**EXPERIMENT NO: 7**

**Aim:** Use simulator(Eg.NS2) to understand functioning of ALOHA,CMSA/CD.NS2/Netsim.

**Theory:**

**Advocates of Linux Open-source Hawaii Association** (**ALOHA):**

ALOHA is a protocol for satellite and terrestrial radio transmissions. In pure Aloha, a user can transmit at any time but risks collisions with other users' messages. "Slotted Aloha" reduces the chance of collisions by dividing the channel into time slots and requiring that the user send only at the beginning of a time slot. Aloha was the basis for Ethernet, a local area network protocol.

**Pure ALOHA**

Graph of frames being sent from 4 different stations according to the pure ALOHA protocol with respect to time, with overlapping frames shaded to denote collision. The first version of the protocol (now called "Pure ALOHA", and the one implemented in ALOHAnet) was quite simple:

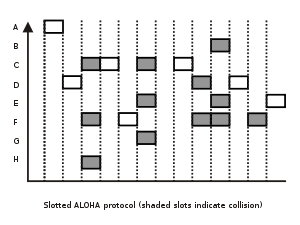
* If you have data to send, send the data
* If, while you are transmitting data, you receive any data from another station, there has been a message collision. All transmitting stations will need to try re sending "later".

|  |
| --- |
| [IMG_256](https://4.bp.blogspot.com/-bLVqBLZ2x6o/U9n0RrZXTfI/AAAAAAAAAls/zg2JKVbEjOk/s1600/300px-Pure_ALOHA1.svg.png) |
| Boxes indicate frames. Shaded boxes indicate frames which have collided. |

Note that the first step implies that Pure ALOHA does not check whether the channel is busy before transmitting. Since collisions can occur and data may have to be sent again, ALOHA cannot use 100% of the capacity of the communications channel. How long a station waits until it transmits, and the likelihood a collision occurs are interrelated, and both effect how efficiently the channel can be used. This means that the concept of "transmit later" is a critical aspect: the quality of the backoff scheme chosen significantly influences the efficiency of the protocol, the ultimate channel capacity, and the predictability of its behavior.

**Slotted ALOHA**

An improvement to the original ALOHA protocol was "Slotted ALOHA", which introduced discrete time slots and increased the maximum throughput.A station can send only at the beginning of a time slot, and thus collisions are reduced. In this case, we only need to worry about the transmission-attempts within 1 frame-time and not 2 consecutive frame-times, since collisions can only occur during each time slot.

[](https://2.bp.blogspot.com/-wfj9CAE-_Ps/U9n1nPZ9rfI/AAAAAAAAAl4/pVxcX2ESFK8/s1600/300px-Slotted_ALOHA.svg.png)

Thus, the probability of there being zero transmission-attempts in a single time slot is:

**Prob\_{slotted} = e^{-G}**

the probability of k packets is:

**Prob\_{slotted} k =  e^{-G} ( 1 - e^{-G} )^{k-1}**

The throughput is:

**S\_{slotted}=Ge^{-G}**

The maximum throughput is 1/e frames per frame-time (reached when G = 1), which is approximately 0.368 frames per frame-time, or **36.8%.** Slotted ALOHA is used in low-data-rate tactical satellite communications networks by military forces, in subscriber-based satellite communications networks, mobile telephony call setup, set-top box communications and in the contact less RFID technologies.

The use of a random-access channel in ALOHAnet led to the development of carrier sense multiple access (CSMA), a "listen before send" random-access protocol that can be used when all nodes send and receive on the same channel. The first implementation of CSMA was Ethernet. CSMA in radio channels was extensively modeled.[11] The AX.25 packet radio protocol is based on the CSMA approach with collision recovery,[12] based on the experience gained from ALOHAnet.

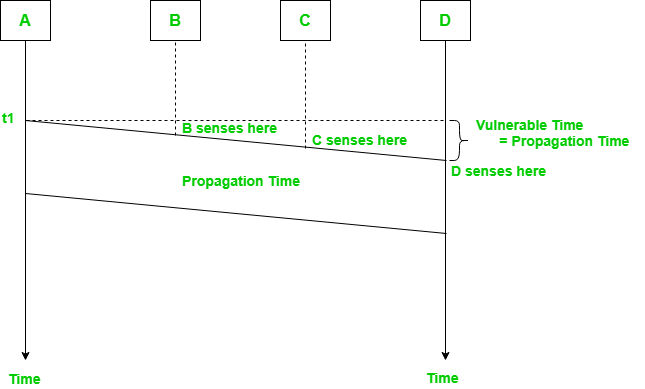
ALOHA and the other random-access protocols have an inherent variability in their throughput and delay performance characteristics. For this reason, applications which need highly deterministic load behavior sometimes used polling or token-passing schemes (such as token ring) instead of contention systems. For instance ARCNET was popular in embedded data applications in the 1980 network.

# Carrier Sense Multiple Access (CSMA)

Prerequisite – [Multiple Access Protocols](https://www.geeksforgeeks.org/computer-network-multiple-access-protocols/" \t "https://www.geeksforgeeks.org/carrier-sense-multiple-access-csma/_blank)  
This method was developed to decrease the chances of collisions when two or more stations start sending their signals over the datalink layer. Carrier Sense multiple access requires that each station **first check the state of the medium** before sending.

**Vulnerable Time –**

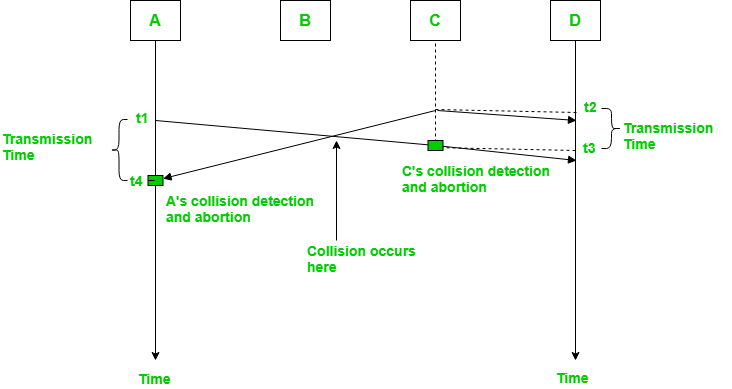
Vulnerable time = Propagation time (Tp)



The persistence methods can be applied to help the station take action when the channel is busy/idle.

### 1. Carrier Sense Multiple Access with Collision Detection (CSMA/CD) –

In this method, a station monitors the medium after it sends a frame to see if the transmission was successful.If succcessful, the station is finished, if not, the frame is sent again.



In the diagram, A starts send the first bit of its frame at t1 and since C sees the channel idle at t2, starts sending its frame at t2. C detects A’s frame at t3 and aborts transmission. A detects C’s frame at t4 and aborts its transmission. Transmission time for C’s frame is therefore Rendered by QuickLaTeX.com and for A’s frame is Rendered by QuickLaTeX.com.

So, the **frame transmission time (Tfr) should be at least twice the maximum propagation time (Tp)**. This can be deduced when the two stations involved in collision are maximum distance apart.

**Program:**

set ns [new Simulator]

#Define different colors for data flows (for NAM)

$ns color 1 Blue

$ns color 2 Red

#Open the Trace files

set file1 [open out.tr w]

set winfile [open WinFile w]

$ns trace-all $file1

#Open the NAM trace file

set file2 [open out.nam w]

$ns namtrace-all $file2

#Define a 'finish' procedure

proc finish {} {

global ns file1 file2

$ns flush-trace

close $file1

close $file2

exec nam out.nam &

exit 0

}

#Create six 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]

$n1 color red

$n1 shape box

#Create links between the nodes

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

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

$ns simplex-link $n2 $n3 0.3Mb 100ms DropTail

$ns simplex-link $n3 $n2 0.3Mb 100ms DropTail

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

# $ns duplex-link $n3 $n4 0.5Mb 40ms DropTail

# $ns duplex-link $n3 $n5 0.5Mb 30ms DropTail

#Give node position (for NAM)

# $ns duplex-link-op $n0 $n2 orient right-down

# $ns duplex-link-op $n1 $n2 orient right-up

# $ns simplex-link-op $n2 $n3 orient right

# $ns simplex-link-op $n3 $n2 orient left

# $ns duplex-link-op $n3 $n4 orient right-up

# $ns duplex-link-op $n3 $n5 orient right-down

#Set Queue Size of link (n2-n3) to 10

# $ns queue-limit $n2 $n3 20

#Setup a 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 window\_ 8000

$tcp set packetSize\_ 552

#Setup a FTP over TCP connection

set ftp [new Application/FTP]

$ftp attach-agent $tcp

$ftp set type\_ FTP

#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

$ns at 0.1 "$cbr start"

$ns at 1.0 "$ftp start"

$ns at 124.0 "$ftp stop"

$ns at 124.5 "$cbr stop"

# next procedure gets two arguments: the name of the

# tcp source node, will be called here "tcp",

# and the name of output file.

proc plotWindow {tcpSource file} {

global ns

set time 0.1

set now [$ns now]

set cwnd [$tcpSource set cwnd\_]

set wnd [$tcpSource set window\_]

puts $file "$now $cwnd"

$ns at [expr $now+$time] "plotWindow $tcpSource $file" }

$ns at 0.1 "plotWindow $tcp $winfile"

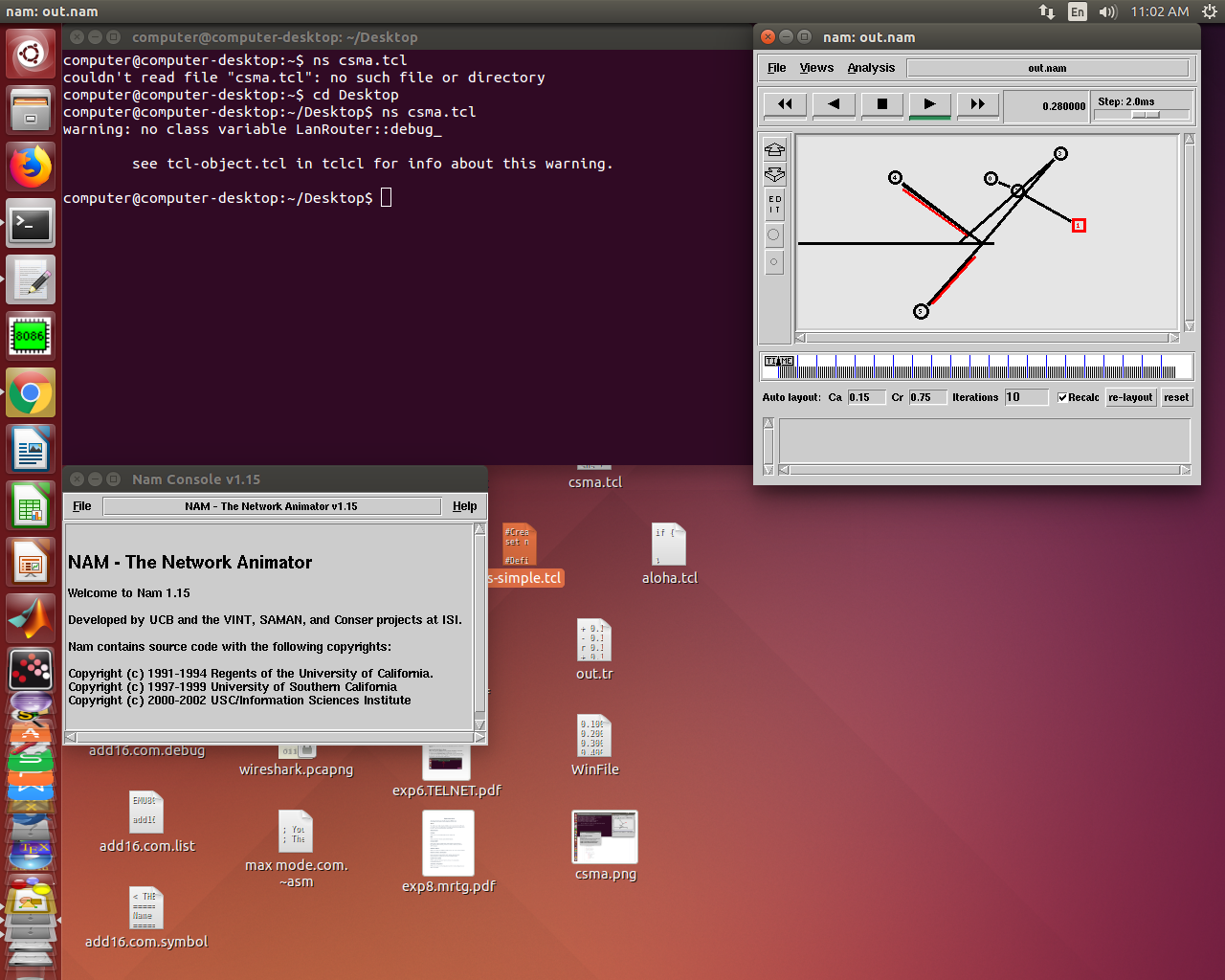
$ns at 5 "$ns trace-annotate \"packet drop\""

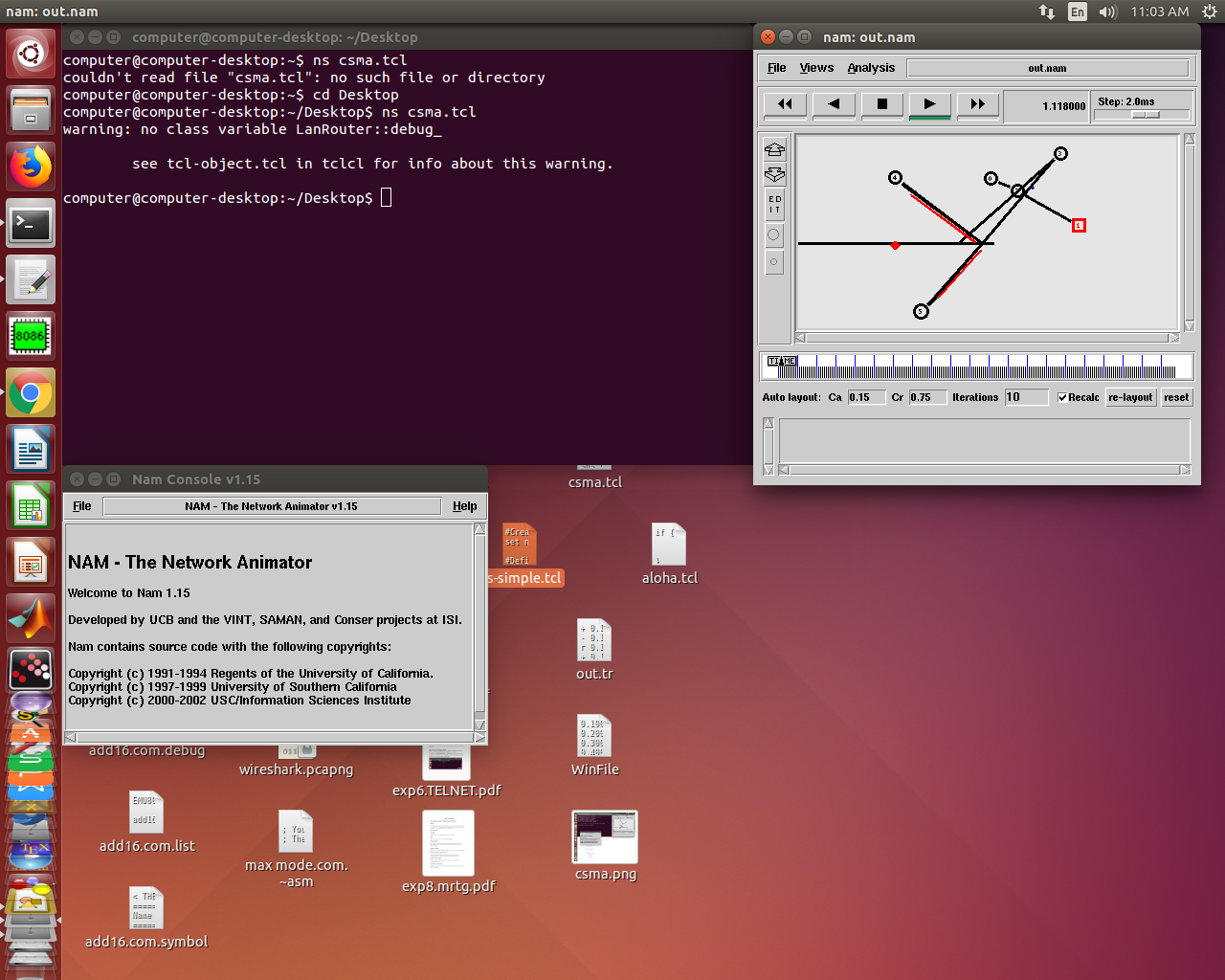
# PPP

$ns at 125.0 "finish"

$ns run

**Output:**

****

****

****

**Conclusion:**

Thus we understand functioning of ALOHA,CSMA/CD.