Теория параллелизма

Отчет

“Уравнение теплопроводности на нескольких GPU”

Выполнила: Володина Софья, г. 21932

Дата: 31 мая 2023г

Цели работы:

Программа должна быть реализована на CUDA. Распараллеливание на несколько GPU должно производиться с использованием MPI. Операцию редукции(подсчет максимальной ошибки) в рамках одного MPI процесса реализовать с использованием библиотеки CUB. Подсчет глобального значения ошибки, обмен граничными условиями реализовать с использованием MPI.

Сравнить скорость работы и масштабирование для разных размеров сеток на разном количестве графических процессоров (1, 2, 4). Обратить внимание на корректность отображения MPI процессов на ядра центрального процессора и корректный выбор графических процессоров. Результаты, их анализ и выводы представить в виде отчета.

Используемый компилятор: mpic++

Используемый профилировщик: nsys

Как производили замер времени работы: time при запуске программы с помощью mpiexec, библиотека <chrono>

Данные из предыдущих задач.

CPU-onecore

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	1.49 s	0.000001	30074
256*256	20.3 s	0.000001	102885
512*512	269.4 s	0.000001	339599

CPU-multicore

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	0.35 s	0.000001	30074
256*256	3.57 s	0.000001	102885
512*512	41.6 s	0.000001	339599
1024*1024	483.7 s	0.000001	1000000

GPU

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	0.9 s	9.9998e-07	30080
256*256	2.4 s	9.99984e-07	102912
512*512	19.4 s	9.99984e-07	339968
1024*1024	46.8 s	1.36929e-06	1000000

GPU_cuBLAS

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	1.2 s	9.9998e-07	30074
256*256	3.1 s	9.99984e-07	102885
512*512	12.1 s	9.99984e-07	339599
1024*1024	40.4 s	0.00140936	1000000

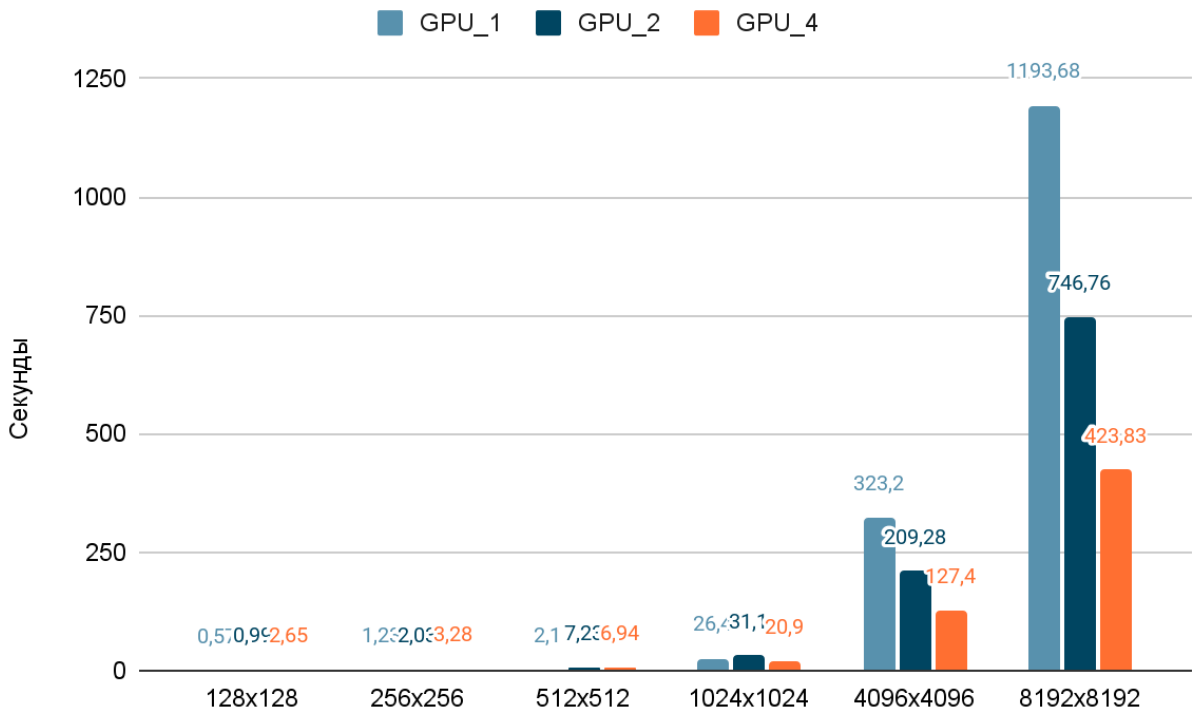
GPU_CUDA

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	0.24 s	9.47552e-07	30500
256*256	0.54 s	9.91241e-07	103000
512*512	2.9 s	9.97135e-07	340000
1024*1024	29.5 s	1.36929e-06	1000000

Данные работы с MPI

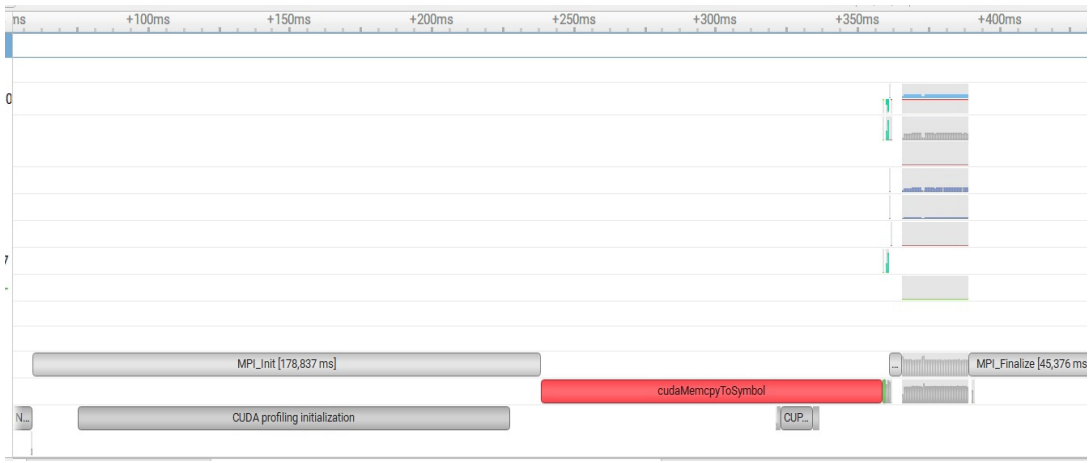
Количество процессов	Размер сетки	Время выполнения	Точность	Количество итераций
1	128x128 256x256 512x512 1024x1024 4096x4096 8192x8192	0.57 s 1.23 s 2.1 s 26.4 s 323.2 s 1193.68 s	8.77769e-07 9.91241e-07 9.91241e-07 1.37256e-06 9.73408e-06 1.01748e-05	30500 103000 340000 1000000 1000000 1000000
2	128x128 256x256 512x512 1024x1024 4096x4096 8192x8192	0.99 s 2.03 s 7.23 s 31.1 s 209.28 s 746.76 s	8.77769e-07 9.91241e-07 9.91241e-07 1.37256e-06 9.73408e-06 1.01748e-05	30500 103000 340000 1000000 1000000 1000000
4	128x128 256x256 512x512 1024x1024 4096x4096 8192x8192	2.65 s 3.28 s 6.94 s 20.9 s 127.4 s 423.83 s	8.77769e-07 9.91241e-07 9.91241e-07 1.37256e-06 9.73408e-06 1.01748e-05	30500 103000 340000 1000000 1000000 1000000

Скорость работы для разных размеров сеток на разном количестве графических процессоров (1, 2, 4).

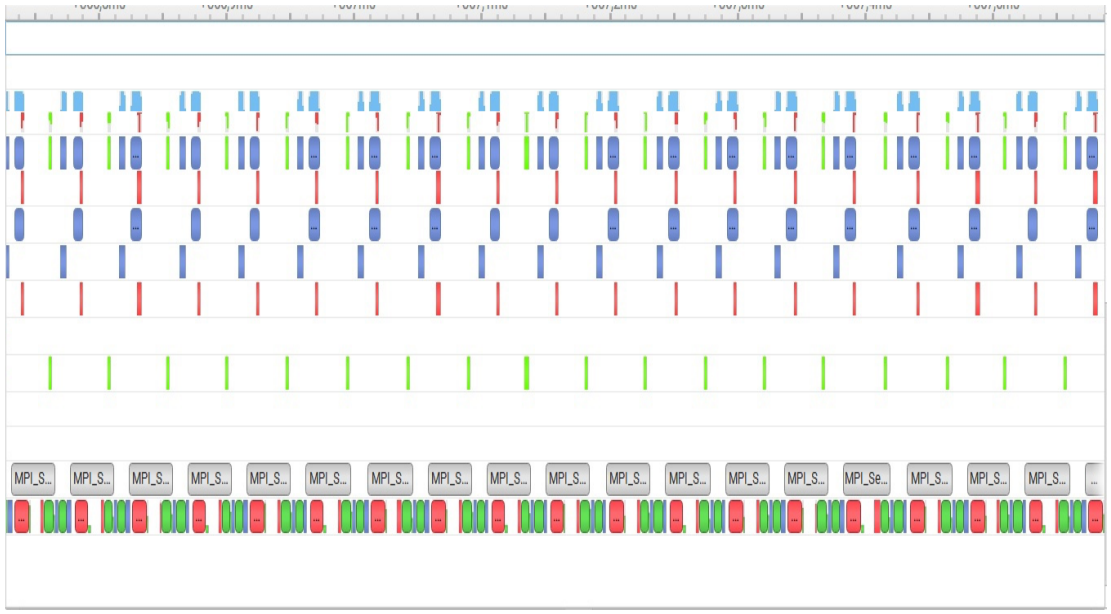


Скриншоты с результатами профилирования в NSightSystems

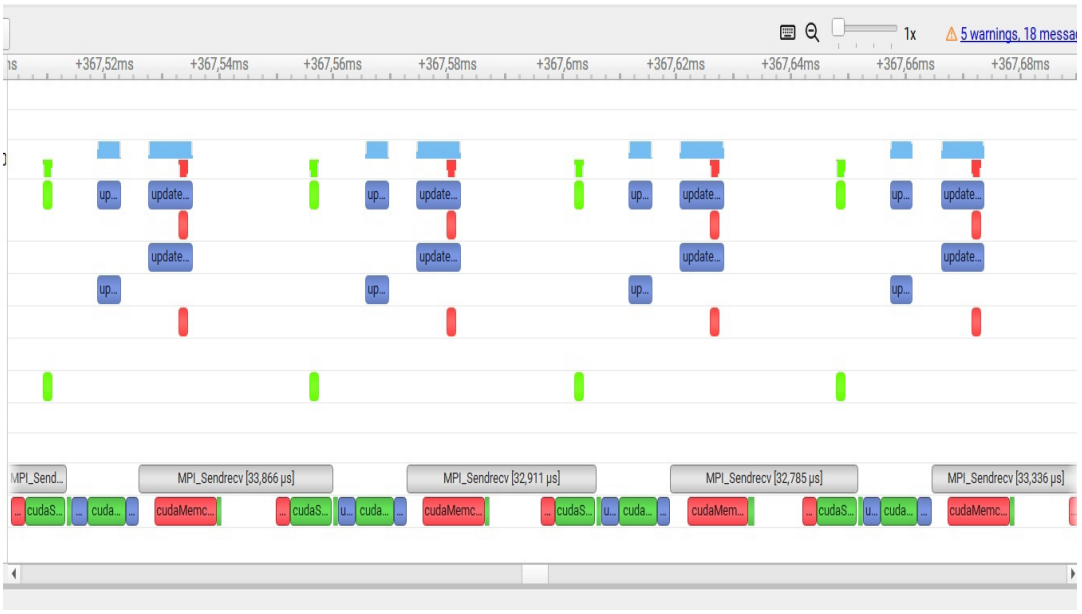
Полная визуализация программы.

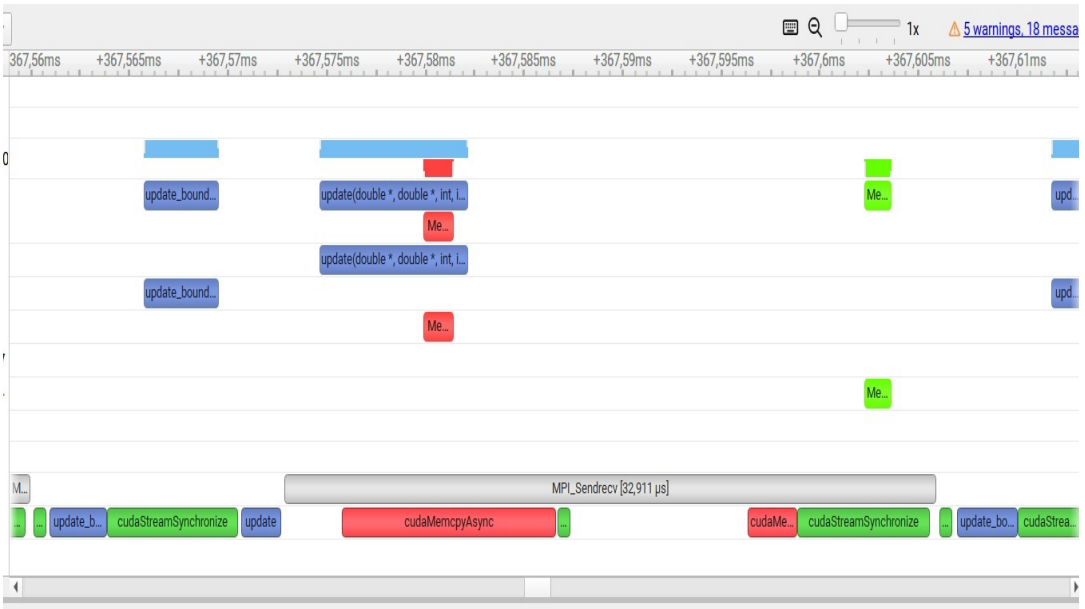


Главный цикл программы 372ms.

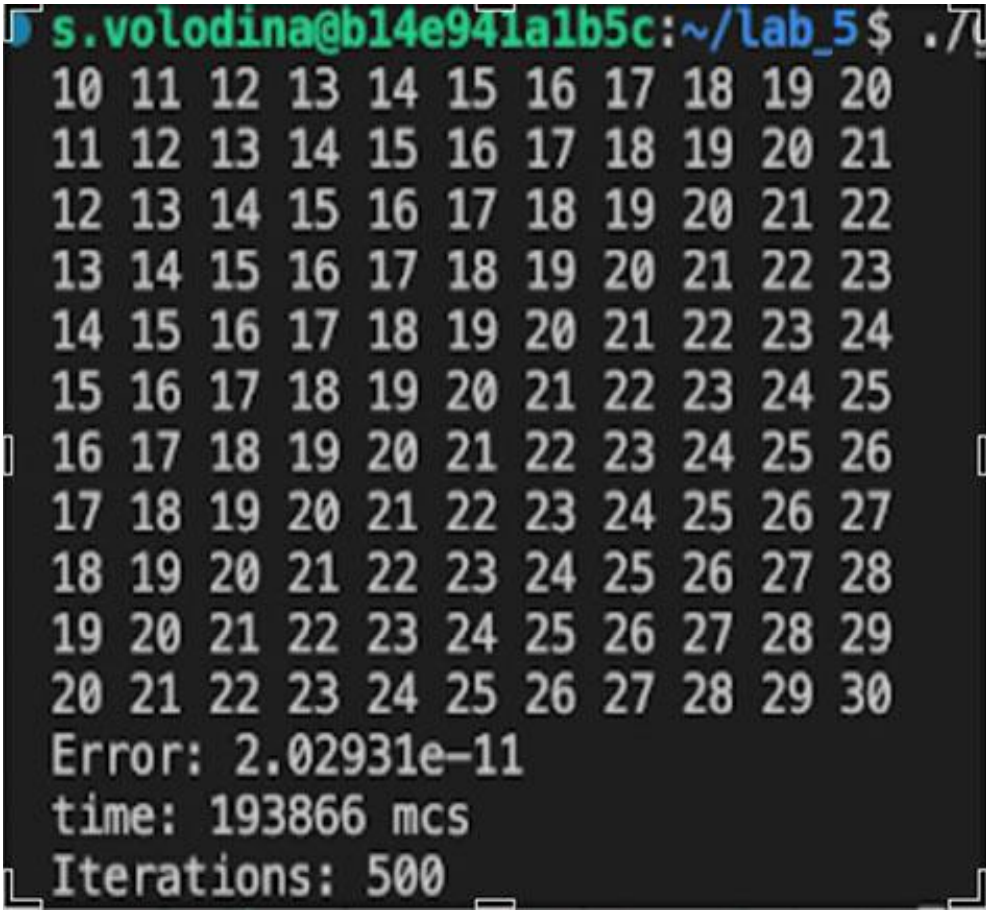


Итерации: обновление границ, обновление внутренних значений, обновление ошибки, обмен границами.





Вывод решения сетки 11 на 11



Вывод

Было реализовано решение уравнения теплопроводности в двумерной области на равномерных сетках с использованием функций CUDA и библиотеки CUB. Распараллеливание на несколько GPU производилось с использованием MPI. Проведено сравнение скорости работы данной программы на разном количестве процессов. Из данных наблюдений можно сделать вывод, что данный подход решения задачи будет эффективен только при больших размерах матрицы и на большем количестве GPU. Если говорить обратно, при меньших матрицах результат по времени гораздо хуже, чем у предыдущих решений задачи, при этом число процессов не поможет улучшить скорость расчетов.

Приложение

Ссылка на git: https://github.com/PiroJOJO/Parallel_5