# Using of the Technology of Parallel Programming for Solving Applied Problems

Gulnar Balakaeva
Department of Information Technologies
Kazakh-British Technical University, Almaty, Kazakhstan
g.balakaeva@kbtu.kz

Galymbek Seralin [1]

***Abstract* – It is known that there are various ways of realizing parallel calculations. In this paper we consider calculations for the heat transfer process in a rectangular region. The parallelization mechanism is presented in along with results of calculations and of testing.**

*Index Terms* – boundary conditions, communicator, filtration, hosts, Laplace equation, MPI, MPICH, MPI_Comm, MPI_COMM_RANK, MPI_COMM_SELF, MPI_COMM_NULL, MPI_COMM_WORLD, MPMD, nodes, parallelize, processes, rank, TCP/IP.

## Introduction

Recently there has been increasing attention paid to the speed and accuracy of executing modules to solve problems arising from the mathematical modeling of physical problems. So, powerful supercomputers or multiprocessing computers have been created, which allow speeding up of the performance of the set operations and that therefore increase the speed of achieving results.

Development of high-efficiency technical equipment has occurred in the following directions: vector-processor computers (*CRAY companies, Cray Research*); array-parallel computers with allocated memory; parallel computers with common/shared memory; computers with cluster architecture. As a result program applications supported by high-efficiency technical equipment are used in various areas of science and economy.

Parallel data processing is based on the use of parallel processing algorithms. To be able to work with the special software and to create our own products using parallel algorithms and parallel programming languages it is necessary to acquire the corresponding knowledge.

In the model of programming which is provided by MPI, the program generates some processes cooperating among themselves by means of references to subroutines of transfer and reception of messages. Usually at the initialization of MPI programs a fixed set of processes is created, and each process is carried out on a separate processor. In these processes different programs can be carried out, therefore the model of MPI programming is sometimes named MPMD model (Multiple Program Multiple Data).

## The problem

We consider the problem of finding the stationary distribution of the temperature of a square plate [0, 1] x [0,1].

The distribution of the temperature is governed by Laplace's equation with two independent variables.

$$\Delta T = \partial^2 u / \partial x^2 + \partial^2 u / \partial y^2 = 0. \qquad (1)$$

To define a unique solution we need to specify boundary conditions, which we choose to be:

at x = 0.0 and any values of y: T = 100-200*y,
at y = 0.0 and any values of x: T = 200*x-100,
at x = 1.0 and any values of y: T = 100-200*y,
at y = 1.0 and any values of x: T = 200*i-100.

To solve the problem, we discretise the plate using a 2-dimensional grid N×N. Let N=50. The distances between the nodes are then h =1/N=0.02. The grid contains 2500 nodes, in 196 of which the value of the temperature is set according to the boundary conditions.

The problem consists in finding the temperature at all internal nodes by iteration. (For N=50 there will be 2304 nodes.) Various schemes are available but here we employ Jacobi's method. Initial values of the temperature in the internal nodes of a grid are initially arbitrarily set to zero.

## Hardware and Software Environment

For parallelizing the problem we employ the following hardware/software:

- MPICH version 1.2.5
- Operating System Windows NT/2000 or XP
- Visual Studio 6.0 Enterprise Edition
- Host processors (in the case here, we employ 4 hosts, resulting in 4 communicating processes)
- Local area network (LAN) to connect hosts with TCP/IP sockets.

[1] al-Farabi Kazakh National University, Department of Computing Sciences, Almaty, Kazakhstan, gseralin@gmail.com

In the work described here we use MPICH version 1.2.5. MPICH (MPI CHamelton), is an implementation of the MPI specification which supports work across a wide range of platforms and with various communication interfaces, including TCP/IP.

MPICH includes library and header files and contains more than one hundreds subroutine. Delivered with the MPICH package are resources for visual debugging and profiling of parallel programs.

An interaction area (area of communication) is defined by a group of processes. All processes which belong to an interaction area can communicate with each other. This set of processes describes a special information structure, referred to as a communicator.

A communicator describes the context of the communications for operation of an exchange. Each context sets a separate interaction area. Messages are accepted in the context in which they have been sent, and the messages sent in different contexts do not interfere with each other. Processes connected with an MPI program can cooperate, if they are connected with one communicator (See Figure 1). The value of the communicator used by default (MPI_COMM_WORLD) corresponds to all processes of the given program. To all processes in the field of interaction whole positive numbers from 0 up to some maximum are assigned, and the number of concurrent processes can be obtained by means of a call to the subroutine MPI_COMM_RANK.

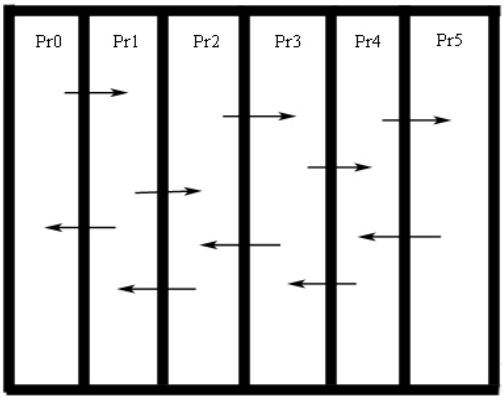


FIGURE 1
Interaction area of MPI program. Here Pr0 – Pr5 are processes.

For the processes entering into an existing interaction area new areas of interaction can be created. The numbering/labeling of processes in different interaction areas is independent. The communicator describes an interaction area. For any one area several communicators can be used. The standard communicator MPI_COMM_WORLD is created automatically.

In the programming language C++ communicators are constants of type MPI_Comm. In addition to MPI_COMM_WORLD the following values of MPI_Comm are also available: MPI_COMM_SELF - communicator, containing the originating process only, and MPI_COMM_NULL – an empty communicator.

The number of a process is referred to us as its rank. The rank is used by each process to, for example, identify a process to transfer messages to. Generally speaking, each process can belong to several interaction areas but in each one it will only have one rank.

For parallelizing a given task we build a cluster with 4 hosts using the operating system Windows XP (SP 2). All characteristics of the hosts can be seen in Table I:

TABLE I
CHARACTERISTICS OF HOSTS

| Host name | Processor | RAM | IP address/ Subnetwork Mask |
|---|---|---|---|
| NIIT8_1 | Intel (R) Pentium(R) 4 CPU 2,66 GHz | 256 MB | 10.44.0.32 / 255.255.255.0 |
| NIIT1_4 | Intel(R) Pentium(R) 4 CPU 2,66 GHz | 256 MB | 10.44.0.31 / 255.255.255.0 |
| NIIT8_2 | Intel(R) Pentium(R) 4 CPU 2,66 GHz | 256 MB | 10.44.0.33 / 255.255.255.0 |
| NIIT7_1 | Intel(R) Pentium(R) 4 CPU 2,66 GHz | 256 MB | 10.44.0.34 / 255.255.255.0 |

**Solution**

The new value of the temperature in each node can be found with the help of a crosswise computing pattern (Jacobi iteration).

$$T_{i,j} := (T_{i-1,j} + T_{i+1,j} + T_{i,j-1} + T_{i,j+1}) \ / \ 4 \qquad (2)$$

Initial values of the temperature at internal nodes of the grid we set to zero and then iterate until convergence. Note that in (2) according to Jacobi's method we use values at the old iteration on the right-hand side when computing the updated values on the left-hand side.

To speed up the process of the solution of the given problem we use a parallelization method with 4 processes. To implement this solution we have written a program in C++. For communication non-blocking communication was used for data transfer (send and receive).

To create a process cluster using the operating system Windows XP I we perform the following steps:

- Install MPICH 1.2.5 and Visual Studio 6.0 Enterprise Edition to all computers which are to be employed in the cluster
- With the help of Visual C ++ 6.0 Enterprise Edition create a C++ project. Compile the project to generate an exe-file.
- Copy this exe-file to all other computers in directory "C:\temp".
- Start MPICH versions 1.2.5 and choose the "MPICH Configuration tool" option. See Figure 2.

- Press the "SELECT" button and choose from the list of computers which appears those that are to be used in the cluster. See Figure 3.
- Press the "Apply" button and the connection will have been obtained. See Figure 4.
- These steps should be done for all computers which are to be used.
- Now start MPICH 1.2.5 and choose MPIRun
- Choose the "…" button to start the problem
- Now specify the number of processes. In our case the number of processes is equal to 4. Press the "run" button and a window will appear for authorization. Login as administrator of a host and the problem solution starts.

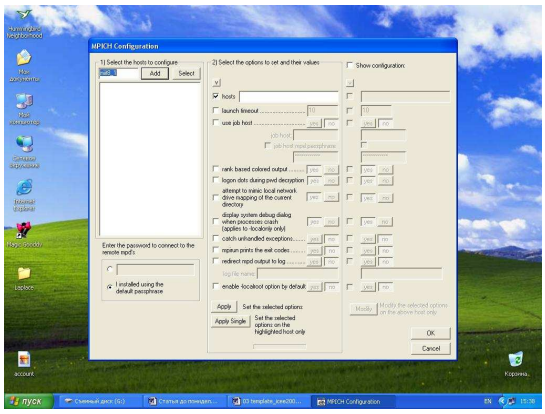

FIGURE 2.
Configuration of MPICH 1.2.5.

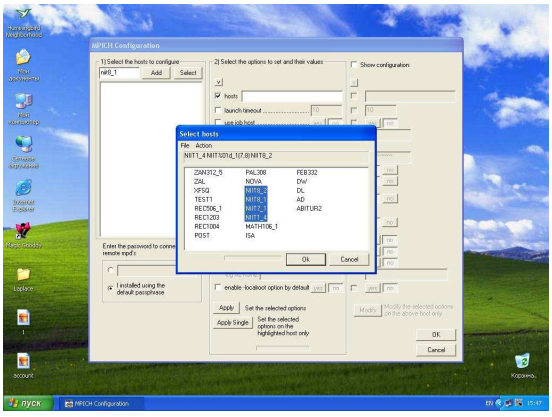

FIGURE 3.
Connection of 4 hosts in MPICH 1.2.5.

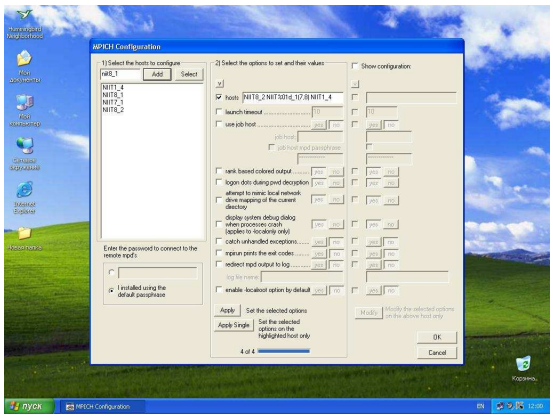

FIGURE 4.
Connection in MPICH 1.2.5.

The results for the solution of our given problem can be seen in Figure 5.

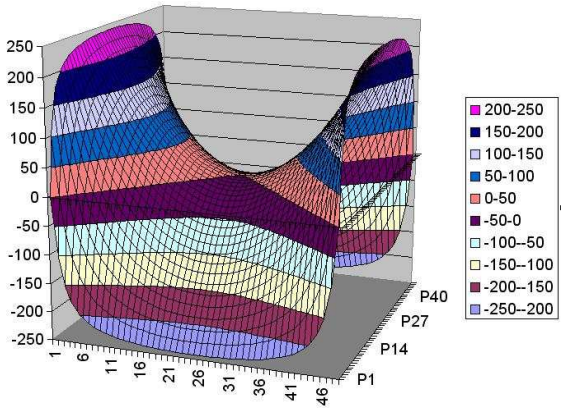


FIGURE 5
Distribution of Temperature in Rectangular Area.

## The way forward

We have developed a programme for numerical calculation using the technology of parallel MPI programming version 1.2.5. Results for the calculation of solution of Laplace's equation (1) on a unit square with appropriate boundary conditions are shown in Figure 1. This computing experiment has shown that results of calculations obtained with the application of parallel programming technologies can lead to an increase in speed of performance (4 times with the program for the solution of Laplace's equation). This relatively straightforward application needs to be extended to our ultimate goal - to perform parallel calculations for a set of coupled PDEs (partial differential equations) that arise from the mathematical consideration of a problem of filtration.