A, B, a, C,MO,MX

П7

П6

П5

П8

П1

П2

П3

П4

**Этап 1. Разработка параллельного математического алгоритма.**

Математическая задача: AH = BH + a \*C\*( MOH \* MX)

Средства программирования: C++, библиотека MPI

Общие ресурсы: a, C, MX

**Этап 2. Разработка алгоритмов задач.**

**T1**

1. Ввод alpha, MX, MO, B,C
2. Передача alpha, MO[H..4H], B[H..4H],MX,Cв задачу Т2
3. Передача alpha, MO[4H..5H], B[4H..5H] MX,C в задачу Т5
4. Передача alpha, MO[5H..7H], B[5H..7H], MX,C в задачу Т6
5. Передача alpha, MO[7H..8H], B[7H..8H], MX,C в задачу Т8
6. Счёт AH = BH + a \*C\*( MOH \* MX)
7. Принять А[H..4H] от задачи Т2
8. Принять А[4H..5H] от задачи Т5
9. Принять А[5H..7H] от задачи Т6
10. Принять А[7H..8H] от задачи Т8
11. Вывод А

**T2**

1. Принять alpha, MO[H..4H], B[H..4H],MX,C от задачи Т1
2. Передача alpha, MO[2H..4H], B[2H..4H],MX,C в задачу Т3
3. Счёт AH = BH + a \*C\*( MOH \* MX)
4. Принять A[2H..4H] от задачи Т3
5. Передача A[H..4H] в задачу Т1

**T3**

1. Принять alpha, MO[2H..4H], B[2H..4H],MX,C от задачи Т2
2. Передача alpha, MO[3H..4H], B[3H..4H],MX,C в задачу Т4
3. Счёт AH = BH + a \*C\*( MOH \* MX)
4. Принять A[3H..4H] от задачи Т4
5. Передача A[2H..4H] в задачу Т2

**Т4**

1. Принять alpha, MO[3H..4H], B[3H..4H],MX,C от задачи Т3
2. Счёт AH = BH + a \*C\*( MOH \* MX)
3. Передача A[3H..4H] в задачу Т3

**Т5**

1. Принять alpha, MO[4H..5H], B[4H..5H] MX,C от задачи Т1
2. Счёт AH = BH + a \*C\*( MOH \* MX)
3. Передача MA[4H..5H] в задачу Т1

**T6**

1. Принять alpha, MO[5H..7H], B[5H..7H], MX,C от задачи Т1
2. Передача alpha, MO[6H..7H], B[6H..7H], MX,C в задачу Т7
3. Счёт AH = BH + a \*C\*( MOH \* MX)
4. Принять A[6H..7H] от задачи Т7
5. Передача A[5H..7H] в задачу Т1

**Т7**

1. Принять alpha, MO[6H..7H], B[6H..7H], MX,C от задачи Т6
2. Счёт AH = BH + a \*C\*( MOH \* MX)
3. Передача A[6H..7H] в задачу Т6

**Т8**

1. Принять alpha, MO[7H..8H], B[7H..8H], MX,C от задачи Т1
2. Счёт AH = BH + a \*C\*( MOH \* MX)
3. Передача A[7H..8H] в задачу Т1

**Этап 4. Разработка программы**

//---------------Лабораторна робота 8-----------------------------

//---------------Матвійчук богдан--------------------

//---------------A = B + alpha \* C \* ( MO \* MX )-------

//---------------16.05.2012---------------------------------

#include <mpi.h>

#include <iostream>

using namespace std;

int N = 8;

int P = 8;

int H = N / P;

int \*\* MX = new int\* [N];

int \*\* MO = new int\* [N];

int \* C = new int\* [N];

int \* B = new int\* [N];

int \* A = new int\* [N];

int alpha;

MPI\_Status s;

static void inputMatrix(int\*\* matr, int temp)

{

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

{

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

{

matr[i][j] = temp;

}

}

}

static void inputVector(int\* matr, int temp)

{

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

{

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

{

matr[i][j] = temp;

}

}

}

static void calc()

{

for (int i = 0; i < H; i++)

{

C[i] = alpha \* C[i];

}

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

{

for (int j = 0; j < H; j++)

{

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

{

MO[i][j] += MO[i][k] \* MX[k][j];

}

}

}

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

{

for (int j = 0; j < H; j++)

{

C[j] += C[j] \* MO[j][H];

}

}

for (int i = 0; i < H; i++)

{

A[i] = B[i] + C[i];

}

}

static void initMatrix(int\*\* matr)

{

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

{

matr[i] = new int [N];

}

}

static void send\_OR(int left, int right, int tag, int dest)

{

for (int i = left; i < right; i++)

{

MPI\_Send(MO[i], N, MPI\_INT, dest, tag, MPI\_COMM\_WORLD);

}

}

static void recieve\_OR(int left, int right, int source, int tag)

{

for (int i = left; i < right; i++)

{

MPI\_Recv(MO[i], N, MPI\_INT, source, tag, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

}

}

static void send\_a(int tag, int dest)

{

MPI\_Send(&alpha, 1, MPI\_INT, dest, tag, MPI\_COMM\_WORLD);

}

static void recieve\_a(int source, int tag)

{

MPI\_Recv(&alpha, N, MPI\_INT, source, tag, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

}

static void send\_vector(int dest, int tag)

{

for (int i = 0; i < H; i++)

{

MPI\_Send(B[i], N, MPI\_INT, dest, tag, MPI\_COMM\_WORLD);

}

}

static void recieve\_vector(int source, int tag)

{

for (int i = 0; i < H; i++)

{

MPI\_Recv(B[i], N, MPI\_INT, source, tag, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

}

}

static void send\_A(int left, int right, int dest, int tag)

{

for (int i = left; i < right; i++)

{

MPI\_Send(MA[i], N, MPI\_INT, dest, tag, MPI\_COMM\_WORLD);

}

}

static void recieve\_A(int left, int right, int source, int tag)

{

for (int i = left; i < right; i++)

{

MPI\_Recv(MA[i], N, MPI\_INT, source, tag, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

}

}

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

{

MPI\_Init(&argc, &argv);

int rank;

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

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

//---------------Program Started-----------------------------

std::cout << "Main Started" << std::endl;

std::cout << "T" << P << " Started" << std::endl;

initMatrix(MO);

initMatrix(MX);

initVector(C);

initVector(B);

if (rank == 0)

{

std::cout << "T1 Inputing" << std::endl;

inputMatrix(MO,1);

inputMatrix(MX,1);

inputVector(C,1);

inputVector(B,1);

alpha = 1;

send\_OR(H, 4 \* H, 1, 0);

send\_ab(1, 0);

send\_vector(1, 0);

send\_OR(4 \* H, 5 \* H, 4, 0);

send\_ab(4, 0);

send\_vector(4, 0);

send\_OR(5\* H, 7 \* H, 5, 0);

send\_ab(5, 0);

send\_vector(5, 0);

send\_OR(7 \* H, 8 \* H, 8, 0);

send\_ab(8, 0);

send\_vector(8, 0);

calc();

recieve\_MA(H, 4 \* H, 1, 0);

recieve\_MA(4 \* H, 5 \* H, 4, 0);

recieve\_MA(5 \* H, 7 \* H, 7, 0);

recieve\_MA(7 \* H, 8 \* H, 8, 0);

if (N <= 20)

{

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

cout << " " << A[i] << endl;

}

}

if (rank == 1)

{

recieve\_ab(0, 0);

recieve\_OR(H, 4 \* H, 0, 0);

recieve\_vector(0, 0);

send\_ab(2, 0);

send\_OR(2 \* H, 4 \* H, 2, 0);

send\_vector(2, 0);

calc();

recieve\_A(2 \* H, 4 \* H, 2, 0);

send\_A(H, 4 \* H, 0, 0);

}

if (rank == 2)

{

recieve\_ab(1, 0);

recieve\_OR(2 \* H, 4 \* H, 1, 0);

recieve\_vector(1, 0);

send\_ab(3, 0);

send\_OR(3 \* H, 4 \* H, 3, 0);

send\_vector(3, 0);

calc();

recieve\_A(3 \* H, 4 \* H, 3, 0);

send\_A(2 \* H, 4 \* H, 1, 0);

}

if (rank == 3)

{

recieve\_ab(2, 0);

recieve\_OR(3 \* H, 4 \* H, 2, 0);

recieve\_vector(2, 0);

calc();

send\_A(3 \* H, 4 \* H, 2, 0);

}

if (rank == 4)

{

recieve\_ab(0, 0);

recieve\_OR(4 \* H, 5 \* H, 0, 0);

recieve\_vector(0, 0);

send\_ab(5, 0);

send\_OR(4 \* H, 5\* H, 5, 0);

send\_vector(5, 0);

calc();

recieve\_A(4 \* H, 5 \* H, 5, 0);

send\_A(4\* H, 5 \* H, 0, 0);

}

if (rank == 5)

{

recieve\_ab(4, 0);

recieve\_OR(5 \* H, 7 \* H, 4, 0);

recieve\_vector(4, 0);

send\_ab(6, 0);

send\_OR(6 \* H, 7 \* H, 6, 0);

send\_vector(6, 0);

calc();

recieve\_A(6 \* H, 7 \* H, 6, 0);

send\_A(5 \* H, 7 \* H, 4, 0);

}

if (rank == 6)

{

recieve\_ab(5, 0);

recieve\_OR(6 \* H, 7 \* H, 5, 0);

recieve\_vector(5, 0);

calc();

send\_A(6 \* H, 7 \* H, 5, 0);

}

if (rank == 7)

{

recieve\_ab(0, 0);

recieve\_OR(7 \* H, 8 \* H, 0, 0);

recieve\_vector(0, 0);

calc();

recieve\_A(7 \* H, 8 \* H, 8, 0);

send\_A(7 \* H, 8 \* H, 0, 0);

}

std::cout << "T " << P << " Ended" << std::endl;

//---------------MPI\_Finilize()-----------------------------

MPI\_Finalize();

std::cout << "Main Ended" << std::endl;

std::cin;

return 0;

}