



R N S INSTITUTE OF TECHNOLOGY

UTTARAHALLI ROAD, CHANNASANDRA, BANGALORE -560 098



Department of Information Science & Engineering

Computer Networks Laboratory Manual V-Sem ISE 15CSL48

**Faculty-in-charge
Dr. Prakasha S and Mrs. Sudha V**

R N S INSTITUTE OF TECHNOLOGY

UTTARAHALLI ROAD, CHANNASANDRA, BANGALORE -560 098

Department of Information Science and Engineering



VISION of the College

MISSION of the College

VISION of the Department

Fostering winning professionals of strong informative potentials.

MISSION of the Department

Imparting high quality education in the area of Information Science so as to graduate the students with good fundamentals, "Information System Integration", "Software Creation" capability & suitably train them to thrive in Industries, higher schools of learning and R & D centers with a comprehensive perspective.

PROGRAM EDUCATIONAL OBJECTIVES (PSOs) of the Department

- **PEO1:** Learn fundamentals of computers, applied knowledge of Information Science and continue to develop their technical competencies by problem solving using programming.
- **PEO2:** Ability to formulate problems, acquire the capability to develop system/application software in a scalable and robust manner, creation of back-end and front-end components for the distributed environment with various platforms, tools, and frameworks to provide cost effective solutions.
- **PEO3:** To build a capacity to investigate the necessities of the software Product, adapt to technological advancement, promote collaboration and interdisciplinary activities, Protecting Environment and developing Comprehensive leadership.
- **PEO4:** Enable students to be employed in IT industries and provide innovative solutions to real-world problems across different domains.
- **PEO5:** To prepare graduates with communication abilities, able to work in teams, professional ethics, socially responsible, entrepreneur and management, to achieve higher career goals, and pursue higher studies.

PROGRAM OUTCOMES (POs) of the Department**Engineering Graduates will be able to:**

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the

professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1 – Problem Solving Abilities:** Ability to demonstrate the fundamental and theoretical concepts, analyze the real-time problems and develop customized software solutions by applying the knowledge of mathematics and algorithmic techniques.
- **PSO2 – Applied Engineering Skills:** Enable creative thinking, Ability to apply standard practices and strategies, technical skills in software design, development, integration of systems and management for improving the security, reliability and survivability of the infrastructure.
- **PSO3 – General Expertise and Higher Learning** – Ability to exchange knowledge effectively, demonstrate the ability of team work, documentation skills, professional ethics, entrepreneurial skills and continuing higher education in the field of Information technology.

R N S INSTITUTE OF TECHNOLOGY

UTTARAHALLI ROAD, CHANNASANDRA, BANGALORE -560 098

Department of Information Science and Engineering Computer Networks Laboratory

Subject Code: 18CSL57

Hours/Week: 03

Total Hours: 42

I.A. Marks : 20

Exam Hours: 03

Exam Marks: 80

Course objectives

This course will enable students to

- Demonstrate operation of network and its management commands
- Simulate and demonstrate the performance of GSM and CDMA
- Implement data link layer and transport layer protocols

Course Outcomes

After studying this course, students will be able to:

CO1	Develop programs to demonstrate the communication for wired networks using NS-3 Simulator
CO2	Develop programs to demonstrate the communication for wireless networks using NS-3 Simulator.
CO3	Develop programs to related to Data integrity and Security.
CO4	Create programs to transfer data using TCP and UDP transport layer protocols.
CO5	Develop programs to find the shortest path using routing algorithms.
CO6	Develop programs to avoid congestion in the network.

CO mapping to PO/PSOs

CO / PO & PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
15CSL48.1	2	3	3							2				2	1
15CSL48.2		3	3	2						2				2	1
15CSL48.3		3	3	2						2				2	
15CSL48.4		3	3	2						2				2	
15CSL48.5		3	3	2						2				2	
15CSL48.6		3	3	2						2				2	

Computer Networks Laboratory

Subject Code: 18CSL57**Hours/Week: 03****Total Hours: 42****I.A. Marks : 20****Exam Hours: 03****Exam Marks: 80****List of Programs**

Sl. No	Name of Experiment	CO
	PART A	
1	Implement three nodes point-to-point network with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.	CO1
2	Implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.	CO1
3	Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source/destination.	CO1
4	Implement simple ESS and with transmitting nodes in wireless LAN by simulation and determine the performance with respect to transmission of packets.	CO2
5	Implement and study the performance of GSM on NS2/NS3 (using MAC layer) or equivalent environment.	CO2
6	Implement and study the performance of CDMA on NS2/NS3 (using Stack called Call net) or equivalent environment.	CO2

Sl. No	Name of Experiment	CO
	PART B	
7	Write a program for error detecting code using CRC-CCITT(16 bits)	CO3
8	Write a program to find the shortest path between vertices using bellman- ford algorithm.	CO5
9	Using TCP/IP sockets, write a client-server program to make the client send the File name and to make the server send back the contents of the requested file if present.	CO4
10	Write a program on datagram socket for client/server to display the messages on Client side, typed at the server side.	CO4
11	Write a program for simple RSA algorithm to encrypt and decrypt the data.	CO3
12	Write a program for congestion control using leaky bucket algorithm.	CO6

Computer Networks Laboratory Evaluation Rubrics

Subject Code: 18CSL57**Hours/Week: 03****Total Hours: 42****I.A. Marks : 20****Exam Hours: 03****Exam Marks: 80****Lab Write-up and EXECUTION rubrics (Max: 12 marks)**

		Above Average	Average	Below Average
a.	Understanding of problem and approach to solve. (4 Marks)	Able to analyze the given problem and efficiently implement using suitable assembly language instructions.(4)	Able to analyze the problem and moderate understanding of assembly language instruction. (2-3)	No program write-up. (0 -1)
b.	Execution and Testing (4 Marks)	Program is executed for varied inputs with valid results.(4)	Program is executed for some inputs. (2-3)	No Execution. (0-1)
c.	Results and Documentation (4 Marks)	Program and results obtained is well documented(4)	Program and results obtained is acceptably documented(2-3)	No Proper results and poor documentation. (0-1)

LAB Internal Assesment rubrics (Max: 8 marks)

		Above Average	Average	Below Average
a.	Write-up (2 Marks)	Able to write the complete code (2)	Able to write the code with few errors. (1)	Unable to write . (0)
b.	Execution (4 Marks)	Executed successfully for all the input set given(4)	Obtained Partially correct results.(2-3)	No Execution. (0)
c.	Viva (2 Marks)	Able to answer all the questions correctly(2)	Able to answer few questions (1)	Not answered any.(0)

Computer Networks Laboratory

Subject Code: 18CSL57**Hours/Week: 03****Total Hours: 42****I.A. Marks : 20****Exam Hours: 03****Exam Marks: 80**

Lesson Planning / Schedule of Experiments

Sl. No	Name of Experiment	To be completed
1		Week1
2		Week2
3		Week3
4	4. Develop an assembly language program to compute nCr using recursive procedure. Assume that 'n' and 'r' are non-negative integers. 5. Design and develop an assembly language program to read the current time and Date from the system and display it in the standard format on the screen.	Week4
5	8a. Design and develop an assembly program to demonstrate BCD Up-Down Counter (00-99) on the Logic Controller Interface. 8b. Design and develop an assembly program to read the status of two 8-bit inputs (X & Y) from the Logic Controller Interface and display $X*Y$.	Week5
6	9. Design and develop an assembly program to display messages "FIRE" and "HELP" alternately with flickering effects on a 7-segment display interface for a suitable period of time. Ensure a flashing rate that makes it easy to read both the messages (Examiner does not specify these delay values nor is it necessary for the student to compute these values).	Week6
7	Lab Test I	Week7
8	10. Design and develop an assembly program to drive a Stepper Motor interface and rotate the motor in specified direction (clockwise or counter-clockwise) by N steps (Direction and N are specified by the examiner). Introduce suitable delay between successive steps. (Any arbitrary value for the delay may be assumed by the student).	Week8
9	11a. Design and develop an assembly language program to generate the Sine Wave using DAC interface (The output of the DAC is to be displayed on the CRO). 11b. Generate a Half Rectified Sine waveform using the DAC interface. (The output of the DAC is to be displayed on the CRO).	Week9

10	6. To write and simulate ARM assembly language programs for data transfer, arithmetic and logical operations (Demonstrate with the help of a suitable program). 7.To write and simulate C Programs for ARM microprocessor using KEIL (Demonstrate with the help of a suitable program)	Week10
11	12. To interface LCD with ARM processor-- ARM7TDMI/LPC2148. Write and execute programs in C language for displaying text messages and numbers on LCD	Week11
12	13.To interface Stepper motor with ARM processor-- RM7TDMI/LPC2148. Write a program to rotate stepper motor	Week12
13	Repetition of lab experiments/ extra programs.	Week13
14	Lab Test II	Week14

Microprocessor and Microcontroller Laboratory

Subject Code: 15CSL48

Hours/Week: 03

Total Hours: 42

I.A. Marks : 20

Exam Hours: 03

Exam Marks: 80

Procedure for Program Execution

Execution Commands

C :/> cd foldername

C:/foldername>edit filename.asm

After this command executed in command prompt an editor window will open. Program should be typed in this window and saved.

To run the program, the following steps have to be followed:

C:/foldername>tasm filename.asm

After this command is executed in command prompt if there are no errors in program regarding to syntax the assembler will generates an object file.

C:/foldername>tlink filename.obj

After verifying the program for correct syntax, the generated object files should be linked together. It will give an EXE file.

To execute EXE file, if the input is from keyboard, **command (A)** is used.

After generating EXE file by the assembler, if the inputs are in the data segment, the program execution can be viewed in single stepping using the debugger mode with the command **(B)**.

(A) C:/foldername> filename.exe

(B) C:/foldername>td filename.exe

Note:

- Here onwards lab experiments and solutions continues.
- Marks distribution in Rubrix table change as per ur lab subject.
15 scheme as given above.
10 scheme is 10 for daily assess+5 for record+10 for test
17 scheme 8 for observation +8 for record +8 for viva+16 for test (3+3+10)

- After each experiment show the following:

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	✓	✓	✓							✓			✓		

PART -A

- Implement three nodes point – to – point network with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.

Network topology

```

      10.1.1.0          10.1.2.0
n0 ----- n1..... n2
point-to-point

```

In this program we have created 3 point to point nodes n0, n1, n2. Node n0 has IP address 10.1.1.1 and n3 has 10.1.2.2. Node n1 has 2 interfaces(10.1.1.2 and 10.1.2.1). OnOffHelper application is used to generate the traffic at source node n0. Packets move from n0 to n2 via n1. Acknowledgment is sent from n2 to n0 via n1. Details of the flow(Number of packets sent, received and dropped) can be verified by using tracemetrics(lab1.tr file).

Program

```

#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"
#include "ns3/traffic-control-module.h"

```

```
using namespace ns3;
```

```
NS_LOG_COMPONENT_DEFINE ("Lab-Program-1");
```

```

int main (int argc, char *argv[])
{
std::string socketType= "ns3::TcpSocketFactory";;

```

```

CommandLine cmd;
cmd.Parse (argc, argv);

```

```

NodeContainer nodes;
nodes.Create (3);                //3 point-to-point nodes are created

```

```
InternetStackHelper stack;
```

```
stack.Install (nodes); //TCP-IP layer functionality configured on all nodes

//Bandwidth and delay set for the point-to-point channel. Vary these parameters to //see the variation in
number of packets sent/received/dropped.
PointToPointHelper p2p1;
p2p1.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
p2p1.SetChannelAttribute ("Delay", StringValue ("1ms"));

//Set the base address for the first network(nodes n0 and n1)
Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");

NetDeviceContainer devices;

devices = p2p1.Install (nodes.Get (0), nodes.Get (1));Ipv4InterfaceContainer
interfaces = address.Assign (devices);

//Set the base address for the second network(nodes n1 and n2)
devices = p2p1.Install (nodes.Get (1), nodes.Get (2)); address.SetBase
("10.1.2.0", "255.255.255.0"); interfaces = address.Assign (devices);

//RateErrorModel allows us to introduce errors into a Channel at a given rate. //Vary the error rate
value to see the variation in number of packets dropped
Ptr<RateErrorModel>em = CreateObject<RateErrorModel> (); em-
>SetAttribute ("ErrorRate", DoubleValue (0.00002));
devices.Get (1)->SetAttribute ("ReceiveErrorModel", PointerValue (em));

//create routing table at all nodes
Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

uint32_t payloadSize = 1448;
OnOffHelper onoff (socketType, Ipv4Address::GetAny ());

//Generate traffic by using OnOff application
onoff.SetAttribute ("OnTime", StringValue
("ns3::ConstantRandomVariable[Constant=1]"));
onoff.SetAttribute ("OffTime", StringValue
("ns3::ConstantRandomVariable[Constant=0]"));
onoff.SetAttribute ("PacketSize", UIntegerValue (payloadSize));
onoff.SetAttribute ("DataRate", StringValue ("50Mbps")); //bit/s
```

```
uint16_t port = 7;
//Install receiver (for packetsink) on node 2
Address localAddress1 (InetSocketAddress (Ipv4Address::GetAny (), port));
PacketSinkHelper packetSinkHelper1 (socketType, localAddress1);
ApplicationContainer sinkApp1 = packetSinkHelper1.Install (nodes.Get (2));
sinkApp1.Start (Seconds (0.0));
sinkApp1.Stop (Seconds (10));

//Install sender app on node 0
ApplicationContainer apps;
AddressValue remoteAddress (InetSocketAddress (interfaces.GetAddress (1), port)); onoff.SetAttribute
("Remote", remoteAddress);
apps.Add (onoff.Install (nodes.Get (0))); apps.Start
(Seconds (1.0)); apps.Stop (Seconds (10));

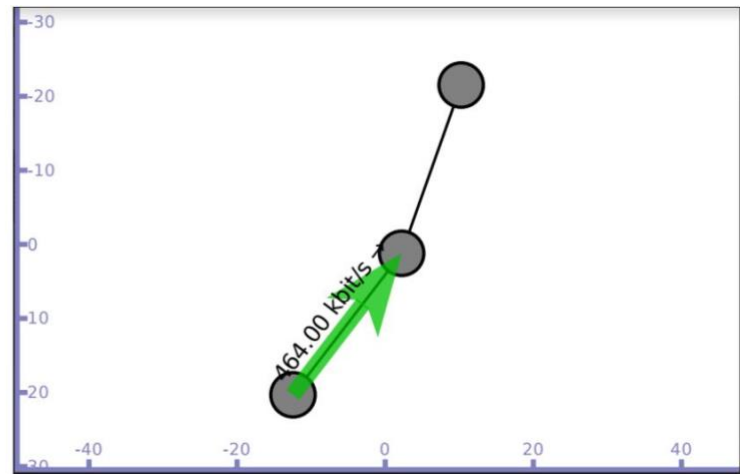
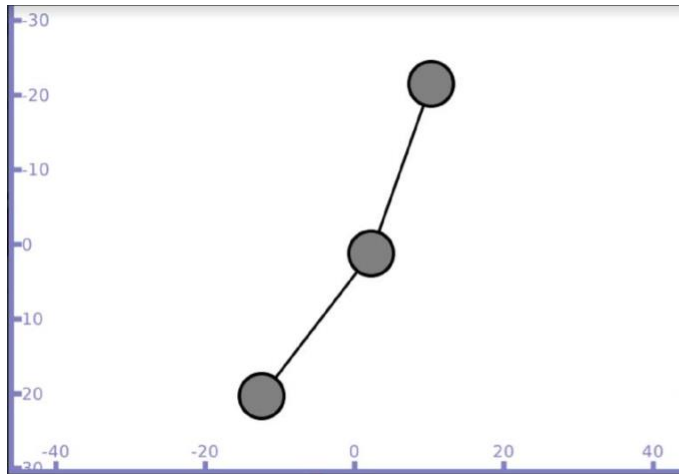
Simulator::Stop (Seconds (10));

AsciiTraceHelper ascii;
p2p1.EnableAsciiAll (ascii.CreateFileStream ("lab1.tr"));

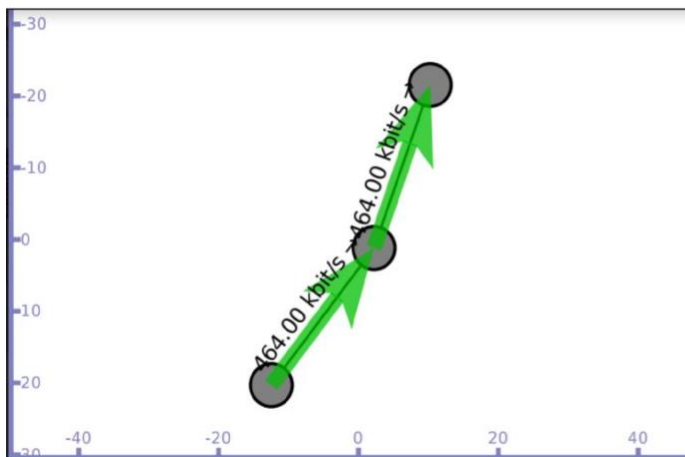
//Run the simulator
Simulator::Run ();
Simulator::Destroy ();
return 0;
}
```

```
./waf --run scratch/Program1 --vis
```

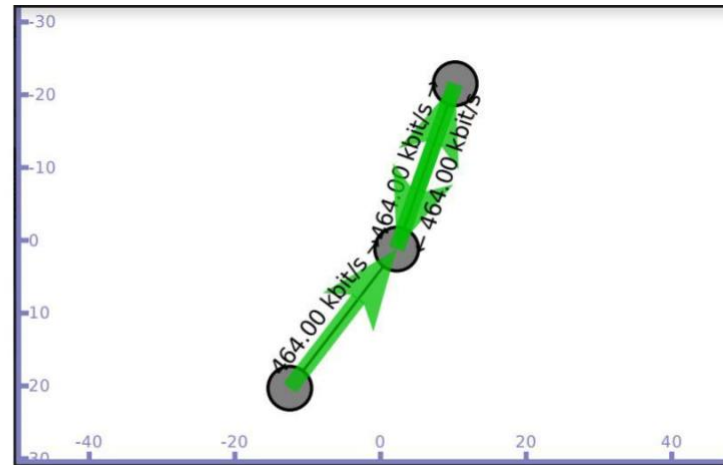
Output



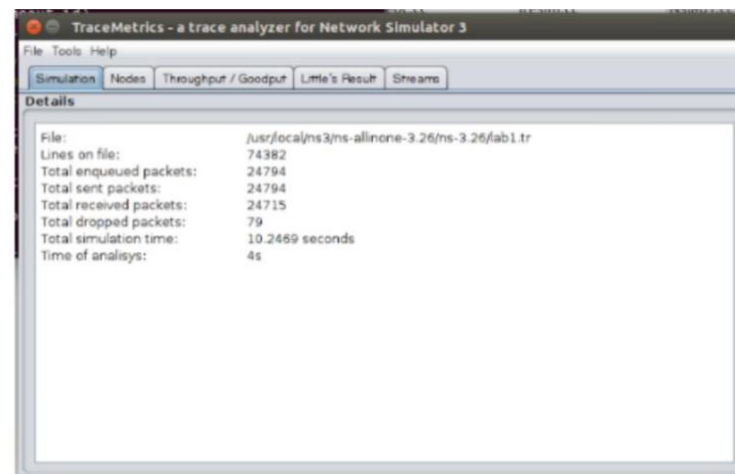
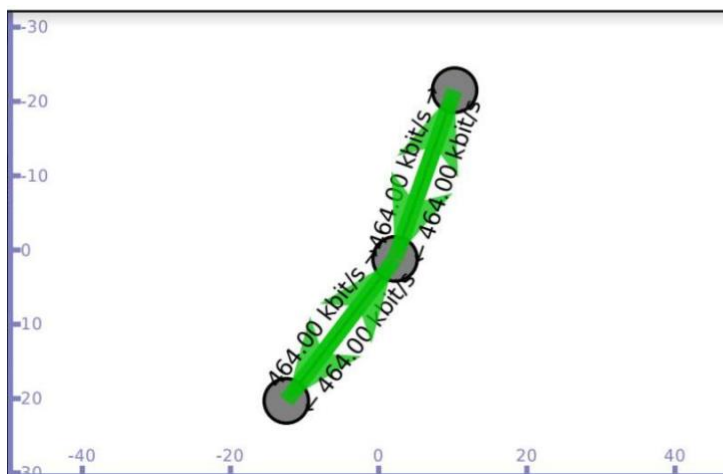
Packet sent from n0 to n1 and then to n2



Acknowledgment sent from n2



Flow details on trace file lab1.tr



2. Implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

Network topology

```
n0      n1      n2      n3      n4      n5
|        |        |        |        |        |
=====
CSMA channel with base IP 10.1.1.0
```

In this program we have created 6 CSMA nodes n0, n1, n2, n3, n4 and n5 with IP addresses 10.1.1.1, 10.1.1.2, 10.1.1.3, 10.1.1.4, 10.1.1.5 and 10.1.1.6 respectively.

Nodes n0 and n1 ping node n2, we can visualize the ping messages transferred between the nodes. Data transfer is also simulated between the nodes n0 and n2 using `UdpSocketFactory` to generate traffic.

Program

```
#include <iostream>
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/csma-module.h"
#include "ns3/applications-module.h"
#include "ns3/internet-apps-module.h"
#include "ns3/internet-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("Lab-Program-2");

static void PingRtt (std::string context, Time rtt)
{
    std::cout << context <<" "<< rtt << std::endl;
}

int main (int argc, char *argv[])
{
    CommandLine cmd;
    cmd.Parse (argc, argv);

    // Here, we will explicitly create six nodes.
    NS_LOG_INFO ("Create nodes.");
    NodeContainer c;

    c.Create (6);
```



```
// connect all our nodes to a shared channel.
NS_LOG_INFO ("Build Topology.");
CsmHelper csm;

csm.SetChannelAttribute ("DataRate", DataRateValue (DataRate (10000)));
csm.SetChannelAttribute ("Delay", TimeValue (Milliseconds
(0.2)));NetDeviceContainer devs = csm.Install (c);

// add an ip stack to all nodes.

NS_LOG_INFO ("Add ip stack.");
InternetStackHelper ipStack;

ipStack.Install (c);

// assign ip addresses
NS_LOG_INFO ("Assign ip addresses.");
Ipv4AddressHelper ip;
ip.SetBase ("192.168.1.0", "255.255.255.0");Ipv4InterfaceContainer
addresses = ip.Assign (devs);

NS_LOG_INFO ("Create Sink.");

// Create an OnOff application to send UDP datagrams from node zero to node 1. NS_LOG_INFO
("Create Applications.");

uint16_t port = 9; // Discard port (RFC 863)

OnOffHelper onoff ("ns3::UdpSocketFactory",
Address (InetSocketAddress (addresses.GetAddress (2), port)));
onoff.SetConstantRate (DataRate ("500Mb/s"));
```

```
ApplicationContainer app = onoff.Install (c.Get (0));
// Start the application app.Start
  (Seconds (6.0)); app.Stop
  (Seconds (10.0));

// Create an optional packet sink to receive these packets
PacketSinkHelper sink ("ns3::UdpSocketFactory",

Address (InetSocketAddress (Ipv4Address::GetAny (), port))); app = sink.Install
(c.Get (2));
app.Start (Seconds (0.0));

NS_LOG_INFO ("Create pinger");
V4PingHelper ping = V4PingHelper (addresses.GetAddress (2));
NodeContainer pingers;
pingers.Add (c.Get (0));
pingers.Add (c.Get (1));

ApplicationContainer apps;
apps = ping.Install (pingers);
apps.Start (Seconds (1.0));
apps.Stop (Seconds (5.0));

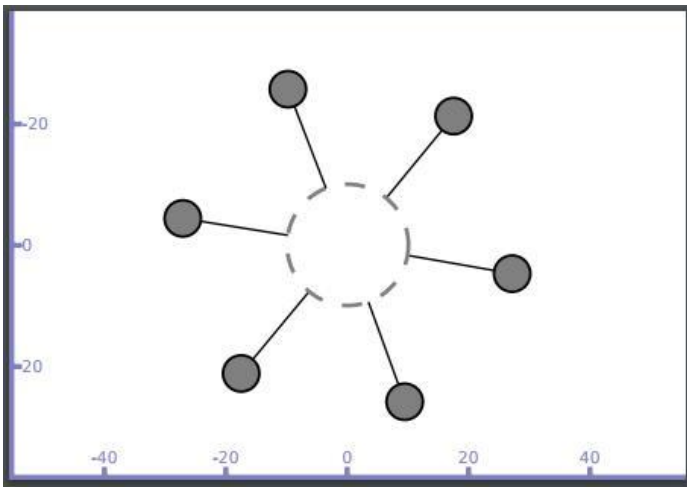
// finally, print the ping rtts.
Config::Connect ("/NodeList/*/ApplicationList*/$ns3::V4Ping/Rtt", MakeCallback
(&PingRtt));

NS_LOG_INFO ("Run Simulation.");

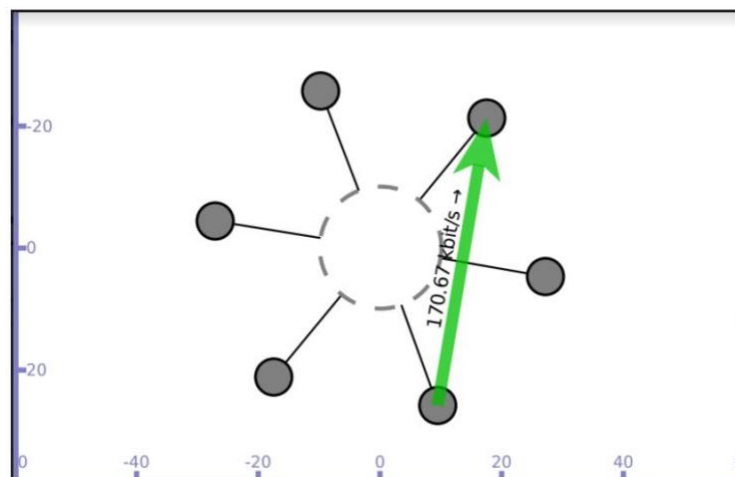
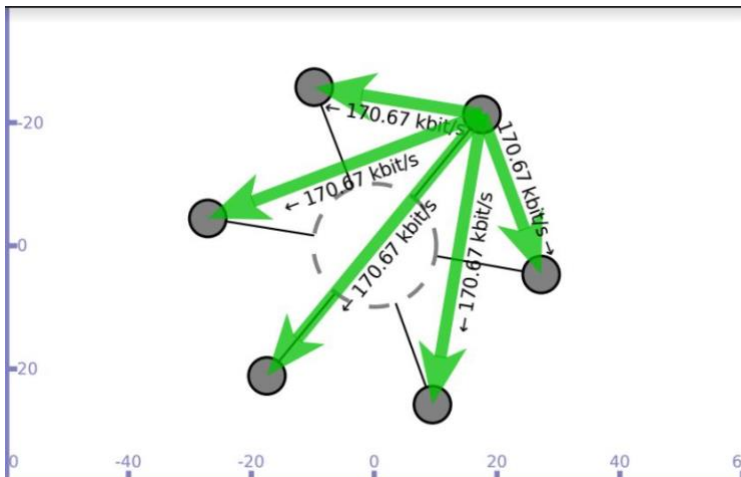
AsciiTraceHelper ascii;
csma.EnableAsciiAll (ascii.CreateFileStream ("ping1.tr"));

Simulator::Run ();
Simulator::Destroy ();
NS_LOG_INFO ("Done.");
}
```

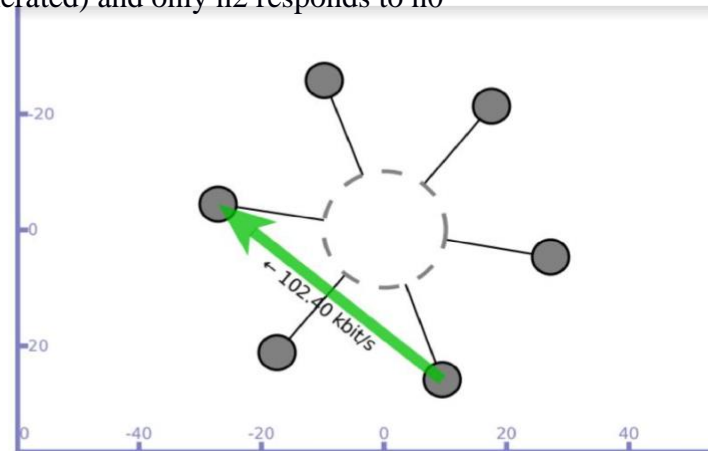
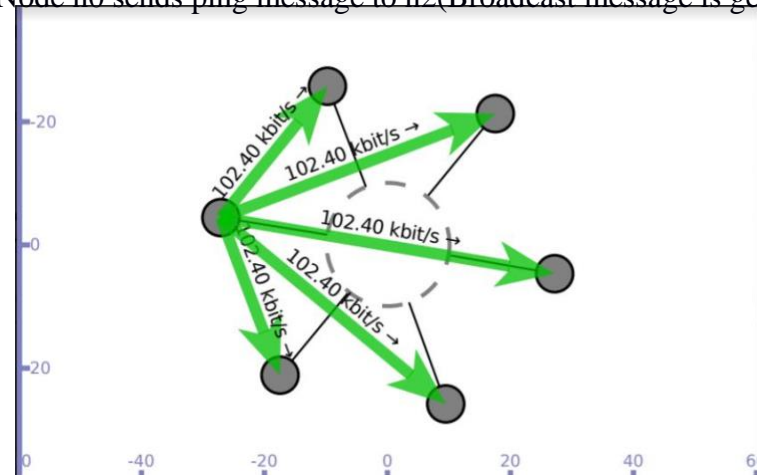
```
./waf - - run scratch/Program2 - -vis
```



Node n1 sends ping message to n2(Broadcast message is generated) and only n2 responds to n1

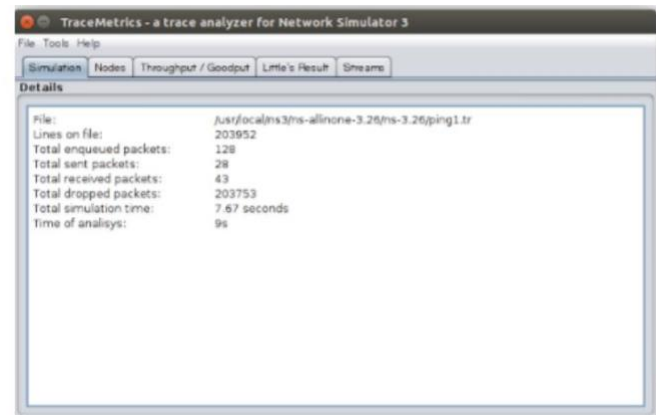
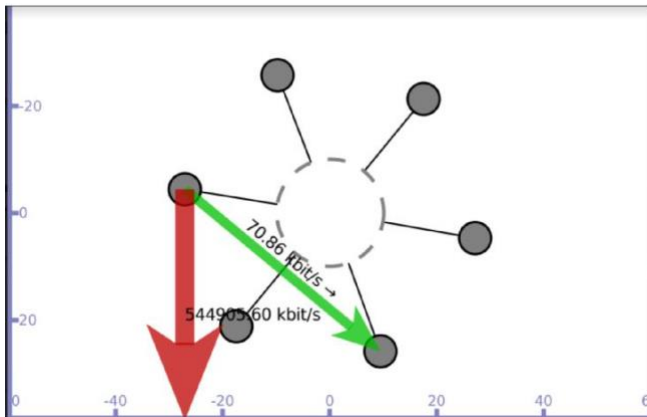


Node n0 sends ping message to n2(Broadcast message is generated) and only n2 responds to n0



Data transfer simulated between nodes n0 and n2

Trace file(ping1.tr) generated



- Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.

Network topology

n0 n1 n2 n3
| | | |

===== CSMA channel with base IP 10.1.1.0

Source node – n0

sink node - n1

In this program we have created 4 CSMA nodes n0, n1, n2 and n3 with IP addresses 10.1.1.1, 10.1.1.2, 10.1.1.3 and 10.1.1.4 respectively. Data transmission is simulated between nodes n0 and n1. Once the cwnd values are generated, they are exported to .dat file and congestion graph is plot using gnuplot.

Program

```
#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"
#include <iostream>
#include "ns3/csma-module.h"
```

using namespace ns3;

```
NS_LOG_COMPONENT_DEFINE ("3rd Lab Program");
```

//MyApp class inherits the ns-3 Application class defined in
//src/network/model/application.h.

//The MyApp class is obligated to override the StartApplication and //StopApplication methods. These methods are automatically called when MyApp is //required to start and stop sending data during the simulation.

```
class MyApp : public Application
{
public:
```

```
    MyApp ();
    virtual ~MyApp();
```

```
    void Setup (Ptr<Socket> socket, Address address, uint32_t packetSize, uint32_t nPackets, DataRate
dataRate);
```

```
private:
    virtual void StartApplication (void);
    virtual void StopApplication (void);
```

```
    void ScheduleTx (void);
    void SendPacket (void);
```

```
    Ptr<Socket>      m_socket;
    Address           m_peer;
    uint32_t          m_packetSize;
    uint32_t          m_nPackets;
    DataRate          m_dataRate;
    EventId           m_sendEvent;
    bool              m_running;
    uint32_t          m_packetsSent;
};
```

```
MyApp::MyApp ()          // constructor
:   m_socket (0),
    m_peer (),
    m_packetSize
(0), m_nPackets
(0), m_dataRate
(0), m_sendEvent
(0), m_running
(false),
    m_packetsSent
(0)
```

```
{
```

```
}
MyApp::~MyApp()          // destructor
{
    m_socket = 0;
}

// initialize member variables.
void MyApp::Setup (Ptr<Socket> socket, Address address, uint32_t packetSize,
uint32_t nPackets, DataRate dataRate)
{
    m_socket = socket;
    m_peer = address;
    m_packetSize = packetSize;
    m_nPackets = nPackets;
    m_dataRate = dataRate;
}

// Below code is the overridden implementation of Application::StartApplication. It //does a socket bind
operation and establishes TCP connection with the address //specified in m_peer.

void MyApp::StartApplication (void)
{
    m_running = true;
    m_packetsSent = 0;
    m_socket->Bind ();
    m_socket->Connect (m_peer);
    SendPacket ();
}

//The next bit of code explains to the Application how to stop creating simulation //events.

void MyApp::StopApplication (void)
{
    m_running = false;
    if (m_sendEvent.IsRunning ())
    {
        Simulator::Cancel (m_sendEvent);
    }
    if (m_socket)
    {
        m_socket->Close ();
    }
}

//StartApplication calls SendPacket to start the chain of events that describes the //Application behavior.
void MyApp::SendPacket (void)
{

```

```
Ptr<Packet> packet = Create<Packet> (m_packetSize);
m_socket->Send (packet);
```

```
if (++m_packetsSent < m_nPackets)
{
ScheduleTx ();
}
}
```

//It is the responsibility of the Application to keep scheduling the chain of //events, so the next lines call ScheduleTx to schedule another transmit event //(a SendPacket) until the Application decides it has sent enough.

```
void MyApp::ScheduleTx (void)
{
if (m_running)
{
Time tNext (Seconds (m_packetSize * 8 / static_cast<double> (m_dataRate.GetBitRate ())));
m_sendEvent = Simulator::Schedule (tNext, &MyApp::SendPacket, this);
}
}
```

//Below function logs the current simulation time and the new value of the congestion window every time it is changed.

```
static void CwndChange (uint32_t oldCwnd, uint32_t newCwnd)
{
NS_LOG_UNCOND (Simulator::Now ().GetSeconds () <<"\t"<< newCwnd);
}
```

//trace sink to show where packets are dropped

```
static void RxDrop (Ptr<const Packet> p)
{
NS_LOG_UNCOND ("RxDrop at "<< Simulator::Now ().GetSeconds ());
}
```

//main function

```
int main (int argc, char *argv[])
{
CommandLine cmd;
cmd.Parse (argc, argv);
```

```
NS_LOG_INFO ("Create nodes.");
NodeContainer nodes;
nodes.Create (4);//4 csma nodes are created
```

```
CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("5Mbps"));
```

```
csma.SetChannelAttribute ("Delay", TimeValue (Milliseconds (0.0001)));

NetDeviceContainer devices;
devices = csma.Install (nodes);

//RateErrorModel allows us to introduce errors into a Channel at a given rate.

Ptr<RateErrorModel>em = CreateObject<RateErrorModel> (); em-
>SetAttribute ("ErrorRate", DoubleValue (0.00001));
devices.Get (1)->SetAttribute ("ReceiveErrorModel", PointerValue (em));

InternetStackHelper stack;
stack.Install (nodes);

Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");Ipv4InterfaceContainer
interfaces = address.Assign (devices);

uint16_t sinkPort = 8080;

//PacketSink Application is used on the destination node to receive TCP connections //and data.

Address sinkAddress (InetSocketAddress (interfaces.GetAddress (1), sinkPort)); PacketSinkHelper
packetSinkHelper ("ns3::TcpSocketFactory", InetSocketAddress (Ipv4Address::GetAny (),
sinkPort));

ApplicationContainer sinkApps = packetSinkHelper.Install (nodes.Get (1));

sinkApps.Start (Seconds (0.));
sinkApps.Stop (Seconds (20.));

//next two lines of code will create the socket and connect the trace source.

Ptr<Socket> ns3TcpSocket = Socket::CreateSocket (nodes.Get (0),
TcpSocketFactory::GetTypeId ());
ns3TcpSocket->TraceConnectWithoutContext ("CongestionWindow", MakeCallback
(&CwndChange));

//creates an Object of type MyApp

Ptr<MyApp> app = CreateObject<MyApp> ();

//tell the Application what Socket to use, what address to connect to, how much //data to send at each send
event, how many send events to generate and the rate at //which to produce data from those events.

app->Setup (ns3TcpSocket, sinkAddress, 1040, 1000, DataRate ("50Mbps")); nodes.Get
(0)->AddApplication (app); app->SetStartTime (Seconds (1.));
```



```
app->SetStopTime (Seconds (20.));
```

```
devices.Get (1)->TraceConnectWithoutContext ("PhyRxDrop", MakeCallback (&RxDrop));
```

```
Simulator::Stop (Seconds (20));
```

```
Simulator::Run ();
```

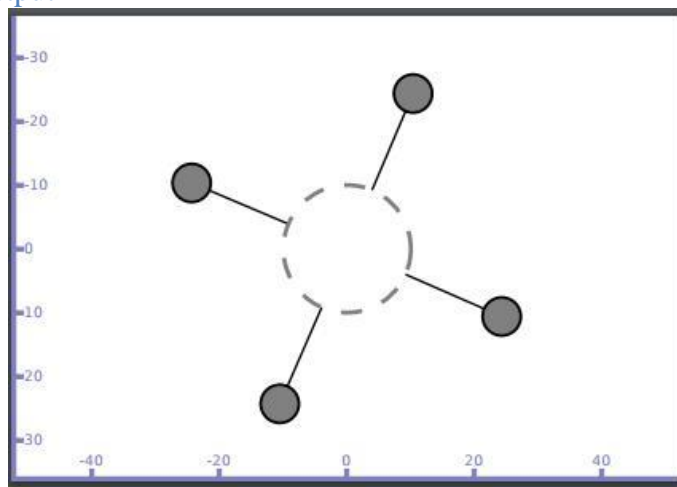
```
Simulator::Destroy ();
```

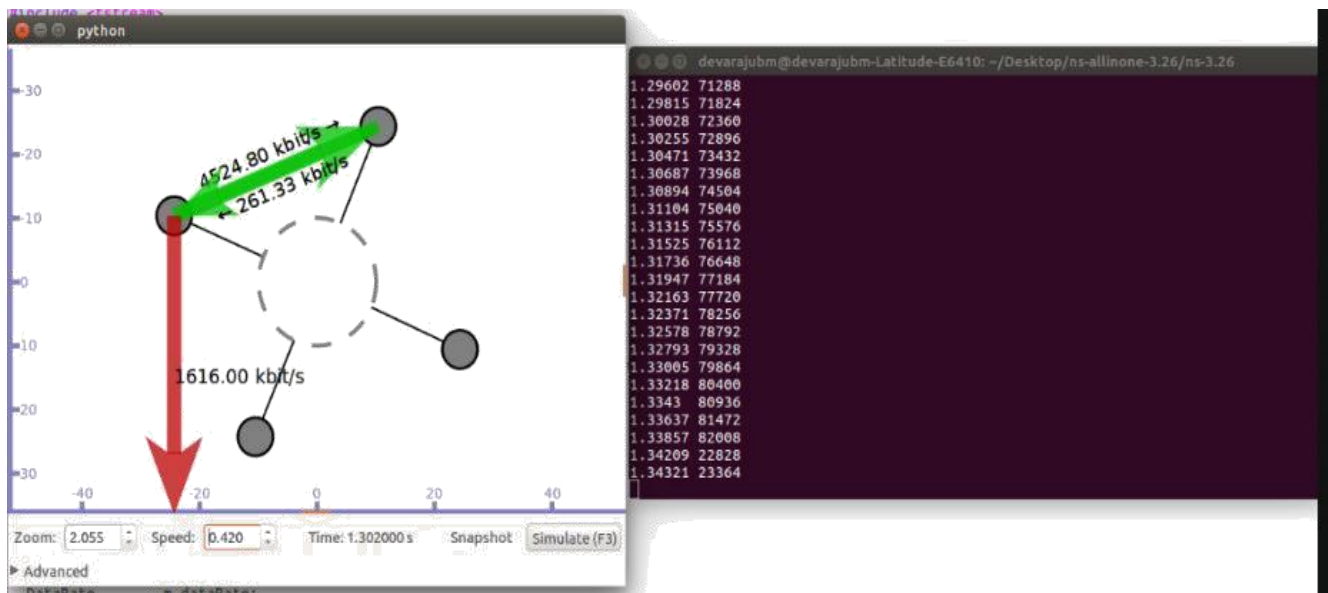
```
return 0;
```

```
}
```

```
./waf - - run scratch/Program3 - -vis
```

Output





Redirect the output to a file called cwnd.dat

```
./waf --run scratch/Program3 > cwnd.dat 2>&1
```

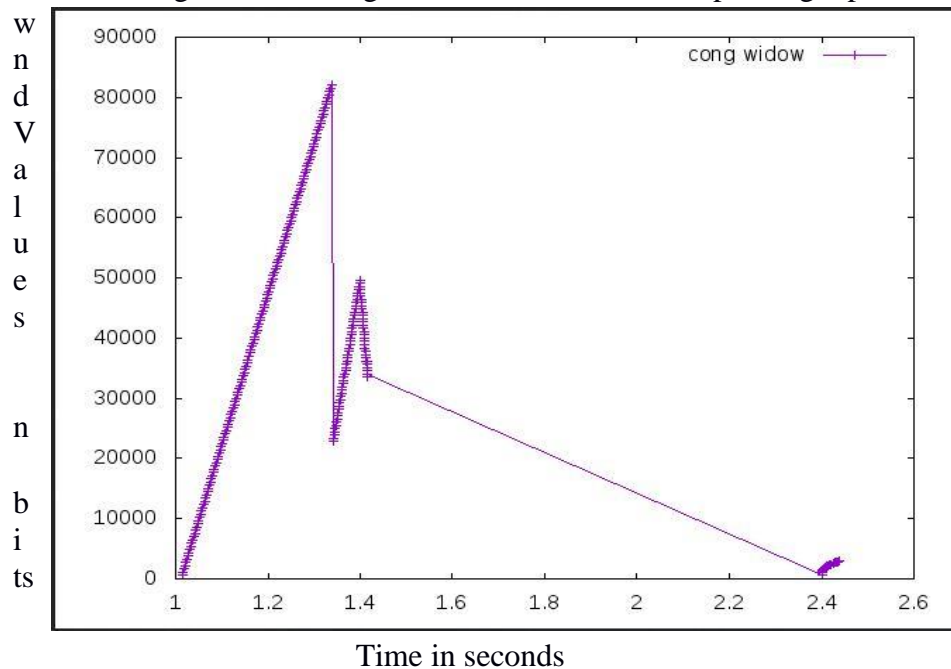
Now run gnuplot

```
gnuplot> set terminal png size 640,480
```

```
gnuplot> set output "cwnd.png"
```

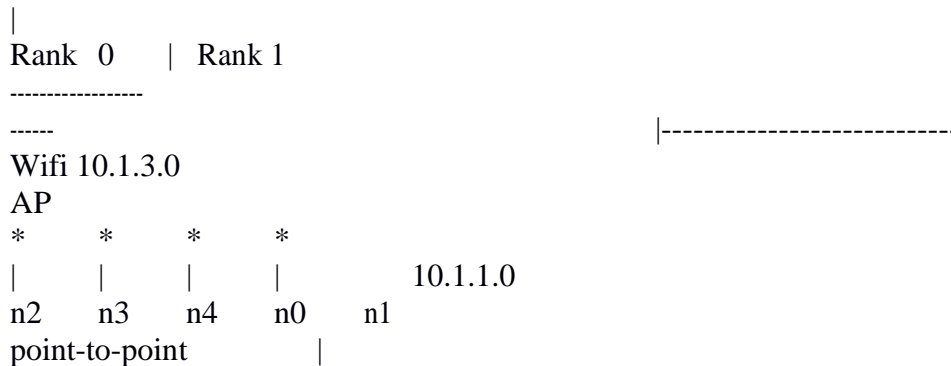
```
gnuplot> plot "cwnd.dat" using 1:2 title 'Congestion Window' with linespoints gnuplot> exit
```

i



4.Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets.

Default Network Topology



In this program we have created 3 wifi (STA/mobile) nodes (n2, n3, n4), 2 point to point nodes (n0, n1) where n0 acts as access point and n1 is a base station. This program establishes connection between n2 (10.1.3.3) and n1 (10.1.1.2) through access point (10.1.1.1). The Performance is measured in terms of throughput of the nodes. It can be verified using tracemetrics (Files generated : Tracefilewifides and Tracefilewifisrc).

Program

```

#include "ns3/core-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/network-module.h"
#include "ns3/applications-module.h"
#include "ns3/wifi-module.h"
#include "ns3/mobility-module.h"
#include "ns3/internet-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("ThirdScriptExample");

int main (int argc, char *argv[])
{
    bool verbose = true;
    uint32_t nWifi = 3; // 3 wi-fi nodes are created

    CommandLine cmd;
    cmd.AddValue ("nWifi", "Number of wifi STA devices", nWifi);
    cmd.AddValue ("verbose", "Tell echo applications to log if true", verbose); cmd.Parse (argc, argv);

    if (verbose)
    {
  
```

```
LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO); }
```

```
NodeContainer p2pNodes;
p2pNodes.Create (2); // 2 nodes are n0,n1 are created
```

```
PointToPointHelper pointToPoint;
pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
```

```
NetDeviceContainer p2pDevices;
p2pDevices = pointToPoint.Install (p2pNodes);
```

```
NodeContainer wifiStaNodes;
wifiStaNodes.Create (nWifi);
NodeContainer wifiApNode = p2pNodes.Get (0); // 1st node of p2p is also access point
```

```
// default PHY layer configuration is used for wifi
YansWifiChannelHelper channel = YansWifiChannelHelper::Default ();
YansWifiPhyHelper phy = YansWifiPhyHelper::Default ();
phy.SetChannel (channel.Create ());
```

```
WifiHelper wifi;
wifi.SetRemoteStationManager ("ns3::AarfWifiManager"); // AARF= rate control algorithm
```

```
WifiMacHelper mac;
Ssid ssid = Ssid ("ns-3-ssid"); // ssid=service set identifier in 802.11
mac.SetType ("ns3::StaWifiMac",
"Ssid", SsidValue (ssid),
"ActiveProbing", BooleanValue (false));
```

```
NetDeviceContainer staDevices;
staDevices = wifi.Install (phy, mac, wifiStaNodes);
```

```
mac.SetType ("ns3::ApWifiMac", "Ssid", SsidValue (ssid));
```

```
NetDeviceContainer apDevices;
apDevices = wifi.Install (phy, mac, wifiApNode);
```

```
MobilityHelper mobility;
```

```
// 2 dimensional grid to initially place sta(stationary nodes)
mobility.SetPositionAllocator ("ns3::GridPositionAllocator", "MinX",
DoubleValue (10.0),
```

```
"MinY", DoubleValue (-10.0), "DeltaX",
DoubleValue (7.0), "DeltaY", DoubleValue
```

```
(12.0), "GridWidth", UIntegerValue (3),  
"LayoutType", StringValue ("RowFirst"));
```

```
mobility.SetMobilityModel  
("ns3::RandomWalk2dMobilityModel", "Bounds", RectangleValue  
(Rectangle (-50, 50, -50, 50))); mobility.Install (wifiStaNodes);
```

```
mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");  
mobility.Install (wifiApNode);
```

```
InternetStackHelper stack;  
stack.Install (p2pNodes.Get(1)); // stack installed on n1 of p2p  
stack.Install (wifiApNode); // stack installed on access point
```

```
stack.Install (wifiStaNodes); // stack installed on mobile nodes
```

```
Ipv4AddressHelper address;  
address.SetBase ("10.1.1.0",  
"255.255.255.0"); Ipv4InterfaceContainer p2pInterfaces;  
p2pInterfaces = address.Assign (p2pDevices);
```

```
address.SetBase ("10.1.3.0", "255.255.255.0"); address.Assign  
(staDevices); address.Assign (apDevices);
```

```
//install echo server application on n1  
UdpEchoServerHelper echoServer (9);  
ApplicationContainer serverApps = echoServer.Install (p2pNodes.Get (1));  
serverApps.Start (Seconds (1.0));  
serverApps.Stop (Seconds (10.0));
```

```
//install echo client application on n3  
UdpEchoClientHelper echoClient (p2pInterfaces.GetAddress (1), 9);  
echoClient.SetAttribute ("MaxPackets", UIntegerValue (1)); echoClient.SetAttribute  
("Interval", TimeValue (Seconds (1.0))); echoClient.SetAttribute ("PacketSize",  
UIntegerValue (1024));
```

```
ApplicationContainer clientApps =  
echoClient.Install (wifiStaNodes.Get (nWifi - 1)); clientApps.Start  
(Seconds (2.0)); clientApps.Stop (Seconds (10.0));
```

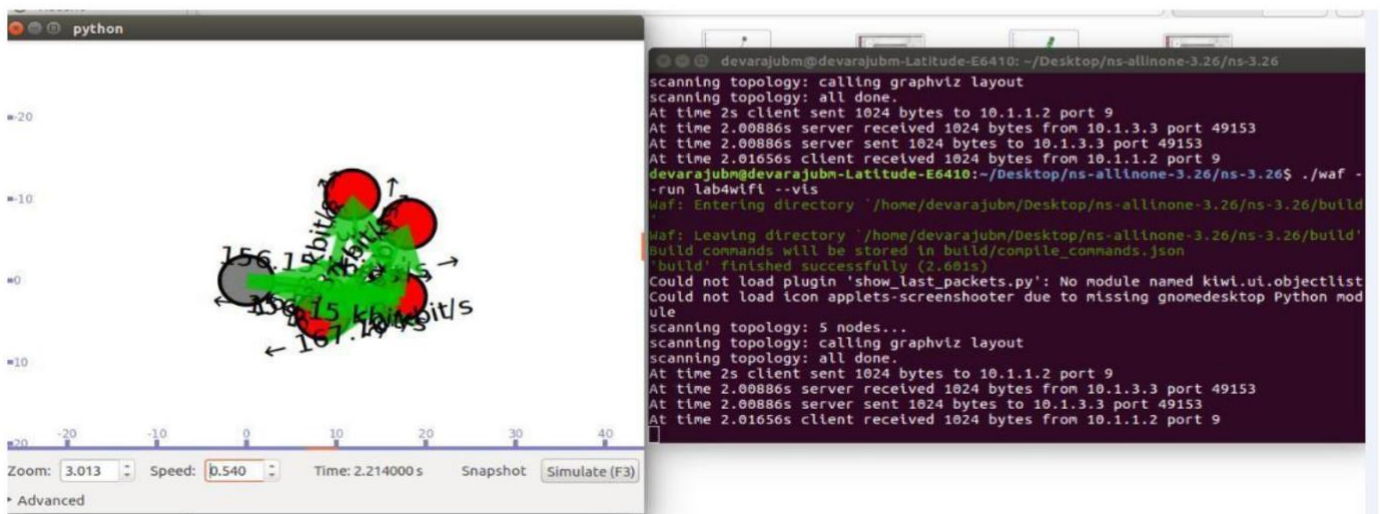
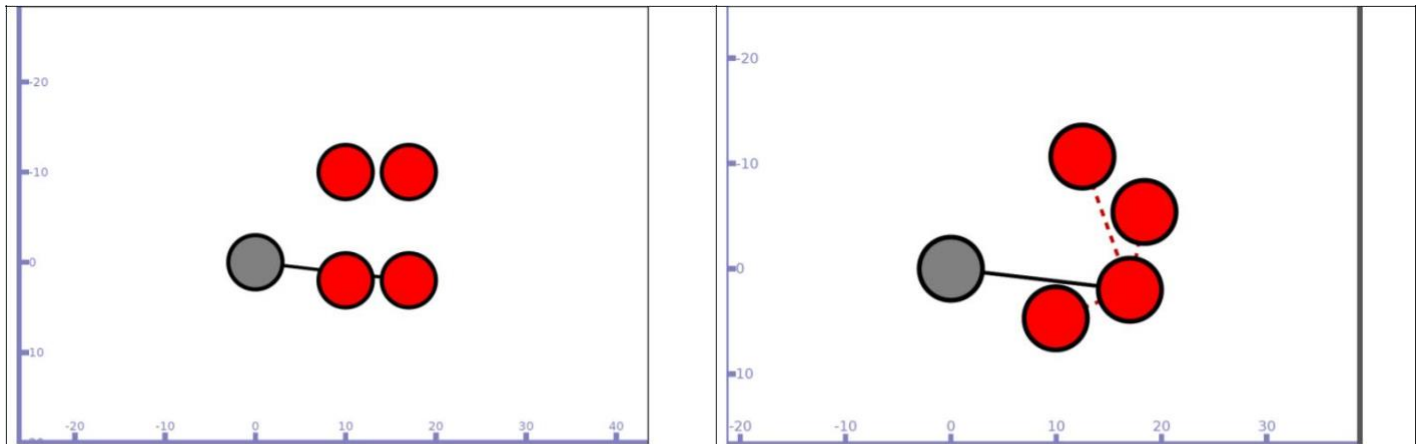
```
Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
```

```
Simulator::Stop (Seconds (10.0));  
AsciiTraceHelper ascii;  
pointToPoint.EnableAsciiAll (ascii.CreateFileStream ("Tracefilewifides.tr")); phy.EnableAsciiAll  
(ascii.CreateFileStream ("Tracefilewifisrc.tr"));
```

```
Simulator::Run ();  
Simulator::Destroy ();  
return 0;  
}
```

./waf --run scratch/Program4 --vis

Output



Trace file is used to see the throughput by using TraceMetrics

TraceMetrics - a trace analyzer for Network Simulator 3

File Tools Help

Simulation Nodes Throughput / Goodput Little's Result Streams

Node	Throughput	Goodput
0	523.7136965541229	508.80723460286697
1	523.7136965541229	508.80723460286697

TraceMetrics - a trace analyzer for Network Simulator 3

File Tools Help

Simulation Nodes Throughput / Goodput Little's Result Streams

Details

File:	/home/devarajubm/Desktop/ns-allinone-3.26/ns-3.26/Tracefilewifi.tr
Lines on file:	6
Total enqueued packets:	2
Total sent packets:	2
Total received packets:	2
Total dropped packets:	0
Total simulation time:	2.01255 seconds
Time of analysis:	0s

5. Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.
6. Implement and study the performance of CDMA on NS2/NS3 (Using stack called Call net) or equivalent environment.

```
#include "ns3/lte-helper.h"
#include "ns3/epc-helper.h"
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/internet-module.h"
#include "ns3/mobility-module.h"
#include "ns3/lte-module.h"
#include "ns3/applications-module.h"
#include "ns3/point-to-point-helper.h"
#include "ns3/config-store.h"
// #include "ns3/gtk-config-store.h"
// .....
using namespace ns3;
```

```
NS_LOG_COMPONENT_DEFINE ("EpcFirstExample");
```

```
int
main (int argc, char *argv[])
{

    uint16_t numberOfNodes = 2; // numberOfNodes = 6 for CDMA
    double simTime = 1.1;
    double distance = 60.0;
    double interPacketInterval = 100;

    // Command line arguments
    CommandLine cmd;

    cmd.Parse(argc, argv);

    Ptr<LteHelper> lteHelper = CreateObject<LteHelper> ();
    //This will instantiate some common objects (e.g., the Channel object) and provide the methods to add
    eNBs and UEs and configure them.
    Ptr<PointToPointEpcHelper> epcHelper = CreateObject<PointToPointEpcHelper> ();
    //PointToPointEpcHelper, which implements an EPC based on point-to-point links.
    //EpcHelper will also automatically create the PGW node and configure it so that it can properly handle
    traffic from/to the LTE radio access network.
    lteHelper->SetEpcHelper (epcHelper);
    //Then, you need to tell the LTE helper that the EPC will be used:
    ConfigStore inputConfig;
```



```
inputConfig.ConfigureDefaults();
//Specify configuration parameters of the objects that are being used for the simulation
// parse again so you can override default values from the command line
cmd.Parse(argc, argv);

Ptr<Node> pgw = epcHelper->GetPgwNode ();
//EpcHelper will also automatically create the PGW node and configure it so that it can properly handle
traffic from/to the LTE radio access network.

// Create a single RemoteHost
NodeContainer remoteHostContainer;
remoteHostContainer.Create (1);
Ptr<Node> remoteHost = remoteHostContainer.Get (0);
InternetStackHelper internet;
internet.Install (remoteHostContainer);

// Create the Internet
PointToPointHelper p2ph;
p2ph.SetDeviceAttribute ("DataRate", DataRateValue (DataRate ("100Gb/s")));
p2ph.SetDeviceAttribute ("Mtu", UIntegerValue (1500));
p2ph.SetChannelAttribute ("Delay", TimeValue (Seconds (0.010)));
NetDeviceContainer internetDevices = p2ph.Install (pgw, remoteHost);
Ipv4AddressHelper ipv4h;
ipv4h.SetBase ("1.0.0.0", "255.0.0.0");
Ipv4InterfaceContainer internetIpIfaces = ipv4h.Assign (internetDevices);
// interface 0 is localhost, 1 is the p2p device
Ipv4Address remoteHostAddr = internetIpIfaces.GetAddress (1);

Ipv4StaticRoutingHelper ipv4RoutingHelper;
Ptr<Ipv4StaticRouting> remoteHostStaticRouting = ipv4RoutingHelper.GetStaticRouting (remoteHost-
>GetObject<Ipv4> ());
remoteHostStaticRouting->AddNetworkRouteTo (Ipv4Address ("7.0.0.0"), Ipv4Mask ("255.0.0.0"), 1);

NodeContainer ueNodes;
NodeContainer enbNodes;
enbNodes.Create(numberOfNodes);
ueNodes.Create(numberOfNodes);

// Install Mobility Model
Ptr<ListPositionAllocator> positionAlloc = CreateObject<ListPositionAllocator> ();
for (uint16_t i = 0; i < numberOfNodes; i++)
{
    positionAlloc->Add (Vector(distance * i, 100, 100));
}
MobilityHelper mobility;
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
```

```
mobility.SetPositionAllocator(positionAlloc);
mobility.Install(enbNodes);
mobility.Install(ueNodes);

// Install LTE Devices to the nodes
NetDeviceContainer enbLteDevs = lteHelper->InstallEnbDevice (enbNodes);
NetDeviceContainer ueLteDevs = lteHelper->InstallUeDevice (ueNodes);

// Install the IP stack on the UEs
internet.Install (ueNodes);
Ipv4InterfaceContainer ueIpIface;
ueIpIface = epcHelper->AssignUeIpv4Address (NetDeviceContainer (ueLteDevs));
// Assign IP address to UEs, and install applications
for (uint32_t u = 0; u < ueNodes.GetN (); ++u)
{
    Ptr<Node> ueNode = ueNodes.Get (u);
    // Set the default gateway for the UE
    Ptr<Ipv4StaticRouting> ueStaticRouting = ipv4RoutingHelper.GetStaticRouting (ueNode-
>GetObject<Ipv4> ());
    ueStaticRouting->SetDefaultRoute (epcHelper->GetUeDefaultGatewayAddress (), 1);
}

// Attach one UE per eNodeB
for (uint16_t i = 0; i < numberOfNodes; i++)
{
    lteHelper->Attach (ueLteDevs.Get(i), enbLteDevs.Get(i));
    // side effect: the default EPS bearer will be activated
}

// Install and start applications on UEs and remote host
uint16_t dlPort = 1234;
uint16_t ulPort = 2000;
uint16_t otherPort = 3000;
ApplicationContainer clientApps;
ApplicationContainer serverApps;
for (uint32_t u = 0; u < ueNodes.GetN (); ++u)
{
    ++ulPort;
    ++otherPort;
    PacketSinkHelper dlPacketSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress
(Ipv4Address::GetAny (), dlPort));
    PacketSinkHelper ulPacketSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress
(Ipv4Address::GetAny (), ulPort));
    PacketSinkHelper packetSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress
(Ipv4Address::GetAny (), otherPort));
```

```
serverApps.Add (dlPacketSinkHelper.Install (ueNodes.Get(u)));
serverApps.Add (ulPacketSinkHelper.Install (remoteHost));
serverApps.Add (packetSinkHelper.Install (ueNodes.Get(u)));

UdpClientHelper dlClient (ueIpIface.GetAddress (u), dlPort);
dlClient.SetAttribute ("Interval", TimeValue (Milliseconds(interPacketInterval)));
dlClient.SetAttribute ("MaxPackets", UIntegerValue(1000000));

UdpClientHelper ulClient (remoteHostAddr, ulPort);
ulClient.SetAttribute ("Interval", TimeValue (Milliseconds(interPacketInterval)));
ulClient.SetAttribute ("MaxPackets", UIntegerValue(1000000));

UdpClientHelper client (ueIpIface.GetAddress (u), otherPort);
client.SetAttribute ("Interval", TimeValue (Milliseconds(interPacketInterval)));
client.SetAttribute ("MaxPackets", UIntegerValue(1000000));

clientApps.Add (dlClient.Install (remoteHost));
clientApps.Add (ulClient.Install (ueNodes.Get(u)));
if (u+1 < ueNodes.GetN ())
{
    clientApps.Add (client.Install (ueNodes.Get(u+1)));
}
else
{
    clientApps.Add (client.Install (ueNodes.Get(0)));
}
}
serverApps.Start (Seconds (0.01));
clientApps.Start (Seconds (0.01));
lteHelper->EnableTraces ();
// Uncomment to enable PCAP tracing
AsciiTraceHelper ascii;
p2ph.EnableAsciiAll(ascii.CreateFileStream("lab5.tr"));
p2ph.EnablePcapAll("lana-epc-first");

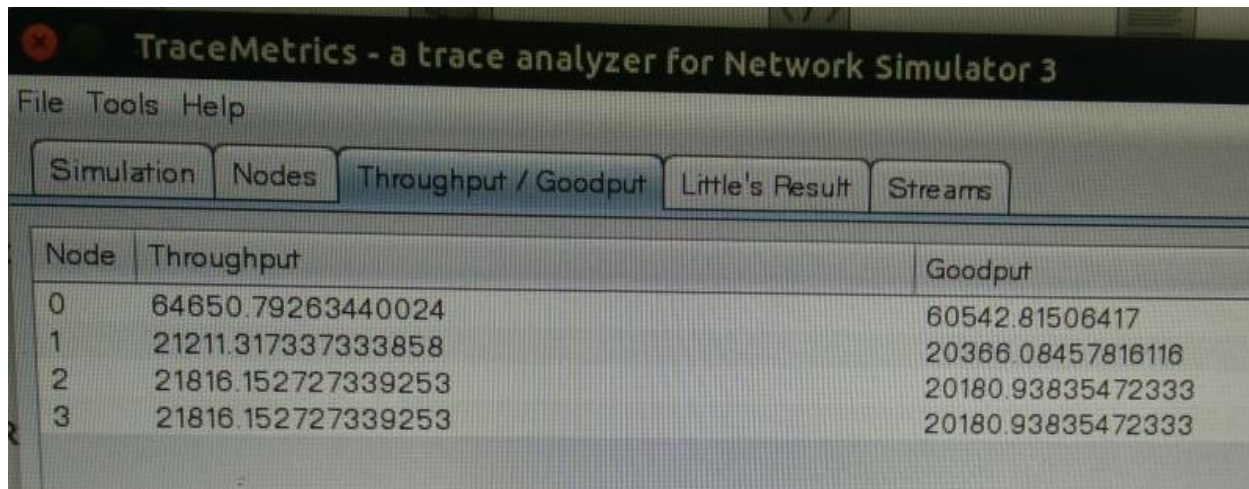
Simulator::Stop(Seconds(simTime));
Simulator::Run();

/*GtkConfigStore config;
config.ConfigureAttributes();*/

Simulator::Destroy();
return 0;

}
```

Output

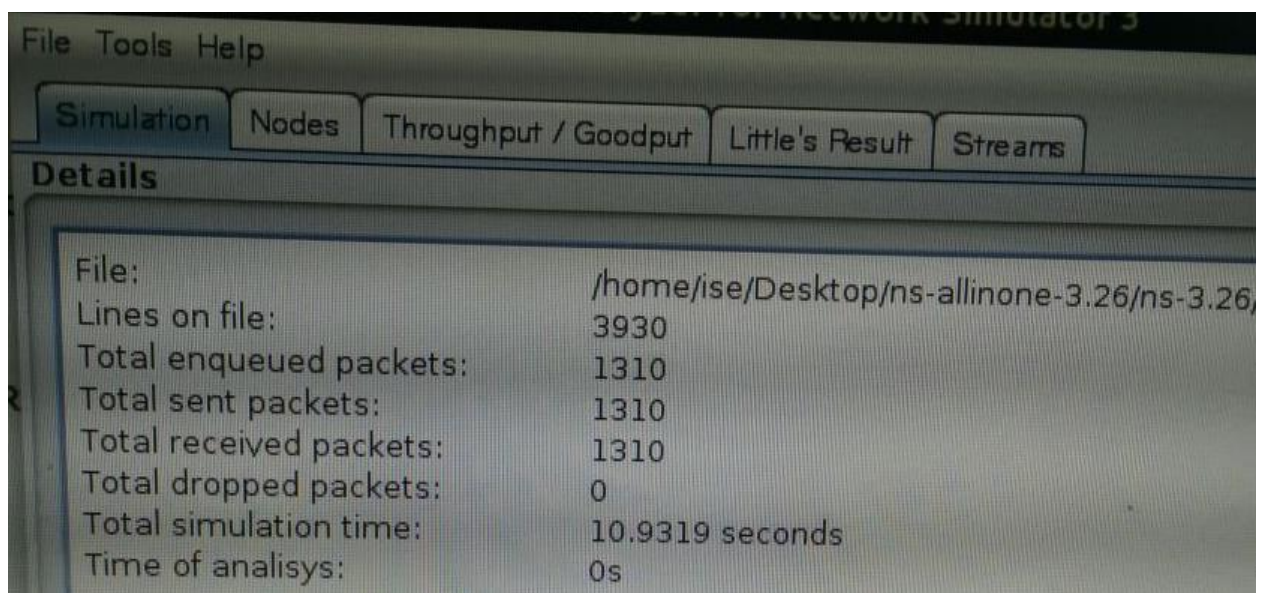


TraceMetrics - a trace analyzer for Network Simulator 3

File Tools Help

Simulation Nodes Throughput / Goodput Little's Result Streams

Node	Throughput	Goodput
0	64650.79263440024	60542.81506417
1	21211.317337333858	20366.08457816116
2	21816.152727339253	20180.93835472333
3	21816.152727339253	20180.93835472333

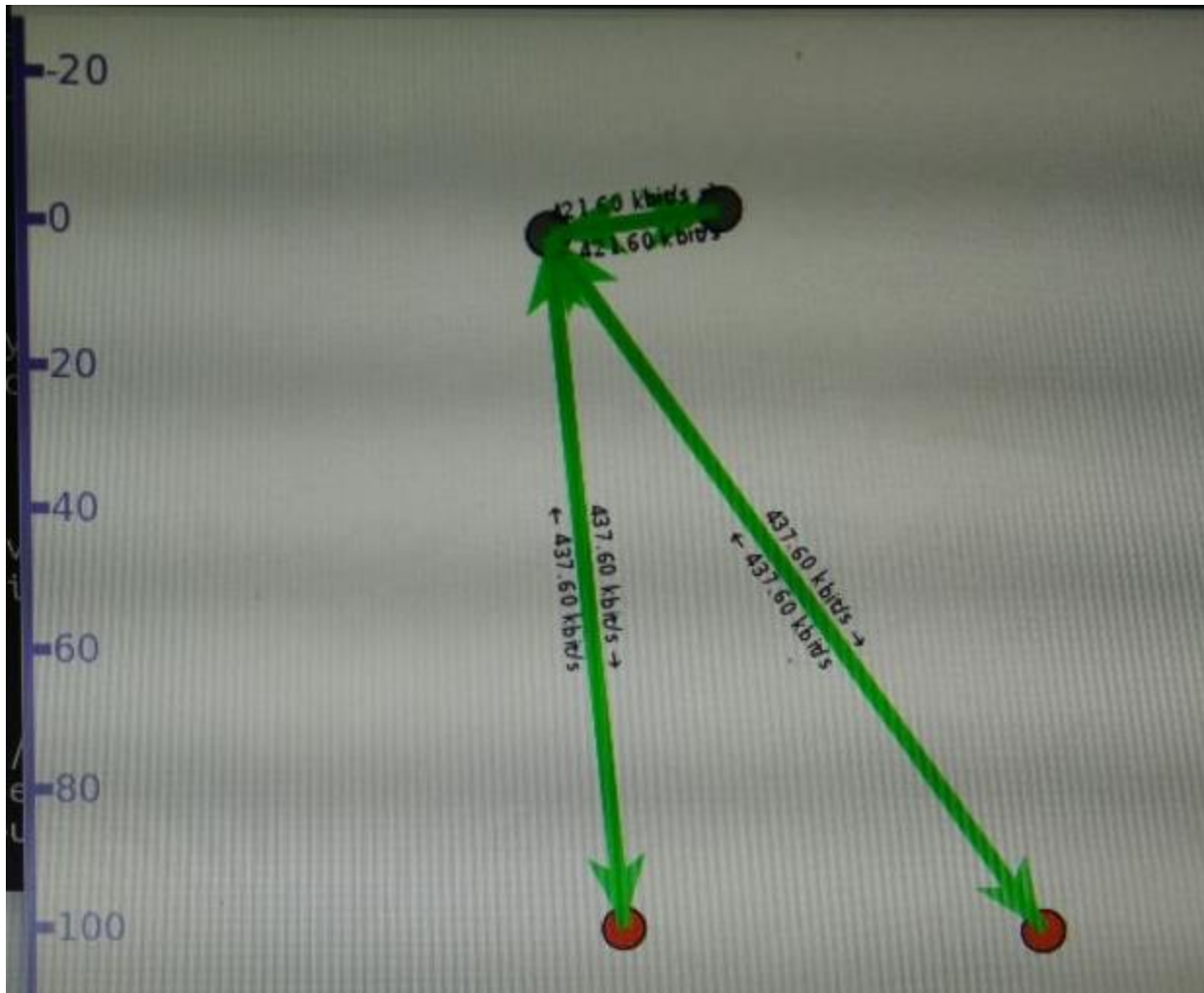


File Tools Help

Simulation Nodes Throughput / Goodput Little's Result Streams

Details

File:	/home/ise/Desktop/ns-allinone-3.26/ns-3.26,
Lines on file:	3930
Total enqueued packets:	1310
Total sent packets:	1310
Total received packets:	1310
Total dropped packets:	0
Total simulation time:	10.9319 seconds
Time of analysis:	0s



Part –B**7. Write a program for error detecting code using CRC-CCITT (16- bits).**

```
import java.io.*;
import java.util.Scanner;

class crcscanner
{
    public static void main(String a[]) throws IOException
    {

        Scanner sc=new Scanner(System.in);
        int[] message;
        int[] gen;
        int[] app_message;
        int[] rem;
        int[] trans_message;
        int message_bits,gen_bits, total_bits;

        System.out.println("\n Enter number of bits in message : ");
        message_bits=sc.nextInt();

        message=new int[message_bits];
        System.out.println("\n Enter message bits : ");
        for(int i=0; i<message_bits; i++)
            message[i]=sc.nextInt();
        System.out.println("\n Enter number of bits in gen : ");
        gen_bits=sc.nextInt();

        gen=new int[gen_bits];
        System.out.println("\n Enter gen bits : ");
        for(int i=0; i<gen_bits; i++)
        {
            gen[i]=sc.nextInt();
        }

        total_bits=message_bits+gen_bits-1;

        app_message=new int[total_bits];
        rem=new int[total_bits];
        trans_message=new int[total_bits];

        for(int i=0;i<message.length;i++)
        {
            app_message[i]=message[i];
        }

        System.out.print("\n Message bits are : ");
        for(int i=0; i< message_bits; i++)
        {
```

```
System.out.print(message[i]);
    }
    System.out.print("\n Generators bits are : ");
    for(int i=0; i< gen_bits; i++)
    {
        System.out.print(gen[i]);
    }

    System.out.print("\n Appended message is : ");
    for(int i=0; i< app_message.length; i++)
    {
        System.out.print(app_message[i]);
    }

    for(int j=0; j<app_message.length; j++)
    {
        rem[j] = app_message[j];
    }

    rem=computeecrc(app_message, gen, rem);

    for(int i=0;i<app_message.length;i++)
    {
        trans_message[i]=(app_message[i]^rem[i]);
    }

    System.out.println("\n Transmitted message from the transmitter is : ");
    for(int i=0;i<trans_message.length;i++)
    {
        System.out.print(trans_message[i]);
    }

    System.out.println("\n Enter received message of "+total_bits+" bits at receiver end : ");
    for(int i=0; i<trans_message.length; i++)
    {
        trans_message[i]=sc.nextInt();
    }
    System.out.println("\n Received message is :");
    for(int i=0; i< trans_message.length; i++)
    {
        System.out.print(trans_message[i]);
    }

    for(int j=0; j<trans_message.length; j++)
    {
        rem[j] = trans_message[j];
    }
    rem=computeecrc(trans_message, gen, rem);
    for(int i=0; i< rem.length; i++)
    {
        if(rem[i]!=0)
```



```

        {
            System.out.println("\n There is Error in the received me    ");
            break;
        }
        if(i==rem.length-1)

        System.out.println("\n There is No Error in the received m ");
    }
}

static int[] computecrc(int app_message[],int gen[], int rem[])
{
    int current=0;
    while(true)
    {
        for(int i=0;i<gen.length;i++)
        {
            rem[current+i]=(rem[current+i]^gen[i]);
        }
        while(rem[current]==0 && current!=rem.length-1)
        {
            current++;
        }
        if((rem.length-current)<gen.length)
        {
            break;
        }
    }
    return rem;
}
}

```

8. Write a program to find the shortest path between vertices using bellman-ford algorithm.

```

import java.util.Scanner;

public class BellmanFord
{
    private int D[]; private int num_ver;
    public static final int MAX_VALUE = 999;
    public BellmanFord(int num_ver)
    {
        this.num_ver = num_ver; D = new int[num_ver + 1];
    }
    public void BellmanFordEvaluation(int source, int A[][])
    {

```

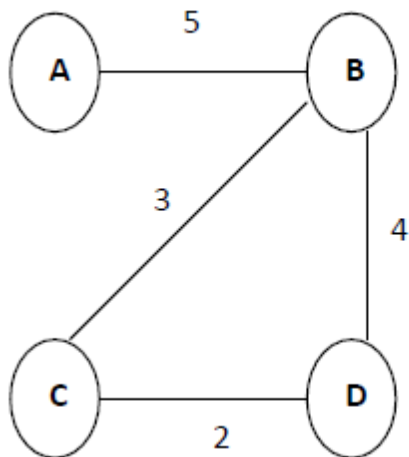


```
for (int node = 1; node <= num_ver; node++)
{
    D[node] = MAX_VALUE;

}
D[source] = 0;
for (int node = 1; node <= num_ver - 1; node++)
{
    for (int sn = 1; sn <= num_ver; sn++)
    {
        for (int dn = 1; dn <= num_ver; dn++)
        {
            if (A[sn][dn] != MAX_VALUE)
            {
                if (D[dn] > D[sn] + A[sn][dn])
                D[dn] = D[sn] + A[sn][dn];
            }
        }
    }
}
for (int sn = 1; sn <= num_ver; sn++)
{
    for (int dn = 1; dn <= num_ver; dn++)
    {
        if (A[sn][dn] != MAX_VALUE)
        {
            if (D[dn] > D[sn] + A[sn][dn])
            System.out.println("The Graph contains negative egde cycle");
        }
    }
}
for (int vertex = 1; vertex <= num_ver; vertex++)
{
    System.out.println("distance of source " + source + " to " + vertex + " is " + D[vertex]);
}
```

```
}  
public static void main(String[ ] args)  
{  
    int num_ver = 0; int source;  
    Scanner scanner = new Scanner(System.in); System.out.println("Enter the number of  
vertices"); num_ver = scanner.nextInt();  
    int A[][] = new int[num_ver + 1][num_ver + 1]; System.out.println("Enter the adjacency  
matrix"); for (int sn = 1; sn <= num_ver; sn++)  
    {  
        for (int dn = 1; dn <= num_ver; dn++)  
        {  
  
            A[sn][dn] = scanner.nextInt(); if (sn == dn)  
            {  
                A[sn][dn] = 0; continue;  
            }  
            if (A[sn][dn] == 0)  
            {  
                A[sn][dn] = MAX_VALUE;  
            }  
        }  
    }  
    System.out.println("Enter the source vertex"); source = scanner.nextInt();  
    BellmanFord b = new BellmanFord (num_ver); b.BellmanFordEvaluation(source, A);  
    scanner.close();  
}  
}
```

Input graph:



Output:

```
Enter the number of vertices
4
Enter the adjacency matrix
0 5 0 0
5 0 3 4
0 3 0 2
0 4 2 0
Enter the source vertex
2
distance of source 2 to 1 is 5
distance of source 2 to 2 is 0
distance of source 2 to 3 is 3
distance of source 2 to 4 is 4
```

9.Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.

Server Program

```
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;

public class SimpleFileServer {

    public final static int SOCKET_PORT = 13267; // you may change this
    public final static String FILE_TO_SEND = "e:/source1.txt"; // you may change this

    public static void main (String [] args ) throws IOException {
        FileInputStream fis = null;
        BufferedInputStream bis = null;
        OutputStream os = null;
        ServerSocket servsock = null;
        Socket sock = null;
        try {
            servsock = new ServerSocket(SOCKET_PORT);
            while (true) {
                System.out.println("Waiting...");
                try {
                    sock = servsock.accept();
                    System.out.println("Accepted connection : " + sock);
                    // send file
                    File myFile = new File (FILE_TO_SEND);
```

```
byte [] mybytearray = new byte [(int)myFile.length()];
fis = new FileInputStream(myFile);
bis = new BufferedInputStream(fis);
bis.read(mybytearray,0,mybytearray.length);
os = sock.getOutputStream();
System.out.println("Sending " + FILE_TO_SEND + "(" + mybytearray.length + " bytes)");
os.write(mybytearray,0,mybytearray.length);
os.flush();
System.out.println("Done.");
}
finally {
    if (bis != null) bis.close();
    if (os != null) os.close();
    if (sock!=null) sock.close();
}
}
}
finally {
    if (servsock != null) servsock.close();
}
}
}
```

Client Program

```
import java.io.BufferedOutputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.net.Socket;
```

```
public class SimpleFileClient {

    public final static int SOCKET_PORT = 13267;    // you may change this

    public final static String SERVER = "127.0.0.1"; // localhost

    public final static String

        FILE_TO_RECEIVED = "e:/source-downloaded.txt"; // you may change this, I give a

                // different name because i don't want to

                // overwrite the one used by server...

    public final static int FILE_SIZE = 6022386; // file size temporary hard coded

                // should bigger than the file to be downloaded

    public static void main (String [] args ) throws IOException {

        int bytesRead;

        int current = 0;

        FileOutputStream fos = null;

        BufferedOutputStream bos = null;

        Socket sock = null;

        try {

            sock = new Socket(SERVER, SOCKET_PORT);
```

```
System.out.println("Connecting...");

// receive file

byte [] mybytearray = new byte [FILE_SIZE];

InputStream is = sock.getInputStream();

fos = new FileOutputStream(FILE_TO_RECEIVED);

bos = new BufferedOutputStream(fos);

bytesRead = is.read(mybytearray,0,mybytearray.length);

current = bytesRead;

do {

    bytesRead =

        is.read(mybytearray, current, (mybytearray.length-current));

    if(bytesRead >= 0) current += bytesRead;

} while(bytesRead > -1);

bos.write(mybytearray, 0 , current);

bos.flush();

System.out.println("File " + FILE_TO_RECEIVED

    + " downloaded (" + current + " bytes read)");

}
```

```
finally {  
  
    if (fos != null) fos.close();  
  
    if (bos != null) bos.close();  
  
    if (sock != null) sock.close();  
  
}  
  
}  
  
}
```

10. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.

UDP Client

```
import java.io.*;  
import java.net.*;  
public class UDPC  
{  
    public static void main(String[] args)  
    {  
        DatagramSocket skt;  
        try {  
            skt=new DatagramSocket(); String msg= "text message "; byte[] b = msg.getBytes();  
            InetAddress host=InetAddress.getByName("127.0.0.1"); int serverSocket=6788;  
            DatagramPacket request =new DatagramPacket (b,b.length,host,serverSocket); skt.send(request);  
            byte[] buffer =new byte[1000];  
            DatagramPacket reply= new DatagramPacket(buffer,buffer.length); skt.receive(reply);  
            System.out.println("client received:" +new String(reply.getData())); skt.close();  
        }  
        catch(Exception ex)
```



```
{  
}  
}  
}
```

UDP Server

```
import java.io.*; import java.net.*;  
  
public class UDPS  
{  
    public static void main(String[] args)  
    {  
        DatagramSocket skt=null;  
        try  
        {  
            skt=new DatagramSocket(6788); byte[] buffer = new byte[1000];  
            while(true)  
            {  
                DatagramPacket request = new DatagramPacket(buffer,buffer.length);  
                skt.receive(request);  
                String[] message = (new String(request.getData())).split("");  
                byte[] sendMsg= (message[1]+ " server processed").getBytes();  
                DatagramPacket reply = new  
                DatagramPacket(sendMsg,sendMsg.length,request.getAddress(),request.getPort());  
                skt.send(reply);  
            }  
        }  
        catch(Exception ex)  
        {  
        }  
    }  
}
```

11. Write a program for simple RSA algorithm to encrypt and decrypt the data.

Implementation of RSA Algorithm(Encryption and Decryption) in Java

```
import java.math.BigInteger;

import java.util.Random;

import java.io.*;

public class RSA {

    private BigInteger p;

    private BigInteger q;

    private BigInteger N;

    private BigInteger phi;

    private BigInteger e;

    private BigInteger d;

    private int bitlength = 1024;

    private int blocksize = 256;

    //blocksize in byte

    private Random r;

    public RSA() {

        r = new Random();

        p = BigInteger.probablePrime(bitlength, r);

        q = BigInteger.probablePrime(bitlength, r);

        N = p.multiply(q);

        phi = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));

        e = BigInteger.probablePrime(bitlength/2, r);

        while (phi.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(phi) < 0 ) {
```

```
        e.add(BigInteger.ONE);

    }

    d = e.modInverse(phi);
}

public RSA(BigInteger e, BigInteger d, BigInteger N) {

    this.e = e;

    this.d = d;

    this.N = N;

}

public static void main (String[] args) throws IOException {

    RSA rsa = new RSA();

    DataInputStream in=new DataInputStream(System.in);

    String teststring ;

    System.out.println("Enter the plain text:");

    teststring=in.readLine();

    System.out.println("Encrypting String: " + teststring);

    System.out.println("String in Bytes: " + bytesToString(teststring.getBytes()));

    // encrypt

    byte[] encrypted = rsa.encrypt(teststring.getBytes());

    System.out.println("Encrypted String in Bytes: " + bytesToString(encrypted));

    // decrypt

    byte[] decrypted = rsa.decrypt(encrypted);

    System.out.println("Decrypted String in Bytes: " + bytesToString(decrypted));
```

```
        System.out.println("Decrypted String: " + new String(decrypted));
    }

    private static String bytesToString(byte[] encrypted) {

        String test = "";

        for (byte b : encrypted) {

            test += Byte.toString(b);

        }

        return test;

    }

    //Encrypt message

    public byte[] encrypt(byte[] message) {

        return (new BigInteger(message)).modPow(e, N).toByteArray();

    }

    // Decrypt message

    public byte[] decrypt(byte[] message) {

        return (new BigInteger(message)).modPow(d, N).toByteArray();

    }

}
```

12. Write a program for congestion control using leaky bucket algorithm.

filename:Licky.java

```
import java.io.*;
import java.util.*;
```

```
class Queue
{
int q[],f=0,r=0,size;
void insert(int n)
{
Scanner in = new Scanner(System.in);
q=new int[10];
for(int i=0;i<n;i++)
{
System.out.print("\nEnter " + i + " element: ");
int ele=in.nextInt();

if(r+1>10)
{
System.out.println("\nQueue is full \nLost Packet: "+ele);
break;
}
else
{
r++;
q[i]=ele;
}
}
}

void delete()
{
Scanner in = new Scanner(System.in);
Thread t=new Thread();
if(r==0)
```

```
System.out.print("\nQueue empty ");
```

```
        else
        {
            for(int i=f;i<r;i++)
            {
                try
                {
                    t.sleep(1000);
                }
                catch(Exception e){ }
                System.out.print("\nLeaked Packet: "+q[i]);
                f++;
            }
        }
        System.out.println();
    }
}
```

```
class Licky extends Thread
{
    public static void main(String ar[])throws Exception
    {
        Queue q=new Queue();
        Scanner src=new Scanner(System.in);
        System.out.println("\nEnter the packets to be sent:");
        int size=src.nextInt();
        q.insert(size);
        q.delete();
    }
}
```

```
}  
}
```

```
/*
```

OUTPUT

```
bash-3.00$ javac Licky.java  
bash-3.00$ java Licky
```

```
Enter the packets to be sent:  
11
```

```
Enter 0 element: 1
```

```
Enter 1 element: 0
```

```
Enter 2 element: 2
```

```
Enter 3 element: 3
```

```
Enter 4 element: 4
```

```
Enter 5 element: 5
```

```
Enter 6 element: 6
```

```
Enter 7 element: 7
```

```
Enter 8 element: 8
```

```
Enter 9 element: 9
```

```
Enter 10 element: 10
```

```
Queue is full  
Lost Packet: 10
```

```
Leaked Packet: 1
```

```
Leaked Packet: 0
```

```
Leaked Packet: 2
```

```
Leaked Packet: 3
```

```
Leaked Packet: 4
```

Leaked Packet: 5

Leaked Packet: 6

Leaked Packet: 7

Leaked Packet: 8

Leaked Packet: 9

Part –B

8. Write a program for error detecting code using CRC-CCITT (16- bits).

```
import java.io.*;
import java.util.Scanner;

class crcscanner
{
    public static void main(String a[]) throws IOException
    {

        Scanner sc=new Scanner(System.in);
        int[] message;
        int[] gen;
        int[] app_message;
        int[] rem;
        int[] trans_message;
        int message_bits,gen_bits, total_bits;

        System.out.println("\n Enter number of bits in message : ");
        message_bits=sc.nextInt();

        message=new int[message_bits];
        System.out.println("\n Enter message bits : ");
        for(int i=0; i<message_bits; i++)
            message[i]=sc.nextInt();
        System.out.println("\n Enter number of bits in gen : ");
        gen_bits=sc.nextInt();

        gen=new int[gen_bits];
        System.out.println("\n Enter gen bits : ");
        for(int i=0; i<gen_bits; i++)
        {
            gen[i]=sc.nextInt();
        }

        total_bits=message_bits+gen_bits-1;

        app_message=new int[total_bits];
        rem=new int[total_bits];
        trans_message=new int[total_bits];

        for(int i=0;i<message.length;i++)
        {
            app_message[i]=message[i];
        }

        System.out.print("\n Message bits are : ");
        for(int i=0; i< message_bits; i++)
        {
```

```

System.out.print(message[i]);
    }
    System.out.print("\n Generators bits are : ");
    for(int i=0; i< gen_bits; i++)
    {
        System.out.print(gen[i]);
    }

    System.out.print("\n Appended message is : ");
    for(int i=0; i< app_message.length; i++)
    {
        System.out.print(app_message[i]);
    }

    for(int j=0; j<app_message.length; j++)
    {
        rem[j] = app_message[j];
    }

    rem=computeecrc(app_message, gen, rem);

    for(int i=0;i<app_message.length;i++)
    {
        trans_message[i]=(app_message[i]^rem[i]);
    }

    System.out.println("\n Transmitted message from the transmitter is : ");
    for(int i=0;i<trans_message.length;i++)
    {
        System.out.print(trans_message[i]);
    }

    System.out.println("\n Enter received message of "+total_bits+" bits at receiver end : ");
    for(int i=0; i<trans_message.length; i++)
    {
        trans_message[i]=sc.nextInt();
    }
    System.out.println("\n Received message is :");
    for(int i=0; i< trans_message.length; i++)
    {
        System.out.print(trans_message[i]);
    }

    for(int j=0; j<trans_message.length; j++)
    {
        rem[j] = trans_message[j];
    }
    rem=computeecrc(trans_message, gen, rem);
    for(int i=0; i< rem.length; i++)
    {
        if(rem[i]!=0)

```

```

        {
            System.out.println("\n There is Error in the received me ");
            break;
        }
        if(i==rem.length-1)

        System.out.println("\n There is No Error in the received m ");
    }
}

static int[] computecrc(int app_message[],int gen[], int rem[])
{
    int current=0;
    while(true)
    {
        for(int i=0;i<gen.length;i++)
        {
            rem[current+i]=(rem[current+i]^gen[i]);
        }
        while(rem[current]==0 && current!=rem.length-1)
        {
            current++;
        }
        if((rem.length-current)<gen.length)
        {
            break;
        }
    }
    return rem;
}
}

```

8. Write a program to find the shortest path between vertices using bellman-ford algorithm.

```

import java.util.Scanner;

public class BellmanFord
{
    private int D[]; private int num_ver;
    public static final int MAX_VALUE = 999;
    public BellmanFord(int num_ver)
    {
        this.num_ver = num_ver; D = new int[num_ver + 1];
    }
    public void BellmanFordEvaluation(int source, int A[][])
    {

```

```

for (int node = 1; node <= num_ver; node++)
{
D[node] = MAX_VALUE;

}
D[source] = 0;
for (int node = 1; node <= num_ver - 1; node++)
{
for (int sn = 1; sn <= num_ver; sn++)
{
for (int dn = 1; dn <= num_ver; dn++)
{
if (A[sn][dn] != MAX_VALUE)
{
if (D[dn] > D[sn] + A[sn][dn])
D[dn] = D[sn] + A[sn][dn];
}
}
}
}
for (int sn = 1; sn <= num_ver; sn++)
{
for (int dn = 1; dn <= num_ver; dn++)
{
if (A[sn][dn] != MAX_VALUE)
{
if (D[dn] > D[sn] + A[sn][dn])
System.out.println("The Graph contains negative egde cycle");
}
}
}
for (int vertex = 1; vertex <= num_ver; vertex++)
{
System.out.println("distance of source " + source + " to " + vertex + " is " + D[vertex]);
}

```

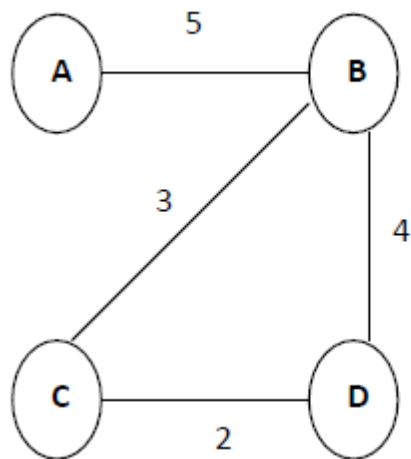
```

}
public static void main(String[ ] args)
{
int num_ver = 0; int source;
Scanner scanner = new Scanner(System.in); System.out.println("Enter the number of
vertices"); num_ver = scanner.nextInt();
int A[][] = new int[num_ver + 1][num_ver + 1]; System.out.println("Enter the adjacency
matrix"); for (int sn = 1; sn <= num_ver; sn++)
{
for (int dn = 1; dn <= num_ver; dn++)
{

A[sn][dn] = scanner.nextInt(); if (sn == dn)
{
A[sn][dn] = 0; continue;
}
if (A[sn][dn] == 0)
{
A[sn][dn] = MAX_VALUE;
}
}
}
System.out.println("Enter the source vertex"); source = scanner.nextInt();
BellmanFord b = new BellmanFord (num_ver); b.BellmanFordEvaluation(source, A);
scanner.close();
}
}

```

Input graph:



Output:

Enter the number of vertices

4

Enter the adjacency matrix

0 5 0 0

5 0 3 4

0 3 0 2

0 4 2 0

Enter the source vertex

2

distance of source 2 to 1 is 5

distance of source 2 to 2 is 0

distance of source 2 to 3 is 3

distance of source 2 to 4 is 4

9.Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.

Server Program

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;

public class SimpleFileServer {

    public final static int SOCKET_PORT = 13267; // you may change this
    public final static String FILE_TO_SEND = "e:/source1.txt"; // you may change this

    public static void main (String [] args ) throws IOException {
        FileInputStream fis = null;
        BufferedReader bis = null;
        OutputStream os = null;
        ServerSocket servsock = null;
        Socket sock = null;
        try {
            servsock = new ServerSocket(SOCKET_PORT);
            while (true) {
                System.out.println("Waiting...");
                try {
                    sock = servsock.accept();
                    System.out.println("Accepted connection : " + sock);
                    // send file
                    File myFile = new File (FILE_TO_SEND);
                    byte [] mybytearray = new byte [(int)myFile.length()];
                    fis = new FileInputStream(myFile);
```



```

        bis = new BufferedInputStream(fis);
        bis.read(mybytearray,0,mybytearray.length);
        os = sock.getOutputStream();
        System.out.println("Sending " + FILE_TO_SEND + "(" + mybytearray.length + "
bytes)");
        os.write(mybytearray,0,mybytearray.length);
        os.flush();
        System.out.println("Done.");
    }
    finally {
        if (bis != null) bis.close();
        if (os != null) os.close();
        if (sock!=null) sock.close();
    }
}
}
finally {
    if (servsock != null) servsock.close();
}
}
}

```

Client Program

```

import java.io.BufferedOutputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.net.Socket;


public class SimpleFileClient {

```

```
public final static int SOCKET_PORT = 13267;    // you may change this

public final static String SERVER = "127.0.0.1"; // localhost

public final static String

    FILE_TO_RECEIVED = "e:/source-downloaded.txt"; // you may change this, I give a

                // different name because i don't want to

                // overwrite the one used by server...

public final static int FILE_SIZE = 6022386; // file size temporary hard coded

                // should bigger than the file to be downloaded

public static void main (String [] args ) throws IOException {

    int bytesRead;

    int current = 0;

    FileOutputStream fos = null;

    BufferedOutputStream bos = null;

    Socket sock = null;

    try {

        sock = new Socket(SERVER, SOCKET_PORT);

        System.out.println("Connecting...");

        // receive file

        byte [] mybytearray = new byte [FILE_SIZE];

        InputStream is = sock.getInputStream();
```

```
fos = new FileOutputStream(FILE_TO_RECEIVED);

bos = new BufferedOutputStream(fos);

bytesRead = is.read(mybytearray,0,mybytearray.length);

current = bytesRead;


do {

    bytesRead =

        is.read(mybytearray, current, (mybytearray.length-current));

    if(bytesRead >= 0) current += bytesRead;

} while(bytesRead > -1);


bos.write(mybytearray, 0 , current);

bos.flush();

System.out.println("File " + FILE_TO_RECEIVED

    + " downloaded (" + current + " bytes read)");

}

finally {

    if (fos != null) fos.close();

    if (bos != null) bos.close();

    if (sock != null) sock.close();

}

}
```

```
}
```

10. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.

UDP Client

```
import java.io.*;
import java.net.*;
public class UDPC
{
    public static void main(String[] args)
    {
        DatagramSocket skt;
        try {
            skt=new DatagramSocket(); String msg= "text message "; byte[] b = msg.getBytes();
            InetAddress host=InetAddress.getByName("127.0.0.1"); int serverSocket=6788;
            DatagramPacket request =new DatagramPacket (b,b.length,host,serverSocket);
            skt.send(request);
            byte[] buffer =new byte[1000];
            DatagramPacket reply= new DatagramPacket(buffer,buffer.length); skt.receive(reply);
            System.out.println("client received:" +new String(reply.getData())); skt.close();
        }
        catch(Exception ex)
        {
        }
    }
}
```

UDP Server

```
import java.io.*; import java.net.*;
public class UDPS
{
    public static void main(String[] args)
    {
        DatagramSocket skt=null;
        try
```

```

{
    skt=new DatagramSocket(6788); byte[] buffer = new byte[1000];
    while(true)
    {
        DatagramPacket request = new DatagramPacket(buffer,buffer.length);
        skt.receive(request);
        String[] message = (new String(request.getData())).split("");
        byte[] sendMsg= (message[1]+ " server processed").getBytes();
        DatagramPacket reply = new
        DatagramPacket(sendMsg,sendMsg.length,request.getAddress(),request.getPort());
        skt.send(reply);
    }
}
catch(Exception ex)
{
}
}
}

```

13. Write a program for simple RSA algorithm to encrypt and decrypt the data.

Implementation of RSA Algorithm(Encryption and Decryption) in Java

```

import java.math.BigInteger;

import java.util.Random;

import java.io.*;

public class RSA {

    private BigInteger p;

    private BigInteger q;

    private BigInteger N;

    private BigInteger phi;

    private BigInteger e;

```

```

private BigInteger d;

private int bitlength = 1024;

private int blocksize = 256;

//blocksize in byte

private Random r;

public RSA() {

    r = new Random();

    p = BigInteger.probablePrime(bitlength, r);

    q = BigInteger.probablePrime(bitlength, r);

    N = p.multiply(q);

    phi = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));

    e = BigInteger.probablePrime(bitlength/2, r);

    while (phi.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(phi) < 0
) {

        e.add(BigInteger.ONE);

    }

    d = e.modInverse(phi);

}

public RSA(BigInteger e, BigInteger d, BigInteger N) {

    this.e = e;

    this.d = d;

    this.N = N;

}

public static void main (String[] args) throws IOException {

    RSA rsa = new RSA();

    DataInputStream in=new DataInputStream(System.in);

```

```

String teststring ;

System.out.println("Enter the plain text:");

teststring=in.readLine();

System.out.println("Encrypting String: " + teststring);

System.out.println("String in Bytes: " + bytesToString(teststring.getBytes()));

// encrypt

byte[] encrypted = rsa.encrypt(teststring.getBytes());

System.out.println("Encrypted String in Bytes: " + bytesToString(encrypted));

// decrypt

byte[] decrypted = rsa.decrypt(encrypted);

System.out.println("Decrypted String in Bytes: " +
bytesToString(decrypted));

System.out.println("Decrypted String: " + new String(decrypted));

}

private static String bytesToString(byte[] encrypted) {

String test = "";

for (byte b : encrypted) {

test += Byte.toString(b);

}

return test;

}

//Encrypt message

public byte[] encrypt(byte[] message) {

return (new BigInteger(message)).modPow(e, N).toByteArray();

}

```

```

// Decrypt message

public byte[] decrypt(byte[] message) {

    return (new BigInteger(message)).modPow(d, N).toByteArray();

}

}

```

14. Write a program for congestion control using leaky bucket algorithm.

filename:Licky.java

```

import java.io.*;
import java.util.*;

class Queue
{
    int q[],f=0,r=0,size;
    void insert(int n)
    {
        Scanner in = new Scanner(System.in);
        q=new int[10];
        for(int i=0;i<n;i++)
        {
            System.out.print("\nEnter " + i + " element: ");
            int ele=in.nextInt();

            if(r+1>10)
            {
                System.out.println("\nQueue is full \nLost Packet: "+ele);
                break;
            }
            else
            {
                r++;
                q[i]=ele;
            }
        }
    }
}

```



```
}  
}  
}
```

```
void delete()
```

```
{  
    Scanner in = new Scanner(System.in);  
    Thread t=new Thread();  
    if(r==0)  
        System.out.print("\nQueue empty ");  
  
        else  
    {  
        for(int i=f;i<r;i++)  
        {  
            try  
            {  
                t.sleep(1000);  
            }  
            catch(Exception e){ }  
            System.out.print("\nLeaked Packet: "+q[i]);  
            f++;  
        }  
    }  
    System.out.println();  
}
```

```
class Licky extends Thread
```

```
{  
    public static void main(String ar[])throws Exception  
    {  
        Queue q=new Queue();
```

```
Scanner src=new Scanner(System.in);
System.out.println("\nEnter the packets to be sent:");
int size=src.nextInt();
q.insert(size);
q.delete();

}

}
```

/*

OUTPUT

```
bash-3.00$ javac Licky.java
bash-3.00$ java Licky
```

```
Enter the packets to be sent:
11
```

```
Enter 0 element: 1
```

```
Enter 1 element: 0
```

```
Enter 2 element: 2
```

```
Enter 3 element: 3
```

```
Enter 4 element: 4
```

```
Enter 5 element: 5
```

```
Enter 6 element: 6
```

```
Enter 7 element: 7
```

```
Enter 8 element: 8
```

```
Enter 9 element: 9
```

```
Enter 10 element: 10
```

```
Queue is full
Lost Packet: 10
```

```
Leaked Packet: 1
```

```
Leaked Packet: 0
```

Leaked Packet: 2

Leaked Packet: 3

Leaked Packet: 4

Leaked Packet: 5

Leaked Packet: 6

Leaked Packet: 7

Leaked Packet: 8

Leaked Packet: 9