**Министерство науки и высшего образования Российской Федерации**

**федеральное государственное автономное образовательное учреждение высшего образования**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**(Университет ИТМО)**

**Факультет программной инженерии и компьютерной техники**

Отчет по лабораторной работе №4

по дисциплине «Параллельное программирование»

***Тема* «Метод доверительных интервалов при измерении времени выполнения параллельной OpenMP-программы»**

Выполнили:

студенты гр. № P33212

Зиганшин Геннадий Муратович

Ян Цзяфэн

Проверил:

Балакшин Павел Валерьевич

Санкт-Петербург

2021

Содержание

[Краткая характеристика 3](#_Toc72886575)

[Описание задачи 3](#_Toc72886576)

[Коды программы 4](#_Toc72886577)

[Графики 4](#_Toc72886578)

[Вывод 6](#_Toc72886579)

# Краткая характеристика

* Процессор: AMD Ryzen 5 3600 6-Core Processor
* Операционная система: 20.04.1-Ubuntu
* Компилятор GCC: gcc 9.3.0
* ОЗУ 32 гб

# Описание задачи

1. В программе, полученной в результате выполнения ЛР-3, так изменить этап Generate, чтобы генерируемый набор случайных чисел не зависел от количества потоков, выполняющих программу. Например, на каждой итерации i перед вызовом rand\_r можно вызывать функцию srand(f(i)), где f – произвольно выбранная функция. Можно придумать и использовать любой другой способ.

2. Заменить вызовы функции gettimeofday на omp\_get\_wtime.

3. Распараллелить вычисления на этапе Sort, для чего выполнить сортировку в два этапа:

* Отсортировать первую и вторую половину массива в двух независимых нитях (можно использовать OpenMP-директиву ”parallel sections”);
* Объединить отсортированные половины в единый массив.

4. Написать функцию, которая один раз в секунду выводит в консоль сообщение о текущем проценте завершения работы программы. Указанную функцию необходимо запустить в отдельном потоке, параллельно работающем с основным вычислительным циклом.

5. Обеспечить прямую совместимость (forward compatibility) написанной параллельной программы. Для этого все вызываемые функции вида «omp\_\*» можно условно переопределить в препроцессорных директивах.

6. Провести эксперименты, варьируя N от min( Nx/2 , N1) до N2, где значения N1 и N2 взять из ЛР-1, а Nx – это такое значение N, при котором накладные расходы на распараллеливание превышают выигрыш от распараллеливания. Написать отчёт о проделанной работе. Подготовиться к устным вопросам на защите.

7. Необязательное задание на «четвёрку» и «пятёрку». Уменьшить количество итераций основного цикла с 100 до 10 и провести эксперименты, замеряя время выполнения следующими методами:

* Использование минимального из десяти полученных замеров;
* Расчёт по десяти измерениям доверительного интервала с уровнем доверия 95%. Привести графики параллельного ускорения для обоих методов в одной системе координат, при этом нижнюю и верхнюю границу доверительного интервала следует привести двумя независимыми графиками.

8. Необязательное задание на «пятёрку»: в п.3 задания на этапе Sort выполнить параллельную сортировку не двух частей массива, а k частей в k нитях (тредах), где k – это количество процессоров (ядер) в системе, которое становится известным только на этапе выполнения программы с помощью команды «k = omp\_get\_num\_procs()».

# Коды программы

lab4.c в виде отдельных файлов.

# Графики

*Рис. 1 – График времени выполнения при N1<N<N2*

*Рис. 2 – График времени выполнения при N<N1*

*Рис. 3 – График параллельного ускорения при N1<N<N2*

*Рис. 4 – График параллельного ускорения при N<N1*

*Рис. 5 – График времени выполнения* *для доверительного интервала с распараллеливанием*

*Рис. 6 – График времени выполнения* *для доверительного интервала без распараллеливания*

*Рис. 6 – График параллельного ускорения для пункта 7*

# Вывод