Основы параллельного программирования

Лабораторная работа № 3

Решение задачи Пуассона явным методом в трехмерной области

Цель - практическое освоение методов распараллеливание алгоритмов задач, решаемых сеточными методами на примере решения задачи Пуассона в трехмерной области. В лабораторной работе дано объяснение способов декомпозиции данных, связанных с разными способами распределения данных по компьютерам. В этом разделе приведена последовательная программа решения задачи Пуассона явным методом.

Постановка задачи.

Решить уравнение Пуассона

$$\frac{\partial^2\varphi}{\partial x}+\frac{\partial^2\varphi}{\partial y}+\frac{\partial^2\varphi}{\partial z}-a\varphi=\rho(x,y,z) \qquad (1)$$

$$\varphi=\varphi(x,y,z), a>0$$

в области Ω с краевыми условиями 1-го рода

$$\varphi|_G=F(x,y,z)$$

Область решения Ω имеет вид прямоугольного параллелепипеда с размерами X_m x Y_m x Z_m.

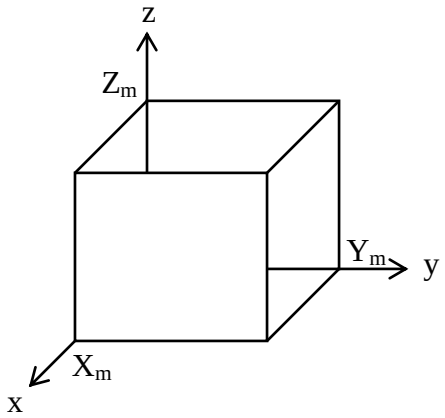


Рис. 1. Область решения Ω

В области введена равномерная прямоугольная сетка с шагами:

$$h_x = X_m/I_m, \quad h_y = Y_m/J_m, \quad h_z = Z_m/K_m,$$

где I_m, J_m, K_m - количество ячеек в направлениях x, y, z.
Значения сеточной функции задаются в узлах сетки

$$\varphi_{i,j,k} = \varphi(x_i, y_j, z_k)$$

где $i = 0,\dots,I_m$, $j = 0,\dots,J_m$, $k = 0,\dots,K_m$.
В выбранной сетке апроксимируется уравнение (1):

$$\frac{\varphi_{i+1} - 2\varphi_{i,j,k} + \varphi_{i-1}}{h_x^2} + \frac{\varphi_{j+1} - 2\varphi_{i,j,k} + \varphi_{j-1}}{h_y^2} + \frac{\varphi_{k+1} - 2\varphi_{i,j,k} + \varphi_{k-1}}{h_z^2} - \alpha\varphi = \rho_{i,j,k}$$

$i = 1,\dots,I_m-1$, $j = 1,\dots,I_m-1$, $k = 1,\dots,I_m-1$.

Полученная система линейных алгебраических уравнений решается итерационным явным методом:

$$(\frac{2}{h_x^2} + \frac{2}{h_y^2} + \frac{2}{h_z^2} + a)\varphi_{i,j,k}^{m+1} = \frac{\varphi_{i+1,j,k}^m + \varphi_{i-1,j,k}^m}{h_x^2} + \frac{\varphi_{i,j+1,k}^m + \varphi_{i,j-1,k}^m}{h_y^2} + \frac{\varphi_{i,j,k+1}^m + \varphi_{i,j,k-1}^m}{h_z^2} - \rho_{i,j,k}$$

или

$$\varphi_{ijk}^{m+1} = \frac{\frac{\varphi_{i+1\,jk}^m + \varphi_{i-1\,jk}^m}{h_x^2} + \frac{\varphi_{ij+1k}^m + \varphi_{ij-1k}^m}{h_y^2} + \frac{\varphi_{ijk+1}^m + \varphi_{ijk-1}^m}{h_z^2} - \rho_{ijk}}{\frac{2}{h_x^2} + \frac{2}{h_y^2} + \frac{2}{h_z^2} + a}$$

Здесь m номер итерации.

Входные параметры: X_m, Y_m, Z_m, I_m, J_m, K_m , $\acute{e}$, α функция $\rho(x,y,z)$, $\varphi|_{\dot{G}}$.
Итерации вести до выполнения условия:

$$\max\ |\varphi^{m+1} - \varphi^m| < \varepsilon$$

Выходные данные:

1) $\max\ |\varphi^{m+1} - \varphi_{tochnoe}|$

2) функция $\varphi^{m+1} - \varphi_{tochnoe}$

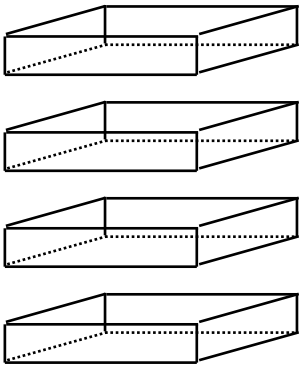
Общая схема распараллеливания алгоритма задачи Пуассона

Здесь приведена общая схема решения задачи Пуассона в трехмерной области явным методом на параллельной вычислительной системе. Схема решения поясняется на рисунках и приведенном фрагменте основного итерационного цикла параллельной программы для решения задачи Пуассона явным методом в трехмерной области с применением коммуникационных функций высокого уровня.

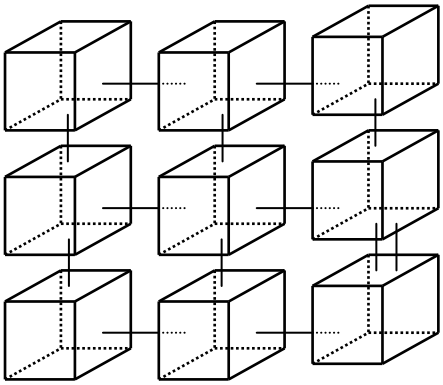
Задача решается методом декомпозиции вычислительного пространства на подобласти. В каждый компьютер помещается соответствующая подобласть вычислительного пространства. Поскольку вычисления осуществляются по схеме «крест», то указанные подобласти имеют дополнительные плоскости для дублирования граничных

плоскостей соседних подобластей. Т.е. осуществляется декомпозиция вычислительного пространства на подобласти с перекрытием граничных плоскостей. Используется модель вычислений - SPMD (Single program - Multiple Data. Все ветви исполняются по одной и той же программе).

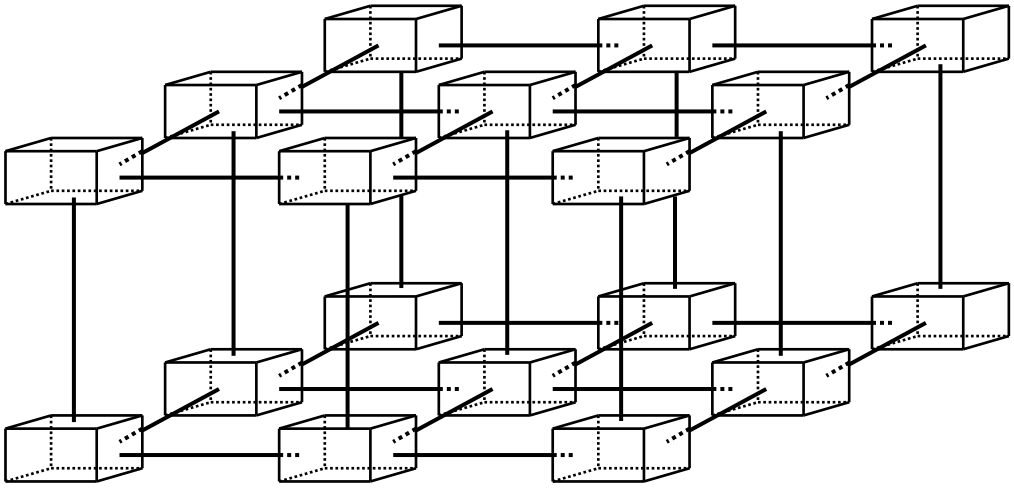
Здесь предполагается, что область вычислений задачи прямоугольная и заданы ее граничные значения. Топология вычислительной системы, решающей данную задачу, может быть: "линейка", или "двумерная решетка", или "трехмерная решетка" (см. рис. 2).



а) декомпозиция на "линейке"



б) декомпозиция на "двумерной решетке"



в) декомпозиция на "трехмерной решетке"

Рис. 2. Декомпозиция трехмерной области вычислений на решетках вычислительных узлов разной мерности.

Кратко, схема вычислений осуществляется следующим образом. Все процессы обсчитывают свою подобласть внутри одной итерации одновременно. Так как коммуникации между процессам является времяемкой операцией имеет смысл выполнять коммуникации на фоне вычислений. Для этого в каждом процессе изначально обсчитываются граничные значения области, после чего осуществляется асинхронная коммуникация по обмену этих значений необходимых для обсчета следующей итерации. И только после этого начинается вычисления внутренних точек подобласти.

Схема обменов границами показана ниже на рис. 3.

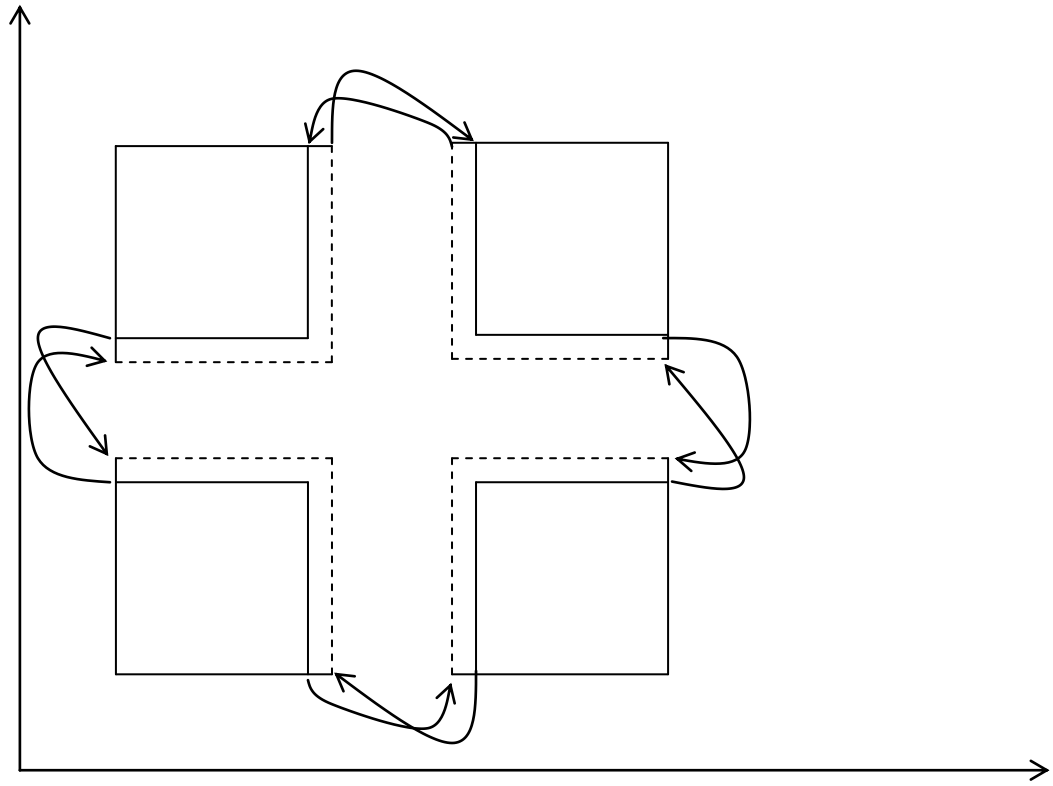


Рис. 3. Схема обмена границами между процессами

Ниже приведена программа решения задачи Пуассона явным методом для последовательного алгоритма.

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<time.h>
#include<sys/time.h>

/* Количество ячеек вдоль координат x, y, z */
#define in 20
#define jn 20
#define kn 20

#define a 1

double Fresh(double,double,double);
double Ro(double,double,double);
void Inic();

/* Выделение памяти для 3D пространства для текущей и предыдущей итерации */
double F[2][in+1][jn+1][kn+1];

double hx, hy, hz;


/* Функция определения точного решения */
```

```
 double Fresh(double x,double y,double z)
          { double res;
          res = x + y + z;
          return res;
        }

/* Функция задания правой части уравнения */
 double Ro(double x, double y, double z)
        { double d;
          d = -a*(x+y+z);
          return d;
        }

/* Подпрограмма инициализации границ 3D пространства */
 void Inic()
      { int i, j, k;
        for(i = 0; i <= in; i++)
          { for(j = 0; j <= jn; j++)
             { for(k = 0; k <= kn; k++)
                { if((i!=0)&&(j!=0)&&(k!=0)&&(i!=in)&&(j!=jn)&&(k!=kn))
                  { F[0][i][j][k] = 0;
                    F[1][i][j][k] = 0;
                  }
                 else
                  { F[0][i][j][k] = Fresh(i*hx,j*hy,k*hz);
                    F[1][i][j][k] = Fresh(i*hx,j*hy,k*hz);
                   }
                }
             }
            }
       }


int main()
  {
    double X, Y, Z;
    double max, N, t1, t2;
    double owx, owy, owz, c, e;
    double Fi, Fj, Fk, F1;

    int i, j, k, mi, mj, mk;
    int R, fl, fl1, fl2;
    int it,f, L0, L1;
    long int osdt;


    struct timeval tv1,tv2;

    it = 0;
    X = 2.0;
    Y = 2.0;
    Z = 2.0;
    e = 0.00001;
    L0 = 1;
    L1 = 0;

    /* Размеры шагов */
    hx = X/in;
    hy = Y/jn;
    hz = Z/kn;

    owx = pow(hx,2);
    owy = pow(hy,2);
    owz = pow(hz,2);
```

```
    c = 2/owx + 2/owy + 2/owz + a;

    gettimeofday(&tv1,(struct timezone*)0);
/* Инициализация границ пространства */
    Inic();

/* Основной итерационный цикл */
    do
     { f = 1;
       L0 = 1-L0;
       L1 = 1-L1;
       for(i = 1; i < in; i++)
         for(j = 1; j < jn; j++)
        { for(k = 1; k < kn; k++)
            {
              Fi = (F[L0][i+1][j][k] + F[L0][i-1][j][k]) / owx;
              Fj = (F[L0][i][j+1][k] + F[L0][i][j-1][k]) / owy;
              Fk = (F[L0][i][j][k+1] + F[L0][i][j][k-1]) / owz;
              F[L1][i][j][k] = (Fi + Fj + Fk - Ro(i*hx,j*hy,k*hz)) / c;
              if(fabs(F[L1][i][j][k] - F[L0][i][j][k]) > e)
              f = 0;
            }
         }
      it++;
      }
     while(f == 0);

    gettimeofday(&tv2,(struct timezone*)0);
    osdt = (tv2.tv_sec - tv1.tv_sec)*1000000 + tv2.tv_usec-tv1.tv_usec;

    printf("\n in = %d  iter = %d E = %f  T = %ld\n",in,it,e,osdt);

/* Нахождение максимального расхождения полученного приближенного решения
 * и точного решения */
    max = 0.0;
      { for(i = 1; i < in; i++)
         { for(j = 1; j < jn; j++)
           { for(k = 1; k < kn; k++)
               { if((F1 = fabs(F[L1][i][j][k] - Fresh(i*hx,j*hy,k*hz))) > max)
                         { max = F1;
                      mi = i; mj = j; mk = k;
                   }
               }
               }
       }

  printf("Max differ = %f\n in point(%d,%d,%d)\n",max,mi,mj,mk);
      }
      return(0);

  }
```

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Написать параллельную программу на языке C/C++ с использованием MPI, реализующую решение уравнение Пуассона явным методом в трехмерной области в случае одномерной декомпозиции области. Уделить внимание тому, чтобы коммуникации по обмену граничными значениями подобластей между процессами были асинхронными и выполнялись на фоне счета.
2. Замерить время работы программы при использовании различного числа процессорных ядер: 1..16. Исходные данные и параметры задачи подобрать таким образом, чтобы решение задачи на одном ядре занимало не менее 30 секунд. Построить графики зависимости времени работы программы, ускорения и эффективности распараллеливания от числа используемых ядер.