Теория Параллелизма

Отчет
Уравнение теплопроводности
(cublas + openacc)

Выполнил 21932, Бабенко Егор Степанович

18.03.2022

Цель работы

Реализовать решение уравнение теплопроводности методом Якоба (пятиточечный шаблон) для двумерной сетки. Произвести профилирование программы для GPU и оптимизацию кода. Произвести сравнение времени работы на CPU и GPU. Для CPU использовалась программа из прошлого задания.

Используемый компилятор: *pgc++*

Для компиляции версии с cublas использовалось:

- pgc++ t3.cpp -o t3_GPU.pg -fast -acc=gpu -O2 -D OPENACC__ -Mcudalib=cublas

Для дополнительной профилировки: -D NVPROF_

Используемый профилировщики: *nvprof, nsys*

Nsys использовался с OpenACC trace, cuBLAS trace, а также NVTX trace.

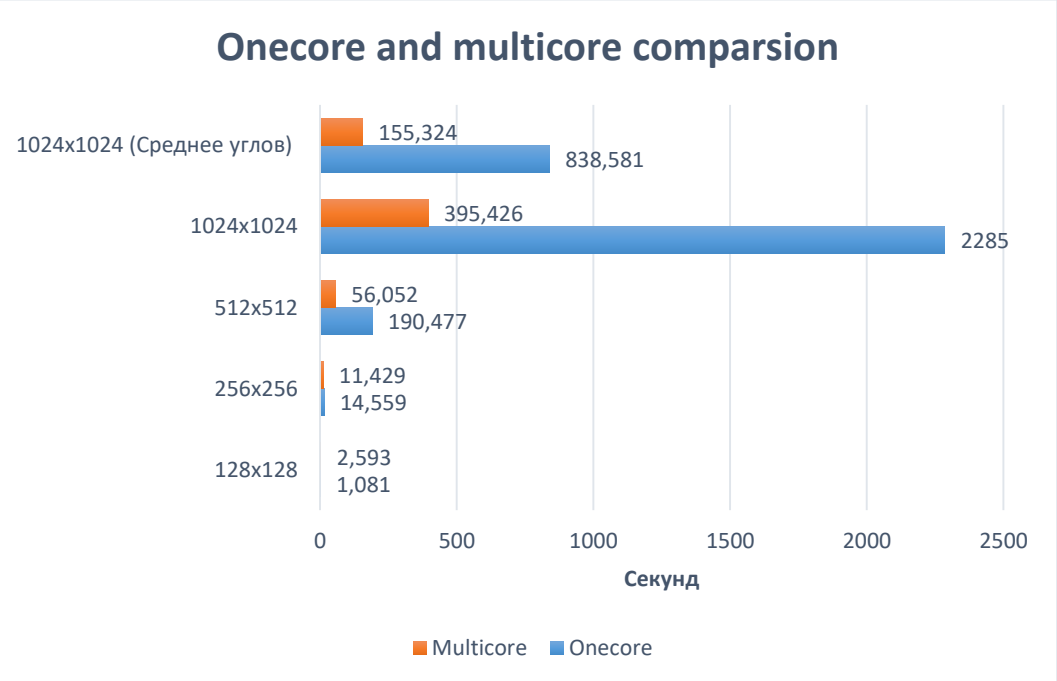
Замер времени работы всей программы производился с помощью команды time. Время высчитывалось как среднее между пятью запусками.

CPU-onecore

Размер сетки	Время выполнения (сек)	Ошибка	Количество итераций
128x128	1.081	9.9998e-07	30074
256x256	14.559	9.9993e-07	102885
512x512	190.477	9.99984e-07	339599
1024x1024	2285.354	1.36929e-06	1000000
1024x1024 *(среднее значение углов сетки как стартовое значение)*	838.581	9.99993e-07	364620

CPU-multicore

Размер сетки	Время выполнения (сек)	Ошибка	Количество итераций
128x128	2.593	9.9998e-07	30074
256x256	11.429	9.9993e-07	102885
512x512	56.052	9.99984e-07	339599
1024x1024	395.426	1.36929e-06	1000000
1024x1024 *(среднее значение углов сетки как стартовое значение)*	155.324	9.99993e-07	364620

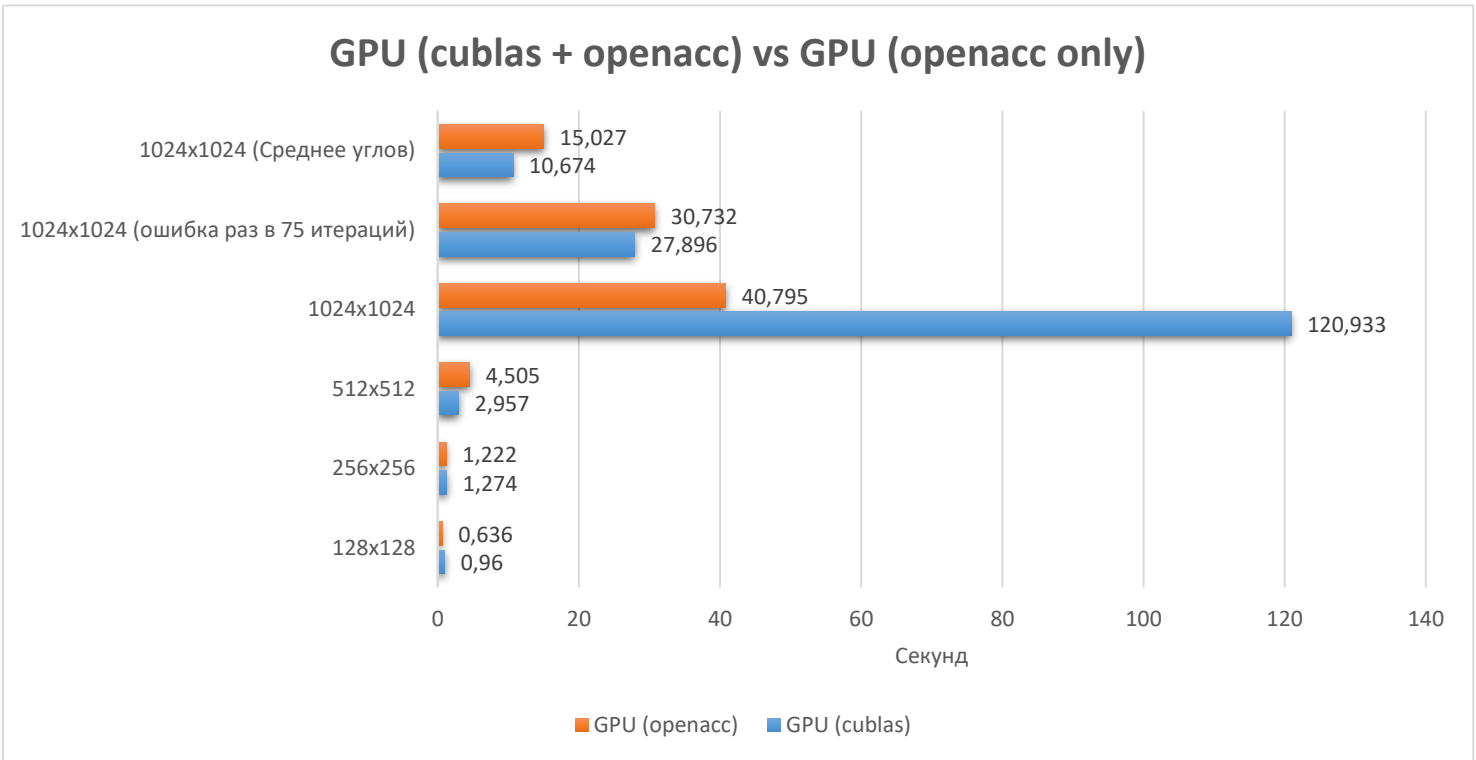


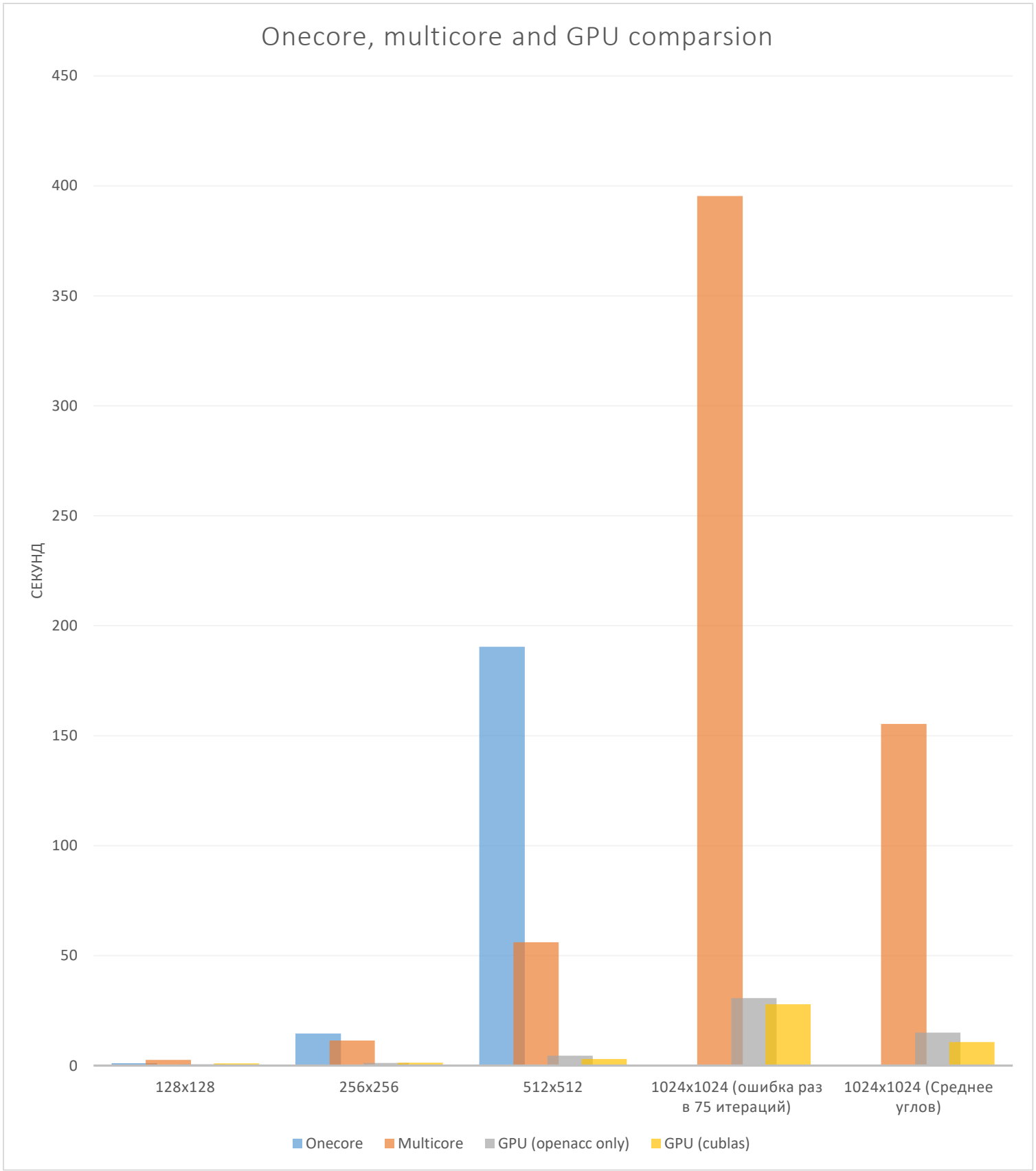
GPU-optimized (openacc only)

Размер сетки	Время выполнения (сек)	Ошибка	Количество итераций
128x128	0.636	9.93271e-07	30096
256x256	1.222	9.98489e-07	102904
512x512	4.505	9.99134e-07	339644
1024x1024 *(расчет ошибки каждую итерацию*	40.795	1.36929e-06	1000000
1024x1024 *(расчет ошибки каждые 75 итераций)*	30.732	1.36929e-06	1000000
1024x1024 *(среднее значение углов сетки как стартовое значение)*	15.027	9.99663e-07	364648

GPU-optimized (cublas)

Размер сетки	Время выполнения (сек)	Ошибка	Количество итераций
128x128	0.960	9.93271e-07	30096
256x256	1.274	9.98489e-07	102904
512x512	2.957	9.99134e-07	339644
1024x1024 *(расчет ошибки каждую итерацию*	120.933	1.36973e-06	1000000
1024x1024 *(расчет ошибки каждые 75 итераций)*	27.896	1.36973e-06	1000000
1024x1024 *(среднее значение углов сетки как стартовое значение)*	10.674	9.99663e-07	364648

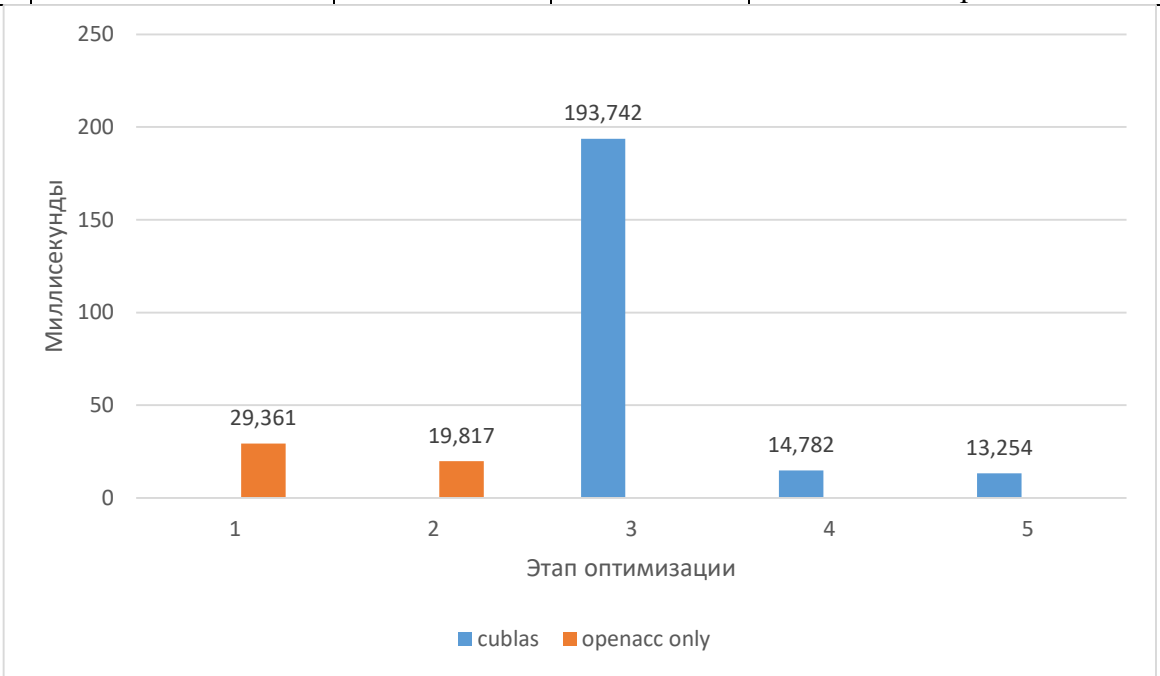



Onecore, multicore and GPU comparsion
450
400
350
300
250
200
150
100
50
0
СЕКУНД
128x128
256x256
512x512
1024x1024 (ошибка раз в 75 итераций)
1024x1024 (Среднее углов)
Onecore
Multicore
GPU (openacc only)
GPU (cublas)

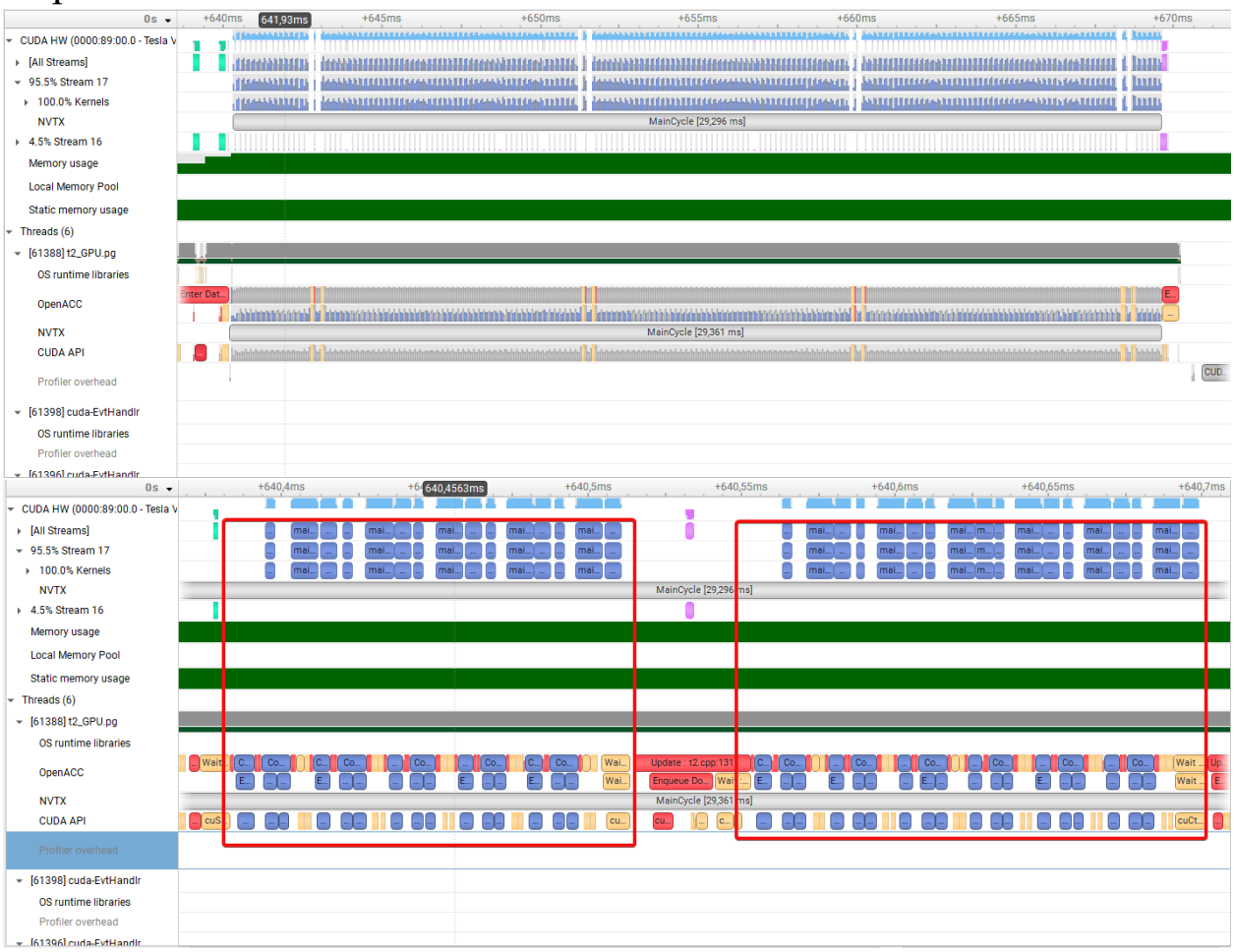
Этапы оптимизации кода на GPU (cublas)

Этапы оптимизации на сетке 512x512

№	Время выполнения основного цикла	Ошибка	Кол-во итераций	Комментарий
1	29,361 мс	0.0105525	1000	Код написанный только с использованием openacc (reduction выполнено через прагмы openacc, результат прошлого задания)
2	18,817	0.0106815	1000	Код написаный только с использованием openacc (перерасчет ошибки каждые 75 итераций)
3	193,742 мс	0.0105524	1000	Выполнение расчета ошибки с помощью cublas (перерасчет каждую итерацию)
4	14,782 мс	0.0140183	1000	Перерасчет ошибки каждые 250 итераций. Значение подобрано эмпирически на основе данных профилировщика, как оптимальное для разных размеров сеток и с учетом времени на синхронизацию потоков.
5	13,254 мс	0.0106815	1000	При обновлении ошибки перенос на CPU только одного элемента. Обновление ошибки каждые 75 итераций.

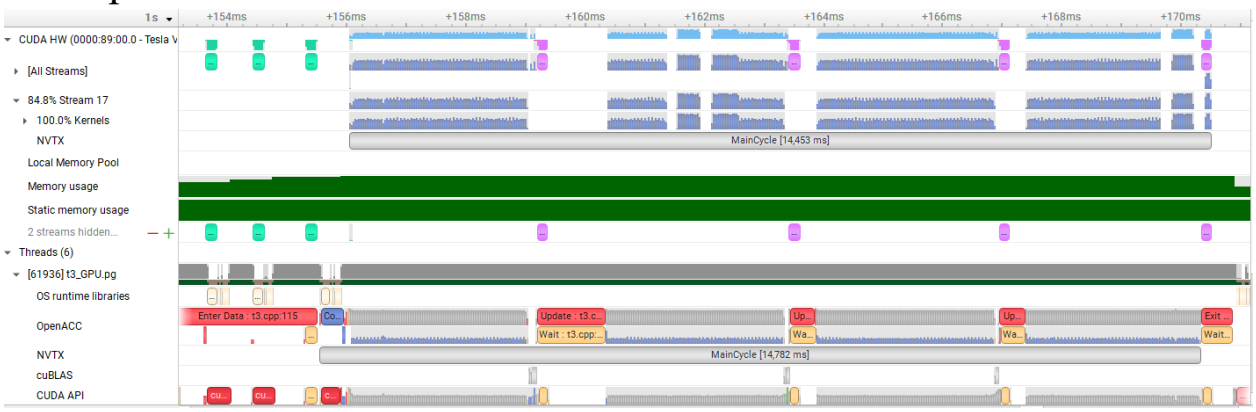


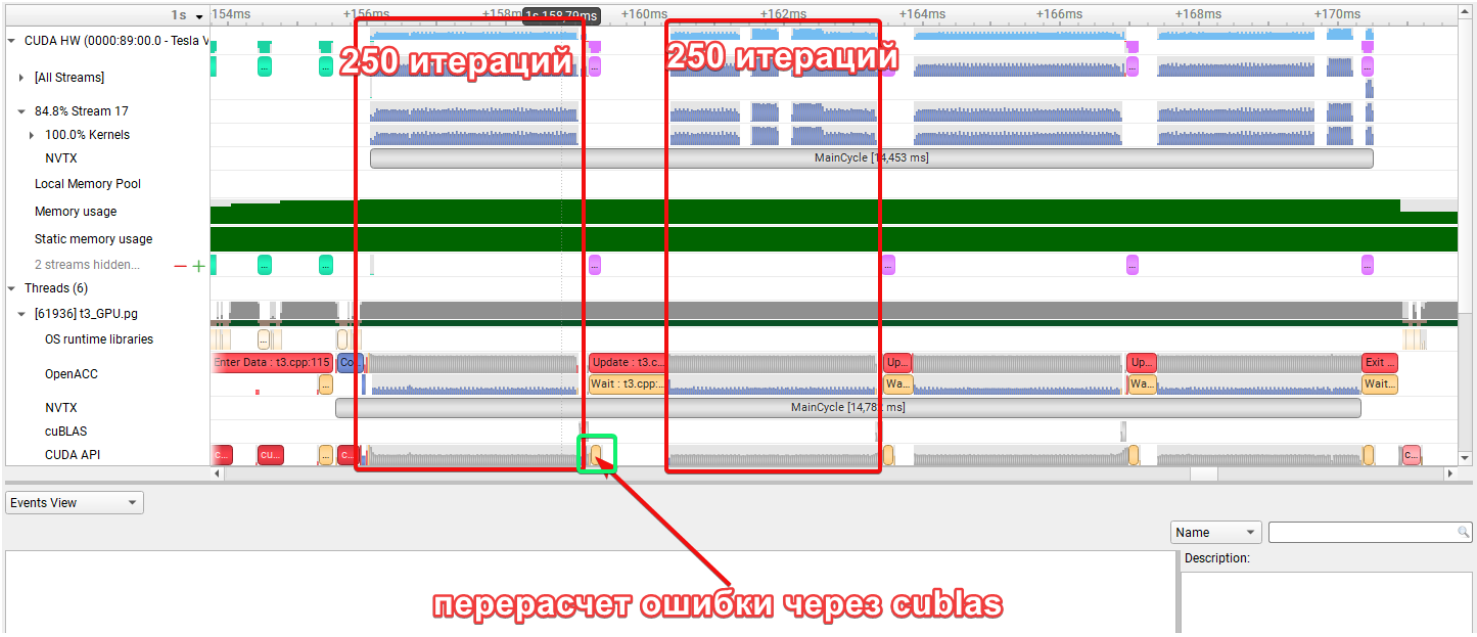
Первый этап:



Третий этап

Четвертый этап





Пятый этап

Вывод:

Используя cublas мы можем добиться меньшего времени за счет того что производим синхронизацию вычислительных потоков как можно реже, а само вычисление ошибки производим достаточно быстро. В итоге получаем время вычисление уравнения на маленьких сетках меньше, чем на CPU, за счет грамотного использования ресурсов видеокарты и минимизации синхронизаций данных между CPU/GPU и вычислительными потоками.

Github: https://github.com/JooudDoo/Parallelism-Tasks