

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»

Студентка гр. 6304

Блинникова Ю.И.

Преподаватель

Кирияничков В.А.

Санкт-Петербург

2020

Формулировка задания

Выполнить оценивание структурной сложности двух программ с помощью критериев:

- Минимального покрытия дуг графа;
- Выбора маршрутов на основе цикломатического числа графа.

Варианты программ:

- Программа с заданной преподавателем структурой управляющего графа;
- Программа из 1-ой лабораторной работы (управляющий граф составить самостоятельно).

Оцениваемые характеристики структурной сложности:

- Число учитываемых маршрутов проверки программы для заданного критерия;
- Цикломатическое число;
- Суммарное число ветвлений по всем маршрутам.

Ход работы

1. Анализ заданной программы (вариант 1)

1.1. Граф программы 1

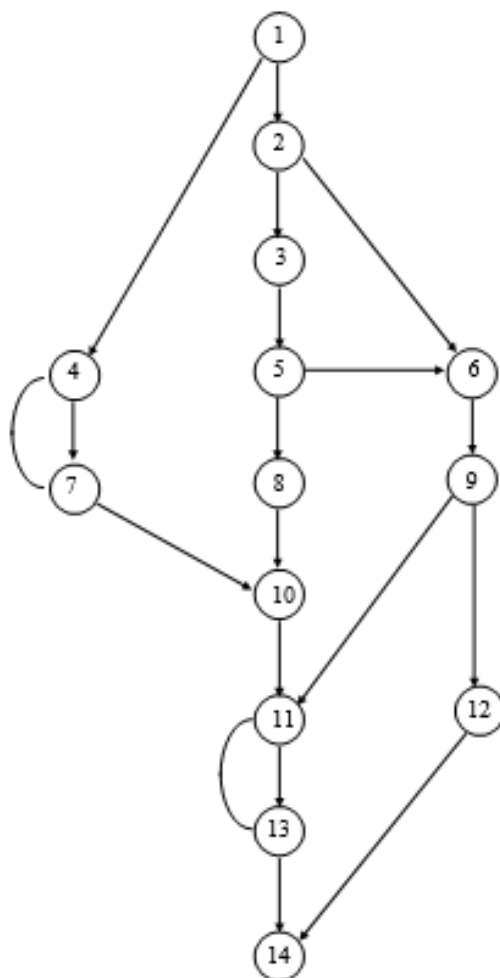


Рисунок 1 – Заданный управляющий граф

1.2. Ручные расчеты

Критерий 1:

М1: **1-2-3-5-6-9-11-13-11-13-14** | 6 ветвлений

М2: **1-2-6-9-12-14** | 3 ветвления

М3: **1-4-7-4-7-10-11-13-14** | 4 ветвления

М4: **1-2-3-5-8-10-11-13-14** | 4 ветвления

Количество маршрутов $M = 4$

Сложность: $S_2 = \sum_{i=1}^M \xi_i = 6 + 3 + 4 + 4 = 17$

Критерий 2:

$$Z = Y - N + 2 * P = 19 - 14 + 2 * 1 = 7$$

Количество проверяемых маршрутов равно цикломатическому числу графа.

M1: 1-2-3-5-6-9-11-13-14	5 ветвлений
M2: 1-2-3-5-6-9-12-14	4 ветвления
M3: 1-2-6-9-12-14	3 ветвления
M4: 1-4-7-10-11-13-14	3 ветвления
M5: 1-2-3-5-8-10-11-13-14	4 ветвления
M6: 11-13-11	1 ветвление
M7: 4-7-4	1 ветвление

$$\text{Сложность: } S_2 = \sum_{i=1}^M \xi_i = 5 + 4 + 3 + 3 + 4 + 1 + 1 = 21$$

1.3. Автоматические расчёты

Описание структуры графа:

Nodes{ 1, 2, 3,4,5,6,7,8,9,10,11,12,13,14 }

Top{ 1 }

Last{ 14 }

Arcs{

arc(1,2);

arc(1,4);

arc(2,3);

arc(2,6);

arc(3,5);

arc(4,7);

arc(5,6);

arc(5,8);

arc(6,9);

arc(7,4);

arc(7,10);

arc(8,10);

arc(9,11);

arc(9,12);

arc(10,11);

arc(11,13);

arc(12,14);

arc(13,11);

arc(13,14);

}

Результат анализа структурной сложности:

```
Min ways....
----- Path #1 -----
-> 1 -> 2 -> 3 -> 5 -> 6 -> 9 -> 11 -> 13 -> 11 -> 13 -> 14
-----Press a key to continue -----
----- Path #2 -----
-> 1 -> 4 -> 7 -> 4 -> 7 -> 10 -> 11 -> 13 -> 14
-----Press a key to continue -----
----- Path #3 -----
-> 1 -> 2 -> 6 -> 9 -> 12 -> 14
-----Press a key to continue -----
----- Path #4 -----
-> 1 -> 2 -> 3 -> 5 -> 8 -> 10 -> 11 -> 13 -> 14
-----Press a key to continue -----

Complexity = 17
Press a key...
```

Рисунок 2 - Результат выполнения ways.exe для 1-го критерия

```
----- Path #1 -----
-> 4 -> 7 -> 4
-----Press a key to continue -----
----- Path #2 -----
-> 11 -> 13 -> 11
-----Press a key to continue -----
----- Path #1 -----
-> 1 -> 2 -> 3 -> 5 -> 6 -> 9 -> 11 -> 13 -> 14
-----Press a key to continue -----
----- Path #2 -----
-> 1 -> 2 -> 3 -> 5 -> 6 -> 9 -> 12 -> 14
-----Press a key to continue -----
----- Path #3 -----
-> 1 -> 2 -> 3 -> 5 -> 8 -> 10 -> 11 -> 13 -> 14
-----Press a key to continue -----
----- Path #4 -----
-> 1 -> 2 -> 6 -> 9 -> 12 -> 14
-----Press a key to continue -----
----- Path #5 -----
-> 1 -> 4 -> 7 -> 10 -> 11 -> 13 -> 14
-----Press a key to continue -----

Complexity = 21
Press a key...
```

Рисунок 3 - Результат выполнения ways.exe для 2-го критерия

2. Анализ программы из 1-ой лабораторной работы

2.1. Граф программы 2

Код программы представлен в приложении А.

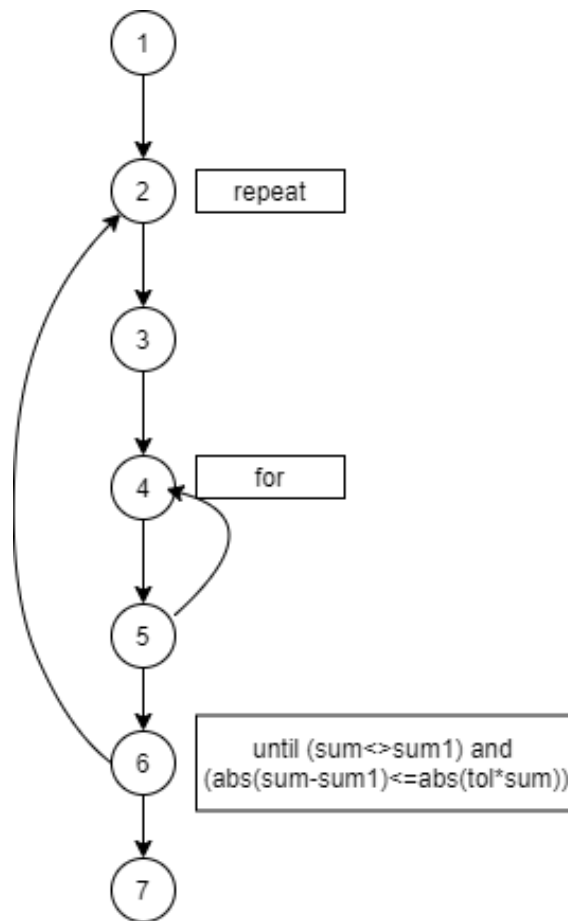


Рисунок 4 – управляющий граф программы 2

2.2. Ручные расчеты

Критерий 1:

M1: 1-2-3-4-**5**-4-**5**-6-2-3-4-**5**-6-7 | 5 ветвлений

Количество маршрутов $M = 1$

Сложность: $S=2$.

Критерий 2:

$$Z = Y - N + 2 * P = 8 - 7 + 2 * 1 = 3$$

Количество проверяемых маршрутов равно цикломатическому числу графа.

M1: 1-2-3-4-**5**-6-7 | 2 ветвления

M2: 4-**5**-4 | 1 ветвление

M3: 2-3-4-**5**-6-2 | 2 ветвления

$$\text{Сложность: } S_2 = \sum_{i=1}^M \xi_i = 2 + 1 + 2 = 5$$

2.3. Автоматические расчёты

Описание структуры графа:

```
Nodes{1, 2, 3,4,5,6,7}  
Top{1}  
Last{7}  
Arcs{  
  arc(1,2);  
  arc(2,3);  
  arc(3,4);  
  arc(4,5);  
  arc(5,4);  
  arc(5,6);  
  arc(6,2);  
  arc(6,7);  
}
```

Результат анализа структурной сложности:

```
Min ways....  
----- Path #1 -----  
-> 1 -> 2 -> 3 -> 4 -> 5 -> 4 -> 5 -> 6 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7  
-----Press a key to continue -----  
  
Complexity = 5  
Press a key...
```

Рисунок 5 - Результат выполнения ways.exe для 1-го критерия

```
Z ways....  
----- Path #1 -----  
-> 4 -> 5 -> 4  
-----Press a key to continue -----  
----- Path #2 -----  
-> 2 -> 3 -> 4 -> 5 -> 6 -> 2  
-----Press a key to continue -----  
----- Path #1 -----  
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7  
-----Press a key to continue -----  
  
Complexity = 5  
Press a key...
```

Рисунок 6 - Результат выполнения ways.exe для 2-го критерия

Вывод

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

ИСХОДНЫЙ КОД ИЗ ЛАБОРАТОРНОЙ №1

```
program simp1;
const      tol      = 1.0E-6;
var sum, upper, lower : real;

function fx(x: real): real;
begin
    fx:=exp(-x/2)
end;      { function fx }
function dfx(x: real): real;
begin
    dfx:=- (exp(-x/2))/2
end;      { function fx }

procedure simps(
    lower, upper, tol : real;
    var sum : real);

var i : integer;
    x, delta_x, even_sum,
    odd_sum, end_sum,
    end_cor, sum1 : real;
    pieces : integer;

begin
    pieces:=2;
    delta_x:=(upper-lower)/pieces;
    odd_sum:=fx(lower+delta_x);
    even_sum:=0.0;
    end_sum:=fx(lower)+fx(upper);
    end_cor:=dfx(lower)-dfx(upper);
    sum:=(end_sum+4.0*odd_sum)*delta_x/3.0;
    repeat
        pieces:=pieces*2;
        sum1:=sum;
        delta_x:=(upper-lower)/pieces;
        even_sum:=even_sum+odd_sum;
        odd_sum:=0.0;
        for i:=1 to pieces div 2 do
            begin
                x:=lower+delta_x*(2.0*i-1.0);
                odd_sum:=odd_sum+fx(x)
            end;
        sum:=(7.0*end_sum+14.0*even_sum+16.00*odd_sum
            +end_cor*delta_x)*delta_x/15.0;
    until (sum<>sum1) and (abs(sum-sum1)<=abs(tol*sum))
end;      { simps }
begin
    { main program }
    lower:=1.0;
    upper:=9.0;
    simps(lower, upper, tol, sum);
    writeln;
    writeln(chr(7), 'area= ', sum)
end.
```