

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Качество и метрология программного обеспечения»

Тема: «Построение операционной графовой модели программы (ОГМП)
и расчет характеристик эффективности ее выполнения методом
эквивалентных преобразований»

Студент гр. 8304

Чешуин Д. И.

Преподаватель

Кирияничков В. А.

Санкт-Петербург

2022

Цель работы.

Построение операционной графовой модели программы (ОГМП) и расчет характеристик эффективности ее выполнения методом эквивалентных преобразований.

Задание.

Для задания из лабораторных работ 1-3 разработать операционную модель управляющего графа программы на основе схемы алгоритма. При выполнении работы рекомендуется для упрощения обработки графа исключить диалог при выполнении операций ввода-вывода данных, а также привести программу к структурированному виду.

Выбрать вариант графа с нагруженными дугами, каждая из которых должна представлять фрагмент программы, соответствующий линейному участку или ветвлению. При расчете вероятностей ветвлений, зависящих от распределения данных, принять равномерное распределение обрабатываемых данных в ограниченном диапазоне (например, $[0,100]$ - для положительных чисел или $[-100,100]$ - для произвольных чисел). В случае ветвлений, вызванных проверкой выхода из цикла, вероятности рассчитываются исходя априорных сведений о числе повторений цикла. Сложные случаи оценки вероятностей ветвлений согласовать с преподавателем.

В качестве параметров, характеризующих потребление ресурсов, использовать времена выполнения команд соответствующих участков программы. С помощью монитора Sampler выполнить оценку времен выполнения каждого линейного участка в графе программы.

Полученную ОГМП, представить в виде графа с нагруженными дугами, у которого в качестве параметров, характеризующих потребление ресурсов на дуге ij , использовать тройку $\{P_{ij}, M_{ij}, D_{ij}\}$, где:

P_{ij} - вероятность выполнения процесса для дуги ij ;

M_{ij} - мат. ожидание потребления ресурса процессом для дуги ij ;

D_{ij} - дисперсия потребления ресурса процессом для дуги ij .

В качестве потребляемого ресурса в данной работе рассматривается время процессора, а оценками мат. ожиданий времен для дуг исходного графа следует принять времена выполнения операторов (команд), соответствующих этим дугам участков программы. Дисперсиям исходных дуг следует присвоить нулевые значения.

Получить описание полученной ОГМП на входном языке пакета CSA III в виде поглощающей марковской цепи (ПМЦ) – (англ.) AMC (absorbing Markov chain) и/или эргодической марковской цепи (ЭМЦ) - EMC (ergodic Markov chain).

С помощью предоставляемого пакетом CSA III меню действий выполнить расчет среднего времени и дисперсии времени выполнения как для всей программы, так и для ее фрагментов, согласованных с преподавателем. Сравнить полученные результаты с результатами измерений, полученными в работе 3.

Ход работы.

1. Текст программы (исходный).

Была взята программа из 1-ой лабораторной работы. Исходный код программы представлен в приложении А.

2. Профилирование.

Программы из приложения А была разбита на функциональные участки. Код программы для профилирования, разделенной на функциональные участки, представлен в приложении Б.

3. Граф управления программы.

Был построен граф управления программы на основе разбиения программы на функциональные участки (см. пункт 2). Граф представлен на рисунке 1.

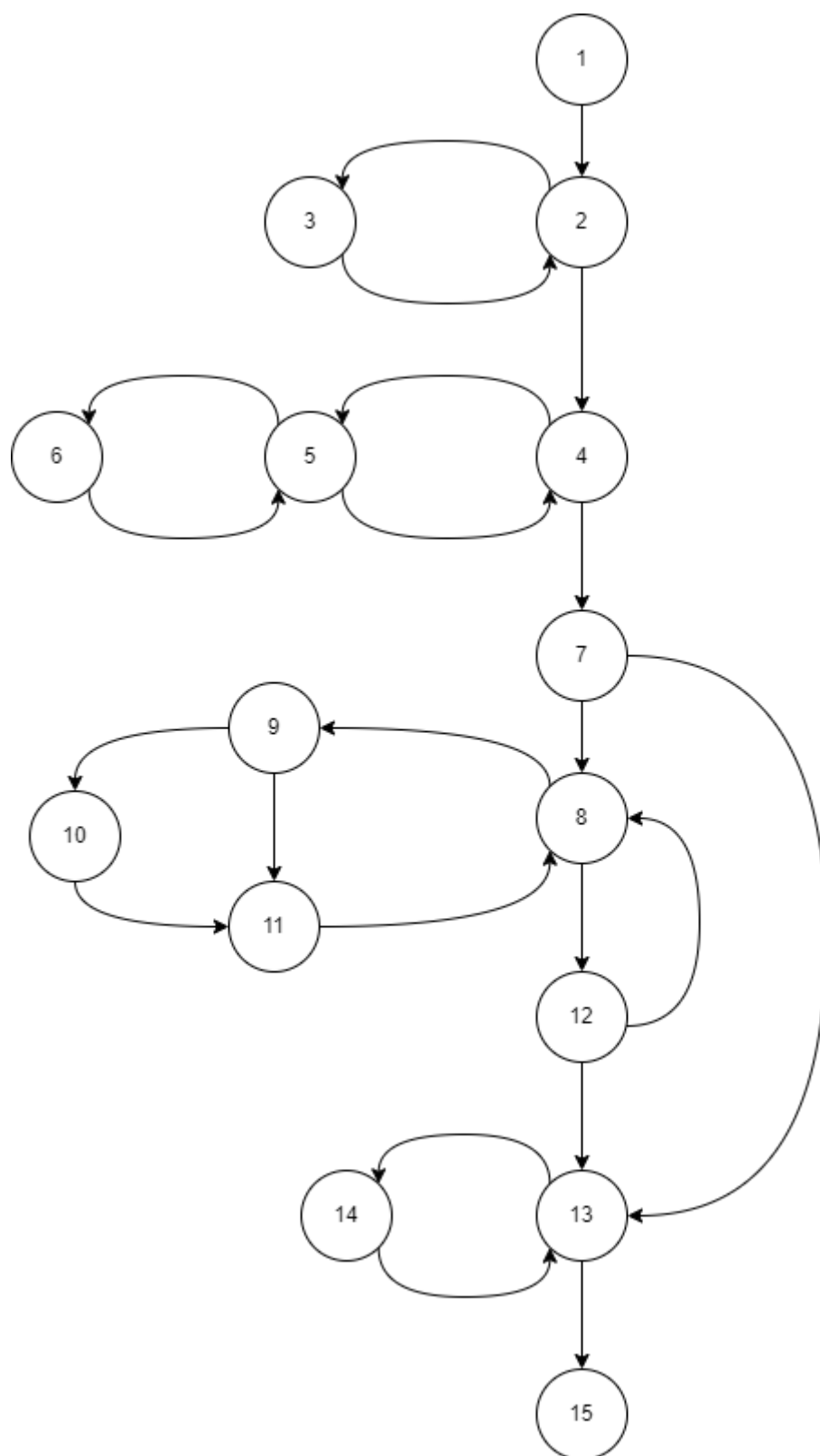


Рисунок 1 - Граф управления программы

4. Результаты профилирования.

Было рассчитано время выполнения ФУ для программы из приложения Б. Результаты профилирования представлены на рисунке 2.

исх	прием	общее время	кол-во проходов	среднее время
158	86	35.850	1	35.850
86	102	17.412	1	17.412
102	103	12.540	1	12.540
103	104	23.040	9	2.560
103	118	19.161	1	19.161
104	116	131.051	9	14.561
116	103	15.875	9	1.764
118	134	4.169	1	4.169
134	52	17.277	1	17.277
52	57	2.526	1	2.526
57	58	10.356	1	10.356
58	59	3.833	3	1.278
58	67	10.339	1	10.339
59	60	5.678	3	1.893
60	61	26.987	9	2.999
60	65	15.145	3	5.048
61	63	41.152	9	4.572
63	60	27.973	9	3.108
65	58	0.949	3	0.316
67	70	31.614	1	31.614
70	76	21.656	1	21.656
76	32	25.644	1	25.644
32	34	2.755	3	0.918
34	35	11.586	3	3.862
35	36	25.783	9	2.865
35	46	22.827	3	7.609
36	38	39.965	9	4.441
38	44	10.894	3	3.631
38	40	15.979	6	2.663
44	35	11.236	9	1.248
46	48	73.747	3	24.582
48	32	33.288	2	16.644
48	80	9.594	1	9.594
40	42	34.071	6	5.679
42	44	7.647	6	1.275
80	82	1.758	1	1.758
82	136	32.227	1	32.227
136	137	11.818	1	11.818
137	138	19.242	9	2.138
137	143	8.264	1	8.264
138	141	6440.486	9	715.610
141	137	23.862	9	2.651
143	145	174.133	1	174.133
145	160	26.770	1	26.770

Рисунок 2 – Результаты профилирования

5. Расчет вероятностей и затрат ресурсов для дуг управляющего графа.

Были произведены расчеты для дуг управляющего графа. Некоторые дуги соответствуют нескольким последовательным участкам кода, их вес рассчитывался суммированием времени выполнения. Результаты расчета представлены в таблице 1.

Таблица 1 – Расчет вероятностей и затрат ресурсов для дуг.

	Номера строк	Количество проходов	Вероятность
$L_{1-2} = 65,802$	158-86, 86-102, 102-103	1	1
$L_{2-3} = 17,121$	103-104, 104-116	9	0,9
$L_{3-2} = 1,764$	116-103	9	1
$L_{2-4} = 53,489$	103-118, 118-134, 134-52, 52-57, 57-58	1	0,1
$L_{4-5} = 3,171$	58-59, 59-60	3	0,75
$L_{4-7} = 41,953$	58-67, 67-70	1	0,25
$L_{5-4} = 5,364$	60-65, 65-58	3	0,25
$L_{5-6} = 7,571$	60-61, 61-63	9	0,75
$L_{6-5} = 3,108$	63-60	9	1
$L_{7-8} = 52,08$	70-76, 76-32, 32-34, 34-35	1	1
$L_{7-13} = 11,818$	70-72, 72-136, 136-137	0	0
$L_{8-9} = 7,306$	35-36, 36-38	9	0,75
$L_{8-12} = 32,191$	35-46, 46-48	3	0,25
$L_{9-10} = 8,342$	38-40, 40-42	6	0,67
$L_{9-11} = 3,631$	38-44	3	0,33
$L_{10-11} = 1,275$	42-44	6	1
$L_{11-8} = 1,248$	44-35	9	1
$L_{12-8} = 21,424$	48-32, 32-34, 34-35	2	0,67
$L_{12-13} = 55,397$	48-80, 80-82, 82-136, 136-137	1	0,33
$L_{13-14} = 717,748$	137-138, 138-141	9	0,9
$L_{13-15} = 209,364$	137-143, 143-145, 145-160	1	0,1
$L_{14-13} = 2,651$	141-137	9	1

6. Операционная графовая модель программы.

Была составлена операционная графовая модель программы с помощью рисунка 1 и таблицы 1. Операционная графовая модель программы представлена на рисунке 3.

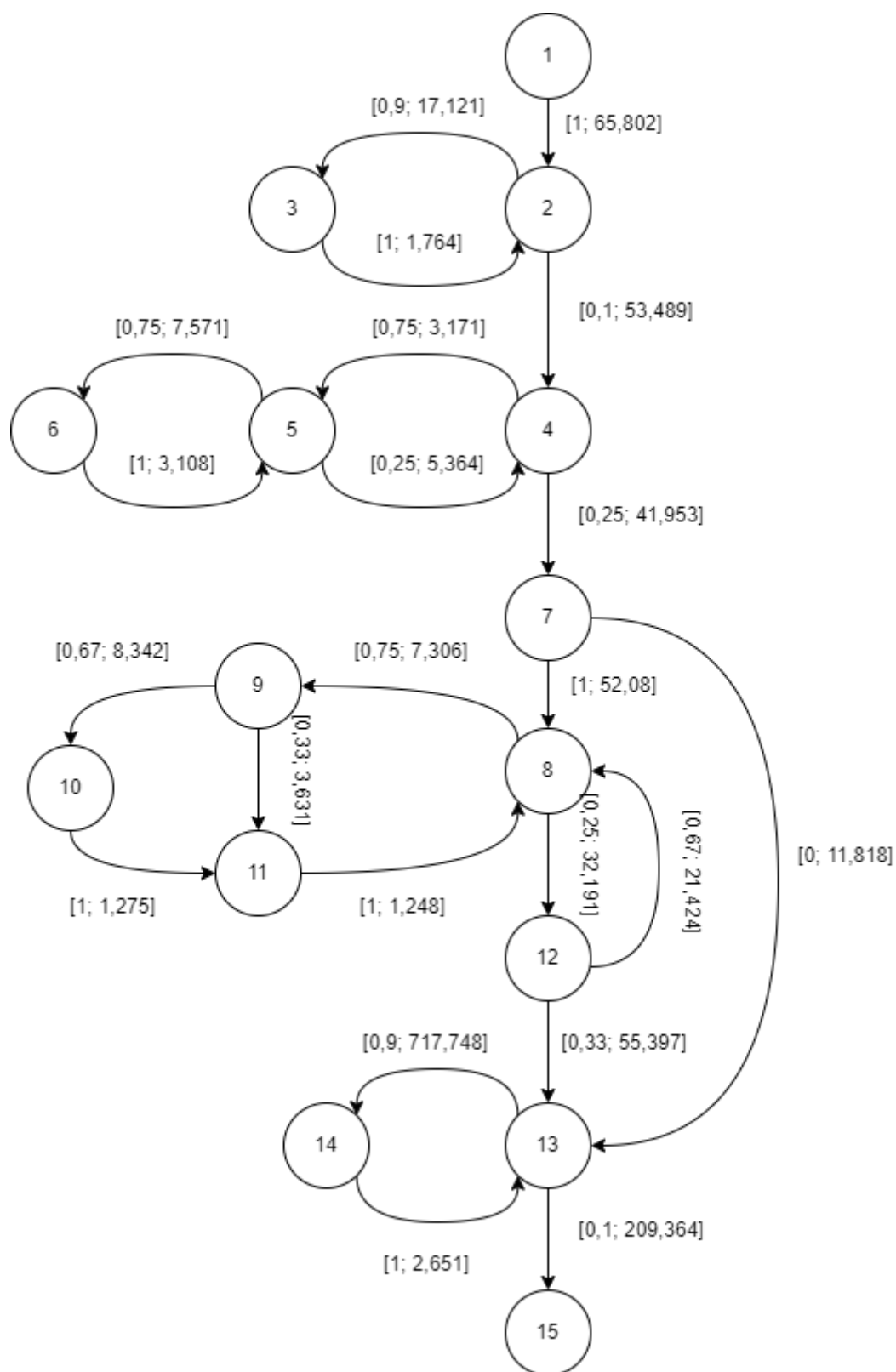


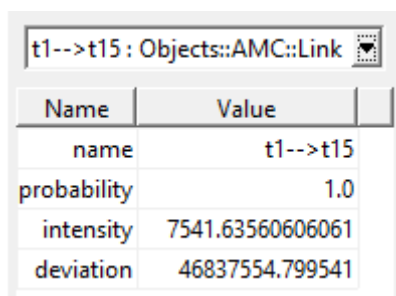
Рисунок 3 - Операционная графовая модель

7. Описание модели model.xml.

Было составлено описание модели в формате xml. Результат представлен в приложении В.

8. Результаты.

Была запущена программа по обработке ОГМП. Результаты работы программы представлены на рисунке 4.



The screenshot shows a window titled "t1-->t15 : Objects::AMC::Link". Inside the window is a table with two columns: "Name" and "Value". The table contains four rows of data.

Name	Value
name	t1-->t15
probability	1.0
intensity	7541.63560606061
deviation	46837554.799541

Рисунок 4 - Результат работы программы

Согласно расчётам программы, среднее время выполнения составляет 7541,635 мкс. В пункте 4 данного отчёта приведен результат профилирования программы с использованием SAMPLER_v2, где суммарное время выполнения составило 7511,39 мкс. В итоге, разница между результатами составляет 0.7 %.

Выводы.

В ходе выполнения лабораторной работы была построена операционная графовая модель заданной программы, нагрузочные параметры которой были оценены с помощью профилировщика SAMPLER_v2 и методом эквивалентных преобразований с помощью пакета CSA III были вычислены математическое ожидание и дисперсия времени выполнения для всей программы. Результаты сравнения этих характеристик с полученными в ходе выполнения лабораторной работы №3 согласуются.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define MAXR 9
#define MAXC 3

void get_data(double x[MAXR], double y[MAXR]) {
    int i;
    for (i = 0; i < MAXR; i++) {
        x[i] = (double)i + 1;
    }
    y[0] = 2.07;
    y[1] = 8.6;
    y[2] = 14.42;
    y[3] = 15.8;
    y[4] = 18.92;
    y[5] = 17.96;
    y[6] = 12.98;
    y[7] = 6.45;
    y[8] = 0.27;
}

double deter(double a[MAXC][MAXC]) {
    return(a[0][0] * (a[1][1] * a[2][2] - a[2][1] * a[1][2])
        - a[0][1] * (a[1][0] * a[2][2] - a[2][0] * a[1][2])
        + a[0][2] * (a[1][0] * a[2][1] - a[2][0] * a[1][1]));
}

void setup(double a[MAXC][MAXC], double b[MAXC][MAXC], double
y[MAXC], double coef[MAXC], int j, double det) {
    int i;
    for (i = 0; i < MAXC; i++) {
        b[i][j] = y[i];
        if (j > 0) {
            b[i][j - 1] = a[i][j - 1];
        }
    }
    coef[j] = deter(b) / det;
}

void solve(double a[MAXC][MAXC], double y[MAXC], double
coef[MAXC]) {
    double b[MAXC][MAXC];
    int i, j;
    double det;

    for (i = 0; i < MAXC; i++) {
        for (j = 0; j < MAXC; j++) {
            b[i][j] = a[i][j];
        }
    }
}
```

```

    }

    det = deter(b);
    if (det == 0) {
        return;
    }
    else {
        setup(a, b, y, coef, 0, det);
        setup(a, b, y, coef, 1, det);
        setup(a, b, y, coef, 2, det);
    }
}

void linfit(double x[MAXR], double y[MAXR], double y_calc[MAXR],
double coef[MAXC], double* corel_coef) {
    double sum_x, sum_y, sum_xy, sum_x2, sum_y2;
    double xi, yi, x2, sum_x3, sum_x4, sum_2y, srs;
    int i;
    double a[MAXC][MAXC];
    double g[MAXC];

    sum_x = 0.0;
    sum_y = 0.0;
    sum_xy = 0.0;
    sum_x2 = 0.0;
    sum_y2 = 0.0;
    sum_x3 = 0.0;
    sum_x4 = 0.0;
    sum_2y = 0.0;

    for (i = 0; i < MAXR; i++) {
        xi = x[i];
        yi = y[i];
        x2 = xi * xi;
        sum_x = sum_x + xi;
        sum_y = sum_y + yi;
        sum_xy = sum_xy + xi * yi;
        sum_x2 = sum_x2 + x2;
        sum_y2 = sum_y2 + yi * yi;
        sum_x3 = sum_x3 + xi * x2;
        sum_x4 = sum_x4 + x2 * x2;
        sum_2y = sum_2y + x2 * yi;
    }

    a[0][0] = MAXR;
    a[1][0] = sum_x;
    a[0][1] = sum_x;
    a[2][0] = sum_x2;
    a[0][2] = sum_x2;
    a[1][1] = sum_x2;
    a[2][1] = sum_x3;
    a[1][2] = sum_x3;
    a[2][2] = sum_x4;

```

```

    g[0] = sum_y;
    g[1] = sum_xy;
    g[2] = sum_2y;

    solve(a, g, coef);
    srs = 0.0;

    for (i = 0; i < MAXR; i++) {
        y_calc[i] = coef[0] + coef[1] * x[i] + coef[2] * pow(x[i],
2);
        srs += pow(y[i] - y_calc[i], 2);
    }
    *corel_coef = sqrt(1.0 - srs / (sum_y2 - pow(sum_y, 2) /
MAXR));
}

int main() {
    double x[MAXR];
    double y[MAXR];
    double y_calc[MAXR];
    double coef[MAXC];
    double corel_coef;

    get_data(x, y);
    linfit(x, y, y_calc, coef, &corel_coef);

    return 0;
}

```

ПРИЛОЖЕНИЕ Б.

КОД ПРОГРАММЫ С РАЗДЕЛЕНИЕМ НА ФУ

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4 #include "sampler.h"
5
6 #define MAXR 9
7 #define MAXC 3
8
9 void get_data(double x[MAXR], double y[MAXR]) {
10     int i;
11     for (i = 0; i < MAXR; i++) {
12         x[i] = (double)i + 1;
13     }
14     y[0] = 2.07;
15     y[1] = 8.6;
16     y[2] = 14.42;
17     y[3] = 15.8;
18     y[4] = 18.92;
19     y[5] = 17.96;
20     y[6] = 12.98;
21     y[7] = 6.45;
22     y[8] = 0.27;
23 }
24
25 double deter(double a[MAXC][MAXC]) {
26     return(a[0][0] * (a[1][1] * a[2][2] - a[2][1] * a[1][2])
27         - a[0][1] * (a[1][0] * a[2][2] - a[2][0] * a[1][2])
28         + a[0][2] * (a[1][0] * a[2][1] - a[2][0] * a[1][1]));
29 }
30
31 void setup(double a[MAXC][MAXC], double b[MAXC][MAXC], double
y[MAXC], double coef[MAXC], int j, double det) {
32     SAMPLE;
33     int i;
34     SAMPLE;
35     for (i = 0; SAMPLE, i < MAXC; i++) {
36         SAMPLE;
37         b[i][j] = y[i];
38         SAMPLE;
39         if (j > 0) {
40             SAMPLE;
41             b[i][j - 1] = a[i][j - 1];
42             SAMPLE;
43         }
44         SAMPLE;
45     }
46     SAMPLE;
47     coef[j] = deter(b) / det;
48     SAMPLE;
```

```

49 }
50
51 void solve(double a[MAXC][MAXC], double y[MAXC], double
coef[MAXC]) {
52     SAMPLE;
53     double b[MAXC][MAXC];
54     int i, j;
55     double det;
56
57     SAMPLE;
58     for (i = 0; SAMPLE, i < MAXC; i++) {
59         SAMPLE;
60         for (j = 0; SAMPLE, j < MAXC; j++) {
61             SAMPLE;
62             b[i][j] = a[i][j];
63             SAMPLE;
64         }
65         SAMPLE;
66     }
67     SAMPLE;
68
69     det = deter(b);
70     SAMPLE;
71     if (det == 0) {
72         SAMPLE;
73         return;
74     }
75     else {
76         SAMPLE;
77         setup(a, b, y, coef, 0, det);
78         setup(a, b, y, coef, 1, det);
79         setup(a, b, y, coef, 2, det);
80         SAMPLE;
81     }
82     SAMPLE;
83 }
84
85 void linfit(double x[MAXR], double y[MAXR], double
y_calc[MAXR], double coef[MAXC], double* corel_coef) {
86     SAMPLE;
87     double sum_x, sum_y, sum_xy, sum_x2, sum_y2;
88     double xi, yi, x2, sum_x3, sum_x4, sum_2y, srs;
89     int i;
90     double a[MAXC][MAXC];
91     double g[MAXC];
92
93     sum_x = 0.0;
94     sum_y = 0.0;
95     sum_xy = 0.0;
96     sum_x2 = 0.0;
97     sum_y2 = 0.0;
98     sum_x3 = 0.0;
99     sum_x4 = 0.0;

```

```

100     sum_2y = 0.0;
101
102     SAMPLE;
103     for (i = 0; SAMPLE, i < MAXR; i++) {
104         SAMPLE;
105         xi = x[i];
106         yi = y[i];
107         x2 = xi * xi;
108         sum_x = sum_x + xi;
109         sum_y = sum_y + yi;
110         sum_xy = sum_xy + xi * yi;
111         sum_x2 = sum_x2 + x2;
112         sum_y2 = sum_y2 + yi * yi;
113         sum_x3 = sum_x3 + xi * x2;
114         sum_x4 = sum_x4 + x2 * x2;
115         sum_2y = sum_2y + x2 * yi;
116         SAMPLE;
117     }
118     SAMPLE;
119
120     a[0][0] = MAXR;
121     a[1][0] = sum_x;
122     a[0][1] = sum_x;
123     a[2][0] = sum_x2;
124     a[0][2] = sum_x2;
125     a[1][1] = sum_x2;
126     a[2][1] = sum_x3;
127     a[1][2] = sum_x3;
128     a[2][2] = sum_x4;
129     g[0] = sum_y;
130     g[1] = sum_xy;
131     g[2] = sum_2y;
132     srs = 0.0;
133
134     SAMPLE;
135     solve(a, g, coef);
136     SAMPLE;
137     for (i = 0; SAMPLE, i < MAXR; i++) {
138         SAMPLE;
139         y_calc[i] = coef[0] + coef[1] * x[i] + coef[2] *
pow(x[i], 2);
140         srs += pow(y[i] - y_calc[i], 2);
141         SAMPLE;
142     }
143     SAMPLE;
144     *corel_coef = sqrt(1.0 - srs / (sum_y2 - pow(sum_y, 2)) /
MAXR);
145     SAMPLE;
146 }
147
148 int main(int argc, char **argv)
149 {
150     sampler_init(&argc, argv);

```

```
151     double x[MAXR];
152     double y[MAXR];
153     double y_calc[MAXR];
154     double coef[MAXC];
155     double corel_coef;
156
157     get_data(x, y);
158     SAMPLE;
159     linfit(x, y, y_calc, coef, &corel_coef);
160     SAMPLE;
161     return 0;
162 }
```

ПРИЛОЖЕНИЕ В.

ОПИСАНИЕ МОДЕЛИ .XML

```
<model type = "Objects::AMC::Model" name = "model">
  <node type = "Objects::AMC::Top" name = "t1"></node>
  <node type = "Objects::AMC::Top" name = "t2"></node>
  <node type = "Objects::AMC::Top" name = "t3"></node>
  <node type = "Objects::AMC::Top" name = "t4"></node>
  <node type = "Objects::AMC::Top" name = "t5"></node>
  <node type = "Objects::AMC::Top" name = "t6"></node>
  <node type = "Objects::AMC::Top" name = "t7"></node>
  <node type = "Objects::AMC::Top" name = "t8"></node>
  <node type = "Objects::AMC::Top" name = "t9"></node>
  <node type = "Objects::AMC::Top" name = "t10"></node>
  <node type = "Objects::AMC::Top" name = "t11"></node>
  <node type = "Objects::AMC::Top" name = "t12"></node>
  <node type = "Objects::AMC::Top" name = "t13"></node>
  <node type = "Objects::AMC::Top" name = "t14"></node>
  <node type = "Objects::AMC::Top" name = "t15"></node>
  <link type = "Objects::AMC::Link" name = "t1-->t2" probability
= "1.0" intensity = "65.802" deviation = "0.0" source = "t1" dest =
"t2"></link>
  <link type = "Objects::AMC::Link" name = "t2-->t3" probability
= "0.9" intensity = "17.121" deviation = "0.0" source = "t2" dest =
"t3"></link>
  <link type = "Objects::AMC::Link" name = "t3-->t2" probability
= "1.0" intensity = "1.764" deviation = "0.0" source = "t3" dest =
"t2"></link>
  <link type = "Objects::AMC::Link" name = "t2-->t4" probability
= "0.1" intensity = "53.489" deviation = "0.0" source = "t2" dest =
"t4"></link>
  <link type = "Objects::AMC::Link" name = "t4-->t5" probability
= "0.75" intensity = "3.171" deviation = "0.0" source = "t4" dest =
"t5"></link>
  <link type = "Objects::AMC::Link" name = "t4-->t7" probability
= "0.25" intensity = "41.953" deviation = "0.0" source = "t4" dest =
"t7"></link>
  <link type = "Objects::AMC::Link" name = "t5-->t4" probability
= "0.25" intensity = "5.364" deviation = "0.0" source = "t5" dest =
"t4"></link>
  <link type = "Objects::AMC::Link" name = "t5-->t6" probability
= "0.75" intensity = "7.571" deviation = "0.0" source = "t5" dest =
"t6"></link>
  <link type = "Objects::AMC::Link" name = "t6-->t5" probability
= "1.0" intensity = "3.108" deviation = "0.0" source = "t6" dest =
"t5"></link>
  <link type = "Objects::AMC::Link" name = "t7-->t8" probability
= "1.0" intensity = "52.08" deviation = "0.0" source = "t7" dest =
"t8"></link>
```



```

    <link type = "Objects::AMC::Link" name = "t7-->t13"
probability = "0.0" intensity = "11.818" deviation = "0.0" source = "t7"
dest = "t13"></link>
    <link type = "Objects::AMC::Link" name = "t8-->t9" probability
= "0.75" intensity = "7.306" deviation = "0.0" source = "t8" dest =
"t9"></link>
    <link type = "Objects::AMC::Link" name = "t8-->t12"
probability = "0.25" intensity = "32.191" deviation = "0.0" source =
"t8" dest = "t12"></link>
    <link type = "Objects::AMC::Link" name = "t9-->t10"
probability = "0.67" intensity = "8.342" deviation = "0.0" source = "t9"
dest = "t10"></link>
    <link type = "Objects::AMC::Link" name = "t9-->t11"
probability = "0.33" intensity = "3.631" deviation = "0.0" source = "t9"
dest = "t11"></link>
    <link type = "Objects::AMC::Link" name = "t10-->t11"
probability = "1.0" intensity = "1.275" deviation = "0.0" source = "t10"
dest = "t11"></link>
    <link type = "Objects::AMC::Link" name = "t11-->t8"
probability = "1.0" intensity = "1.248" deviation = "0.0" source = "t11"
dest = "t8"></link>
    <link type = "Objects::AMC::Link" name = "t12-->t8"
probability = "0.67" intensity = "21.424" deviation = "0.0" source =
"t12" dest = "t8"></link>
    <link type = "Objects::AMC::Link" name = "t12-->t13"
probability = "0.33" intensity = "55.397" deviation = "0.0" source =
"t12" dest = "t13"></link>
    <link type = "Objects::AMC::Link" name = "t13-->t14"
probability = "0.9" intensity = "717.748" deviation = "0.0" source =
"t13" dest = "t14"></link>
    <link type = "Objects::AMC::Link" name = "t13-->t15"
probability = "0.1" intensity = "209.364" deviation = "0.0" source =
"t13" dest = "t15"></link>
    <link type = "Objects::AMC::Link" name = "t14-->t13"
probability = "1.0" intensity = "2.651" deviation = "0.0" source = "t14"
dest = "t13"></link>
</model>

```