Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

Направление 09.03.04 – «Программная инженерия»

Дисциплина: «Технологии блокчейн и распределенные информационные системы»

Семестр 8

ОТЧЕТ

по лабораторной работе №4

Тема: «Параллельное умножение матриц MPI»

Выполнил: студент группы РИС-20-2б

Гулуа И.Ю. \_\_\_\_\_\_\_\_\_\_\_

Проверил: старший преподаватель кафедры ИТАС

Щапов В. А. \_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_

Пермь, 2024

**Цели работы**

Рассмотреть параллельное умножение матриц с помощью MPI.

**Задание к работе**

1. Написать программу для параллельного умножения матриц с помощью MPI

**Ход работы**

Напишем на языке программирования C++ программу для параллельного умножения матриц с помощью MPI. Функция main представлена на листинге 1:

Листинг 1 – Функция main

int main(int argc, char\* argv[])

{ //теги сообщений 1 - от мастера, 2 - от рабочих

int numt, rank;

int rowsForA, extraRows, rows; //Дополнительные строки если кол-во строк не делится на целое

int buff = 0;

MPI\_Status status;

double beginTime, endTime;

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

matrixResult[i][j] = 0;

}

}

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &numt);

MPI\_Barrier(MPI\_COMM\_WORLD);

if (rank == 0) { // Мастер распределяет строки между процессами

printf("Master is here\n");

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

matrix1[i][j] = rand();

}

}

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

matrix2[i][j] = rand();

}

}

beginTime = MPI\_Wtime();

rowsForA = N / (numt - 1); //общее кол-во процессов минус 1

extraRows = N % (numt - 1); //Остаток требующий распределения

for (int i = 1; i <= numt - 1; i++) { //Отправка начальный значений

if (i <= extraRows) {

rows = rowsForA + 1;

}

else {

rows = rowsForA;

}

MPI\_Send(&buff, 1, MPI\_INT, i, 1, MPI\_COMM\_WORLD);//начальный номер строки

MPI\_Send(&rows, 1, MPI\_INT, i, 1, MPI\_COMM\_WORLD);//количество строк

MPI\_Send(&matrix1[buff][0], N \* N, MPI\_DOUBLE, i, 1, MPI\_COMM\_WORLD);//Отправка строки

MPI\_Send(&matrix2, N \* N, MPI\_DOUBLE, i, 1, MPI\_COMM\_WORLD);

buff += rows;

}

for (int i = 1; i <= numt - 1; i++) { //Получение результата

MPI\_Recv(&buff, 1, MPI\_INT, i, 2, MPI\_COMM\_WORLD, &status);

MPI\_Recv(&rows, 1, MPI\_INT, i, 2, MPI\_COMM\_WORLD, &status);

MPI\_Recv(&(matrixResult[buff][0]), N \* N, MPI\_DOUBLE, i, 2, MPI\_COMM\_WORLD, &status);

}

endTime = MPI\_Wtime();

double time = endTime - beginTime;

printf("Time %f s\n", time);

//printf("Matrix Result:\n");

//printMatrix(matrixResult);

}

else {

MPI\_Recv(&buff, 1, MPI\_INT, 0, 1, MPI\_COMM\_WORLD, &status);

MPI\_Recv(&rows, 1, MPI\_INT, 0, 1, MPI\_COMM\_WORLD, &status);

MPI\_Recv(&matrix1[buff][0], N \* N, MPI\_DOUBLE, 0, 1, MPI\_COMM\_WORLD, &status);

MPI\_Recv(&matrix2, N \* N, MPI\_DOUBLE, 0, 1, MPI\_COMM\_WORLD, &status);

for (int i = buff; i < buff + rows; i++) {

for (int k = 0; k < N; k++) {

for (int j = 0; j < N; j++) {

matrixResult[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

MPI\_Send(&buff, 1, MPI\_INT, 0, 2, MPI\_COMM\_WORLD);

MPI\_Send(&rows, 1, MPI\_INT, 0, 2, MPI\_COMM\_WORLD);

MPI\_Send(&(matrixResult[buff][0]), N \* N, MPI\_DOUBLE, 0, 2, MPI\_COMM\_WORLD);

}

MPI\_Finalize();

}

Процессор с рангом 0 принимает на себя роль мастера, который распределяет строки матрицы 1 между остальными процессами. В отправки сообщений используется метод MPI\_Send, для принятия сообщений MPI\_Recv.

Пример работы программы представлен на рисунке 1.

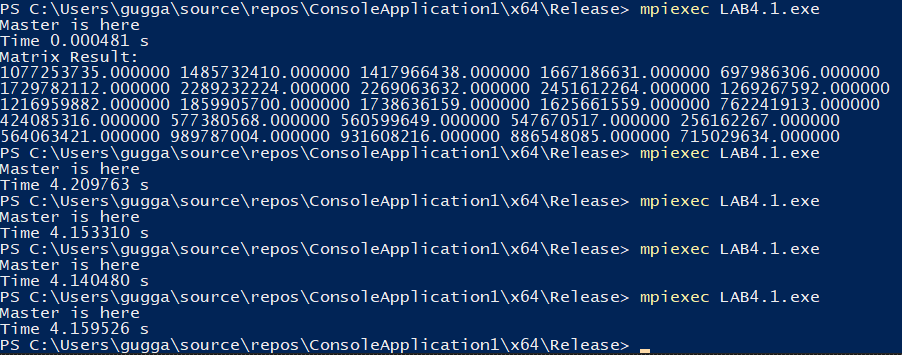


Рисунок 1 – Выполнение программы

На рисунке 1 представлен запуск программы с матрицами 5 на 5 для проверки на правильность полученных значений, которые оказались верны.

Далее был запущен алгоритм для умножения матриц размерности 2500 на 2500

**Заключение**

В ходе выполнения лабораторной работы была написана программа программу для параллельного умножения матриц с помощью MPI.