ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ВОСТОЧНО-СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Алтаев А.А., Бундаев В.В.

Лабораторный практикум по дисциплине
«Компьютерное моделирование»

**МЕТОДИЧЕСКОЕ ПОСОБИЕ**

**для студентов специальностей 230105**

**«Программное обеспечение ВТ и АС» и 010503 «Математическое обеспечение и администрирование информационных систем»**

Издательство ВСГТУ

Улан-Удэ, 2007

УДК: 519.21(075.8)

Лабораторный практикум по дисциплине
«Компьютерное моделирование»/ Сост. Алтаев А.А., Бундаев В.В. – Улан-Удэ, Изд-во ВСГТУ, 2007. – 61 с.

Методическое пособие включает краткую теорию по моделированию систем и задания по лабораторным работам. Предназначено студентам специальностей 230105

«Программное обеспечение ВТ и АС» и 010503 «Математическое обеспечение и администрирование информационных систем», изучающих компьютерное моделирование

Рецензент: *Дарибазарон С.Б.*, к.ф.-м.н., доц. ВСГТУ

Печатается по решению редакционно-издательского совета ВСГТУ

© ВСГТУ, 2007 г.

© Алтаев А.А., Бундаев В.В.

# Оглавление

[Оглавление 3](#_Toc157211129)

[Лабораторная работа №1. Дискретно-детерминированные модели 4](#_Toc157211130)

[Лабораторная работа №2. Дискретно-стохастические модели 8](#_Toc157211131)

[Лабораторная работа №3. Непрерывно-стохастические модели 11](#_Toc157211132)

[Лабораторная работа №4. Система имитационного моделирования GPSS 26](#_Toc157211133)

[Лабораторная работа №5. Проведение однофакторных экспериментов 31](#_Toc157211134)

[Лабораторная работа №6. Проведение многофакторных экспериментов 41](#_Toc157211135)

[Литература 50](#_Toc157211136)

[Приложение 52](#_Toc157211137)

# Введение

# Дискретно-детерминированные модели

### Теория

Дискретно-детерминированные модели называются также конечными автоматами (англ. finite automat), или F-схемами. F-схемы характеризуются шестью элементами: конечным множеством *Х* входных сигналов (входным алфавитом); конечным множеством *Y* выходных сигналов (выходным алфавитом); конечным множеством Z внутренних состояний (внутренним алфавитом или алфавитом состояний); начальным состоянием автомата; функцией переходов *ϕ(z,х)*; функцией выходов *ψ(z,х).* Работа конечного автомата происходит по следующей схеме: в каждом *i-*м такте на вход автомата, находящегося в состоянии *zi*, подается некоторый сигнал *xi,* на который он реагирует переходом в новое состояние *zi+1* и выдачей некоторого выходного сигнала *yi*. Существуют разновидности конечных автоматов:

*автомат Мили* первого рода

*zi+1=ϕ[zi, xi],*

*yi=ψ[zi*, *хi],* *i=0,1,2,…;*

*автомат Мили* второго рода

*zi+1=ϕ[zi, xi],*

*yi+1=ψ[zi+1, хi], i=0,1,2,…;*

*автомат Мура*

*zi+1=ϕ[zi, xi],*

*yi=ψ[zi], i=0,1,2,….*

Для описания *F*-автоматов применяют табличный, графический и матричный способы.

В таблице на пересечении строки *xi* и столбца *zi* записываются функции переходов *ϕ[zi, xi]* и выходов *ψ[zi*, *хi].*

Примеры табличного способа задания *F*-автомата Мили с тремя состояниями, двумя входными и двумя выходными сигналами приведены в табл. 1.1, а для *F*-автомата Мура - в табл. 1.2.

Таблица 1.1.

|  |  |
| --- | --- |
| *xi* | *zk* |
| *z1* | *z2* | *z3* |
| **Переходы** |
| *x1* | *z2* | *z3* | *z3* |
| *x2* | *z3* | *z2* | *z1* |
| **Выходы** |
| *x1* | *y1* | *y1* | *y2* |
| *x2* | *y1* | *y2* | *y1* |

Таблица 1.2.

|  |  |
| --- | --- |
| *xi* | *yj* |
| *y1* | *y1* | *y2* |
| *z1* | *z2* | *z3* |
| *x1* | *z3* | *z3* | *z2* |
| *x2* | *z1* | *z1* | *z3* |

При графическом способе задания конечного автомата применяется направленный граф, вершины которого соответствуют состояниям автомата, дуги - переходам.

 На рис. 1.1, *а, б* приведены заданные ранее таблицами *F*-автоматы Мили и Мура соответственно.

*y1*

*x1*

*x1, y2*

*x1, y1*

*x2, y2*

*x2, y1*

*z1*

*z2*

*z3*

*x2, y1*

*x1, y1*

*а)*

*x2*

*x1*

*x2*

*x1*

*z1*

*z2*

*z3*

*б)*

*y1*

*x2*

*y2*

Рис. 1.1. Графы автоматов Мили *(а)* и Мура *(б)*

При матричном задании конечного автомата применяют квадратную матрицу *С=||cij||,* строки которой соответствуют исходным состояниям, а столбцы - состояниям перехода. Элемент *cij=xk/yS,* соответствует входному сигналу *xk,* вызывающему переход из состояния *zi* в состояние *zj* и выходному сигналу *уS,* выдаваемому при этом переходе. Для автомата Мили*,* рассмотренного выше, матрица соединений имеет вид

.

Для *F*-автомата Мура элемент *cij* равен множеству входных сигналов на переходе *(zi,zj),* а выход описывается вектором выходов. Для рассмотренного выше *F*-автомата Мура матрица соединений и вектор выходов имеют вид:

 

### Задание

Придумать F-автомат, представив его в табличном, графическом и матричном видах и составить программу, моделирующую его работу. Начальное состояние автомата и входное слово генерируются случайным образом. Выходные данные программы должны представлять собой таблицу с полями: *x, z\_old, z\_new, y*. Например, если моделируется работа рассмотренного выше автомата Мили (будем считать что он первого рода), начальное состояние которого равно z2 и на вход подается слово: x1 x2 x1 x2 x2, то таблица будет иметь вид

*x z\_old z\_new y*

*x1 z2 z3 y1*

*x2 z3 z1 y1*

*x1 z1 z2 y1*

*x2 z2 z2 y2*

*x2 z2 z2 y2*

Подсчитайте статистику.

Функции переходов и выходов оформить в виде типизированных констант. Для рассмотренного автомата Мили начало программы может иметь следующий вид:

uses crt;

const

n=2; {входной алфавит состоит из двух символов}

m=2; {выходной алфавит состоит из двух символов}

p=3; {количество состояний}

phi: array[1..n, 1..p] of integer =

((2,3,3), (3,2,1)); {функция переходов}

ksi: array[1..n, 1..p] of integer =

((1,1,2), (1,2,1)); {функция выходов}

### Варианты

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант | Тип автомата | n | m | p | Подсчитать статистику в абсолютном и процентном выражении по |
| 1 | Автомат Мили первого рода | 2 | 2 | 4 | состояниям |
| 2 | Автомат Мили второго рода | 3 | 2 | 4 | выходным сигналам |
| 3 | Автомат Мура | 4 | 2 | 4 | состояниям |
| 4 | Автомат Мили первого рода | 4 | 3 | 3 | выходным сигналам |
| 5 | Автомат Мили второго рода | 3 | 3 | 3 | состояниям |
| 6 | Автомат Мура | 2 | 3 | 3 | выходным сигналам |
| 7 | Автомат Мили первого рода | 3 | 4 | 2 | состояниям |
| 8 | Автомат Мили второго рода | 4 | 4 | 2 | выходным сигналам |
| 9 | Автомат Мура | 2 | 2 | 2 | состояниям |
| 10 | Автомат Мили первого рода | 3 | 5 | 3 | выходным сигналам |

# Дискретно-стохастические модели

### Теория

Дискретно-стохастические модели принято называть *Р*-автоматами (англ. probabilistic automat). В отличие от *F*-автоматов функции переходов *ϕ*  и выходов *ψ* в *Р*‑автоматах описываются матрицами вероятностей переходов и выходов. Каждый символ *xi* входного алфавита имеет свою матрицу вероятностей переходов *Ai*=(*ajk*) и матрицу вероятностей выходов *Bi*=(*bjℓ*), где *ajk* – вероятность перехода автомата из *j* состояния в *k* состояние при поступлении на вход *i*-го сигнала, *bjℓ* –вероятность выработки на выходе сигнала *ℓ*. Если обозначить через *N* и *M* - соответственно, количество символов во входном и выходном алфавитах, через *P* – число состояний, то для описания вероятностного автомата потребуется *N* матриц *A* и *B*. Сумма элементов одной строки этих матриц должна быть равна 1:  и . Помимо указанных матриц описание *Р*‑автомата должно быть дополнено вектором вероятностей начальных состояний C, элемент *ci* которого *—* вероятность того, что в начале работы *Р*-автомат будет находиться в состоянии *i.* Должно выполняться условие .

Рассмотрим пример задания *Р*‑автомата для следующих данных: *N*=2, *M*=2 и *P*=3. Примем, что , , . Для задания автомата в таком случае достаточно заполнить четыре матрицы и один вектор (табл. 2.1).

Таблица 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| *xi* | *Ai* | *Bi* | *C* |
| 1 |  |  |  |
| 2 |  |  |

Такой автомат назовем полным вероятностным *P*-автоматом. Существуют другие разновидности *P*-автоматов, у которых либо переход в новое состояние, либо выходной сигнал определяются детерминировано. Если выходной сигнал *Р*-автомата определяется детерминировано, то такой автомат называется *Y-*детерминированным вероятностным автоматом*.* Аналогично, *Z-*детерминированным вероятностным автоматом называется *Р*‑автомат, у которого выбор нового состояния является детерминированным. *Y‑*детерминированные вероятностные автоматы по аналогии с *F*-автоматами можно еще подразделить на автоматы Мили и Мура*.*

### Задание

В соответствии с заданием, разработать программу, моделирующую работу вероятностного автомата. Программа должна выводить информацию о состоянии автомата на каждом такте в виде таблицы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *x* | *z\_old* | *r\_1* | *z\_new* | *r\_2* | *y* |

где *x* и *y* – символы входного и выходного слова, *z\_old* и *z\_new* – соответственно, текущее и будущее состояния автомата, *r\_1* и  *r\_2* – случайные числа из диапазона [0; 1], используемые для определения перехода *z\_new* и выходного сигнала *y*. Обеспечить получение статистики.

Алгоритм определения следующего состояния *z\_new* рассмотрим на следующем примере. Пусть автомат находится в состоянии *’c’* и на вход поступает сигнал *1*. Для вычисления перехода выбирается матрица *A1* (*x=1*)и в ней - 3-я строка (*z=’c’*): (0,3; 0,3; 0,1). Если случайное число *r\_1* будет меньше или равно *a31*, то новым состоянием будет *’a’*. При выполнении условия *a31< r\_1≤* *a31+ a32* следующее состояние автомата - *’b’.* Если не выполняются условия для *’a’* и *’b’* , автомат на следующем такте перейдет в состояние *’c’*. Например, если *r\_1=0,643* , то *z\_new=’c’.* Аналогично определяются и выходные сигналы.

Рассмотрим более полный пример. Входное слово пусть состоит из пяти символов, полученных случайным образом: ‘11212’. Генератор случайных чисел (*random*) выдал следующую последовательность (значения округлены): (*0,23; 0,46; 0,08; 0,79; 0,49, 0,94; 0,73; 0,37; 0,48, 0,91; 0,57*). Первое случайное число (0,23) используется для определения начального состояния автомата. Поскольку выполняется условие *c1<0,23< c1+ c2* , значение *z\_old* принимает значение *‘b’*. Полученная таблица переходов и выхода представлена ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *x* | *z\_old* | *r\_1* | *z\_new* | *r\_2* | *y* |
| *1* | *b* | *0,46* | *a* | *0,08* | *m* |
| *1* | *a* | *0,79* | *c* | *0,49* | *n* |
| *2* | *c* | *0,94* | *c* | *0,73* | *m* |
| *1* | *c* | *0,37* | *b* | *0,48* | *n* |
| *2* | *b* | *0,91* | *c* | *0,57* | *n* |

В программе описание автомата представить в виде типизированных констант:

uses crt;

const

n=2; {входной алфавит состоит из двух символов}

m=2; {выходной алфавит состоит из двух символов}

p=3; {количество состояний}

a: array[1..n, 1..p, 1..p] of real =

(((0.5, 0, 0.5), (1, 0, 0), (0.3, 0.3, 0.1)),

 ((0, 0.5, 0.5), (0, 0.3,0.7), (0.9, 0, 0.1)));

b: array[1..n, 1..p, 1..m] of real =

(((0.4, 0.6), (1, 0), (0, 1)),

 ((0, 1), (0.5, 0.5), (0.7, 0.3)));

c: array[1..p] of real = (0.2, 0.2, 0.6);

var

x: 1..2; {входной символ}

y: ‘m’..’n’; {выходной символ}

z\_new, z\_old: ‘a’..’c’; {состояния}

### Варианты

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант | Тип вероятностного автомата | n | m | p | Подсчитать статистику в абсолютном и процентном выражении по |
| 1 | полный | 1 | 4 | 4 | состояниям |
| 2 | *Y-*детерминированный Мили | 2 | 4 | 4 | выходным сигналам |
| 3 | *Y-*детерминированный Мура | 3 | 2 | 2 | состояниям |
| 4 | *Z-*детерминированный | 1 | 5 | 3 | выходным сигналам |
| 5 | полный | 2 | 5 | 3 | состояниям |
| 6 | *Y-*детерминированный Мили | 3 | 5 | 3 | выходным сигналам |
| 7 | *Y-*детерминированный Мура | 1 | 3 | 5 | состояниям |
| 8 | *Z-*детерминированный | 2 | 3 | 5 | выходным сигналам |
| 9 | полный | 2 | 3 | 3 | состояниям |
| 10 | *Z-*детерминированный | 1 | 4 | 4 | выходным сигналам |

#  Непрерывно-стохастические модели

### Теория

Одной из разновидностей непрерывно-стохастических моделей являются *системы массового обслуживания (СМО)*, называемые ***Q-схемами*** (queue - очередь). В качестве примера СМО можно привести обслуживание читателей в библиотеке, клиентов в парикмахерской и т.п. Рассмотрим основные понятия Q-схем.

Простейшая СМО (рис.3.1), называемая прибором, состоит из накопителя заявок H*,* в котором может одновременно находиться  заявок, где LH *—* емкость накопителя, и канала обслуживания заявок K*.* В СМО наблюдаются следующие потоки: w - поток поступающих на обслуживание заявок, u *—* поток управляющих воздействий, y - поток обслуженных заявок.

K

y

H

w

u

Рис. 3.1. Прибор обслуживания заявок

*Потоком событий* называется последовательность событий, происходящих одно за другим в какие-то случайные моменты времени. Различают потоки однородных и неоднородных собы­тий. Поток событий называется *однородным,* если он характеризу­ется только моментами поступления этих событий (вызываю­щими моментами) и задается последовательностью *{ti}={0 ≤ t1 ≤ t2 … ≤ ti≤ …},* где *ti* — момент наступления *i*-го собы­тия. Однородный поток событий также может быть задан в виде последовательности про­межутков времени между *i*-м и *(i—1)-*м событиями *{τi},* где *τi = ti – ti-1, i ≥1, t0 = 0,* т.е. *τ1 = t1*. Таким образом, для отсчета времени наступления событий в СМО применяются два множества времен (рис. 3.2): множество абсолютных времен *{t1, t2, …}* и множество относительных времен *{τ1, τ2, …}.*

*Потоком неоднородных событий* называется последовательность *{tt, ft},* где *tt* — вызывающие моменты; *ft* — набор признаков события. Например, примени­тельно к процессу обслуживания для не­однородного потока заявок могут быть заданы принадлежность к тому или иному источнику заявок, наличие приоритета, возможность обслуживания тем или иным типом канала и т. п.

Начало моделирования

Наступление первого события

Наступление

второго

события

t1

t2

τ1

τ2

Рис. 3.2. Два принципа отсчета времени в СМО

Рассмотрим поток, в котором события разделены интервалами времени *τ1, τ2, …,* которые вообще являются случайными величинами. Пусть интервалы *τ1, τ2, ...* независимы между собой. Тогда поток событий называется *потоком с ограниченным последействием.*

Рассмотрим малый интервал времени *[t, t+Δt].* Вероятность наступления на этом интервале одного события обозначим через *Р1(t, Δt),* вероятность наступления двух и более событий через *Р>1(t, Δt),* вероятность ненаступления события через *Р0(t, Δt).* В таком случае получаем

*Р0(t, Δt) + Р1(t, Δt) + Р>1(t, Δt) = 1.*

Поток событий называется *ординарным*, если *Р>1(t, Δt)* пренебрежительно мала по сравнению с *Р1(t, Δt),* т.е.

*Р1(t, Δt) >> Р>1(t, Δt)* и *Р0 (t, Δt) + Р1 (t, Δt) ≈1*

Введем такое понятие как *интенсивность* ординарного потока событий . *Стационарным потоком* событий называется поток, для которого интенсивность не зависит от времени *t* и представляет собой постоянное значение, равное среднему числу событий, наступающих в единицу времени .

Процесс функционирования СМО можно представить как процесс изменения состояний его элементов во времени. Переход в новое состояние для СМО означает изменение количества заявок, которые в нем находятся (в канале K и в накопи­теле H)*.* Возможные состояния СМО: по накопителю - накопитель пуст, накопитель непуст, накопи­тель полностью заполнен; по каналу - канал свободен, канал занят.

В сложных Q-схемах применяют композиции из множества элементарных приборов обслужива­ния. Если каналы различных приборов обслуживания соединены параллельно, то имеет место *многоканальное* обслуживание (многоканальная Q-схема), а если приборы и их параллельные композиции соединены последовате­льно, то имеет место *многофазное* обслуживание (многофазная Q-схема).

Связи между элементами Q-схемы изображают в виде стрелок (линий потока, отражающих направление движения заявок). Раз­личают разомкнутые и замкнутые Q-схемы. В разомкнутой Q-схеме выходной поток обслуженных заявок не может снова поступить на какой-либо элемент, т. е. обратная связь отсутствует, а в замкнутых Q-схемах имеются обратные связи, по которым заявки двигаются в направлении, обратном движению вход-выход.

Для задания Q-схемы также необходимо описать алгоритмы ее функционирования, которые определяют набор правил поведения заявок в системе в различных неоднозначных ситуациях. В зависи­мости от места возникновения таких ситуаций различают алгорит­мы (дисциплины) ожидания заявок в накопителе и обслуживания заявок каналомкаждого элементарного обслуживающего прибо­ра Q-схемы.

Заявки могут иметь различные приоритеты. Приоритеты могут быть *статические* и *динамические*. Статические приоритеты назначаются заявкам заранее и не зависят от состояний Q-схемы. Динамические приоритеты могут изменяться во время модели­рования в зависимости от возникающих ситуаций. Исходя из правил выбора заявок из накопителя Hна обслуживание каналом K*,* можно выделить относительные и абсолютные приоритеты. *От­носительный приоритет* означает, что заявка с более высоким при­оритетом, поступившая в накопитель ожидает окончания об­служивания предшествующей заявки каналом и только после этого занимает канал. *Абсолютный приоритет* означает, что заявка с более высоким приоритетом, минуя накопитель*,* пре­рывает обслуживание каналом заявки с более низким приорите­том и сама занимает канал (при этом вытесненная изканалазаявка может либо покинуть систему, либо может быть снова записана на какое-то место в накопителе). В таких СМО канал может находиться в трех состояниях: свободен, занят, прерывания.

При рассмотрении алгоритмов функционирования приборов обслуживания (каналов Kи накопителей H) необходимо также задать набор правил, по которым заявки покидают H и K:дляH *—* либо правила переполнения, по которым заявки в зависимости от заполнения Hпокидают систему, либо правила ухода, связанные с истечением времени ожидания заявки в H, для K *—* правила выбора маршрутов или направлений ухода. Кроме того, для заявок необходимо задать правила, по которым они остаются в канале K или не допускаются до обслуживания каналом K*,* т. е. правила блокировок канала. При этом различают блокировки K по выходу и по входу. Такие блокировки отражают наличие управляющих связей в Q-схеме, регулирующих поток заявок в зависимости от состояний Q-схемы.

Возможны три варианта значений емкости *LН* накопителя:

1. *LН = 0* *–* прибор накопитель не имеет;
2. *LН = const –* накопитель ограниченной длины (возможны потери заявок из-за переполнения накопителя);
3. *LН = ∞* *–* накопитель неограниченной длины.

y1

y2

x

u

H

В1

В2

Рис. 3.3. Прибор обслуживания заявок при наличии очереди ограниченной длины.

К

Для прибора с накопителем ограниченной длины (рис. 3.3.) следует предусматривать логические вентили В1 и В2. В таком случае, если текущая длина накопителя *ℓ* меньше емкости *LH* вентиль В1 открыт и заявки поступают в накопитель, вентиль В2 при этом закрыт. При *ℓ= LH* состояние вентилей меняется на противоположное и заявки будут уходить из прибора необслуженными (поток *y2*). Следовательно, накопитель “следит” за состоянием вентилей с помощью потока управляющих воздействий *u*.

Эффективность СМО можно характеризовать большим числом различных показателей [17]. К числу наиболее часто применяемых показателей относятся следующие показатели (за *λ* и А здесь приняты интенсивности входного и выходного потоков заявок):

* вероятность потери заявки *p*отк, для систем с потерями она равна сумме вероятности занятости всех обслуживающих приборов и вероятности переполнения очередей;
* относительная пропускная способность системы *q* = 1 – *p*отк – доля обслуженных заявок;
* абсолютная пропускная способность системы *A = λ∙q* – интенсивность заявок, обслуженных системой;
* вероятность того, что обслуживанием занято *k* приборов – *pk*, частным случаем этого показателя являются *pn* – вероятность занятости всех приборов и *p*0 – вероятность того, что все приборы свободны;
* среднее число занятых приборов *n*з =  = ;
* среднее число свободных приборов *n*0 = ;
* коэффициент простоя приборов *K*п = *n*0/*n*;
* коэффициент занятости приборов *K*з = *n*з/*n*;
* средняя длина очереди  = , ( = );
* среднее время ожидания в очереди  = /;
* среднее число заявок, находящихся в системе ;
* средняя длительность обслуживания заявки ;
* среднее время пребывания заявки в системе =  + ;
* вероятность того, что число заявок в очереди больше некоторого числа *m* *Pr>m*= .

Для контроля получаемых результатов удобно пользоваться функциональными связями параметров системы:

*n*0 + *n*з = *n*,

 = ,

= ,

 = o+п,

где o = A – интенсивность обслуженных заявок,

 п – интенсивность потерянных заявок.

В СМО события подразделяются на основные и на вспомогательные. К основным событиям относятся приход заявки в модель (выход ее из источника И) и конец обслуживания заявки в канале (уход ее из канала К и удаление из модели). Время наступления основных событий вычисляется случайным образом. Вспомогательные события являются следствием наступления основных событий. Для расчета времени наступления основных событий используют среднее время наступления события  и равномерный (рис 3.4), экспоненциальный *I* или нормальный *II* (рис 3.5) законы распределения *f(t)* случайной величины. В моделировании времена принято выражать в целых числах. Таким образом, если применять равномерное распределение для расчета времени, то время наступления будущего события *τ* определится по формуле ** и будет принимать целые значения из диапазона [-B,+B]. Здесь *B* – полуразмах отклонения от среднего. Если рассматривать только приходы заявок, то такой поток событий относится к потоку однородных и одинарных событий и время будущего события (прихода следующей заявки) *ti+1* будет рассчитано через время уже наступившего события *ti* (времени прихода только что поступившей в модель заявки):.







*f(t)*

*f(t)*

*B*

*B*

*II*

*I*









Рис. 3.5

Рис. 3.4

Рассмотрим таблицу соответствия основных и вспомогательных событий для простейшей СМО с одним каналом и с одной очередью (рис. 3.11,а). Средний интервал между приходами заявок в модель , среднее время обслуживания в канале , полуразмахи отклонений *Bпр* и *Bобсл* известны. Хотя момент завершения моделирования (*t=T*) не является основным событием, его также включим в таблицу.

Таблица 3.1

| **Основное событие** | **Вспомогательные события и сопутствующие вычисления** |
| --- | --- |
| Приход заявки | 1. Канал свободен?

Да* 1. Канал перевести в состояние «занято»;
	2. Рассчитать *τобсл* для заявки, занявшей канал

Нет1. Заявка занимает очередь (длина очереди увеличивается на 1);
2. Уменьшить время *τобсл* = *τобсл* - *Δt* для заявки, обслуживающейся в канале
3. Вычислить время прихода в модель следующей заявки *τпр*
 |
| Конец обслуживания заявки в канале | 1. Очередь пуста?

ДаКанал перевести в состояние «свободно»Нет1. Заявка занимает канал (длина очереди уменьшается на 1);
2. Рассчитать *τобсл* для заявки, занявшей канал
3. Уменьшить время *τпр* = *τпр* - *Δt* для заявки, ожидающей прибытия в модель
 |
| Завершение моделирования | Вывод накопленной статистики |

В СМО с двумя каналами и с двумя очередями (рис. 3.11,b) будут наблюдаться три основных события: приход заявки в модель и завершение обслуживания в двух каналах. Таблица соответствия событий для данной системы будет иметь вид

Таблица 3.2

| **Основное событие** | **Вспомогательные события и сопутствующие вычисления** |
| --- | --- |
| Приход заявки в модель | 1. Канал *К1* свободен?

Да* 1. Канал *К1* перевести в состояние «занято»;
	2. Рассчитать *τобсл1* для заявки, занявшей канал

Нет1. Заявка занимает очередь *H1*  (длина очереди увеличивается на 1);
2. Уменьшить время *τобсл1 = τобсл1 - Δt* для заявки, обслуживающейся в канале *К1*
3. Уменьшить время *τобсл2 = τобсл2 - Δt* для заявки, обслуживающейся в канале *К2*
4. Вычислить время прихода в модель следующей заявки *τпр*
 |
| Конец обслуживания заявки в канале *К1* | 1. Очередь *H1* пуста?

ДаКанал *К1* перевести в состояние «свободно»Нет1. Заявка покидает очередь *H1* и занимает канал *К1* (длина очереди *H1* уменьшается на 1);
2. Рассчитать *τобсл1* для заявки, занявшей канал;
3. Уменьшить время *τпр* = *τпр* - *Δt* для заявки, ожидающей прибытия в модель;
4. Канал *К2* свободен?

Да* 1. Канал *К2* перевести в состояние «занято»;
	2. Рассчитать *τобсл2* для заявки, занявшей канал

Нет1. Заявка занимает очередь *H2*  (длина очереди увеличивается на 1);
2. Уменьшить время *τобсл2 = τобсл2 - Δt* для заявки, обслуживающейся в канале *К2*
 |
| Конец обслуживания заявки в канале *К2* | 1. Очередь *H2*  пуста?

ДаКанал *К2* перевести в состояние «свободно»Нет* 1. Заявка покидает очередь *H2* и занимает канал *К2* (длина очереди *H2* уменьшается на 1);
	2. Рассчитать *τобсл2* для заявки, занявшей канал *К2*
1. Уменьшить время *τпр* = *τпр* - *Δt* для заявки, ожидающей прибытия в модель
2. Уменьшить время *τобсл1 = τобсл1 - Δt* для заявки, обслуживающейся в канале *К1*
 |
| Завершение моделирования | Вывод накопленной статистики |

Для трехканальной СМО, приведенной на рис. 3.11,с, предлагаем таблицу основных и вспомогательных событий разработать самостоятельно.

При моделировании Q-схем применяются два алгоритма изменения модельного времени *t*:

* принцип Δ*t* - с постоянным шагом приращения *t* (как правило, *Δ t=1*);
* принцип δz - с переменным шагом.

Алгоритм работы модели по принципу Δ*t* можно представить в виде схемы (рис. 3.6.)

Нет

t=t+Δt

Должно совершиться основное событие?

Да

Выполнить основное событие и сопутствующие вспомогательные события

Рис. 3.6.

Алгоритм модели согласно δz представлен на рис. 3.7.

Должны совершиться в этот момент времени другие основные события?

Да

Нет

Рис. 3.7.

Выполнить событие

Увеличить t до момента наступления события

Определить наиближайшее основное событие

Рассмотрим пример моделирования одноканальной СМО (рис. 3.11,а) по принципу Δ*t*. Приняты следующие данные: =3; *Bпр*=2; =4; *Bобсл*=1, Δ*t*=1 единице времени (ед. вр.). Следовательно, времена прихода заявок *τпр* будут выбираться из множества {1, 2, 3, 4, 5}, времена обслуживания *τобсл* – из множества {3, 4, 5}, Пусть для *τпр* была сгенерирована следующая последовательность случайных чисел: 2, 4, 2, 3, 5, для *τобсл* – последовательность: 3, 5, 4, 4.

Время завершения моделирования примем *T*=15 ед. вр. В моменты времени, когда канал переводится в состояние «свободно», *τобсл* задается в виде фиктивного числа*.* Использование фиктивного времени объясняется тем, что поскольку в канале нет заявок, то наиближайшим событием никак не может быть конец завершения обслуживания. Оно должно быть подобрано таким образом, чтобы было заведомо больше *τпр* и *T-t.*

*τпр*

2

1

4

3

2

1

2

1

3

2

1

5

4

3

2

1

*l*

0

0

0

0

0

0

0

0

1

1

1

1

1

1

1

0

*τобсл* 16

15

3

2

1

11

5

4

3

2

1

4

3

2

1

4

*t*

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

*T-t*

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

1-я заявка (обслужена)

2-я заявка (обслужена)

3-я заявка (обслужена)

5-я заявка (в модель не успела прийти)

4-я заявка (осталась в канале - не дообслужена)

Рис. 3.8

В рассмотренном примере фиктивное значение вычисляется по формуле

*τобсл*=*T-t+1*, (3.1)

но с таким же успехом могла быть использована, например, формула *τобсл*=*T+1*.

В начале моделирования, когда модельное время *t*=0, в модели нет заявок. Следовательно, длина очереди *l*=0, канал пуст: *τобсл*= *T-t+1=16* ед. вр. Время прихода первой заявки принято равным 2 ед. вр.

Блок-схема модели простейшей СМО (рис. 3.11,а.), работающей по принципу δz, представлена на рис. 3.9. При *t=0* длина очереди *l* равна *0*, канал находится в состоянии «свободно» - *f=false*.

Нет

Начало

Ввод , *Bпр*, , *Bобсл*, T

*t=0; τобсл=T+1; f=false; l=0;*

*τпр=*- *Bпр+random(2\* Bпр+1)*

T=t

Да

Да

*τобсл* ≤ *τпр*

нет

*Δt=τобсл*

Нет

*f*

Да

 *f=true;*

*τобсл=*- *Bпр+random(2\* Bобсл+1)*

*l=l+1*

Вывод

статистики

Нет

*l=0*

Да

*f=false;*

*τобсл=T-t+1*

Конец

*l=l-1;*

*τобсл=*- *Bпр+random(2\* Bобсл+1)*

*Δt= τпр*

Нет

*Δt* = *τпр*

Да

*t=t+Δt*

Нет

*T-t<Δt*

Да

*τобсл= τобсл -Δt*

*τпр= τпр -Δt*

*Δt= T-t*

рис. 3.9.

*τприх=*- *Bприх+random(2\* Bобсл+1)*

Приращение времени определяется по формуле *Δt=min{τпр*, *τобсл*, *T-t}*, следующее значение модельного времени по формуле *ti+1=ti+Δt*. на место кружка вставить формулу

 *τпр=*- *Bпр+random(2\* Bпр+1)*

Условие *Δt*= *τпр* учитывает вероятность одновременного наступления двух основных событий: прихода заявки и ухода заявки из канала. В этом случае логично вначале освободить канал, а затем обработать приход заявки.

Рассмотрим пример моделирования одноканальной СМО (рис. 3.11,а). Приняты следующие данные: =5; *Bпр*=3; =4; *Bобсл*=1; *T*=1000. Одной звездочкой обозначены числа, выбранные случайным образом из множества {3, 4, 5, 6, 7}, двумя звездочками - из множества {3, 4, 5}, тремя звездочками – вычисленные по формуле (3.1).

| *t* | *τпр* | *l*  | *τобсл* | *f* | *T-t* |
| --- | --- | --- | --- | --- | --- |
| 0 | 4\* | 0 | 1001\*\*\* | False | 1000 |
| 4 | 7\* | 0 | 5\*\* | True | 996 |
| 9 | 2 | 0 | 992\*\*\* | False | 991 |
| 11 | 3\* | 0 | 4\*\* | True | 989 |
| 14 | 7\* | 1 | 1 | True | 986 |
| 15 | 6 | 0 | 3\*\* | True | 985 |
| … |

На рис. 3.10 приведена иллюстрация потока событий, совершающихся в модели.

В начале моделирования (*t=*0) вычисляется время прихода первой заявки в модель. Оно выбирается из интервала времени [3,7]: *τпр1*= 4. Так как, канал свободен, то время обработки приравнивается *T+1*= 1001. Множество времен {4, 1001, 1000} можно назвать списком будущих событий. Список указывает, какое событие является наиближайшим. Им является приход заявки. Модельное время увеличивается до значения 4 и обрабатывается приход первой заявки. Пришедшая заявка занимает канал и для нее вычисляется время обслуживания в канале *τобсл1* (*τобсл1*= 5). Это время берется из интервала [3,5]. Рассчитывается время прихода второй заявки *τпр2*= 7. Время, оставшееся до конца моделирования равно *T-t*= 996. Список будущих событий теперь имеет вид {7, 5, 996}. Таким образом, следующим значением модельного времени будет *t+Δt*= 4+5= 9. На этом этапе обрабатывается конец обслуживания заявки и, так как, вторая заявка еще не пришла, канал переводится в состояние «свободно». Дальнейший процесс понятен из таблицы.

При приближении конца моделирования в списке будущих событий наименьшим значением станет величина *T-t*, которая и будет принята за последнее приращение.

Статистические данные вычисляются для канала, для очереди и для всей модели. Для модели вычисляется среднее время пребывания заявки в модели (без учета заявок, оставшихся в модели в момент завершения моделирования). Для очереди вычисляются максимальная длина очереди, среднее время пребывания заявки в очереди с учетом и без учета нулевых входов, средняя длина очереди с учетом и без учета нулевых

входов. Нулевым входом в очередь считается случай, когда заявка проходит через очередь без задержки в ней (в очередь нет заявок, канал свободен). Для канала вычисляется коэффициент его использования, являющийся отношением суммарного времени пребывания канала в состоянии «занято» к времени моделирования *T.* И для канала и для очереди подсчитывается количество входов. Если очередь имеет ограниченную длину, то возможно ее переполнение. Заявки, не попадающие в очередь из-за переполнения, покидают модель необслуженными. Такая модель называется моделью с потерями. В этом случае подсчитывается количество потерянных заявок. Потери возможны также, если часть приходящих заявок имеет абсолютный приоритет и дисциплина обслуживания канала не разрешает дообслуживание прерванных заявок. В СМО с абсолютными приоритетами канал может находиться в трех состояниях: «свободно», «занято» и «прерывания».

свободен

свободен

*t*

15

9

*τобсл1*

*τпр4*

14

*τпр3*

11

*τпр1*

4

0

*Приход заявок*

*τпр2*

*τобсл2*

*τобсл3*

*Состояние канала*

занят

занят

*Уход заявок*

Рис. 3.10.

### Задание

Составить программу, моделирующую СМО в соответствии с вариантами. Считать, что время прихода заявки и время обслуживания равномерно распределены на интервале. Выходные данные оформить в виде таблицы, иллюстрирующей состояние модели при каждом изменении модельного времени. Таблица должна включать поля: *t, τпр , l, τобсл , f, T-t.* Если система с потерями, то в таблицу включить поле с данными по количеству потерянных заявок. Если СМО имеет два и более каналов, то таблица должна иметь соответствующие этим каналам поля *τобсл1 , τобсл2* , … . Для СМО с двумя и более источниками предусмотреть поля *τобсл1 , τобсл2 ,* … .

Если в СМО предусмотрены разноприоритетные заявки, то 15% заявок присвоить повышенный, а остальным обычный приоритет. Если заявки имеют относительные приоритеты, то в таблице предусмотреть столбцы  *l* (длина очереди заявок с обычными приоритетами) и *lприор* (длина очереди заявок с повышенными приоритетами).

Обеспечить сбор статистики.

### Варианты

| № варианта | Схема СМО (рис. 3.10) | Длина очереди Lн:- ограниченная,+ неограниченная | приращение времени:+ с переменным шагом- с постоянным шагом | наличиеприоритетов | Определяемые величины |
| --- | --- | --- | --- | --- | --- |
| относительного | абсолютного |
| 1 | а | **-** | **+** | **-** | **-** | max длина очереди |
| 2 | а | **-** | **+** | **-** | **-** | средняя длина очереди |
| 3 | а | **-** | **+** | **-** | **-** | среднее время пребывания заявки в очереди |
| 4 | а | **-** | **+** | **-** | **-** | среднее время пребывания заявки в модели |
| 5 | а | **-** | **+** | **-** | **-** | число потерянных заявок |
| 6 | a | **+** | **-** | **-** | **-** | max длина очереди |
| 7 | a | **+** | **-** | **-** | **-** | средняя длина очереди |
| 8 | a | **+** | **-** | **-** | **-** | среднее время пребывания заявки в очереди |
| 9 | a | **+** | **-** | **-** | **-** | среднее время пребывания заявки в модели |
| 10 | a | **+** | **-** | **-** | **-** | коэффициент занятости канала |
| 11 | a | **+** | **+** | **+** | **-** | max длина очереди |
| 12 | a | **+** | **+** | **+** | **-** | средняя длина очереди |
| 13 | a | **+** | **+** | **+** | **-** | среднее время пребывания заявки в очереди |
| 14 | a | **+** | **+** | **+** | **-** | среднее время пребывания заявки в модели |
| 15 | a | **+** | **+** | **+** | **-** | коэффициент занятости канала |
| 16 | a | **+** | **+** | **-** | **-** | max длина очереди |
| 17 | a | **+** | **+** | **-** | **-** | средняя длина очереди |
| 18 | a | **+** | **+** | **-** | **-** | среднее время пребывания заявки в очереди |
| 19 | a | **+** | **+** | **-** | **-** | среднее время пребывания заявки в модели |
| 20 | a | **+** | **+** | **-** | **-** | коэффициент занятости канала |
| 21 | a | **+** | **-** | **-** | **+** | max длина очереди |
| 22 | a | **+** | **-** | **-** | **+** | средняя длина очереди |
| 23 | a | **+** | **-** | **-** | **+** | среднее время пребывания заявки в очереди |
| 24 | a | **+** | **-** | **-** | **+** | среднее время пребывания заявки в модели |
| 25 | a | **+** | **-** | **-** | **+** | коэффициент занятости канала |
| 26 | b | **+** | **+** | **-** | **-** | max длина очереди |
| 27 | b | **+** | **+** | **-** | **-** | средняя длина очереди |
| 28 | b | **+** | **+** | **-** | **-** | среднее время пребывания заявки в очереди |
| 29 | c | **+** | **+** | **-** | **-** | среднее время пребывания заявки в модели |
| 30 | c | **+** | **+** | **-** | **-** | коэффициент занятости канала |

|  |
| --- |
| 1.

KHИ |
| K1H1ИK2H2 |
| K1H1И1K3H3K2H2И2 |

Рис. 3.11.

# Система имитационного моделирования GPSS

### Теория

Существует множество программных комплексов имитационного моделирования (AUTOMOD, ARENA, EXTEND, MAST, Deneb, PROQUEST, Taylor II, Simplex3 и т.д.). Одним из наиболее известных комплексов является система имитационного моделирования общего применения GPSS (General Purpose Simulation System).

GPSS предназначена для описания и исследования моделей систем массового обслуживания (СМО). Модель СМО состоит из элементов, называемых объектами аппаратной категории (устройства, памяти и логические ключи). Динамическими объектами в СМО являются транзакты (сообщения, заявки). Функционирование СМО представляется как процесс прохождения транзактов через объекты аппаратной и ряда других категорий.

В процессе выполнения программы автоматически вычисляется статистическая информация в виде стандартных арифметических атрибутов (СЧА), которые выводятся в файл отчета по окончании моделирования.

### Задание

Построить модель, используя среду имитационного моделирования GPSS.

Очередь ограниченной длины представить в виде многоканального устройства (памяти), логический вентиль, проверяющий очередь на переполнение, реализовать либо с помощью блока GATE, либо блока TEST. Для сбора статистики по очереди неограниченной длины использовать блок QUEUE

В моделях с разноприоритетными заявками обеспечить разбиение входящего потока заявок на два потока (безприоритетных и с приоритетами) с помощью блока TRANSFER. Предусмотреть в такой модели три очереди: общую для обоих потоков и по одной для каждого потока.

Собираемые в ходе моделирования статистические данные вывести в файл отчета, поместив их в сохраняемые ячейки. Прогнать модель при разных наборах исходных данных, объяснить полученные результаты.

### Варианты

* 1. ([видео](https://youtu.be/2uzEQkYwdVc))

Создать программу, моделирующую процесс прохождения заявок через прибор. Одна треть всех поступающих заявок имеет повышенный приоритет. Поступление заявок подчиняется равномерному закону с интервалом X плюс/минус 2 мин, обработка – экспоненциальному со средним временем X. Если у прибора нет возможности принять заявку, она становится в очередь.

Проанализируйте характеристики СМО, попытайтесь изменить их, меняя параметры системы. Оцените среднее время пребывания безприоритетных заявок в модели, затабулировав соответствующий СЧА. Аналогичную оценку сделайте и для высокоприоритетных заявок. Начертите блок-диаграмму модели.

Рассматривается система с потерями. Число мест в очереди ограничено Z. В случае, если все Z мест в очереди заняты, заявка теряется. Поступление заявок подчинено экспоненциальному закону, обработка - равномерному закону.

Меняя Z, смоделируйте систему без потерь. Начертите блок-диаграмму.

Рассматривается система с потерями. Число мест в очереди ограничено Z. В случае, если все Z мест в очереди заняты, заявка теряется. Одна треть всех поступающих заявок имеет повышенный приоритет. Поступление заявок подчинено равномерному закону, обработка - экспоненциальному закону.

Оцените среднее время пребывания безприоритетных заявок в модели, затабулировав соответствующий СЧА. Аналогичную оценку сделайте и для высокоприоритетных заявок. Начертите блок-диаграмму.

* 1. ([видео](https://youtu.be/Rk69ZFBSfnk))

Моделирование многолинейной системы с потерями. Система состоит из 2-х приборов (1 и 2). Каждому прибору соответствует накопитель (sys1, sys2) ёмкостью 2 и 3 соответственно. Заявка пытается захватить прибор 1, а если он занят, направляется в накопитель sys1. Если накопитель заполнен, заявка пытается захватить прибор 2, или направляется в накопитель sys2. Если заявка не может войти ни в один из указанных блоков, она уничтожается. Поступление и обработка заявок подчинены экспоненциальному закону.

Дополните программу, введя прибор 3 и накопитель sys3.

Проанализируйте параметры. Попытайтесь обеспечить одинаковую загрузку 3-х приборов. Изменяя параметры, создайте систему без потерь, начертите блок-диаграмму.

Моделирование системы с отказами. Система состоит из одного прибора, интенсивность потока отказов = 0,1. Распределение времени восстановления - экспоненциальное. Среднее время восстановления равно 20. Оцените среднее время пребывания прибора в состоянии отказа. Введите дублирующий прибор с такими же параметрами, который включается при отказе основного, начертите блок-диаграмму.

Рассматривается модель из 2-х приборов с отказами. Одинаковые параметры приборов: среднее время обработки заявок = 20; разные параметры: интенсивность отказов для 1-го прибора равна 0.0125, для 2-го прибора - 0.01.

Описание работы модели.

Заявки поступают в среднем через 10 ед.вр. Приборы последовательно проверяются на занятость и заявка занимает первый свободный прибор. Если все приборы заняты, заявка теряется. При отказе прибора заявка также теряется. Оба прибора могут при отказе ремонтироваться одновременно, среднее время восстановления равно 25 ед.вр.

Оцените количество потерянных заявок. Измените программу, введя 3-й прибор, проанализируйте поведение системы (повысилась ли надежность?). Можно ли без ущерба для надежности заменить 2-х ремонтников одним? Начертите блок-диаграмму.

Дана система, состоящая из двух приборов (основного и дублирующего). Длина очереди к основному прибору не превышает 3, заявки, не попавшие в очередь, становятся в другую очередь с ограниченным числом мест равным 3 (к дублирующему прибору). Если и здесь мест нет, заявка теряется. Среднее время задержки у обоих приборов 20 ед.вр. Заявки поступают в среднем через 8 ед.вр.

Подсчитайте количество потерянных заявок за 1000 ед.вр. Начертите блок-диаграмму.

Дана система, состоящая из двух параллельно работающих приборов Среднее время задержки у приборов равно 20 ед.вр. Длина общей очереди к ним не превышает 3, заявки, не попавшие в очередь, теряются. Одна треть всех поступающих заявок имеет повышенный приоритет. Заявки поступают в среднем через 8 ед.вр.

Подсчитайте количество потерь за 1000 ед.вр. отдельно для высокоприоритетных, и отдельно для безприоритетных заявок. Начертите блок-диаграмму.

Дана система из 2-х последовательно соединенных приборов. Заявки поступают в среднем через 12 ед.вр. сначала на 1-й прибор, задерживаются в нем на 14 ед.вр., затем на 2-й прибор, где обрабатываются в течении 16 ед.вр. Заявки, обрабатывавшиеся в первом приборе за время более 16 ед.вр., получают более высокий приоритет перед другими заявками при прохождении через вторую очередь. Поступление и обработка заявок подчинены экспоненциальному закону.

Необходимо написать программу, моделирующую эту систему. Выведите статистику по очередям в виде таблиц (для этого определить места возможного образования очередей и описать их). Подсчитайте количество высокоприоритетных заявок, обслуженных за 1000 ед.вр. Начертите блок-диаграмму.

Дана система из 2-х последовательно соединенных приборов. Заявки поступают в среднем через 12 ед.вр. сначала на 1-й прибор, задерживаются в нем на 14 ед.вр., затем на 2-й прибор, где обрабатываются в течении 16 ед.вр. Поступление и обработка заявок подчинены экспоненциальному закону. 2-й прибор может выходить из строя с интенсивностью отказов 0.01. Время восстановления равно 25 ед.вр. При отказе прибора заявка теряется.

Необходимо написать программу, моделирующую эту систему в течении 3000 ед.вр. Выведите статистику по очередям в виде таблиц (для этого определить места возможного образования очередей и описать их). Оцените количество отказов. Начертите блок-диаграмму.

Дана система, состоящая из двух последовательно соединенных приборов. Заявки поступают в среднем через 8 ед.вр. Среднее время задержки у обоих приборов 20 ед.вр. Десятая часть заявок после обработки в первом приборе возвращается на повторную обработку, причем среднее время задержки в первом приборе при этом уменьшается до 10.

Выведите статистику по очередям в виде таблиц за 1000 ед.вр. Определите среднее время пребывания заявок в модели и вероятность обслуживания заявки в первом приборе три и более раз. Начертите блок-диаграмму.

Создать программу, моделирующую процесс прохождения заявок через прибор. Модель имеет два источника поступления заявок, один из которых выпускает заявки с повышенным приоритетом. Безприоритетные заявки поступают в среднем через 10 ед.вр., высокоприоритетные - через 20 ед.вр. Если у прибора нет возможности принять заявку, она становится в очередь, число мест которой ограничено Z. Очередь организована таким образом, что на две высокоприоритетные заявки пропускается одна низкоприоритетная. Среднее время обработки в приборе равно 25.

Оцените среднее время пребывания низкоприоритетных заявок в модели, затабулировав соответствующий СЧА. Аналогичную оценку сделайте и для высокоприоритетных заявок. Определите количество потерь для каждого типа заявок для времени моделирования 1000 ед.вр. Начертите блок-диаграмму модели.

Дана система, состоящая из двух приборов (основного и дублирующего). Заявки поступают в среднем через 8 ед.вр. и после обработки в основном приборе покидают систему. Если средняя длина очереди к основному прибору превышает 3, заявки становятся в очередь к дублирующему прибору. Среднее время задержки у первого прибора 12 ед.вр., у второго - 6 ед.вр.

Выведите статистику по очередям. Подсчитайте количество обслуженных заявок за время моделирования 1000 ед.вр. Начертите блок-диаграмму.

Рассматривается модель из 2-х параллельно соединенных приборов с отказами. Параметры приборов одинаковы: среднее время обработки заявок = 20; интенсивность отказов = 0.01. Длина очереди к первому прибору равна 5, к второму прибору – 8.

Заявки поступают в среднем через 8 ед.вр. Пришедшая в модель заявка занимает очередь с меньшей длиной, и после обслуживания в соответствующем приборе покидает модель. Если в обеих очередях нет места, заявка теряется. При отказе прибора заявка также теряется. Оба прибора могут при отказе ремонтироваться одновременно, среднее время восстановления равно 25 ед.вр.

Оцените количество потерянных заявок. Измените программу, введя 3-й прибор, проанализируйте поведение системы (повысилась ли надежность?). Можно ли без ущерба для надежности заменить 2-х ремонтников одним? Начертите блок-диаграмму.

# Проведение однофакторных экспериментов

### Теория

Построение математических моделей с использованием только аналитических методов, как правило, трудно реализуемо на практике из-за недостатка необходимой информации о моделируемом процессе или объекте. Поэтому, помимо аналитического подхода используют и экспериментальные методы, позволяющие либо получить необходимые для построения модели данные, либо получить собственно математическое описание объектов в упрощенном виде.

Имитационное моделирование является машинным экспериментом над моделью исследуемой или проектируемой системы. Эффективность же использования экспериментальных ресурсов зависит от выбора плана эксперимента. Для этого необходимо проектировать не только саму модель, но и планировать процесс ее использования, т. е. проведения с ней экспериментов на ЭВМ. Основная задача планирования машинных экспериментов — получение необходимой информации об исследуемой системе с учетом таких ограничений, как уменьшение затрат машинного времени на моделирование, увеличение точности и достоверности результатов моделирования, проверка адекватности модели и т. д.

Весь процесс проведения эксперимента можно разбить на три этапа.

1. Разработка плана проведения серии экспериментов.
2. Проведение экспериментов.
3. Обработка полученных данных (регрессионный анализ).

*yk*

*xm*

“черный ящик”

*x1*

*y1*

Рис 5.1 - Представление системы в виде «черного ящика»

Планирование экспериментов основано на представлении объекта как «*черного ящика»* (рис. 5.1), на вход, которого подаются переменные: *х1, х2,..., xk,* и на выходе получают *у1, у2, ..., уk.* Целью экспериментов является изучение влияния переменных *х* (называемыхфакторами) на переменные *у* (называемых реакциями, откликами). В экспериментах над моделями фактор является экзогенной или управляемой (входной) переменной, а реакция — эндогенной (выходной) переменной.

Соотношения между реакциями (откликами) системы и факторами



называют функциями реакции (или уравнениями регрессии), а геометрический образ, соответствующий функциям реакции, - поверхностями реакции. Каждый фактор *хi* (*i=1, m*) может принимать в эксперименте одно из нескольких значений, называемых уровнями, как показано на рис.5.2 для случая двух факторов *х1* и *х2* и одной функции отклика (для *х1* выбрано три уровня, для *х2* - два).

В лабораторной работе будет рассматриваться случай, когда наблюдение ведется лишь за одной функцией отклика (*k=1*) при одном изменяемом факторе (*m=1*). Все дальнейшие рассуждения касаются именно этого случая.

0

x2

x1

y

Рис. 5.2

Любая функция может быть представлена в табличном, аналитическом или в графическом видах. Получаемая в ходе проведения эксперимента функция отклика имеет табличное представление, т.е. задается *n* парами значений абсцисс *xi* и ординат *yi* (рис. 5.3а), где *n* – количество уровней изменения фактора *x*. Для аналитического описания функции *y = ψ(x)* можно было бы применить полином. Например, при наличии всего лишь двух экспериментальных точек, через них можно провести простейшую кривую в виде прямой, т.е. уравнение регрессии имело бы вид . В случае *n=3* применима формула параболы , для *n* точек - полином *n* степени (рис. 5.3б). Однако следует учитывать тот момент, что в машинных экспериментах изучаемая модель является лишь приближенной копией реального объекта или процесса и, следовательно, полученные в ходе проведения опытов значения *yi* содержат некоторые погрешности. Данное обстоятельство не позволяет требовать, чтобы кривая уравнения регрессии проходила обязательно через все экспериментальные точки, а требовать, чтобы она лишь приближенно описывала зависимость функции отклика *y* от фактора *x* (рис. 5.3в). Вид уравнения регрессии выбирается достаточно простым и содержащим небольшое число параметров. Наибольшее применение получили такие зависимости, как

* линейная ;

а) б) в)

Рис. 5.3.

* гиперболическая ;
* степенная ;
* показательные:  и ;
* экспоненциальные:  и ;
* логарифмические:  и ;
* параболическая .

Таким образом, результатом проведенных экспериментов является таблица значений *xi* и *yi*. Далее выбирается аналитический вид функции *y(x).* На следующем этапе вычисляются параметры *bj* выбранной функции, которая бы с минимальной погрешностью приближалась бы к исходной функции *yi(xi).*Для определения *bj*используется **метод наименьших квадратов** (МНК), основная цель которого состоит в минимизации функции среднеквадратичного отклонения:

 (5.1)

Если бы в качестве аналитической функции была бы выбрана линейная зависимость, то функция *Z* будет иметь вид:



Итак, весь процесс сводится к нахождению таких значений коэффициентов *b0, b1,* которые давали бы минимальное расхождение между расчетными и измеренными значениями *y.*

Имеется другой вариант функции (5.1)

, (5.2)

которая позволяет достичь более точной аппроксимации для малых значений ординат функции отклика. Недостатком этой формулы является «завышение» графика получаемой аналитической зависимости.

Для вышеприведенных аналитических зависимостей имеются готовые алгоритмы нахождения *bj*, однако в данной работе мы применим общий подход, основанный на использовании оптимизирующих методов. Отметим, что функция, значение которой минимизируется, называется **целевой**, а методы, предназначенные для поиска минимума целевой функции, носят название **оптимизирующих**, одним из которых является метод деформируемого многогранника (метод Нелдера-Мида) [1]. Описание алгоритма данного метода дано в Приложении.

Для вычисления *bj* также воспользуемся возможностями MS Excel.

### Задание

**Цель работы:** выбор вида уравнения регрессии *y=ψ(x*) и вычисление его коэффициентов на основе обработки результатов эксперимента.

 В соответствии с вариантом составить GPSS-модель. Обеспечить вывод в файл отчета с помощью сохраняемых ячеек значений заданного уровня фактора и получаемой в ходе эксперимента функции отклика.

Провести эксперименты над GPSS-моделью и полученные результаты *yi* занести в таблицу Excel под заголовками *xi* и *yi (experim)*.

Выбрать необходимый вид аналитической зависимости *y = ψ(x)* и для вычисления коэффициентов *bj* выбранного уравнения регрессии формулу целевой функции внести в программу nm.pas. Также в программу ввести значения *xi* и *yi*. Учесть, что в программе искомые значения обозначены как *x,* т.е. коэффициент *b0* следует записать как *x[1], b1* — как *x[2]* и т.д. Для фактора следует создать одномерный массив, определив его как типизированную константу. Например,

const

ur=5; {число уровней}

fact:array[1.. ur] of real = …

Значения функции отклика следует также занести в типизированную константу-вектор. Впрочем, значения фактора и функции отклика можно записать и в операторе вычисления целевой функции.

 Провести вычисления, повторив их не менее пяти раз, меняя каждый раз координаты начальной вершины многогранника (см. Приложение). Результаты расчетов занести в табл. 5.1.

 Таблица 5.1

|  |  |  |
| --- | --- | --- |
| Номер прогона | Исходные данные | Результат |
| *b0* | … | *bn* | Размер многогранника | *b0* | … | *bn* | Значение целевой функции |
| 1 |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |

Повторные расчеты необходимы для выявления наличия у целевой функции помимо глобального минимума еще и локальных. При наличии у функции нескольких минимумов расчеты по программе могут привести к различным результатам, т.е. к разным значениям координат точки минимума с разными значениями целевой функции в них. Глобальному минимуму будет соответствовать точка с наименьшим значением целевой функции.

После определения коэффициентов *bi* сравнить теоретические и экспериментальные значения функции отклика, добавив к таблице Excel столбец *yi (teoret1)* с вычисленными по уравнению регрессии теоретическими значениями. Вычисления проводить с помощью формул Excel, для чего найденные значения *bi* также занести в рабочий лист Excel.

Используя первые два столбца таблицы Excel, построить диаграмму типа точечная/точечная. Добавить линию тренда, активировав при этом флажки «Поместить уравнение на диаграмме» и «Поместить на диаграмму величину достоверности аппроксимации R2». Чем ближе значение R2 к единице, тем точнее аппроксимация. Используя R2, выбрать из списка предлагаемых Excel функций наиболее подходящий вид уравнения регрессии. Значения *yi*, рассчитанные по выбранному уравнению, добавить к таблице Excel в виде четвертого столбца *yi (teoret2)*.

Используя возможности Excel, рассчитать среднеквадратичные отклонения по формуле (5.1) для обоих уравнений регрессии и выбрать наиболее оптимальный вариант уравнения.

На основе анализа полученного регрессионного уравнения сделайте вывод о том, какие меры следует принять для улучшения характеристик модели.

### Пример выполнения задания

Задание.

Поступление заявок подчиняется равномерному закону с интервалом X плюс/минус 2 мин, обработка – экспоненциальному со средним значением 10 мин. Если у прибора нет возможности принять заявку, она становится в очередь OR. Исследовать зависимость средней длины очереди от времени прихода заявок.

Решение.

Выберем экспериментальное окно для значения X в пределах [7, 11] с количеством уровней равным 5. Таким образом, X при каждом новом прогоне модели будет принимать значения 7, 8, …, 11. Значения X и средней длины очереди OR будем выводить в отчет с помощью сохраняемых ячеек INTER и SRDLOR. Время моделирования выберем равным 1000. Текст программы имеет вид

 SIMULATE

1 EXPON FUNCTION RN3,C24

0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69

.6,.915/.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12

.9,2.3/.92,2.52/.94,2.81/.95,2.99/.96,3.2/.97,3.5

.98,3.9/.99,4.6/.995,5.3/.998,6.2/.999,7/1.,8

2 INITIAL X$INTER,7

3 GENERATE X$INTER,2

4 QUEUE OR

5 SEIZE 1

6 DEPART OR

7 ADVANCE 10,FN$EXPON

8 RELEASE 1

9 TERMINATE

10 GENERATE 1000

11 SAVEVALUE SRDLOR,QA$OR

12 TERMINATE 1

 START 1

 END

Ниже приведены результаты 5 прогонов модели

|  |  |
| --- | --- |
| SAVEVALUE | VALUE |
| Номер прогона |
| 1 | 2 | 3 | 4 | 5 |
| INTER | +7 | +8  | +9  | +10  | +11  |
| SRDLOR | +11 | +5  | +2  | +1  | +0  |

На рис. 5.4 показаны три варианта уравнения регрессии, полученных с помощью Excel. Наилучшая аппроксимация достигнута для квадратного полинома. Отметим, что две другие функции приводят к отрицательным значениям, тогда как действительная кривая функции средней длины очереди должна асимптотически приближаться к оси абсцисс, не пересекая ее. Также отметим, что и полиномиальная кривая при ее экстраполяции вправо (за пределы экспериментального окна) приводит к увеличению значения средней длины, что тоже неверно.



Рис. 5.4.

Проведем расчеты с помощью программы nm.pas. Учитывая, что полученные с помощью Excel функции плохо описывают поведение исследуемой величины в последней четверти экспериментального окна, выберем в качестве целевой функцию (5.2). Добавим проверку функции отклика на неотрицательность - если указанное условие будет нарушено, то значение функции отклика будет увеличиваться в 10 раз, что приведет к возрастанию целевой функции. Проверим на пригодность в качестве аппроксимирующего уравнения несколько измененную версию гиперболической зависимости: 

const

 ur=5;

 fact:array[1..ur] of real=(7,8,9,10,11);

 funct:array[1..ur] of real=(11,5,2,1,0);

 {Вычисление целевой функции}

 procedure functmin(var z:real; b:vect1);

 var i:integer;

 y\_theor,y\_theor1:real;

 begin

 tev:=tev+1;

 z:=0;

 for i:=1 to ur do

 begin

 y\_theor:=b[1]+b[2]/(b[3]+fact[i]);

 if y\_theor<0 then y\_theor1:=y\_theor\*10

 else y\_theor1:=y\_theor;

 z:=z+sqr((funct[i]-y\_theor1)/y\_theor);

 end;

 z:=sqrt(z/ur)

 end;

Результаты пяти прогонов сведены в табл. 5.2

Таблица 5.2

|  |  |  |
| --- | --- | --- |
| Номер прогона | Исходные данные | Результат |
| *b0* | *b1* | *b2* | Размер многогранника | *b0* | *b1* | *b2* | Значение целевой функции |
| 1 | 1 | 1 | 1 | 1 | -1,04 | 7,70 | -6,45 | 0,471 |
| 2 | 3 | 3 | 3 | 3 | -7,78 | 58,32 | -3,50 | 0,484 |
| 3 | -5 | 35 | -5 | 1 | -2,95 | 16,35 | -5,84 | 0,451 |
| 4 | -3 | 10 | -5 | 2 | -2,66 | 14,72 | -5,96 | 0,451 |
| 5 | -6 | 40 | 1 | 1 | -5,51 | 34,35 | -4,77 | 0,463 |

Анализ показывает, что наблюдается довольно большой разброс значений коэффициентов. Выберем третий вариант: . Отметим, что экстраполяция по этой формуле также ведет к ошибочным результатам: за правую границу экспериментального окна дает отрицательные значения, а за левой границей функция стремится к бесконечности при *x → 5,84*. Данный пример показывает, что регрессионные уравнения применимы только в пределах экспериментального окна, в рамках которого они получены, а экстраполяция допустима лишь на небольшое расстояние.

Второй проверенной формулой стала логарифмическая зависимость , приведшая к результатам *b0 = 25,33, b1 = -12,03* и *b2 = -2,78* при значении целевой функции 0,536*.* На рис 5.5. приведены сравнительные графики полученных уравнений регрессии. Здесь Ряд1 соответствует экспериментальным точкам, Ряд2 – гиперболе, Ряд3 - логарифмической зависимости. Более точной является гиперболическая зависимость.

Отметим, что хотя расчеты с использованием оптимизирующих методов являются более трудоемкими, чем например, с помощью Excel, они дают исследователям более гибкие возможности по подбору вида аппроксимирующей зависимости.

### Варианты



Рис. 5.5.

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | Вариант задания из лаб. раб. 4 | Варьируемый фактор | Функция отклика  |
| 1 | 14 | Интенсивность отказов приборов | Количество потерянных заявок |
| 2 | 1 | Процент заявок с повышенным приоритетом | Среднее время пребывания безприоритетных заявок в модели |
| 3 | 2 | Максимально допустимая длина очереди Z | Число потерянных заявок |
| 4 | 3 | Максимально допустимая длина очереди Z | Среднее время пребывания безприоритетных заявок в модели |
| 5 | 4 | Интервал поступления заявок | Количество потерянных заявок |
| 6 | 5 | Интенсивность потока отказов | Коэффициент использования дублирующего прибора |
| 7 | 6 | Интервал поступления заявок | Количество потерянных заявок |
| 8 | 7 | Максимально допустимая длина очереди к основному прибору | Количество потерянных заявок |
| 9 | 8 | Максимально допустимая длина очереди | Количество потерянных заявок |
| 10 | 9 | Время обслуживания в первом приборе | Среднее время пребывания безприоритетных заявок в модели |
| 11 | 10 | Время восстановления 2-го прибора | Средняя длина очереди к 2-му прибору |
| 12 | 11 | Процент возвращаемых на вторичную обработку заявок | Среднее время пребывания заявок в модели |
| 13 | 12 | Число мест в очереди | Среднее время пребывания низкоприоритетных заявок в модели |
| 14 | 13 | Средняя длина очереди к основному прибору | Коэффициент использования дублирующего прибора |

# Проведение многофакторных экспериментов

### Теория

Как и в предыдущей работе наблюдение будет вестись за одной функцией отклика, но отличие от той работы будет состоять в том, что будет несколько факторов:

*y= ψ(x1, х2, ..., xm)*

При планировании эксперимента с моделью необходимо отобрать факторы *хi,* влияющие на искомую характеристику, установить диапазоны изменения факторов – границы экспериментального окна: [*ximin*; *ximax*],определить требуемое количество уровней для каждого фактора, оценить необходимое число реализации и их порядок в эксперименте. Выбранные факторы должны быть управляемы. *Фактор* называется *управляемым,* если его уровни целенаправ­ленно выбираются исследователем в процессе эксперимента. Выбранные факторы также должны быть совместимы и независимы. Совместимость факторов означает, что все их комбинации осуществимы, а независимость соответствует воз­можности установления фактора на любом уровне независимо от уровней других.

Эксперимент, в котором реализуются все возможные состояния уровней факторов, называется *полным факторным экспериментом (ПФЭ).* Общее число различных состояний уровней в ПФЭ можно определить по формуле:

*S = k1 ∙ k2 ∙ k3 ∙… ki ∙… km ,*

где *ki* – число уровней *i-*го фактора. Если число уровней для всех факторов одинаково, то *S = km*. Каждому сочетанию уровней факторов соответствует одно наблюдение функции отклика. Очевидно, что увеличение числа факторов и числа уровней приводит к резкому росту числа наблюдений. Если имеется два фактора и для каждого из них задаются по три уровня, то для проведения эксперимента потребуется *S = 32 = 9* наблюдений, для четырех факторов и четырех уровней - *S = 44 = 256* наблюдений. Для уменьшения числа наблюдений применяют план эксперимента с варьированием всех *m* факторов на двух уров­нях: *ximin*, *ximax*.  Количество наблюдений функции откликав этом случае составит *S = 2m*, а саму функцию*ψ* прибли­женно можно представить в виде линейного полинома:

 (6.1)

Коэффициент *bi* отражает влияние фактора *xi* на реакцию *y*, коэффициент *bij* – взаимовлияние факторов *xi* и *xj*, коэффициент *bij…k* – взаимовлияние факторов *xi* , *xj* , …, *xk*.

В табл. 6.1 приве­ден план проведения двухфакторного эксперимента: *S = 22* .

Таблица 6.1

|  |  |  |
| --- | --- | --- |
| Номер испытания | План ПФЭ | Реакция *y* |
| *x1* | *x2* |
| 1 | *x1min* | *x2min* | *y1* |
| 2 | *x1min* | *x2max* | *y2* |
| 3 | *x1max* | *x2min* | *y3* |
| 4 | *x1max* | *x2max* | *y4* |

Функция отклика *ψ* при этом будет иметь вид

*y = b0 + b1 x1+ b2 x2+ b3 x1 x2.*

Для ПФЭ типа 23 план эксперимента приведен в табл. 6.2.

Таблица 6.2.

|  |  |  |
| --- | --- | --- |
| Номер испытания | План ПФЭ | Реакция *y* |
| *x1* | *x3* | *x3* |
| 1 | *x1min* | *x2min* | *x3min* | y1 |
| 2 | *x1min* | *x2min* | *x3max* | y2 |
| 3 | *x1min* | *x2max* | *x3min* | y3 |
| 4 | *x1min* | *x2max* | *x3max* | y4 |
| 5 | *x1max* | *x2min* | *x3min* | y5 |
| 6 | *x1max* | *x2min* | *x3max* | y6 |
| 7 | *x1max* | *x2max* | *x3min* | y 7 |
| 8 | *x1max* | *x2max* | *x3max* | y 8 |

Уравнение регрессии *ψ* можно представить в виде

*y = b0 + b1 x1+ b2 x2+ b3 x3+ b4 x1 x2+ b5 x2 x3+ b6 x1 x3+ b7 x1 x2 x3*

Коэффициент *b7* учитывает взаимовлияние всех трех факторов друг на друга.

Основная идея регрессионного анализа

Для исследования взаимосвязи между величинами *Х* (вход) и *y* (выход) используются методы корреляционного и регрессионного анализа. Результаты корреляционного анализа позволяют сделать вывод о степени зависимости между переменными, а форма зависимости уточняется методами регрессионного анализа.

На значение величины *y* оказывают влияние стохастические воздействия разного рода, поэтому форма связи между величинами *Х* и *y* определяется линией регрессии, показывающей, как в *среднем* изменяется величина *y*при изменении входной величины *Х***,** т.е. приходится говорить о связи средних значений величины*y* c *X*. Эту связь характеризуют условным математическим ожиданием величины *y*, вычисляемым при условии, что величина *Х*приняла определенное значение, а аппроксимирующая функция строится*как функция регрессии* *ψ(X, B) ≈ M[y/X],* где *B* - неизвестные параметры уравнения регрессии.

Задача регрессионного анализа ставится следующим образом. Для каждого i-го опыта имеется набор значений (*xi1,...,xim*) входных параметров
*x1*÷ *xm* и соответствующее им значение выходного параметра *yi*. Пример опытных данных приведен в табл. 6.3.

Таблица 6.3

|  |  |  |
| --- | --- | --- |
| Номер испытания | Факторы | Измеренная в ходе эксперимента реакция *y* |
| *x1* | *x2* | *---* | *xm* |
| 1 | *x11* | *x12* | *---* | *x1m* | *y1* |
| 2 | *x21* | *x22* | *---* | *x2m* | *y2* |
| ----- | *-----* | *-----* | *---* | *-----* | *-----* |
| *k* | *xk1* | *xk2* | *---* | *xkm* | *yk* |

Задача сводится к определению значений коэффициентов уравнения регрессии *b0, b1, ... , bn,* которые с определенной степенью вероятности будут отражать влияние аргументов *xi1,..., xim* на *y*.

Для определения *bk*используется **метод наименьших квадратов** (МНК), смысл которого сводится к минимизации функции:

 (6.2)

где *yi* - фактическое (экспериментальное) значение выходной переменной,

- рассчитанное по уравнению регрессии значение выходной переменной.

Выражение (6.2) с учетом (6.1) будет иметь вид:

 (6.3)

В данной работе для вычисления *b0* следует использовать метод Нелдера-Мида, который реализован в программе nm.pas.

Отметим, что современные системы моделирования позволяют полностью автоматизировать весь процесс проведения многофакторных экспериментов, начиная с составления плана эксперимента, непосредственно проведения экспериментов и завершая оценкой влияния факторов на функцию отклика. В качестве примера можно привести общецелевую систему моделирования GPSS World [3]. Исследователь для своей модели указывает функцию отклика и факторы, назначает нижнюю и верхнюю границы интервала варьирования каждого фактора, выбирает тип эксперимента: дисперсионный или регрессионный анализ. Дисперсионный анализ (отсеивающий эксперимент) показывает силу влияния каждого фактора на наблюдаемую величину (отклик). Регрессионный анализ (оптимизирующий эксперимент) позволяет получить уравнение регрессии и определить для полученного уравнения координаты точки экстремума.

### Задание

**Цель работы:** определить коэффициенты *b0, b1, ... , bn* уравнения регрессии *y=ψ(X*) на основе обработки результатов эксперимента.

 В соответствии с вариантом составить GPSS-модель. Разработать план проведения эксперимента, следуя следующим указаниям:

* для каждого фактора определить по два уровня - *ximin, ximax*;
* в Excel построить таблицу, аналогичную табл. 6.3.

Провести эксперименты над GPSS-моделью и полученные результаты *yi* занести в табл. 6.3.

Для вычисления коэффициентов *bi* уравнения регрессии формулу (6.3) внести в программу nm.pas. Для факторов следует создать двумерный массив, определив его как типизированную константу. Например,

const

ur=2; {число уровней}

m=2; {число факторов}

fact:array[1.. ur, 1..m] of real = …

Методика вычислений такая же как и в предыдущей работе. Результаты расчетов занести в таблицу, аналогичную 5.1.

После определения коэффициентов *bi* следует сравнить теоретические и экспериментальные значения функции отклика, добавив к табл. 6.3. столбец  с вычисленными по (6.1) теоретическими значениями.

В случае, если проводился двухфакторный эксперимент, построить график поверхности отклика. Для этого заполнить табл. 6.4.

Таблица 6.4

|  |  |  |
| --- | --- | --- |
|  | *x1min* | *x1max* |
| *x2min* | *y1* | *y3* |
| *x2max* | *y2* | *y4* |

Тип графика выбрать *Поверхность, проволочная*.

На основе анализа полученного регрессионного уравнения сделайте вывод о том, какие меры следует принять для улучшения характеристик модели.

### Варианты

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | Задача | Функция отклика | Варьируемые факторы |
| Первый фактор | Второй фактор |
| 1 | 1 | коэффициент использования первого процессора | интервал моделирования*x 1min* =100 задач*x 1max* =500 задач | интервалы времени между поступлениями задач*x 2min* =2 ед.вр.*x 2max* =8 ед.вр. |
| 2 | коэффициент использования второго процессора |
| 3 | коэффициент использования внешней памяти |
| 4 | среднее время нахождения транзакта в системе |
| 5, 6, 7 | средняя длина очереди:5) A1; 6) A2;7) A3 |
| 8, 9, 10 | максимальная длина очереди: 8) A1; 9) A2;10) A3 |
| 11 | В задаче 1 предусмотреть уход транзактов из системы без обслуживания, если длина очереди А1 превысит 5 | коэффициент использования первого процессора |
| 12 | коэффициент использования второго процессора |
| 13 | коэффициент использования внешней памяти |
| 14 | среднее время нахождения транзакта в системе |
| 15,16,17 | средняя длина очереди:15) A1;16) A2;17) A3 |
| 18, 19,20 | максимальная длина очереди: 18) A1; 19) A2;20) A3 |
| 21 | количество заявок, покинувших систему без обработки. |
| 22,23,24 | 2 | коэффициент использования:22) A1;23) A2;24) A3 | единица времени*x 1min* = 1 мс*x 1max* = 1,7 мс | количество дисководов*x 2min* = 3*x 2max* = 4 |
| 25,26 | средняя длина очереди:25) A1;26) A4 |

#### Задача №1

Требуется промоделировать решение задач в двухпроцессорной ЭВМ с общей памятью, разделенной на восемь блоков. Каждой задаче отводится при ее решении один блок. Интервалы времени между поступлениями задач распределены по экспоненциальному закону со средним временем 8 единиц времени, время обработки порции информации также подчинено экспоненциальному закону с интенсивностью v1=5 в процессоре CPU1 и с v2=2 в процессоре CPU2.

Между обработкой порций с вероятностью 0,6 возможно обращение к внешней памяти, в которой время обслуживания распределено равномерно в диапазоне [2,8]. С вероятностью 0,4 задачи оказываются решенными и покидают систему. Моделирование выполнить на отрезке времени, соответствующем решению не менее 100 задач. Ниже представлена схема имитационной модели и текст программы на языке GPSS.

 MEM STORAGE 8

 EXP FUNCTION RN1,C12

 0,0/.2,.22/.4,.51/.5,.69/.6,.92/.7,1.2/.8,1.61/

 .9,2.3/.95,3/.99,4.6/.999,6.9/1,100

 GENERATE 8,FN$EXP,,100

 QUEUE A1

 ENTER MEM,1

 DEPART A1

 M6 QUEUE A2

 TRANSFER BOTH,M1,M2

 M1 SEIZE CPU1

 DEPART A2

 ADVANCE 5,FN$EXP

 RELEASE CPU1

 TRANSFER ,M3

 M2 SEIZE CPU2

 DEPART A2

 ADVANCE 2,FN$EXP

 RELEASE CPU2

 M3 TRANSFER .6,M5,M4

 M4 QUEUE A3

 SEIZE DISK

 DEPART A3

 ADVANCE 5,3

 RELEASE DISK

 TRANSFER ,M6

 M5 LEAVE MEM,1

 TERMINATE 1

#### Задача №2

Промоделировать работу устройства дисковой памяти при наличии одного канала и трех дисководов. Запросы поступают равновероятные ко всем дисководам. Обработка запроса включает установку головки (при этом канал не требуется) и обмен данными через канал. Интервалы времени между поступлениями запросов распределены по экспоненциальному закону с v=6. Время установки головки равномерно распределено в интервале 0 - 50 мс. Время обмена данными равно 1,7 мс (за единицу времени принять 1,7 мс).

Ниже приведен текст программы на языке GPSS.

 EXP FUNCTION RN1,C12

 0,0/.2,.22/.4,.51/.5,.69/.6,.92/.7,1.2/.8,1.61/

 .9,2.3/.95,3/.99,4.6/.999,6.9/1,100

 GENERATE 6,FN$EXP

 TRANSFER .333,M2,M1

 M2 TRANSFER .5,M4,M3

 M1 QUEUE A1

 SEIZE DISK1

 DEPART A1

 ASSIGN 1,DISK1

 ADVANCE 15,15

 TRANSFER ,M5

 M3 QUEUE A2

 SEIZE DISK2

 DEPART A2

 ASSIGN 1,DISK2

 ADVANCE 15,15

 TRANSFER ,M5

 M4 QUEUE A3

 SEIZE DISK3

 DEPART A3

 ASSIGN 1,DISK3

 ADVANCE 15,15

 M5 QUEUE A4

 SEIZE CAN

 DEPART A4

 ADVANCE 1

 RELEASE CAN

 RELEASE P1

 TERMINATE 1

# Литература

1. Банди Б. Методы оптимизации. Вводный курс. Пер. с англ. – М.: Радио и связь, 1988.
2. Боев В.Д. Моделирование систем. Инструментальные средства GPSS World: Учеб. пособие. – СПб.: БХВ‑Петербург, 2004. – 368 с.
3. Голованов О. В., Дуваков С., Смирнов В. Н. Моделирование сложных дискретных систем на ЭВМ третьего поколения (опыт применения GPSS). М.: Энергия, 1978. - 160 с.
4. Курс лекций по дисциплине «Компьютерное моделирование»/ Сост. Алтаев А.А. – Улан-Удэ, Изд-во ВСГТУ, 2001. – 63 с.
5. Наставление по GPSS/PC. Minuteman Software перевод с английского под ред. Якимова И. М. Казань, 1997. - 320 с.
6. Советов Б. Я., Яковлев С. А. Моделирование систем. Учебник для ВУЗов. М.: Высшая школа, 1999.
7. Советов Б. Я., Яковлев С. А. Моделирование систем. Лабораторный практикум. М.: Высшая школа, 1989.
8. Федоров В.Н. Моделирование дискретных систем. Учебное пособие – М.: МГАПИ, 2005. – 92 с.
9. Шеннон Р. Дж. Имитационное моделирование систем - искусство и наука. М.: Мир, 1978. - 418 с.
10. Шрайбер Т. Дж. Моделирование на GPSS. М.: Машиностроение, 1980. - 592 с.

# Приложение

Метод деформируемого многогранника

Метод деформируемого многогранника или, второе его название, - метод Нелдера-Мида (Nelder-Mead) относится к методам оптимизации нулевого порядка, то есть к методам, использующим в ходе поиска минимума значения только самой функции. Функции первого порядка для поиска минимума помимо значений самой целевой функции используют еще и значения ее первой производной, методы второго порядка – значения второй производной и т.д.

*x1*

*x1*

Рис. П.1.

*x2*

*а)*

*б)*

Рассмотрим алгоритм метода в его классическом варианте. Итак, пусть необходимо найти минимум целевой функции от *n* переменных. В *n*-мерном эвклидовом пространстве строится многогранник с *n+1* вершиной. Например, в случае одной переменной многогранником будет отрезок (рис. П.1,а), для двух переменных − треугольник (рис. П.1,б), при *n=3* − это тетраэдр.

Далее используются следующие обозначения: *xij* − координата *i*-й вершины, *j* − номер переменной, *fi* — значение целевой функции *i*-й вершины. Помимо матрицы *X(N+1,N)* и вектора *F(N+1)* применяются вспомогательные векторы *XO(N), XC(N), XR(N), XE(N).*

В качестве исходных данных задаются:

* *n* − число переменных;
* координаты первой вершины многогранника − *x1j*, где ;
* начальный размер многогранника Δ.

Координаты других вершин определяются по формуле

, 

При этом создается равносторонний многогранник, называемый симплексом. Сам метод по этой причине имеет третье название − симплексный метод (не путать с симплекс-методом, который используется при решении задач линейного программирования).

Далее вычисляются значения целевой функции в вершинах − *fi* .

Дальнейшие действия состоят в перемещении построенного многогранника в область минимума функции. Алгоритм перемещения рассмотрим на примере целевой функции от двух переменных (рис. П.2).

1. Определяется вершина с наибольшим значением целевой функции. Номер такой вершины обозначим через *H*. Таким образом, *fh = max{f1, f2,…, fn}*. Также определяется вершина *L* с наименьшей функцией: *fl = min{f1, f2,…, fn}*.

 *Комментарий по алгоритму.* Основная идея метода состоит в том, что на каждом шаге очередная «наихудшая» вершина будет заменяться на новую, в которой значение целевой функции будет меньше. Это позволит смещать многогранник в направлении минимума.

2. Вычисляются координаты центра тяжести *O* всех вершин многогранника, исключая *H* :

*x1*

*R*

*E*

*O*

*C*

*H*

Рис. П.2.

*x2*

Линии равного уровня функции

.

3. Вычисляются координаты точки *R*, лежащей на продолжении отрезка *HO*, причем длины отрезков *HO* и *OR* равны:

*xrj=2∙xoj  −  xhj*

4. Вычисляется целевая функция в точке *R* − *fr* .

5. Если *fr < fh* , то идти к п. 6, иначе к п. 13.

*Комментарий по п.5*. Если выполняется условие *fr < fh* , то это означает, что точка *R* вполне может быть принята в качестве новой вершины. Операция замены *H* на *R* называется *отражением*. Но имеет смысл проверить и более отдаленную точку − *E*, выбор которой позволит увеличить размеры многогранника. Такая возможность увеличения размера позволяет повысить скорость движения многогранника, что повышает эффективность алгоритма в случае, если размеры исходного многогранника были выбраны небольшими и он находится далеко от точки минимума.

6. Проверяется точка *E* в качестве альтернативы точке *R*:

*xej=2∙xrj  −  xoj*

7. Вычисляется *fe* .

8. Если *fe < fh* , то идти к п. 9, иначе к п. 11.

9. В качестве новой вершины выбирается точка *E* (выполняется *растяжение*):

*xhj=xej*

10. Идти к п. 20.

11. В качестве новой вершины выбирается точка *R* (выполняется *отражение*):

*xhj=xrj*

12. Идти к п. 20.

13. Так как точка *R* не подходит в качестве новой вершины (*fr > fh*), то следующей рассматривается точка *C*, лежащая посередине отрезка *HO*:



14. Вычисляется *fc.*

15. Если *fc < fh* , то идти к п. 16, иначе к п. 18.

16. В качестве новой вершины выбирается точка *C* (выполняется *сжатие*):

*xhj=xcj*

17. Идти к п. 20.

18. Так как точки *R* и *C* не подошли в качестве альтернативы вершине *H,* выполняется всеобщее сжатие многогранника, причем наилучшая вершина *L* остается на месте, а координаты других пересчитываются так, чтобы они сместились в сторону наилучшей вершины:



19. Вычисляются функции *fi .*

20. Выводятся на печать: *l* − номер наилучшей вершины, *fl* − значение целевой функции в данной вершине, *xlj* − координаты вершины.

21. Проверяется условие выхода из цикла. Для этого вычисляется среднеквадратичное отклонение целевых функций для всех вершин, которое сравнивается с погрешностью расчета ε.



*Комментарий по п.21*. После перемещения многогранника в область минимума начинают преобладать операции сжатия и вершины постепенно стягиваются в одну точку, целевые функции при этом выравниваются по своим значениям. Это обстоятельство используется для завершения вычислений.

22. Если σ < ε, то идти к п. 23, иначе к п. 1.

23. Выводятся на печать: *fl* − значение целевой функции в наилучшей вершине, *xlj* − координаты вершины.

*Комментарий по алгоритму.* Поскольку алгоритм метода предусматривает возможность изменения размеров и формы многогранника за счет растяжения и сжатия, метод получил название *деформируемого* многогранника.

Ниже дана распечатка текста программы, в которой реализован алгоритм несколько отличающийся от рассмотренного выше. В частности, для построения исходного многогранника используется датчик случайных чисел. Имеются и другие отличия.

program nm;

const

 nMax=10;

type dim= array[1..nMax+1,1..nMax] of real;

 vect1=array [1..nMax] of real;

 vect2=array [1..nMax+1] of real;

var

 s: dim;

 x,xh,xg,xl,xo,xr,xc,xe:vect1;

 f: vect2;

 ko,z,k,fl,fh,s1,s2,sig:real;

 j,n,jc,i,tev,l,u,u1,h,g:integer;

 fg,fr,fc,fe:real;

{Выбор вершины}

 procedure choice(fn:real; x:vect1);

 var j: integer;

 begin

 for j:=1 to n do

 s[h,j]:=x[j];

 f[h]:=fn

 end;

 {Вычисление целевой функции}

 procedure functmin(var z:real; x:vect1);

 begin

 tev:=tev+1;

 z:=100\*sqr(x[2]-x[1])+sqr(1-x[1]);

 end;

begin

 randomize;

 writeln('симплексный метод');

 writeln('введи число пеpеменных');

 readln(n);

 tev:=0;

 ko:=1;

 writeln('Начальное пpиближение (x><0)');

 for j:=1 to n do

 begin

 writeln('введите x',j);

 read(s[1,j]);

 if s[1,j]=0 then

 s[1,j]:=0.001

 end;

 writeln('введите длину шага ( >1 )');

 readln(k);

 if k=1 then k:=1.1;

 u1:=n+trunc(1.2\*ln(k));

 fl:=1e+20;

 l:=1;

 for i:=1 to n+1 do

 begin

 f[i]:=1e+20;

 for u:=1 to u1 do

 begin

 for j:=1 to n do

 x[j]:=s[l,j]\*exp((1-2\*random)\*ln(k));

 functmin(z,x);

 ko:=0.5\*k+0.5;

 if f[i]>z then

 begin

 f[i]:=z;

 for j:=1 to n do

 s[i,j]:=x[j];

 if z<fl then

 begin

 l:=i;

 fl:=f[i];

 ko:=k\*3-2;

 k:=ko;

 end;

 end;

 end;

 end;

 repeat

 fh:=-1e+20;

 fl:=1e+20;

 for i:=1 to n+1 do

 begin

 if f[i]>fh then

 begin

 fh:=f[i];

 h:=i;

 end;

 if f[i]<fl then

 begin

 fl:=f[i];

 l:=i;

 end;

 end;

 fg:=-1e+20;

 for i:=1 to n+1 do

 begin

 if i<>h then

 if f[i]>fg then

 begin

 fg:=f[i];

 g:=i;

 end;

 end;

 for j:=1 to n do

 begin

 xo[j]:=0;

 for i:=1 to n+1 do

 xo[j]:=xo[j]+s[i,j];

 xo[j]:=(xo[j]-s[h,j])/n;

 xh[j]:=s[h,j];

 xg[j]:=s[g,j];

 xl[j]:=s[l,j];

 end;

 for j:=1 to n do

 xr[j]:=2\*xo[j]-xh[j];

 functmin(fr,xr);

 if fr<fl then

 begin

 for j:=1 to n do

 xe[j]:=2\*xr[j]-xo[j];

 functmin(fe,xe);

 if fe<fr then choice(fe,xe)

 else choice(fr,xr)

 end

 else

 begin

 if fr>fg then

 begin

 if fr<fh then choice(fr,xr)

 else

 begin

 for j:=1 to n do

 xc[j]:=(xh[j]+xo[j])/2;

 functmin(fc,xc);

 if fc>fh then

 begin {}

 for i:=1 to n+1 do

 begin

 for j:=1 to n do

 begin

 s[i,j]:=(s[i,j]+xl[j])/2;

 x[j]:=s[i,j];

 end;

 functmin(f[i],x);

 end

 end

 else choice(fc,xc)

 end

 end

 else choice(fr,xr)

 end;

 s1:=0;

 s2:=0;

 for i:=1 to n+1 do

 begin

 s1:=s1+f[i];

 s2:=s2+sqr(f[i]);

 end;

 sig:=s2-s1\*s1/(n+1);

 sig:=sig/(n+1);

 writeln(l:3,f[l]:12:8);

 for j:=1 to n do

 write(xl[j]);

 writeln;

 until sig<1e-10;

 writeln('Минимум найден в точке');

 for j:=1 to n do

 writeln('x[',j,']=',xl[j]);

 writeln;

 writeln('Значение минимума функции',f[l]);

 writeln('Количество вычислений функции=',tev);

 readln;

 end.

Целевая функция может иметь несколько минимумов. На рис. П.3 показана топология функции от двух переменных, которая имеет два минимума: глобальный *A* и локальный *B*. Если координаты начальной вершины многогранника *x1j* будут выбраны в окрестности точки B, то велика вероятность что будет получено неверное решение: многогранник «сползет» в овраг B. Во избежание подобных ошибок рекомендуется выполнить несколько расчетов, меняя каждый раз координаты начальной вершины. Из полученных решений следует выбрать результат с наименьшим значением целевой функции.

35

30

255

200

15

A

B

30

10

Рис. П.3.

В процедуре functmin записана функция , координаты точки минимума которой *x1 = x2 =1* при значении *z = 0*.