МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МОЭВМ

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. 6304 _____ Цыганов М.А.

Преподаватель _____ Кирьянчиков В.А.

Санкт-Петербург

2020

Цель работы:

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

Формулировка задания:

1.1. Построение ОГМП.

Для рассматривавшегося 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] - 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.

1.2. Расчет характеристик эффективности выполнения программы методом эквивалентных преобразований.

Полученную в части 2.1 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.

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.

Построение операционной графовой модели программы.

Исходный текст программы:

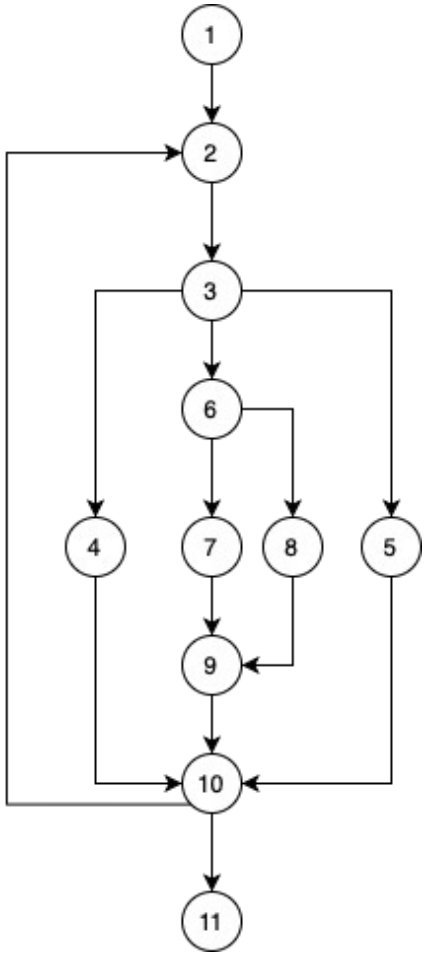
File: lab1_2.cpp

```
01: #include "sampler.h"
02: #include <stdio.h>
03: #include <math.h>
04:
05: double erf(double x)
06: {
07:     const double sqrtpi = 1.7724538;
08:     const double t2 = 0.66666667;
09:     const double t3 = 0.66666667;
10:     const double t4 = 0.07619048;
11:     const double t5 = 0.01693122;
12:     const double t6 = 3.078403E-3;
13:     const double t7 = 4.736005E-4;
14:     const double t8 = 6.314673E-5;
15:     const double t9 = 7.429027E-6;
16:     const double t10 = 7.820028E-7;
17:     const double t11 = 7.447646E-8;
18:     const double t12 = 6.476214E-9;
19:
20:     double x2, rez;
21:     int i;
22:     x2 = x * x;
23:      rez = 2.0 * exp(-x2) / sqrtpi * (x * (1 + x2 * (t2 + x2
* (t3 + x2 * (t4 + x2 * (t5 + x2 * (t6 + x2 * (t7 + x2 * (t8 + x2 *
(t9 + x2 * (t10 + x2 * (t11 + x2 * t12)))))))))));
24:     return rez;
25: }
26:
27: double erfc(double x)
28: {
29:     const double sqrtpi = 1.7724538;
30:     double rez, v;
31:     v = 1.0 / (2.0 * x*x);
32:      rez = 1.0 / (exp(x*x) * x * sqrtpi * (1 + v / (1 + 2 * v
/ (1 + 3 * v / (1 + 4 * v / (1 + 5 * v / (1 + 6 * v / (1 + 7 * v /
(1 + 8 * v / (1 + 9 * v / (1 + 10 * v / (1 + 11 * v / (1 + 12 *
v)))))))))));
33:     return rez;
34: }
35:
36: int main()
37: {
38:
39: #1  double x, er, ec;
40:     int done;
41:     done = 1;
42:     x = 3.0;
43:        do {
44: #2      x -= 1;
45: #3      if (x < 0.0){
46: #4          done = 0;
47:         }
48: #3       else if (x == 0.0)
49:         {
```

```
50: #5           er = 0.0;
51:              ec = 1.0;
52:          }
53: #3       else
54:          {
55: #6           if (x < 1.5)
56:              {
57: #7               er = erf(x);
58:                  ec = 1.0 - er;
59:              }
60: #6           else
61:              {
62: #8               ec = erfc(x);
63:                  er = 1.0 - ec;
64:              }
65: #9       }
66: #10    } while (done);
67:
68: #10  return 0;
69: }
```

Для данной программы была разработана операционная модель управляющего графа на основе схемы алгоритма.

Операционная модель управляющего графа программы.



Профилирование.

Текст программы (подготовленный для профилирования).

```
   File: lab1_3.cpp
   01: #include "sampler.h"
   02: #include <stdio.h>
   03: #include <math.h>
   04:
   05: double erf(double x)
   06: {
   07:     SAMPLE;
   08:     const double sqrtpi = 1.7724538;
   09:     const double t2 = 0.66666667;
   10:     const double t3 = 0.66666667;
   11:     const double t4 = 0.07619048;
   12:     const double t5 = 0.01693122;
   13:     const double t6 = 3.078403E-3;
   14:     const double t7 = 4.736005E-4;
   15:     const double t8 = 6.314673E-5;
   16:     const double t9 = 7.429027E-6;
   17:     const double t10 = 7.820028E-7;
   18:     const double t11 = 7.447646E-8;
   19:     const double t12 = 6.476214E-9;
   20:
   21:     double x2, rez;
   22:     int i;
   23:     SAMPLE;
   24:     x2 = x * x;
   25:      rez = 2.0 * exp(-x2) / sqrtpi * (x * (1 + x2 * (t2 + x2
* (t3 + x2 * (t4 + x2 * (t5 + x2 * (t6 + x2 * (t7 + x2 * (t8 + x2 *
(t9 + x2 * (t10 + x2 * (t11 + x2 * t12)))))))))));
   26:     SAMPLE;
   27:     return rez;
   28: }
   29:
   30: double erfc(double x)
   31: {
   32:     const double sqrtpi = 1.7724538;
   33:     double rez, v;
   34:     SAMPLE;
   35:     v = 1.0 / (2.0 * x*x);
   36:      rez = 1.0 / (exp(x*x) * x * sqrtpi * (1 + v / (1 + 2 * v
/ (1 + 3 * v / (1 + 4 * v / (1 + 5 * v / (1 + 6 * v / (1 + 7 * v /
(1 + 8 * v / (1 + 9 * v / (1 + 10 * v / (1 + 11 * v / (1 + 12 *
v)))))))))));
   37:     SAMPLE;
   38:     return rez;
   39: }
   40:
   41: int main()
   42: {
   43: #1  double x, er, ec;
   44:     int done;
   45:     SAMPLE;
   46:     done = 1;
   47:     x = 3.0;
   48:     SAMPLE;
   49:     do {
   50:         SAMPLE;
   51: #2      x -= 1;
   52:         SAMPLE;
   53: #3      if (x < 0.0){
   54:             SAMPLE;
```

```
55: #4          done = 0;
56:             SAMPLE;
57:         }
58: #3        else if (x == 0.0)
59:         {
60:             SAMPLE;
61: #5          er = 0.0;
62:             ec = 1.0;
63:             SAMPLE;
64:         }
65: #3      else
66:         {
67:             SAMPLE;
68: #6          if (x < 1.5)
69:             {
70:                 SAMPLE;
71: #7              er = erf(x);
72:                 ec = 1.0 - er;
73:                 SAMPLE;
74:             }
75: #6          else
76:             {
77:                 SAMPLE;
78: #8              ec = erfc(x);
79:                 er = 1.0 - ec;
80:                 SAMPLE;
81:             }
82:            SAMPLE;
83: #9     }SAMPLE;
84: #10  } while (done);
85:     SAMPLE;
86: #11 return 0;
87: }
```

Результаты профилирования.

Файл отчёта:

```
-------------------------------------------------------------------------
  NN           Имя обработанного файла
-------------------------------------------------------------------------
  1.   ..\TEST\LAB1.CPP
-------------------------------------------------------------------------
  Таблица с результатами измерений ( используется 21 из 416 записей )
-------------------------------------------------------------------------
Исх.Поз. Прием.Поз.  Общее время(мкс)  Кол-во прох. Среднее время(мкс)
-------------------------------------------------------------------------
 1 :   7  1 :  23               3.35            1               3.35
-------------------------------------------------------------------------
 1 :  23  1 :  26               5.03            1               5.03
-------------------------------------------------------------------------
 1 :  26  1 :  73               1.68            1               1.68
-------------------------------------------------------------------------
 1 :  34  1 :  37               9.22            1               9.22
-------------------------------------------------------------------------
 1 :  37  1 :  80               0.84            1               0.84
-------------------------------------------------------------------------
 1 :  45  1 :  48               0.84            1               0.84
-------------------------------------------------------------------------
 1 :  48  1 :  50               0.00            1               0.00
-------------------------------------------------------------------------
 1 :  50  1 :  52               2.51            4               0.63
-------------------------------------------------------------------------
 1 :  52  1 :  67               0.84            2               0.42
 1 :  52  1 :  60               0.00            1               0.00
 1 :  52  1 :  54               0.84            1               0.84
-------------------------------------------------------------------------
 1 :  54  1 :  56               0.00            1               0.00
-------------------------------------------------------------------------
 1 :  56  1 :  83               0.84            1               0.84
-------------------------------------------------------------------------
 1 :  60  1 :  63               0.00            1               0.00
-------------------------------------------------------------------------
 1 :  63  1 :  83               0.00            1               0.00
-------------------------------------------------------------------------
 1 :  67  1 :  77               0.84            1               0.84
 1 :  67  1 :  70               0.84            1               0.84
-------------------------------------------------------------------------
 1 :  70  1 :   7               0.84            1               0.84
-------------------------------------------------------------------------
 1 :  73  1 :  82               0.00            1               0.00
-------------------------------------------------------------------------
 1 :  77  1 :  34               1.68            1               1.68
-------------------------------------------------------------------------
 1 :  80  1 :  82               0.00            1               0.00
-------------------------------------------------------------------------
 1 :  82  1 :  83               0.84            2               0.42
-------------------------------------------------------------------------
 1 :  83  1 :  50               0.00            3               0.00
 1 :  83  1 :  85               0.00            1               0.00
-------------------------------------------------------------------------
```

Общее время выполнения программы: 31,03мкс

Расчет вероятностей и затрат ресурсов для дуг управляющего графа.

	Номера строк	Количество проходов
L_{1-2} = 0.84 мкс	45:48 ; 48:50	1;1
L_{2-3} = 0.63 мкс	50:52	4
L_{3-4} = 0.84 мкс	52:54	1
L_{3-5} = 0.00 мкс	52:60	1
L_{3-6} = 0.42 мкс	52:67	2
L_{4-10} = 0.84 мкс	54:56;56:83	1;1
L_{5-10} = 0.00 мкс	60:63;63:83	1;1
L_{6-7} = 0.84 мкс	67:70	1
L_{6-8} = 0.84 мкс	67:77	1
L_{7-9} = 10.90 мкс	70:7; 7:23; 23:26; 26:73	1;1;1;1
L_{8-9} = 11.74 мкс	77:34; 34:37; 37:80	1;1;1
L_{9-10} = 0.42 мкс	82:83	2
L_{10-2} = 0.00 мкс	83:50	3
L_{10-11} = 0.00 мкс	83:85	1

Для цикла do…while:

При проверке условия выполнения цикла значения true и false получены соответственно 3 и 1 раз (N_{true}=3, N_{false}=1), а в совокупности условие проверялось $N = N_{true}+N_{false} = 4$ раз.

Тогда вероятности входа в цикл и выхода из него соответственно равны:

Вероятность входа в цикл : $P = N_{true} / N = 3/4 = 0,75$

Вероятность выхода из цикла : $Q = N_{false} / N = 1/4 = 0,25$

Для if:

При проверке условия значения true и false получены соответственно:

1. If-(#3-#4) 1 и 3 раз (N_{true}=1, N_{false}=3), а в совокупности условие проверялось $N = N_{true}+N_{false} = 4$ раз. Тогда вероятности входа и выхода соответственно равны:

Вероятность входа: $P = N_{true} / N = 1/4 = 0,25$

Вероятность выхода: $Q = N_{false} / N = 3/4 = 0,75$

2. If-(#3-#5) 1 и 3 раз (N_{true}=1, N_{false}=3), а в совокупности условие проверялось N = N_{true}+N_{false} = 4 раз. Тогда вероятности входа и выхода соответственно равны:

Вероятность входа: $P = N_{true} / N = 1/4 = 0,25$

Вероятность выхода: $Q = N_{false} / N = 3/4 = 0,75$

3. If-(#3-#6) 2 и 2 раз (N_{true}=2, N_{false}=2), а в совокупности условие проверялось N = N_{true}+N_{false} = 4 раз. Тогда вероятности входа и выхода соответственно равны:

Вероятность входа: $P = N_{true} / N = 2/4 = 0,5$

Вероятность выхода: $Q = N_{false} / N = 2/4 = 0,5$

4. If-(#6-#7) 1 и 1 раз (N_{true}=1, N_{false}=1), а в совокупности условие проверялось N = N_{true}+N_{false} = 2 раз. Тогда вероятности входа и выхода соответственно равны:

Вероятность входа: $P = N_{true} / N = 1/2 = 0,5$

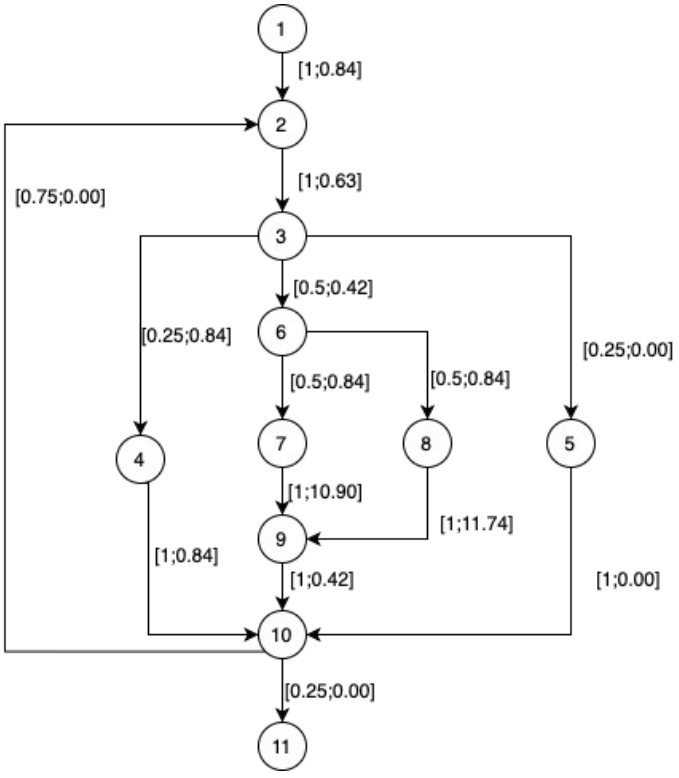
Вероятность выхода: $Q = N_{false} / N = 1/2 = 0,5$

5. If-(#6-#8) 1 и 1 раз (N_{true}=1, N_{false}=1), а в совокупности условие проверялось N = N_{true}+N_{false} = 2 раз. Тогда вероятности входа и выхода соответственно равны:

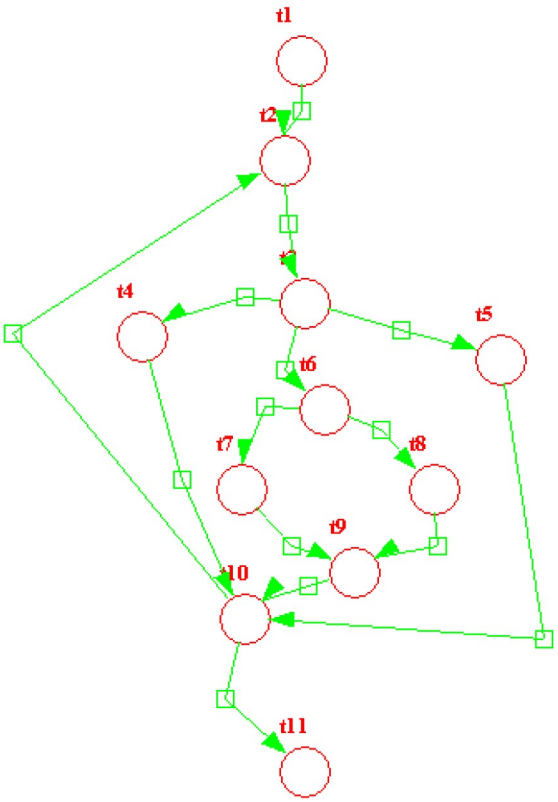
Вероятность входа: $P = N_{true} / N = 1/2 = 0,5$

Вероятность выхода: $Q = N_{false} / N = 1/2 = 0,5$

Операционная графовая модель программы.



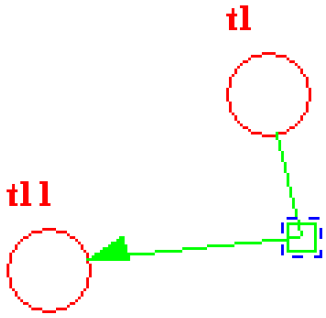
Расчет характеристик эффективности выполнения программы с помощью пакета CSA III методом эквивалентных преобразований ГНД

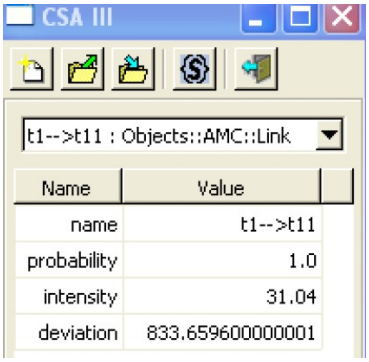


Описание модели.

Lab4.xml

Результаты.





По данным профилирования время выполнения функции составляет 31.04 мкс – результаты расчетов согласуются.

Вывод

При выполнении 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 CSA III были вычислены математическое ожидание и дисперсия времени выполнения.