Министерство Науки и Образования Российской Федерации

Санкт-Петербургский Государственный Электротехнический Университет «ЛЭТИ»

Кафедра МО ЭВМ

Отчёт по лабораторной работе №1-2

**«**Построение операционных графовых моделей последовательных

программ и их анализ на основе аппарата

поглощающих цепей Маркова**»**

**Вариант 19.**

**Выполнил: Эмман П.А.**

**гр. 3351**

**Проверил: Кирьянчиков В.А.**

**Санкт-Петербург**

**2007**

Оглавление

[Задание 3](#_Toc186207852)

[Ход работы 4](#_Toc186207853)

[Программа для анализа 4](#_Toc186207854)

[Граф программы 6](#_Toc186207855)

[Cтруктурирование графа для метода поуровневой детализации: 11](#_Toc186207856)

[Результаты вычисления матриц при помощи fm.exe 12](#_Toc186207857)

[На входе 12](#_Toc186207858)

[Поуровневая детализация 12](#_Toc186207859)

[Вычисление средних и дисперсий по вычисленной ФМ 14](#_Toc186207860)

[Вывод 16](#_Toc186207861)

# Задание

# Для рассматривавшегося в лабораторных работах курса «Метрология программного обеспечения» индивидуального задания разработать операционную модель управляющего графа программы на основе схемы алгоритма и ассемблерного представления программы.

## При выполнении работы рекомендуется для упрощения обработки графа и ПЦМ исключить диалог при выполнении операций ввода-вывода данных, а также привести программу к структурированному виду.

# Выбрать вариант графа с нагруженными вершинами, 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 из априорных сведений о числе повторений цикла.

# В качестве параметров, характеризующих потребление ресурсов, использовать времена выполнения команд соответствующих участков программы, задаваемые в тактах процессора.

# Выполнить оценку времен выполнения каждого линейного участка и каждого ветвления в графе программы. Оценку времен выполнения участков производить либо с использованием монитора (например, Sampler или Vtune), либо прямым подсчетом по тексту программы.

# Ход работы

## Программа для анализа

Из лабораторных работ курса «МПО»:

#include <math.h>

#include <conio.h>

#include <stdio.h>

double x,er,ec;

int done;

const double sqrtpi = 1.7724538;

const double t2 = 0.66666667;

const double t3 = 0.66666667;

const double t4 = 0.07619048;

const double t5 = 0.01693122;

const double t6 = 3.078403E-3;

const double t7 = 4.736005E-4;

const double t8 = 6.314673E-5;

const double t9 = 7.429027E-6;

const double t10 = 7.820028E-7;

const double t11 = 7.447646E-8;

const double t12 = 6.476214E-9;

double erf(double x)

{

double x2 = x \* x;

double 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)))));

}// { function erf }

double erfc(double x)

{

double x2 = x \* x;

double v =1.0/(2.0\*x2);

double 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))));

}// { function ercf }

void main(void)

{

//clrscr();

done = false;

printf("\n");

do{

printf("Arg? ");

scanf("%fl",&x);

if (x<0.0) {done = true;}

else

{

if (x = 0.0)

{

er = 0.0;

ec = 1.0;

}else

{

if (x<1.5)

{

er = erf(x);

ec = 1.0 - er;

}else

{

ec = erfc(x);

er = 1.0 - ec;

}

}

printf("\nX=%d, Erf=%d, Erfc=%d",x,er,ec);

}

}while(done);

getch();

}

Измененная программа в ходе лабораторных работ по курсу «МПО»:

#include "SAMPLER.H"

#include <math.h>

#include <conio.h>

#include <stdio.h>

double x,er,ec;

int done;

const double sqrtpi = 1.7724538;

const double t2 = 0.66666667;

const double t3 = 0.66666667;

const double t4 = 0.07619048;

const double t5 = 0.01693122;

const double t6 = 3.078403E-3;

const double t7 = 4.736005E-4;

const double t8 = 6.314673E-5;

const double t9 = 7.429027E-6;

const double t10 = 7.820028E-7;

const double t11 = 7.447646E-8;

const double t12 = 6.476214E-9;

double erf(double x)

{

double x2 = x \* x;

double 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)))));

}// { function erf }

double erfc(double x)

{

double x2 = x \* x;

double v =1.0/(2.0\*x2);

double 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))));

}// { function ercf }

void main(void)

{

x = 1.01f;

while(x < 2){

if (x == 0.0)

{

er = 0.0;

ec = 1.0;

}else

{

if (x<1.5)

{

er = erf(x);

ec = 1.0 - er;

}else

{

ec = erfc(x);

er = 1.0 - ec;

}

}

printf("\nX=%f, Erf=%f, Erfc=%f",x,er,ec);

x += 0.1;

}

}

## Граф программы

|  |  |
| --- | --- |
| № | Текст программы |
| 1 | x = 1.01f; |
| 2 | while(x < 2){ |
| 3 | if (x == 0.0) |
| 4 | {  er = 0.0;  ec = 1.0;  } |
|  | else  { |
| 5 | if (x<1.5) |
| 6 | {  er = erf(x);  ec = 1.0 - er;  } |
|  | else |
| 7 | {  ec = erfc(x);  er = 1.0 - ec;  } |
|  | } |
| 8 | printf("\nX=%f, Erf=%f, Erfc=%f",x,er,ec);  x += 0.1; |
| 9 | }  } |



Особенность данной программы то, что она по сути линейна. При «правильном» ее использовании, вычисление будут о(1). Для проведения измерений мы ее «зацикливаем». Но количество повторов всегда заранее известно: сколько значений будет перебираться до 1,5, и сколько после – по этому соотношению будет соотношение выбора ветки 5-6 5-7. ветка 3-4 представляет собой лишь «страховой случай».

double erf(double x)

{

double x2 = x \* x;

double 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)))));

}// { function erf }

double erfc(double x)

{

double x2 = x \* x;

double v =1.0/(2.0\*x2);

double 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))));

}// { function ercf }

void main(void)

{

|  |  |
| --- | --- |
| № | Текст программы |
| 1 | 40-42 |
| 2 | 42-45 |
| 3 | 45-54 |
| 4 | - |
| 5 | 54-57 |
| 6 | 57-60 |
| 7 | 63-66 |
| 8 | 69-72 |
| 9 | 72-45 |

SAMPLE; //40

x = 1.01f;

SAMPLE; //42

while(x < 2){

SAMPLE; //45

if (x == 0.0)

{

SAMPLE; //

er = 0.0;

ec = 1.0;

SAMPLE; //

}else

{

SAMPLE; //54

if (x<1.5)

{

SAMPLE; //57

er = erf(x);

ec = 1.0 - er;

SAMPLE; //60

}else

{

SAMPLE; //63

ec = erfc(x);

er = 1.0 - ec;

SAMPLE; //66

}

}

SAMPLE; //69

printf("\nX=%f, Erf=%f, Erfc=%f",x,er,ec);

x += 0.01;

SAMPLE; //72

}

}

Проверим это по семплеру:

Установлено время коррекции 5 [минитиков] <==> 4.19 [микросекунд].

-----

Таблица с результатами измерений ( используется 10 из 416 записей )

----------------------------------------------------------------------

Исх.Поз. Прием.Поз. Общее время(мкс) Кол-во прох. Среднее время(мкс)

----------------------------------------------------------------------

1 : 40 1 : 42 0.00 1 0.00

----------------------------------------------------------------------

1 : 42 1 : 45 0.84 1 0.84

----------------------------------------------------------------------

1 : 45 1 : 54 67.89 100 0.68

----------------------------------------------------------------------

1 : 54 1 : 57 27.66 50 0.55

1 : 54 1 : 63 36.88 50 0.74

----------------------------------------------------------------------

1 : 57 1 : 60 80.46 50 1.61

----------------------------------------------------------------------

1 : 60 1 : 69 34.36 50 0.69

----------------------------------------------------------------------

1 : 63 1 : 66 54.48 50 1.09

----------------------------------------------------------------------

1 : 66 1 : 69 36.04 50 0.72

----------------------------------------------------------------------

1 : 69 1 : 72 2207.55 100 22.08

----------------------------------------------------------------------

1 : 72 1 : 45 75.43 99 0.76

----------------------------------------------------------------------

Проведем тестирование для вычислений, начиная с 0.

Таблица с результатами измерений ( используется 12 из 416 записей )

----------------------------------------------------------------------

Исх.Поз. Прием.Поз. Общее время(мкс) Кол-во прох. Среднее время(мкс)

----------------------------------------------------------------------

1 : 40 1 : 42 0.00 1 0.00

----------------------------------------------------------------------

1 : 42 1 : 45 0.84 1 0.84

----------------------------------------------------------------------

1 : 45 1 : 48 0.84 1 0.84

1 : 45 1 : 54 227.12 300 0.76

----------------------------------------------------------------------

1 : 48 1 : 51 0.00 1 0.00

----------------------------------------------------------------------

1 : 51 1 : 69 0.84 1 0.84

----------------------------------------------------------------------

1 : 54 1 : 57 290.82 149 1.95

1 : 54 1 : 63 109.79 151 0.73

----------------------------------------------------------------------

1 : 57 1 : 60 144.99 149 0.97

----------------------------------------------------------------------

1 : 60 1 : 69 104.76 149 0.70

----------------------------------------------------------------------

1 : 63 1 : 66 173.49 151 1.15

----------------------------------------------------------------------

1 : 66 1 : 69 113.98 151 0.75

----------------------------------------------------------------------

1 : 69 1 : 72 7039.17 301 23.39

----------------------------------------------------------------------

1 : 72 1 : 45 457.60 300 1.53

----------------------------------------------------------------------

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | для интервала (1…2) | | для интервала (0…3) | | Такты |
| Текст программы | Время выполнения,  мксек | Текст программы | Время выполнения,  мксек |
| 1 | 40-42 | 0 | 40-42 | 0 | 1 |
| 2 | 42-45 | 0,84 | 42-45 | 0,84 | 1 |
| 3 | 45-54 | 0,68 | 45-54 | 0,76 | 1 |
| 4 | - | - | 45-48 | 0,84 | 1 |
| 5 | 54-57 | 0,55 | 54-57 | 1,95 | 1 |
| 6 | 57-60 | 0,74 | 57-60 | 0,97 | 1 |
| 7 | 63-66 | 1,09 | 63-66 | 1,15 | 1 |
| 8 | 69-72 | 22,08 | 69-72 | 23,39 | 6 |
| 9 | 72-45 | 0,76 | 72-45 | 1,53 | 1 |

# Для полученного графа построить соответствующую ему поглощающую цепь Маркова (ПЦМ), определить ее фундаментальную матрицу(ФМ) и вектор нагрузочных парметров L.

# Вычислить оценки средних времен, дисперсии и СКО времен выполнения как всей программы, так и ее основных фрагментов, на которые она может быть разбита.

# Определение ФМ ПЦМ требуется выполнить двумя способами:

# путем непосредственного обращения матрицы (I-Q), полученной по переходной матрице ПЦМ, соответствующей графу всей программы;

# путем структурной детализации фундаментальных матриц, соответствующих подграфам элементарных вычислительных процессов.

# Оценки времен выполнения следует определить как в тактах, так и в абсолютных единицах времени (сек, мсек или мксек). Результаты расчетов представить в виде таблиц как для всей программы, так и для ее фрагментов.

**ВЕКТОР ЗАТРАТ**



|  |  |
| --- | --- |
| ВЕРШИНА | ВЕС |
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 1 |
| 5 | 1 |
| 6 | 1 |
| 7 | 1 |
| 8 | 6 |
| 9 | 1 |

|  |  |  |
| --- | --- | --- |
| ВЕРШИНА | ПЕРЕХОД | ВЕРОЯТНОСТЬ |
| 2 | 2-9 | 1/301 = 0,33% |
| 2-3 | 300/301 = 99,66% |
| 3 | 3-4 | 1/301 = 0,33% |
| 3-5 | 300/301 = 99,66% |
| 5 | 5-6 | 149/300 = 49,66% |
| 5-7 | 151/300 = 50,33% |

## Cтруктурирование графа для метода поуровневой детализации:

**S3**



|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 |

**S2**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1/(1-q) | Q/(1-q) | 1 |
| 0 | 1/(1-q) | Q/(1-q) | 1 |
| 0 | 1/(1-q) | 1/(1-q) | 1 |
| 0 | 0 | 0 | 1 |

Q = 0.9966

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 303,0303 | 302 | 1 |
| 0 | 303,0303 | 302 | 1 |
| 0 | 1303,0303 | 303,0303 | 1 |
| 0 | 0 | 0 | 1 |

**S1**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | p | 1-p | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 |

p = 0.4966

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0.4966 | 0.5033 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 |

## Результаты вычисления матриц при помощи fm.exe

## На входе

tops {

C1(1),

C2(1),

B1(1),

B2(1),

A1(1),

A2(1),

A3(1),

A4(1),

B3(6),

C3(1)

}

links {

C1->C2(1),

C2->B1(0.9966),

B1->B2(0.33),

B1->A1(0.9966),

A1->A2(0.4966),

A1->A3(0.5033),

A2->A4(1),

A3->A4(1),

A4->B3(1),

B2->B3(1),

B3->C2(1),

C2->C3(0.0034)

}

## Поуровневая детализация

Файл получен с помощью программы fm.exe Version 1.0.1 (build Dec 14 2000 17:32:35)

Вызвано распознавание шаблона для подграфа @ROOT:

начальная вершина C1, конечная вершина C3

Текущее состояние фундаментальной матрицы:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@ROOT ║ 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Конструкция ЦИКЛ

Вершина проверки условия: C2

Дуга, входящая в тело цикла: C2->B1

Дуга, выходящая из тела цикла: B3->C2

Вероятность повторения тела цикла: 0.9966

Вероятность выхода из цикла: 0.0034

Матрица для подстановки:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 1 ║

C2 ║ 0 294.1 293.1 1 ║

@TMP0 ║ 0 294.1 294.1 1 ║

C3 ║ 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Вызвано распознавание шаблона для подграфа @TMP0:

начальная вершина B1, конечная вершина B3

Текущее состояние фундаментальной матрицы:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 1 ║

C2 ║ 0 294.1 293.1 1 ║

@TMP0 ║ 0 294.1 294.1 1 ║

C3 ║ 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Конструкция ВЕТВЛЕНИЕ

Матрица для подстановки:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

B1 ║ 1 0.9966 0.0034 1 ║

@TMP1 ║ 0 1 0 1 ║

@TMP2 ║ 0 0 1 1 ║

B3 ║ 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Вызвано распознавание шаблона для подграфа @TMP1:

начальная вершина A1, конечная вершина A4

Текущее состояние фундаментальной матрицы:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 292.1 0.9966 293.1 1 ║

C2 ║ 0 294.1 293.1 292.1 0.9966 293.1 1 ║

B1 ║ 0 294.1 294.1 293.1 1 294.1 1 ║

@TMP1 ║ 0 294.1 293.1 293.1 0.9966 294.1 1 ║

@TMP2 ║ 0 294.1 293.1 292.1 1.997 294.1 1 ║

B3 ║ 0 294.1 293.1 292.1 0.9966 294.1 1 ║

C3 ║ 0 0 0 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Конструкция ВЕТВЛЕНИЕ

Матрица для подстановки:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

A1 ║ 1 0.5034 0.4966 1 ║

@TMP3 ║ 0 1 0 1 ║

@TMP4 ║ 0 0 1 1 ║

A4 ║ 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Вызвано распознавание шаблона для подграфа @TMP3:

начальная вершина A3, конечная вершина A3

Текущее состояние фундаментальной матрицы:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

C2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

B1 ║ 0 294.1 294.1 293.1 147.6 145.6 293.1 1 294.1 1 ║

A1 ║ 0 294.1 293.1 293.1 147.6 145.6 293.1 0.9966 294.1 1 ║

@TMP3 ║ 0 294.1 293.1 292.1 148.1 145.1 293.1 0.9966 294.1 1 ║

@TMP4 ║ 0 294.1 293.1 292.1 147.1 146.1 293.1 0.9966 294.1 1 ║

A4 ║ 0 294.1 293.1 292.1 147.1 145.1 293.1 0.9966 294.1 1 ║

@TMP2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 1.997 294.1 1 ║

B3 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 294.1 1 ║

C3 ║ 0 0 0 0 0 0 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Шаблон распознан как атомарная операция

Вызвано распознавание шаблона для подграфа @TMP4:

начальная вершина A2, конечная вершина A2

Текущее состояние фундаментальной матрицы:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

C2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

B1 ║ 0 294.1 294.1 293.1 147.6 145.6 293.1 1 294.1 1 ║

A1 ║ 0 294.1 293.1 293.1 147.6 145.6 293.1 0.9966 294.1 1 ║

A3 ║ 0 294.1 293.1 292.1 148.1 145.1 293.1 0.9966 294.1 1 ║

@TMP4 ║ 0 294.1 293.1 292.1 147.1 146.1 293.1 0.9966 294.1 1 ║

A4 ║ 0 294.1 293.1 292.1 147.1 145.1 293.1 0.9966 294.1 1 ║

@TMP2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 1.997 294.1 1 ║

B3 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 294.1 1 ║

C3 ║ 0 0 0 0 0 0 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Шаблон распознан как атомарная операция

Вызвано распознавание шаблона для подграфа @TMP2:

начальная вершина B2, конечная вершина B2

Текущее состояние фундаментальной матрицы:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

C2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

B1 ║ 0 294.1 294.1 293.1 147.6 145.6 293.1 1 294.1 1 ║

A1 ║ 0 294.1 293.1 293.1 147.6 145.6 293.1 0.9966 294.1 1 ║

A3 ║ 0 294.1 293.1 292.1 148.1 145.1 293.1 0.9966 294.1 1 ║

A2 ║ 0 294.1 293.1 292.1 147.1 146.1 293.1 0.9966 294.1 1 ║

A4 ║ 0 294.1 293.1 292.1 147.1 145.1 293.1 0.9966 294.1 1 ║

@TMP2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 1.997 294.1 1 ║

B3 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 294.1 1 ║

C3 ║ 0 0 0 0 0 0 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Шаблон распознан как атомарная операция

Результат:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

C2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 293.1 1 ║

B1 ║ 0 294.1 294.1 293.1 147.6 145.6 293.1 1 294.1 1 ║

A1 ║ 0 294.1 293.1 293.1 147.6 145.6 293.1 0.9966 294.1 1 ║

A3 ║ 0 294.1 293.1 292.1 148.1 145.1 293.1 0.9966 294.1 1 ║

A2 ║ 0 294.1 293.1 292.1 147.1 146.1 293.1 0.9966 294.1 1 ║

A4 ║ 0 294.1 293.1 292.1 147.1 145.1 293.1 0.9966 294.1 1 ║

B2 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 1.997 294.1 1 ║

B3 ║ 0 294.1 293.1 292.1 147.1 145.1 292.1 0.9966 294.1 1 ║

C3 ║ 0 0 0 0 0 0 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Алгебраический метод**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 294.1 293.1 0.9966 292.1 145.1 147.1 292.1 293.1 1 ║

C2 ║ 0 294.1 293.1 0.9966 292.1 145.1 147.1 292.1 293.1 1 ║

B1 ║ 0 294.1 294.1 1 293.1 145.6 147.6 293.1 294.1 1 ║

B2 ║ 0 294.1 293.1 1.997 292.1 145.1 147.1 292.1 294.1 1 ║

A1 ║ 0 294.1 293.1 0.9966 293.1 145.6 147.6 293.1 294.1 1 ║

A2 ║ 0 294.1 293.1 0.9966 292.1 146.1 147.1 293.1 294.1 1 ║

A3 ║ 0 294.1 293.1 0.9966 292.1 145.1 148.1 293.1 294.1 1 ║

A4 ║ 0 294.1 293.1 0.9966 292.1 145.1 147.1 293.1 294.1 1 ║

B3 ║ 0 294.1 293.1 0.9966 292.1 145.1 147.1 292.1 294.1 1 ║

C3 ║ 0 0 0 0 0 0 0 0 0 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Вычисление средних и дисперсий по вычисленной ФМ

Потребление ресурсов:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1 ║

C2 ║ 1 ║

B1 ║ 1 ║

B2 ║ 1 ║

A1 ║ 1 ║

A2 ║ 1 ║

A3 ║ 1 ║

A4 ║ 1 ║

B3 ║ 6 ║

C3 ║ 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Средние значения потребления ресурсов:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 3225 ║

C2 ║ 3224 ║

B1 ║ 3234 ║

B2 ║ 3231 ║

A1 ║ 3233 ║

A2 ║ 3232 ║

A3 ║ 3232 ║

A4 ║ 3231 ║

B3 ║ 3230 ║

C3 ║ 1 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Дисперсии потребления ресурсов:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

C1 ║ 1.042e+007 ║

C2 ║ 1.042e+007 ║

B1 ║ 1.042e+007 ║

B2 ║ 1.042e+007 ║

A1 ║ 1.042e+007 ║

A2 ║ 1.042e+007 ║

A3 ║ 1.042e+007 ║

A4 ║ 1.042e+007 ║

B3 ║ 1.042e+007 ║

C3 ║ 0 ║

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Вывод

В ходе лабораторной работы построен управляющий граф программы, определены вероятности выполнения ветвей графа, потребление процессорного времени участками программы.

При помощи ПС fm.exe построена фундаментальная матрица для заданной программы — двумя способами: алгебраическим (преобразованием матрицы переходов и обращением результата) и методом поуровневой детализации. Вычислены средние знания потребления процессорного времени операциями программы в тактах и микросекундах.

Из-за особой простоты программы, получены несколько сомнительные результаты.