Pradeep KM
3122225001092

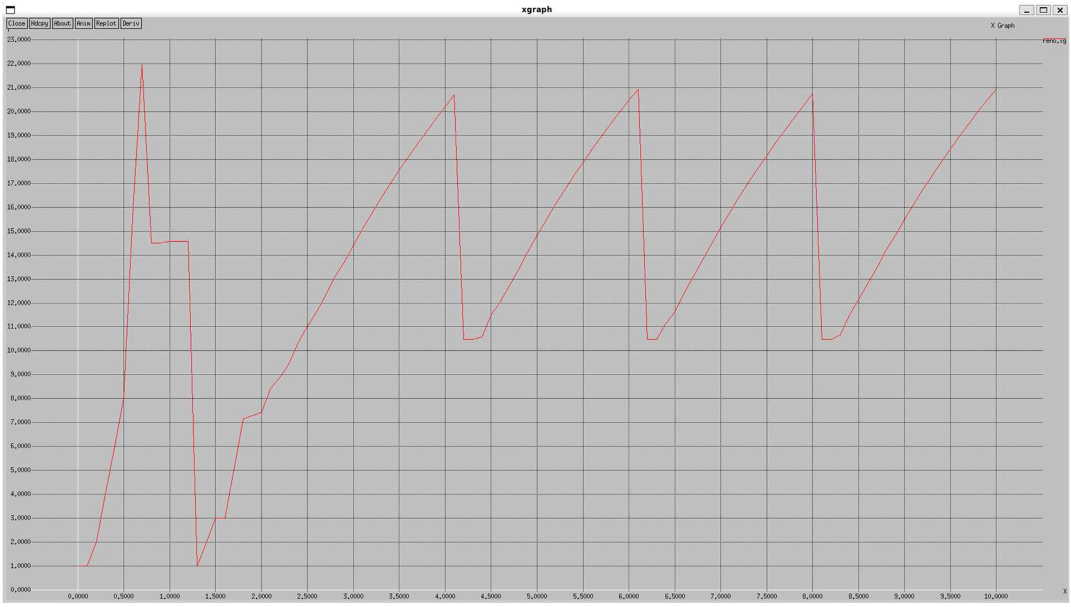
Exercise 12: TCP Congestion Control Algorithms

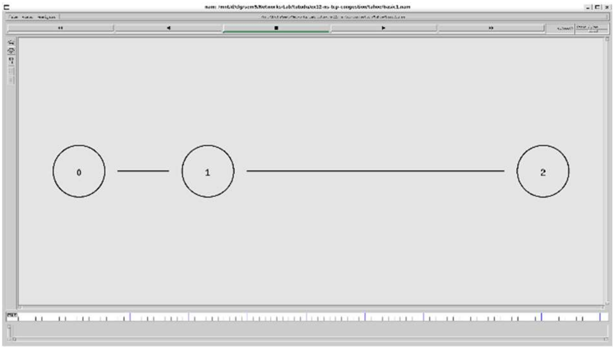
reno.tcl

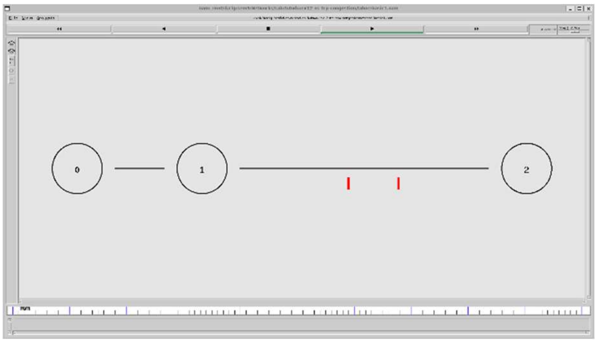
```
# Create a simulator object
set ns [new Simulator]

# Open the NAM file and the trace file
set nf [open basic1.nam w]
$ns namtrace-all $nf
set tf [open basic1.tr w]
$ns trace-all $tf

# Define a 'finish' procedure
proc finish {} {
   global ns nf tf
   $ns flush-trace
   close $nf
   close $tf
   exec nam basic1.nam &
   exec xgraph reno.xg &
   exit 0
}

# Create the network nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

# Create duplex links
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n2 800Kb 50ms DropTail

# Set queue limit for the router
$ns queue-limit $n1 $n2 7

# Visual hints for NAM
$ns color 0 Red
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n1 $n2 orient right
$ns duplex-link-op $n1 $n2 queuePos 0.5

# Create and configure TCP sending agent
set tcp [new Agent/TCP/Reno]
$tcp set class_ 0
$tcp set window_ 100
$tcp set packetSize_ 960
$ns attach-agent $n0 $tcp

# Create and attach TCP receive agent (sink)
```

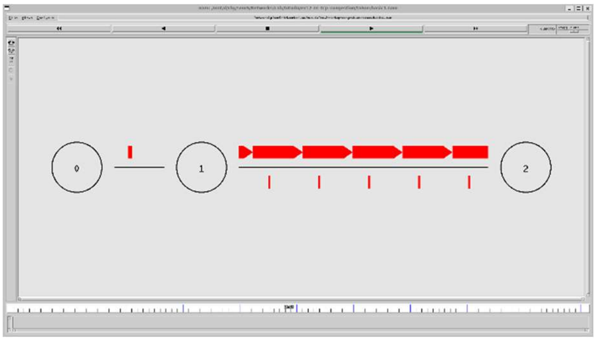
```
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
$ns connect $tcp $sink
# Schedule the data flow
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 0.0 "$ftp start"
$ns at 10.0 "finish"

# Procedure to plot the congestion window
proc plotWindow {tcpSource outfile} {
    global ns
    set now [$ns now]
    set cwnd [$tcpSource set cwnd_]
    puts $outfile "$now $cwnd"
    $ns at [expr $now + 0.1] "plotWindow $tcpSource $outfile"
}

# Open file to log congestion window
set outfile [open "reno.xg" w]
$ns at 0.0 "plotWindow $tcp $outfile"

# Run the simulation
$ns run
```

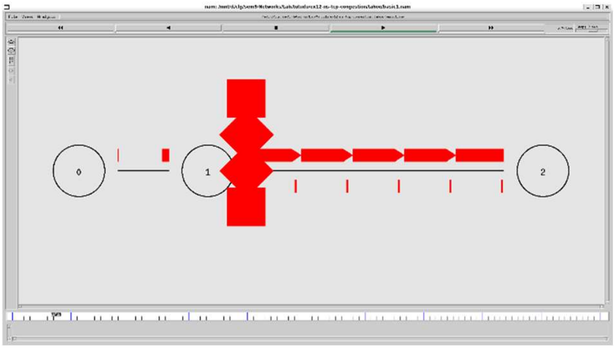
Output









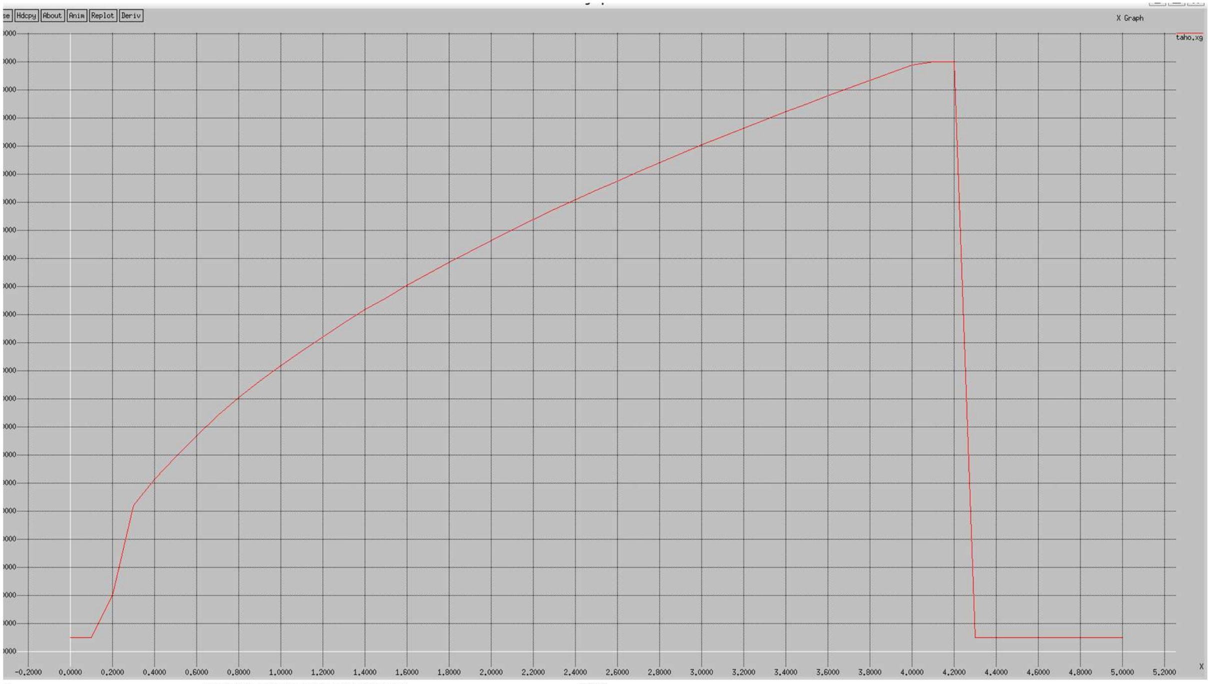


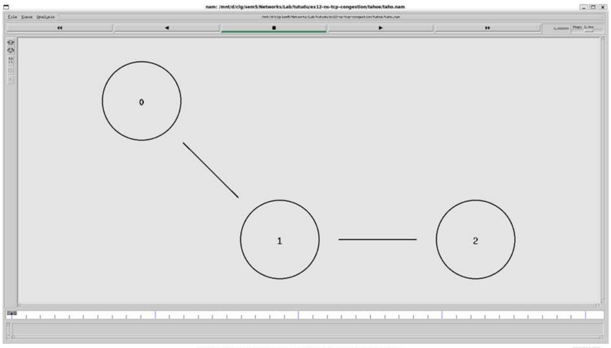
tahoe.tcl

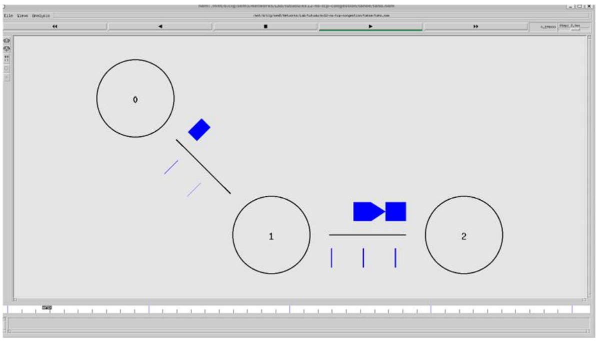
```
# Create a simulator object
set ns [new Simulator]

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

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

# Open the trace file for general simulation data
set tf [open taho.tr w]
$ns trace-all $tf

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

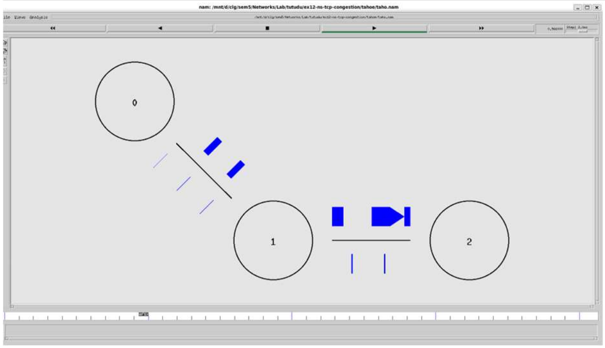
```
# Create three nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

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

# Set Queue Size of link (n0-n1) to 10 packets
$ns queue-limit $n0 $n1 10

# Position nodes for visualization in NAM
$ns duplex-link-op $n0 $n1 orient right-down
$ns duplex-link-op $n1 $n2 orient right

# Monitor the queue for link (n0-n1). (for NAM)
$ns duplex-link-op $n0 $n1 queuePos 0.5

# Setup a TCP connection using the default TCP agent
set tcp [new Agent/TCP] ;# Use default TCP, which should be Tahoe
$tcp set window_ 10 ;# Set the window size (e.g., 10 packets)
$tcp set packetSize_ 1000 ;# Set the packet size (e.g., 1000 bytes)
$tcp set timeout_ 1.0 ;# Set the timeout (e.g., 1.0 seconds)
$ns attach-agent $n0 $tcp

# Create a TCP Sink on the destination node
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

# Setup an FTP application over the TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp

# Schedule the FTP events
$ns at 0.1 "$ftp start"
$ns at 4.0 "$ftp stop"

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

# Procedure to plot the congestion window
proc plotWindow {tcpSource outfile} {
    global ns
    set now [$ns now]
    set cwnd [$tcpSource set cwnd_]
    # Record the data in a file
    puts $outfile "$now $cwnd"
    $ns at [expr $now + 0.1] "plotWindow $tcpSource $outfile"
}

# Prepare to record the congestion window
set outfile [open "taho.xg" w]
$ns at 0.0 "plotWindow $tcp $outfile"

# Run the simulation
$ns run
```

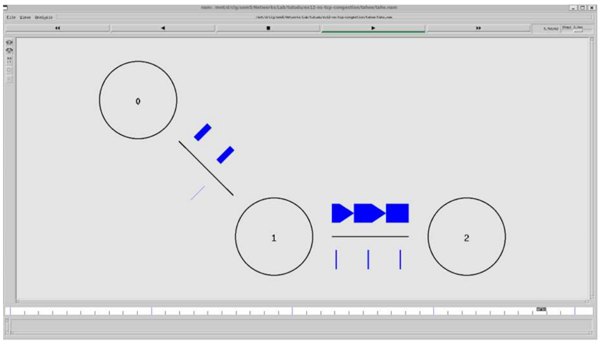
Output









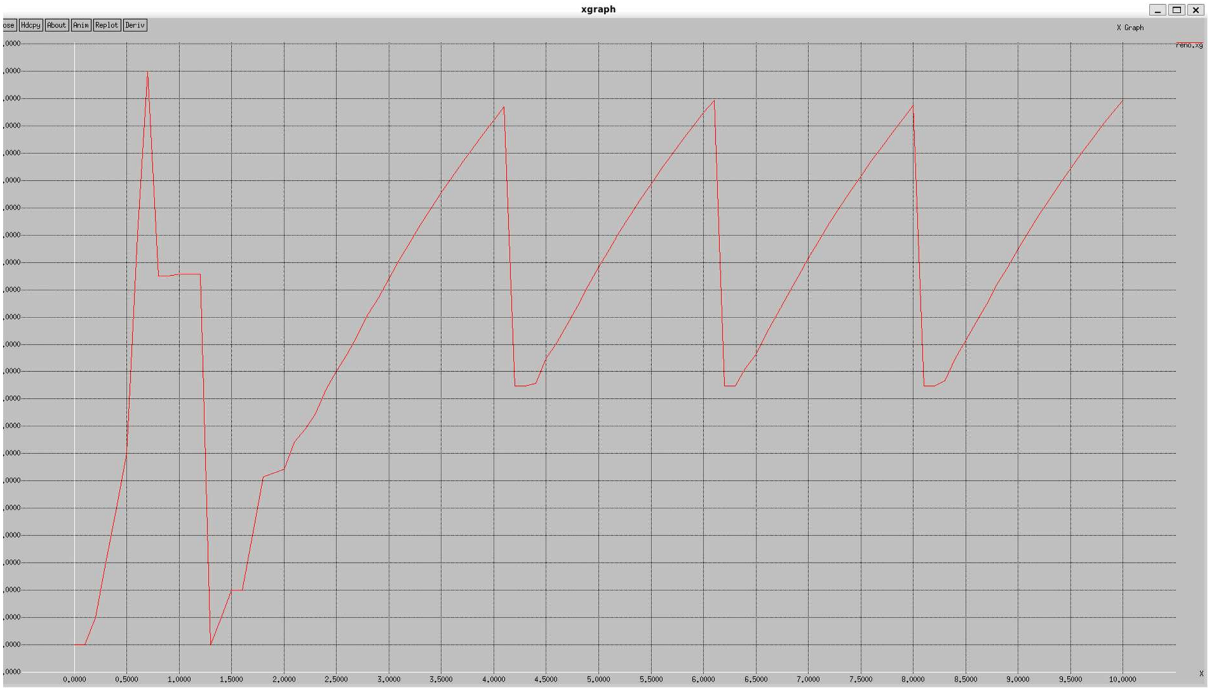


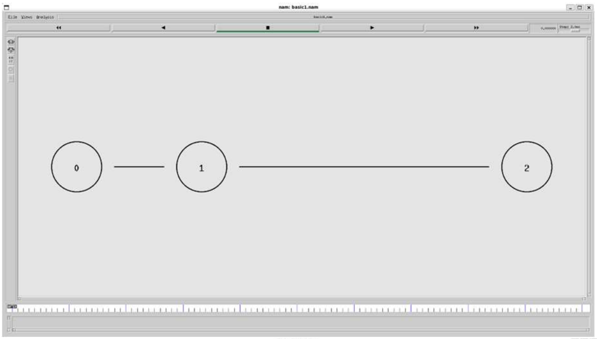
newreno.tcl

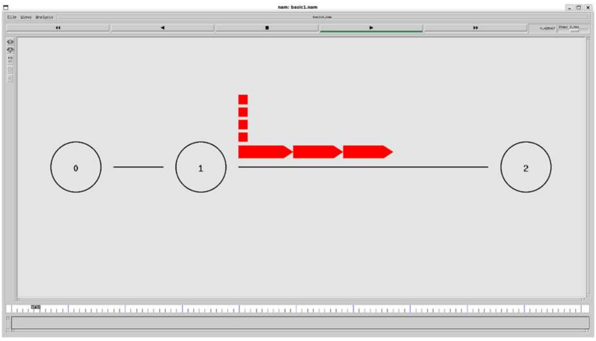
```
# Create a simulator object
set ns [new Simulator]

# Open the NAM file and the trace file
set nf [open basic1.nam w]
$ns namtrace-all $nf
set tf [open basic1.tr w]
$ns trace-all $tf

# Define a 'finish' procedure
proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam basic1.nam &
exec xgraph reno.xg &
exit 0
}

# Create the network nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

# Create duplex links
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n2 800Kb 50ms DropTail

# Set queue limit for the router
$ns queue-limit $n1 $n2 7

# Visual hints for NAM
$ns color 0 Red
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n1 $n2 orient right
$ns duplex-link-op $n1 $n2 queuePos 0.5

# Create and configure TCP sending agent
set tcp [new Agent/TCP/Reno]
$tcp set class_ 0
$tcp set window_ 100
$tcp set packetSize_ 960
$ns attach-agent $n0 $tcp

# Create and attach TCP receive agent (sink)
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
$ns connect $tcp $sink
```

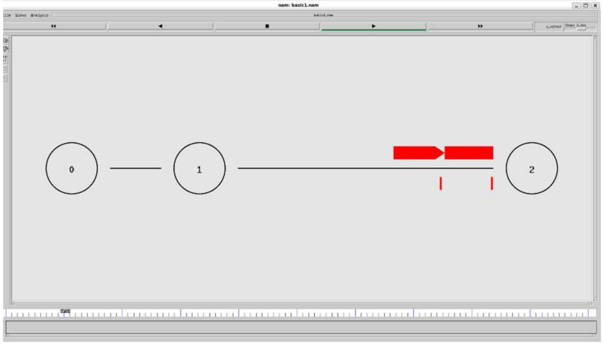
```
# Schedule the data flow
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 0.0 "$ftp start"
$ns at 10.0 "finish"

# Procedure to plot the congestion window
proc plotWindow {tcpSource outfile} {
    global ns
    set now [$ns now]
    set cwnd [$tcpSource set cwnd_]
    puts $outfile "$now $cwnd"
    $ns at [expr $now + 0.1] "plotWindow $tcpSource $outfile"
}

# Open file to log congestion window
set outfile [open "reno.xg" w]
$ns at 0.0 "plotWindow $tcp $outfile"

# Run the simulation
$ns run
```

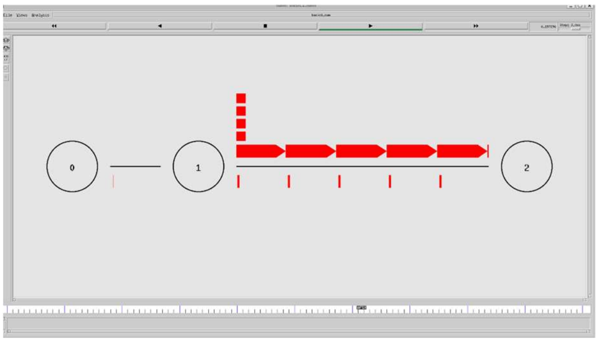
Output









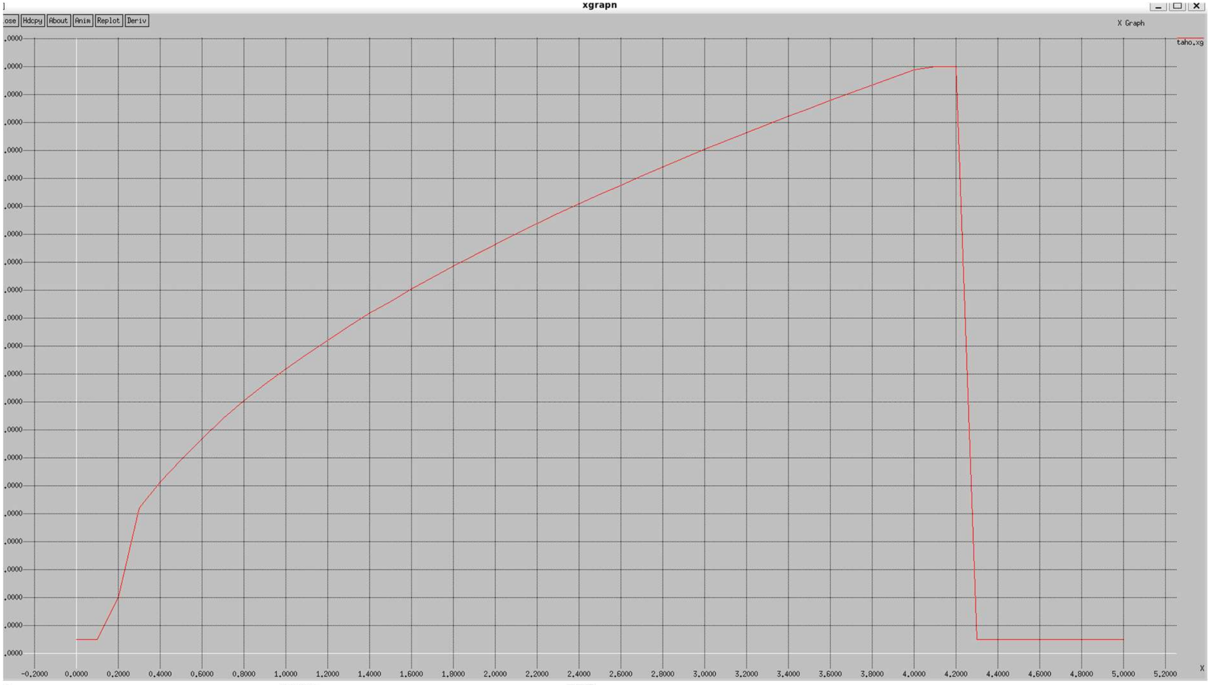


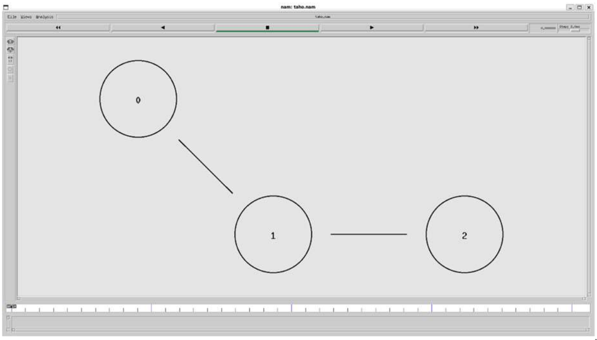
stack.tcl

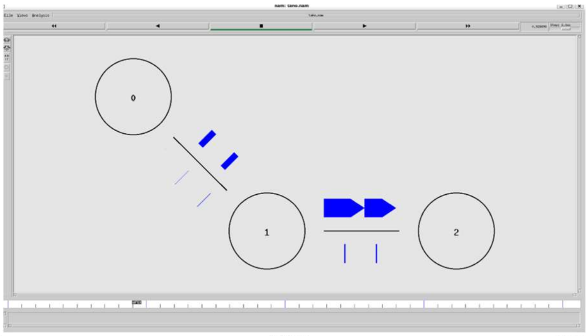
```
# Create a simulator object
set ns [new Simulator]

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

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

# Open the trace file for general simulation data
set tf [open taho.tr w]
$ns trace-all $tf

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

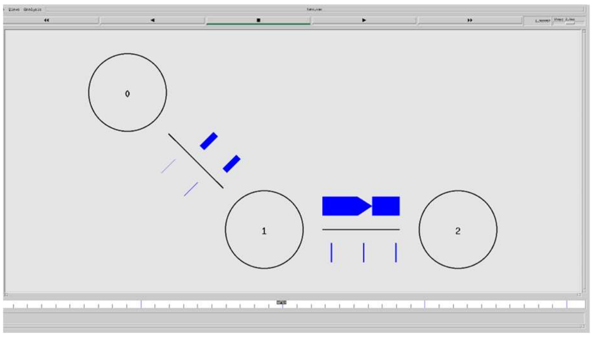
```
# Create three nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

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

# Set Queue Size of link (n0-n1) to 10 packets
$ns queue-limit $n0 $n1 10
# Position nodes for visualization in NAM
$ns duplex-link-op $n0 $n1 orient right-down
$ns duplex-link-op $n1 $n2 orient right

# Monitor the queue for link (n0-n1). (for NAM)
$ns duplex-link-op $n0 $n1 queuePos 0.5

# Setup a TCP connection using the default TCP agent
set tcp [new Agent/TCP]  ;# Use default TCP agent
$tcp set tcpType_ "Tahoe"  ;# Set the congestion control algorithm to Tahoe
$tcp set window_ 10  ;# Set the window size (e.g., 10 packets)
$tcp set packetSize_ 1000  ;# Set the packet size (e.g., 1000 bytes)
$tcp set timeout_ 1.0  ;# Set the timeout (e.g., 1.0 seconds)
$ns attach-agent $n0 $tcp

# Create a TCP Sink on the destination node
set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

# Setup an FTP application over the TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp

# Schedule the FTP events
$ns at 0.1 "$ftp start"
$ns at 4.0 "$ftp stop"

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

# Procedure to plot the congestion window
proc plotWindow {tcpSource outfile} {
   global ns
   set now [$ns now]
   set cwnd [$tcpSource set cwnd_]
   # Record the data in a file
   puts $outfile "$now $cwnd"
   $ns at [expr $now + 0.1] "plotWindow $tcpSource $outfile"
}

# Prepare to record the congestion window
set outfile [open "taho.xg" w]
$ns at 0.0 "plotWindow $tcp $outfile"

# Run the simulation
$ns run
```

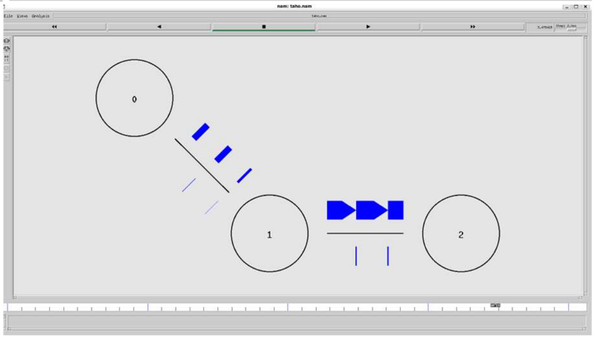
Output


xgraph
Close
Hdcpy
About
Anim
Replot
Deriv
X Graph
taho.xg
X


nam: taho.nam
0
1
2


nam: taho.nam
0
1
2




nam: taho.nam
0
1
2