

# Лабораторная работа 1

Простые модели компьютерной сети

---

Оганнисян Д.Б.

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Оганнисян Давит Багратович
- студент
- Российский университет дружбы народов
- 1132226440@pfur.ru
- <https://dbogannisyanNKA.github.io/ru/>



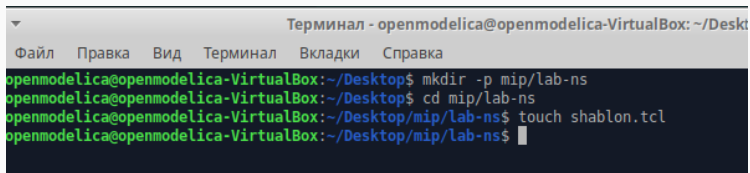
Приобрести навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также проанализировать полученные результаты моделирования.

1. Создать шаблон сценария для NS-2;
2. Выполнить простой пример описания топологии сети, состоящей из двух узлов и одного соединения;
3. Выполнить пример с усложнённой топологией сети;
4. Выполнить пример с кольцевой топологией сети;
5. Выполнить упражнение.

## **Выполнение лабораторной работы**

---

## Шаблон сценария для NS-2



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

**Рис. 1:** Создание директорий и файла

## Шаблон сценария для NS-2

```
# создание объекта Simulator
set ns [new Simulator]

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

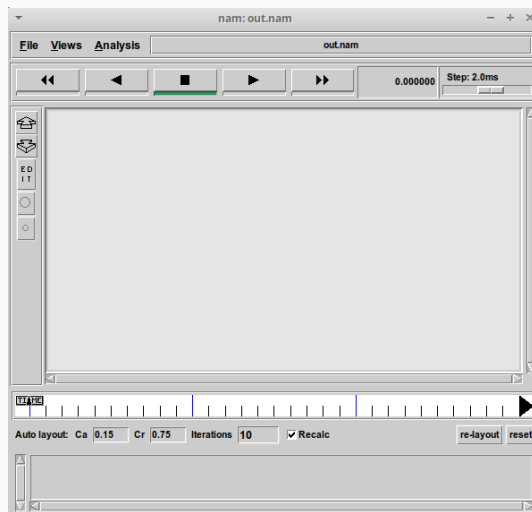
# все результаты моделирования будут записаны в переменную nf
$ns namtrace-all $nf

# 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
    $ns flush-trace
    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
```

Рис. 2: Редактирование файла shablon.tcl



## Шаблон сценария для NS-2



**Рис. 3:** Запуск шаблона сценария для NS-2

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

```
}
# 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"
||
```

Рис. 4: Пример описания топологии сети, состоящей из двух узлов и одного соединения

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

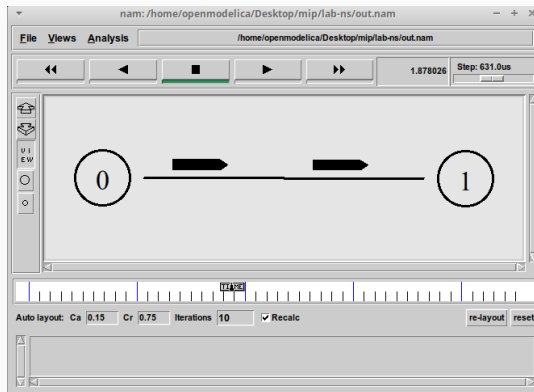


Рис. 5: Визуализация простой модели сети с помощью nam

## Пример с усложнённой топологией сети

```
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. 6: Визуализация простой модели сети с помощью nam

## Пример с усложнённой топологией сети

```
# 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
```

Рис. 7: Описание усложненной топологии сети

## Пример с усложнённой топологией сети

```
# 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

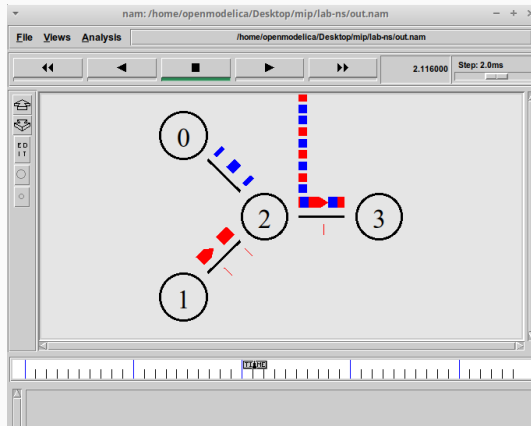
$ns duplex-link-op $n(2) $n(3) queuePos 0.5

$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 5.0 "$cbr0 stop"
# at 5.0 "$ftp stop"
```

Рис. 8: Описание усложненной топологии сети

## Пример с усложнённой топологией сети



**Рис. 9:** Описание усложненной топологии сети

## Пример с кольцевой топологией сети

```
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

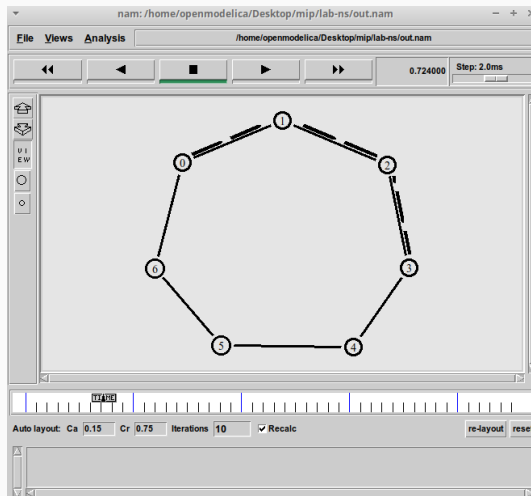
$ns at 0.5 "$cbr0 start"
$ns rtmodel-at 1.0 down $n(1) $n(2)
$ns rtmodel-at 2.0 up $n(1) $n(2)
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"

# at-событие для планировщика событий, которое запускает
```

**Рис. 10:** Описание кольцевой топологии сети и динамической маршрутизацией пакетов

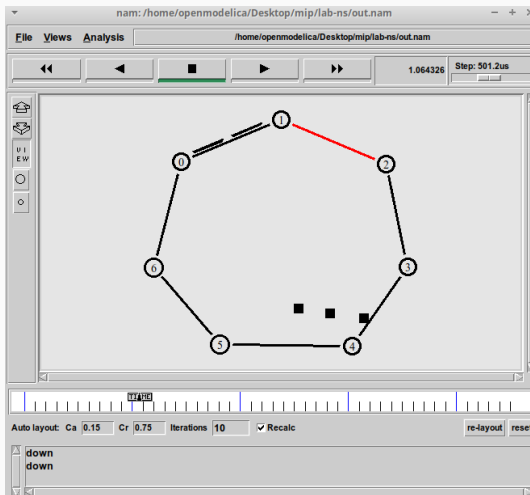


## Пример с кольцевой топологией сети



**Рис. 11:** Передача данных по кратчайшему пути сети с кольцевой топологией

## Пример с кольцевой топологией сети



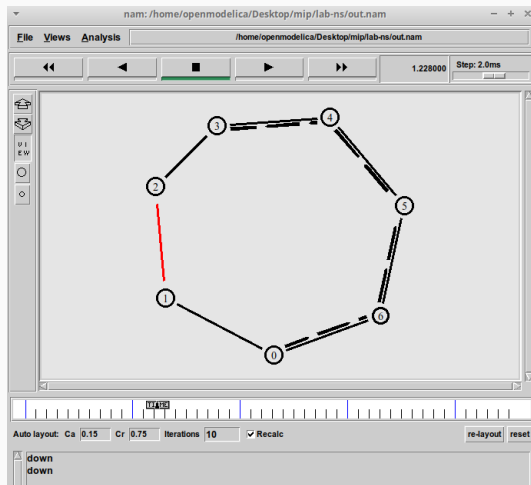
**Рис. 12:** Передача данных по сети с кольцевой топологией в случае разрыва соединения

Добавив в начало скрипта после команды создания объекта Simulator:

```
$ns rtproto DV
```

увидим, что сразу после запуска в сети отправляется небольшое количество маленьких пакетов, используемых для обмена информацией, необходимой для маршрутизации между узлами.

## Пример с кольцевой топологией сети



**Рис. 13:** Маршрутизация данных по сети с кольцевой топологией в случае разрыва соединения

```
}  
  
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/TCP/Sink/DelAck]  
$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"  
$ns at 5.0 "finish"
```

Рис. 14: Программа для упражнения по построению топологии сети

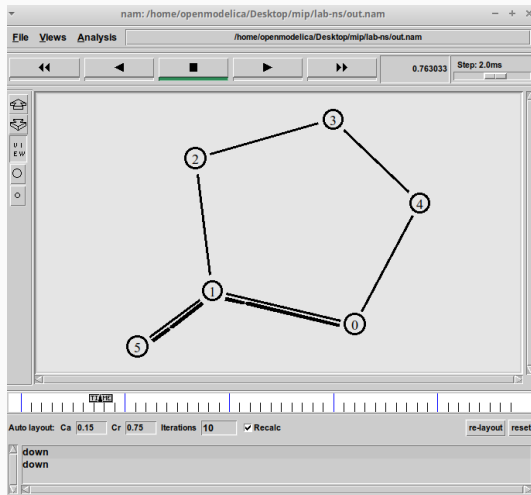


Рис. 15: Передача данных по изменённой кольцевой топологии сети

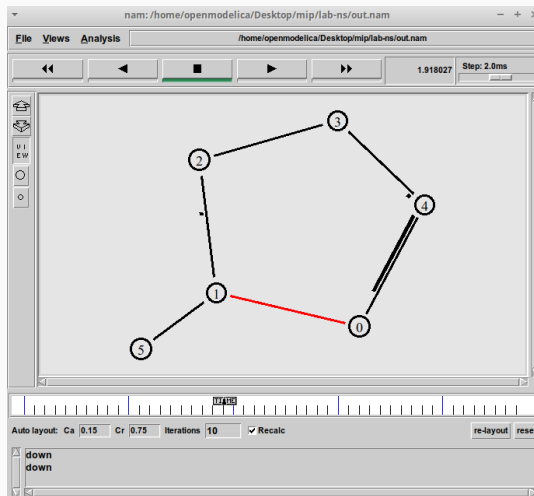


Рис. 16: Передача данных по сети в случае разрыва соединения

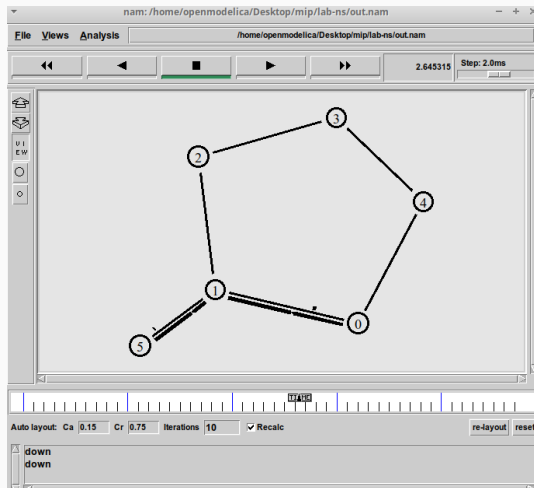


Рис. 17: Передача данных после восстановления соединения



В процессе выполнения данной лабораторной работы я приобрел навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также проанализировал полученные результаты моделирования.