

Моделирование информационных процессов

Отчёт по лабораторной работе №1

Аминов Зулфикор Мирзокаримович

Содержание

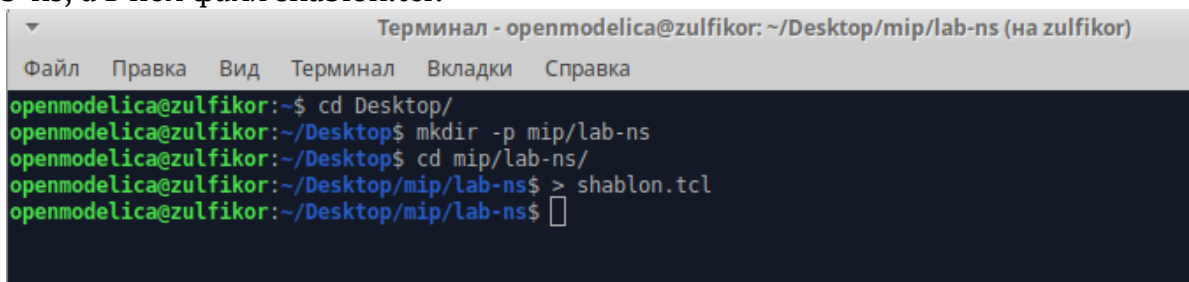
1. Цель работы	3
2. Шаблон сценария для NS-2	4
3. Простой пример описания топологии сети, состоящей из двух узлов и одного соединения	7
4. Пример с усложнённой топологией сети	10
5. Пример с кольцевой топологией сети	14
6. Упражнение	19
7. Выводы	26

1. Цель работы

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

2. Шаблон сценария для NS-2

В своём рабочем каталоге создали директорию `mip`, внутри `mip` директорию `lab-ns`, а в ней файл `shablon.tcl`:



```
Терминал - openmodelica@zulfikor: ~/Desktop/mip/lab-ns (на zulfikor)
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@zulfikor:~$ cd Desktop/
openmodelica@zulfikor:~/Desktop$ mkdir -p mip/lab-ns
openmodelica@zulfikor:~/Desktop$ cd mip/lab-ns/
openmodelica@zulfikor:~/Desktop/mip/lab-ns$ > shablon.tcl
openmodelica@zulfikor:~/Desktop/mip/lab-ns$
```

Открыли на редактирование файл `shablon.tcl` и записали код:

```

/home/openmodelica/Desktop/lab1/mip/lab-ns/shablon.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка

# 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

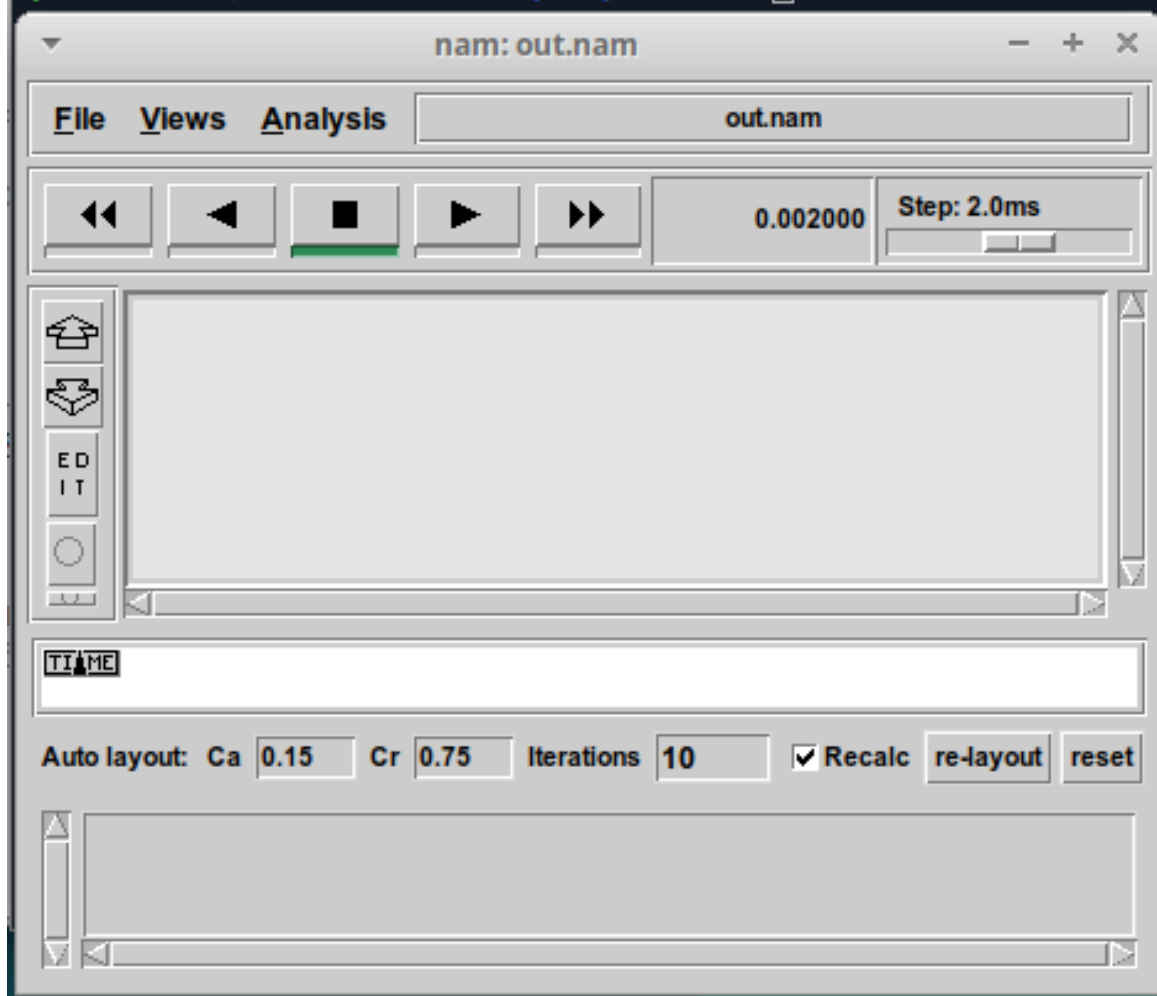
# процедура finish закрывает файлы трассировки
# и запускает визуализатор nam
proc finish {} {
    global ns f nf
    # описание глобальных переменных
    $ns flush-trace
    # прекращение трассировки
    close $f
    # закрытие файлов трассировки
    close $nf
    # закрытие файлов трассировки nam
    # 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

```

Сохранили изменения в файле shablon.tcl и запустили симулятор:

```
openmodelica@zulfikor:~/Desktop/mip/lab-ns$ ns shablon.tcl
openmodelica@zulfikor:~/Desktop/mip/lab-ns$
```



3. Простой пример описания топологии сети, состоящей из двух узлов и одного соединения

Скопировали содержимое созданного шаблона в новый файл:

```
openmodelica@zulfikor:~/Desktop/mip/lab-ns$ cp shablon.tcl example1.tcl  
openmodelica@zulfikor:~/Desktop/mip/lab-ns$
```

Откроем example.tcl на редактирование и добавим изменение:

```

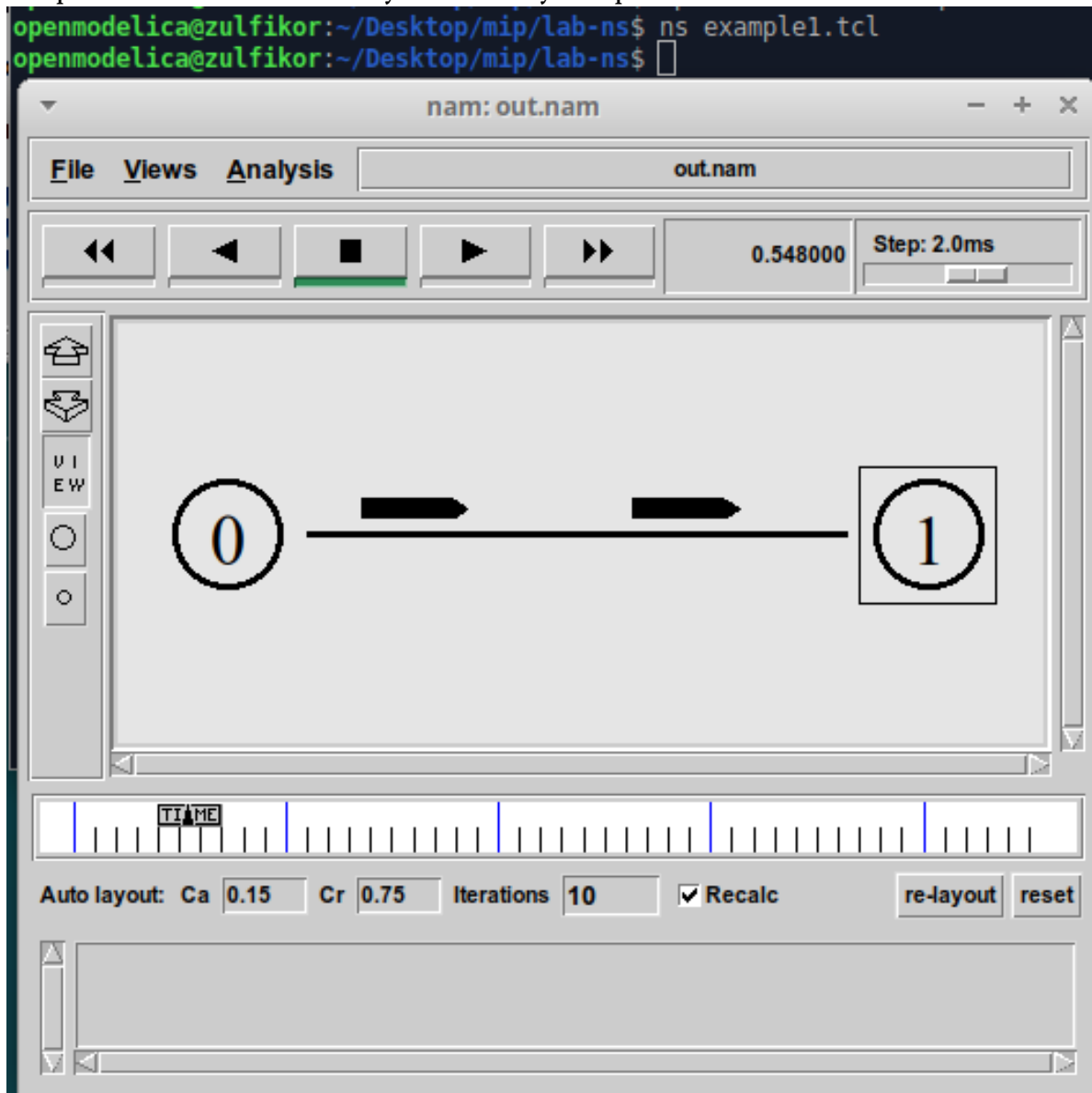
* /home/openmodelica/Desktop/lab1/mip/lab-ns/example1.tcl - Mousepad (на zulfikor)
Файл  Правка  Поиск  Вид  Документ  Справка

# создание объекта Simulator
set ns [new Simulator]
# открытие на запись файла out.nam для визуализатора nam
set nf [open out.nam w]
$ns namtrace-all $nf
set f [open out.tr w]
$ns trace-all $f
proc finish {} {
    global ns f nf
    # описание глобальных переменных
    $ns flush-trace
    # прекращение трассировки
    close $f
    # закрытие файлов трассировки
    close $nf
    # закрытие файлов трассировки nam
    # запуск nam в фоновом режиме
    exec nam out.nam &
    exit 0
}
# создание 2-х узлов:
set N 2
for {set i 0} {$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,
# очередь 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"

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

```


Сохраним изменения и запускаем симулятор:



Получили в качестве результата запуск аниматора nam в фоновом режиме.

4. Пример с усложнённой топологией сети

Скопируем содержимое созданного шаблона в example2.tcl.

```
openmodelica@zulfikor:~/Desktop/mip/lab-ns$ cp shablon.tcl example2.tcl  
openmodelica@zulfikor:~/Desktop/mip/lab-ns$
```

Откроем example.tcl на редактирование и внесем изменения.

```
/home/openmodelica/Desktop/lab1/mip/lab-ns/example2.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка

# 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
}

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
$ns color 2 Red
$udp0 set class_ 1
$tcp1 set class_ 2

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

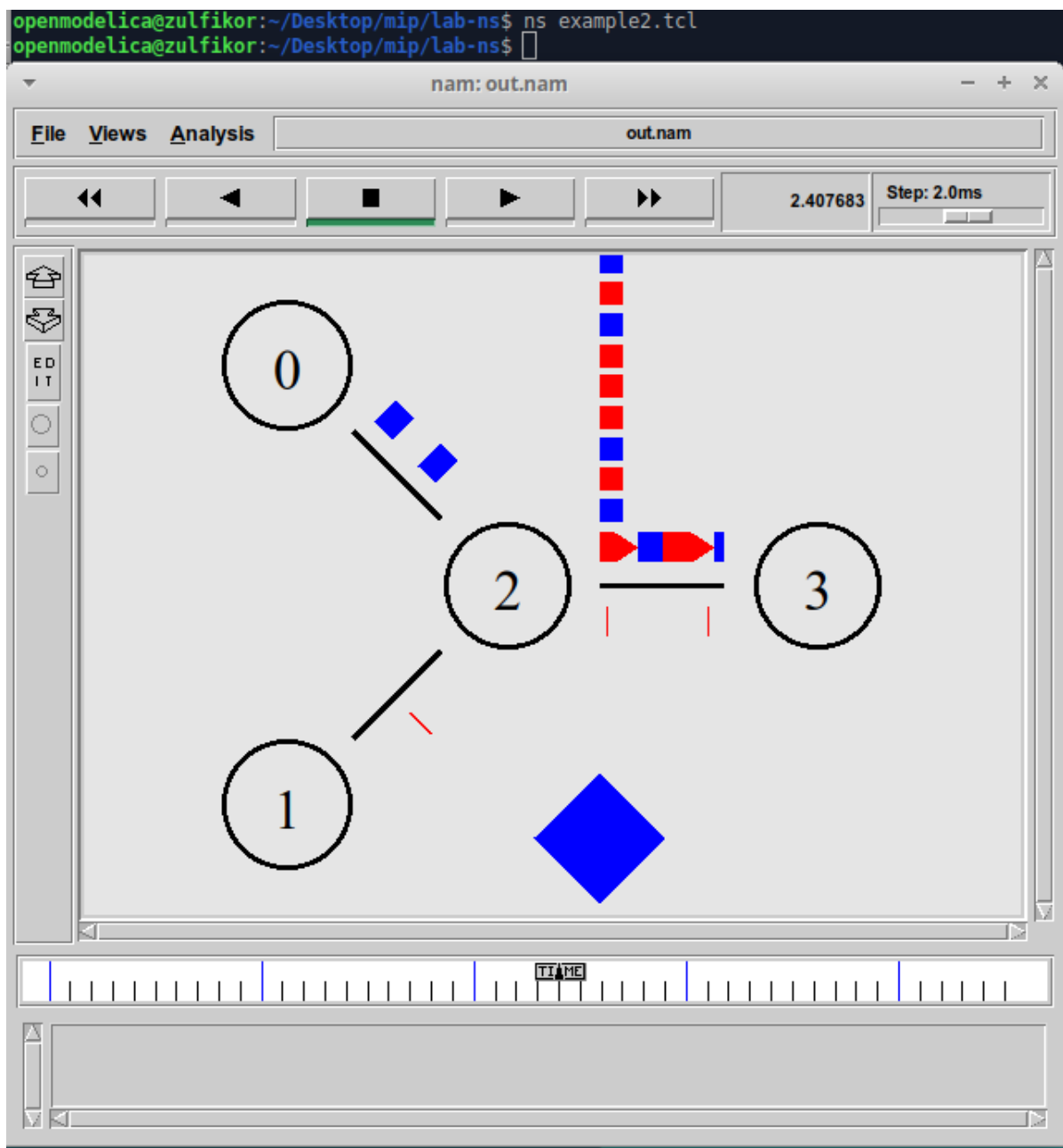
$ns queue-limit $n(2) $n(3) 20

$ns at 0.5 "$cbr0 start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr0 stop"

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

```

Сохранив изменения в файле запустим симулятор



Получили анимированный результат моделирования.

5. Пример с кольцевой топологией сети

Скопируем содержимое созданного шаблона в новый файл:

```
openmodelica@zulfikor:~/Desktop/mip/lab-ns$ cp shablon.tcl example3.tcl  
openmodelica@zulfikor:~/Desktop/mip/lab-ns$
```

Откроем example.tcl на редактирование.

```

* /home/openmodelica/Desktop/lab1/mip/lab-ns/example3.tcl - Mousepad (на zulfikor)
Файл  Правка  Поиск  Вид  Документ  Справка

# 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

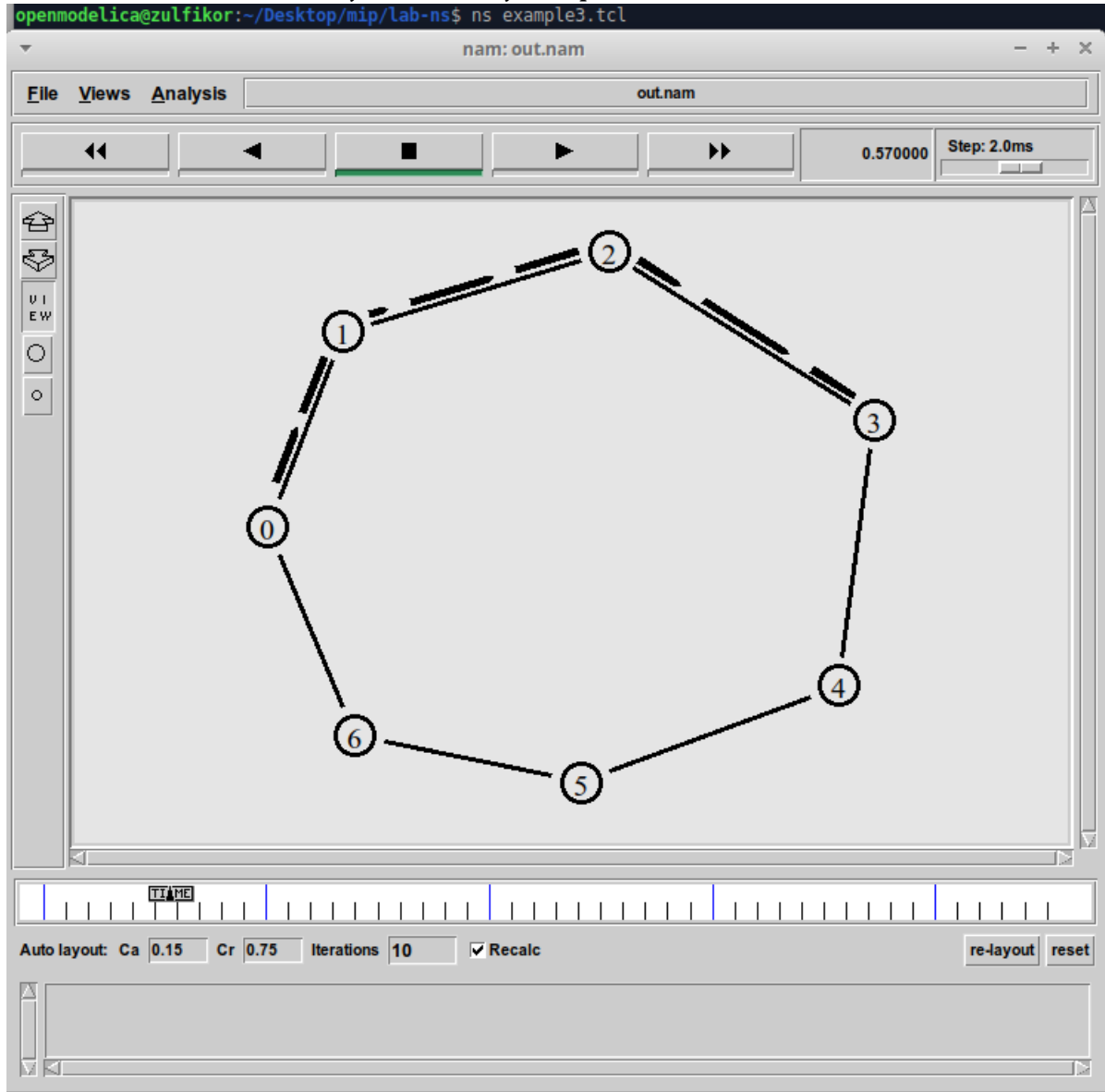
# открытие на запись файла трассировки out.tr
# для регистрации всех событий
set f [open out.tr w]
# все регистрируемые события будут записаны в переменную f
$ns trace-all $f
# процедура finish закрывает файлы трассировки
# и запускает визуализатор nam
proc finish {} {
    global ns f nf
    # описание 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
}

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
}

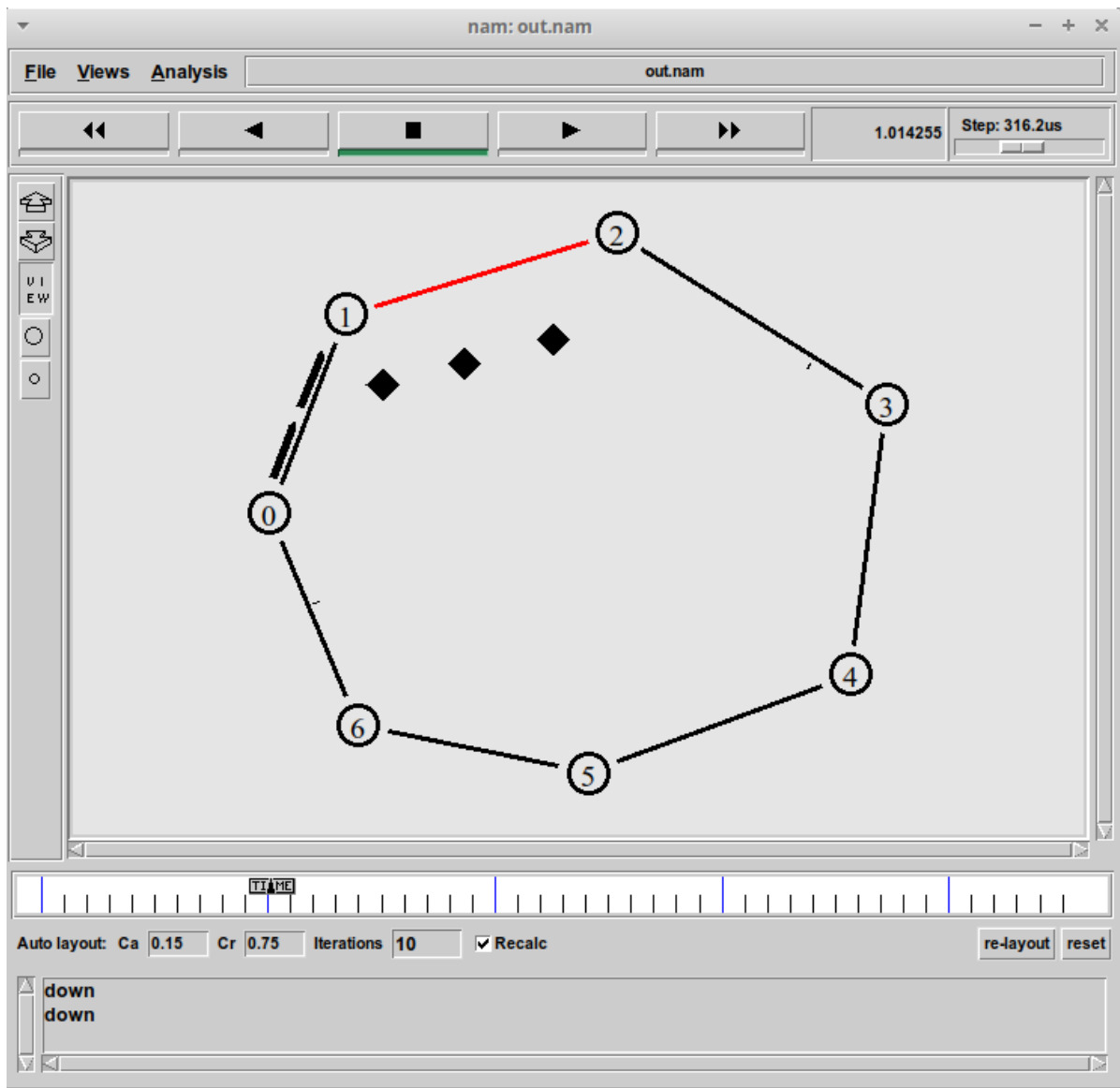
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Agent/CBR]
$ns attach-agent $n(0) $cbr0
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
set null0 [new Agent/Null]
$ns attach-agent $n(3) $null0
$ns connect $cbr0 $null0
$ns at 0.5 "$cbr0 start"
$ns rtmodel-at 1.0 down $n(1) $n(2)
$ns rtmodel-at 2.0 up $n(1) $n(2)
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"
# запуск модели
$ns run

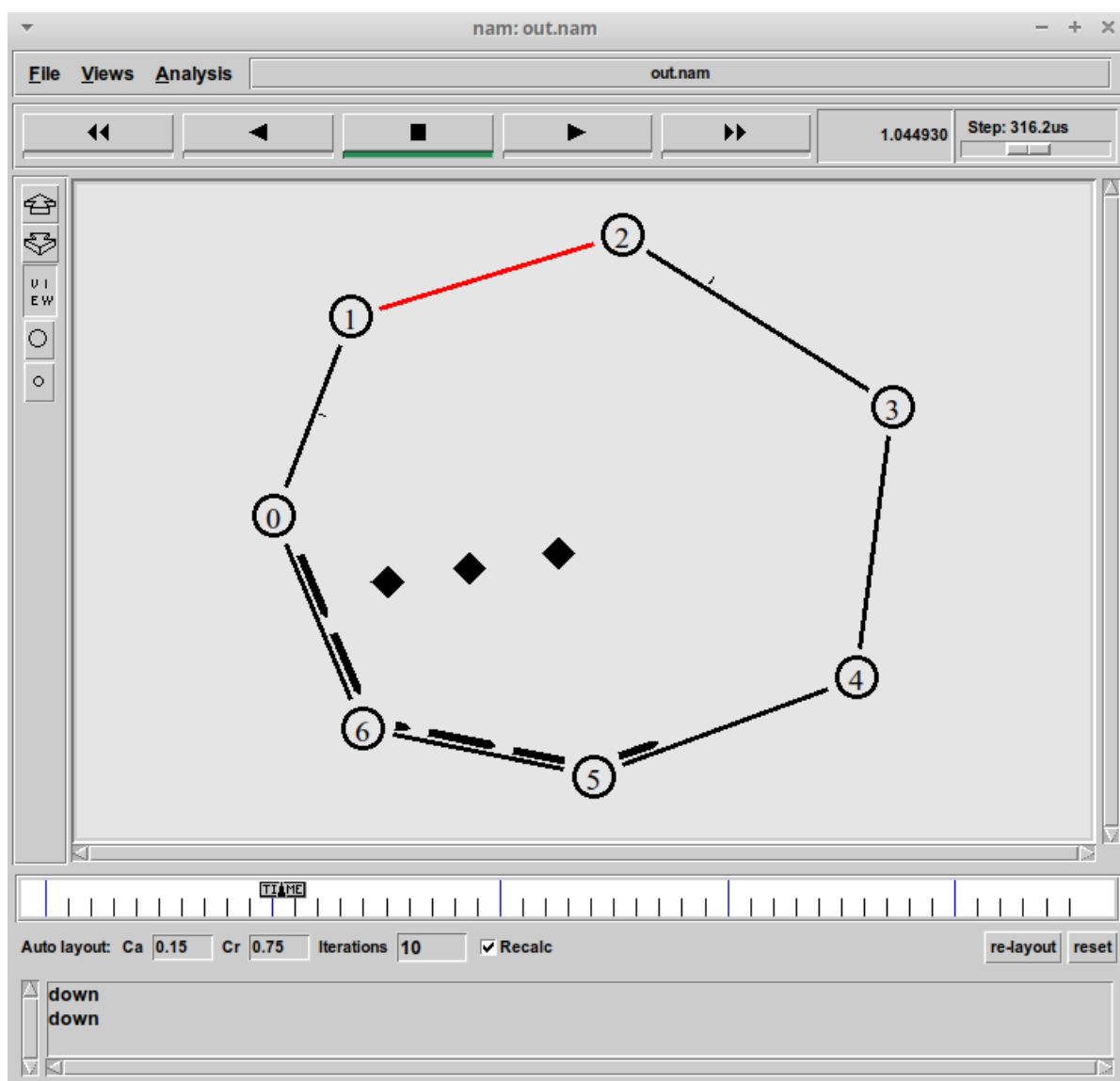
```

Внесли изменения и запустили симулятор:



После запуска в сети отправляется небольшое количество маленьких пакетов, используемых для обмена информацией, необходимой для маршрутизации между узлами





6. Упражнение

Создали файл `example4.tcl` и написали код для данного упражнения:

```

* /home/openmodelica/Desktop/lab1/mip/lab-ns/example4.tcl - Mousepad (на zulfikor)
Файл  Правка  Поиск  Вид  Документ  Справка

# создание объекта Simulator
set ns [new Simulator]
$ns rtproto DV
set nf [open out.nam w]
# все результаты моделирования будут записаны в переменную nf
$ns namtrace-all $nf
set f [open out.tr w]
# все регистрируемые события будут записаны в переменную f
$ns trace-all $f
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 6
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

for {set i 0} {$i < $N-2} {incr i} {
    $ns duplex-link $n($i) $n([expr ($i+1)%$N]) 1Mb 10ms DropTail
}

$ns duplex-link $n(1) $n(5) 2Mb 10ms DropTail
$ns duplex-link $n(4) $n(0) 2Mb 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

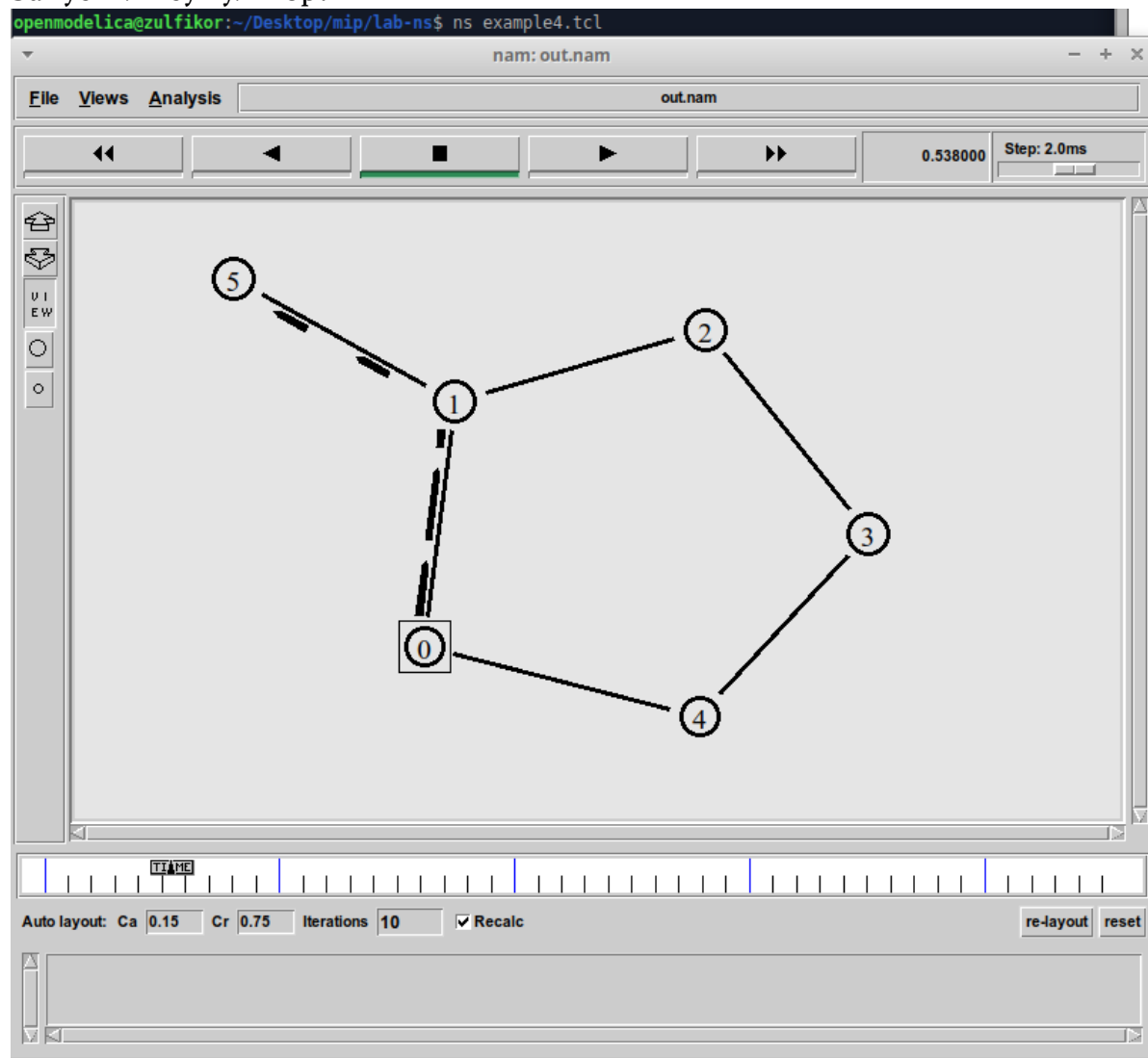
set null0 [new Agent/Null]
$ns attach-agent $n(5) $null0

$ns connect $cbr0 $null0

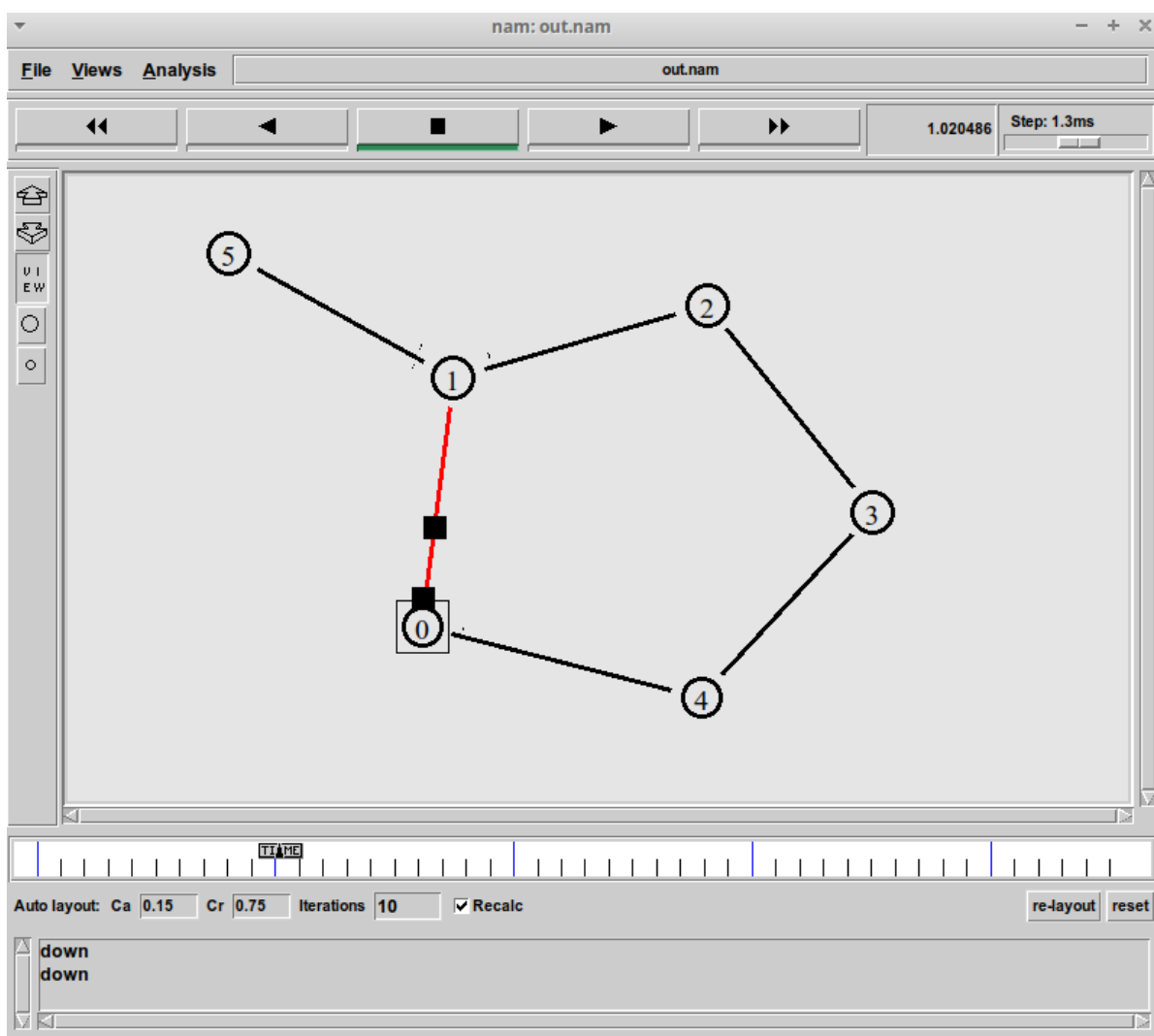
$ns at 0.5 "$cbr0 start"
$ns rtmodel-at 1.0 down $n(0) $n(1)
$ns rtmodel-at 2.0 up $n(0) $n(1)
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"
# запуск модели
$ns run

```

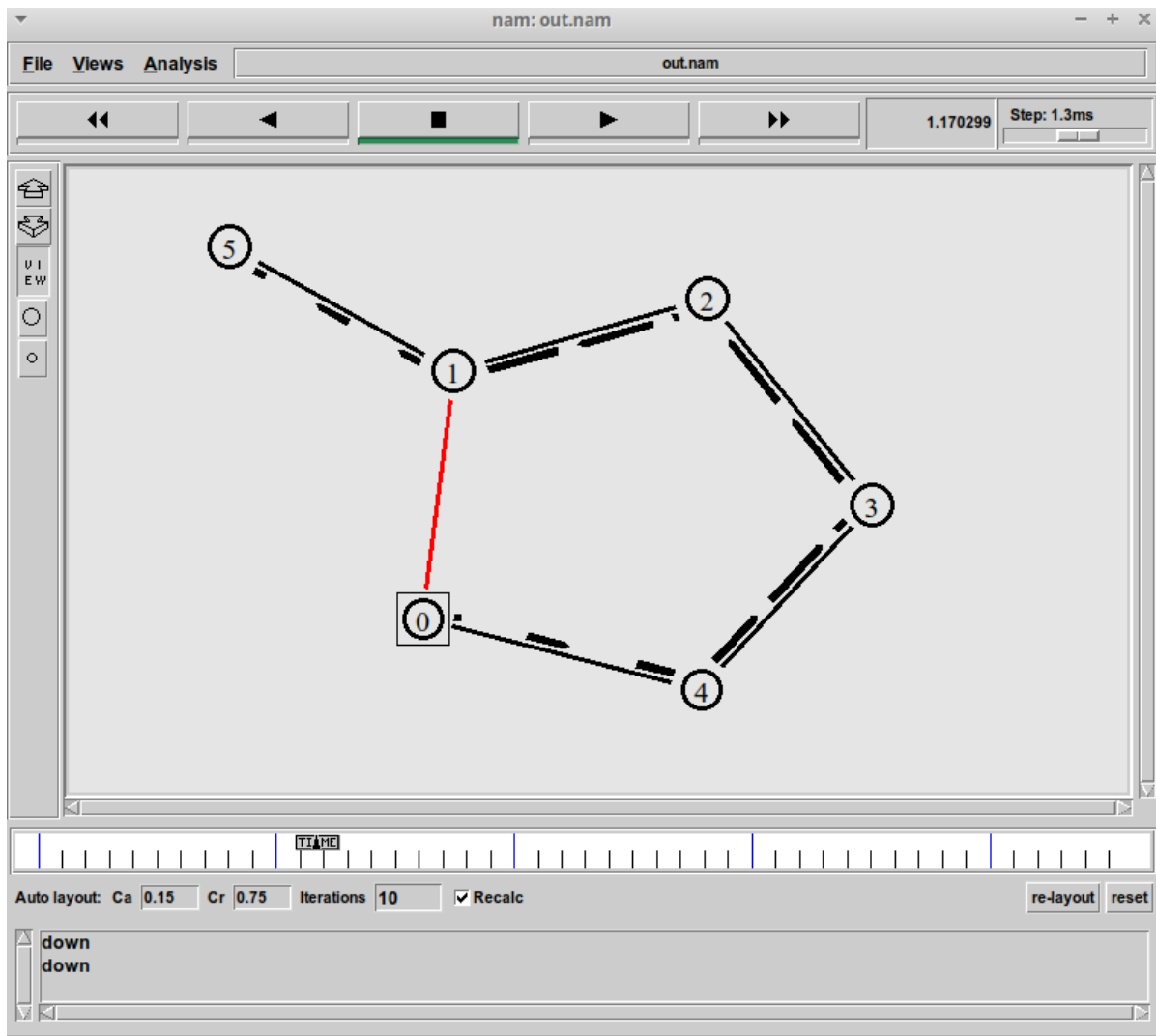
Запустили симулятор:



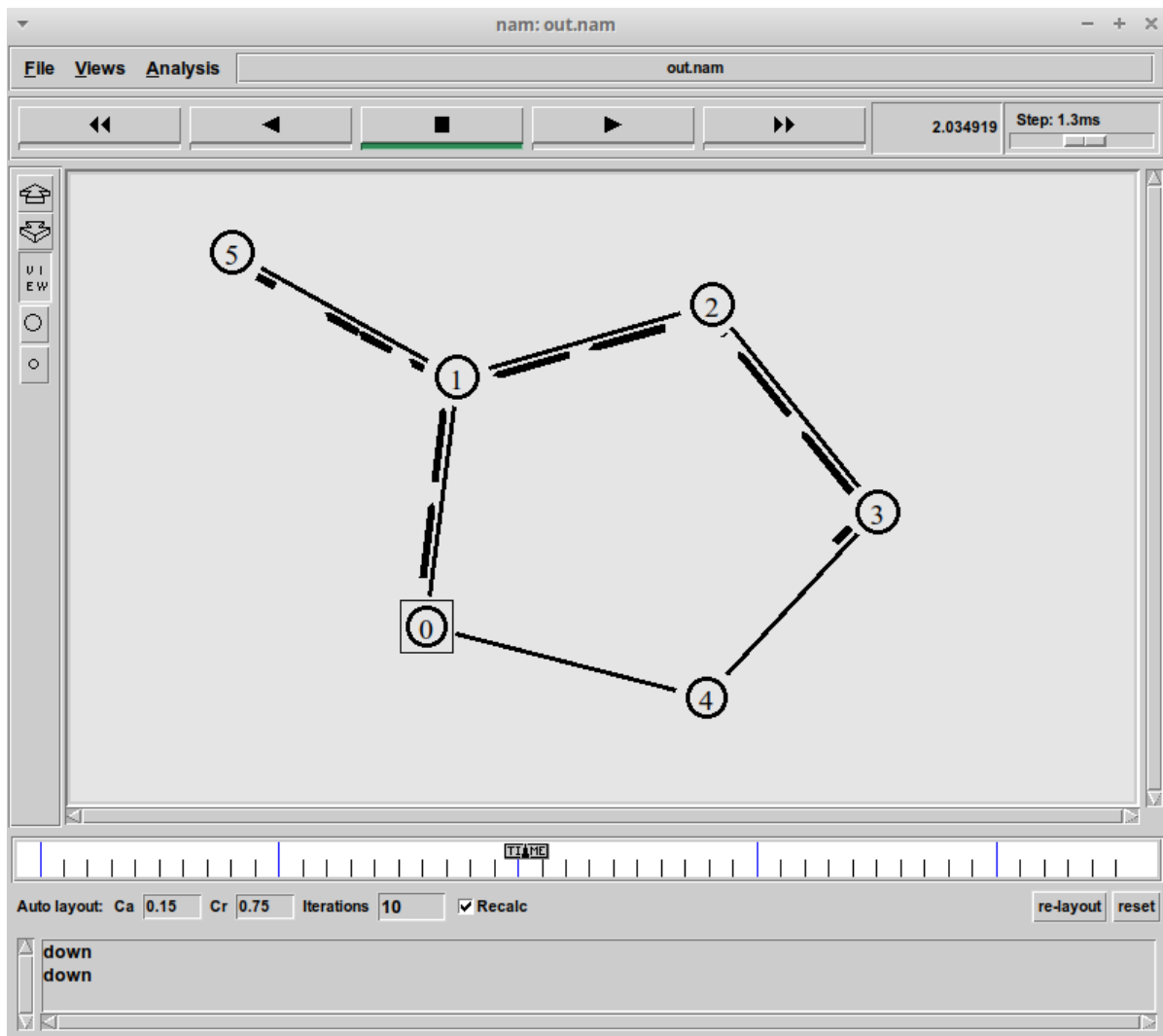
Осуществление передача данных от узла $n(0)$ до узла $n(5)$



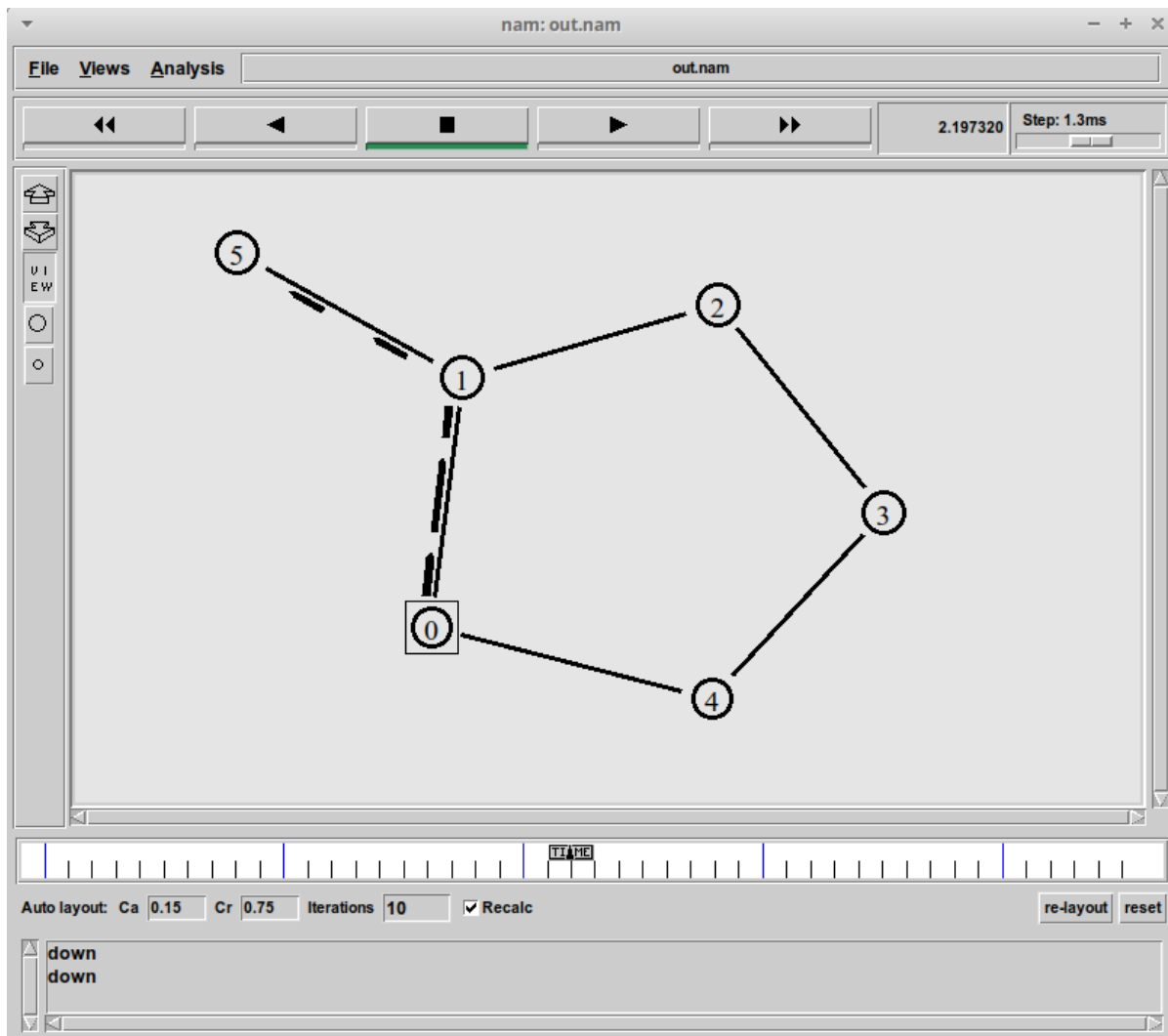
разрыв соединения между узлами $n(0)$ и $n(1)$



Изменение маршрут передачи данных на резервный:



Восстановления соединения пакеты по кратчайшему пути:



7. Выводы

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