

Лабораторная работа № 4

Имитационное моделирование

Королёв Иван Андреевич

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	8
4	Выполнение лабораторной работы	9
4.1	Разработка имитационной модели по схеме	9
4.2	Полный код реализованной модели	11
4.3	График изменения размера окна TCP (в Xgraph и в GNUPlot)	17
4.4	Построить график изменения длины очереди и средней длины очереди на первом маршрутизаторе.	20
4.5	Демонстрация работы модели	22
5	Выводы	24
	Список литературы	25

Список иллюстраций

4.1 Simulator	9
4.2 Узлы	10
4.3 Мониторинг	10
4.4 Формирование файла	10
4.5 finish	11
4.6 at-события	11
4.7 График изменения размера окна TCP в Xgraph	18
4.8 График изменения размера окна TCP в Xgraph	19
4.9 Реализация графика в GNUPlot	19
4.10 График изменения размера окна TCP в GNUPlot	20
4.11 Изменение размера длины очереди	21
4.12 Изменение размера средней длины очереди	22
4.13 Передача пакетов	23
4.14 Сброс очереди	23

Список таблиц

1 Цель работы

Закрепить и продемонстрировать навыки самостоятельной разработки имитационной модели в пакете NS-2 и построении графиков.

2 Задание

1. По приведенной схеме разработать имитационную модель в пакете NS-2.

Схема:

- сеть состоит из N TCP-источников, N TCP-приёмников, двух маршрутизаторов $R1$ и $R2$ между источниками и приёмниками (N — не менее 20);
 - между TCP-источниками и первым маршрутизатором установлены дуплексные соединения с пропускной способностью 100 Мбит/с и задержкой 20 мс очередью типа DropTail;
 - между TCP-приёмниками и вторым маршрутизатором установлены дуплексные соединения с пропускной способностью 100 Мбит/с и задержкой 20 мс очередью типа DropTail;
 - между маршрутизаторами установлено симплексное соединение ($R1-R2$) с пропускной способностью 20 Мбит/с и задержкой 15 мс очередью типа RED, размером буфера 300 пакетов; в обратную сторону — симплексное соединение ($R2-R1$) с пропускной способностью 15 Мбит/с и задержкой 20 мс очередью типа DropTail;
 - данные передаются по протоколу FTP поверх TCP Reno;
 - параметры алгоритма RED: $q_{min} = 75$, $q_{max} = 150$, $qw = 0,002$, $p_{max} = 0.1$;
 - максимальный размер TCP-окна 32; размер передаваемого пакета 500 байт; время моделирования — не менее 20 единиц модельного времени.
2. Построить график изменения размера окна TCP (в Xgraph и в GNUPlot);
 3. Построить график изменения длины очереди и средней длины очереди на

первом маршрутизаторе.

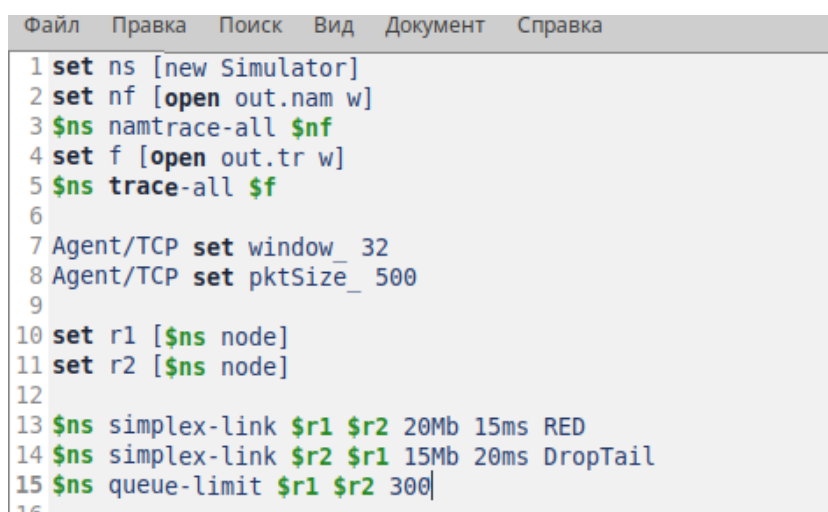
3 Теоретическое введение

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. NS-2 является объектно-ориентированным программным обеспечением. Его ядро реализовано на языке C++. В качестве интерпретатора используется язык скриптов (сценариев) OTcl (Object oriented Tool Command Language). NS-2 полностью поддерживает иерархию классов C++ и подобную иерархию классов интерпретатора OTcl. Обе иерархии обладают идентичной структурой, т.е. существует однозначное соответствие между классом одной иерархии и таким же классом другой. Объединение для совместного функционирования C++ и OTcl производится при помощи TclCl (Classes Tcl). В случае, если необходимо реализовать какую-либо специфическую функцию, не реализованную в NS-2 на уровне ядра, для этого используется код на C++.

4 Выполнение лабораторной работы

4.1 Разработка имитационной модели по схеме

Создание объекта типа Simulator. Затем создаём переменную nf и указываем, что требуется открыть на запись nam-файл для регистрации выходных результатов моделирования. Далее создаём переменную f и открываем на запись файл трассировки для регистрации всех событий модели. Установка максимального размера окна и размера передаваемого пакета. Создание двух маршрутизаторов и установка между маршрутизаторами симплексного соединения. (рис. 4.1).



```
Файл  Правка  Поиск  Вид  Документ  Справка
1 set ns [new Simulator]
2 set nf [open out.nam w]
3 $ns namtrace-all $nf
4 set f [open out.tr w]
5 $ns trace-all $f
6
7 Agent/TCP set window_ 32
8 Agent/TCP set pktSize_ 500
9
10 set r1 [$ns node]
11 set r2 [$ns node]
12
13 $ns simplex-link $r1 $r2 20Mb 15ms RED
14 $ns simplex-link $r2 $r1 15Mb 20ms DropTail
15 $ns queue-limit $r1 $r2 300
16
```

Рис. 4.1: Simulator

Создание и соединение узлов. (рис. 4.2).

```

10
17 set N 20
18
19 for {set i 0} {$i < $N} {incr i} {
20     set n1($i) [$ns node]
21     set n2($i) [$ns node]
22     $ns duplex-link $n1($i) $r1 100Mb 20ms DropTail
23     $ns duplex-link $n2($i) $r2 100Mb 20ms DropTail
24     set tcp($i) [$ns create-connection TCP/Reno $n1($i) TCPSink $n2($i) $i]
25     set ftp($i) [$tcp($i) attach-source FTP]
26 }
27

```

Рис. 4.2: Узлы

Мониторинг размера окна TCP и очереди (рис. 4.3).

```

27
28 set windowVsTimeOne [open WindowVsTimeRenoOne w]
29 puts $windowVsTimeOne "0.Color: Black"
30 set windowVsTimeAll [open WindowVsTimeRenoAll w]
31 puts $windowVsTimeAll "0.Color: Black"
32 set qmon [$ns monitor-queue $r1 $r2 [open qm.out w] 0.1];
33 [$ns link $r1 $r2] queue-sample-timeout;
34
35 set redq [$ns link $r1 $r2] queue]
36 $redq set thresh_ 75
37 $redq set maxthresh_ 150
38 $redq set q_weight_ 0.002
39 $redq set linterm_ 10
40 set tchan_ [open all.q w]
41 $redq trace curq_
42 $redq trace ave_
43

```

Рис. 4.3: Мониторинг

Формирование файла с данными о размере окна TCP (рис. 4.4).

```

44 proc plotWindow {tcpSource file} {
45     global ns
46     set time 0.01
47     set now [$ns now]
48     set cwnd [$tcpSource set cwnd_]
49     puts $file "$now $cwnd"
50     $ns at [expr $now+$time] "plotWindow $tcpSource $file"
51 }
52

```

Рис. 4.4: Формирование файла

Процедура finish, которая завершает симуляцию и запускает анализ результатов (рис. 4.5).

```

52
53 proc finish {} {
54     global tchan
55     set awkCode {
56         {
57             if ($1 == "Q" && NF>2) {
58                 print $2, $3 >> "temp.q";
59                 set end $2
60             }
61             else if ($1 == "a" && NF>2)
62                 print $2, $3 >> "temp.a";
63         }
64     }
65     exec rm -f temp.q temp.a
66     exec touch temp.a temp.q
67
68     set f [open temp.q w]
69     close $f
70
71     set f [open temp.a w]
72     close $f
73
74
75     exec awk $awkCode all.q
76     exec xgraph -bb -tk -x time -t "TCPrenoCWND" WindowVsTimeRenoOne &
77     exec xgraph -bb -tk -x time -t "TCPrenoCWND" WindowVsTimeRenoAll &
78     exec xgraph -bb -tk -x time -y queue temp.q &
79     exec xgraph -bb -tk -x time -y queue temp.a &
80     exec nam out.nam &
81     exit 0
82 }

```

Рис. 4.5: finish

Добавление at-событий и запуск модели. (рис. 4.6).

```

84 $redq attach $tchan_
85 for {set i 0} {$i < $N} {incr i} {
86     $ns at 0.0 "$ftp($i) start"
87     $ns at 0.0 "plotWindow $tcp($i) $windowVsTimeAll"
88 }
89 $ns at 0.0 "plotWindow $tcp(1) $windowVsTimeOne"
90 $ns at 20.0 "finish"
91 $ns run

```

Рис. 4.6: at-события

4.2 Полный код реализованной модели

Создание объекта симулятора

```
set ns [new Simulator]
```

Открытие файла out.nam для записи данных визуализации NAM

```
set nf [open out.nam w]
```

```
# Настройка записи трассировочных данных для визуализатора NAM в файл out.nam
$ns namtrace-all $nf
```

```
# Открытие файла out.tr для записи событий симуляции
set f [open out.tr w]
```

```
# Настройка записи всех событий симуляции в файл out.tr
$ns trace-all $f
```

```
# Установка параметров TCP-агента: размер окна TCP равен 32
Agent/TCP set window_ 32
```

```
# Установка размера пакетов TCP на 500 байт
Agent/TCP set pktSize_ 500
```

```
# Определение процедуры finish, которая завершает симуляцию и запускает анализ результатов
proc finish {} {
    global tchan_
```

```
    # Код на AWK для обработки выходных данных
```

```
    set awkCode {
        {
            if ($1 == "Q" && NF>2) {
                print $2, $3 >> "temp.q";
                set end $2
            }
            else if ($1 == "a" && NF>2)
                print $2, $3 >> "temp.a";
        }
    }
```

```

    }
}

# Удаление временных файлов, если они существуют
exec rm -f temp.q temp.a

# Создание пустых файлов temp.q и temp.a
exec touch temp.a temp.q

# Добавление заголовка цвета для файла temp.q
set f [open temp.q w]
puts $f "0.Color: Purple"
close $f

# Добавление заголовка цвета для файла temp.a
set f [open temp.a w]
puts $f "0.Color: Purple"
close $f

# Запуск обработки файлов через AWK
exec awk $awkCode all.q

# Запуск графиков xgraph для визуализации окна TCP и очереди
exec xgraph -fg pink -bg purple -bb -tk -x time -t "TCPRenoCWND" WindowVsTimeReno0
exec xgraph -fg pink -bg purple -bb -tk -x time -t "TCPRenoCWND" WindowVsTimeRenoA
exec xgraph -bb -tk -x time -y queue temp.q &
exec xgraph -bb -tk -x time -y queue temp.a &

# Запуск NAM для визуализации симуляции

```

```

exec nam out.nam &

# Завершение работы симулятора
exit 0
}

# Определение процедуры plotWindow для мониторинга размера окна TCP
proc plotWindow {tcpSource file} {
    global ns
    set time 0.01

    # Получение текущего времени симуляции
    set now [$ns now]

    # Получение текущего размера окна TCP
    set cwnd [$tcpSource set cwnd_]

    # Запись значения окна TCP в файл
    puts $file "$now $cwnd"

    # Запланировать повторное выполнение через 0.01 секунды
    $ns at [expr $now+$time] "plotWindow $tcpSource $file"
}

# Создание двух узлов маршрутизаторов
set r1 [$ns node]
set r2 [$ns node]

# Создание симплексных каналов с различными параметрами

```

```

$ns simplex-link $r1 $r2 20Mb 15ms RED # Пропускная способность 20 Мбит/с, задержка 1
$ns simplex-link $r2 $r1 15Mb 20ms DropTail # Пропускная способность 15 Мбит/с, задер

# Ограничение размера очереди между r1 и r2 до 300 пакетов
$ns queue-limit $r1 $r2 300

# Количество создаваемых узлов
set N 30

# Цикл создания 30 TCP-соединений
for {set i 0} {$i < $N} {incr i} {
    # Создание узлов отправителя и получателя
    set n1($i) [$ns node]
    set n2($i) [$ns node]

    # Создание дуплексных каналов с параметрами
    $ns duplex-link $n1($i) $r1 100Mb 20ms DropTail
    $ns duplex-link $n2($i) $r2 100Mb 20ms DropTail

    # Создание TCP-соединения между узлами
    set tcp($i) [$ns create-connection TCP/Reno $n1($i) TCPSink $n2($i) $i]

    # Привязка TCP-источника к FTP-приложению
    set ftp($i) [$tcp($i) attach-source FTP]
}

# Открытие файлов для записи данных о размере окна TCP
set windowVsTimeOne [open WindowVsTimeRenoOne w]
puts $windowVsTimeOne "0.Color: White"

```

```

set windowVsTimeAll [open WindowVsTimeRenoAll w]
puts $windowVsTimeAll "0.Color: White"

# Мониторинг очереди между r1 и r2 с интервалом 0.1 секунды
set qmon [$ns monitor-queue $r1 $r2 [open qm.out w] 0.1];

# Запуск таймера выборки для очереди
[$ns link $r1 $r2] queue-sample-timeout;

# Получение ссылки на очередь RED и настройка параметров RED-буфера
set redq [[$ns link $r1 $r2] queue]
$redq set thresh_ 75      # Минимальный порог очереди
$redq set maxthresh_ 150  # Максимальный порог очереди
$redq set q_weight_ 0.002 # Вес очереди
$redq set linterm_ 10     # Линейный интервал

# Открытие файла для записи данных об очереди
set tchan_ [open all.q w]

# Настройка трассировки параметров очереди RED
$redq trace curq_  # Текущий размер очереди
$redq trace ave_   # Средний размер очереди

# Привязка файла к RED-очереди
$redq attach $tchan_

# Запуск всех TCP-источников и мониторинг окон TCP
for {set i 0} {$i < $N} {incr i} {
    $ns at 0.0 "$ftp($i) start" # Запуск передачи FTP
}

```



```

    $ns at 0.0 "plotWindow $tcp($i) $windowVsTimeAll" # Мониторинг окна TCP
}

# Мониторинг окна TCP для конкретного TCP-соединения
$ns at 0.0 "plotWindow $tcp(1) $windowVsTimeOne"

# Планирование завершения симуляции через 20 секунд
$ns at 20.0 "finish"

# Запуск симуляции
$ns run

```

4.3 График изменения размера окна TCP (в Xgraph и в GNUPlot)

График изменения размера окна TCP на линке 1-го источника в Xgraph. Текущий размер очереди показывает высокие колебания. Максимальное значение размера окна TCP равняется 32, минимальное значение примерно 1. (рис. 4.7).

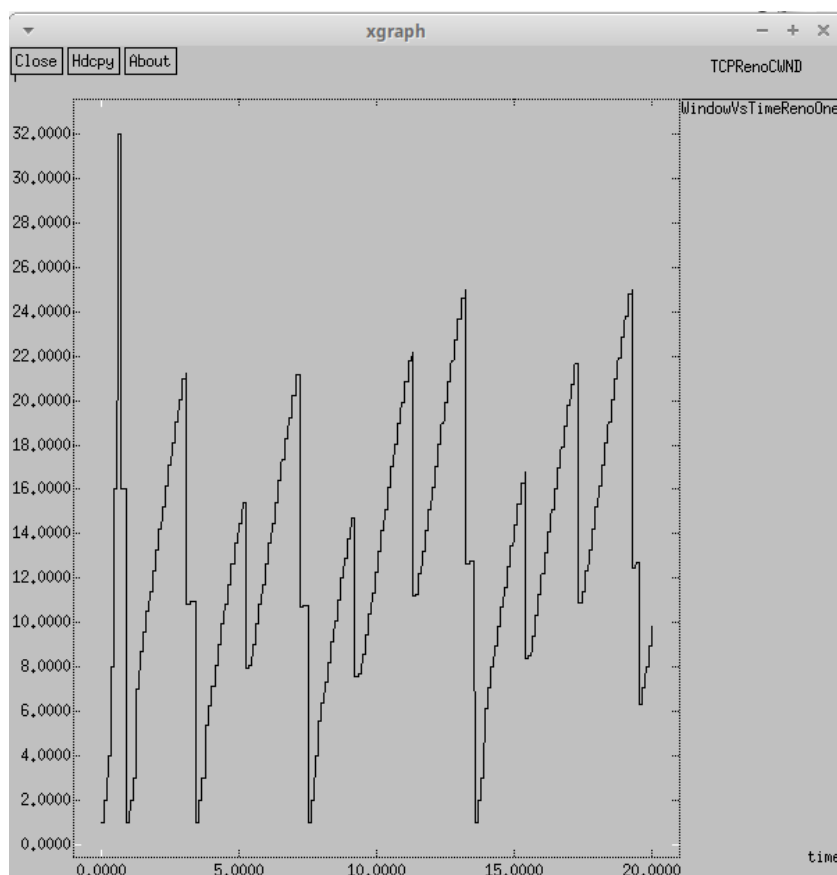


Рис. 4.7: График изменения размера окна TCP в Xgraph

График изменения размера окна TCP R на всех источниках при $N=20$ в Xgraph. Текущий размер очереди показывает высокие колебания. Максимальное значение размера окна TCP равняется чуть больше 32, минимальное значение примерно 1. (рис. 4.8).

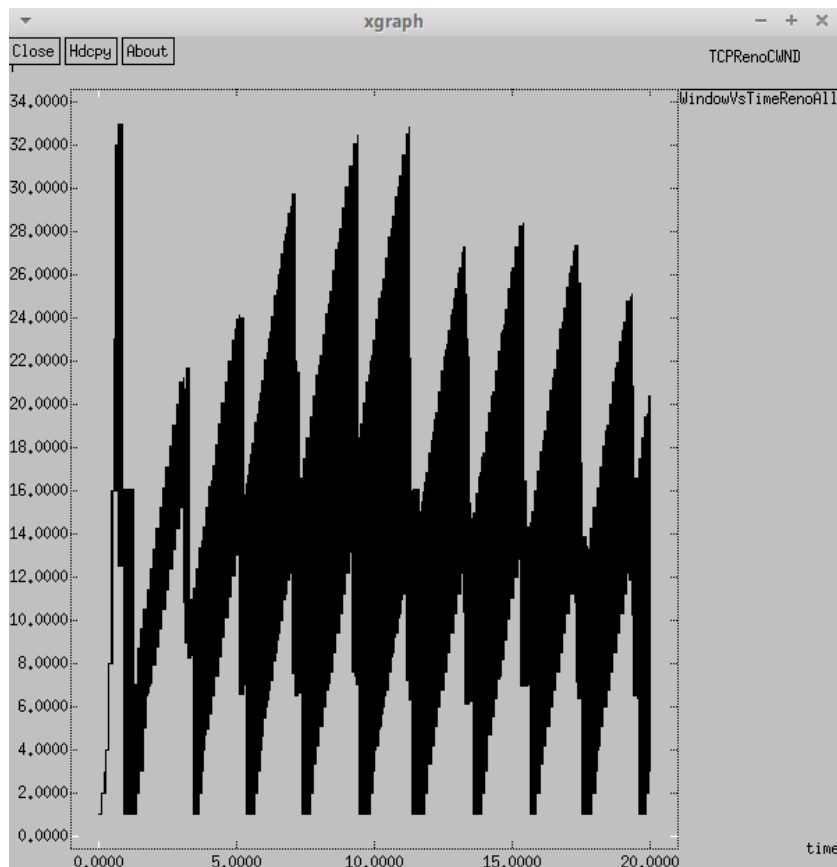


Рис. 4.8: График изменения размера окна TCP в Xgraph

График изменения размера окна TCP в GNUPlot. Текущий размер очереди показывает высокие колебания. Максимальное значение размера окна TCP равняется 32, минимальное значение примерно 1. И описание кода (рис. 4.9), (рис. 4.10)

```

/home/openmodelica/mip/lab-ns/lab04/graph_plot.gpi - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка
1 #!/usr/bin/gnuplot -persist
2 # задаём текстовую кодировку,
3 # тип терминала, тип и размер шрифта
4
5 set encoding utf8
6 set term pdfcairo font "Arial,9"
7
8 set out 'tcp.pdf'
9
10 set title "График изменения размера окна TCP"
11
12 set style line 2
13
14 set xlabel "Время"
15 set ylabel "Размер окна"
16
17 plot "WindowVsTimeRenoOne" using 1:2 with lines title "Размер окна TCP"

```

Рис. 4.9: Реализация графика в GNUPlot

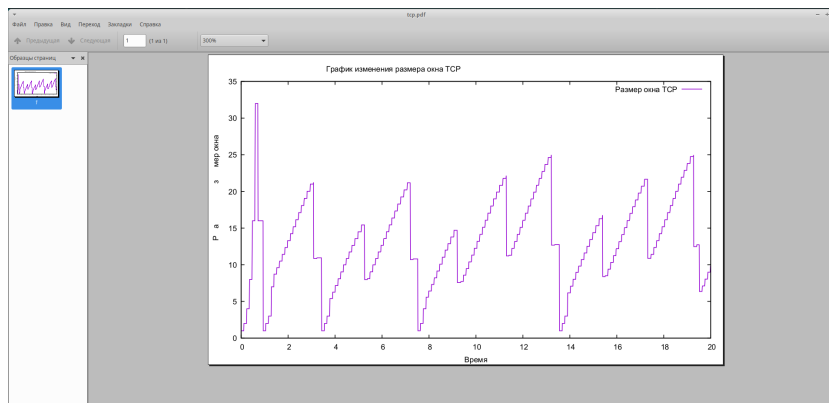


Рис. 4.10: График изменения размера окна TCP в GNUPlot

4.4 Построить график изменения длины очереди и средней длины очереди на первом маршрутизаторе.

Изменение размера длины очереди на линке (R1–R2) при $N=20$, $q_{\min} = 75$, $q_{\max} = 150$. Максимальное значение около 150, минимальное значение ноль. (рис. 4.11)

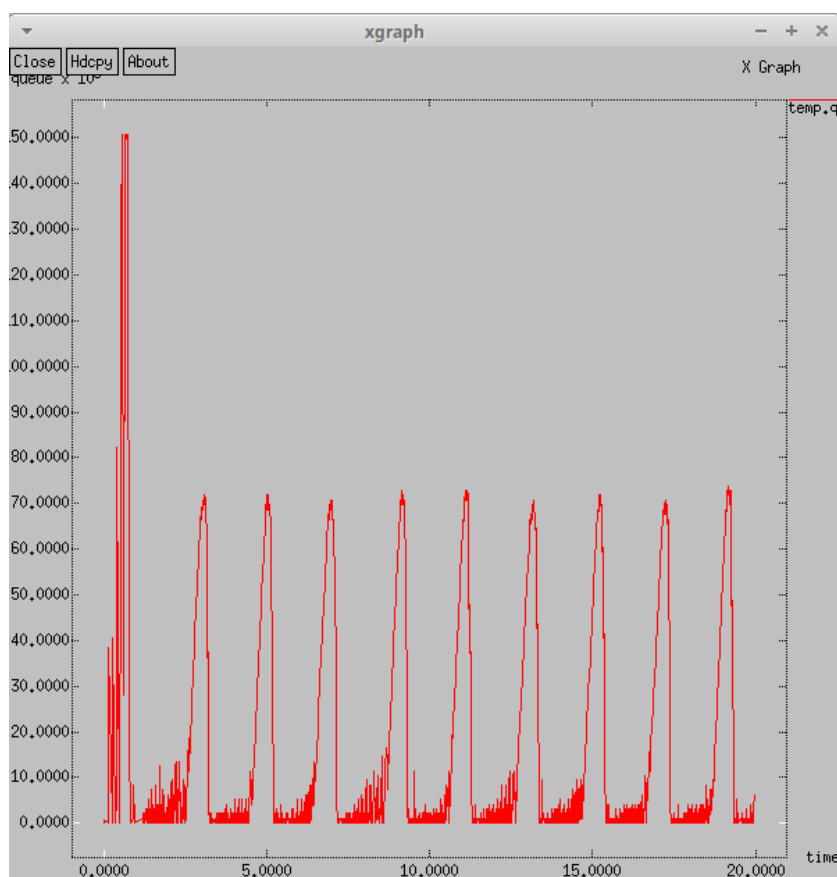


Рис. 4.11: Изменение размера длины очереди

Изменение размера средней длины очереди на линке (R1–R2) при $N=20$, $q_{\min} = 75$, $q_{\max} = 150$ Максимальное значение около 110, минимальное значение ноль. (рис. 4.12)

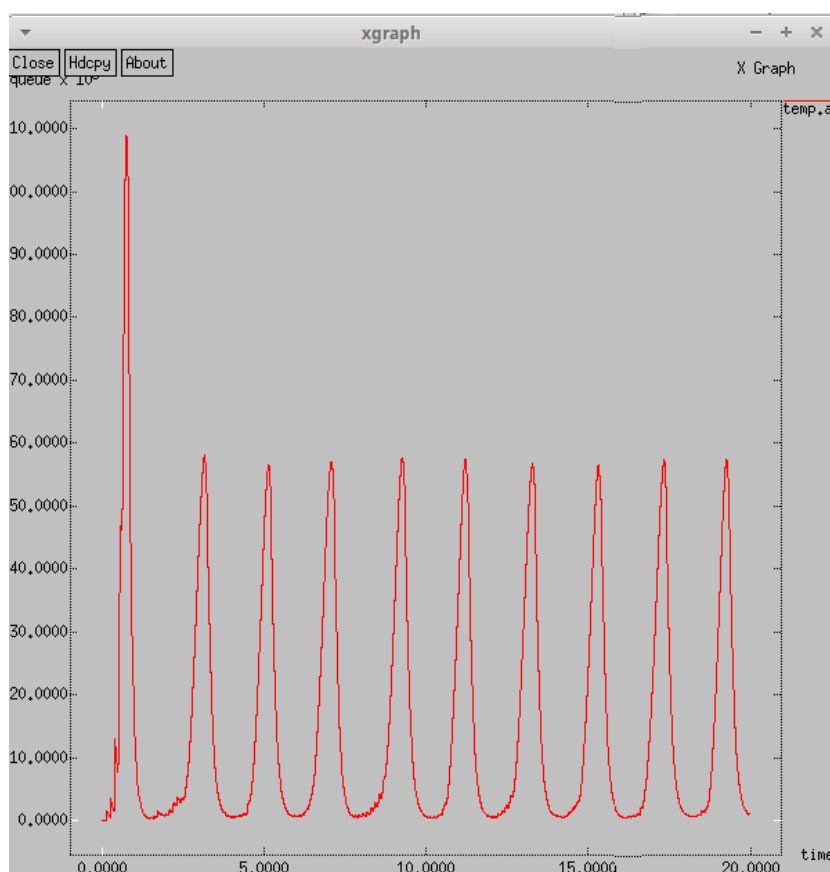


Рис. 4.12: Изменение размера средней длины очереди

4.5 Демонстрация работы модели

Передача пакетов из узлов к маршрутизатору ноль. От маршрутизатора ноль пакеты идут к маршрутизатору 1 и распределяются от него по узлам, соединенным с ним. (рис. 4.13).

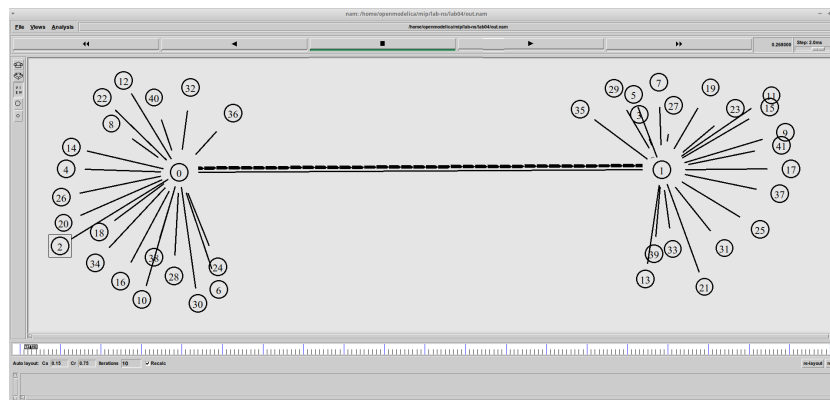


Рис. 4.13: Передача пакетов

При переполнении очереди происходит сброс (рис. 4.14).



Рис. 4.14: Сброс очереди

5 Выводы

Закрепл и продемонстрировал навыки самостоятельной разработки имитационной модели в пакете NS-2 и построил графики.

Список литературы