Data Communication and Networking Lab

Part A: C Programs

Program 1: Write a program for error detection using CRC-CCITT(16-bits).

```
#include<stdio.h>
#include<string.h>
#define N strlen(g)

char t[28],cs[28],g[]="10001000000100001";
int a,e,c;

void xor()
{
    for(c = 1;c < N; c++)
            cs[c] = (( cs[c] == g[c])?'0':'1');
}

void crc()
{
    for(e=0;e<N;e++)
            cs[e]=t[e];
    //If leftmost bit is '1' then perform XOR operation
    do{
            if(cs[0]=='1')
                    xor();
            // shift to right by 1 bit
            for(c=0;c<N-1;c++)
                    cs[c]=cs[c+1];
            cs[c]=t[e++];
    }while(e<=a+N-1);
}

int main()
{
    printf("\nEnter data : ");
    scanf("%s",t);
    printf("\n---------------------------------------");
    printf("\nGeneratng polynomial : %s",g);
    a=strlen(t);
    // Add N-1 Redundunt bit to Data
    for(e=a;e<a+N-1;e++)
            t[e]='0';
    printf("\n---------------------------------------");
    printf("\nModified data is : %s",t);
    printf("\n---------------------------------------");
    crc();
    printf("\n CRC checksum is : %s",cs);
    for(e=a;e<a+N-1;e++)
            t[e]=cs[e-a];
    printf("\n---------------------------------------");
    printf("\nFinal codeword transmitted is : %s",t);
    printf("\n---------------------------------------");
```

```
    printf("\nTest error detection 0(yes) 1(no)? : ");
    scanf("%d",&e);
    if(e==0)
    {
            do{
                    printf("\nEnter the position where error is to be inserted : ");
                    scanf("%d",&e);
            }while(e==0 || e>a+N-1);
    t[e-1]=(t[e-1]=='0')?'1':'0';
    printf("\n----------------------------------------");
    printf("\nErroneous data : %s\n",t);
    }
    crc();
    for(e=0;(e<N-1) && (cs[e]!='1');e++);
    if(e<N-1)
       {
            printf("\n CRC checksum is : %s",cs);
            printf("\nError detected\n\n");
       }
    else
       {
            printf("\n CRC checksum is : %s",cs);
            printf("\nNo error detected\n\n");
       }
    printf("\n----------------------------------------\n");
    return 0;
    }
```

Program 2: Write a program for error detection and correction using Hamming Code

```
#include<stdio.h>
void main() {
int data[10];
int dataatrec[10],c,c1,c2,c3,i;
printf("Enter 4 bits of data one by one\n");
scanf("%d",&data[0]);
scanf("%d",&data[1]);
scanf("%d",&data[2]);
scanf("%d",&data[4]);
   //Calculation of even parity
data[6]=data[0]^data[2]^data[4];
data[5]=data[0]^data[1]^data[4];
data[3]=data[0]^data[1]^data[2];
printf("\nEncoded data is\n");
for(i=0;i<7;i++)
printf("%d",data[i]);
printf("\n\nEnter received data bits one by one\n");
for(i=0;i<7;i++)
scanf("%d",&dataatrec[i]);
   c1=dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];
   c2=dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];
   c3=dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];
```

```
   c=c3*4+c2*2+c1 ;
if(c==0) {
printf("\nNo error while transmission of data\n");
   }
else {
printf("\nError on position %d",c);
printf("\nData sent : ");
for(i=0;i<7;i++)
    printf("%d",data[i]);
printf("\nData received : ");
for(i=0;i<7;i++)
    printf("%d",dataatrec[i]);
printf("\nCorrect message is\n");
      //if errorneous bit is 0 we complement it else vice versa
if(dataatrec[7-c]==0)
dataatrec[7-c]=1;
else
dataatrec[7-c]=0;
for (i=0;i<7;i++) {
printf("%d",dataatrec[i]);
      }
   }
}
```

Program 3: Write a program for congestion control using leaky bucket algorithm.

```
#include<stdio.h>

int main(){

    int incoming, outgoing, buck_capacity, n, store = 0;

    printf("Enter bucket capacity, outgoing rate and No.of inputs: ");

    scanf("%d %d %d", &buck_capacity, &outgoing, &n);

    while (n != 0) {          //loop over total number of inputs

       printf("Enter the number of incoming packets: ");

       scanf("%d", &incoming);

       printf("No. of Incoming packets:%d\n", incoming);

if((incoming-outgoing) <= (buck_capacity-store)) //it is possible to send without
dropping

{

int sent = outgoing>=incoming?incoming:outgoing; //if incoming is more than
outgoing, total sent will be outgoing rest will be buffered

if(sent < outgoing && store != 0)  //if incoming<outgoing, we can add values from
the store to be sent till Top cap is reached
```

```
{
int remaining = outgoing-sent;
while(remaining > 0 && store != 0) //keeps adding one to sent until we run out of top cap or nothing left in buff
{
remaining -= 1;
store -= 1;
sent += 1;
}
}
printf("%d packets sent out\n", sent);
if(outgoing<incoming)
{
store = store + (incoming - outgoing);
} }
else //packets need to be dropped
{
int dropped = (incoming-outgoing)-(buck_capacity-store); //excess packets are the ones that after sending cant be accomodated in buff
printf("%d packets dropped\n", dropped);
store = buck_capacity;
}
printf("%d out of %d space used in the buffer\n", store, buck_capacity);
      n--;
   }}
```

Program 4: Design an RPC (Remote Procedure Call) application for a server performing integer arithmetic operations. Describe call and reply message contents, error handling, and client-server interaction.

Refer video

Program 5: Using TCP/IP sockets, write a client – server program where the client send the file name and the server send back the contents of the requested file if present.

```
/*SERVER - Create a file called hello.txt in the current directory and pass that as the file name for server */

#include<stdio.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<netdb.h>

#include<stdlib.h>

#include<string.h>

#include<unistd.h>

#define SERV_TCP_PORT 5035

#define MAX 60

int i, j, tem;

char buff[4096], t;

FILE *f1;

int main(int afg, char *argv)

{

int sockfd, newsockfd, clength;

struct sockaddr_in serv_addr,cli_addr;

char t[MAX], str[MAX];

strcpy(t,"exit");

sockfd=socket(AF_INET, SOCK_STREAM,0);

serv_addr.sin_family=AF_INET;

serv_addr.sin_addr.s_addr=INADDR_ANY;
```

```
serv_addr.sin_port=htons(SERV_TCP_PORT);
printf("\nBinded");
bind(sockfd,(struct sockaddr*)&serv_addr, sizeof(serv_addr));
printf("\nListening...");
listen(sockfd, 5);
clength=sizeof(cli_addr);
newsockfd=accept(sockfd,(struct sockaddr*) &cli_addr,&clength);
close(sockfd);
read(newsockfd, &str, MAX);
printf("\nClient message\n File Name : %s\n", str);
f1=fopen(str, "r");
while(fgets(buff, 4096, f1)!=NULL) {
write(newsockfd, buff,MAX);
printf("\n"); }
fclose(f1);
printf("\nFile Transferred\n");
return 0;
}

//CLIENT
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<netdb.h>
#include<stdlib.h>
#include<string.h>
```

```
#include<unistd.h>
#define SERV_TCP_PORT 5035
#define MAX 60
int main(int arg,char*argv[])
{
int sockfd,n;
struct sockaddr_in serv_addr;
struct hostent*server;
char send[MAX],recvline[MAX],s[MAX],name[MAX];
sockfd=socket(AF_INET,SOCK_STREAM,0);
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=inet_addr("127.0.0.1");
serv_addr.sin_port=htons(SERV_TCP_PORT);
connect(sockfd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));
printf("\nEnter the source file name : \n");
scanf("%s",send);
write(sockfd,send,MAX);
while((n=read(sockfd,recvline,MAX))!=0) {
printf("%s",recvline);
}
close(sockfd);
return 0;
}
```

Part B: OPNET

Simulation 1:

Simulate three node point-to-point networks with duplex links between them. Set the queue size and vary the bandwidth and find the number of packets dropped.

Solution:

Step 1: Create a New Project

Step 2: Create the Network

- Select the **Object Palette** box.
- Select **Client-server** from the drop-down menu.
- Choose **eth4_slip4_multihomed_client objects** (3 numbers).
- Choose **Application Config** and **Profile Config** objects.
- Choose the **10baseT** link and connect the client nodes.
- Close the **Object Palette** box.

Step 3: Configure the Application Config

- Select the **Application config** object.
- Right-click and select **Edit Attributes**.
- Select **Application Definitions** => set row = 1.
- In row go to row0 => set **Name** = video. Select **description** => set **Video Conferencing = High Resolution Video**.
- Check **Apply Changes to Selected Objects**.
- Click **Ok**.

Step 4: Configure the Profile config

- Select **Profile config** object.
- Right-click and select **Edit Attributes**.
- Select **Profile Configuration**. Set row = 1,
- In row0, set **Profile Name** = video profile.
- Select Applications. Set row=1.Go to row0 set **Name** = video, **Start Time Offset** to constant(1), in **Repeatability**, **Inter Repetition Time** to constant(1) and **Number of Repetition** to unlimited and **Repetition Pattern** to concurrent.
- In **Repeatability**, **Inter Repetition Time** to constant (1) and **Number of Repetition** to exponential (0.1) and
- **Repetition Pattern** to Concurrent.
- Check **Apply Changes to Selected Objects**.
- Click **Ok**.

Step 5: Configure Node Objects

- Select any **eth4_slip4_multihomed_client**.
- Right click and **Select Similar Nodes**.
- Right click and **select Edit Attributes**.
- Select **Application Support Profiles** => set rows to 1.
- In rows => go to row0 => set **Profile Name** = video profile.
- Select **Application Support Services**. Select edit => set rows =1. Set Name = video for that row.
- Select **IP Processing Information** => set **Memory Size** to 8MB => set **Datagram Forwarding Rate** to 5000.
- Check **Apply Changes to Selected Objects**.
- Click **Ok**.

Step 6: Selecting Statistics for viewing results

- Right click on the work space and select choose individual DES statistics => in **Global Statistics** go to **IP**,
  select **traffic dropped**.
- In **Node Statistics** => select **IP** =>**Traffic Dropped (Packets/Sec)**, **Traffic Received (Packets/Sec)** and
  Traffic Sent (Packets/Sec).
- Click **Ok**.

Step 7: Run the simulation

- Click **run simulation** icon from the toolbar.
- Set the **Duration** to 120 seconds.
- Click **Run**.

Step 8: View Results

- Right click on the work space and select **View Results**.
- Select the statistics from the **View Results**.
- Select **Sample Sum** instead of **As Is**.
- Click **Show** button to view the graphs.

To Vary Bandwidth and Queue Size:

1. Select **Duplicate Scenario** from the **Scenario** menu. Name the scenario.
2. You can vary the queue size and bandwidth by changing the values of **Memory size** and **Datagram Forwarding**
   Rate as in Step 5 of the procedure above. Change it to 16 Mb and 10000 respectively.
3. Run the Simulation. You can compare the results by clicking on **Compare Results** from the **Results** menu. ***Ideally the number of packets dropped must be less .***

Simulation 2:

Using OPNET Simulate a four node point-to-point network, and connect the links as follows: n0-n2, n1-n2 and n2-n3. Apply TCP agent between n0-n3 and UDP agent between n1-n3. Apply relevant applications over TCP and UDP agents by changing the parameters and determine the number of packets sent by TCP/UDP.

Step 1: Create a New Project

Step 2: Creating network topology:

- Select eth4_slip4_multihomed_client (4 numbers) from the client_server tool in Object Palette.
- elect an ethernet_server.
- Select ethernet from Object Palette.
- Connect them using the 10BaseT links.
- Select Application Config and Profile config objects.

Step 3: Configuring network application Right click on the Application definition object.

- Select Edit Attributes.
- Select 2 rows for applications
- Select the FTP application for TCP traffic and set the traffic to High Load.
- Select the video conferencing application for UDP traffic and set the traffic to High-Resolution Video.
- Check Apply Changes to Selected Objects and click Ok.

Step 4: Configure profile

- Right click on Profile Definition object and select Edit Attributes.
- TCP profile: Assign the values for the various fields as
- In row0.set **Profile Name** = tcp
- Select Applications. Set row=1.Go to row0 set **Name** = video, **Start Time Offset** to constant(1), in **Repeatability**, **Inter Repetition Time** to constant(1) and **Number of Repetition** to unlimited and **Repetition Pattern** to concurrent.
- In **Repeatability**, **Inter Repetition Time** to constant (1) and **Number of Repetition** to exponential (0.1) and
- **Repetition Pattern** to Concurrent.
- Repeat the same setting for row1 set **Profile Name** = udp
- Click on Apply Changes to Selected Objects and click on OK.

Step 5: Configure the network objects

- Right click on the appropriate object i.e. client node or ethernet server.
- Select Edit Attributes.
- Click on Application Supported Profiles and choose edit.
- Apply both the profiles to the client nodes.
- Click Ok.
- Click on Application Supported Services and choose edit.
- Apply the FTP application in case of Ethernet server object and for client nodes select the Video application.
- This is to simulate FTP-TCP-IP for Server and Video-UDP-IP for client nodes.
- Click Ok.
- Check Apply Changes to Selected Objects and click on Ok.

Step 6: Choose statistics:

- Right click on the workspace and Select Choose Individual Statistics from Node Statistics as Traffic Received and Traffic sent.

Step 7: Run the simulation

- Click **run simulation** icon from the toolbar.
- Set the **Duration** to 120 seconds.
- Click **Run**.

Step 8: View Results

- Right click on the work space and select **View Results**.
- Select the statistics from the **View Results**.
- Select **Sample Sum** instead of **As Is**.
- Click **Show** button to view the graphs.

Interpretation:

Notice the difference in scale in the Y-Axis. For TCP the range is in 1000s and in UDP in 10000s. This means the amount of data sent and received in UDP is much higher than that in UDP, which implies a higher data rate and therefore a higher data loss.

These graphs show that for UDP the loss rate is high especially since this is a video conferencing application with high data rate. However, for TCP, since congestion control is applied the loss rate is less and we see a more even graph with less loss.

Simulation 3:

Simulate the transmission of PING message over a network topology consisting of 4 nodes and find the number of packets dropped due to congestion.

Step 1: Create a new Project.

Step 2: Create the network:

- Select four Ethernet work stations from Ethernet too in object palette.
- Select an Ethernet16 hub from Ethernet tools in obje palette.
- Connect the components using 10Base_T links.
- Select the IP attribute definition from Ethernet tools object palette.
- Select IP_ping_traffic object palette. pics trotic from internet tools in

Step 3: Configuring the IP Attribute Definition object.

- Right Click on IP Attribute Definition object in work space
- Select Edit Attribute=>IP Ping Parameters.
- Set parameters as shown below.
    - Intervel(sec)=1.0
    - Time Out=0.1

Please Note: It is 10Base T Connection between the nodes and the hub in a Star Topology. It is a PING Model between the nodes. You need to connect to and from the node to simulate a bi-directional ping as is required in this question.

Step 4: Configuring IP Ping Links:

- Right Click on any one IP ping link and select similar Demands.
- Select any one Ping link and Edit its attributes as below:
    - Interrepetition time=Constant(1)
    - Max Repetition count = unlimited
    - Start Time= Constant(1)

Step 5: Choose Individual Statistics:

- Right click on the work space and choose individual statistics.
- Node Statistics => IP => (Ping Replies, Ping Request Sent, Ping Response Time, Traffic Dropped)

Step 6: Duplicate the scenario

In this scenario remove any link between the hub and workstation.

Step 7.Run the Simulation:

- Click on Run Button
- Set Duration to 100 seconds.
- Click on Run.

Step 8: View and Compare the results

Result Interpretation:

When node 12 is disconnected both the ping packets are dropped. This is shown in justified by the graph. We see similar results for node 11 and node 9 which must drop 1 packet. However there should be no change for node 10. Packet generation rate is 1 ping packet per destination per second.

Simulation 4: Simulate an Ethernet LAN using N nodes (6-10). Change the data rate and compare throughput and bit error rate.

Step 1: Create a New Project

Step 2: Create the network

Select topology=>rapid configuration. From the drop down menu choose star and click Ok.

In the rapid configuration dialog: star box, set the following values:

Central node model = ethernet16_hub, Periphery node model = Ethernet_station, Link model = 10baseT.

Step 3: Configuring Ethernet Stations:

Right click on any Ethernet station => Select similar nodes Edit Attributes

Set Traffic generation parameters and Packet Generation Arguments.

Start Time=Constant(0)

ON State Time=Constant(100)

OFF State Time=Constant(0)

Interarrival Time=Constant(1.0)

Check Apply changes to selected objects.

Click Ok.

Step 4: Choose Statistics

Right Click on work space => Choose Individual Statistics In Link Statistics => low level point-to-point=> bit error per packet

=> point-to-point =>throughput[packets/sec] -> & <-

Click OK

Step 5: Run Simulation

Click on Run Button

Set Duration to 70 seconds

Click Run

Step 5: View & Compare results:

Change the data rate by changing the inter arrival time

Scenario 1: inter arrival time is constant 1

Scenario 2: inter arrival time is constant 0.1

Result Interpretation: The data rate is changed by changing the Inter Arrival Rate.

First Scenario Constant 1 implies the interarrival rate is less than Constant 0.1, Second Scenario. When Interarrival time increases, data rate increases implying that the throughput is increased as long as there is no collision.

Simulation 5: Simulate an Ethernet LAN using N nodes and set multiple traffic nodes and plot the congestion window for different source/destination.

Step 1: Create a New Project

Step 2: Create the Network

Select Object Palette box.

Select internet_toolbox from drop down menu. Choose ethernet_wkstn objects (3 numbers).

Choose ehternet_server object (1 number). Ethernet16hub (1) from Ethernet tool box

Choose Application Config, and Profile Config objects.

Choose 10baseT link and connect the nodes.

Close the Object Palette box.

Step 3: Configure the Network Application

Select Application config object.

Right click and select Edit Attributes.

Select Application Definitions => set row = 2.

In row go to row0 => set Name = ftp. Select description => set ftp = > High Load.

In row go to row1 => set Name = volce. Select description => set voice = > GSM Quality Speech.

Check Apply Changes to Selected Objects.

Step 4: Configure the Profile

Select Profile config node.

Right click and select Edit Attributes.

Select Profile Configuration. Set row = 2.

In rowo, set Profile Name = ftp profile. Select Applications. Set row=1.Go to rowo set Name = ftp, Start Time Offset to constant(1), in Repeatability, Inter Repetition Time to exponential (0.1) and Number of Repetition to unlimited and Repetition Pattern to concurrent.

At rowo set Name = ftp, Start Time Offset to constant(1), in Repeatability, Inter Repetition Time to exponential(0.1) and Number of Repetition to unlimited and Repetition Pattern to concurrent.

In row1, set Profile Name = voice profile. Select Applications. Set row=1.Go to row1 set Name = voice, Start Time Offset to constant(1), in Repeatability, Inter Repetition Time to exponential (0.1) and Number of Repetition to unlimited and Repetition Pattern to concurrent.

At row1 set Name = video, Start Time Offset to constant(1), in Repeatability, Inter Repetition Time to exponential (0.1) and Number of Repetition to unlimited and Repetition Pattern to concurrent.

Check Apply Changes to Selected Objects.

Step 5: Configure Network Objects

Select any Ethernet_wskt.

Right click and Select Similar Nodes.

Right click and select Edit Attributes.

Select Application Support Profiles => set rows to 2.

In rows => go to rowo => set Profile Name = ftp profile

In rows => go to row1 => set Profile Name = voice profile.

Select Application Support Services. Select edit => set rows 1. Set Name = voice for that row.

Check Apply Changes to Selected Objects.

Click Ok.

Select ethernet_server object.

Right click and select Edit Attributes.

Select Application Support Services. Select edit => se rows 1. Set Name = ftp for that row.

Check Apply Changes to Selected Objects.

Click Ok.

Step 6: Selecting Statistics for viewing:

In Node Statistics => select TCP Connection => select Congestion Window Size (bytes)

Step 7: Run the simulation

Click run simulation icon from the toolbar.

Set the Duration to 20 seconds.

Click Run.

Simulation 6: Simulate simple BSS and with transmitting nodes in wireless LAN by simulation and determine the performance with respect to transmission of packets.

Step 1: Create a New Project

Step 2: Create the Network

Select Object Palette box.

Select wireless_lan_adv from drop down menu.

Choose wlan_station_sdv(fix) 4 numbers

Step 3: Configure the Network Application (Select Topology icon ->open Annotation Palette >(choose circle) and annotation.) include all 4 nodes in a circle

Select any wlan_station_adv object in the workspace.

Right click on the selected object and Select Similar Nodes.

Right click and select Edit Attributes.

Select Traffic Generation Parameters => set Start Time to constant (> set ON State to constant (100) => set OFF State to constant (0).

Please Note: If the field to set Start Time to constant(1) is disabled for some reason, please change the value from 'Never' to 'Not Used'

Select Packet Generation Arguments =>set Interarrival Time to exponential(1).

Check Apply Changes to Selected Objects.

Click Ok.

Step 4: Selecting Statistics for viewing:

Right click on the workspace and select Choose Individual Statistics.

In Node Statistics => select Wireless LAN => select Data Traffic Rcvd (packets/sec) and select Data Traffic send (packets/sec).

Step 5: Run the simulation

Click run simulation icon from the toolbar.

Set the Duration to 100 seconds.

Click Run.

Step 6: View Results

Right click on the work space and select View Results

Change As Is to Sample_Sum.

Result Interpretation: We see that on an average 1 packet is sent per second and around 2.5 packets are received per second. This is because the transmissions of the other 3 stations are also being received by this station. There are some losses as well which can be seen by including global statistics (bit/sec).