Московский государственный технический университет

имени Н.Э. Баумана (Калужский филиал)

Кафедра высшей математики

Курсовая работа

по курсу «Исследование операций»

Имитационное моделирование системы массового обслуживания

Калуга 2009

Задание

Задание на работу: Составить имитационную модель и рассчитать показатели эффективности системы массового обслуживания (СМО) со следующими характеристиками:

- число каналов обслуживания n; максимальная длина очереди т;

- поток поступающих в систему заявок простейший со средней интенсивностью λ и показательным законом распределения времени между поступлением заявок;

- поток обслуживаемых в системе заявок простейший со средней интенсивностью µ и показательным законом распределения времени обслуживания.

Сравнить найденные значения показателей с результатами. полученными путем численного решения уравнении Колмогорова для вероятностей состояний системы. Значения параметров СМО приведены в таблице.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер варианта | n | m | λ | µ |
| 7 | 3 | 2 | 4.0 | 1.0 |

Оглавление

Введение

Глава 1. Основные характеристики CМО и показатели их эффективности

1.1 Понятие марковского случайного процесса

1.2 Потоки событий

1.3 Уравнения Колмогорова

1.4 Финальные вероятности и граф состояний СМО

1.5 Показатели эффективности СМО

1.6 Основные понятия имитационного моделирования

1.7 Построение имитационных моделей

Глава 2. Аналитическое моделирование СМО

2.1 Граф состояний системы и уравнения Колмогорова

2.2 Расчет показатели эффективности системы по финальным вероятностям

Глава 3. Имитационное моделирование СМО

3.1 Алгоритм метода имитационного моделирования СМО (пошаговый подход)

3.2 Блок-схема программы

3.3 Расчет показателей эффективности СМО на основе результатов ее имитационного моделирования

3.4 Статистическая обработка результатов и их сравнение с результатами аналитического моделирования

Заключение

Литература

Приложение 1

Приложение 2

Введение

При исследовании операций часто приходится сталкиваться с системами, предназначенными для многоразового использования при решении однотипных задач. Возникающие при этом процессы получили название процессов обслуживания, а системы – систем массового обслуживания (СМО).

Каждая СМО состоит из определенного числа обслуживающих единиц (приборов, устройств, пунктов, станций), которые называются каналами обслуживания. Каналами могут быть линии связи, рабочие точки, вычислительные машины, продавцы и др. По числу каналов СМО подразделяют на одноканальные и многоканальные.

Заявки поступают в СМО обычно не регулярно, а случайно, образуя так называемый случайный поток заявок (требований). Обслуживание заявок также продолжается какое-то случайное время. Случайный характер потока заявок и времени обслуживания приводит к тому, что СМО оказывается загруженной неравномерно: в какие-то периоды времени скапливается очень большое количество заявок (они либо становятся в очередь, либо покидают СМО не обслуженными), в другие же периоды СМО работает с недогрузкой или простаивает.

Предметом теории массового обслуживания является построение математических моделей, связывающих заданные условия работы СМО (число каналов, их производительность, характер потока заявок и т.п.) с показателями эффективности СМО, описывающими ее способность справляться с потоком заявок.

В качестве показателей эффективности СМО используются:

- Абсолютная пропускная способность системы (А), т.е. среднее число заявок, обслуживаемых в единицу времени;

- относительная пропускная способность (Q), т.е. средняя доля поступивших заявок, обслуживаемых системой;

- вероятность отказа обслуживания заявки ();

- среднее число занятых каналов (k);

- среднее число заявок в СМО ();

- среднее время пребывания заявки в системе ();

- среднее число заявок в очереди ();

- среднее время пребывания заявки в очереди ();

- среднее число заявок, обслуживаемых в единицу времени;

- среднее время ожидания обслуживания;

- вероятность того, что число заявок в очереди превысит определенное значение и т.п.

СМО делят на 2 основных типа: СМО с отказами и СМО с ожиданием (очередью). В СМО с отказами заявка, поступившая в момент, когда все каналы заняты, получает отказ, покидает СМО и в дальнейшем процессе обслуживания не участвует (например, заявка на телефонный разговор в момент, когда все каналы заняты, получает отказ и покидает СМО не обслуженной). В СМО с ожиданием заявка, пришедшая в момент, когда все каналы заняты, не уходит, а становится в очередь на обслуживание.

Одним из методов расчета показателей эффективности СМО является метод имитационного моделирования. Практическое использование компьютерного имитационного моделирования предполагает построение соответствующей математической модели, учитывающей факторы неопределенности, динамические характеристики и весь комплекс взаимосвязей между элементами изучаемой системы. Имитационное моделирование работы системы начинается с некоторого конкретного начального состояния. Вследствие реализации различных событий случайного характера, модель системы переходит в последующие моменты времени в другие свои возможные состояния. Этот эволюционный процесс продолжается до конечного момента планового периода, т.е. до конечного момента моделирования.

Глава 1. Основные характеристики CМО и показатели их эффективности

1.1 Понятие марковского случайного процесса

Пусть имеется некоторая система, которая с течением времени изменяет свое состояние случайным образом. В этом случае говорят, что в системе протекает случайный процесс.

Процесс называется процессом с дискретными состояниями, если его состояния можно заранее перечислить и переход системы из одного состояния в другое происходит скачком. Процесс называется процессом с непрерывным временем, если переходы системы из состояния в состояние происходят мгновенно.

Процесс работы СМО – это случайный процесс с дискретными состояниями и непрерывным временем.

Случайный процесс называют марковским или случайным процессом без последействия, если для любого момента времени вероятностные характеристики процесса в будущем зависят только от его состояния в данный момент и не зависят от того, когда и как система пришла в это состояние.

Рис.1.

При анализе процессов работы СМО удобно пользоваться геометрической схемой – графом состояний. Обычно состояния системы изображаются прямоугольниками, а возможные переходы из состояния в состояние – стрелками. Пример графа состояний приведен на рис.1.

1.2 Потоки событий

Поток событий – последовательность однородных событий, следующих одно за другим в случайные моменты времени.

Поток характеризуется интенсивностью λ – частотой появления событий или средним числом событий, поступающих в СМО в единицу времени.

Поток событий называется регулярным, если события следуют одно за другим через определенные равные промежутки времени.

Поток событий называется стационарным, если его вероятностные характеристики не зависят от времени. В частности, интенсивность стационарного потока есть величина постоянная: .

Поток событий называется ординарным, если вероятность попадания на малый участок времени двух и более событий мала по сравнению с вероятностью попадания одного события, т.е., если события появляются в нем поодиночке, а не группами.

Поток событий называется потоком без последействия, если для любых двух непересекающихся участков времени и число событий, попадающих на одно из них, не зависит от числа событий, попадающих на другие.

Поток событий называется простейшим (или стационарным пуассоновским), если он одновременно стационарен, ординарен и не имеет последействия.

1.3 Уравнения Колмогорова

Все переходы в системе из состояния в состояние происходят под некоторым потоком событий. Пусть система находится в некотором состоянии , из которого возможен переход в состояние , тогда можно считать, что на систему воздействует простейший поток с интенсивностью , переводящий ее из состояния в . Как только появляется первое событие потока, происходит ее переход . Для наглядности на графе состояний у каждой стрелки, соответствующей переходу, указывается интенсивность . Такой размеченный граф состояний позволяет построить математическую модель процесса, т.е. найти вероятности всех состояний как функции времени. Для них составляются дифференциальные уравнения, называемые уравнениями Колмогорова.

Правило составлений уравнений Колмогорова: В левой части каждого из уравнений стоит производная по времени от вероятности данного состояния. В правой части стоит сумма произведений всех состояний, из которых возможен переход в данное состояние, на интенсивности соответствующих потоков событий минус суммарная интенсивность всех потоков, выводящих систему из данного состояния, умноженная на вероятность данного состояния.

Например, для графа состояний, приведенного на рис.1, уравнения Колмогорова имеют вид:

Т.к. в правой части системы каждое слагаемое входит 1 раз со знаком и 1 раз со знаком , то, складывая все уравнений, получим, что

,

,

. (1.3.1)

Следовательно, одно из уравнений системы можно отбросить и заменить уравнением (1.3.1).

Чтобы получить конкретное решение надо знать начальные условия, т.е. значения вероятностей в начальный момент времени.

1.4 Финальные вероятности и граф состояний СМО

При достаточно большом времени протекания процессов в системе (при ) могут устанавливаться вероятности состояний, не зависящие от времени, которые называются финальными вероятностями, т.е. в системе устанавливается стационарный режим. Если число состояний системы конечно, и из каждого из них за конечное число шагов м. перейти в любое другое состояние, то финальные вероятности существуют, т.е.

Смысл финальных вероятностей состоит в том, что они равны среднему относительному времени нахождения системы в данном состоянии.

Т.к. в стационарном состоянии производные по времени равны нулю, то уравнения для финальных вероятностей получаются из уравнений Колмогорова путем приравнивания нулю их правых частей.

Графы состояний, используемые в моделях систем массового обслуживания, называются схемой гибели и размножения. Такое название обусловлено тем, что эта схема используется в биологических задачах, связанных с изучением численности популяции. Его особенность состоит в том, что все состояния системы можно представить в виде цепочки, в которой каждое из состояний связано с предыдущим и последующим (рис 2).

Рис. 2

Предположим, что все потоки, переводящие систему из одного состояния в другое, простейшие. По графу, представленному на рис. 2, составим уравнения для финальных вероятностей системы. Они имеют вид:



Получается система из (n+1) уравнения, которая решается методом исключения. Этот метод заключается в том, что последовательно все вероятности системы выражаются через вероятность .

,

,

.

Подставляя эти выражения в последнее уравнение системы, находим , затем находим остальные вероятности состояний СМО.

1.5 Показатели эффективности СМО

Цель моделирования СМО состоит в том, чтобы рассчитать показатели эффективности системы через ее характеристики. В качестве показателей эффективности СМО используются:

- абсолютная пропускная способность системы (А), т.е. среднее число заявок, обслуживаемых в единицу времени;

- относительная пропускная способность (Q), т.е. средняя доля поступивших заявок, обслуживаемых системой;

- вероятность отказа (), т.е. вероятность того, что заявка покинет СМО не обслуженной;

- среднее число занятых каналов (k);

- среднее число заявок в СМО ();

- среднее время пребывания заявки в системе ();

- среднее число заявок в очереди () – длина очереди;

- среднее число заявок в системе ();

- среднее время пребывания заявки в очереди ();

- среднее время пребывания заявки в системе ()

- степень загрузки канала (), т.е. вероятность того, что канал занят;

- среднее число заявок, обслуживаемых в единицу времени;

- среднее время ожидания обслуживания;

- вероятность того, что число заявок в очереди превысит определенное значение и т.п.

Доказано, что при любом характере потока заявок, при любом распределении времени обслуживания, при любой дисциплине обслуживания, среднее время пребывания заявки в системе (очереди) равна среднему числу заявок в системе (очереди), деленному на интенсивность потока заявок, т.е.

 (1.5.1)

 (1.5.2)

Формулы (1.5.1) и (1.5.2) называются формулами Литтла. Они вытекают из того, что в предельном стационарном режиме среднее число заявок, прибывающих в систему, равно среднему числу заявок, покидающих ее, т.е. оба потока заявок имеют одну и ту же интенсивность .

Формулы для вычисления показателей эффективности приведены в таб. 1.

Таблица 1.

|  |  |  |
| --- | --- | --- |
| Показатели | Одноканальная СМО с ограниченной очередью | Многоканальная СМО с ограниченной очередью |
| Финальные вероятности | ,  |  |
| Вероятность отказа |  |  |
| Абсолютная пропускнаяспособность |  |  |
| Относительная пропускная способность |  |  |
| Среднее число заявок в очереди |  |  |
| Среднее число заявок под обслуживанием |  |  |
| Среднее число заявок в системе |  |  |

1.6 Основные понятия имитационного моделирования

Основная цель имитационного моделирования заключается в воспроизведении поведения изучаемой системы на основе анализа наиболее существенных взаимосвязей ее элементов.

Компьютерное имитационное моделирование следует рассматривать как статический эксперимент.

Из теории функций случайных величин известно, что для моделирования случайной величины с любой непрерывной и монотонно возрастающей функцией распределения достаточно уметь моделировать случайную величину , равномерно распределенную на отрезке . Получив реализацию случайной величины , можно найти соответствующую ей реализацию случайной величины , так как они связаны равенством

 (1.6.1)

Предположим, что в некоторой системе массового обслуживания время обслуживания одной заявки распределено по экспоненциальному закону с параметром , где - интенсивность потока обслуживания. Тогда функция распределения времени обслуживания имеет вид

Пусть - реализация случайной величины , равномерно распределенной на отрезке , а - соответствующая ей реализация случайного времени обслуживания одной заявки. Тогда, согласно (1.6.1),

.

1.7 Построение имитационных моделей

Первый этап создания любой имитационной модели – этап описания реально существующей системы в терминах характеристик основных событий. Эти события, как правило, связаны с переходами изучаемой системы из одного возможного состояния в другое и обозначаются как точки на временной оси. Для достижения основной цели моделирования достаточно наблюдать систему в моменты реализации основных событий.

Рассмотрим пример одноканальной системы массового обслуживания. Целью имитационного моделирования подобной системы является определение оценок ее основных характеристик, таких, как среднее время пребывания заявки в очереди, средняя длина очереди и доля времени простоя системы.

Характеристики самого процесса массового обслуживания могут изменять свои значения либо в момент поступления новой заявки на обслуживание, либо при завершении обслуживания очередной заявки. К обслуживанию очередной заявки СМО может приступить немедленно (канал обслуживания свободен), но не исключена необходимость ожидания, когда заявке придется занять место в очереди (СМО с очередью, канал обслуживания занят). После завершения обслуживания очередной заявки СМО может сразу приступить к обслуживанию следующей заявки, если она есть, но может и простаивать, если таковая отсутствует. Необходимую информацию можно получить, наблюдая различные ситуации, возникающие при реализациях основных событий. Так, при поступлении заявки в СМО с очередью при занятом канале обслуживания длина очереди увеличивается на 1. Аналогично длина очереди уменьшается на 1, если завершено обслуживание очередной заявки и множество заявок в очереди не пусто.

Для эксплуатации любой имитационной модели необходимо выбрать единицу времени. В зависимости от природы моделируемой системы такой единицей может быть микросекунда, час, год и т.д.

Так как по своей сути компьютерное имитационное моделирование представляет собой вычислительный эксперимент, то его наблюдаемые результаты в совокупности должны обладать свойствами реализации случайной выборки. Лишь в этом случае будет обеспечена корректная статистическая интерпретация моделируемой системы.

При компьютерном имитационном моделировании основной интерес представляют наблюдения, полученные после достижения изучаемой системой стационарного режима функционирования, так как в этом случае резко уменьшается выборочная дисперсия.

Время, необходимое для достижения системой стационарного режима функционирования, определяется значениями ее параметров и начальным состоянием.

Поскольку основной целью является получение данных наблюдений с возможно меньшей ошибкой, то для достижения этой цели можно:

1) увеличить длительность времени имитационного моделирования процесса функционирования изучаемой системы. В этом случае не только увеличивается вероятность достижения системой стационарного режима функционирования, но и возрастает число используемых псевдослучайных чисел, что также положительно влияет на качество получаемых результатов.

2) при фиксированной длительности времени Т имитационного моделирования провести N вычислительных экспериментов, называемых еще прогонами модели, с различными наборами псевдослучайных чисел, каждый из которых дает одно наблюдение. Все прогоны начинаются при одном и том же начальном состоянии моделируемой системы, но с использованием различных наборов псевдослучайных чисел. Преимуществом этого метода является независимость получаемых наблюдений , показателей эффективности системы. Если число N модели достаточно велико, то границы симметричного доверительного интервала для параметра определяются следующим образом:

, , т.е. , где

математическое ожидание (среднее значение), находится по формуле

,

 исправленная дисперсия,

,

N – число прогонов программы, – надежность, .

Глава 2. Аналитическое моделирование СМО

2.1 Граф состояний системы и уравнения Колмогорова

Рассмотрим четырехканальную систему массового обслуживания (n = 3) с максимальной длиной очереди равной трем (m = 2). В СМО поступает простейший поток заявок со средней интенсивностью λ = 4.0 и показательным законом распределения времени между поступлением заявок. Поток обслуживаемых в системе заявок является простейшим со средней интенсивностью μ = 1.0 и показательным законом распределения временем обслуживания.

Данная система имеет 9 состояний, обозначим их:

S0 – все каналы пусты, очередь пуста;

S1 – 1 канал занят, очередь пуста;

S2 – 2 канала заняты, очередь пуста;

S3 – 3 канала заняты, очередь пуста;

S4 – 3 канала заняты, в очереди 1 заявка;

S5 – 3 канала заняты, в очереди 2 заявки;

Вероятности прихода системы в состояния S0, S1, S2, …, S5 соответственно равны Р0, Р1, Р2, …, Р5.

Граф состояний системы массового обслуживания представляет собой схему гибели и размножения. Все состояния системы можно представить в виде цепочки, в которой каждое из состояний связано с предыдущим и последующим.

Рис. 3

Для построенного графа запишем уравнения Колмогорова:

Чтобы решить данную систему зададим начальные условия:

Систему уравнений Колмогорова (систему дифференциальных уравнений) решим численным методом Эйлера с помощью программного пакета Maple 8 (см. Приложение 1).

Метод Эйлера

где- в нашем случае, это правые части уравнений Колмогорова, n=7.

 (1)

Выберем шаг по времени . Предположим , где Т – это время, за которое система выходит на стационарный режим. Отсюда получаем число шагов

.

Последовательно N раз вычисляя по формуле (1) получим зависимости вероятностей состояний системы от времени, приведенной на рис.4. Очевидно, что уже при система выходит на стационарный режим. Значения вероятностей СМО при равны:

Зависимости вероятностей состояний системы от времени

**P0 - черный**

**P1 – синий**

**P2 – красный**

**P3 – голубой**

**P4 – малиновый**

**P5 – зеленый**

Рис. 4

2.2 Финальные вероятности системы

При достаточно большом времени протекания процессов в системе () могут устанавливаться вероятности состояний, не зависящие от времени, которые называются финальными вероятностями, т.е. в системе устанавливается стационарный режим. Если число состояний системы конечно, и из каждого из них за конечное число шагов можно перейти в любое другое состояние, то финальные вероятности существуют, т.е.

Т.к. в стационарном состоянии производные по времени равны 0, то уравнения для финальных вероятностей получаются из уравнений Колмогорова путем приравнивания правых частей 0. Запишем уравнения для финальных вероятностей для нашей СМО.

Решим данную систему линейных уравнений с помощью программного пакета Maple 8 (см. Приложение 1).

Получим финальные вероятности системы:

Сравнение вероятностей, полученных из системы уравнений Колмогорова при , с финальными вероятностями показывает, что ошибки

равны:

Т.е. достаточно малы. При увеличении интервала времени до , погрешности должны были стать еще меньше.

Это подтверждает правильность полученных результатов.

2.3 Расчет показатели эффективности системы по финальным вероятностям

Найдем показатели эффективности системы массового обслуживания. Наиболее важными являются следующие показатели:

1) Вероятность отказа в обслуживании заявки, т.е. вероятность того, что заявка покидает систему не обслуженной. В нашем случае заявке отказывается в обслуживании, если все 4 канала заняты, и очередь максимально заполнена (т.е. 3 человека в очереди), это соответствует состоянию системы S7. Т.к. вероятность прихода системы в состояние S7 равна Р7 , то

2) Относительная пропускная способность – это средняя доля поступивших заявок, обслуживаемых системой.

3) Абсолютная пропускная способность – это среднее число заявок, обслуживаемых в единицу времени.

4) Длина очереди, т.е. среднее число заявок в очереди. Длина очереди равна сумме произведений числа человек в очереди на вероятность соответствующего состояния.

5) Среднее время пребывания заявки в очереди определяется формулой Литтла

6) Среднее число занятых каналов определяется следующим образом:

Глава 3. Имитационное моделирование СМО

3.1 Алгоритм метода имитационного моделирования СМО (пошаговый подход)

Рассмотрим четырехканальную систему массового обслуживания (n = 3) с максимальной длиной очереди равной четырем (m = 2). В СМО поступает простейший поток заявок со средней интенсивностью λ = 4 и показательным законом распределения времени между поступлением заявок. Поток обслуживаемых в системе заявок является простейшим со средней интенсивностью μ = 1 и показательным законом распределения временем обслуживания.

Для имитации СМО воспользуемся одним из методов статистического моделирования – имитационным моделированием. Будем использовать пошаговый подход. Суть этого подхода в том, что состояния системы рассматриваются в последующие моменты времени, шаг между которыми является достаточно малым, чтобы за его время произошло не более одного события.

Выберем шаг по времени (). Он должен быть много меньше среднего времени поступления заявки () и среднего времени ее обслуживания (), т.е.

, где (3.1.1)

Исходя из условия (3.1.1) определим шаг по времени .

Время поступления заявки в СМО и время ее обслуживания являются случайными величинами. Поэтому, при имитационном моделировании СМО их вычисление производится с помощью случайных чисел.

Рассмотрим поступление заявки в СМО. Вероятность того, что на интервале в СМО поступит заявка, равна:

.

Сгенерируем случайное число , и, если , то будем считать, что заявка на данном шаге в систему поступила, если , то не поступила.

В программе это осуществляет функция . Интервал времени примем постоянным и равным 0,001, тогда отношение будет равно 1000 . Если заявка поступила, то она принимает значение «истина», в противном случае значение «ложь».

function sob: boolean;

var r:real;

begin

r := Random(1000)/1000;

if r <= (i\_deltaT\*r\_Lamda) then

sob:= true

else

sob:= false;

end;

Рассмотрим теперь обслуживание заявки в СМО. Время обслуживания заявки в системе определяется выражением

,

где – случайное число. В программе время обслуживания определяется с помощью функции .

function tok: real;

var r:real;

t\_ob: real;

begin

r:= Random(1000)/1000;

t\_ob:= -1/r\_Mu\*ln(1-r);

tok:= t\_ob;

end;

Алгоритм метода имитационного моделирования можно сформулировать следующим образом. Время работы СМО (Т) разбивается на шаги по времени dt, на каждом из них выполняется ряд действий. Вначале определяются состояния системы (занятость каналов, длина очереди), затем, с помощью функции , определяется, поступила ли на данном шаге заявка или нет.

Если поступила, и, при этом имеются свободные каналы, то с помощью функции генерируем время обработки заявки и ставим ее на обслуживание. Если все каналы заняты, а длина очереди меньше 4, то помещаем заявку в очередь, если же длина очереди равна 4, то заявке будет отказано в обслуживании.

В случае, когда на данном шаге заявка не поступала, а канал обслуживания освободился, проверяем, есть ли очередь. Если есть, то из очереди заявку ставим на обслуживание в свободный канал. После проделанных операций время обслуживания для занятых каналов уменьшаем на величину шага dt. По истечении времени Т, т.е., после моделирования работы СМО, вычисляются показатели эффективности работы системы и результаты выводятся на экран.

3.2 Блок-схема программы

Блок-схема программы, реализующей описанный алгоритм, приведена на рис.5. Распишем некоторые блоки более подробно.

Блок 1. Задание начальных значений параметров. Пользователем задаются значения: Time – время работы системы,

r\_la – интенсивность потока поступления заявок (),

r\_mu – интенсивность потока обслуживания заявок ().

Значения, задаваемые программой:

i\_deltaT:=0.001 – шаг по времени;

i\_post:=0;

i\_otk:= 0;

i\_obsl:=0;

i\_och:=0;

i\_tobs:=0;

t:=0 – параметр, следящий, сколько времени проработала система;

i\_obsl:=0 – число обслуженных заявок;

i\_tobs:=0 – общее время обслуживания заявок в системе;

i\_post:=0 – число поступивших в СМО заявок;

i\_otk:=0 – число заявок, которым было отказано в обслуживании;

i\_obsl:=0 – число обслуженных заявок;

sost:='s1' – состояние системы (изначально устанавливаем СМО в состояние s1, т.е. все каналы свободны);

for i :=1 to 3 do

begin

t\_och[i] := 0;- обнуляем время пребывания СМО в состояниях с длиной очереди 1, 2.

t\_okonch[i] := 0; – время окончания обслуживания заявки во всех 3 каналах устанавливаем в 0, т.к. каналы пусты;

end;

Рис.5.

Блок 3. Задание состояний системы. Выделим у данной 3-х канальной системы 9 различных состояний: s1, s2,.. s9. СМО находится в состоянии s1, когда система свободна; s2..s7 – хотя бы один канал свободен; в состоянии s8, когда все каналы заняты, и есть место в очереди; в состоянии s9 - все каналы заняты, и очередь достигла максимальной длины (och=2).

Задаем состояния системы:

\_sost:=1;

for i:=0 to 3 do

begin

if t\_okonch[i+1]>0 then i\_temp:= 1

else i\_temp:= 0;

\_sost:=\_sost + i\_temp\* round(Exp(i\*ln(2)));

end;

if \_sost<7 then begin Sost:= \_sost; exit; end;

if (\_sost= 7) and (i\_och< 2) then Sost:= \_sost

else

Sost := \_sost+ 1;

Блок 4. Изменение времени пребывания СМО в состояниях с длиной очереди 1, 2. Это реализуется следующим программным кодом:

if i\_och > 0 then t\_och[i\_och]:= t\_och[i\_och]+i\_deltaT;

В блоках 9 и 16 присутствует такая операция, как помещение заявки на обслуживание в свободный канал. Просматриваются, начиная с первого, все каналы, когда выполняется условие (канал свободен), в него подается заявка, т.е. генерируется время окончания обслуживания заявки.

for i:=1 to 2 do

begin

if t\_okonch[i]>0 then t\_okonch[i]:= t\_okonch[i] - i\_deltaT

else

if i\_och >0 then

begin

Dec(i\_och);

t\_okonch[i]:=tok;

i\_tobs:=i\_tobs + t\_okonch[i];

end;

end;

Блок 17 реализуется следующим программным кодом:

for i:=1 to 2 do

if t\_okonch[i]>0 then

t\_okonch[i]:= t\_okonch[i] - i\_deltaT;

Алгоритм метода имитационного моделирования реализован на языке программирования C#.

3.3 Расчет показателей эффективности СМО на основе результатов ее имитационного моделирования

Наиболее важными являются такие показатели, как:

1) Вероятность отказа в обслуживании заявки, т.е. вероятность того, что заявка покидает систему не обслуженной. В нашем случае заявке отказывается в обслуживании, если все 3 канала заняты, и очередь максимально заполнена (т.е. 2 человека в очереди). Для нахождения вероятности отказа разделим время пребывания СМО в состоянии с очередью 4 на общее время работы системы.

2) Относительная пропускная способность – это средняя доля поступивших заявок, обслуживаемых системой.

3) Абсолютная пропускная способность – это среднее число заявок, обслуживаемых в единицу времени.

4) Длина очереди, т.е. среднее число заявок в очереди. Длина очереди равна сумме произведений числа человек в очереди на вероятность соответствующего состояния. Вероятности состояний найдем как отношение времени нахождения СМО в этом состоянии к общему времени работы системы.

5) Среднее время пребывания заявки в очереди определяется формулой Литтла

6) Среднее число занятых каналов определяется следующим образом:

7) Процент заявок, которым было отказано в обслуживании, находится по формуле

8) Процент обслуженных заявок находится по формуле

3.4 Статистическая обработка результатов и их сравнение с результатами аналитического моделирования

Т.к. показатели эффективности получаются в результате моделирования СМО в течение конечного времени, они содержат случайную компоненту. Поэтому, для получения более надежных результатов нужно провести их статистическую обработку. С этой целью оценим доверительный интервал для них по результатам 20 прогонов программы.

Величина попадает в доверительный интервал, если выполняется неравенство

, где

математическое ожидание (среднее значение), находится по формуле

,

 исправленная дисперсия,

,

N=20 – число прогонов,

 – надежность. При и N=20 .

Результат работы программы представлен на рис.6.

Рис.6.

Для удобства сравнения результатов, полученных различными методами моделирования, представим их в виде таблицы.

Таблица 2.

|  |  |  |
| --- | --- | --- |
| Показатели эффективности СМО | Результаты аналитического моделирования | Результаты имитационного моделирования |
| Нижняя граница доверительного интервала  | Верхняя граница доверительного интервала  |
| Вероятность отказа | 0,33355 | 0,238 | 0,35979 |
| Относительная пропускная способность | 0,66645 | 0,64 | 0,761 |
| Абсолютная пропускная способность | 2.66579 | 2,56 | 3,048 |
| Средняя длина очереди | 0,917264 | 0,696 | 0,9566 |
| Среднее время пребывания заявки в очереди | 0,229316 | 0,1739 | 0,2394 |
| Среднее число занятыхканалов | 2,665798 | 2,561 | 3,048 |

Из табл. 2 видно, что результаты, полученные при аналитическом моделировании СМО, попадают в доверительный интервал, полученный по результатам имитационного моделирования. Т.е., результаты, полученные разными методами, согласуются.

Заключение

В данной работе рассмотрены основные методы моделирования СМО и расчета показателей их эффективности.

Проведено моделирование четырехканальной СМО с максимальной длиной очереди равной 4 с помощью уравнений Колмогорова, а также, найдены финальные вероятности состояний системы. Рассчитаны показатели ее эффективности.

Проведено имитационное моделирование работы такой СМО. На языке программирования Delphi составлена программа, имитирующая ее работу. Проведена серия расчетов, по результатам которых найдены значения показателей эффективности системы и выполнена их статистическая обработка.

Полученные при имитационном моделировании результаты согласуются с результатами аналитического моделирования.

Литература

1. Вентцель Е.С. Исследование операций. – М.: Дрофа, 2004. – 208 с.
2. Волков И.К., Загоруйко Е.А. Исследование операций. – М.: Изд.-во МГТУ им. Н.Э. Баумана, 2002. – 435 с.
3. Волков И.К., Зуев С.М., Цветкова Г.М. Случайные процессы. – М.: Изд.-во МГТУ им. Н.Э. Баумана, 2000. – 447 с.
4. Гмурман В.Е. Руководство к решению задач по теории вероятностей и математической статистике. – М.: Высшая школа, 1979. – 400 с.
5. Ивницкий В.Л. Теория сетей массового обслуживания. – М.: Физматлит, 2004. – 772 с.
6. Исследование операций в экономике/ под ред. Н.Ш. Кремера. – М.: Юнити, 2004. – 407 с.
7. Таха Х.А. Введение в исследование операций. – М.: ИД «Вильямс», 2005. – 902 с.
8. Харин Ю.С., Малюгин В.И., Кирлица В.П. и др. Основы имитационного и статистического моделирования. – Минск: Дизайн ПРО, 1997. – 288 с.

Приложение 1

Листинг программы аналитического моделирования

Программа написана в математическом пакете Maple 8.

> # Дано

la:=4:

m:=1:

Находим финальные вероятности

> usl := {m\*Pf[1]-la\*Pf[0] = 0,

2\*m\*Pf[2]-la\*Pf[1] = 0,

3\*m\*Pf[3]-la\*Pf[2] = 0,

3\*m\*Pf[4]-la\*Pf[3] = 0,

3\*m\*Pf[5]-la\*Pf[4] = 0,

Pf[0]+Pf[1]+Pf[2]+Pf[3]+Pf[4]+Pf[5]= 1 }:

> s:=evalf(solve(usl) ):

> for i from 0 by 1 to 5 do

Pf[i]:=subs( s, Pf[i] );

end;

>

> # Уравнения Колмогорова

> del\_t:=0.01: #шаг по времени

T:=10: # Время, за которое система выходит на стационарный режим

N:=trunc(T/del\_t); # Число шагов

>

> # Присваиваем начальные значения

p[i,c]:=array(0..5,0..N):

p[0,0]:=1:

for i from 1 by 1 to 5 do p[i,0]:=0 end:

> # Метод Эйлера

jj:=0:

for k from 1 to N do

v:=jj:

j:=trunc(100\*v):

jj:=jj+del\_t:

p[0,k] := p[0,j]+( m\*p[1,j]-la\*p[0,j])\*del\_t:

p[1,k] := p[1,j]+( la\*p[0,j]+2\*m\*p[2,j]-p[1,j]\*(m+la) )\*del\_t:

p[2,k] := p[2,j]+( la\*p[1,j]+3\*m\*p[3,j]-p[2,j]\*(2\*m+la) )\*del\_t:

p[3,k] := p[3,j]+( la\*p[2,j]+3\*m\*p[4,j]-p[3,j]\*(3\*m+la) )\*del\_t:

p[4,k] := p[4,j]+( la\*p[3,j]+3\*m\*p[5,j]-p[4,j]\*(3\*m+la) )\*del\_t:

p[5,k] := p[5,j]+( la\*p[4,j]-3\*m\*p[5,j] )\*del\_t:

end:

> for i from 0 by 1 to 5 do

P[i]=p[i,N];

end;

> # Cравним финальные вероятности с вероятностями при Т=10, водно, что они близки. R-ошибка

for i from 0 by 1 to 5 do

# Pf[i]=p[i,N]:

R[i]:=abs(Pf[i]-p[i,500]);

end;

>

> # Показатели эффективности системы

# вероятность отказа

p\_otk:=Pf[5];

> # относительная пропускная способность

Q := 1-p\_otk;

> # абсолютная пропускная способность

A := la\*Q;

> # длина очереди

lo := Pf[4]+2\*Pf[5];

> # среднее время в очереди

t0 := lo/la;

> # среднее число каналов

K := (la/m)\*Q;

>

> # Вывод графиков вероятностей

cur[i]:=array(0..5):

for i from 0 by 1 to 5 do

cur[i]:=CURVES([[0,p[i,0]],[0.05,p[i,5]],[0.07,p[i,7]],[0.1,p[i,10]],[0.15,p[i,15]],

[0.2,p[i,20]],[0.3,p[i,30]],[0.4,p[i,40]],[0.5,p[i,50]],[0.75,p[i,75]],[1,p[i,100]],

[1.25,p[i,125]],[1.5,p[i,150]],[1.75,p[i,175]],[2,p[i,200]],[2.5,p[i,250]],[3,p[i,300]],

[4,p[i,400]],[5,p[i,500]] ]):

end:

> i:=0: a[i]:=PLOT(cur[i],COLOR(RGB,0,0,0)): # P0 - черный

i:=1: a[i]:=PLOT(cur[i],COLOR(RGB,0,0,1)): # P1 - синий

i:=2: a[i]:=PLOT(cur[i],COLOR(RGB,1,0,0)): # P2 - красный

i:=3: a[i]:=PLOT(cur[i],COLOR(RGB,0,1,1)): # P3 - голубой

i:=4: a[i]:=PLOT(cur[i],COLOR(RGB,1,0,1)): # P4 - малиновый

i:=5: a[i]:=PLOT(cur[i],COLOR(RGB,0,1,0)): # P5 - зеленый

> display({a[0],a[1],a[2],a[3],a[4],a[5]});

Приложение 2

Листинг программы имитационного моделирования

Программа написана на языке Delphi7.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace CMO\_misiagin

{

class CMO\_work

{

Random R; // Генератор случайных чисел

public uint Max\_dlina\_ocheredi; // Максимальная длина очереди

public uint chislo\_kanalov\_in\_system; // Число каналов в системе

public double Lamda; // Интенсивность потока поступления заявок

public double Mu; // Интенсивность потока обслуживания заявок

public double shag\_by\_time; // Шаг по времени

public double[] t\_oconch\_obsl\_vo\_vsex\_canalax; // Время окончания обслуживания заявки во всех каналах

public double[] t\_v\_sostoijaniax\_s\_ocher; // Время пребывания СМО в состояниях с очередью

public double t\_rab\_sistem; // Время работы системы

public double summ\_t\_obsl\_zaijvok; // Суммарное время обслуживания заявок

public uint chislo\_post\_zaijavok; // Число поступивших заявок

public uint chislo\_otkaz\_zaijavok; // Число отказанных заявок

public uint chislo\_obslu\_zaijavok; // Число обслуженных заявок

uint dlina\_ocheredi; // Длина очереди

public uint Dlina\_ocheredi

{

get

{ return dlina\_ocheredi; }

set

{

if (value < 0)

{ dlina\_ocheredi = 0; }

else if (value > 2)

{ }

else

{ dlina\_ocheredi = value;}

}

}

// Состояния СМО {ССС,ССЗ,СЗС,ЗСС,СЗЗ,ЗСЗ,ЗЗС,ЗЗЗ,ЗЗЗ+1,ЗЗЗ+2}

enum Sostoijanija\_sustem { S0, S1, S2, S3, S4, S5, S6, S7, S8, S9};

Sostoijanija\_sustem tekushee\_sost\_sustem; // Текущее состояние системы

/// <summary>

/// Представляет модель системы массового обслуживания (СМО)

/// </summary>

/// <param name="p\_chislo\_kanalov\_in\_system">Число каналов в системе</param>

/// <param name="p\_Max\_dlina\_ocheredi">Максимальная длина очереди</param>

/// <param name="p\_Lamda">Интенсивность потока поступления завок</param>

/// <param name="p\_Mu">Интенсивность потока обслуживания завок</param>

/// <param name="p\_shag\_by\_time">Шаг по времени</param>

public CMO\_work(uint p\_chislo\_kanalov\_in\_system, uint p\_Max\_dlina\_ocheredi, double p\_Lamda, double p\_Mu, double p\_shag\_by\_time)

{

SetDefaults();

R = new Random();

chislo\_kanalov\_in\_system = p\_chislo\_kanalov\_in\_system;

Max\_dlina\_ocheredi = p\_Max\_dlina\_ocheredi;

Lamda = p\_Lamda;

Mu = p\_Mu;

shag\_by\_time = p\_shag\_by\_time;

t\_oconch\_obsl\_vo\_vsex\_canalax = new double[chislo\_kanalov\_in\_system];

t\_v\_sostoijaniax\_s\_ocher = new double[Max\_dlina\_ocheredi];

}

/// <summary>

/// Устанавливает значения параметров СМО по умолчанию

/// </summary>

public void SetDefaults()

{

tekushee\_sost\_sustem = Sostoijanija\_sustem.S0;

chislo\_post\_zaijavok = 0;

chislo\_otkaz\_zaijavok = 0;

chislo\_obslu\_zaijavok = 0;

t\_rab\_sistem = 0;

summ\_t\_obsl\_zaijvok = 0;

dlina\_ocheredi = 0;

for (int i = 0; i < chislo\_kanalov\_in\_system; i++)

{ t\_oconch\_obsl\_vo\_vsex\_canalax[i] = 0; }

for (int i = 0; i < Max\_dlina\_ocheredi; i++)

{ t\_v\_sostoijaniax\_s\_ocher[i] = 0; }

}

/// <summary>

/// Определяет случайным образом, поступила ли заявка

/// </summary>

/// <returns>true - поступила, false = не поступила</returns>

bool prichla\_li\_sluchain\_zaijavka()

{

double r = R.NextDouble();

if (r < (shag\_by\_time \* Lamda))

{ return true; }

return false;

}

/// <summary>

/// Определяет случайным образом время облуживания заявки

/// </summary>

/// <returns>Время облуживания заявки</returns>

double t\_obslujivania\_zaijvki()

{

double r = R.NextDouble();

return (-1/Mu\*Math.Log(1-r, Math.E));

}

/// <summary>

/// Определение текущего состояния

/// </summary>

/// <returns>Состояние системы</returns>

Sostoijanija\_sustem opred\_tec\_sost()

{

Sostoijanija\_sustem \_sostoijanija\_sustem = Sostoijanija\_sustem.S0;

int k = 0;

int zanijat\_chislo\_kanalov\_in\_system = 0;

for (int i = 0; i < chislo\_kanalov\_in\_system; i++)

{

if (t\_oconch\_obsl\_vo\_vsex\_canalax[i] > 0)

{

k = 1;

zanijat\_chislo\_kanalov\_in\_system++;

}

else

{ k = 0; }

\_sostoijanija\_sustem += k \* (i + 1);

}

if (zanijat\_chislo\_kanalov\_in\_system > 1)

{ \_sostoijanija\_sustem ++; }

return \_sostoijanija\_sustem + (int)dlina\_ocheredi;

}

/// <summary>

/// Один шаг моделирования системы

/// </summary>

public void modelirovanie\_sustem()

{

t\_rab\_sistem += shag\_by\_time;

tekushee\_sost\_sustem = opred\_tec\_sost();

if (dlina\_ocheredi > 0)

{

t\_v\_sostoijaniax\_s\_ocher[dlina\_ocheredi - 1] += shag\_by\_time;

for (int i = 0; i < chislo\_kanalov\_in\_system; i++)

{

if (t\_oconch\_obsl\_vo\_vsex\_canalax[i] <= 0)

{

t\_oconch\_obsl\_vo\_vsex\_canalax[i] = t\_obslujivania\_zaijvki();

summ\_t\_obsl\_zaijvok += t\_oconch\_obsl\_vo\_vsex\_canalax[i];

dlina\_ocheredi--;

break;

}

}

}

if (prichla\_li\_sluchain\_zaijavka())

{

chislo\_post\_zaijavok++;

if (dlina\_ocheredi < Max\_dlina\_ocheredi)

{

chislo\_obslu\_zaijavok++;

if (tekushee\_sost\_sustem < Sostoijanija\_sustem.S7)

{

for (int i = 0; i < chislo\_kanalov\_in\_system; i++)

{

if (t\_oconch\_obsl\_vo\_vsex\_canalax[i] <= 0)

{

t\_oconch\_obsl\_vo\_vsex\_canalax[i] = t\_obslujivania\_zaijvki();

summ\_t\_obsl\_zaijvok += t\_oconch\_obsl\_vo\_vsex\_canalax[i];

break;

}

}

}

else

{ dlina\_ocheredi++; }

}

else

{ chislo\_otkaz\_zaijavok++; }

}

for (int i = 0; i < chislo\_kanalov\_in\_system; i++)

{

if (t\_oconch\_obsl\_vo\_vsex\_canalax[i] > 0)

{

t\_oconch\_obsl\_vo\_vsex\_canalax[i] -= shag\_by\_time;

}

}

}

}

}

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;

namespace CMO\_misiagin

{

public partial class Form1 : Form

{

CMO\_work work;

uint chislo\_progonov;

List<double> P, Q, A, K, Lo, To, Percents\_otkaz, Percents\_Prin;

uint n = 3; // Число каналов

uint m = 2; // MAX длина очереди

public Form1()

{

InitializeComponent();

work = new CMO\_work(n, m, (double)numericUpDown1.Value, (double)numericUpDown2.Value, (double)numericUpDown3.Value);

chislo\_progonov = 0;

P = new List<double>();

Q = new List<double>();

A = new List<double>();

K = new List<double>();

Lo = new List<double>();

To = new List<double>();

Percents\_otkaz = new List<double>();

Percents\_Prin = new List<double>();

}

private void Form1\_Load(object sender, EventArgs e)

{

button1.Enabled = false;

}

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

{

if (chislo\_progonov>1) button1.Enabled = true;

richTextBox1.Clear();

richTextBox1.Text += " Показатели эффективности: \n";

work.SetDefaults();

work.chislo\_kanalov\_in\_system = n;

work.Max\_dlina\_ocheredi = m;

work.Lamda = (double)numericUpDown1.Value;

work.Mu = (double)numericUpDown2.Value;

work.shag\_by\_time = (double)numericUpDown3.Value;

double Time = (double)numericUpDown4.Value;

while (work.t\_rab\_sistem < Time)

{

work.modelirovanie\_sustem();

}

richTextBox1.Text += "Поступило заявок: \t\t\t" + work.chislo\_post\_zaijavok.ToString() + "\n";

richTextBox1.Text += "Очередь: \t\t\t\t" + work.Dlina\_ocheredi.ToString() + "\n";

richTextBox1.Text += "Отклонено заявок: \t\t\t" + work.chislo\_otkaz\_zaijavok.ToString() + "\n";

// Вероятность отказа

double ver\_otcaza = work.t\_v\_sostoijaniax\_s\_ocher[work.Max\_dlina\_ocheredi - 1] / work.t\_rab\_sistem;

P.Add(ver\_otcaza);

richTextBox1.Text += "Вероятность отказа, Ротк: \t\t\t" + ver\_otcaza.ToString() + "\n";

// Относительная пропускная способность

double otnos\_prop\_spos = 1 - ver\_otcaza;

Q.Add(otnos\_prop\_spos);

richTextBox1.Text += "Относительная пропускная способность, Q: \t" + otnos\_prop\_spos.ToString() + "\n";

// Абсолютная пропускная способность

double absol\_prop\_spos = work.Lamda \* otnos\_prop\_spos;

A.Add(absol\_prop\_spos);

richTextBox1.Text += "Абсолютная пропускная способность, A: \t" + absol\_prop\_spos.ToString() +"\n";

// Длина очереди

double dlina\_ocheredi = 0;

for (int i = 0; i < work.Max\_dlina\_ocheredi; i++)

{

dlina\_ocheredi += (i + 1) \* (work.t\_v\_sostoijaniax\_s\_ocher[i] / work.t\_rab\_sistem);

}

Lo.Add(dlina\_ocheredi);

richTextBox1.Text += "Ср длина очереди, L\x03BF: \t\t\t" + dlina\_ocheredi.ToString() + "\n";

// Ср время пребывания заявки в очереди

double sr\_t\_preb\_v\_ocheredi = dlina\_ocheredi / work.Lamda;

To.Add(sr\_t\_preb\_v\_ocheredi);

richTextBox1.Text += "Ср время преб-ия з. в очереди, T\x03BF, c: \t" + sr\_t\_preb\_v\_ocheredi.ToString()+"\n";

// Ср число занятых каналов

double sr\_chislo\_zan\_can\_k = absol\_prop\_spos / work.Mu;

K.Add(sr\_chislo\_zan\_can\_k);

richTextBox1.Text += "Ср число занятых каналов, k: \t\t" + sr\_chislo\_zan\_can\_k.ToString() + "\n";

// % отказанных заявок

double curPercents\_otkaz = ((double)work.chislo\_otkaz\_zaijavok / (double)work.chislo\_post\_zaijavok) \* 100d;

Percents\_otkaz.Add(curPercents\_otkaz);

richTextBox1.Text += "% отказанных заявок: \t\t\t" + curPercents\_otkaz.ToString() + "\n";

// % обслуженных заявок

double absol\_prop\_sposcceptedPercents = ((double)work.chislo\_obslu\_zaijavok / (double)work.chislo\_post\_zaijavok) \* 100d;

Percents\_Prin.Add(absol\_prop\_sposcceptedPercents);

richTextBox1.Text += "% обслуженных заявок: \t\t\t" + absol\_prop\_sposcceptedPercents.ToString() + "\n";

// Число прогонов

chislo\_progonov++;

vivod\_chisla\_pasch.Text = chislo\_progonov.ToString();

/\* if (chislo\_progonov > 1)

{ RasStatistici\_Click(); }\*/

}

// 222222222222222 доверительные интервалы 222222222222222222222222222222222222222222222222

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

{

richTextBox1.Text += "\n Доверительные интервалы: \n";

double sredn\_Potk = 0, sredn\_Q = 0, sredn\_A = 0, sredn\_Lo = 0, sredn\_do = 0, sredn\_K = 0,

sredn\_Percents\_otkaz = 0, sredn\_Percents\_Prin = 0;

double sum\_Potk = 0, sum\_Q = 0, sum\_A = 0, sum\_Lo = 0, sum\_do = 0, sum\_K = 0,

sum\_Percents\_otkaz = 0, sum\_Percents\_Prin = 0;

double Potk\_ot = 0, Potk\_do = 0, Q\_ot = 0, Q\_do = 0, A\_ot = 0, A\_do = 0,

Lo\_ot = 0, Lo\_do = 0, To\_ot = 0, To\_do = 0, K\_ot = 0, K\_do = 0,

Percents\_otkaz\_ot = 0, Percents\_otkaz\_do = 0,

Percents\_Prin\_ot = 0, Percents\_Prin\_do = 0;

for (int i = 0; i < chislo\_progonov; i++)

{

sredn\_Potk += P[i];

sredn\_Q += Q[i];

sredn\_A += A[i];

sredn\_Lo += Lo[i];

sredn\_do += To[i];

sredn\_K += K[i];

sredn\_Percents\_otkaz += Percents\_otkaz[i];

sredn\_Percents\_Prin += Percents\_Prin[i];

}

// Математические ожидания величин

sredn\_Potk /= chislo\_progonov;

sredn\_Q /= chislo\_progonov;

sredn\_A /= chislo\_progonov;

sredn\_Lo /= chislo\_progonov;

sredn\_do /= chislo\_progonov;

sredn\_K /= chislo\_progonov;

sredn\_Percents\_otkaz /= chislo\_progonov;

sredn\_Percents\_Prin /= chislo\_progonov;

// Вычисление доверительных интервалов

for (int i = 0; i < chislo\_progonov; i++)

{

sum\_Potk += Math.Pow((P[i] - sredn\_Potk), 2);

sum\_Q += Math.Pow((Q[i] - sredn\_Q), 2);

sum\_A += Math.Pow((A[i] - sredn\_A), 2);

sum\_Lo += Math.Pow((sredn\_Lo - Lo[i]), 2);

sum\_do += Math.Pow((sredn\_do - To[i]), 2);

sum\_K += Math.Pow((K[i] - sredn\_K), 2);

sum\_Percents\_otkaz += Math.Pow((Percents\_otkaz[i] - sredn\_Percents\_otkaz), 2);

sum\_Percents\_Prin += Math.Pow((Percents\_Prin[i] - sredn\_Percents\_Prin), 2);

}

Potk\_ot = sredn\_Potk - (2.093d) \* (Math.Sqrt((sum\_Potk / (chislo\_progonov - 1)) / chislo\_progonov));

Potk\_do = sredn\_Potk + (2.093d) \* (Math.Sqrt((sum\_Potk / (chislo\_progonov - 1)) / chislo\_progonov));

Q\_ot = sredn\_Q - (2.093d) \* (Math.Sqrt((sum\_Q / (chislo\_progonov - 1)) / chislo\_progonov));

Q\_do = sredn\_Q + (2.093d) \* (Math.Sqrt((sum\_Q / (chislo\_progonov - 1)) / chislo\_progonov));

A\_ot = sredn\_A - (2.093d) \* (Math.Sqrt((sum\_A / (chislo\_progonov - 1)) / chislo\_progonov));

A\_do = sredn\_A + (2.093d) \* (Math.Sqrt((sum\_A / (chislo\_progonov - 1)) / chislo\_progonov));

Lo\_ot = sredn\_Lo - (2.093d) \* (Math.Sqrt((sum\_Lo / (chislo\_progonov - 1)) / chislo\_progonov));

Lo\_do = sredn\_Lo + (2.093d) \* (Math.Sqrt((sum\_Lo / (chislo\_progonov - 1)) / chislo\_progonov));

To\_ot = sredn\_do - (2.093d) \* (Math.Sqrt((sum\_do / (chislo\_progonov - 1)) / chislo\_progonov));

To\_do = sredn\_do + (2.093d) \* (Math.Sqrt((sum\_do / (chislo\_progonov - 1)) / chislo\_progonov));

K\_ot = sredn\_K - (2.093d) \* (Math.Sqrt((sum\_K / (chislo\_progonov - 1)) / chislo\_progonov));

K\_do = sredn\_K + (2.093d) \* (Math.Sqrt((sum\_K / (chislo\_progonov - 1)) / chislo\_progonov));

Percents\_otkaz\_ot = sredn\_Percents\_otkaz - (2.093d) \* (Math.Sqrt((sum\_Percents\_otkaz / (chislo\_progonov - 1)) / chislo\_progonov));

Percents\_otkaz\_do = sredn\_Percents\_otkaz + (2.093d) \* (Math.Sqrt((sum\_Percents\_otkaz / (chislo\_progonov - 1)) / chislo\_progonov));

Percents\_Prin\_ot = sredn\_Percents\_Prin - (2.093d) \* (Math.Sqrt((sum\_Percents\_Prin / (chislo\_progonov - 1)) / chislo\_progonov));

Percents\_Prin\_do = sredn\_Percents\_Prin + (2.093d) \* (Math.Sqrt((sum\_Percents\_Prin / (chislo\_progonov - 1)) / chislo\_progonov));

richTextBox1.Text += "Вероятность отказа, Ротк: \t\nот\t" + Potk\_ot.ToString() + "\nдо\t" + Potk\_do.ToString() + "\n";

richTextBox1.Text += "Относительная пропускная способность, Q: \nот\t" + Q\_ot.ToString() + "\nдо\t" + Q\_do.ToString() + "\n";

richTextBox1.Text += "Абсолютная пропускная способность, A: \nот\t" + A\_ot.ToString() + "\nдо\t" + A\_do.ToString() + "\n";

richTextBox1.Text += "Ср длина очереди, L\x03BF: \t\nот\t" + Lo\_ot.ToString() + "\nдо\t" + Lo\_do.ToString() + "\n";

richTextBox1.Text += "Ср время преб-ия з. в очереди, T\x03BF, c: \nот\t" + To\_ot.ToString() + "\nдо\t" + To\_do.ToString() + "\n";

richTextBox1.Text += "Ср число занятых каналов, k: \nот\t" + K\_ot.ToString() + "\nдо\t" + K\_do.ToString() + "\n";

richTextBox1.Text += "% отказанных заявок: \t\nот\t" + Percents\_otkaz\_ot.ToString() + "\nдо\t" + Percents\_otkaz\_do.ToString() + "\n";

richTextBox1.Text += "% обслуженных заявок: \t\nот\t" + Percents\_Prin\_ot.ToString() + "\nдо\t" + Percents\_Prin\_do.ToString() + "\n";

button1.Enabled = false;

}

}

}