BITS Pilani - Hyderabad Campus
CS F303 (Computer Networks)
Second Semester 2023-24, Lab Sheet 6
Go-Back-N and Selective Repeat

1. Overview

In this lab, we will implement two sliding-window protocols, namely, Go-Back-N and Selective Repeat. We will use the Socket programming to utilize sockets for the communication between application and transport layer. Additionally, we will also utilize the Mininet, to implement a virtualized environment for simulating hosts, switches, and routers. We will emulate network configurations such as delay and packet losses to study their impact on these two protocols.

2. Sliding Window Protocols: Go-Back-N and Selective Repeat

The Go-Back-N protocol is a sliding window protocol used for reliable data transfer in computer networks. It is given by the following Finite State Machines (FSMs):

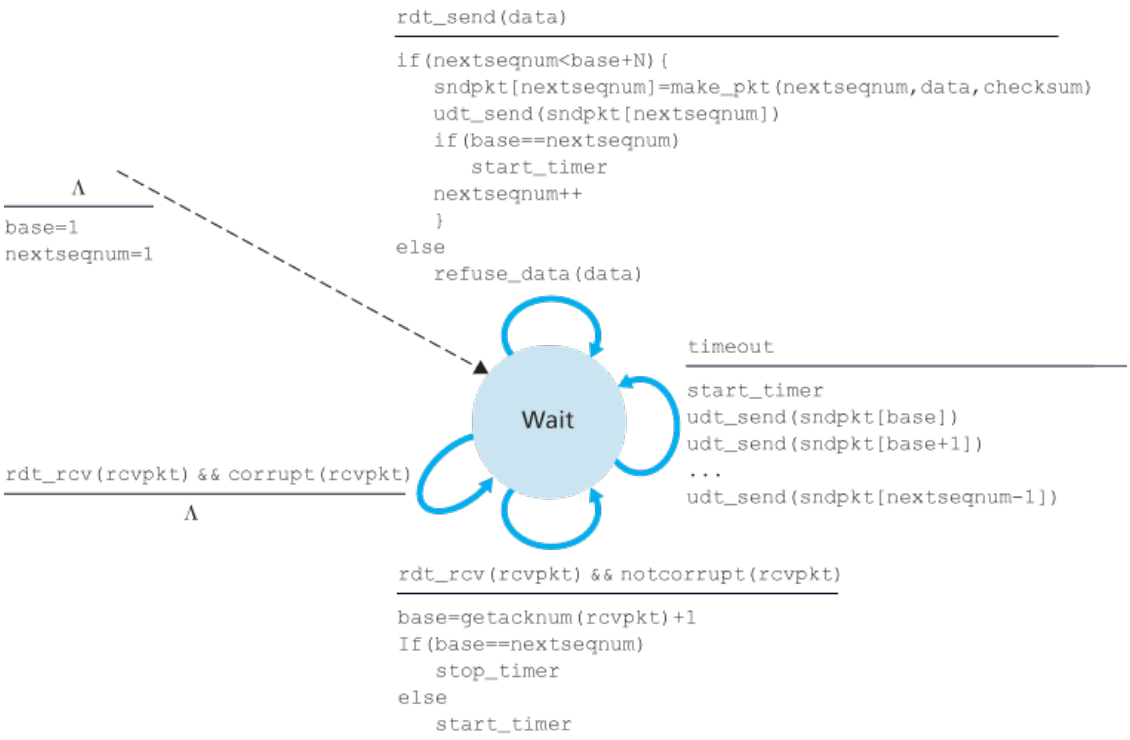

Figure 1 GBN operation at the Sender side

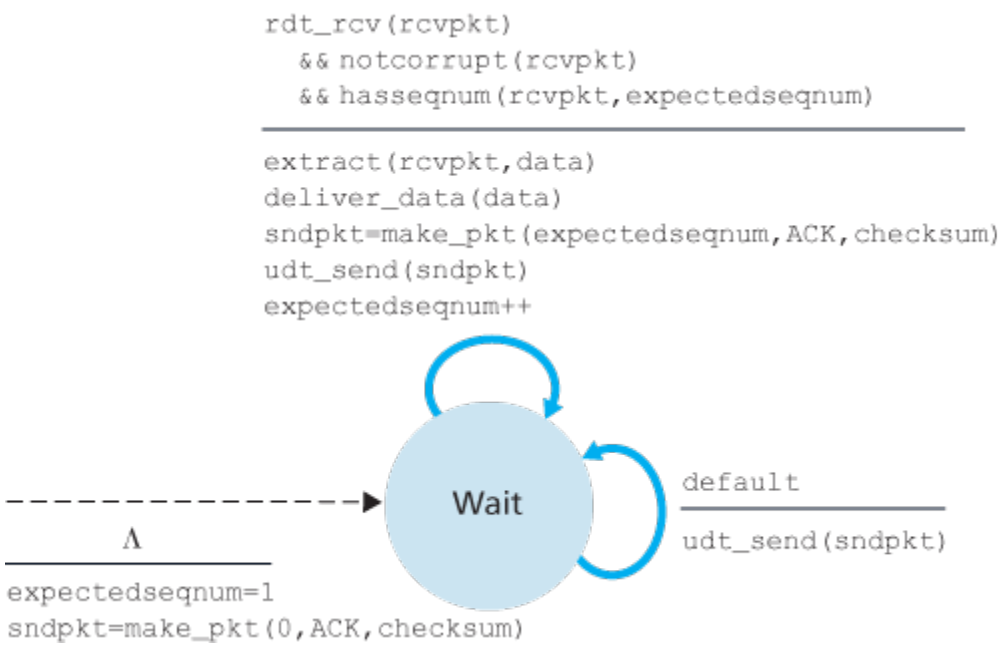

Figure 2 GBN operation at the receiver side

Although, Go-Back-N uses pipelining, it may be inefficient due to a high number of re-transmissions. Hence, Selective Repeat, another sliding window protocol, uses retransmissions of only selected packets. The events and actions of Selective Repeat are given as follows:

1. *Data received from above.* When data is received from above, the SR sender checks the next available sequence number for the packet. If the sequence number is within the sender's window, the data is packetized and sent; otherwise it is either buffered or returned to the upper layer for later transmission, as in GBN.
2. *Timeout.* Timers are again used to protect against lost packets. However, each packet must now have its own logical timer, since only a single packet will be transmitted on timeout. A single hardware timer can be used to mimic the operation of multiple logical timers [Varghese 1997].
3. *ACK received.* If an ACK is received, the SR sender marks that packet as having been received, provided it is in the window. If the packet's sequence number is equal to `send_base`, the window base is moved forward to the unacknowledged packet with the smallest sequence number. If the window moves and there are untransmitted packets with sequence numbers that now fall within the window, these packets are transmitted.

Figure 3 Selective Repeat Sender: Events and Actions

1. *Packet with sequence number in* `[rcv_base, rcv_base+N-1]` *is correctly received.* In this case, the received packet falls within the receiver's window and a selective ACK packet is returned to the sender. If the packet was not previously received, it is buffered. If this packet has a sequence number equal to the base of the receive window (`rcv_base` in Figure 3.22), then this packet, and any previously buffered and consecutively numbered (beginning with `rcv_base`) packets are delivered to the upper layer. The receive window is then moved forward by the number of packets delivered to the upper layer. As an example, consider Figure 3.26. When a packet with a sequence number of `rcv_base=2` is received, it and packets 3, 4, and 5 can be delivered to the upper layer.
2. *Packet with sequence number in* `[rcv_base-N, rcv_base-1]` *is correctly received.* In this case, an ACK must be generated, even though this is a packet that the receiver has previously acknowledged.
3. *Otherwise.* Ignore the packet.

Figure 4 Selective Repeat Receiver: Events and Actions

3. Deployment of Go Back N and Selective Repeat protocol over Mininet

In this lab, we will be using Mininet to develop a network topology (one sender and one receiver) and implement sliding window protocols. Please follow these steps:

- To setup and use Mininet, follow the previous labsheets.
- Create a standard topology as shown in Figure 3.

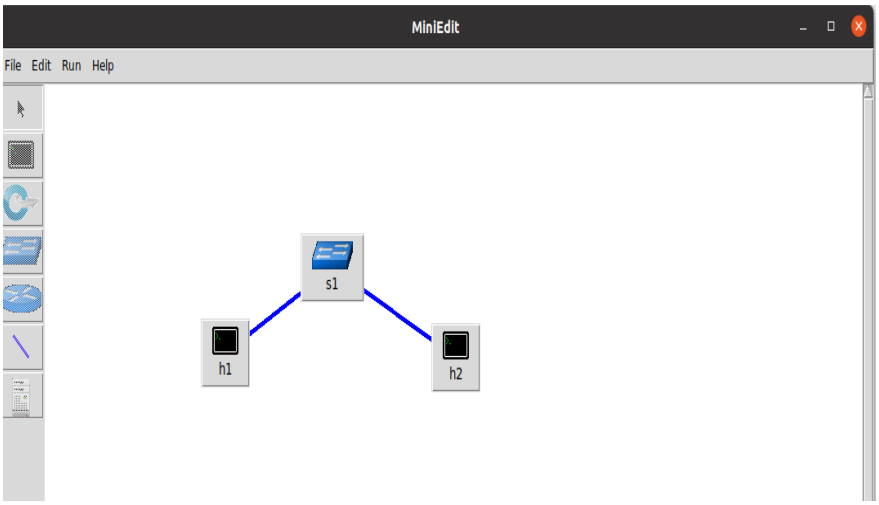

Figure 5 Topology (One sender and one receiver)

- Configure IP addresses of h1 and h2 nodes, and start running mininet.
- Develop C programs for both the sender and receiver functionalities. Place the **sender.c file** on host 1 (h1) and the **receiver.c** file on host 2 (h2). These codes are given in the next section.
- Execute the program without adding delay by running receiver.c first and then sender.c.
- Add a delay on both the link between h1-s1 and s1-h2. Execute the programs and observe the impact of the delay introduced.

Note: Prior to compilation, ensure to input the IP address of the receiver into the sender program. Then, compile both the sender and receiver programs on mininet terminals of both h1 and h2.

4. C programs for implementation of Sliding Window Protocols

4.1 C programs for Go-Back-N

Code for Go-Back-N (sender.c)

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<time.h>
#include<stdlib.h>
#include<ctype.h>

#define W 5

char a[10];
char b[10];

void alpha9(int);

int main()
{
   int s,f,wl,c=1,x,i=0,j,n,p=0,e=0;
   struct sockaddr_in ser;
   s=socket(AF_INET,SOCK_STREAM,0);
   ser.sin_family=AF_INET;
   ser.sin_port=6500;
```

```
ser.sin_addr.s_addr=inet_addr("127.0.0.1");
connect(s,(struct sockaddr *) &ser, sizeof(ser));
printf("\nTCP Connection Established.\n");
printf("\nEnter the number of Frames: ");
scanf("%d",&f);
alpha9(f);
send(s,a,sizeof(a),0);
strcpy(b,"Time Out ");
while(1)
{
   for(i=0;i<W;i++)
   {
      alpha9(c);
      send(s,a,sizeof(a),0);
      if(c<=f)
      {
         printf("\nFrame %d Sent",c);
         c++;
      }
   }
   i=0;
   wl=W;
   while(i<W)
   {
      recv(s,a,sizeof(a),0);
      p=atoi(a);
      if(strcmp(a,b)==0)
      {
         e=c-wl;
                  if(e<f)
                  {
                     printf("\nTime Out, Resent Frame %d onwards",e);
                  }
                  break;
      }
      else
      {
         if(p<=f)
```

```
            {
               printf("\nFrame %s Acknowledged",a);
               wl--;
            }
            else
            {
               break;
            }
         }
         if(p>f)
         {
            break;
         }
         i++;
      }
      if(wl==0 && c>f)
      {
         send(s,b,sizeof(b),0);
         break;
      }
      else
      {
         c=c-wl;
         wl=W;
      }
   }
   close(s);
   return 0;
}

void alpha9(int z)
{
   int k,i=0,j,g;
   k=z;
   while(k>0)
   {
      i++;
      k=k/10;
```

```
    }
    g=i;
    i--;
    while(z>0)
    {
        k=z%10;
        a[i]=k+48;
        i--;
        z=z/10;
    }
    a[g]='\0';
}
```

Code for Go-Back-N (receiver.c)

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<time.h>
#include<stdlib.h>
#include<ctype.h>
#include<arpa/inet.h>

#define W 5
#define P1 50
#define P2 0

char a[10];
char b[10];
void alpha9(int);

int main()
{
    struct sockaddr_in ser,cli;
    int s,n,sock,i,j,c=1,f;
    unsigned int s1;
```

```
s=socket(AF_INET,SOCK_STREAM,0);
ser.sin_family=AF_INET;
ser.sin_port=6500;
ser.sin_addr.s_addr=inet_addr("127.0.0.1");
bind(s,(struct sockaddr *) &ser, sizeof(ser));
listen(s,1);
n=sizeof(cli);
sock=accept(s,(struct sockaddr *)&cli, &n);
printf("\nTCP Connection Established.\n");
s1=(unsigned int) time(NULL);
srand(s1);
strcpy(b,"Time Out ");
recv(sock,a,sizeof(a),0);
f=atoi(a);
while(1)
{
   for(i=0;i<W;i++)
   {
      recv(sock,a,sizeof(a),0);
      if(strcmp(a,b)==0)
      {
         break;
      }
   }
   i=0;
   while(i<W)
   {
      j=rand()%P1;
      if(j<=P2)
      {
         send(sock,b,sizeof(b),0);
         break;
      }
      else
      {   //printf("%d",c);
      //printf("%d",f);
         alpha9(c);
         //printf("\nFrame %s Received new ",a);
```

```
            if(c<=f)
            { //printf("%d",c);
          //printf("%d",f);

              if(c==f)
              {printf("\nFrame %d received",c);}
              printf("\nFrame %s Received ",a);
              send(sock,a,sizeof(a),0);
            }
            else
            {
              break;
            }
           // printf("%d--->",c);
            c++;
          }
          if(c>f)
          {
            break;
          }
          i++;
      }
   }
   close(sock);
   close(s);
   return 0;
}

void alpha9(int z)
{
   int k,i=0,j,g;
   k=z;
   while(k>0)
   {
      i++;
      k=k/10;
   }
   g=i;
```

```
   i--;
   while(z>0)
   {
      k=z%10;
      a[i]=k+48;
      i--;
      z=z/10;
            }
   a[g]='\0';
}
```

4.2 C programs for Selective Repeat

Code for Selective Repeat (Sender.c)

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<time.h>
#include<stdlib.h>
#include<ctype.h>

#define W 5

char a[10];
char b[10];
void alpha9(int);
int con();

int main()
{
   int s,f,wl,c=1,x,i=0,j,n,p=0,e=0;
   struct sockaddr_in ser;
   s=socket(AF_INET,SOCK_STREAM,0);
   ser.sin_family=AF_INET;
   ser.sin_port=6500;
   ser.sin_addr.s_addr=inet_addr("127.0.0.1");
   connect(s,(struct sockaddr *) &ser, sizeof(ser));
```

```
printf("\nTCP Connection Established.\n");
printf("\nEnter the number of Frames: ");
scanf("%d",&f);
alpha9(f);
send(s,a,sizeof(a),0);
strcpy(b,"Time Out ");
while(1)
{
   for(i=0;i<W;i++)
   {
      alpha9(c);
      send(s,a,sizeof(a),0);
      if(c<=f)
      {
         printf("\nFrame %d Sent",c);
         c++;
      }
   }
   i=0;
   wl=W;
   while(i<W)
   {
      recv(s,a,sizeof(a),0);
      p=atoi(a);
      if(a[0]=='N')
      {
         e=con();
         if(e<f)
         {
            printf("\nNAK %d",e);
            printf("\nFrame %d sent",e);
            i--;
         }
      }
      else
      {
         if(p<=f)
         {
```

```
            printf("\nFrame %s Acknowledged",a);
            wl--;
          }
          else
          {
            break;
          }
        }
        if(p>f)
        {
          break;
        }
        i++;
      }
      if(wl==0 && c>f)
      {
        send(s,b,sizeof(b),0);
        break;
      }
      else
      {
        c=c-wl;
        wl=W;
      }
    }
    close(s);
    return 0;
}

void alpha9(int z)
{
    int k,i=0,j,g;
    k=z;
    while(k>0)
    {
      i++;
      k=k/10;
    }
```

```
   g=i;
   i--;
   while(z>0)
   {
      k=z%10;
      a[i]=k+48;
      i--;
      z=z/10;
   }
   a[g]='\0';
}

int con()
{
   char k[9];
   int i=1;
   while(a[i]!='\0')
   {
      k[i-1]=a[i];
      i++;
   }
   k[i-1]='\0';
   i=atoi(k);
   return i;
}
```

Code for Selective Repeat (Receiver.c)

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<time.h>
#include<stdlib.h>
#include<ctype.h>
#include<arpa/inet.h>
#define W 5
#define P1 50
```

```
#define P2 10

char a[10];
char b[10];
void alpha9(int);
void alp(int);

int main()
{
   struct sockaddr_in ser,cli;
   int s,n,sock,i,j,c=1,f;
   unsigned int s1;
   s=socket(AF_INET,SOCK_STREAM,0);
   ser.sin_family=AF_INET;
   ser.sin_port=6500;
   ser.sin_addr.s_addr=inet_addr("127.0.0.1");
   bind(s,(struct sockaddr *) &ser, sizeof(ser));
   listen(s,1);
   n=sizeof(cli);
   sock=accept(s,(struct sockaddr *)&cli, &n);
   printf("\nTCP Connection Established.\n");
   s1=(unsigned int) time(NULL);
   srand(s1);
   strcpy(b,"Time Out ");
   recv(sock,a,sizeof(a),0);
   f=atoi(a);
   while(1)
   {
      for(i=0;i<W;i++)
      {
         recv(sock,a,sizeof(a),0);
         if(strcmp(a,b)==0)
         {
            break;
         }
      }
      i=0;
      while(i<W)
```

```
      {
         L:
            j=rand()%P1;
         if(j<P2)
         {
            alp(c);
            send(sock,b,sizeof(b),0);
            goto L;
         }
         else
         {
            alpha9(c);
            if(c<=f)
            {
               printf("\nFrame %s Received ",a);
               send(sock,a,sizeof(a),0);
            }
            else
            {
               break;
            }
            c++;
         }
         if(c>f)
         {
            break;
         }
         i++;
      }
   }
   close(sock);
   close(s);
   return 0;
}

void alpha9(int z)
{
   int k,i=0,j,g;
```

```
   k=z;
   while(k>0)
   {
      i++;
      k=k/10;
   }
   g=i;
   i--;
   while(z>0)
   {
      k=z%10;
      a[i]=k+48;
      i--;
      z=z/10;
   }
   a[g]='\0';
}

void alp(int z)
{
   int k,i=1,j,g;
   k=z;
   b[0]='N';
   while(k>0)
   {
      i++;
      k=k/10;
   }
   g=i;
   i--;
   while(z>0)
   {
      k=z%10;
      b[i]=k+48;
      i--;
      z=z/10;
   }
   b[g]='\0';
```

}

4. Lab Exercise

Exercise: Modify the program of Go Back N and Selective Repeat to introduce a packet loss of 20% with window size as 5. If the sender needs to send 50 frames, how many total frames (including retransmissions) will be sent in GBN and SR when there is a lossy channel?