MINISTRY OF SCIENCE AND HIGHER EDUCATION
OF THE RUSSIAN FEDERATION
Federal State Autonomous Educational Institution
higher education "NORTH CAUCASUS FEDERAL UNIVERSITY"

METHODOLOGICAL INSTRUCTIONS

for completing laboratory work
in the discipline "Parallel computing" for master's students in the training program 09.04.03 "Applied computer science" (focus (profile) "Management knowledge")

Stavropol, 2026

CONTENT

INTRODUCTION

Lab #1. Basic MPI Functions Lab #2. Virtual Topologies Lab #3.
Introduction to OpenMP

Lab #4. Solving SLAE using iterative methods Lab #5. Solving SLAE using Gauss's method Lab #6. Solving the Poisson problem in 3D using Seidel's method Lab #7. Parallelization of a non-computational problem Lab #8. Working with the Gepard parallel program debugger

Lab #9. Parallel programming in DVM RECOMMENDED REFERENCES

INTRODUCTION

This course of laboratory work is necessary for teaching practical skills in programming parallel algorithms, given in the lectures "Parallel Computing".

As a basis for organizing parallel computing, students are invited to become familiar with the idea of building a parallel programming environment PVM (Parallel Virtual Machine).

Lab #1. Basic MPI Functions

The purpose of the laboratory work:give an idea of the construction of simple parallel programs in the parallel programming language MPI.

Competencies to be developed:PC-3; PC-10; PC-12

Theoretical part

The MPI system is the main programming tool for such modern high- performance multicomputers as Silicon Graphics Origin 2000, Cray T3D, Cray T3E, IBM SP2 and many others. MPI works on a variety of multicomputer architectures, both with distributed memory and with shared memory. In addition, MPI works on computer networks (clusters), homogeneous and heterogeneous. The number of computers in the network can be from one or more. The most important feature of MPI is that the user does not have to take into account all these architectural features of specific multicomputers when writing their parallel programs, since MPI provides the user with a virtual multicomputer with distributed memory and a virtual communication network between virtual computers. The user orders the number of computers necessary to solve his problem and determines the topology of connections between these computers. MPI implements this order on a specific physical system.
The limitation is the amount of RAM of the physical multicomputer. Thus, the user works in a virtual environment, which ensures the portability of his parallel programs. The MPI system is a library of parallel programming tools for the C and Fortran 77 languages.

One of the goals pursued when solving problems on computing systems, including parallel ones, is efficiency. The efficiency of a parallel program depends significantly on the ratio of computation time to the time of communication between computers (during data exchange). And the smaller the percentage of time spent on communications in the total computation time, the greater the efficiency. For parallel systems with message passing, the optimal ratio between computation and communication is provided by coarse-grained parallelization methods, when parallel algorithms are built from large and rarely interacting blocks [2–8]. Linear algebra problems, problems solved by grid methods, and many others are quite effectively *are parallelized using coarse-grained methods*.

MPMD - model of computation . An MPI program is a set of autonomous processes that operate under the control of their own programs and interact via a standard set of library procedures for transmitting and receiving messages. Thus, in the most general case, an MPI program implements the MPMD (Multiple Program - Multiple Data) programming model.

SPMD - model of computation . All processes generally execute different branches of the same program. This approach is due to the fact that the task can be quite naturally divided into subtasks solved by one algorithm. In practice, this programming model (Single program - Multiple Data) is most often encountered [1,12].

The latest model can otherwise be called a model*data parallelization.* Briefly, the essence of this method is as follows. The initial data of the task is distributed among the processes (branches of the parallel algorithm), and the algorithm is the same in all processes, but the actions of this algorithm are distributed in accordance with those available in these processes.*data*. Distribution of algorithm actions consists, for example, in assigning different values to variables of the same cycles in different branches, or in executing different numbers of cycles in different branches.

turns of the same cycles, etc. In other words, the process in each branch follows different execution paths on the same program.

The specified models, in addition to support at the language level, are supported by the architectures of the most modern supercomputers such as: ASCI RED (more than 9000 Pentium PRO/200 combined into a single system, has a performance of about 3.2 Tflops), and ASCI WAIT (8192 - , has a performance of 12.2 Tflops), Cray T3D, Cray T3E, IBM SP2.

Lab assignment

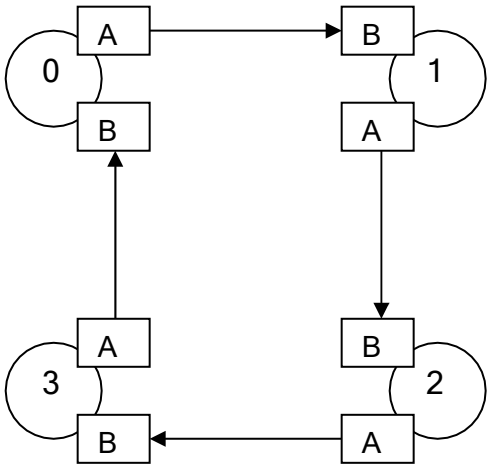
1. Study the theoretical material of this laboratory work.
2. Complete the tasks

Task 1. Conveyor programming.

Branch 0 forwards some data to branch 1, branch 1 forwards the received data to branch 2, branch 2 forwards the received data to branch 3, and so on in ascending order. Finally, branch 0 receives the forwarded data from branch size-1 and displays its number and the received data. That is, forwarding information over a ring of computers. In this example, the algorithm program must be written to be automatically adjusted to the size of the computing system, i.e. the algorithm must not depend on the number of computers and the program must run on any acceptable number of computers.

Task 2. Programming circular data shifts

All branches simultaneously send some of their data to branches with numbers one greater than the transmitting branches. That is, sending information along a ring of computers, see the figure below. In this example, the algorithm program must be written to be automatically adjusted to the size of the computing system, i.e. the algorithm should not depend on the number of computers and the program must run on any acceptable number of computers..



.

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in MS Word and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the laboratory work, the purpose of the work and a description of the process of completing your work. At the end of the report are given

conclusions about the work done. The report must include screenshots of the work done and a description of them. Each figure must be centered on the page, have a caption (Figure 1 – Creating a subsystem) and a link to it in the text.

Test questions:

1. Parallelization methods and program models supported by MPI.
2. MPI properties that ensure portability of parallel programs and independence from computing systems.
3. Explain the basic principles of performing the functions listed below.
4. MPI program compilation command.
5. Command to launch MPI programs.
6. System actions on the mpirun command.

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1
2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL: http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Laboratory work No. 2. Virtual topologies.

The purpose of the laboratory work:give an idea of the construction of simple parallel programs in the parallel programming language MPI.

Competencies to be developed:PC-3; PC-10; PC-12

Theoretical part

To master the programming of Cartesian topologies and to master the combined functions of data reception and transmission.

In the example, the data is shifted to adjacent branches along both computer coordinates on the two-dimensional torus topology by one step, i.e. all branches of the parallel program simultaneously transmit data to adjacent branches, for example, in the direction of increasing values along one and along the other computer coordinate.

```
# include <mpi.h>
# include <stdio.h> #
defineDIMS2
int main(int argc, char** argv)
   { int          rank, size, i, A, B, dims[DIMS]; periods[DIMS], sourc1, sourc2, dest1,
     int          dest2; reorder = 0;
     int
     MPI_Com      tor_2D;
     m
     MPI_Status status; MPI_Init(&argc,
     &argv);
  /*Each branch knows the total number of branches*/
     MPI_Comm_rank(MPI_COMM_WORLD, &rank); /*and
  your number: from0 to (size-1)*/
     MPI_Comm_size(MPI_COMM_WORLD, &size); A = rank;
     B = -1;
  /*Zeroing out the arraydims and fill the periods array for the "two- dimensional torus"
topology*/
     for(i = 0; i < DIMS; i++) { dims[i] = 0; periods[i] = 1;
}
  /*Filling the arraydims, which specifies the dimensions of the grid*/ MPI_Dims_create(size,
     DIMS, dims);
  /*Create a "2D torus" topology with a communicatortop2D*/
     MPI_Cart_create(MPI_COMM_WORLD, DIMS, dims, periods,
reorder,
                                                                    &
tor_2D);
  /*Each branch finds its neighbors along the coordinates, in the direction greater values
    *  coordinates*/
     MPI_Cart_shift(tor_2D, 0, 1, &sourc1, &dest1);
     MPI_Cart_shift(tor_2D, 1, 1, &sourc2, &dest2);
  /*Each branch simultaneously transmits its data (the value of the variable
A) adjacent branch
     *   with a larger coordinate value and accepts data inB from
neighboring branch
     *  with a smaller coordinate value along the "ring".*/ MPI_Sendrecv(&A, 1,

     MPI_INT, dest1, 2, &B, 1, MPI_INT, sourc1, 2,
```

```
                                                                    tor_2D,
&status);
                printf("rank = %d B=%d\n", rank, B); MPI_Sendrecv(&A, 1, MPI_INT, dest2, 2,
                &B, 1, MPI_INT, sourc2, 2,


                                                                    tor_2D,
&status);
          /*Each branch prints its own number (which was also sent to the neighboring branch) and variable
value
           * B (rank of adjacent branch)*/

                printf("rank = %d B=%d\n", rank, B);
          /*All branches complete system processes associated with the topology top_2D and
finish
            *  program execution*/
               MPI_Comm_free(&tor_2D);
               MPI_Finalize();
           return 0;
          }
```

Lab assignment

Parallel matrix-vector multiplication on the "ring" topology. Develop an algorithm, write and debug a parallel program for matrix-vector multiplication in the "ring" topology: A x B = C, provided that the number of matrix rows and vector elements is not evenly divisible by the number of computers. For example, matrix A of size [22x22] and vector B of size [22] on four computers.

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in the MS Word editor and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the lab work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions on the work done are provided. The report must include screenshots of the work done and a description of them. Each figure must be centered on the page, have a caption (Figure 1 - Creating a subsystem) and a link to it in the text.

Test questions:

1. What are virtual topologies used for?
2. How are "Cartesian" topologies defined in MPI?
3. How are graph topologies defined in MPI?
4. What parallelization method is used in parallel algorithms matrix-vector and matrix-matrix multiplication using MPI?
5. How are the elements of a matrix and a vector distributed when a matrix is multiplied by vector on the "ring" topology?
6. How are the elements of both matrices distributed when the matrix is multiplied by matrix on the "ring" topology?
7. What is the best way to represent the second matrix in the computer memory when multiplying? matrix to matrix, to speed up the time it takes to solve the problem?

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming.
BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1

2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8).
BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL:
http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Lab #3. Introduction to OpenMP

The purpose of the laboratory work:getting an idea aboutOpenMP.Formable competencies:PC-3; PC-10; PC-12 Theoretical part

The OpenMP interface is intended as a standard for programming on scalable SMP systems (SSMP, ccNUMA, etc.) in the shared memory model. The OpenMP standard includes specifications for a set of compiler directives, procedures, and environment variables. Examples of shared memory systems scalable to a large number of processors include the Cray Origin2000 (up to 128 processors), HP 9000 V- class (up to 32 processors in a single node, and up to 128 processors in a 4-node configuration), and Sun Starfire (up to 64 processors).

A shared memory process can consist of multiple threads, several threads of control that share a common address space but different instruction streams and separate stacks. In the simplest case, a process consists of a single thread. Threads are sometimes also called streams, lightweight processes, LWP (light-weight processes).

OpenMP is based on the existence of multiple threads in a shared memory programming paradigm.

OpenMP has an explicit (not automatic) programming model, offering the programmer complete control over parallelization.

Fork-Join Model

OpenMP uses the Fork-Join model of parallel execution:

All OpenMP programs start as a single process: the main thread. The main thread executes sequentially until it encounters the first region of a parallel construct.

Fork: The main thread creates a group of parallel threads. The instructions in the program that are included by the parallel region construct are then executed in parallel among the different threads of the group.

Join: When the threads of a group complete the statements in a parallel construct region, they synchronize and close, leaving only the main thread.

Lab assignment

1. Develop an algorithm, write and debug a parallel multiplication program matrices to vectors using work distribution for parallel processes with the sections directive.
2. Develop an algorithm, write and debug a parallel multiplication program matrices to vectors using work distribution for parallel processes with direct assignment of work ("manually").

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in MS Word and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the laboratory work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions about the work done are provided. Screenshots of the work done must be inserted into the report and a description must be added to them. Each figure must be centered

pages, have a signature (Figure 1 – Creating a subsystem) and a link to it in the text.

Test questions:

1. Parallelization methods and program models supported by OpenMP.

3. How are parallel processes defined when multiplying a matrix by a vector and matrix to matrix in OpenMP?

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1

2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL: http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Laboratory work No. 4. Solution of SLAE by iterative methods

The purpose of the laboratory work:getting an idea of the SLAE solution iterative methods.
Competencies to be developed:PC-3; PC-10; PC-12
Theoretical part
Solution of SLAE by simple iteration method
Here are the formulas and sequential programs of this method. The system of linear algebraic equations is given:

a00x0+a01x1+ ... +a0nxn= f0
a10x0+a11x1+ ... +a1nxn= f1 (1)
.
.....an0x0+ a
n1x1+ .. +an.n.xn= fn

Brief entry:

$$Ax = f \qquad (2)$$

The roots of this system are calculated by the simple iteration method using the following formula:

$$x_i^{k+1} = x_i^k - \tau \cdot \left(-a_{ij} x_j^k + f_i\right) \quad i = 0,1,\ldots,N \qquad (3)$$

Here the superscript denotes the iteration step number, i is the root or right-hand side number, τ is the iteration step.

Completion of calculations according to the condition:

$$\frac{\left\| \sum_{j=0}^{N} a_{ij} x_j^k - f_i \right\|}{\|f\|} \qquad (4)$$

Comment.

$$\|f\| = \sqrt{\sum_{i=0}^{N} f_i \cdot f_i} \qquad (5)$$

The numerator of expression (4) is calculated using a similar formula.
Below is a program for the sequential algorithm:

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

# define N 100

double A[N][N], F[N], mf, X[N], X1[N], S[N], msf;

main()
 { int i, j;
   long int dt;
   double t, e;
   struct timeval tv1, tv2;

 /* Data generation. Here a matrix with elements equal to 1 is specified,
```

* equal to 2 on the diagonal. The matrix is taken to be well-conditioned.
* If the right side is equal to N+1, all roots will be equal to 1. */

```
  mf = 0;
  for(i = 0; i < N; i++)
   { for(j = 0; j < N; j++)
     (i==j)?(A[i][j]=2):(A[i][j]=1);
     F[i] = N+1;
 /* We immediately calculate the sum of the squares of the elements of the vector F,
  * i.e. the radical expression of formula (4). */ mf +=
     F[i]*F[i];
   }

/* Set the step t and e. */
  t = 0.01;
  e = 0.00001;

/* Set the initial approximation of the roots. X1 stores the values of the roots
 *  to  +1  iteration.  */
  for(i = 0; i < N; i++)
  X1[i] = 0.6;

/* We record the start time of calculations. */
    gettimeofday(&tv1,NULL);

    do
      { for(i = 0; i < N; i++)
/* X stores the values of the roots of the k-th iteration. */
        X[i] = X1[i];

       for(msf=0,i = 0; i < N; i++)
        { for(S[i] = 0,j = 0; j < N; j++)
            S[i] += A[i][j] * X[j];
              X1[i] = X[i] - t*(S[i] - F[i]);
 /* We calculate the sum of the squares of the residual elements, i.e. the radical
  * expression of the numerator of formula (4).
         */ msf += (S[i] – F[i])*(S[i] – F[i]);
        }
      }
    while(msf/mf > e*e); /* Checking the condition using formula (3). */

/* We record the end time of the calculations. */
    gettimeofday(&tv2,NULL);
    dt = (tv2.tv_sec - tv1.tv_sec) * 1000000 + tv2.tv_usec - tv1.tv_usec;

/* Display the calculation time in seconds */
    printf(" Time= %d\n",dt);
/* For control, we output the first 4 roots */ for(i = 0; i
    < 4; i++)
     printf(" %f\n",X[i]);
 return(0);
  }
```

Note the time measurement function gettimeofday(&tv,NULL). This function can be used for parallel programs as well.

Lab assignment

Task 1. Working through examples from the section “Examples of parallel programs” of this same laboratory.

Task 2. Parallel algorithm for solving SLAE using simple iteration method in MPI

Task 3. Parallel algorithm for solving SLAE by the adjoint method gradients in MPI

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in the MS Word editor and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the lab work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions on the work done are provided. The report must include screenshots of the work done and a description of them. Each figure must be centered on the page, have a caption (Figure 1 - Creating a subsystem) and a link to it in the text.

Test questions:

1. What parallelization method is used in parallel algorithms solving SLAEs using simple iteration and conjugate gradient methods in MPI?
2. How are the elements of the coefficient matrix and the solution vector distributed when solving SLAE using simple iteration method in MPI?
3. How are the elements of the coefficient matrix and the solution vector distributed when solving SLAE using the conjugate gradient method in MPI?
4. How are algorithms for solving SLAEs parallelized using simple iteration methods? and conjugate gradients in OpenMP?

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1

2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL: http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Laboratory work #5. Solution of SLAE by Gauss method

The purpose of the laboratory work:getting an idea of the solution of the SLAE by the method Gauss.

Competencies to be developed:PC-3; PC-10; PC-12

Theoretical part

It is required to find a solution to a system of linear algebraic equations:

$$a11x1+ a12x2+ a21x2 \quad \dots + a1nxn= f1$$
$$+ a22x2+ \quad \dots + a2nxn= f2$$
$$\dots\dots\dots\dots\dots\dots an1x1+ an2x2+ \dots + an.n.$$
$$xn= fn$$

The Gauss method is based on the successive elimination of unknowns. Here, two algorithms for solving SLAEs using the Gauss method are considered. They are associated with different ways of representing*data*(matrices of coefficients and right-hand sides) in the distributed memory of a multicomputer. The data distribution schemes among computers for both examples are given below. Although the data are distributed differently in the memory of a multicomputer in each algorithm, both of them are implemented on the same topology of computer communication - "complete graph".

Lab assignment

Task 1. Parallel algorithm #1 for solving SLAE using the Gauss method in MPI Task 2. Parallel algorithm #2 for solving SLAE using the Gauss method in MPI Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in the MS Word editor and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the lab work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions on the work done are provided. The report must include screenshots of the work done and a description of them. Each figure must be centered on the page, have a caption (Figure 1 - Creating a subsystem) and a link to it in the text.

Test questions:

1. What parallelization method is used in parallel algorithms solving SLAE using Gaussian method in MPI?

2. How are the elements of the coefficient matrix and the vectors of the right-hand sides distributed? when solving SLAE using the Gauss method using algorithm No. 1?

3. How are the elements of the coefficient matrix and the vector of the right parts distributed? when solving SLAE using the Gauss method using algorithm No. 2?

4. Which of the Gauss methods is faster? List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1

2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL:

http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Lab work #6. Solution of the Poisson problem in 3D by the method Seidel

The purpose of the laboratory work:getting an idea of the solution to the problem Poisson in 3D by Seidel method.

Competencies to be developed:PC-3; PC-10; PC-12

Theoretical part

Solve Poisson's equation

$$\frac{-2-}{-x} - \frac{-2-}{-y} - \frac{-2-}{-z} - a\text{---}(x,y,z) \qquad (1)$$

---(x,y,z),a-0

in the area-with boundary conditions of the 1st kind

-|G-$F(x,y,z)$

Solution area-has the form of a rectangular parallelepiped with dimensions Xmx Ymx Zm

.

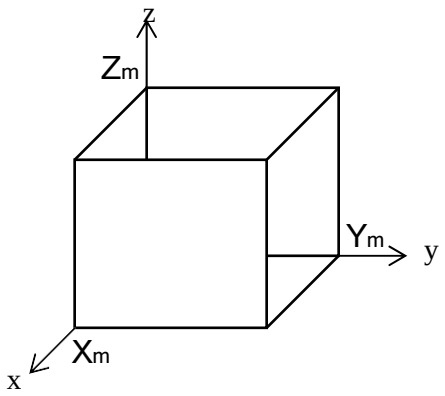


Fig. 6.1. Solution domain-

A uniform rectangular grid with steps of:

hx= Xm/Im, hy= Ym/Jm, hz= Zm/Km,

where Im, Jm, Km- the number of cells in the x, y, z directions. The values of the grid function are specified in the grid nodes

-i,j,k--(xi,yj,zk)

Wherei = 0,…,Im, j = 0,…,Jm, k = 0,…,KmIn the selected grid, equation (1) is approximated:

$$\frac{2\text{-}i,j,k\text{--}}{i\text{-}1} \quad \frac{\text{-}j\text{-}1\text{-}\ 2\text{-}i,j,k\text{--}}{j\text{-}1} \quad \frac{\text{-}k\text{-}1\text{-}\ 2\text{-}}{i,j,k\text{--}} \quad k\text{-}1\text{-----}$$

$$h_x 2 \qquad h_y 2 \qquad h_z 2 \qquad i,j,k$$

i = 1,…,Im-1 , j = 1,…,Im-1 , k = 1,…,Im-1.

The resulting system of linear algebraic equations is solved using the Seidel iterative method:

$$\left(\frac{2}{h_x 2} - \frac{2}{h_y 2} - \frac{2}{h_z 2} - a\right) \text{-}m1\text{-} - \quad i,j,k \quad \frac{\text{-}m_i\text{-}1\text{-}m\text{-}1\ i\text{-}1}{h^2_x} - \frac{\text{-}m_j\text{-}1\ \text{--}m\text{-}1_{j\text{-}1}}{h^2_y} - \frac{\text{-}m_k\text{-}1\ \text{-}\ \text{-}m_k\text{-}1 1}{h^2_z} \text{--} \quad i,j,k$$

or

$$\text{-}m_{ijk} = \frac{\text{-}m_{i\text{-}1jk}\text{-}m_{i\text{-}1}1_{jk} \quad \text{-}m_{ij\text{-}1} \quad k\text{-} \quad m\text{-}\ 1\ \text{-}\ m \quad \text{--}m1\ \text{-}}{h_x 2 \quad h_y 2 \quad h_z 2} \quad \text{-}ij \quad \text{-}1k \quad ijk\text{-}1 \quad ijk\text{-}1\text{--} \quad ijk$$

$$\frac{2}{h_x 2} - \frac{2}{h_y 2} - \frac{2}{h_z 2} - a$$

Herem is the iteration number. Recall that Seidel's scheme is implicit.

Input parameters:Xm, Ym, Zm, Im, Jm, Km, έ, α function-(*x*,*y*,*z*), φ|Ġ.

Iterations are carried out until the following condition is met:

$$\max |\text{-}m\text{-}1\text{--}m| \text{--}$$

Output data:

1) max |-*m*-1--*precise*|

2) function -*m*-1--*precise*

Lab assignment

Task 1. Construction of a parallel algorithm for solving the Poisson problem using the Seidel method in MPI

Task 2. Launching a parallel algorithm for solving the Poisson problem for different functions

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in the MS Word editor and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the lab work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions on the work done are provided. The report must include screenshots of the work done and a description of them. Each figure must be centered on the page, have a caption (Figure 1 - Creating a subsystem) and a link to it in the text.

Test questions:

1. What parallelization method is used to solve problems using in your implementation of grid methods?
2. How can the 3D computing space be distributed on a distributed computing system?
3. Which of the parallel algorithms for solving the Poisson problem in 3D do you think will be faster: 1) on a line of computers; 2) on a two-dimensional grid of computers; or 3) on a three-dimensional grid of computers?

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1
2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL: http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Lab #7. Parallelization of non-computing tasks

The purpose of the laboratory work:getting an idea of parallelization is not computational problem.

Competencies to be developed:PC-3; PC-10; PC-12

Theoretical part

Solution of SLAE by simple iteration method

The formulas and sequential programs of this method are given here.

A system of linear algebraic equations is given:

$$\begin{aligned} &a00x0+a01x1+ \dots +a0nxn= f0\\ &a10x0+a11x1+ \dots +a1nxn= f1 \qquad (1)\\ &\dots\dots\dots\dots\dots\\ &an0x0+an1x1+ \dots +an.n.xn= fn \end{aligned}$$

Brief entry:

$$Ax = f \qquad (2)$$

The roots of this system are calculated by the simple iteration method using the following formula:

$$x_i^{k-1} - X^{i--*}\left(-\sum_{j=0}^{N} a_i x_j^{k} - f\right) \quad i).....i\text{-}0,...,N \qquad (3)$$

Here the superscript denotes the iteration step number,i- number of the root or right part.τ– iteration step.

Completion of calculations according to the condition:

$$\left\| \sum_{j=0}^{N} a\ xk - f \right\| \Big/ \|f\| \qquad (4)$$

Comment.

$$\|f\| = \sqrt{\sum_{i=0}^{N} (f*f)_i\ _i} \qquad (5)$$

The numerator of expression (4) is calculated using a similar formula.

Lab assignment

Task 1. Sorting sets. Develop a parallel algorithm, write and debug a parallel program for sorting sets in MPI. (Any sorting method is optional).

Task 3. Parallel algorithm for solving SLAE by the adjoint method gradient locally optimal scheme in OpenMP

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in MS Word editor and formatted according to the requirements. Formatting requirements: TimesNewRoman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm,

top and bottom – 2 cm. Paragraph indent – 1.25. Text should be aligned by width. The report should contain a title page with the topic of the lab work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions are given on the work done. Screenshots of the work done must be inserted into the report and a description must be added to them. Each figure must be centered on the page, have a caption (Figure 1 – Creating a subsystem) and a link to it in the text.

Test questions:

1. What parallelization method is used to solve problems using in your implementation of grid methods?

2. How can the 3D computing space be distributed on a distributed computing system?

3. Which of the parallel algorithms for solving the Poisson problem in 3D do you think will be faster: 1) on a line of computers; 2) on a two-dimensional grid of computers; or 3) on a three-dimensional grid of computers?

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1

2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL: http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Lab #8. Working with the parallel debugger

Gepard programs

The purpose of the laboratory work:getting an idea of how to work with the debugger Gepard parallel programs.

Competencies to be developed:PC-3; PC-10; PC-12

Theoretical part

Parallel algorithms and programs are usually more complex than sequential ones. Parallel programs are also more difficult to debug. In addition, they may contain errors that are not typical for sequential programs, caused by incorrect synchronization of processes or threads and incorrect use of means that provide parallelism. Debugging of parallel programs is significantly complicated by their non- deterministic behavior, since it complicates the use of the usual method of gradual localization of errors by multiple launches of the program under the control of the debugger. The complexity of creating good means for debugging and analyzing parallel programs is, on the one hand, a consequence of the problems of developing parallel programs, and on the other hand, the debugger of parallel programs is also a parallel program that must interact with the debugged program, also parallel, which complicates the problem even more.

In dialog debugging and analysis of parallel programs, the problem of visualization of the obtained results is especially relevant. Unlike a simple sequential program, where there is one current point of program execution, and the variable has one value, in a parallel program there can be many execution points, hundreds or even thousands. The variable can also have many values - each process has its own. The creators of debugging and analysis tools for parallel programs face the difficult task of presenting all the available information and providing means of managing it, on the one hand, in a convenient, understandable form for the user, and on the other hand, giving the user an exhaustive, complete picture of the behavior of the program being studied.

Lab assignment

Task 1. View MPI programs: parallel matrix-vector and matrix-matrix multiplication, running on 4 and 8 processors.

Task 2. View the MPI program for solving the Poisson problem in a 3- dimensional domain, running on 4 and 8 processors.

Task 3. View the MPI set sorting program running on 4 and 8 processors.

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in the MS Word editor and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the lab work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions on the work done are provided. The report must include screenshots of the work done and a description of them. Each figure must be centered on the page, have a caption (Figure 1 - Creating a subsystem) and a link to it in the text.

Test questions:

1. Behavioral errors in a parallel program.

2. Methods for detecting behavioral errors.
3. Approaches to describing partial behavior.
4. Technical requirements for a software debugger oriented towards debugging the behavior of the PP.
5. Debugger architecture.
6. The GEPARD debugger, focused on debugging programs for multicomputers.

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1
2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL: http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

Lab #9. Parallel programming in DVM

The purpose of the laboratory work:getting an idea of parallel programming in DVM.

Competencies to be developed:PC-3; PC-10; PC-12

Theoretical part

The C-DVM language is designed for developing mobile and efficient parallel programs of a computing nature. It is an extension of the C language in accordance with the DVM model developed at the Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences.

Computer programs for sequential computers have traditionally been written in Fortran 77 and C. For multiprocessor computers with distributed memory and computer networks, such programs are currently usually written in Fortran 77 and C, extended by message-passing libraries (PVM, MPI). The development of such parallel programs requires much more effort from the application programmer than the development of sequential programs, since he must not only distribute data and calculations between different processors, but also organize the interaction of processors by exchanging messages. In fact, a parallel program is a system of interacting programs, each of which is executed on its own processor. Programs on different processors may be completely different, may differ only slightly, or may be a single program whose behavior depends on the processor number. But even in this last case, the programmer is usually forced to develop and maintain two different versions of his program - sequential and parallel.

When using the C-DVM language, the programmer has only one version of the program for both sequential and parallel execution. This program, in addition to describing the algorithm using the usual means of the C language, contains rules for parallel execution of this algorithm. These rules are designed syntactically in such a way that they are “invisible” to standard C compilers for sequential computers and do not interfere with the execution and debugging of the DVM program on workstations as an ordinary sequential program.

Thus, the C-DVM language is a standard C language supplemented with means for specifying parallel program execution:

- distribution of array elements between processors;
- distribution of cycle turns between processors;
- specification of parallel executing sections of the program (parallel tasks) and mapping them to processors;
- organization of effective access to remote (located on processors) others data;
- organization of effective implementation of reduction operations - global operations with data located on different processors (such as summing them or finding their minimum value).

To create programs in the C-DVM language, a special system for automating the development of parallel programs is used – the DVM system.

The DVM system provides a unified set of tools for developing parallel programs for scientific and technical calculations in the C and Fortran 77 languages.

Lab assignment

Task 1. Multiplication of a matrix by a vector. Write a program in C_DVM and in F-DVM. The algorithm for multiplying a matrix by a vector is in lab work #1. Run the program on 4 processors. Look at different options for displaying vector elements on matrix elements.

Task 2. Multiplication of a matrix by a matrix. C = A * B Write a program in C_DVM and in F-DVM. Run the program on 4 processors. Look at different display options, for example, of elements of matrices A and B on elements of matrix C.

Equipment and materials:to complete this lab work A computer with the Windows 7 operating system and software products installed is required: MS Office package.

Safety instructions:to perform laboratory work Students who are familiar with the rules of working in the laboratory and have undergone safety training are allowed to attend.

Contents of the report:The laboratory report must be completed in in the MS Word editor and formatted according to the requirements. Formatting requirements: Times New Roman font, one and a half spacing, left margins - 3 cm, right - 1.5 cm, top and bottom - 2 cm. Paragraph indent - 1.25. The text should be aligned by width. The report should contain a title page with the topic of the lab work, the purpose of the work and a description of the process of completing your work. At the end of the report, conclusions on the work done are provided. The report must include screenshots of the work done and a description of them. Each figure must be centered on the page, have a caption (Figure 1 - Creating a subsystem) and a link to it in the text.

Test questions:

1. Behavioral errors in a parallel program.
2. Methods for detecting behavioral errors.
3. Approaches to describing partial behavior.
4. Technical requirements for a software debugger oriented towards debugging the behavior of the PP.
5. Debugger architecture.
6. The GEPARD debugger, focused on debugging programs for multicomputers.

List of literature recommended for use on this topic:

1. Billig V. A. Parallel computing and multithreaded programming. BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1
2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8). BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL: http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

RECOMMENDED REFERENCES

1. Billig V. A. Parallel computing and multithreaded programming.
BBK: 32.973. Moscow: National Open University "INTUIT", 2016. P-311. Additional information: 2nd ed., revised. http://biblioclub.ru/index.php? page=book_red&id=428948&sr=1
2. Nikolaev E. I. Parallel computing: a tutorial UDC: 004.41 (075.8).
BBK: 22.18 I73. Stavropol: SKFU, 2016 P-185. URL:
http://biblioclub.ru/index.php?page=book_red&id=459124&sr=1

MINISTRY OF SCIENCE AND HIGHER EDUCATION
OF THE RUSSIAN FEDERATION
FEDERAL STATE AUTONOMOUS EDUCATIONAL
INSTITUTION OF HIGHER EDUCATION "NORTH
CAUCASUS FEDERAL UNIVERSITY"

METHODOLOGICAL INSTRUCTIONS

on organizing independent work on the subject "Parallel computing"
for master's students in the training program 09.04.03 "Applied Informatics"
(focus (profile) "Knowledge Management")

Stavropol
2022

Introduction

MPI is the main programming tool for such modern high- performance multicomputers as Silicon Graphics Origin 2000, Cray T3D, Cray T3E, IBM SP2 and many others. MPI works on a variety of multicomputer architectures, both with distributed memory and with shared memory. In addition, MPI works on computer networks (clusters), homogeneous and heterogeneous. The number of computers in the network can be from one or more. The most important feature of MPI is that the user does not have to take into account all these architectural features of specific multicomputers when writing their parallel programs, since MPI provides the user with a virtual multicomputer with distributed memory and a virtual communication network between virtual computers. The user orders the number of computers necessary to solve his problem and determines the topology of connections between these computers. MPI implements this order on a specific physical system. The limitation is the amount of RAM of the physical multicomputer. Thus, the user works in a virtual environment, which ensures the portability of his parallel programs. The MPI system is a library of parallel programming tools for the C and Fortran 77 languages.

1. General characteristics of independent work of a student studying the disciplineParallel computing - set formation professional competencies of the future master's degree student in the field of study
09.04.03 "Applied Informatics".

2. Objectives of mastering the discipline:

- study of the features of the architecture of multiprocessors computing systems;
- study the basics of parallel programming based on computing complex;
- development of the ability to create parallel algorithms programming for solving typical linear algebra problems.

3. Formed competencies- PC-3; PC-10; PC-12

4. Schedule for completing independent work

No. p/p	Chapter disciplines	Week semesters	SRS in hours	Forms of the current control academic performance
1	Laboratory Job No. 1. Basic functions of MPI	1.	4	Interview on questions of the topic
2	Laboratory Job No. 2. Virtual topologies.	2.	4	Interview on questions of the topic
3	Laboratory Job No. 3. Introduction to OpenMP	3.	4	Interview on questions of the topic
4	Laboratory Job No. 4. Solution of SLAE by iterative methods	4.	4	Interview on questions of the topic
5	Laboratory Job No. 5. Solution of SLAE by Gauss method	5.	4.5	Interview on questions of the topic
6	Laboratory Job No. 6. Solution of the Poisson problem in 3D by Seidel's method	6.	5	Interview on questions of the topic
7	Laboratory Job No. 7. Parallelization Not computational problem	7.	5	Interview on questions of the topic
8	Lab #8. Working with the Gepard parallel program debugger	8.	5	Interview on questions of the topic
9	Laboratory Job No. 9. Parallel programming in DVM	9.	5	Interview on questions of the topic
		TOTAL	40.5	

5. Methodological recommendations for studying theoretical material

At the first stage, it is necessary to familiarize yourself with the working program of the discipline, which examines the content of the topics of the lecture course, the relationship of lecture topics with laboratory classes, topics and types of independent work. For each type of independent work, certain reporting forms are provided.

Sample list of interview questions Lab 1. Basic MPI functions

1. Parallelization methods and program models supported

MPI. 2. MPI properties that ensure portability of parallel
programs and independence from computing systems.

3. Explain the basic principles of implementation of the following
functions.

4. MPI program compilation command.

5. Command to launch MPI programs.

6. System actions on the mpirun command.

Laboratory work No. 2. Virtual topologies.

1. What are virtual topologies used for?

2. How are "Cartesian" topologies defined in MPI?

3. How are graph topologies defined in MPI?

4. What parallelization method is used in parallel
matrix-vector and matrix-matrix multiplication algorithms using MPI?

5. How are matrix and vector elements distributed during multiplication? matrices to vectors on the "ring" topology?

6. How are the elements of both matrices distributed during multiplication?
matrix to matrix on the "ring" topology?

7. What is the best way to represent the second matrix in the computer memory, when matrix-matrix multiplication to speed up the time it takes to solve a problem?

Lab #3. Introduction to OpenMP

1. Parallelization methods and program models supported
OpenMP.

3. How are parallel processes defined when multiplying a matrix by
vector and matrix to matrix in OpenMP?

Laboratory work No. 4. Solution of SLAE by iterative methods

1. What parallelization method is used in parallel
algorithms for solving SLAEs using simple iteration and conjugate gradient methods in MPI?

2. How are the elements of the coefficient matrix and vector distributed?
solutions when solving SLAE using the simple iteration method in MPI?

3. How are the elements of the coefficient matrix and vector distributed? solutions when solving SLAE using the conjugate gradient method in MPI?

4. How are algorithms for solving SLAEs parallelized using methods? simple iteration and conjugate gradients in OpenMP?

Laboratory work #5. Solution of SLAE by Gauss method

1. What parallelization method is used in parallel algorithms for solving SLAE using the Gauss method in MPI?
2. How are the elements of the coefficient matrix and vector distributed? right-hand sides when solving SLAE using the Gauss method using algorithm No. 1?
3. How are the elements of the coefficient matrix and vector distributed? right-hand sides when solving SLAE using the Gauss method using algorithm No. 2?
4. Which of the Gauss methods is faster?

Lab #6. Solving the Poisson problem in 3D using Seidel's method

1. What parallelization method is used to solve problems, using grid methods in their implementation?
2. How can the 3D computing space be distributed on distributed computing system?
3. Which of the parallel algorithms for solving the Poisson problem in 3D In your opinion, which will be faster: 1) on a line of computers; 2) on a two-dimensional grid of computers; or 3) on a three-dimensional grid of computers?

Lab #7. Parallelization of a non-computational task

1. What parallelization method is used to solve problems, using grid methods in their implementation?
2. How can the 3D computing space be distributed on distributed computing system?
3. Which of the parallel algorithms for solving the Poisson problem in 3D In your opinion, which will be faster: 1) on a line of computers; 2) on a two-dimensional grid of computers; or 3) on a three-dimensional grid of computers?

Lab #8. Working with the Gepard parallel program debugger

1. Behavioral errors in a parallel program.
2. Methods for detecting behavioral errors.
3. Approaches to describing partial behavior.
4. Technical requirements for the PP debugger oriented towards debugging the behavior of the software.
5. Debugger architecture.
6. The GEPARD debugger, focused on debugging programs for multicomputers.

Lab #9. Parallel programming in DVM

1. Behavioral errors in a parallel program.
2. Methods for detecting behavioral errors.

3. Approaches to describing partial behavior.
4. Technical requirements for the PP debugger

oriented towards debugging the behavior of the software.

5. Debugger architecture.
6. The GEPARD debugger, focused on debugging programs for multicomputers.

6. Methodological instructions for preparing for the exam

Before the exam, the student must fully complete all assignments for laboratory work. If there are outstanding debts for the current certification in this discipline, the student is not allowed to take the exam. The exam in the discipline is provided in oral form using tickets.

Questions to check the level of training Basic level.Questions to test your knowledge:

1. Distributed memory systems. Interacting processes.
2. Routing and flow control. Process allocation.
3. Reactive systems. Systems with infinite channels.
4. Calculations in anonymous systems. Calculation of Boolean functions.
5. Algorithms for information dissemination.
6. Checking the connectivity of a graph. Algorithm for calculating the shortest paths distances.
7. Consideration of the "subtasks - messages" graph.
8. Data transmission channel in parallel systems.
9. Key indicators of success in distributing subtasks between processors.
10. Distribution of subtasks between processors.
11. Static data transmission scheme.
12. Methods of dividing calculations into independent parts.
13. Necessary conditions for the possibility of organizing parallel calculations.
14. Definition and classification of cluster.
15. Flynn's classifications for multiprocessor computing systems.
16. Definition of supercomputer systems. Advanced level.Questions to test your knowledge:

1. Classes of parallel tasks are supported in systems simulation modeling.
2. Definition of the parallel computation schedule.
3. Graphs of information dependencies in parallel programs.
4. The problem of summing a sequence of numbers.
5. Definition of the computation model.
6. Dependence of program execution times on the number of processors.
7. Methods for debugging distributed programs.

7. Methodological materials defining the assessment procedures knowledge, skills, abilities and (or) experience of activities characterizing the stages of formation of competencies

Procedure of implementationexamis carried out in accordance with the Regulation on the current monitoring of academic performance and midterm assessment of students in higher education programs at SKFU.

The examination ticket includes three questions. You have 45 minutes to prepare for the ticket.

When preparing an answer, the student is given the right to use written reports on completed laboratory work.

When checking the practical assignment, the following is assessed: the practical assignment is not provided.

Current assessment of students is carried out by teachers conducting laboratory classes in the discipline in the following forms: a written report and an interview.

Admission to laboratory work occurs if students have a printed version of the report. The report is defended in the form of a student's report on the work performed and answers to the teacher's questions.

The student receives the maximum number of points if the report design meets the established requirements, and the report fully reveals the essence of the work. The grounds for reducing the grade are:

- Submission of laboratory work late;
- Incomplete execution.

The report may be sent back for revision in the following cases:

- Incomplete execution;
- Weak level of presentation of results;
- Inadequate design. Self-assessment criteria studies literature are listed in the Fund of Evaluation Tools for the discipline "Parallel Computing"