Министерство образования Республики Беларусь

Учреждение образования БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей Кафедра электронных вычислительных машин Дисциплина: Моделирование

ОТЧЕТ

по лабораторной работе № 4

Вариант № 3

Группа 050503

Студенты: А.С. Сикорин

И.А. Григорик

Проверил: И.Г. Алексеев

МИНСК 2023

# 1. Расчёт варианта

# 05050143 % 15 = 3

# 2. Задание

Построить аналитическую и имитационную модели и сравнить результаты исследования

СМО с ожиданием ответа

…

n1=6 n2

λ =2.5 μ

Определить среднее число ожидающих ответа источников, среднее время ожидания ответа и абсолютную пропускную способность (интенсивность на выходе системы) Входные потоки и потоки обслуживаний – простейшие.

а) n2=3, μ=6

б) n2=2, μ=9

в) n2=1, μ=18

В данном случае:

λ - интенсивность входящего потока (в данном случае, число поступающих запросов в единицу времени).

μ - интенсивность обслуживания (число запросов, обслуживаемых в единицу времени).

Общие формулы для простейших потоков:

1. Среднее число ожидающих ответа источников (*Lq*​):
2. Среднее время ожидания ответа (*Wq*​):

​​

1. Абсолютная пропускная способность (интенсивность на выходе системы) (*X*):

X=​

# Построение аналитической модели

* Случай «А» (n1 = 6), (λ = 2.5), (n2=3), ( = 6):

1. Среднее число ожидающих ответа источников (Lq):
2. Среднее время ожидания ответа (Wq):
3. Абсолютная пропускная способность (интенсивность на выходе системы) (X):

* Случай «Б» (n1 = 6), (λ = 2.5), (n2=2), ( = 9):

1. Среднее число ожидающих ответа источников (Lq):
2. Cреднее время ожидания ответа (Wq):
3. Абсолютная пропускная способность (интенсивность на выходе системы) (X):

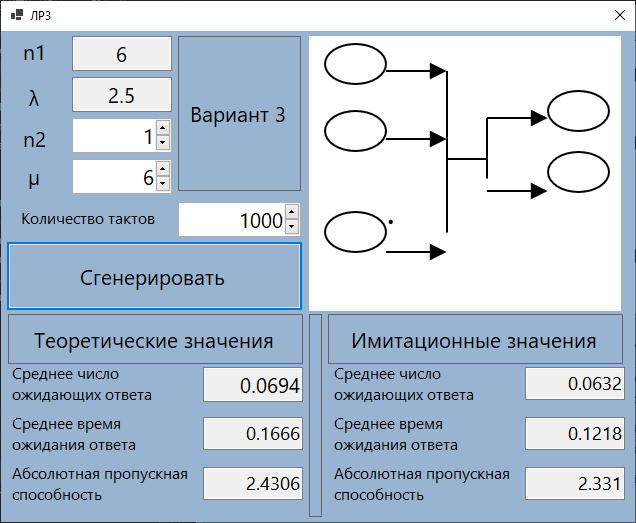
* Случай «В» (n1 = 6), (λ = 2.5), (n2=1), ( = 18).

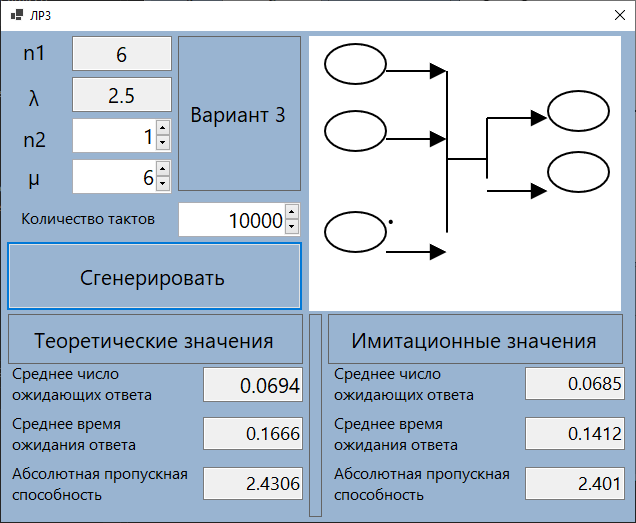
1. Среднее число ожидающих ответа источников (Lq):
2. Cреднее время ожидания ответа (Wq):
3. Абсолютная пропускная способность (интенсивность на выходе системы) (X):

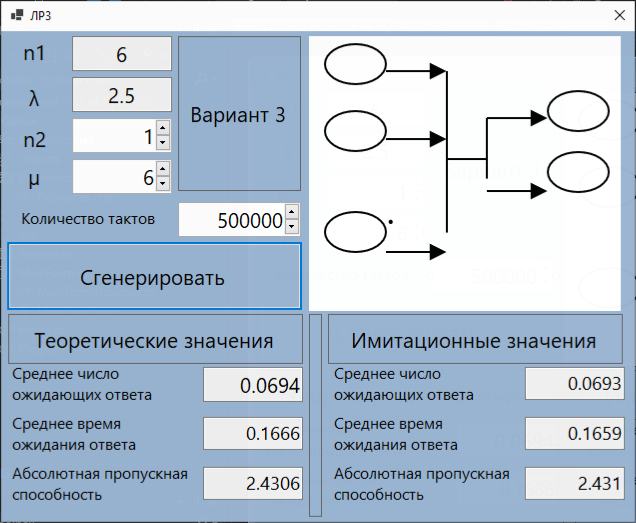
# Построение имитационной модели:

Результаты работы программы при количестве заявок 1000, 10000 и 500000:

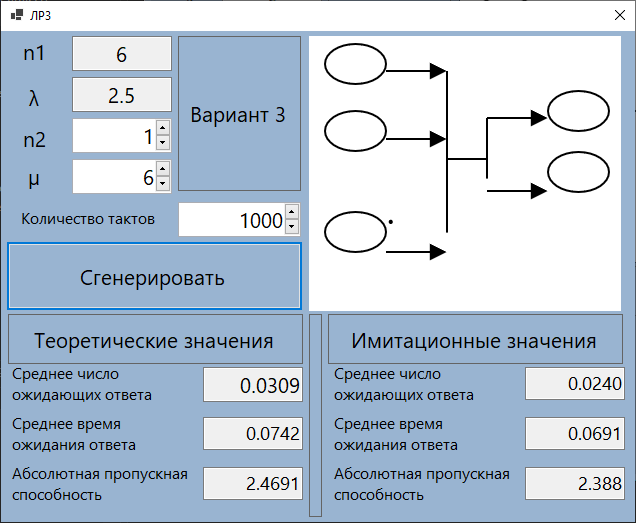
Случай «А»:

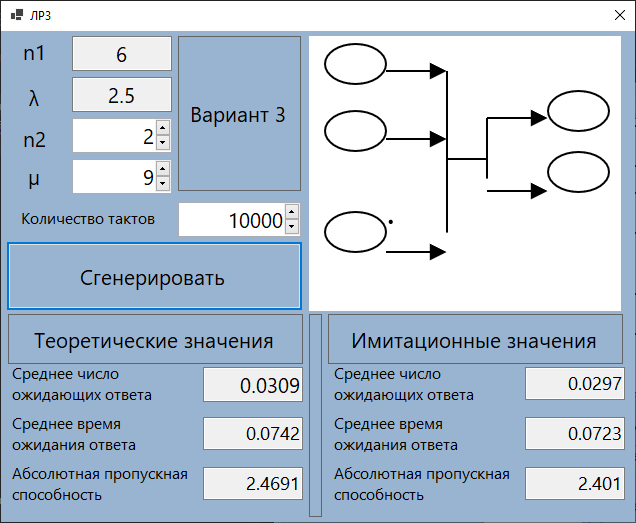


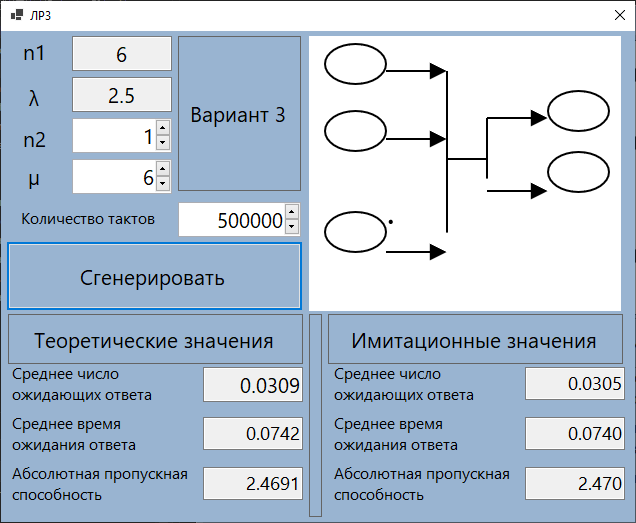




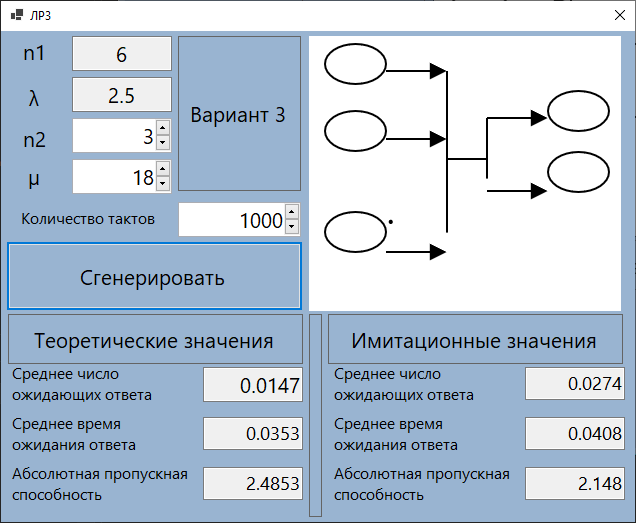
Случай «Б»:

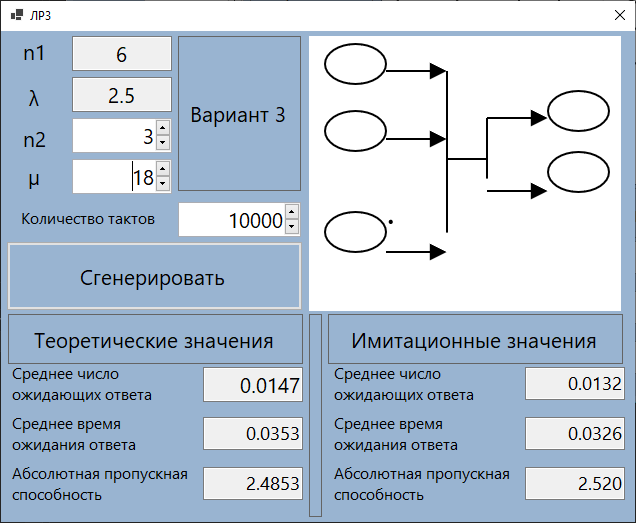


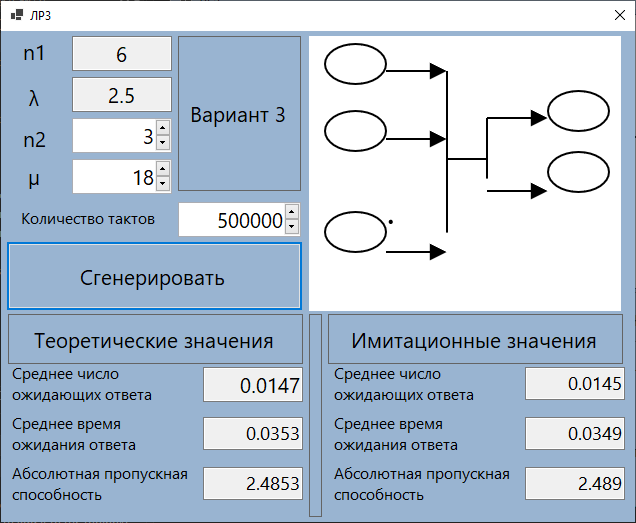




Случай «В»:







Исходный текст программы:

internal class Quetch

{

private readonly Random randChannel1, randChannel2;

private readonly double n1 = 6, lambda = 2.5;

public byte queue\_ { get; private set; }

public byte channels { get; private set; }

public int count { get; private set; }

public int req\_number { get; private set; }

public double queue\_time{ get; private set; }

public Quetch(int numberOfRequests)

{

Queue\_ = 0;

Channel = 0;

QueueCount = 0;

NumberOfRequests = numberOfRequests;

}

public void Work()

{

Random rnd = new Random();

List<double> requestStream = new List<double>();

double newRequest = 0;

requestStream.Add(0);

for (int i = 1; i < NumberOfRequests; i++)

{

newRequest += -Math.Log(rnd.NextDouble()) / h;

requestStream.Add(newRequest);

}

Imitate(requestStream);

void Imitate(List<double> requests)

{

Random rand = new Random();

int count = 0;

double timeInChannel = 0;

List<double> outputStream = new List<double>();

while (count < requests.Count)

{

if (timeInChannel > tl)

{

Channel = 0;

Queue\_++;

}

if (Channel == 0)

{

if (Queue\_ > 0)

{

TimeInQueue += to;

Queue\_--;

}

Channel = 1;

timeInChannel = 0;

}

if(Math.Log(rand.NextDouble()) < m)

{

timeInChannel += to;

}

else

{

outputStream.Add(requests[count] + Math.Log(rand.NextDouble())/m);

Channel = 0;

timeInChannel = 0;

}

count++;

QueueCount += Queue\_;

}

}

}