Simulation with NS-2 and CPN tools

Ying-Dar Lin
Department of Computer Science,
National Chiao Tung University
Outline

- NS-2 simulator
  - NS-2 basics
  - Basic syntax
  - Tracing a simple network
  - Mini and term projects

- CPN tools
  - Terminology and operation of CPN tools
  - Interface of CPN tools
  - Mini and term projects
A brief introduction to NS-2

- An event simulator for network research
- Support for simulation of TCP, routing, multicast, etc.
- Have a C++ (simulator) and Object Tcl (OTcl) interpreter

“NS by Example”, Jae Chung and Mark Claypool

User view of NS-2
Class hierarchy in NS-2

- C++ implementation
  - for efficiency in simulation
  - detailed definition and operation of protocols
- OTcl script
  - description of topology, protocols and applications
  - Specification of the output form
Initialization and termination (1/2)

☐ Initialization
  - set ns [new Simulator]

☐ Opening trace and visualization files
  - set tracefile1 [open out.tr w]
  - $ns trace-all $tracefile1
  - set namfile [open out.nam w]
  - $ns namtrace-all $namfile
Initialization and termination (2/2)

- **Termination**
  - proc finish {}
    ```
    proc finish {} {
        global ns tracefile1 namfile
        $ns flushtrace
        close $tracefile1
        close $namfile
        exec nam out.nam &
        exit 0
    }
    ```
    - declaration outside the procedure
    - dump the trace
    - execute `nam` for visualization

- **Start simulation**
  - $ns run
  - schedule "finish" at time 125 sec
Definition of a simple network

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

$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns simplex-link $n1 $n2 0.3Mb 100ms DropTail
$ns simplex-link $n2 $n1 0.5Mb 50ms DropTail
$ns duplex-link $n2 $n3 2Mb 5ms DropTail

$ns queue-limit $n1 $n2 10 assign queue size (default: 50)
Simulation of FTP over TCP

source

$n0$

1 Mbps

10 ms

$n1$

300 kbps

100 ms

$n2$

500 kbps

50 ms

$n3$
sink

2 Mbps

5 ms

set tcp [new Agent/TCP]

$ns\ attach-agent\ $n0\ $tcp$

define source

set sink [new Agent/TCPSink]

$ns\ attach-agent\ $n4\ $sink$

define sink

$ns\ connect\ $tcp\ $sink$

$tcp\ set\ fid\_1$

set flow id to 1

$tcp\ set\ packetSize\_552$

default:1000

set ftp [new Application/FTP]

$ftp\ attach-agent\ $tcp$

$ns\ at\ 1.0\ "ftp\ start"

$ns\ at\ 124.0\ "ftp\ stop"
Visualization using nam

- Give node position
  \$ns duplex-link-op $n0 $n2 orient right-down

- Set color
  \$ns color 1 Blue

- Color nodes
  $n0 color red

- Color links
  \$ns duplex-link-op $n0 $n2 color “green”

- Add and remove marks
  $ns at 2.0 “$n2 add-mark m3 blue box”
  $ns at 30.0 “$n2 delete-mark m3”

- Add label
  $ns at 1.2 “$n2 label \"active node\"”
  \$ns duplex-link-op $n1 label “TCP input link”

- Add text
  $ns at 5 “$ns trace-annotate \"packet drop\"”

- Monitor queue size
  $ns simplex-link-op $n1 $n2 queuePos 0.5
Tracing

EnqT → Queue → DeqT → Delay → TTL → RecvT

Drop → DrpT

Agent/Null

simplex link

components for simulation

components for tracing

<table>
<thead>
<tr>
<th>Event</th>
<th>Time</th>
<th>From node</th>
<th>To node</th>
<th>Pkt type</th>
<th>Pkt size</th>
<th>Flags</th>
<th>Fid</th>
<th>Src addr</th>
<th>Dst addr</th>
<th>Seq num</th>
<th>Pkt id</th>
</tr>
</thead>
</table>

Fields in the trace
Mini-project: TCP simulation

1. Download and install the NS-2 simulator at http://www.isi.edu/nsnam/ns/ns-build.html
   - We suggest to download the all-in-one version to save your time

2. Refer to the document of “NS simulator for beginners”. Prepare the TCL script in Table 2.4 “ex1.tcl”.

3. Set up a loss module between $n2$ and $n3$ (See the code in page 42-43)

4. Set up the parameters as indicated in the questions of the project

(Note: the configuration in each question of the mini-project and term project is tried independently, meaning that you will restore the original configuration after each question and set the configuration as requested in the next.)
Mini-project: questions

1. Tune the loss rate between n2 and n3 to 0%, 10% and 20%. See the changes in TCP throughput.

2. Increase the UDP traffic (n1 to n5) from 0.01Mb/s to 0.1Mb/s. How is the TCP traffic affected in terms of window size (or throughput)?

3. Add one more link n6→n2. Set three nodes (n0, n1 and n6) to be TCP sources. Compare the window sizes when the queue mechanism is DropTail and RED.

4. Explain your observation in the above questions according to the flow control and congestion control of TCP.

ps. You may set the parameters at will if they are unspecified in the questions.
Term project: random early discard

- Please refer to these references when doing this term project.

- Build the simulation model of “Drop tail buffer” and “RED buffer” as listed in Table 6.1 and Table 6.2.

Simulation model for this project
Term project: questions

1. Repeat the experiments as indicated in Fig. 6.3 and Fig. 6.6. Make sure your models coincide with their results.

2. Add one more UDP link to n2 as described in Table 2.4 “ex1.tcl”. How does the UDP traffic affect the window size in DropTail and RED?

3. Suppose one of the links from S(i) to n2 is upgraded to 100Mb/s. How will the other links to n2 (10 Mb/s) affected (in terms of window size)?

4. How about one source S[i] with RED rather than DropTail (Change from DropTail to RED in one of the duplex-links from s(i) to n2)?

5. Design your reconfiguration of the model. Tell why you have the reconfiguration and what you want to watch.
Important references

- Homepage of NS-2 simulator
  - http://www.isi.edu/nsnam/ns/

- NS simulator for beginners

- Tutorial for the network simulator “ns”
What is Colored Petri Nets (CPN) tools?

- A graphical modeling language to build an executable model.
- For design, specification, simulation and verification systems.
- Typical target applications (distributed systems with communication and synchronization):
  - Communication protocols
  - Data networks
  - Distributed algorithms
  - Embedded systems
  - Workflow modeling
  - Agent systems
Terminology in CPN modeling

- **Transition/event**
- **Place/state**
- **Inscription**
- **Initial marking**
- **Marking (tokens/token values (colors))**
- **Arc**

*Diagram with CPN modeling elements labeled.*
Transitions

Packets to Send

SendPacket

NextSend

Transition enabled

1 \((1, \text"COL")\)++
2 \((2, \text"OUR")\)++
...

n is bound to 1

 initial marking

The token \((1, \text"COL")\)
also goes to place A
(i.e., the network)

n is bound to 1

ps. the marking in both
places are unchanged due
to the double-headed arcs

n is bound to 1

initial marking

i.e., retransmission is possible

d is bound to \text"COL"

1\ '1

1\ '1
Basic syntax in CPN tools

- **Data Type (color set)**
  - colset DATA = string;
  - colset NOxDATA = product NO * DATA;

- **Multi-set**
  - Six packets:
    1` (1,"COL") ++ 1` (2,"OUR") ++ 1` (3,"ED") ++ 1` (4,"PET") ++ 1` (5,"RI") ++ 1` (6,"NET")
  - ‘: number of appearances of some element
  - +++: Union of two multi-sets
Interface of CPN tools

Binder
(A set of sheets in the tabs)
Installation of CPN tools

- Download the CPN tools from the Web site: http://www.daimi.au.dk/~cpntools/bin/license/cpntools_setup.php
  - username: pclin_cis_nctu_edu_tw
  - password: JSG8304
- Install the tool in the Windows (note: your computer must support OpenGL to run CPN tools.
- Read “Getting started with CPN tools” after installation
  http://wiki.daimi.au.dk/cpntools-help/getting_started_with_cpn_.wiki
Mini project: simple protocol

1. Launch CPN tools
2. Load the net from \textless\text{installation path}\textgreater\>/Samples/SimpleProtocol/SimpleProtocol.cpn
3. Drag the “simulation palette” from Toolbox $\rightarrow$ Simulation to the workspace. (See the “Tool box” section from “Getting started with CPN tools”)
4. Press the button of “Executes a transition”, move the mouse cursor to the binder of the Net, and Click on it.
5. Click again and again. See how the tokens are moved in the net.
Mini project: questions

1. What token will Place A receive after the first transition? Explain why.

2. How does the network guarantee the packets are transmitted in order?

3. Modify the model of simple protocol so that the sender can send two packets (tokens) before receiving a ACK, but the receiver responds with a ACK for every packet.

4. (optional/bonus) Design a model to simulate UDP transmission without a ACK.
Term project: dining philosophers

- Load and execute the CPN of dining philosophers under `<installation path>/Samples/DiningPhilosophers/DiningPhilosophers_COMM_Mon.cpn`

- Check whether a deadlock will occur in this model. If a deadlock occurs, please explain how.
Important references

- Homepage of CPN tools
  - http://wiki.daimi.au.dk/cpntools/cpntools.wiki
  - http://www.informatik.uni-hamburg.de/TGI/PetriNets/
  - http://www.daimi.au.dk/designCPN/