Simple NS Tutorial

CS 556 – Advanced Computer Networks
@ Boston University

Instructor: Abraham Matta

1

Overview (1 of 2)

- NS is an event driven simulator
  - World is modeled as a list of events
  - Take one event, process it until it is completed
  - Uses simulated virtual time rather than real time
  - Single threaded

Overview (2 of 2)

- NS contains functionalities and support for:
  - Protocols: TCP, UDP, …
  - Traffic Behavior: FTP, Telnet, Web, CBR, VBR
  - Queue Mgmt: Drop Tail, RED, CBQ
  - Routing Algorithms: LS, DV, Dijkstra's Algorithm
  - Multicast, MAC protocols
  - Animation
  - Tracing

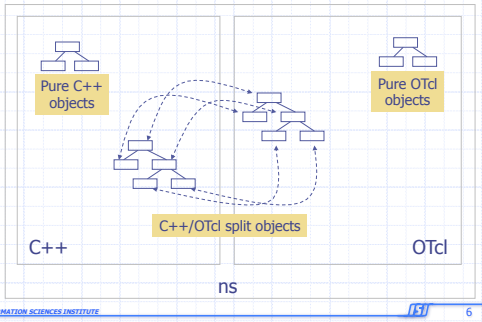
NS Goals

- Support networking research & education
  - Protocol design, traffic studies
  - Protocol comparison
- Collaborative environment
  - Free
- Multiple levels of detail in one simulator

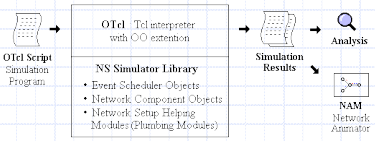
NS Current Status

- Platform Support
  - FreeBSD, Linux, Solaris, Windows & MAC
- Latest Release
  - ns-2.3X
  - ns-3

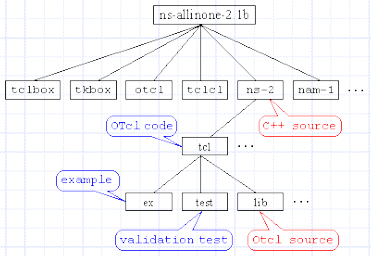
OTcl and C++: The Duality



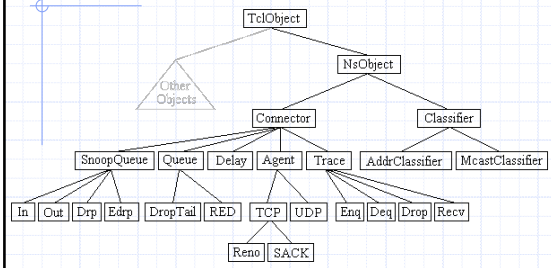
NS input & output



NS Directory map



NS Class Hierarchy



Basic Tcl

Defining a variable.

```
set var1 1
set var2 “Hello“
```

Variable Reference

```
puts "var1=$var1, var2=$var2"
```

Assign the results of a function

```
set var3 [expr 5*10]
```


Basic Tcl

For Loop

```
for {set i 0} {$i < 100} {incr i} {
    puts “I = $i "
}
```

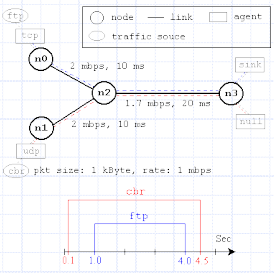
While Loop

```
set i 0
while {$i < 10} {
    set n($i) [new Node]
    incr i
}
```

Procedure

```
proc proc1 {}  {
    puts "in procedure proc1"
}
proc1
```


Simple Simulation Example



Creating Event Scheduler

- Create event scheduler
  - set ns [new Simulator]
- Schedule events
  - $ns at <time> <event>
  - <event>: any legitimate ns/tcl commands
- Start scheduler
  - $ns run

Creating Network

- Nodes
  - set n0 [$ns node]
  - set n1 [$ns node]
- Links and queuing
  - $ns duplex-link $n0 $n1 <bandwidth> <delay> <queue_type>
  - <queue_type>: DropTail, RED, CBQ, FQ, SFQ, DRR

Creating Connection: UDP

- UDP
  - set udp [new Agent/UDP]
  - set null [new Agent/Null]
  - $ns attach-agent $n0 $udp
  - $ns attach-agent $n1 $null
  - $ns connect $udp $null

Creating Traffic: On Top of UDP

- CBR
  - set src [new Application/Traffic/CBR]
- Exponential or Pareto on-off
  - set src [new Application/Traffic/Exponential]
  - set src [new Application/Traffic/Pareto]

Creating Connection: TCP

- TCP
  - set tcp [new Agent/TCP]
  - set tcpsink [new Agent/TCPSink]
  - $ns attach-agent $n0 $tcp
  - $ns attach-agent $n1 $tcpsink
  - $ns connect $tcp $tcpsink

Creating Traffic: On Top of TCP

- FTP
  - set ftp [new Application/FTP]
  - $ftp attach-agent $tcp
- Telnet
  - set telnet [new Application/Telnet]
  - $telnet attach-agent $tcp

Tracing

- Trace packets on all links
  - $ns trace-all [open out.tr w]

event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id

r : receive (at to_node)
+ : enqueue (at queue)
- : dequeue (at queue)
d : drop (at queue)

src_addr : node.port (3.0)
dst_addr : node.port (0.0)

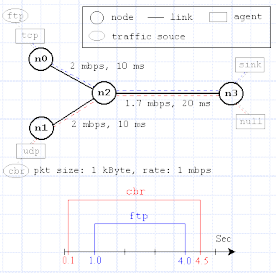
```
r 1.3556 3 2 ack 40 ------- 1 3.0 0.0 15 201
+ 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
r 1.35576 0 2 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
d 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
- 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
```

- + 0.112731 1 3 tcp 512 ------- 0 0.1 3.1 0 0
- - 0.112731 1 3 tcp 512 ------- 0 0.1 3.1 0 0
- r 0.125461 1 3 tcp 512 ------- 0 0.1 3.1 0 0

Summary: Generic Script Structure

```
set ns [new Simulator]
# [Turn on tracing]
# Create topology
# Setup packet loss, link dynamics
# Create:
#    - protocol agents
#    - application and/or setup traffic sources
# Post-processing procs
# Start simulation
```


Simple Simulation Example 1



TCL Script Step 1

```
#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 out.nam w]
$ns namtrace-all $nf

#Open the All trace file
Set f [open out.tr w]
$ns trace-all $f
```


TCL Script Step 2

```
#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 $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail
```


TCL Script Step 3

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

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$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
```

TCL Script Step 4

```
#Setup a TCP connection
set tcp [new Agent/TCP]
$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
```


TCL Script Step 5

```
#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 packet_size_ 1000
$cbr set rate_ 1mb
```


TCL Script Step 6

```
#Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$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_]"
```

TCL Script Step 7

```
#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
}
```


TCL Script Step 8

```
# Trace Congestion Window and RTT
set file [open cwnd_rtt.tr w]
$tcp attach $file
$tcp trace cwnd_
$tcp trace rtt_

#Run the simulation
$ns run
```


Visualization Tools

- nam (Network AniMator)
  - Packet-level animation
  - Well supported by ns
- gnuplot
- xgraph
- Conversion from ns trace to gnuplot or xgraph format

NAM Interface: Color

- Color mapping

```
$ns color 40 red
$ns color 41 blue
$ns color 42 chocolate
```

- Color ↔ flow id association

```
$tcp0 set fid_ 40 ;# red packets
$tcp1 set fid_ 41 ;# blue packets
```


Awk Script (Throughput)

- Do “man awk” on csa2 to know more
- awk ‘{if(($3==“0”)&&($4==“2”)&&($1==“r”)) print $2, $6}’ out.tr > Temp
- awk -f throughput.awk Temp > File_Throughput
- xgraph File_Throughput
- gnuplot
  - plot “File_Throughput” with lines

throughput.awk script file

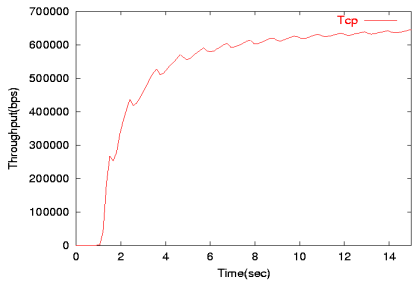
```
BEGIN{
        sum = 0;
}
{
    if ($1>0)
      printf(“%lg\t%lg”,$1,sum/$1);
    sum = sum + $2;
}
END{
   puts “Done”
}
```

GNUPLOT

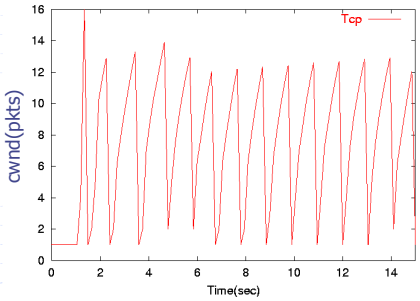
```
#simple.gnu
set xrange [0:200]
set yrange [0:0.4]
set xlabel 'Time (sec)'
set ylabel 'Throughput (bps)'
plot ' out.data' title ' Tcp' with lines
set term postscript
set output "tcp.ps"

%gnuplot simple.gnu
```


Throughput vs Time



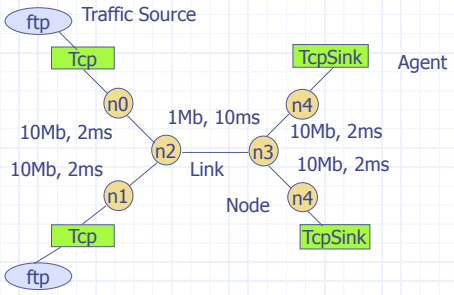
Congestion Window vs Time



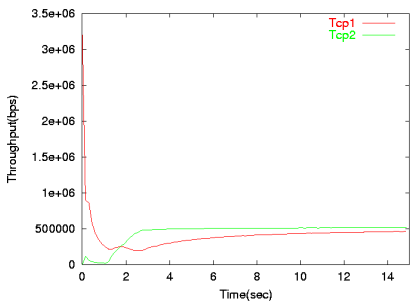
Awk script (Congestion Wnd)

- awk '{if($6==“cwnd_”) print $1,$7}' cwnd_rtt.tr > cwnd.data
- gnuplot
  - plot “cwnd.data” with lines

Simple Simulation Example 2



Throughputs vs Time



Useful Links

- NS web page (Download & Build NS)
  - http://nsnam.isi.edu/nsnam/index.php/Main_Page
- NAM web page
  - http://www.isi.edu/nsnam/nam
- NS tutorials (Simplest tutorial first)
  - http://nile.wpi.edu/NS/
  - http://www.isi.edu/nsnam/ns/tutorial/index.html
- NS Manual (Complete Reference)
  - http://www.isi.edu/nsnam/ns/ns-documentation.html