ALGORITHM

1. Create Nodes
2. Create Link
3. Join link to node
4. Install TCP/IP protocol stack
5. Set base address
6. Assign address
7. Start server at port 9. Install server at node 1.
8. Start and stop the server
9. Set the Client attributes.
10. Install client at node 0
11. Start and stop the client
12. Run the client-server simulation.

D D M M Y Y Y Y

PART-B

1. Simulate peer-to-peer communication between client and server using Point-to-Point protocol. Apply NetAnim software to demonstrate the scenario graphically. Analyze packet parameters by creating files using Ascii trace metrics.

THEORY:

A socket is a mechanism used to realise client-server or peer-to-peer communication in a network. It is a protocol independent method to create a connection between two processes residing in the same computer or different computer in a network.

In ns3, real data is sent between two processes to analyse the network behaviour. Socket programming is used in ns3 to set up inter-process communication between two processes such that actual data can be sent between them. In order to set up communication between two processes, there should by 2 sockets, one at sender's side and the other at receiver's side.

Often, one behaves like server socket and the other behaves like client socket. The client tries to connect to the server and obtain data. The address of an Internet socket consists of IP address and the port number, thus involving the transport layer and network layer.

D D M M Y Y Y Y

PROGRAM:

```
#include "ns3/netanim-module.h"
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
using namespace ns3;
int main (int argc, char *argv[])
{
    Time::SetResolution (Time::NS);
    NodeContainer nodes;
    nodes.Create (2);
    PointToPointHelper PointToPoint;
    PointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
    PointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
    NetDeviceContainer devices;
    devices = PointToPoint.Install (nodes);
    InternetStackHelper stack;
    stack.Install (nodes);
    Ipv4AddressHelper address;
    address.SetBase ("10.1.1.0", "255.255.255.0");
    Ipv4InterfaceContainer interfaces = address.Assign (devices);
    UdpEchoServerHelper echoServer (9);
    ApplicationContainer serveApps = echoServer.Install (nodes.Get(1));
    ServeApps.Start (Seconds (1.0));
    serveApps.Stop (Seconds (10.0));
    UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
```

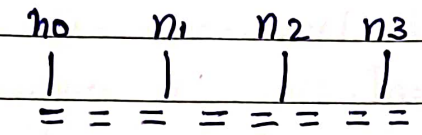
D D M M Y Y Y Y

```
echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));
AnimationInterface anim ("first.xml");
Simulator::Run ();
Simulator::Destroy ();
return 0;
}
```

D D M M Y Y Y Y

2. Create UDP client and server application on CSMA connection.

THEORY:



The ns-3 CSMA models a simple network in the spirit of Ethernet. A real Ethernet uses CSMA/CD schema with exponentially increasing backoff to contend for the shared transmission medium. The ns3 CSMA device and channels models are only a subset of this. In this program, a two way connection is established between nodes through Ethernet.

PROGRAM:

```
#include <fstream>
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/netanim-module.h"
#include "ns3/csma-module.h"
using namespace ns3;
int main (int argc, char *argv[])
{
    Address serverAddress;
    NodeContainer n;
    n.Create (4);
    InternetStackHelper internet;
    internet.Install (n);
```

D D M M Y Y Y Y

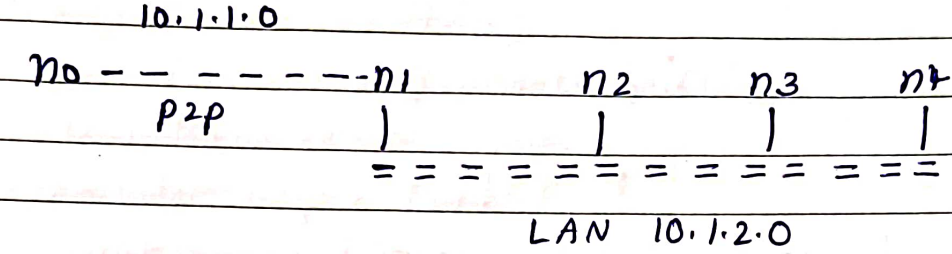
```
CsmaHelper csma;
  csma.SetChannelAttribute ("DataRate", DataRateValue (DataRate(5000000)));
  csma.SetChannelAttribute ("Delay", TimeValue (MilliSeconds(2)));
  csma.SetDeviceAttribute ("Mtu", UintegerValue (1400));
NetDevicContainer d = csma.Install (n);
Ipv4AddressHelper ipv4;
  ipv4.SetBase ("10.1.1.0", "255.255.255.0");
  Ipv4InterfaceContainer i = ipv4.Assign (d);
  serveAddress = Address (i.GetAddress (1));

  uint16_t port = 9;
  UdpEchoServerHelper server (port);
  ApplicationContainer apps = server.Install (n.Get (1));
  apps.Start (Seconds (1.0));
  apps.Stop (Seconds (10.0));

  uint32_t PpacketSize = 1024;
  uint32_t maxPacketCount = 1;
  UdpEchoClientHelper client (serverAddress, port);
  client.SetAttribute ("MaxPackets", UintegerValue (maxPacketCount));
  client.SetAttribute ("Interval", TimeValue (interPacketintoval));
  client.SetAttribute ("PacketSize", UintegerValue (packetSize));
  apps= client.Install (n.Get (0));
  apps.Start (Seconds (2.0));
  apps.Stop (Seconds (10.0));
  AnimationInterface anim ("second.xml");
   Simulator::Run ();
   Simulator::Destroy ();
}
```

D D M M Y Y Y Y

3. Create UDP echo client and server Application on combination of P2P and CSMA connection



THEORY:

The program is an extension of UDP echo client-server P2P program by hanging a bus network off the rightside. This is the default network topology since one can vary the no. of nodes created on LAN. If you set nCSMA to one. There will be a total of two nodes on LAN (CSMA channel()) one required node and one extra node.

PROGRAM:

```
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/netanim-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/csma-module.h"
#include "ns3/network-module.h"
#include "ns3/wifi-module.h"
#include "ns3/mobility-module.h"
using namespace ns3;
int main (int argc, char *argv[])
{
    uint32_t nCsma = 3;
```

D D M M Y Y Y Y

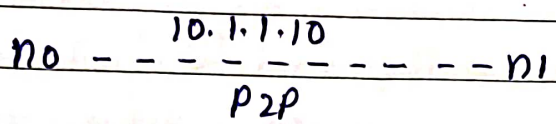
```
NodeContainer p2pNodes;
  p2pNodes.Create (2);
NodeContainer csmaNodes;
  csmaNodes.Add (p2pNodes.Get(1));
  csmaNodes.Create (nCsma);
PointToPointHelper PointToPoint;
  PointToPoint.SetDeviceAttribute("DataRate", StringValue ("5Mbps"));
  PointToPoint.SetChannelAttribute("Delay", StringValue ("2ms"));
NetDeviceContainer p2pDevices;
  p2pDevices = pointToPoint.Install (p2pNodes);
CsmaHelper csma;
  csma.SetChamelAttributes("DataRati", StringValue ("100Mbps"));
  csma.SetChannelAttributes("Delay", TimeValue (NanoSeconds (6560)));
NetDeviceContainer csmaDevices;
  csmaDevices = csma.Install(csmaNodes);
InternetStackHelper stack;
  stack.Install (p2pNodes.Get(0));
  stack.Install (csmaNodes);
Ipv4AddressHelper address;
  address.SetBase ("10.1.1.0","255.255.255.0");
  Ipv4InterfaceContainer p2pInterfaces;
  p2pInterfaces = address.Assign(p2pDevices);
  address.SetBase ("10.1.2.0","255.255.255.0");
  Ipv4InterfaceContainer csmaInterfaces;
  csmaInterfaces = address.Assign (csmaDevices);
  UdpEchoServerHelper echoServer (9);
ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get(nCsma));
  serverApps.Start (Seconds (1.0));
  serverApps.Stop (Seconds (10.0));
```

D D M M Y Y Y Y

```
UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (nCsma), 9);
    echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
    echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
    echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get (0));
    clientApps.Start (Seconds (2.0));
    clientApps.Stop (Seconds (10.0));
Ipv4GlobalRoutingHelper :: PopulateRoutingTables ();
 pointToPoint.EnablePcapAll ("second");
  csma.EnablePcap ("second", csmaDevices.Get (1), true);
AnimationInterface anim ("third.xml");
Simulator :: Run ();
  Simulator :: Destroy ();
return 0;
}
```

D D M M Y Y Y Y

4. Create TCP source and sink application on P2P connection



THEORY:

The point-to-point TCP source and sink application is similar to that of point-to-point UDP server and Client application. The TCP protocol follows three-way handshake communication which is reflected on the output.

Here, the servers of UDP is replaced by source of TCP and the client of UDP is replaced by sink of TCP.
The program follows point-to-point communication between the nodes n0 and n1.

PROGRAM:

```
#include <string>
#include <fstream>
#include "ns3/core
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/netanim-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/network-module.h"
#include "ns3/packet-sink.h"
int using namespace ns3;
int main (int argc, char *argv[])
{   uint32_t maxBytes =0;
    NodeContainer nodes;
      nodes.Create (2);
```

D D M M Y Y Y Y

```
PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("500kbps"));
  pointToPoint.SetChannelAttribute ("Delay", StringValue ("5ms"));
NetDeviceContainer devices;
  devices = pointToPoint.Install (nodes);
InternetStackHelper internet;
  internet.Install (nodes);
Ipv4AddressHelper ipv4;
  ipv4.SetBase ("10.1.1.0","255.255.255.0");
  Ipv4InterfaceContainer i = ipv4.Assign (devices);
uint16_t port = 9;
BulkSendHelper source ("ns3::TcpSocketFactory", InetSocketAddress
                                    (i.GetAddress (1), port));
  source.SetAttribute ("MaxBytes", UintegerValue (maxBytes));
ApplicationContainer sourceApps = source.Install (nodes.Get (0));
  sourceApps.Start (Seconds (0.0));
  sourceApps.Stop (Seconds (10.0));
PacketSinkHelper sink ("ns3::TcpSocketFactory", InetSocketAddress
                                    (Ipv4Address::GetAny (), port));
ApplicationContainer sinkApps = sink.Install (nodes.Get (1));
  sinkApps.Start (Seconds (0.0));
  sinkApps.Stop (Seconds (10.0));
Simulator::Stop (Seconds (10.0));
AnimationInterface anim ("fourth.xml");
anim.EnablePacketMetadata (true);
  Simulator::Run ();
  Simulator::Destroy ();
}
```