МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3
по дисциплине «Качество и метрология программного обеспечения»
Тема: Измерение характеристик динамической сложности программ с помощью профилировщика SAMPLER

Студент гр. 7304 Шарапенков И.И.

Преподаватель Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Изучение возможности измерения динамических характеристик программ с помощью профилировщиков на примере профилировщика SAMPLER.

Постановка задачи.

1. Ознакомиться с документацией на монитор SAMPLER и выполнить под его управлением тестовые программы test_cyc.c и test_sub.c с анализом параметров повторения циклов, структуры описания циклов, способов профилирования процедур и проверкой их влияния на точность и чувствительность профилирования.
2. Скомпилировать и выполнить под управлением SAMPLER'a программу на C, разработанную в 1-ой лабораторной работе.

   Выполнить разбиение программы на функциональные участки и снять профили для двух режимов:

   a. измерение только полного времени выполнения программы;
   b. измерение времен выполнения функциональных участков (ФУ);
3. Выявить "узкие места", связанные с ухудшением производительности программы, ввести в программу усовершенствования и получить новые профили. Объяснить смысл введенных модификаций программ.

Ход выполнения.

1. Была изучена документация монитора SAMPLER, после чего под его управлением были запущены тестовые программы test_cyc.c и test_sub.c. Для проведения измерений использовалась старая версия профилировщика SAMPLER, запуск которого осуществлялся через DOSBox. Результаты работы монитора программ test_cyc.c и test_sub.c продемонстрированы в Таблицах 1 и 2 соответственно:

Исход. Поз.	Прием. Поз.	Общее время (мкс)	Кол-во проходов	Среднее время (мкс)
1 : 9	1 : 11	4335.47	1	4335.47
1 : 11	1 : 13	8668.43	1	8668.43
1 : 13	1 : 15	21678.20	1	21678.20
1 : 15	1 : 17	43348.87	1	43348.87
1 : 17	1 : 20	4335.47	1	4335.47
1 : 20	1 : 23	8670.11	1	8670.11
1 : 23	1 : 26	21672.34	1	21672.34
1 : 26	1 : 29	43348.03	1	43348.03
1 : 29	1 : 35	4334.64	1	4334.64
1 : 35	1 : 41	8670.11	1	8670.11
1 : 41	1 : 47	21676.53	1	21676.53
1 : 47	1 : 51	43348.87	1	43348.87

Таблица 1: Результаты профилирования тестовой программы test_cyc.c

Исход. Поз.	Прием. Поз.	Общее время (мкс)	Кол-во проходов	Среднее время (мкс)
1 : 24	1 : 26	433697.35	1	433697.35
1 : 26	1 : 28	867391.34	1	867391.34
1 : 28	1 : 30	2168480.87	1	2168480.87
1 : 30	1 : 32	4336936.59	1	4336936.59

Таблица 2: Результаты профилирования тестовой программы test_sub.c

2. При помощи Borland C++ была скомпилирована программа, написанная на Си, из первой лабораторной работы (program.cpp) после чего была запущена под управлением SAMPLER'a в режиме измерения полного времени выполнения программы. Результаты измерения приведены на Таблице 3:

Исход. Поз.	Прием. Поз.	Общее время (мкс)	Кол-во проходов	Среднее время (мкс)
1 : 45	1 : 61	2060.04	1	2060.04

Таблица 3: Результаты профилирования программы из первой лабораторной работы (полное время работы программы)

3. Программ из первой лабораторной работы была разбита на функциональные участки следующим образом:
    a. Функция main:
        i. строка 42 – строка 49: начало работы функции, объявление переменных, инициализация генератора псевдослучайных чисел;
        ii. строка 49 – строка 55: цикл по генерации исходных данных;
        iii. строка 55 – строка 57: вызов функции linfit1, окончание работы функции;
    b. Функция linfit1:
        i. строка 7 – строка 15: начало работы функции, объявление переменных;
        ii. строка 15 – строка 26: цикл по вычислению сумм;
        iii. строка 26 – строка 32: вычисление параметров для линеаризации;
        iv. строка 32 – строка 37: цикл для заполнения результирующего массива;
4. Разбитая на функциональные участки программа была скомпилирована и запущена под управлением SAMPLER'a. Результаты профилирования показаны на Таблице 4:

Исход. Поз.	Прием. Поз.	Общее время (мкс)	Кол-во проходов	Среднее время (мкс)
1 : 8	1 : 19	207.85	1	207.85
1 : 19	1 : 26	464.31	1	464.31
1 : 26	1 : 66	284.95	1	284.95
1 : 31	1 : 40	294.17	1	294.17
1 : 40	1 : 45	238.86	1	238.86
1 : 45	1 : 66	253.94	1	253.94
1 : 52	1 : 8	247.24	1	247.24
1 : 52	1 : 31	64.53	1	64.53
1 : 66	1 : 52	1.68	1	1.68

Таблица 4: Результаты профилирования программы из первой лабораторной работы (разбиение на функциональные участки)

Суммарное время работы T = 2057,53 мкс.

По результатам профилирования видно, что наибольшее время выполнения у функционального участка с циклом в функции erf. Однако в целом большинство участков имеют примерно одинаковое время выполнения. С помощью избавления от ненужных промежуточных переменных, констант, избавления от ненужных условий цикла, удалось сократить время выполнения многих участков. Измененная программа была записана в файл program_update.cpp.

5. Изменённая программа была скомпилирована и запущена под управлением SAMPLER'a. Результаты профилирования показаны на Таблице 5:

Исход. Поз.	Прием. Поз.	Общее время (мкс)	Кол-во проходов	Среднее время (мкс)
1 : 8	1 : 15	87.16	1	87.16
1 : 15	1 : 21	421.56	1	421.56
1 : 21	1 : 52	302.55	1	302.55
1 : 26	1 : 32	207.85	1	207.85
1 : 32	1 : 36	190.25	1	190.25
1 : 36	1 : 52	270.71	1	270.71
1 : 43	1 : 8	160.08	1	160.08
1 : 43	1 : 26	61.18	1	61.18
1 : 52	1 : 43	1.68	1	1.68

Таблица 5: Результаты профилирования измененной программы из первой лабораторной работы

Суммарное время работы T = 1703,02 мкс, уменьшение времени работы составило 354,51 мкс (~ 17%).

Выводы.

В ходе выполнения лабораторной работы была изучена возможность измерения динамических характеристик программ с помощью профилировщиков и было измерено с помощью профилировщика SAMPLER время выполнения всего кода и время выполнения функциональных участков тестовых программ test_cyc.c и test_sub.c, а также программы из первой лабораторной работы.

В ходе профилирования было выяснено, что на цикл в функции erf приходится наибольшее время выполнения среди всех функциональных участков, после чего была проведена оптимизация программы за счёт удаления ненужных переменных и вследствие этого сокращения ненужных вычислений,

что привело к уменьшению времени работы на 354,51 мкс, то есть на 17% от времени работы неоптимизированной программы.