Семинар 10. Решение двумерного уравнения теплопроводности.

1. Постановка задачи.

$$\frac{\partial u}{\partial t}=\frac{\partial}{\partial x_1}\left(k(x_1,x_2)\frac{\partial u}{\partial x_1}\right)+\frac{\partial}{\partial x_2}\left(k(x_1,x_2)\frac{\partial u}{\partial x_2}\right)+f(x_1,x_2,t),\quad (x_1,x_2)\in D=(a_1,b_1)\times(a_2,b_2),$$

$$u(x_1,x_2,0)=g_0(x_1,x_2),\quad u(a_1,x_2,t)=g_{11}(t),\quad u(b_1,x_2,t)=g_{12}(t),\quad u(x_1,a_2,t)=g_{21}(t),\quad u(x_1,b_2,t)=g_{22}(t),$$

$$k(x_1,x_2)=\begin{cases}k_1, & (x_1,x_2)\in D_0,\\ k_2, & (x_1,x_2)\notin D_0,\end{cases}\quad D_0=[x_{11},x_{12}]\times[x_{21},x_{22}],$$

$$f(x_1,x_2,t)=Q_0\exp\left[-(x_1-x_{10})^2/r_0^2-(x_2-x_{20})^2/r_0^2\right]\left(1-\exp\left[-t/\tau_0\right]\right),$$

$$g_0(x_1,x_2)=u_0,\quad g_{11}(t)=u_0,\quad g_{12}(t)=u_0,\quad g_{21}(t)=u_0+\left(u_1-u_0\right)\left(1-\exp\left[-t/\tau_1\right]\right),\quad g_{22}(t)=u_0.$$

2. Численные алгоритмы.

МКР на равномерной сетке $\Omega=\omega_{x_1}\times\omega_{x_2}\times\omega_t$, $\omega_{x_\alpha}=\left\{x_{\alpha,i_\alpha}=i_\alpha h_\alpha, i_\alpha=0,...,N_\alpha, h_\alpha=\dfrac{b_\alpha-a_\alpha}{N_\alpha}\right\}$, $\alpha=1,2,$

$\omega_t=\left\{t_j=j\tau,\ j=0,...,N_t,\ \tau=\dfrac{t_{\max}}{N_t}\right\}$.

Схема с весами:

$$\frac{y_{i_1i_2}^{j+1}-y_{i_1i_2}^{j}}{\tau}=\sigma\frac{1}{\hbar_1}\left\{k_{i_1+1/2,i_2}\frac{y_{i_1+1,i_2}^{j+1}-y_{i_1i_2}^{j+1}}{h_1}-k_{i_1-1/2,i_2}\frac{y_{i_1i_2}^{j+1}-y_{i_1-1,i_2}^{j+1}}{h_1}\right\}+\sigma\frac{1}{\hbar_2}\left\{k_{i_1,i_2+1/2}\frac{y_{i_1,i_2+1}^{j+1}-y_{i_1i_2}^{j+1}}{h_2}-k_{i_1,i_2-1/2}\frac{y_{i_1i_2}^{j+1}-y_{i_1,i_2-1}^{j+1}}{h_2}\right\}+$$

$$+(1-\sigma)\frac{1}{\hbar_1}\left\{k_{i_1+1/2,i_2}\frac{y_{i_1+1,i_2}^{j}-y_{i_1i_2}^{j}}{h_1}-k_{i_1-1/2,i_2}\frac{y_{i_1i_2}^{j}-y_{i_1-1,i_2}^{j}}{h_1}\right\}+(1-\sigma)\frac{1}{\hbar_2}\left\{k_{i_1,i_2+1/2}\frac{y_{i_1,i_2+1}^{j}-y_{i_1i_2}^{j}}{h_2}-k_{i_1,i_2-1/2}\frac{y_{i_1i_2}^{j}-y_{i_1,i_2-1}^{j}}{h_2}\right\}+$$

$$+\sigma f_{i_1i_2}^{j+1}+(1-\sigma)f_{i_1i_2}^{j},\quad 0<i_\alpha<N_\alpha,\quad \alpha=1,2,\quad 0\le j<N_t,$$

$$y_{i_1i_2}^0=g_0(x_{1,i_1},x_{2,i_2}),\quad y_{0,i_2}^j=g_{11}(t_j),\quad y_{N_1,i_2}^j=g_{12}(t_j),\quad y_{i_1,0}^j=g_{21}(t_j),\quad y_{i_1,N_2}^j=g_{22}(t_j),$$

$$k_{i_1\pm1/2,i_2}=\frac{2k_{i_1i_2}k_{i_1\pm1,i_2}}{k_{i_1i_2}+k_{i_1\pm1,i_2}},\quad k_{i_1,i_2\pm1/2}=\frac{2k_{i_1i_2}k_{i_1,i_2\pm1}}{k_{i_1i_2}+k_{i_1,i_2\pm1}},\quad k_{i_1i_2}=k(x_{1,i_1},x_{2,i_2}),\quad f_{i_1i_2}^j=f(x_{1,i_1},x_{2,i_2},t_j),\quad \hbar_\alpha=\begin{cases}0.5h_\alpha, & i=0,N_\alpha,\\ h_\alpha, & 1<i<N_\alpha.\end{cases}$$

Выбор сеток и шагов:

1) $h_\alpha\sim\dfrac{r_0}{m}$, $m=1,2,3,...$; 2) $t_{\max}\sim\max\left(\tau_0,\tau_1\right)\cdot M$, $M=5,10,15,...$;

3) если имеется стационар, то норма $g_t=\max\limits_{i_1,i_2}\left|\dfrac{y_{i_1,i_2}^{j+1}-y_{i_1,i_2}^{j}}{\tau y_{i_1,i_2}^{j}}\right|$ уменьшается и можно поставить условие $g_t\le\varepsilon_t$;

4) $\tau\le\min\left(\tau^{(\sigma)},\tau^{(f)}\right)$ – условия устойчивости для схемы с весами и ОДУ:

$$\tau^{(\sigma=0)}=\frac{0.5h_{\min}^2}{\max\left(k_1,k_2\right)},\ \tau^{(\sigma=0.5)}=\frac{h_{\min}^2}{\max\left(k_1,k_2\right)},\ \tau^{(\sigma=1)}=\frac{0.5h_{\min}}{\sqrt{\max\left(k_1,k_2\right)}},\ \tau^{(f)}=\frac{1}{\max\left|f\left(x_1,x_2,t\right)\right|L}=\frac{1}{Q_0L},\quad L=1,5,10,....$$

Тестирование – расчеты на последовательности вложенных сеток.

Расчетные формулы для явной схемы:

$$y_{i_1i_2}^{j+1}=y_{i_1i_2}^{j}+B_{1i_1i_2}\left(y_{i_1+1,i_2}^{j}-y_{i_1i_2}^{j}\right)-A_{1i_1i_2}\left(y_{i_1i_2}^{j}-y_{i_1-1,i_2}^{j}\right)+B_{2i_1i_2}\left(y_{i_1,i_2+1}^{j}-y_{i_1i_2}^{j}\right)-A_{2i_1i_2}\left(y_{i_1i_2}^{j}-y_{i_1,i_2-1}^{j}\right)+\tau f_{i_1i_2}^{j},$$

$$A_{1i_1i_2}=\gamma_1k_{i_1-1/2,i_2},\quad B_{1i_1i_2}=\gamma_1k_{i_1+1/2,i_2},\quad A_{2i_1i_2}=\gamma_2k_{i_1,i_2-1/2},\quad B_{2i_1i_2}=\gamma_2k_{i_1,i_2+1/2},\quad \gamma_1=\frac{\tau}{\hbar_1h_1},\quad \gamma_2=\frac{\tau}{\hbar_2h_2}.$$

Неявная локально одномерная схема:

$$\frac{y_{i_1i_2}^{j+1/2}-y_{i_1i_2}^{j}}{\tau}=\frac{1}{\hbar_1}\left\{k_{i_1+1/2,i_2}\frac{y_{i_1+1,i_2}^{j+1/2}-y_{i_1i_2}^{j+1/2}}{h_1}-k_{i_1-1/2,i_2}\frac{y_{i_1i_2}^{j+1/2}-y_{i_1-1,i_2}^{j+1/2}}{h_1}\right\}+\frac{1}{\hbar_2}\left\{k_{i_1,i_2+1/2}\frac{y_{i_1,i_2+1}^{j}-y_{i_1i_2}^{j}}{h_2}-k_{i_1,i_2-1/2}\frac{y_{i_1i_2}^{j}-y_{i_1,i_2-1}^{j}}{h_2}\right\}+\frac{1}{2}f_{i_1i_2}^{j},$$

$$\frac{y_{i_1i_2}^{j+1}-y_{i_1i_2}^{j+1/2}}{\tau}=\frac{1}{\hbar_1}\left\{k_{i_1+1/2,i_2}\frac{y_{i_1+1,i_2}^{j+1/2}-y_{i_1i_2}^{j+1/2}}{h_1}-k_{i_1-1/2,i_2}\frac{y_{i_1i_2}^{j+1/2}-y_{i_1-1,i_2}^{j+1/2}}{h_1}\right\}+\frac{1}{\hbar_2}\left\{k_{i_1,i_2+1/2}\frac{y_{i_1,i_2+1}^{j+1}-y_{i_1i_2}^{j+1}}{h_2}-k_{i_1,i_2-1/2}\frac{y_{i_1i_2}^{j+1}-y_{i_1,i_2-1}^{j+1}}{h_2}\right\}+\frac{1}{2}f_{i_1i_2}^{j+1},$$

$$0<i_\alpha<N_\alpha,\quad \alpha=1,2,\quad 0\le j<N_t.$$

Расчетные формулы:

$$y_{i_1 i_2}^{j+1/2} - B_{1 i_1 i_2}\left(y_{i_1+1,i_2}^{j+1/2} - y_{i_1 i_2}^{j+1/2}\right) + A_{1 i_1 i_2}\left(y_{i_1 i_2}^{j+1/2} - y_{i_1-1,i_2}^{j+1/2}\right) = y_{i_1 i_2}^{j} + B_{2 i_1 i_2}\left(y_{i_1,i_2+1}^{j} - y_{i_1 i_2}^{j}\right) - A_{2 i_1 i_2}\left(y_{i_1 i_2}^{j} - y_{i_1,i_2-1}^{j}\right) + \frac{\tau}{2} f_{i_1 i_2}^{j},$$

$$y_{i_1 i_2}^{j+1} - B_{2 i_1 i_2}\left(y_{i_1,i_2+1}^{j+1} - y_{i_1 i_2}^{j+1}\right) + A_{2 i_1 i_2}\left(y_{i_1 i_2}^{j+1} - y_{i_1,i_2-1}^{j+1}\right) = y_{i_1 i_2}^{j+1/2} + B_{1 i_1 i_2}\left(y_{i_1+1,i_2}^{j+1/2} - y_{i_1 i_2}^{j+1/2}\right) - A_{1 i_1 i_2}\left(y_{i_1 i_2}^{j+1/2} - y_{i_1-1,i_2}^{j+1/2}\right) + \frac{\tau}{2} f_{i_1 i_2}^{j+1},$$

$$A_{1 i_1 i_2} = \gamma_1 k_{i_1-1/2,i_2}, \quad B_{1 i_1 i_2} = \gamma_1 k_{i_1+1/2,i_2}, \quad A_{2 i_1 i_2} = \gamma_2 k_{i_1,i_2-1/2}, \quad B_{2 i_1 i_2} = \gamma_2 k_{i_1,i_2+1/2}, \quad \gamma_1 = \frac{\tau}{\hbar_1 h_1}, \quad \gamma_2 = \frac{\tau}{\hbar_2 h_2}.$$

Проверка на точность с помощью вложенных сеток.

3. Параллельная реализация.

В случае явной схемы параллелим расчетные формулы.

В случае неявной ЛОС применяем комбинацию параллельных прогонок.

4. Реализация примеров.

Пример 1. Реализация явной схемы на решетке процессоров (ex14a.c).

```
#include <stdio.h> #include <stdlib.h> #include <string.h> #include <math.h>
#include "mycom.h" #include "mynet.h" #include "myio.h"
static int np, mp, nl, ier, lp;
static int np1, np2, mp1, mp2, mp_l, mp_r, mp_b, mp_t;
static char pname[MPI_MAX_PROCESSOR_NAME];
static char vname[10] = "ex14a"; static char sname[20];
static MPI_Status status; static union_t buf;
static double tick, t1, t2, t3;
static FILE *Fi = NULL; static FILE *Fo = NULL;
static int n1, n2, ntp, ntm, ntv;
static double a1, b1, a2, b2;
static double x10, x20, r0, q0, tau0;
static double x11, x12, x21, x22, k1, k2;
static double u0, u1, tau1, tmax, epst;
static double tv, u10, omg0, omg1, gt;
double k(double x1, double x2); double k(double x1, double x2) {
  if ((x11<=x1) && (x1<=x12) && (x21<=x2) && (x2<=x22)) return k1; else return k2; }
double f(double x1, double x2, double t); double f(double x1, double x2, double t) {
  double s1 = (x1-x10)/r0; double s2 = (x2-x20)/r0; double s3 = omg0 * t;
  return q0*exp(-s1*s1-s2*s2)*(1.0-exp(-s3)); }
double g0(double x1, double x2); double g0(double x1, double x2) { return u0; }
double g11(double t); double g11(double t) { return u0; }
double g12(double t); double g12(double t) { return u0; }
double g21(double t); double g21(double t) {
  double s1 = omg1 * t; return u0 + u10 * (1.0-exp(-s1)); }
double g22(double t); double g22(double t) { return u0; }
int main(int argc, char *argv[])
{
  int m, i1, i2, j1, j2, i11, i12, i21, i22, nc1, nc2, nc1m, nc2m, nc12;
  double h1, h12, h2, h22, tau, gam1, gam2, s0, s1, s2, s3, s4, s5;
  double *xx1, *xx2, *aa1, *bb1, *aa2, *bb2, *yy0, *yy1;
  double *ss_l, *rr_l, *ss_r, *rr_r, *ss_b, *rr_b, *ss_t, *rr_t;
  MyNetInit(&argc,&argv,&np,&mp,&nl,pname,&tick);
  fprintf(stderr,"Netsize: %d, process: %d, system: %s, tick=%12le\n", np,mp,pname,tick);
  sleep(1);
  sprintf(sname,"%s.p%02d",vname,mp);
  ier = fopen_m(&Fo,sname,"wt");
  if (ier!=0) mpierr("Protocol file not opened",1);
  if (mp==0) {
    sprintf(sname,"%s.d",vname);
    ier = fopen_m(&Fi,sname,"rt");
    if (ier!=0) mpierr("Data file not opened",2);
    fscanf(Fi,"a1=%le\n",&a1); fscanf(Fi,"b1=%le\n",&b1);
    fscanf(Fi,"a2=%le\n",&a2); fscanf(Fi,"b2=%le\n",&b2);
    fscanf(Fi,"x10=%le\n",&x10); fscanf(Fi,"x20=%le\n",&x20);
    fscanf(Fi,"x11=%le\n",&x11); fscanf(Fi,"x12=%le\n",&x12);
    fscanf(Fi,"x21=%le\n",&x21); fscanf(Fi,"x22=%le\n",&x22);
    fscanf(Fi,"r0=%le\n",&r0); fscanf(Fi,"q0=%le\n",&q0);
    fscanf(Fi,"u0=%le\n",&u0); fscanf(Fi,"u1=%le\n",&u1);
    fscanf(Fi,"k1=%le\n",&k1); fscanf(Fi,"k2=%le\n",&k2);
    fscanf(Fi,"tau0=%le\n",&tau0); fscanf(Fi,"tau1=%le\n",&tau1);
    fscanf(Fi,"tmax=%le\n",&tmax); fscanf(Fi,"epst=%le\n",&epst);
    fscanf(Fi,"n1=%d\n",&n1); fscanf(Fi,"n2=%d\n",&n2);
    fscanf(Fi,"ntp=%d\n",&ntp); fscanf(Fi,"ntm=%d\n",&ntm);
    fscanf(Fi,"lp=%d\n",&lp);
    fclose_m(&Fi);
    if (argc>1) sscanf(argv[1],"%d",&n1); if (argc>2) sscanf(argv[2],"%d",&n2);
    if (argc>3) sscanf(argv[3],"%d",&ntp); if (argc>4) sscanf(argv[4],"%d",&ntm);
  }
  if (np>1) {
    if (mp==0) {
      buf.ddata[0] = a1; buf.ddata[1] = b1; buf.ddata[2] = a2; buf.ddata[3] = b2;
      buf.ddata[4] = x10; buf.ddata[5] = x20; buf.ddata[6] = x11; buf.ddata[7] = x12;
      buf.ddata[8] = x21; buf.ddata[9] = x22; buf.ddata[10] = r0; buf.ddata[11] = q0;
      buf.ddata[12] = u0; buf.ddata[13] = u1; buf.ddata[14] = k1; buf.ddata[15] = k2;
      buf.ddata[16] = tau0; buf.ddata[17] = tau1; buf.ddata[18] = tmax;
      buf.ddata[19] = epst; buf.idata[40] = n1; buf.idata[41] = n2;
      buf.idata[42] = ntp; buf.idata[43] = ntm; buf.idata[44] = lp;
    }
    MPI_Bcast(buf.ddata,23,MPI_DOUBLE,0,MPI_COMM_WORLD);
    if (mp>0) {
      a1 = buf.ddata[0]; b1 = buf.ddata[1]; a2 = buf.ddata[2]; b2 = buf.ddata[3];
      x10 = buf.ddata[4]; x20 = buf.ddata[5]; x11 = buf.ddata[6]; x12 = buf.ddata[7];
      x21 = buf.ddata[8]; x22 = buf.ddata[9]; r0 = buf.ddata[10]; q0 = buf.ddata[11];
      u0 = buf.ddata[12]; u1 = buf.ddata[13]; k1 = buf.ddata[14]; k2 = buf.ddata[15];
      tau0 = buf.ddata[16]; tau1 = buf.ddata[17]; tmax = buf.ddata[18];
      epst = buf.ddata[19]; n1 = buf.idata[40]; n2 = buf.idata[41];
      ntp = buf.idata[42]; ntm = buf.idata[43]; lp = buf.idata[44];
    }
  }
  fprintf(Fo,"Netsize: %d, process: %d, system: %s, tick=%12le\n",np,mp,pname,tick);
  fprintf(Fo,"a1=%le b1=%le a2=%le b2=%le\n",a1,b1,a2,b2);
  fprintf(Fo,"x10=%le x20=%le r0=%le q0=%le tau0=%le\n",x10,x20,r0,q0,tau0);
  fprintf(Fo,"x11=%le x12=%le x21=%le x22=%le k1=%le k2=%le\n",x11,x12,x21,x22,k1,k2);
  fprintf(Fo,"u0=%le u1=%le tau1=%le tmax=%le epst=%le\n",u0,u1,tau1,tmax,epst);
  fprintf(Fo,"n1=%d n2=%d ntp=%d ntm=%d lp=%d\n",n1,n2,ntp,ntm,lp);
  t1 = MPI_Wtime();
  u10 = u1 - u0; omg0 = 1.0 / tau0; omg1 = 1.0 / tau1;
  h1 = (b1-a1)/n1; h12 = h1 * h1; h2 = (b2-a2)/n2; h22 = h2 * h2;
  tau = 0.25 * dmin(h12,h22) / dmax(k1,k2); tau = dmin(tau,1.0/q0);
  gam1 = tau / h12; gam2 = tau / h22;
  s0 = dmin(tmax/tau,1000000000.0); ntm = imin(ntm,(int)s0);
  fprintf(Fo,"u10=%le omg0=%le omg1=%le\n",u10,omg0,omg1);
  fprintf(Fo,"h1=%le h2=%le tau=%le ntm=%d\n",h1,h2,tau,ntm);
  My2DGrid(np,mp,n1,n2,&np1,&np2,&mp1,&mp2);
// mp = np1 * mp2 + mp1
  if (mp1 ==     0) mp_l = -1; else mp_l = mp - 1;
  if (mp1 == np1-1) mp_r = -1; else mp_r = mp + 1;
  if (mp2 ==     0) mp_b = -1; else mp_b = mp - np1;
  if (mp2 == np2-1) mp_t = -1; else mp_t = mp + np1;
  MyRange(np1,mp1,0,n1,&i11,&i12,&nc1); nc1m = nc1-1;
  MyRange(np2,mp2,0,n2,&i21,&i22,&nc2); nc2m = nc2-1;
  nc12 = nc1 * nc2;
  fprintf(Fo,"Grid=%dx%d coord=(%d,%d)\n",np1,np2,mp1,mp2);
  fprintf(Fo,"i11=%d i12=%d nc1=%d\n",i11,i12,nc1);
  fprintf(Fo,"i21=%d i22=%d nc2=%d\n",i11,i12,nc2);
  if (mp == 0) {
    fprintf(stderr,"n1=%d n2=%d h1=%le h2=%le tau=%le ntm=%d\n",n1,n2,h1,h2,tau,ntm);
    fprintf(stderr,"Grid=%dx%d\n",np1,np2);
  }
  xx1 = (double*)(malloc(sizeof(double)*nc1));
  xx2 = (double*)(malloc(sizeof(double)*nc2));
  yy0 = (double*)(malloc(sizeof(double)*nc12));
  yy1 = (double*)(malloc(sizeof(double)*nc12));
  aa1 = (double*)(malloc(sizeof(double)*nc12));
  bb1 = (double*)(malloc(sizeof(double)*nc12));
  aa2 = (double*)(malloc(sizeof(double)*nc12));
  bb2 = (double*)(malloc(sizeof(double)*nc12));
  if (mp_l>=0) {
    rr_l = (double*)(malloc(sizeof(double)*nc2));
    ss_l = (double*)(malloc(sizeof(double)*nc2));
  }
  if (mp_r>=0) {
    rr_r = (double*)(malloc(sizeof(double)*nc2));
    ss_r = (double*)(malloc(sizeof(double)*nc2));
  }
  if (mp_b>=0) {
    rr_b = (double*)(malloc(sizeof(double)*nc1));
    ss_b = (double*)(malloc(sizeof(double)*nc1));
  }
  if (mp_t>=0) {
    rr_t = (double*)(malloc(sizeof(double)*nc1));
    ss_t = (double*)(malloc(sizeof(double)*nc1));
  }
  for (i1=0; i1<nc1; i1++) xx1[i1] = a1 + h1 * (i11 + i1); // grid for x1
  for (i2=0; i2<nc2; i2++) xx2[i2] = a2 + h2 * (i21 + i2); // grid for x2
  for (i2=0; i2<nc2; i2++) { j2 = i21 + i2;
    for (i1=0; i1<nc1; i1++) { j1 = i11 + i1;
      m = nc1 * i2 + i1;
      if ((j1==0) || (j1==n1)) { aa1[m] = 0.0; bb1[m] = 0.0; }
      else {
        s0 = k(xx1[i1],xx2[i2]); s1 = k(xx1[i1]-h1,xx2[i2]); s2 = k(xx1[i1]+h1,xx2[i2]);
        aa1[m] = gam1*2.0*s0*s1/(s0+s1); bb1[m] = gam1*2.0*s0*s2/(s0+s2); }
      if ((j2==0) || (j2==n2)) { aa2[m] = 0.0; bb2[m] = 0.0; }
      else {
        s0 = k(xx1[i1],xx2[i2]); s1 = k(xx1[i1],xx2[i2]-h2); s2 = k(xx1[i1],xx2[i2]+h2);
        aa2[m] = gam2*2.0*s0*s1/(s0+s1); bb2[m] = gam2*2.0*s0*s2/(s0+s2); }
    }
  }
  ntv = 0; tv = 0.0; gt = 1.0;
  for (i2=0; i2<nc2; i2++) for (i1=0; i1<nc1; i1++) {
    m = nc1 * i2 + i1; yy1[m] = g0(xx1[i1],xx2[i2]); }
// Time loop:
  do {
    ntv++; tv += tau;
    for (m=0; m<nc12; m++) yy0[m] = yy1[m];
    if (np>1) {
      if (mp_l>=0) {
        i1 = 0; for (i2=0; i2<nc2; i2++) { m = nc1 * i2 + i1; ss_l[i2] = yy0[m]; }
      }
      if (mp_r>=0) {
        i1 = nc1-1; for (i2=0; i2<nc2; i2++) { m = nc1 * i2 + i1; ss_r[i2] = yy0[m]; }
      }
      if (mp_b>=0) {
        i2 = 0; for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; ss_b[i1] = yy0[m]; }
      }
      if (mp_t>=0) {
        i2 = nc2-1; for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; ss_t[i1] = yy0[m]; }
      }
      BndAExch2D(mp_l,nc2,ss_l,rr_l,
                 mp_r,nc2,ss_r,rr_r,
                 mp_b,nc1,ss_b,rr_b,
                 mp_t,nc1,ss_t,rr_t);
    }
    for (i2=0; i2<nc2; i2++) { j2 = i21 + i2;
      for (i1=0; i1<nc1; i1++) { j1 = i11 + i1;
        m = nc1 * i2 + i1;
        if      (j1== 0) yy1[m] = g11(tv);
        else if (j1==n1) yy1[m] = g12(tv);
        else if (j2== 0) yy1[m] = g21(tv);
        else if (j2==n2) yy1[m] = g22(tv);
        else {
          if (i1==0)     s1 = aa1[m] * (yy0[m] - rr_l[i2]);
          else           s1 = aa1[m] * (yy0[m] - yy0[m-1]);
          if (i1==nc1m) s2 = bb1[m] * (rr_r[i2] - yy0[m]);
          else           s2 = bb1[m] * (yy0[m+1] - yy0[m]);
          if (i2==0)     s3 = aa2[m] * (yy0[m] - rr_b[i1]);
          else           s3 = aa2[m] * (yy0[m] - yy0[m-nc1]);
          if (i2==nc2m) s4 = bb2[m] * (rr_t[i1] - yy0[m]);
          else           s4 = bb2[m] * (yy0[m+nc1] - yy0[m]);
          s5 = tau * f(xx1[i1],xx2[i2],tv-tau);
          yy1[m] += (s2 - s1 + s4 - s3 + s5);
        }
      }
    }
    if (ntv % ntp == 0) {
      gt = 0.0;
      for (m=0; m<nc12; m++) { s0 = (yy1[m]/yy0[m]-1.0); gt = dmax(gt,dabs(s0)); }
      gt = gt / tau;
      if (np>1) { s0 = gt; MPI_Allreduce(&s0,&gt,1,MPI_DOUBLE,MPI_MAX,MPI_COMM_WORLD); }
      if (mp == 0) {
        t2 = MPI_Wtime() - t1;
        fprintf(stderr,"ntv=%d tv=%le gt=%le tcpu=%le\n",ntv,tv,gt,t2);
      }
    }
    if (lp>0) {
      fprintf(Fo,"ntv=%d tv=%le gt=%le\n",ntv,tv,gt);
      for (i2=0; i2<nc2; i2++) { j2 = i21 + i2;
        for (i1=0; i1<nc1; i1++) { j1 = i11 + i1;
          m = nc1 * i2 + i1;
          fprintf(Fo,"i1=%8d i2=%8d x1=%12le x2=%12le y1=%12le\n",
            j1,j2,xx1[i1],xx2[i2],yy1[m]);
        }
      }
    }
  } while ((ntv<ntm) && (gt>epst));
  t1 = MPI_Wtime() - t1;
  sprintf(sname,"%s_%02d.dat",vname,np);
  OutFun2DP(sname,np,mp,nc1,nc2,xx1,xx2,yy1);
  fprintf(Fo,"ntv=%d tv=%le gt=%le time=%le\n",ntv,tv,gt,t1);
  if (mp == 0) fprintf(stderr,"Grid=%dx%d n1=%d n2=%d ntv=%d tv=%le gt=%le tcpu=%le\n",
    np1,np2,n1,n2,ntv,tv,gt,t1);
  ier = fclose_m(&Fo);
  MPI_Finalize();
  return 0;
}
```

Трансляция:

```
>mpicc -o ex14a.px -O2 ex14a.c mycom.c mynet.c myio.c -lm
```

Результаты расчетов:

```
>mpirun -np <1-16> -nolocal -machinefile hosts ex14a.px 100 100 5000 25000  (в классе)
>mpirun -np <1-16> ex14a.px 100 100 5000 25000                               (на сервере)
Grid=1x1 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=2.241195e+01
Grid=1x2 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=1.596867e+01
Grid=1x3 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=1.253931e+01
Grid=2x2 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=1.030746e+01
Grid=1x5 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=9.375307e+00
Grid=2x3 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=8.089331e+00
Grid=1x7 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=7.802669e+00
Grid=2x4 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=6.910250e+00
Grid=3x3 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=6.936690e+00
Grid=2x5 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=7.533998e+00
Grid=1x11 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=7.455552e+00
Grid=3x4 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=7.639191e+00
Grid=1x13 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=7.125131e+00
Grid=2x7 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=5.894694e+00
Grid=3x5 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=6.417959e+00
Grid=4x4 n1=100 n2=100 ntv=25000 tv=6.250000e-02 gt=2.196164e+00 tcpu=5.550545e+00
```

Пример 2. Реализация неявной локально одномерной схемы на решетке процессоров (ex14b.c).

```
#include <stdio.h> #include <stdlib.h> #include <string.h> #include <unistd.h>
#include <math.h> #include "mycom.h" #include "mynet.h" #include "myio.h"
#include "myprog.h"
static int np, mp, nl, ier, lp; static int np1, np2, mp1, mp2;
static int mp_l, mp_r, mp_b, mp_t; static char pname[MPI_MAX_PROCESSOR_NAME];
static char vname[10] = "ex14b"; static char sname[20]; static MPI_Status status;
static union_t buf; static double tick, t1, t2, t3;
static FILE *Fi = NULL; static FILE *Fo = NULL;
static int n1, n2, ntp, ntm, ntv; static double a1, b1, a2, b2;
static double x10, x20, r0, q0, tau0; static double x11, x12, x21, x22, k1, k2;
static double u0, u1, tau1, tmax, epst; static double tv, u10, omg0, omg1, gt;
double k(double x1, double x2); double k(double x1, double x2) {
  if ((x11<=x1) && (x1<=x12) && (x21<=x2) && (x2<=x22)) return k1; else return k2; }
double f(double x1, double x2, double t); double f(double x1, double x2, double t) {
  double s1 = (x1-x10) / r0; double s2 = (x2-x20) / r0; double s3 = omg0 * t;
  return q0*exp(-s1*s1-s2*s2)*(1.0-exp(-s3)); }
double g0(double x1, double x2); double g0(double x1, double x2) { return u0; }
double g11(double t); double g11(double t) { return u0; }
double g12(double t); double g12(double t) { return u0; }
double g21(double t); double g21(double t) {
  double s1 = omg1 * t; return u0 + u10 * (1.0-exp(-s1)); }
double g22(double t); double g22(double t) { return u0; }
int main(int argc, char *argv[])
{
  int m, i1, i2, j1, j2, i11, i12, i21, i22, nc1, nc2, nc1m, nc2m, nc12;
  int ncp, ncp1, ncp2, ncx, ncx1, ncx2, ncpx;
  double h1, h12, h2, h22, tau, tau05, gam1, gam2, s0, s1, s2, s3, s4, s5;
  double *xx1, *xx2, *aa1, *bb1, *aa2, *bb2, *yy0, *yy1, *yy2;
  double *aa, *bb, *cc, *ff, *al, *y1, *y2, *y3, *y4;
  double *ss_l, *rr_l, *ss_r, *rr_r, *ss_b, *rr_b, *ss_t, *rr_t;
  int ranks[128];
  MPI_Group gr0, gr1, gr2; MPI_Comm  cm0, cm1, cm2;
  MyNetInit(&argc,&argv,&np,&mp,&nl,pname,&tick);
  fprintf(stderr,"Netsize: %d, process: %d, system: %s, tick=%12le\n",np,mp,pname,tick);
  sleep(1);
  sprintf(sname,"%s.p%02d",vname,mp); ier = fopen_m(&Fo,sname,"wt");
  if (ier!=0) mpierr("Protocol file not opened",1);
  if (mp==0) {
    sprintf(sname,"%s.d",vname); ier = fopen_m(&Fi,sname,"rt");
    if (ier!=0) mpierr("Data file not opened",2);
    fscanf(Fi,"a1=%le\n",&a1); fscanf(Fi,"b1=%le\n",&b1);
    fscanf(Fi,"a2=%le\n",&a2); fscanf(Fi,"b2=%le\n",&b2);
    fscanf(Fi,"x10=%le\n",&x10); fscanf(Fi,"x20=%le\n",&x20);
    fscanf(Fi,"x11=%le\n",&x11); fscanf(Fi,"x12=%le\n",&x12);
    fscanf(Fi,"x21=%le\n",&x21); fscanf(Fi,"x22=%le\n",&x22);
    fscanf(Fi,"r0=%le\n",&r0); fscanf(Fi,"q0=%le\n",&q0);
    fscanf(Fi,"u0=%le\n",&u0); fscanf(Fi,"u1=%le\n",&u1);
    fscanf(Fi,"k1=%le\n",&k1); fscanf(Fi,"k2=%le\n",&k2);
    fscanf(Fi,"tau0=%le\n",&tau0); fscanf(Fi,"tau1=%le\n",&tau1);
    fscanf(Fi,"tmax=%le\n",&tmax); fscanf(Fi,"epst=%le\n",&epst);
    fscanf(Fi,"n1=%d\n",&n1); fscanf(Fi,"n2=%d\n",&n2);
    fscanf(Fi,"ntp=%d\n",&ntp); fscanf(Fi,"ntm=%d\n",&ntm);
    fscanf(Fi,"lp=%d\n",&lp);
    fclose_m(&Fi);
    if (argc>1) sscanf(argv[1],"%d",&n1); if (argc>2) sscanf(argv[2],"%d",&n2);
    if (argc>3) sscanf(argv[3],"%d",&ntp); if (argc>4) sscanf(argv[4],"%d",&ntm);
  }
  if (np>1) {
    if (mp==0) {
      buf.ddata[0]  = a1; buf.ddata[1]  = b1; buf.ddata[2]  = a2; buf.ddata[3]  = b2;
      buf.ddata[4]  = x10; buf.ddata[5]  = x20; buf.ddata[6]  = x11; buf.ddata[7]  = x12;
      buf.ddata[8]  = x21; buf.ddata[9]  = x22; buf.ddata[10] = r0; buf.ddata[11] = q0;
      buf.ddata[12] = u0; buf.ddata[13] = u1; buf.ddata[14] = k1; buf.ddata[15] = k2;
      buf.ddata[16] = tau0; buf.ddata[17] = tau1; buf.ddata[18] = tmax;
      buf.ddata[19] = epst; buf.idata[40] = n1; buf.idata[41] = n2;
      buf.idata[42] = ntp; buf.idata[43] = ntm; buf.idata[44] = lp;
    }
    MPI_Bcast(buf.ddata,23,MPI_DOUBLE,0,MPI_COMM_WORLD);
    if (mp>0) {
      a1  = buf.ddata[0]; b1  = buf.ddata[1]; a2 = buf.ddata[2]; b2 = buf.ddata[3];
      x10 = buf.ddata[4]; x20 = buf.ddata[5]; x11 = buf.ddata[6]; x12 = buf.ddata[7];
      x21 = buf.ddata[8]; x22 = buf.ddata[9]; r0 = buf.ddata[10]; q0 = buf.ddata[11];
      u0 = buf.ddata[12]; u1 = buf.ddata[13]; k1 = buf.ddata[14]; k2 = buf.ddata[15];
      tau0 = buf.ddata[16]; tau1 = buf.ddata[17]; tmax = buf.ddata[18];
      epst = buf.ddata[19]; n1 = buf.idata[40]; n2 = buf.idata[41];
      ntp  = buf.idata[42]; ntm  = buf.idata[43]; lp = buf.idata[44];
    }
  }
  fprintf(Fo,"Netsize: %d, process: %d, system: %s, tick=%12le\n",np,mp,pname,tick);
  fprintf(Fo,"a1=%le b1=%le a2=%le b2=%le\n",a1,b1,a2,b2);
  fprintf(Fo,"x10=%le x20=%le r0=%le q0=%le tau0=%le\n",x10,x20,r0,q0,tau0);
  fprintf(Fo,"x11=%le x12=%le x21=%le x22=%le k1=%le k2=%le\n",x11,x12,x21,x22,k1,k2);
  fprintf(Fo,"u0=%le u1=%le tau1=%le tmax=%le epst=%le\n",u0,u1,tau1,tmax,epst);
  fprintf(Fo,"n1=%d n2=%d ntp=%d ntm=%d lp=%d\n",n1,n2,ntp,ntm,lp);
  t1 = MPI_Wtime();
  u10 = u1 - u0; omg0 = 1.0 / tau0; omg1 = 1.0 / tau1;
  h1 = (b1-a1)/n1; h12 = h1 * h1; h2 = (b2-a2)/n2; h22 = h2 * h2;
  tau = 0.5 * dmin(h1,h2) / dmax(k1,k2); tau = dmin(tau,1.0/q0);
  tau05 = 0.5 * tau; gam1 = tau05 / h12; gam2 = tau05 / h22;
  s0 = dmin(tmax/tau,1000000000.0); ntm = imin(ntm,(int)s0);
  fprintf(Fo,"u10=%le omg0=%le omg1=%le\n",u10,omg0,omg1);
  fprintf(Fo,"h1=%le h2=%le tau=%le ntm=%d\n",h1,h2,tau,ntm);
  My2DGrid(np,mp,n1,n2,&np1,&np2,&mp1,&mp2);
//
// mp = np1 * mp2 + mp1
//
  if (mp1 ==     0) mp_l = -1; else mp_l = mp - 1;
  if (mp1 == np1-1) mp_r = -1; else mp_r = mp + 1;
  if (mp2 ==     0) mp_b = -1; else mp_b = mp - np1;
  if (mp2 == np2-1) mp_t = -1; else mp_t = mp + np1;
//
// Base group:
//
  MPI_Comm_group(MPI_COMM_WORLD,&gr0);
  cm0 = MPI_COMM_WORLD;
//
// Horizontal group:
//
  for (m=0; m<np1; m++) ranks[m] = np1 * mp2 + m;
  MPI_Group_incl(gr0,np1,ranks,&gr1);
  MPI_Comm_create(MPI_COMM_WORLD,gr1,&cm1);
//
// Vertical group:
//
  for (m=0; m<np2; m++) ranks[m] = np2 * m + mp1;
  MPI_Group_incl(gr0,np2,ranks,&gr2);
  MPI_Comm_create(MPI_COMM_WORLD,gr2,&cm2);
  MyRange(np1,mp1,0,n1,&i11,&i12,&nc1); nc1m = nc1-1;
  MyRange(np2,mp2,0,n2,&i21,&i22,&nc2); nc2m = nc2-1;
  nc12 = nc1 * nc2;
  ncp1 = 2*(np1-1); ncx1 = imax(nc1,ncp1);
  ncp2 = 2*(np2-1); ncx2 = imax(nc2,ncp2);
  ncp = imax(ncp1,ncp2); ncx = imax(ncx1,ncx2);
  ncpx = imax(ncp,ncx);
  fprintf(Fo,"Grid=%dx%d coord=(%d,%d)\n",np1,np2,mp1,mp2);
  fprintf(Fo,"i11=%d i12=%d nc1=%d\n",i11,i12,nc1);
  fprintf(Fo,"i21=%d i22=%d nc2=%d\n",i11,i12,nc2);
  fprintf(Fo,"ncp1=%d ncp2=%d ncp=%d\n",ncp1,ncp2,ncp);
  fprintf(Fo,"ncx1=%d ncx2=%d ncx=%d\n",ncx1,ncx2,ncx);
  if (mp == 0) {
    fprintf(stderr,"n1=%d n2=%d h1=%le h2=%le tau=%le ntm=%d\n",
      n1,n2,h1,h2,tau,ntm);
    fprintf(stderr,"Grid=%dx%d\n",np1,np2);
  }
  xx1 = (double*)(malloc(sizeof(double)*nc1));
  xx2 = (double*)(malloc(sizeof(double)*nc2));
  yy0 = (double*)(malloc(sizeof(double)*nc12));
  yy1 = (double*)(malloc(sizeof(double)*nc12));
  aa1 = (double*)(malloc(sizeof(double)*nc12));
  bb1 = (double*)(malloc(sizeof(double)*nc12));
  aa2 = (double*)(malloc(sizeof(double)*nc12));
  bb2 = (double*)(malloc(sizeof(double)*nc12));
  aa = (double*)(malloc(sizeof(double)*ncx));
  bb = (double*)(malloc(sizeof(double)*ncx));
  cc = (double*)(malloc(sizeof(double)*ncx));
  ff = (double*)(malloc(sizeof(double)*ncx));
  al = (double*)(malloc(sizeof(double)*ncpx));
  y1 = (double*)(malloc(sizeof(double)*ncx));
  if (np>1) {
    y2 = (double*)(malloc(sizeof(double)*ncx));
    y3 = (double*)(malloc(sizeof(double)*ncx));
    y4 = (double*)(malloc(sizeof(double)*9*ncp));
  }
  if (mp_l>=0) {
    rr_l = (double*)(malloc(sizeof(double)*nc2));
    ss_l = (double*)(malloc(sizeof(double)*nc2));
  }
  if (mp_r>=0) {
    rr_r = (double*)(malloc(sizeof(double)*nc2));
    ss_r = (double*)(malloc(sizeof(double)*nc2));
  }
  if (mp_b>=0) {
    rr_b = (double*)(malloc(sizeof(double)*nc1));
    ss_b = (double*)(malloc(sizeof(double)*nc1));
  }
  if (mp_t>=0) {
    rr_t = (double*)(malloc(sizeof(double)*nc1));
    ss_t = (double*)(malloc(sizeof(double)*nc1));
  }
  for (i1=0; i1<nc1; i1++) xx1[i1] = a1 + h1 * (i11 + i1); // grid for x1
  for (i2=0; i2<nc2; i2++) xx2[i2] = a2 + h2 * (i21 + i2); // grid for x2
  for (i2=0; i2<nc2; i2++) { j2 = i21 + i2;
    for (i1=0; i1<nc1; i1++) { j1 = i11 + i1; m = nc1 * i2 + i1;
      if ((j1==0) || (j1==n1)) { aa1[m] = 0.0; bb1[m] = 0.0; }
      else {
        s0 = k(xx1[i1],xx2[i2]); s1 = k(xx1[i1]-h1,xx2[i2]); s2 = k(xx1[i1]+h1,xx2[i2]);
        aa1[m] = gam1*2.0*s0*s1 / (s0 + s1); bb1[m] = gam1*2.0*s0*s2 / (s0 + s2);
      }
      if ((j2==0) || (j2==n2)) { aa2[m] = 0.0; bb2[m] = 0.0; }
      else {
        s0 = k(xx1[i1],xx2[i2]); s1 = k(xx1[i1],xx2[i2]-h2); s2 = k(xx1[i1],xx2[i2]+h2);
        aa2[m] = gam2*2.0*s0*s1 / (s0 + s1); bb2[m] = gam2*2.0*s0*s2 / (s0 + s2);
      }
    }
  }
  ntv = 0; tv = 0.0; gt = 1.0;
  for (i2=0; i2<nc2; i2++) for (i1=0; i1<nc1; i1++) {
    m = nc1 * i2 + i1; yy1[m] = g0(xx1[i1],xx2[i2]); }
// Time loop:
  do {
    ntv++;
    // step 1: yy0 => yy1
    tv += tau05;
    for (m=0; m<nc12; m++) yy0[m] = yy1[m];
    if (np1>1) {
      if (mp_b>=0) {
        i2 = 0;
        for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; ss_b[i1] = yy0[m]; }
      }
      if (mp_t>=0) {
        i2 = nc2m;
        for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; ss_t[i1] = yy0[m]; }
      }
      BndAExch1D(mp_b,nc1,ss_b,rr_b,mp_t,nc1,ss_t,rr_t);
    }
    for (i2=0; i2<nc2; i2++) { j2 = i21 + i2;
      if (j2==0) {
        for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; yy1[m] = g21(tv); }
      }
      else if (j2==n2) {
        for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; yy1[m] = g22(tv); }
      }
      else {
        for (i1=0; i1<nc1; i1++) { j1 = i11 + i1; m = nc1 * i2 + i1;
          if (j1==0) {
            aa[i1] = 0.0; bb[i1] = 0.0; cc[i1] = 1.0; ff[i1] = g11(tv);
          }
          else if (j1==n1) {
            aa[i1] = 0.0; bb[i1] = 0.0; cc[i1] = 1.0; ff[i1] = g11(tv);
          }
          else {
            aa[i1] = aa1[m]; bb[i1] = bb1[m]; cc[i1] = 1.0 + aa[i1] + bb[i1];
            if (i2==0)     s3 = aa2[m] * (yy0[m] - rr_b[i1]);
            else           s3 = aa2[m] * (yy0[m] - yy0[m-nc1]);
            if (i2==nc2m) s4 = bb2[m] * (rr_t[i1] - yy0[m]);
            else           s4 = bb2[m] * (yy0[m+nc1] - yy0[m]);
            s5 = tau05 * f(xx1[i1],xx2[i2],tv);
            ff[i1] = yy0[m] + s4 - s3 + s5;
          }
        }
        ier = prog_rightpn(np1,mp1,cm1,nc1,1,aa,bb,cc,ff,al,y1,y2,y3,y4);
        if (ier!=0) mpierr("Bad solution",1);
        for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; yy1[m] = y1[i1]; }
      }
    }
    // step 2: yy1 => yy2
      ???
    if (ntv % ntp == 0) {
      gt = 0.0;
      for (m=0; m<nc12; m++) {
        s0 = (yy2[m]/yy0[m]-1.0); gt = dmax(gt,dabs(s0));
      }
      gt = gt / tau;
      if (np>1) { s0 = gt; MPI_Allreduce(&s0,&gt,1,MPI_DOUBLE,MPI_MAX,MPI_COMM_WORLD); }
      if (mp == 0) {
        t2 = MPI_Wtime() - t1;
        fprintf(stderr,"ntv=%d tv=%le gt=%le tcpu=%le\n",ntv,tv,gt,t2);
      }
    }
    if (lp>0) {
      fprintf(Fo,"ntv=%d tv=%le gt=%le\n",ntv,tv,gt);
      for (i2=0; i2<nc2; i2++) {
        j2 = i21 + i2;
        for (i1=0; i1<nc1; i1++) {
          j1 = i11 + i1;
          m = nc1 * i2 + i1;
          fprintf(Fo,"i1=%8d i2=%8d x1=%12le x2=%12le y1=%12le\n",
            j1,j2,xx1[i1],xx2[i2],yy1[m]);
        }
      }
    }
  } while ((ntv<ntm) && (gt>epst));
  t1 = MPI_Wtime() - t1;
  sprintf(sname,"%s_%02d.dat",vname,np);
  OutFun2DP(sname,np,mp,nc1,nc2,xx1,xx2,yy1);
  fprintf(Fo,"ntv=%d tv=%le gt=%le time=%le\n",ntv,tv,gt,t1);
  if (mp == 0)
    fprintf(stderr,"Grid=%dx%d n1=%d n2=%d ntv=%d tv=%le gt=%le tcpu=%le\n",
      np1,np2,n1,n2,ntv,tv,gt,t1);
  ier = fclose_m(&Fo);
  MPI_Finalize();
  return 0;
}
```

Трансляция:

```
>mpicc -o ex14b.px -O2 ex14b.c mycom.c mynet.c myio.c -lm
```

Запуск:

```
>mpirun -np <1-16> -nolocal -machinefile hosts ex14b.px 100 100 100 2500  (в классе)
>mpirun -np <1-16> ex14b.px 100 100 100 2500                              (на сервере)
```