

Лабораторная работа № 1

Имитационное моделирование

Королёв И.А.

Российский университет дружбы народов, Москва, Россия

Цель работы

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

Теоретическое введение

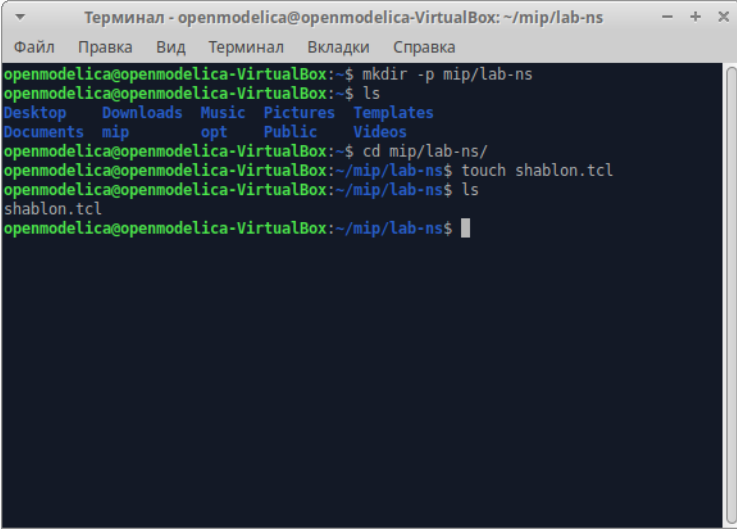
Network Simulator (NS-2) — один из программных симуляторов моделирования процессов в компьютерных сетях. NS-2 позволяет описать топологию сети, конфигурацию источников и приёмников трафика, параметры соединений (полосу пропускания, задержку, вероятность потерь пакетов и т.д.) и множество других параметров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

Выполнение лабораторной работы

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



```
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$
```


Написанные код шаблона сценария для NS-2

Написанные код шаблона сценария для NS-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.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

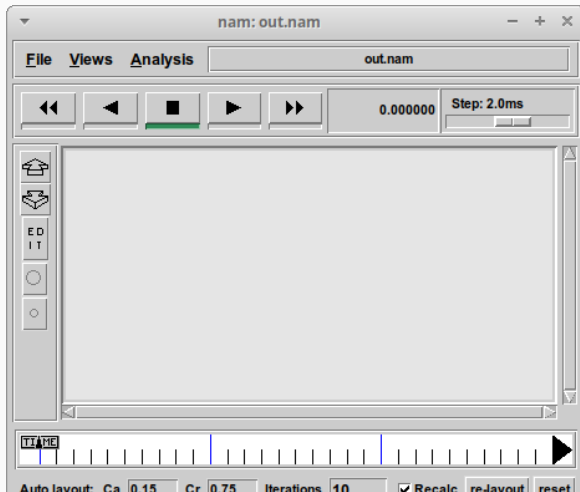
# процедура finish закрывает файлы трассировки
# и запускает визуализатор nam
proc finish {} {
    global ns f nf # описание глобальных переменных
    $ns flush-trace # прекращение трассировки
    close $f # закрытие файлов трассировки
    close $nf # закрытие файлов трассировки nam
    # запуск nam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

Визуальное отображение работающей программы nam.

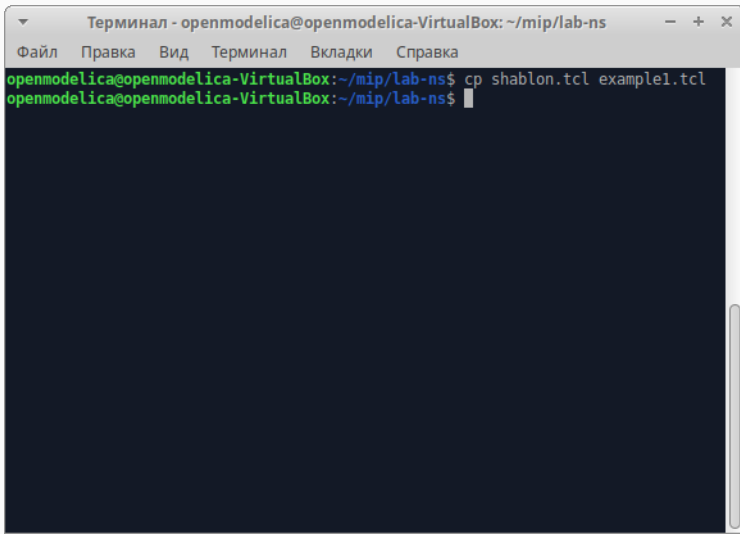
В данном этапе никакого визуального отображения нет, т.к. нет прописанных протоколов передачи данных, агента для генерации и приёма трафика и at-событий.



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.



```
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$
```


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.

```

/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 Мб/с 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

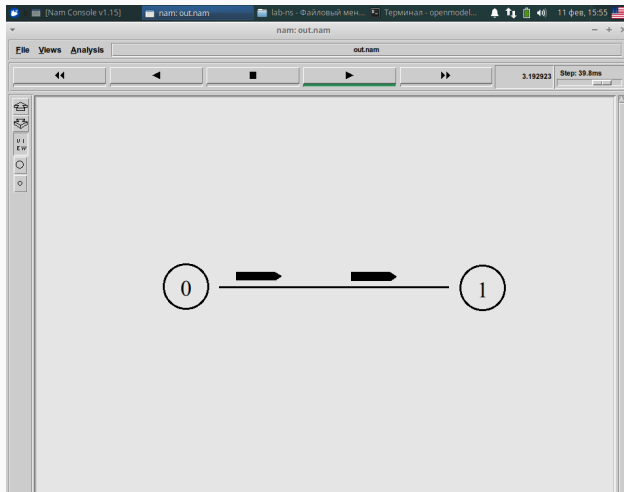
# присоединение источника трафика CBR к агенту udp0
$cbr0 attach-agent $udp0

# 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
```

Результат добавления описания топологии сети.

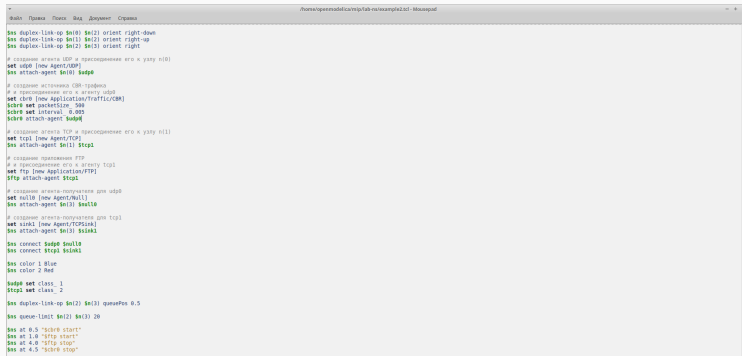
Видим, что через 0.5 секунд из узла 0 данные поступают к узлу 1. Поступление остановится через 4.5 секунды.



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.



```
#!/usr/bin/tclsh /home/opensimodica/isp/fab-ns/example2.tcl -Moosepad

# Файл  Правка  Поиск  Вид  Документ  Справка

$ns duplex-link-ep $n0 $n2 orient right-down
$ns duplex-link-ep $n1 $n2 orient right-up
$ns duplex-link-ep $n2 $n3 orient right

# создание агента UDP и присоединение его к узлу n0)
set udb0 [new Agent/UDP]
$ns attach-agent $n0 $udb0

# создание источника CBR-трафика
# и присоединение его к агенту udb0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize 500
$cbr0 set interval 0.005
$cbr0 attach-agent $udb0

# создание агента TCP и присоединение его к узлу n1)
set tcp1 [new Agent/TCP]
$ns attach-agent $n1 $tcp1

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

# создание агента-получателя для udb0
set null0 [new Agent/Null]
$ns attach-agent $n3 $null0

# создание агента-получателя для tcp1
set sink1 [new Agent/CSMKA]
$ns attach-agent $n3 $sink1

$ns connect $udb0 $null0
$ns connect $tcp1 $sink1

$ns color 1 Blue
$ns color 2 Red

$udb0 set class 1
$tcp1 set class 2

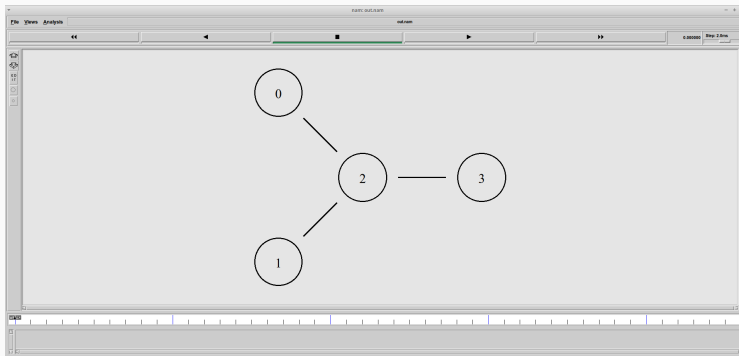
$ns duplex-link-ep $n2 $n3 queuePos 0.5
$ns queue-limit $n2 $n3 20

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

Рис. 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.



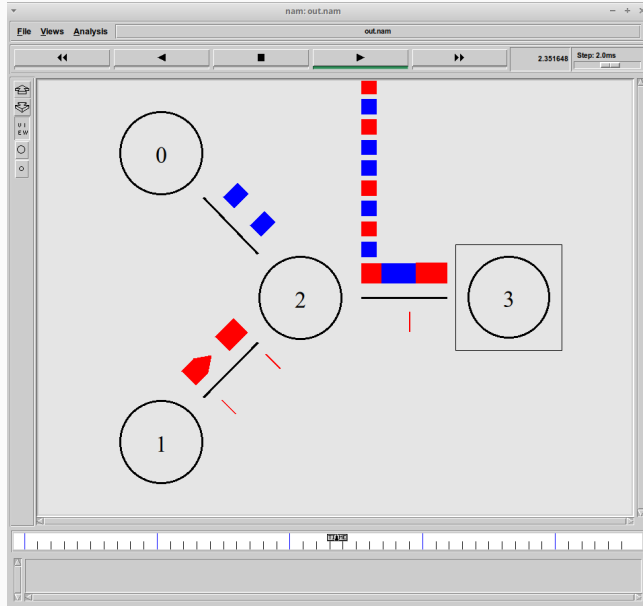
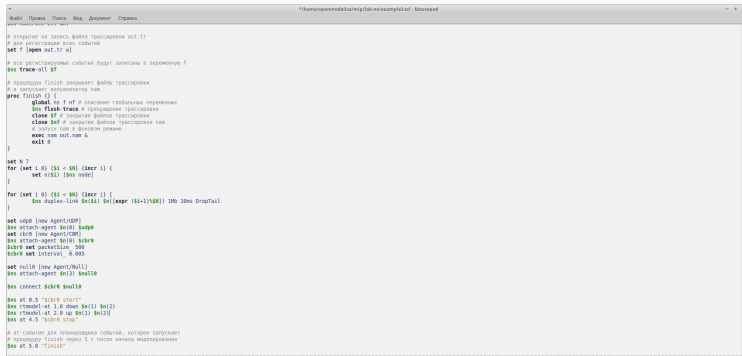


Рис. 9: out.nam

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.



```
#!/usr/bin/env tclsh
# example3.tcl

# 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 интерпретатор nam
proc finish () {
    global ns f nf # глобальные переменные
    $ns flush-trace # промывание трассировки
    close $f # закрытие файла трассировки
    close $nf # закрытие файла трассировки nam
    # запись nam в файл out.nam &
    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}]) 100 10ms DropTail
}

set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr [new Agent/CBR]
$ns attach-agent $n(0) $cbr
$cbr set packetSize 300
$cbr set interval 0.005

set null0 [new Agent/Null]
$ns attach-agent $n(3) $null0
$ns connect $cbr $null0

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

# at-событие для генерации событий, которые запустят
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"
```

Рис. 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.

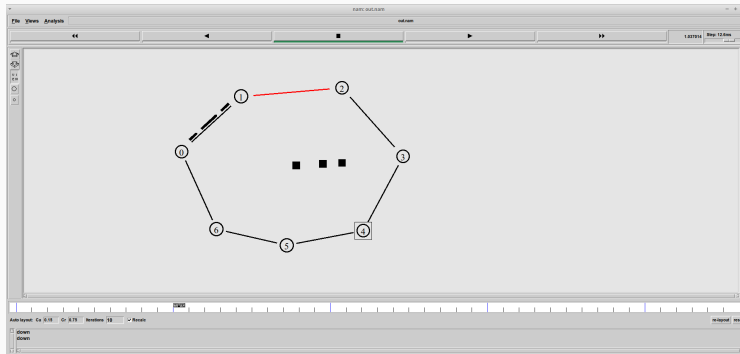


Рис. 11: out.nam

Чтобы пакеты доходили до конечного узла при разрыве, необходимо в начала программы, а после команды создания объекта Simulator добавить `$ns rtproto DV`

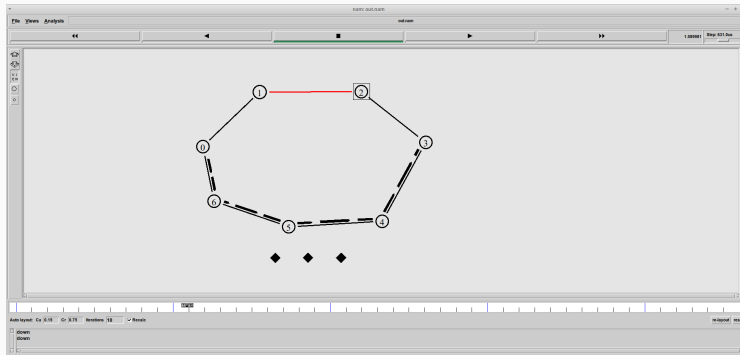


Рис. 12: out.nam

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.

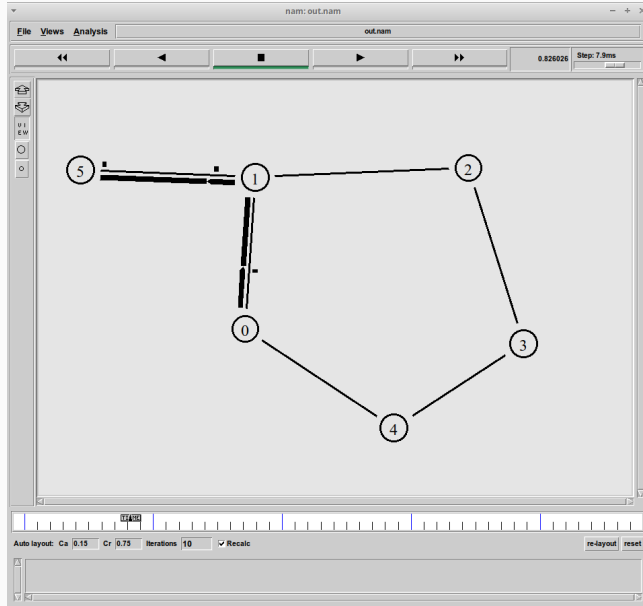


Рис. 13: out.nam

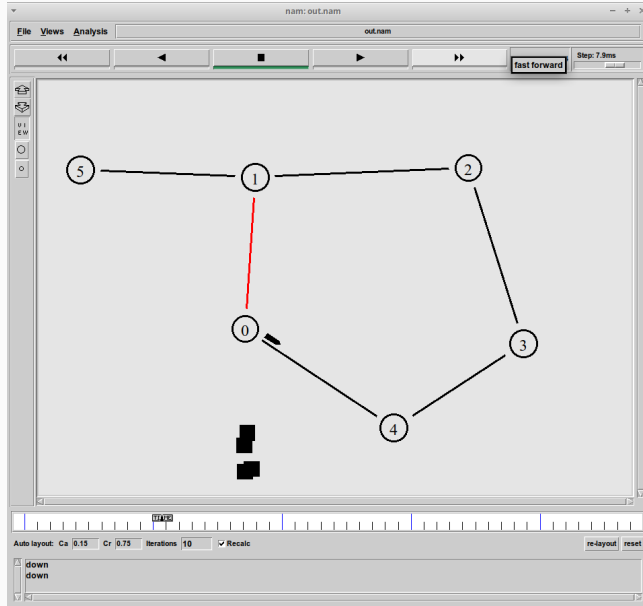


Рис. 14: out.nam

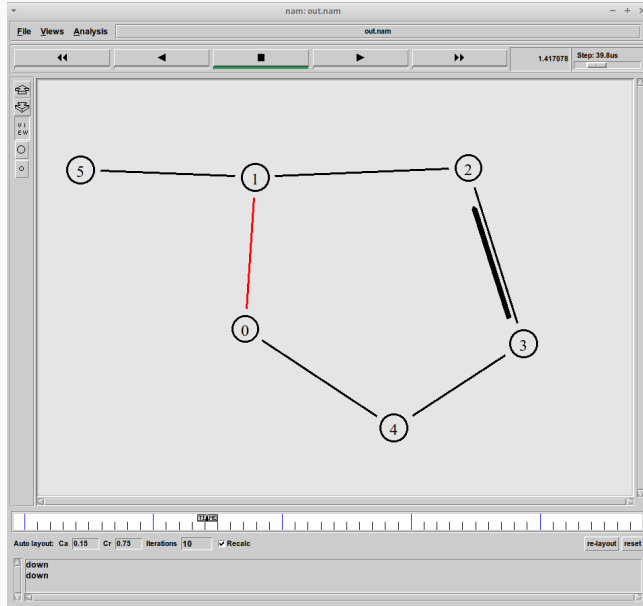


Рис. 15: out.nam

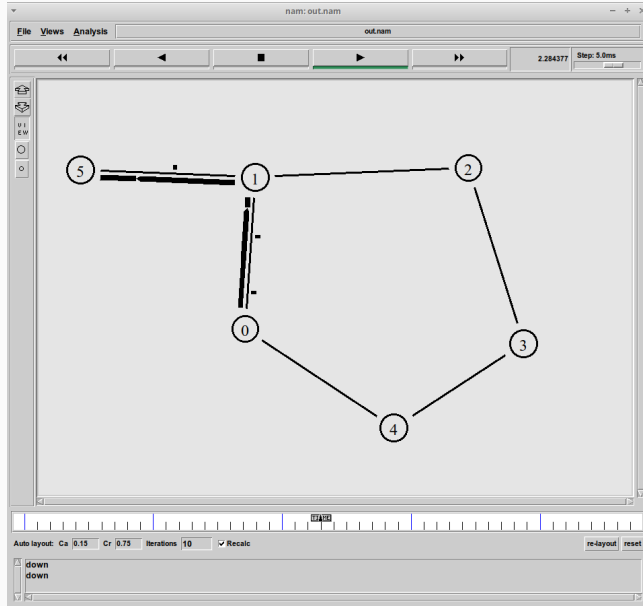


Рис. 16: out.nam

Код реализации.

```
/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
```

Выводы

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

...