МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий и управления в технических системах

Кафедра «Информационные системы»

**ОТЧЕТ**

о выполнении лабораторной работы № 2

по дисциплине “[Технологии распределенных систем и параллельных вычислений](https://do.sevsu.ru/course/view.php?id=7205)”

Выполнил: ст. гр. ИС/б-20-2-о

Белик Г. М.

Проверил: доц. каф.

«Информационные системы»

Кротов К. В.

Севастополь

2023

**Цель работы**

Исследовать способы обмена данными между процессами в режиме широковещания или группового обмена с использованием MPI-функций.

**Вариант 1**

Реализовать блочный алгоритм распределенного параллельного перемножения матриц Aи Bс размерами (8\*5) и (5\*3) соответственно. Вид распределяемых между процессами блоков представлен на рисунке 1:



Рисунок 1 – Перемножение матриц

Корневой процесс реализует рассылку:

1) блоков матрицы A по две строки;

2) широковещательную рассылку элементов матрицы B между обрабатывающими процессами внутри своей группы (по умолчанию) – режим **One-To-All**.

Организуется четыре процесса, обрабатывающих данные и формирующих фрагменты (2\*3) матрицы результата C. После подготовки блоков матрицы со всеми обрабатывающими процессами (взаимная синхронизация функцией **MPI\_Barrier**) выполняется совместная передача результатов (блоков матрицы C) корневому процессу – режим **All-To-One**.

**Код программы**

#include <iostream>

#include <iomanip>

#include "mpi.h"

using namespace std;

const int N = 8;

const int M = 5;

const int K = 3;

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

{

int rank, processes;

MPI\_Init(&argc, &argv);

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

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

int masterProcess = 0;

int A[N][M];

int B[M][K];

int C[N][K];

int blockA[2][M];

int blockC[2][K];

if (rank == masterProcess)

{

int dataA[N][M] = {

{1, 2, 3, 4, 5},

{6, 7, 8, 9, 10},

{11, 12, 13, 14, 15},

{16, 17, 18, 19, 20},

{21, 22, 23, 24, 25},

{26, 27, 28, 29, 30},

{31, 32, 33, 34, 35},

{36, 37, 38, 39, 40},

};

int dataB[M][K] = {

{1, 2, 3},

{4, 5, 6},

{7, 8, 9},

{10, 11, 12},

{13, 14, 15},

};

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

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

B[i][j] = dataB[i][j];

}

}

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

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

A[i][j] = dataA[i][j];

}

}

}

// Широковещательная рассылка

MPI\_Bcast(B, M \* K, MPI\_INT, masterProcess, MPI\_COMM\_WORLD);

MPI\_Scatter(A, 10, MPI\_INT, blockA, 10, MPI\_INT, masterProcess, MPI\_COMM\_WORLD);

// Вычисления

for (int i = 0; i < 2; i++) {

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

blockC[i][j] = 0;

for (int g = 0; g < M; g++) {

blockC[i][j] += blockA[i][g] \* B[g][j];

}

}

}

for (int i = 0; i < 2; i++) {

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

cout << setw(6) << blockC[i][j] << " ---" << rank << "---";

}

cout << endl;

}

// Взаимная синхронизация

MPI\_Barrier(MPI\_COMM\_WORLD);

// Совместная передача данных процессу-мастеру

MPI\_Gather(blockC, 6, MPI\_INT, C, 6, MPI\_INT, masterProcess, MPI\_COMM\_WORLD);

if (rank == masterProcess) {

cout << endl << endl << "Result matrix: " << endl;

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

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

cout << setw(6) << C[i][j] << " ";

}

cout << endl;

}

}

MPI\_Finalize();

return 0;

}

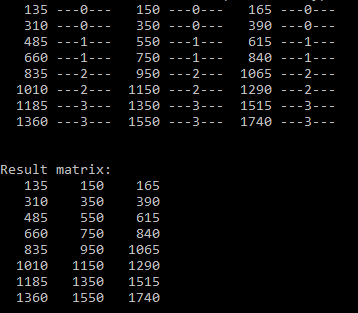


Рисунок 2 – Результат работы программы

**Выводы**

В ходе выполнения лабораторной работы были исследованы способы обмена данными между процессами в режиме широковещания или группового обмена с использованием MPI-функций. Был реализован блочный алгоритм распределенного параллельного перемножения матриц A и B с размерами (8\*5) и (5\*3) соответственно.