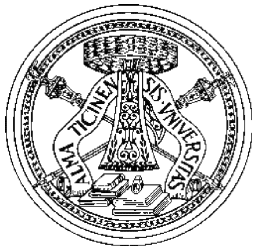
UNIVERSITÀ DEGLI STUDI DI PAVIA
FACOLTÀ DI INGEGNERIA



Primi passi con Ns-2

Emanuele Goldoni

Riferimenti

- Emanuele Goldoni
  - Laboratorio Reti (MN)
    - Tel. 0376-286234
    - Web: http://netlab-mn.unipv.it
  - E-mail: emanuele.goldoni@gmail.com
- Slide sul sito del corso
  - http://www.unipv.it/retical/

NS-2 Elements

- Create the event scheduler
- Turn on tracing
- Create network
- Setup routing
- Insert errors
- Create transport connection
- Create traffic
- Transmit application-level data

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`**

Example 1

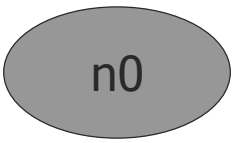
simple.tcl

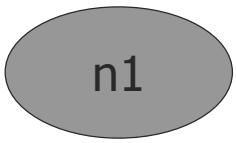
```
set ns [new Simulator]
$ns at 1 “puts \“Hello World!\””
$ns at 1.5 “exit”
$ns run
```

```
swallow 74% ns simple.tcl
Hello World!
swallow 75%
```


Creating Network

- Nodes
  - **set n0 [$ns node]**
  - **set n1 [$ns node]**





Creating Network

- Links and queuing
  - **$ns duplex-link $n0 $n1 <bandwidth> <delay> <queue_type>**
    - **<bandwidth>:** 1000b, 1kb, 0.001Mb, …
    - **<delay>:** 1ms, 0.001s, …
    - **<queue_type>:** DropTail, RED, CBQ, FQ, SFQ, DRR.

Creating Network

- LAN
  - **$ns make-lan <node_list> <bandwidth> <delay> <ll_type> <ifq_type> <mac_type> [<channel_type>]**
    - **<ll_type>:** LL
    - **<ifq_type>:** Queue/DropTail,
    - **<mac_type>:** MAC/802_3
  - Es:

```
$ns make-lan "$n1 $n2 $n3 $n4 $n5" 10Mb
20ms LL Queue/DropTail Mac/802_3
```

Tracing

- Trace packets on all links (!)
  - #Open the NAM trace file
    ```
    set nf [open out.nam w]
    $ns namtrace-all $nf
    ```
    #Open the Trace file
    ```
    set tf [open out.tr w]
    $ns trace-all $tf
    ```
  - Must appear immediately after creating scheduler

- Turn on tracing on specific links
  - **`$ns trace-queue $n0 $n1`**
  - **`$ns namtrace-queue $n0 $n1`**

Trace Files

(event, time, from_node, to_node, pkt type, pkt size, flags, fid, src_addr, dst_addr, seq_num, pkt_id)

Events: r received, **+** enqueued, **-** dequeued, **d** dropped

```
r 1.152965 0 2 tcp 1040 ------- 1 0.0 3.0 4 137
+ 1.152965 2 3 tcp 1040 ------- 1 0.0 3.0 4 137
r 1.154 1 2 cbr 1000 ------- 2 1.0 3.1 130 138
+ 1.154 2 3 cbr 1000 ------- 2 1.0 3.1 130 138
r 1.154706 2 3 cbr 1000 ------- 2 1.0 3.1 127 133
- 1.1556 2 3 tcp 1040 ------- 1 0.0 3.0 4 137
...
d 1.337868 2 3 tcp 1040 ------- 1 0.0 3.0 26 193
r 1.338 1 2 cbr 1000 ------- 2 1.0 3.1 153 197
+ 1.338 2 3 cbr 1000 ------- 2 1.0 3.1 153 197
d 1.338 2 3 cbr 1000 ------- 2 1.0 3.1 153 197
```

Post Analysis

```
r 1.152965 0 2 tcp 1040 ------- 1 0.0 3.0 4 137
+ 1.152965 2 3 tcp 1040 ------- 1 0.0 3.0 4 137
r 1.154 1 2 cbr 1000 ------- 2 1.0 3.1 130 138
+ 1.154 2 3 cbr 1000 ------- 2 1.0 3.1 130 138
r 1.154706 2 3 cbr 1000 ------- 2 1.0 3.1 127 133
- 1.1556 2 3 tcp 1040 ------- 1 0.0 3.0 4 137
...
d 1.337868 2 3 tcp 1040 ------- 1 0.0 3.0 26 193
r 1.338 1 2 cbr 1000 ------- 2 1.0 3.1 153 197
+ 1.338 2 3 cbr 1000 ------- 2 1.0 3.1 153 197
d 1.338 2 3 cbr 1000 ------- 2 1.0 3.1 153 197
```


Example 2

```
set ns [new Simulator]

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

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

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

$ns run
```

Creating Connection: UDP

- UDP

```
set udp [new Agent/UDP]
set null [new Agent/Null]

$ns attach-agent $n0 $udp0
$ns attach-agent $n1 $null

$ns connect $udp0 $null
```

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 Connection: TCP

- Full TCP

```
set tcp0 [new Agent/TCP/FullTcp]
$tcp0 set window_ 30
$tcp0 set segsize_ 536
$ns attach-agent $n0 $tcp0

set sink0 [new Agent/TCP/FullTcp]
$ns attach-agent $n5 $sink0
$sink0 listen

$ns connect $tcp0 $sink0
```


Example 3

```
set ns [new Simulator]

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

set n0 [$ns node]
set n1 [$ns node]
$ns duplex-link $n0 $n1 1Mb 10ms DropTail

set udp0 [new Agent/UDP]
set null [new Agent/Null]
$ns attach-agent $n0 $udp0
$ns attach-agent $n1 $null
$ns connect $udp $null

$ns run
```

Creating Traffic: On Top of UDP

- CBR

```
set src [new Application/Traffic/CBR]
$src attach-agent $udp0
```

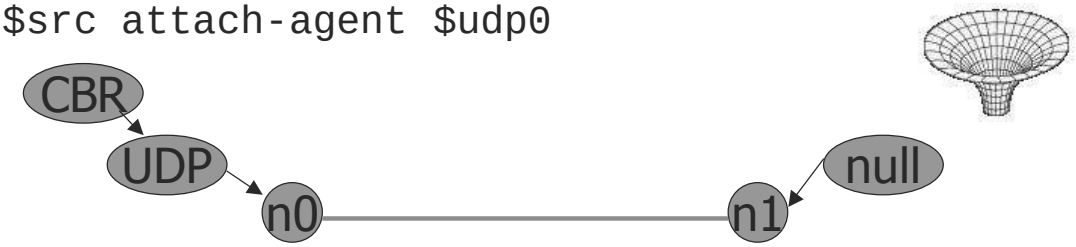
- Exponential or Pareto on-off

```
set src [new Application/Traffic/Exponential]
```

Or

```
set src [new Application/Traffic/Pareto]

$src attach-agent $udp0
```



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
```

Creating Traffic: Trace Driven

- Trace driven

```
set tfile [new Tracefile]
$tfile filename <file>
set src [new
  Application/Traffic/Trace]
$src attach-tracefile $tfile
```

- `<file>`:
  - Binary format (native!)
  - inter-packet time (msec) and packet size (byte)

Creating Traffic: Exponential

- Exponential On/Off

```
set ex [new Application/Traffic/Exponential]
$ex set packet-size <pktsize>
$ex set burst-time <burst-time>
$ex set idle-time <idle-time>
$ex set rate <rate>
```

Where

- **`<pktsize>:`** the constant size of the packets generated
- **`<burst-time>:`** the average “on” time for the generator
- **`<idle-time>:`** the average “off” time for the generator
- **`<rate>:`** the sending rate during “on” times

The ‘finish()’ Procedure

- Flush NS tracing
- Close tracing files
- Executing any post-analysis programs (display results, run Network Animator NAM, …)

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


Setting Traffic

- Setting Traffic Parameters

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

- Scheduling Events

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

Simulation Example

```
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
```

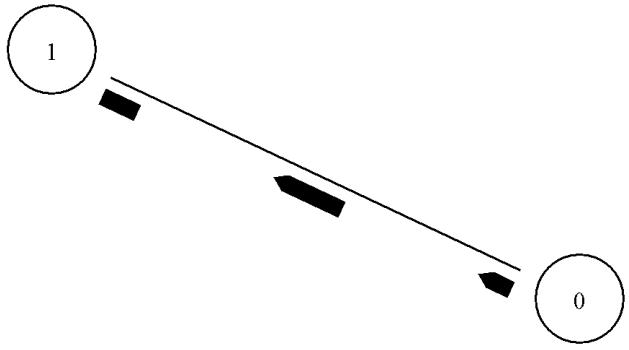

Simulation Example

```
set udp0 [new Agent/UDP]
set null [new Agent/Null]
$ns attach-agent $n0 $udp0
$ns attach-agent $n1 $null
$ns connect $udp $null

set src [new Application/Traffic/CBR]
$src set packetsize_ 500
$src set interval_ 0.005
$src attach-agent $udp0

$ns at 0.5 "$src start"
$ns at 4.5 "$src stop"
$ns at 20.0 "finish"

$ns run
```



Inserting Errors

- Creating Error Module
  - `set loss_module [new ErrorModel]`
  - `$loss_module set rate_ 0.01`
  - `$loss_module unit pkt`
  - `$loss_module ranvar [new RandomVariable/Uniform]`
  - `$loss_module drop-target [new Agent/Null]`
- Inserting Error Module
  - `$ns lossmodel $loss_module $n0 $n1`

Network Dynamics

- Link failures
  - Hooks in routing module to reflect routing changes
- Four models

```
$ns rtmodel Trace <config_file> $n0 $n1
$ns rtmodel Exponential {<params>} $n0 $n1
$ns rtmodel Deterministic {<params>} $n0 $n1
$ns rtmodel-at <time> up|down $n0 $n1
```

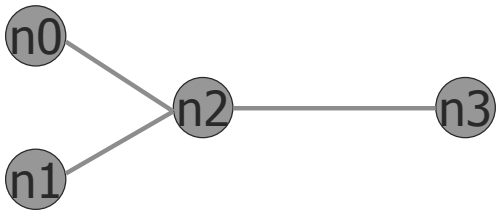
- Parameter list

```
[<start>] <up_interval> <down_interval> [<finish>]
```

Nodes Layout

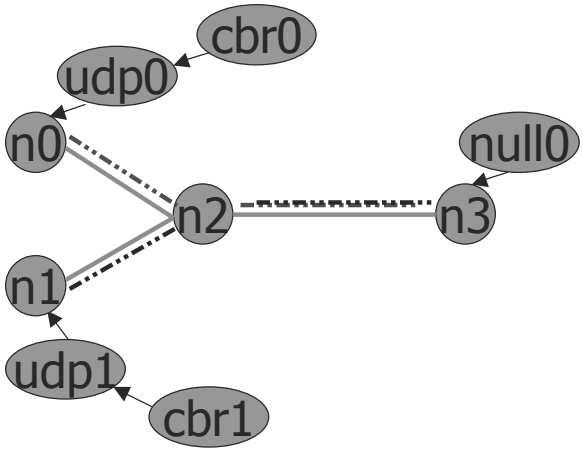
```
# create nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
# add links
$ns duplex-link $n0 $n2 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n3 $n2 1Mb 10ms DropTail
# layout
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

# right(0), right-up(45), right-down(315), left(180),
# left-up(135),left-down(225), up(90), down(270)
```



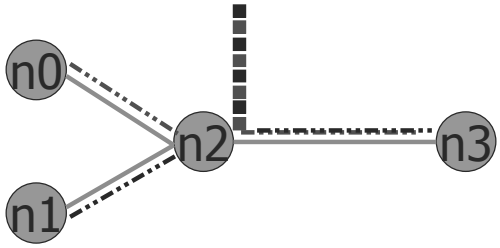
Marking the Flow

```
# create color classes
$ns color 1 Blue
$ns color 2 Red
# add connection between n0-n2,
# n1-n2, n2-n3
……
# add udp agents
……
# add sink agent
……
# connect agents
……
# add CBRs (cbr0 and cbr1)and connect them to udp agents
……
# mark the two flows with different colors
$udp0 set class_ 1
$udp1 set class_ 2
```



SFQ: Statically Fair Queue

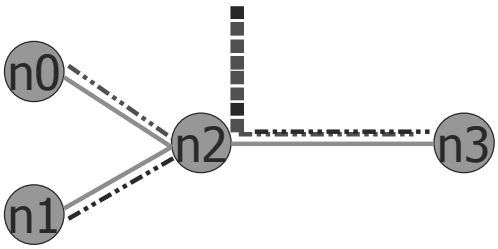
```
# add links, with an SFQ on n2-n3 link
$ns duplex-link $n0 $n2 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n3 $n2 1Mb 10ms SFQ
```



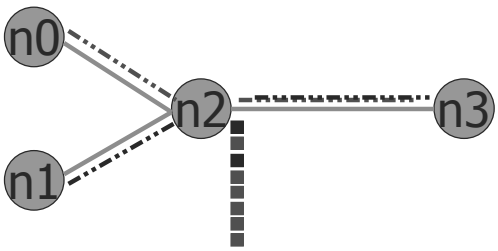
Statically fair queues applies the "round robin" approach (fair share).

Monitoring the Queue

```
# monitor the queue at 90 degree angle
$ns duplex-link-op $n2 $n3 queuePos 0.5
```



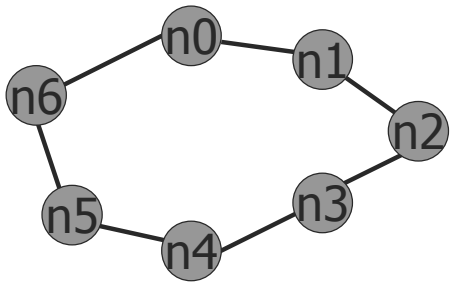

```
# monitor the queue at 270 degree angle
$ns duplex-link-op $n2 $n3 queuePos 1.5
```




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

Larger Topologies

```
# create 7 nodes
for {set i 0} {$i < 7} {incr i}  {
     set n($i) [$ns node]
}
# add 7 circular links
for {set i 0} {$i < 7} {incr i} {
     $ns  duplex-link
          $n($i) $n([expr [expr $i + 1]%7])
          1Mb 10ms DropTail
}
```



Riferimenti

- *ns2 Manual*, http://www.isi.edu/nsnam/ns/ns-documentation.html
- *Ns2 Tutorial*, http://www.isi.edu/nsnam/ns/tutorial/index.html
- *La simulazione di reti di calcolatori con ns*, aavv, http://www.dia.uniroma3.it/~impianti/1998-99/ns_due/
- *Primi passi con NS2*, E. Fasolo, http://www.dei.unipd.it/wdyn/?IDsezione=2562
- *Ns2 Simulazione di Reti Wireless 802.11*, M. Di Felice, http://www.cs.unibo.it/~difelice/risorse.htm
- *NS2 - Materiale di Laboratorio*, G. Maier, http://home.dei.polimi.it/maier/materiale.html