МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Нижегородский государственный университет им. Н.И. Лобачевского»**

**Национальный исследовательский университет**

**Отчет по лабораторной работе**

**«Умножение плотных матриц. Элементы типа double. Блочная схема, алгоритм Кэннона.»**

**Выполнил:** студент группы 381506-1

Герасимов Дмитрий Геннадьевич

Нижний Новгород

2018

**Содержание**

[**Постановка задачи** 3](#_Toc514605545)

[**Последовательный алгоритм** 4](#_Toc514605546)

[**Схема распараллеливания** 4](#_Toc514605547)

[**Описание программной реализации** 5](#_Toc514605548)

[**Подтверждение корректности** 6](#_Toc514605549)

[**Результаты экспериментов по оценке масштабируемости** 7](#_Toc514605550)

[***OpenMP:*** 7](#_Toc514605551)

[***TBB:*** 8](#_Toc514605552)

[**Вывод** 10](#_Toc514605553)

# **Постановка задачи**

Цель данной работы – написать последовательную и параллельные версии (OpenMP и TBB) программы перемножения двух плотных матриц. Для параллельных версий реализовать алгоритм Кэннона блочного перемножения матриц.

Таким образом, выполнение задачи включает в себя:

1. Реализацию последовательной версии перемножения матриц.
2. Реализацию параллельной версии алгоритма Кэннона перемножения матриц с использованием технологии OpenMP.
3. Реализацию параллельной версии алгоритма Кэннона перемножения матриц с использованием технологии Intel TBB.
4. Проведение экспериментов, сравнение времени выполнения программ.
5. Вывод, основанный на проведенных экспериментах.

# **Последовательный алгоритм**

Последовательный алгоритм перемножения двух плотных матриц A и B размерности [n \*n] реализуется формулой:

# **Схема распараллеливания**

Для корректной работы алгоритма Кэннона необходимо, чтобы матрицы были квадратными [*n \* n*], число потоков *p* было полным квадратом, а размер матрицы был кратен корню из числа потоков.

В алгоритме Кэннона матрицы *A* и *B* разбиваются на *p* квадратных блоков размерностью ( каждый. Блоки организуют своеобразную квадратную решетку. Для каждой строки *i* решетки блоки матрицы A сдвигаются на (*i-1*)позиций влево, для каждого столбца *j* решетки блоки матрицы B сдвигаются на (*j-1*)позиций вверх. Далее проводится итераций, во время которых сначала происходит перемножение блоков по методу трех вложенных циклов, и произведение складывается с текущим значением результирующего блока. Затем выполняется циклический сдвиг блоков матрицы *А* по строкам и циклический сдвиг блоков матрицы В вверх по столбцам решетки. После выполнения всех итераций результирующие блоки собираются в результирующую матрицу.

В OpenMP и TBB используется общая память, поэтому алгоритм сводится к тому, что каждый поток перемножает свои блоки и записывает результат в результирующую матрицу.

# **Описание программной реализации**

* OmpV1 – решение задачи последовательного алгоритма.
* Checker – программа для проверки корректности полученных результатов.

Через командную строку программе подаются входные данные, эталонный результат, результат решения программы. Checker выдает текстовый файл с результатом проверки.

* Generator – программа для генерации тестов.

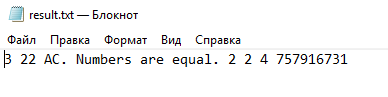
Через командную строку программе подается номер теста, который программа должна сгенерировать. Программа генерирует эталонный результат с помощью стандартного алгоритма перемножения матриц. Generator выдает 2 файла: входные данные и эталонный результат данного теста.

* Typer – программа, преобразующая входные данных из текстового вида в бинарный.
* Viewer – программа, преобразующая ответ из бинарного вида в текстовый.
* Cannon\_Omp2 – решение задачи параллельным алгоритмом с помощью OpenMP.
* Tbb\_pp – решение задачи параллельным алгоритмом с помощью Intel TBB.
* Папка «test» - набор тестов для проверки корректности результатов.

# **Подтверждение корректности**

Для подтверждения корректности нужно запустить программу Checker, в котором эталонный результат сравнивается с результатом одной из программ решения задачи (OmpV1, Cannon\_Omp2, Tbb\_pp). Checker выдает текстовый файл «Result.txt» с результатом проверки.

Примерный вид файла:



**Результаты экспериментов по оценке масштабируемости**

Все эксперименты проводились на компьютере, который имеет следующие характеристики:

* Процессор: Intel Core i5- 4210U с частотой 1.70GHz (boost 2.40GHz)
* Объем ОЗУ: 8Gb
* Система: Windows 10 с разрядностью 64-бит

## **OpenMP:**

Таблица 1 Время выполнения OpenMP версии

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Размер матрицы | | 540х540 | 720х720 | 1260х1260 | 1980х1980 |
| Время выполнения последовательной реализации | | 0,215894 | 0,716164 | 17,0827451 | 76,4489761 |
| Время выполнения параллельной реализации (с) | 4 потоков | 0,111173 | 0,262098 | 9,2159849 | 40,9470471 |
| 9 потоков | 0,097241 | 0,248838 | 5,21958885 | 34,2941696 |
| 16 потоков | 0,089883 | 0,24239 | 3,571285 | 12,6560556 |
| 25 потоков | 0,08781 | 0,234352 | 2,228515 | 7,90946445 |
| 36 потоков | 0,087856 | 0,206557 | 1,4183975 | 6,59867255 |

Рисунок . Зависимость времени выполнения от числа потоков

Таблица 2 Ускорение в OpenMP версии

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Размер  матрицы Число потоков | 540х540 | 720х720 | 1260х1260 | 1980х1980 |
| 4 | 1,94196507 | 2,732433 | 1,8536 | 1,86702049 |
| 9 | 2,22019689 | 2,87804 | 3,272814 | 2,22921205 |
| 16 | 2,40195255 | 2,954593 | 4,783361 | 6,04050571 |
| 25 | 2,45865206 | 3,055941 | 7,665528 | 9,665506 |
| 36 | 2,45737034 | 3,467151 | 12,04369 | 11,5855084 |

Рисунок Ускорение OpenMP версии

## **TBB:**

Таблица 3 Время выполнения TBB версии

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Размер матрицы | | 540х540 | 720х720 | 1260х1260 | 1980х1980 |
| Время выполнения последовательной реализации | | 0,215894 | 0,716164 | 17,0827451 | 76,44897613 |
| Время выполнения параллельной реализации (с) | 4 потоков | 0,264973 | 0,689882 | 9,321013 | 41,17223 |
| 9 потоков | 0,249133 | 0,634278 | 6,931648 | 32,11245 |
| 16 потоков | 0,248069 | 0,604308 | 5,85921 | 22,683 |
| 25 потоков | 0,233691 | 0,587638 | 4,72879 | 20,1775 |
| 36 потоков | 0,233089 | 0,569631 | 3,735035 | 17,87503 |

Рисунок Зависимость времени выполнения от числа потоков

Таблица 4 Ускорение в TBB версии

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Размер  матрицы Число потоков | 540х540 | 720х720 | 1260х1260 | 1980х1980 |
| 4 | 0,8147767 | 1,038097 | 1,832713 | 1,85680922 |
| 9 | 0,86657999 | 1,129102 | 2,464457 | 2,3806647 |
| 16 | 0,87029512 | 1,185098 | 2,915537 | 3,37032033 |
| 25 | 0,92384293 | 1,218716 | 3,612498 | 3,788823 |
| 36 | 0,92622894 | 1,257244 | 4,573651 | 4,27685862 |

Рисунок Ускорение TBB версии

# **Вывод**

Из проведенных экспериментов можно сделать вывод, что в основном на больших значениях матрицы с увеличением числа потоков, уменьшается время выполнения программы. Алгоритм Кэннона показывает ускорение, близкое к линейному.

Алгоритм Кэннона может работать как быстрее, так и медленнее теоретической оценки. Не всегда слишком больше число потоков положительно сказывается на времени выполнении программы. В каждом случае есть предел, после которого увеличение числа потоков почти не дает никакого ускорения. Это особенно заметно на относительно небольших матрицах.

Также можно отметить, что на больших размерах матрицы время выполнения OpenMP и TBB версий примерно совпадает.