НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

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

Виконав:

студент 3 курсу

ФІОТ гр. ІО-34

Кривоносов Олексій

Перевірив:

Корочкін О. В.

Київ – 2016 р.

Тема: Програмування для комп’ютерних систем зі спільною пам’яттю. Бібліотека Open MP

Розробити програму для розв’язання в ПКС зі СП (структура на рис. Рисунок 1) математичної задачі: A = sort(α\*B + Z\*(MO\*MK))

Бібліотека: Open MP.

Засоби організації взаємодії: бар’єри, критичні секції, замки.



Структурна схема ПКС

**Виконання роботи:**

**Етап 1. Побудова паралельного алгоритму**

1. ВH = α\*BH +Z(MO\*MKH)
2. *BH =*
3. *B2H=merge(BH, BH)*
4. *B=merge(B2H, B2H)*

Спільні ресурси:Z, MO, α

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

|  |  |
| --- | --- |
| **Задача Т1** | КД, бар’єри |
| 1. Введення B,α |  |
| 2. Бар’єр. Синхронизація вводу | бар’єр |
| 3. Копіювання α1 = α | КД |
| 4. Копіювання Z1 = Z, MO1 = MO | КД |
| 5. Обчислення 1 АH = α1\*ВH + Z1\*(MO1\*MKH) |  |
| 6. Обчислення 2 *АH =* |  |
| 7. Бар’єр. Синхронизація обчислення 2 | бар’єр |
| 9. Бар’єр. Синхронизація обчислення 3 | бар’єр |
| 9. Обчислення 4 А= *merge(А2H, А2H)* |  |
| 10. Вивід А |  |
| **Задача Т2** | КД, бар’єри |
| 1. Бар’єр. Синхронизація вводу | бар’єр |
| 2. Копіювання α2 = α | КД |
| 3. Копіювання Z2 = Z, MO2 = MO | КД |
| 4. Обчислення 1 АH = α2\*ВH + Z2\*(MO2\*MKH) |  |
| 5. Олчислення 2 *АH =* |  |
| 6. Бар’єр. Синхронизація обчислення 2 | бар’єр |
| 7. Обчислення 3*A2H=merge(AH, AH)* |  |
| 8. Бар’єр. Синхронизація обчислення 3 | бар’єр |

|  |  |
| --- | --- |
| **Задача T3** | КД, бар’єри |
| 1. Введення Z, MO |  |
| 2. Бар’єр. Синхронизація вводу | бар’єр |
| 3. Копіювання α3 = α | КД |
| 4. Копіювання Z3 = Z, MO3 = MO | КД |
| 5. Обчислення 1 АH = α3\*ВH + Z3\*(MO3\*MKH) |  |
| 6. Обчислення 2 *АH =* |  |
| 7. Бар’єр. Синхронизація обчислення 2 | бар’єр |
| 8. Обчислення 3 *A2H=merge(AH, AH)* |  |
| 9. Бар’єр. Синхронизація обчислення 3 | бар’єр |

|  |  |
| --- | --- |
| **Задача Т4** | КД, бар’єри |
| 1. Введення MK |  |
| 2. Бар’єр. Синхронизація вводу | бар’єр |
| 3. Копіювання α4 = α | КД |
| 4. Копіювання Z4 = Z, MO4 = MO | КД |
| 5. Обчислення 1АH = α4\*ВH + Z4\*(MO4\*MKH) |  |
| 6.Обчислення 2 *АH =* |  |
| 7. Бар’єр. Синхронизація обчислення 2 | бар’єр |
| 8. Бар’єр. Синхронизація обчислення 3 | бар’єр |

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

На основі алгоритму,розробленому у попередньому етапі, було розроблено структурну схему взаємодії задач. Вона дозволяє наочно контролювати бар’єри та критичні ділянки. Крім того, на структурній схемі уводяться також замок, та критична секція, що будуть використовуватись в програмі.

Етап 4. Розроблення програми

Файл pro2\_lab4.cpp

/\*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* Програмування паралельний комп'ютерних сисем \*

\* Лабораторна робота №4. Бібліотека OpenMP \*

\* \*

\* Завдання: A = A=sort(alfa\*B+Z(MO\*MK)) \*

\* \*

\* Автор Кривоносов Олексій \*

\* Група ІО-34 \*

\* Дата 02.04.16 \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\*/

#include "stdafx.h"

#include "omp.h"

#include <windows.h>

#include <iostream>

#include "operations.h"

#include <ctime>

#include <clocale>

using namespace std;

#pragma comment(linker, "/stack:160000000")

const int P = 4;

int main()

{

vector A = new int[N], B, Z;

matrix MO, MK;

matrix BufMatr1;

vector BufVect1;

int alfa;

BufMatr1 = initMatrix();

BufVect1 = initVector();

B = initVector();

Z = initVector();

MO = initMatrix();

MK = initMatrix();

const int H = N / P;

omp\_lock\_t lock\_Copy;

setlocale(LC\_ALL, "Russian");

omp\_init\_lock(&lock\_Copy);

omp\_set\_dynamic(0);

omp\_set\_num\_threads(P);

#pragma omp parallel

{

int tid = omp\_get\_thread\_num();

cout << "Задача " << tid+1 << " стартувала" << endl;

/\* Введення B,α у задачі Т1\*/

if (tid == 0){

B = inputVector(1);

// B[2] = 0;

alfa = 1;

}

/\* Введення Z, MO у задачі Т3 \*/

if (tid == 2){

Z = inputVector(1);

MO = inputMatrix(1);

}

/\* Введення MK у задачі Т4\*/

if (tid == 3){

MK = inputMatrix(1);

}

/\*Бар’єр для усіх задач. Синхронізація по вводу\*/

#pragma omp barrier

matrix MOid;

vector Zid;

int alfaId;

/\*Копіювати MOid= MO, MRid=MR, αid = α, \*/

#pragma omp\_atomic

{

alfaId = alfa;

}

omp\_set\_lock(&lock\_Copy);

Zid = copyVector(Z);

MOid = copyMatrix(MO);

omp\_unset\_lock(&lock\_Copy);

/\*Обчислення 1 АH = αid\*ВH + Zid\*(MOid\*MKH)\*/

for (int i = tid \* H; i < (tid + 1) \* H; i++) {

BufVect1[i] = 0;

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

BufMatr1[i][j] = 0;

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

BufMatr1[i][j] += MOid[j][z] \* MK[i][j];

}

BufVect1[i] += Zid[j] \* BufMatr1[i][j];

}

}

for (int i = tid\*H; i < (tid + 1)\*H; i++)

{

B[i] = B[i] \* alfaId;

}

for (int i = tid\*H; i < (tid + 1)\*H; i++)

{

A[i] = B[i] + BufVect1[i];

}

/\*Обчислення 2 АH = sort(AH)\*/

for (int i = tid\*H; i < (tid + 1)\*H; i++)

{

for (int j = tid\*H; j < ((tid + 1)\*H)-1; j++)

{

if (A[j + 1] < A[j])

{

int tmp = A[j + 1];

A[j + 1] = A[j];

A[j] = tmp;

}

}

}

/\*Бар’єр. Синхронизація обчислення 2\*/

#pragma omp barrier

/\*Обчислення 3 A2H=merge(AH, AH)\*/

if (tid == 1){

merge(0, H, H,H\*2,A);

}

/\*Обчислення 3 A2H=merge(AH, AH)\*/

if (tid == 2){

merge(H\*2, H\*3, H\*3, H \* 4, A);

}

/\*Бар’єр. Синхронизація обчислення 3\*/

#pragma omp barrier

if (tid == 0){

/\*Обчислення 4 А= merge(А2H, А2H)\*/

merge(0, H \* 2, H \* 2, H \* 4, A);

/\*Вивід А\*/

output(A);

}

cout << "Задача " << tid+1 << " завершилась" << endl;

}

int stop1;

cin >> stop1;

return 0;

}

Файл operation.h

/\*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* Програмування паралельний комп'ютерних сисем \*

\* Лабораторна робота №4. Бібліотека OpenMP \*

\* \*

\* Завдання: A = A=sort(alfa\*B+Z(MO\*MK)) \*

\* \*

\* Автор Кривоносов Олексій \*

\* Група ІО-34 \*

\* Дата 02.04.16 \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\*/

typedef int\* vector;

typedef int\*\* matrix;

const int N = 8;

vector inputVector(int);

matrix inputMatrix(int);

void output(vector);

void sort(int, int);

void merge(int, int, int,int, vector);

vector copyVector(vector);

matrix copyMatrix(matrix);

matrix initMatrix();

vector initVector();

Файл operations.cpp

/\*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* \*

\* Програмування паралельний комп'ютерних сисем \*

\* Лабораторна робота №4. Бібліотека OpenMP \*

\* \*

\* Завдання: A = A=sort(alfa\*B+Z(MO\*MK)) \*

\* \*

\* Автор Кривоносов Олексій \*

\* Група ІО-34 \*

\* Дата 02.04.16 \*

\* \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\*/

#include "stdafx.h"

#include <windows.h>

using namespace std;

#include "operations.h"

#include <iostream>

void merge(int start1, int end1, int start2, int end2, int Z[]){

vector L;

int globstart = start1;

L = copyVector(Z);

while (start1 != end1 && start2 != end2){

if (L[start1] <= L[start2]){

Z[globstart] = L[start1];

globstart++;

start1++;

}

else if (L[start2] < L[start1]){

Z[globstart] = L[start2];

globstart++;

start2++;

}

}

if (start1 == end1)

while (start2 != end2){

Z[globstart] = L[start2];

globstart++;

start2++;

}

else if (start2 == end2){

while (start1 != end1){

Z[globstart] = L[start1];

globstart++;

start1++;

}

}

}

vector inputVector(int value){

vector result = new int[N];

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

{

result[i] = value;

}

return result;

}

matrix inputMatrix(int value){

matrix result = new vector[N];

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

{

result[i] = new int[N];

}

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

{

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

{

result[i][j] = value;

}

}

return result;

}

void output(vector v) {

if (N <= 20) {

cout << endl;

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

cout << v[i] << " ";

}

cout << endl;

}

}

vector copyVector(vector v){

vector result = new int[N];

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

{

result[i] = v[i];

}

return result;

}

matrix copyMatrix(matrix m){

matrix result = new vector[N];

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

{

result[i] = new int[N];

}

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

{

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

{

result[i][j] = m[i][j];

}

}

return result;

}

matrix initMatrix(){

matrix result = new vector[N];

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

{

result[i] = new int[N];

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

{

result[i][j] = 0;

}

}

return result;

}

vector initVector(){

vector result = new int[N];

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

{

result[i] = 0;

}

return result;

}