МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Информационные технологии и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №1
по курсу «Параллельная обработка данных»

Message Passing Interface (MPI)

Выполнил: А. В. Куликов
Группа: 8О-408Б
Преподаватели: К.Г. Крашенинников,
А.Ю. Морозов

Москва, 2020

Условие

Цель работы: Знакомство с технологией MPI. Реализация метода Якоби.
Решение задачи Дирихле для уравнения Лапласа в трехмерной области с граничными условиями первого рода.
Вариант 7. Обмен граничными слоями через sendrecv, контроль сходимости allreduce;

Программное и аппаратное обеспечение

Видеокарта	GeForce GTX 1650
Compute capability	7.5
Графическая память	3911 Мб
Разделяемая память	48 Кб
Константная память	64 Кб
Количество регистров на блок	65536
Максимальное кол-во блоков	2147483647*65535*65535
Максимальное кол-во нитей в блоке	1024
Кол-во мультипроцессоров	16
Ядер CUDA	896

Процессор	AMD Ryzen 5 3550H
ОЗУ	8 Гб
ЖД	

Операционная система	Ubuntu 20.04.6 LTS
IDE	VS Code
Компилятор	nvcc V10.1, mpi V3.3.2

Метод решения

Корневой процесс читает входные данные и рассылает их всем остальным процессам при помощи bcasat. Далее начинается сам рассчет.
Каждая его итерация состоит из 3 шагов: обмен граничными условиями, пересчет, вычисление погрешности.
Обмен границами происходи посредством sendrecv. Каждый процесс посылает все свои граничные значения «соседям» и принимает обратно их граничные значения.
Далее для каждой ячейки рассчитывается новое значение на основе предыдущих значений соседних ячеек. В ходе рассчета вычисляется максимальная погрешность по всем ячейкам для каждого блока. Далее при помощи allreduce вычисляется максимум погрешности уже по всем ячейкам.

Программа продолжает вычисления, пока погрешность не станет меньше заданной пользователем точности.

После проведения расчета корневой процесс собирает результаты рассчета ото всех остальных процессов, попутно выводя данные в выходной файл.

Описание программы

Вся программа реализована в одном файле main.cpp. Основная логика реализована прямо в функции main. В отдельную функцию compute вынесен только расчет функции для всех ячеек блока по формуле Якоби.

Для идентификации сообщений в процессе обмена граничными условиями используется как номер процесса, так и тег т. к. каждый процесс обменивается от 2-4 сообщениями. В качестве тега передается значение LEFT, RIGHT и т. д.

При сборе данных корневым процессом после рассчета другие процессы вынуждены посылать данные маленькими порциями по block_size_x элементов, чтобы обеспечить вывод в нужном формате. Поэтому тут тоже необходима идентификация сообщений.

Результаты

Тестирование ядер с различными конфигурациями

область 8x8x8 ячеек

Количество процессов	Время
1	0m 0,044s
2	0m 0,051s
4	0m 0,055s
8	0m 0,082s
16	0m 10,345s

Лучший результат: 0m 0,044s при одном запущенном процессе.

область 64x64x64 ячеек

Количество процессов	Время
1	0m 9,747s
2	0m 6,056s
4	0m 3,419s
8	0m 3,819s
16	3m 59,394s

Лучший результат: 0m 3,419s при 4-х запущенных процессах.

область 128x128x128 ячеек

Количество процессов	Время
1	1m 29,012s
2	2m 6,144s
4	1m 13,876s
8	1m 38,173s
16	7m 38,481s

Лучший результат: 1m 13,876s при 4-х запущенных процессах.

Сравнение с CPU
область 8x8x8 ячеек
Результат: 0m 0,003s

область 64x64x64 ячеек
Результат: 0m 5,446s

область 128x128x128 ячеек
Результат: 1m 41,747s

Выводы

Данный алгоритм может быть использован для решения задачи Дирихле для уравнения Лапласа с граничными условиями 1-го рода в прямоугольной трехмерной области.

Основные сложности в реализации программы возникли в написании обмена граничными условиями между блоками. В решении данной задачи программа с использованием MPI незначительно превзошла стандартную реализацию на C++. Эта незначительность обусловлена тем, что при тестировании все процессы запускались на одной машине с 8-ю потоками.