МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ) Институт №8
«Информационные технологии и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №3
по курсу «Параллельная обработка данных»
Технология MPI и технология CUDA. MPI-IO.

Выполнила: Алексюнина Ю.В.
Группа:М8О-407Б
Преподаватели:
К.Г.Крашенинников, А.Ю. Морозов

Москва, 2021

Условие

Совместное использование технологии MPI и технологии CUDA. Применение библиотеки алгоритмов для параллельных расчетов Thrust. Реализация метода Якоби. Решение задачи Дирихле для уравнения Лапласа в трехмерной области с граничными условиями первого рода. Использование механизмов MPI-IO и производных типов данных.

Запись результатов в файл должна осуществляться параллельно всеми процессами. Необходимо создать производный тип данных, определяющий шаблон записи данных в файл.

Вариант: 1. Конструктор типа MPI_Type_create_subarray

Программное и аппаратное обеспечение

GPU:

- Name: GeForce GTX 750 Ti
- Compute capability: 5.0
- Графическая память: 4294967295
- Разделяемая память: 49152
- Константная память: 65536
- Количество регистров на блок: 65536
- Максимальное количество блоков: (2147483647, 65535, 65535)
- Максимальное количество нитей: (1024, 1024, 64)
- Количество мультипроцессоров: 5

Сведения о системе:

- Процессор: Intel Core i5-4460 3.20GHz
- ОЗУ: 16 ГБ
- HDD: 930 ГБ

Программное обеспечение:

- OS: Windows 8.1
- IDE: Visual Studio 2019

Метод решения:

До этого мы уже писали лабораторную работу, логика работы которой и была взята в основу данной. Отличие состоит в том, что в данной лабораторной также используется CUDA, а еще организация записи в файл осуществляется при помощи MPI-IO.

Для копирования значений и подсчета были написаны несколько ядер. Было написано 3 ядра для копирования и инициализации данных. Эти ядра отличались лишь поверхностью, через которую происходит копирование. Так же были написаны два ядра для вычисления основного цикла и вычисления ошибки. Основную сложность представляли собой множественные индексы, в которых было очень легко запутаться.

Параллельная запись в файл была рассмотрена подробно на лекции, оставалось лишь разобраться с вариантом (MPI_Type_create_subarray). После осознания того, что с помощью MPI_Type_create_subarray можно создавать сетку для вывода данных, я смогла написать многопроцессорную запись в файл.

Описание программы

Для многопроцессорной записи в файл сначала был создан тип данных с помощью которого происходила запись одной ячейки, т.е. одного значения. Далее был создан подмассив, который расчерчивал данные из одной MPI node. Потом мы расчерчивали глобальную сетку, чтобы мы смогли вписать наш подмассив из node в нужное место.
В конце открывали файл на запись и записывали данные из MPI node.

Результаты

	MPI	**MPI+CUDA**	**MPI+Open MP**
1 1 1, 20 20 20	**18933ms**	**8347ms**	**7562ms**
2 2 2, 20 20 20	**5981ms**	**5729ms**	**4822ms**

Вывод

Эта лабораторная работа научила меня не только тому, как использовать CUDA с MPI, но и тому, что нужно быть предельно внимательным, когда используешь код из предыдущей лабораторной. Из-за невнимательности я потратила несколько дней на то, чтобы найти ошибку.