

Теория параллелизма

Отчет

Решение уравнения теплопроводности с использованием метода Якоби

Выполнила Новикова Полина Павловна, гр. 23931
20.05.2025

Цель работы: Реализовать решение уравнения теплопроводности в двумерной области с использованием разностной схемы (пятиточечный шаблон) на равномерных сетках. Программа должна учитывать линейную интерполяцию на границах и заданные значения в углах, ограничивать точность до 10^{-6} и максимальное число итераций до 1006. Реализация должна быть на C++ с использованием OpenACC для переноса на GPU. Необходимо сравнить производительность на CPU и GPU, провести профилирование и оптимизацию кода.

Компиляторы:
nvcc++ 23.11-0

Визуализатор параллельного кода:
NVIDIA Nsight Systems

Инструмент для измерения времени работы:
chrono.

Выполнение на CPU (по факту добавления критических оптимизаций)

CPU-onecore

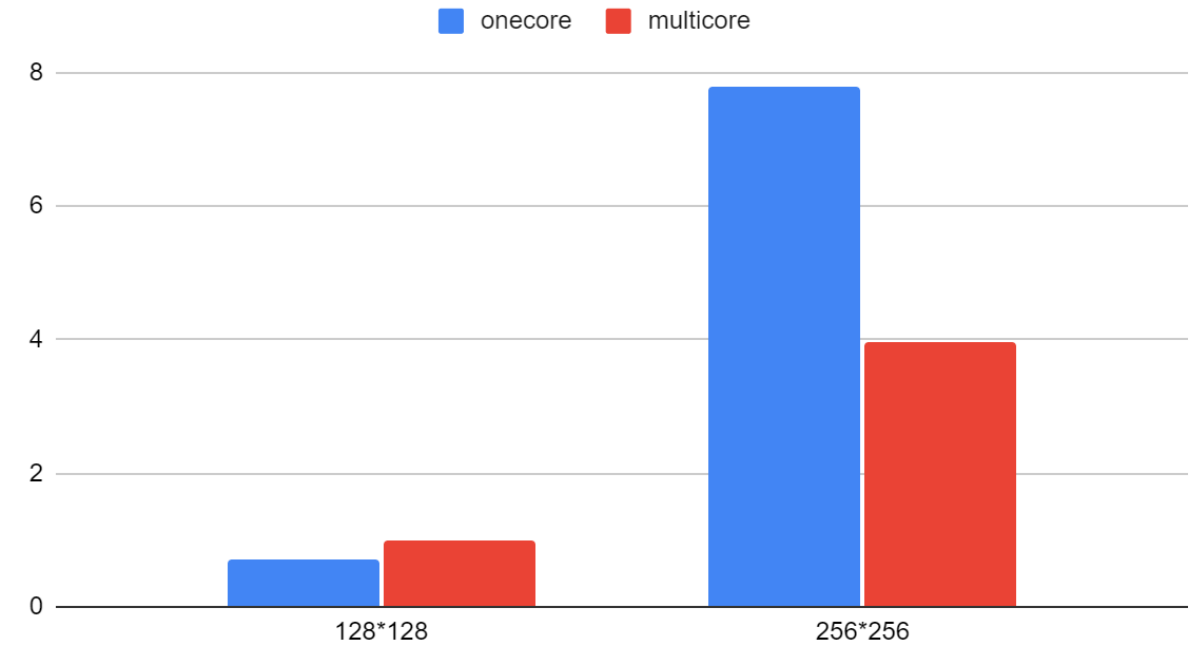
Размер сетки	Время выполнения, сек	Точность	Количество итераций
128*128	0.722	1e-6	40000
256*256	7.799	1e-6	110000
512*512	94.848	1e-6	340000
1024*1024	1273.686	1e-6	1000000

CPU-multicore

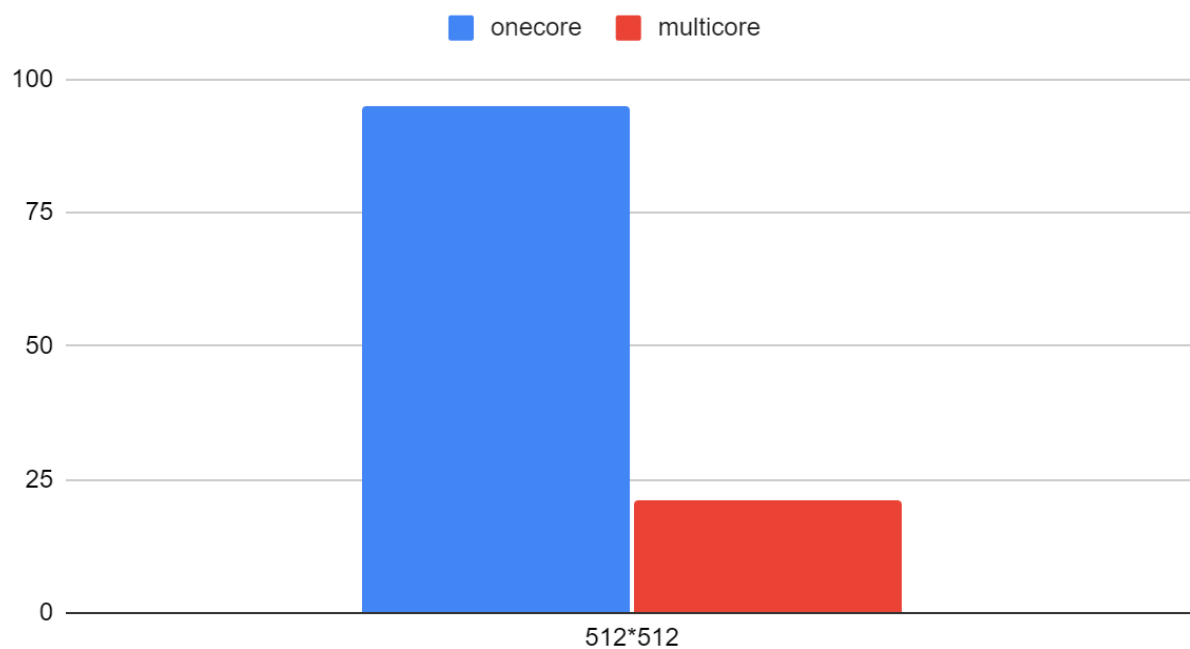
Размер сетки	Время выполнения, сек	Точность	Количество итераций
128*128	0.983	1e-6	40000
256*256	3.952	1e-6	110000
512*512	21.096	1e-6	3400000
1024*1024	194.531	1e-6	1000000

Диаграмма сравнения времени работы CPU-onecore и CPU-multicore

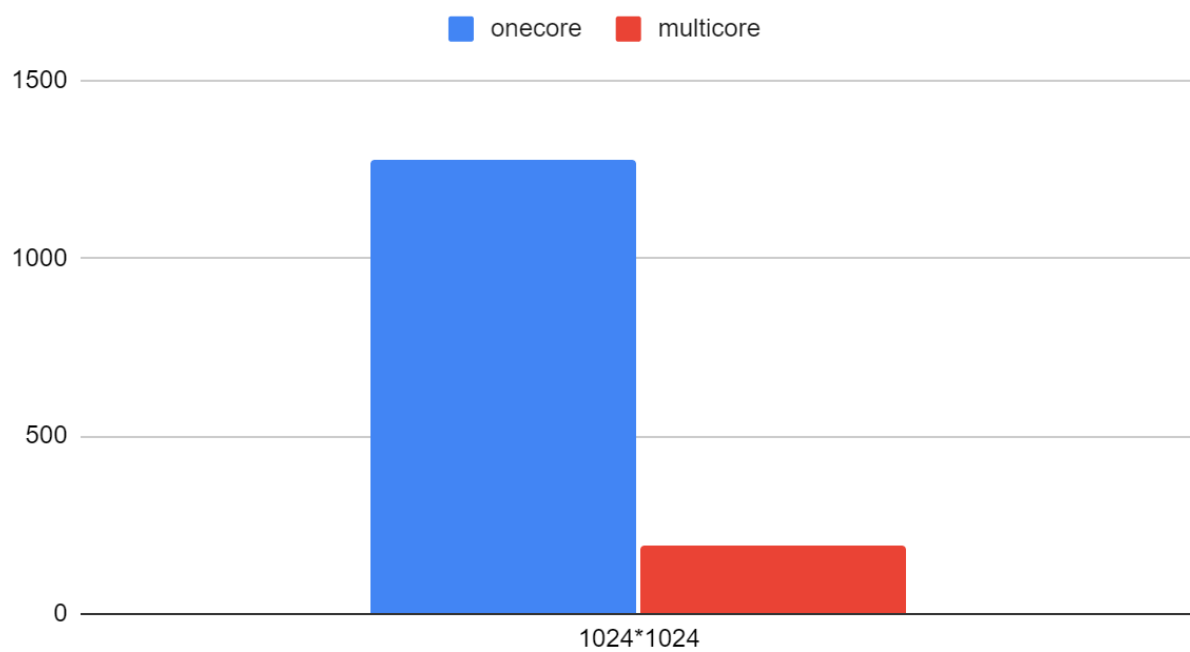
Сравнение onecore и multicore



Сравнение onecore и multicore



Сравнение onecore и multicore

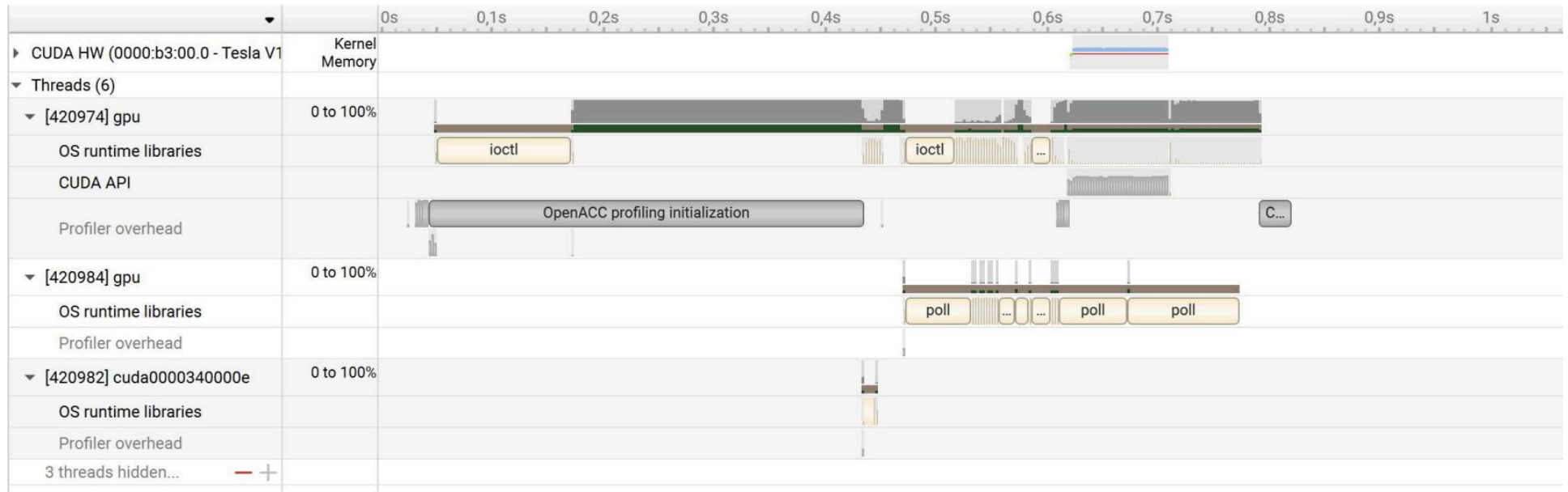


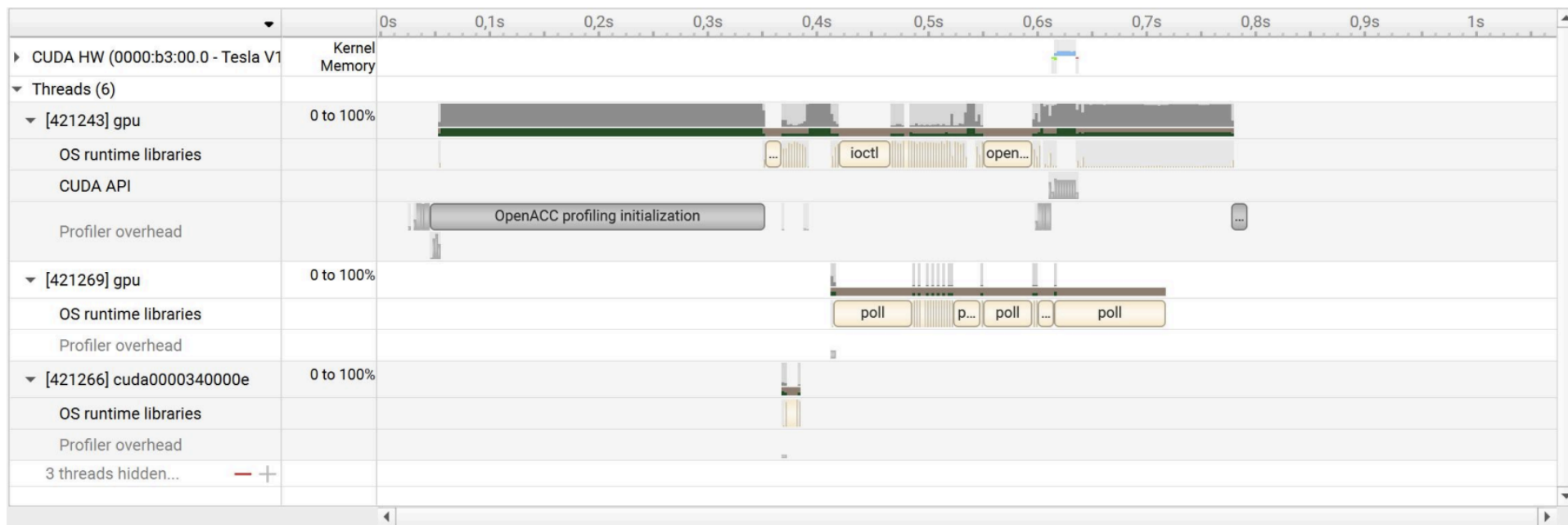
Выполнение на GPU

Измерения на матрице 1024*1024 с указанием оптимизаций для ускорения

№	Время выполнения, сек	Точность	Описание решения
1	98.902	1e-6	Без применения оптимизаций.
2	36.998	1e-6	Убран вывод промежуточного результата по окончании каждой итерации.
3	32.938	1e-6	Замена swap на temp через указатели.

Визуализация в nsys-ui (5000 итераций)





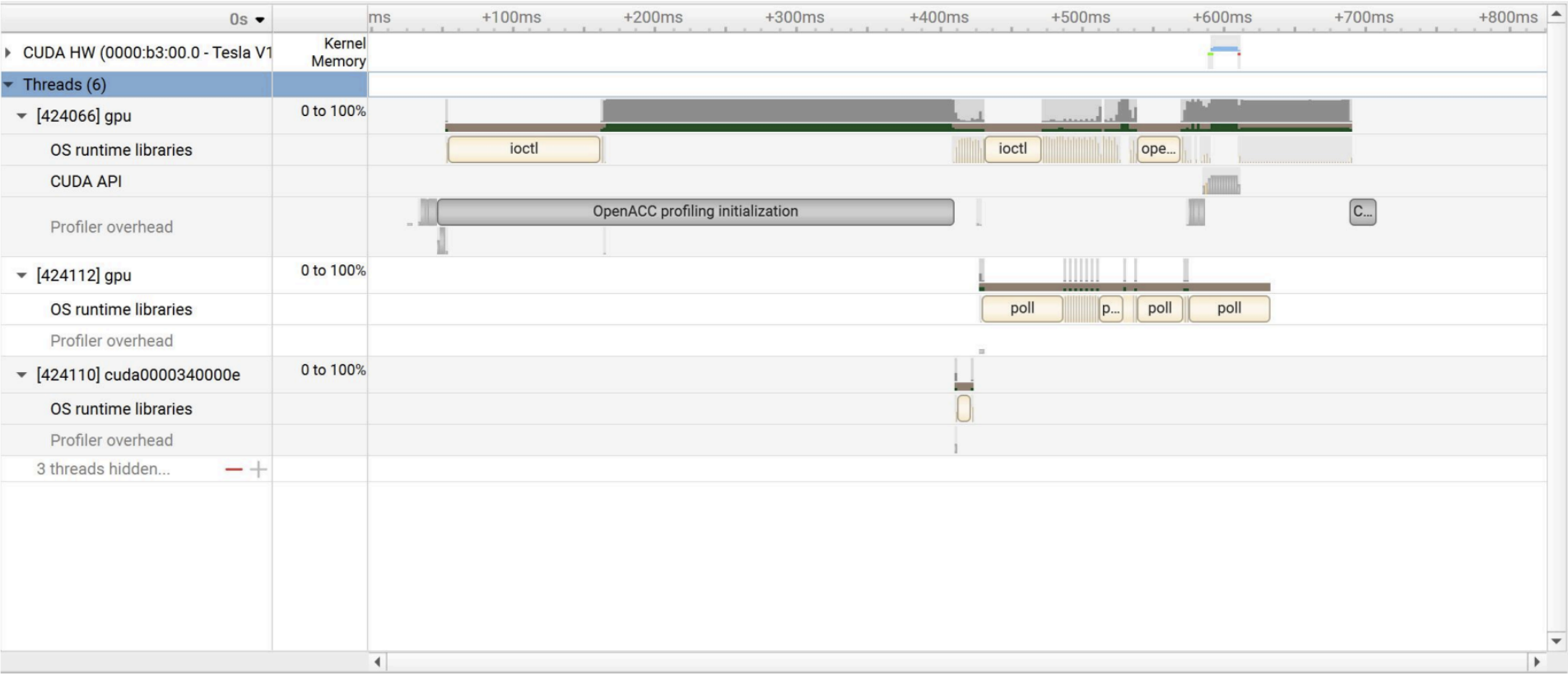
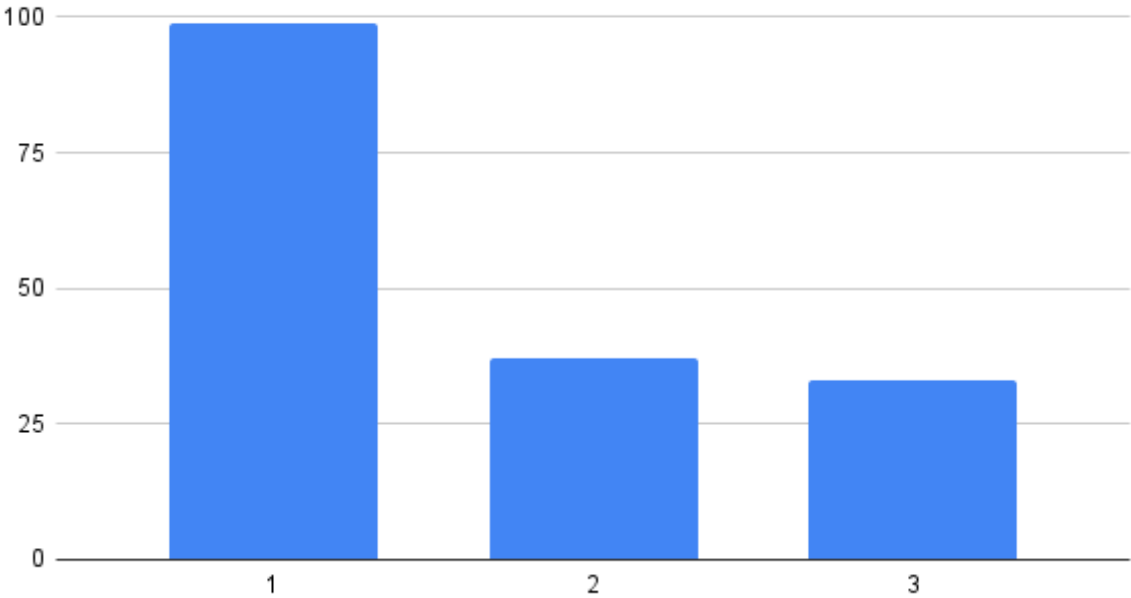


Диаграмма оптимизации

Диаграмма оптимизации (в сек.)

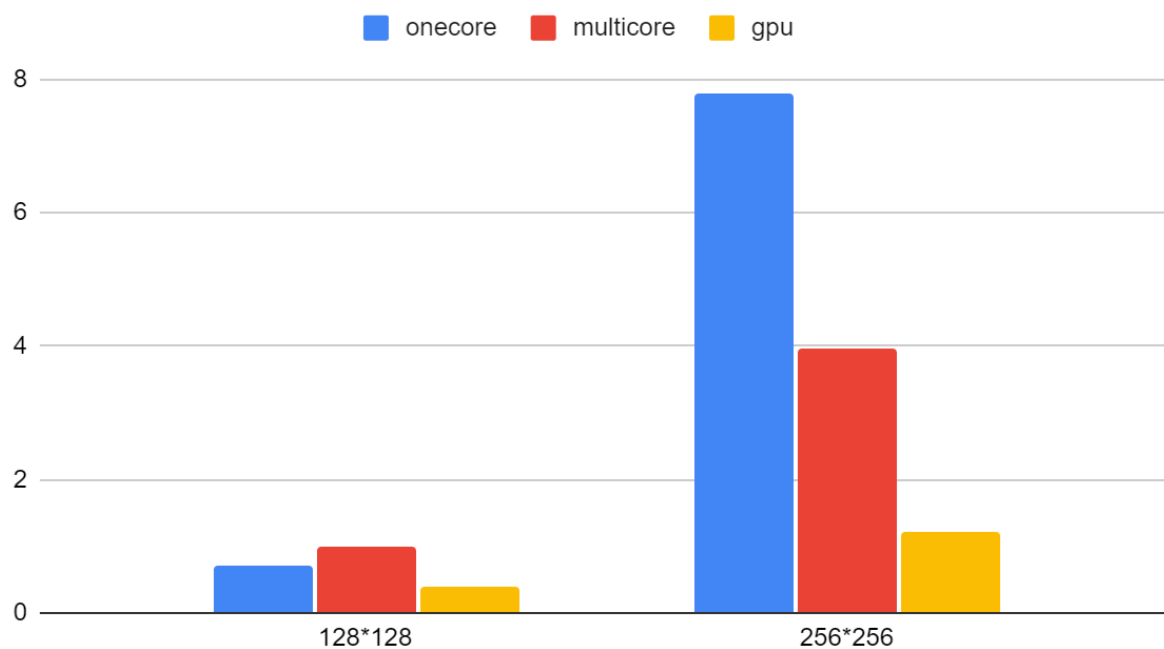


Измерения оптимизированного варианта на GPU

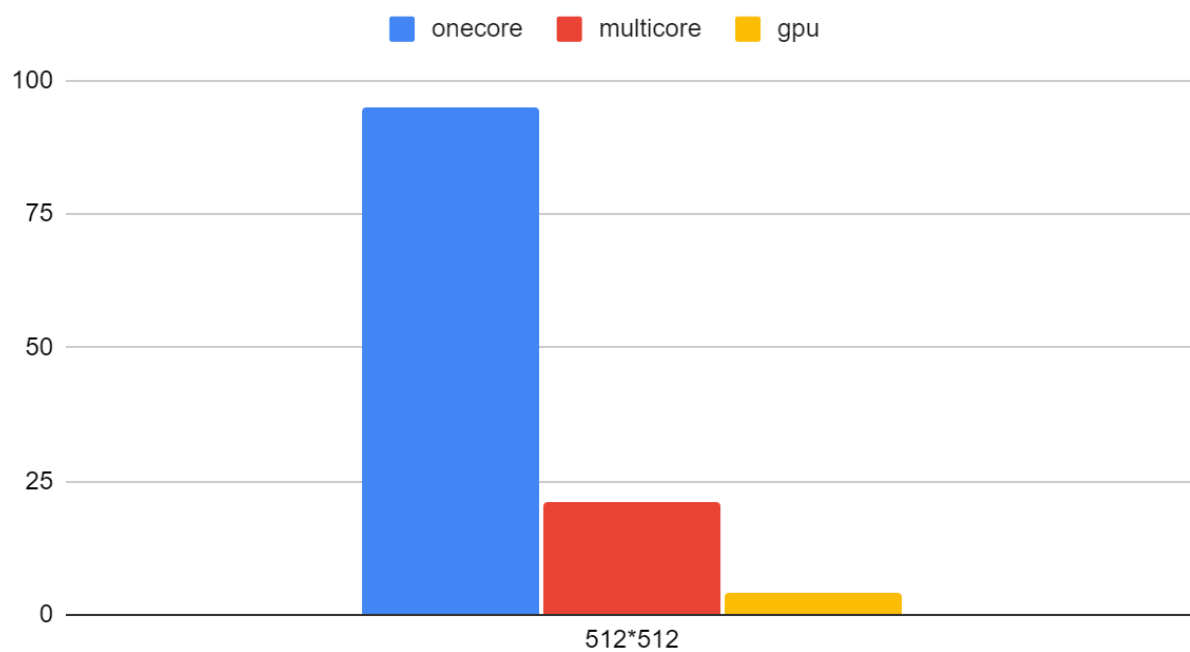
Размер сетки	Время выполнения, сек	Точность	Количество итераций
128*128	0.402	1e-6	40000
256*256	1.205	1e-6	110000
512*512	4.304	1e-6	340000
1024*1024	32.932	1e-6	1000000

Диаграмма сравнения времени работы CPU-onecore, CPU-multicore, GPU- optimized для разных размеров матриц

Сравнение onecore, multicore и gpu



Сравнение onecore, multicore и gpu



Сравнение onecore, multicore и gpu

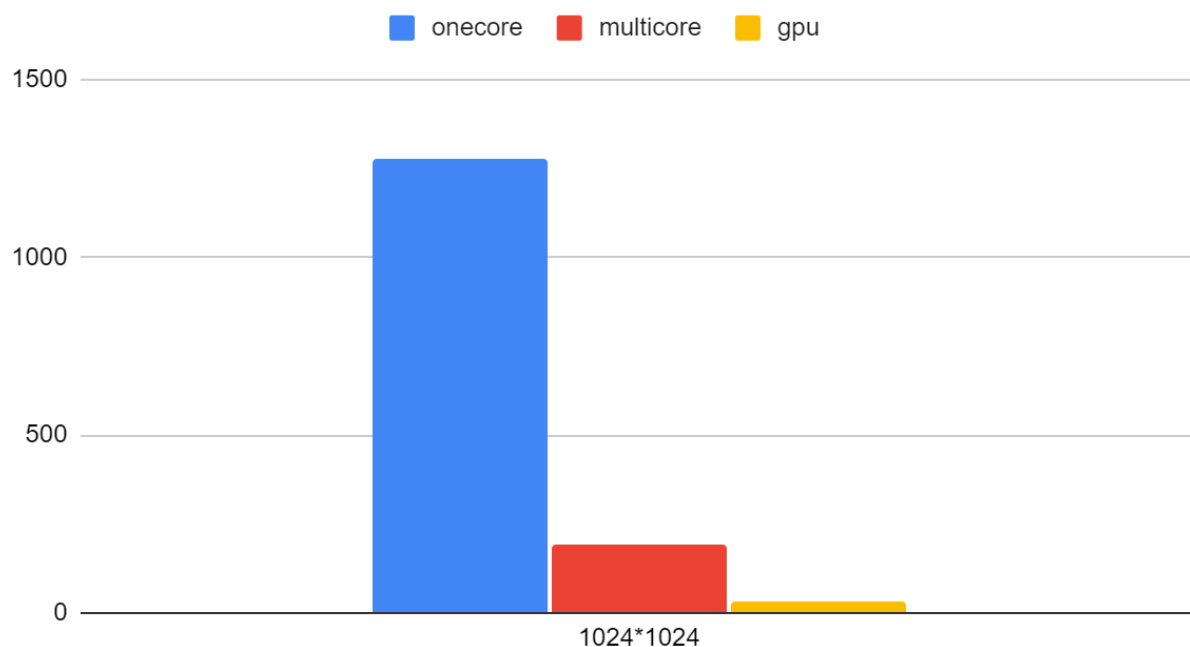
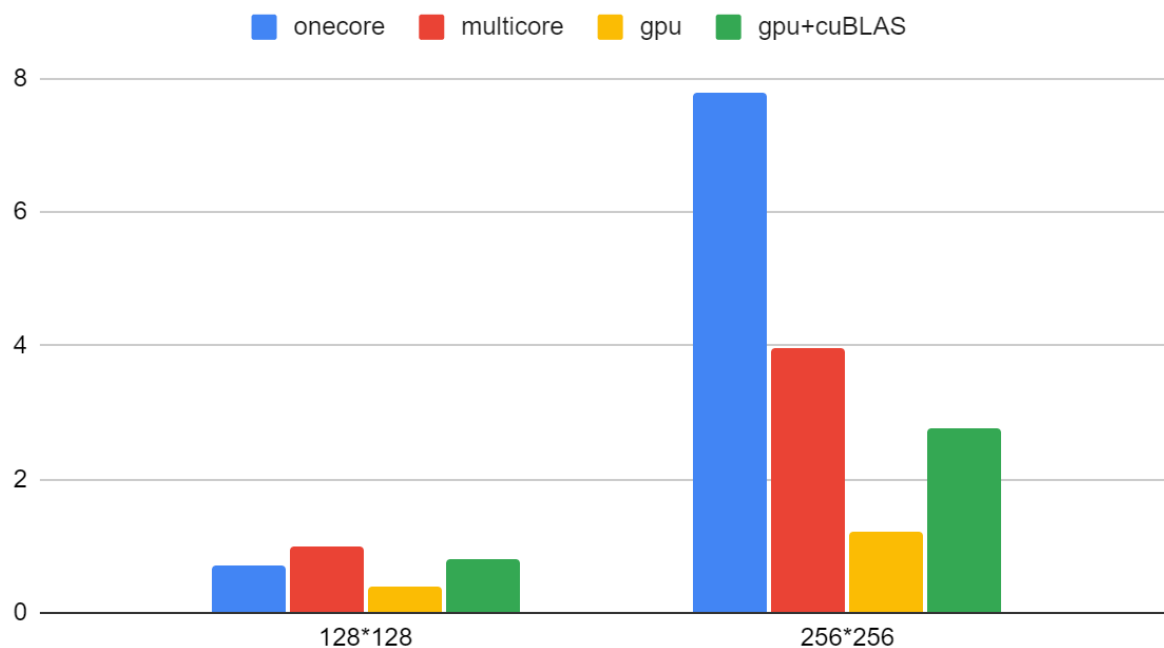
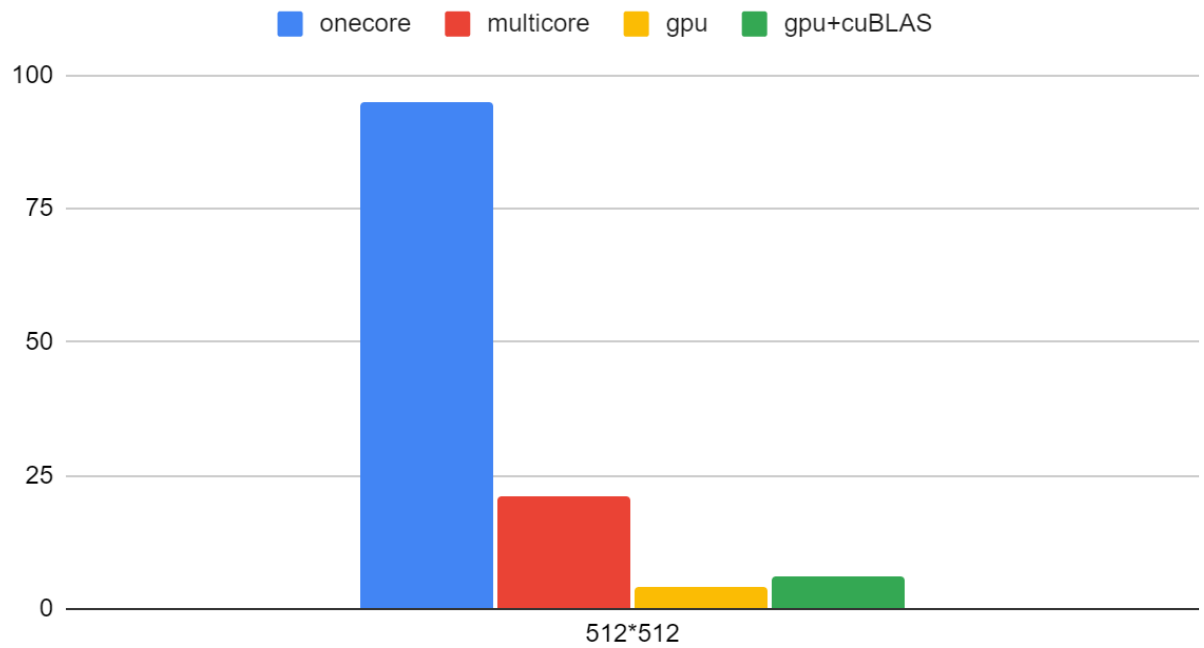


Диаграмма сравнения времени работы CPU + CUBLAS

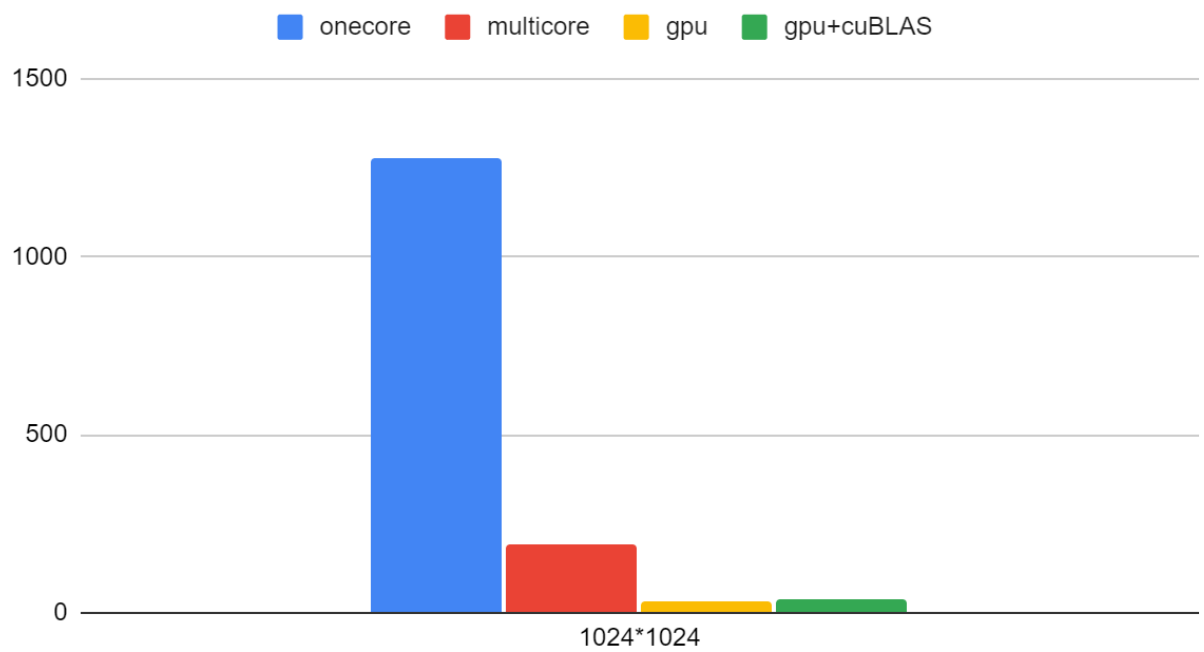
Сравнение onecore, multicore, gpu и gpu + cuBLAS



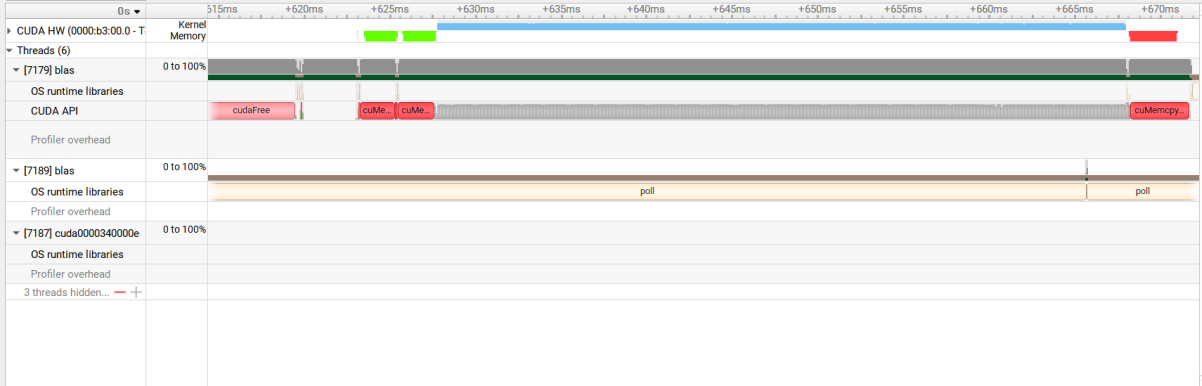
Сравнение onecore, multicore, gpu и gpu + cuBLAS



Сравнение onecore, multicore, gpu и gpu + cuBLAS



Размер сетки	Время выполнения, сек	Точность	Количество итераций
128*128	0.799	1e-6	40000
256*256	2.761	1e-6	110000
512*512	6.229	1e-6	340000
1024*1024	35.409	1e-6	1000000



Вывод

Использование нескольких ядер CPU позволяет значительно увеличить производительность, в то время как на любых матрицах лучше всего себя показывает GPU. Разница в работе между программами, где ошибка ищется через директивы openACC или cuBLAS незначительна.