Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра программного обеспечения информационных технологий

ОТЧЁТ

по лабораторной работе №3

по курсу «Системный анализ и машинное моделирование»

на тему

«Построение аналитической модели дискретно-стохастической системы»

Вариант 7

|  |  |
| --- | --- |
| Проверил:  Мельник Н.И. | Выполнил:  студент гр. 851001  Касперович В.Л. |

Минск 2011

1. ЦЕЛЬ РАБОТЫ

Построение аналитической модели дискретно-стохастической системы и расчет параметров данной системы с использованием построенной модели.

Р-схема (вариант 7)

2

π

π

2

1. ИСХОДНЫЕ ДАННЫЕ К РАБОТЕ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант | ρ | π1 | π2 | Цель исследования |
| 7 | - | 0,75 | 0,6 | Lоч, А |

1. РАСЧЕТ МОДЕЛИ Р-СХЕМЫ

Граф состояний кодируется четырехкомпонентным вектором *abcd,* где *a* – время до выдачи очередной заявки источником, *a*={2,1,0} (значение 0 означает, что источник заблокирован); *b* – количество заявок, находящихся в накопителе (длина очереди), *b*={0,1,2}; *c* и *d* – состояние каналов обслуживания, принимает следующие значения: 0 – канал свободен, 1 – канал занят обслуживанием заявки.

Граф состояний представлен на рис.1.

Исходя из графа состояний, построим СЛАУ для нахождения вероятностей состояний:



и дополним ее нормировочным уравнением 



 Рис.1. Граф состояний СМО.

Подставив в СЛАУ значения  и  и приведя СЛАУ к каноническому виду, получим:



Или, в матричном виде:

.

Решив данную СЛАУ с помощью пакета MATLAB, получим значения вероятностей состояний:



Используя данные значения, подсчитаем следующие величины:

– среднюю длину очереди:

где – число заявок в очереди при состоянии *i.*

– абсолютную пропускную способность (интенсивность потока обработки заявок):



где *р* – сумма вероятностей состояний, при которых не происходит обслуживание заявок (в данном случае это состояние 2), *Т* – количество тактов, за которые источник выдает 1 заявку.

1. ВЫВОДЫ

В данной работе была исследована дискретно-стохастическая модель (Р-схема). Для нее был построен граф состояний, на основе которого затем была составлена СЛАУ, из которой были получены вероятности каждого из состояний системы. Были посчитаны необходимые характеристики системы, которые затем будут сравнены с результатами, получившимися при имитационном моделировании.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД ПРОГРАММЫ

Модуль MainForm:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.IO;

namespace SAMM\_3\_4

{

public partial class MainForm : Form

{

Random rnd;

const int numtacts = 100000;

List<Label> lbs = new List<Label>();

public MainForm()

{

InitializeComponent();

rnd = new Random();

#region lbsadd

lbs.Add(label5);

lbs.Add(label6);

lbs.Add(label7);

lbs.Add(label8);

lbs.Add(label9);

lbs.Add(label10);

lbs.Add(label11);

lbs.Add(label12);

lbs.Add(label13);

lbs.Add(label14);

lbs.Add(label15);

lbs.Add(label16);

#endregion

}

private void button1\_Click(object sender, EventArgs e)

{

int[] prob = new int[13];

QueueSystem qs = new QueueSystem(Double.Parse(textBox1.Text), Double.Parse(textBox2.Text));

// QueueSystem qs = new QueueSystem(0.75,0.6);

int[] firstValue = new int[4] {2, 0, 0, 0 };

int[] nv = qs.Tact(firstValue);

double MeanQLen=0,MeanTP=0;

StreamWriter sw = new StreamWriter("1.csv");

sw.WriteLine(Convert.ToString(firstValue[0]) + Convert.ToString(firstValue[1]) + Convert.ToString(firstValue[2]) + Convert.ToString(firstValue[3]));

sw.WriteLine(Convert.ToString(nv[0]) + Convert.ToString(nv[1]) + Convert.ToString(nv[2]) + Convert.ToString(nv[3]));

for (int i = 0; i <= numtacts; i++)

{

int t = nv[0];

nv = qs.Tact(nv);

if ((nv[0] == 2) && (t == 1)) MeanTP++;

MeanQLen += nv[1];

string s = Convert.ToString(nv[0]) + Convert.ToString(nv[1]) + Convert.ToString(nv[2]) + Convert.ToString(nv[3]);

switch (s)

{

case "2000":

prob[0]++;

break;

case "1000":

prob[1]++;

break;

case "2010":

prob[2]++;

break;

case "1010":

prob[3]++;

break;

case "2011":

prob[4]++;

break;

case "1001":

prob[5]++;

break;

case "1011":

prob[6]++;

break;

case "2111":

prob[7]++;

break;

case "1111":

prob[8]++;

break;

case "2211":

prob[9]++;

break;

case "1211":

prob[10]++;

break;

case "0211":

prob[11]++;

break;

default:

prob[12]++;

break;

}

sw.WriteLine(Convert.ToString(nv[0]) + Convert.ToString(nv[1]) + Convert.ToString(nv[2]) + Convert.ToString(nv[3]));

}

for (int j =0;j<=11;j++)

{

lbs[j].Text = Convert.ToString(j + 1) + " | " + Convert.ToString((double)prob[j] / numtacts);

}

sw.Close();

MeanTP -= nv[1];

MeanTP -= nv[2];

MeanTP -= nv[3];

label3.Text = "A="+Convert.ToString(MeanTP / numtacts);

label4.Text ="Lоч="+Convert.ToString(MeanQLen / numtacts);

}

}

}

Модуль QueueSystem:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace SAMM\_3\_4

{

class QueueSystem

{

double \_pi1, \_pi2;

Random rnd;

public QueueSystem(double pi1, double pi2)

{

\_pi1 = pi1;

\_pi2 = pi2;

rnd = new Random();

//!!!Вынесение инициализации рандома в конструктор обусловлено тем, что при повторной инициализации на каждом такте СМО в качестве затравки рандома

//среда разработки берет текущее значение времени в мс, что влечет за собой одинаковые значения моделируемых вероятностей на нескольких тактах,

//следовательно, вероятность выхода СМО из циклического состояния уменьшается во много раз!

}

public int[] Tact(int[] prevState)

{

int[] newState = new int[4];

int pi1Prob = rnd.Next(100) + 1;//вероятность прохода заявки в pi1

int pi2Prob = rnd.Next(100) + 1;//вероятность прохода заявки в pi2

if ((pi1Prob > \_pi1 \* 100) && (prevState[2] == 1))

newState[2] = 0;// заявка в первом канале выполнена

else if (prevState[2] == 1)

newState[2] = 1;

if ((pi2Prob > \_pi2 \* 100) && (prevState[3] == 1))

newState[3] = 0;// заявка во втором канале выполнена

else if (prevState[3] == 1)

newState[3] = 1;

switch (prevState[1]) //обработка заявок, находящихся в очереди

{

case 0:

break;

case 1:

if (newState[2] == 0)

newState[2] = 1;

else

{

if (newState[3] == 0)

newState[3] = 1;

else

newState[1] = 1;

}

break;

case 2:

if (newState[2] == 0)

{

newState[2] = 1;

if (newState[3] == 0)

{

newState[3] = 1;

newState[1] = 0;

}

else

newState[1] = 1;

}

else

{

if (newState[3] == 0)

{

newState[3] = 1;

newState[1] = 1;

}

else

newState[1] = 2;

}

break;

default:

break;

}

if (prevState[0] == 2)

{

newState[0] = 1;

}

else

{

if (prevState[0] < 2)

{

if (newState[2] == 0)

{

newState[2] = 1;

newState[0] = 2;

}

else

{

if (newState[3] == 0)

{

newState[3] = 1;

newState[0] = 2;

}

else

{

if (newState[1] < 2)

{

newState[1]++;

newState[0] = 2;

}

else

{

newState[1] = 2;

newState[0] = 0;

}

}

}

}

}

return newState;

}

}

}