18CSL57 COMPUTER NETWORK LABORATORY

g.srinivasachar

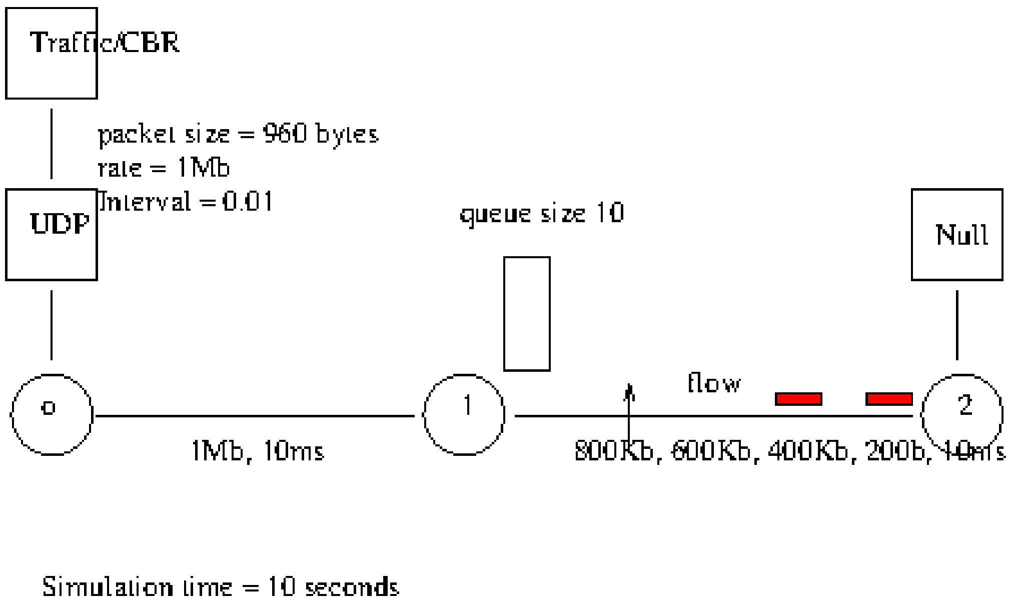
October 4, 2021

Contents

1 A.1 Network of three nodes

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.

DESIGN:



CODE:

```
# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 1.tcl
# Three nodes network & measure packets dropped


set ns [new Simulator]
set tf [open out.tr w]
set nf [open out.nam w]

$ns trace-all $tf
$ns namtrace-all $nf

# Create nodes
set num 3
```

```
for {set i 0} {$i < $num} {incr i} { set node($i) [$ns node]}

# Create links
$ns duplex-link $node(0) $node(1) 1Mb 10ms DropTail
$ns duplex-link $node(1) $node(2) 400Kb 10ms DropTail ;#800, 600, 400, 200

# Create queues 0.5*PI radians
$ns duplex-link-op $node(1) $node(2) queuePos 0.5
$ns queue-limit $node(1) $node(2) 10

# Label nodes
$node(0) label "UDP"
$node(2) label "Null"

# Label flows
$ns color 0 Red

# Create connections
set udp [$ns create-connection UDP $node(0) Null $node(2) 0]
set cbr [$udp attach-app Traffic/CBR]

# Traffic
$cbr set packetSize_ 960
$cbr set rate_ 1Mb
$cbr set interval_ 0.01 ;# choose 0.01 only; 0.001, 0.01, 0.1

$ns at 0.0 "$cbr start"
$ns at 10 "finish"

proc finish {} {
        global ns tf nf
        $ns flush-trace
        close $tf
        close $nf
        exit 0
}

# Start simulation
$ns run


# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 1.awk
# Count dropped packets

BEGIN {
    count=0;
}
{
    if($1=="d") count++;
}
END {
    printf("Number of packets dropped is %d\n", count);
}
```

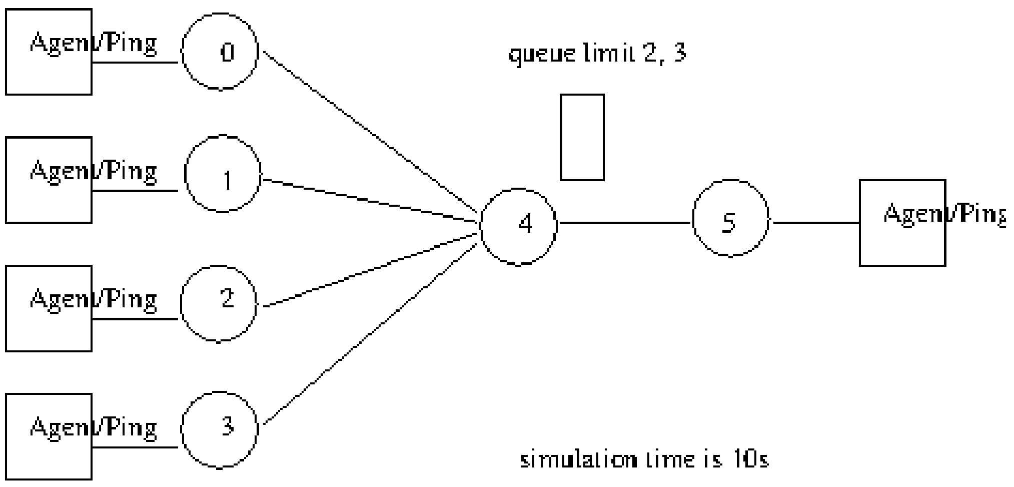
RUN:

```
ns 1.tcl
nam out.nam
awk -f 1.awk out.tr
BW(Kb/s) 800 600 400 200
Dropped  0   210 470 730
```

2 A.2 Ping traffic

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.

DESIGN:



CODE:

```
# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 2.tcl
# Simulate Ping & count dropped packets due to congestion

set ns [new Simulator]
set tf [open out.tr w]
set nf [open out.nam w]
```

```
$ns trace-all $tf
$ns namtrace-all $nf

# Create nodes
set num 6
for {set i 0} {$i < $num} {incr i} {
    set node($i) [$ns node]
}

# Create links
$ns duplex-link $node(0) $node(4) 1Mb 10ms DropTail
$ns duplex-link $node(1) $node(4) 1Mb 10ms DropTail
$ns duplex-link $node(2) $node(4) 1Mb 10ms DropTail
$ns duplex-link $node(3) $node(4) 1Mb 10ms DropTail
$ns duplex-link $node(4) $node(5) 1Mb 10ms DropTail

# Create queue
$ns duplex-link-op $node(4) $node(5) queuePos 0.5
$ns queue-limit $node(4) $node(5) 3 ;# different from normal 3, 2

# Label flows
$ns color 1 "red"
$ns color 2 "blue"
$ns color 3 "green"
$ns color 4 "yellow"
$ns color 5 "orange"

# Define a 'recv' function for the class 'Agent/Ping'
Agent/Ping instproc recv {from rtt} {
    $self instvar node_
    puts "node [$node_ id] received ping answer from $from with round-trip-time $rt
}
```

```
# Create connections
set p0 [$ns create-connection Ping $node(0) Ping $node(5) 1]
set p1 [$ns create-connection Ping $node(1) Ping $node(5) 2]
set p2 [$ns create-connection Ping $node(2) Ping $node(5) 3]
set p3 [$ns create-connection Ping $node(3) Ping $node(5) 4]
set p5 [$ns create-connection Ping $node(5) Ping $node(4) 5]

# Schedule events
for { set i 0} {$i < 10} {incr i} {
    for {set j 0} {$j < 10} {incr j} {
        $ns at [expr $i+.1+$j/10] "$p0 send"
        $ns at [expr $i+.1+$j/10] "$p5 send"
        $ns at [expr $i+.2+$j/10] "$p1 send"
        $ns at [expr $i+.3+$j/10] "$p2 send"
        $ns at [expr $i+.4+$j/10] "$p3 send"
        $ns at [expr $i+.5+$j/10] "$p5 send"
    }
}
$ns at 10 "finish"

proc finish {} {
        global ns tf nf
        $ns flush-trace
        close $tf
        close $nf
        exit 0
}

# Start simulation
$ns run


# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 2.awk
# Count dropped packets due to congestion

BEGIN {
    count=0;
}

{
    if($1=="d") count++;
}
END {
    printf("total no of packets dropped due to cngestion : %d\n", count);
}
```

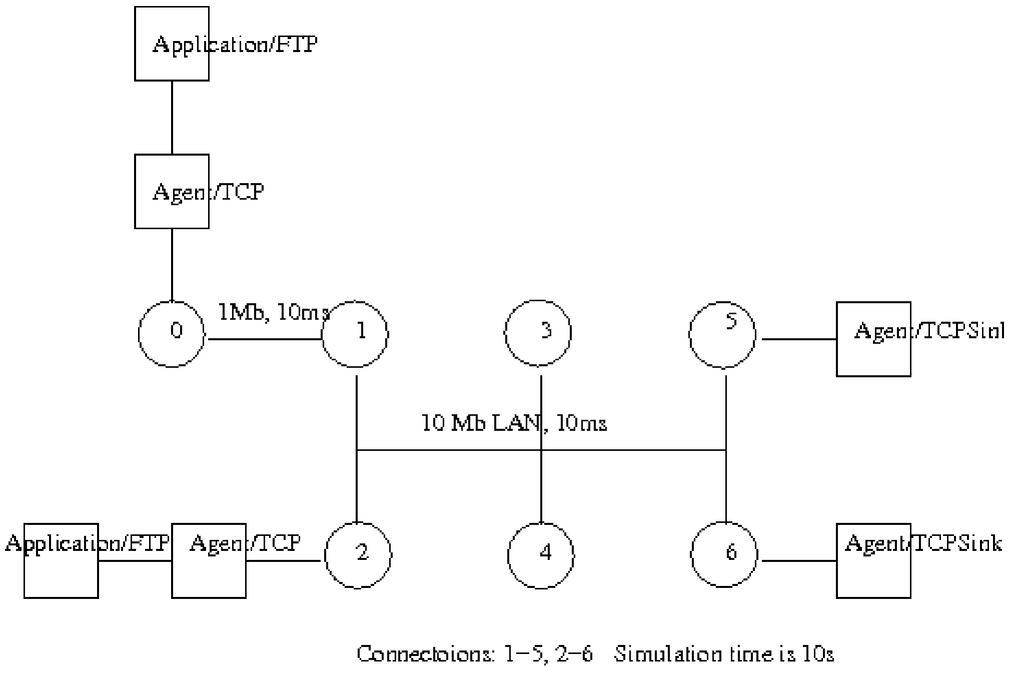
RUN:

```
ns 2.tcl
nam out.nam
awk -f 2.awk out.tr
1. qsize(n4,n5) = 2, 100 packets dropped due to congestion
2. qsize(n4,n5) = 3, 89  packets dropped
```

3 A.3 LAN

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

DESIGN



CODE:

```
# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 3.tcl
# LAN simulation (congestion window size with time)
```

```
set ns [new Simulator]
set tf [open out.tr w]
set nf [open out.nam w]

$ns trace-all $tf
$ns namtrace-all $nf

# Create nodes
set node(0) [$ns node]

set num 6
for {set i 1} {$i <= $num} {incr i} {
    set node($i) [$ns node]
    lappend nodelist $node($i)
}

# create LAN and links
$ns make-lan $nodelist 10Mb 10ms LL Queue/DropTail Mac/802_3 Channel

$ns duplex-link $node(0) $node(1) 1Mb 10ms DropTail
$ns duplex-link-op $node(0) $node(1) queuePos 0.5
$ns duplex-link-op $node(0) $node(1) orient right

# Create connections
set tcp0 [$ns create-connection TCP $node(0) TCPSink $node(5) 0]
set tcp1 [$ns create-connection TCP $node(2) TCPSink $node(6) 0]

set ftp0 [$tcp0 attach-app FTP]
set ftp1 [$tcp1 attach-app FTP]

#create error model for 1/1000 packets between node(0) and node(1)
set err [new ErrorModel]
$err set rate_ 0.001  ;# 0.001, 0.005, 0.010
```

```
$ns lossmodel $err $node(0) $node(1)

$tcp0 attach $tf
$tcp0 trace cwnd_

$tcp1 attach $tf
$tcp1 trace cwnd_

$ns at 0.1 "$ftp0 start"
$ns at 0.2 "$ftp1 start"

$ns at 10 "finish"

proc finish {} {
        global ns tf nf
        $ns flush-trace
        close $tf
    close $nf
        exit 0
}

# Start simulator
$ns run


# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 3.awk
# Plot conestion window X time

BEGIN{
}
{
    if($6=="cwnd_")
    {
        if ($2 == 0 && $4 == 5) printf("%4.2f\t%4.2f\t\n",$1,$7); # $1=time, $7=cu
#       if ($2 == 2 && $4 == 6) printf("%4.2f\t%4.2f\t\n",$1,$7);
    }
}
END{
    puts "DONE";
}
```

RUN:

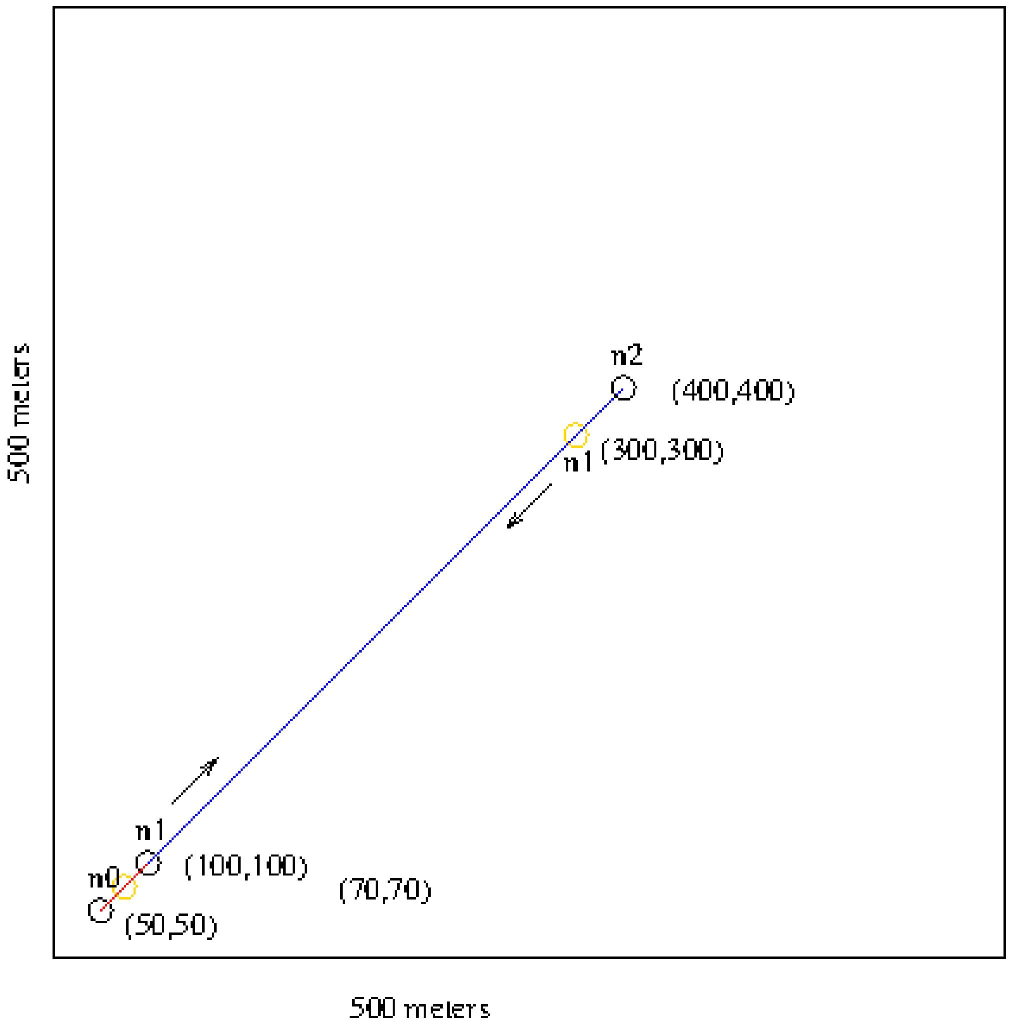
```
ns 3.tcl
nam out.nam
awk -f 3.awk out.tr > out.txt
xgraph out.txt
modify awk script to use another tcp connection
```

4 A.4 Simple ESS

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

DESIGN:

A service set (also known as extended service set or ESS) is a group of wireless network devices which are identified by the same SSID (service set identifier).



Simulation time 25 s; n1 moves towards n2 at 10s; retracts back at 20 s.

CODE:

```
# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 4.tcl
# Wireless LAN simulation
```

```
set ns [new Simulator]
set tf [open out.tr w]
set nf [open out.nam w]

$ns trace-all $tf
$ns namtrace-all-wireless $nf 500 500

set topo [new Topography]
$topo load_flatgrid 500 500

$ns node-config \
    -adhocRouting DSDV \
    -llType       LL \
    -macType      Mac/802_11 \
    -ifqType      Queue/DropTail \
    -ifqLen       10 \
    -phyType      Phy/WirelessPhy \
    -propType     Propagation/TwoRayGround \
    -antType      Antenna/OmniAntenna \
    -topoInstance $topo \
    -agentTrace   ON \
    -routerTrace  ON \
    -macTrace     ON \
    -channel      [new Channel/WirelessChannel]

create-god 3 ;# General Operations Director

set num 3
for {set i 0} {$i < $num} {incr i} {
    set node($i) [$ns node]
}
```

```
$node(0) label "TCP"
$node(1) label "TCPSink, TCP"
$node(2) label "TCPSink"

$node(0) set X_ 50
$node(0) set Y_ 50
$node(0) set Z_ 0

$node(1) set X_ 100
$node(1) set Y_ 100
$node(1) set Z_ 0

$node(2) set X_ 400
$node(2) set Y_ 400
$node(2) set Z_ 0

# Create connections
set tcp0 [$ns create-connection TCP $node(0) TCPSink $node(1) 1]
set tcp1 [$ns create-connection TCP $node(1) TCPSink $node(2) 2]

$ns color 1 "red"
$ns color 2 "blue"

set ftp0 [$tcp0 attach-app FTP]
set ftp1 [$tcp1 attach-app FTP]

$ns at 0 "$node(0) setdest 50 50 100"
$ns at 0 "$node(1) setdest 100 100 100"
$ns at 0 "$node(2) setdest 400 400 100"

$ns at 1 "$ftp0 start"
$ns at 1 "$ftp1 start"
```

```
$ns at 10 "$node(1) setdest 300 300 100"
$ns at 15 "$node(1) setdest 100 100 100"

$ns at 20 "finish"

proc finish {} {
        global ns tf nf
        $ns flush-trace
        close $tf
        close $nf
        exit 0
}

# Start simulation
$ns run


# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 4.awk
# Wireless LAN link performance

BEGIN {
        cnt_1 = pkt_1 = 0;
        cnt_2 = pkt_2 = 0;
}

{
        if($1=="r" && $3=="_1_" && $4=="AGT")
        {
                cnt_1++;
                pkt_1 += $8
                t1 = $2;

        }
        if($1=="r" && $3=="_2_" && $4=="AGT")
        {
                cnt_2++;
                pkt_2 += $8;

                t2 = $2;
        }
}

END {
        printf("node(0) to node(1) link performance : %6.2f Mbps\n", (cnt_1*pkt_1*
        printf("node(1) to node(2) link performance : %6.2f Mbps\n", (cnt_2*pkt_2*
}
```

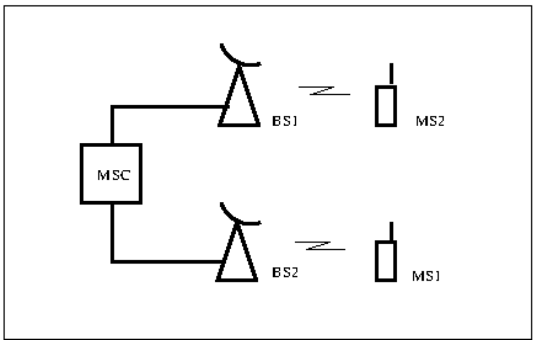
RUN:

```
ns 4.tcl
nam out.nam
awk -f 4.awk out.tr
The throughput from node(0) to node(1): 643.49 Mb/s
The throughput from node(1) to node(2):   8.31 Mb/s
```

5 A.5 GSM

Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.

DESIGN



CODE:

```
# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 5.tcl
# GSM Performance

set ns [new Simulator]
set tf [open out.tr w]
set nf [open out.nam w]

$ns trace-all $tf
$ns namtrace-all $nf
```

```
# Create network nodes

set node(ms1) [$ns node]
set node(bs1) [$ns node]
set node(msc) [$ns node]
set node(bs2) [$ns node]
set node(ms2) [$ns node]

# Create links

$ns duplex-link $node(ms1) $node(bs1) 1Mb 1ms DropTail
$ns duplex-link $node(bs1) $node(msc) 1Mb 10ms DropTail
$ns duplex-link $node(msc) $node(bs2) 1Mb 10ms DropTail
$ns duplex-link $node(bs2) $node(ms2) 1Mb 1ms DropTail

puts "Cell Topology"

$ns bandwidth $node(ms1) $node(bs1) 9.6Kb simplex      ;#uplink
$ns bandwidth $node(bs1) $node(ms1) 9.6Kb simplex      ;#downlink
$ns insert-delayer $node(ms1) $node(bs1) [new Delayer]

# Create connection & attach applications

set tcp [$ns create-connection TCP $node(ms1) TCPSink $node(ms2) 0]
set ftp [$tcp attach-app FTP]

proc finish {} {
        global ns tf nf
        $ns flush-trace
        close $tf
        close $nf
        exit 0
}
```

```
$ns at 0.1 "$ftp start"
$ns at 20 "finish"

$ns run

#xgraph -P -bar -x TIME -y DATA gsm.xg


# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 5.awk
# GSM performance

BEGIN {
    packets_sent = 0;
    packets_acks = 0;
}
{
    if($1 == "r" && $5 == "tcp")
    {
        packets_sent++;
    }
    if($1 == "r" && $5 == "ack")
    {
        packets_acks++;
    }
}
END {
    printf("Packets sent = %d\n", packets_sent);
    printf("Packets acks = %d\n", packets_acks);
}
```

RUN:

ns 5.tcl

This programs must have,

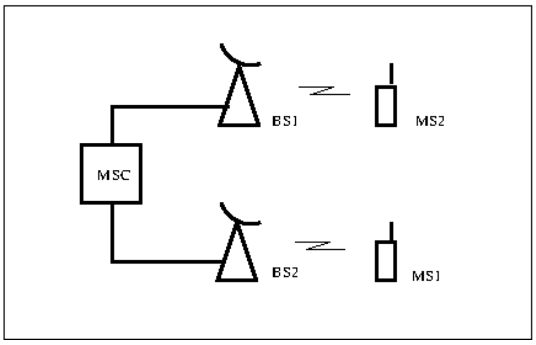
1. the files getrc, raw2xg, xg2gp.awk in ".'
2. the file getopts.pl in /etc/perl

```
nam out.nam
awk -f 4.awk out.tr
Packets sent = 92
Packets acks = 92
```

6 A.6 CDMA

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

DESIGN



CODE:

```
# Author: G. Srinivasachar
# Date:   3/6/16
#
# File 6.tcl
# CDMA Performance

set ns [new Simulator]
set tf [open out.tr w]
set nf [open out.nam w]

$ns trace-all $tf
$ns namtrace-all $nf
```

```
set node(ms1) [$ns node]
set node(bs1) [$ns node]
set node(msc) [$ns node]
set node(bs2) [$ns node]
set node(ms2) [$ns node]


$ns duplex-link $node(ms1) $node(bs1) 1Mb 1ms DropTail
$ns duplex-link $node(bs1) $node(msc) 1Mb 10ms DropTail
$ns duplex-link $node(msc) $node(bs2) 1Mb 10ms DropTail
$ns duplex-link $node(bs2) $node(ms2) 1Mb 1ms DropTail

puts "Cell Topology"


$ns bandwidth $node(ms1) $node(bs1) 64Kb simplex ;#uplink
$ns bandwidth $node(bs1) $node(ms1) 384Kb simplex ;#downlink
$ns insert-delayer $node(ms1) $node(bs1) [new Delayer]


set tcp1 [$ns create-connection TCP $node(ms1) TCPSink $node(ms2) 0]
set ftp1 [$tcp1 attach-app FTP]
$ns at 0.1 "$ftp1 start"

proc finish {} {
        global ns tf nf
        $ns flush-trace
        close $tf
        close $nf
        exit 0
}
```

```
$ns at 20 "finish"
$ns run
```

RUN:

ns 5.tcl

This programs must have,

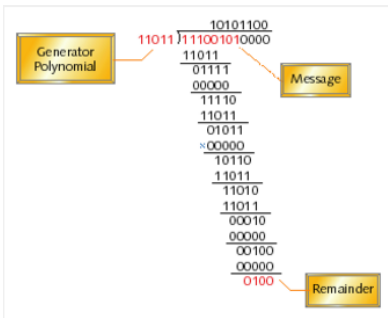
1. the files getrc, raw2xg, xg2gp.awk in '.'
2. the file getopts.pl in /etc/perl

```
nam out.nam
awk -f 4.awk out.tr
Packets sent = 612
Packets acks = 612
```

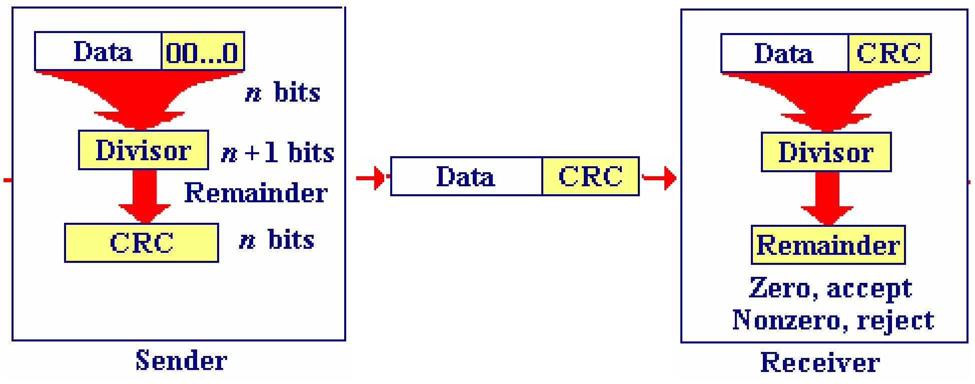
7 B.1 CRC

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

DESIGN:



```
               3   2   1   0   Bits

              +---+---+---+---+
     Pop! <-- |   |   |   |   | <----- Augmented message
              +---+---+---+---+
           1    0   1   1   1   = The Poly

http://www.sunshine2k.de/articles/coding/crc/understanding_crc.html

Load the register with zero bits.
Augment the message by appending W zero bits to the end of it.
While (more message bits)
   Begin
   Shift the register left by one bit, reading the next bit of the
      augmented message into register bit position 0.
   If (a 1 bit popped out of the register during shifting)
      Register = Register XOR Poly.
   End
The register now contains the remainder.
```



CODE:

```java
import java.util.Scanner;

public class CRC {
    int W;
    String P;
    String checksum;
    String message;

    CRC()
    {
        W = 16;
        P = "10001000000100001"; //0, 5, 12, 16
    }

    void crc()
    {
        String msg = message + "0000000000000000";  // augmented message
        char[] rem = new char[P.length()];

        for (int i=0; i < msg.length(); i++)
        {
            // take the next digit
            rem[W] = msg.charAt(i);

            // compute the reminder and shift left
            boolean xor = rem[0] == '1';
            for (int j=1; j <= W; j++)
            {
                if (xor) rem[j] = (rem[j]==P.charAt(j)) ? '0':'1';
                rem[j-1] = rem[j];
            }
        }
        checksum = String.valueOf(rem).substring(0, W); //excludes W
    }

    void input()
    {
        Scanner scanner = new Scanner(System.in);

        System.out.print("MESSAGE:");
        message = scanner.next();

        scanner.close();
    }

    void output()
    {
        System.out.println("Checksum:"+checksum);
    }
    public static void main(String[] args)
    {
        CRC crc = new CRC();

        crc.input();
        crc.crc();
        crc.output();
    }
}
```

RUN:

```
javac CRC.java
java  CRC

MESSAGE: 0101
Checksum: 0101000010100101

MESSAGE: 1011101
Checksum: 1000101101011000

MESSAGE: MESSAGE + Checksum.
Checksum: 0000000000000000

MESSAGE: (MESSAGE + CHecksum) error bits
Checksum: NOT Zero
```

8 B.2 Bellman Ford

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

DESIGN

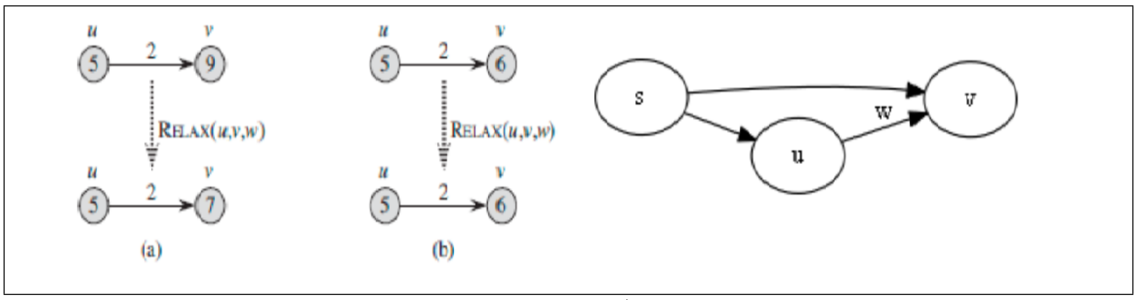
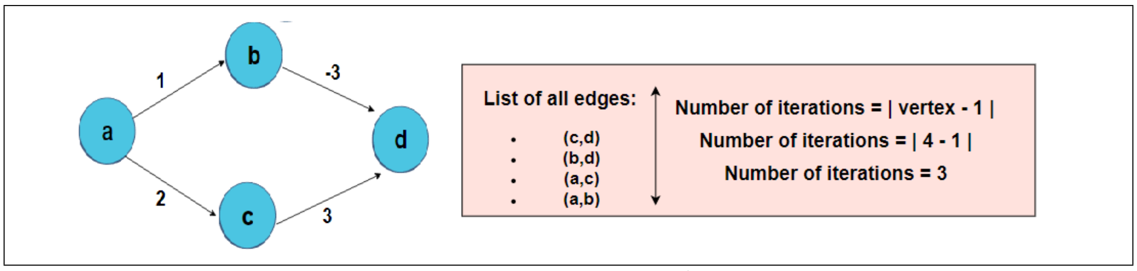


Figure 1: RELAX Edge



List of all edges:
- (c,d)
- (b,d)
- (a,c)
- (a,b)

Number of iterations = | vertex - 1 |
Number of iterations = | 4 - 1 |
Number of iterations = 3

Figure 2: Example

```
BELLMAN-FORD(G, w, s)

// Initialization
for each vertex v ∈ G.V
    v.d = ∞
    v.π = NIL
s.d = 0

// Relaxation
for i = 1 to |G.V|-1
  for each edge(u,v) ∈ G.E
```

```
            if v.d > u.d + w(u,v)
                  v.d = u.d + w(u,v)
                  v.$\pi$ = u

      for each edge(u,v) $\in$ G.E
            if v.d > u.d + w(u,v)
                  return FALSE

      return TRUE;
```

CODE:

```java
import java.util.Scanner;

public class BellmanFord {
    int n;
    int[][] a;
    int[] d;
    int[] p;
    int s;
    public final static int INFTY=999;

    BellmanFord(int n)
    {
        this.n = n;

        a = new int[n][n];
        d = new int[n];
        p = new int[n];
    }

    void bellmanFord()
    {
        // Initialization
        for(int i=0; i < n; i++)
            {
                d[i] = a[s][i];
                p[i] = a[s][i] == INFTY ? -1 : s;
            }
        p[s] = -1;

        for(int i=0; i < n-1; i++)
            {
                // Relax all edges iteratively (n-1) times
                for(int u=0; u < n; u++)
                    {
                        for(int v=0; v < n; v++)
                            {
                                if(d[v] > d[u]+a[u][v])
                                    {
                                        d[v] = d[u]+a[u][v];
                                        p[v] = u;
                                    }
                            }
                    }
            }
    }

    void input(Scanner scanner)
    {
        System.out.println("Enter G: ");

        for(int i=0; i<n; i++)
            {^^I
                for(int j=0; j<n; j++)
                    {
```

```
                    a[i][j] = scanner.nextInt();
                    if (i != j && a[i][j] == 0) a[i][j] = INFTY;
                }
            }

        System.out.print("Enter the source vertex: ");
        s = scanner.nextInt();

        scanner.close();
    }

    void path(int v)
    {
        if (v == -1) return;

        path(p[v]);
        System.out.print("."+v);
    }

    void output()
    {
        int i;

        for(i=0; i < n; i++)
            {
                System.out.print("d(" + s + "," + i + ")=" + d[i]+" :p");
                path(i);
                System.out.println();
            }
    }

    public static void main(String[] args)
    {
        int ^^I^^In;
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter n: ");
        n = scanner.nextInt();

        BellmanFord bf = new BellmanFord(n);

        bf.input(scanner);
        bf.bellmanFord();
        bf.output();
    }
}
```

RUN:

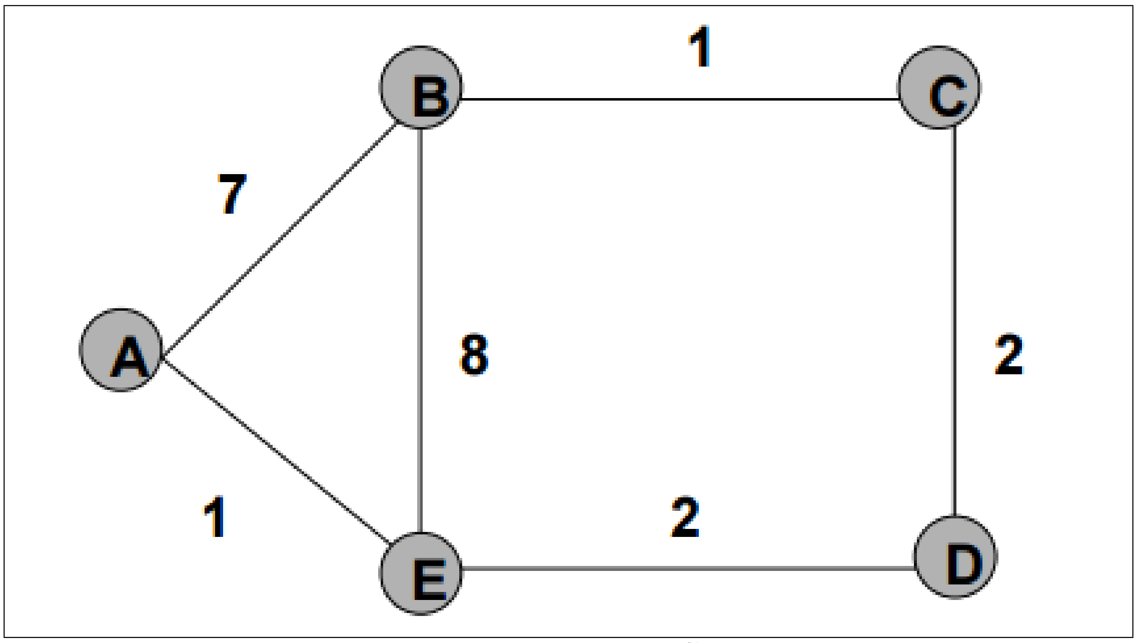


Figure 3: Input G

```
javac BellmanFord.java
```

```
java  BellmanFord

INPUT:

Enter n: 5
Enter a:
0  7  0  0  1
7  0  1  0  8
0  1  0  2  0
0  0  2  0  2
1  8  0  2  0

Dest Dist path...
1    6    4.3.2.1.
2    5    4.3.2.
3    3    4.3.
4    1    4.
```

9 B.3 File transfer using TCP

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. Implement the above program using as message queues or FIFOs as IPC channels.

DESIGN:

```
Client                                         Server

1. OutputStream -->\                                        /--> 2. InputStream -->
                     Socket <--> network <--> ServerSocket                          |
4. InputStream  <--/                                        \<--3. OutputStream <--
```

SERVER

- Create a server socket and bind to a specific address
- Wait for client connection
- Create input and output stream for the client socket
- Read file name from the input stream
- Read all lines from the file
- Write the lines to output stream

CLIENT

- Create a socket with the server address

- Create input and output stream for the socket
- Read file name from the console
- Write file name to the socket
- Read, in a loop, the lines from server until the line is "stop"

CODE:

TcpServer.java

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.nio.file.*;

public class TcpServer
{
    void server() throws Exception
    {
        System.out.println("Server waiting for connection from client");
        ServerSocket serverSocket = new ServerSocket(3333);

        Socket socket = serverSocket.accept();
        DataInputStream din = new DataInputStream(socket.getInputStream());
        DataOutputStream dout = new DataOutputStream(socket.getOutputStream());

        String fileName = din.readUTF();
        List<String> lines = Files.readAllLines(Paths.get(fileName));

        for (int i=0; i < lines.size(); i++)
            {
                System.out.println("server: "+lines.get(i));
                dout.writeUTF(lines.get(i));
            }

        din.close();
        dout.close();

        serverSocket.close();
        socket.close();
    }

    public static void main(String[] args) throws Exception
    {
        TcpServer ts = new TcpServer();
        ts.server();
    }
}
```

TcpClient.java

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

public class TcpClient {
    void client() throws Exception
    {
        Socket socket = new Socket("localhost",3333);

        DataInputStream din=new DataInputStream(socket.getInputStream());
        DataOutputStream dout=new DataOutputStream(socket.getOutputStream());
        System.out.print("Enter filename:");

        Scanner scanner = new Scanner(System.in);
        String fileName = scanner.next();

        dout.writeUTF(fileName);

        String message;
        do
            {
                message = din.readUTF();
                System.out.println("Client: " + message);
            }
        while(!message.equals("stop"));

        scanner.close();

        din.close();
        dout.close();

        socket.close();
    }

    public static void main(String[] args) throws Exception
    {
        TcpClient tc = new TcpClient();
        tc.client();
    }
}
```

f.txt

```
1
2
3
4
stop
```

RUN:

javac TcpServer.java TcpClient.java

```
>java TcpServer                              >java TcpClient
Server waiting for connection from client    Enter filename:f.txt
server: 1                                    Client: 1
server: 2                                    Client: 2
server: 3                                    Client: 3
server: 4                                    Client: 4
server: stop                                 Client: stop
>                                            >
```

10 B.4 Data transfer using UDP

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

DESIGN:

SERVER

- Create a datagram socket and bind to a specific address
- Create a datagram packet
- Receive datagram packet and extract the client address
- Read line from the console and write to the socket until "stop condition"

CLIENT

- Create a datagram socket
- Create a datagram packet with server address
- Send the packet
- Receive, in a loop, packet and display the message until the "stop condition"

CODE

UdpServer.java

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

class UdpServer
{
    public void server() throws Exception
    {
        DatagramSocket socket = new DatagramSocket(3333);
        DatagramPacket packet = new DatagramPacket(new byte[1024], 1024);

        socket.receive(packet);

        InetAddress address = packet.getAddress();
        int port = packet.getPort();

        Scanner scanner = new Scanner(System.in);
        System.out.println("Server: type lines of text to send");
        do
            {
                String message = scanner.nextLine();
                packet = new DatagramPacket(message.getBytes(), message.length(),
                socket.send(packet);
            }
        while (Boolean.TRUE);

        scanner.close();
        socket.close();
    }

    public static void main(String args[]) throws Exception
    {
        UdpServer us = new UdpServer();
        us.server();
    }
}
```

UdpClient.java

```
import java.net.*;

class UdpClient
{
    public void display(DatagramPacket packet)
    {
        byte[] p = packet.getData();

        for (int i=0; i < packet.getLength(); i++)
            System.out.print((char)p[i]);

        System.out.println();
    }

    public void client() throws Exception
    {
        DatagramSocket socket = new DatagramSocket();
        DatagramPacket packet;

        packet = new DatagramPacket(new byte[1024], 1024, InetAddress.getByName("l
        socket.send(packet);

        packet = new DatagramPacket(new byte[1024], 1024);
        do
            {
                socket.receive(packet);
                display(packet);
            }
        while (Boolean.TRUE);
        socket.close();
    }

    public static void main(String args[]) throws Exception
    {
        UdpClient uc = new UdpClient();
        uc.client();
    }
}
```

RUN

javac UdpServer.java UdpClient.java

```
>java UdpServer                        >java UdpClient
Server: type lines of text to send     1
1                                      2
2                                      3
3                                      A
A                                      B
B                                      C
C
```

11 B.5 RSA

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

DESIGN

RSA: Rivest, Shamir, and Adelman

1. Find n

   Choose two large prime numbers, a and b, and derive n = ab.

2. Find x.

   Select encryption key x such that x and (a - 1)(b - 1) are relatively prime.

3. Find y.

   Calculate decryption key y such that y is inverse of x,

   xy mod (a-1)(b-1) = 1

4. The public key = {x, n}.

5. The private key = {y, n}.

6. Encryption: $C = P(M) = M^x$ (mod n); Decryption: $M = S(C) = C^y$ (mod n)

CODE

```
import java.util.Scanner;

public class RSA
{
```

```
int x;
int y;
int n; // S=(x,n), P=(y,n)

String T;

// gcd(m,n) = gcd(n, m%n)
int gcd(int m, int n)
{
    if (n == 0) return m;
    return gcd(n, m%n);
}

// a^m % n
int pow(int a, int m, int n)
{
    int r = 1;
    while (m-- != 0)
        r = (r*a) % n;

    return r ;
}

void rsa()
{
    int a;
    int b;
    int z;

    //odd prime numbers 3, 5 (3, 11?)
    a = 13;
    b = 11;
```

```
        n = a*b;
        z = (a-1)*(b-1);

        // choose e as relative prime
        for (x=2; gcd(x, z) != 1; x++);

        // choose d as inverse of e
        for (y=2; (y*x) % z != 1; y++);

        System.out.println("\n(x,n)=(" + x + "," + n + ")" + " Public Key");
        System.out.println("\n(y,n)=(" + y + "," + n + ")" + " Private Key");
        System.out.println("\n");
    }

    void cipher(char[] T, int k, int n) // k = e or d
    {
        int r;

        for (int i=0; i < T.length; i++)
            {
                T[i] = (char)pow((int)T[i], k, n);
            }
        System.out.println("C: " + String.valueOf(T));
    }

    void input()
    {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter T:");
        T = scanner.next();

        scanner.close();
    }

    void output()
    {
        System.out.println("T: " + T);

        char[] M = T.toCharArray();
        cipher(M, x, n); // M^x % n
        cipher(M, y, n); // C^y % n
    }


    public static void main(String[] args)
    {
        RSA r = new RSA();

        r.rsa();
        r.input();
        r.output();
    }
}
```

RUN:

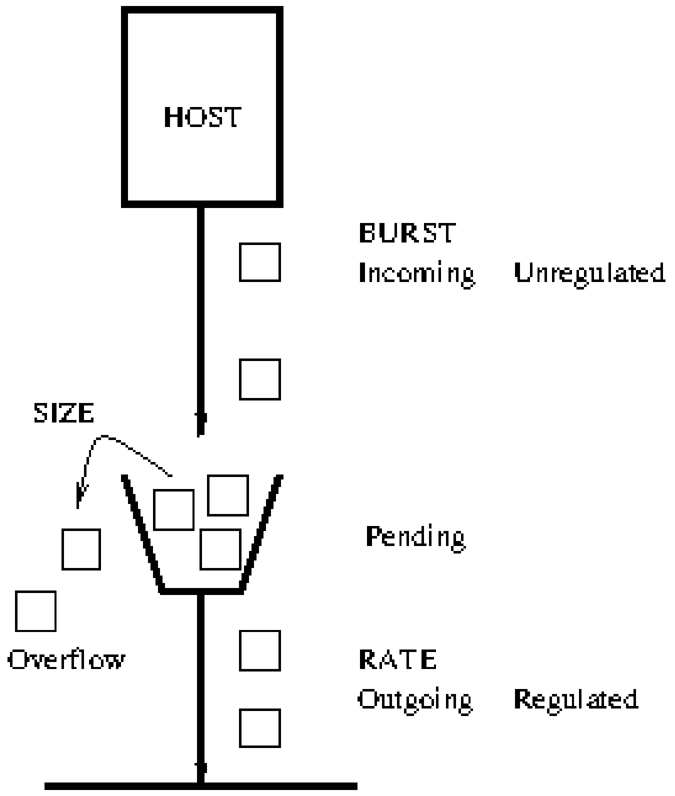
```
javac RSA.java
java  RSA

S=(d,n)=(103,143)
P=(e,n)=(7,143)

Enter T:ABCabc123
```

```
C: ABY; ,$)t
T: ABCabc123
```

12 B.6 Leaky Bucket

Write a program for congestion control using leaky bucket algorithm.

DESIGN



1. The leaky bucket algorithm used to control rate in a network
2. In this algorithm the input rate can vary but the output rate remains constant
3. This algorithm saves bursty traffic into fixed rate traffic by averaging the data rate
4. If the bucket (buffer) overflows then packets are discarded

5. It is implemented as a single-server queue with constant service rate

CODE

```
import java.util.*;

public class LeakyBucket
{
    int burst;
    int rate;
    int size;

    int incoming;
    int outgoing;
    int pending;
    int overflow;

    LeakyBucket()
    {
        pending = 0;
        incoming = 0;
        overflow = 0;
        outgoing = 0;
    }


    void leakyBucket()
    {
        System.out.println("Time    Incoming Pending Overflow Outgoing");

        Random rand = new Random();
        int time=0;
        while (time < 8)
            {
                incoming = rand.nextInt(burst);
                if ((pending + incoming) > size)
                    {
                        overflow = (pending + incoming) - size;
                        pending = size;
                    }
                else
                    {
                        overflow = 0;
                        pending += incoming;
                    }

                output(time, incoming, pending, overflow, outgoing);
                outgoing = Math.min(rate, pending);
                pending -= outgoing;

                incoming = 0;
                ++time;
            }
    }

    void input()
    {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter burst size: ");
        burst = scanner.nextInt();

        System.out.println("Enter bucket size: ");
        size = scanner.nextInt();

        System.out.println("Enter outgoing rate: ");
        rate = scanner.nextInt();

        scanner.close();
    }

    void output(int time, int incoming, int pending, int overflow, int outgoing)
    {
        System.out.printf("%d\t%d\t%d\t%d\t%d\n",time,incoming,pending,overflow,ou
    }


    public static void main(String[] args)
    {
        LeakyBucket lb = new LeakyBucket();

        lb.input();
        lb.leakyBucket();
    }
}
```

RUN

```
javac LeakyBucket.java
java  LeakyBucket

Enter burst size: 8
Enter bucket size: 8
Enter outgoing rate: 2

Time    Incoming Pending Overflow Outgoing
0        2        2        0        0
1        4        4        0        2
2        5        7        0        2
3        4        8        1        2
4        5        8        3        2
5        6        8        4        2
6        5        8        3        2
7        0        6        3        2
```