

Отчёт по лабораторной работе №3

Дисциплина: Имитационное моделирование

Ганина Таисия Сергеевна, НФИбд-01-22

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Листинги	12
6	Выводы	16
	Список литературы	17

Список иллюстраций

4.1	Теоретическая вероятность потери и средняя длина очереди . . .	8
4.2	Создание файла, сделала его исполняемым	9
4.3	Код в файле	9
4.4	График	10

Список таблиц

1 Цель работы

Исследование работы системы массового обслуживания, моделирование передачи пакетов в сети с использованием симулятора NS-2, анализ зависимости вероятности потерь и длины очереди от интенсивности входного потока.

2 Задание

1. Реализация модели на NS-2.
2. График в GNUplot.

3 Теоретическое введение

M|M|1 — однолинейная СМО с накопителем бесконечной ёмкости. Поступающий поток заявок — пуассоновский с интенсивностью λ . Времена обслуживания заявок — независимые в совокупности случайные величины, распределённые по экспоненциальному закону с параметром μ .

Коэффициент загрузки системы определяется формулой:

$$\rho = \frac{\lambda}{\mu}$$

где:

- λ — интенсивность поступления пакетов.
- μ — интенсивность обработки пакетов.

Вероятность потери пакетов вычисляется как:

$$P_{\text{loss}} = \frac{(1 - \rho) \cdot \rho^{qsize}}{1 - \rho^{qsize+1}}$$

Средняя длина очереди:

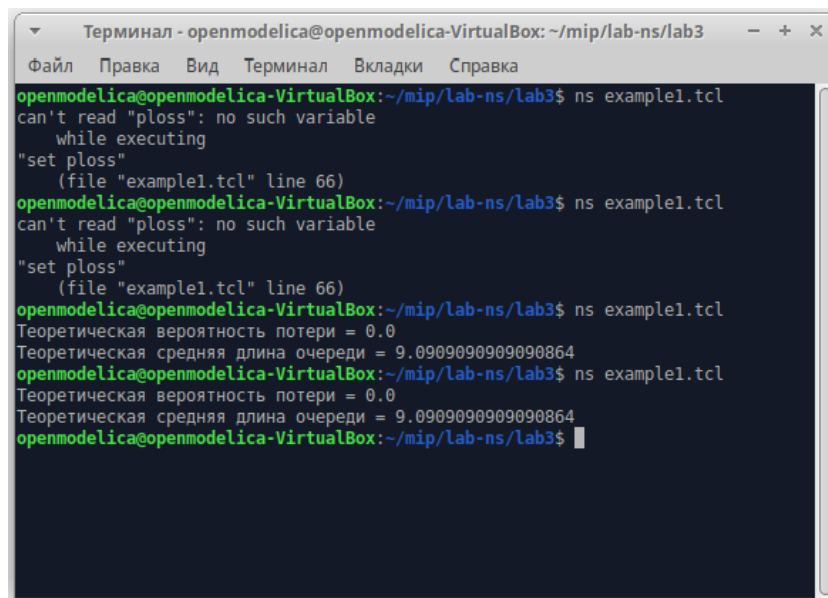
$$L_q = \frac{\rho^2}{1 - \rho}$$

В данной лабораторной работе с помощью **NS-2** моделируется передача пакетов по каналу связи между двумя узлами. Анализируются потери пакетов и длина очереди при разных параметрах системы.

4 Выполнение лабораторной работы

1. Реализация модели на NS-2.

Результат выполнения кода: (рис. 4.1). Так как очередь мы задали очень большую (set qsize 100000), а загружается она в среднем на 9 позиций, то и вероятность потери равна 0.0.



```
Терминал - openmodelica@openmodelica-VirtualBox: ~/mip/lab-ns/lab3
Файл  Правка  Вид  Терминал  Вкладки  Справка
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ ns example1.tcl
can't read "ploss": no such variable
while executing
"set ploss"
(file "example1.tcl" line 66)
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ ns example1.tcl
can't read "ploss": no such variable
while executing
"set ploss"
(file "example1.tcl" line 66)
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ ns example1.tcl
Теоретическая вероятность потери = 0.0
Теоретическая средняя длина очереди = 9.0909090909090864
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ ns example1.tcl
Теоретическая вероятность потери = 0.0
Теоретическая средняя длина очереди = 9.0909090909090864
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$
```

Рис. 4.1: Теоретическая вероятность потери и средняя длина очереди

2. График в GNUplot.

В каталоге с проектом создала отдельный файл graph_plot (рис. 4.2): touch graph_plot


```
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ touch graph_plot
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ ls
example1.tcl  graph_plot  out.tr  qm.out
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ vim graph_plot

Command 'vim' not found, but can be installed with:

sudo apt install vim          # version 2:8.1.2269-1ubuntu5.7, or
sudo apt install vim-tiny     # version 2:8.1.2269-1ubuntu5.7
sudo apt install vim-athena   # version 2:8.1.2269-1ubuntu5.7
sudo apt install vim-gtk3     # version 2:8.1.2269-1ubuntu5.7
sudo apt install vim-nox      # version 2:8.1.2269-1ubuntu5.7

openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ chmod +x graph_plot
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ ls -l
итого 5612
-rw-rw-r-- 1 openmodelica openmodelica 2725 фев 22 12:35 example1.tcl
-rwxrwxr-x 1 openmodelica openmodelica 929 фев 22 12:42 graph_plot
-rw-rw-r-- 1 openmodelica openmodelica 4830276 фев 22 12:35 out.tr
-rw-rw-r-- 1 openmodelica openmodelica 902143 фев 22 12:35 qm.out
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns/lab3$ ./graph_plot
```

Рис. 4.2: Создание файла, сделала его исполняемым

Открыла его на редактирование и добавила следующий код(рис. 4.3), обращая внимание на синтаксис GNUplot.

```
1#!/usr/bin/gnuplot -persist
2# задаём текстовую кодировку,
3# тип терминала, тип и размер шрифта
4set encoding utf8
5set term pngcairo font "Arial,9"
6# задаём выходной файл графика
7set out 'qm.png'
8# задаём название графика
9set title "График поведения длины очереди"
10# подписи осей графика
11set xlabel "t"
12set ylabel "Пакеты"
13# построение графика, используя значения
14# 1-го и 5-го столбцов файла qm.out
15
16
17plot "qm.out" using ($1):($5) with lines lt rgb "red" title "Размер очереди (в пакетах)", \
18      "qm.out" using ($1):($5) smooth csplines lt rgb "blue" title "Приближение сплайном", \
19      "qm.out" using ($1):($5) smooth bezier lt rgb "green" title "Приближение Безье "
20|
```

Рис. 4.3: Код в файле

Результат выполнения кода видно на изображении. (рис. 4.4)

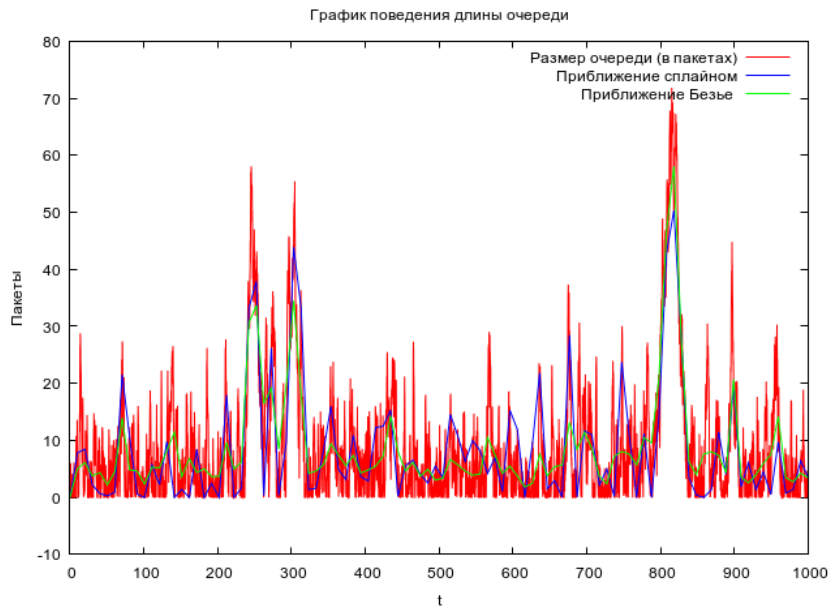


Рис. 4.4: График

На графике изображена динамика изменения длины очереди в зависимости от времени.

- **Оси:**

- Ось X (горизонтальная): “t” (время). Показывает время в условных единицах.
- Ось Y (вертикальная): “Пакеты”. Показывает количество пакетов в очереди в данный момент времени.

- **Линии:**

- Красная линия: “Размер очереди (в пакетах)”. Это фактический размер очереди в каждый момент времени, отражающий мгновенную загрузку очереди.
- Синяя линия: “Приближение сплайном”. Это сглаженная версия данных о размере очереди, полученная с использованием сплайн-интерполяции. Сплаины помогают выделить общие тенденции и сгладить случайные колебания.

- Зелёная линия: “Приближение Безье”. Еще один метод сглаживания данных, использующий кривые Безье. Как и сплайны, он помогает увидеть общую картину без резких колебаний.

График показывает, что длина очереди постоянно меняется. Есть периоды с низкой загрузкой (близко к 0), и периоды с высокой загрузкой (пики до 60-70 пакетов). Приближения сплайном и Безье показывают общую тенденцию изменения длины очереди. Они помогают отфильтровать краткосрочные колебания и увидеть более плавное изменение нагрузки.

5 Листинги

1. Реализация модели на NS-2.

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

# открытие на запись файла out.tr для регистрации событий
set tf [open out.tr w]
$ns trace-all $tf

# задаём значения параметров системы
set lambda 30.0
set mu 33.0

# размер очереди для M|M|1 (для M|M|1|R: set qsize R)
set qsize 100000

# устанавливаем длительность эксперимента
set duration 1000.0

# задаём узлы и соединяем их симплексным соединением
# с полосой пропускания 100 Кб/с и задержкой 0 мс,
# очередью с обслуживанием типа DropTail
set n1 [$ns node]
set n2 [$ns node]

set link [$ns simplex-link $n1 $n2 100kb 0ms DropTail]

# наложение ограничения на размер очереди:
```

```

$ns queue-limit $n1 $n2 $qsize

# задаём распределения интервалов времени
# поступления пакетов и размера пакетов
set InterArrivalTime [new RandomVariable/Exponential]
$InterArrivalTime set avg_ [expr 1/$lambda]
set pktSize [new RandomVariable/Exponential]
$pktSize set avg_ [expr 100000.0/(8*$mu)]

# задаём агент UDP и присоединяем его к источнику,
# задаём размер пакета
set src [new Agent/UDP]
$src set packetSize_ 100000
$ns attach-agent $n1 $src

# задаём агент-приёмник и присоединяем его
set sink [new Agent/Null]
$ns attach-agent $n2 $sink
$ns connect $src $sink

# мониторинг очереди
set qmon [$ns monitor-queue $n1 $n2 [open qm.out w] 0.1]
$link queue-sample-timeout
# процедура finish закрывает файлы трассировки

proc finish {} {
    global ns tf
    $ns flush-trace
    close $tf
}

```

```

        exit 0
    }
    # процедура случайного генерирования пакетов
    proc sendpacket {} {
        global ns src InterArrivalTime pktSize
        set time [$ns now]
        $ns at [expr $time +[$InterArrivalTime value]] "sendpacket"
        set bytes [expr round ([$pktSize value])]
        $src send $bytes
    }
    # планировщик событий
    $ns at 0.0001 "sendpacket"
    $ns at $duration "finish"
    # расчет загрузки системы и вероятности потери пакетов
    set rho [expr $lambda/$mu]
    set ploss [expr (1-$rho)*pow($rho,$qsize)/(1-pow($rho,($qsize+1)))]
    puts "Теоретическая вероятность потери = $ploss"

    set aveq [expr $rho*$rho/(1-$rho)]
    puts "Теоретическая средняя длина очереди = $aveq"
    # запуск модели
    $ns run

```

2. График в GNUplot.

```

#!/usr/bin/gnuplot -persist
# задаём текстовую кодировку,
# тип терминала, тип и размер шрифта
set encoding utf8
set term pngcairo font "Arial,9"

```

```

# задаём выходной файл графика
set out 'qm.png'
# задаём название графика
set title "График поведения длины очереди"
# подписи осей графика
set xlabel "t"
set ylabel "Пакеты"
# построение графика, используя значения
# 1-го и 5-го столбцов файла qm.out

plot "qm.out" using ($1):($5) with lines
    lt rgb "red" title "Размер очереди (в пакетах)", \
"qm.out" using ($1):($5) smooth csplines
    lt rgb "blue" title "Приближение сплайном", \
"qm.out" using ($1):($5) smooth bezier
    lt rgb "green" title "Приближение Безье "

```

6 Выводы

В ходе работы была смоделирована передача пакетов в сети с заданной пропускной способностью и ограниченной очередью. Было показано, что:

- При низкой загрузке системы потери минимальны, а средняя длина очереди мала.

Список литературы

1. Руководство к лабораторной работе №3