Vaibhav Goel

Original Title: VaibhavGoel

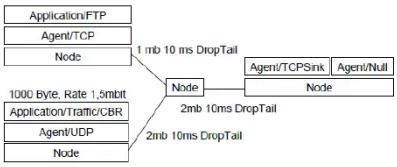
ns2 recordssjdj **Full description**

– Sender agent: Agent/UDP
– Receiver agent: Agent/Null
– Connect agents
– Data source: Application/Traffic/CBR
– Run from 0.5 to 4.5 sec, finish at 5.0 sec


```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out2a.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out2a.nam &
        exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n1 2Mb 10ms DropTail

#Set Queue Size of link (n0-n1) to 10
```

```
$ns attach-agent $n0 $udp
set null [new Agent/Null]
$ns attach-agent $n1 $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_ 1.5mb
$cbr set random_ false


#Schedule events for the CBR
$ns at 0.5 "$cbr start"
$ns at 4.5 "$cbr stop"

#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

#Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"

#Run the simulation
$ns run
```

Visualize the bottle neck queue




```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out2b.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out2b.nam &
        exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
```

```
$ns duplex-link-op $n2 $n3 orient right

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5


#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#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 $n3 $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_ 1.5mb
$cbr set random_ false


#Schedule events for the CBR and FTP agents
$ns at 1.0 "$cbr start"
```

```
#Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"

#Run the simulation
$ns run
```

```
Assume typical values for other parameters.
set val(stop)   150.0
set val(file_size)      [expr 10*1024*1024]

#Create a ns simulator
set ns [new Simulator]

#Open the NS trace file
set tracefile [open ftp.tr w]
$ns trace-all $tracefile

#Create 2 nodes
set n0 [$ns node]
set n1 [$ns node]

#Createlinks between nodes
$ns duplex-link $n0 $n1 5.0Mb 50ms DropTail
$ns queue-limit $n0 $n1 10


#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0

set sink1 [new Agent/TCPSink]
$ns attach-agent $n1 $sink1
$ns connect $tcp0 $sink1
$tcp0 set packetSize_ 1500


#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ftp0 set type_ FTP

$ns at 1.5 "$ftp0 send $val(file_size)"

#Define a 'finish' procedure
proc finish {} {
    global ns tracefile
    $ns flush-trace
    close $tracefile
    exit 0
}

# Schedule events
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"

$ns run
```

n2 through n4. The duration of the simulation time is 10 seconds.

```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out2d.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out2d.nam &
        exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n4 2Mb 10ms DropTail
$ns duplex-link $n1 $n4 2Mb 10ms DropTail
$ns duplex-link $n2 $n4 2Mb 10ms DropTail
$ns duplex-link $n3 $n4 2Mb 10ms DropTail

#Set Queue Size of link (n2-n3) to 10
#$ns queue-limit $n2 $n3 10
```

```
#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP
$ftp set packet_size_ 1000
$ftp set rate_ 1mb
$ftp set random_ false

#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n2 $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_ 1mb
$cbr set random_ false


#Schedule events for the CBR and FTP agents
$ns at 0.0 "$cbr start"
$ns at 0.0 "$ftp start"
$ns at 10.0 "$ftp stop"
$ns at 10.0 "$cbr stop"
```

```
#Run the simulation
$ns run
```

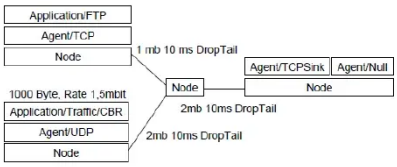
Consider the LAN with seven nodes to be an isolated one i.e. not connected to the Internet. Node # 0 in the LAN act as a UDP traffic source, and node # 6 is the destination node. Assume CBR traffic to be flowing between the nodes. The simulation lasts for 25 seconds. In Ethernet a packet is broadcasted in the shared medium, and only the destination node accepts the packet. Other nodes simply drop it. How many hops a packet should take to travel from node # 0 to node # 6? Verify this from the "Hop Count" plot.

```
set ns [new Simulator]

# Open tracefile
set tracefile [open out.tr w]
$ns trace-all $tracefile

# Define the finish procedure
proc finish {} {
   global ns tracefile
   $ns flush-trace
   close $tracefile
   exit 0
}

# Create the 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]


# Setup LAN
set lan [$ns newLan "$n0 $n1 $n2 $n3 $n4 $n5 $n6" 1Mb 40ms LL Queue/DropTail MAC/Csma/Cd
Channel]

# A gateway
$ns duplex-link $n0 $n7 1.0Mb 50ms DropTail


# Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n7 $udp
set null [new Agent/Null]

$ns attach-agent $n6 $null
$ns connect $udp $null

# Setup a CBR over UDP connection
```

– Connect agents
– Data source: Application/Traffic/CBR
– Run from 0.5 to 4.5 sec, finish at 5.0 sec


```
set ns [ new Simulator ]
set nf  [ open o.nam w ]
$ns namtrace-all $nf
set n0 [ $ns node ]
set n1 [ $ns node ]
$ns duplex-link $n0 $n1 2mb 10ms DropTail
$ns queue-limit $n0 $n1 10
$ns duplex-link-op $n1 $n0 queuePos 0.5

set udp [ new Agent/UDP ]
$ns attach-agent $n0 $udp


set cbr [ new Application/Traffic/CBR ]
$cbr attach-agent $udp
$cbr set packet_size_ 1000
$cbr set rate_ 1.5Mb
$cbr set random_ false

set null1 [ new Agent/Null ]
$ns attach-agent $n1 $null1

$ns connect $udp $null1

$ns at 0.1 "$cbr start"
$ns at 4.5 "$cbr stop"

proc finish { } {
global ns nf
$ns flush-trace
close $nf
exec nam o.nam
exit 0
}

$ns at 5.0 "finish"
$ns run
```

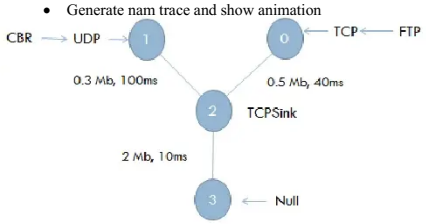
Visualize the bottle neck queue




```
set ns [ new Simulator ]
set nf  [ open o.nam w ]
$ns namtrace-all $nf

set n0 [ $ns node ]
set n1 [ $ns node ]
set n2 [ $ns node ]
set n3 [ $ns node ]

$ns duplex-link $n0 $n2 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n3 $n2 2Mb 10ms DropTail

$ns duplex-link-op $n2 $n0 orient left-up
$ns duplex-link-op $n2 $n1 orient left-down
$ns duplex-link-op $n2 $n3 orient right

set tcp [ new Agent/TCP ]
$ns attach-agent $n0 $tcp
set udp [ new Agent/UDP ]
$ns attach-agent $n1 $udp

set ftp [ new Application/FTP ]
$ftp attach-agent $tcp
set cbr [ new Application/Traffic/CBR ]
$cbr attach-agent $udp
$cbr set packet_size_ 1000
$cbr set rate_ 1.5Mb
$cbr set random_ false

set null1 [ new Agent/LossMonitor ]
$ns attach-agent $n3 $null1
set null2 [ new Agent/LossMonitor ]
$ns attach-agent $n3 $null2
```

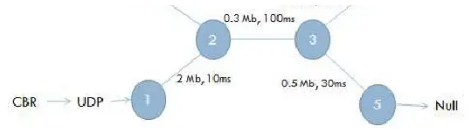
```
close $nf
exec nam o.nam
exit 0
}

$ns at 5.0 "finish"
$ns run
```

- CBR traffic packet size: 1000B, inter-arrival time: 8ms, start at time 1.0; TCP window size 8000, packet size 512B
- Generate nam trace and show animation




```
set ns [ new Simulator ]
set nf  [ open o.nam  w ]
$ns namtrace-all $nf

set n0 [ $ns node ]
set n1 [ $ns node ]
set n2 [ $ns node ]
set n3 [ $ns node ]

$ns simplex-link $n0 $n2 0.5Mb 40ms DropTail
$ns duplex-link $n1 $n2 0.3Mb 100ms DropTail
$ns duplex-link $n3 $n2 2Mb 10ms DropTail
$ns queue-limit $n2 $n3 40
$ns duplex-link-op $n3 $n2 queuePos 0.5

$ns simplex-link-op $n2 $n0 orient right-up
$ns duplex-link-op $n2 $n1 orient left-up
$ns duplex-link-op $n2 $n3 orient down

#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP connection
```

```
set null [new Agent/Null]
$ns attach-agent $n3 $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_ 1mb
$cbr set random_ false


#Schedule events for the CBR and FTP agents
$ns at 0.5 "$ftp start"
$ns at 4.5 "$ftp stop"
$ns at 1.0 "$cbr start"
$ns at 4.5 "$cbr stop"

proc finish { } {
global ns nf
$ns flush-trace
close $nf
exec nam o.nam
exit 0
}

$ns at 5.0 "finish"
$ns run
```




```
set ns [ new Simulator ]
set nf  [ open o.nam w ]
$ns namtrace-all $nf

set n0 [ $ns node ]
set n1 [ $ns node ]
set n2 [ $ns node ]
set n3 [ $ns node ]
set n4 [ $ns node ]
set n5 [ $ns node ]

$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 0.3Mb 100ms DropTail
$ns duplex-link $n3 $n4 0.5Mb 40ms DropTail
$ns duplex-link $n3 $n5 0.5Mb 30ms DropTail

$ns duplex-link-op $n2 $n0 orient left-up
$ns duplex-link-op $n2 $n1 orient left-down
$ns duplex-link-op $n2 $n3 orient right
$ns duplex-link-op $n3 $n4 orient right-up
$ns duplex-link-op $n3 $n5 orient right-down

#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP
$ftp set packet_size_ 1000
```

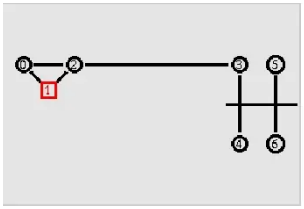
```
#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false


#Schedule events for the CBR and FTP agents
$ns at 0.5 "$ftp start"
$ns at 4.5 "$ftp stop"
$ns at 1.0 "$cbr start"
$ns at 4.5 "$cbr stop"

proc finish { } {
global ns nf
$ns flush-trace
close $nf
exec nam o.nam
exit 0
}

$ns at 5.0 "finish"
$ns run
```

```
#Create a simulator object
        set ns [new Simulator]

        #Define different colors for data flows (for NAM)
        #$ns color 1 Green
        #$ns color 2 Red

        #Open the NAM trace file
        set nf [open out.nam w]
        $ns namtrace-all $nf

        #Define a 'finish' procedure
        proc finish {} {
              global ns nf
              $ns flush-trace
              #Close the NAM trace file
              close $nf
              #Execute NAM on the trace file
              exec nam out.nam &
              exit 0
        }

              #Create four nodes

    for { set i 0 } { $i<10 } { incr i } {
  set n($i) [ $ns node ]
}


        #Create links between the nodes
    for { set i 0 } { $i<9} { incr i } {
    $ns duplex-link $n($i) $n([ expr $i+1 ]) 1Mb 10ms DropTail
    }
$ns duplex-link $n(9) $n(0) 1Mb 10ms DropTail

        #Set Queue Size of link (n2-n3) to 10
        #$ns queue-limit $n2 $n3 5

        #Give node position (for NAM)
        $ns duplex-link-op $n(0) $n(1) orient right-up
        #$ns duplex-link-op $n(4) $n(5) orient right-bottom
        #$ns duplex-link-op $n(5) $n(6) orient left-bottom
     #$ns duplex-link-op $n(9) $n(0) orient right-up
     for {set i 1} {$i < 4} {incr i} {
  $ns duplex-link-op $n($i) $n([ expr $i+1 ]) orient right
}
```

```
#Monitor the queue for link (n2-n3). (for NAM)
#$ns duplex-link-op $n2 $n3 queuePos 0.5


#Setup a TCP connection
#set tcp [new Agent/TCP]
#$tcp set class_ 2
#$ns attach-agent $n(0) $tcp
#set sink [new Agent/TCPSink]
#$ns attach-agent $n(5) $sink
#$ns connect $tcp $sink
#$tcp set fid_ 1

#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 $n(0) $udp
set null [new Agent/Null]
$ns attach-agent $n(5) $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_ 1.5mb
$cbr set random_ false


#Schedule events for the CBR and FTP agents
$ns at 1.0 "$cbr start"
#ns at 0.5 "$ftp start"
#$ns at 9.5 "$ftp stop"
$ns at 10.0 "$cbr stop"

#Detach tcp and sink agents (not really necessary)
#$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"

#Call the finish procedure after 5 seconds of simulation time
$ns at 10.0 "finish"
```

Experiment 3 : Simulating link errors using ns2

Consider the following topology:



Here node # 2 act as a router. Any traffic to or from the LAN passes through it. Consider node # 1 running a FTP server, and node # 5 is downloading a file of size 4 MB. However, the link between node # 2 and # 3 is fault. It drops packets with a fixed probability of 0.2. Implement a link error model to reflect this.

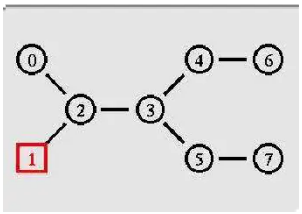
It may be noted here that the file download time will be more than the we had in exercise # 2 of experiment # 1. Try different values of the simulation time to ensure that the file has been entirely transferred. Is the plot of bytes received a linear curve or non-linear? Why?

Presence of link errors cause one or more packets to be retransmitted. Verify this from the "Packet Retransmissions" plot.

```
set val(file_size)    [expr 4*1024*1024];    #  Send a file of size 4 MB

set ns [new Simulator]

# Open tracefile
set tracefile [open out.tr w]
$ns trace-all $tracefile

# Define the finish procedure
proc finish {} {
   global ns tracefile
   $ns flush-trace
   close $tracefile
   exit 0
}
```

```
$ns duplex-link $n0 $n1 2Mb 10ms DropTail
$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 queue size of link(n2-n3) to 20
$ns queue-limit $n2 $n3 20

# Set up the LAN
set lan [$ns newLan "$n3 $n4 $n5 $n6" 0.5Mb 40ms LL Queue/DropTail MAC/Csma/Cd
Channel]

# Set error model
set loss_module [new ErrorModel]
$loss_module set rate_ 0.2
$loss_module ranvar [new RandomVariable/Uniform]
$loss_module drop-target [new Agent/Null]
$ns lossmodel $loss_module $n2 $n3

# Setup TCP connection
set tcp [new Agent/TCP]
$ns attach-agent $n1 $tcp

set sink [new Agent/TCPSink]
$ns attach-agent $n5 $sink
$ns connect $tcp $sink
$tcp set packet_size_ 1500

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

# Scheduling the events
$ns at 1.5 "$ftp send $val(file_size)"

$ns at 2000.0 "finish"

$ns run
```

1. **Simulating Bottleneck in the network using ns2:**
   Consider a dumbbell topology with eight nodes as shown as in the following figure. Consider nodes # 2 and 3 to be two routers connecting two different networks. When the bandwidth of the link 2-3 is much lower than the sum of bandwidths of the other links in the network, it act as a bottleneck.



Assume node # 0 running a FTP application (over TCP) and sending data to node # 6. Node # 1 is sending CBR data node # 7. Assume all the links except 2-3 has a bandwidth of 1 Mb, propagation delay of 10ms and queue type as DropTail. (All are duplex links).

Tasks:

- The link 2-3 has a propagation delay of 10 ms. Vary it's bandwidth from 0.5 Mb to 2.5 Mb in steps of 0.25Mb.
- Compute the throughput for node # 3 in each case
- Plot the throughput vs. bandwidth data in the "Custom Plot" section below

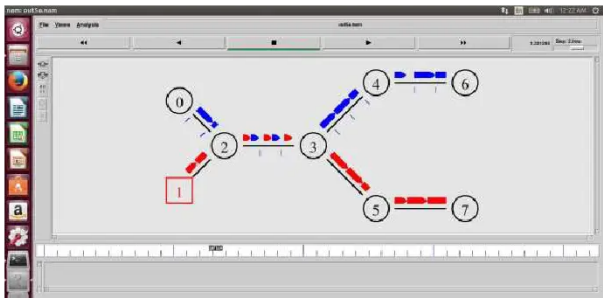
Based on the above plots, suggest what should be the recommended bandwidth of the link 2-3.

Now, plot the end-to-end delay between nodes 0 and 6 for the above chosen values of link 2-3 bandwidth. Revisit your previous answer (i.e. optimum bandwidth of link 2-3) based on these graphs.

```
set ns [new Simulator]

#open tracefiles
set tracefile1 [open out.tr w]
$ns trace-all $tracefile1
```

```
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]

#create links between the nodes
$ns duplex-link $n0 $n2 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n2 $n3 0.5Mb 10ms RED
$ns duplex-link $n3 $n4 1Mb 10ms DropTail
$ns duplex-link $n3 $n5 1Mb 10ms DropTail
$ns duplex-link $n4 $n6 1Mb 10ms DropTail
$ns duplex-link $n5 $n7 1Mb 10ms DropTail

#set queue size of link(n2-n3) to 10
$ns queue-limit $n2 $n3 10

#setup TCP connection
set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n6 $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 $n7 $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 interval_ 0.005
$cbr set random_ false
```

Output:



Part-II Bandwidth sharing between TCP and UDP

Consider the dumbbell topology from our previous exercise: Dumbbell

topology with bottleneck

Node # 0 is a TCP source, and the corresponding sink is at node # 6. Node # 1 is a UDP source (CBR traffic) with a null agent attached to node # 7. These two traffic flows through the common link 2-3. The aim of this exercise is to examine how TCP and UDP share the bandwidth between themselves when the rate of CBR traffic is changed.

Set the TCP packet size to 1460 B. The UDP and CBR packet sizes are 1500 B. All the links in the network have same bandwidths (say, 4 Mb), delay and queue types.

Part 1:

Set the initial rate of CBR traffic to 0.5 Mb. Run the simulation, and plot the "Bytes Received" by node #s 4 and 5 (sinks for TCP and UDP traffic)

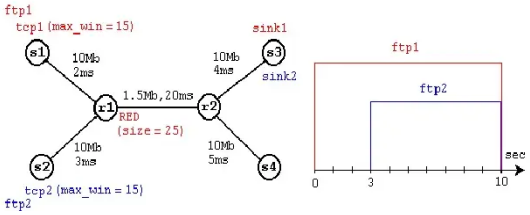
Now, increment the rate up to 4 Mb, the link bandwidth, in steps of 0.5 Mb. Run the simulation and plot the graphs again.

How does the graphs change after each run? In particular, what's the nature of the

```
set ns [new Simulator]

# Open tracefiles
set tracefile1 [open out.tr w]
$ns trace-all $tracefile1


# Define the finish procedure
proc finish {} {
    global ns tracefile1
    $ns flush-trace
    close $tracefile1
    exit 0
}

# Create 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]

# Create links between the nodes
$ns duplex-link $n0 $n2 4Mb 10ms DropTail
$ns duplex-link $n1 $n2 4Mb 10ms DropTail
$ns duplex-link $n2 $n3 4Mb 10ms DropTail
$ns duplex-link $n3 $n4 4Mb 10ms DropTail
$ns duplex-link $n3 $n5 4Mb 10ms DropTail
$ns duplex-link $n4 $n6 4Mb 10ms DropTail
$ns duplex-link $n5 $n7 4Mb 10ms DropTail


# Set queue size of link(n2-n3) to 10
$ns queue-limit $n2 $n3 20

# Setup TCP connection
set tcp [new Agent/TCP/Reno]
$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n6 $sink
$ns connect $tcp $sink
```

```
set null [new Agent/Null]
ns attach-agent $n7 $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 rate 1Mb
$cbr set random_ false

$udp set packetSize_ 1500
$cbr set packetSize_ 1500

# Scheduling the events
$ns at 1.0 "$ftp start"
$ns at 1.5 "$cbr start"
$ns at 24.0 "$cbr stop"
$ns at 24.5 "$ftp stop"

$ns at 25.0 "finish"

$ns run
```

2. Set up the network topology and run the simulation scenario as shown below



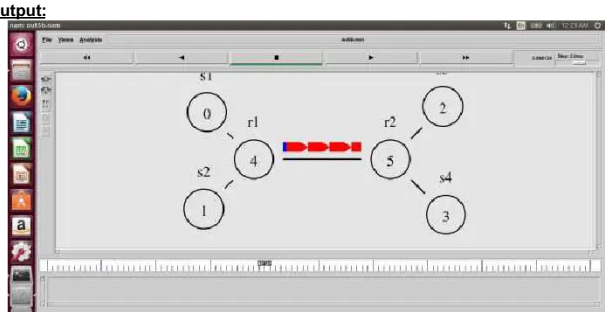

```
#Create a simulator
object set ns [new
```

```
#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace
        file close $nf
        #Execute NAM on the trace file
        exec nam out5b.nam &
        exit 0
}

#Create six
nodes set s1 [$ns
node] set s2 [$ns
node] set s3 [$ns
node] set s4 [$ns
node] set r1 [$ns
node] set r2 [$ns
node]

#Create links between the nodes
$ns duplex-link $s1 $r1 10Mb 2ms DropTail
$ns duplex-link $s2 $r1 10Mb 3ms DropTail
$ns duplex-link $r1 $r2 1.5Mb 20ms DropTail
$ns duplex-link $r2 $s3 10Mb 4ms DropTail
$ns duplex-link $r2 $s4 10Mb 5ms DropTail

#Labelling
$ns at 0.0 "$s1 label s1"
$ns at 0.0 "$s2 label s2"
$ns at 0.0 "$s3 label s3"
$ns at 0.0 "$s4 label s4"
$ns at 0.0 "$r1 label r1"
$ns at 0.0 "$r2 label r2"


#Set Queue Size of link (r1-r2) to
25 $ns queue-limit $r1 $r2 25

#Give node position (for NAM)
$ns duplex-link-op $s1 $r1 orient right-down
$ns duplex-link-op $s2 $r1 orient right-up
$ns duplex-link-op $r1 $r2 orient right
$ns duplex-link-op $r2 $s3 orient right-up
$ns duplex-link-op $r2 $s4 orient right-down

#Monitor the queue for link (n2-n3). (for NAM)
#$ns duplex-link-op $n2 $n3 queuePos 0.5


#Setup        a        TCP
```

```
set sink [new Agent/TCPSink]
$ns attach-agent $s3 $sink $ns
connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP
connection set ftp1 [new
Application/FTP] $ftp1 attach-
agent $tcp
$ftp1 set type_ FTP
$ftp1 set packet_size_ 15

#Setup a TCP
connection set tcp [new
Agent/TCP] $tcp set
class_ 2
$ns attach-agent $s2 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $s3 $sink $ns
connect $tcp $sink
$tcp set fid_ 2

#Setup a FTP over TCP
connection set ftp2 [new
Application/FTP] $ftp2 attach-
agent $tcp
$ftp2 set type_ FTP
$ftp2 set packet_size_ 15

#Schedule events for the CBR and FTP agents
$ns at 0.0 "$ftp1 start"
$ns at 3.0 "$ftp2 start"
$ns at 10.0 "$ftp1 stop"
$ns at 10.0 "$ftp2 stop"
```

#Run the
simulation $ns run

Output:



Kory Findley talks about h…
26 min listen

MacLife
2 min read

Diagnose Persistent Problems

Magazines
Podcasts
Sheet Music

12holi11-12
Niti Arora

The Subtle Art of Not Giving a F*ck: A Counterintuitive Approach to Living a Good…
Mark Manson

The IPv6 Switch
Communicatio
internet occurs via small…

Network Topology

Computer Network

Transmission Control Protocol

Internet Standards

MacFormat
1 min read

Diagnose Persistent Problems

Network Administrator Technical Interview Questions
kaleebu

Never Split the Difference: Negotiating As If Your Life Depended On It
Chris Voss

Heavy Network Broadcast Medi Networks Part 2: Today's…

APC
2 min read

IP Freely

c2
Marbidin Muhammad

Show more