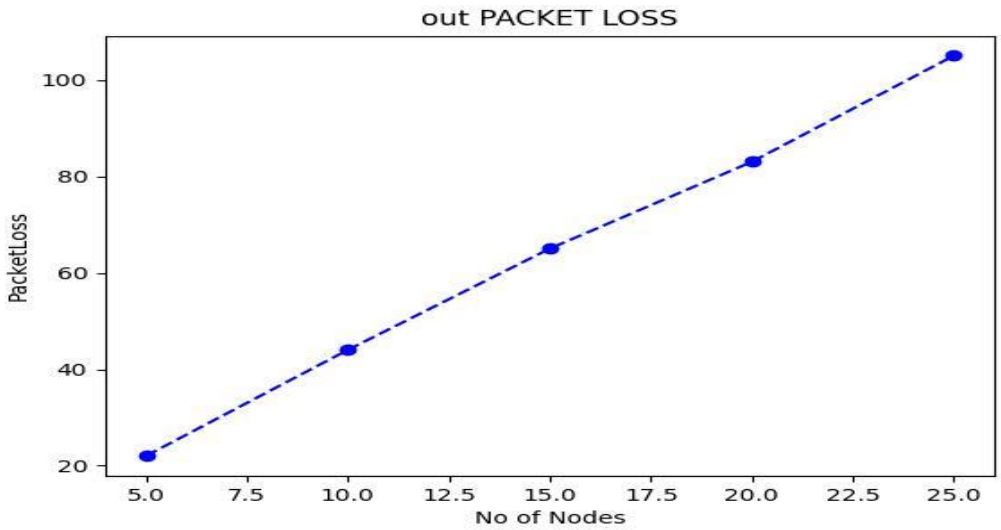
LAB-4, NS2 SIMULATOR

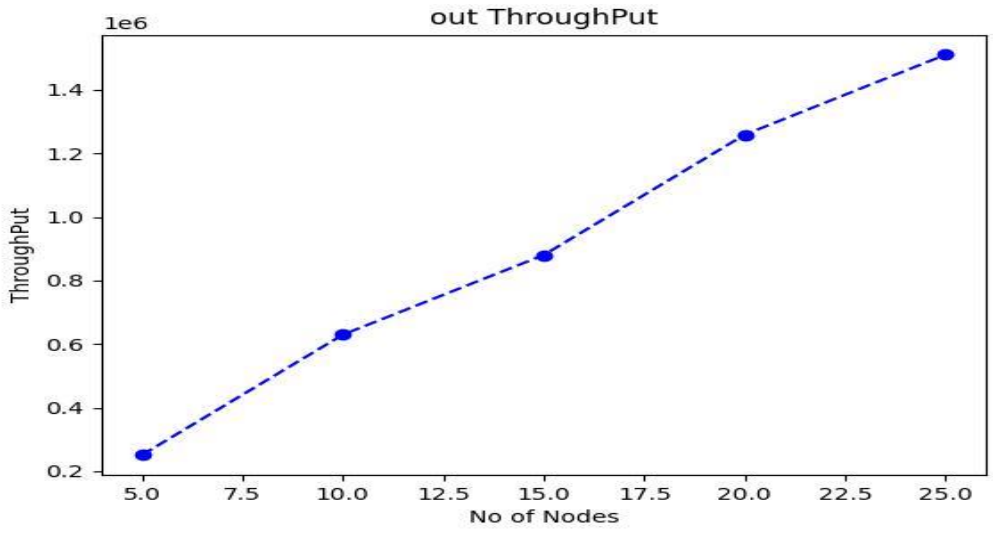
a) Ring Topology:

```
#This Program will create a ring topology using less number of statements in
TCL Language
set ns [new Simulator]
$ns rtproto DV
set nf [open "out.nam" w]
set tr [ open "out25.tr" w]
$ns trace-all $tr
$ns namtrace-all $nf
set num 25
for {set i 1} {$i<$num} {incr i 2} {
        set ftp($i) [new Application/FTP]
        }
proc finish {} {
        global ns nf tr
        $ns flush-trace
        close $nf
        close $tr
        exec nam out.nam &
        exit 0
        }
#Creating Nodes
set packet_size 1024
for {set i 0} {$i<$num} {incr i} {
set n($i) [$ns node]
}
#Creating Links
for {set i 0} {$i<$num} {incr i} {
$ns duplex-link $n($i) $n([expr ($i+1)%$num]) 1Mb 10ms DropTail
}
for {set i 0} {$i < $num} {incr i} {
    if {$i % 2 == 0} {
        # Even-numbered nodes are destinations
        set sink($i) [new Agent/TCPSink]
        $ns attach-agent $n($i) $sink($i)
        #if { $i != 0 } {
        #$ns connect $tcp1([expr $i-1]) $sink($i)
        #}
        } else {
        # Odd-numbered nodes are sources
        set tcp1($i) [new Agent/TCP]
        $ns attach-agent $n($i) $tcp1($i)
        $ftp($i) attach-agent $tcp1($i)
    }
}
for {set i 0} {$i<$num} {incr i} {
```

```
        if {$i%2!=0} {
                $ns connect $sink([expr $i-1]) $tcp1($i)
                #$ns connect $sink([expr $i+1]) $tcp1($i)
        $tcp1($i) set packetSize_ $packet_size
        $tcp1($i) set rate_ [expr {$packet_size / 0.001}]
        $tcp1($i) set interval_ 0.005
        $ns at 0.1 "$ftp($i) start"
        $ns at 10 "$ftp($i) stop"
        }
        }
$ns queue-limit $n(3) $n(2) 10
$ns queue-limit $n(9) $n(8) 10
$ns queue-limit $n(13) $n(12) 10
$ns queue-limit $n(17) $n(16) 10
$ns queue-limit $n(21) $n(20) 10
$ns at 10.1 "finish"
$ns run
```

OUTPUT:

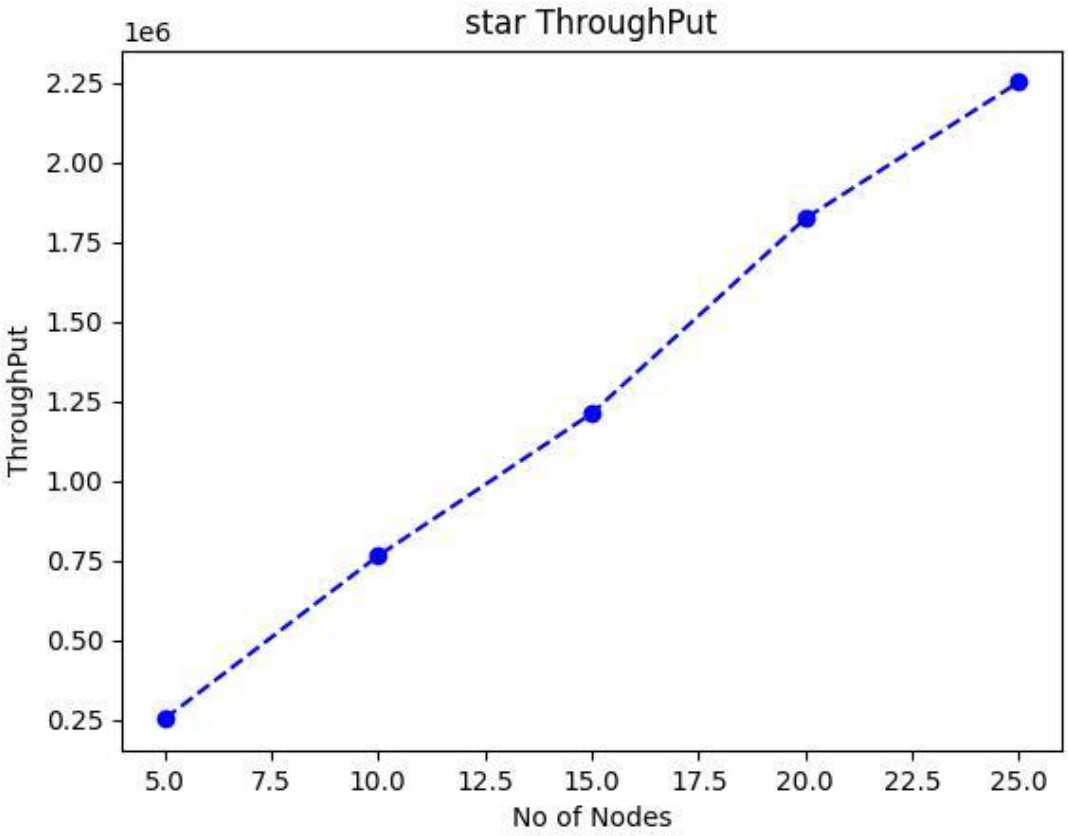


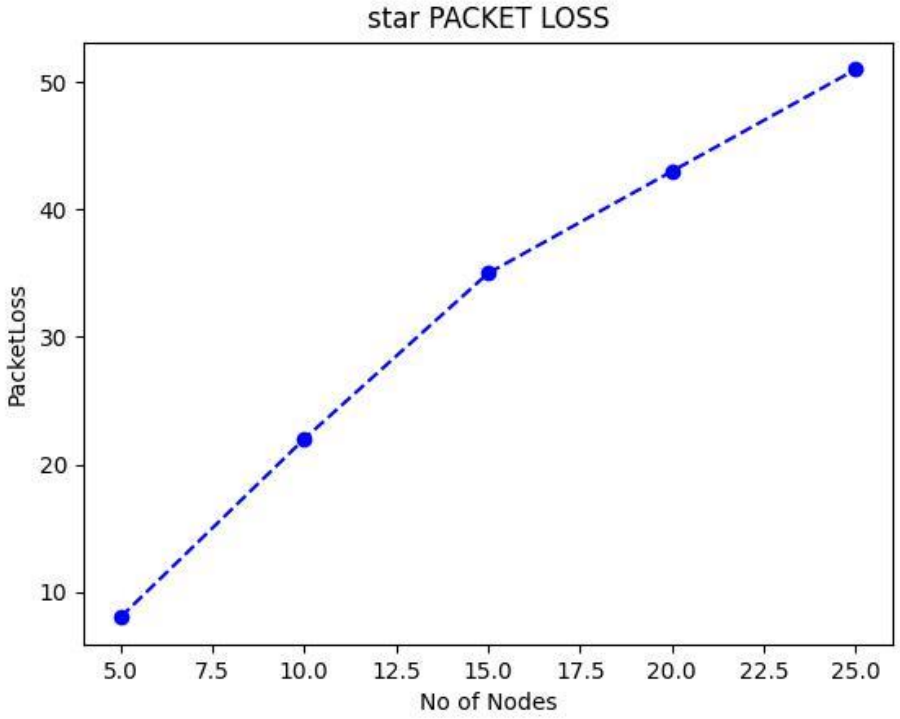


b) Star Topology

```
set noOfNodes 25
set ns [new Simulator]
set tr [open star25.tr w ]
$ns trace-all $tr
set ftr [open out.nam w ]
$ns namtrace-all $ftr
proc finish { } {
    global tr ftr ns
    $ns flush-trace
    close $tr
    close $ftr
    exec nam out.nam &
    exit 0
}
for {set i 0} {$i < $noOfNodes} {incr i} {
    set n($i) [ $ns node ]
}
for {set i 1} {$i<$noOfNodes} {incr i} {
    $ns duplex-link $n(0) $n($i) 1Mb 10ms DropTail
}
for {set i 0} {$i<$noOfNodes} {incr i} {
    if { $i % 2 == 0 } {
        set tcp($i) [new Agent/TCP]
        $ns attach-agent $n($i) $tcp($i)
    } else {
        set sink($i) [new Agent/TCPSink]
        $ns attach-agent $n($i) $sink($i)
    }
}
for {set i 1} {$i<$noOfNodes} {incr i} {
    if { $i % 2 != 0 } {
        $ns connect $sink($i) $tcp([expr $i-1])
        set ftp([expr $i-1]) [new Application/FTP]
        $ftp([expr $i-1])  attach-agent $tcp([expr $i-1])
        $ns at 0.$i "$ftp([expr $i-1]) start"
    }
}
$ns queue-limit $n(2) $n(0) 3
$ns queue-limit $n(6) $n(0) 5
$ns queue-limit $n(8) $n(0) 5
$ns queue-limit $n(12) $n(0) 6
$ns queue-limit $n(16) $n(0) 3
$ns queue-limit $n(22) $n(0) 5
$ns at 3.0 "finish"
$ns run
```

OUTPUT:


star ThroughPut
1e6
2.25
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
ThroughPut
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
No of Nodes


star PACKET LOSS
50
40
30
20
10
PacketLoss
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
No of Nodes

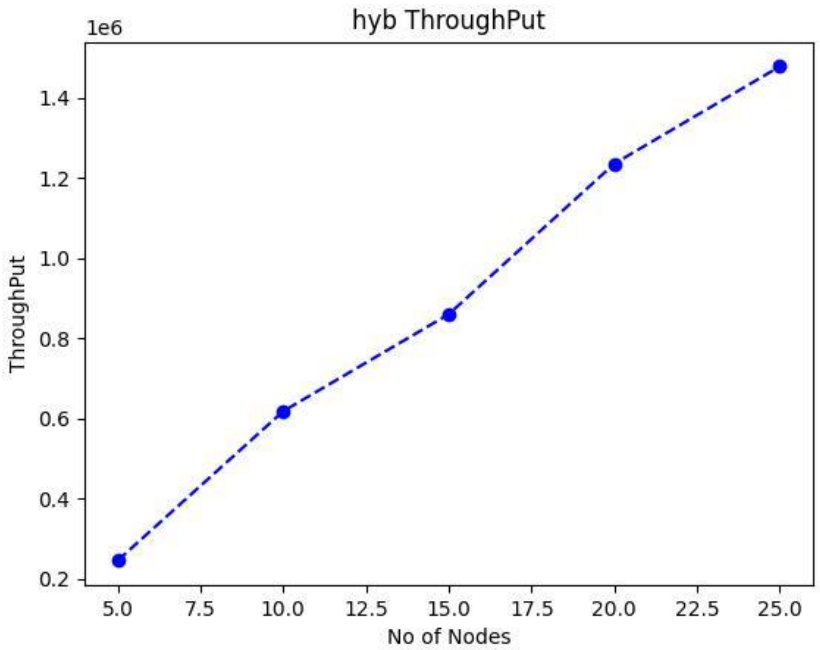
c) Hybrid Topology:

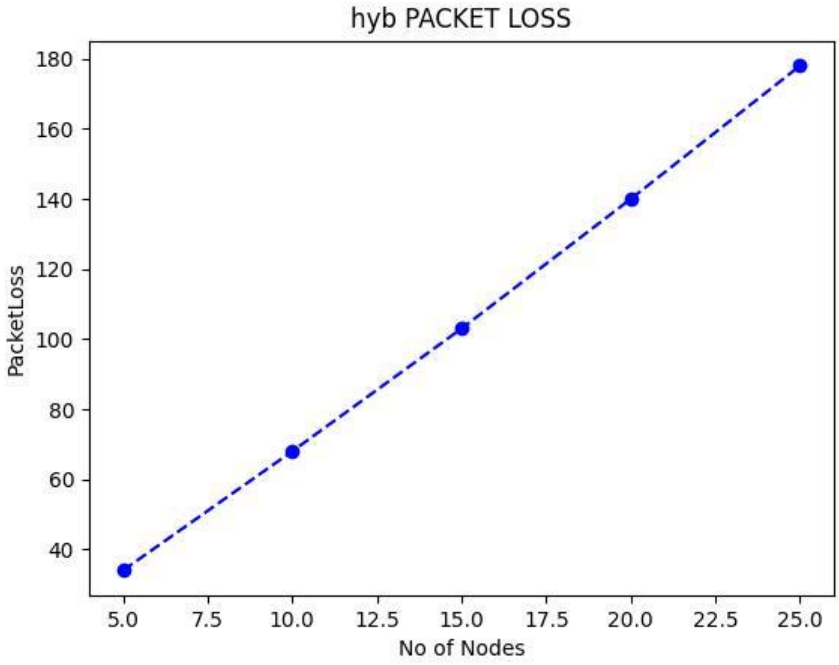
```
#This Program will create a ring topology using less number of statements in
TCL Language
set ns [new Simulator]
$ns rtproto DV
set nf [open "out.nam" w]
set tr [ open "hyb25.tr" w]
$ns namtrace-all $nf
$ns trace-all $tr
set num 25
set center 0
set ne 1
for {set i 1} {$i<$num} {incr i 2} {
        set ftp($i) [new Application/FTP]
        }

proc finish {} {
        global ns nf tr
        $ns flush-trace
        close $nf
        close $tr
        exec nam out.nam &
        exit 0
        }
#Creating Nodes
set packet_size 1024
for {set i 0} {$i<$num} {incr i} {
set n($i) [$ns node]
}
#Creating Links
for {set i 1} {$i<$num} {incr i} {
$ns duplex-link $n($i) $n([expr ($i+1)%$num]) 1Mb 10ms DropTail
}
for {set i 1} {$i<$num} {incr i} {
$ns duplex-link $n($center) $n($i) 1Mb 10ms DropTail
}
$ns duplex-link $n($ne) $n([expr ($num-1)]) 1Mb 10ms DropTail
for {set i 0} {$i < $num} {incr i} {
    if {$i % 2 == 0} {
        # Even-numbered nodes are destinations
        set sink($i) [new Agent/TCPSink]
        $ns attach-agent $n($i) $sink($i)
        #if { $i != 0 } {
        #$ns connect $tcp1([expr $i-1]) $sink($i)
        #}
        } else {
        # Odd-numbered nodes are sources
        set tcp1($i) [new Agent/TCP]
```

```
        $ns attach-agent $n($i) $tcp1($i)
        $ftp($i) attach-agent $tcp1($i)
    }
}
for {set i 0} {$i<$num} {incr i} {
        if {$i%2!=0} {
                $ns connect $sink([expr $i-1]) $tcp1($i)
                #$ns connect $sink([expr $i+1]) $tcp1($i)
        $tcp1($i) set packetSize_ $packet_size
        $tcp1($i) set rate_ [expr {$packet_size / 0.001}]
        $tcp1($i) set interval_ 0.005
        $ns at 0.1 "$ftp($i) start"
        $ns at 10 "$ftp($i) stop"
        }
        }
$ns queue-limit $n(1) $n(0) 5
$ns queue-limit $n(7) $n(6) 5
$ns queue-limit $n(11) $n(10) 5
$ns queue-limit $n(19) $n(18) 5
$ns queue-limit $n(23) $n(22) 5
$ns at 10.1 "finish"
$ns run
```

OUTPUT:





PLOT FILE:

```
from matplotlib import pyplot as plt
def printVal(f,name):
    print(name)
    data = f.read()
    data = data.split("\n")
    startTime = float('inf')
    endTime = float('-inf')
    recv = 0
    drop = 0
    for dat in data:
        t = dat.split()
        if not t:
            continue
        if t[0] == 'r':
            recv += int(t[5])
        if t[0] == 'd':
            drop += 1
        startTime = min(startTime,float(t[1]))
        endTime = max(endTime, float(t[1]))
    return recv/(endTime - startTime),drop
L = ['out','star','hyb']
Y = [5,10,15,20,25]
for i in L:
    pX = []
    tX = []
    for j in range(1,6):
```

```
        filename = (f"{i}{j*5}.tr")
        f = open(filename,'r')
        t = printVal(f,i + " " + str(j*5))
        pX.append(t[1])
        tX.append(t[0])
    plt.plot(Y,pX,'--bo')
    plt.ylabel("PacketLoss")
    plt.xlabel("No of Nodes")
    plt.title(f"{i} PACKET LOSS")
    plt.savefig(f"./graphs/{i}_PACKET_LOSS.jpg")
    plt.figure()
    plt.plot(Y,tX,'--bo')
    plt.ylabel("ThroughPut")
    plt.xlabel("No of Nodes")
    plt.title(f"{i} ThroughPut")
    plt.savefig(f"./graphs/{i}_ThroughPut.jpg")
    plt.figure()
```