

**Title: Write a program to simulate Go back N Sliding Window Protocol in peer-to-peer mode.**

## **Sliding Window Protocol**

The sliding window is a technique for sending multiple frames at a time. It controls the data packets between the two devices where reliable and gradual delivery of data frames is needed. It is also used in TCP (Transmission Control Protocol).

In this technique, each frame has sent from the sequence number. The sequence numbers are used to find the missing data in the receiver end. The purpose of the sliding window technique is to avoid duplicate data, so it uses the sequence number.

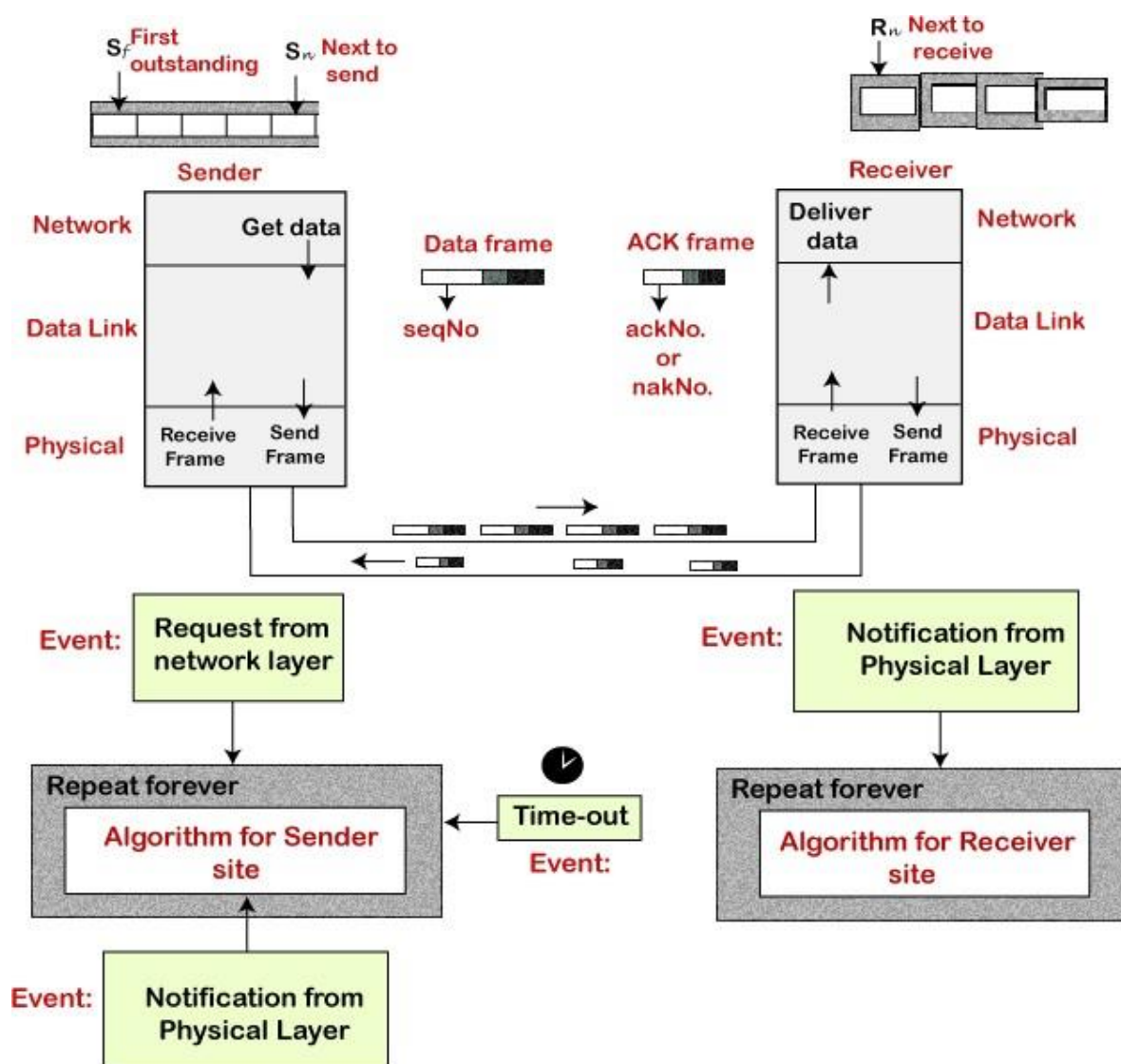
Types of Sliding Window Protocol Sliding window protocol has two types:

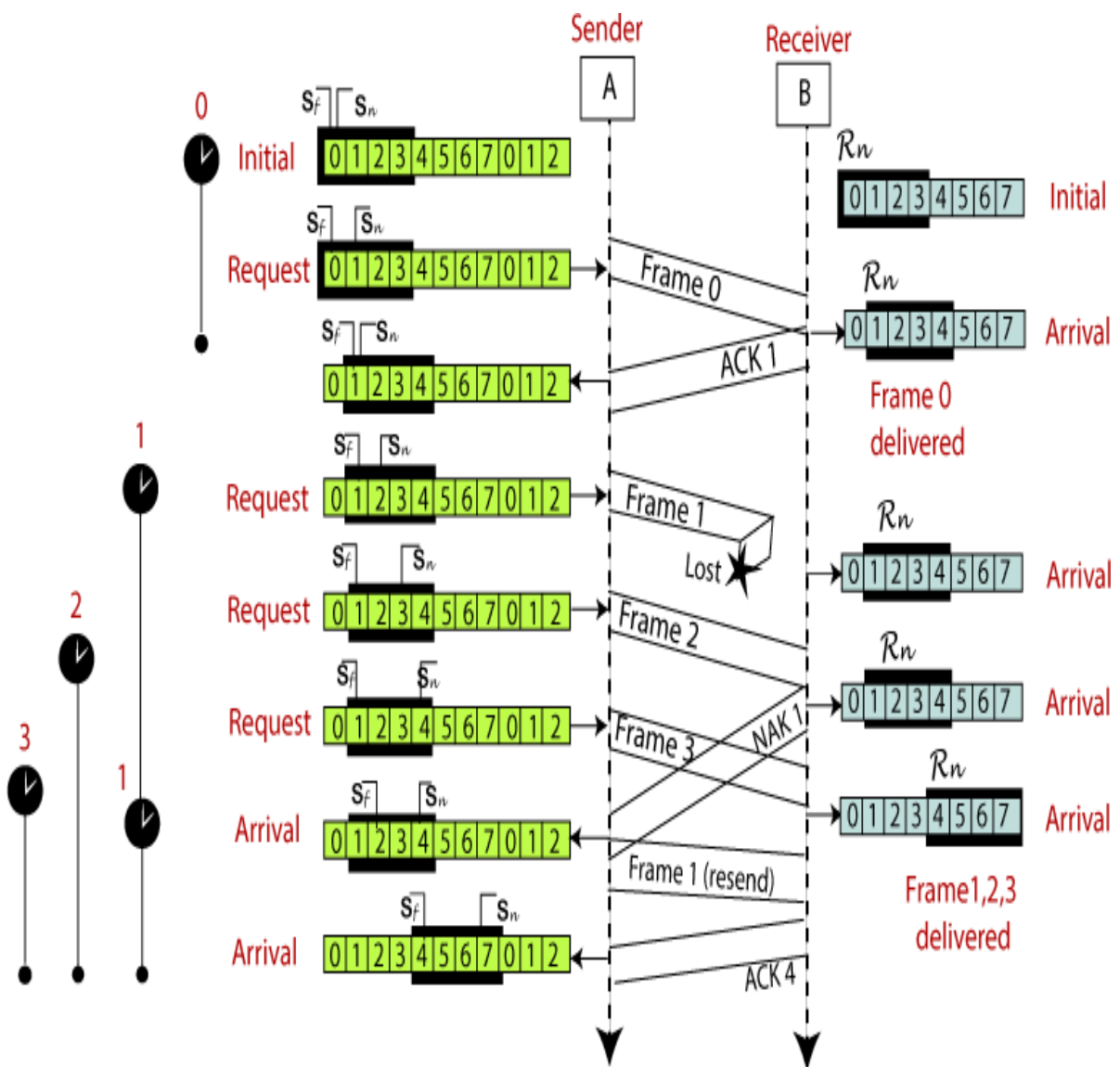
- Go-Back-N ARQ
- Selective Repeat ARQ

## Selective Repeat ARQ

Selective Repeat ARQ is also known as the Selective Repeat Automatic Repeat Request. It is a data link layer protocol that uses a sliding window method. The Go-back-N ARQ protocol works well if it has fewer errors. But if there is a lot of error in the frame, lots of bandwidth loss in sending the frames again. So, we use the Selective Repeat ARQ protocol. In this protocol, the size of the sender window is always equal to the size of the receiver window. The size of the sliding window is always greater than 1.

If the receiver receives a corrupt frame, it does not directly discard it. It sends a negative acknowledgment to the sender. The sender sends that frame again as soon as on the receiving negative acknowledgment. There is no waiting for any time-out to send that frame. The design of the Selective Repeat ARQ protocol is shown below.





## Code:-

```
import java.util.Scanner;
import java.util.HashMap;
import java.util.Map;

public class SelectiveRepeatSimulation {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of frames to be sent: ");
        int numFrames = scanner.nextInt();

        System.out.print("Enter the size of the sender's window: ");
        int senderWindowSize = scanner.nextInt();

        System.out.println("\nSimulating Selective Repeat Sliding Window
Protocol...");

        Sender sender = new Sender(numFrames, senderWindowSize);
        Receiver receiver = new Receiver(numFrames);

        while (!sender.isTransmissionComplete() || !receiver.isReceivedAllFrames())
        {
            sender.sendFrames(receiver);
            receiver.receiveFrames(sender);
        }

        System.out.println("\nTransmission complete. All frames have been sent and
acknowledged.");
    }
}

class Sender {
    private int[] frames;
    private int windowSize;
    private int nextFrameToSend;
    private Map<Integer, Boolean> acknowledged;

    public Sender(int numFrames, int windowSize) {
        this.frames = new int[numFrames];
        for (int i = 0; i < numFrames; i++) {
            this.frames[i] = i + 1;
        }
        this.windowSize = windowSize;
        this.nextFrameToSend = 0;
        this.acknowledged = new HashMap<>();
    }

    public int getWindowSize() {
        return windowSize;
    }

    public int getNumFrames() {
```

```

        return frames.length;
    }

    public int getFrame(int index) {
        return frames[index];
    }

    public boolean isTransmissionComplete() {
        return nextFrameToSend == frames.length;
    }

    public void sendFrames(Receiver receiver) {
        System.out.println("\nSender transmitting frames " + nextFrameToSend + " to
" +
            Math.min(nextFrameToSend + windowSize - 1, frames.length - 1));

        for (int i = nextFrameToSend; i < Math.min(nextFrameToSend + windowSize,
frames.length); i++) {
            if (!acknowledged.containsKey(i)) {
                System.out.println("Sending Frame " + frames[i]);
            }
        }

        int lastAcknowledgedFrame = receiver.getLastAcknowledgedFrame();

        for (int i = nextFrameToSend; i <= lastAcknowledgedFrame; i++) {
            acknowledged.put(i, true);
        }

        nextFrameToSend = lastAcknowledgedFrame + 1;
    }

    public int getNextFrameToSend() {
        return nextFrameToSend;
    }
}

class Receiver {
    private int[] receivedFrames;
    private int lastAcknowledgedFrame;

    public Receiver(int numFrames) {
        this.receivedFrames = new int[numFrames];
        this.lastAcknowledgedFrame = -1;
    }

    public boolean isReceivedAllFrames() {
        for (int frame : receivedFrames) {
            if (frame == 0) {
                return false;
            }
        }
    }
}

```

```
        return true;
    }

    public void receiveFrames(Sender sender) {
        System.out.println("\nReceiver waiting for frames...");

        int windowStart = sender.getNextFrameToSend();
        int windowEnd = Math.min(windowStart + sender.getWindowSize() - 1,
sender.getNumFrames() - 1);

        for (int i = windowStart; i <= windowEnd; i++) {
            if (receivedFrames[i] == 0) {
                int frame = sender.getFrame(i);
                System.out.println("Receiver received Frame " + frame);
                receivedFrames[i] = frame;
                lastAcknowledgedFrame = i;
            }
        }

        System.out.println("Sending acknowledgment for Frame " +
(lastAcknowledgedFrame + 1));
    }

    public int getLastAcknowledgedFrame() {
        return lastAcknowledgedFrame;
    }
}
```

## Output:

```
PS C:\Rushikesh_Tanpure_AIDS_B_75> cd "c:\Rushikesh_Tanpure_AIDS_B_75\" ; if ($?) { javac SelectiveRepeatSimulation.java
} ; if ($?) { java SelectiveRepeatSimulation }
Enter the number of frames to be sent: 10
Enter the size of the sender's window: 4

Simulating Selective Repeat Sliding Window Protocol...

Sender transmitting frames 0 to 3
Sending Frame 1
Sending Frame 2
Sending Frame 3
Sending Frame 4

Receiver waiting for frames...
Receiver received Frame 1
Receiver received Frame 2
Receiver received Frame 3
Receiver received Frame 4
Sending acknowledgment for Frame 4

Sender transmitting frames 0 to 3
Sending Frame 1
Sending Frame 2
Sending Frame 3
Sending Frame 4
```

Sender transmitting frames 0 to 3

Sending Frame 1

Sending Frame 2

Sending Frame 3

Sending Frame 4

Receiver waiting for frames...

Receiver received Frame 5

Receiver received Frame 6

Receiver received Frame 7

Receiver received Frame 8

Sending acknowledgment for Frame 8

Sender transmitting frames 4 to 7

Sending Frame 5

Sending Frame 6

Sending Frame 7

Sending Frame 8

Receiver waiting for frames...

Receiver received Frame 9

Receiver received Frame 10

Sending acknowledgment for Frame 10

Sender transmitting frames 8 to 9

Sending Frame 9

Sending Frame 10

Receiver waiting for frames...

Sending acknowledgment for Frame 10

Transmission complete. All frames have been sent and acknowledged.