Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №8**

**з дисципліни «Програмування паралельних комп’ютерних систем»**

**на тему «MPI»**

Виконав:

студент ІІI курсу

групи ІО-53

Лісовий В. О.

Перевірив:

доц. Корочкін О. В.

Київ – 2018

**Технічне завдання**

Розробити програму для розв’язання ПКС із ЛП (структура на рис. 1) математичної задачі: MA = MB \* MC + (B \* C) \* MO

Мова програмування: C/C+, MPI.



Рис. 1 Структурна схема ПКС

**Виконання роботи**

**Етап 1. Побудова паралельного алгоритму**

1. ai = (BH \* CH), i = 1..12
2. a = a + ai
3. MAh = MB \* MCh + a \* MOh;

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

|  |
| --- |
| **T0** |
| 1. Введення MB, MC, B, C, MO |
| 2. Передати у Т1 MB, MC3h, B3h, C3h, MO3h |
| 3. Передати у Т4 MB, MC3h, B3h, C3h, MO3h |
| 4. Передати у Т7 MB, MC4h, B4h, C4h, MO4h |
| 5. Передати у Т11 MB, MCh, Bh, Ch, MOh |
| 6. Обчислення a=scalar(Bh, Ch) |
| 7. Прийняти від Т1 a1 |
| 8. Обчислення a += a1 |
| 9. Прийняти від Т4 a4. |
| 10. Обчислення a += a4 |
| 11. Прийняти від Т7 a7 |
| 12. Обчислення a += a7 |
| 13. Прийняти від Т11 a11 |
| 14. Обчислення a += a11 |
| 15. Передати у T1, T4, T7, T11 a |
| 16. Обчислення MAh = MB \* MCh + a \* MOh |
| 17. Прийняти від Т1 MA3H |
| 18. Прийняти від Т4 MA3H |
| 19. Прийняти від Т7 MA4H |
| 20. Прийняти від Т11 MAH |
| 21. Виведення MA |
|  |
| **T1** |
| 1. Прийняти від Т0 MB, MC3h, B3h, C3h, MO3h |
| 2. Передати у Т2 i T3 MB, MCh, Bh, Ch, MOh |
| 3. Обчислення a1=scalar(Bh, Ch) |
| 4. Прийняти від Т2 a2. |
| 5. Обчислення a1 += a2 |
| 6. Прийняти від Т3 a3 |
| 7. Обчислення a1 += a3 |
| 8. Передати у Т0 a1 |
| 9. Прийняти від Т0 a |
| 10. Передати у Т2, Т3 a |
| 11. Обчислення MAh = MB \* MCh + a \* MOh. |
| 12. Прийняти від Т2 MAH. |
| 13. Прийняти від Т3 MAH. |
| 14. Передати у Т0 MA3H. |

|  |
| --- |
| **T2** |
| 1. Прийняти від Т1 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a2 = scalar(Bh, Ch) |
| 3. Передати у Т1 a2 |
| 4. Прийняти від Т1 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т1 MAH. |

|  |
| --- |
| **T3** |
| 1. Прийняти від Т1 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a3 = scalar(Bh, Ch) |
| 3. Передати у Т1 a3 |
| 4. Прийняти від Т1 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т3 MAH. |

|  |
| --- |
| **T4** |
| 1. Прийняти від Т0 MB, MC3h, B3h, C3h, MO3h |
| 2. Передати у Т5 i T6 MB, MCh, Bh, Ch, MOh |
| 3. Обчислення a4=scalar(Bh, Ch) |
| 4. Прийняти від Т5 a5. |
| 5. Обчислення a4 += a5 |
| 6. Прийняти від Т6 a6 |
| 7. Обчислення a4 += a6 |
| 8. Передати у Т0 a4 |
| 9. Прийняти від Т0 a |
| 10. Передати у Т5 Т6 a |
| 11. Обчислення MAh = MB \* MCh + a \* MOh. |
| 12. Прийняти від Т5 MAH. |
| 13. Прийняти від Т6 MAH. |
| 14. Передати у Т0 MA3H. |

|  |
| --- |
| **T5** |
| 1. Прийняти від Т4 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a5 = scalar(Bh, Ch) |
| 3. Передати у Т4 a5 |
| 4. Прийняти від Т4 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т4 MAH. |

|  |
| --- |
| **T6** |
| 1. Прийняти від Т4 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a6 = scalar(Bh, Ch) |
| 3. Передати у Т4 a6 |
| 4. Прийняти від Т4 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т4 MAH. |

|  |
| --- |
| **T7** |
| 1. Прийняти від Т0 MB, MC4h, B4h, C4h, MO4h |
| 2. Передати у T8, Т9 i T10 MB, MCh, Bh, Ch, MOh |
| 3. Обчислення a7=scalar(Bh, Ch) |
| 4. Прийняти від Т8 a8. |
| 5. Обчислення a7 += a8 |
| 6. Прийняти від Т9 a9 |
| 7. Обчислення a7 += a9 |
| 6. Прийняти від Т10 a10 |
| 8. Обчислення a7 += a10 |
| 9. Передати у Т0 a7 |
| 10. Прийняти від Т0 a |
| 11. Передати у T8, Т9 i Т10 a |
| 12. Обчислення MAh = MB \* MCh + a \* MOh. |
| 13. Прийняти від Т8 MAH. |
| 14. Прийняти від Т9 MAH. |
| 15. Прийняти від Т10 MAH. |
| 16. Передати у Т0 MA4H. |

|  |
| --- |
| **T8** |
| 1. Прийняти від Т7 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a8 = scalar(Bh, Ch) |
| 3. Передати у Т7 a8 |
| 4. Прийняти від Т7 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т7 MAH. |

|  |
| --- |
| **T9** |
| 1. Прийняти від Т7 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a9 = scalar(Bh, Ch) |
| 3. Передати у Т7 a9 |
| 4. Прийняти від Т7 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т7 MAH. |

|  |
| --- |
| **T10** |
| 1. Прийняти від Т7 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a10 = scalar(Bh, Ch) |
| 3. Передати у Т7 a10 |
| 4. Прийняти від Т7 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т7 MAH. |

|  |
| --- |
| **T11** |
| 1. Прийняти від Т0 MB, MCh, Bh, Ch, MOh |
| 2. Обчислення a11 = scalar(Bh, Ch) |
| 3. Передати у Т0 a11 |
| 4. Прийняти від Т0 a. |
| 5. Обчислення MAh = MB \* MCh + a \* MOh. |
| 6. Передати у Т0 MAH. |

**Етап 3. Розробка програми**

#include "functions.h"

using namespace std;

int tid;

int n = 12, p, num = 1;

float h;

int leftIndex, rightIndex;

int a;

int \*B, \*C;

int \*MA, \*MB, \*MC, \*MO;

int \*Bi, \*Ci, \*MAi, \*MCi, \*MOi;

int \*scalars;

void init\_memory();

void free\_memory();

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

MPI\_Comm graph\_comm;

int \*index = new int[12]{ 4, 7, 8, 9, 12, 13, 14, 18, 19, 20, 21, 22};

int \*edges = new int[22]{1,4,7,11, 0,2,3, 1, 1, 0,5,6, 4, 4, 0,8,9,10, 7, 7, 7, 0};

MPI\_Init(&argc, &argv);

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

MPI\_Graph\_create(MPI\_COMM\_WORLD, p, index, edges, false, &graph\_comm);

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

init\_memory();

h = (float) n / p;

leftIndex = (int)(tid \* h);

rightIndex = (int)((tid + 1) \* h);

cout << "Thread" << tid << " start" << endl;

// T0 input

if (tid == 0) {

fillVector(num, B, n);

fillVector(num, C, n);

fillMatrix(num, MB, n);

fillMatrix(num, MC, n);

fillMatrix(num, MO, n);

}

// Send MB, MC, B, C, MO

MPI\_Bcast(MB, n\*n, MPI\_INT, 0, graph\_comm);

MPI\_Scatter(B, (int) h, MPI\_INT, Bi, (int) h, MPI\_INT, 0, graph\_comm);

MPI\_Scatter(C, (int) h, MPI\_INT, Ci, (int) h, MPI\_INT, 0, graph\_comm);

MPI\_Scatter(MC, (int)(h\*n), MPI\_INT, MCi, (int)(h\*n), MPI\_INT, 0, graph\_comm);

MPI\_Scatter(MO, (int)(h\*n), MPI\_INT, MOi, (int)(h\*n), MPI\_INT, 0, graph\_comm);

// Scalar

a = scalar(Bi, Ci, (int) h);

MPI\_Gather(&a, 1, MPI\_INT, scalars, 1, MPI\_INT, 0, graph\_comm);

if (tid == 0) {

int buf = 0;

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

buf += scalars[i];

}

a = buf;

}

MPI\_Bcast(&a, 1, MPI\_INT, 0, graph\_comm);

F(MAi, MB, MCi, a, MOi, (int)h, n);

MPI\_Gather(MAi, (int)(h \* n), MPI\_INT, MA, (int)(h \* n), MPI\_INT, 0, graph\_comm);

if (tid == 0) {

cout << "Thread " << tid << ": MA=" << endl;

printMatrix(MA, n);

}

cout << "Thread" << tid << " finish"<< endl;

free\_memory();

MPI\_Finalize();

return 0;

}

void init\_memory() {

MA = (int \*)calloc(n \* n, 4);

MB = (int \*)malloc(n \* n \* 4);

MC = (int \*)malloc(n \* n \* 4);

MO = (int \*)malloc(n \* n \* 4);

MCi = (int \*)malloc((int)(n \* n \* 4));

MOi = (int \*)malloc((int)(n \* n \* 4));

MAi = (int \*)malloc((int)(n \* n \* 4));

B = (int \*)malloc(n \* 4);

C = (int \*)malloc(n \* 4);

Bi = (int \*)malloc((int)(n \* 4));

Ci = (int \*)malloc((int)(n \* 4));

scalars = (int \*)malloc(p \* 4);

}

void free\_memory() {

free(MA);

free(MB);

free(MC);

free(MO);

free(MAi);

free(MOi);

free(MCi);

free(B);

free(C);

free(Bi);

free(Ci);

free(scalars);

}