РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Моделирование информационных процессов

Студент: Худицкий Василий Олегович

Группа: НКНбд-01-19

МОСКВА

2022 г.

Постановка задачи

1. Пример с дисциплиной RED.

   Описание моделируемой сети:

   - сеть состоит из 6 узлов;
   - между всеми узлами установлено дуплексное соединение с различными пропускной способностью и задержкой 10 мс;
   - узел r1 использует очередь с дисциплиной RED для накопления пакетов, максимальный размер которой составляет 25;
   - TCP-источники на узлах s1 и s2 подключаются к TCP-приёмнику на узле s3;
   - генераторы трафика FTP прикреплены к TCP-агентам.

2. Упражнение.

   - Измените в модели на узле s1 тип протокола TCP с Reno на Newreno, затем на Vegas. Сравните и поясните результаты.
   - Внесите изменения при отображении окон с графиками (измените цвет фона, цвет траекторий, подписи к осям, подпись траектории в легенде).

Выполнение работы

1. Пример с дисциплиной RED.

Перешёл в директорию, в которой выполняются лабораторные работы, создал файл для реализации модели на NS-2, открыл его на редактирование и добавил указанный в листинге код, запустил симулятор (рис. 1):

```
openmodelica@openmodelica-VirtualBox:~$ cd mip/lab-ns
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ touch lab2.tcl
openmodelica@openmodelica-VirtualBox:~/mip/lab-ns$ ns lab2.tcl
```

Рисунок 1Команды в терминале, Пример с дисциплиной RED

В результате получил в xgraph следующую визуализацию(рис. 2):

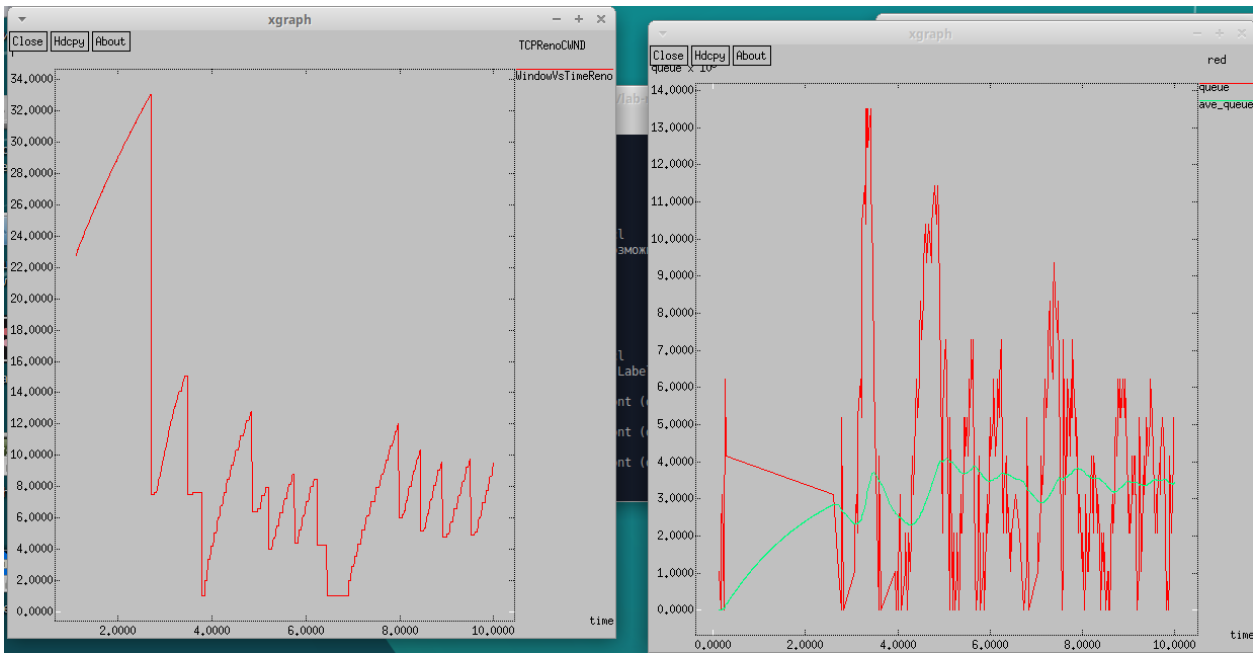

Рисунок 2Графики динамики размера окна TCP, динамики длины очереди и средней длины очереди

Листинг:

```
set ns [new Simulator]

# Узлы сети:
set N 5
for {set i 1} {$i < $N} {incr i} {
set node_(s$i) [$ns node]
}
set node_(r1) [$ns node]
set node_(r2) [$ns node]

# Соединения:
$ns duplex-link $node_(s1) $node_(r1) 10Mb 2ms DropTail
$ns duplex-link $node_(s2) $node_(r1) 10Mb 3ms DropTail
$ns duplex-link $node_(r1) $node_(r2) 1.5Mb 20ms RED
$ns queue-limit $node_(r1) $node_(r2) 25
$ns queue-limit $node_(r2) $node_(r1) 25
$ns duplex-link $node_(s3) $node_(r2) 10Mb 4ms DropTail
$ns duplex-link $node_(s4) $node_(r2) 10Mb 5ms DropTail

# Агенты и приложения:
set tcp1 [$ns create-connection TCP/Reno $node_(s1) TCPSink $node_(s3) 0]
```

```
$tcp1 set window_ 15
set tcp2 [$ns create-connection TCP/Reno $node_(s2) TCPSink $node_(s3) 1]
$tcp2 set window_ 15
set ftp1 [$tcp1 attach-source FTP]
set ftp2 [$tcp2 attach-source FTP]

# Мониторинг размера окна TCP:
set windowVsTime [open WindowVsTimeReno w]
set qmon [$ns monitor-queue $node_(r1) $node_(r2) [open qm.out w] 0.1];
[$ns link $node_(r1) $node_(r2)] queue-sample-timeout;
# Мониторинг очереди:
set redq [[$ns link $node_(r1) $node_(r2)] queue]
set tchan_ [open all.q w]
$redq trace curq_
$redq trace ave_
$redq attach $tchan_

# Добавление at-событий:
$ns at 0.0 "$ftp1 start"
$ns at 1.1 "plotWindow $tcp1 $windowVsTime"
$ns at 3.0 "$ftp2 start"
$ns at 10 "finish"

# Формирование файла с данными о размере окна TCP:
proc plotWindow {tcpSource file} {
global ns
set time 0.01
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file"
}

# Процедура finish:
proc finish {} {
```

```
global tchan_

# подключение кода AWK:
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";
}
}
set f [open temp.queue w]
puts $f "TitleText: red"
puts $f "Device: Postscript"
if { [info exists tchan_] } {
close $tchan_
}
exec rm -f temp.q temp.a
exec touch temp.a temp.q
exec awk $awkCode all.q
puts $f \"q
exec cat temp.q >@ $f
puts $f \n\"a_q
exec cat temp.a >@ $f
close $f
# Запуск xgraph с графиками окна TCP и очереди:
exec xgraph -bb -tk -x time -t "TCPRenoCWND" WindowVsTimeReno &
exec xgraph -bb -tk -x time -y queue temp.queue &
exit 0
}

$ns run
```

2. Упражнение.

Заменив в программе строку

set tcp1 [$ns create-connection TCP/Reno $node_(s1) TCPSink $node_(s3) 0]

на строку

set tcp1 [$ns create-connection TCP/Newreno $node_(s1) TCPSink $node_(s3) 0]

получил следующий результат (рис. 3):

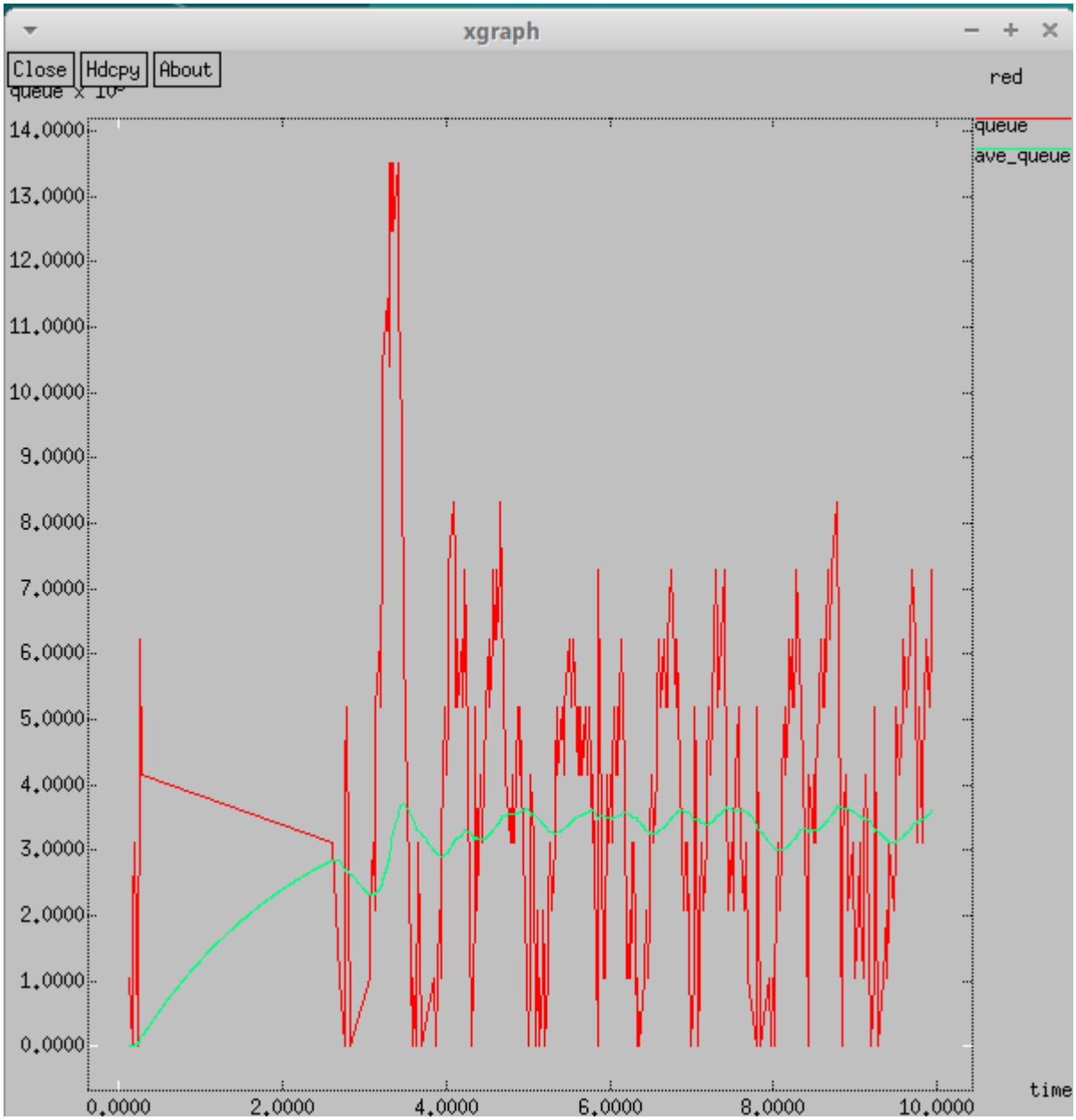

Рисунок 3Графики динамики длины очереди и средней длины очереди для протокола TCP/Newreno

Заменив эту строку на

set tcp1 [$ns create-connection TCP/Vegas $node_(s1) TCPSink $node_(s3) 0]

и строку

exec xgraph -bb -tk -x time -y queue temp.queue &

на

exec xgraph -bb -bg light-grey -fg cyan -tk -x t -y q temp.queue &

получил результат, представленный на рис.4.

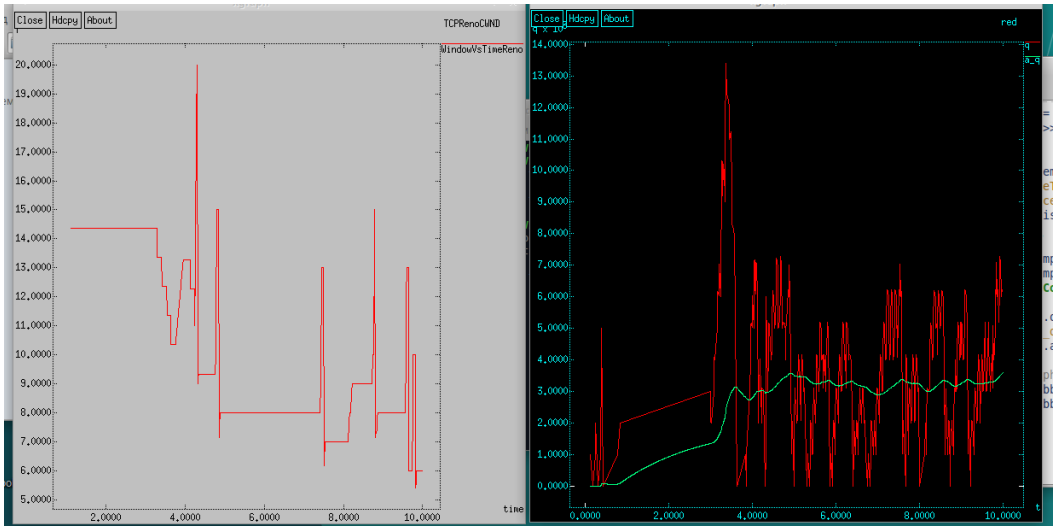

Рисунок 4Графики динамики размера окна TCP, динамики длины очереди и средней длины очереди для протокола TCP/Vegas с изменёнными цветами и подписями

Сравнил полученные результаты.

Заключение

Графики динамики длины очереди и средней длины очереди для TCP Reno, TCP NewReno и TCP Vegas очень похожи.

Значительно различаются только графики динамики размера окна, так как алгоритмы изменения размера окна отличаются у этих TCP-агентов.

Также, в ходе выполнения лабораторной работы я приобрёл навыки визуализации результатов моделирования с помощью средства xgraph.