

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

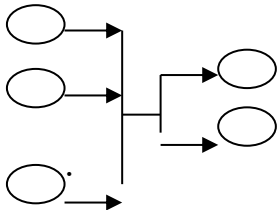
## 1. Расчёт варианта

$$05050143 \% 15 = 3$$

## 2. Задание

Построить аналитическую и имитационную модели и сравнить результаты исследования

СМО с ожиданием ответа



$$\begin{array}{ll} n_1=6 & n_2 \\ \lambda = 2.5 & \mu \end{array}$$

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)  $n_2=3, \mu=6$
- б)  $n_2=2, \mu=9$
- в)  $n_2=1, \mu=18$

В данном случае:

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

$\mu$  - интенсивность обслуживания (число запросов, обслуживаемых в единицу времени).

Общие формулы для простейших потоков:

1. Среднее число ожидающих ответа источников ( $L_q$ ):

$$L_q = \frac{n_1(\lambda \setminus n_1)^2}{n_1(\mu - \lambda \setminus n_1)}$$

2. Среднее время ожидания ответа ( $W_q$ ):

$$W_q = \frac{L_q}{\lambda/n_1}$$

3. Абсолютная пропускная способность (интенсивность на выходе системы) ( $X$ ):

$$X = \lambda - L_q$$

## 2.1 Построение аналитической модели

- Случай «А» ( $n_1 = 6$ ), ( $\lambda = 2.5$ ), ( $n_2=3$ ), ( $\mu = 6$ ):

1. Среднее число ожидающих ответа источников ( $L_q$ ):

$$L_q = \frac{n_1(\lambda/n_1)^2}{\mu(\mu - \lambda/n_1)} = \frac{6(2.5/6)^2}{6(6 - 2.5/6)} \approx 0.0694$$

2. Среднее время ожидания ответа ( $W_q$ ):

$$W_q = \frac{L_q}{\lambda/n_1} = \frac{0.0694}{2.5/6} \approx 0.1666$$

3. Абсолютная пропускная способность (интенсивность на выходе системы) ( $X$ ):

$$X = \lambda - L_q = 2.5 - 0.0694 \approx 2.4306$$

- Случай «Б» ( $n_1 = 6$ ), ( $\lambda = 2.5$ ), ( $n_2=2$ ), ( $\mu = 9$ ):

1. Среднее число ожидающих ответа источников ( $L_q$ ):

$$L_q = \frac{n_1(\lambda/n_1)^2}{\mu(\mu - \lambda/n_1)} = \frac{6(2.5/6)^2}{9(9 - 2.5/6)} \approx 0.0309$$

2. Среднее время ожидания ответа ( $W_q$ ):

$$W_q = \frac{L_q}{\lambda/n_1} = \frac{0.0309}{2.5/6} \approx 0.0742$$

3. Абсолютная пропускная способность (интенсивность на выходе системы) ( $X$ ):

$$X = \lambda - L_q = 2.5 - 0.0309 \approx 2.4691$$

- Случай «В» ( $n_1 = 6$ ), ( $\lambda = 2.5$ ), ( $n_2=1$ ), ( $\mu = 18$ ).

1. Среднее число ожидающих ответа источников ( $L_q$ ):

$$L_q = \frac{n_1(\lambda/n_1)^2}{\mu(\mu - \lambda/n_1)} = \frac{6(2.5/6)^2}{18(18 - 2.5/6)} \approx 0.0147$$

2. Среднее время ожидания ответа ( $W_q$ ):

$$W_q = \frac{L_q}{\frac{\lambda}{n_1}} = \frac{0.0147}{\frac{2.5}{6}} \approx 0.0353$$

3. Абсолютная пропускная способность (интенсивность на выходе системы) ( $X$ ):

$$X = \lambda - L_q = 2.5 - 0.0147 \approx 2.4853$$

## 2.2 Построение имитационной модели:

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

Случай «А»:

ЛРЗ

n1: 6  
 $\lambda$ : 2.5  
n2: 1  
 $\mu$ : 6  
Количество тактов: 1000  
Вариант 3

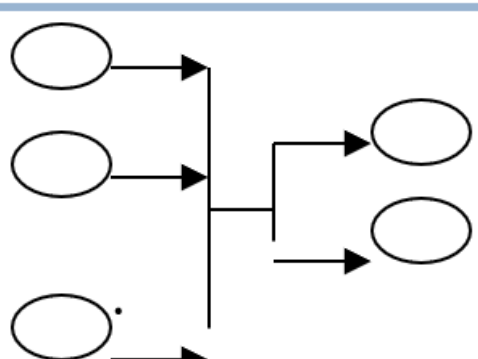
Сгенерировать

Теоретические значения

Среднее число ожидающих ответа	0.0694
Среднее время ожидания ответа	0.1666
Абсолютная пропускная способность	2.4306

Имитационные значения

Среднее число ожидающих ответа	0.0632
Среднее время ожидания ответа	0.1218
Абсолютная пропускная способность	2.331



ЛРЗ

n1: 6  
 $\lambda$ : 2.5  
n2: 1  
 $\mu$ : 6  
Количество тактов: 10000  
Вариант 3

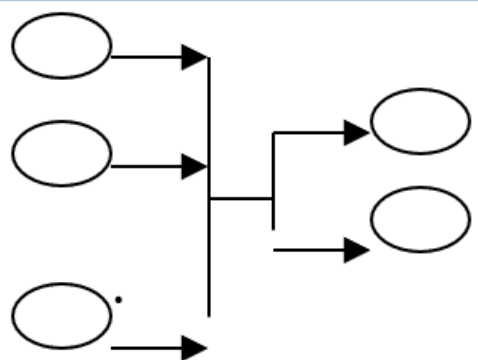
Сгенерировать

Теоретические значения

Среднее число ожидающих ответа	0.0694
Среднее время ожидания ответа	0.1666
Абсолютная пропускная способность	2.4306

Имитационные значения

Среднее число ожидающих ответа	0.0685
Среднее время ожидания ответа	0.1412
Абсолютная пропускная способность	2.401



ЛРЗ

n1: 6

$\lambda$ : 2.5

n2: 1

$\mu$ : 6

Количество тактов: 500000

Вариант 3

Сгенерировать

**Теоретические значения**

Среднее число ожидающих ответа	0.0694
Среднее время ожидания ответа	0.1666
Абсолютная пропускная способность	2.4306

**Имитационные значения**

Среднее число ожидающих ответа	0.0693
Среднее время ожидания ответа	0.1659
Абсолютная пропускная способность	2.431

Случай «Б»:

ЛРЗ

n1: 6

$\lambda$ : 2.5

n2: 1

$\mu$ : 6

Количество тактов: 1000

Вариант 3

Сгенерировать

**Теоретические значения**

Среднее число ожидающих ответа	0.0309
Среднее время ожидания ответа	0.0742
Абсолютная пропускная способность	2.4691

**Имитационные значения**

Среднее число ожидающих ответа	0.0240
Среднее время ожидания ответа	0.0691
Абсолютная пропускная способность	2.388

ЛРЗ

$n_1$  6  
 $\lambda$  2.5  
 $n_2$  2  
 $\mu$  9  
 Количество тактов 10000  
 Вариант 3

Сгенерировать

```

    graph LR
      A1(( )) --> J1(( ))
      A2(( )) --> J1
      A3(( )) --> J1
      J1 --> J2(( ))
      J1 --> J3(( ))
      J2 --> B1(( ))
      J3 --> B2(( ))
  
```

Теоретические значения		Имитационные значения	
Среднее число ожидающих ответа	0.0309	Среднее число ожидающих ответа	0.0297
Среднее время ожидания ответа	0.0742	Среднее время ожидания ответа	0.0723
Абсолютная пропускная способность	2.4691	Абсолютная пропускная способность	2.401

ЛРЗ

$n_1$  6  
 $\lambda$  2.5  
 $n_2$  1  
 $\mu$  6  
 Количество тактов 500000  
 Вариант 3

Сгенерировать

```

    graph LR
      A1(( )) --> J1(( ))
      A2(( )) --> J1
      A3(( )) --> J1
      J1 --> J2(( ))
      J1 --> J3(( ))
      J2 --> B1(( ))
      J3 --> B2(( ))
  
```

Теоретические значения		Имитационные значения	
Среднее число ожидающих ответа	0.0309	Среднее число ожидающих ответа	0.0305
Среднее время ожидания ответа	0.0742	Среднее время ожидания ответа	0.0740
Абсолютная пропускная способность	2.4691	Абсолютная пропускная способность	2.470

Случай «В»:

■ ЛРЗ

n1

$\lambda$

n2

$\mu$

Количество тактов

Вариант 3

Сгенерировать

Теоретические значения

Среднее число ожидающих ответа	<input type="text" value="0.0147"/>
Среднее время ожидания ответа	<input type="text" value="0.0353"/>
Абсолютная пропускная способность	<input type="text" value="2.4853"/>

Имитационные значения

Среднее число ожидающих ответа	<input type="text" value="0.0274"/>
Среднее время ожидания ответа	<input type="text" value="0.0408"/>
Абсолютная пропускная способность	<input type="text" value="2.148"/>

```
graph LR; A(( )) --> J(( )); B(( )) --> J; C((.)) --> J; J --> D(( )); J --> E(( ))
```

■ ЛРЗ

n1

$\lambda$

n2

$\mu$

Количество тактов

Вариант 3

Сгенерировать

Теоретические значения

Среднее число ожидающих ответа	<input type="text" value="0.0147"/>
Среднее время ожидания ответа	<input type="text" value="0.0353"/>
Абсолютная пропускная способность	<input type="text" value="2.4853"/>

Имитационные значения

Среднее число ожидающих ответа	<input type="text" value="0.0132"/>
Среднее время ожидания ответа	<input type="text" value="0.0326"/>
Абсолютная пропускная способность	<input type="text" value="2.520"/>

```
graph LR; A(( )) --> J(( )); B(( )) --> J; C((.)) --> J; J --> D(( )); J --> E(( ))
```

ЛРЗ

n1

$\lambda$

n2

$\mu$

Количество тактов

Вариант 3

Сгенерировать

```

graph LR
    A(( )) --> J(( ))
    B(( )) --> J
    C(( )) --> J
    J --> D(( ))
    J --> E(( ))
  
```

Теоретические значения

Среднее число ожидающих ответа	0.0147
Среднее время ожидания ответа	0.0353
Абсолютная пропускная способность	2.4853

Имитационные значения

Среднее число ожидающих ответа	0.0145
Среднее время ожидания ответа	0.0349
Абсолютная пропускная способность	2.489

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_;
    }
}

```