Міністерство освіти і науки України

Одеський національний політехнічний університет

Інститут комп’ютерних систем  
Кафедра комп’ютерних інтелектуальних систем і мереж

Протокол

Лабораторної роботи №6

На тему “Разработка параллельного умножения матриц, блочные схемы распределения данных (алгоритмы Фокса и Кеннона). Оценка эффективности параллельного алгоритма”

По предмету: «Основи розподілених обчислень»

Виконали:

студенти групи АМ-181

Борщов М.І.

Перевірив:

Тішин П.М.

Одеса 2021

**Перелік завдань до лабораторної роботи**

1. Напишите и отладьте программу умножения матрицы на матрицу.

2. Элементы матрицы  и вектора  должны определяться по формулам

,

, , 

где  - номер варианта,  - размерность матрицы.

**Хід роботи**

**Код програми**

#include <mpi.h>

#include <conio.h>

#include <time.h>

#include <iostream>

#include <windows.h>

#include <math.h>

using namespace std;

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

{

srand(time(0));

int size, rank;

const int MAX\_ROWS = 3;

const int MAX\_COLS = 3;

double A[MAX\_ROWS][MAX\_COLS], B[MAX\_ROWS][MAX\_COLS], buffer[MAX\_ROWS], result[MAX\_COLS];

MPI\_Status status;

MPI\_Init(&argc, &argv);

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

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

if (rank == 0) {

for (int a = 0; a < MAX\_ROWS; a++) {

for (int b = 0; b < MAX\_COLS; b++) {

A[a][b] = double(rand() % 10) + 1;

B[a][b] = double(rand() % 10) + 1;

}

}

for (int a = 0; a < 5; a++) {

MPI\_Bcast(B[a], 5, MPI\_DOUBLE, 0, MPI\_COMM\_WORLD);

}

for (int a = 0; a < MAX\_ROWS; a++) {

for (int b = 0; b < MAX\_COLS; b++) {

buffer[b] = A[a][b];

}

MPI\_Send(&buffer, MAX\_ROWS, MPI\_DOUBLE, (a + 1), (a + 1), MPI\_COMM\_WORLD);

}

}

else {

for (int b = 0; b < MAX\_COLS; b++) {

result[b] = 0;

}

for (int a = 0; a < MAX\_ROWS; a++) {

MPI\_Bcast(B[a], 5, MPI\_DOUBLE, 0, MPI\_COMM\_WORLD);

}

MPI\_Recv(&buffer, MAX\_ROWS, MPI\_DOUBLE, 0, MPI\_ANY\_TAG, MPI\_COMM\_WORLD, &status);

if (status.MPI\_TAG == 0) {

}

else {

for (int a = 0; a < MAX\_ROWS; a++) {

for (int b = 0; b < MAX\_COLS; b++) {

result[a] += buffer[b] \* B[b][a];

}

}

cout << status.MPI\_TAG << ": ";

for (int a = 0; a < MAX\_ROWS; a++) {

cout << result[a] << " ";

}

cout << endl;

}

}

MPI\_Finalize();

return 0;

**Висновок:**

У результаті виконання даної лабораторної роботи ми розробили програму паралельного множення матриці на матрицю шляхом використання методів MPI та зміцнили свої знання у передачі даних між процесами за допомогою методів MPI\_Send, MPI\_Bcast, MPI\_Recv.