

Лабораторная работа 1. Простые модели компьютерной сети

Ильин Никита Евгеньевич

Содержание

| | | |
|---|--------------|----|
| 1 | Цель работы | 5 |
| 2 | Ход работы | 6 |
| 3 | Выводы | 17 |
| 4 | Библиография | 18 |

List of Figures

| | | |
|------|---|----|
| 2.1 | Рис 1. Создание директории | 6 |
| 2.2 | Рис 2. Код программы для шаблона | 7 |
| 2.3 | Рис 2. Код программы для моделирования простой сети | 8 |
| 2.4 | Рис 3. Моделирование простой сети из двух узлов | 9 |
| 2.5 | Рис 4. Код программы для усложненной сети из четырех узлов(1) . | 10 |
| 2.6 | Рис 5. Код программы для усложненной сети из четырех узлов(2) . | 11 |
| 2.7 | Рис 6. Моделирование усложненной сети | 12 |
| 2.8 | Рис 7. Код программы для кольцевой сети | 13 |
| 2.9 | Рис 7. Моделирование простой сети из двух узлов | 14 |
| 2.10 | Рис 8. Код программы для упражнения | 15 |
| 2.11 | Рис 9. Результат моделирования для упражнения | 16 |

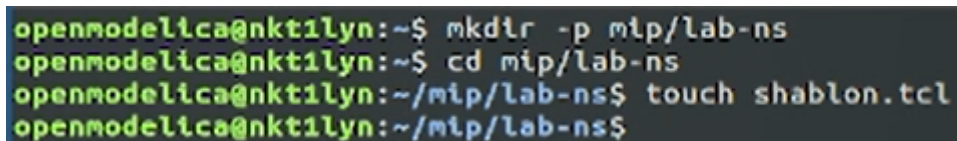
List of Tables

1 Цель работы

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

2 Ход работы

1. В своем рабочем каталоге создаем директорию `mip`, в которой будут выполняться лабораторные работы. Внутри нее создаем директорию `lab-ns`, а в ней файл `shablon.tcl`.



```
openmodelica@nkt1lyn:~$ mkdir -p mip/lab-ns
openmodelica@nkt1lyn:~$ cd mip/lab-ns
openmodelica@nkt1lyn:~/mip/lab-ns$ touch shablon.tcl
openmodelica@nkt1lyn:~/mip/lab-ns$
```

Figure 2.1: Рис 1. Создание директории

2. Пишем программу в файл `shablon.tcl`.

```

# создание объекта Simulator
set ns [new Simulator]
# открытие на запись файла out.nam для визуализатора nam
set nf [open out.nam 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 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
}
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"
# запуск модели
$ns run

```

Figure 2.2: Рис 2. Код программы для шаблона

3. Пишем программу для моделирования простой сети в файл example1.tcl.

```

# 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 pamtrace-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 $nf # закрытие файлов трассировки pam
    # запуск pam в фоновом режиме
    exec pam out.pam &
    exit 0
}
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
# создание 2-х узлов:
set N 2
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}
# соединение 2-х узлов дуплексным соединением
# с полосой пропускания 2 Мб/с и задержкой 10 мс,
# очередью с обслуживанием типа DropTail
$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"

$ns at 5.0 "finish"
# запуск модели
$ns run

```

Figure 2.3: Рис 2. Код программы для моделирования простой сети

4. Запускаем моделирование с помощью команды `nam out.nam`.

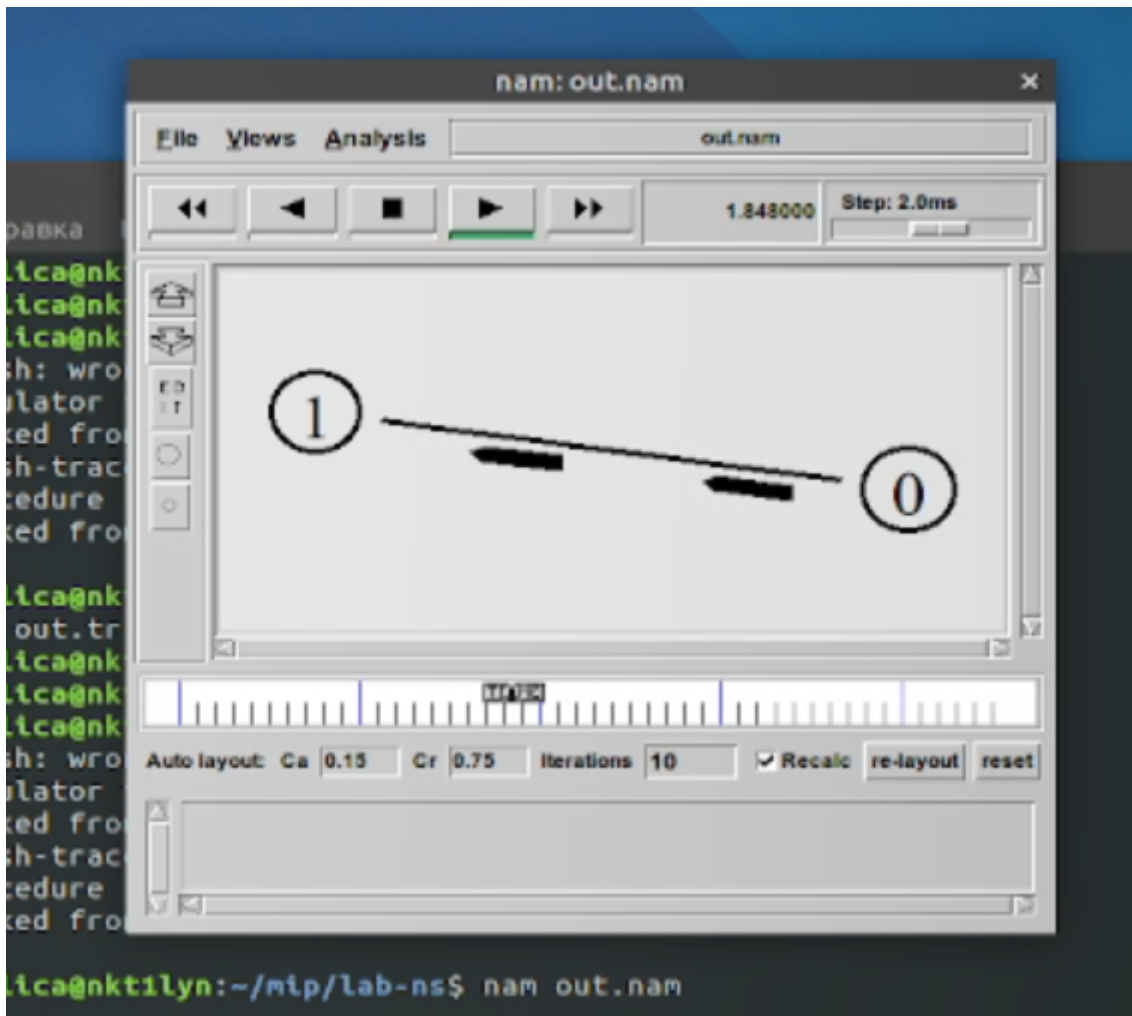


Figure 2.4: Рис 3. Моделирование простой сети из двух узлов

5. Пишем программу для усложненной сети из четырех узлов в файл `example2.tcl`.

```

# создание объекта Simulator
set ns [new Simulator]
# открытие на запись файла out.nam для визуализатора nam
set nf [open out.nam 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 nam
proc finish {} {
    global ns f nf # описание глобальных переменных
    $ns flush-trace # прекращение трассировки
    close $f # закрытие файлов трассировки
    close $nf # закрытие файлов трассировки nam
    # запуск nam в фоновом режиме
    exec nam out.nam &
    exit 0
}
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
set N 4
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}
$ns duplex-link $n(0) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(1) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(3) $n(2) 2Mb 10ms DropTail
$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]
$cbr0 set packetSize_ 500
$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]
$ftp attach-agent $tcp1
# создание агента-получателя для udp0
set null0 [new Agent/Null]
$ns attach-agent $n(3) $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

```

Figure 2.5: Рис 4. Код программы для усложненной сети из четырех узлов(1)

```
$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"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
# запуск модели
$ns run
```

Figure 2.6: Рис 5. Код программы для усложненной сети из четырех узлов(2)

6. Запускаем моделирование с помощью команды `nam out.nam`


```

# создание объекта Simulator
set ns [new Simulator]
$ns rtproto DV
# открытие на запись файла out.nam для визуализатора nam
set nf [open out.nam 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 nam
proc finish {} {
    global ns f nf # описание глобальных переменных
    $ns flush-trace # прекращение трассировки
    close $f # закрытие файлов трассировки
    close $nf # закрытие файлов трассировки nam
    # запуск nam в фоновом режиме
    exec nam out.nam &
    exit 0
}
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
set N 7
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}
for {set i 0} {$i < $N} {incr i} {
    $ns duplex-link $n($i) $n([expr ($i+1)%$N]) 1Mb 10ms DropTail
}
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Agent/CBR]
$ns attach-agent $n(0) $cbr0
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
set null0 [new Agent/Null]
$ns attach-agent $n(3) $null0
$ns connect $cbr0 $null0
$ns at 0.5 "$cbr0 start"
$ns rtmodel-at 1.0 down $n(1) $n(2)
$ns rtmodel-at 2.0 up $n(1) $n(2)
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns at 5.0 "finish"
# запуск модели
$ns run

```

Figure 2.8: Рис 7. Код программы для кольцевой сети

8. Запускаем моделирование с помощью команды `nam out.nam`

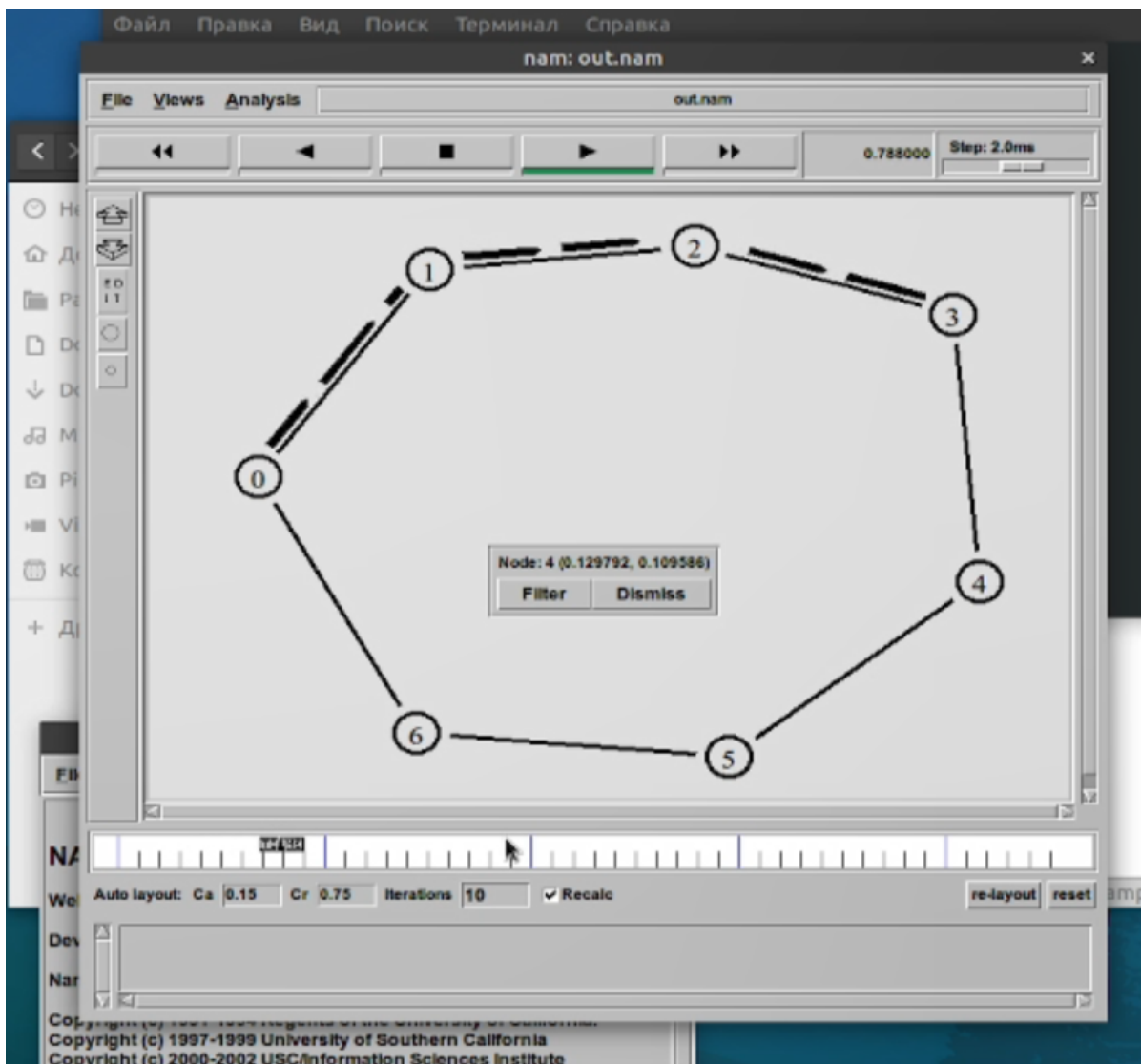


Figure 2.9: Рис 7. Моделирование простой сети из двух узлов

9. Выполняем упражнение, записывая код в файл `example4.tcl`.

```

# создание объекта Simulator
set ns [new Simulator]
$ns rtproto DV
# открытие на запись файла out.nam для визуализатора nam
set nf [open out.nam 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 nam
proc finish {} {
global ns f nf # описание глобальных переменных
$ns flush-trace # прекращение трассировки
close $f # закрытие файлов трассировки
close $nf # закрытие файлов трассировки nam
# запуск nam в фоновом режиме
exec nam out.nam &
exit 0
}
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
set N 5
for {set i 0} {$i < $N} {incr i} {
set n($i) [$ns node]
}
set n(5) [$ns node]
for {set i 0} {$i < $N} {incr i} {
$ns duplex-link $n($i) $n([expr ($i+1)%$N]) 1Mb 10ms DropTail
}
$ns duplex-link $n(1) $n(5) 1Mb 10ms DropTail

set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Agent/CBR]
$ns attach-agent $n(0) $cbr0
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
set null0 [new Agent/Null]
$ns attach-agent $n(5) $null0
$ns connect $cbr0 $null0
$ns at 0.5 "$cbr0 start"
$ns rtmodel-at 1.0 down $n(0) $n(1)
$ns rtmodel-at 2.0 up $n(0) $n(1)
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns at 5.0 "finish"
# запуск модели
$ns run

```

Figure 2.10: Рис 8. Код программы для упражнения

9. Запускаем моделирование с помощью команды `nam out.nam`

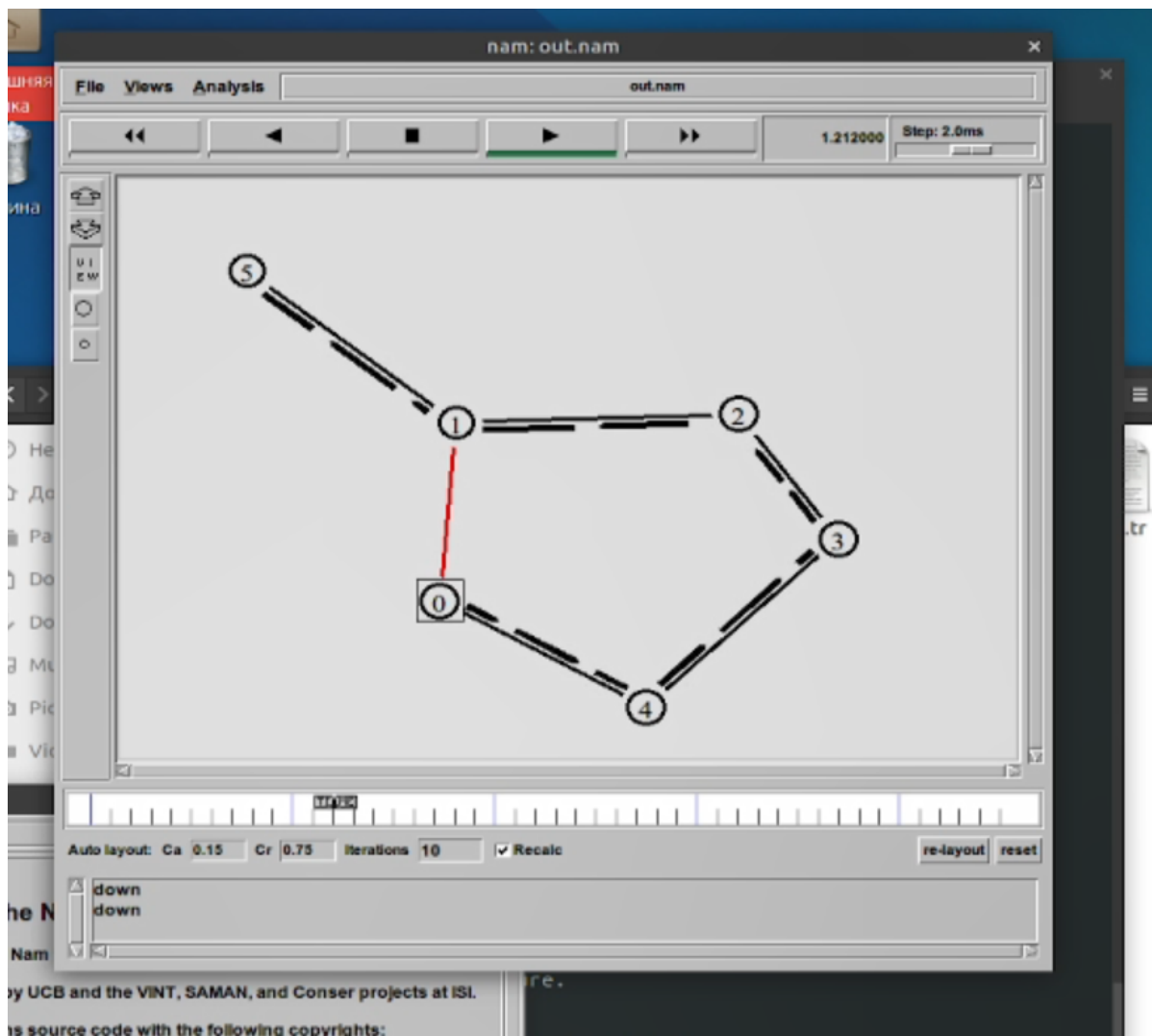


Figure 2.11: Рис 9. Результат моделирования для упражнения

3 Выводы

Получены навыки моделирования сетей передачи данных в NS-2.

4 Библиография

1. Методические материалы курса