Protocoale de Comunicatii - Laborator 5

Protocoale de Dirijare cu Network Simulator (ns2)

NS-2 este unul dintre cele mai folosite simulatoare de retea. NS poate fi instalat in urmatorii pasi simpli (pt Ubuntu recent, aveti nevoie de root):

apt-get install ns2
apt-get install nam

Folosirea NS necesita descrierea experimentelor in limbajul TCL. Pentru a porni ns se poate executa direct si specifica experimentul din linia de comanda; se prefera totusi descrierea experimentului intr-un fisier separat (de ex. test.tcl) si rularea "ns test.tcl".

NS poate fi configurat sa salveze datele rezultate din simulare pe disk, iar acestea pot fi vizualizate folosind utilitarul network animator (nam): "nam fisier_iesire".

In acest laborator vom studia sintaxa de baza a NS pornind de la exemplul dat in fisierul routing.tcl atasat. Vom descrie acum pe scurt ce functionalitate implementeaza fiecare parte a script-ului.

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

Prima linie creaza simulatorul (prima linie) iar a doua linie deschide un fisier numit out.nam pentru salvarea rezultatelor rezultatelor. A treia linie apeleaza functia namtrace-all cu parametrul $nf; simulatorul va scrie rezultatele simularii in acest fisier.

Pentru a opri simularea, se defineste o functie simpla care goleste output-ul ns (apelul flush-trace), inchide fisierul si apeleaza nam. Aceasta functie va fi executata atunci cand timpul simulat ajunge la 5s.

```
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
}

$ns at 5.0 "finish"
```

Pentru a face lucruri utile in ns trebuie sa definim graful retelei. Un nod se creaza apeland functia $ns node. Mai jos cream doua noduri numite n0 si n1.

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

Pentru a conecta doua noduri trebuie sa definim o legatura de date astfel (parametrii sunt nodurile capete, viteza, latenta, si tipul de coada pe legatura respectiva; droptail inseamna ca mesajele se pierd atunci cand coada este plia). Duplex inseamna ca legatura poate trimite mesaje in ambele directii:

$ns duplex-link $n0 $n1 100Kb 100ms DropTail

In sfarsit, se poate configura un cost pe o legatura de date care va fi folosit de algoritmul de dirijare; costurile se configureaza pe o singura directie;

$ns cost $n0 $n1 3

In mod standard, ns calculeaza static rutele intre noduri folosind algoritmul disjkstra. Daca dorim ca rutarea sa se faca dinamic trebuie sa configuram ns explicit. Parametrii pot fi DV (distance vector) sau LS (link state):

$ns rtproto DV

Pentru a transmite date trebuie sa cream un transmitator de nivel transport (TCP sau UDP), sa ii atasam unui nod, sa cream un receptor si sa il atasam unui nod, si sa conectam transmitatorul cu receptorul. In routing.tcl se creaza o sursa UDP de la nodul n0 la nodul n2. Sursa va fi configurata sa inceapa transmisia dupa 0.5 secunde de la inceputul simularii, si sa se opreasca dupa 4.5 secunde.

In sfarsit, putem simula defecte in retea, de exemplu oprind legatura n1 n2 la secunda 1 si repornind-o la secunda 2:

$ns rtmodel-at 1.0 down $n1 $n2
$ns rtmodel-at 2.0 up $n1 $n2

Ultima linie din fisier ruleaza simulatorul:

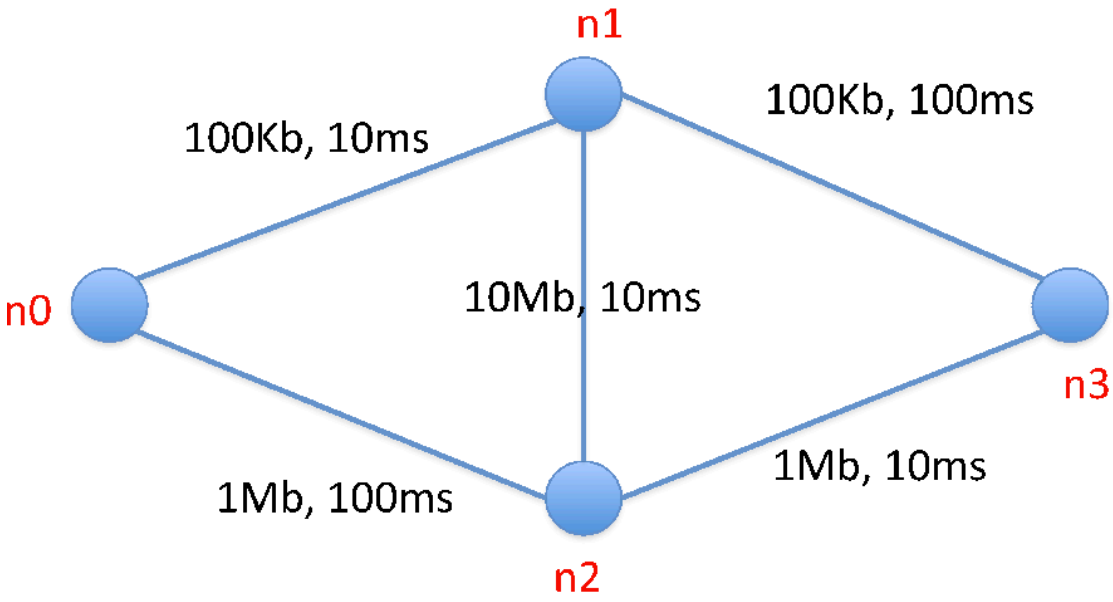
$ns run

Exercitii:

- modificati costurile astfel incat protocolul de rutare sa prefere n0-n1-n2 intre n0 si n2; simulati un defect pe legatura n1-n2 si observati ce se intampla

Creati topologia descrisa in figura de mai jos, unde nodul n0 va trimite catre nodul n3:

- Simulati defecte si observati ce se intampla. Cate pachete se pierd?
- Setari costurile pentru topologia data pentru a obtine viteza maxima
- Setari costurile pentru a obtine latenta minima.


n1
100Kb, 10ms
100Kb, 100ms
n0
10Mb, 10ms
n3
1Mb, 100ms
1Mb, 10ms
n2