Univerzitet u Beogradu
Elektrotehnički fakultet

Jelena Živković, 251/2012

Simulacija dinamičkog rutiranja u mrežnom simulatoru ns3

mentor:
Dr Milan Bjelica

Beograd, maj 2015.

Abstract

U ovom radu je prikazana simulacija dinamičkog rutiranja uz korišćenje simulatora ns3. Kod programa je pisan u programskom jeziku C++. Dobijeni rezultati su prikazani pomoću... Aplikativni prikaz simulacije odradjen je uz pomoć simulacionog softvera NetAnim.
Ključne reči: simulacija, ns3, NetAnim, Wireshark, C++

Sadržaj

Slike

Chapter 1

Uvod

1.1 Simulator ns3

Simulator ns -3 je slobodni softverski alat za simulaciju fiksnih i bežicnih telekomunikacionih mreža, namenjen za istraživačke i obrazovne svrhe. Projekat njegovog razvoja je pokrenut 2006. godine, s ciljem da se napravi potpuno novi simulator, koji bi bio jednostavniji za korišćenje i čiji bi kod bio konzistentniji od starog simulatora ns -2, koji je do tada široko korišćen u istraživačke svrhe.
Simulator ns -3 nije naredna niti unapredjena verzija simulatora ns -2, već se radi o potpuno novom programu. Nije predvidjena uzajamna kompatibilnost, pa se simulaciona skripta za ns -2 ne mogu direktno primeniti u ns -3.
Preporučuje se instaliranje simulatora pod operativnim sistemom Linux. Programski kod je napisan u jeziku C++, dok se sama simulacija izvršava po principu simulacije diskretnih događaja.

1.1.1 Organizacija simulacionog modela ns-3

Organizacija simulacionog modela u ns -3 prikazana je na slici 1.1.
Elementi mreže se nazivaju čvorovima (node). Neki čvorovi odgovaraju mrežnim uredjajima, kao što su to npr. ruteri, a

neki krajnjim sistemima, poput umreženih računara. Ako čvorovi odgovaraju krajnjim sistemima, na njima se izvršavaju aplikacije (application). Na jednom čoru se može izvršavati više aplikacija, a primeri onih koje BulkSendApplication , OnOffApplication , PacketSink , Ping6 , Radvd , UdpEcho i UdpClientServer . Aplikacije generišu i primaju saobraćaj, koji se posredstvom primenjenih protokola (npr. UDP, TCP, IPv6, IPv4, OSPF itd) prenosi ka sprezi s mrežom (mrežnom interfejsu – net device); nju možemo shvatiti kao mrežnu karticu na računaru. Sprega odgovara primenjenom kanalu (channel), pa tako postoje posebne sprege za kanale tipa tačka–tačka, CSMA, Aloha, WiFi, Wimax, LTE itd.Jedan čvor se preko odgovarajućih sprega može istovremeno povezati na više kanala.

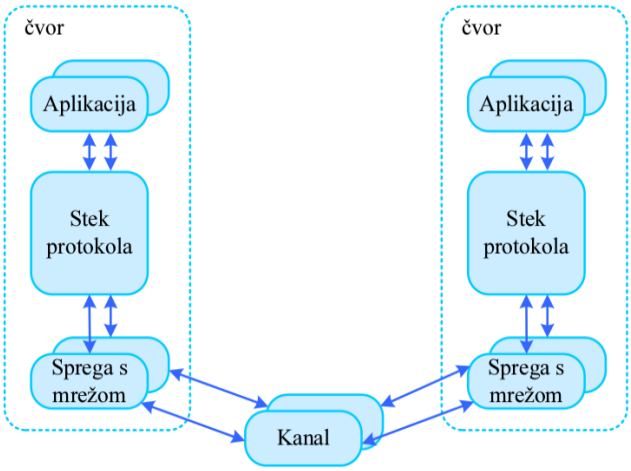

Figure 1.1: Organizacija simulacionog modela u ns-3.

Chapter 2

Dinamičko rutiranje

2.1 Zadatak

U mreži sa slike , izvor on-off TCP saobraćaja je priključen na čvor n0, a primalac na čvor n4. Parametri linkova su dati u tabeli 2.1 tab:par).

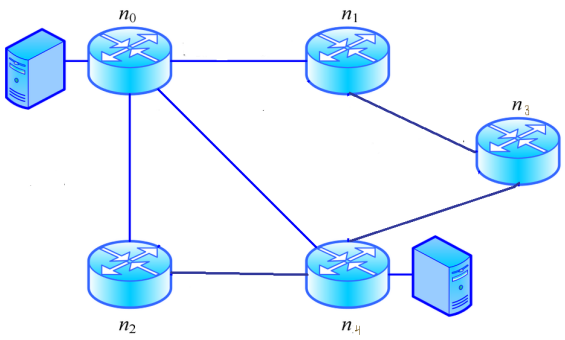

Figure 2.1: Izgled topologije mreže.

Posmatra se sledeći scenario dešavanja:

- u trenutku t=1s,izvor počinje emitovati saobračaj vršnim protokom 100 kb/s, uz veličinu TCP paketa 150 B,
- u trenutku t =2 s, ukida se link izmedju čvorova n2 i n4 ,
- u trenutku t =3 s, ukida se link izmedju čvorova n0 i n4 ,

Table 2.1: Parametri linkova

Link	Kapacitet	Propagaciono kašnjenje	Metrika
n0-n1	10 Mb/s	4 ms	4
n0-n2	10 Mb/s	4 ms	2
n0-n4	10 Mb/s	3 ms	6
n1-n3	10 Mb/s	2 ms	2
n2-n4	10 Mb/s	2 ms	1
n3-n4	10 Mb/s	2 ms	3

  - u trenutku t =4 s, ponovo se uspostavlja link izmedju čvorova n2 i n4

- u trenutku t =5 s, isključuje se izvor.
  imullacijom je potrebno odrediti putanje saobraćaja

2.2 Opis simulacije

Da bismo ilustrovali razli cite mogućnosti konfigurisanja simulacionog modela u ns-3, izvor saobraćaja smo pridružili postojećem čvoru mreže n 0 , u vidu OnOff aplikacije, dok smo prijemnik paketa (PacketSink) pridru zili novom cvoru n5. Čvorovi n4 i 5 povezani su linkom zanemarivog ka2njenja, kome dodeljujemo IP adrese iz opsega 164.70.7.0/24.
Metodom SetMetric zadajmo metriku linkova.Primetimo da se, iako su linkovi „tačka-tačka" dvosmerni, mogu zadati različite vrednosti metrike za dva smera prenosa.
Dogadjaji tokom izvršavanja simulacije se specifciraju metodom Simulator::Schedule . U našem slučju, njen poslednji argument se odnosi na indeks posmatranog mrežnog interfejsa, pri čemu 0 odgovara tzv. loopback interfejsu, dok su interfejsi ka fizičkim linkovima numerisani po redu kojim su pravljeni. Zanimljivo je da se linkovi ne isključuju direktno, već raskidanjem veze protokolskog steka sa mrežnim interfejsom.
Izveštaje ćemo snimiti kao trace, pcap i xml datoteke, a saču-

vaćemo i tabele rutiranja posle karakteristiÂŁnih dogadjaja u mreži. Na početku programa moramo eksplicitno tražiti da se tabele rutiranja ažuriraju posle svake promene stanja linkova, tako što ćemo vrednost parametra ns3::Ipv4GlobalRouting::RespondToInterfaceEvents postaviti na true .

2.3 Kod rešenja

```
#include <iostream>
#include <fstream>
#include <string>
#include <cassert>

#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/network-module.h"
#include "ns3/applications-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/netanim-module.h"
using namespace ns3;

int main (int argc, char *argv[])
{

// zadajemo azuriranje tabela rutiranja posle svake izmene stanja li
Config::SetDefault ("ns3::Ipv4GlobalRouting::RespondToInterfaceEvent
BooleanValue (true));


// pravimo cvorove:

NodeContainer c;
c.Create (6);

// instaliramo stek protokola na cvorovima:

InternetStackHelper internet;
internet.Install (c);
```

```
NodeContainer n0n1 = NodeContainer (c.Get (0), c.Get (1));
NodeContainer n0n2 = NodeContainer (c.Get (0), c.Get (2));
NodeContainer n0n4 = NodeContainer (c.Get (0), c.Get (4));
NodeContainer n1n3 = NodeContainer (c.Get (1), c.Get (3));
NodeContainer n2n4 = NodeContainer (c.Get (2), c.Get (4));
NodeContainer n3n4 = NodeContainer (c.Get (3), c.Get (4));
NodeContainer n4n5 = NodeContainer (c.Get (4), c.Get (5));

// pravimo linkove:
PointToPointHelper p2p;

p2p.SetDeviceAttribute ("DataRate", StringValue ("10Mbps"));
p2p.SetChannelAttribute ("Delay", StringValue ("4ms"));

//instaliranje linka izmedju cvorova
NetDeviceContainer d0d1 = p2p.Install (n0n1);
NetDeviceContainer d0d2 = p2p.Install (n0n2);
p2p.SetChannelAttribute ("Delay", StringValue ("3ms"));

NetDeviceContainer d0d4 = p2p.Install (n0n4);

p2p.SetChannelAttribute ("Delay", StringValue ("2ms"));

NetDeviceContainer d1d3 = p2p.Install (n1n3);
NetDeviceContainer d2d4 = p2p.Install (n2n4);
NetDeviceContainer d3d4 = p2p.Install (n3n4);

p2p.SetChannelAttribute ("Delay", StringValue ("0ms"));
NetDeviceContainer d4d5 = p2p.Install (n4n5);

// dodeljujemo IP adrese:
```

```
Ipv4AddressHelper ipv4;

ipv4.SetBase ("164.70.1.0", "255.255.255.0");
Ipv4InterfaceContainer i0i1 = ipv4.Assign (d0d1);
ipv4.SetBase ("164.70.2.0", "255.255.255.0");
Ipv4InterfaceContainer i0i2 =ipv4.Assign (d0d2);
ipv4.SetBase ("164.70.3.0", "255.255.255.0");
Ipv4InterfaceContainer i0i4 =ipv4.Assign (d0d4);
ipv4.SetBase ("164.70.4.0", "255.255.255.0");
Ipv4InterfaceContainer i1i3 =ipv4.Assign (d1d3);
ipv4.SetBase ("164.70.5.0", "255.255.255.0");
Ipv4InterfaceContainer i2i4 = ipv4.Assign (d2d4);
ipv4.SetBase ("164.70.6.0", "255.255.255.0");
Ipv4InterfaceContainer i3i4 = ipv4.Assign (d3d4);
ipv4.SetBase ("164.70.7.0", "255.255.255.0");
Ipv4InterfaceContainer i4i5 = ipv4.Assign (d4d5);


// zadajemo metrike linkova
// (pretpostavljena vrednost je 1):

i0i1.SetMetric (0, 4);
i0i1.SetMetric (1, 4);
i0i2.SetMetric (0, 2);
i0i2.SetMetric (1, 2);
i0i4.SetMetric (0, 6);
i0i4.SetMetric (1, 6);
i1i3.SetMetric (0, 2);
i1i3.SetMetric (1, 2);
i3i4.SetMetric (0, 3);
i3i4.SetMetric (1, 3);

// pravimo tabele rutiranja:
```

```
Ipv4GlobalRoutingHelper::PopulateRoutingTables ();


// pravimo izvor saobracaja
// i instaliramo ga na cvor 0:

uint16_t port = 4001;
OnOffHelper onoff ("ns3::TcpSocketFactory",InetSocketAddress (i4i5.G
onoff.SetAttribute ("OnTime",  StringValue("ns3::ConstantRandomVaria
onoff.SetAttribute ("OffTime",  StringValue("ns3::ConstantRandomVari
onoff.SetAttribute ("DataRate", StringValue ("100kbps"));
onoff.SetAttribute ("PacketSize", UintegerValue (150));

ApplicationContainer apps = onoff.Install (c.Get (0));

apps.Start (Seconds (1.0));
apps.Stop (Seconds (5.0));

// pravimo prijemnik saobracaja
// i instaliramo ga na cvor 4:

PacketSinkHelper sink ("ns3::TcpSocketFactory",Address (InetSocketAd

apps = sink.Install (c.Get (5));
apps.Start (Seconds (1.0));
apps.Stop (Seconds (5.0));

// zadajemo snimanje izvestaja:

AsciiTraceHelper ascii;
Ptr<OutputStreamWrapper> stream = ascii.CreateFileStream ("dinamicko
```

```
p2p.EnableAsciiAll (stream);
internet.EnableAsciiIpv4All (stream);

p2p.EnablePcapAll ("dinamicko");

// postavljamo pointere:

Ptr<Node> n0 = c.Get (0);
Ptr<Ipv4> ipv4n0 = n0->GetObject<Ipv4> ();
Ptr<Node> n2 = c.Get (2);
Ptr<Ipv4> ipv4n2 = n2->GetObject<Ipv4> ();


// u t = 2 s iskljucujemo link n2-n4:

Simulator::Schedule (Seconds (2), &Ipv4::SetDown, ipv4n2, 2);

// u t = 3 s iskljucujemo link n0-n4:

Simulator::Schedule (Seconds (3), &Ipv4::SetDown, ipv4n0, 3);

  // u t = 4 s ukljucujemo link n1-n2:

Simulator::Schedule (Seconds (4), &Ipv4::SetUp, ipv4n2, 2);

// snimamo tabele rutiranja:

Ipv4GlobalRoutingHelper g;
Ptr<OutputStreamWrapper> routingStream =Create<OutputStreamWrapper>

g.PrintRoutingTableAllAt (Seconds (1), routingStream);
g.PrintRoutingTableAllAt (Seconds (2), routingStream);
g.PrintRoutingTableAllAt (Seconds (3), routingStream);
```

```
g.PrintRoutingTableAllAt (Seconds (4), routingStream);
g.PrintRoutingTableAllAt (Seconds (5), routingStream);

//NetAnim
std::string animFile = "dini.xml";
AnimationInterface anim(animFile);
// postavljamo potrebne pointere
Ptr<Node> n1 = c.Get(1);
Ptr<Node> n3 = c.Get(3);
Ptr<Node> n4 = c.Get(4);
Ptr<Node> n5 = c.Get(5);
// zadajemo poziciju cvora n0 u NetAnim animaciji
anim.SetConstantPosition( n0, 100, 100);
anim.SetConstantPosition( n1, 200, 100);
anim.SetConstantPosition( n2, 100, 200);
anim.SetConstantPosition( n3, 200, 200);
anim.SetConstantPosition( n4, 300, 150);
anim.SetConstantPosition( n5, 300, 200);


// pokre\' cemo simulaciju
// i po njenom zavrsetku uni\v stavamo sve napravljene objekte:

Simulator::Run ();
Simulator::Destroy ();
}
```

2.4 Rezultati

Razmotrimo sada analizu rezultata koje dobijamo po izvršavanju simulacije. Primenjeni protokol rutiranja (OSPF) bira rutu ka odredištu koja ima najmanju metriku; na početku je to n0 - n2 - n4 -n5.Kada u trenutku t = 2 s bude ispao link n2-n4, sabraćaj

će se preusmeriti na rutu n0 - n4 - n5. U trenutku t = 3 s ispada i link izmedju čvorova n0-n4, nova putanja će bii n0-n1-n3-n4-n5. Kada se u trenutu t = 4 s bude ponovo uspostavio link izmedju čvorova n 2 i n 2 , saobra caj će se vratiti na prvobitnu putanju n 0 - n2 - n4 - n 5 .

Gornja zapažanja potvrdjuju i tabele rutranja, koje zbog obimnosti navodimo samo za čvor 0 u trenucima t =1s, t =2 s i t=3s.

```
Node: 0 Time: 1s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -
164.70.1.0      0.0.0.0         255.255.255.0   U     0      -
164.70.2.0      0.0.0.0         255.255.255.0   U     0      -
164.70.3.0      0.0.0.0         255.255.255.0   U     0      -
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting
Destination     Gateway         Genmask         Flags Metric Ref
164.70.2.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.5.1      164.70.2.2      255.255.255.255 UH    -      -
164.70.3.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.5.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.6.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.7.1      164.70.2.2      255.255.255.255 UH    -      -
164.70.1.2      164.70.1.2      255.255.255.255 UH    -      -
164.70.4.1      164.70.1.2      255.255.255.255 UH    -      -
164.70.7.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.4.2      164.70.1.2      255.255.255.255 UH    -      -
164.70.4.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.6.1      164.70.1.2      255.255.255.255 UH    -      -
164.70.6.1      164.70.2.2      255.255.255.255 UH    -      -
127.0.0.0       164.70.2.2      255.0.0.0       UG    -      -
164.70.2.0      164.70.2.2      255.255.255.0   UG    -      -
164.70.5.0      164.70.2.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.2.2      255.0.0.0       UG    -      -
```

```
164.70.3.0      164.70.2.2      255.255.255.0   UG    -      -
164.70.5.0      164.70.2.2      255.255.255.0   UG    -      -
164.70.6.0      164.70.2.2      255.255.255.0   UG    -      -
164.70.7.0      164.70.2.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.2.2      255.0.0.0       UG    -      -
164.70.7.0      164.70.2.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
127.0.0.0       164.70.2.2      255.0.0.0       UG    -      -
164.70.4.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.4.0      164.70.2.2      255.255.255.0   UG    -      -
164.70.6.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.6.0      164.70.2.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
164.70.1.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.4.0      164.70.1.2      255.255.255.0   UG    -      -

Node: 0 Time: 2s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -
164.70.1.0      0.0.0.0         255.255.255.0   U     0      -
164.70.2.0      0.0.0.0         255.255.255.0   U     0      -
164.70.3.0      0.0.0.0         255.255.255.0   U     0      -
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting
Destination     Gateway         Genmask         Flags Metric Ref
164.70.2.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.1.2      164.70.1.2      255.255.255.255 UH    -      -
164.70.4.1      164.70.1.2      255.255.255.255 UH    -      -
164.70.3.2      164.70.3.2      255.255.255.255 UH    -      -
164.70.6.2      164.70.3.2      255.255.255.255 UH    -      -
164.70.7.1      164.70.3.2      255.255.255.255 UH    -      -
164.70.4.2      164.70.1.2      255.255.255.255 UH    -      -
164.70.6.1      164.70.1.2      255.255.255.255 UH    -      -
```

```
164.70.7.2      164.70.3.2      255.255.255.255 UH    -      -
127.0.0.0       164.70.2.2      255.0.0.0       UG    -      -
164.70.2.0      164.70.2.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
164.70.1.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.4.0      164.70.1.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
164.70.4.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.6.0      164.70.1.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.3.2      255.0.0.0       UG    -      -
164.70.3.0      164.70.3.2      255.255.255.0   UG    -      -
164.70.5.0      164.70.3.2      255.255.255.0   UG    -      -
164.70.6.0      164.70.3.2      255.255.255.0   UG    -      -
164.70.7.0      164.70.3.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.3.2      255.0.0.0       UG    -      -
164.70.7.0      164.70.3.2      255.255.255.0   UG    -      -

Node: 0 Time: 3s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -
164.70.1.0      0.0.0.0         255.255.255.0   U     0      -
164.70.2.0      0.0.0.0         255.255.255.0   U     0      -
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting
Destination     Gateway         Genmask         Flags Metric Ref
164.70.2.2      164.70.2.2      255.255.255.255 UH    -      -
164.70.1.2      164.70.1.2      255.255.255.255 UH    -      -
164.70.4.1      164.70.1.2      255.255.255.255 UH    -      -
164.70.4.2      164.70.1.2      255.255.255.255 UH    -      -
164.70.6.1      164.70.1.2      255.255.255.255 UH    -      -
164.70.6.2      164.70.1.2      255.255.255.255 UH    -      -
164.70.7.1      164.70.1.2      255.255.255.255 UH    -      -
164.70.7.2      164.70.1.2      255.255.255.255 UH    -      -
```

```
127.0.0.0       164.70.2.2      255.0.0.0       UG    -      -
164.70.2.0      164.70.2.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
164.70.1.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.4.0      164.70.1.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
164.70.4.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.6.0      164.70.1.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
164.70.3.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.5.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.6.0      164.70.1.2      255.255.255.0   UG    -      -
164.70.7.0      164.70.1.2      255.255.255.0   UG    -      -
127.0.0.0       164.70.1.2      255.0.0.0       UG    -      -
164.70.7.0      164.70.1.2      255.255.255.0   UG    -      -
```

U trenutku t =1 s, najbolje rute iz čvora n0 ka svim ostaim članovima idu preko intrfejsa interfejsa 2, koji vodi ka čvoru n2. U trenutku t= 2 , ispada lik izmedju čvorova n2 i n4. U tabeli rutiranja čvora n0 stoga će rute ka čvoru 4 ići preko interfejsa 3.

Chapter 3

Animacija rezultata simulacije

3.1 NetAnim

Animator NetAnim je namenje radu s C++ programskim skriptama. Da bismo mogučili animaciju rezultata našeg programa, pozivmo načtku modul netanim

```
#include "ns3/netanim-module.h"
```

Potom definišemo izlaznu xml datoteku i položaje čvorova, kao što je prikazano u kodu u poglavlju 2.3.
Nakon izvšavanja simulacije, u osnonom direktorijumu se pravi datoteka dini.xml . Kada je budemo otvorili u programu NetAnim, videÂ˘emo sledeÂ˘u sliku:

Pokretanjem animacije, videćemo tokove paketa po linkovima. Prikazi u karakteristi cnim trenucima dati su na slici

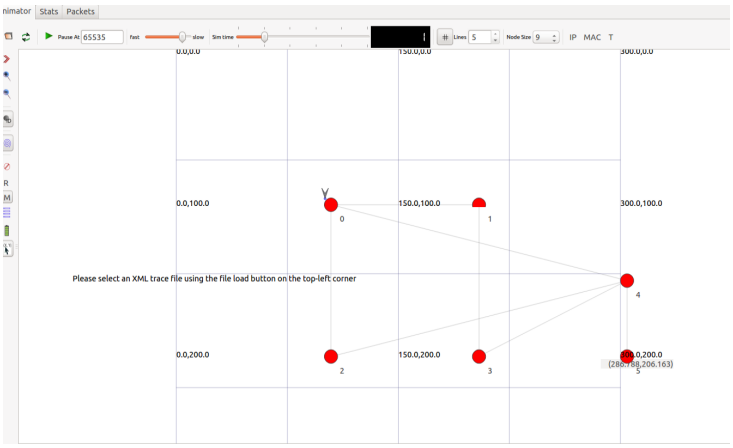

Figure 3.1: Početno stanje mreže.

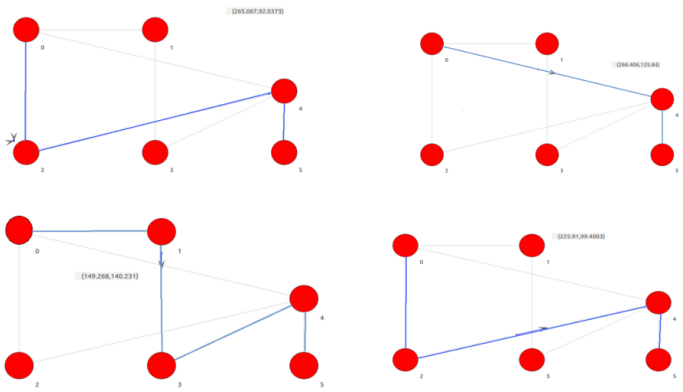

Figure 3.2: Stanje mreže u trenucima 1s, 2s, 3s i 4s respektivno.

Chapter 4

Zaključak

U radu je prkazan prenos TCP paketa kroz mrežu odredjene topologije uz izmenu stanja u mreži u odredjenim trenucima. Cilj je bio da koristeći vec implementirane klase i metode, koje nam pruža ns-3, prikažemo ponašanje mreže tokom prenosa paketa kroz nju. Dato rešenje e može e nadogradjiti u smislu povečanja broja čvorova, promeni topologije ili pak promeni tipa paketa koji se prenosi.

Literatura

[1] Dr Milan Bjelica **MODELIRANJE I SIMULACIJA U TELEKOMUNIKACIJAMA**, Beograd 2013., Elektrotehnički fakultet

[2] Asistent dr Nataša Maksić, **Mrežni simulator ns-3**, Elektrotehnički fakultet.