

Лабораторная работа № 1

простые модели компьютеров сети

Джахангиров Илгар Залид оглы

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	вывод	13

Список иллюстраций

3.1	таблица	7
3.2	таблица	8
3.3	Мониторинг очереди в визуализаторе nam	9
3.4	Передача данных по кратчайшему пути сети с кольцевой топологией	10
3.5	Передача данных по сети с кольцевой топологией в случае разрыва соединения	11
3.6	Изменённая кольцевая топология сети	12

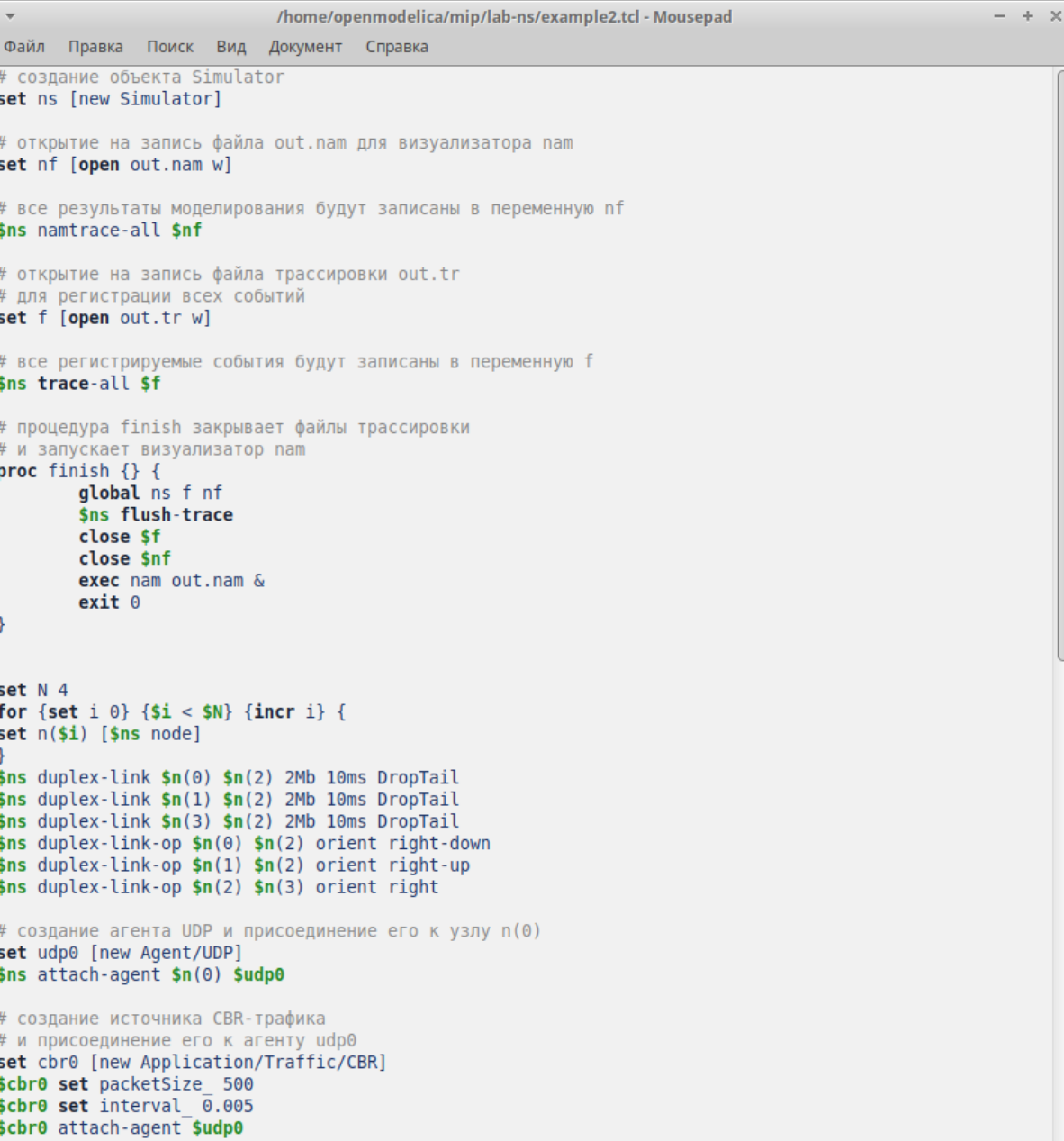
1 Цель работы

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

2 Задание

1. создать Шаблон сценария для NS-2
2. Простой пример описания топологии сети, состоящей из двух

3 Выполнение лабораторной работы



The screenshot shows a text editor window titled "/home/openmodelica/mip/lab-ns/example2.tcl - Mousepad". The window contains a TCL script for NetSim. The script includes comments in Russian and various commands for creating a simulator, opening files for output, setting up tracing, and creating network nodes and agents. The script is as follows:

```
# 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
}

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
```

Рис. 3.1: таблица

```

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

```

Рис. 3.2: таблица

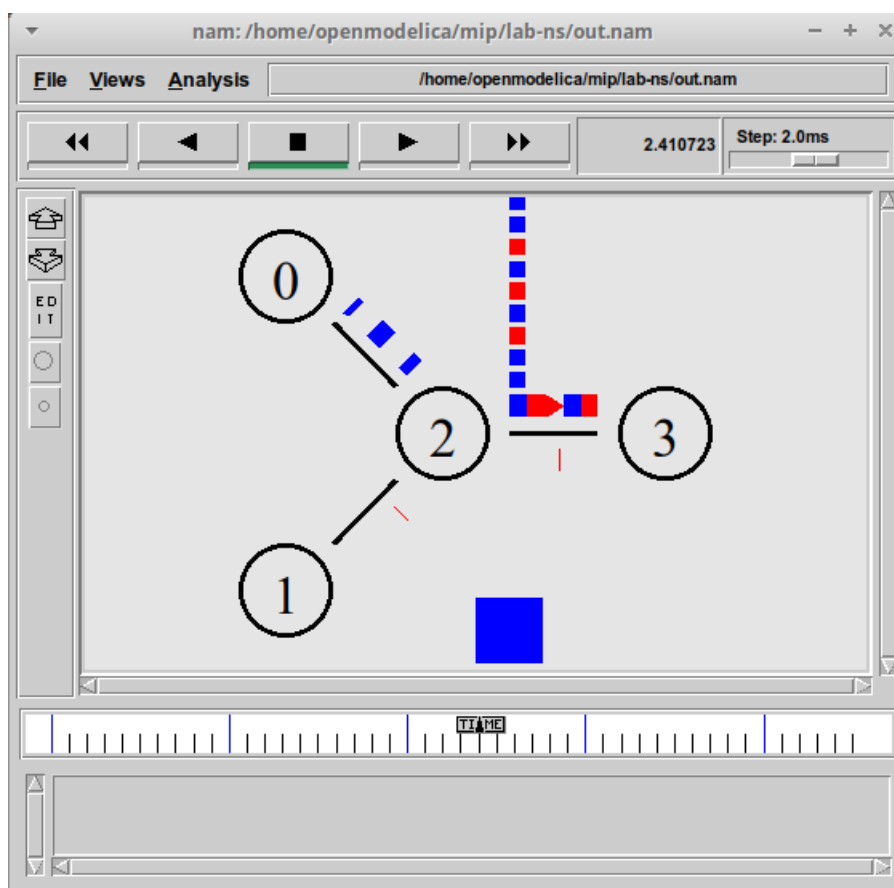


Рис. 3.3: Мониторинг очереди в визуализаторе nam

Сохранив изменения в отредактированном файле и запустив симулятор, получим анимированный результат моделирования (рис. 1.3). При запуске скрипта можно заметить, что по соединениям между узлами $n(0)$ – $n(2)$ и $n(1)$ – $n(2)$ к узлу $n(2)$ передаётся данных больше, чем способно передаваться по соединению от узла $n(2)$ к узлу $n(3)$. Действительно, мы передаём 200 пакетов в секунду от каждого источника данных в узлах $n(0)$ и $n(1)$, а каждый пакет имеет размер 500 байт. Таким образом, полоса каждого соединения 0, 8 Mb, а суммарная — 1, 6 Mb. Но соединение $n(2)$ – $n(3)$ имеет полосу лишь 1 Mb. Следовательно, часть пакетов должна теряться. В окне аниматора можно видеть пакеты в очереди, а также те пакеты, которые отбрасываются при переполнении

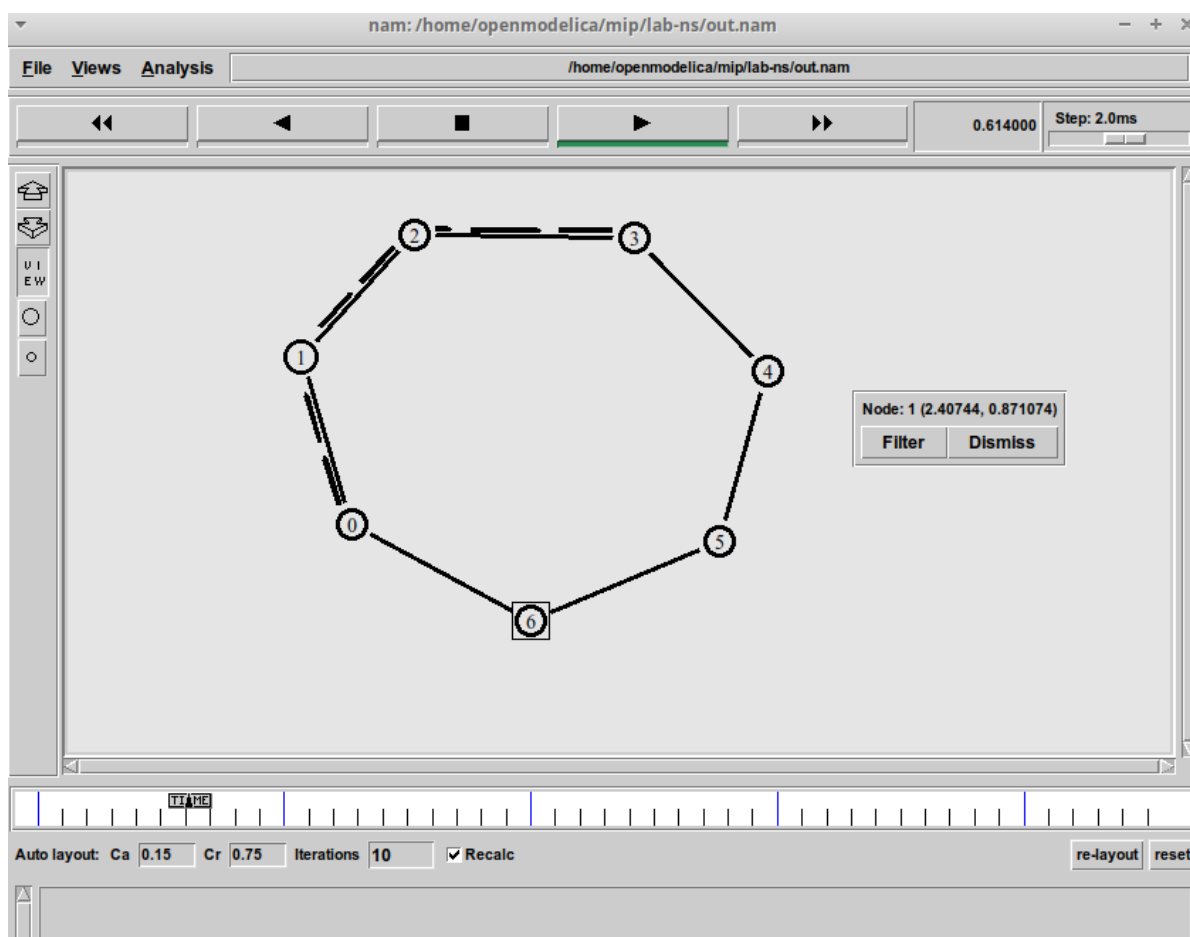


Рис. 3.4: Передача данных по кратчайшему пути сети с кольцевой топологией

Постановка задачи. 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 $n(2)$;

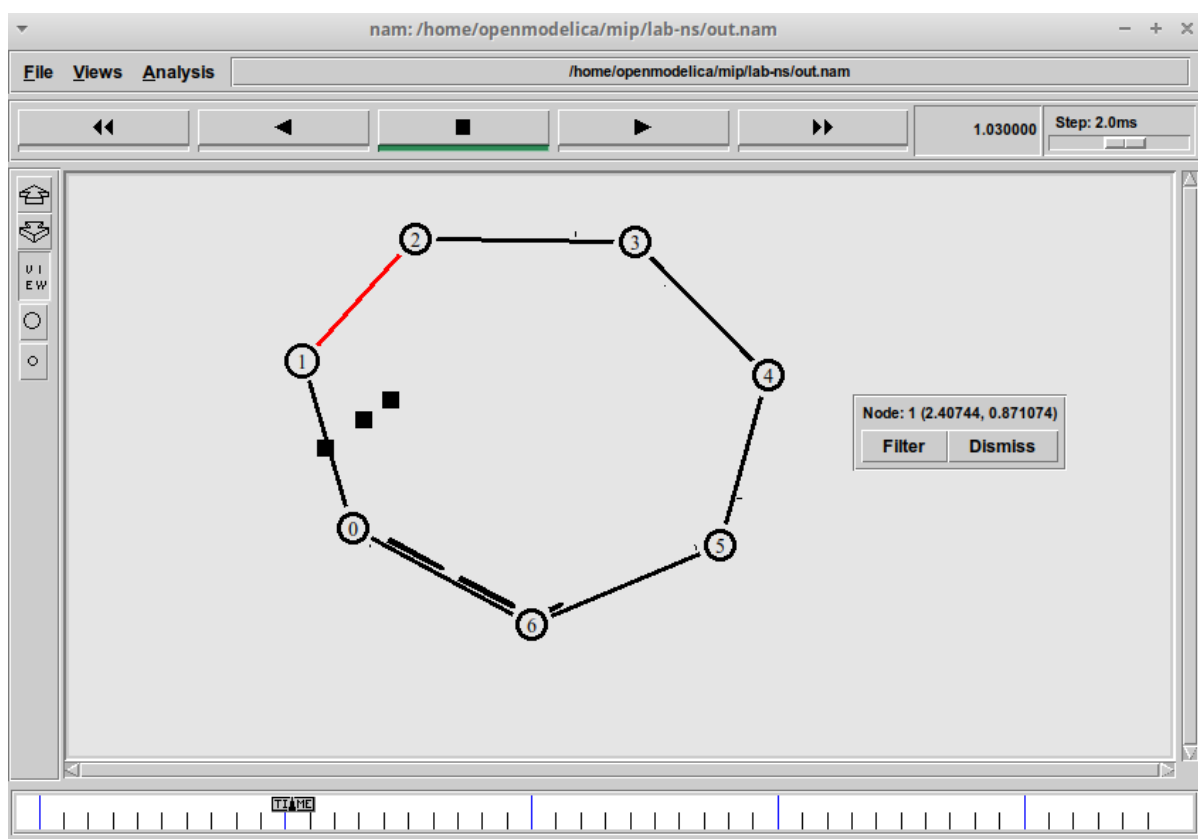


Рис. 3.5: Передача данных по сети с кольцевой топологией в случае разрыва соединения

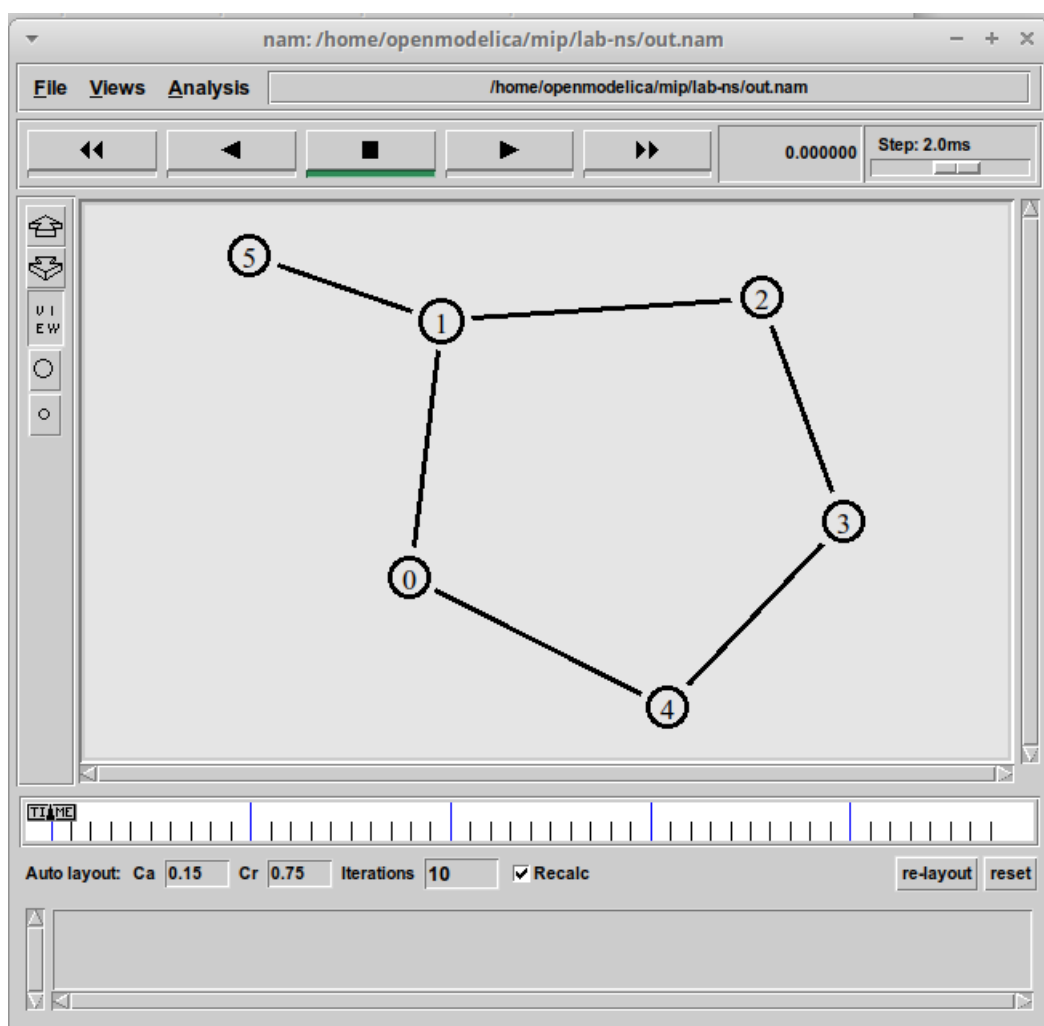


Рис. 3.6: Изменённая кольцевая топология сети

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.

4 ВЫВОД