Міністерство освіти і науки, молоді та спорту України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

# Лабораторна робота №9

# з курсу «Паралельні та розподілені обчислення-2»

**Тема:** MPI

Виконав: студент групи ІО-93

Глухенько Костянтин Анатолійович

2012 р.

**Завдання**

Математичний вираз: A=B\*max(C)+Z\*(MO\*MT)

Засоби програмування: MPI

Структура ПКС:

m= √P

T(P-m)+1

TP-1

ТP

Z, MO

.

.

.

T(P-m)

Tm+1

Tm

T2

Т1

A, B, C, MT

r – порядковий номер процесора

**Виконання**

**Етап 1.** Побудова паралельного алгоритму.

1. ai=max(СH); i=1,P; H=N/P
2. a=max(a, ai); i=1,P
3. AH=BH\*а+Z\*(MO\*MTH)

**Етап 2.** Розроблення алгоритмів роботи кожного процесу.

|  |  |
| --- | --- |
|  | Т1 |
| 1) | Введення B, C, MT |
| 2) | Передача B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H в задачу Т2 |
| 3) | Передача B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H в задачу Т1+m |
| 4) | Отримання Z, MO від задачі Т2 |
| 5) | Обчислення a1=max(СH); |
| 6) | Отримання a2 від задачі Т2 |
| 7) | Обчислення a1=max(a1, a2); |
| 8) | Отримання am+1 від задачі Тm+1 |
| 9) | Обчислення a1=max(a1, am); |
| 10) | Передача a1 в задачу Т2 |
| 11) | Передача a1 в задачу Т1+m |
| 12) | Обчислення AH=BH\*а+Z\*(MO\*MTH) |
| 13) | Отримання AH\*m від задачі Т2 |
| 14) | Отримання AN-((r+1+m)\*H) від задачі Тm+1 |
| 15) | Виведення A |

|  |  |
| --- | --- |
|  | Т2..m |
| 1) | Отримання B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H від задачі Т1 |
| 2) | Передача B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H в задачу Тr+1 |
| 3) | Передача B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H в задачу Тr+m |
| 4) | Отримання Z, MO від задачі Тr+1 |
| 5) | Передача Z, MO в задачу Тr-1 |
| 6) | Обчислення ar=max(СH); |
| 7) | Отримання ar+1 від задачі Тr+1 |
| 8) | Обчислення ar=max(ar, ar+1); |
| 9) | Передача ar в задачу Тr-1 |
| 10) | Отримання ar від задачі Тr-1 |
| 11) | Передача ar в задачу Тr+1 |
| 12) | Передача ar в задачу Тr+m |
| 13) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 14) | Отримання AH\*m від задачі Тr+1 |
| 15) | Передача Aв задачу Тr-1 |

|  |  |
| --- | --- |
|  | Тm |
| 1) | Отримання B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H від задачі Тr-1 |
| 2) | Передача B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H в задачу Тr+m |
| 3) | Передача Z, MO в задачу Тr-1 |
| 4) | Обчислення ar=max(СH); |
| 5) | Передача ar в задачу Тr-1 |
| 6) | Отримання ar від задачі Тr-1 |
| 7) | Передача ar в задачу Тr+m |
| 8) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 9) | Передача Aв задачу Тr-1 |

|  |  |
| --- | --- |
|  | Тm..P (r+1%m == 0) |
| 1) | Отримання B(r/(m-1))\*H, С(r/(m-1))\*H, MT(r/(m-1))\*H від задачі Тr-m |
| 2) | Передача B(r/(m-1))\*H, С(r/(m-1))\*H, MT(r/(m-1))\*H в задачу Тr+m |
| 3) | Отримання Z, MO від задачі Тr+1 |
| 4) | Обчислення ar=max(СH); |
| 5) | Отримання ar+1 від задачі Тr+1 |
| 6) | Обчислення ar=max(ar, ar+1); |
| 7) | Отримання ar+m від задачі Тr+m |
| 8) | Обчислення ar=max(ar, ar+m); |
| 9) | Передача ar в задачу Тr-m |
| 10) | Отримання ar від задачі Тr-m |
| 11) | Передача ar в задачу Тr+m |
| 12) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 13) | Отримання AH\*m від задачі Тr+1 |
| 14) | Отримання AN-((r+1+m)\*H) від задачі Тm+r |
| 15) | Передача Aв задачу Тr-m |

|  |  |
| --- | --- |
|  | Trii || (r >m && r+1%m !=0 && r < P-m) |
| 1) | Отримання B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H від задачі Тr-m |
| 2) | Отримання Z, MO від задачі Тr+m |
| 3) | Передача B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H в задачу Тr+1 |
| 4) | Передача Z, MO в задачу Тr-1 |
| 5) | Обчислення ar=max(СH); |
| 6) | Отримання ar+1 від задачі Тr+1 |
| 7) | Обчислення ar=max(ar, ar+1); |
| 8) | Передача ar в задачу Тr-1 |
| 9) | Отримання ar від задачі Тr-m |
| 10) | Передача ar в задачу Тr+m |
| 11) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 12) | Отримання AH\*m від задачі Тr+1 |
| 13) | Передача Aв задачу Тr-1 |

|  |  |
| --- | --- |
|  | Тm..P (r%m == 0) |
| 1) | Отримання Z, MO від задачі Тr+m |
| 2) | Передача Z, MO в задачу Тr-m |
| 3) | Отримання B(r/(m-1))\*H, С(r/(m-1))\*H, MT(r/(m-1))\*H від задачі Тr-1 |
| 4) | Обчислення ar=max(СH); |
| 5) | Передача ar в задачу Тr-1 |
| 6) | Отримання ar від задачі Тr-m |
| 7) | Передача ar в задачу Тr+m |
| 8) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 9) | Передача Aв задачу Тr-1 |

|  |  |
| --- | --- |
|  | ТP+1-m |
| 1) | Отримання Z, MO від задачі Тr+1 |
| 2) | Отримання Bm\*H, СH\*m, MTH\*m від задачі Тr-m |
| 3) | Передача Bm\*H, СH\*m, MTH\*m в задачу Тr+1 |
| 4) | Обчислення ar=max(СH); |
| 5) | Отримання ar+1 від задачі Тr+1 |
| 6) | Обчислення ar=max(ar, ar+1); |
| 7) | Передача ar в задачу Тr-m |
| 8) | Отримання ar від задачі Тr-m |
| 9) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 10) | Отримання AH\*m від задачі Тr+1 |
| 11) | Передача Aв задачу Тr-m |

|  |  |
| --- | --- |
|  | ТP-m..TP-1 |
| 1) | Отримання Z, MO від задачі Тr+1 |
| 2) | Передача Z, MO в задачу Тr-1 |
| 3) | Передача Z, MO в задачу Тr-m |
| 4) | Отримання Bm\*H, СH\*m, MTH\*m від задачі Тr-1 |
| 5) | Передача Bm\*H, СH\*m, MTH\*m в задачу Тr+1 |
| 6) | Обчислення ar=max(СH); |
| 7) | Отримання ar+1 від задачі Тr+1 |
| 8) | Обчислення ar=max(ar, ar+1); |
| 9) | Передача ar в задачу Тr-1 |
| 10) | Отримання ar від задачі Тr-m |
| 11) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 12) | Отримання AH\*m від задачі Тr+1 |
| 13) | Передача Aв задачу Тr-1 |

|  |  |
| --- | --- |
|  | ТP |
| 1) | Введення Z, MO |
| 2) | Передача Z, MO в задачу ТP-1 |
| 3) | Передача Z, MO в задачу ТP-m |
| 4) | Отримання BH, СH, MTH від задачі ТP-1 |
| 5) | Обчислення aP=max(СH); |
| 6) | Передача aP в задачу ТP-1 |
| 7) | Отримання ar від задачі Тr-m |
| 8) | Обчислення AH=BH\*аp+Z\*(MO\*MTH) |
| 9) | Передача Aв задачу Тr-1 |
|  | Тr (r >Trii && r+1%m !=0 && r < P-m) |
| 1) | Отримання B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H від задачі Тr+m |
| 2) | Отримання Z, MO від задачі Тr-m |
| 3) | Передача B(m-r)\*(m-r)Н, С(m-r)\*(m-r)\*Н, MT(m-r)\*(m-r)\*H в задачу Тr+1 |
| 4) | Передача Z, MO в задачу Тr-1 |
| 5) | Обчислення ar=max(СH); |
| 6) | Отримання ar+1 від задачі Тr+1 |
| 7) | Обчислення ar=max(ar, ar+1); |
| 8) | Передача ar в задачу Тr-1 |
| 9) | Отримання ar від задачі Тr-m |
| 10) | Передача ar в задачу Тr+m |
| 11) | Обчислення AH=BH\*аr+Z\*(MO\*MTH) |
| 12) | Отримання AH\*m від задачі Тr+1 |
| 13) | Передача Aв задачу Тr-1 |

**Етап 3.** Розроблення структурної схеми взаємодії задач.

Т2..Tm

Send Z, MO to r-1

Send B, C, MT to r+m

Recv Z, MO fr r+1

Send a to r+1

Send a to r-1

Send Z, MO to r+m

Recv a fr r+1

Send A to r-1

Recv A fr r+1

Recv B, C, MT fr r-1

T1

Send B, C, MT to r+1

Send B, C, MT to r+m

Recv Z, MO fr r+1

Send a to r+m

Send a to r+1

Recv A fr r+1

Recv a fr r+1

Recv a fr r+m

Recv a fr r-1

Send a to r+m

recv A fr r+m

Тm..P

(r+1%m == 0)

Recv Z, MO fr r+1

Send a to r-m

Recv B, C, MT fr r-m

Recv a fr r-m

Send B, C, MT to r+m

Recv a fr r+m

Recv A fr r+1

Send A to r-m

Recv A fr r+m

Тm

Send Z, MO to r-m

Recv Z, MO fr r+m

Send a to r+m

Send a to r-1

Recv B, C, MT fr r-1

Send A to r-1

Recv a fr r+1

Recv a fr r-1

Send a to r+m

Trii

|| (r >m && r+1%m !=0

&& r < P-m)

Send Z, MO to r-1

Send B, C, MT to r+1

Recv Z, MO fr r+m

Send a to r+m

Send a to r-1

Recv B, C, MT fr r-m

Recv a fr r-m

Recv a fr r+1

Send A to r-1

Recv A fr r+1

Тm..P

(r%m == 0)

Recv Z, MO fr r+m

Send a to r+m

Send a to r-1

Recv B, C, MT fr r-m

Send Z, MO to r-m

Recv a fr r-m

Send A to r-1

ТP-m..TP-1

Send Z, MO to r-1

Send B, C, MT to r+1

Recv Z, MO fr r+1

Send a to r-1

Recv B, C, MT fr r-1

Send Z, MO to r+m

Recv a fr r-m

Recv a fr r+1

Send A to r-1

Recv A fr r+1

ТP+1-m

Send B, C, MT to r+1

Recv a to r-m

Recv Z, MO fr r+1

Send a to r-m

Recv B, C, MT fr r-m

Recv a fr r+1

Recv A fr r+1

Send A to r-m

Тr

(r >Trii && r+1%m !=0

&& r < P-m)

Send B, C, MT to r+1

Recv Z, MO fr r+m

Send a to r+m

Send a to r-1

Recv B, C, MT fr r-1

Send Z, MO to r+m

Recv a fr r-m

Recv a fr r+1

Send A to r-1

Recv A fr r+1

ТP

Send Z, MO to r-1

Send a to r-1

Send Z, MO to r-m

Recv B, C, MT fr r-1

Recv a fr r+m

Send A to r-1

**Етап 4.** Розроблення програми**.**

/\* Глухенько Костянтин Анатолійович, група ІО-93

лаб№ 9. MPI

A=B\*max(C)+Z\*(MO\*MT)

\*/

#include "stdafx.h"

#include "mpi.h"

#include <iostream>

#include <math.h>

using namespace std;

int N = 4;

int sm = N\*N;

int r, P;

int H, m;

int\* A = new int[N];

int\* B = new int[N];

int\* C= new int[N];

int\* Z= new int[N];

int\* MO= new int[sm];

int\* MT= new int[sm];

const int value = 1;

MPI\_Status s;

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

{

MPI\_Init(&argc, &argv);

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

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

H = N/P;

m = sqrt((double)P);

int\* Z = new int[N];

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

cout << "Task " << r+1 << " started " << endl;

if(r == (P-1)){

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

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

MO[(i\*N)+j] = value;;

Z[i] = value;

}

MPI\_Send(Z, N, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(Z, N, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

int len = H;

int size = 2\*len + len\*N;

int\* buf2 = new int[size];

MPI\_Recv(buf2, size, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD, &s);//======<-----------

B = new int[len];

C = new int[len];

MT = new int[len \* N];

int index2 = 0;

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

B[i] = buf2[index2];

index2++;

}

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

C[i] = buf2[index2];

index2++;

}

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

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

MT[(i \* N) + j] = buf2[index2];

index2++;

}

delete[] buf2;

}

else if(r == 0){

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

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

MT[(i\*N)+j] = value;

B[i] = value;

C[i] = value;

}

int len = (m - r) \* (m - r) \* H;

int size = (m - r - 1) \* (m - r - 1) \* H \* 2 + (m - r - 1) \* (m - r - 1) \* H \* N;

int\* buf = new int[size];

int index = 0;

for (int i = 0; i < len - (m - r) \* H; i += H) {

if (i % ((m - r) \* H) != 0) {

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

buf[index] = B[i + j];

index++;

}

}

}

for (int i = 0; i < len - (m - r) \* H; i += H) {

if (i % ((m - r) \* H) != 0) {

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

buf[index] = C[i + j];

index++;

}

}

}

for (int i = 0; i < len - (m - r) \* H; i += H) {

if (i % ((m - r) \* H) != 0) {

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

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

buf[index] = MT[((i + j) \* N) + z];

index++;

}

}

}

MPI\_Send(buf, size, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD);//======= -------->

delete[] buf;

len = (m - r) \* (m - r) \* H;

size = ((H \* (m - r - 1)) \* 2 + H \* (m - r - 1) \* N) \* 2;

int\* buf1 = new int[size];

index = 0;

for (int i = (m - r) \* H; i < len; i += (m - r) \* H) {

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

buf1[index] = B[i + j];

index++;

}

}

for (int i = (len - (m - r) \* H) + H; i < len; i++) {

buf1[index] = B[i];

index++;

}

for (int i = (m - r) \* H; i < len; i += (m - r) \* H) {

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

buf1[index] = C[i + j];

index++;

}

}

for (int i = (len - (m - r) \* H) + H; i < len; i++) {

buf1[index] = C[i];

index++;

}

for (int i = (m - r) \* H; i < len; i += (m - r) \* H) {

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

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

buf1[index] = MT[((i + j) \* N) + k];

index++;

}

}

}

for (int i = (len - (m - r) \* H) + H; i < len; i++) {

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

buf1[index] = MT[(i \* N) + j];

index++;

}

}

MPI\_Send(buf1, size, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD);//======= ======== --------UP

delete[] buf1;

MPI\_Recv(Z, N, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

}

else if( r < m){//snizy

int len = (m - r) \* (m - r) \* H;

int size = 2\*len + len\*N;

int\* buf2 = new int[size];

MPI\_Recv(buf2, size, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD, &s);//====== <-----------

B = new int[len];

C = new int[len];

MT = new int[len \* N];

int index2 = 0;

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

B[i] = buf2[index2];

index2++;

}

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

C[i] = buf2[index2];

index2++;

}

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

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

MT[(i \* N) + j] = buf2[index2];

index2++;

}

delete[] buf2;

if((r+1)%m != 0){// !m

size = (m - r - 1) \* (m - r - 1) \* H \* 2 + (m - r - 1) \* (m - r - 1) \* H \* N;

int\* buf = new int[size];

int index = 0;

for (int i = 0; i < len - (m - r) \* H; i += H) {

if (i % ((m - r) \* H) != 0) {

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

buf[index] = B[i + j];

index++;

}

}

}

for (int i = 0; i < len - (m - r) \* H; i += H) {

if (i % ((m - r) \* H) != 0) {

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

buf[index] = C[i + j];

index++;

}

}

}

for (int i = 0; i < len - (m - r) \* H; i += H) {

if (i % ((m - r) \* H) != 0) {

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

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

buf[index] = MT[((i + j) \* N) + z];

index++;

}

}

}

MPI\_Send(buf, size, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD);//=== ------------->

delete[] buf;

size = ((H \* (m - r - 1)) \* 2 + H \* (m - r - 1) \* N) \* 2;

int\* buf3 = new int[size];

index = 0;

for (int i = (m - r) \* H; i < len; i += (m - r) \* H) {

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

buf3[index] = B[i + j];

index++;

}

}

for (int i = (len - (m - r) \* H) + H; i < len; i++) {

buf3[index] = B[i];

index++;

}

for (int i = (m - r) \* H; i < len; i += (m - r) \* H) {

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

buf3[index] = C[i + j];

index++;

}

}

for (int i = (len - (m - r) \* H) + H; i < len; i++) {

buf3[index] = C[i];

index++;

}

for (int i = (m - r) \* H; i < len; i += (m - r) \* H) {

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

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

buf3[index] = MT[((i + j) \* N) + k];

index++;

}

}

}

for (int i = (len - (m - r) \* H) + H; i < len; i++) {

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

buf3[index] = MT[(i \* N) + j];

index++;

}

}

MPI\_Send(buf3, size, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD);//==================== --------UP

delete[] buf3;

MPI\_Recv(Z, N, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

MPI\_Send(Z, N, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

}

else{

MPI\_Recv(Z, N, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

MPI\_Send(Z, N, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

}

}else{

int len = 0;

if ((r > m - 1) && (r % (m - 1) != 0))

if(r > (m-(r%m))\*(m-1))

len = (m-(r%m))\*H;

else

len = ((m - (r % m) - (r/(m-1)))+(m - (r % m)) -1) \* H;

else

if((r+1)%m == 0)

len = H;

else

len = (r/(m-1))\*H;

int size = 2\*len + len\*N;

int\* buf2 = new int[size];

if (r > (m-(r%m))\*(m-1)){//Up i,i

if(r > (P- m)){

MPI\_Recv(Z, N, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

MPI\_Send(Z, N, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(Z, N, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

}

else{

MPI\_Recv(Z, N, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

MPI\_Send(Z, N, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

}

MPI\_Recv(buf2, size, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD, &s);//================= <-------

}else

MPI\_Recv(buf2, size, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD, &s);//================= down

int index2 = 0;

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

B[i] = buf2[index2];

index2++;

}

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

C[i] = buf2[index2];

index2++;

}

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

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

MT[(i \* N) + j] = buf2[index2];

index2++;

}

delete[] buf2;

if((r+1)%m != 0){

if ((r > m - 1) && (r % (m - 1) != 0)) {// !i,i

if (r > (m-(r%m))\*(m-1)){ //up i,i

size = 2\*len + len\*N - 2\*H - H\*N;

int\* buf = new int[size];

int index = 0;

for (int i = H; i < len; i++) {

buf[index] = B[i];

index++;

}

for (int i = H; i < len; i++) {

buf[index] = C[i];

index++;

}

for (int i = H; i < len; i++) {

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

buf[index] = MT[(i \* N) + j];

index++;

}

}

MPI\_Send(buf, size, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD);//==================== -------->

delete[] buf;

}

else{

size = 2\*len + len\*N - 2\*H - H\*N;

int\* buf = new int[size];

int index = 0;

for (int i = H; i < len; i++) {

buf[index] = B[i];

index++;

}

for (int i = H; i < len; i++) {

buf[index] = C[i];

index++;

}

for (int i = H; i < len; i++) {

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

buf[index] = MT[(i \* N) + j];

index++;

}

}

MPI\_Send(buf, size, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD);//==================== --------UP

delete[] buf;

MPI\_Recv(Z, N, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

if(r%m !=0){

MPI\_Send(Z, N, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

}

}

}

else{// i, i

if(r%m == 0){

MPI\_Recv(Z, N, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

}

else{

MPI\_Recv(Z, N, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

MPI\_Recv(MO, sm, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

}

size = 2\*len + len\*N - 2\*H - H\*N;

int\* buf = new int[size];

int index = 0;

for (int i = H; i < len; i++) {

buf[index] = B[i];

index++;

}

for (int i = H; i < len; i++) {

buf[index] = C[i];

index++;

}

for (int i = H; i < len; i++) {

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

buf[index] = MT[(i \* N) + j];

index++;

}

}

MPI\_Send(buf, size, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD);//==================== -------->

delete[] buf;

if(r%m != 0){

MPI\_Send(Z, N, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

MPI\_Send(MO, sm, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

}

}

}

}

//==============================

int ai = -999999;

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

if(C[i] > ai)

ai = C[i];

int size = 1;

int\* a = new int[size];

if((r+1)%m != 0){

MPI\_Recv(a, size, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

if(ai < a[0])

ai = a[0];

}

if((r < (P - m))&&(r%m == 0)){

MPI\_Recv(a, size, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

if(ai < a[0])

ai = a[0];

}

a[0] = ai;

if(r%m != 0)

MPI\_Send(a, size, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

else if(r != 0)

MPI\_Send(a, size, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

//==========================

if((r != 0)&&(r < m))

MPI\_Recv(a, size, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD, &s);

if(r == 0)

a[0] = ai;

if(r < m){

if((r+1)%m !=0)

MPI\_Send(a, size, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD);

MPI\_Send(a, size, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD);

}

else{

MPI\_Recv(a, size, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD, &s);

if(r < (P - m))

MPI\_Send(a, size, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD);

}

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

A[i] = 0;

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

int t = 0;

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

t+=MO[(k\*N)+j]\*MT[(i\*N)+k];

A[i] += t\*Z[j];

}

A[i] += B[i]\*a[0];

}

//===================

int\* Ai = new int[N];

if((r+1)%m != 0){

int size = (m-((r+1)%m))\*H;

MPI\_Recv(Ai, size, MPI\_INTEGER, r+1, 0, MPI\_COMM\_WORLD, &s);

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

A[i] = Ai[i-H];

}

//=====================

if((r < (P - m))&&(r%m == 0)){

int size = N-((r+m)\*H);

MPI\_Recv(Ai, size, MPI\_INTEGER, r+m, 0, MPI\_COMM\_WORLD, &s);

for(int i = H\*m; i < size+H\*m; i++)

A[i] = Ai[i-(H\*m)];

}

if(r%m != 0){

int size = (m-(r%m))\*H;

MPI\_Send(A, size, MPI\_INTEGER, r-1, 0, MPI\_COMM\_WORLD);

}

else if(r != 0){

int size = N-(r\*H);

MPI\_Send(A, size, MPI\_INTEGER, r-m, 0, MPI\_COMM\_WORLD);

}

else if(r == 0){

if(N < 10)

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

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

else

cout << A[0] << endl;

}

cout << "Task " << r+1 << " finished " << endl;

MPI\_Finalize();

return 0;

}