

Московский государственный университет имени М. В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра суперкомпьютеров и квантовой информатики

**Отчёт по заданию 1:**

**Разработка параллельной версии решения СЛАУ методом отражения и обратным ходом метода Гаусса**

Аксой Тевфик Огузхан

323 группа

**Постановка задачи**

* **Написать последовательный алгоритм, приводящий матрицу в верхний треугольный вид методом отражения**
* **Найти элементы вектора столбца x решив приведенную в верхний треугольный вид матрицу методом обратного хода Гаусса**
* **Найти и вывести норму невязки**
* **Распараллелить алгоритмы приведения матрицы в верхний треугольный вид методом отражения и обратного хода метода Гаусса.**
* **Измерить времена выполнений параллельных алгоритмов**
* **Запустить программу на вычислительном комплексе IBM Polus с разными размерами вводных данных и количествами OpenMP нитей. Сделать выводы о времени выполнения.**

**Кратко алгоритм решения**

**Метод отражения**

**Обратный ход метода Гаусса**

****

**Компиляция и запуск программы**

***В программе используется такие особенности С++11, как ламбда функции, равномерные распределения чисел, чтобы создать рандомные числа в стиле современного С++ и cинонимы шаблонов и типов (type alias, alias template)***

Программа скомпилировалась с помощью команды **g++ main.cpp -o main -std=c++11 -fopenmp** и запустилась с помощью [**скрипта**](https://gist.github.com/AxoyTO/7e64d2cb535efa2729fc0109fb1818a2), написанная на языке Python. **Скрипт создаёт планировщик ompjob.lsf для размещения работы на очередь по указанным данным и вызывает команду bsub < ompjob.lsf.** Так же можно при запуске скрипта указать число нитей и размеры матриц.

**def make\_script(num\_threads\_list=[1, 2, 4, 8, 16, 20, 32, 64],**

**sizes\_list=[128, 256, 512, 1024]):**

**with open('ompjob.lsf', 'w') as f:**

**header = 'source /polusfs/setenv/setup.SMPI\n' + \**

**'#BSUB -n 1 -q normal\n' + \**

**'#BSUB -W 00:45\n' + \**

**'#BSUB -o result.out\n' + \**

**'#BSUB -e result.err\n'**

**f.write(header)**

**for size in sizes\_list:**

**f.write(**

**f'echo "========= MATRIX: {size} x {size} ========="\n')**

**for num\_threads in num\_threads\_list:**

**f.write(**

**f'echo "============ THREADS: {num\_threads} ============"\n')**

**f.write(**

**f'OMP\_NUM\_THREADS={num\_threads} mpiexec ./main {size} {size} {num\_threads}\n')**

**if \_\_name\_\_ == '\_\_main\_\_':**

**make\_script()**

**os.system('g++ main.cpp -o main -fopenmp -std=c++11')**

**os.system('bsub < ompjob.lsf')**

**Результаты и выводы**

После запуска и обработки программы в Полюсе, файл с результатами ***«result.out»*** на свой компьютер скопировал, дальше для составления графиков по результатам, перенёс данные от результатов в Excel с помощью [скрипта](https://gist.github.com/AxoyTO/a466d586dd16bfbaf02b5037d6ece2ba). В Excel составил графики ускорения в зависимости от количества нитей. В графику данные о Т2 не добавил так как они очень маленькие и по сравнению с Т1 невидимыми становятся.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MatrixSize** | **Threads** | **T1** | **T2** | **Total** | **SPEEDUP** |
| **128x128** | **1** | 2,55403 | 0,00019 | 2,55423 | 1,00 |
| **128x128** | **2** | 1,33581 | 0,00020 | 1,33601 | 1,91 |
| **128x128** | **4** | 0,72668 | 0,00020 | 0,72688 | 3,51 |
| **128x128** | **8** | 0,43829 | 0,00020 | 0,43849 | 5,83 |
| **128x128** | **16** | 0,44007 | 0,00020 | 0,44027 | 5,80 |
| **128x128** | **32** | 0,31886 | 0,00021 | 0,31906 | 8,01 |
| **128x128** | **64** | 0,30478 | 0,00020 | 0,30499 | 8,37 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MatrixSize** | **Threads** | **T1** | **T2** | **Total** | **SPEEDUP** |
| **256x256** | **1** | 39,07 | 0,0007624 | 39,07 | 1,00 |
| **256x256** | **2** | 19,95 | 0,0007665 | 19,95 | 1,96 |
| **256x256** | **4** | 10,50 | 0,0007656 | 10,50 | 3,72 |
| **256x256** | **8** | 5,97 | 0,0007831 | 5,97 | 6,55 |
| **256x256** | **16** | 4,63 | 0,0010098 | 4,63 | 8,43 |
| **256x256** | **32** | 3,74 | 0,0010595 | 3,74 | 10,46 |
| **256x256** | **64** | 3,40 | 0,0011946 | 3,40 | 11,48 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MatrixSize** | **Threads** | **T1** | **T2** | **Total** | **SPEEDUP** |
| **512x512** | **1** | 617,7790 | 0,0030 | 617,7820 | 1,00 |
| **512x512** | **2** | 313,1760 | 0,0031 | 313,1790 | 1,97 |
| **512x512** | **4** | 159,8050 | 0,0030 | 159,8080 | 3,87 |
| **512x512** | **8** | 87,2194 | 0,0031 | 87,2225 | 7,08 |
| **512x512** | **16** | 65,5751 | 0,0033 | 65,5784 | 9,42 |
| **512x512** | **32** | 51,0282 | 0,0036 | 51,0318 | 12,11 |
| **512x512** | **64** | 42,6757 | 0,0036 | 42,6793 | 14,47 |