Introduction to ns-2

Zhiqiang Ma

Why simulating a network?

- Suppose we are designing a 10Gbps network for HKUST
  - What kinds of routers should we choose?
  - What kinds of topology should we use?
  - What is the aggregate speed at one switch/router?
- Build a prototype?
  - Network devices are usually expensive, and hard to install and configure, especially for large network configurations
- Why not make use of the power of modern computer to simulate the network?

What is ns-2?

- Ns-2 is a discrete event simulator targeted at networking research. Ns-2 provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless networks
- Ns-2 is written in C++ and OTcl, an object oriented version of Tcl
- Nam is a Tcl/TK based animation tool for viewing network simulation traces and real world packet traces. It is mainly intended as a companion animator to the NS simulator

Installation of ns-2

Please refer to:

http://fclose.com/b/linux/3376/install-ns-2-and-ns-3-on-fedora-linux/

Simulate a simple topology

- You can write your Tcl scripts in any text editor
- As an example, we call the first Tcl script ‘example1.tcl’
- Create a simulator object by adding this line to the script

```
set ns [new Simulator]
```

Simulate a simple topology

- Open a file for writing the trace data. The file is going to be used by nam
  - First, open the file out.nam for writing and gives it the file handle “nf”
  - Then, tell the simulator object that we create out.nam to write all simulation data that is going to be relevant with nam into this file

```
set nf [open out.nam w]
$ns namtrace-all $nf
```

Simulate a simple topology

- Add a finish procedure that closes the trace file and starts nam

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

Simulate a simple topology

- Tell the simulator object to execute the finish procedure after 5.0 seconds of simulation time

```
$ns at 5.0 “finish”
```

- Start the simulation

```
$ns run
```

Your script should look like this

zma@localhost:~/working/ns-2/comp362ta

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

#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 trace file
        close $nf
    #Execute nam on the trace file
        exec nam out.nam &
        exit 0
}

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

#Run the simulation
$ns run
```

0-1 100%

Simulate a simple topology

- Save the file and try to run it in the shell

```
ns example1.tcl
```

- You are going to see that nam starts automatically with an empty window
  - This is because you have not defined any topology or event

Simulate a simple topology

- Define a simple topology with two nodes that are connected by a link
- Define two nodes: insert the following lines into example1.tcl before the line “$ns at 5.0 finish”

```
set n0 [$ns node]
set n1 [$ns node]
```

- Command ‘$ns node’ creates a new node. We create two nodes and assign them to the handles ‘n0’ and ‘n1’ by these two commands

Simulate a simple topology

- Tell the simulator object to connect the nodes n0 and n1 with a duplex link with the bandwidth of 1 Megabit, a delay of 10ms and a DropTail queue

```
$ns duplex-link  $n0 $n1 1Mb 10ms DropTail
```

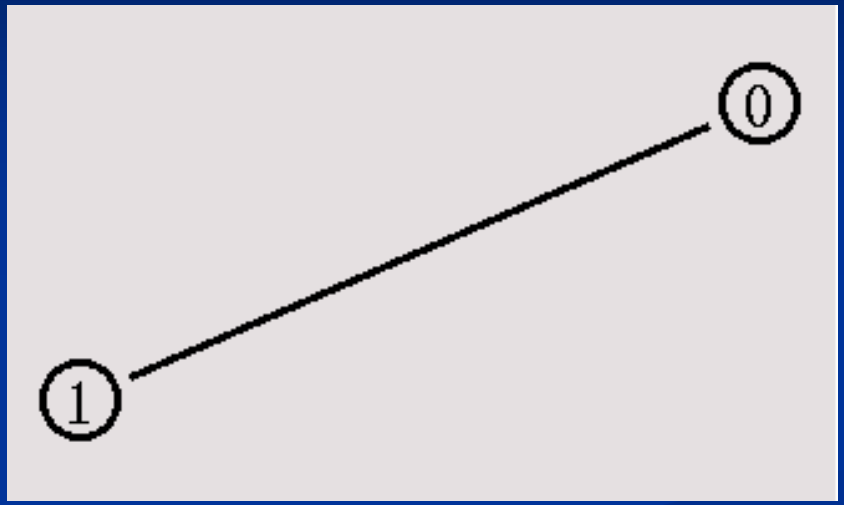
Simulate a simple topology

- Save the file now and try to run it in the shell

```
ns example1.tcl
```

- You are going to see that nam starts automatically and generates an output

Simulate a simple topology



- A simple topology is set successfully
- We will show you how to define events so that the nodes can send messages

Send data between nodes

In ns-2, data is always being sent from one “agent” to another

The idea

- Create an agent object that sends data from node n0
- Create another agent object that receives data on node n1

Send data between nodes

- Put more lines of script just before the line “$ns at 5.0 finish”

- First, we create a UDP agent and attach it to node n0

```
# Create an UDP agent and attach it to node n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
```

Send data between nodes

- What does n0 send? We use a traffic generator
  - Attach a CBR (constant bit rate) traffic generator to the UDP agent

```
# Create a CBR traffic source
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
# Attach the traffic generator to udp0
$cbr0 attach-agent $udp0
```

- We have created an agent object that sends packet with size of 500 bytes through UDP from node n0 every 0.005 simulation second

Send data between nodes

- Create another agent object that receives data on node n1
  - The Null agent is the traffic sink in ns-2
  - We create a Null agent and attach it to n1

```
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
```

Send data between nodes

- Connect these two agents with each other

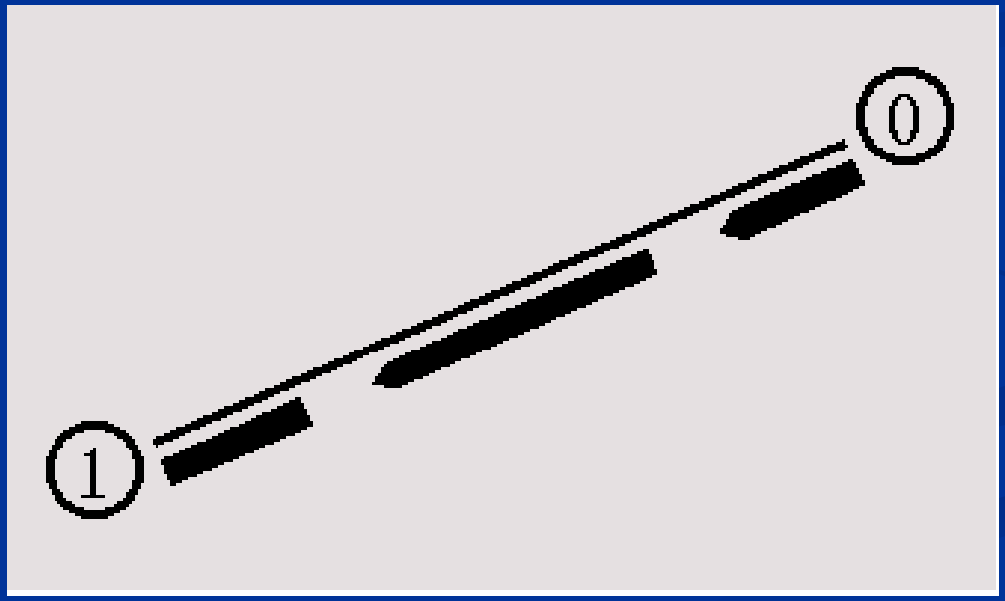
```
$ns connect $udp0 $null0
```

- Tell the CBR agent to send data at 0.5 second and stop at 4.5 second

```
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
```

Send data between nodes

- Save the file and start the simulation again
- Click on the “play” button in the nam window, you will see that after 0.5 simulation second, node n0 starts sending packets to node n1
- You can slow down nam with the “Step” slider



Thanks you!

Zhiqiang Ma