Bibliography

1. www.lustre.org/
2. http://www.netlib.org/atlas/
3. http://www.netlib.org/blacs/
4. http://www.netlib.org/scalapack/
5. http://www.intel.com/cd/software/products/asmo-na/eng/perflib/mkl/index.htm
6. http://www.gromacs.org/
7. http://www.wien2k.at/
8. www.msg.ameslab.gov/GAMESS/GAMESS.html
9. http://www.scali.com/
10. http://www.open-mpi.org/
11. www.t-platforms.ru/english/about/dnd.html
12. www.intel.com/design/servers/ipmi/spec.htm
13. www.redhat.com/software/rha/gfs/
14. oss.oracle.com/projects/ocfs2/

Authors' Information

Ahdrey L. Golovinskiy – Institute of Cybernetics NAS Ukraine; Prospekt Academika Glushkova,40, Kiev, 03680 MCP, Ukraine; e-mail: tikus@ukr.net

Sergey G. Ryabchun – Institute of Cybernetics NAS Ukraine; Prospekt Academika Glushkova,40, Kiev, 03680 MCP, Ukraine; e-mail: sr67@voliacable.com

Anatoliy A. Yakuba – Institute of Cybernetics NAS Ukraine; Prospekt Academika Glushkova,40, Kiev, 03680 MCP, Ukraine; e-mail: ayacuba@gmail.com

THE TECHNOLOGY OF PROGRAMMING FOR A CLUSTER COMPUTER BY THE REMOTE TERMINAL WITH OS WINDOWS

Dmitry Cheremisinov, Liudmila Cheremisinova

Abstract. The problem of preparation of a program to perform it on multiprocessor system of a cluster type is considered. When developing programs for a cluster computer the technology based on use of the remote terminal is applied. The situation when such remote terminal is the computer with operational system Windows is considered. The set of the tool means, allowing carrying out of editing program texts, compiling and starting programs on a cluster computer, is suggested. Advantage of an offered way of preparation of programs to execution is that it allows as much as possible to use practical experience of programmers used to working in OS Windows environment.

Keywords: parallel programming, cluster computer, programming technology.

ACM Classification Keywords: D.2.1 [Software engineering]: Requirements/Specifications – Tools; D.1.3 [Programming techniques]: Concurrent Programming – Parallel programming.

Introduction

In recent years, it has become clear that the future of the high performance computing industry is in cluster computing. Cluster computers are found in the Top 500 List [1] of the highest performance computers in the

world. Major universities, military agencies and scientific research laboratories are potential users of clusters. A cluster computer is a group of processors (computers) that are tightly-coupled through dedicated fast network designed to capture their cumulative processing power for running parallel-processing applications. Cluster computers are specifically designed to take large programs and sets of data and subdivide them into component parts, allowing the individual cluster nodes (processors) to process their own individual parts of the program [2, 3]. The cluster nodes can solve the same problem (as a rule, of the big computing complexity), exchanging data during the computation and sharing some resources, and so they can be viewed as a very complex single computer. One of the cluster nodes is marked out usually as a head one (master node), it starts the running acts as the front-end of the cluster system.

To use multiple processors of the cluster the application has to be partitioned into components that could run concurrently and separately in a distributed programming environment. Since the cluster nodes are in separate locations it is still necessary for intra-process communication in the form of messages sent between processes. The Message Passing Interface (MPI) is a widely used communications protocol that is de facto standard for communication among concurrent processes. MPI implementations consist of a library of routines that can be called from C programs, for example.

The most of the software installed on cluster computers is open-source, freely available. One of the more popular high-performance cluster implementations are cluster computers with cluster nodes running Linux (free version of UNIX-based OS) as the operating system (OS) and communications libraries of MPI, which are specially designed for writing scientific applications for such computers.

A cluster computer is a complex enough object and for skilled job on it one needs of special knowledge on software and hardware of networks, computers and operational systems. The often case is when a real user well acquainted with widespread OS Windows and developing applications in C for personal computers needs to prepare them for running in cluster computer. It would be desirable for him in that case to study as small as possible the technical details connected with management by new operational system when editing programs, compiling and starting them on execution, and to concentrate on development of a parallel algorithm solving his problem. It can be reached if he could use his PC computer with Windows as the remote terminal of a cluster computer. It turns out but nevertheless some minimal knowledge on the functions of OS UNIX is necessary.

The purpose of this paper consists in describing the most basic actions which should be carried out during process of a program development for a cluster computer using PC computer as the remote terminal; and the features of the process of preparation of the program application from the point of view of the user working in OS Windows environment are considered.

The access to a cluster computer

There are many cluster configurations, but a simple architecture such as the one shown in Fig. 1, is used to visualize the basic concept [2]. In a typical cluster, it is common to have a dedicated master node that is the only machine connected to the outside world. This machine then acts as the file server, and the compile node. This provides a single-system image to the user, who launches the jobs from the master node without ever logging into any nodes. However, the computational work is split-up and parsed out to be done by multiple nodes in the cluster.

To have an access to a cluster computer, it is necessary to be the registered user of the master machine of the cluster. To make that unaided, it is necessary to have access to the identification information of the superuser of a cluster (with UNIX-based OS). Usually registration of the new user is carried out by the system administrator of a cluster. As the result of registration the user receives memory volume for his data on the master machine of the cluster and a login (the alphanumeric line of symbols used for identification of the user in operational system) and the password for access.
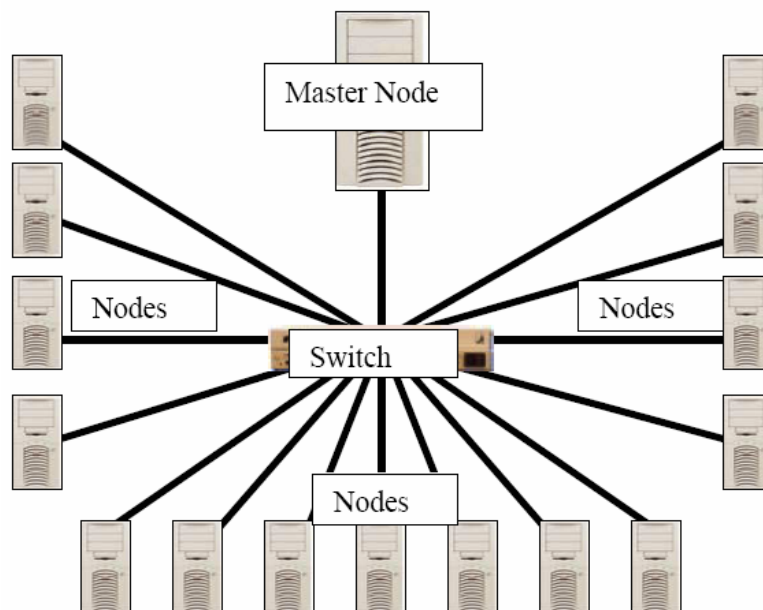
Fig. 1. Typical architecture of a cluster computer

The computer with OS Windows should be connected to the same network, as the master machine of the cluster. For interactive communication with the cluster and an exchange of files it is required to use a protocol with enciphering the information – *SSH* (Secure SHell). If to use the protocol *telnet* in which passwords and commands are passed in the open code, the password of the user can be intercepted (when it is transferred on a network) and could be used for not authorized access (by strangers) to the cluster. To connect a user of Windows to a cluster the terminal program of a cluster access, that uses the protocol *SSH*, is required. It is necessary to establish and adjust it. For this purpose it is convenient to use freely accessible program *PuTTY* which can be found in the following Internet address:

ftp://linux4u.jinr.ru/pub/win9x/crypt/putty-0.53b/x86/putty.exe.

The program *PuTTY* represents remote terminal service, i.e. enables to enter the information through the keyboard and to display it. If source code of the developed program is already on the cluster, *PuTTY* is that is enough for the further job. If the source code is still on the user remote computer the program providing transportation of data from the computer on a cluster is necessary. As a matter of fact a program intended for this purpose is available in OS Windows – that is *ftp*. However the appropriate protocol *FTP* is not protected, it provides transportation of data in open code. So the usage of such a protocol is impossible for safety reasons.

To transport data using the protected transmission channel one should use the program *SecureFX*, for example. It is a client program of the protocol *SHH* that supports the ciphered file transfer and allows to adjust its configuration in a wide range and to choose transfer protocol in use. *SecureFX* supports a mode of downloading and restoration of communication in the case of its breakage. This program extends for a payment. The program *SecureFX* is a client of the service of transport of data that can pass data both by means of traditional protocol *FTP*, and through ciphering protocol *SSH2*.
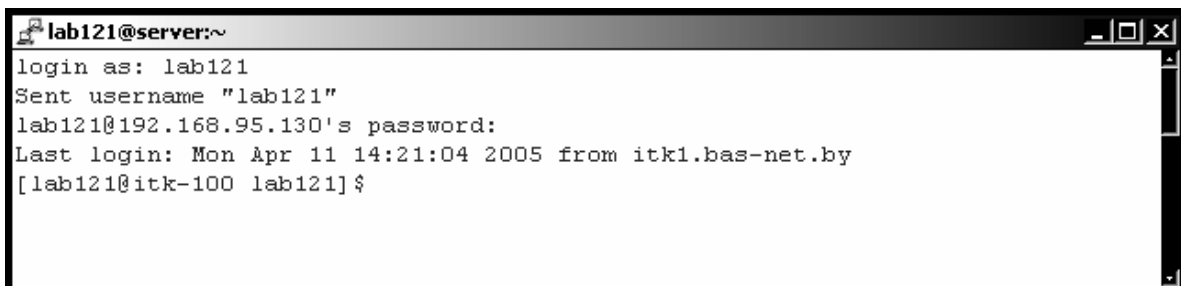
Use of Midnight Commander

In a standard status for execution of a session of a program debugging on a cluster the source code of the program is situated in one of catalogues of the master machine of the cluster. The procedure of connection to the cluster is finished after starting the program *PuTTY*, and on its screen (on the client computer with OS Windows) there is the prompt for typing the command line for the command interpreter of the master machine of the cluster (Fig. 2).

One could find the information on the language of the command line in any OS Linux manual [7]. Usually a manual on OS of UNIX type is a thick enough book in which the description of the instructions of command lines and their parameters makes the overwhelming part. For Windows user it is more convenient to control carrying out calculations on a cluster through its console (remote terminal). In that case he can use the program *mc* (Midnight Commander). It is included into the distributive of OS Linux last versions. For the program *mc* starting it should be typed *mc* in the command line invitation (when using *PuTTY*). After that the *PuTTY* screen becomes as the screen of a known Norton Commander environment (Fig. 3). The majority of control buttons work as well, as in Norton Commander for DOS [4]. It should be noted for those who use Norton Commander for Windows (Windows or Total Commander [5]) or *FAR*, when working in framework of Midnight Commander [6] the difference is connected only with peculiarities of OS Linux file system. So with the help of *mc* file searching and scanning on a cluster (with OS Linux) can be carried out in the same way, as in OS Windows.

```
lab121@server:~                                    _ □ ×
login as: lab121
Sent username "lab121"
lab121@192.168.95.130's password:
Last login: Mon Apr 11 14:21:04 2005 from itk1.bas-net.by
[lab121@itk-100 lab121]$
```

Fig. 2. The sample of the screen contents of the program *PuTTY* after finishing its connection to a cluster

When it is required to correct a little the source code of the debugged program (that is the usual text), it is possible to use the built in text editor of the program shell *mc* (softkey F4) and to edit the program text directly on a cluster. In the case of need of greater program source code corrections it is more convenient to edit it on the user terminal machine, using the habitual editor. The basic inconveniences at editing on a cluster are connected with impossibility of execution of corrections by means of habitual operations *copy-paste*, the difficulty of switching on and off of language (Russian/Latin, for example) and so on.

Program start on the execution is carried out in framework of Midnight Commander by the same way, as in Norton Commander. It is necessary to take into account, that program files in a binary mode in OS Linux have no filename extension. An attribute of that, the file could be run, is the asterisk before the filename (Fig. 3). To assign the access rights, the instruction *chmod* which gets out through the menu command (softkey F9, as well as in Norton Commander for access to the main menu) could be used.

Some difficulty arises when the user would want to shut down immediately his program being in progress. In most cases program execution is stopped after the combination *Ctrl-C*. But before keying it is necessary to pass in a screen mode of the command interpreter of OS Linux, i.e. to close panels of the program *mc*. Closing of the

console on remote terminal machine (by means of program *PuTTY* exit) generally speaking does not stop the user program.

Fig. 3. The sample of the screen contents of the program *PuTTY* after starting Midnight Commander

Program start modes

To start a program it is enough to type in the command line field of the program *mc* its name and the necessary parameters and then to press input. While the started program runs, the user can do nothing with the program, he should wait for its termination. Instead of the mentioned usual mode there exists still the program start mode to run the program background. It allows releasing the console for execution of any other jobs. To start a program in a background mode, it is necessary to append the name of the appropriate file with the symbol «&». To look through the list of background running programs one can use the command *jobs*. In the received list all applications are numbered, and if it is necessary to move up any from them into the usual mode of program running (real time running) it should execute the command *fg* (from foreground) appended with the number of the program, for example *fg 2*. And vice versa to move up a real time started program into background mode it should use the command *bg* (from background). But if the program already runs, it is impossible to enter any command and *bg* too. But the user could bring to a stop the program, pressing the key combination *Ctrl-Z*, and then already use the commands *jobs* and *bg*. Which of the two mentioned program start modes is assigned default when entering program name in the command line field of the program *mc*, depends on the program *mc* setup (usually in a background mode).

Preparation for program compilation

In OS Linux there are environments for developing programs on C++, which provide C++ programmer with the programming support similar to that of MSVC in OS Windows. However the job within the environment C++ in Linux considerably differs from the job within MSVC environment in Windows. One of the grave disadvantages of using a remote terminal computer with Windows for access to a cluster is that in this case one has to be limited to those Linux resources which are accessible through the console. Therefore through the remote terminal of the program *PuTTY* it is impossible in principle to use Linux programs with the graphic interface. On the other hand, the usage of development environment of OS Linux for programming requires special knowledge different from that got in MSVC OS Windows.

Possible strategy of programming for clusters is based on the concept of a code portability which provides independence of the program code from development environment when it has been prepared. For a user accustomed to work in OS Windows environment, the variant of this strategy, allowing as far as possible to provide application of his knowledge of MSVC environment, is convenient. In that case the sequential prototype (realizing sequential algorithm) of the parallel program is developed within MSVC environment taking into account the requirement of program code portability. The code portability requires taking into account the architecture of the console program (its potentiality) when developing program code within MSVC OS Windows. This program code is then transformed to the parallel program which is intended for execution on a cluster. For this purpose texts of the program should be compiled on the Linux platform.

By the master machine of a cluster always there exists a program compiler which is controlled by means of the command line. There is no OS UNIX without such a compiler. The matter is there is no program compatibility at the level of binary codes between different computers, even with the same type of OS UNIX. As for computers with OS Linux there is program compatibility at the level of binary codes only between computers with the same version of a kernel of operational system. Thus, installation of any program in UNIX (except for programs which are carried out by interpreters) usually demands compilation of its source code.

The task for a program compilation with the aid of C++ Linux compiler represents a control file for the program *make*. Thus, to specify program compilation it is necessary to have (in addition to program source code on C++) a file of the task for the program *make*. MSVC OS Windows environment allows exporting a file of the project in the format *make*. However the file of the project exported from MSVC is poorly suitable for control of program compilation on Linux. It demands a manual addition to the great extent that it is easier to write it anew from nothing.

It is possible to compile without resorting to services of the program *make*, calling the compiler C++ direct from the command line, but in this case the compilable program should consist of the only file, and all other parts of the program text should be pointed out obviously by means of the instructions *#include*. It is very inconvenient for many reasons. One of the essential reasons of use of the program *make* consists in that, it gives an opportunity to accelerate compilation of the program consisting of several files. The program *make* allows using object files and carries out recompilation only of those files with the source code which have been changed.

The format of a task for the program *make* is complicated enough, and making of the task demands significant job: it is necessary to remember interdependencies between files and to enumerate every single one file which be required at the program compilation. Therefore often it is necessary to specify in a control file for the program *make* very many data. If at the process of compilation it is required to use various tools (for example, the linkage editor besides the compiler), it is necessary to provide control of each of them. There exist programs, allowing to automatize generating the task file for the program *make*. The program *genmake* is one of them, it can be found in the following Internet address:

http://www.robertnz.net/genmake.htm

When using this program the project, for which the task file for the program *make* would be generated, should be specifically prepared. For this purpose it is necessary to include special instructions into each *h-file* of the project in a format of comments. In these instructions the names of *cpp-files* containing definitions of the objects declared in the corresponding *h-file* are specified. It is necessary for the program *genmake* to assign the name of *cpp-file* containing the function *main* of the program. The program *genmake* looking through this file finds instructions *#include* to include corresponding *h-files* into the formed project. *h-files* that have been found are looked through with the purpose of searching instructions in the format of *genmake*, in which cpp-files containing definitions of objects (declared in a corresponding *h-file*) are specified. Thus, all necessary files are searched and dependences between them are formed. Unfortunately, the file generated by the program *genmake* demands small manual modification

Problem of Russian language

The fundamental difference of situation with use of Russian and other languages in OS Linux is connected with that today there exist several coding pages for Russian –1251, KOI-8r, ISO 8859-5, etc. By default the codepage of Russian in Linux is KOI-8r, and in Windows it is Windows 1251. The distinction in standards of the coding results in that the program texts prepared for compilation in Windows, will look incorrectly in OS Linux. But the problem is connected not only with viewing the program text, but the main problem lies in difficulty to switch keyboard layout on Russian when editing the program text with Russian comments. It is the main reason on which program text editing is more convenient to carry out on the remote computer of the user with OS Windows using MSVC environment as the editor. This editor is irreplaceable, when it is necessary to analyze the structure of brackets nesting. On the other hand, for correct job with Russian the editor should provide a capability of transformation of coding Windows 1251 in KOI-8r and vice versa.

From the point of view of capabilities of program text viewing and editing the text editor *Aditor* is convenient enough. It can be found in the following Internet address:

http://nrd.pnpi.spb.ru/UseSoft/tools/Aditor/ aditor.htm.

Its basic advantage – the complete support almost all codings of Russian alphabet (KOI, Win, DOS, ISO, Mac). "Complete support" means that it is possible to look through and to edit files in any of these codings, and when a text file is opened its coding is recognized automatically and almost always accurately (and if a mistake took place nevertheless, it is possible to correct it). It is possible also to translate files from one coding in another by user request, at that Clipboard will automatically be recoded, etc. Besides in this editor syntactic structures of C++ are displayed in different colors.

Technology of job with the program for a cluster

To start program running on a cluster using remote terminal with OS Windows you need to make a folder (catalogue) on the computer. It is meant to keep texts of the programs for a cluster. These texts differ from texts of the initial prototype prepared for Windows not only in parallelism instructions, but also in that they are recoded (as it is required for OS Linux) in KOI-8r (by means of editor Aditor) and instructions of the program *genmake* are included in *h-files*. The creation and filling such a folder is the first step of program preparation to run it on a cluster computer. Then by means of the program *genmake* it is necessary to construct *a make-file* and to correct it manually. Further the folder with data is duplicated in a catalogue on a master machine of a cluster. For this purpose data from the folder on the computer of the remote terminal with OS Windows are transported in the catalogue on the master machine the cluster (by means of the program *SecureFX*). This is the last step of the spadework for compilation and execution of the program on the cluster.

Further the catalogue on the cluster serves as a mirror of the folder on a user computer of the remote terminal. It is convenient to hold the active files (with which there is a job) opened in the program *Aditor*. To start program compilation and execution on a cluster it is necessary to start the program *PuTTY* and, using it, to run the

program *mc* (Midnight Commander). If at compilation there were mistakes, the user can look numbers of the program lines and error diagnostics under panels of the program *mc*. Using the files opened in *Aditor* on the user computer and the program lines numbers, it is possible to find corresponding lines of the program code and to correct errors in them. After code editing it is necessary to save corrections (not finishing the program *Aditor*) on the user computer and by means of the program *SecureFX* to transport the corrected files to the catalogue on the cluster. After updating the catalogue (contents of the catalogue on the cluster should be identical to that contained in the folder on the user computer) it is necessary to repeat the corrected program compilation.

If the program compilation has passed successfully, the binary file of the program appears on the panel of the program *mc* being labeled by an asterisk. Now it is possible to start this file on execution in the cluster computer again. The results of program run can be watched under the panels of the program *mc*.

Conclusion

The paper makes some recommendations on organization of a software complex for providing performance of typical operations on preparation for starting and execution of programs for a cluster. The case is considered when the master machine of the cluster represents UNIX server, and programs for a cluster are prepared, corrected and started execution using a remote computer with OS Windows. The suggested configuration of software is not the only one of this kind. However using offered technology the programmer, that used to working with program tools for OS Windows, will be able simply enough to develop and debug programs for a cluster with OS Linux. It, certainly, is given by refusal of use of some program tools providing by program system on the basis of Linux. But the suggested approach suffers from disadvantage: it does not enable to use program tools of development of programs for Linux with the graphic interface.

The suggested technology has been applied for development and investigation of several parallel programs [8, 9] intended for computer-aided logical design system, their sequential prototypes have been constructed in MSVC OS Windows environment. The program transfer demands practically no skill of working in OS Linux environment. The prime time was spent on paralleling the sequential programs and their debugging, rather than on mastering at the suggested complex of tool means.

Bibliography

1. TOP500 Supercomputer Sites, http://www.top500.org/; http://parallel.ru/computers/top500.list27.html.
2. M. Baker, A. Apon, R. Buyya, and H. Jin, "Cluster computing and applications". In A. Kent & J. Williams (Eds.), Encyclopedia of Computer Science and Technology, New York: Marcel Dekker, p. 87-125, 2002 (http://www.gridbus.org/~raj/papers/encyclopedia.pdf).
3. M. Baker, R. Buyya, "Cluster computing at a glance". In R. Buyya (Ed.), High performance cluster computing: Architecture and systems, Prentice Hall, p. 3-47, 1999.
4. Norton Commander, http://en.wikipedia.org/wiki/Norton_Commander.
5. Total Commander, http://en.wikipedia.org/wiki/Total_Commander.
6. Midnight Commander, http://www.ibiblio.org/mc/.
7. Linux, http://en.wikipedia.org/wiki/Linux
8. N.R. Toropov, " Parallel DNF verification on tautology", Informatica, no 2 (6), p. 35–42, 2005.
9. N.R. Toropov, "Parallel logic-combinatorial calculations with MPI", Informatica, no 3 (7), p. 82–90, 2005.

Authors' Information

Dmitry Cheremisinov, **Liudmila Cheremisinova** – The United Institute of Informatics Problems of National Academy of Sciences of Belarus, Surganov str., 6, Minsk, 220012, Belarus, Tel.: (10-375-17) 284-20-76, e-mail: cher@newman.bas-net.by, cld@newman.bas-net.by