Индивидуальная работа по курсу «суперкомпьютеры»

«Исследование блочного умножения матриц»

Выполнил:

Тихонов Алексей

4 курс 2 группа

Ростов-на-Дону – 2018

**Реферат**

Поставлена задача разработать программу осуществляющую блочное перемножение матриц:

***C***=***A*⋅*B***

где матрица ***A*** является блочной верхней треугольной матрицей, а матрица ***B***

– блочной симметричной. Обе блочные матрицы являются согласованными (т.е. разбиение матрицы ***A*** на блоки по столбцам совпадает с разбиением ***B*** по строкам). Это означает, что для таких матриц возможно использовать алгоритм блочного перемножения:

***Cij=Ai1⋅B1j+Ai2⋅B2j+…+Ais⋅Bsj,***

где ***Aij, Bij, Cij*** - блоки матриц ***A, B, C.*** Для перемножения блоков используется стандартный алгоритм перемножения матриц, известный из курса линейной алгебры.

Матрицы ***A, B, C*** должны быть представлены в памяти компьютера в линеаризованном виде, при этом поставлено условие исключения хранения незначащих блоков в памяти. Обе матрицы по условию задачи имеют блочно-строковое размещение.

Разрабатываемая программа должна содержать в себе несколько вариантов перемножения блочных матриц:

1. Последовательное перемножение матриц;
2. Параллельное перемножение блоков;
3. Параллельное перемножение блоков в разных вычислительных ядрах;

Необходимо провести исследование зависимости времени перемножения от размера блока и от метода выполнения расчетов

**Архитектура компьютера**

Для проведения расчетов использовалась ЭВМ со следующими характеристиками:



|  |  |
| --- | --- |
| **Процессор** | Intel Core i7-3770 |
| **RAM** | DDR 3, 2133МГц |
|  |  |

*Таблица 1. Технические характеристики ЭВМ*

Характеристики процессора:



|  |  |
| --- | --- |
| **Семейство** | Ivy Bridge |
| **Количество ядер** | 4 |
| **Количество потоков** | 8 |
| **Базовая частота** | 3500 МГц |
| **Частота в режиме Turbo Boost** | 4200 МГц |
| **Кэш-память L1** | 128 КБ |
| **Кэш-память L2** | 1 МБ |
| **Кэш-память L3** | 8 МБ |



*Таблица 2. Технические характеристики Intel Core i7-3770*

**Параметры запуска приложения**

Язык разработки C++. Использовался компилятор G++. Время работы алгоритма перемножения подсчитывалось с помощью функции omp\_get\_wtime

Запуск приложения осуществлялся для матриц размером 2880 х 2880 при следующих размерах блока: 1, 6, 10, 15, 20, 24, 30, 36, 40, 60, 72, 80, 96, 120, 144, 160, 180, 240, 360, 480, 720.

* методах, использующих параллельное программирование, применялась библиотека Open MP.

**Результаты эксперимента (G++)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Блок |  | Double |  |  | Float |  |  |
|  | I | II | III | I | II | III |  |
| 1 | 492.704 | >1000 | 170.804 | 394.563 | >1000 | 112.45 |  |
| 6 | 31.0419 | 98.1309 | 7.96773 | 30.1815 | 112.114 | 7.29048 |  |
| 10 | 20.9746 | 25.6178 | 5.40317 | 19.8879 | 27.5806 | 5.14797 |  |
| 15 | 17.5528 | 11.2576 | 4.64506 | 17.3327 | 11.1717 | 4.48575 |  |
| 20 | 15.9692 | 7.00313 | 4.21378 | 15.1241 | 6.7845 | 3.95111 |  |
| 24 | 16.9288 | 6.31637 | 4.45223 | 15.5525 | 5.95332 | 4.36902 |  |
| 30 | 16.3219 | 5.82413 | 4.31941 | 15.8256 | 5.57612 | 4.15389 |  |
| 36 | 16.0544 | 5.16226 | 4.26194 | 15.3706 | 4.84441 | 4.04006 |  |
| 40 | 17.3123 | 5.60891 | 4.5761 | 15.9689 | 5.14618 | 4.17053 |  |
| 60 | 17.2252 | 5.10523 | 4.5514 | 16.6265 | 4.78514 | 4.33457 |  |
| 72 | 18.6332 | 5.45296 | 4.8878 | 17.5695 | 5.09706 | 4.59697 |  |
| 80 | 20.2478 | 5.68016 | 5.35827 | 18.7118 | 5.40011 | 4.87884 |  |
| 96 | 22.2482 | 6.09372 | 6.20003 | 20.4519 | 5.61595 | 5.65569 |  |
| 120 | 19.4272 | 5.4355 | 5.11794 | 18.7751 | 5.2318 | 4.88952 |  |
| 144 | 20.9306 | 5.75151 | 5.46397 | 19.9728 | 5.4897 | 5.2079 |  |
| 160 | 22.8095 | 6.2623 | 6.58211 | 21.1489 | 5.75456 | 6.21428 |  |
| 180 | 20.2863 | 5.60903 | 5.38438 | 19.6169 | 5.33857 | 5.12649 |  |
| 240 | 23.5457 | 6.43972 | 6.18661 | 20.5551 | 5.57781 | 5.36592 |  |
| 360 | 22.569 | 6.12762 | 5.92585 | 20.908 | 5.64313 | 5.45426 |  |
| 480 | 25.4676 | 6.86743 | 8.78502 | 23.4861 | 6.26939 | 8.05591 |  |
| 720 | 27.6126 | 7.59788 | 8.81702 | 22.6343 | 6.02284 | 6.2861 |  |
| 2880 | 66.0709 | 40.4537 | 40.4537 | 44.1086 | 19.1472 | 43.8966 |

*Таблица 3. Время работы алгоритмов для компилятора G++. I – последовательный алгоритм. II – параллельное перемножение двух блоков.*

1. *– параллельное перемножение разных пар блоков.*

*Рисунок 1. Зависимость времени исполнения от размера блока. Компилятор G++. Тип double.*

*Рисунок 2. Зависимость времени исполнения от размера блока. Компилятор G++. Тип float.*

**Анализ результатов эксперимента**

Полученные результаты показывают, что алгоритм блочного перемножения матриц, при правильном подборе размера блока, позволяет сократить время выполнения расчетов в 4 раза. Однако, алгоритм II - параллельное перемножение пар блоков, является крайне неэффективным в случае размеров блока 1, 6 и 10

Запуск алгоритма для разных типов данных, демонстрирует нам, вычисления происходят быстрее для типа float, что связано с возможностью размещения большего количества элементов матрицы в кэш-памяти и, соответственно, уменьшению количества кэш-промахов.

Алгоритм III - перемножение блоков в разных вычислительных ядрах, проявляет свою эффективность в случае небольшого размера блоков. Лучший результат был получен при размере блока равному 20. В свою очередь алгоритм II является наиболее эффективным для блоков большого размера – 480 и 720.