

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук Кафедра прикладной  
информатики и теории вероятностей**

**ОТЧЕТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1**

*дисциплина: Моделирование информационных процессов*

Студент: Кузнецов Юрий Владимирович

СТ/Б: 1032200533

Группа: НФИбд 01-20

**МОСКВА**

2023 г.

## Цель работы:

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

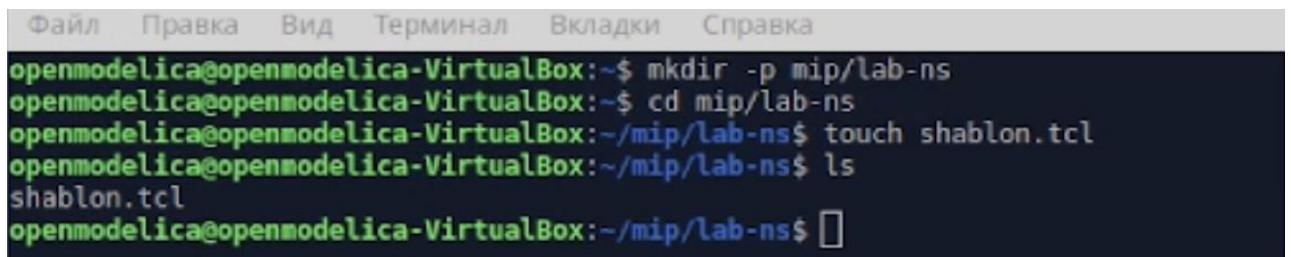
## Задачи:

- Ознакомиться с **NS-2**.
- Смоделировать сеть передачи данных.
- Проанализировать результаты.
- Выполнить упражнение.

## Ход работы:

### 1.1. Создаём Шаблон сценария для NS-2:

- В своём рабочем каталоге создаём директорию **mip**, к которой будут выполняться лабораторные работы. Внутри **mip** создаём директорию **lab-ns**, а в ней файл **shablon.tcl**:



```
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@openmodelica-VirtualBox:~$ mkdir -p mip/lab-ns
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$
```

- Открываем на редактирование файл **shablon.tcl**. Можно использовать любой текстовый редактор типа emacs.
- Сначала создадим объект типа Simulator
- Затем создадим переменную **nf** и укажем, что требуется открыть на запись **nam**-файл для регистрации выходных результатов моделирования
- Далее создадим переменную **f** и откроем на запись файл трассировки для регистрации всех событий модели:
- После этого добавим процедуру **finish**, которая закрывает файлы трассировки и запускает **nam**:
- Наконец, с помощью команды **at** указываем планировщику событий, что процедуру **finish** следует запустить через 5 с после начала моделирования, после чего запустить симулятор **ns**:

```
*/home/openmodelica/mip/lab-ns/shablon.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка
set ns [new Simulator]
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
    $ns flush-trace
    close $f
    close $nf
    exec nam out.nam &
    exit 0
}
$ns at 5.0 "finish"
$ns run
```

- Сохранив изменения в отредактированном файле **shablon.tcl** и закрыв его, можно запустить симулятор командой:

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

- Получившийся шаблон можно использовать в дальнейшем в большинстве разрабатываемых скриптов NS-2, добавляя в него до строки **\$ns at 5.0 "finish"** описание объектов и действий моделируемой системы.

## 1.2. Моделируем сеть передачи данных, состоящую из двух узлов:

- Скопируем содержимое созданного шаблона в новый файл:

```
Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/lab-ns
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@openmodelica-VirtualBox:~$ mkdir -p mip/lab-ns
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$ ns shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ns shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ cp shablon.tcl example1.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$
```

и откроем **example1.tcl** на редактирование. Добавим в него до строки **\$ns at 5.0 "finish"** описание топологии сети

- Создадим агенты для генерации и приёма трафика
- Далее создадим Null-агент, который работает как приёмник трафика, и прикрепим его к узлу n1
- Соединим агенты между собой
- Для запуска и остановки приложения CBR добавляются at-события в планировщик событий (перед командой `$ns at 5.0 "finish"`)

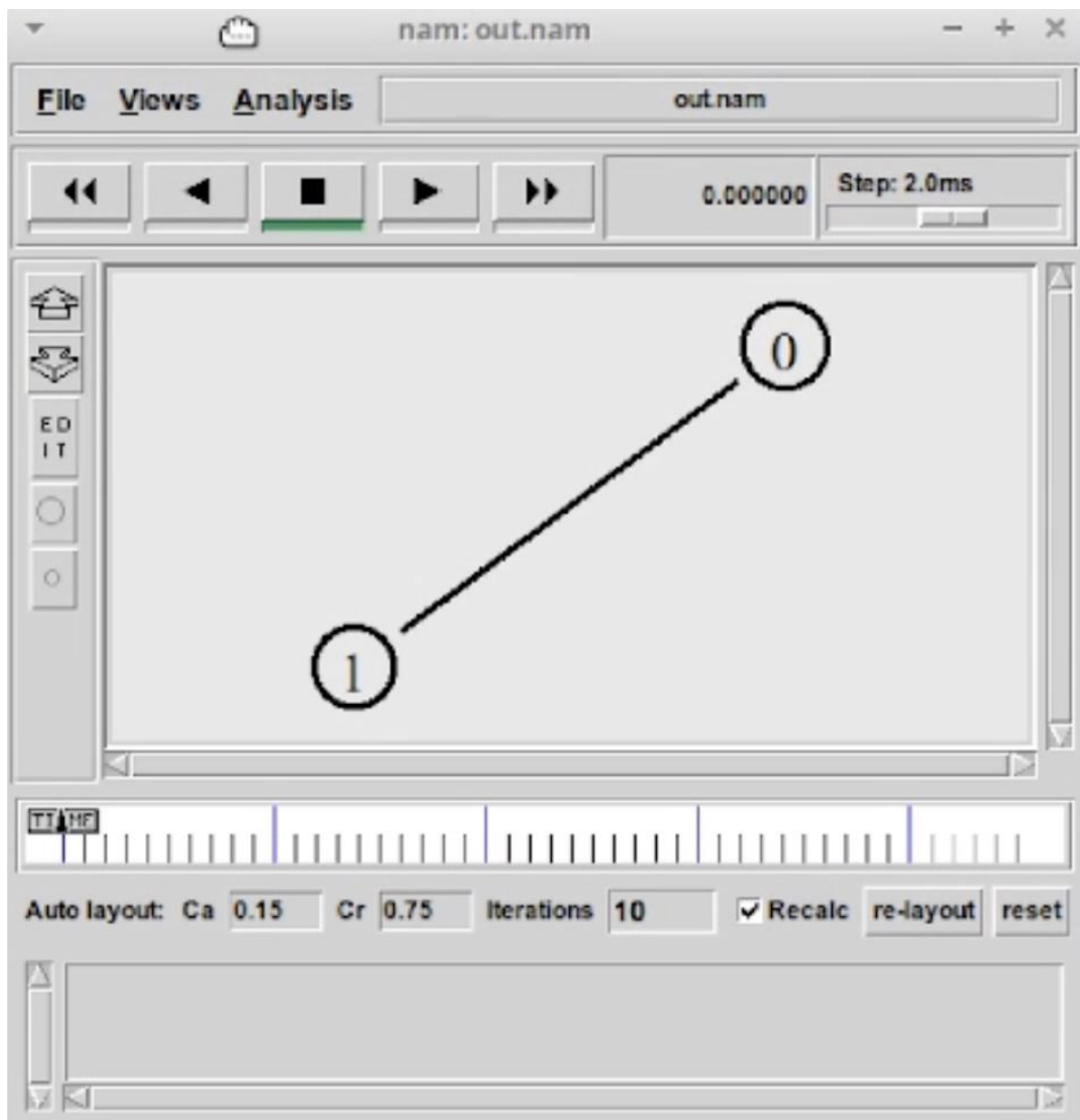
```
set N 2
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns nide]
}

$ns duplex-link $n(0) $n(1) 2Mb 10ms DropTail
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set intercal_ 0.005
$cbr0 attach-agent $udp0
set null0 [new Agent/Null]
$ns attach-agent $n(1) $null0
$ns connect $udp0 $null0

$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```

Сохранив изменения в отредактированном файле и запустив симулятор:

```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ns example1.tcl
```



### 1.3. Моделируем сеть передачи данных, с усложнённой топологией сети:

- Скопируем содержимое созданного шаблона в новый файл:

```

Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/lab-ns
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@openmodelica-VirtualBox:~$ cd mip
openmodelica@openmodelica-VirtualBox:~/mip$ ls
lab-ns
openmodelica@openmodelica-VirtualBox:~/mip$ cd mip/lab-ns
bash: cd: mip/lab-ns: Нет такого файла или каталога
openmodelica@openmodelica-VirtualBox:~/mip$ cd lab-ns
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ls
example1.tcl  out.nam  out.tr  shablon.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ cp shablon.tcl example2.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$

```

и откроем **example2.tcl** на редактирование.

Создадим 4 узла и 3 дуплексных соединения с указанием направления:

- Создадим агент UDP с прикреплённым к нему источником CBR и агент TCP с прикреплённым к нему приложением FTP:
- Создадим агенты-получатели:
- Соединим агенты udr0 и tcp1 и их получателей:
- Зададим описание цвета каждого потока:
- Отслеживание событий в очереди:
- Наложение ограничения на размер очереди:
- Добавление at-событий:

```

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

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

set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0

set tcp1 [new Agent/TCP]
$ns attach-agent $n(1) $tcp1

set ftp [new Application/FTP]
$ftp attach-agent $tcp1

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

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

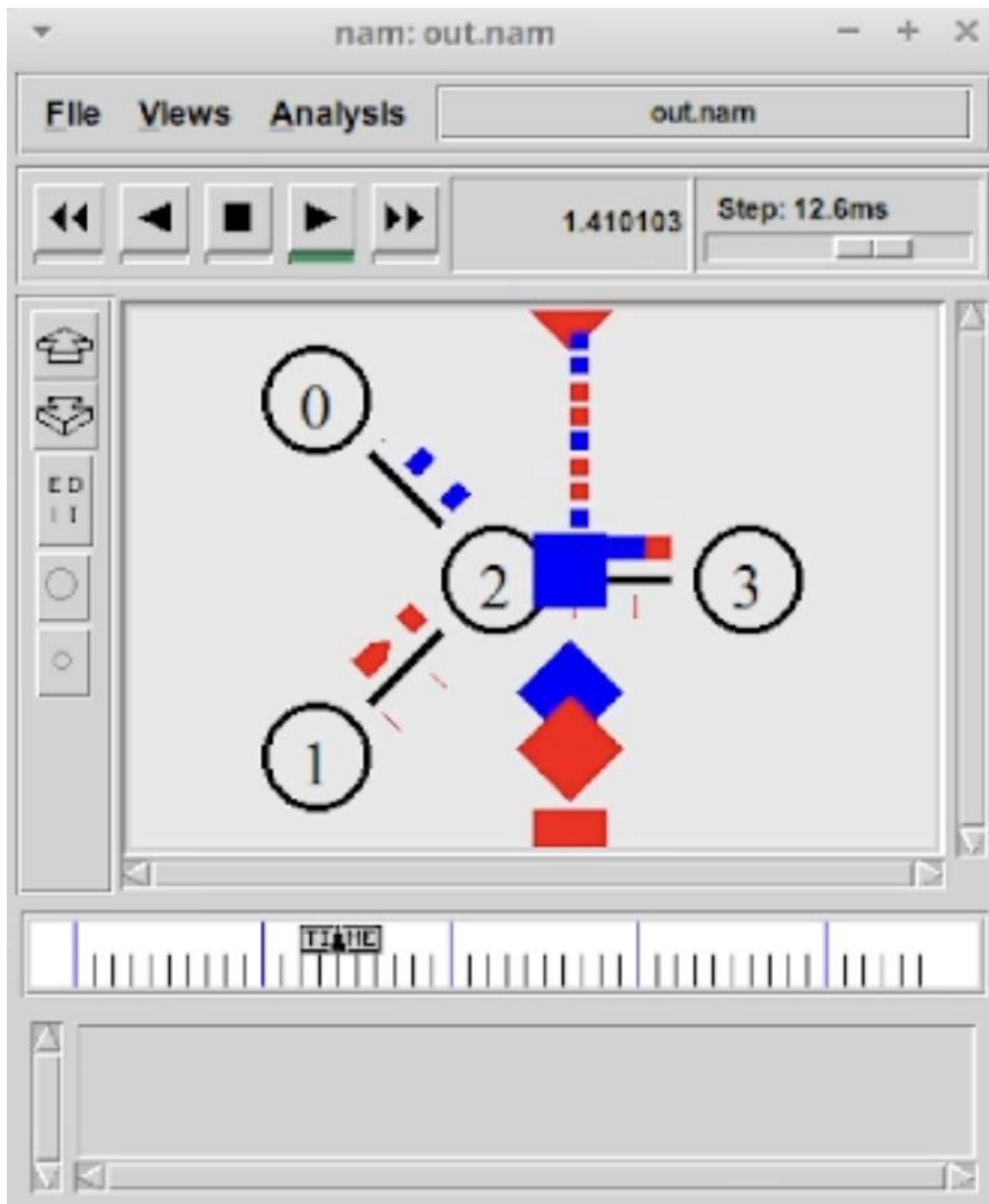
$ns at 0.5 "$cbr0 start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr0 stop"

$ns run

```

- Сохранив изменения в отредактированном файле и запустив симулятор, получим анимированный результат моделирования:





#### 1.4. Моделируем сеть передачи данных, с усложнённой топологией сети:

Скопируем содержимое созданного шаблона в новый файл: `cp shablon.tcl example3.tcl`

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

и откроем `example3.tcl` на редактирование. Опишем топологию моделируемой сети

- Далее соединим узлы так, чтобы создать круговую топологию
- Зададим передачу данных от узла  $n(0)$  к узлу  $n(3)$
- Добавим команду разрыва соединения между узлами  $n(1)$  и  $n(2)$  на время в одну секунду, а также время начала и окончания передачи данных

```

set N 6
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

for {set i 0} {$i < $N} {incr i} {
    if {$i==4} {
        $ns duplex-link $n($i) $n([expr ($i+2)%$N]) 1Mb 10ms DropTail
    }
    if {$i==5} {
        $ns duplex-link $n($i) $n([expr ($i+2)%$N]) 1Mb 10ms DropTail
    }
    if {($i!=4) && ($i!=5)} {
        $ns duplex-link $n($i) $n([expr ($i+1)%$N]) 1Mb 10ms DropTail
    }
}

set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n(0) $tcp
set ftp [new Application/FTP]
$ftp attach-agent $tcp

$ftp set packetSize_ 500
$ftp set interval_ 0.005

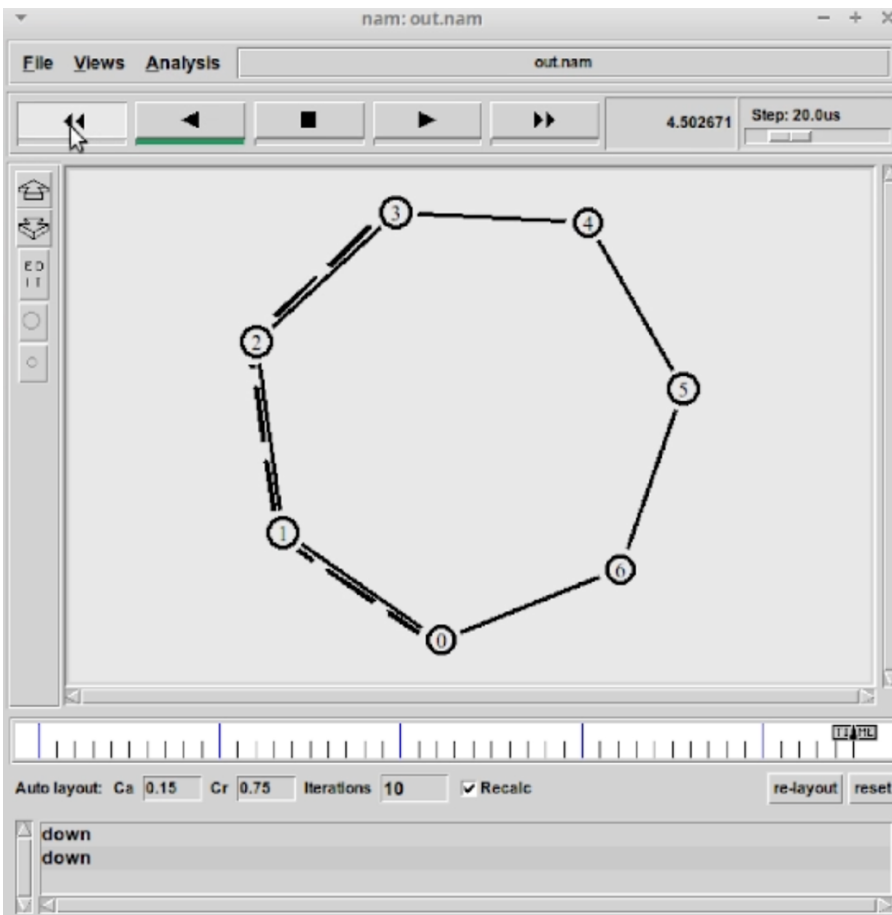
set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n(5) $sink

$ns connect $tcp $sink

$ns at 0.5 "$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"

$ns run

```





## 2. Моделируем сеть передачи данных, в соответствии с требованиями, описанными в упражнении:

Требования:

- топология сети должна соответствовать представленной в материалах
- 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.

Выполнение:

```
set N 6
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

for {set i 0} {$i < $N} {incr i} {
    if {$i==4} {
        $ns duplex-link $n($i) $n([expr ($i+2)%$N]) 1Mb 10ms DropTail
    }
    if {$i==5} {
        $ns duplex-link $n($i) $n([expr ($i+2)%$N]) 1Mb 10ms DropTail
    }
    if {$i!=4} && {$i!=5} {
        $ns duplex-link $n($i) $n([expr ($i+1)%$N]) 1Mb 10ms DropTail
    }
}

set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n(0) $tcp
set ftp [new Application/FTP]
$ftp attach-agent $tcp

$ftp set packetSize_ 500
$ftp set interval_ 0.005
```

```
set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n(0) $tcp
set ftp [new Application/FTP]
$ftp attach-agent $tcp

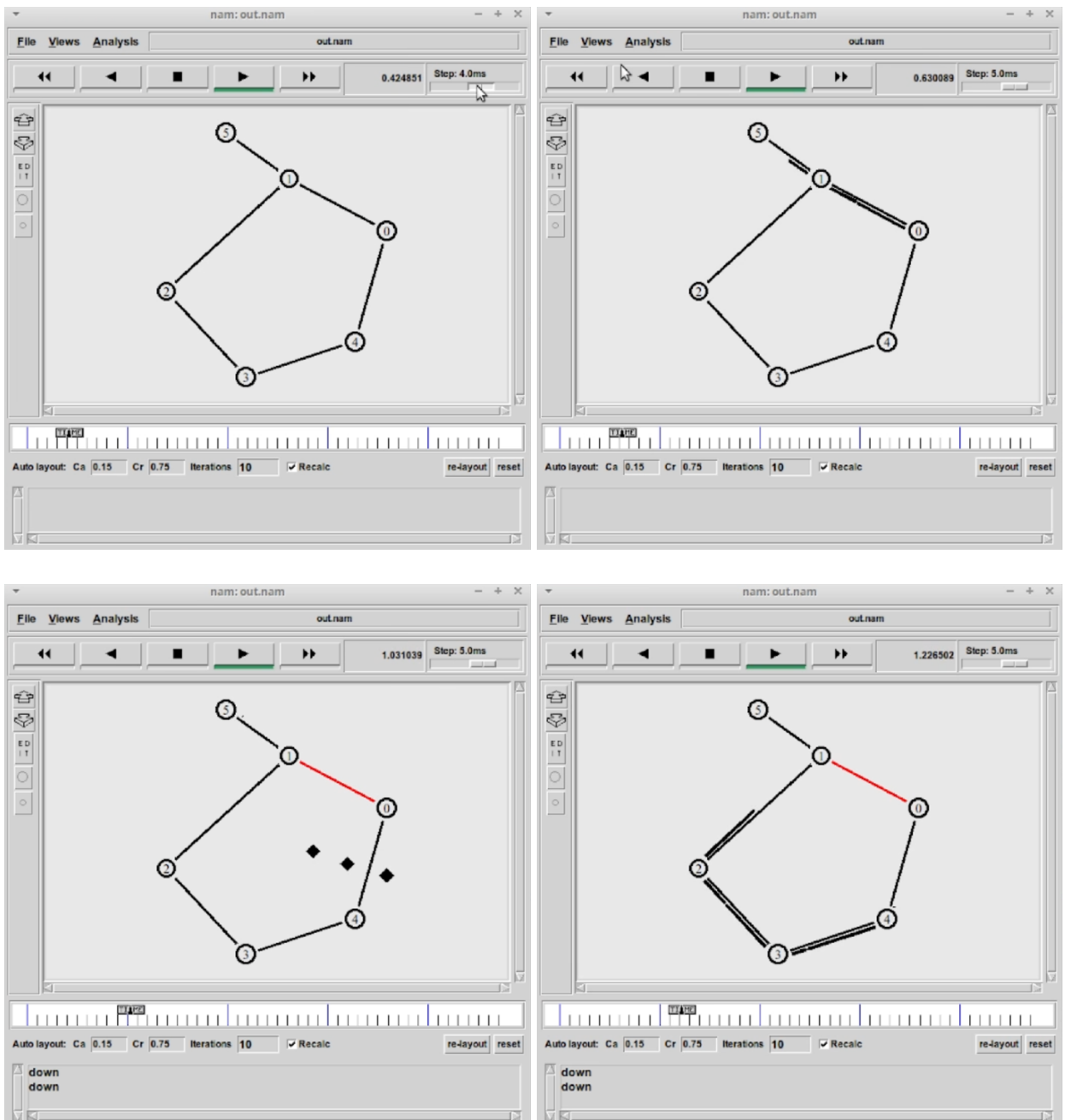
$ftp set packetSize_ 500
$ftp set interval_ 0.005

set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n(5) $sink

$ns connect $tcp $sink

$ns at 0.5 "$ftp start"
$ns rtmodel-at 1.0 down $n(0) $n(1)
$ns rtmodel-at 2.0 up $n(0) $n(1)
$ns at 4.5 "$scbr0 stop"
$ns at 5.0 "finish"

$ns run
```



В данной сети передача данных от узла  $n(0)$  до узла  $n(5)$  осуществляется по кратчайшему пути, через узел  $n(1)$ , с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами  $n(0)$  и  $n(1)$ , а при разрыве соединения маршрут передачи данных должен измениться на резервный. Подробные изменения со временем можно наблюдать на представленных выше рисунках.

## ВЫВОД:

При выполнении данной лабораторной работы были приобретены навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также был проведён анализ полученных результатов моделирования.