МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Южный федеральный университет»

Институт математики, механики и компьютерных наук им. И.И.Воровича Кафедра алгебры и дискретной математики

Направление подготовки 02.04.02 – Фундаментальная информатика и информационные технологии

ОТЧЕТ ПО ТЕМЕ
«Блочное перемножение матриц»

Выполнил:
магистр 1 года 5 группы
Погорелов А. А.

Описание задачи

Основой эксперимента является задача блочного перемножения двух матриц вещественных чисел. При этом первая матрица является верхне-треугольной и хранится по блочным строкам, а вторая матрица так же является верхне-треугольной, но хранится по блочным столбцам. При программной реализации данной задачи был выбран язык программирования C++.

С помощью данного эксперимента будет выяснено, как на производительность программной реализации влияют разные виды распараллеливания, различные размеры блоков и различные виды матричных данных.

Характеристики компьютера

Эксперименты проводились на компьютере со следующими характеристиками:

Processor: Intel® Core™ i5-8300H CPU @ 2.30GHz; 4 cores; (4.1 GHz TurboBoost)
L3: 8MB cache;
L2: 1MB cache;
L1: 256 KB cache;
RAM: DDR4, 16 GB, clock speed: 2667 MHz;
и с установленной операционной системой Windows 10 PRO x64.

Детали реализации

В ходе экспериментов были получены результаты времени работы программы блочного перемножения матриц при разных размерах блоков для двух типов вещественных данных, при этом тесты производились для двух компиляторов:
1) MSVC v14.16.27023, compiler option: -O2; OpenMP v. 2.0;
2) GCC v. 8.1.0, compiler option: -Ofast; OpenMP v. 4.5;

Также стоить отметить что при реализации алгоритма использовались матрицы смещения. Заранее посчитав смещения одномерного массива необходимые для получения каждого блока, можно поместить их в двумерный массив, где каждый элемент соответствует блоку. Таким образом, избавиться от дорогостоящих вычислений для получения каждого блока. Построение таких структур и другие вспомогательные операции в подсчете времени учтены не были.

Результаты экспериментов

MSVC, Double, миллисекунды

Размер блока	Без параллельности	Паралельность блоков	Параллельность внутри блоков
1	>30000	>60000	>60000
6	1657	766	36903
10	1791	861	10230
15	2024	846	4929
20	2274	1003	2371
24	2490	947	1737
30	2547	1023	1629
36	2859	1207	1085
40	3090	1182	1483
60	3296	1310	1323
72	3655	1383	1296
80	3954	1596	870
96	4342	1756	1073
120	4151	1558	1163
144	4486	1936	1580
160	4789	2173	1381
180	4496	1728	1583
240	5238	2381	1695
360	5703	2123	1372
480	6535	2923	1994
720	8293	4557	2036

MSVC, Float, миллисекунды

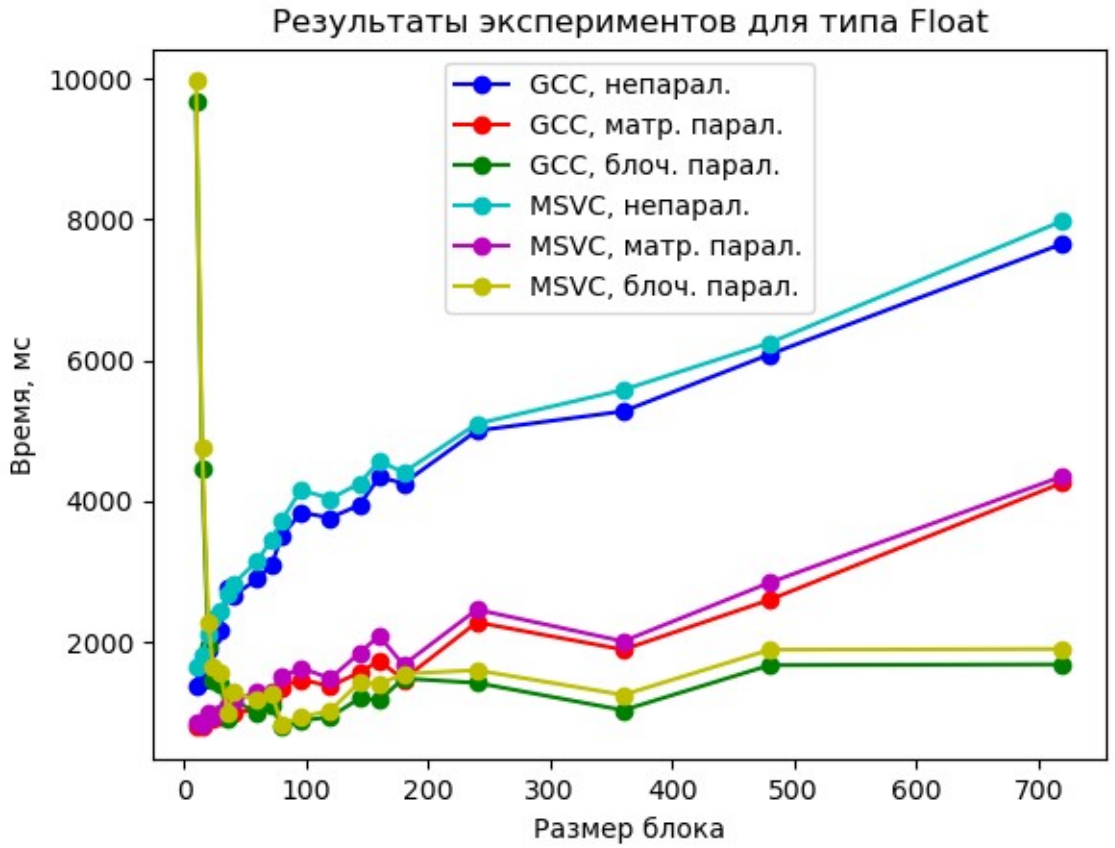
Размер блока	Без параллельности	Паралельность блоков	Параллельность внутри блоков
1	>30000	>60000	>60000
6	1553	752	35894
10	1651	849	9967
15	1822	831	4763
20	2114	996	2284
24	2340	931	1647
30	2455	992	1563
36	2679	1102	989
40	2817	1165	1293
60	3152	1298	1183
72	3459	1302	1274
80	3728	1503	834
96	4162	1637	938
120	4039	1484	1029
144	4253	1849	1435
160	4573	2089	1403
180	4410	1678	1553
240	5101	2463	1602
360	5583	2013	1252
480	6251	2847	1893
720	7982	4357	1904

GCC, Double, миллисекунды

Размер блока	Без параллельности	Паралельность блоков	Параллельность внутри блоков
1	>30000	>60000	>60000
6	1374	731	32914
10	1463	812	9746
15	1563	827	4563
20	1938	985	2084
24	2173	937	1564
30	2273	976	1463
36	2837	1057	956
40	2647	1083	1204
60	2984	1189	1094
72	3174	1254	1174
80	3584	1476	813
96	3984	1563	904
120	3847	1398	987
144	4074	1675	1301
160	4557	1846	1285
180	4371	1564	1584
240	5098	2395	1523
360	5463	1974	1134
480	6173	2703	1785
720	7765	4391	1797

GCC, Float, миллисекунды

Размер блока	Без параллельности	Паралельность блоков	Параллельность внутри блоков
1	>30000	>60000	>60000
6	1302	705	31723
10	1387	798	9674
15	1465	812	4463
20	1904	973	1945
24	2098	924	1465
30	2174	945	1395
36	2784	975	910
40	2653	993	1175
60	2896	1048	1006
72	3096	1174	1103
80	3504	1362	794
96	3842	1473	886
120	3767	1373	940
144	3948	1574	1204
160	4345	1734	1176
180	4256	1473	1485
240	5003	2283	1423
360	5278	1894	1035
480	6087	2603	1674
720	7659	4265	1685



Из графика видно, что блочная параллельность показывает лучшие результаты для обоих компиляторов. Однако, при маленьких размерах блоков, блочная параллельность требует большого количества временных ресурсов, в то время как матричная параллельность работает гораздо быстрее. Возможно, это связано с количеством ресурсов, которые нужны для создания большого количества потоков при блочной параллельности и маленьком размере блока. Также стоит отметить, что из таблиц видно, что для типа данных Float программная реализация работает чуть быстрей. Оба компилятора показывают примерно одинаковые результаты, хотя GCC показывает себя немного лучше, чем MSVC.

Сравнительный анализ

Из результатов экспериментов можно заключить, что параллельное выполнение вычислений для обоих компиляторов работает в 3-3.5 раза быстрее, чем последовательное (при условии рассмотрения оптимальных размеров блоков), а в некоторых случаях ускорение доходит до семикратного.

Параллельность блоков в целом показывает себя лучше матричной параллельности, однако при маленьких размерах блоков абсолютно неэффективна.

Заключение

В ходе проделанной работы были выполнены и проанализированы различные варианты программной реализации блочного перемножения матриц. Для каждого из вариантов был определен оптимальный размер блоков, на вычисление которого затрачивается минимальное время выполнения программы. Критичным для скорости работы программы оказались выбор метода параллельности и размера блока.