The Ministry of Education and Science of the Russian Federation

Lobachevsky State University of Nizhni Novgorod

Computing Mathematics and Cybernetics faculty

The competitiveness enhancement program

of the Lobachevsky State University of Nizhni Novgorod

among the world's research and education centers

Strategic initiative

“Achieving leading positions in the field of supercomputer technology

and high-performance computing”

INTRODUCTION TO PARALLEL PROGRAMMING

Lecture 9. Parallel Programming with MPI.

Base Data Communication Operation.

Nizhni Novgorod

2014

Lecture_9_. Parallel Programming with MPI. Base data communication operation.

This Section is devoted to the description of parallel programming methods for the distributed memory computer systems with the use of MPI.

The processors in the computer systems with distributed memory work independently of each other. It is necessary to have a possibility to distribute the computational load and to organize the information interaction (data transmission) among the processors in order to arrange parallel computations under these conditions.

The solution to the above mentioned problems is provided by *message passing interface - MPI.*

1. Generally, it is necessary to analyze the algorithm of solving a problem, to select the fragments of the computations, which are independent with regard to the information, in order to distribute the computations among the processors. It is also necessary to perform the program implementation of the fragments and then distribute the obtained program parts on different processors. A simpler approach is used in MPI. According to this approach, *a program is developed for solving the stated problem and this single program is executed simultaneously on all the available processors*. In order to avoid the identity of computations on different processor, it is first of all possible to substitute different data for executing the program on different processors. Secondly, there are means in MPI to identify the processor, on which the program is executed. Thus, it provides the possibility to organize the computations differently depending on the processor used by the program. This method of organizing parallel computations is referred to as the model *single program multiple processes* or *SPMP*[1)].

2. The operation of data communication is sufficient for the minimum variant of organizing the information interaction among the processors (it should also be technically possible to provide communication among the processors –we mean the availability of channels or communication lines). There are many data transmission operations in MPI. They provide various means of data passing and implement practically all the communication operations discussed in Section 3. These possibilities are the main advantages of MPI (the very name of the method MPI testifies to it).

[1)] It is more often referred to as *single program multiple data* or *SPMD.* With regard to MPI it would be more logical to use the term SPMP.

It should be noted that the attempts to create software for data transmission among the processors began practically right after the local computer networks came into being. Such software, for instance, are described in Buyya (1999), Andrews (2000) and many others. This software, however, were often incomplete and, moreover, incompatible. Thus, the portability of programs in case of transferring the software to other computer systems is one of the most serious problems in software development. This problem is strongly felt in development of parallel programs. As a result, the efforts to standardize the software for organizing message transmission in multiprocessor computational systems have been made since the 90-s. The work, which directly led to the creation of MPI, was initiated by the Workshop on Standards for Message Passing in a Distributed Memory Environment, Williamsburg, Virginia, USA, April 1992. According to the results of the workshop a special workgroup was created, which was later transformed into an international community (*MPI Forum*). The result of its activity was the creation and adaptation in 1994 of the *Message Passing Interface* standard, version 1.0. Later the standard was sequentially developed. In 1997 MPI, version 2.0, was adopted.

So now it is reasonable to explain the concept MPI. First of all, MPI is a standard for organizing message passing. Secondly, MPI is the software, which should provide the possibility of message passing and correspond to all the requirements of MPI standard. According to the standard this software should be arranged as program module libraries (*MPI libraries*) and should be comprehensible for the most widely used algorithmic languages C and Fortran. This "duality" of MPI should be taken into account while using terminology. As a rule, the abbreviation MPI is used to refer to the standard and the phrase "MPI library" points to a standard software implementation. However, the term MPI is often used to denote MPI libraries also, and thus, the correct interpretation of the term depends on the context.

There are many works devoted to the problems connected with the development of parallel programs using MPI. Brief survey of useful reference materials may be found at the end of the section. Before we started to describe MPI in details, we find useful to mention some of its advantages:

- MPI makes possible to decrease considerably the complexity of the parallel program portability among different computer systems. A parallel program developed in the algorithmic languages C or Fortran with the use of MPI library will, as a rule, operate on different computer platforms,
- MPI contributes to the increase of parallel computation efficiency, as there are MPI library implementations for practically every type of computer system nowadays. These realizations are carried out with regard to the possibilities of the hardware being used,

MPI decreases the complexity of parallel program development as on the one hand, the greater part of the basic data transmission operations discussed in Section 3 are provided by MPI standard. On the other hand, there are many parallel numerical libraries available nowadays created with the use of MPI.

It must be noted that on the one hand MPI is complex enough, as the MPI standard provides for more than 125 functions. On the other hand, MPI has an elaborate structure: one can start developing parallel software as soon as only 6 MPI functions have been studied. All the advanced MPI capabilities may be mastered as the developed algorithms and programs become more complex. This is the structure - simple-to-complex – the MPI training materials will have.

This section is devoted to the description of the parallel programming methods for the computational systems with the distributed memory and the use of MPI.

It was mentioned at the very beginning of the section that MPI is a *message passing interface*. It is currently one of the basic approaches to develop parallel programs for the distributed memory computer systems. The use of MPI makes possible to distribute the computational load and arrange the information interaction, data transmission among the processors. The term MPI means on the one hand the standard to which the software of arranging message passing must meet. On the other hand, this term denotes the program libraries, which provide the possibility of message passing and meet all the requirements of the standard.

In the second part of the lecture discusses a number of concepts and definitions, which are the basic ones for the standard MPI. Thus, it presents a *parallel program* as a number of simultaneously executed *processes*. These processes may be carried out on different processors but several processes may be located on a processor (in these cases they are executed in the time-sharing mode). A brief characteristic of the concept needed for the description of the message passing operations is given further. The subsection also describes the data types, the communicators and virtual topologies.

In the final part of the lecture offers a brief and simple introduction into the development of MPI based parallel programs. The material of the subsection is a good basis for starting the development of parallel programs of various complexity.

Test questions

1. What minimum set of operations of sufficient for the organization of parallel computations in the distributed memory systems?
2. Why is it important to standardize message passing?
3. How can a parallel program be defined?
4. What minimum set of MPI functions makes possible to start the development of parallel programs?

5. How are the messages being passed described?
6. How do we organize the reception of messages from concrete processes?
7. How do we determine the execution time of an MPI based program?

Practice

1. Develop a program to find the minimum (maximum) vector value.
2. Develop a program to find a scalar product of two vectors.
3. Develop a program where two processes repeatedly exchange messages with a length of *n* bytes. Perform experiments and evaluate dependence of execution time on the message length.

Reference

1. Pacheco, P. (1996). Parallel Programming with MPI. - Morgan Kaufmann.
2. Gropp, W., Lusk, E., Skjellum, A. (1999a). Using MPI - 2nd Edition: Portable Parallel Programming with the Message Passing Interface (Scientific and Engineering Computation). - MIT Press.
3. Gropp, W., Lusk, E., Thakur, R. (1999b). Using MPI-2: Advanced Features of the Message Passing Interface (Scientific and Engineering Computation). - MIT Press.
4. Snir, M., Otto, S., Huss-Lederman, S., Walker, D., Dongarra, J. (1996). MPI: The Complete Reference. - MIT Press, Boston, 1996.