Batch: C2	Roll No.: 110
Experiment / assignment / tutorial No._____	
Grade: AA / AB / BB / BC / CC / CD /DD	

Signature of the Staff In-charge with date

Experiment No.:5

TITLE: Flow control Mechanism: Go-Back- N ARQ Sliding Window Protocol using Socket programming

AIM: Implementation of Flow Control Mechanism: Stop and Wait ARQ and Go-Back-N Sliding Window Protocol ARQ using sockets.

Expected Outcome of Experiment:
CO:

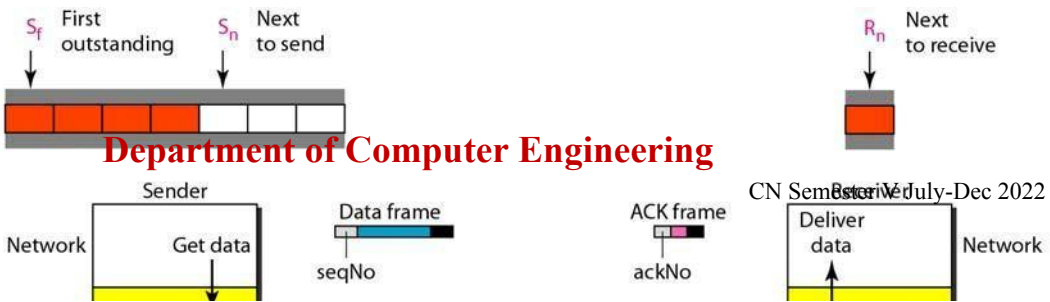
Books/ Journals/ Websites referred:

1. A. S. Tanenbaum, “Computer Networks”, Pearson Education, Fourth Edition
2. B. A. Forouzan, “Data Communications and Networking”, TMH, Fourth Edition

Pre-Lab/ Prior Concepts:
Java Socket Programming, Flow Control, Go-Back-Stop and Wait

New Concepts to be learned: Window Flow Control

Design of Go-Back-N ARQ



1. Take data from user about how many bit windows is case of go back n and selective repeat.
2. Generate frames randomly and show the transmission
3. Generate the random number for the frame to be lost.
4. For Go – Back – N transmit all the frames after that number till max number
5. For Selective repeat transmit the selected frame which is not received by the receiver.

IMPLEMENTATION: (printout of code)

```
# first of all import the socket library
import socket


# next create a socket object
s = socket.socket()
print ("Socket successfully created")


# reserve a port on your computer in our
# case it is 12345 but it can be anything
port = 12345


# Next bind to the port
# we have not typed any ip in the ip field
# instead we have inputted an empty string
# this makes the server listen to requests
# coming from other computers on the network
s.bind(('', port))
print ("socket binded to %s" %(port))
```

```
# put the socket into listening mode
s.listen(5)
print ("socket is listening")

# a forever loop until we interrupt it or
# an error occurs

while True:
#sliding window of 7


# Establish connection with client.
c, addr = s.accept()
print ('Got connection from', addr )
recieved = [0,1,2,3,4,5,6,7] # sliding window
counter = 0
time = 0
for i in range(100):
print("time = ", time)
print("counter = ", counter)
time +=1
if(time % 7 > counter + 6):
time = counter # go back
print("going back")
c.send(str(time % 7).encode()) #send to client
val = int(c.recv(1024).decode())
if(val>=counter):
counter = val +1
counter = counter % 7
```

```
# proceed
print("val = ", val)

# Close the connection with the client
c.close()

# Breaking once connection closed
break

# Import socket module
import socket

# Create a socket object
s = socket.socket()

# Define the port on which you want to connect
port = 12345
recieved = []

# connect to the server on local computer
s.connect(('127.0.0.1', port))
frame = 0
while True:
# receive data from the server and decoding to get the
string.

val = s.recv(1024).decode()
print(val)
val = int(val)
recieved.append(val)
```

```
if(frame == val):
print ("recieved")
frame = frame +1
frame = frame % 7
s.send(str(val).encode())  # ack
else:
s.send("-1".encode())  # nack
# close the connection


s.close()
```

```
time =  96
counter =  6
val =  6
time =  97
counter =  0
val =  0
time =  98
counter =  1
val =  1
time =  99
counter =  2
val =  2
```

```
recieved
4
recieved
5
recieved
6
recieved
0
recieved
1
recieved
2
recieved
```

CONCLUSION:

Thus we have implemented go back N protocol in python using socket programming. This protocol is useful for sending the data from two processes over a network. This is more efficient than the stop-and-wait protocol. In this experiment we have learnt how to do socket programming in python and we have implemented the go back n arq protocol.

Post Lab Questions

1. **Compare Go-Back-N and Stop and Wait.**
   Go-Back-N (GBN):
   a. Operation: In the Go-Back-N protocol, multiple frames are sent in sequence without waiting for acknowledgment. The sender continues sending frames until it reaches its sender's window size. The receiver acknowledges the correctly received frames but can only handle frames in order. If any frame is lost or corrupted, the receiver discards it along with all subsequent frames until the error is detected. The sender, upon receiving a NAK (negative acknowledgment) or timing out, retransmits all unacknowledged frames from the last acknowledged one.
   b. Efficiency: GBN is more efficient than Stop-and-Wait as it allows for pipelining of frames, which means the sender can have multiple unacknowledged frames in transit.

Stop and wait

c. Operation: In the Stop-and-Wait protocol, the sender transmits one frame and then waits for an acknowledgment from the receiver. The receiver acknowledges the frame if it's received correctly; otherwise, it requests the sender to retransmit the frame. Only when an acknowledgment is received does the sender proceed to the next frame.
d. Efficiency: Stop-and-Wait is less efficient compared to GBN because it doesn't take full advantage of the available bandwidth as it operates in a start-stop manner.

2. **What is Flow Control and why it is necessary?**
Flow control is a fundamental concept in data communication that ensures that data is transmitted between sender and receiver at a rate that both can handle. It is necessary for several reasons:

Preventing Data Overflow: Flow control prevents the sender from overwhelming the receiver with data. Without flow control, the sender could send data at a rate faster than the receiver can process it, leading to data loss and congestion.

Resource Management: It helps in managing resources efficiently. By controlling the flow of data, it prevents the waste of network bandwidth and memory resources.

Error Prevention: Flow control mechanisms can detect and prevent errors due to data loss or corruption. For example, if a receiver can't keep up with the sender, it can request a retransmission rather than accepting potentially corrupted or incomplete data.

Congestion Control: Flow control also plays a role in managing network congestion. By slowing down the rate of data transmission, it helps in avoiding network congestion and subsequent packet loss

3. The maximum window size for data transmission using the selective reject protocol with n-bit frame sequence numbers is
a) 2n b) 2n-1 c) 2n-1 d)2n-2
C

Date :21 oct 2023 **Signature of Faculty In-charge**