CN ESE LAB

Name: Yash Oswal
Div: B   Roll no.:38
SRN: 201901226

CHIT NO: 17

Q: Simulate sliding window protocol using Selective Repeat with Positive Acknowledgement

Algorithm:

This protocol(SRP) is mostly identical to GBN protocol, except that buffers are used and the receiver, and the sender, each maintains a window of size. SRP works better when the link is very unreliable. Because in this case, retransmission tends to happen more frequently, selectively retransmitting frames is more efficient than retransmitting all of them. SRP also requires full-duplex link. backward acknowledgments are also in progress.

➢ Sender's Windows (Ws) = Receiver's Windows (Wr).
➢ Window size should be less than or equal to half the sequence number in SR protocol. This is to avoid packets being recognized incorrectly. If the size of the window is greater than half the sequence number space, then if an ACK is lost, the sender may send new packets that the receiver believes are retransmissions.
➢ Sender can transmit new packets as long as their number is with W of all unACKed packets.
➢ Sender retransmit un-ACKed packets after a timeout – Or upon a NAK if NAK is employed.
➢ Receiver ACKs all correct packets.
➢ Receiver stores correct packets until they can be delivered in order to the higher layer.
➢ In Selective Repeat ARQ, the size of the sender and receiver window must be at most one-half of 2^m.

CODE:

SERVER.CPP

```
#include <stdio.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define cls() printf("\t")

//structure definition for designing the packet.
struct frame
{
   int packet[40];
};
//structure definition for accepting the acknowledgement.
struct ack
{
    int acknowledge[40];
};

int main()
{
    int serversocket;
    sockaddr_in serveraddr, clientaddr;
    socklen_t len;
    int windowsize, totalpackets, totalframes, framessend = 0, i = 0, j = 0, k, l, m, n,
repacket[40];
    ack acknowledgement;
    frame f1;
    char req[50];
    serversocket = socket(AF_INET, SOCK_DGRAM, 0);
    bzero((char *)&serveraddr, sizeof(serveraddr));
    serveraddr.sin_family = AF_INET;
    serveraddr.sin_port = htons(5018);
    serveraddr.sin_addr.s_addr = INADDR_ANY;
    bind(serversocket, (sockaddr *)&serveraddr, sizeof(serveraddr));
    bzero((char *)&clientaddr, sizeof(clientaddr));
    len = sizeof(clientaddr);
    //connection establishment.
    printf("\nWaiting for client connection.\n");
    recvfrom(serversocket, req, sizeof(req), 0, (sockaddr *)&clientaddr, &len);
    printf("\nThe client connection obtained.\t %s\n", req);
    //sending request for windowsize.
    printf("\nSending request for window size.\n");
    sendto(serversocket, "REQUEST FOR WINDOWSIZE.", sizeof("REQUEST FOR WINDOWSIZE."), 0,
(sockaddr *)&clientaddr, sizeof(clientaddr));
    //obtaining windowsize.
    printf("\nWaiting for the windowsize.\n");
    recvfrom(serversocket, (char *)&windowsize, sizeof(windowsize), 0, (sockaddr
*)&clientaddr, &len);
    cls();
    printf("\nThe windowsize obtained as:\t %d\n", windowsize);
    printf("\nObtaining packets from network layer.\n");
    printf("\nTotal packets obtained:\t %d\n", (totalpackets = windowsize * 3));
    printf("\nTotal frames or windows to be transmitted:\t %d\n", (totalframes = 3));
    //sending details to client.
    printf("\nSending total number of packets.\n");
    sendto(serversocket, (char *)&totalpackets, sizeof(totalpackets), 0, (sockaddr
*)&clientaddr, sizeof(clientaddr));
    recvfrom(serversocket, req, sizeof(req), 0, (sockaddr *)&clientaddr, &len);
    printf("\nSending total number of frames.\n");
    sendto(serversocket, (char *)&totalframes, sizeof(totalframes), 0, (sockaddr
*)&clientaddr, sizeof(clientaddr));
    recvfrom(serversocket, req, sizeof(req), 0, (sockaddr *)&clientaddr, &len);
    printf("\nPRESS ENTER TO START THE PROCESS.\n");
    fgets(req, 2, stdin);
    cls();
    j = 0;
    l = 0; //starting the process of sending
    while (l < totalpackets)
    {
        //initialising the transmit buffer.
        bzero((char *)&f1, sizeof(f1));
        printf("\nInitialising the transmit buffer.\n");
        printf("\nThe frame to be send is %d with packets:\t", framessend);
        //Builting the frame.
        for (m = 0; m < j; m++)
        {
            //including the packets for which negative acknowledgement was received.
            printf("%d  ", repacket[m]);
            f1.packet[m] = repacket[m];
```

```
       }

       while (j < windowsize && i < totalpackets)
       {
          printf("%d  ", i);
          f1.packet[j] = i;
          i++;
          j++;
       }
       printf("\nSending frame %d\n", framessend);
       //sending the frame.
       sendto(serversocket, (char *)&f1, sizeof(f1), 0, (sockaddr *)&clientaddr, 
sizeof(clientaddr));
       //Waiting for the acknowledgement.
       printf("\nWaiting for the acknowledgement.\n");
       recvfrom(serversocket, (char *)&acknowledgement, sizeof(acknowledgement), 0, 
(sockaddr *)&clientaddr, &len);
       cls();
       //Checking acknowledgement of each packet.

       j = 0;
       k = 0;
       m = 0;
       n = l;

       while (m < windowsize && n < totalpackets)
       {
          if (acknowledgement.acknowledge[m] == -1)
          {
             printf("\nNegative acknowledgement received for packet: %d\n", 
f1.packet[m]);
             k = 1;
             repacket[j] = f1.packet[m];
             j++;
          }
          else
          {
             l++;
          }
          m++;
          n++;
       }
       if (k == 0)
       {
          printf("\nPositive acknowledgement received for all packets within the 
frame: %d\n", framessend);
       }
       framessend++;
       printf("\nPRESS ENTER TO PROCEED……\n");
       fgets(req, 2, stdin);
       cls();
    }
    printf("\nAll frames send successfully.\n\nClosing connection with the client.\n");
    close(serversocket);
}
```

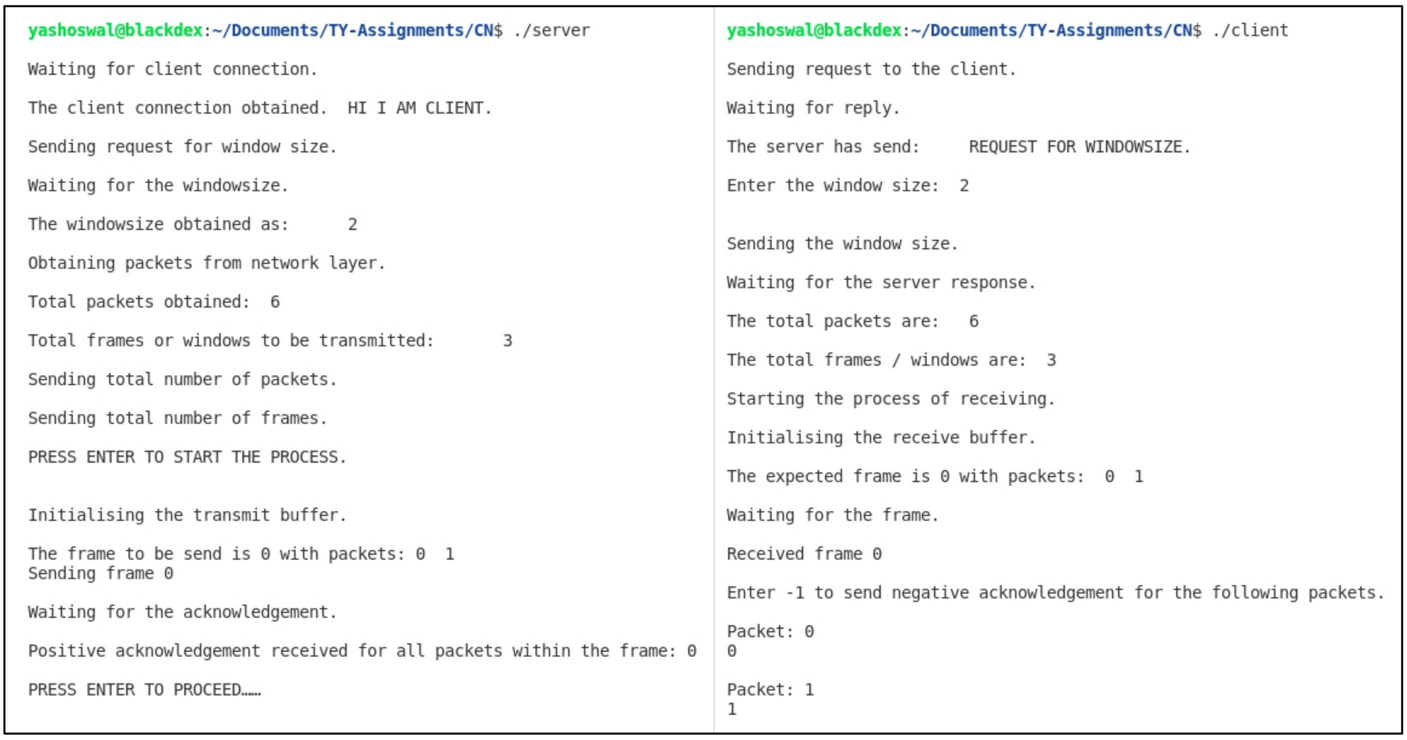
CLIENT.CPP

```
#include <iostream>
#include <stdio.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
```

```c
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define cls() printf("\t");
//structure definition for accepting the packets.

struct frame
{
   int packet[40];
};

//structure definition for constructing the acknowledgement frame

struct ack
{
   int acknowledge[40];
};

int main()
{
   int clientsocket;
   sockaddr_in serveraddr;
   socklen_t len;
   hostent *server;
   frame f1;
   int windowsize, totalpackets, totalframes, i = 0, j = 0, framesreceived = 0, k, l, m,
repacket[40];
   ack acknowledgement;
   char req[50];
   clientsocket = socket(AF_INET, SOCK_DGRAM, 0);
   bzero((char *)&serveraddr, sizeof(serveraddr));
   serveraddr.sin_family = AF_INET;
   serveraddr.sin_port = htons(5018);
   server = gethostbyname("127.0.0.1");
   bcopy((char *)server->h_addr, (char *)&serveraddr.sin_addr.s_addr,
sizeof(server->h_addr));
   //establishing the connection.
   printf("\nSending request to the client.\n");
   sendto(clientsocket, "HI I AM CLIENT.", sizeof("HI I AM CLIENT."), 0, (sockaddr
*)&serveraddr, sizeof(serveraddr));
   printf("\nWaiting for reply.\n");
   recvfrom(clientsocket, req, sizeof(req), 0, (sockaddr *)&serveraddr, &len);
   printf("\nThe server has send:\t %s\n", req);
   //accepting window size from the user.
   printf("\nEnter the window size:\t");
   scanf("%d", &windowsize);
   //sending the window size.
   printf("\n\nSending the window size.\n");
   sendto(clientsocket, (char *)&windowsize, sizeof(windowsize), 0, (sockaddr
*)&serveraddr, sizeof(serveraddr));
   cls();

   //collecting details from server.

   printf("\nWaiting for the server response.\n");
   recvfrom(clientsocket, (char *)&totalpackets, sizeof(totalpackets), 0, (sockaddr
*)&serveraddr, &len);
   printf("\nThe total packets are:\t %d\n", totalpackets);
   sendto(clientsocket, "RECEIVED.", sizeof("RECEIVED."), 0, (sockaddr *)&serveraddr,
sizeof(serveraddr));
   recvfrom(clientsocket, (char *)&totalframes, sizeof(totalframes), 0, (sockaddr
*)&serveraddr, &len);
   printf("\nThe total frames / windows are:\t %d\n", totalframes);
```

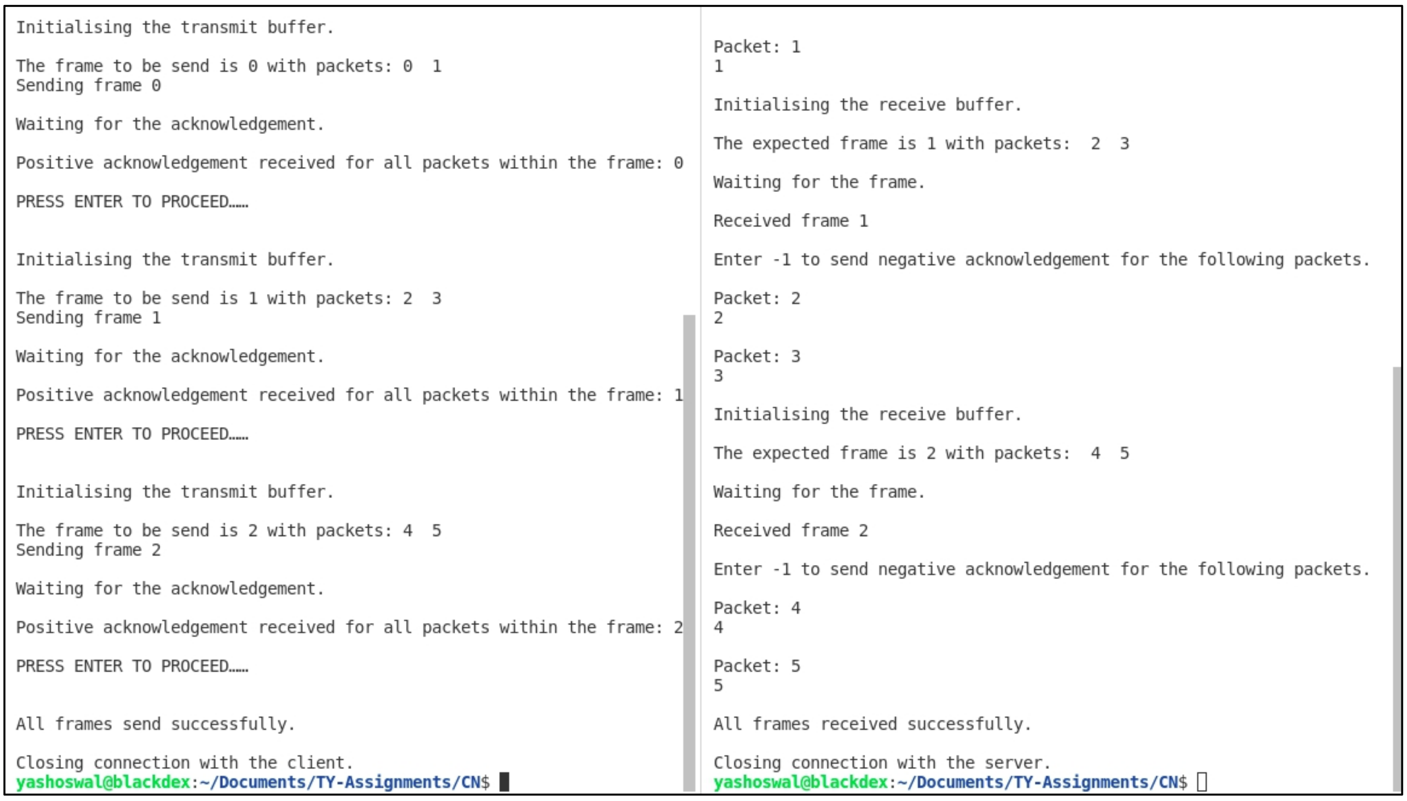
```
    sendto(clientsocket, "RECEIVED.", sizeof("RECEIVED."), 0, (sockaddr *)&serveraddr,
sizeof(serveraddr));

    //starting the process.
    printf("\nStarting the process of receiving.\n");
    j = 0;
    l = 0;
    while (l < totalpackets)
    { //initialising the receive buffer.
        printf("\nInitialising the receive buffer.\n");
        printf("\nThe expected frame is %d with packets:  ", framesreceived);
        for (m = 0; m < j; m++)
        { //readjusting for packets with negative acknowledgement.
            printf("%d  ", repacket[m]);
        }
        while (j < windowsize && i < totalpackets)
        {
            printf("%d  ", i);
            i++;
            j++;
        }
        printf("\n\nWaiting for the frame.\n");
        //accepting the frame.
        recvfrom(clientsocket, (char *)&f1, sizeof(f1), 0, (sockaddr *)&serveraddr, &len);
        printf("\nReceived frame %d\n\nEnter -1 to send negative acknowledgement for the
following packets.\n", framesreceived);
        //constructing the acknowledgement frame.
        j = 0;
        m = 0;
        k = l;
        while (m < windowsize && k < totalpackets)
        {
            printf("\nPacket: %d\n", f1.packet[m]);
            //accepting acknowledgement from the user.
            scanf("%d", &acknowledgement.acknowledge[m]);
            if (acknowledgement.acknowledge[m] == -1)
            {
                repacket[j] = f1.packet[m];
                j++;
            }
            else
            {
                l++;
            }
            m++;
            k++;
        }
        framesreceived++;
        //sending acknowledgement to the server.
        sendto(clientsocket, (char *)&acknowledgement, sizeof(acknowledgement), 0,
(sockaddr *)&serveraddr, sizeof(serveraddr));
        cls();
    }
    printf("\nAll frames received successfully.\n\nClosing connection with the server.\n");
    close(clientsocket);
}
```

OUTPUT:



```
yashoswal@blackdex:~/Documents/TY-Assignments/CN$ ./server

Waiting for client connection.

The client connection obtained.  HI I AM CLIENT.

Sending request for window size.

Waiting for the windowsize.

The windowsize obtained as:      2

Obtaining packets from network layer.

Total packets obtained:  6

Total frames or windows to be transmitted:       3

Sending total number of packets.

Sending total number of frames.

PRESS ENTER TO START THE PROCESS.


Initialising the transmit buffer.

The frame to be send is 0 with packets: 0  1
Sending frame 0

Waiting for the acknowledgement.

Positive acknowledgement received for all packets within the frame: 0

PRESS ENTER TO PROCEED……
```

```
yashoswal@blackdex:~/Documents/TY-Assignments/CN$ ./client

Sending request to the client.

Waiting for reply.

The server has send:     REQUEST FOR WINDOWSIZE.

Enter the window size:  2


Sending the window size.

Waiting for the server response.

The total packets are:    6

The total frames / windows are:  3

Starting the process of receiving.

Initialising the receive buffer.

The expected frame is 0 with packets:  0  1

Waiting for the frame.

Received frame 0

Enter -1 to send negative acknowledgement for the following packets.

Packet: 0
0

Packet: 1
1
```



```
Initialising the transmit buffer.

The frame to be send is 0 with packets: 0  1
Sending frame 0

Waiting for the acknowledgement.

Positive acknowledgement received for all packets within the frame: 0

PRESS ENTER TO PROCEED……


Initialising the transmit buffer.

The frame to be send is 1 with packets: 2  3
Sending frame 1

Waiting for the acknowledgement.

Positive acknowledgement received for all packets within the frame: 1

PRESS ENTER TO PROCEED……


Initialising the transmit buffer.

The frame to be send is 2 with packets: 4  5
Sending frame 2

Waiting for the acknowledgement.

Positive acknowledgement received for all packets within the frame: 2

PRESS ENTER TO PROCEED……


All frames send successfully.

Closing connection with the client.
yashoswal@blackdex:~/Documents/TY-Assignments/CN$
```

```
Packet: 1
1

Initialising the receive buffer.

The expected frame is 1 with packets:  2  3

Waiting for the frame.

Received frame 1

Enter -1 to send negative acknowledgement for the following packets.

Packet: 2
2

Packet: 3
3

Initialising the receive buffer.

The expected frame is 2 with packets:  4  5

Waiting for the frame.

Received frame 2

Enter -1 to send negative acknowledgement for the following packets.

Packet: 4
4

Packet: 5
5

All frames received successfully.

Closing connection with the server.
yashoswal@blackdex:~/Documents/TY-Assignments/CN$
```