МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №1
по курсу «Параллельная обработка данных»

Message Passing Interface (MPI)

Выполнил: И.А. Мариничев
Группа: 8О-408Б
Преподаватели: К.Г. Крашенинников
А.Ю. Морозов

Москва, 2022

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 8: обмен граничными слоями через isend/irecv, контроль сходимости allreduce.

Программное и аппаратное обеспечение

Compute capability	:	2.1
Name	:	GeForce GT 545
Total Global Memory	:	3150381056
Shared memory per block	:	49152
Registers per block	:	32768
Warp size	:	32
Max threads per block	:	(1024, 1024, 64)
Max block	:	(65535, 65535, 65535)
Total constant memory	:	65536
Multiprocessors count	:	3
Processor	:	Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz
RAM	:	16 GB
Drive	:	349G
OS	:	Ubuntu 16.04.6 LTS
IDE	:	Visual Studio Code
Compiler	:	NVIDIA (R) Cuda compiler driver V7.5.17

Метод решения

Считываем входные данные и параметры расчета. Передаем параметры расчета всем процессам через MPI_Bcast. Затем запускаем основной цикл метода Якоби, внутри которого обеспечиваем межпроцессорное взаимодействие при помощи MPI_Isend и MPI_Irecv, отправка и прием происходит в шесть сторон (вверх, вниз, вправо, влево, вперед, назад). После того как метод сошелся отправляем все необходимые данные корневому процессу, который выводит полученные данные в файл.

Описание программы

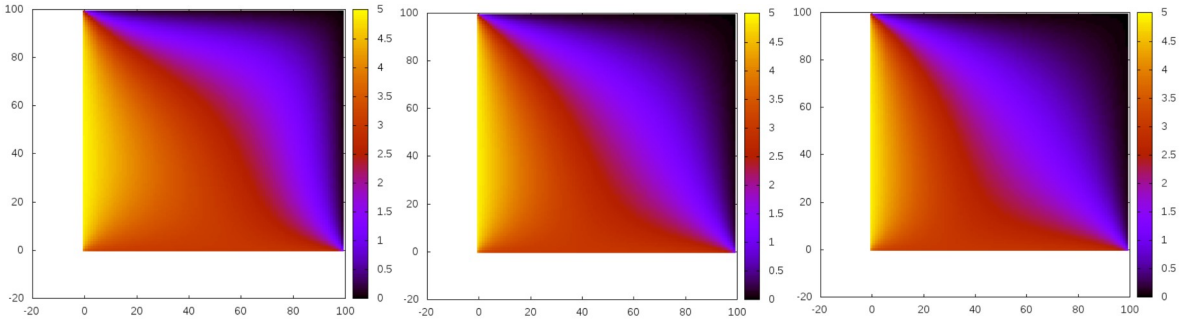
```
// пример межпроцессорного взаимодействия: отправка вправо
if (ib + 1 < nbx) {
        MPI_Request request1, request2;
        for (int k = 0; k < nz; ++k) {
                for (int j = 0; j < ny; ++j) {
            send_buff1[j + k * ny] = data[_i(nx - 1, j, k)];
          }
        }
        MPI_CALL(MPI_Isend(send_buff1, ny * nz, MPI_DOUBLE, _ib(ib + 1, jb, kb), 0,
MPI_COMM_WORLD, &request1));
        MPI_CALL(MPI_Irecv(receive_buff1, ny * nz, MPI_DOUBLE, _ib(ib + 1, jb, kb), 0,
MPI_COMM_WORLD, &request2));
        MPI_CALL(MPI_Wait(&request1, MPI_STATUS_IGNORE));
        MPI_CALL(MPI_Wait(&request2, MPI_STATUS_IGNORE));
        for (int k = 0; k < nz; ++k) {
          for (int j = 0; j < ny; ++j) {
            data[_i(nx, j, k)] = receive_buff1[j + k * ny];
          }
        }
      } else {
        for (int k = 0; k < nz; ++k) {
          for (int j = 0; j < ny; ++j) {
            data[_i(nx, j, k)] = u_right;
          }
        }
}

// перевычисление на текущей итерации
error = 0.0;
for (int i = 0; i < nx; ++i) {
        for (int j = 0; j < ny; ++j) {
          for (int k = 0; k < nz; ++k) {
            next[_i(i, j, k)] = ((data[_i(i - 1, j, k)] + data[_i(i + 1, j, k)]) * h2x +
                                 (data[_i(i, j - 1, k)] + data[_i(i, j + 1, k)]) * h2y +
                                 (data[_i(i, j, k - 1)] + data[_i(i, j, k + 1)]) * h2z) /
                                (2 * (h2x + h2y + h2z));
            error = std::max(error, std::fabs(next[_i(i, j, k)] - data[_i(i, j, k)]));
          }
        }
}

MPI_CALL(MPI_Allreduce(&error, &error, 1, MPI_DOUBLE, MPI_MAX,
MPI_COMM_WORLD));
```

Результаты

Сетка процессоров	Размер сетки	Время в ms
(1, 1, 1)	(120, 10, 10)	8.27494e+06
(2, 1, 1)	(60, 10, 10)	3.95423e+06
(3, 1, 1)	(40, 10, 10)	2.85953e+06
(2, 2, 1)	(60, 5, 10)	2.72952e+06
(5, 1, 1)	(24, 10, 10)	2.83332e+06
(3, 2, 1)	(40, 5, 10)	2.72788e+06

Распределение «температуры»

Ниже представлены изображения, иллюстрирующие полученное распределение «температуры» в рассматриваемой области в виде набора сечений:



Описание области: сетка процессов: (1, 1, 1), сетка: (100, 100, 3), точность: 1e-10, l = (1, 1, 2), u_bottom = 7, u_top =0, u_left= 5, u_right= 0, u_front =3, u_back =0, u_0 = 5.

Выводы

MPI – это программный интерфейс для передачи информации, который позволяет обмениваться сообщениями между процессами, выполняющими одну задачу. В данной лабораторной работе была решена задача Дирихле для уравнения Лапласа в трехмерной области с граничными условиями первого рода. MPI является наиболее распространённым стандартом интерфейса обмена данными в параллельном программировании, существуют его реализации для большого числа компьютерных платформ. Как видно по результатам работы, при переходе к двум процессам наблюдается значительное ускорение, при дальнейшем увеличении числа процессов ускорение также присутствует, но разница уже значительно меньше.