**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

**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. 7304 |  | Субботин А.С. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2021

**Цель работы**

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

**Исходные данные**

Вариант 17. Исходный код для задания был получен преобразованием исходного кода на Си из лабораторной работы №1 соответствующего варианта.

**Ход работы**

1. Для программы вычисления функции ошибок распределения Гаусса на Си из предыдущих работ (см. Приложение А) был построен управляющий граф. Результат представлен на Рисунке 1.

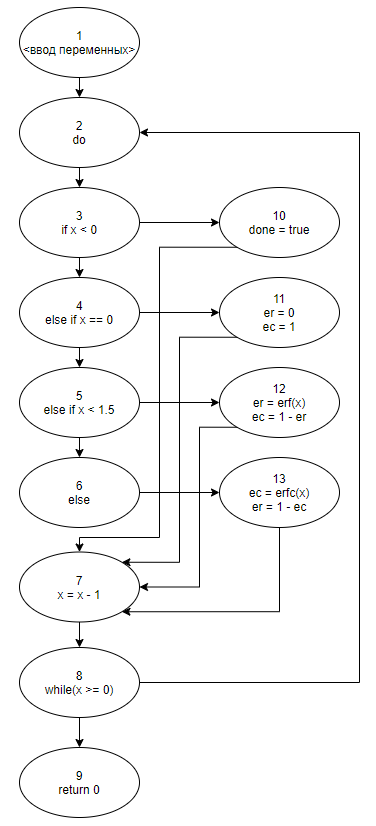


Рисунок 1 – Управляющий граф программы

1. Код программы был подготовлен к профилированию с использованием Sampler, код с пронумерованными строками представлен в Приложении Б. Результат профилирование представлен на Рисунке 2.

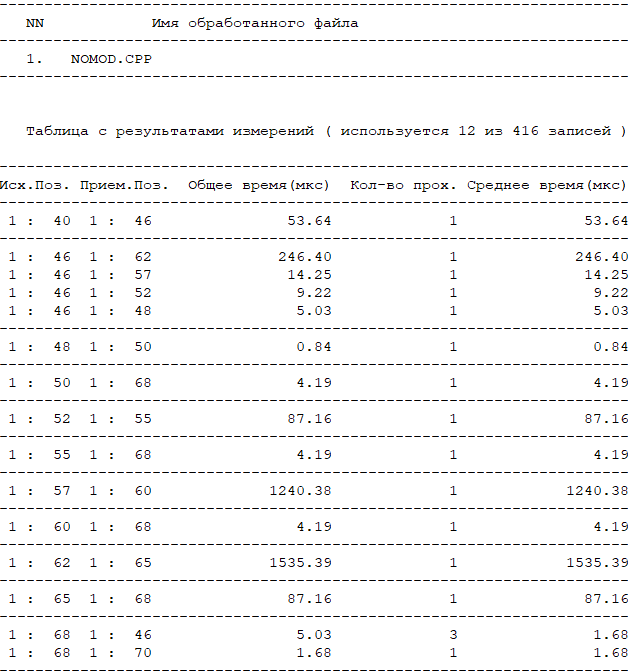


Рисунок 2 – Результат профилирования программы

Суммарное время отдельных замеров – 3298.75 мкс

1. Расчет вероятностей и затрат ресурсов для дуг управляющего графа представлен в Таблице 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Дуги | Номера строк | Количество проходов | Время, мкс |
| 1-2 | 40:46 | 1 | 53.64 |
| 2-3 | 46:48 | 1 | 5.03 |
| 3-10 | 48:50 | 1 | 0.84 |
| 10-7-8 | 50:68 | 1 | 4.19 |
| 8-2 | 68:46 | 3 | 1.68\*3~5.03 |
| 2-4 | 46:52 | 1 | 9.22 |
| 4-11 | 52:55 | 1 | 87.16 |
| 11-7-8 | 55:68 | 1 | 4.19 |
| 2-5 | 46:57 | 1 | 14.25 |
| 5-12 | 57:60 | 1 | 1240.38 |
| 12-7-8 | 60:68 | 1 | 4.19 |
| 2-6 | 46:62 | 1 | 246.40 |
| 6-13 | 62:65 | 1 | 1535.39 |
| 13-7-8 | 65:68 | 1 | 87.16 |
| 8-9 | 68:70 | 1 | 1.68 |

Таблица 1 – Вероятности и затраты ресурсов для дуг управляющего графа

Do..while:

Программа вошла в цикл 4 раза и вышла из него 1 раз, следовательно вероятность дуги t7-t1 = 0.75, а дуги t7-t13 = 0.25

If..else:

Вершина 3 – Программа посетила вершину 4 раза. T/F = 1/3

Вершина 4 – Программа посетила вершину 3 раза. T/F = 1/2

Вершина 5 – Программа посетила вершину 2 раза. T/F = 1/1

1. С использованием полученных вероятностей и затрат ресурсов с помощью пакета CSA III был получен граф с нагруженными дугами. Результат представлен на рисунке 3.

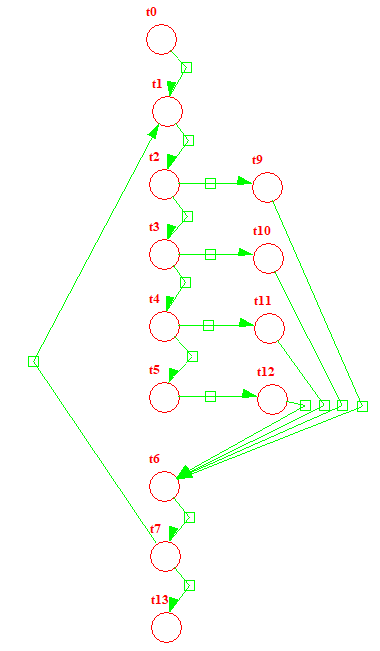


Рисунок 3 – Граф с нагруженными дугами

Описание модели представлено в Приложении В.

Результат оценки программой (математическое ожидание и дисперсия) времени выполнения для всей программы представлен на Рисунке 4.

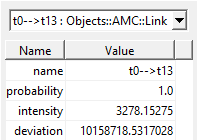


Рисунок 4 – Результат

**Выводы**

При выполнении лабораторной работы была построена операционная графовая модель программы из ЛР1, было оценено время выполнения программы с помощью профилировщика Sampler и методом эквивалентных преобразований c помощью пакета CSA III. Результаты сравнения этих характеристик показали, что метод эквивалентных преобразований даёт очень близкие результаты к результатам работы программы Sampler (разница менее одного процента свидетельствует об адекватности построенной модели).

**Приложение А. Код программы вычисления функции ошибок распределения Гаусса на Си**

1. #include <stdio.h>

2. #include <stdlib.h>

3. #include <math.h>

4.

5. double erf(double x){

6. const double sqrtpi = 1.7724538;

7. double t2 = 0.66666667;

8. double t3 = 0.66666667;

9. double t4 = 0.07619048;

10. double t5 = 0.01693122;

11. double t6 = 3.078403E-3;

12. double t7 = 4.736005E-4;

13. double t8 = 6.314673E-5;

14. double t9 = 7.429027E-6;

15. double t10 = 7.820028E-7;

16. double t11 = 7.447646E-8;

17. double t12 = 6.476214E-9;

18.

19. double x2, sum;

20. x2 = x\*x;

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

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

23.

24. }

25.

26. double erfc(double x){

27. const double sqrtpi = 1.7724538;

28. double x2,v,sum;

29. x2 = x\*x;

30. v = 1/(2\*x2);

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

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

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

34. }

35.

36. int main()

37. {

38. double x,er,ec;

39. bool done;

40. x = 2.0;

41. done = false;

42. do{

43. if(x<0){

44. done = true;

45. }else if (x == 0){

46. er = 0;

47. ec = 1;

48. }else if (x < 1.5){

49. er = erf(x);

50. ec = 1 - er;

51. }else{

52. ec = erfc(x);

53. er = 1-ec;

54. }

55. x = x - 1;

56.

57. }while (done == false);

58.

59. return 0;

60. }

**Приложение Б. Текст программы для профилирования**

1. #include <stdio.h>

2. #include <stdlib.h>

3. #include <math.h>

4. #include "SAMPLER.H"

5. typedef enum { false, true } bool;

6.

7. double erf(double x){

8. const double sqrtpi = 1.7724538;

9. double t2 = 0.66666667;

10. double t3 = 0.66666667;

11. double t4 = 0.07619048;

12. double t5 = 0.01693122;

13. double t6 = 3.078403E-3;

14. double t7 = 4.736005E-4;

15. double t8 = 6.314673E-5;

16. double t9 = 7.429027E-6;

17. double t10 = 7.820028E-7;

18. double t11 = 7.447646E-8;

19. double t12 = 6.476214E-9;

20.

21. double x2, sum;

22. x2 = x\*x;

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

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

25.

26. }

27.

28. double erfc(double x){

29. const double sqrtpi = 1.7724538;

30. double x2,v,sum;

31. x2 = x\*x;

32. v = 1/(2\*x2);

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

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

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

36. }

37.

38. int main()

39. {

40. SAMPLE;

41. double x,er,ec;

42. bool done;

43. x = 2.0;

44. done = false;

45. do{

46. SAMPLE;

47. if(x<0){

48. SAMPLE;

49. done = true;

50. SAMPLE;

51. }else if (x == 0){

52. SAMPLE;

53. er = 0;

54. ec = 1;

55. SAMPLE;

56. }else if (x < 1.5){

57. SAMPLE;

58. er = erf(x);

59. ec = 1 - er;

60. SAMPLE;

61. }else{

62. SAMPLE;

63. ec = erfc(x);

64. er = 1-ec;

65. SAMPLE;

66. }

67. x = x - 1;

68. SAMPLE;

69. }while (done == false);

70. SAMPLE;

71. return 0;

72. }

**Приложение В. XML файл для CSA III**

<model type = "Objects::AMC::Model" name = "lab4">

<node type = "Objects::AMC::Top" name = "t0"></node>

<node type = "Objects::AMC::Top" name = "t1"></node>

<node type = "Objects::AMC::Top" name = "t2"></node>

<node type = "Objects::AMC::Top" name = "t3"></node>

<node type = "Objects::AMC::Top" name = "t4"></node>

<node type = "Objects::AMC::Top" name = "t5"></node>

<node type = "Objects::AMC::Top" name = "t6"></node>

<node type = "Objects::AMC::Top" name = "t7"></node>

<node type = "Objects::AMC::Top" name = "t9"></node>

<node type = "Objects::AMC::Top" name = "t10"></node>

<node type = "Objects::AMC::Top" name = "t11"></node>

<node type = "Objects::AMC::Top" name = "t12"></node>

<node type = "Objects::AMC::Top" name = "t13"></node>

<link type = "Objects::AMC::Link" name = "t0-->t1" probability = "1.0" intensity = "53.64" deviation = "0.0" source = "t0" dest = "t1"></link>

<link type = "Objects::AMC::Link" name = "t1-->t2" probability = "1.0" intensity = "5.03" deviation = "0.0" source = "t1" dest = "t2"></link>

<link type = "Objects::AMC::Link" name = "t2-->t3" probability = "0.75" intensity = "9.22" deviation = "0.0" source = "t2" dest = "t3"></link>

<link type = "Objects::AMC::Link" name = "t3-->t4" probability = "0.67" intensity = "14.25" deviation = "0.0" source = "t3" dest = "t4"></link>

<link type = "Objects::AMC::Link" name = "t4-->t5" probability = "0.5" intensity = "246.3" deviation = "0.0" source = "t4" dest = "t5"></link>

<link type = "Objects::AMC::Link" name = "t6-->t7" probability = "1.0" intensity = "4.19" deviation = "0.0" source = "t6" dest = "t7"></link>

<link type = "Objects::AMC::Link" name = "t2-->t9" probability = "0.25" intensity = "0.84" deviation = "0.0" source = "t2" dest = "t9"></link>

<link type = "Objects::AMC::Link" name = "t3-->t10" probability = "0.33" intensity = "87.16" deviation = "0.0" source = "t3" dest = "t10"></link>

<link type = "Objects::AMC::Link" name = "t4-->t11" probability = "0.5" intensity = "1240.68" deviation = "0.0" source = "t4" dest = "t11"></link>

<link type = "Objects::AMC::Link" name = "t5-->t12" probability = "1.0" intensity = "1535.39" deviation = "0.0" source = "t5" dest = "t12"></link>

<link type = "Objects::AMC::Link" name = "t9-->t6" probability = "1.0" intensity = "0.0" deviation = "0.0" source = "t9" dest = "t6"></link>

<link type = "Objects::AMC::Link" name = "t10-->t6" probability = "1.0" intensity = "0.0" deviation = "0.0" source = "t10" dest = "t6"></link>

<link type = "Objects::AMC::Link" name = "t11-->t6" probability = "1.0" intensity = "0.0" deviation = "0.0" source = "t11" dest = "t6"></link>

<link type = "Objects::AMC::Link" name = "t12-->t6" probability = "1.0" intensity = "0.0" deviation = "0.0" source = "t12" dest = "t6"></link>

<link type = "Objects::AMC::Link" name = "t7-->t13" probability = "0.25" intensity = "1.68" deviation = "0.0" source = "t7" dest = "t13"></link>

<link type = "Objects::AMC::Link" name = "t7-->t1" probability = "0.75" intensity = "1.68" deviation = "0.0" source = "t7" dest = "t1"></link>

</model>