Лабораторная работа №3

Имитационное моделирование

Серёгина Ирина Андреевна

Содержание

# 1 Цель работы

Смоделировать систему массового обслуживания (СМО).

# 2 Задание

1. Реализация модели *M|M|1*
2. Подсчитать вероятность потери пакетов и загрузку системы
3. Построить график изменения размера очереди

# 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.

# 4 Выполнение лабораторной работы

Для начала я создаю файл, в котором буду выполнять лабораторную работу (рис. 1).

Создание файла

Рис. 1: Создание файла

В нем реализую модель 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

После запуска данного файла мы видим теоретическую вероятность потери пакетов и теоретическую среднюю длину очереди (рис. 2).

Результат работы программы

Рис. 2: Результат работы программы

В каталоге с проектом создаю отдельный файл graph\_plot. Открываю его на редактирование и добавляю следующий код, обращая внимание на синтаксис GNUplot (рис. 3).

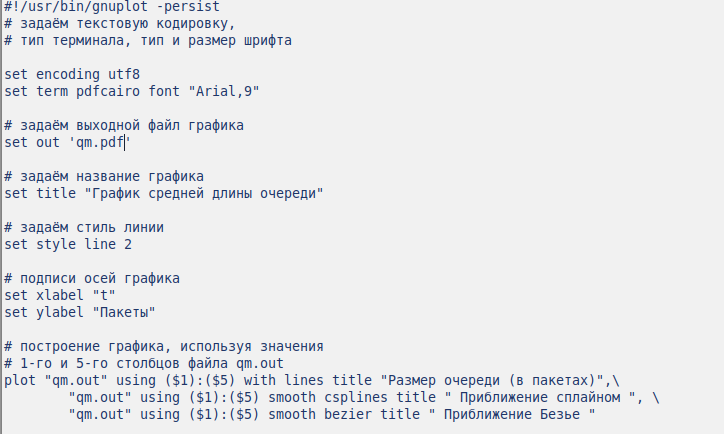


Рис. 3: Листинг программы для создания графика

Затем делаю файл исполняемым и компилирую его (рис. 4).

Компиляция файла

Рис. 4: Компиляция файла

В папке, где находился файл с программой появился файл pdf, содержащий в себе график поведения длины очереди, его приближение сплайном и Безье (рис. 5).

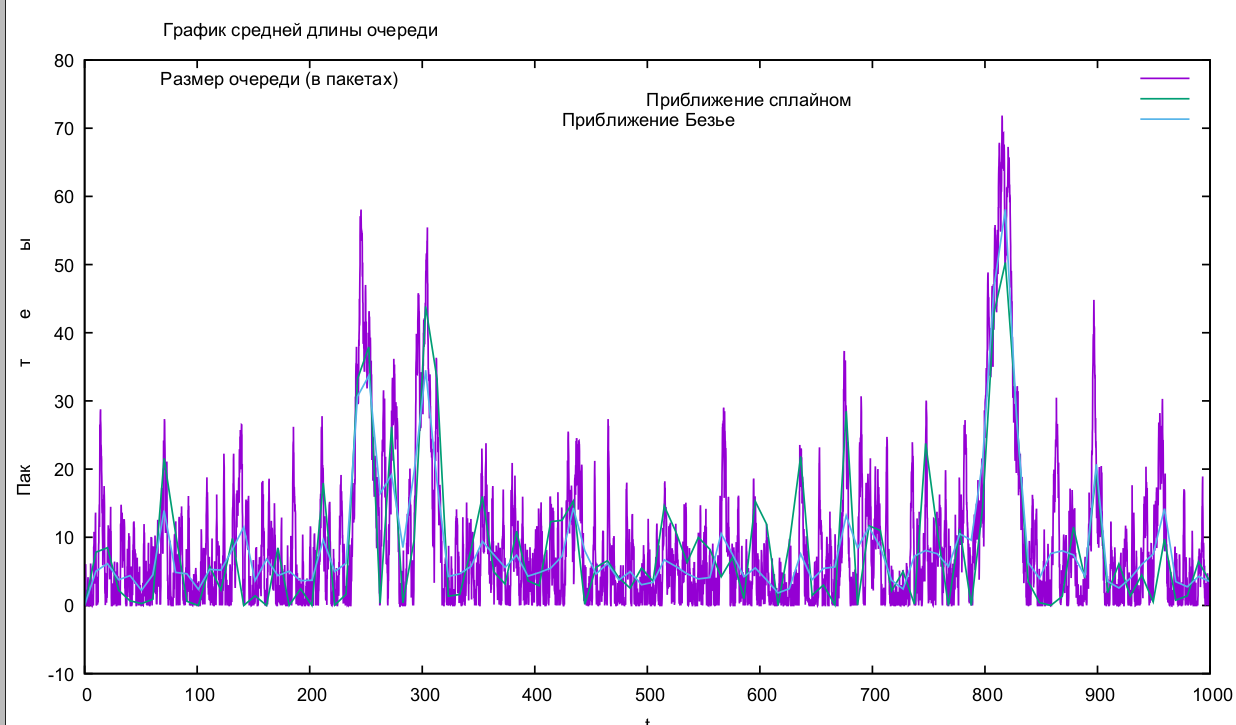


Рис. 5: График поведения длины очереди

# 5 Выводы

Я смоделировала систему массового обслуживания