IMPLEMENTATION AND SIMULATION(NS2) OF NON-PERSISTENT CSMA/CD PROTOCOL WITH BACK-OFF ALGORITHM(java)

Report submitted to the SASTRA Deemed to be University
as the requirement for the course

CSE302: COMPUTER NETWORKS

Submitted by

Divya Dharshini V

(Reg. No.:124003086, CSE)

December 2022

SCHOOL OF COMPUTING

THANJAVUR, TAMIL NADU, INDIA – 613 401

Bonafide Certificate

This is to certify that the report titled **"Implementation and simulation of Non-Persistent CSMA/CD Protocol with Back-off Algorithm "** submitted as a requirement for the course, **CSE302: COMPUTER NETWORKS** for B.Tech. is a bonafide record of the work done by **Ms. Divya Dharshini (Reg. No.124003086, CSE)** during the academic year 2022-23, in the School of Computing

Project Based Work *Viva voce* held on **13.12.2022**

Examiner 1 **Examiner 2**

ACKNOWLEDGEMENT

First of all, I would like to thank God Almighty for his endless blessings.

I would like to express my sincere gratitude to **Dr S. Vaidyasubramaniam, Vice-Chancellor** for his encouragement during the span of my academic life at SASTRA Deemed University.

I would forever remain grateful and I would like to thank **Dr A.Umamakeswri, Dean, School of Computing** and **R. Chandramouli, Registrar** for their overwhelming support provided during my course span in SASTRA Deemed University.

I am extremely grateful to **Dr. Shankar Sriram, Associate Dean**, **School of Computing** for his constant support, motivation and academic help extended for the past three years of my life in School of Computing.

I would specially thank and express my gratitude to **Dr.Shankar Sriram, Associate Professor,Associate Dean School of Computing** for providing me an opportunity to do this project and for his guidance and support to successfully complete the project.

I also thank all the Teaching and Non-teaching faculty, and all other people who have directly or indirectly help me through their support, encouragement and all other assistance extended for completion of my project and for successful completion of all courses during my academic life at SASTRA Deemed University. Finally, I thank my parents and all others who help me acquire this interest in project and aided me in completing it within the deadline without much struggle.

TABLE OF CONTENTS

ABBREVIATIONS

CSMA	Carrier Sense Multiple Access
CSMA/CD	Carrier Sense Multiple Access/Collision Detection
MAC	Media Access Control
NP-CSMA	Non-Persistent Carrier Sense Multiple Access
HOL	Head of Line
NIC	Network Interface Controller

ABSTRACT

CSMA first allows stations to sense carrier and then transmit the data if carrier is free. It is equipped to handle collisions. CSMA is widely used Media Access (MAC) Protocol that allows multiple users to share a common transmission channel such as 802.11 wireless LAN and Ethernet. A Back-off algorithm is proposed to schedule re-transmissions after collisions. The result shows the complete analysis of the collision detection using non-persistent sensing. After heading to collision , station has to wait for random back off time before transmitting again. The service time of head-of-line packets depends on the Back-off scheduling algorithm when collision occurs. In a collision detection , the station will stops transmitting and sends a congestion signal and waits for a random time interval before re-transmission.

In simulation of the CSMA /CD , the packets are sent to different nodes and if the packet sent meet collision, then the packet tend to drop . With this effect you can know that node has collision. The different packets sent are denoted by different colours.

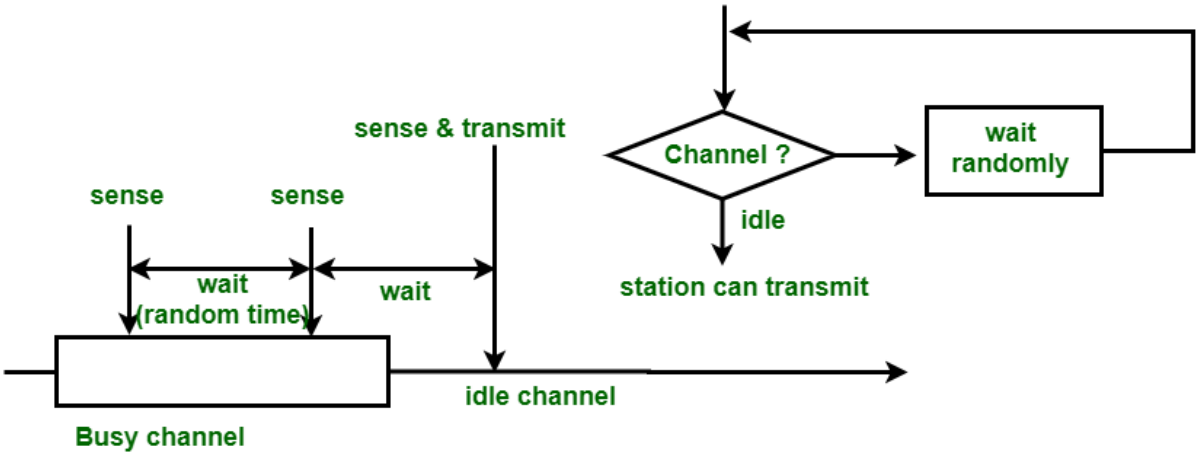
INTRODUCTION

In the data link layer , we use multiple access control to decrease collision and avoid crosstalk. Mulitple access protocol can be further classified as random access , controlled access and channelization access protocols. CSMA refers each station first checks whether the state of the medium before sending. The persistence methods help the station to take action when the channel is busy/idle.

CSMA/CD is used to improve CSMA performance by ending transmission of data as collision is detected, so short the time before the entry can be attempted.

When the transmitting station has the frame to send but it senses the channel is busy , so , it has to wait for a random period of time without sensing the channel in the interimediate, and repeats the algorithm again. Station does not immediately sense for the channel for only purpose of capturing it when it detects end of previous transmission.

In the Non-Persistent Carrier-Sense Multiple Access (NP-CSMA) protocol, the node will take action after sensing the channel's status. Collisions takes place when two or more packets are transmitted at a time.



Back-off algorithm is a widely adopted collision resolution scheme in most MAC protocols.

Back off algorithm is collision resolution method which is used in random access MAC protocols (CSMA/CD). This algorithm is used in Ethernet to schedule re-transmissions after collisions. If a collision occurs between two stations, they may restart transmission as soon as they can after the collision leading to a deadlock.
The random amount of time taken is directly proportional to number of attempts that has made to transmit the signal . It helps to avoid collision in wireless networks.

To prevent this scenario, back-off algorithm is used in CSMA/CD protocol, after the occurrence of collision, station waits for some random back-off time and then starts retransmits.

This is the waiting time for which the station waits before retransmitting the data is called as **Back-off time**.

Backoff algorithm is used for calculating the back off time.

Back off algorithm , after undergoing the collision, transmitting station choosed a random number in range [0, $2^n - 1$] if the packet is undergoing collision for the n^{th} time.

If station chooses a number k, then ,

Back off time = k x Time Slot,

Where the value of one time slot = 1 RTT.

A consent on the stability issue of the Back-off scheme cannot be concluded from these controversial results.

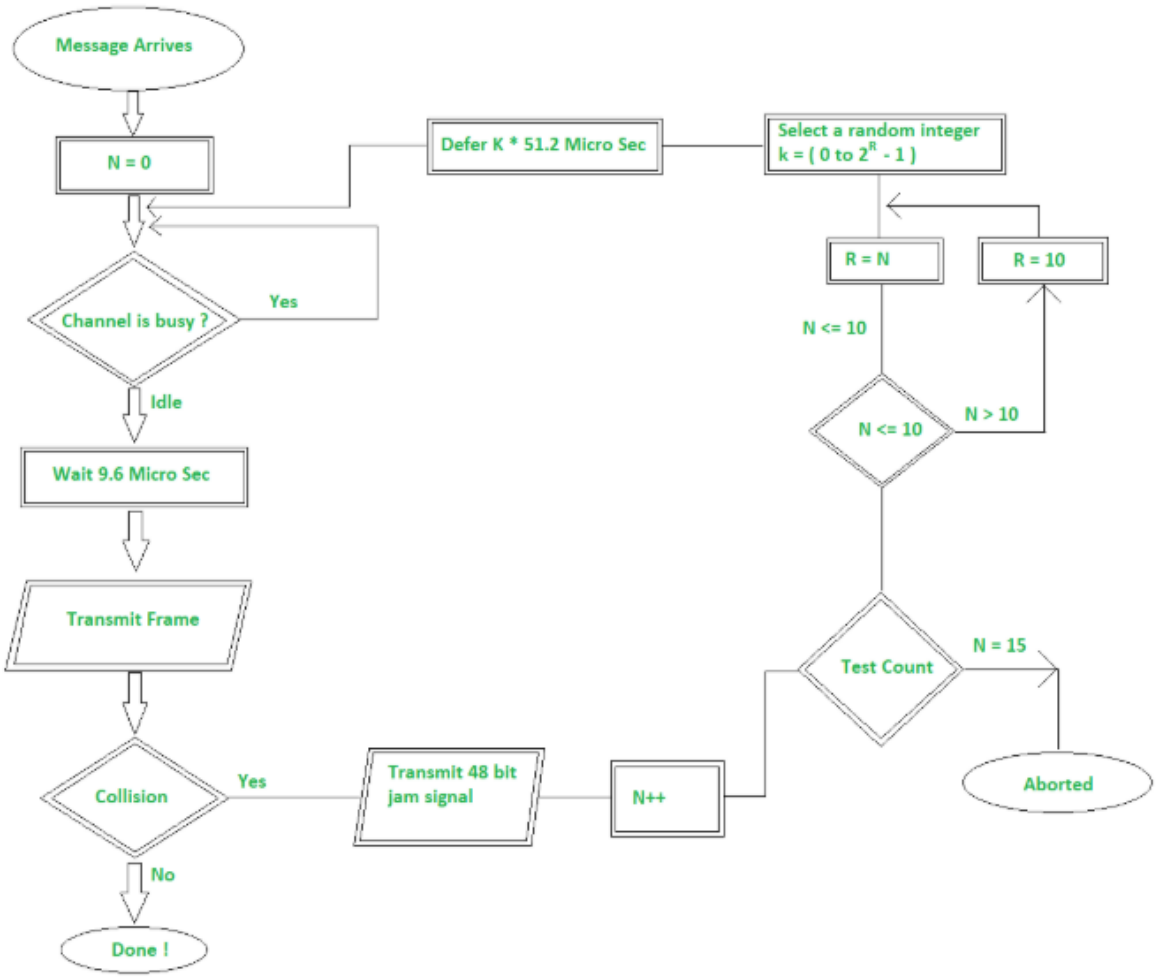
If it detects a collision, it stops transmission immediately and instead transmits a 48-bit jam sequence.

The transmitter sets the number of transmission of the current frame (N) to zero and starts listening to the cable to check if any message bits are moving around . If the cable is not idle, it has to wait till the cable becomes idle.

To avoid starvation ,the carrier after being idle, it waits more 9.6 ms to allow the nodes to prepare themselves for the transmission about to happen next.

It provides a full analysis of NP-CSMA/CD to retransmit the data if any collision occurs. It is impossible to achieve a comprehensive analysis of the system without including the impact of the scheduling algorithm applied on the input queuing behavior. It is naturally unstable if the number of nodes in the network is infinite.

ALGORITHM



After each iteration, the value of N increases , and so, the range will be as [0,2^n-1], the probability of collisions decreases In this way.

In some cases it can be a drawback because the continuous backoff cause discard packets of some nodes. Maximum attempts limits will be reached.

After the collision, each node has to wait for a certain amount of time which is denoted by the formula,

T_{slot} refer to the discrete time slot - length 2t,

where t is the max prop delay in the network.

$K = [0, 2^n - 1]$. n is the collision number.

Advantangaes of CSMA /CD:

Very efficient for light to moderate load.

It is for wireless networks, inexpensive , fast , easy to implement, avoid collision, if so, detects within short time, avoids wastefull transmission.

Disadvantages of CSMA / CD:

Not useful for broadcasting , high power consumption and network may not be available when needed.

Merits of non persistent CSMA

As the station waits for a random time before retransmission , the rate of collision is reduced than 1-persistent CSMA. There is less probability for multiple stations to wait for the same amount of time. The collision between stations is reduced greatly.

Collision probability decreases exponentially.

Demerits of non persistent CSMA

It reduces the bandwidth usage of network.

Collision probability decreases exponentially

Works only for two stations or hosts

CODE:

NewThreads.java

```
import java.util.Random;
import java.util.concurrent.atomic.*;
class NewThread implements Runnable, ChannelConstants
{
String StationNo;
Thread t;
static int distance, stat=0,frame;
static int ChnStatus; //Indicates if channel is being used
int FrameNumber,MaxFrameNumber;
private AtomicBoolean CheckIfSuccessfulTransmission;
static int tfr=50; //Transmission time
private int NumberOfAttempts;
NewThread(String threadname, int MaxFrameNumber)
{
StationNo = threadname;
t = new Thread(this, StationNo);
FrameNumber = 1;
this.MaxFrameNumber=MaxFrameNumber;
CheckIfSuccessfulTransmission = new AtomicBoolean();
t.start();
}
public void run()
{
Random rand = new Random();
while (!CheckIfSuccessfulTransmission.get())
{
NumberOfAttempts++;
```

```
while(FrameNumber <= MaxFrameNumber)
{
if (NumberOfAttempts < 15)
{ //15 is the maximum number of attempts
try
{
if (ChnStatus == chnused)
{

System.out.println(StationNo + " is using Non-Persistent sensing,channel is
busy");
try
{
Thread.sleep(rand.nextInt(50)+1000);
}
catch (InterruptedException e)
{
System.out.println(("Interrupt"));
}
}
else
{
System.out.println(StationNo + " is trying to transmit frame number : "+
FrameNumber);

if (ChnStatus == chnfree && distance == 0)
{//Successful transmission
stat = CheckThreads.checking(Thread.currentThread().getName());
frame = this.FrameNumber;
```

```
ChnStatus = chnused;//set channel to in use
for (; distance < 9000000; distance++)
for(int i =0;i<1000;i++); //simulate transmission over some distance

System.out.println(StationNo + " frame " + FrameNumber + " issuccessful");
CheckIfSuccessfulTransmission.set(true);
FrameNumber++;
distance = 0; //reset distance for next frame's transmission
ChnStatus = chnfree;
}
else
{//Collision has occurred
System.out.println("Collision for frame " + FrameNumber + " of " +StationNo
+ " and frame " + frame + " of Station " + stat);

System.out.println("Retransmitting Station " + stat + "'s frame " + frame);
CheckIfSuccessfulTransmission.set(false);
ChnStatus = chnfree;
NumberOfAttempts++;

try
{
int R = rand.nextInt((int) (Math.pow(2, NumberOfAttempts - 1)));
int BackOffTime = R * tfr;
Thread.sleep(BackOffTime);
}
catch (InterruptedException e)
{
System.out.println("Interrupted");
```

```
}
}
Thread.sleep(1000);
}
}
catch (InterruptedException e)
{
System.out.println(StationNo + "Main Interrupted");
}
}
else
{
CheckIfSuccessfulTransmission.set(true);
System.out.println("Too many attempts for frame " + FrameNumber+ "of "
+StationNo + ". Transmission stopped");
}
}
}
}
}
```

CheckThreads.java

```
/*Class to check which station's frame the current transmission is colliding
with. */
public class CheckThreads implements ChannelConstants
{
public static int checking(String StationName)
{ int stat;
```

```
switch (StationName)
{
case("Station 1") : stat = 1;
break;
case ("Station 2") : stat = 2;
break;
case("Station 3") : stat = 3;
break;
case ("Station 4") : stat = 4;
break;
default : stat = 0;

}
return(stat);
}
}
```

CSMACD.java

```
import java.util.Scanner;
interface ChannelConstants
{
int chnfree=0; //Indicates Channel is free
int chnused=1; //Indicates Channel is being used
}
class CSMACD implements ChannelConstants
{
public static void main(String args[])
{
```

```
Scanner sc=new Scanner(System.in);
NewThread.ChnStatus=chnfree; //initially channel is free
System.out.println("Enter no of stations");
int Noofstations=sc.nextInt();
NewThread ArrayOfObjects[]=new NewThread[Noofstations+1];
int FrameArray[]=new int[Noofstations+1];
for(int i=1;i<=Noofstations;i++)
{
System.out.println("Enter no of frames for station"+i);
FrameArray[i]=sc.nextInt();
}
for(int i=1;i<=Noofstations;i++)
ArrayOfObjects[i]=new
NewThread("Station"+Integer.toString(i),FrameArray[i]);
try
{
//wait for stations to complete transmission
for(int i=1;i<=Noofstations;i++)
ArrayOfObjects[i].t.join();
}
catch(InterruptedException e)
{
System.out.println("Main Thread Interrupted");
}
System.out.println("Transmission completed");
}
}
```

NS2 Simulation

CSMA.tcl

```
#Lan simulation
set ns [new Simulator]

#define color for data flows
$ns color 1 Blue
$ns color 2 Red

 #open tracefiles
set tracefile1 [open out_node20.tr w]
set winfile [open winfile w]
$ns trace-all $tracefile1

#open nam file
set namfile [open out_node20.nam w]
$ns namtrace-all $namfile

#define the finish procedure
proc finish {} {
global ns tracefile1 namfile
$ns flush-trace
close $tracefile1
close $namfile
exec nam out_node20.nam &
exec gawk -f analysis.awk out_node20.tr &
exit 0
}
```

```
#create twenty nodes

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
set n10 [$ns node]
set n11 [$ns node]
set n12 [$ns node]
set n13 [$ns node]
set n14 [$ns node]
set n15 [$ns node]
set n16 [$ns node]
set n17 [$ns node]
set n18 [$ns node]
set n19 [$ns node]
set n20 [$ns node]

$n1 color Red
$n1 shape box

#create links between the nodes
```

$ns duplex-link $n0 $n1 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail

$ns duplex-link $n1 $n4 2Mb 10ms DropTail
$ns duplex-link $n2 $n14 2Mb 10ms DropTail
$ns duplex-link $n14 $n5 2Mb 10ms DropTail
$ns duplex-link $n5 $n3 2Mb 10ms DropTail

$ns duplex-link $n3 $n9 2Mb 10ms DropTail
$ns duplex-link $n9 $n7 2Mb 10ms DropTail
$ns duplex-link $n7 $n16 2Mb 10ms DropTail
$ns duplex-link $n16 $n12 2Mb 10ms DropTail

$ns duplex-link $n12 $n20 2Mb 10ms DropTail
$ns duplex-link $n4 $n6 2Mb 10ms DropTail
$ns duplex-link $n6 $n18 2Mb 10ms DropTail
$ns duplex-link $n8 $n10 2Mb 10ms DropTail

$ns duplex-link $n10 $n3 2Mb 10ms DropTail
$ns duplex-link $n3 $n15 2Mb 10ms DropTail
$ns duplex-link $n15 $n19 2Mb 10ms DropTail
$ns duplex-link $n8 $n11 2Mb 10ms DropTail

```
$ns duplex-link $n11 $n17 2Mb 10ms DropTail



#set lan [$ns newLan "$n3 $n4 $n5" 0.5Mb 40ms LL Queue/DropTail
MAC/Csma/Cd Channel]

set lan [$ns newLan "$n0 $n1 $n2 $n3 $n4 $n5 $n7 $n8 $n9 $n10 $n11 $n12
$n13 $n14 $n15 $n16 $n18 $n19 $n20 " 0.5Mb 40ms LL Queue/DropTail
MAC/Csma/Cd Channel]



 #setup TCP connection
set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
$tcp set packet_size_ 1000

#set ftp over tcp connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp

#setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
```

```
set null [new Agent/Null]
$ns attach-agent $n5 $null
$ns connect $udp $null
$udp set fid_ 2

#setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 0.01Mb
$cbr set random_ false

#scheduling the events
$ns at 0.1 "$cbr start"
$ns at 0.3 "$ftp start"
$ns at 90.0 "$ftp stop"
$ns at 100.0 "$cbr stop"
proc plotWindow {tcpSource file} {
global ns
set time 0.1
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$now $cwnd"
 $ns at [expr $now+$time] "plotWindow $tcpSource $file"
}
$ns at 0.1 "plotWindow $tcp $winfile"
 $ns at 100.0 "finish"
 $ns run
```

Analysis.awk

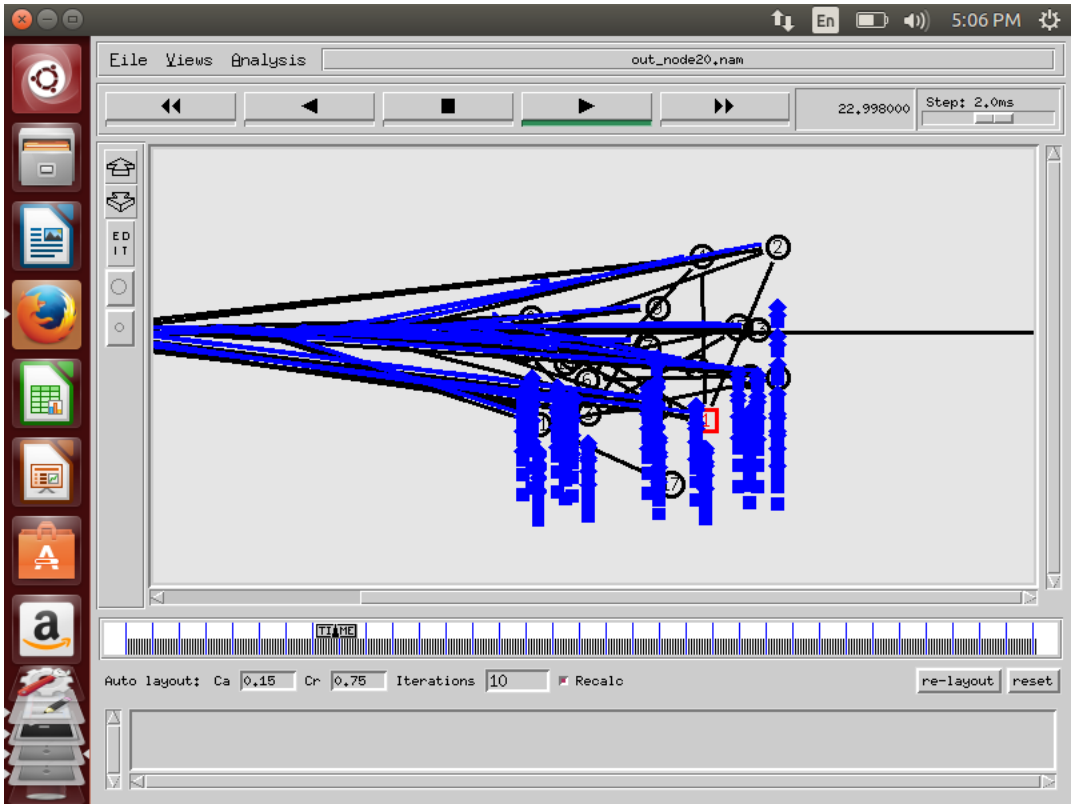
```
BEGIN {
    node = 1;
    time1 = 0.0;
    time2 = 0.0;
    num_packet=0;
    num_total_packet_sent=0;
    bytes_counter=0;
    num_packet_drop=0;
    num_queued=0;
    num_dequeued=0;

   }

    {
num_total_packet_sent++;
       time2 = $2;
       if (time2 - time1 > 0.050) {
          thru = bytes_counter / (time2-time1);
          printf("%f %f\n", time2, thru) > "dataset";
          time1=$2;
          #bytes_counter=0;

       }

       if (($1=="r" && $5=="cbr")||( $1=="r" && $5=="ack")||(($1=="r" &&
$5=="tcp"))) {
```

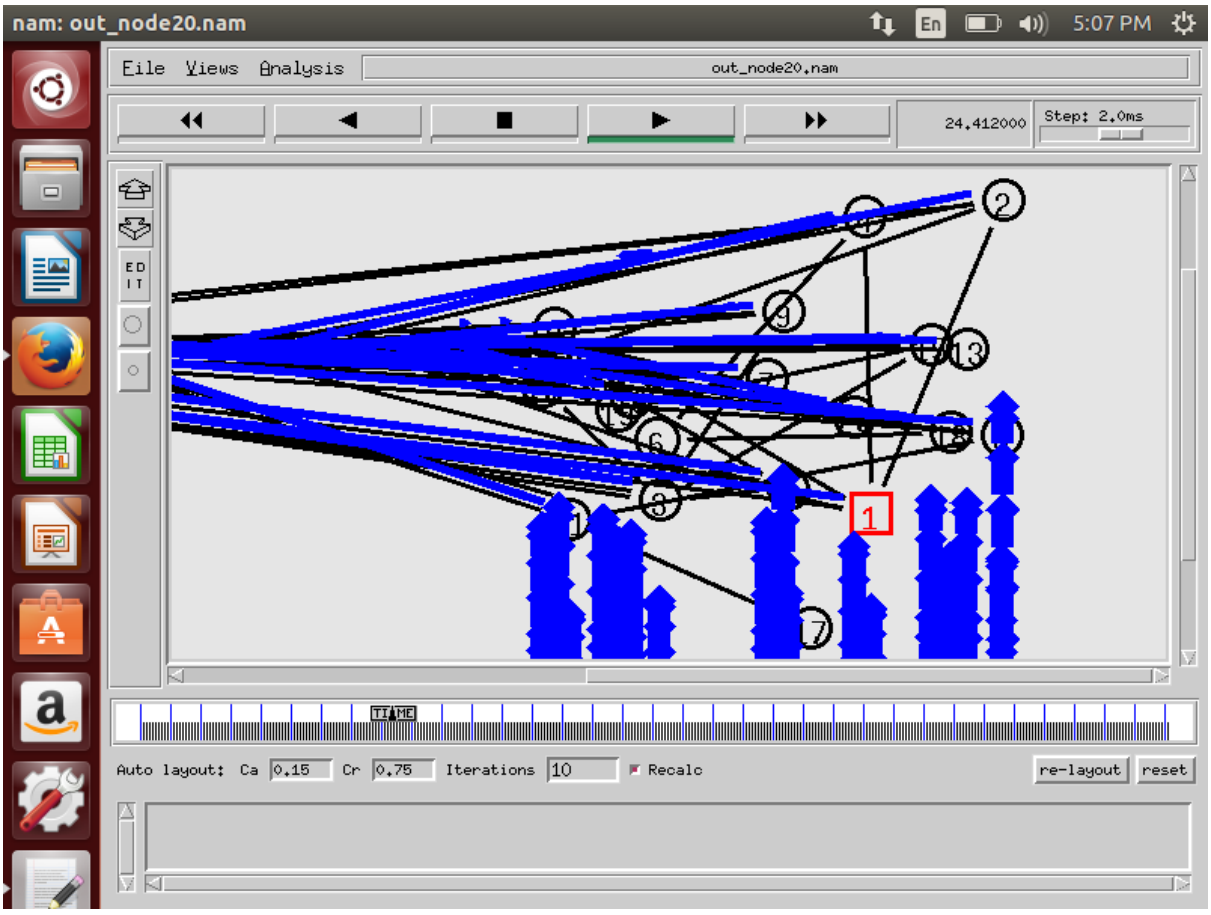
```
            bytes_counter += $6;
            num_packet++;
        }

if (($1=="d" && $5=="cbr")||( $1=="r" && $5=="ack")||(($1=="r" &&
$5=="tcp"))) {
            num_packet_drop++;
        }
if ($1== "+") {
            num_queued++;
        }
        if ($1== "-") {
            num_dequeued++;
        }



}

END {
    printf("\n *******");
    printf("Total number of Packet Sent: \n%d\n",num_total_packet_sent );
    printf("Total number of packets received: \n%d\n", num_packet);
    printf("Total number of packets Dropped: \n%d\n", num_packet_drop);
    printf("Total number of Packet Queued: \n%d\n",num_queued );
    printf("Total number of Packet DeQueued: \n%d\n",num_dequeued );
    printf("Total number of Byte Sent: \n%d\n", bytes_counter);

    printf("********");
}
```

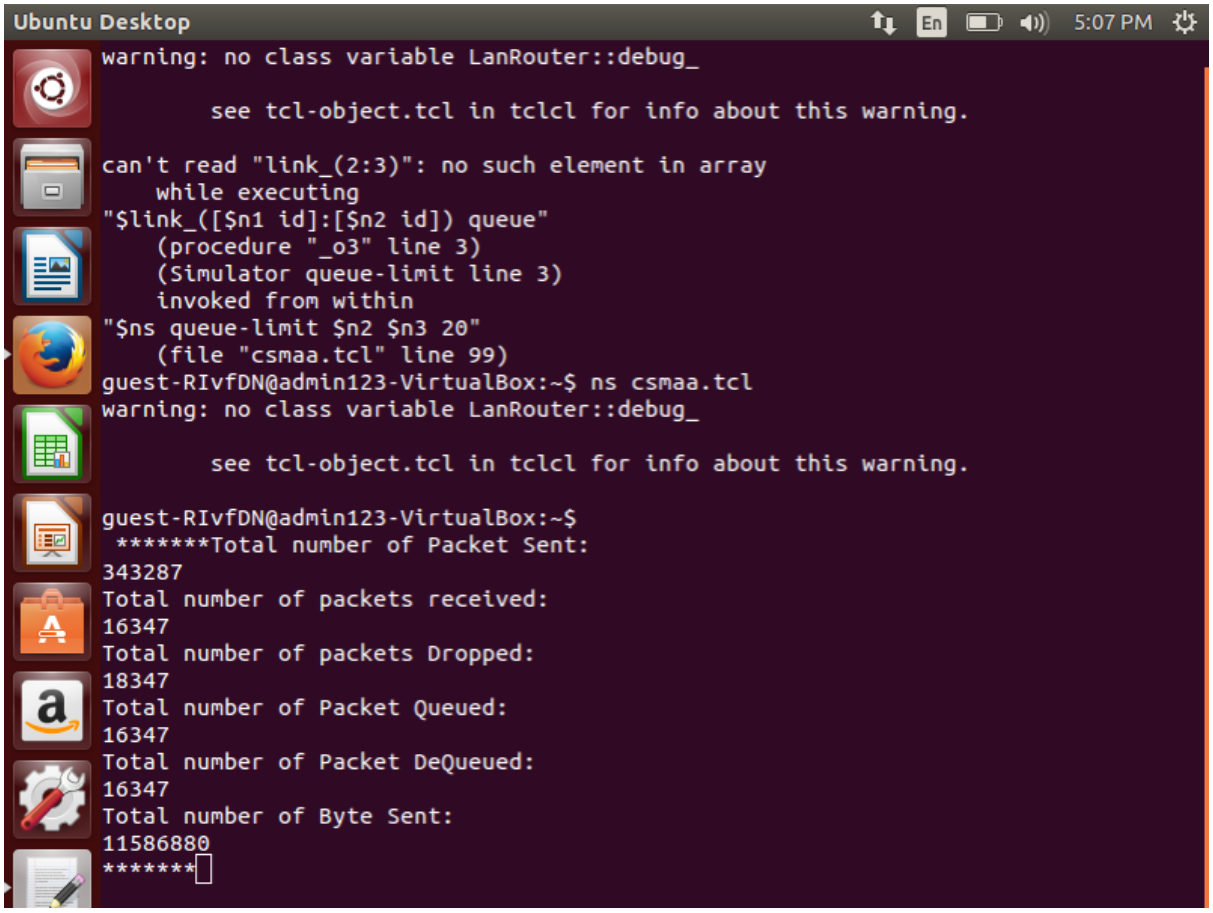
SNAPSHOTS

```
Enter no of stations
5
Enter no of frames for station1
3
Enter no of frames for station2
2
Enter no of frames for station3
4
Enter no of frames for station4
1
Enter no of frames for station5
2
Station1 is trying to transmit frame number : 1
Station3 is trying to transmit frame number : 1
Station5 is trying to transmit frame number : 1
Station2 is trying to transmit frame number : 1
Station4 is trying to transmit frame number : 1
Station3 frame 1 issuccessful
Station2 frame 1 issuccessful
Station5 frame 1 issuccessful
Station4 frame 1 issuccessful
Station1 frame 1 issuccessful
Station2 is trying to transmit frame number : 2
Station3 is trying to transmit frame number : 2
Station1 is trying to transmit frame number : 2
Station5 is trying to transmit frame number : 2
```

```
Collision for frame 2 of Station5 and frame 2 of Station 0
Retransmitting Station 0's frame 2
Collision for frame 2 of Station1 and frame 2 of Station 0
Retransmitting Station 0's frame 2
Collision for frame 2 of Station3 and frame 2 of Station 0
Retransmitting Station 0's frame 2
Station2 frame 2 issuccessful
Station1 is trying to transmit frame number : 2
Station5 is trying to transmit frame number : 2
Collision for frame 2 of Station5 and frame 2 of Station 0
Retransmitting Station 0's frame 2
Station1 frame 2 issuccessful
Station3 is trying to transmit frame number : 2
Station3 frame 2 issuccessful
Station1 is trying to transmit frame number : 3
Station5 is using Non-Persistent sensing,channel is busy
Station1 frame 3 issuccessful
Station3 is trying to transmit frame number : 3
Station3 frame 3 issuccessful
Station5 is trying to transmit frame number : 2
Station5 frame 2 issuccessful
Station3 is trying to transmit frame number : 4
Station3 frame 4 issuccessful
Transmission completed
```


5:06 PM
File Views Analysis
out_node20.nam
22.998000
Step: 2.0ms
TIME
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout
reset


nam: out_node20.nam
5:07 PM
File Views Analysis
out_node20.nam
24.412000
Step: 2.0ms
TIME
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout
reset


Ubuntu Desktop
En
5:07 PM
warning: no class variable LanRouter::debug_
see tcl-object.tcl in tclcl for info about this warning.
can't read "link_(2:3)": no such element in array
while executing
"$link_([$n1 id]:[$n2 id]) queue"
(procedure "_o3" line 3)
(Simulator queue-limit line 3)
invoked from within
"$ns queue-limit $n2 $n3 20"
(file "csmaa.tcl" line 99)
guest-RIvfDN@admin123-VirtualBox:~$ ns csmaa.tcl
warning: no class variable LanRouter::debug_
see tcl-object.tcl in tclcl for info about this warning.
guest-RIvfDN@admin123-VirtualBox:~$
*******Total number of Packet Sent:
343287
Total number of packets received:
16347
Total number of packets Dropped:
18347
Total number of Packet Queued:
16347
Total number of Packet DeQueued:
16347
Total number of Byte Sent:
11586880
****

CONCLUSION

Back-off algorithm is to remove repeated retransmissions of the same block of data to avoid network congestion. The Back-off scheduling algorithm is considered in this analysis of the service time of HOL packets. We found that, the collision is detected using Non-persistent CSMA/CD, after the detection of collision, the station stops transmitting, sends a jam signal, and then waits for a random time interval before re-transmission. Based on this model, the probability of collision is minimized in the data link layer. Simulation of nodes done to show the packets fall from collision and how is detected in CSMA.

References

1 . L. Kleinrock and F. A. Tobagi, “Packet Switching in Radio Channels: Part 1-Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics,” *IEEE Trans. Commun.*, vol. COM-23, pp. 1400-1416, 1975.

2 . P. Chatzimisios, V. Vitsas, and A. C. Boucouvalas, “Throughput and delay analysis of IEEE 802.11 protocol,” in *Proc. 5th IEEE Workshop Networked Appliances*, Oct. 2003, pp. 168–174.

3 . B-J Kwok, N-O Song and L. E. Miller, “Performance analysis of exponential backoff,” *IEEE/ACM Trans. Networking*, pp. 343-335, April 2005.

4. A. Nasipuri, J. Zhuang and S.R. Das, “A multichannel CSMA MAC protocol for multihop wireless networks,” *wireless commun and Netw. Conf.*, pp. 1402-1406, 1999.

[12] T. T. Lee and L. Dai, “Stability and Throughput of Buffered Aloha with Backoff,” *Networking and Internet Architecture*, Apr. 2008.