Федеральное государственное бюджетное образовательное учреждение высшего образования «Сибирский государственный университет телекоммуникаций и информатики» (СибГУТИ)

Кафедра вычислительных систем

**ОТЧЁТ**

по лабораторной работе №5

«**ПАРАЛЛЕЛИЗМ ЗАДАЧ**»

|  |  |  |
| --- | --- | --- |
| Выполнил:  студент гр. ИС-241  «19» мая 2024 г. |  | Бондаренко А.А. |
|  |  |  |
| Проверил:  преподаватель  «20» мая 2024 г. |  | Челканова Т.В. |

Новосибирск 2024

**ОГЛАВЛЕНИЕ**

[ЗАДАНИЕ 3](#_Toc167039584)

[ОПИСАНИЕ ФУНКЦИЙ 4](#_Toc167039585)

[ partition 4](#_Toc167039586)

[ quicksort\_serial 4](#_Toc167039587)

[ quicksort\_tasks 4](#_Toc167039588)

[ run\_parallel 4](#_Toc167039589)

[ run\_serial 4](#_Toc167039590)

[АЛГОРИТМ РАБОТЫ 5](#_Toc167039591)

[РЕЗУЛЬТАТЫ РАБОТЫ 6](#_Toc167039592)

# **ЗАДАНИЕ**

**Задание**

* На базе директив #pragma omp task реализовать многопоточный рекурсивный алгоритм быстрой сортировки (QuickSort). Опорным выбирать центральный элемент подмассива (функция partition, см. слайды к лекции). При достижении подмассивами размеров THREASHOLD = 1000 элементов переключаться на последовательную версию алгоритма.
* Выполнить анализ масштабируемости алгоритма для различного числа сортируемых элементов и порогового значения THRESHOLD.

**Защита работы**

1. Продемонстрировать код программы и графики ускорения
2. Описать суть распараллеливания алгоритма
3. Охарактеризовать эффективность созданной параллельной программы

# **ОПИСАНИЕ ФУНКЦИЙ**

## partition

Вспомогательная функция для алгоритма быстрой сортировки.

Функция выбирает опорный элемент массива и переставляет элементы участка массива таким образом, чтобы массив разбился на 2 части: левая часть содержит элементы, которые меньше опорного, а правая часть содержит элементы, которые больше него или ему (int pivot).

## quicksort\_serial

Реализует алгоритм последовательной (нераспараллеленной) быстрой сортировки. Сначала вызывается функция partition, после с заданными индексами рекурсивно запускается следующая итерация сортировки.

## quicksort\_tasks

Представляет собой распараллеленную версию сортировки с использованием директивы #pragma omp task untied – она преобразовывает рекурсивные запуски в задачи (task-и) и распределяет их между потоками. По мере освобождения потоки могут перенимать чужие task-и на выполнение (work stealing).

После вызова функции partition проверяется, достигли ли подмассивы пороговой величины THREASHOLD = 1000. В случае, если это так, функция переходит в последовательный режим работы, не создавая новые task-и. Если же пороговое значение ещё не достигнуто, то работа продолжается по вышеописанной распараллеленной схеме.

## run\_parallel

Осуществляет запуск параллельной версии на заданном количестве потоков с замером времени работы алгоритма.

## run\_serial

Осуществляет запуск последовательной с замером времени работы алгоритма.

# **АЛГОРИТМ РАБОТЫ**

1. Создание массива имён файлов, в которые будут сохранятся показатели ускорения при тестировании программы на разном количестве элементов (массивы с SIZE = 100000, 200000, 300000, 400000, 500000 и THRESHOLD = 1000, 10000, 100000)
2. Запуск цикла для наблюдения за ускорение в зависимости от вышеприведённых объёмов входного массива, производится пять проходов по циклу
3. Создание массива для последовательной сортировки, элементы задаются случайно
4. Открытие файла для записи с соответствующим именем в зависимости от индекса j цикла for
5. Вызов run\_serial для текущего объёма данных
6. Запуск цикла для вызова run\_parallel на разном количестве потоков (2, 4, 6, 8). Также внутри цикла создаётся локальная версия неотсортированного массива, там же память освобождается
7. В конце цикла (2) очищается память, хранящая первый из сортируемых массивов, и закрывается файл
8. Программа запускается трижды с разными значениями THRESHOLD (1000, 10000, 100000)

# **РЕЗУЛЬТАТЫ РАБОТЫ**

Ниже приведён **скриншот запуска программы.**



Команды:

1. Запуск скрипта для компиляции исполняемого файла с соответствующими флагами
2. Запуск исполняемого файла
3. Запуск скрипта для построения графика утилитой GNU-polt

В выходные файлы data1.dat, data2.dat, data3.dat, data4.dat, data5.dat, производится запись результатов в формате:

«количество потоков коэффициент ускорения»

**График**, построенный по полученным результатам: