

# Лабораторная работа №1

Простые модели компьютерной сети

---

Астраханцева А. А.

14 февраля 2025

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Астраханцева Анастасия Александровна
- НФИбд-01-22, 1132226437
- Российский университет дружбы народов
- 1132226437@pfur.ru
- <https://github.com/aaastrakhantseva>



## Вводная часть

---

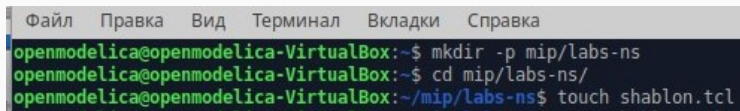
Приобретение навыков моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также анализ полученных результатов моделирования.

1. Создание шаблона сценария для NS-2.
2. Выполнение примера описания топологии сети, состоящей из двух узлов и одного соединения.
3. Выполнение примера описания с усложнённой топологией сети.
4. Выполнение примера с кольцевой топологией сети
5. Выполнение упражнения

## Выполнение ЛР

---

## Создание необходимых директорий и файла



```
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@openmodelica-VirtualBox:~$ mkdir -p mip/labs-ns
openmodelica@openmodelica-VirtualBox:~$ cd mip/labs-ns/
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ touch shablon.tcl
```

Рис. 1: Создание необходимых директорий и файла



```
# 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]

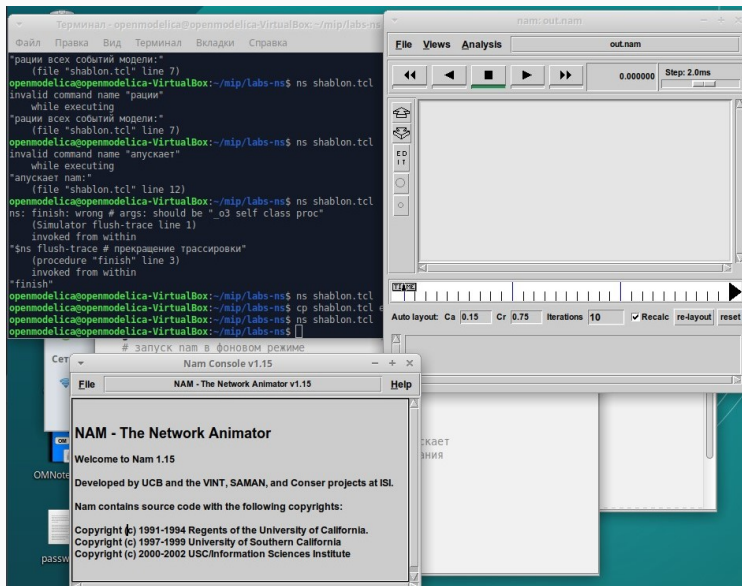
# все регистрируемые события будут записаны в переменную f
$ns trace-all $f

# процедура finish закрывает файлы трассировки
# и запускает визуализатор nam
proc finish {} {
    global ns f nf # объявление глобальных переменных
    # запуск nam в фоновом режиме
    close $f
    close $nf
    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
```

# Запуск симулятора



Выполнение примера описания  
топологии сети, состоящей из двух  
узлов и одного соединения.

---

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.

```
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl  
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ cp shablon.tcl example1.tcl  
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl  
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$
```

Рис. 4: Копирование шаблона в файл example1.tcl

# Скрипт для описания простой сети

```
# 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

# 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

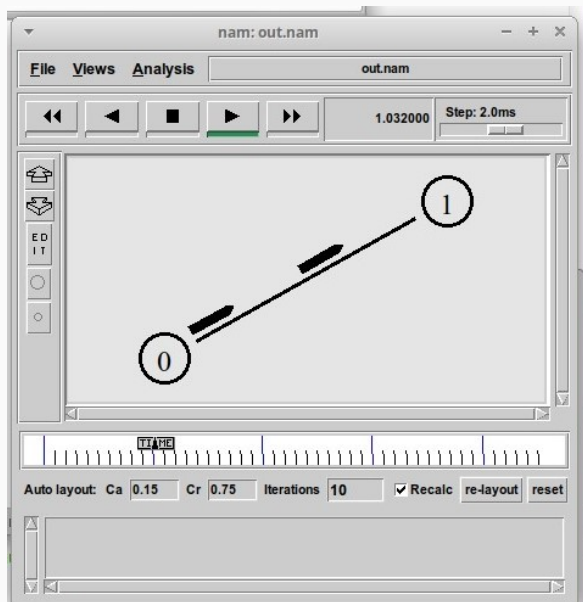
# присоединение источника трафика CBR к агенту udp0
$cbr0 attach-agent $udp0

# запуск приложения через 0,5 с
$ns at 0.5 "$cbr0 start"
# остановка приложения через 4,5 с
$ns at 4.5 "$cbr0 stop"

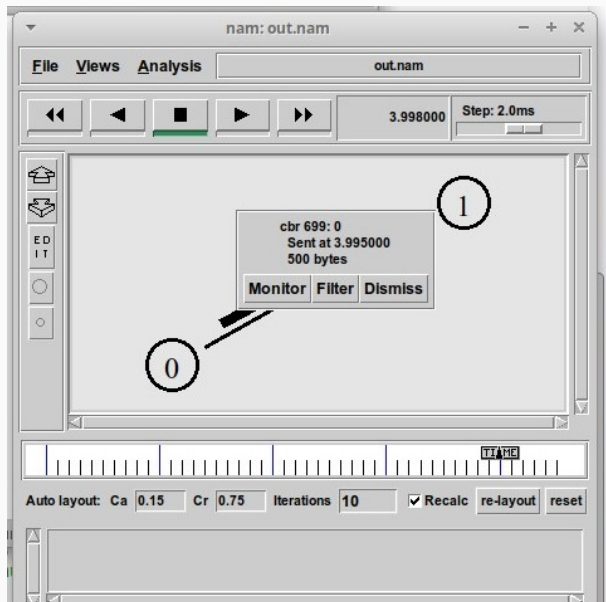
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"

# запуск модели
```

## Запуск аниматора



## Запуск аниматора





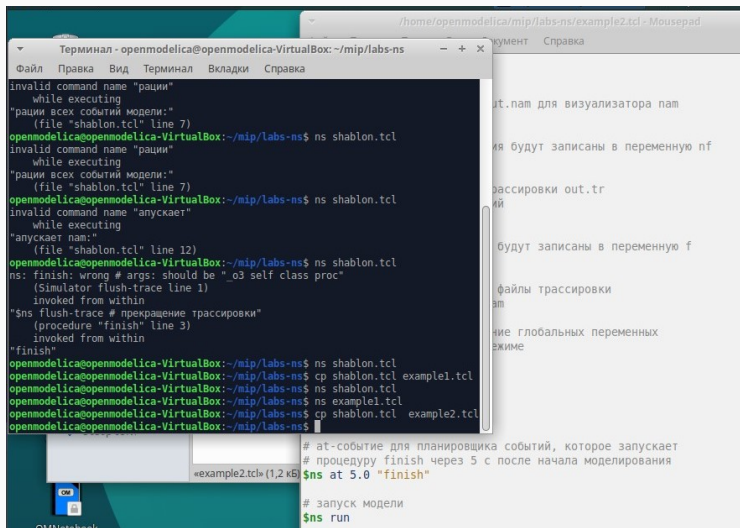
Выполнение примера описания с  
усложнённой топологией сети.

---

## Постановка задачи.

- сеть состоит из 4 узлов ( $n_0$ ,  $n_1$ ,  $n_2$ ,  $n_3$ );
- между узлами  $n_0$  и  $n_2$ ,  $n_1$  и  $n_2$  установлено дуплексное соединение с пропускной способностью 2 Мбит/с и задержкой 10 мс;
- между узлами  $n_2$  и  $n_3$  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  $n_0$  подключается к TCP-приёмнику на узле  $n_3$  (по-умолчанию, максимальный размер пакета, который TCP-агент может генерировать, равняется 1KByte)
- TCP-приёмник генерирует и отправляет ACK пакеты отправителю и откидывает полученные пакеты;
- UDP-агент, который подсоединён к узлу  $n_1$ , подключён к null-агенту на узле  $n_3$  (null-агент просто откидывает пакеты);
- генераторы трафика ftp и cbr прикреплены к TCP и UDP агентам соответственно;

## Копирование шаблона



The screenshot shows a terminal window titled "Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/labs-ns". The terminal displays the execution of a TCL script named "shablon.tcl". The script contains several commands, including "ns shablon.tcl", "ns flush-trace", and "finish". The terminal output shows errors for "invalid command name" and "while executing". The user then copies the contents of "shablon.tcl" into "example2.tcl" using the command "cp shablon.tcl example2.tcl". The terminal also shows the execution of "example2.tcl" with the command "ns example2.tcl".

```
openmodelica@openmodelica-VirtualBox: ~/mip/labs-ns
invalid command name "рации"
while executing
"рации всех событий модели:"
(file "shablon.tcl" line 7)
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
invalid command name "рации"
while executing
"рации всех событий модели:"
(file "shablon.tcl" line 7)
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
invalid command name "анускает"
while executing
"анускает nam:"
(file "shablon.tcl" line 12)
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
ns: finish: wrong # args: should be "_o3 self class proc"
(Simulator flush-trace line 1)
invoked from within
"$ns flush-trace # прекращение трассировки"
(procedure "finish" line 3)
invoked from within
"finish"
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ cp shablon.tcl example1.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns example1.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ cp shablon.tcl example2.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$
```

«example2.tcl» (1,2 кБ)

```
# 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. 8: Копирование шаблона в файл example2.tcl

```
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
```

Рис. 9: Создание 4-х узлов и 3-х дуплексных соединения с указанием направления

```
# 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
```

Рис. 11: Создание агентов-получателей. Соединение агентов udp0 и tcp1 и их получателей

```
$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
```

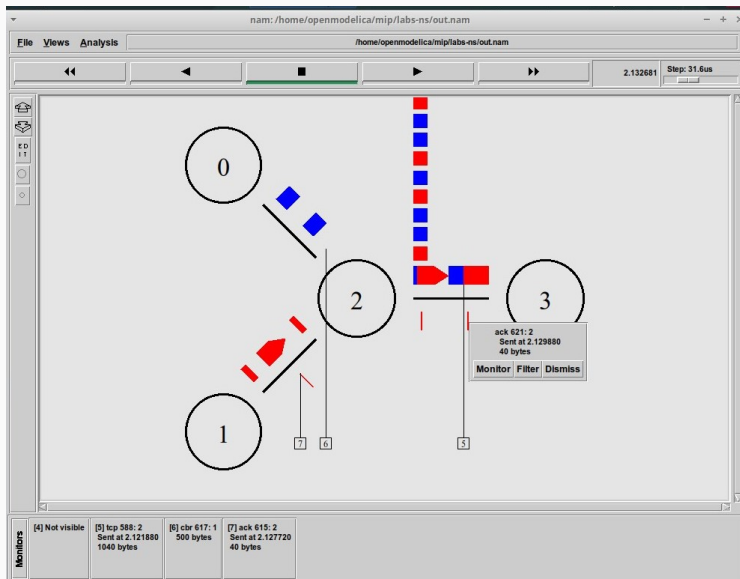
Рис. 12: Задание цвета потока. Отслеживание событий в очереди и наложение ограничения на размер очереди

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

Рис. 13: Добавление at-событий



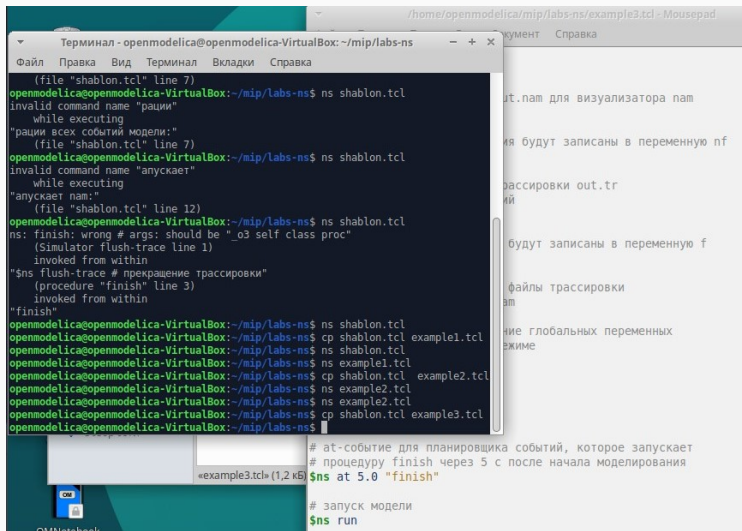
# Запуск аниматора



Выполнение примера с кольцевой  
топологией сети

---

- 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.



The screenshot shows a terminal window titled "Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/labs-ns". The terminal displays the following commands and output:

```
(file "shablon.tcl" line 7)
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
invalid command name "рации"
while executing
"рации всех событий модели:"
(file "shablon.tcl" line 7)
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
invalid command name "пускает"
while executing
"пускает nam:"
(file "shablon.tcl" line 12)
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
ns: finish: wrong # args: should be "_o3 self class proc"
(Simulator flush-trace line 1)
invoked from within
"$ns flush-trace # прекращение трассировки"
(procedure "finish" line 3)
invoked from within
"finish"
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ cp shablon.tcl example1.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns example1.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ cp shablon.tcl example2.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns example2.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ ns example2.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$ cp shablon.tcl example3.tcl
openmodelica@openmodelica-VirtualBox:~/mip/labs-ns$
```

Below the terminal window, a file explorer shows the contents of "example3.tcl" (1,2 KB):

```
# 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. 15: Копирование шаблона в файл example3.tcl

```
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
}
```

Рис. 16: Создание круговой топологии

```
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0

set cbr0 [new Agent/CBR]
$ns attach-agent $n(0) $cbr0

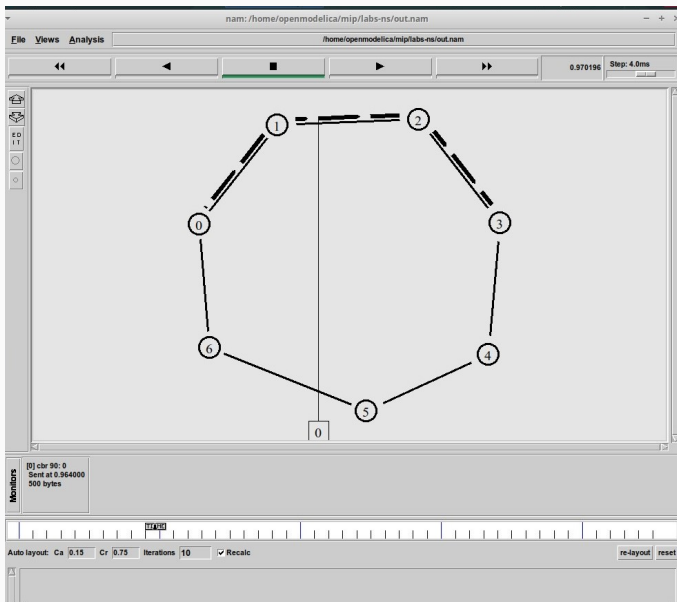
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005

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

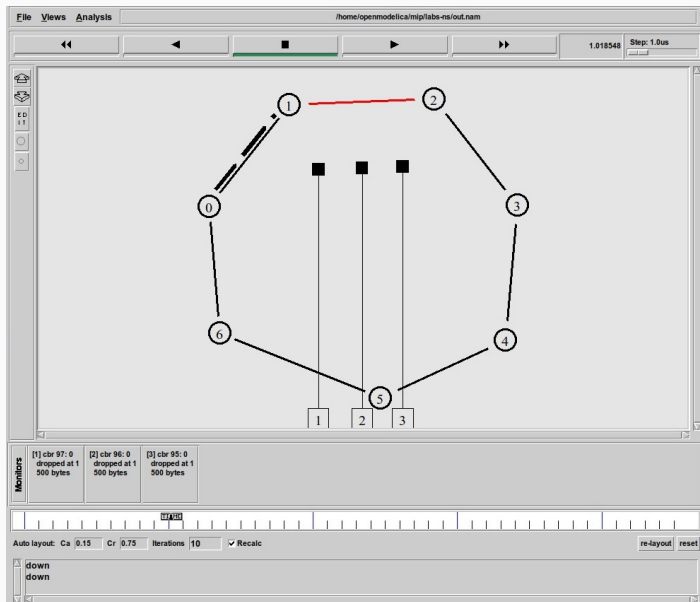
$ns connect $cbr0 $null0

$ns at 0.5 "$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"
```

# Создание скрипта



# Создание скрипта





```
▼ /home/openmodelica/mip/labs-ns/example3.tcl - Mousepad
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]

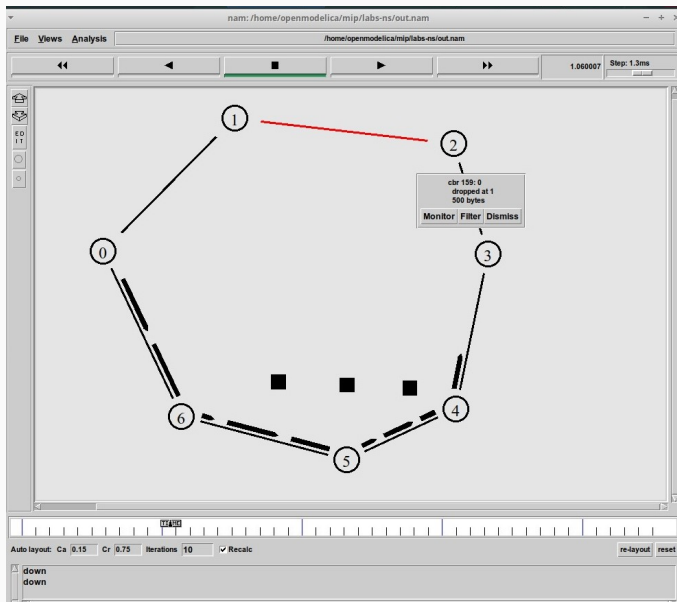
# все результаты моделирования будут записаны в переменную nf
$ns namtrace-all $nf

# 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 # объявление глобальных переменных
    # запуск nam в фоновом режиме
    close $f
    close $nf
    exec nam out.nam &
    exit 0
}
```

# Запуск аниматора



## Выполнение упражнения

---

## Постановка задачи

- Внесите следующие изменения в реализацию примера с кольцевой топологией сети:
- топология сети должна соответствовать представленной на рисунке в тексте описания ЛР.
- 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.

```
,
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

# создание агента TCP и присоединение его к узлу n(0)
set tcp1 [new Agent/TCP]
$ns attach-agent $n(0) $tcp1

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

# создание агента-получателя для tcp1
set sink1 [new Agent/TCPSink]
$ns attach-agent $n5 $sink1

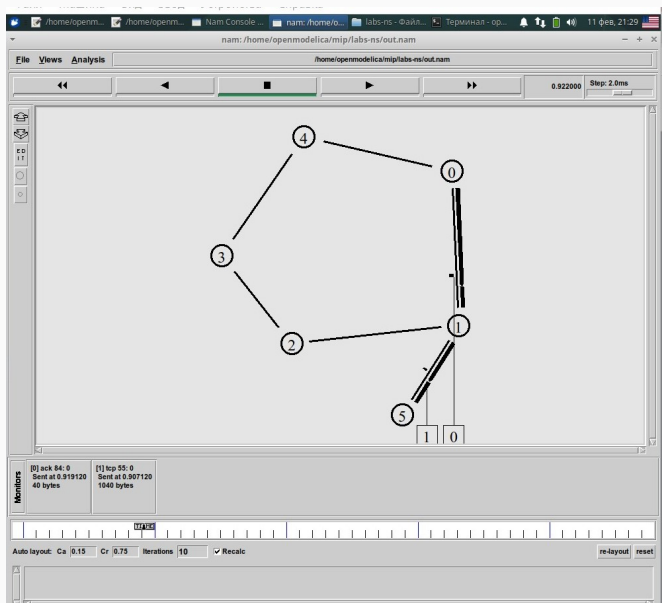
$ns connect $tcp1 $sink1

$ns at 0.5 "$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"

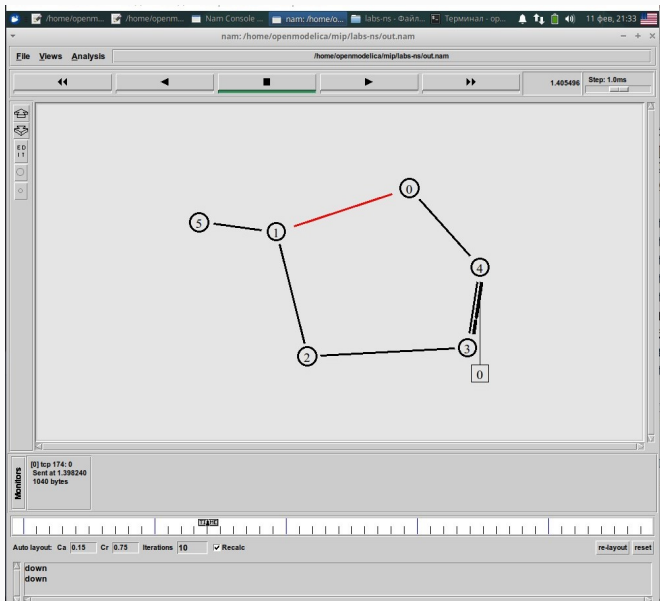
# at-событие для планировщика событий, которое запускает
# процедуру finish через 5 с после начала моделирования
$ns at 5.0 "finish"

# запуск модели
run
```

# Запуск аниматора



# Запуск аниматора



В ходе выполнения лабораторной работы я приобрела навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также провела анализ полученных результатов моделирования.



Спасибо за внимание!

---