Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра программного обеспечения информационных технологий

ОТЧЁТ

по лабораторной работе №3-4

по курсу «Системный анализ и машинное моделирование»

на тему

«Аналитическое моделирование дискретно-стохастической СМО»

Вариант 3

|  |  |
| --- | --- |
| Проверил:  Мельник Н.И. | Выполнил:  студент гр. 951003  А.В. |

Минск 2022

**Цель работы:**

* Изучить принципы построения и функционирования СМО.
* Освоение методики построения графа состояний СМО.
* Научиться по графу состояний строить аналитическую модель и на ее основании определять вероятности состояний
* Разработать имитационную модель и статически определять вероятности состояний

**Схема СМО:**

3.

2

2

π

π

p1 = 0.5

p2 = 0.7

t = {1, 2} – кол-во тактов до заявки

j = {0, 1, 2} – кол-во заявок в очереди

t1 = {0, 1} – кол-во заявок в канале №1

t2 = {0, 1} – кол-во заявок в канале №2

Общий вид кодировки состояния системы:

{t, j, t1, t2}



По графу построим аналитическую модель и, решив ее, определим вероятности состояний.

P1 = 0

P2 = P1+ P3\*(1-р1) + P6\*(1- р1)\*(1- р2)

P3 = P2+ P4\*(1- р1) + P5\*(1- р2) + P7\*(1- р1)\*(1- р2)

P4 = P3\*р1 + P6\*(1- р1)\*(1- р2) + P8\*(1- р1)\*(1- р2)

P5 = P6\*р2\*(1- р1)

P6 = P4\* р1 + P7\*((1- р1)\*р2+(1- р2)\*р1) + P5\*р2 + P9\*(1- р1)\*(1- р2)

P7 = P6\*р1\*р2 + P8\*((1- р1)\*р2+(1- р2)\*р1) + P10\*(1- р1)\*(1- р2)

P8 = P7\*р1\*р2 + P9\*((1- р1)\*р2+(1- р2)\*р1) + P11\*(1- р1)\*(1- р2)

P9 = P8\*р1\*р2 + P10\*((1- р1)\*р2+(1- р2)\*р1)

P10=P9\*р1\*р2 + P11\*( р1\* р2+(1- р1)\*р2+(1- р2)\*р1)

P11=P10\*р1\*р2

P1+P2+P3+P4+P5+P6+P7+P8+P9+P10+P11 = 1

P1 = 0

P2 = P1+ P3\*0.5 + P6\*0.5\*0.3

P3 = P2+ P4\*0.5 + P5\*0.3 + P7\*0.5\*0.3

P4 = P3\*0.5 + P6\*0.5\*0.3 + P8\*0.5\*0.3

P5 = P6\*0.7\*0.5

P6 = P4\*0.5 + P7\*(0.7\*0.5+0.3\*0.5) + P5\*0.7 + P9\*0.3\*0.5

P7 = P6\*0.7\*0.5 + P8\*(0.5\*0.7+0.3\*0.5) + P10\*0.5\*0.3

P8 = P7\*0.5\*0.7 + P9\*(0.5\*0.7+0.3\*0.5) + P11\*0.5\*0.3

P9 = P8\*0.7\*0.5 + P10\*(0.5\*0.7+0.3\*0.5)

P10=P9\*0.7\*0.5 + P11\*(0.5\*0.7+0.3\*0.5+0.5\*0.7)

P11=P10\*0.7\*0.5

P1+P2+P3+P4+P5+P6+P7+P8+P9+P10+P11 = 1

P1=P2000 = 0

P2=P1000 = 0.1675

P3=P2010 = 0.2839

P4=P1010 = 0.1731

P5=P1001 = 0.0597

P6=P2011 = 0.1706

P7=P1011 = 0.0794

P8=P2111 = 0.0368

P9=P1111 = 0.0171

P10=P2211 = 0.0085

P11=P1211 = 0.0029

A = (P2010+P1010)\*(1-р1)+P1001\*(1–р2)+ (P2011+P1111+P2111+P1211+P1011+P2211)\*(1- р2)\*р1+ (P2011+P1111+P2111+P1211+P1011+ P2211)\*(1- р1)\*р2+2\*(P2011+P1111+P2111+P1211+P1011+P2211) \*(1-р1)\*(1- р2)

= (0,2839+0,1731)\*0,5+0,0597\*0,3+(0,1706+0,0171+0,0368+0,0029+0,0794+0,0085)\*(1-0,5)\*0,7 +(0,1706+0,0171+0,0368+0,0029+0,0794+0,0085)\*(1-0,7)\*0,5 +2\*(0,1706+0,0171+0,0368+0,0029+0,0794+0,0085)\*0,5\*0,3

= 0,49857

Ротк= P1211/ (P1000+ P1010 +P1001 +P1011 +P1111 +P1211)\*р1\*р2

= 0,0029/(0,1675+0,1731+0,0597+0,0794+0,0171+0,0029)\*0,5\*0,7

=0,0020

Рбл = 0

Lоч = P2111 + P1111 + 2\* P2211 + 2\* P1211

= 0,0368 + 0,0171+ 2\*0,0085 + 2\*0,0029

=0,0767

Lc =1\*(P2010+ P1010+ P1001)+2\*( P2011+ P1011)+3\*( P2111+ P1111)+4\*(P2211+P1211)

= 0,2839+0,1731+0,0597+2\*(0,1706+0,0793)+3\*(0,0368+0,0171)+4\*(0,0085+0,0029)

=1,2238

Q = 1 – Ротк= 1- 0,0020 = 0,998

A = (P2010+P1010)\*(1-р1)+P1001\*(1–р2)+ (P2011+P1111+P2111+P1211+P1011+P2211)\*(1- р2)\*р1+ (P2011+P1111+P2111+P1211+P1011+ P2211)\*(1- р1)\*р2+2\*(P2011+P1111+P2111+P1211+P1011+P2211) \*(1-р1)\*(1- р2)

= (0,2839+0,1731)\*0,5+0,0597\*0,3+(0,1706+0,0171+0,0368+0,0029+0,0794+0,0085)\*(1-0,5)\*0,7 +(0,1706+0,0171+0,0368+0,0029+0,0794+0,0085)\*(1-0,7)\*0,5 +2\*(0,1706+0,0171+0,0368+0,0029+0,0794+0,0085)\*0,5\*0,3

= 0,4986

Wоч = Lоч/A = 0,0767/0,49644 = 0,1545

Wс = Lc/A = 1,2238/0,49644 = 2,4651

Kкан1 = P2010+P1010+P2011+P1011+P1111+ P2111+P2211+P1211

=0,2839 +0,1731+0,1705+0,0794+0,0171+0,0368+0,0085+0,0029

= 0,7722

Kкан2 = P1001+P2011+P1011+P1111+P2111+P2211+P1211

= 0,0597 +0,1705+0,0794+0,0171+0,0368+0,0085+0,0029

= 0,3749

Имитационная модель

P2000 = 0

P1000 = 0.16809

P2010 = 0.28469

P1010 = 0.17322

P1001 = 0.05979

P2011 = 0.17117

P1011 = 0.07941

P2111 = 0.03588

P1111 = 0.01666

P2211 = 0.00826

P1211 = 0.00283

Sym(P) = 1.00000

P declined: 0.00198

P block: 0

L queue: 0.07472

L system: 1.22084

Q relative: 0.99802

A: 0.499

W queue: 0.14944

W system: 2.4417

W channel1: 0.77212

W channel2: 0.374

public class Channel

{

private static Random \_random = new Random();

private readonly int \_number;

private readonly double \_probability;

private Request \_request;

private readonly Statistics \_statistics;

public Channel(double probability, Statistics statistics, int number)

{

\_probability = probability;

\_statistics = statistics;

\_number = number;

}

public bool IsBusy { get; set; } = false;

public bool TryProcessRequest()

{

var currentProb = \_random.NextDouble();

\_statistics.BusyTicks[\_number]++;

if (currentProb > \_probability)

{

IsBusy = false;

\_statistics.ProcessedInChannelRequests[\_number]++;

return true;

}

\_request.InChannelTicks++;

return false;

}

public void AddRequest(Request request)

{

IsBusy = true;

\_request = request;

request.InChannelTicks++;

}

}

public class CustomQueue

{

private const int queueCount = 2;

private readonly Queue<Request> \_queue = new();

private readonly Statistics \_statistics;

public Channel FirstChannel { get; }

public Channel SecondChannel { get; }

public int Count => \_queue.Count;

public event EventHandler RequestEvents;

public CustomQueue(Channel firstChannel, Channel secondChannel, Statistics statistics)

{

FirstChannel = firstChannel;

SecondChannel = secondChannel;

\_statistics = statistics;

}

public void AddRequest(Request request)

{

if (!FirstChannel.IsBusy)

{

FirstChannel.AddRequest(request);

// Console.WriteLine("Request was added to first channel");

return;

}

if (!SecondChannel.IsBusy)

{

SecondChannel.AddRequest(request);

//Console.WriteLine("Request was added to second channel");

return;

}

if (\_queue.Count == queueCount)

{

\_statistics.DeclinedRequests++;

//Console.WriteLine("Request was declined");

return;

}

RequestEvents += request.InQueueHandler;

\_queue.Enqueue(request);

//Console.WriteLine("Request was added to queue");

}

public void SendHeldRequestsToChannels()

{

RequestEvents?.Invoke(this, EventArgs.Empty);

if (\_queue.Count < 1)

{

return;

}

if (!FirstChannel.IsBusy)

{

var request = \_queue.Dequeue();

FirstChannel.AddRequest(request);

RequestEvents -= request.InQueueHandler;

//Console.WriteLine("Held request was added to first channel");

}

if (\_queue.Count < 1)

{

return;

}

if (!SecondChannel.IsBusy)

{

var request = \_queue.Dequeue();

SecondChannel.AddRequest(request);

RequestEvents -= request.InQueueHandler;

//Console.WriteLine("Held request was added to second channel");

}

}

}

public class StaxanovSystem

{

private const int tCount = 2;

public CustomQueue Queue { get; }

public Channel FirstChannel { get; }

public Channel SecondChannel { get; }

public Statistics Statistics { get; set; }

public int T { get; set; }

public StaxanovSystem(

CustomQueue queue,

Channel firstChannel,

Channel secondChannel,

Statistics statistics)

{

Queue = queue;

FirstChannel = firstChannel;

SecondChannel = secondChannel;

Statistics = statistics;

}

public void Work(int ticks)

{

//var prev = string.Empty;

for (var i = 1; i <= ticks; i++)

{

T = i % tCount + (tCount - 1);

var code = $"{T}{Queue.Count}"

+ (FirstChannel.IsBusy ? "1" : "0")

+ (SecondChannel.IsBusy ? "1" : "0");

// Console.WriteLine($"{i}) "+code);

Statistics.CollectState(code);

Statistics.CollectQueueState(Queue.Count.ToString());

if (FirstChannel.IsBusy && FirstChannel.TryProcessRequest())

{

// Console.WriteLine("First channel process request");

}

if (SecondChannel.IsBusy && SecondChannel.TryProcessRequest())

{

//Console.WriteLine("Second channel process request");

}

Queue.SendHeldRequestsToChannels();

if (T == 1)

{

var request = new Request();

Queue.AddRequest(request);

Statistics.ProcessedRequests.Add(request);

Statistics.RequestsCount++;

}

/\*if (i != 1)

{

Statistics.NextStates[prev]

.Add(code);

}

prev = code;\*/

}

}

}