|  |
| --- |
| МГУ им. М. В. Ломоносова, факультет ВМК |
| Задание 1 Проблемы масштабируемости в OpenMP. Лишние барьеры для синхронизации работы нитей. |
|  |

|  |
| --- |
| Арбузов Николай Романович  группа 423 |

# Проблема

При реализации параллелизма в решении задач добавление лишней синхронизации в виде барьеров OpenMP повышает время выполнения задачи без влияния на качество выполнения программы.

# Алгоритм

В качестве показательного алгоритма для этой проблемы был выбран параллельный алгоритм перемножения матриц:

Проблемный код:

Matrix matmul(Matrix &A, Matrix &B)

{

int size = A.size;

Matrix C(size);

C.generate();

#pragma omp parallel

for (int i = 0; i < size; i++)

{

for (int k = 0; k < size; k++)

{

#pragma omp for nowait

for (int j = 0; j < size; j++)

{

C.data[i \* size + j] += A.data[i \* size + k] \* B.data[k \* size + j];

}

#pragma omp barrier

}

}

return C;

}

Код без проблемы:

Matrix matmul(Matrix &A, Matrix &B)

{

int size = A.size;

Matrix C(size);

C.generate();

#pragma omp parallel

for (int i = 0; i < size; i++)

{

for (int k = 0; k < size; k++)

{

#pragma omp for nowait

for (int j = 0; j < size; j++)

{

C.data[i \* size + j] += A.data[i \* size + k] \* B.data[k \* size + j];

}

}

}

return C;

}

# Компиляция и запуск

Все вычисления производились на машине Polus.

Сама программа написана на языке C++ и состоит из файлов:

* main\_barriers.cpp / main\_no\_barriers.cpp (в зависимости от того, как мы хотим запустить с или без барьеров соответственно)
* matrix.cpp (Реализация структуры Matrix из практикума прошлого семестра)
* matrix.h

Компилировалась с использованием Makefile:

all: main

main\_no\_barriers: \*.cpp \*.h

g++ main\_no\_barriers.cpp matrix.cpp -o prog -std=c++17 -fopenmp

main\_barriers: \*.cpp \*.h

g++ main\_barriers.cpp matrix.cpp -o prog -std=c++17 -fopenmp

omp\_polus\_no\_barriers: \*.cpp \*.h

xlc++ main\_no\_barriers.cpp matrix.cpp -o prog -Wall -std=c++11 -qsmp=omp -fopenmp

omp\_polus\_barriers: \*.cpp \*.h

xlc++ main\_barriers.cpp matrix.cpp -o prog -Wall -std=c++11 -qsmp=omp -fopenmp

clean:

rm -rf ./prog

Запуск производился постановкой в очередь с помощью lsf-файлов вида:

Для 1 и 2 потоков:

#BSUB -n 1

#BSUB -W 00:30

#BSUB -o \"./out\_files/j/i.out\"

#BSUB -e \"./err\_files/j/i.err\"

#BSUB -R \"span[hosts=1]\"

OMP\_NUM\_THREADS=i ./prog j

Для 4, 8, 16 и 32 потоков:

#BSUB -W 00:15

#BSUB -o "./out\_files/j/i.out"

#BSUB -e "./err\_files/j/i.err"

#BSUB -R "affinity[core(M)]"

OMP\_NUM\_THREADS=i

/polusfs/lsf/openmp/launchOpenMP.py ./prog j

Где **i** – количество потоков, на которых будет запускаться программа, **M** – количество ядер (**M = i / 2**), **j** – размер матрицы, на которой будут производиться вычисления.

Оценка результативности изменений проводилась по средством измерения времени с помощью функции omp\_get\_wtime().

# Результаты

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Nthreads** | **MatrixSize** | **Time\_With** | **Time\_Without** | **diff** |
| **0** | 1 | 1000 | 8.096785 | 5.5252485 | 2.5715365000000006 |
| **1** | 1 | 4000 | 58.6966 | 57.4837 | 1.2128999999999976 |
| **2** | 1 | 6000 | 178.965 | 186.478 | -7.513000000000005 |
| **3** | 2 | 1000 | 5.4884275 | 3.0393950000000003 | 2.4490325 |
| **4** | 2 | 4000 | 54.754466666666666 | 179.8091 | -125.05463333333333 |
| **5** | 2 | 6000 | 272.786 | 119.554 | 153.232 |
| **6** | 4 | 1000 | 2.68982775 | 0.50909325 | 2.1807345 |
| **7** | 4 | 4000 | 41.60926666666666 | 30.527875 | 11.081391666666661 |
| **8** | 4 | 6000 | 364.3845 | 100.79984999999999 | 263.58465 |
| **9** | 8 | 1000 | 1.9252175 | 0.32229474999999996 | 1.60292275 |
| **10** | 8 | 4000 | 31.3425 | 17.302125 | 14.040375000000001 |
| **11** | 8 | 6000 | 203.3278 | 56.6797 | 146.6481 |
| **12** | 16 | 1000 | 1.7224875 | 0.9310156666666667 | 0.7914718333333333 |
| **13** | 16 | 4000 | 66.58706666666667 | 37.4879 | 29.09916666666667 |
| **14** | 16 | 6000 | 106.4727 | 35.94486666666667 | 70.52783333333333 |
| **15** | 32 | 1000 | 2.0339 | 0.571595 | 1.4623050000000002 |
| **16** | 32 | 4000 | 39.998333333333335 | 21.2881 | 18.710233333333335 |
| **17** | 32 | 6000 | 102.0655 | 49.1788 | 52.8867 |

Рисунок . Графики времени выполнения программы: Синим цветом с "ошибочным" использованием барьеров, оранжевым цветом без использования барьеров. Графика по порядку для матриц размером 1000, 4000, 6000.

