Задание

Написать программу на OpenMP и MPI, которая вычисляет распространение тепла в тонком однородном стержне:

*;*

u(x, 0) = g0, u(0, t) = f0(t), u(L, t) = f1(t);

g0(x) = sin(x+0.48);

f0(t) = 0.4618;

f1(t) = 3t+0.882;

Математическая модель

Для решения данной задачи воспользуемся методом сеток.

Расчетные формулы для метода:

U1k+1 = ψ(tk+1);

Uik+1 = Uik + (a2τ/h2)(Ui-1k - 2Uik + Ui+1k), i = 2, 3, …, n-1;

Unk+1 = ψ(tk+1);

tk+1 = tk + τ;

(a2τ/h2) = r < ½;

Код программы(OpenMP)

#include <stdio.h>

#include <omp.h>

#include <math.h>

#include <windows.h>

double g(double x) //Вычисляем значение функции в точке х

{

return sin(x+0.48);

}

const double f0 = 0.4618; //Значение в левой краевой точке

double f1(double t) //Значение в правой краевой точке

{

return 3.0\*t+0.882;

}

int main()

{

//Объявление начальных значений

double a = 1.0, L = 300.0, x = 0.0, time = 0.0, tmax = 100.0, tau = 0.001, h = 0.1;

double r = a\*tau/(h\*h); //Постоянный коэффициент

int n = (int)(L/h)+1;

double \*u = new double[n];

double \*un = new double[n];

double tn = omp\_get\_wtime(); //Начальное время

u[0] = f0; //Начальное значение в левой краевой точке

x+=h; //Шаг по координате

for(int i = 1; i<n-1; i++) //Задаем начальные значения

{

u[i] = g(x);

x+=h;

}

u[n-1] = f1(time); //Начальное значение в правой краевой точке

do{

printf("\r%f", time); // Вывод текущего значения переменной time для отладки

#pragma omp parallel //Открываем параллельную секцию

{

#pragma omp for schedule(dynamic, 1)

for(int i = 1; i<n-1; i++) //Вычисляем новые значения в точках

{

un[i] = u[i]+r\*(u[i-1]-2.0\*u[i]+u[i+1]);

}

#pragma omp for schedule(dynamic, 1)

for(int i = 1; i<n-1; i++) //Переприсваиваем новые значения в массив u

{

u[i] = un[i];

}

}

time += tau; //Шаг по времени

u[0] = f0;

u[n-1] = f1(time);

} while(time<=tmax);

double tk = omp\_get\_wtime(); //Конечное время

FILE \*f;

f = fopen("Temp\_res\_m.dat", "w");

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

fprintf(f, "u[%d] = %f\n", i, u[i]); //Вывод значений

fprintf(f, "\nВремя вычислений: %f\n", tk - tn); //Время выполнения

fclose(f);

delete[] u;

delete[] un;

system("pause");

return 0;

}

Код программы(MPI)

#include <stdio.h>

#include <mpi.h>

#include <math.h>

double g(double x)

{

return sin(x+0.48);

}

const double f0 = 0.4618;

double f1(double t)

{

return 3.0\*t+0.882;

}

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

//Определение начальных значений переменных

int size, rank, msgtag = 12;

double a = 1.0, L = 300.0, x = 0.0, time = 0.0, tmax = 100.0, tau = 0.001, h = 0.1;

double r = a\*tau/(h\*h), upr, uu;

int N = (int)(L/h)+1;

int n, l = 0, i, j;

//шапочка

if (MPI\_Init(&arc, &argv) != MPI\_SUCCESS) return 1;

if (MPI\_Comm\_size(MPI\_COMM\_WORLD, &size) != MPI\_SUCCESS) {

MPI\_Finalize();

return 2;

}

if (MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank) != MPI\_SUCCESS) {

MPI\_Finalize();

return 3;

}

MPI\_Status status;

//Распределение отрезков по процессам

int l1 = N/size;

int l2 = N%size;

int \*kol = new int[size];

for(i = 0; i<size; i++) kol[i] = l1;

if(l2) {

if(rank == size-1) l1++;

l2--;

kol[size - 1]++;

}

if(l2) {

if(rank < l2) l1++;

for(i = 0; i<l2; i++) kol[i]++;

}

n = l1;

if((!rank) || (rank == size - 1)) n++; else n+=2;

double \*u = new double[n];

if(rank) {

int sum=0;

for(i=0; i<rank; i++) sum+=kol[i];

l=sum;

sum--;

x=h\*sum; //Вычисляем смещение для каждого промежутка

}

//Присваиваем начальные значения

for(i=0; i<n; i++) {

u[i] = g(x);

x+=h;

}

if(!rank) u[0]=f0;

if(rank==size-1) u[n-1]=f1(time);

double tn=MPI\_Wtime(); //Начальное время

do{

upr=u[0]; //Запоминаем предыдущее значение

for(i=1;i<n-1;i++){

uu=u[i];

u[i]+=r\*(upr-2.0\*u[i]+u[i+1]); //Вычисление новых значений в точках

upr=uu; //Запоминаем предыдущее значение

}

//Начальные значения в краевых точках

if(!rank) u[0]=f0;

if(rank==size-1) u[n-1]=f1(time + tau);

//Обмен данными между процессами

if (rank&1){

MPI\_Ssend(&u[1],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD);

MPI\_Recv(&u[0],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD,&status);

if (rank!=size-1){

MPI\_Ssend(&u[n-2],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD);

MPI\_Recv(&u[n-1],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD,&status);

}}

else{

if (rank!=size-1){

MPI\_Recv(&u[n-1],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD,&status);

MPI\_Ssend(&u[n-2],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD);

}

if(rank){

MPI\_Recv(&u[0],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD,&status);

MPI\_Ssend(&u[1],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD);

}}

time+=tau; //Шаг по времени

}while(time<=tmax);

double tk=MPI\_Wtime(); //Конечное время

//вывод

int in, ik;

if(!rank) in=0; else in=1;

if(rank == size - 1) ik=n; else ik=n-1;

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

{

if(i == rank)

{

FILE \*f;

f = fopen("Temperature.txt", "a");

for(j=in; j<ik; j++, l++) {

fprintf(f, "u[%d] = %f\n", l, u[j]);

}

fclose(f);

}

MPI\_Barrier(MPI\_COMM\_WORLD);

}

if(!rank) {

FILE \*f;

f = fopen("Temperature.txt", "a");

fprintf(f, "\n¬рем¤ вычислений: %f", tk-tn);

fclose(f);

}

delete []u;

delete []kol;

//тапочки

MPI\_Finalize();

return 0;

}

Код программы (MPI однопроцессорный)

#include <stdio.h>

#include <mpi.h>

#include <math.h>

double g(double x)

{

return sin(x+0.48);

}

const double f0 = 0.4618;

double f1(double t)

{

return 3.0\*t+0.882;

}

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

int size, rank, msgtag = 12;

double a = 1.0, L = 300.0, x = 0.0, time = 0.0, tmax = 100.0, tau = 0.001, h = 0.1;

double r = a\*tau/(h\*h), upr, uu;

int N = (int)(L/h)+1;

int n, l = 0, i, j;

//шапочка

if (MPI\_Init(&arc, &argv) != MPI\_SUCCESS) return 1;

if (MPI\_Comm\_size(MPI\_COMM\_WORLD, &size) != MPI\_SUCCESS) {

MPI\_Finalize();

return 2;

}

if (MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank) != MPI\_SUCCESS) {

MPI\_Finalize();

return 3;

}

MPI\_Status status;

int l1 = N/size;

int l2 = N%size;

int \*kol = new int[size];

for(i = 0; i<size; i++) kol[i] = l1;

if(l2) {

if(rank == size-1) l1++;

l2--;

kol[size - 1]++;

}

if(l2) {

if(rank < l2) l1++;

for(i = 0; i<l2; i++) kol[i]++;

}

n = l1;

//if((!rank) || (rank == size - 1)) n++; else n+=2;

double \*u = new double[n];

if(rank) {

int sum=0;

for(i=0; i<rank; i++) sum+=kol[i];

l=sum;

sum--;

x=h\*sum;

}

for(i=0; i<n; i++) {

u[i] = g(x);

x+=h;

}

/\*if(!rank)\*/ u[0]=f0;

/\*if(rank==size-1)\*/ u[n-1]=f1(time);

double tn=MPI\_Wtime();

do{

upr=u[0];

for(i=1;i<n-1;i++){

uu=u[i];

u[i]+=r\*(upr-2.0\*u[i]+u[i+1]);

upr=uu;

}

/\*if(!rank)\*/ u[0]=f0;

/\*if(rank==size-1)\*/ u[n-1]=f1(time + tau);

/\*if (rank&1){

MPI\_Ssend(&u[1],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD);

MPI\_Recv(&u[0],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD,&status);

if (rank!=size-1){

MPI\_Ssend(&u[n-2],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD);

MPI\_Recv(&u[n-1],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD,&status);

}}

else{

if (rank!=size-1){

MPI\_Recv(&u[n-1],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD,&status);

MPI\_Ssend(&u[n-2],1,MPI\_DOUBLE,rank+1,msgtag,MPI\_COMM\_WORLD);

}

if(rank){

MPI\_Recv(&u[0],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD,&status);

MPI\_Ssend(&u[1],1,MPI\_DOUBLE,rank-1,msgtag,MPI\_COMM\_WORLD);

}}\*/

time+=tau;

}while(time<=tmax);

double tk=MPI\_Wtime();

//вывод

//int in, ik;

//if(!rank) in=0; else in=1;

//if(rank == size - 1) ik=n; else ik=n-1;

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

//{

//if(i == rank)

//{

FILE \*f;

f = fopen("Temperature\_One.txt", "a");

for(j=0; j<n; j++) {

fprintf(f, "u[%d] = %f\n", j, u[j]);

}

//for(j=in; j<ik; j++, l++) {

//fprintf(f, "u[%d] = %f\n", l, u[j]);

//}

//fclose(f);

//}

//MPI\_Barrier(MPI\_COMM\_WORLD);

//}

//if(!rank) {

//FILE \*f;

//f = fopen("Temperature\_One.txt", "a");

fprintf(f, "\n¬рем¤ вычислений: %f", tk-tn);

fclose(f);

//}

delete []u;

delete []kol;

//тапочки

MPI\_Finalize();

return 0;

}

Результаты

u[0] = 0.461800

u[1] = 0.459181

u[2] = 0.456563

u[3] = 0.453945

u[4] = 0.451327

u[5] = 0.448709

u[6] = 0.446093

u[7] = 0.443477

u[8] = 0.440862

u[9] = 0.438248

…

u[2991] = 271.565703

u[2992] = 274.706899

u[2993] = 277.876744

u[2994] = 281.075405

u[2995] = 284.303054

u[2996] = 287.559857

u[2997] = 290.845984

u[2998] = 294.161604

u[2999] = 297.506887

u[3000] = 300.882000

Время вычислений MPI: 19.705067

Время вычислений MPI однопроцессорный: 1.045113

Время вычислений OpenMP: 29.471161