

Лабораторная работа № 1. Простые модели компьютерной сети

Имитационное моделирование

Королёв Иван Андреевич

Содержание

1	Цель работы	5
2	Задание	6
2.1	Шаблон сценария для NS-2	6
2.2	Простой пример описания топологии сети, состоящей из двухузлов и одного соединения	6
2.3	Пример с усложнённой топологией сети	6
2.4	Пример с кольцевой топологией сети	7
2.5	Упражнение Внесите следующие изменения в реализацию примера с кольцевой топологией сети:	7
3	Теоретическое введение	9
4	Выполнение лабораторной работы	11
4.1	Задание 1. Создание шаблона сценария для NS-2	11
4.2	Задание 2. Простой пример описания топологии сети, состоящей из двух узлов и одного соединения	13
4.3	Задание 3. Пример с усложнённой топологией сети	16
4.4	Задание 4. Пример с кольцевой топологией сети	18
4.5	Упражнение для самостоятельной реализации.	20
5	Выводы	26
	Список литературы	27

Список иллюстраций

4.1	Создание директории <code>mir</code> для лабораторной работы	11
4.2	Шаблон NS-2	12
4.3	<code>out.nam</code>	13
4.4	Скопировали шаблон сценария NS-2 в файл <code>example1</code>	14
4.5	Шаблон NS-2	15
4.6	<code>out.nam</code>	16
4.7	Усложнённая топология сети	17
4.8	<code>out.nam</code>	17
4.9	<code>out.nam</code>	18
4.10	Кольцевая топология сети	19
4.11	<code>out.nam</code>	19
4.12	<code>out.nam</code>	20
4.13	<code>out.nam</code>	21
4.14	<code>out.nam</code>	22
4.15	<code>out.nam</code>	23
4.16	<code>out.nam</code>	24
4.17	<code>out.nam</code>	25

Список таблиц

1 Цель работы

Приобретение навыков моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также анализ полученных результатов моделирования.

2 Задание

2.1 Шаблон сценария для NS-2

2.2 Простой пример описания топологии сети, состоящей из двухузлов и одного соединения

Постановка задачи. Требуется смоделировать сеть передачи данных, состоящую из двух узлов, соединённых дуплексной линией связи с полосой пропускания 2 Мб/с и задержкой 10 мс, очередью с обслуживанием типа DropTail. От одного узла к другому по протоколу UDP осуществляется передача пакетов, размером 500 байт, с постоянной скоростью 200 пакетов в секунду.

2.3 Пример с усложнённой топологией сети

Постановка задачи. Описание моделируемой сети: * сеть состоит из 4 узлов (n0, n1, n2, n3); * между узлами n0 и n2, n1 и n2 установлено дуплексное соединение с пропускной способностью 2 Мбит/с и задержкой 10 мс; * между узлами n2 и n3 установлено дуплексное соединение с пропускной способностью 1,7 Мбит/с и задержкой 20 мс; * каждый узел использует очередь с дисциплиной DropTail для накопления пакетов, максимальный размер которой составляет 10; * TCP-источник на узле n0 подключается к TCP-приёмнику на узле n3 (по-умолчанию, максимальный размер пакета, который TCP-агент может генерировать, равняет-

ся 1KByte) * TCP-приёмник генерирует и отправляет ACK пакеты отправителю и откидывает полученные пакеты; * UDP-агент, который подсоединён к узлу n1, подключён к null-агенту на узле n3 (null-агент просто откидывает пакеты); * генераторы трафика ftp и cbr прикреплены к TCP и UDP агентам соответственно; * генератор cbr генерирует пакеты размером 1 Кбайт со скоростью 1 Мбит/с; * работа cbr начинается в 0,1 секунду и прекращается в 4,5 секунды, а ftp начинает работать в 1,0 секунду и прекращает в 4,0 секунды.

2.4 Пример с кольцевой топологией сети

Постановка задачи. Требуется построить модель передачи данных по сети с кольцевой топологией и динамической маршрутизацией пакетов: * сеть состоит из 7 узлов, соединённых в кольцо; * данные передаются от узла n(0) к узлу n(3) по кратчайшему пути; * с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами n(1) и n(2); * при разрыве соединения маршрут передачи данных должен измениться на резервный.

2.5 Упражнение Внесите следующие изменения в

реализацию примера с кольцевой топологией сети:

- передача данных должна осуществляться от узла n(0) до узла n(5) по кратчайшему пути в течение 5 секунд модельного времени;
- передача данных должна идти по протоколу TCP (тип Newreno), на принимающей стороне используется TCPSink-объект типа DelAck; поверх TCP работает протокол FTP с 0,5 до 4,5 секунд модельного времени;
- с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами n(0) и n(1);
- при разрыве соединения маршрут передачи данных должен измениться на резервный, после восстановления соединения пакеты снова должны пойти

по кратчайшему пути.

3 Теоретическое введение

Network Simulator (NS-2) — один из программных симуляторов моделирования процессов в компьютерных сетях. NS-2 позволяет описать топологию сети, конфигурацию источников и приёмников трафика, параметры соединений (полосу пропускания, задержку, вероятность потерь пакетов и т.д.) и множество других параметров моделируемой системы. Данные о динамике трафика, состоянии соединений и объектов сети, а также информация о работе протоколов фиксируются в генерируемом trace-файле.

NS-2 является объектно-ориентированным программным обеспечением. Его ядро реализовано на языке C++. В качестве интерпретатора используется язык скриптов (сценариев) OTcl (Object oriented Tool Command Language). NS-2 полностью поддерживает иерархию классов C++ и подобную иерархию классов интерпретатора OTcl. Обе иерархии обладают идентичной структурой, т.е. существует однозначное соответствие между классом одной иерархии и таким же классом другой. Объединение для совместного функционирования C++ и OTcl производится при помощи TclCl (Classes Tcl). В случае, если необходимо реализовать какую-либо специфическую функцию, не реализованную в NS-2 на уровне ядра, для этого используется код на C++

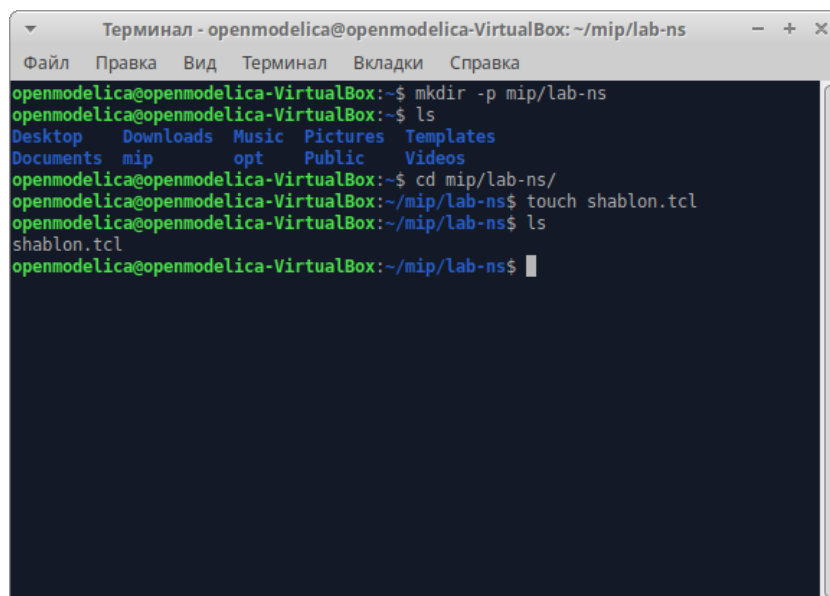
Процесс создания модели сети для NS-2 состоит из нескольких этапов: 1. создание нового объекта класса Simulator, в котором содержатся методы, необходимые для дальнейшего описания модели (например, методы new и delete используются для создания и уничтожения объектов соответственно); 2. описание топологии моделируемой сети с помощью трёх основных функциональных блоков: узлов

(nodes), соединений (links) и агентов (agents); 3. задание различных действий, характеризующих работу сети.

4 Выполнение лабораторной работы

4.1 Задание 1. Создание шаблона сценария для NS-2

Создал директорию `mip/lab-ns` для выполнения лабораторной работы. Первый файл `shablon.tcl` будет содержать шаблон сценария для NS-2 (рис. 4.1).



```
Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/lab-ns
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@openmodelica-VirtualBox:~$ mkdir -p mip/lab-ns
openmodelica@openmodelica-VirtualBox:~$ ls
Desktop  Downloads  Music  Pictures  Templates
Documents  mip      opt    Public   Videos
openmodelica@openmodelica-VirtualBox:~$ cd mip/lab-ns/
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ touch shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ls
shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$
```

Рис. 4.1: Создание директории `mip` для лабораторной работы

Написанные код шаблона сценария для NS-2 (рис. 4.2).

```

/home/openmodelica/mip/lab-ns/shablon.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка

# создание объекта Simulator
set ns [new Simulator]

# открытие на запись файла out.pam для визуализатора pam
set nf [open out.pam w]

# все результаты моделирования будут записаны в переменную nf
$ns namtrace-all $nf

# открытие на запись файла трассировки out.tr
# для регистрации всех событий
set f [open out.tr w]

# все регистрируемые события будут записаны в переменную f
$ns trace-all $f

# процедура finish закрывает файлы трассировки
# и запускает визуализатор pam
proc finish {} {
    global ns f nf # описание глобальных переменных
    $ns flush-trace # прекращение трассировки
    close $f # закрытие файлов трассировки
    close $nf # закрытие файлов трассировки pam
    # запуск pam в фоновом режиме
    exec pam out.pam &
    exit 0
}

# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"

# запуск модели
$ns run

```

Рис. 4.2: Шаблон NS-2

Визуальное отображение работающей программы pam. В данном этапе никакого визуального отображения нет, т.к. нет прописанных протоколов передачи данных, агента для генерации и приёма трафика и at-событий. (рис. 4.3).

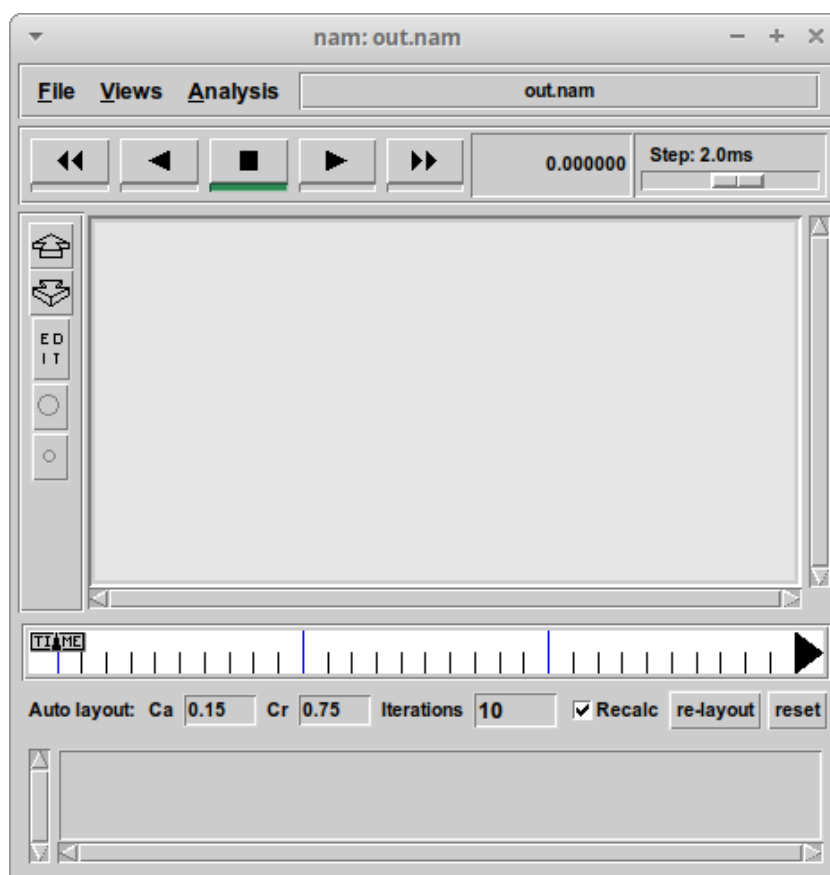
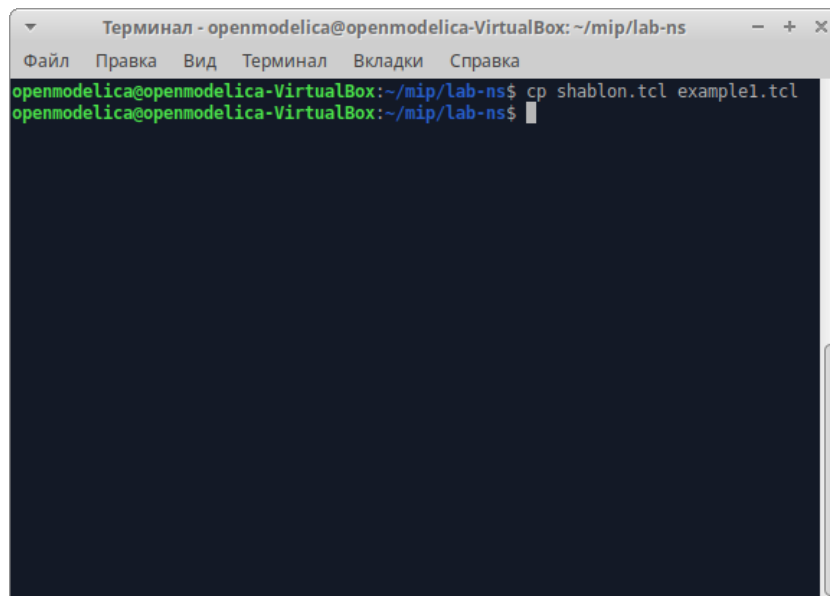


Рис. 4.3: out.nam

4.2 Задание 2. Простой пример описания топологии сети, состоящей из двух узлов и одного соединения

Скопировали написанный в предыдущем задании шаблон NS-2 в файл `example1.tcl`. На основе данного шаблона будем моделировать сеть передачи данных. (рис. 4.4).

A screenshot of a terminal window titled "Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/lab-ns". The window has a menu bar with "Файл", "Правка", "Вид", "Терминал", "Вкладки", and "Справка". The terminal shows two lines of text: "openmodelica@openmodelica-VirtualBox:~/mip/lab-ns\$ cp shablon.tcl example1.tcl" and "openmodelica@openmodelica-VirtualBox:~/mip/lab-ns\$". The prompt is green, and the command and its output are white on a dark background. A vertical scrollbar is visible on the right side of the terminal window.

```
Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/lab-ns
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ cp shablon.tcl example1.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$
```

Рис. 4.4: Скопировали шаблон сценария NS-2 в файл example1

Реализация модели. Добавил 2 узла, соединил узлы дуплексным соединением с полосой пропускания 2 Мб/с и задержкой 10 мс, очередью с обслуживанием типа DropTail. Написал агента для приёма и генерации трафика. Добавил at-события. (рис. 4.5).

```

/home/openmodelica/mip/lab-ns/example1.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка
exit 0
}

# создание 2-х узлов:
set N 2
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

# соединение 2-х узлов дуплексным соединением
# с полосой пропускания 2 Mb/c и задержкой 10 мс,
# очередь с обслуживанием типа DropTail(дисциплина обслуживания)
$ns duplex-link $n(0) $n(1) 2Mb 10ms DropTail

# создание агента UDP и присоединение его к узлу n0
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0

# создание источника трафика CBR (constant bit rate)
set cbr0 [new Application/Traffic/CBR]

# устанавливаем размер пакета в 500 байт
$cbr0 set packetSize_ 500
# задаем интервал между пакетами равным 0.005 секунды,
# т.е. 200 пакетов в секунду
$cbr0 set interval_ 0.005

# присоединение источника трафика CBR к агенту udp0
$cbr0 attach-agent $udp0

# Создание агента-приёмника и присоединение его к узлу n(1)
set null0 [new Agent/Null]
$ns attach-agent $n(1) $null0

# Соединение агентов между собой
$ns connect $udp0 $null0

# запуск приложения через 0,5 с
$ns at 0.5 "$cbr0 start"

# остановка приложения через 4,5 с
$ns at 4.5 "$cbr0 stop"

# at-событие для планировщика событий, которое запускает
# процедуры finish через 5 с после начала моделирования
$ns at 5.0 "finish"

# запуск модели
$ns run

```

Рис. 4.5: Шаблон NS-2

Результат добавления описания топологии сети. Видим, что через 0.5 секунд из узла 0 данные поступают к узлу 1. Поступление остановится через 4.5 секунды. (рис. 4.6).

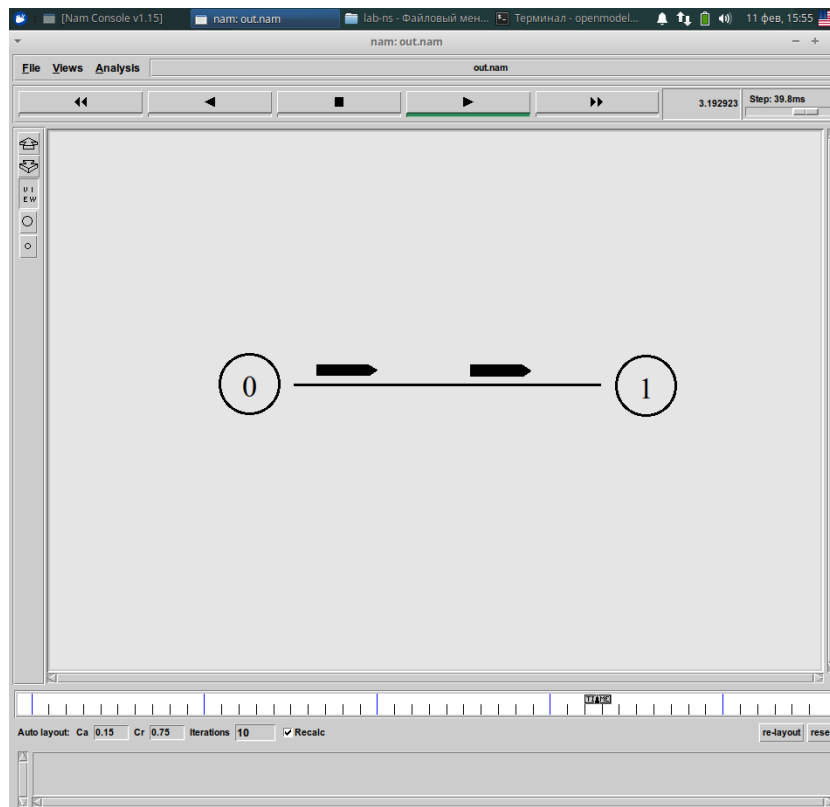


Рис. 4.6: out.nam

4.3 Задание 3. Пример с усложнённой топологией сети

Скопировали написанный в предыдущем задании шаблон NS-2 в файл `example2.tcl`. На основе данного шаблона добавил описание моделируемой сети из 4 узлов. (рис. 4.7).


```

ns duplex-link-op $n(0) $n(2) orient right-down
ns duplex-link-op $n(1) $n(2) orient right-up
ns duplex-link-op $n(2) $n(3) orient right

# создание агента UDP и присоединение его к узлу n(0)
set udp0 [new Agent/UDP]
ns attach-agent $n(0) $udp0

# создание источника CBR-трафика
# и присоединение его к агенту udp0
set cbr0 [new Application/Traffic/CBR]
$udp0 set packetSize_ $m
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0

# создание агента TCP и присоединение его к узлу n(1)
set tcp1 [new Agent/TCP]
ns attach-agent $n(1) $tcp1

# создание приложения FTP
# и присоединение его к агенту tcp1
set ftp [new Application/FTP]
$tcp1 attach-agent $ftp

# создание агента-получателя для udp0
set null0 [new Agent/Null]
ns attach-agent $n(2) $null0

# создание агента-получателя для tcp1
set sink1 [new Agent/TCPSink]
ns attach-agent $n(3) $sink1

ns connect $udp0 $null0
ns connect $tcp1 $sink1

ns color 1 Blue
ns color 2 Red

$udp0 set class_ 1
$tcp1 set class_ 2

ns duplex-link-op $n(2) $n(3) queuePos 0.5
ns queue-limit $n(2) $n(3) 20

ns at 0.5 "cbr0 start"
ns at 1.5 "tcp1 start"
ns at 4.0 "ftp stop"
ns at 4.5 "cbr0 stop"

```

Рис. 4.7: Усложнённая топология сети

На данных изображениях отображена визуальная работа усложнённой топологии сети. На рисунке 9 видим, что от узла 0 к узлу 2, от узла 1 к узлу 2 передаётся трафик, а от узла 2 передается трафик к узлу 3. Соединение 2 и 3 имеет полосу 1Мб, а от каждого узла передается по 200 пакетов. Соответственно, пакеты должны теряться. Так же, мы видим, как накапливается очередь. У нас наложены ограничения на размер очереди, поэтому она сбрасывается при её достижении. (рис. 4.8), (рис. 4.9)

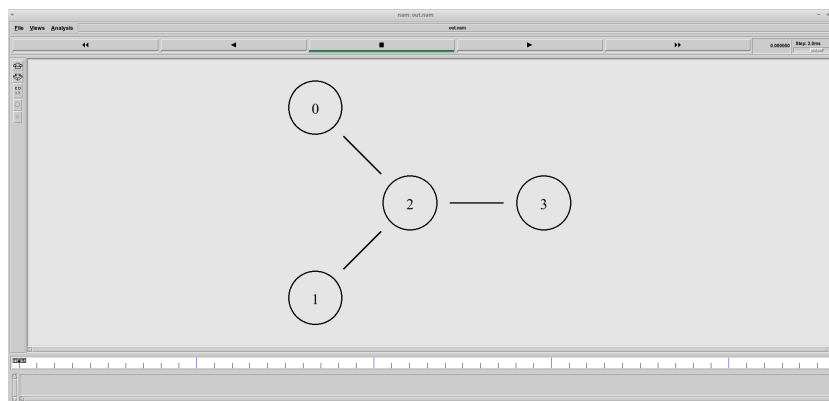


Рис. 4.8: out.nam

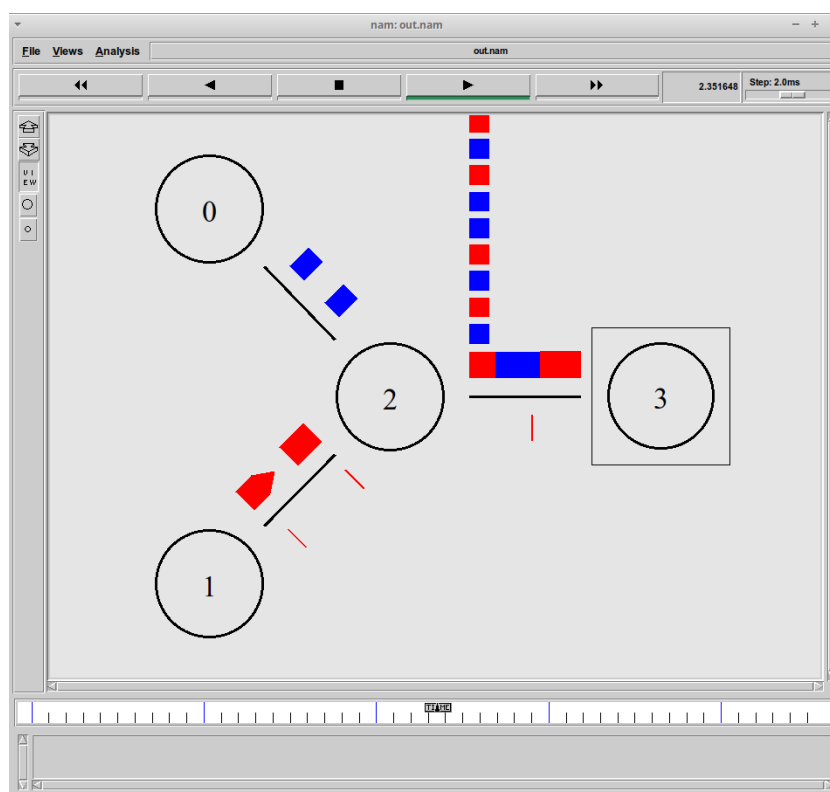


Рис. 4.9: out.nam

4.4 Задание 4. Пример с кольцевой топологией сети

Скопировали написанный в предыдущем задании шаблон NS-2 в файл `example3.tcl`. На основе данного шаблона добавил описание моделируемой сети из 7 узлов. (рис. 4.10).

```

ns
# в начале на запись файла трассировки out.tr
# для разрыва связи события
set f [open out.tr w]

# все регистрируемые события будут записаны в переменную f
$ns trace all $f

# процедура finish завершает файл трассировки
# и запускает интерпретатор ns
proc finish {} {
    global ns f nf # описание глобальных переменных
    $ns flush-trace # опорожнение трассировки
    close $f # закрываем файл трассировки
    close $nf # закрываем файл трассировки ns
    # выводим что в файле нечего
    exec cat out.tram &
    exit 0
}

set n 7
for {set i 0} {$i < $n} {incr i} {
    set n($i) [ns node]
}

for {set i 0} {$i < $n} {incr i} {
    $ns duplex-link $n($i) $n($i+1) 1000Mbps 10ms DropTail
}

set ns0 [new Agent/UDP]
$ns attach-agent $n(0) $ns0
set ns1 [new Agent/UDP]
$ns attach-agent $n(1) $ns1
$ns0 set packetSize_ 500
$ns1 set packetSize_ 500
$ns0 set interval_ 0.005
$ns1 set interval_ 0.005

set ns10 [new Agent/HTTP]
$ns attach-agent $n(3) $ns10
$ns connect $ns0 $ns10

$ns at 0.5 "$ns0 start"
$ns rmodel-at 1.0 down $n(1) $n(2)
$ns rmodel-at 2.0 up $n(1) $n(2)
$ns at 4.5 "$ns0 stop"

# at-события для планирования событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "$ns finish"

```

Рис. 4.10: Кольцевая топология сети

В at-событии прописано событие на разрыва соединения между узлами n(1) и n(2) на время в одну секунду. Во время разрыва пакеты не доходят до узла 3. (рис. 4.11)

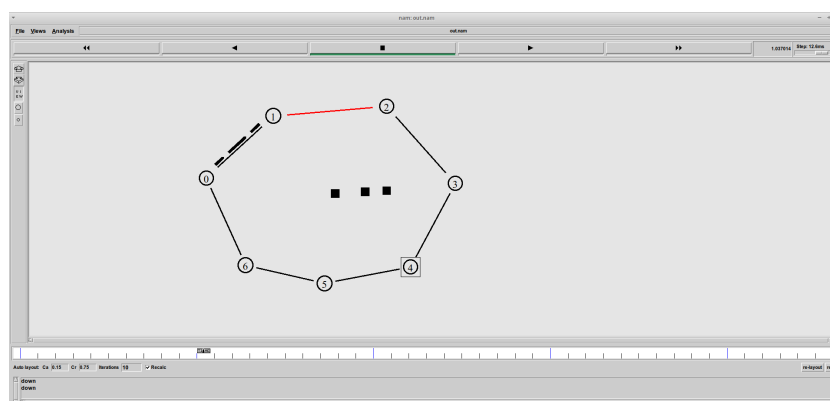


Рис. 4.11: out.tram

Чтобы пакеты доходили до конечного узла при разрыве, необходимо в начале программы, а после команды создания объекта Simulator добавить \$ns rtproto DV (рис. 4.12)

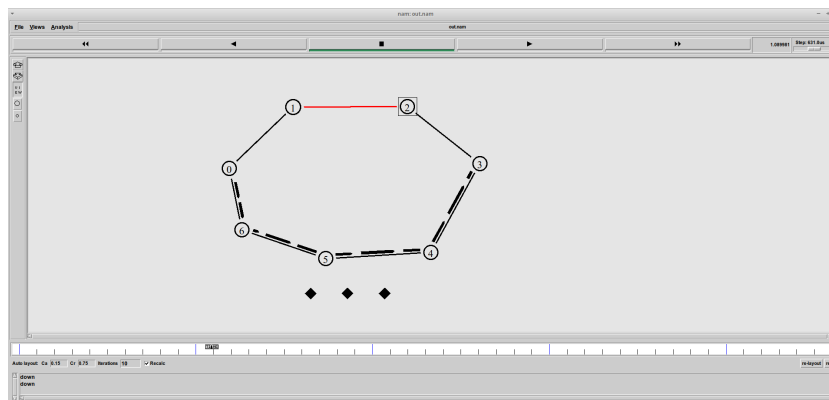


Рис. 4.12: out.nam

4.5 Упражнение для самостоятельной реализации.

Все пункты упражнения выполнены. Результаты представлены в скриншотах. Передача пакетов идет по кратчайшему пути от узла 0 к узлу 5. При разрыве соединения между узлами 0 и 1, строится другой путь до узла 5. Когда разрыв прекращается, передача пакетов дальше идет по кратчайшему. (рис. 4.13), (рис. 4.14), (рис. 4.15), (рис. 4.16)

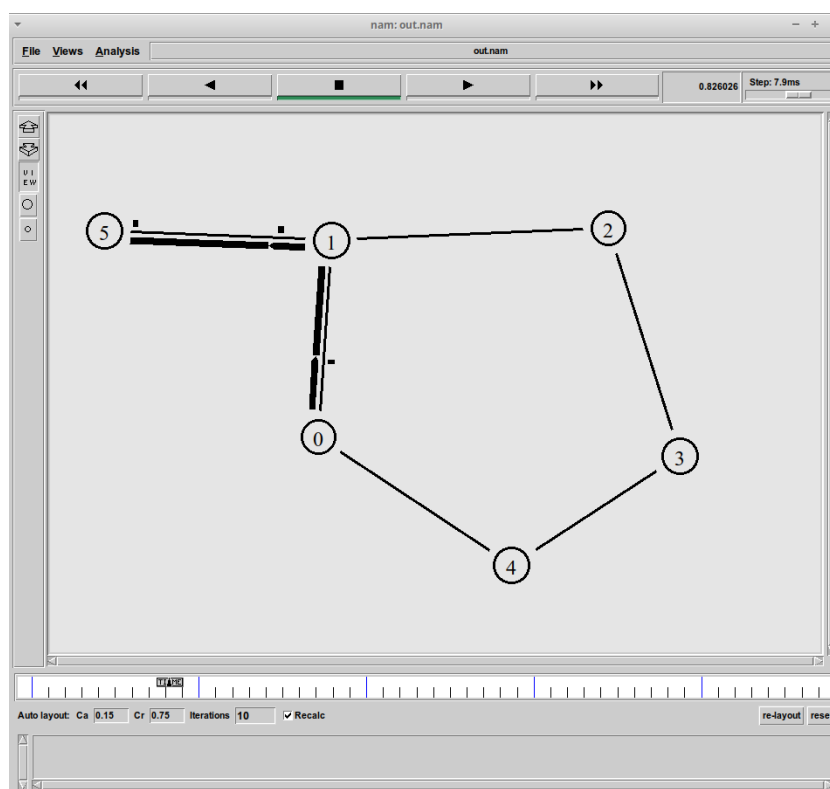


Рис. 4.13: out.nam

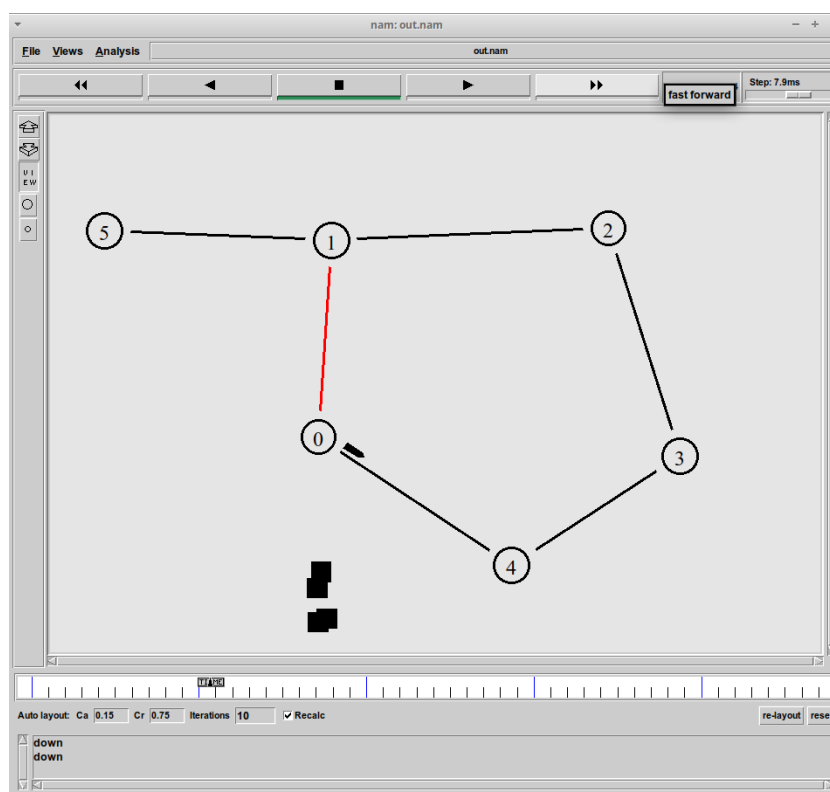


Рис. 4.14: out.nam

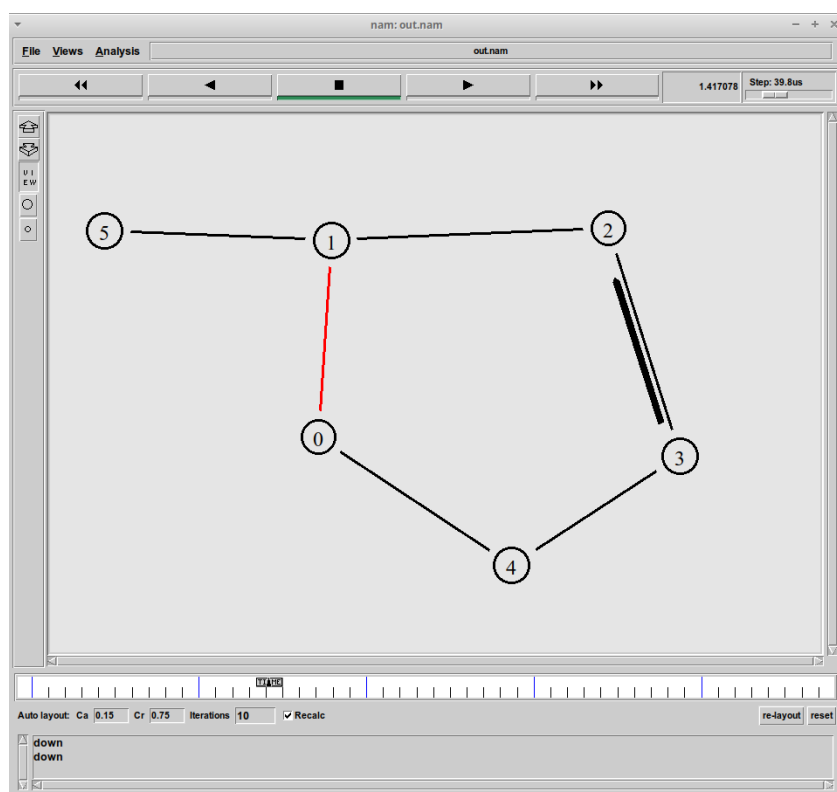


Рис. 4.15: out.nam

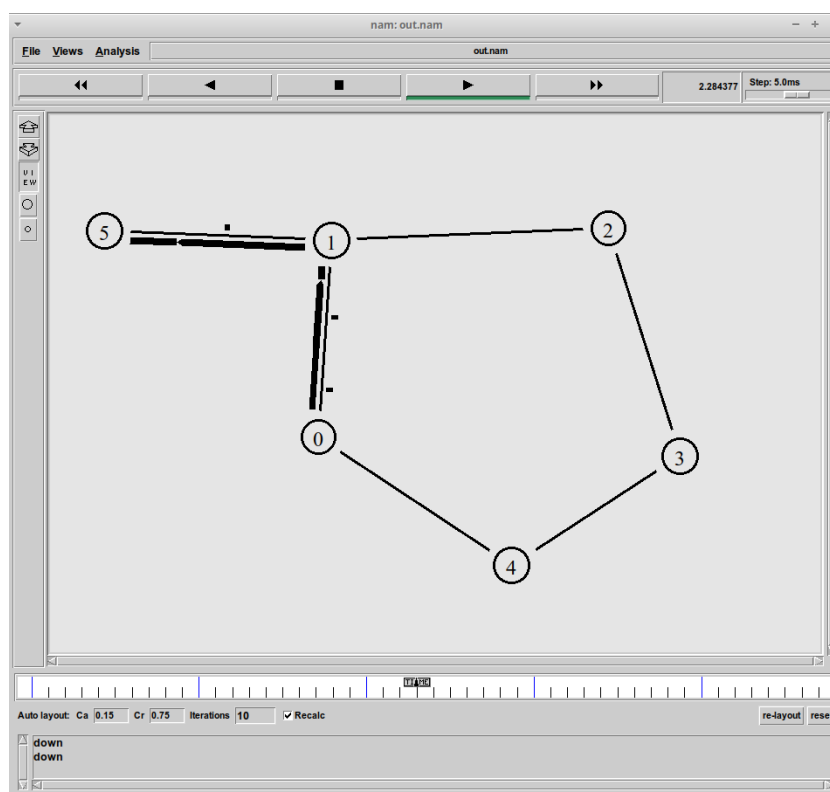


Рис. 4.16: out.nam

Код реализации. (рис. 4.17)


```

/home/openmodelica/mip/lab-ns/z.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка
# все регистрируемые события будут записаны в переменную t
$ns trace-all $f

# процедура finish закрывает файлы трассировки
# и запускает визуализатор nam
proc finish {} {
    global ns f nf # описание глобальных переменных
    $ns flush-trace # прекращение трассировки
    close $f # закрытие файлов трассировки
    close $nf # закрытие файлов трассировки nam
    # запуск nam в фоновом режиме
    exec nam out.nam &
    exit 0
}

set N 5
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

for {set i 0} {$i < $N} {incr i} {
    $ns duplex-link $n($i) $n([expr {$i+1}$N]) 1Mb 10ms DropTail
}

set n5 [$ns node]
$ns duplex-link $n5 $n(1) 1Mb 10ms DropTail

set tcp1 [new Agent/TCP/Newreno]
$ns attach-agent $n(0) $tcp1

set ftp [new Application/FTP]
$ftp attach-agent $tcp1

set sink1 [new Agent/TCPSink]
$ns attach-agent $n5 $sink1

$ns connect $tcp1 $sink1

$ns at 0.5 "$ftp start"
$ns rtmodel-at 1.0 down $n(0) $n(1)
$ns rtmodel-at 2.0 up $n(0) $n(1)
$ns at 4.5 "$ftp stop"

# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"

# запуск модели
$ns run

```

Рис. 4.17: out.nam

5 Выводы

Приобрел навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2.

Список литературы