Національний технічний університет України

«Київський Політехнічний Інститут»

Факультет інформатики і обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №1**

**З предмету «Паралельні та розподілені обчислення»**

**Бібліотека Win32**

Виконав:

Студент  
IІІ курсу ФІОТ  
групи ІО-12  
Бута С. О.

Залікова книжка №1205

Київ-2014

**Техническое задание**

1. Структура ПКС с ОП:

ОП

1

MZ, MK

2

MO

3

A, B, C, l

1. Задача: .
2. Язык программирования: C++, библиотека Win32.
3. Средства взаимодействия задач: множественные семафоры, события, мютексы, критические секции.

**Выполнение работы**

1. Разработка параллельного математического алгоритма.
2. DH = C \* MZH ; OP: C;
3. AH = BH + l\*D\*(MO\*MKH); OP: l, D, MO.
4. Разработка алгоритмов процессов.

**Задача Т1:**

1. Ввод MZ, MK
2. Сигнал Т2, Т3 о завершении ввода S2,3;1
3. Ждать завершение ввода в T2, T3 W2,3;1
4. Копия С КУ
   1. C1 = C
5. Счет №1
   1. DH = C1 \* MZH
6. Сигнал Т2, Т3 о завершении счета №1 S2,3;2
7. Ждать завершение счета №1 в Т2, Т3 W2,3;2
8. Копии l, D, MO КУ
   1. l1 = l;
   2. D1 = D;
   3. MO1 = MO;
9. Счет №2
   1. AH = BH + l1\*D1\*(MO1\*MKH)
10. Сигнал T3 о завершении счета S3,31

**Задача Т2:**

1. Ввод MO
2. Сигнал Т1, Т3 о завершении ввода S1,3;1
3. Ждать завершение ввода в T1, T3 W1,3;1
4. Копия С КУ
   1. C2 = C
5. Счет №1
   1. DH = C2 \* MZH
6. Сигнал Т1, Т3 о завершении счета №1 S1,3;2
7. Ждать завершение счета №1 в Т1, Т3 W1,3;2
8. Копии l, D, MO КУ
   1. l2 = l;
   2. D2 = D;
   3. MO2 = MO;
9. Счет №2
   1. AH = BH + l2\*D2\*(MO2\*MKH)
10. Сигнал T3 о завершении счета S3,32

**Задача Т3:**

1. Ввод B, C, l
2. Сигнал Т1, Т2 о завершении ввода S1,2;1
3. Ждать завершение ввода в T1, T2 W1,2;1
4. Копия С КУ
   1. C3 = C;
5. Счет №1
   1. DH = C3 \* MZH
6. Сигнал Т1, Т2 о завершении счета №1 S1,2;2
7. Ждать завершение счета №1 в Т1, Т2 W1,2;2
8. Копии l, D, MO КУ
   1. l3 = l;
   2. D3 = D;
   3. MO3 = MO;
9. Счет №2
   1. AH = BH + l3\*D3\*(MO3\*MKH)
10. Ждать завершение счета №2 в Т1, Т2 W1,2;3
11. Вывод А

**Разработка схемы взаимодействия процессов:**



**Разработка программы.**

**Листинг**:

Исходный код PRO\_Lab\_2(cpp).cpp

// Лабораторна робота №2. Win32

// Бута С.О.

// Математична операція: A = B + l\*(C\*MZ)\*(MO\*MK)

// Дата: 10.04.14

#include "stdafx.h"

#include <iostream>

#include <Windows.h>

#include "Vector.h"

#include "Matrix.h"

const int STACK\_SIZE = 300000;

const int N = 3;

const int P = 3;

const int H = N/P;

const int FILLER = 1;

int l;

Vector A(N), B(N), C(N), D(N);

Matrix MZ(N), MO(N), MK(N);

HANDLE S1, S2, S3;

HANDLE E1, E2, E3;

HANDLE S4, S5;

HANDLE M;

CRITICAL\_SECTION CrSec;

void Func1(){

std::cout << "Thread 1 started \n";

//Ввод MZ, MK

MZ.input();

MK.input();

//Сигнал Т2, Т3 о завершении ввода

ReleaseSemaphore(S1, 2, NULL);

//Ждать завершение ввода в T2, T3

WaitForSingleObject(S2, INFINITE);

WaitForSingleObject(S3, INFINITE);

//Копия С

WaitForSingleObject(M, INFINITE);

Vector C1(N);

C.copy(C1);

ReleaseMutex(M);

//Счет №1

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

D[j] = 0;

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

D[j] += C1[i] \* MZ[i][j];

}

}

//Сигнал Т2, Т3 о завершении счета №1

SetEvent(E1);

//Ждать завершение счета №1 в Т2, Т3

WaitForSingleObject(E2, INFINITE);

WaitForSingleObject(E3, INFINITE);

//Копии l, D, MO

EnterCriticalSection(&CrSec);

int l1 = l;

Vector D1(N);

D.copy(D1);

Matrix MO1(N);

MO.copy(MO1);

LeaveCriticalSection(&CrSec);

//Счет №2

int acum;

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

A[i] = B[i];

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

acum = 0;

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

acum += MO1[j][k]\*MK[k][i];

}

A[i] += l1\*D1[j]\*acum;

}

}

//Сигнал T3 о завершении счета

ReleaseSemaphore(S4, 1, NULL);

std::cout << "Thread 1 finished \n";

}

void Func2(){

std::cout << "Thread 2 started \n";

//Ввод MO

MO.input();

//Сигнал T1, T3 о завершении ввода

ReleaseSemaphore(S2, 2, NULL);

//Ждать завершение ввода в T1, T3

WaitForSingleObject(S1, INFINITE);

WaitForSingleObject(S3, INFINITE);

//Копия С

WaitForSingleObject(M, INFINITE);

Vector C2(N);

C.copy(C2);

ReleaseMutex(M);

//Счет №1

for( int j = H; j<2\*H; j++){

D[j] = 0;

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

D[j] += C2[i] \* MZ[i][j];

}

}

//Сигнал T1, T3 о завершении счета №1

SetEvent(E2);

//Ждать завершение счета №1 в Т1, Т3

WaitForSingleObject(E1, INFINITE);

WaitForSingleObject(E3, INFINITE);

//Копии l, D, MO

EnterCriticalSection(&CrSec);

int l2 = l;

Vector D2(N);

D.copy(D2);

Matrix MO2(N);

MO.copy(MO2);

LeaveCriticalSection(&CrSec);

//Счет №2

int acum;

for(int i=H; i<2\*H; i++){

A[i] = B[i];

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

acum = 0;

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

acum += MO2[j][k]\*MK[k][i];

}

A[i] += l2\*D2[j]\*acum;

}

}

//Сигнал T3 о завершении счета

ReleaseSemaphore(S5, 1, NULL);

std::cout << "Thread 2 finished \n";

}

void Func3(){

std::cout << "Thread 3 started \n";

//Ввод B, C, l

B.input();

C.input();

l = FILLER;

//Сигнал T1, T2 о завершении ввода

ReleaseSemaphore(S3, 2, NULL);

//Ждать завершение ввода в T1, T2

WaitForSingleObject(S1, INFINITE);

WaitForSingleObject(S2, INFINITE);

//Копия С

WaitForSingleObject(M, INFINITE);

Vector C3(N);

C.copy(C3);

ReleaseMutex(M);

//Счет №1

for( int j = 2\*H; j<N; j++){

D[j] = 0;

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

D[j] += C3[i] \* MZ[i][j];

}

}

//Сигнал T1, T2 о завершении счета №1

SetEvent(E3);

//Ждать завершение счета №1 в Т1, Т2

WaitForSingleObject(E1, INFINITE);

WaitForSingleObject(E2, INFINITE);

//Копии l, D, MO

EnterCriticalSection(&CrSec);

int l3 = l;

Vector D3(N);

D.copy(D3);

Matrix MO3(N);

MO.copy(MO3);

LeaveCriticalSection(&CrSec);

//Счет №2

int acum;

for(int i=2\*H; i<N; i++){

A[i] = B[i];

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

acum = 0;

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

acum += MO3[j][k]\*MK[k][i];

}

A[i] += l3\*D3[j]\*acum;

}

}

//Ждать завершение счета №2 в Т1, Т2

WaitForSingleObject(S4, INFINITE);

WaitForSingleObject(S5, INFINITE);

//Вывод А

if(N<10){

std::cout << "Vector A: \n";

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

std::cout << A[i] << ' ';

}

std::cout << '\n';

}

std::cout << "Thread 3 finished \n";

}

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

{

std::cout << "Main thread started \n";

S1 = CreateSemaphore(NULL, 0, 2, NULL);

S2 = CreateSemaphore(NULL, 0, 2, NULL);

S3 = CreateSemaphore(NULL, 0, 2, NULL);

E1 = CreateEvent(NULL, 1, 0, NULL);

E2 = CreateEvent(NULL, 1, 0, NULL);

E3 = CreateEvent(NULL, 1, 0, NULL);

S4 = CreateSemaphore(NULL, 0, 1, NULL);

S5 = CreateSemaphore(NULL, 0, 1, NULL);

M = CreateMutex(NULL, 0, NULL);

InitializeCriticalSection(&CrSec);

DWORD name1;

DWORD name2;

DWORD name3;

HANDLE thread1;

HANDLE thread2;

HANDLE thread3;

thread1 = CreateThread(NULL, STACK\_SIZE, (LPTHREAD\_START\_ROUTINE) Func1, NULL, 0, &name1);

thread2 = CreateThread(NULL, STACK\_SIZE, (LPTHREAD\_START\_ROUTINE) Func2, NULL, 0, &name2);

thread3 = CreateThread(NULL, STACK\_SIZE, (LPTHREAD\_START\_ROUTINE) Func3, NULL, 0, &name3);

WaitForSingleObject(thread3, INFINITE);

CloseHandle(thread1);

CloseHandle(thread2);

CloseHandle(thread3);

CloseHandle(S1);

CloseHandle(S2);

CloseHandle(S3);

CloseHandle(E1);

CloseHandle(E2);

CloseHandle(E3);

CloseHandle(S4);

CloseHandle(S5);

CloseHandle(M);

DeleteCriticalSection(&CrSec);

std::cout << "Main thread finished \n";

getchar();

return 0;

}