СОЗДАНИЕ ИМИТАЦИОННОЙ МОДЕЛИ ОДНОКАНАЛЬНОГО ПЕРЕДАТЧИКА КАДРОВ С БУФЕРОМ ОГРАНИЧЕННОЙ ЕМКОСТИ С ПОМОЩЬЮ VISUAL STUDIO

CREATION OF A SIMULATION MODEL OF A SINGLE-CHANNEL FRAME TRANSMITTER WITH A LIMITED CAPACITY BUFFER USING VISUAL STUDIO

УДК 004.94

Журавлев Александр Александрович, магистрант, Уральский Федеральный Университет им. Б. Н. Ельцина, ИРИТ-РтФ, г. Екатеринбург

Zhuravlev A.A. SanyaProgrammer2503@gmail.com

Аннотация

В настоящее время моделирование играет огромную роль в жизни людей. Оно помогает предвидеть последствия того или иного события, принимать обдуманные и взвешенные решения, а также строить определенные планы на будущее. В данной статье с помощью Visual Studio разработана модель имитационного передатчика кадров с буфером ограниченной емкости. Передатчик имеет такие составляющие как емкость буфера, количество заявок, текущее состояние и пр. Для данной модели теоретически рассчитаны коэффициент (вероятность) потерь заявок и средняя длина очереди. В соответствии с блок-схемой, описывающей работу передатчика, данные показатели вычислены практически с помощью Visual Studio. Теоретические и практические значения показателей коэффициента

потерь заявок, и средней длины очереди практически совпадают, следовательно, модель, разработанную с помощью Visual Studio, можно считать корректной.

Annotation

Currently, modeling plays a huge role in people's lives. It helps to anticipate the consequences of an event, make informed and balanced decisions, as well as build particular plans for the future. In this article, using visual studio, a model of a simulation model of a single-channel frame transmitter with a limited capacity buffer was developed. The transmitter has such components as the buffer capacity, the number of requests, the current state, etc. For this model, the coefficient (probability) of loss of requests and the average queue length are theoretically calculated. In accordance with the flowchart describing the operation of the transmitter, these indicators are calculated practically using Visual Studio. Theoretical and practical values of applications and the average queue length are almost the same, therefore, the model developed using Visual Studio can be considered correct.

Ключевые слова: создание, имитационная модель, одноканальный передатчик, буфер ограниченной емкости, Visual Studio

Keywords: creation, simulation model, single-channel frame transmitter, limited capacity buffer, Visual Studio

Введение

Роль моделирования для современного человека огромна [1]. Оно позволяет обдуманно и взвешенно принимать решения, прогнозировать события, строить определенные планы на будущее и пр.

В данной статье описана модель одноканального передатчика кадров с буфером ограниченной емкости. Параметры, которые имеет данный передатчик, представлены в разделе «Описание модели».

Для данного передатчика необходимо выполнить следующее:

- с помощью Visual Studio разработать модель работы передатчика, представленной с помощью блок-схемы в разделе «Описание модели»;
- рассчитать и сравнить теоретические значения коэффициента потерь заявок и средней длину очереди.

Цель работы – проверить корректность имитационной модели одноканального передатчика кадром с буфером ограниченной емкости, разработанной с помощью Visual Studio.

В качестве **материала исследования** выступает одноканальный передатчик кадров с буфером ограниченной емкости.

В статье используется **эмпирический метод** исследования, поскольку основной источник результатов – эксперимент.

Описание модели

Работу одноканального передатчика кадров с буфером ограниченной емкости можно представить с помощью блок-схемы, изображенной на рисунке 1.

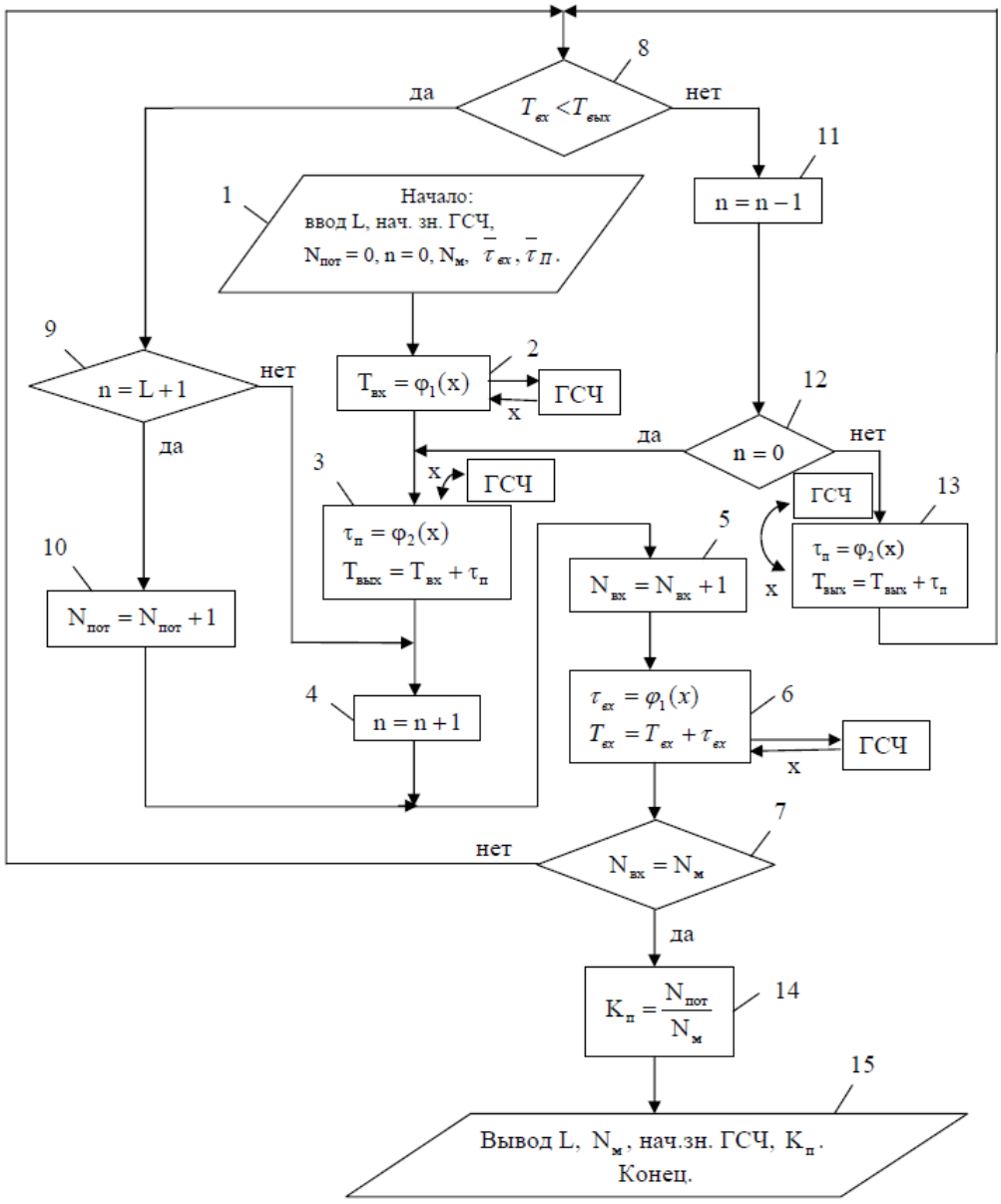

Рисунок 1 – Работа одноканального передатчика кадров с ограниченной емкостью буфера

На схеме использованы обозначения:

- n – переменная состояния (количество заявок в модели);
- $N_{вх}$ – текущее количество поступивших на вход заявок;
- $N_{пот}$ – количество потерянных заявок;
- $K_п$ – коэффициент потерь;
- L – емкость буфера;

- $N_м$ – заданное максимальное число входных заявок – условие окончания моделирования;
- ГСЧ – генератор случайных чисел x;
- φ_1, φ_2 – функции, преобразующие равномерно распределенную случайную величину x в экспоненциально распределенные $\tau_{вх}$ и $\tau_п$.

Начальное значение ГСЧ используется, чтобы получить псевдослучайную воспроизводимую последовательность случайных чисел x.

Блоки на схеме – операции событийного моделирования:

- 1 – задание начальных значений для одного прогона модели;
- 2, 3, 6, 13 – имитация времени будущих событий изменения $z_i \rightarrow z_j$;
- 7 – контроль окончания моделирования;
- 8 – определение времени ближайшего события и вида наступающего состояния в переходе;
- 4, 9, 11, 12 – изменение состояния (фиксация и использование переменной состояния для разветвлений алгоритма);
- 5, 10 – счетчики;
- 14, 15 – получение и вывод результатов.

Типовые параметры модели: среднее время между появлениями на входе $\tau_{вх} = 2$, среднее время обслуживания/передачи $\tau_{вх} = 2$.

Задание

С помощью среды Visual Studio разработать имитационную модель одноканального передатчика кадров с буфером ограниченной емкости. Рассчитать коэффициент (вероятность) потерь $P_П$, среднюю длину очереди $\overline{l}$.

Разработка модели

В первую очередь, для разрабатываемой модели рассчитаны типовые характеристики:

1.Коэффициент/вероятность потерь заявок ($P_{\text{п}} = \frac{N_{\text{п}}}{N_{\text{вх}}}$):

$$\rho = \frac{f_{\text{вх}}}{\mu} = \frac{1}{\tau_{\text{вх}}} : \frac{1}{\tau_{\text{п}}} = \frac{\tau_{\text{п}}}{\tau_{\text{вх}}} = \frac{8}{2} = 4$$

- Для $L = 2$, $P_{\text{п}} = P(L+1) = \frac{1-\rho}{1-\rho^{L+2}}\rho^{L+1} = \frac{1-4}{1-4^4}4^3 \approx 0.752$
- Для $L = 3$, $P_{\text{п}} = P(L+1) = \frac{1-\rho}{1-\rho^{L+2}}\rho^{L+1} = \frac{1-4}{1-4^5}\rho^4 \approx 0.751$
- Для $L = 5$, $P_{\text{п}} = P(L+1) = \frac{1-\rho}{1-\rho^{L+2}}\rho^{L+1} = \frac{1-4}{1-4^7}4^6 \approx 0.750$

2. Средняя длина очереди ($\bar{l} = \sum_{l=1}^{L} lP(l)$):

- Для $L = 2$, $\bar{l} = \frac{\rho^2}{1-\rho^{L+2}} \cdot \frac{1-\rho^L(1+L-L\rho)}{1-\rho} \approx 1{,}69$
- Для $L = 3$, $\bar{l} = \frac{\rho^2}{1-\rho^{L+2}} \cdot \frac{1-\rho^L(1+L-L\rho)}{1-\rho} \approx 2{,}67$
- Для $L = 5$, $\bar{l} = \frac{\rho^2}{1-\rho^{L+2}} \cdot \frac{1-\rho^L(1+L-L\rho)}{1-\rho} \approx 4{,}67$

Далее, реализуем данную модель (используя блок-схему из раздела «Описание модели») на языке C# с помощью среды Visual Studio [2-7]. Модель состоит из трех классов: Simulation (класс для имитации передатчика), ExponentialDistribution (класс экспоненциального распределения – вида распределения, по которому работает передатчик), Main (класс для эксперимента над созданной моделью). Коды каждого из классов представлены ниже.

Класс Simulation

```
public class Simulation
{
    private int L; //емкость буффера
    private int seed; //значение для инициализации распределения
    private double appearanceAverageTime; //время появления заявки
    private double serviceAverageTime; //время обслуживания заявки
    private Random random;
    private int requestNumber; //поступившие заявки
    private int maxRequestNumber; //максимальное количество поступивших заявок
    private int lostNumber; //потерянные заявки
    private int n; //текущее состояние
    private double[] queueLength; //длина очереди
    private double averageQueueLength; // средняя длина очереди
```

```
//Конструктор
public Simulation(int L, int seed)
{
    this.L = L;
    this.seed = seed;
    appearanceAverageTime = 2;
    serviceAverageTime = 8;
    random = new Random(seed);
    requestNumber = 0;
    maxRequestNumber = 10000;
    lostNumber = 0;
    n = 0;
    queueLength = new double[maxRequestNumber + 1];
}

//Получить среднюю длину очереди
public double getAverageQueueLength()
{
    if (queueLength.Length != 0)
    {
        double sigma = 0;
        foreach (double var in queueLength)
            sigma += var;
        return sigma / queueLength.Length;
    }
    return 0;
}

//Получить емкость
public int getL()
{
    return L;
}

//Получить значение для инициализации распределения
public int getSeed()
{
    return seed;
}

//Получить максимальное число входных заявок
public int getMaxRequestNumber()
{
    return maxRequestNumber;
}

//Генерация случайного числа
public double generateRandomNumber(double value)
{
    ExponentialDistribution ed = new ExponentialDistribution();
    return ed.getRandom(random, 1 / value);
}

//Имитация
public double simulate()
{
    double enterTime = generateRandomNumber(appearanceAverageTime);
    double exitTime = enterTime + generateRandomNumber(serviceAverageTime);
    n++;
    requestNumber++;
    enterTime += generateRandomNumber(appearanceAverageTime);
    while (requestNumber != maxRequestNumber)
    {
        if (enterTime < exitTime)
        {
            if (n == L + 1)
                lostNumber++;
            else
```

```
                    n++;
                requestNumber++;
                enterTime += generateRandomNumber(appearanceAverageTime);
            }
            else
            {
                n--;
                if (n == 0)
                {
                    exitTime = enterTime + generateRandomNumber(serviceAverageTime);
                    n++;
                    requestNumber++;
                    enterTime += generateRandomNumber(appearanceAverageTime);
                }
                else
                {
                    exitTime += generateRandomNumber(serviceAverageTime);
                }
            }
            queueLength[requestNumber] = n - 1 >= 0 ? n - 1 : 0;
        }
        return (double)lostNumber / maxRequestNumber;
    }
}
```

Класс ExponentialDistribution

```
public class ExponentialDistribution
{
    //Получить случайное число
    public double getRandom(Random r, double p)
    {
        return -(Math.Log(r.NextDouble()) / p);
    }
}
```

Класс Main

```
class Program
{
    static void Main(string[] args)
    {
          Console.WriteLine("Enter L(Buffer size):");
          int L = Convert.ToInt32(Console.ReadLine());
          Console.WriteLine("Enter seed:");
          int seed = Convert.ToInt32(Console.ReadLine());
          Simulation simulation = new Simulation(L, seed);
          Console.WriteLine("Buffer size = " + simulation.getL());
          Console.WriteLine("Seed = " + simulation.getSeed());
          Console.WriteLine("Maximum request value = " +
          simulation.getMaxRequestNumber());
          Console.WriteLine("Coefficient of lost = " + simulation.simulate());
          Console.WriteLine("Average queue length = " +
          simulation.getAverageQueueLength());
          Console.Write("Press any key to continue: ");
          Console.ReadKey();
      }
}
```

Пример работы программы представлен на рисунках 2-4.

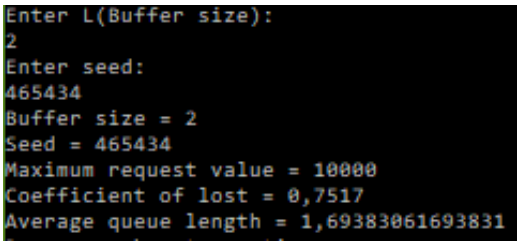

Рисунок 2 – Работа программы для L = 2 и seed = 465434

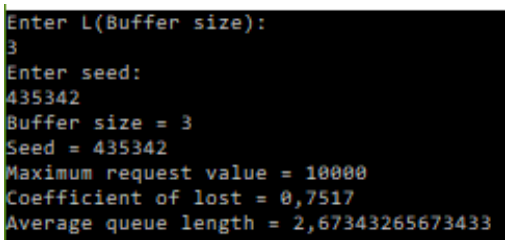

Рисунок 3 – Работа программы для L = 3 и seed = 435342

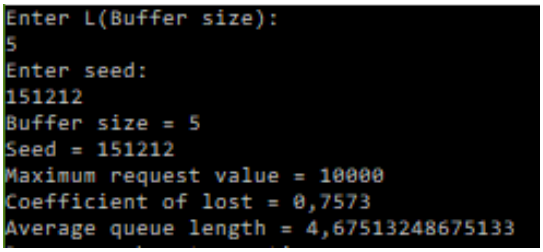

Рисунок 4 – Работа программы для L = 5 и seed = 151212

Как видно, из рисунков 2-4, практические значения для коэффициента (вероятность) потерь (Coefficient of lost), среднюю длину очереди (Average queue length) практически совпадают с теоретическими показателями (рассчитаны в начале раздела «Разработка модели»). Следовательно, разработанную с помощью Visual Studio модель можно считать корректной.

Заключение

В данной статье описана модель одноканального передатчика кадров с буфером ограниченной емкости.

Результаты исследования

Для данной модели передатчика выполнено следующее:

- с помощью Visual Studio разработана модель работы передатчика, представленной с помощью блок-схемы в разделе «Описание модели»;
- рассчитаны и сравнены теоретические значения коэффициента потерь заявок и средней длину очереди;

Теоретические и практические значения коэффициента потерь и средней длины очереди совпадают, следовательно, разработанную модель передатчика можно считать корректной.

Литература

1. Решетков К. Ю. Роль моделирования в управлении конкурентоспособным и устойчивым развитием инновационных предпринимательских структур // Бизнес в законе. 2015. № 4. С. 157-162
2. Строгалев В. П., Толкачева И. О. Имитационное моделирование. Издание: «МГТУ им. Н. Э. Баумана». 2018. 296 С.
3. Документация по Visual Studio. Добро пожаловать в интегрированную среду разработки Visual Studio [Электронный ресурс] // URL: https: //docs. microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019
4. Клейнберг, Дж., Тардос Е. Алгоритмы: разработка и применение. Классика Computers Science / Пер. с англ. Е. Матвеева. — СПб.: Питер, 2016. 800 С.
5. Лафоре Р. Структуры данных и алгоритмы в Java. Классика Computers Science. 2-е изд. — СПб.: Питер, 2013. 704 С.
6. Бхаргава А. Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих. - СПб.: Питер, 2017. 288 С.

Literature

1. Reshetkov K. U. The role of modeling in the management of competitive and sustainable development of innovative entrepreneurial structures // Business in Law. 2015. No. 4. P. 157-162

2. Strogalev V. P., Tolkacheva I. O. Simulation modeling. Edition: “MSTU named after N. E. Bauman". 2018. 296 P.
3. Visual Studio Documentation. Welcome to the Visual Studio integrated development environment [Electronic resource] // URL: https://docs. microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019
4. Kleinberg, J., Tardos E. Algorithms: development and application. Classics Computer Science / Per. from English E. Matveeva. - SPb.: Peter, 2016. 800 P.
5. Lafore R. Data structures and algorithms in Java. Classic Computer Science. 2nd ed. - SPb.: Peter, 2013. 704 P.
6. Bhargava A. Grokai algorithms. Illustrated guide for programmers and curious. - SPb.: Peter, 2017. 288 P.