Lab 3: NS-2 Simulation

Wei-Liang Shen
w.w.l.shen@gmail.com

Tung-Wei Kuo
tungweikuo@gmail.com

Goal

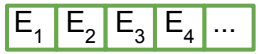
- Use NS-2 to implement a bit-rate selection protocol called SampleRate
  - SampleRate
    - http://pdos.csail.mit.edu/papers/jbicket-ms.pdf
  - NS-2(networking simulator)
    - http://nsnam.isi.edu/nsnam/index.php/Main_Page

Simulator

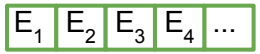
- Real-System might not be available or costly
- Evaluate complex functions
- Quick evaluations
- …

NS-2: a discrete event-driven networking simulator

Event-driven Simulator

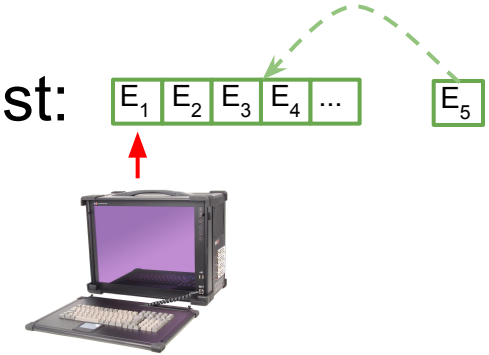
- Event List: 

Event-driven Simulator

- Event List: 

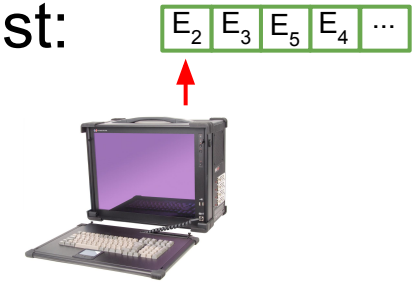
Event-driven Simulator

- Event List:



Event-driven Simulator

- Event List:

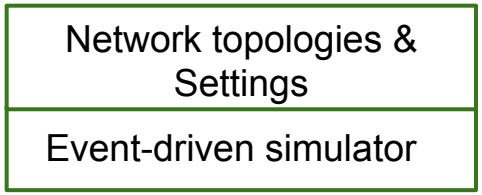


- Initial event list
- while(t < T)
  - Get nearest event
  - t = event time
  - process event
  - add/delete event
  - update

NS-2

- An open-source event-driven simulator
- Simulate the behavior of networks and investigate the performance of network protocols

NS-2



- TCL: build a network systems
- C++: implement network protocols as classes

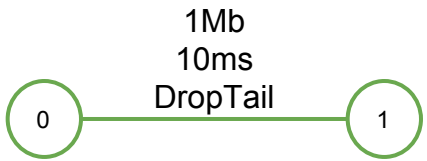
Create Nodes

```
$ns_ node-config -adhocRouting $val(rp) \
                 -llType $val(ll) \
                 -macType $val(mac) \
                 -ifqType $val(ifq) \
                 -ifqLen $val(ifqlen) \
                 -antType $val(ant) \
                 -propType $val(prop) \
                 -phyType $val(netif) \
                 -channelType $val(chan) \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace ON \
                 -macTrace OFF \
                 -movementTrace OFF
```

```
set val(chan) Channel/WirelessChannel;      # channel type
set val(prop)  Propagation/TwoRayGround;  # radio-propagation model
set val(netif)  Phy/WirelessPhy;                # network interface type
set val(mac)  Mac/802_11;                       # MAC type
set val(ifq)     Queue/DropTail/PriQueue;     # interface queue type
set val(ll)       LL;                                  # link layer type
set val(ant)     Antenna/OmniAntenna;        # antenna model
set val(ifqlen) 50;                    # max packet in ifq
set val(nn)      2;                    # number of mobile nodes
set val(rp)       AODV;                # routing protocol
set val(x)       2000;                 # X dimension of the topography
set val(y)       2000                  # Y dimension of the topography
set val(sc)      "Trace/Motion0";      # Apply different motion file
```

Create Links

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
proc finish {} {
        global ns nf
        $ns flush-trace
        close $nf
        exec nam out.nam &
        exit 0
}
set n0 [$ns node]
set n1 [$ns node]
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
...
$ns at 5.0 "finish"
$ns run
```



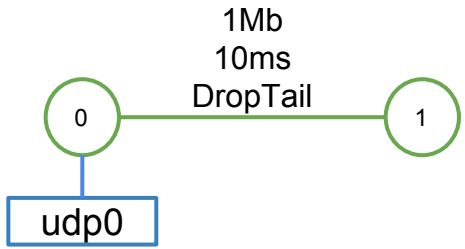
Attach Traffic

Transport layer

```
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0

set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
$ns connect $udp0 $null0

$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```



Attach Traffic

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

Transport layer

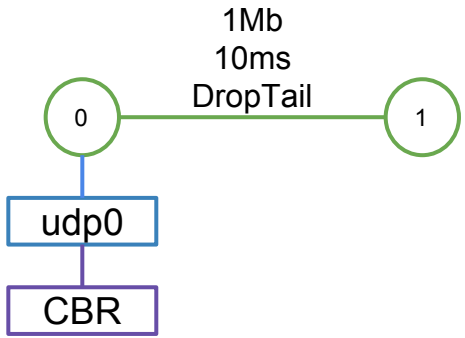
```
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
```

Application layer

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

```
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
$ns connect $udp0 $null0

$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```



Attach Traffic

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

Transport layer

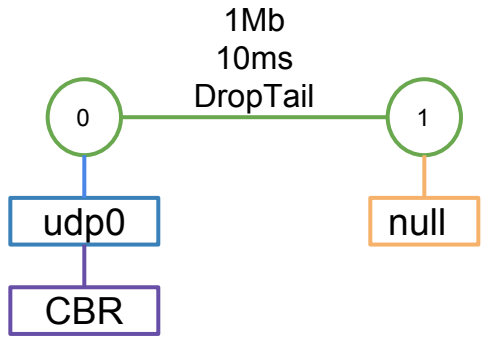
```
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
```

Application layer

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

```
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
$ns connect $udp0 $null0
```

```
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```



Install and Build NS-2

Install and Build NS-2

- OS:
  - Linux: Ubuntu or Debian
  - Windows: Need to install Cygwin

- Recommend to install NS-2 in CSIE workstations

Step 1

- Download NS-2 project
  - http://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.35/ns-allinone-2.35.tar.gz/download

- Extract the compressed file
  - *tar -zxvf ns-allinone-2.35.tar.gz*

Step 2

- Install NS-2
  - *cd ns-allinone-2.35*
  - *./install*
    - Depending on your operating system, you might see some error messages when you install NS-2. Google the solutions!
    - Possible error:

      ls.h:137:20: note: declarations in dependent base ‘std::map<int, LsIdSeq, std:: less<int>, std::allocator<std::pair<const int, LsIdSeq> > >’ are not found by unqualified ...
      - edit ns-2.35/linkstate/ls.h:137
        - change **void eraseAll() { erase…** to **void eraseAll() { this->erase…**

Step 3

- Set environment variables
  - Edit ~/.bashrc and add the following scripts

```
# LD_LIBRARY_PATH
OTCL_LIB=~/ns-allinone-2.35/otcl-1.14
NS2_LIB=~/ns-allinone-2.35/lib
USR_LOCAL_LIB=/usr/local/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:$USR_LOCAL_LIB
# TCL_LIBRARY
TCL_LIB=~/ns-allinone-2.35/tcl8.5.10/library
USR_LIB=/usr/lib
export TCL_LIBRARY=$TCL_LIB:$USR_LIB
# PATH
XGRAPH=~/ns-allinone-2.35/bin:~/ns-allinone-2.35/tcl8.5.10/unix:~/ns-allinone-2.35/tk8.5.10/unix
NS=~/ns-allinone-2.35/ns-2.35/
NAM=~/ns-allinone-2.35/nam-1.15/
export PATH=$PATH:$XGRAPH:$NS:$NAM
```

  - Log out, and log in again.

Step 4

- Test the installation
  - *ns ns-simple.tcl*
    - folder: simple_example/
    - idle output:
      CBR packet size = 1000
      CBR interval = 0.0080000000000000002
    - GUI: Xorg

Lab 3

Implement packet loss & bit-rate selection in .cc files

1. Generate Packet Losses in NS2

- Modify ns-2.35/mac/mac-80211.cc
- Compute the distance between the transmitter and receiver
- Use the path-loss model (Friis) to compute the SNR based on the distance
  - $P_r = (P_t * G_t * G_r * (\lambda/(4\pi R))^2)/L$, (Watts)
    - Tx power(P_t): 18dBm
    - Tx Gain(G_t): 2dB
    - Rx Gain(G_r): 2dB
    - Wavelength(λ): c/2.44GHz
    - Distance(R): m
    - System loss: 17 dB
  - Noise: -95 dBm

1. Generate Packet Losses in NS2

- Lookup the SNR-Rate-PDR table to check the packet loss rate of a given rate (data/sr2p.h)
  - SNR Modulation PDR

- Randomly drop the packet based on the PDR
  - For example, if PDR = 0.9, generate a random number *x* and drop the packet if *x* is larger than 0.9.
  - Implement PDR in mac/mac-80211.cc

- We assume the sender always can receive the ACKs from receiver.

Overview of SampleRate

We only implement a simplified version of SampleRate

2. Bit-Rate Selection

- Initialization:
  - Transmitting at the highest bit-rate
- After Initialization:
  - Do random sampling for every 10th packet
    - Not eligible to be sampled:
      - Average transmission time is larger than average transmission time of the current bit-rate

2. Bit-Rate Selection

- Detail descriptions are illustrated in “Sample Rate Spec”

- Do not select the transmission bit-rate based on the receiving SNR!!

Mobility Patterns

- Simulate SampleRate for a one-link topology
- The transmitter is static, but the receiver moves based on the random-walk mobility model
- Mobility patterns are defined in "tcl" folder
  - tcl/public_[0~3].tcl
- Public testcase will be announced on 2014/04/22

I/O

- Input
  - 3 public and 2 private tcl files, these files are included in “tcl” folder
    - configure default parameters
    - create nodes and links
    - attach traffic
    - log trace
    - specify mobility pattern(tcl/Trace)
    - ...
- Output
  - out.tr
    - timestamp $\text{throughput}_{\text{avg}}$

Score

- Programming: 60%
  - Basic: 3 public test cases: 30%
  - Basic: 2 private test cases: 20%
  - Advanced: 10*(k/N)%, k: rank, N: number of teams
- Report: 30%
  - Your algorithms: 15%
  - Implementation descriptions: 15%
- Demo: 10%
- Required documents:
  - Zip(team_id_version.zip)
    - Report, Source codes files, Output files

Deadline

- 2014/05/05
- e-mail: wn@csie.ntu.edu.tw

Demo

- 2014/05/06 night
  - The exact time will be announced in the next week

References

- **NS-2**
  - http://nsnam.isi.edu/nsnam/index.php/Main_Page
  - http://www.isi.edu/nsnam/ns/tutorial/
  - http://csie.nqu.edu.tw/smallko/ns2_old/ns2.htm
- **Ubuntu(13.04)**
  - http://jantivas.blogspot.se/2012/08/how-to-install-ns-all-in-one-235-in.html
- **Ubuntu(12.04)**
  - http://ajlinx.wordpress.com/2013/06/19/install-ns2-ns-allinone-2-35-on-ubuntu-12-04-for-beginners/

Q&A