Белорусский государственный университет информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Отчет по лабораторным работам № 5, 6

по дисциплине «Системный анализ и машинное моделирование»

Вариант 25

Проверил:

Мельник Н. И.

Выполнил:

студент группы 451006

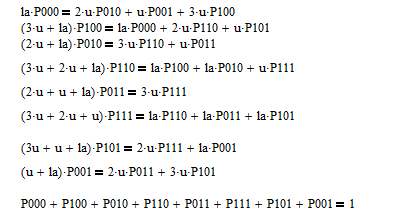
Снитовец М. В.

**1. Построение и исследование аналитической модели непрерывно-стохастической системы массового обслуживания**

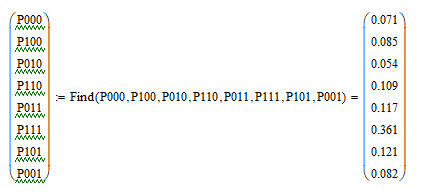
В двухканальной СМО вида M/M/n интенсивность обслуживания первого канала 1=5, второго 2. Определить значение 2 при котором будет обслуживаться не менее 95% поступающих заявок, если=9,5. Все потоки простейшие.

|  |  |
| --- | --- |
| c1 | = {0, 1} – количество заявок в канале 1; |
| c2 | = {0, 1} – количество заявок в канале 2; |

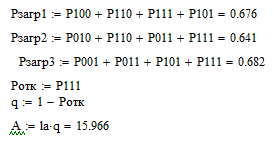
Общий вид кодировки состояния системы: **{c1, c2}** по графу построить аналитическую модель и, решив ее, определить вероятности состояний:

****

Решив систему уравнений (при u = 3, la = 25), получим:



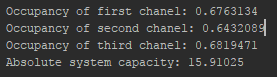
Рассчитать теоретическое значение показателей:



**2. Построение и исследование имитационной модели непрерывно-стохастической системы массового обслуживания**

* для заданной СМО простроить имитационную модель и исследовать ее:

Результаты работы программы (имитация cто тысячи тактов):



**Вывод**:

В ходе лабораторной работы была аналитически смоделирована непрерывно- стохастическая СМО и разработана программа, имитирующая ее поведение. Построенная модель позволяет статистически подсчитать необходимые характеристики СМО. Полученные статистические значения искомых характеристик близки к теоретически рассчитанным: отклонения при моделировании ста тысяч тактов составляют не более нескольких тысячных.

Исходный код программы:

import sys

from ArgumentParser import ArgumentParser

from Chanel import Chanel

from Generator import Generator

u = 0

l = 0

normalizing\_factor = 0

accuracy = 100

def main(argv):

argument\_parser = ArgumentParser()

arguments = argument\_parser.parse(argv)

get\_normalized\_intensity(arguments)

first\_chanel = Chanel(3 \* u)

second\_chanel = Chanel(2 \* u)

third\_chanel = Chanel(u)

generator = Generator(l)

ticks\_number = 100000 \* accuracy

for i in range(0, ticks\_number):

if generator.is\_generated():

generator.start\_generate()

if first\_chanel.is\_processed():

first\_chanel.add()

elif second\_chanel.is\_processed():

second\_chanel.add()

elif third\_chanel.is\_processed():

third\_chanel.add()

first\_chanel.tick()

second\_chanel.tick()

third\_chanel.tick()

generator.tick()

print('Occupancy of first chanel:', first\_chanel.work\_time / ticks\_number)

print('Occupancy of second chanel:', second\_chanel.work\_time / ticks\_number)

print('Occupancy of third chanel:', third\_chanel.work\_time / ticks\_number)

print('Absolute system capacity:', (first\_chanel.processed\_claims + second\_chanel.processed\_claims

+ third\_chanel.processed\_claims) \* normalizing\_factor / ticks\_number)

def get\_normalized\_intensity(arguments):

global u

global l

global normalizing\_factor

u = arguments.u

l = arguments.l

normalizing\_factor = max(u, l) \* accuracy

u = u / normalizing\_factor

l = l / normalizing\_factor

if \_\_name\_\_ == "\_\_main\_\_":

main(sys.argv[1:])