Экзаменационная работа по дисциплине «Параллельное программирование»

Бородаенко Сергей ПВ-31

27.10.14

Задание 9

Разработайте программы для нахождения количества элементов матрицы, значения которых отрицательно.

1. Однопоточное приложение. Проверьте его работоспособность на небольшом объеме исходных данных. + **OpemMP**
2. Приложение с использованием технологии **MPI**. Проверьте его работоспособность на очень большом объеме исходных данных на трех разных количествах узлов.
3. **Код программы**

#include <iostream>

#include <ctime>

#include <omp.h>

#define N 20000

using namespace std;

int ar[N][N];

int GetCount(int ar[N][N], int size);

int GetCountParall(int ar[N][N], int size);

int main()

{

// Инициализация

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

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

ar[i][j] = (rand() % 1000) - 500;

int k;

unsigned int start\_time, search\_time;

// Набор 1

start\_time = clock();

k = GetCount(ar, 5000);

search\_time = clock() -start\_time;

printf("%i time : %u \n", k, search\_time);

// Набор 2

start\_time = clock();

k = GetCount(ar, 10000);

search\_time = clock() - start\_time;

printf("%i time : %u \n", k, search\_time);

// Набор 3

start\_time = clock();

k = GetCount(ar, 20000);

search\_time = clock() - start\_time;

printf("%i time : %u \n\n", k, search\_time);

printf("Parallel:\n");

// Параллельный набор

start\_time = clock();

k = GetCountParall(ar, 5000);

search\_time = clock() - start\_time;

printf("%i time : %u \n", k, search\_time);

// Параллельный набор

start\_time = clock();

k = GetCountParall(ar, 10000);

search\_time = clock() - start\_time;

printf("%i time : %u \n", k, search\_time);

// Параллельный набор

start\_time = clock();

k = GetCountParall(ar, 20000);

search\_time = clock() - start\_time;

printf("%i time : %u \n", k, search\_time);

getchar();

return 0;

}

int GetCount(int ar[N][N], int size)

{

int k = 0;

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

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

if (ar[i][j] < 0)

k++;

return k;

}

int GetCountParall(int ar[N][N], int size)

{

int k = 0;

#pragma omp parallel for

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

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

if (ar[i][j] < 0)

k++;

return k;

}

|  |  |  |  |
| --- | --- | --- | --- |
| Количество данных | 5000 | 10000 | 50000 |
| Однопоточное приложение | 156 | 641 | 2625 |
| OpenMP | 156 | 625 | 2547 |
| MPI |  |  |  |

Коэффициенты ускорения для OpenMP:

1. 156/156 = 1
2. 641/625 = 1,0256
3. 2625/2547 = 1,030624263839812

