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. 7304 _____ Петруненко Д.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Изучение возможности построения операционной графовой модели программы (ОГМП) и расчета характеристик эффективности ее выполнения методом эквивалентных преобразований.

Постановка задачи.

1. Построение ОГМП.

Для рассмотренного в лабораторных работах 1-3 индивидуального задания 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, [0,100.00] - для положительных чисел или [-100.00, 100.00] - для произвольных чисел). В случае ветвлений, вызванных проверкой выхода из цикла, вероятности рассчитываются исходя из 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, полученные с помощью монитора Sampler в процессе выполнения работы №3. Если требуется, оценить с помощью монитора Sampler времена выполнения неучтенных ранее участков программы.

2. Расчет характеристик эффективности выполнения программы методом эквивалентных преобразований.

Полученную в части 1 данной работы ОГМП, представить в виде графа с нагруженными дугами, у которого в качестве параметров, характеризующих потребление ресурсов на дуге ij, использовать тройку {Pij, Mij, Dij}, где:

Pij - вероятность выполнения процесса для дуги ij,

Mij - мат. ожидание потребления ресурса процессом для дуги ij,

Dij - 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.

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

Ход выполнения.

1. Построен граф управления для основной части программы (функция romb). Код программы представлен в приложении А. Граф представлен на Рисунке 1:

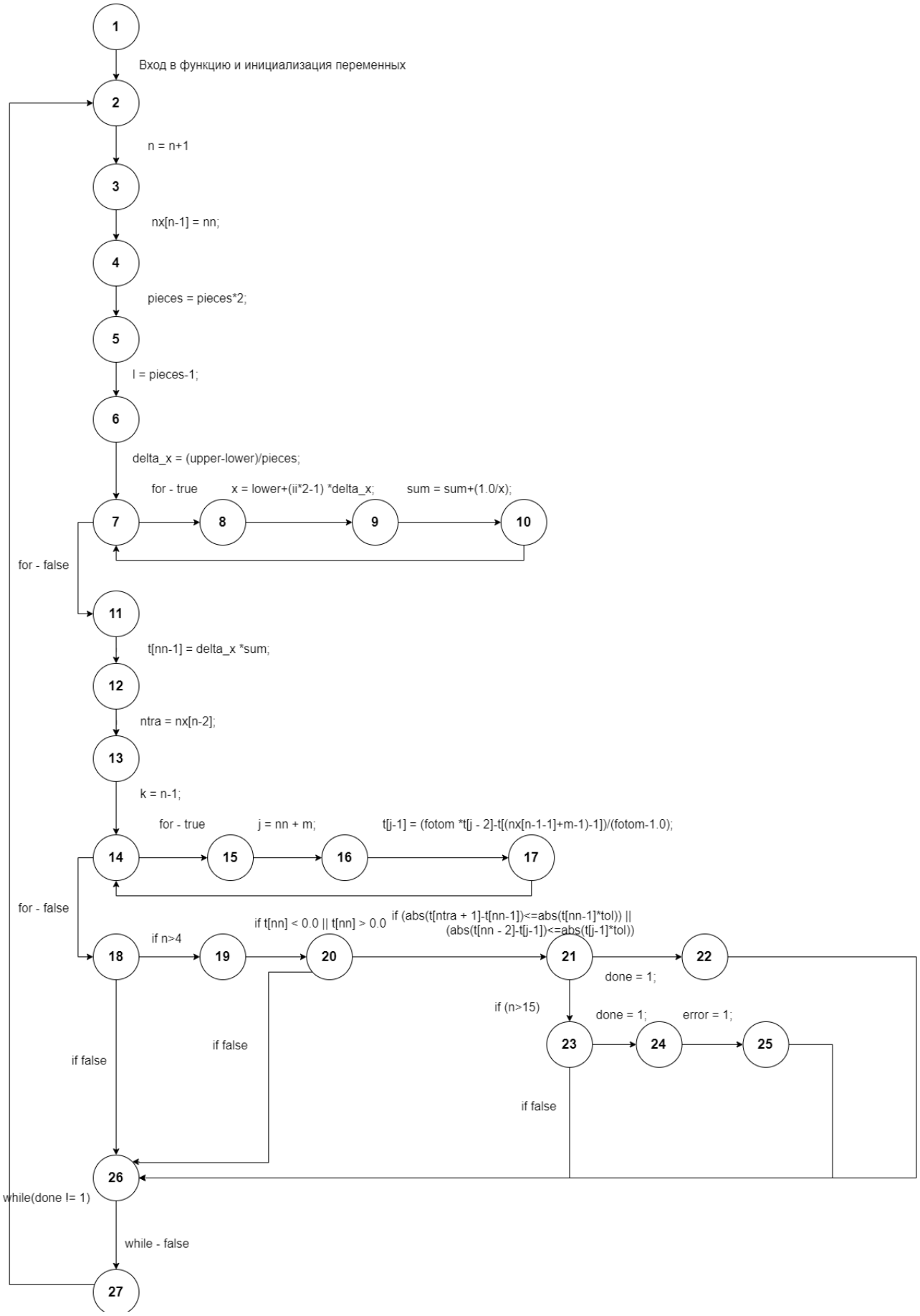

Рисунок 1: Управляющий граф функции romb

2. Составлен текст программы для профилирования с использованием профилировщика SAMPLER (на основе кода из приложения А). Текст программы для профилирования представлен в Приложении Б, результаты профилирования функции romb представлены на Рисунке 2:

```
    Отчет о результатах измерений для программы ROMBC.EXE.

    Создан программой Sampler ( версия от Feb 15 1999 )
        1995-98 (c) СПбГЭТУ, Мойсейчук Леонид.



        Список обработанных файлов.

-------------------------------------------------------------------------
   NN            Имя обработанного файла
-------------------------------------------------------------------------
   1.   ROMBC.CPP
-------------------------------------------------------------------------


   Таблица с результатами измерений ( используется 22 из 416 записей )

-------------------------------------------------------------------------
Исх.Поз. Прием.Поз.  Общее время(мкс)  Кол-во прох. Среднее время(мкс)
-------------------------------------------------------------------------
 1 :  10  1 :  27             550.63            1             550.63
-------------------------------------------------------------------------
 1 :  27  1 :  29               3.35            4               0.84
-------------------------------------------------------------------------
 1 :  29  1 :  31               5.87            4               1.47
-------------------------------------------------------------------------
 1 :  31  1 :  33               4.19            4               1.05
-------------------------------------------------------------------------
 1 :  33  1 :  35               4.19            4               1.05
-------------------------------------------------------------------------
 1 :  35  1 :  37             118.17            4              29.54
-------------------------------------------------------------------------
 1 :  37  1 :  39              11.73            4               2.93
-------------------------------------------------------------------------
 1 :  39  1 :  41             155.05           15              10.34
-------------------------------------------------------------------------
 1 :  41  1 :  43             137.45           15               9.16
-------------------------------------------------------------------------
 1 :  43  1 :  45              10.90            4               2.72
 1 :  43  1 :  39              29.33           11               2.67
-------------------------------------------------------------------------
```

```
1 :  45  1 :  47              83.81              4              20.95
-------------------------------------------------------------------------
1 :  47  1 :  50            1483.43              4             370.86
-------------------------------------------------------------------------
1 :  50  1 :  52               3.35              4               0.84
-------------------------------------------------------------------------
1 :  52  1 :  54               7.54              4               1.89
-------------------------------------------------------------------------
1 :  54  1 :  56              10.90             10               1.09
-------------------------------------------------------------------------
1 :  56  1 :  58             276.57             10              27.66
-------------------------------------------------------------------------
1 :  58  1 :  61            1979.58              4             494.90
1 :  58  1 :  54              10.90              6               1.82
-------------------------------------------------------------------------
1 :  61  1 :  76               5.03              3               1.68
1 :  61  1 :  63               0.00              1               0.00
-------------------------------------------------------------------------
1 :  63  1 :  65             177.68              1             177.68
-------------------------------------------------------------------------
1 :  65  1 :  76             181.03              1             181.03
-------------------------------------------------------------------------
1 :  76  1 :  27               2.51              3               0.84
1 :  76  1 :  79              53.64              1              53.64
-------------------------------------------------------------------------
```

Рисунок 2: Результаты профилирования функции romb

Суммарное время работы T = 5409,91 мкс.

3. На основании полученных с помощью профилировщика SAMPLER данных о работе функции romb был проведён расчёт вероятностей и затрат ресурсов для дуг управляющего графа функции romb. Результаты расчётов приведены в Таблице 1:

Дуга	**Номера строк**	**Количество проходов**	**Расчет вероятности**	**Затраты ресурсов (Среднее время), мкс**
L1 – L2	10 : 27	1	1	550.63
L2 – L3	27 : 29	4	1	0.84
L3 – L4	29 : 31	4	1	1.47
L4 – L5	31 : 33	4	1	1.05
L5 – L6	33 : 35	4	1	1.05

L6 – L7	35 : 37	4	1	29.54
L7 – L8	37 : 39	4	15/(15+4) = 0,79	2.93
L8 – L9	39 : 41	15	1	10.34
L9 – L10	41 : 43	15	1	9.16
L10 – L7	43 : 39	11	1	2.67
L7 – L11	43 : 45	4	1 – 0,79 = 0,21	2.72
L11 – L12	45 : 47	4	1	20.95
L12 – L13	47 : 50	4	1	370.86
L13– L14	50 : 52	4	1	0.84
L14 – L15	52 : 54	4	10/(10+4) = 0,71	1.89
L15 – L16	54 : 56	10	1	1.09
L16 – L17	56 : 58	10	1	27.66
L17 – L14	58 : 54	6	1	1.82
L14 – L18	58 : 61	4	1 – 0,71 = 0,29	494.90
L18– L20	61 : 76	3	3 /(3+1) = 0,75	1.68
L18– L19	61 : 65	1	1 /(3+1) = 0,25	0.00
L19– L20	65 : 76	1	1	181.03
L20– L2	76 : 27	3	3 /(3+1) = 0,75	0.84
L20– L21	76 : 79	1	1 /(3+1) = 0,25	53.64

Таблица 1: Расчёт вероятностей и затрат ресурсов

4. На основании полученных расчётов вероятностей и затрат ресурсов был построен операционная графовая модель программы, представленная на Рисунке 3:

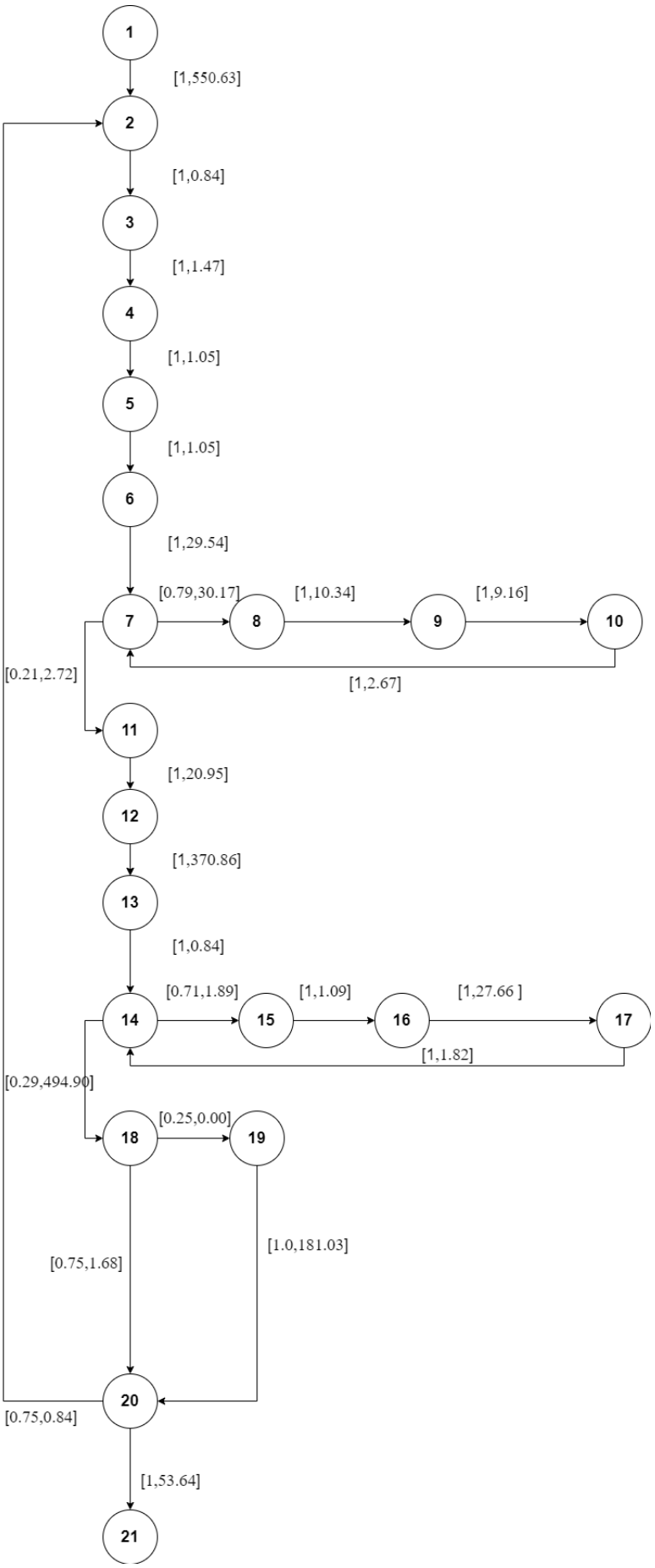

Рисунок 3: Операционная графовая модель программы

5. По основе полученной ОГМП был создан XML-файл модели программы, приведённый в Приложении В, для расчёта характеристик эффективности выполнения программы методом эквивалентных преобразований с помощью пакета CSA III. Графическое отображение модели представлено на Рисунке 4:

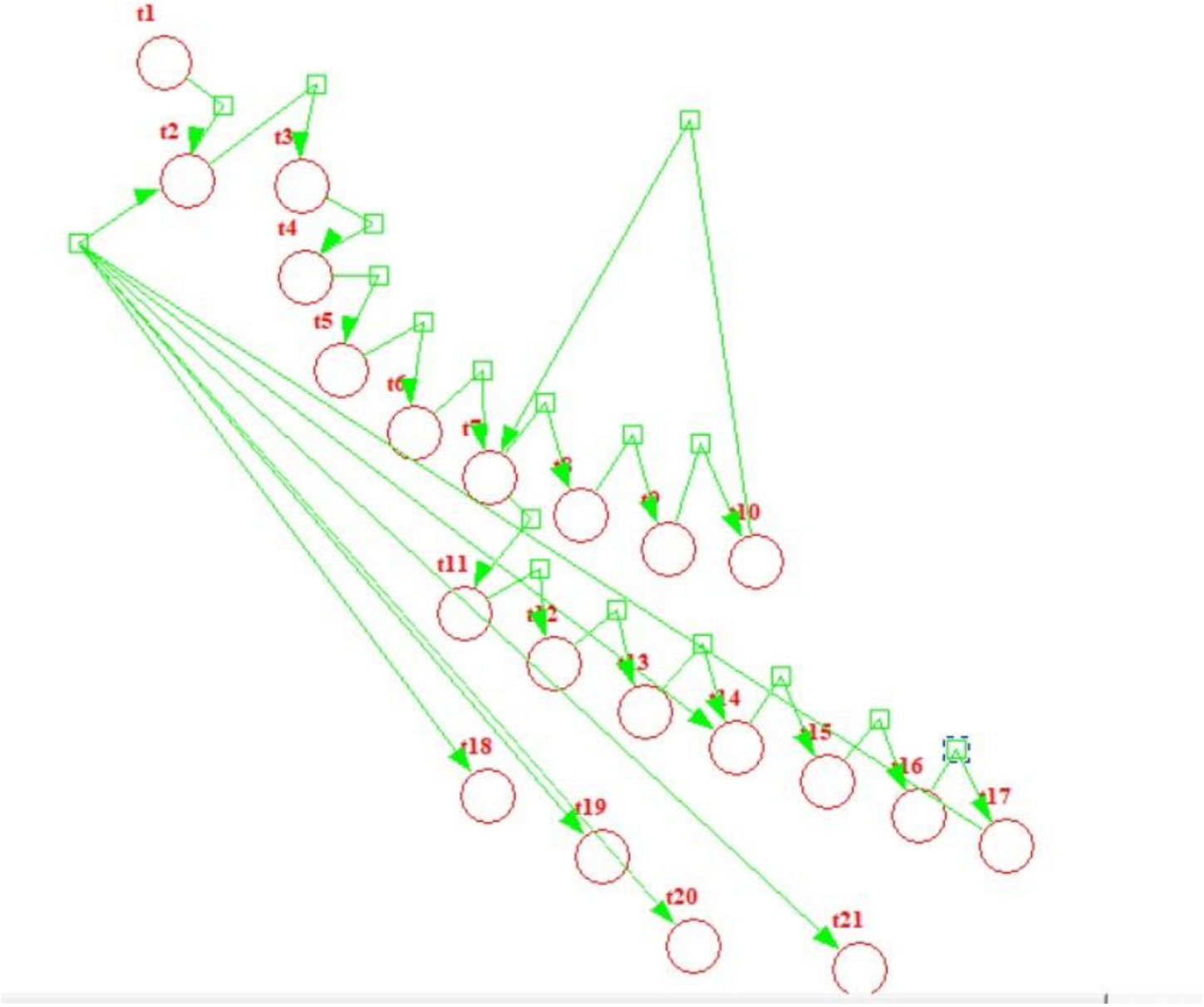

Рисунок 4: Модель программы в CSA III

1. С помощью пакета CSA III были вычислены математическое ожидание и дисперсия времени выполнения основной части программы, а именно функции romb. Результаты вычислений представлены на Рисунке 5:

Name	Value
name	t1-->t21
probability	1
intensity	5389.54
deviation	10332454.34

Рисунок 5: Результаты вычислений

Выводы.

В ходе выполнения лабораторной работы была построена операционная графовая модель заданной программы, нагрузочные параметры которой были оценены с помощью профилировщика SAMPLER и методом эквивалентных преобразований с помощью пакета CSAIII были вычислены математическое ожидание и дисперсия времени выполнения основной части программы, а именно функции romb.

Результаты полученных характеристик с помощью пакета CSAIII не сильно отличаются от полученного с помощью SAMPLER, что говорит о адекватности построенной модели.

Приложение А: Исходный код программы.

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

double tol = 0.0001;
int done = 0;
double sumMain,upper,lower;

void romb(double lower,double upper){
    int p;
    int nx[16] = {0};
    double t[136]= {0};
    int done,error;
    int pieces,ii,n,nn,l,ntra,k,m,j;
    double delta_x,c,sum,fotom,x;
    pieces = 1;
    nx[0] = 1;
    delta_x = (upper-lower)/pieces;
    c = ((1.0/lower)+(1.0/upper))*0.5;
    t[0] = delta_x*c;
    n = 1;
    nn = 2;
    sum = c;
    do {
        n = n+1;
        fotom = 4.0;
        nx[n-1] =  nn;
        pieces = pieces*2;
        l = pieces-1;
        delta_x = (upper-lower)/pieces;
        for (ii =1; ii<=((l+1)/2); ii++) {
            x = lower+(ii*2-1) *delta_x;
            sum = sum+(1.0/x);
        }
        t[nn-1] = delta_x *sum;
        printf("%d%f ", pieces,t[nn]);
        ntra = nx[n-2];
        k = n-1;
        for (m = 1;m<=k;m++) {
            j = nn + m;
            t[j-1]   =   (fotom   *t[j   -   2]-t[(nx[n-1-1]+m-1)-
1])/(fotom-1.0);
        }
        printf("%d%f\n",j-1,t[j-1]);
        if (n>4) {
            if (t[nn] < 0.0 || t[nn] > 0.0) {
                if  ((abs(t[ntra  +  1]-t[nn-1])<=abs(t[nn-1]*tol))
||
                (abs(t[nn - 2]-t[j-1])<=abs(t[j-1]*tol))) {
                    done = 1;
                } else if (n>15) {
                    done = 1;
```

```
                    error = 1;
                }
            }
        }
        nn = j+1;
    }while(done != 1);
    sumMain = t[j-1];
}

int main() {
    lower = 1.0;
    upper = 9.0;
    printf("\n");
    romb(lower,upper);
    printf("\n");
    printf("Area= %f",sumMain);
    return 0;
}
```

Приложение Б: Код программы для профилирования.

```
#include <stdio.h>
#include <math.h>
#include "sampler.h"

double tol = 0.0001;
int done = 0;
double sumMain,upper,lower;

void romb(double lower,double upper){
     SAMPLE ;
    int p;
    int nx[16] = {0};
    double t[136]= {0};
    int done,error;
    int pieces,ii,n,nn,l,ntra,k,m,j;
    double delta_x,c,sum,fotom,x;
    pieces = 1;
    nx[0] = 1;
    delta_x = (upper-lower)/pieces;
    c = ((1.0/lower)+(1.0/upper))*0.5;
    t[0] = delta_x*c;
    n = 1;
    nn = 2;
    sum = c;
    fotom = 4.0;
    do {
        SAMPLE ;
        n = n+1;
          SAMPLE ;
        nx[n-1] =  nn;
          SAMPLE ;
        pieces = pieces*2;
          SAMPLE ;
        l = pieces-1;
          SAMPLE ;
        delta_x = (upper-lower)/pieces;
        SAMPLE ;
        for (ii =1; ii<=((l+1)/2); ii++) {
               SAMPLE ;
            x = lower+(ii*2-1) *delta_x;
               SAMPLE ;
            sum = sum+(1.0/x);
               SAMPLE ;
        }
          SAMPLE ;
        t[nn-1] = delta_x *sum;
          SAMPLE ;
        printf("%d%f ", pieces,t[nn]);
        ntra = nx[n-2];
          SAMPLE ;
```

```
        k = n-1;
        SAMPLE ;
        for (m = 1;m<=k;m++) {
               SAMPLE ;
            j = nn + m;
               SAMPLE ;
            t[j-1]   =   (fotom   *t[j   -   2]-t[(nx[n-1-1]+m-1)-
1])/(fotom-1.0);
               SAMPLE ;
        }
        printf("%d%f\n",j-1,t[j-1]);
        SAMPLE ;
        if (n>4) {
               SAMPLE ;
            if (t[nn] < 0.0 || t[nn] > 0.0) {
                   SAMPLE ;
                if  ((abs(t[ntra  +  1]-t[nn-1])<=abs(t[nn-1]*tol))
||
                (abs(t[nn - 2]-t[j-1])<=abs(t[j-1]*tol))) {
                    done = 1;
                } else if (n>15) {
                    done = 1;
                    error = 1;
                }
            }
        }
        nn = j+1;
        SAMPLE ;
    }while(done != 1);
    sumMain = t[j-1];
    SAMPLE ;
}

int main() {
    lower = 1.0;
    upper = 9.0;
    printf("\n");
    romb(lower,upper);
    printf("\n");
    printf("Area= %f",sumMain);
    return 0;
}
```

Приложение B: XML-файл модели программы.

```
<model type="Objects::AMC::Model" name="lab4">
     <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>
     <node type="Objects::AMC::Top" name="t16"></node>
     <node type="Objects::AMC::Top" name="t17"></node>
     <node type="Objects::AMC::Top" name="t18"></node>
     <node type="Objects::AMC::Top" name="t19"></node>
     <node type="Objects::AMC::Top" name="t20"></node>
     <node type="Objects::AMC::Top" name="t21"></node>
     <link  type="Objects::AMC::Link"  name="t1-t2"  probability="1"
intensity="550.63" deviation="0.0" source="t1" dest="t2"></link>
     <link  type="Objects::AMC::Link"  name="t2-t3"  probability="1"
intensity="0.84" deviation="0.0" source="t2" dest="t3"></link>
     <link  type="Objects::AMC::Link"  name="t3-t4"  probability="1"
intensity="1.47" deviation="0.0" source="t3" dest="t4"></link>
     <link  type="Objects::AMC::Link"  name="t4-t5"  probability="1"
intensity="1.05" deviation="0.0" source="t4" dest="t5"></link>
     <link  type="Objects::AMC::Link"  name="t5-t6"  probability="1"
intensity="1.05" deviation="0.0" source="t5" dest="t6"></link>
     <link  type="Objects::AMC::Link"  name="t6-t7"  probability="1"
intensity="29.54" deviation="0.0" source="t6" dest="t7"></link>
     <link          type="Objects::AMC::Link"          name="t7-t8"
probability="0.79"  intensity="2.93"  deviation="0.0"  source="t7"
dest="t8"></link>
     <link  type="Objects::AMC::Link"  name="t8-t9"  probability="1"
intensity="10.34" deviation="0.0" source="t8" dest="t9"></link>
     <link type="Objects::AMC::Link" name="t9-t10" probability="1"
intensity="10.34" deviation="0.0" source="t9" dest="t10"></link>
     <link type="Objects::AMC::Link" name="t10-t7" probability="1"
intensity="2.67" deviation="0.0" source="t10" dest="t7"></link>
     <link         type="Objects::AMC::Link"         name="t7-t11"
probability="0.21"  intensity="2.72"  deviation="0.0"  source="t7"
dest="t11"></link>
     <link         type="Objects::AMC::Link"         name="t11-t12"
probability="1"  intensity="20.95"  deviation="0.0"  source="t11"
dest="t12"></link>
```

```
     <link         type="Objects::AMC::Link"         name="t12-t13"
probability="1"  intensity="370.86"  deviation="0.0"  source="t12"
dest="t13"></link>
     <link         type="Objects::AMC::Link"         name="t13-t14"
probability="1"   intensity="0.84"   deviation="0.0"   source="t13"
dest="t14"></link>
     <link         type="Objects::AMC::Link"         name="t14-t15"
probability="0.71"  intensity="1.89"  deviation="0.0"  source="t14"
dest="t15"></link>
     <link         type="Objects::AMC::Link"         name="t15-t16"
probability="1"   intensity="1.09"   deviation="0.0"   source="t15"
dest="t16"></link>
     <link         type="Objects::AMC::Link"         name="t16-t17"
probability="1"   intensity="27.66"   deviation="0.0"   source="t16"
dest="t17"></link>
     <link         type="Objects::AMC::Link"         name="t17-t14"
probability="1"   intensity="1.82"   deviation="0.0"   source="t17"
dest="t14"></link>
     <link         type="Objects::AMC::Link"         name="t14-t18"
probability="0.29" intensity="494.90" deviation="0.0" source="t14"
dest="t18"></link>
     <link         type="Objects::AMC::Link"         name="t18-t20"
probability="0.75"  intensity="1.68"  deviation="0.0"  source="t18"
dest="t20"></link>
     <link         type="Objects::AMC::Link"         name="t18-t19"
probability="0.25"  intensity="0.00"  deviation="0.0"  source="t18"
dest="t19"></link>
     <link         type="Objects::AMC::Link"         name="t19-t20"
probability="1"  intensity="181.03"  deviation="0.0"  source="t19"
dest="t20"></link>
     <link          type="Objects::AMC::Link"          name="t20-t2"
probability="0.75"  intensity="0.84"  deviation="0.0"  source="t20"
dest="t2"></link>
     <link         type="Objects::AMC::Link"         name="t20-t21"
probability="0.25"  intensity="53.64"  deviation="0.0"  source="t20"
dest="t21"></link>
</model>
```