Задание 4

Умножение матриц.

Минь Синьжань

1. Постановка задачи

С помощью интринсик используемой целевой архитектуры (AVX для Intel, AMD; NEON для ARM) реализовать векторизованную версию матричного умножения (A\*B = C). Можно предполагать, что матрицы A, B - квадратные. Обязательное требование - хранение всех матриц предполагается в едином порядке (все в row-major либо все в col-major). Тип элементов матрицы - float для 128-битных, double для 256-битных векторных расширений. Сравнить результаты и время выполнения векторизованного алгоритма с его последовательной версией. (N = 512, 1024, 2048).

2. Формат командной строки

Для Apple M3: g++ -O3 -std=c++11 -o file\_name vector\_neon.cpp

Для Intel i7: g++ -O2 -mavx -mfma vector\_avx.cpp -o file\_name

3. Спецификация системы

Процессор: Apple M3/Intel CORE i7-1065G7 CPU@1.30GHz

вычислительных ядер: 8/8

4. Результаты выполнения

Для Apple M3 я использовала NEON, для Intel i7 я использовала AVX. Для каждого значения N, для Sequential\_time, Vectorized\_time и Max\_error я выполнил три запуска и взял среднее значение, которое затем занёс в таблицу. И вот результаты:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Процессор | N\_size | Sequential\_time | Vectorized\_time | Max\_error |
| Apple M3 | 512 | 0.134544 | 0.0265929 | 0 |
| Intel i7 | 512 | 0.179683 | 0.0920539 | 979.675 |
| Apple M3 | 1024 | 0.957385 | 0.957385 | 0 |
| Intel i7 | 1024 | 1.57844 | 0.769351 | 1923.5 |
| Apple M3 | 2048 | 16.006 | 4.00603 | 0 |
| Intel i7 | 2048 | 10.4934 | 5.28732 | 3741.73 |

Apple M3 и Intel i7 показали значительное сокращение времени векторизации по сравнению с последовательным временем. Однако при одинаковом размере N, время векторизации на Apple M3 обычно ниже, чем на Intel i7. Это свидетельствует о том, что оптимизация NEON на Apple M3 в данном тесте более эффективна, чем AVX на Intel i7. Кроме того, максимальная ошибка на Apple M3 в любых условиях равна 0, что указывает на более стабильные результаты вычислений, тогда как у Intel i7 наблюдаются некоторые проблемы с точностью.

Это может быть связано с тем, что при использовании векторизации NEON на Apple M3 умножение матриц выполняется с обработкой 4-х чисел с плавающей запятой за раз, в то время как реализация AVX на Intel i7 обрабатывает 8 чисел за раз. В коде для Intel использование AVX может приводить к накоплению ошибок из-за природы операций с плавающей запятой, особенно в функции \_mm256\_fmadd\_ps, что может вызвать дополнительные ошибки округления.