Теория параллелизма

Задание 6, 7 – OpenACC & cuBLAS

Цель работы

Целью данной работы является исследование производительности “Задачи о распределении тепла в пластинке”. Сравнение проводится между реализацией на OpenACC и cuBLAS с использованием GPU, а также различных вычислительных сред (CPU, Multicore, GPU).

Используемый компилятор

Компилятор: pgc++ (NVIDIA HPC SDK)

Используемый профилировщик

Профилировка производительности выполнялась с использованием замеров времени выполнения на основе стандартной библиотеки C++ (<chrono>).

Также для GPU-реализаций может использоваться pgprof или nsight-systems, однако в данной работе акцент сделан на количественное сравнение по времени выполнения.

Как производили замер времени работы

Замеры времени проводились с использованием высокоточного таймера из стандартной библиотеки C++11:

```
using namespace std::chrono;
auto start = high_resolution_clock::now();
auto end = high_resolution_clock::now();
duration<double> diff = end - start;
cout << "Время выполнения: " << diff.count() << " секунд." << endl;
```

Это позволяет получать точные оценки времени исполнения независимо от реализации.

Выполнение на CPU (OpenACC – Host)

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	442.9078	0.06329410	1000
256*256	6978.5844	0.06329410	1000

Выполнение на CPU (OpenACC – Multicore)

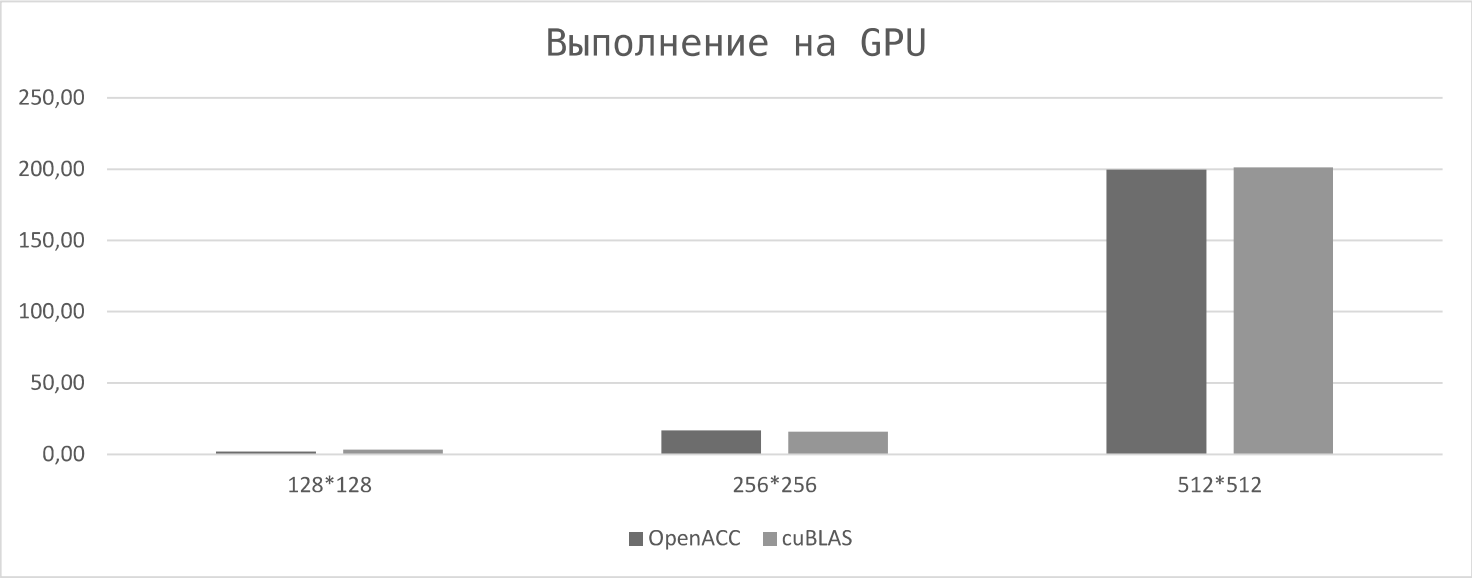
Размер сетки	Время выполнения	Точность	Количество итераций
128*128	27.7537	0.06329410	1000
256*256	394.6631	0.06329410	1000

Выполнение на CPU

Результаты, полученные при запуске программы на CPU (режимы Host и Multicore), показали значительно худшую производительность по сравнению с GPU-реализациями. Особенно это заметно при увеличении размеров матрицы.

Такое существенное падение производительности объясняется высокой вычислительной сложностью метода при использовании матрицы большого размера и отсутствием эффективной распараллеливания или оптимизации кэширования в однопоточном режиме. Даже многопоточность через OpenACC не даёт сравнимого с GPU ускорения из-за ограниченной пропускной способности памяти и меньшего числа доступных вычислительных ядер по сравнению с GPU.

Выполнение на GPU



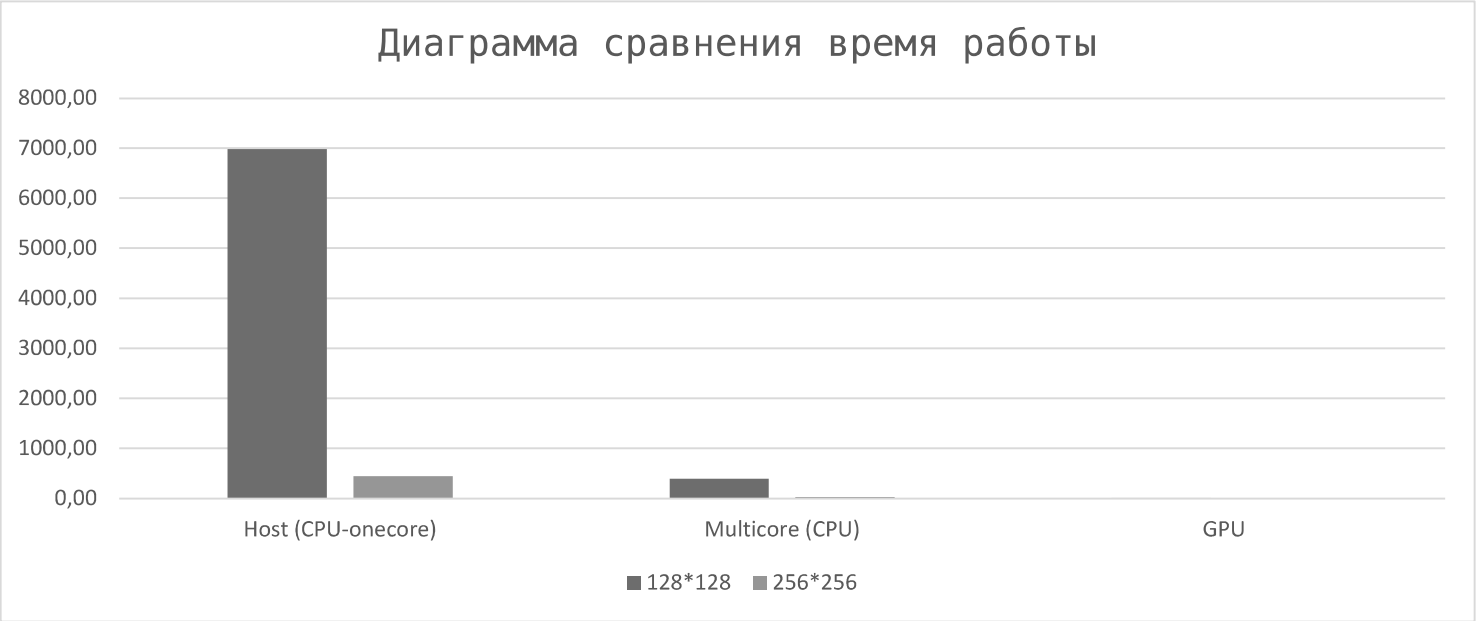
Производительность на GPU:

Для задач малого и среднего масштаба (например, 128×128 и 256×256), cuBLAS и OpenACC показывают сопоставимые результаты. Например:

- 128×128: OpenACC — 1.84 сек, cuBLAS — 3.23 сек.
- 256×256: OpenACC — 16.70 сек, cuBLAS — 15.82 сек.

При увеличении размера задачи (512×512) различие становится незначительным: OpenACC — 199.64 сек, cuBLAS — 201.21 сек.

Диаграмма сравнения время работы (Host/Multicore/GPU)



Выводы

На основании результатов экспериментов можно сделать следующие выводы:

- OpenACC (GPU) показал наилучшую производительность на меньших и средних размерах матриц.
- cuBLAS (GPU) показывает сравнимую или чуть худшую производительность на больших размерах, но выигрывает по удобству использования за счёт высокоуровневых оптимизированных функций.
- Host (CPU) и Multicore демонстрируют значительно более низкую производительность, особенно при увеличении размера задачи.