Sliding window protocols

Sliding window protocols are data link layer protocols for reliable and sequential delivery of data frames. The sliding window is also used in Transmission Control Protocol.

In this protocol, multiple frames can be sent by a sender at a time before receiving an acknowledgment from the receiver. The term sliding window refers to the imaginary boxes to hold frames. Sliding window method is also known as windowing.

Working Principle

In these protocols, the sender has a buffer called the sending window and the receiver has buffer called the receiving window.

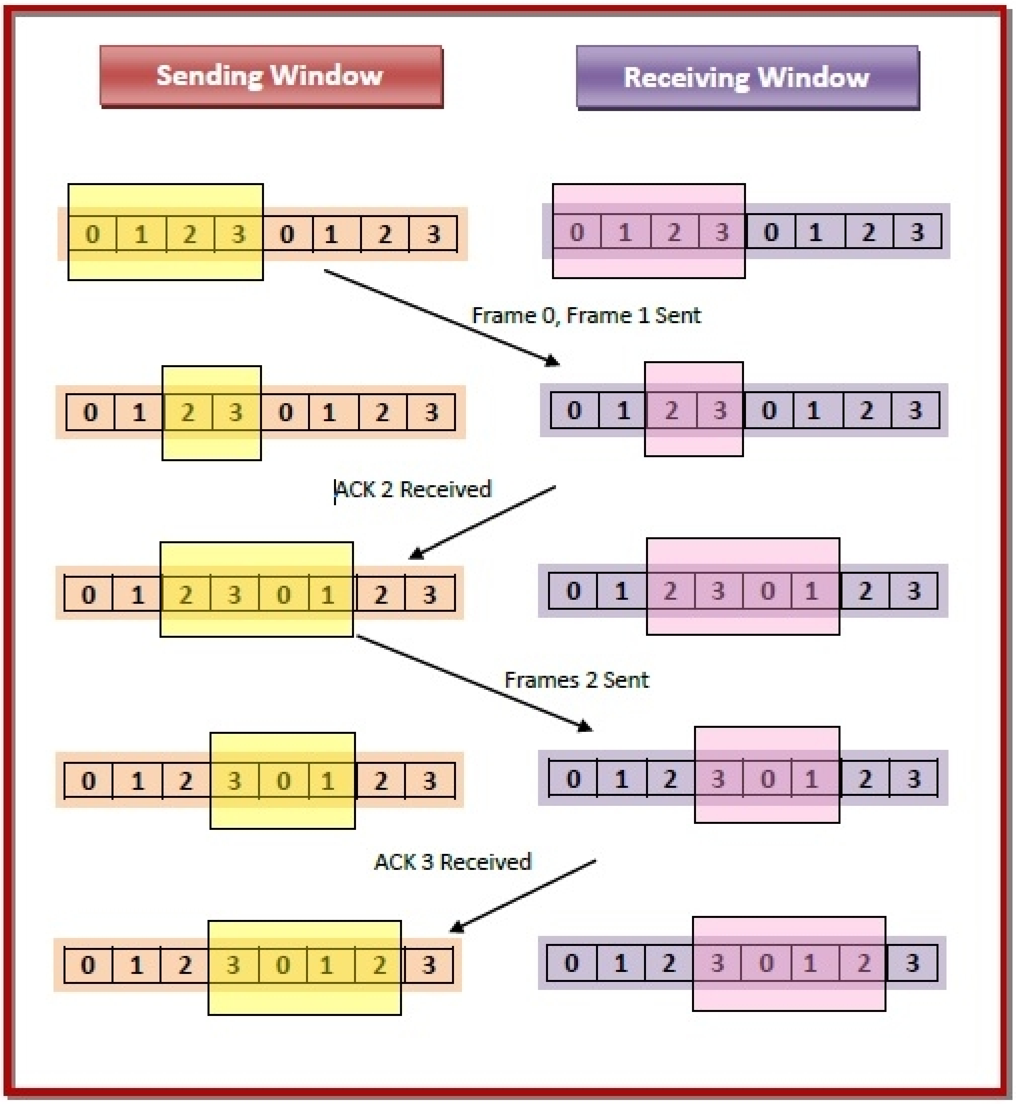
The size of the sending window determines the sequence number of the outbound frames. If the sequence number of the frames is an ***n***-bit field, then the range of sequence numbers that can be assigned is 0 to $2^n - 1$. Consequently, the size of the sending window is $2^n - 1$. Thus in order to accommodate a sending window size of $2^n - 1$, a ***n***-bit sequence number is chosen.

The sequence numbers are numbered as modulo-***n***. For example, if the sending window size is 4, then the sequence numbers will be 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, and so on. The number of bits in the sequence number is 2 to generate the binary sequence 00, 01, 10, 11.

The size of the receiving window is the maximum number of frames that the receiver can accept at a time. It determines the maximum number of frames that the sender can send before receiving acknowledgment.

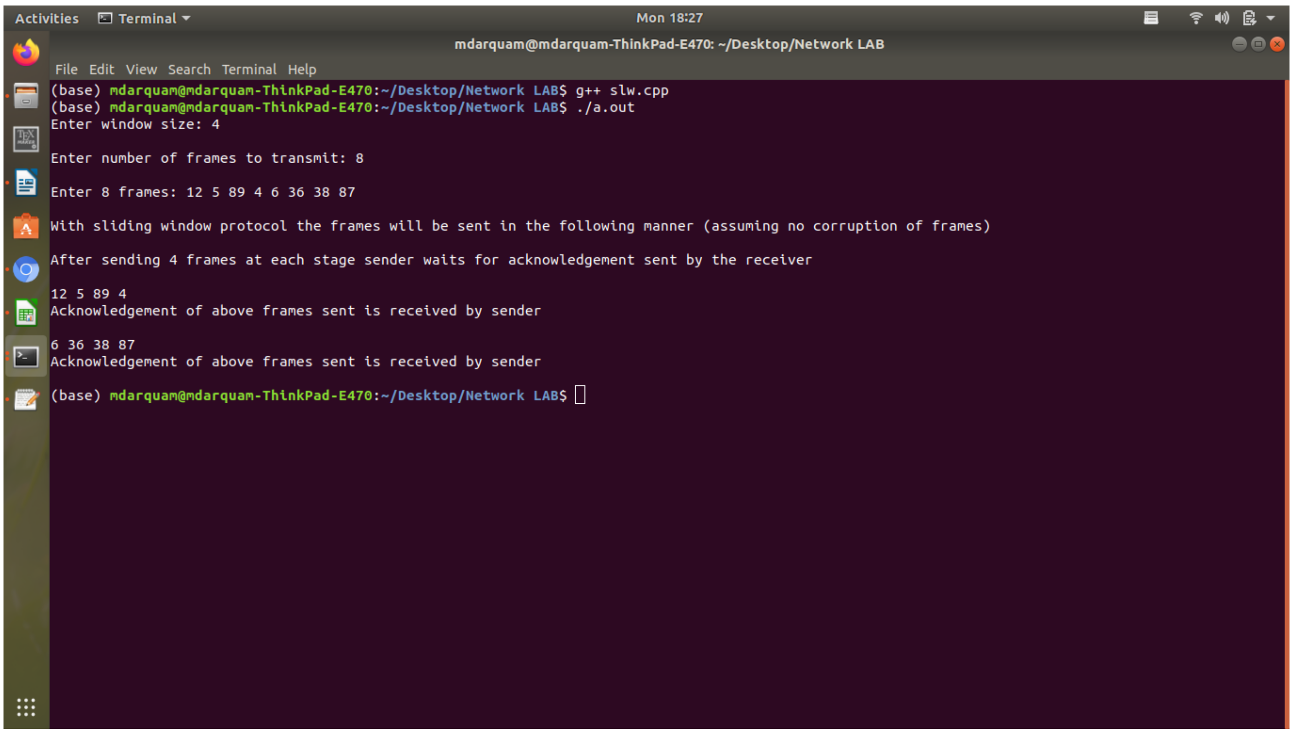
Example

Suppose that we have sender window and receiver window each of size 4. So the sequence numbering of both the windows will be 0,1,2,3,0,1,2 and so on. The following diagram shows the positions of the windows after sending the frames and receiving acknowledgments.


Sending Window
Receiving Window
Frame 0, Frame 1 Sent
ACK 2 Received
Frames 2 Sent
ACK 3 Received

Write a program to implement Sliding window protocol

```
#include<iostream>

using namespace std;

int main()
{
   int w,i,f,frames[50];

   cout<<"Enter window size: ";
   cin>>w;

   cout<<"\nEnter number of frames to transmit: ";
   cin>>f;

   cout<<"\nEnter "<<f<<" frames: ";

   for(i=1;i<=f;i++)
      cin>>frames[i];

   cout<<"\nWith sliding window protocol the frames will be sent in the following manner
(assuming no corruption of frames)\n\n";
   cout<<"After sending "<<w<<" frames at each stage sender waits for acknowledgement sent by
the receiver\n\n";

   for(i=1;i<=f;i++)
   {
      if(i%w==0)
      {
         cout<<frames[i]<<"\n";
         cout<<"Acknowledgement of above frames sent is received by sender\n\n";
      }
      else
         cout<<frames[i]<<" ";
   }

   if(f%w!=0)
      cout<<"\nAcknowledgement of above frames sent is received by sender\n";

   return 0;
}
```



Activities
Terminal
Mon 18:27
mdarquam@mdarquam-ThinkPad-E470: ~/Desktop/Network LAB
File Edit View Search Terminal Help

```
(base) mdarquam@mdarquam-ThinkPad-E470:~/Desktop/Network LAB$ g++ slw.cpp
(base) mdarquam@mdarquam-ThinkPad-E470:~/Desktop/Network LAB$ ./a.out
Enter window size: 4

Enter number of frames to transmit: 8

Enter 8 frames: 12 5 89 4 6 36 38 87

With sliding window protocol the frames will be sent in the following manner (assuming no corruption of frames)

After sending 4 frames at each stage sender waits for acknowledgement sent by the receiver

12 5 89 4
Acknowledgement of above frames sent is received by sender

6 36 38 87
Acknowledgement of above frames sent is received by sender

(base) mdarquam@mdarquam-ThinkPad-E470:~/Desktop/Network LAB$ 
```

Stop and Wait

Characteristics:

•Used in Connection-oriented communication.

•It offers error and flow control

•It is used in Data Link and Transport Layers

•Stop and Wait ARQ mainly implements Sliding Window Protocol concept with Window Size 1

Useful Terms:

•Propagation Delay: Amount of time taken by a packet to make a physical journey from one router to another router.

•Propagation Delay = (Distance between routers) / (Velocity of propagation)

•RoundTripTime (RTT) = 2* Propagation Delay

•TimeOut (TO) = 2* RTT

•Time To Live (TTL) = 2* TimeOut. (Maximum TTL is 180 seconds)

<u>Simple Stop and Wait</u>

Sender:

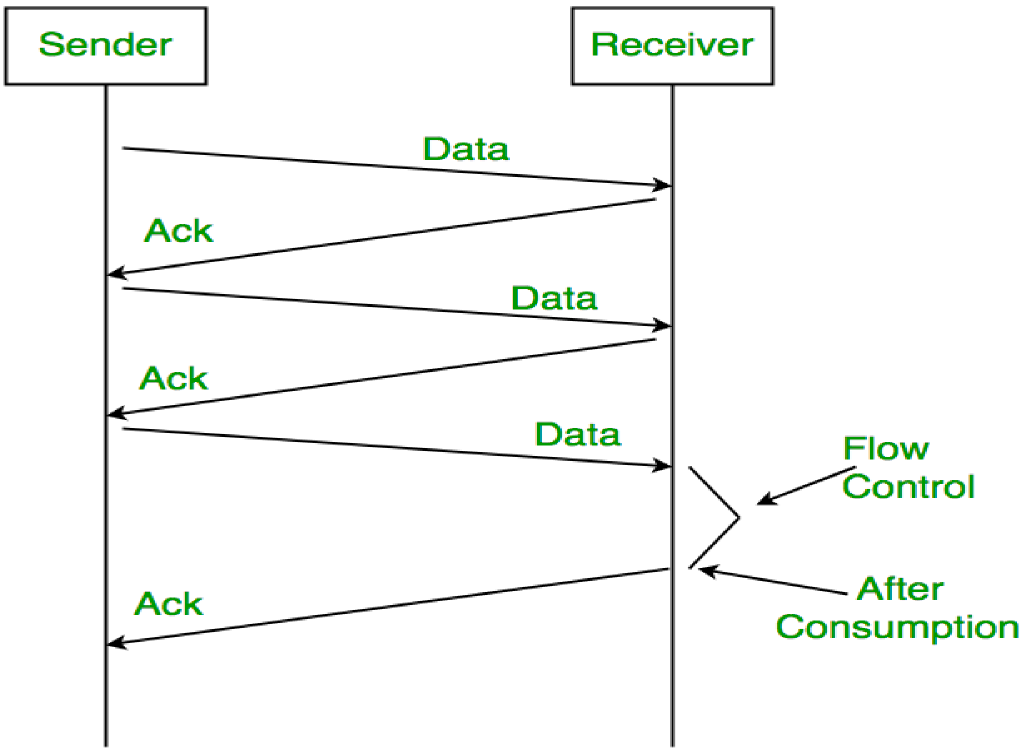
Rule 1) Send one data packet at a time.

Rule 2) Send next packet only after receiving acknowledgement for previous.

Receiver:

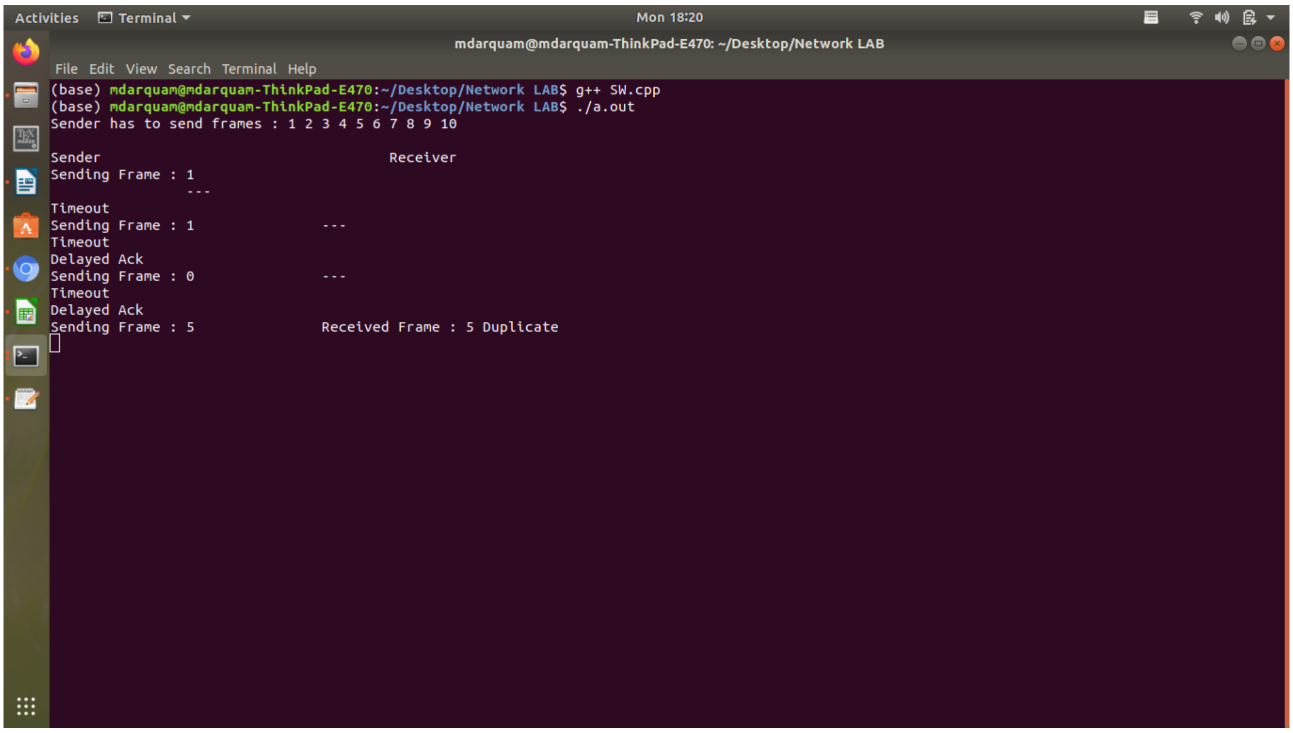
Rule 1) Send acknowledgement after receiving and consuming of data packet.

Rule 2) After consuming packet acknowledgement need to be sent (Flow Control)



Write a program to implement Stop and Wait protocol

```
#include<iostream>

#include <time.h>

#include <cstdlib>
#include<ctime>
#include <unistd.h>

using namespace std;

class timer {
private:
 unsigned long begTime;
public:
 void start() {
 begTime = clock();
 }
unsigned long elapsedTime() {
 return ((unsigned long) clock() - begTime) / CLOCKS_PER_SEC;
 }
 bool isTimeout(unsigned long seconds) {
 return seconds >= elapsedTime();
 }
};

int main()
{
int frames[] = {1,2,3,4,5,6,7,8,9,10};
unsigned long seconds = 5;
srand(time(NULL));
timer t;

cout<<"Sender has to send frames : ";

for(int i=0;i<10;i++)
cout<<frames[i]<<" ";
cout<<endl;
int count = 0;
bool delay = false;
cout<<endl<<"Sender\t\t\t\t\tReceiver"<<endl;
do
{
 bool timeout = false;
 cout<<"Sending Frame : "<<frames[count];
 cout.flush();
 cout<<"\t\t";
 t.start();
 if(rand()%2)
 {
 int to = 24600 + rand()%(64000 - 24600)  + 1;
 for(int i=0;i<64000;i++)
 for(int j=0;j<to;j++) {}
 }
 if(t.elapsedTime() <= seconds)
 {
 cout<<"Received Frame : "<<frames[count]<<" ";
 if(delay)
 {
 cout<<"Duplicate";
 delay = false;
 }
 cout<<endl;
 count++;
 }
 else
 {
 cout<<"---"<<endl;
 cout<<"Timeout"<<endl;
 timeout = true;
 }
 t.start();
 if(rand()%2 || !timeout)
 {
 int to = 24600 + rand()%(64000 - 24600)  + 1;
 for(int i=0;i<64000;i++)
 for(int j=0;j<to;j++) {}
 if(t.elapsedTime() > seconds )
 {
 cout<<"Delayed Ack"<<endl;
 count--;
 delay = true;
 }
 else if(!timeout)
 cout<<"Acknowledgement : "<<frames[count]-1<<endl;
 }
}while(count!=10);
return 0;
}
```


Activities
Terminal
Mon 18:20
mdarquam@mdarquam-ThinkPad-E470: ~/Desktop/Network LAB
File Edit View Search Terminal Help

```
(base) mdarquam@mdarquam-ThinkPad-E470:~/Desktop/Network LAB$ g++ SW.cpp
(base) mdarquam@mdarquam-ThinkPad-E470:~/Desktop/Network LAB$ ./a.out
Sender has to send frames : 1 2 3 4 5 6 7 8 9 10

Sender                                  Receiver
Sending Frame : 1
                ---
Timeout
Sending Frame : 1               ---
Timeout
Delayed Ack
Sending Frame : 0               ---
Timeout
Delayed Ack
Sending Frame : 5               Received Frame : 5 Duplicate
```