are used in SR ARQ. Sorting mechanism at receiver's side adds to more complex implementation with SWS = RWS

When Buffer Space is of more concern than bandwidth then Go Back N Protocol is used,if Bandwidth is of more concern than buffer space then Selective Repeat Protocol is used.In Go-Back-N we have Less complexity,less CPU cycles while for Selective Repeat Protocol More processing power,cpu cycles at receiver.If error rate is low, use Go-back-N else if error rate is high use Selective Repeat Protocol .

2 Experimental Details

2.1 Simulation setup

Selective Repeat Protocol:

This project requires two separate programs, running at the same time, on two different hosts: a sender program that generates and transmits packets; and a receiver, that accepts the packets, and transmits the acknowledgments to the sender. Note that the receiver does not send any data packet; it only sends acknowledgments. Communication between the sender and receiver is through UDP sockets.

Go Back N Protocol:

Setup is same as of 0 Seletive Repeat prorocol But Cumulative ACKs will be used in Go-Back-N protocol

2.2 Entities involved and functions in each entity

Selective Repeat Protocol:

- **Sender:**
  The main loop of sender has these main steps:

  a. Generate a packet of length, where the packet length follows a uniform distribution:
  Uniform(40, MAX_PACKET_LENGTH) bytes, where MAX_PACKET_LENGTH is commandline param eter). The first byte(s) of the packet contains the sequence number (depends on the number of bits in the sequence number field).

  b. Packets are generated at periodic time intervals specified by the PACKET_GEN_RATE parameter (packets/unit time). The transmit buffer has a capacity specified by the BUFFER_SIZE parameter (number of packets, not bytes). A newly generated packet will be dropped if the Buffer is full. A sequence number is assigned ONLY if the packet is added to the buffer.

  c. Transmit the packet based on the Window conditions. Start the timeout timer for this packet's sequence number. The timeout is set to 300 ms for the first 10 packets and then $2 \times RTT_{ave}$ (in milliseconds) for all other packets.

  d. Process the next packet (when available) and transmit it if the sender window is not exhausted, i.e. the total number of unacknowledged packets is at most WINDOW_SIZE. Given an nbit sequence number, the maximum window size will be 2^{n-1} for Selective Repeat.

e. If an ACK packet arrives, process it, update local state variables and cancel timers corresponding to acknowl edged packets. Remove the packet from the Transmit Buffer. Note that selective ACKs are used. For each sequence number acknowledged, calculate the Roundtrip-Time (RTT) for the packet and update the average RTT (RTT_{ave}) for the packets acknowledged so far.

f. If a timer expires, retransmit only the unacknowledged packet.

g. The sender terminates after MAX_PACKETS (a commandline parameter) have been successfully ACKNOWLEDGED (OR) if the maximum retransmission attempts for any sequence number exceeds 10.

- **Receiver:**
  The receiver is always waiting to read a packet from the UDP socket it is listening to. Whenever a packet is delivered to the receiver:

  a. The receiver randomly decides that packet is corrupted and decides to drop the packet; note that you can use rand, rand48, etc. The probability of packet drop is specified as a commandline parameter, denoted by PACKET_ERROR_RATE. This step is used to simulate random network errors.

  b. If the packet is NOT corrupted (per step 1 above), the receiver reads the packet and extracts the sequence number. If the receiver buffer is FULL, then the received packets are discarded even if they were correctly received. Otherwise, it follows the Selective Repeat protocol for generating ACKs, and buffering outoforder packets.

  c. The ACK packets are NOT dropped and are always assumed to be delivered to the sender.

  d. The receiver terminates after acknowledging MAX_PACKETS (a commandline parameter).

Go Back N Protocol:

- **Sender:**
  The main loop of sender has these main steps:

  a. Generate a packet of length PACKET_LENGTH(command-line parameters) bytes. The first bytes of the packet contain the sequence number.Packet generation rate is given by the commandline parameter PACKET_GEN_RATE, in packets per second.

  b. You might need to use a thread that generates packets periodically (based on the above rate) and stores them in a buffer used by the sender's protocol. The maximum size of this sender transmission buffer is given by the commandline parameter, MAX_BUFFER_SIZE (number of packets, not bytes). A newly generated packet will be dropped if the Buffer is full. A sequence number is assigned ONLY if the packet is added to the buffer.

c. Transmit the packet based on the Window conditions. Start the time-out timer for this packet's sequence number. The timeout is set to 100 ms for the first 10 packets and then $2 \times RTT_{ave}$ (in milliseconds) for all other packets.

d. Process the next packet (when available) and transmit it if the sender window is not exhausted, i.e. the total number of unacknowledged packets is at most WINDOW_SIZE.

e. If an ACK packet arrives, process it, update local state variables and cancel timers corresponding to acknowledged packets. Note that cumulative ACKs are used. For each packet received, calculate the Round-tri-pTime (RTT) for the packet and update the average RTT (RTT_{ave}) for the packets acknowledged so far.

f. If a timer expires, retransmit only the unacknowledged packet.

g. The sender terminates after MAX_PACKETS (a commandline parameter) have been successfully ACKNOWLEDGED (OR) if the maximum retransmission attempts for any sequence number exceeds 5.

- **Receiver:**
  The receiver is always waiting to read a packet from the UDP socket it is listening to. Whenever a packet is delivered to the receiver:

  a. The receiver randomly decides that packet is corrupted and decides to drop the packet; note that you can use rand, rand48, etc. The probability of packet drop is specified as a commandline parameter, denoted RANDOM_DROP_PROB. This step is used to simulate random network errors.

  b. If the packet is NOT corrupted (per step 1 above), the receiver reads the packet and extracts the sequence number. If sequence number matches the NEXT EXPECTED sequence number, it transmits an ACK to the sender, and updates local state variables. Note that Cumulative ACKs are used by the receiver.

  c. The ACK packets are NOT dropped and are always assumed to be delivered to the sender.

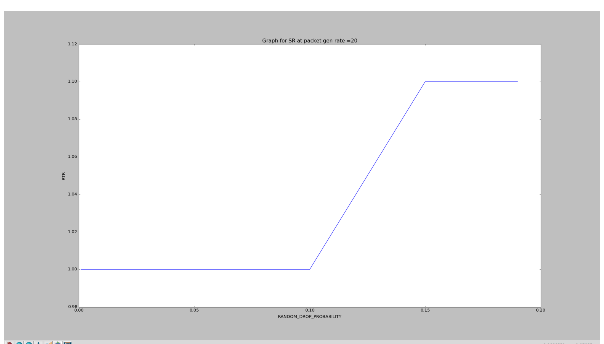
  d. The receiver terminates after acknowledging MAX_PACKETS (a commandline parameter).

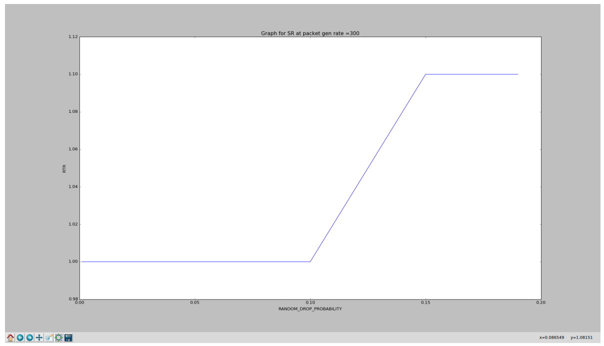
2.3 Additional details

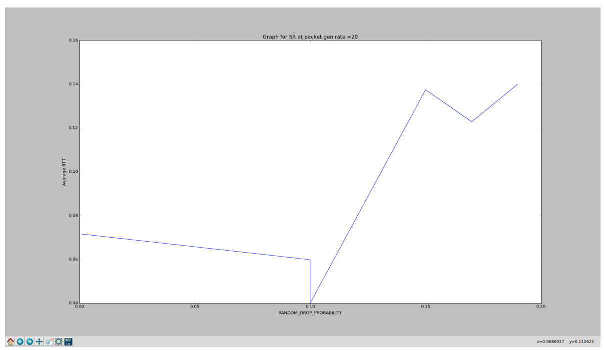
- Negative ACK's are excluded from Selective Repeat Protocol.
- OutPut for both Protocols In Non-Debug mode for Sender:
  - ReTransmission Ratio: Ratio of Total Number of Transmissions (including Retransmissions) to Number of Packets Acknowledged.
  - Average RTT Value for ALL Acknowledged Packets
- OutPut for both Protocols In Debug mode for Sender:
  - the Sender will print the following information for EACH packet when its ACK is received:
  - Seq : Time Generated: xx:yy RTT: zz Number of Attempts: aa where time is in milliseconds:microseconds format.
- OutPut for both Protocols In Debug mode for Receiver:
  - Receiver will print the following information for EACH packet when code stops executing. Note that the receiver will print this information ONLY in Sequence Number order.
  - Seq : Time Received: xx:yy where time is in milliseconds:microseconds format.

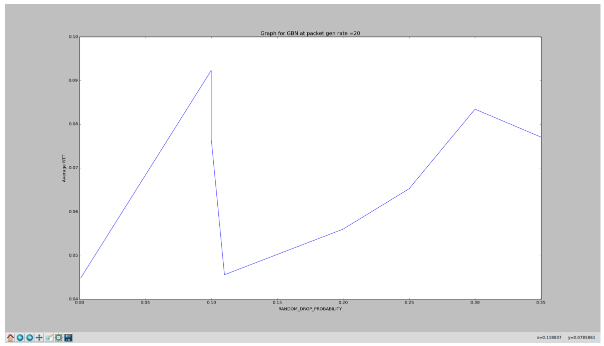
3 Results and Observations

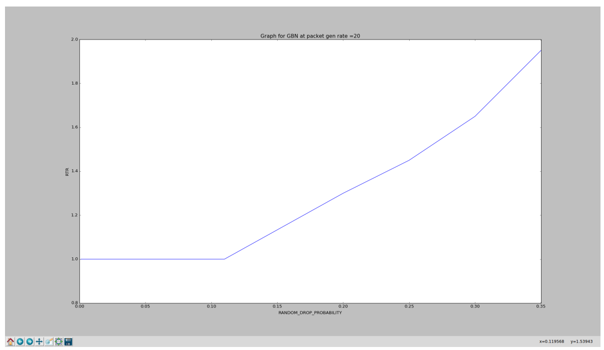
Here for both GBN and SR graphs for RTR vs packeterrorrate and avgrtr vs packet error rate for 2 different packet generation rates are:

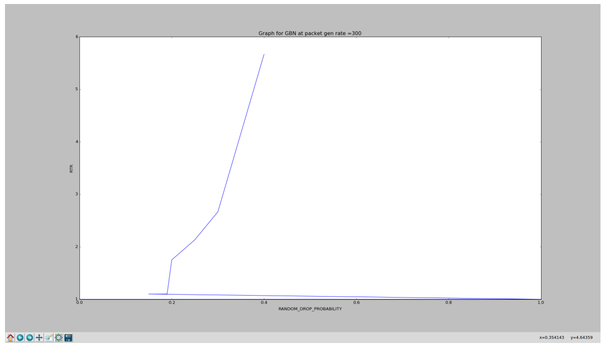



Graph for SR at packet gen rate =300
RTR
1.12
1.10
1.08
1.06
1.04
1.02
1.00
0.98
0.00
0.05
0.10
0.15
0.20
RANDOM_DROP_PROBABILITY
x=0.086549 y=1.08151


Graph for SR at packet gen rate =20
Average RTT
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.00
0.05
0.10
0.15
0.20
RANDOM_DROP_PROBABILITY
x=0.0686027 y=0.112622


Graph for GBN at packet gen rate =20
Average RTT
0.10
0.09
0.08
0.07
0.06
0.05
0.04
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
RANDOM_DROP_PROBABILITY
x=0.118837 y=0.0785861


Graph for GBN at packet gen rate =20
RTR
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
RANDOM_DROP_PROBABILITY
x=0.119568 y=1.53943



4 Learnings

From the above experiment i learned how to connect a UDP server-client setup.I learned how each of the protocols works and were used and i learned in which case which one is preferable.

5 Conclusion

Here in SR by increasing the packet error rate the no of retransmissions increases by which the rtr values increases.Similarly in GBN also the RTR value increased as we can observe the graphs.

References