

NS2: Contents

- NS2 – Introduction to NS2 simulator
- Some NS2 examples

- RED example

- Enhanced RED example

- NS2 project work instructions

24.11.2004

1

Example 1

- Task
 - simulate a queue operating under RED control
 - using elementary topology
 - traffic is a superposition of greedy TCP sources
 - measure instantaneous queue length
 - file: redtcp.tcl

24.11.2004

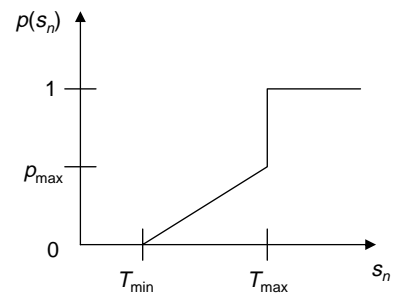
2

RED

- RED (Random Early Detection/Drop)
 - Active Queue Management (AQM) method proposed by S. Floyd
 - designed to cooperate with TCP-friendly congestion control
 - tries to prevent buffer overflows by discarding packets prior to the buffer becoming full
 - TCP friendly rate control reacts to packets losses and (some) sources slow down their sending rates \Rightarrow serious congestion is avoided
 - packet dropping probability depends on load
- RED algorithm (approximately)
 - for each arriving packet, compute exponentially averaged queue length (\approx load), s_n
 - drop packet with probability

$$s_n = (1 - \beta)s_{n-1} + \beta q_n$$

$$p_n = \begin{cases} 0, & s_n < T_{\min} \\ \frac{p_{\max}(s_n - T_{\min})}{T_{\max} - T_{\min}}, & T_{\min} \leq s_n \leq T_{\max} \\ 1, & s_n > T_{\max} \end{cases}$$

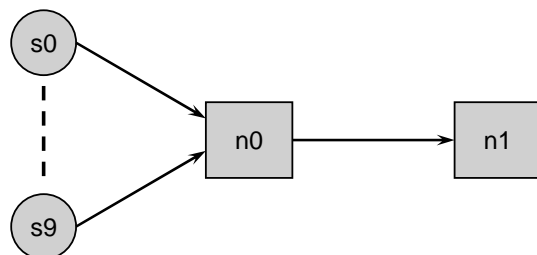


24.11.2004

3

Simulator objects for example 1

- Traffic sources
 - greedy TCP Reno sources (constantly send traffic)
 - need to create a TCP connection and attach an FTP agent to the TCP source
 - parameters
 - nof of sources
 - maximum window size
 - segment size
- Bottleneck link
 - finite buffer with RED queue
 - parameters
 - queue size
 - RED parameters
- Required traffic objects (topology)
 - 2 nodes
 - 1 link (with RED queue)
 - N TCP sources



24.11.2004

4

Tracing for example 1

- Aim:
 - trace the variable that represents instantaneous queue length
- See red.h (in /ns-allinone-2.1b9a/ns-2.1b9a/)
 - variables with type TracedInt (or TracedDouble, etc.) are variables defined in the C++ class that are also visible at the OTcl level
 - to find out what traced variables are defined for RED queue (run in above directory)


```
fgrep Traced red.h
```
 - variable "TracedInt cur_" represents instantaneous queue length (as seen by an arriving packet)
- Creating a trace object
 - create file for output and create a trace object (\$redobj represents a RED queue object)


```
set outfile [open data.txt w]
$redobj trace cur_
$redobj attach $outfile
```
 - tracing started after warm-up time

24.11.2004

5

Plotting the queue length process

- One can plot the realization of the queue length process
 - can experiment with RED parameters to examine stability
- The output file contains rows with following entries:


```
Q 20.0041 11
Q 20.0151 9
Q 20.0225 10 ...
```
- Remove extra 'Q' from beginning of each line by:
 - from command line: `awk '{print $2, $3}' qlen.dat > qq1.dat`
 - from ns2-tcl script: `exec awk {{print $2, $3}} qlen.dat > qq1.dat`
- Plot the data in qq1.dat using xgraph
 - from command line: `xgraph qq1.dat`
 - from n2-tcl script: `exec xgraph qq1.dat`

24.11.2004

6

Running the tcl-scripts

- Ns2 is not installed in the machines of B215
- Take a remote connection to one of the Linux machines in Maarintalo. Try for example some of the pharaoh machines in Maari A
 - <http://www.hut.fi/atk/luokat/Maari-A.html>
- In order to be able to use ns2, you first have to do the following
 - Type in your shell: `source /p/edu/s-38.180/usens2.csh`
 - This file contains the required settings for environmental variables.
 - Give this command each time you start an ns2 session in a shell.
- After that you can use ns2 simply by writing in your shell
 - `ns my_script.tcl`
- Top directory where ns2 source files are is:
 - `/p/edu/s-38.180/src/ns-2.1b9a_standard/`

24.11.2004

7

NS2: Contents

- NS2 – Introduction to NS2 simulator
- Some NS2 examples
 - RED example
 - Enhanced RED example
- NS2 project work instructions

24.11.2004

8

Example 2

- Task
 - measure the steady state mean of the instantaneous queue length as a function of offered load
- We need to ...
 - be able to run consecutive independent simulations, and
 - compute steady state mean from the measured data
 - files: redtcpmain.tcl and redtcpsub.tcl

24.11.2004

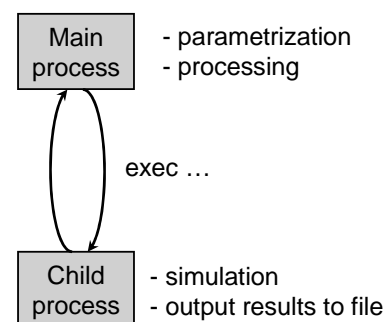
9

Running independent simulations

- To run independent simulations, we must either be able to ...
 - re-initialize all simulator objects (simulation clock, event scheduler, all traffic objects, etc.), or somehow re-execute simulation scripts
 - re-initializing in ns2 is difficult \Rightarrow we need to make repeated executions of ns2 scripts

- In Unix, the operating system executes each Tcl/ns2 script as a process

- For repeated simulations we need ...
 - a main program that controls and parametrizes simulations, and a sub-program that executes each simulation run
 - in unix terminology, we need a main process that spawns a child process for execution of the actual simulation runs
 - in Tcl (and most script languages), the command for this is "exec ..."



```
exec ns myscript.tcl command_line_args
```

24.11.2004

10

Measuring time averages

- Post-processing
 - read data from output file and process it
 - in our case, output consists of tuples <Q, time, queue_len>

- Reading from file
 - open file for reading and read a line from the file

```
set $outfile [open data.txt r]
gets $outfile tmp
```

- To compute time average of the data, use Integrator class
 - creating and adding points

```
set integ [new Integrator]
$integ newpoint $time $value
```

- variable `sum_` contains cumulative sum