

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

Ажель И.В.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Изучение структурной сложности графовых моделей программ и метрик её оценки.

Постановка задачи.

1. Для заданного варианта провести оценивание структурной сложности двух программ с помощью критериев:
 - a. Минимального покрытия вершин и дуг графа управления;
 - b. Выбора маршрутов на основе цикломатического числа графа;
2. Варианты программ для оценки структурной сложности:
 - a. Программа с заданной структурой управляющего графа, выбираемой из файла `zadan_struct.doc` в соответствии с номером в списке группы;
 - b. Программа из 1-ой лабораторной работы (управляющий граф составить самостоятельно).
3. Оцениваемые характеристики структурной сложности:
 - a. Число учитываемых маршрутов проверки программы для заданного критерия;
 - b. Цикломатическое число;
 - c. Суммарное число ветвлений по всем маршрутам – оценка структурной сложности;
4. Для каждой из заданных программ построение маршрутов и оценивание структурной сложности следует выполнять двумя способами:
 - a. «вручную», непосредственно по управляющему графу программы;
 - b. автоматически с помощью программы `ways.exe` (для контроля результата, полученного «вручную»).

Ход выполнения.

1. Был выбран вариант №2 (номер по списку 2) для первой программы, у которой необходимо определить структурную сложность. Управляющий граф первой программы приведён на Рисунке 1:

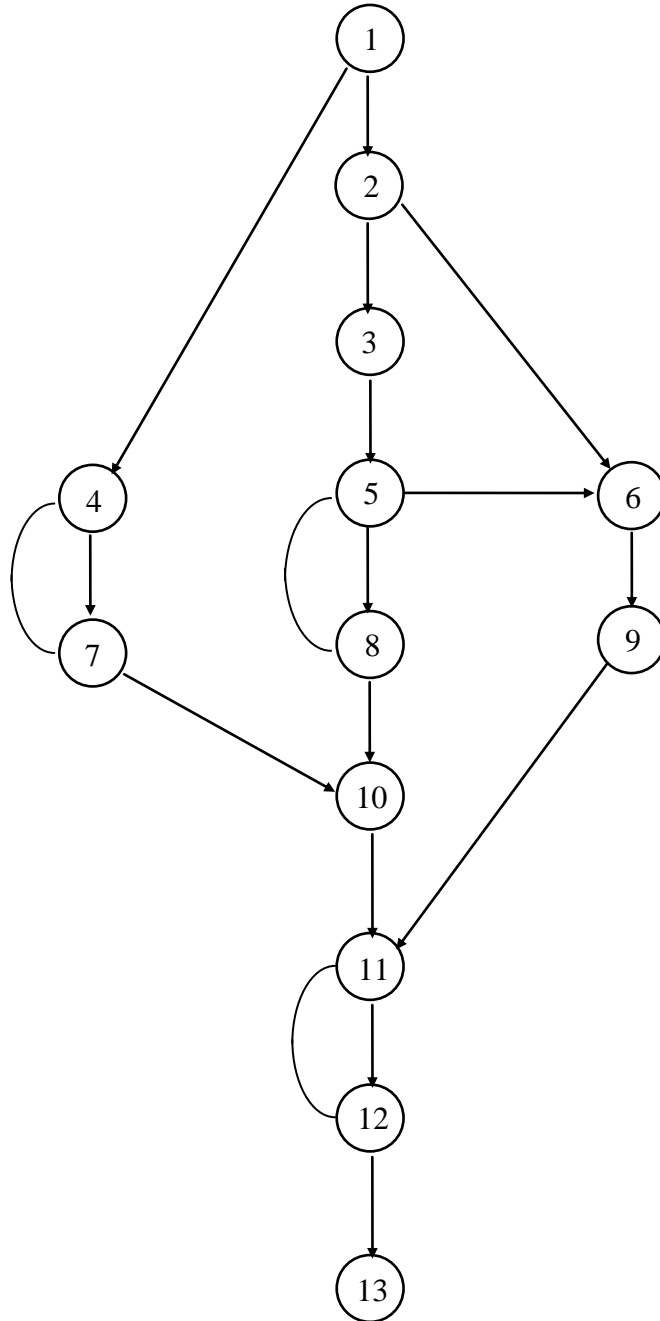


Рисунок 1: Управляющий граф первой программы для определения структурной сложности

2. Была рассчитана структурная сложность первой программы вручную по первому критерию, а именно по минимальному покрытию вершин и дуг графа управления. Результаты расчётов представлены на Таблице 1:

i	mi	Si
1	<u>1</u> – 4 – <u>7</u> – 4 – <u>7</u> – 10 – 11 – <u>12</u> – 11 – <u>12</u> – 13	5
2	<u>1</u> – <u>2</u> – 3 – <u>5</u> – <u>8</u> – <u>5</u> – <u>8</u> – 10 – 11 – <u>12</u> – 13	7
3	<u>1</u> – <u>2</u> – 6 – 9 – 11 – <u>12</u> – 13	3
4	<u>1</u> – <u>2</u> – 3 – <u>5</u> – 6 – 9 – 11 – <u>12</u> – 13	4

Таблица 1: Расчёт структурной сложности первой программы по первому критерию

Сложность программы $S = 19$ единиц.

3. Было рассчитано вручную цикломатическое число графа первой программы для дальнейшего применения второго критерия. Для данной программы число вершин в графе равно 13, число дуг – 18, число связанных компонент графа 1 (максимально связный граф получается путём добавления дугу (16, 1)), тогда:

$$Z = Y - N + 2 * P = 18 - 13 + 2 * 1 = 7,$$

то есть цикломатическое число равно 7.

4. Для управляющего графа первой программы вручную были построены 8 линейно-независимых циклов и линейно-независимых маршрутов, после чего была подсчитана структурная сложность по второму критерию. Результаты представлены на Таблице 2:

i	mi	Si
1	4 – <u>7</u>	1
2	<u>5</u> – <u>8</u>	2
3	11 – <u>12</u>	1
4	<u>1</u> – 4 – <u>7</u> – 10 – 11 – <u>12</u> – 13	3
5	<u>1</u> – <u>2</u> – 3 – <u>5</u> – <u>8</u> – 10 – 11 – <u>12</u> – 13	5
6	<u>1</u> – <u>2</u> – 6 – 9 – 11 – <u>12</u> – 13	3
7	<u>1</u> – <u>2</u> – 3 – <u>5</u> – 6 – 9 – 11 – <u>12</u> – 13	4

Таблица 2: Расчёт структурной сложности первой программы по второму критерию

Сложность программы $S = 19$ единиц.

- Для программы из первой лабораторной работы был составлен управляющий граф, для каждой вершины графа было обозначено, какой фрагмент программы скрывается за ней. Данный управляющий граф представлен на Рисунке 2:

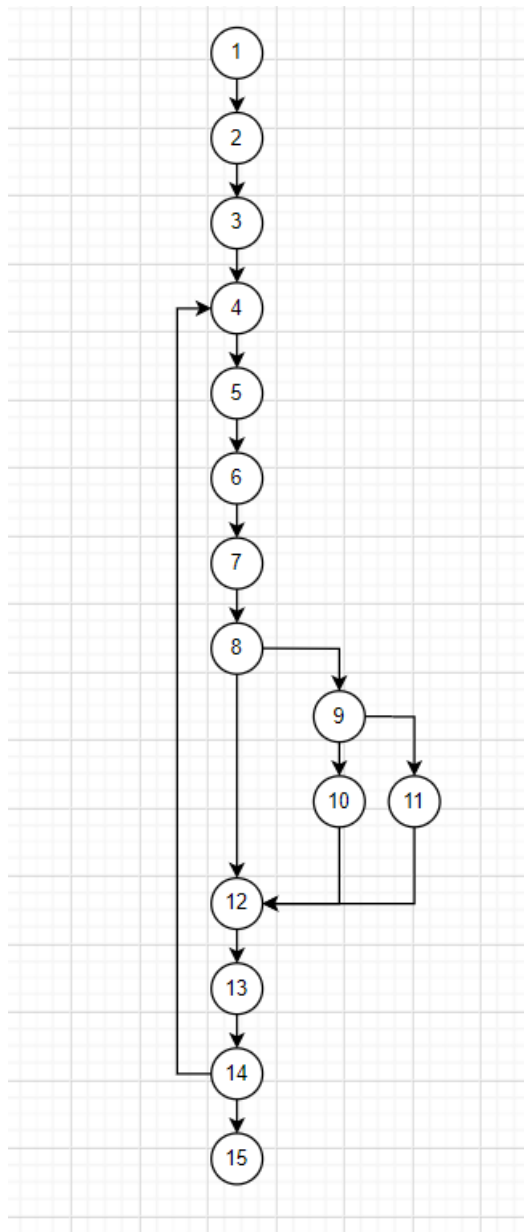


Рисунок 2: Управляющий граф программы из первой лабораторной работы для определения структурной сложности

Соответствие вершин фрагментам программы:

- 1 – `int main() {`
- 2 – `x = rand()%100;`
- 3 – `void newton(float* x) {`
- 4 – `x1 = *x;`
- 5 – `void func(float x, float*fx, float*dfx) {`
- 6 – `*fx = a + b / x + c * log(x) - logp;`

- 7 – $*dfx = -b / (x * x) + c / x$;
- 8 – $fabs(dfx) < tol$
- 9 – $dfx >= 0.0$
- 10 – $dfx = tol$;
- 11 – $dfx = -tol$;
- 12 – $dx = fx / dfx$;
- 13 – $*x = x1 - dx$;
- 14 – $fabs(dx) <= fabs(tol * *x)$
- 15 – }

6. Была рассчитана структурная сложность программы из первой лабораторной работы вручную по первому критерию. Результаты расчётов представлены на Таблице 3:

i	mi	Si
1	1 – 2 – 3 – 4 – 5 – 6 – 7 – <u>8</u> – 12 – 13 – <u>14</u> – 4 – 5 – 6 – 7 – <u>8</u> – <u>9</u> – 10 – 12 – 13 – <u>14</u> – 4 – 5 – 6 – 7 – <u>8</u> – <u>9</u> – 11 – 12 – 13 – <u>14</u> – 15	8

Таблица 3: Расчёт структурной сложности программы из первой лабораторной работы по первому критерию

Сложность программы $S = 8$ единиц.

7. Было рассчитано вручную цикломатическое число графа программы из первой лабораторной работы для дальнейшего применения второго критерия. Для данной программы число вершин в графе равно 15, число дуг – 17, число связных компонент графа 1 (максимально связный граф получается путём добавления дугу (20, 1)), тогда:

$$Z = Y - N + 2 * P = 17 - 15 + 2 * 1 = 4,$$

то есть цикломатическое число равно 4.

8. Для управляющего графа программы из первой лабораторной работы вручную были построены 4 линейно-независимых цикла и линейно-

независимых маршрута, после чего была подсчитана структурная сложность по второму критерию. Результаты представлены на Таблице 4:

i	mi	Si
1	4 – 5 – 6 – 7 – <u>8</u> – 12 – 13 – <u>14</u>	2
2	1 – 2 – 3 – 4 – 5 – 6 – 7 – <u>8</u> – 12 – 13 – <u>14</u> – 15	2
3	1 – 2 – 3 – 4 – 5 – 6 – 7 – <u>8</u> – <u>9</u> – 10 – 12 – 13 – <u>14</u> – 15	3
4	1 – 2 – 3 – 4 – 5 – 6 – 7 – <u>8</u> – <u>9</u> – 11 – 12 – 13 – <u>14</u> – 15	3

Таблица 4: Расчёт структурной сложности программы из первой лабораторной работы по второму критерию

Сложность программы $S = 10$ единиц.

9. Для первой программы была подсчитана структурная сложность по двум критериям с помощью программы ways.exe. Результаты расчёта структурной сложности по первому критерию приведены на Рисунке 3, по второму критерию – на Рисунках 4 и 5:

```

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

Complexity = 18
Press a key...
  
```

Рисунок 3: Программный расчёт структурной сложности первой программы по первому критерию


```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: WAYS

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

```

Рисунок 4: Программный расчёт структурной сложности первой программы по второму критерию, часть 1

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: WAYS

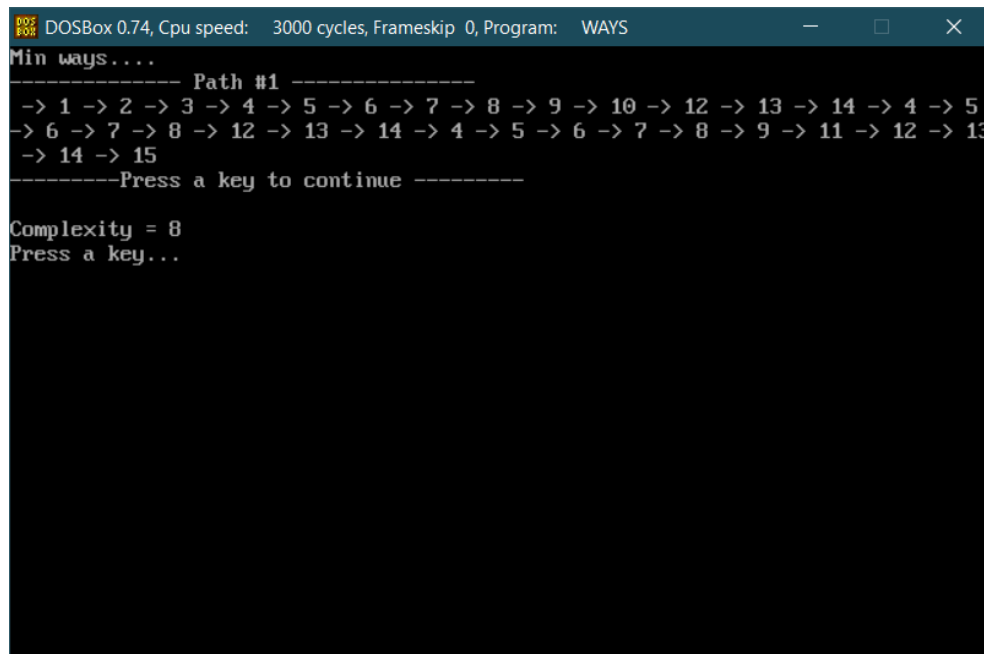
----- Path #1 -----
-> 4 -> 7 -> 4
-----Press a key to continue -----
----- Path #2 -----
-> 5 -> 8 -> 5
-----Press a key to continue -----
----- Path #3 -----
-> 11 -> 12 -> 11
-----Press a key to continue -----
----- Path #1 -----
-> 1 -> 2 -> 3 -> 5 -> 8 -> 10 -> 11 -> 12 -> 13
-----Press a key to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

Complexity = 18
Press a key...
-

```

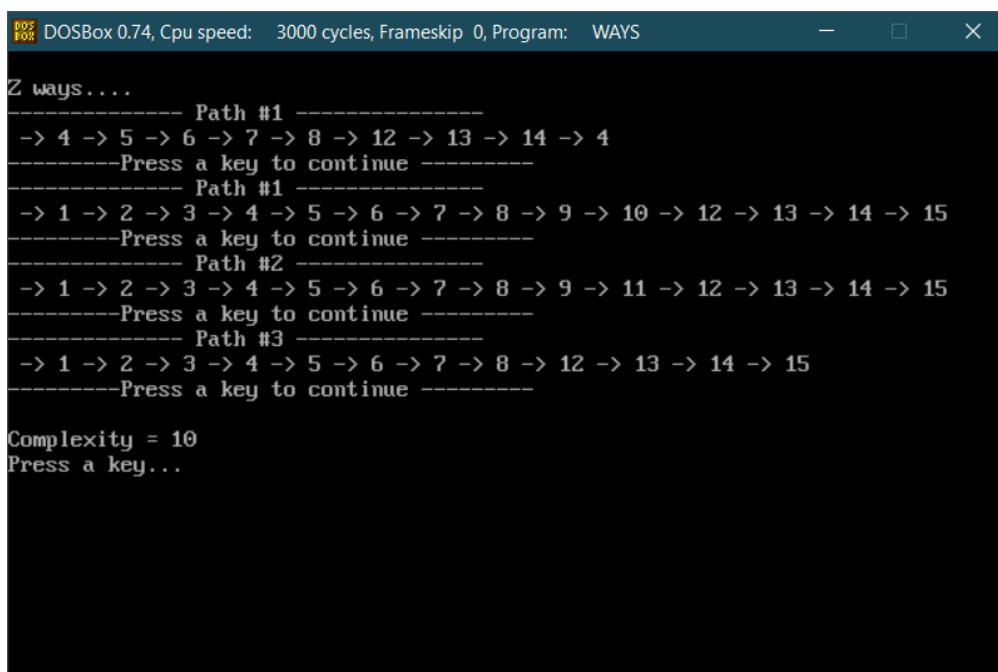
Рисунок 5: Программный расчёт структурной сложности первой программы по второму критерию, часть 2

10. Для программы из первой лабораторной работы была подсчитана структурная сложность по двум критериям с помощью программы ways.exe. Результаты расчёта структурной сложности по первому критерию приведены на Рисунке 6, по второму критерию – на Рисунке 7:



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: WAYS
Min ways....
----- Path #1 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 12 -> 13 -> 14 -> 4 -> 5
-> 6 -> 7 -> 8 -> 12 -> 13 -> 14 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 11 -> 12 -> 13
-> 14 -> 15
-----Press a key to continue -----
Complexity = 8
Press a key...
```

Рисунок 6: Программный расчёт структурной сложности программы из первой лабораторной работы по первому критерию



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: WAYS
Z ways....
----- Path #1 -----
-> 4 -> 5 -> 6 -> 7 -> 8 -> 12 -> 13 -> 14 -> 4
-----Press a key to continue -----
----- Path #1 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 12 -> 13 -> 14 -> 15
-----Press a key to continue -----
----- Path #2 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 11 -> 12 -> 13 -> 14 -> 15
-----Press a key to continue -----
----- Path #3 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 12 -> 13 -> 14 -> 15
-----Press a key to continue -----
Complexity = 10
Press a key...
```

Рисунок 7: Программный расчёт структурной сложности программы из первой лабораторной работы по второму критерию

Выводы.

В ходе выполнения лабораторной работы были изучены методы оценки структурной сложности программы на основе его управляющего графа, была рассчитана структурная сложность двух программ по двум критериям. При этом для одной из программ, взятой из первой лабораторной работы, был составлен управляющий граф. Расчёт структурной сложности программ выполнялся двумя методами: вручную и с помощью программы `ways.exe`. Оба метода для программы из первой лабораторной показали один результат (8 по первому и 10 по второму критериям). Из-за неподобающей структуры граф из файла `zadan_struct.doc` был немного изменен для возможности расчета с помощью программы. В итоге результат при расчете с помощью программы (19 по первому и 19 по второму критериям) немного отличается от полученного при расчете вручную (18 по первому и 18 по второму критериям).

Малая сложность программы из первой лабораторной работы обусловлена наличием всего 2 условных операторов и всего 1 цикла.