МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИЛЬВІВСЬКИЙ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА

Лабораторна робота №8

з курсу «Паралельні та розподілені обчислення»

Виконав:
Гуменюк Станіслав
Група Пмі-33с

Оцінка ___
Перевірив:
Пасічник
Т.В.

2024

Завдання: На основі програмно-апаратної архітектури паралельних обчислень CUDA реалізувати один з методів лабораторних 2,3,6,7. Продемонструвати результати матриць різної розмірності та порівняти з результатами відповідної попередньої лабораторної роботи

Програмна реалізація:

Я обрав 2 лабораторну, тому що CUDA власне і використовується в більшості випадків для множення матриць(відмалювання кадрів відеоіграх, робота штучного інтелекту) Паралельний алгоритм написаний на c++

При написанні паралельного алгоритму я зробив два кернели для генерування матриці та самого множення

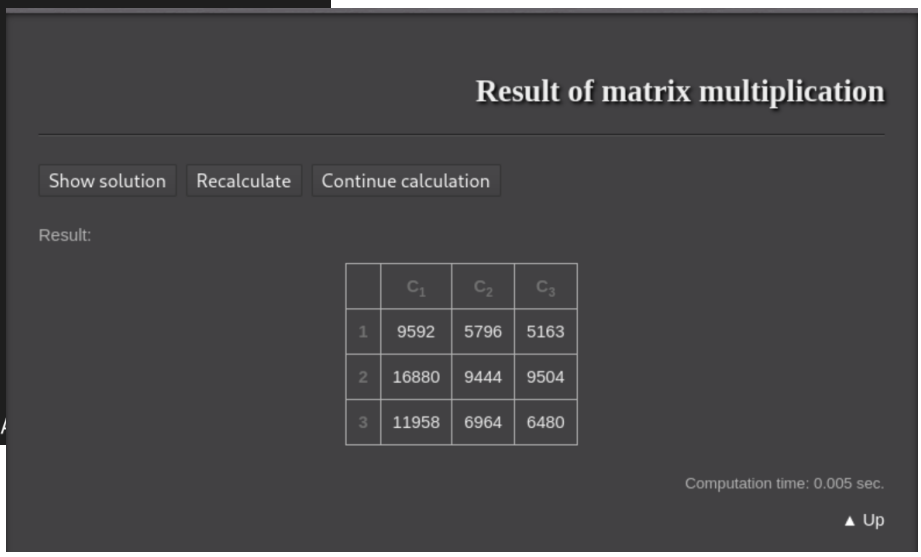
В main я виділяю під все пам'ять, також переводжу матриці в одновимірний вигляд щоб передати їх в пам'ять ГП

Після обчислення я переводжу результат з одновимірного вигляду в двовимірний і друкую

Також треба не забути вивільнити пам'ять, бо компілятор не робить цього автоматично

Для початку маленька матриця для перевірки чи робочий алгоритм:

```
sonorma@hellcat:~/Documents/Study/Parallel/Lab8$ ./par
Enter the size of the square matrix: 3
Matrix A:
50 77 11
94 82 40
59 51 40
Matrix B:
99 31 74
47 45 14
93 71 35
Part of Result Matrix:
9592 5796 5163
16880 9444 9504
11958 6964 6480
Time taken for matrix multiplication (CUDA
```



Паралельний алгоритм з 3 лабораторної роботи при розмірності 1000:

```
Enter the size of the matrix:
n: 1000
m: 1000
p: 1000
Threads k: 16
Time elapsed with 16 threads: 1351ms
Press any key to exit...
```

Паралельний алгоритм з використанням CUDA при розмірності 1000:

```
sonorma@hellcat:~/Documents/Study/Parallel/Lab8$ ./par
Enter the size of the square matrix: 1000
Time taken for matrix multiplication (CUDA): 8.97389 ms
```

Старий алгоритм при розмірності 5000:

```
Enter the size of the matrix:
n: 5000
m: 5000
p: 5000
Threads k: 16
Time elapsed with 16 threads: 184279ms
Press any key to exit...
```

Новий алгоритм при 5000:

```
sonorma@hellcat:~/Documents/Study/Parallel/Lab8$ ./par
Enter the size of the square matrix: 5000
Time taken for matrix multiplication (CUDA): 1200.75 ms
```

Старий алгоритм при 10000:

```
Enter the size of the matrix:
n: 10000
m: 10000
p: 10000
Threads k: 16
Time elapsed with 16 threads: 1877203ms
Press any key to exit...
```

Новий алгоритм при 10000:

```
sonorma@hellcat:~/Documents/Study/Parallel/Lab8$ ./par
Enter the size of the square matrix: 10000
Time taken for matrix multiplication (CUDA): 13909.4 ms
```

Новий алгоритм при 17000:

```
sonorma@hellcat:~/Documents/Study/Parallel/Lab8$ ./par
Enter the size of the square matrix: 17000
Time taken for matrix multiplication (CUDA): 110520 ms
```

Висновок: Як ми бачимо, прискорення є дуже значним. Враховуючи те, що матриці 10 на 10 тисяч стара програма обчислювала 30 хвилин, то далі не було сенсу запускати її для більших. А от паралельний алгоитм з використанням CUDA впорався з матрицями 17 на 17 тисяч за менше ніж дві хвилини. Отже можна чітко сказати, що проводити такі обчислення варто саме на ГП, адже не дивлячись на назву ГРАФІЧНИЙ процесор, він був задуманий саме для великої кількості паралельних обчислень.