**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

**отчет**

**по лабораторной работе №2**

**по дисциплине «Качество и метрология программного обеспечения»**

**Тема: Анализ структурной сложности графовых моделей программ**

|  |  |  |
| --- | --- | --- |
| Студент гр. 7304 |  | Субботин А.С. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2021

**Цель работы**

Изучение применения метрик структурной сложности программ – критерия минимального покрытия и анализа базовых маршрутов.

**Исходные данные**

Вариант 17. Исходный граф представлен на Рисунке 1.

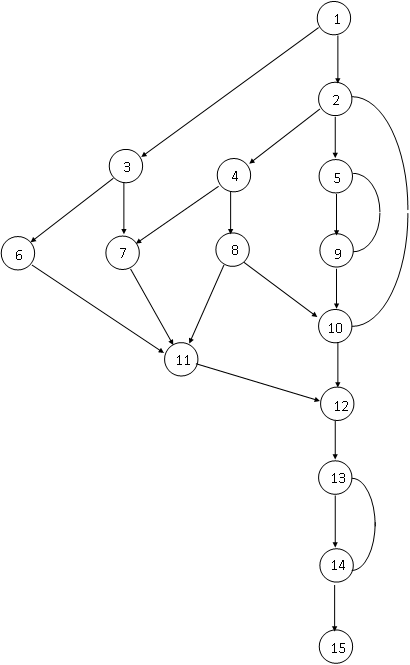


Рисунок 1 – Исходный граф

**Ход работы**

1. Граф был преобразован таким образом, чтобы программа не выявляла некорректных элементов в его структуре:

* Ненаправленная дуга 5-9 была ориентирована из 9 в 5, в обратном случае она повторяет уже имеющуюся в графе дугу
* Ненаправленная дуга 13-14 была ориентирована из 14 в 13, в обратном случае она повторяет уже имеющуюся в графе дугу
* Цикл 2-5-9-10-2 с выходом в вершину 12, а также дуга, например, 2-4-7-11-12, повторяют некорректный вид структуры из методических указаний. Предложенный способ исправления через добавление 16 вершины, и образование путей 1-16-4… и 1-16-2… вместо 1-2-4… и 1-2… не вызывает ошибки в работе программы, но при этом не избавляет от появления предупреждений об ошибочной структуре графа. Предпочтение было отдано попытке решения данной проблемы, поэтому структура исправлена иначе: полностью удалена ненаправленная дуга 2-10

Преобразованный граф представлен на Рисунке 2.

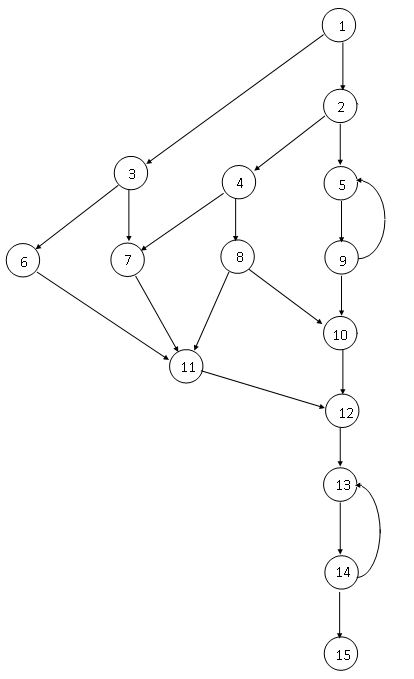
****

Рисунок 2 – Преобразованный граф

1. Ручной расчет по первому критерию:

M1: **1**-**3**-6-11-12-13-**14**-13-**14**-15 = 4

M2: **1**-**3**-7-11-12-13-**14**-15 = 3

M3: **1**-**2**-**4**-7-11-12-13-**14**-15 = 4

M4: **1**-**2**-**4**-**8**-11-12-13-**14**-15 = 5

M5: **1**-**2**-**4**-**8**-10-12-13-**14**-15 = 5

M6: **1**-**2**-5-**9**-5-**9**-10-12-13-**14**-15 = 5

S = 4 + 3 + 4 + 5 + 5 + 5 = 26 – Сложность по первому критерию

Количество маршрутов, необходимое для прохождения по каждой дуге и посещения каждой вершины – 6.

1. Код графа для программы автоматического расчета представлен в Приложении А. Результат автоматического расчета по первому критерию представлен на Рисунке 3.

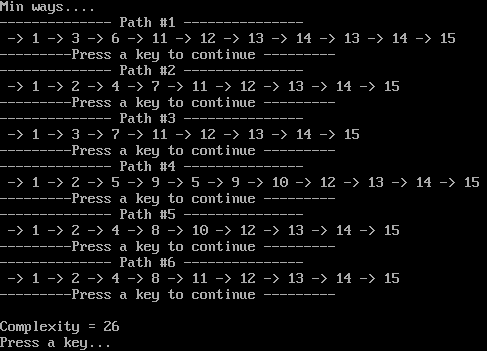


Рисунок 3 – Расчет по первому критерию

Заметим, что построенные маршруты и рассчитанная сложность совпадают с полученными вручную.

1. Ручной расчет по второму критерию:

Y = 20

N = 15

P = 1 (дуга 15-1)

Z = 20 – 15 + 2 \* 1 = 7 – Цикломатическое число

m1: 5-**9**-5 = 1 (цикл)

m2: 13-**14**-13 = 1 (цикл)

m3: **1**-**3**-6-11-12-13-**14**-15 = 3

m4: **1**-**3**-7-11-12-13-**14**-15 = 3

m5: **1**-**2**-**4**-7-11-12-13-**14**-15 = 4

m6: **1**-**2**-**4**-**8**-11-12-13-**14**-15 = 5

m7: **1**-**2**-**4**-**8**-10-12-13-**14**-15 = 5

m8: **1**-**2**-5-**9**-10-12-13-**14**-15 = 4

S = 1 + 1 + 3 + 3 + 4 + 5 + 5 + 4 = 26

1. Результат автоматического расчета по второму критерию представлен на Рисунке 4.

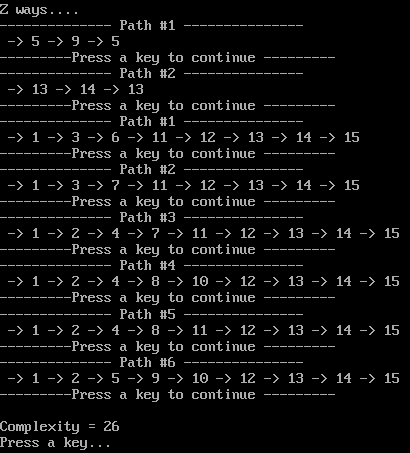


Рисунок 4 – Расчет по второму критерию

Заметим, что построенные маршруты и рассчитанная сложность совпадают с полученными вручную.

1. Исходный код программы, разработанной в ходе первой лабораторной работы, представлен в приложении Б. Граф, построенный для данной программы, представлен на Рисунке 5.

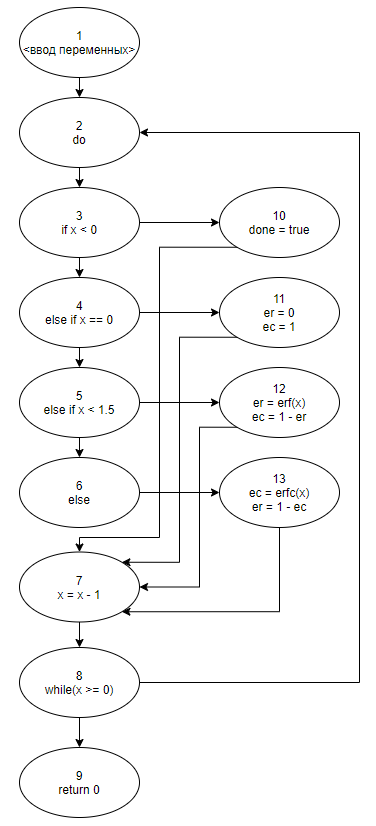


Рисунок 5 – Графовое представление программы из ЛР №1

1. Ручной расчет по первому критерию:

М1: 1-2-**3**-**4**-**5**-6-7-**8**-2-**3**-**4**-**5**-7-**8**-2-**3**-**4**-7-**8**-2-**3**-7-**8**-9 = 13

S = 13 – Сложность по первому критерию

Количество маршрутов, необходимое для прохождения по каждой дуге и посещения каждой вершины – 1.

1. Код графа для программы автоматического расчета представлен в Приложении В. Результат автоматического расчета по первому критерию представлен на Рисунке 6.

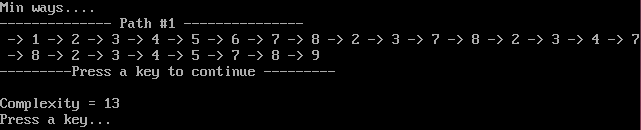


Рисунок 6 – Расчет по первому критерию

Заметим, что построенные маршруты и рассчитанная сложность совпадают с полученными вручную.

1. Ручной расчет по второму критерию:

Y = 12

N = 9

P = 1 (дуга 9-1)

Z = 12 – 9 + 2 \* 1 = 5 – Цикломатическое число

Также возможно использование другого способа:

nв = 4

Z = nв + 1 = 4 + 1 = 5, результаты сходятся

m1: 2-**3**-7-**8**-2 = 2 (цикл)

m2: 1-2-**3**-**4**-**5**-6-7-**8**-9 = 4

m3: 1-2-**3**-**4**-**5**-7-**8**-9 = 4

m4: 1-2-**3**-**4**-7-**8**-9 = 3

m5: 1-2-**3**-7-**8**-9 = 2

S = 2 + 4 + 4 + 3 + 2 = 15

1. Результат автоматического расчета по второму критерию представлен на Рисунке 7.

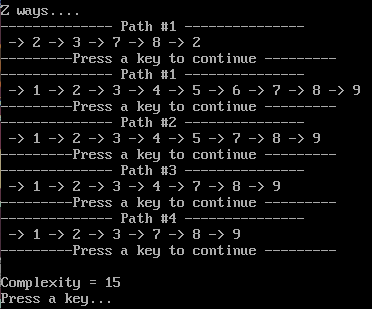


Рисунок 7 – Расчет по второму критерию

Заметим, что построенные маршруты и рассчитанная сложность совпадают с полученными вручную.

**Выводы**

В данной лабораторной работе была выполнена оценка структурной сложности двух программ с помощью критериев: минимального покрытия дуг графа и выбора маршрутов на основе цикломатического числа графа. Расчеты были проведены как ручным, так и программным способом.

**Приложение А. Код преобразованного графа из условия работы**

Nodes{

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

}

Top{1}

Last{15}

Arcs{

arc(1, 3);

arc(1, 2);

arc(2, 4);

arc(2, 5);

arc(3, 6);

arc(3, 7);

arc(4, 7);

arc(4, 8);

arc(5, 9);

arc(6, 11);

arc(7, 11);

arc(8, 10);

arc(8, 11);

arc(9, 5);

arc(9, 10);

arc(10, 12);

arc(11, 12);

arc(12, 13);

arc(13, 14);

arc(14, 13);

arc(14, 15);

}

**Приложение Б. Исходный код программы из ЛР №1 на Си**

#include <stdio.h>

#include <stdbool.h>

#include <math.h>

double erf(double x){

const double sqrtpi = 1.7724538;

double t2 = 0.66666667;

double t3 = 0.66666667;

double t4 = 0.07619048;

double t5 = 0.01693122;

double t6 = 3.078403E-3;

double t7 = 4.736005E-4;

double t8 = 6.314673E-5;

double t9 = 7.429027E-6;

double t10 = 7.820028E-7;

double t11 = 7.447646E-8;

double t12 = 6.476214E-9;

double x2, sum;

x2 = x\*x;

sum = t5+x2\*(t6+x2\*(t7+x2\*(t8+x2\*(t9+x2\*(t10+x2\*(t11+x2\*t12))))));

return (2.0\*exp(-x2)/sqrtpi\*(x\*(1+x2\*(t2+x2\*(t3+x2\*(t4+x2\*sum))))));

}

double erfc(double x){

const double sqrtpi = 1.7724538;

double x2,v,sum;

x2 = x\*x;

v = 1/(2\*x2);

sum=v/(1+8\*v/(1+9\*v/(1+10\*v/(1+11\*v/(1+12\*v)))));

sum=v/(1+3\*v/(1+4\*v/(1+5\*v/(1+6\*v/(1+7\*sum)))));

return (1.0/(exp(x2)\*x\*sqrtpi\*(1+v/(1+2\*sum))));

}

int main()

{

double x,er,ec;

bool done;

x = 2.0;

done = false;

do{

if(x<0){

done = true;

}else if (x == 0){

er = 0;

ec = 1;

}else if (x < 1.5){

er = erf(x);

ec = 1 - er;

}else{

ec = erfc(x);

er = 1-ec;

}

x = x - 1;

}while (done == false);

return 0;

}}

**Приложение В. Код графового представления программы из ЛР №1**

Nodes{

1, 2, 3, 4, 5, 6, 7, 8, 9

}

Top{1}

Last{9}

Arcs{

arc(1, 2);

arc(2, 3);

arc(3, 4);

arc(3, 7);

arc(4, 5);

arc(4, 7);

arc(5, 6);

arc(5, 7);

arc(6, 7);

arc(7, 8);

arc(8, 2);

arc(8, 9);

}