Bull. Nov. Comp. Center, Comp. Science, 21 (2004), 57–68

The NumGRID metacomputing system

N.V. Malyshkin, B. Roux, D. Fougere, V.E. Malyhskin

1. Introduction

With evolution of mathematical modeling, and with creation of high-performance computing systems, many scientific applications have appeared, demanding ever increasing computer costs, which are higher than any of available supercomputers can provide.

In this connection, it is necessary to integrate several remote computers, i.e., to create a grid. Not each application can be effectively solved on grids because of slow communications. However, such applications as search for alien civilizations and decoding the human genome were successfully solved on grids [1].

In recent years, many large-scale problems have appeared in numerical modeling (evolution of a proto-planetary disk, evolution of galaxies, problems of plasma physics). Their solution demands integration of several multicomputers.

Another problem is the fast progress in microprocessor development which forces one to use heterogeneous computer systems for solution of the large-scale problems. In 2004, the ICM&MG (Siberian Branch of the Russian Academy of Sciences) is planning to use two multicomputers: the 96 processor MVS-1000 based on the alpha microprocessor, and the 32 processor multicomputer T-Forge based on Opteron microprocessor. Therefore, there is a necessity to create the software which could help to organize the large-scale simulation in such heterogeneous environments.

In 2003, the co-operative project NumGRID intended for creation of the necessary grid system software, started in Novosibirsk (ICM&MG) and Marseille (L3M CNRS). Before outlining of the objectives of the project, classification of modern grids is briefly considered.

2. Grid classification

Some important definitions:

Interprocess communications – the act of data transfer between processes of a parallel program in the course of execution;

A component of a grid – a part of a grid. For example: a multicomputer, SMP, clusters;

Node of a grid – a subpart of a component of a grid. For example: a personal computer;

A Grid – a union of remote components. Multicomputers (in particular clusters) or multiprocessors can act as components of a grid.

The first existing grids represented a union of personal computers inside one laboratory, research centres or organizations. Usually, personal computers were used at the night time during their idle time. Such systems might include remote computers from all over the world. One of examples is the SETI project [1] that involves about one million personal computers from all over the world. The SETI grid was developed for the search for alien civilizations, and, currently any interested person can join it and have a chance to be the first to find the alien life. Characteristics of the problems which are efficiently solved on systems of this type are the following:

- There are no interprocess communications between nodes of a grid during computation.
- There is a possibility of dynamic addition and removal of nodes during computation.
- All nodes have global IP addresses.

The next step was in connection of several SMP-systems into a grid. The well-known software for grids of such a type is Globus Toolkit. Characteristics of the problems, efficiently solved on grid systems of this type are:

- There are no interprocess communications between nodes of a grid in the course of computation.
- There is no possibility for dynamic addition, removal of nodes in the course of computation.
- All nodes have global IP addresses.

Other types of grids are not widespread.

Unfortunately only a few scientific problems can be parallelized so that there are no interprocess communications between parts of a problem. And, in particular, for parallel implementation of the large-scale numerical models it is necessary to provide communications between nodes of a grid for data exchange between processes during computation. It is necessary to provide data transfer from the process running on one node to another process running on another remote node.

In scientific applications, the MPI library for transferring messages is standard de-facto. Therefore, the next step was to add a possibility to use the MPI in the grids. This step initiated the development of the new technologies and software for grids. For PC-based grids, the CMDE system

[2] was developed. The CMDE enables the use of the MPI library in the user's programs and provides dynamic addition and removing components of a grid using the checkpoint mechanism.

For the grids based on the SMP-systems, the Globus alliance [3] has developed the MPICH-G2 library [4] that works together with the Globus Toolkit. This library allows one to use the MPI calls in the user's applications on the SMP-systems with the Globus Toolkit installed on them.

Recently, clusters have become very popular all over the world. They have a reasonable price and a good scalability. Actually, all the most powerful computer centres are anyhow constructed employing cluster technologies. The MVS-1000M multicomputer, the most powerful supercomputer in Russia, is also of the cluster type. Thus, the next step in the grid construction was to integrate several clusters into a grid. In particular, the objective of the NumGRID project is to integrate clusters in Marseilles and Lyons, and two clusters in the ICM&MG (MVS-1000 and the cluster based on Opteron) for solution of the large-scale scientific problems in numerical modelling.

How the modern clusters look like? Clusters are constructed as follows: a set of computer nodes (currently a node is a dual-processor system) are connected using a high-speed local network (Gigabit Ethernet, Myrinet, SCI). One of the nodes is configured as a master node. The second network card leading to Internet is installed inside it. The other nodes have no global IP addresses and have no access outside the cluster. We will call such nodes internal nodes of a cluster. Also, there is no access to the internal cluster nodes from outside. Starting applications, management, data transfers are carried out completely via the master node. Thus, a modern cluster represents a closed independent system with a unique point of access.This is the problem. All the current grid systems with a possibility of communications demand the following:

- global IP addresses for each node of a grid,
- an opportunity to establish a connection with each other (generally, a full graph).

Such are the PACX MPI [5] and the CMDE and Albatross and even a well-known MPI-G2 (Globus2 device). It is clear that if it is necessary to send messages between nodes of a grid, then there should be a channel between them and their addresses should be known.

At the same time, the cluster nodes of a grid cannot be accessed from outside. This would strongly reduce the safeness of computation, there occurring a parasitic traffic, and it would be difficult to administrate a cluster. Also, it is difficult to find many global IP addresses for all the nodes (for example, the MVS-1000 cluster of the Moscow supercomputer center consists of 768 nodes). Hence, it is necessary to keep an independent cluster structures and, at the same time, to connect them in a grid with a possibility

of the internal communications. The project NumGRID is dealt with the solution to this problem.

Now, a brief description of the Globus Toolkit (as a widely accepted grid package) will be given. In the sequel, there will be shown limitations of the Globus Toolkit and why it cannot be successfully used in the grid construction from clusters.

About the Globus Toolkit. The open source Globus Toolkit is a fundamental enabling technology for the "Grid", letting people share computing power, databases, and other tools securely online across corporate, institutional, and geographic boundaries without sacrificing a local autonomy. The toolkit includes software services and libraries for resource monitoring, discovery, and management, plus security and file management. In addition to being a central part of science and engineering projects that total nearly a half-billion dollars internationally, the Globus Toolkit is a substrate on which leading IT companies are building significant commercial Grid products. The toolkit includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability. It is packaged as a set of components that can be used either independently or together to develop applications. Every organization has unique modes of operation, and collaboration between multiple organizations is hindered by incompatibility of resources such as data archives, computers, and networks. The Globus Toolkit was conceived to remove obstacles that prevent seamless collaboration. Its core services, interfaces and protocols allow users to access remote resources as if they were located within their own machine room while simultaneously preserving local control over who can use resources and when. For more information view http://www-unix.globus.org/toolkit/about.html.

Globus components. The Globus Project provides software tools that make it easier to build computational grids and grid-based applications. These tools are collectively called the Globus Toolkit (tm).

The Globus Toolkit. The GRAM implements a resource management protocol, the MDS implements an information services protocol, and the GridFTP implements a data transfer protocol. They all use the GSI security protocol at the connection layer. These technologies are designed to be modular, but complementary. You can download the components separately, or all together. Additionally, you can obtain the client software independent of the server software if you like. The software is now packaged using the Grid Packaging Toolkit developed in collaboration with the NCSA. For more information view http://www.globus.org/gt2.4/admin/guide-overview.html.

Security component. The Globus Toolkit uses the Grid Security Infrastructure (GSI) for enabling secure authentication and communication over an open network. GSI provides a number of useful services for Grids, including mutual authentication and single sign-on.

To set up a new GRID, we need to get the authority keys from the Globus alliance. Another way to set up your own GRID is to download a special Globus Simple CA Package. The Globus Simple CA package provides a convenient method of setting up a certificate authority (CA) that is interoperable with the Globus Toolkit. It is intended for the users of small, test-grid environments, or users that are not part of a larger grid. For more information view http://www.globus.org/security/simple-ca.html.

A GRAM component. The main component of the Globus Toolkit that we will use is the GRAM. The Globus Toolkit includes a set of service components collectively referred to as the Globus Resource Allocation Manager (GRAM). The GRAM simplifies the use of remote systems by providing a single standard interface for requesting and using remote system resources for the execution of "jobs". The most common use (and the best supported use) of the GRAM is remote job submission and control. This is typically used to support distributed computing applications. For most Grid-based projects, we recommend using the GRAM as a project-wide standard for remote job submission and resource management. The GRAM is designed to provide a single common protocol and API for requesting and using remote system resources, by providing a uniform, flexible interface to, local job scheduling systems. The Grid Security Infrastructure (GSI) provides mutual authentication of both users and remote resources using the GSI (Grid-wide) PKI-based identities. The GRAM provides a simple authorization mechanism based on the GSI identities and a mechanism to map the GSI identities to local the user's accounts. For more information view http://www-unix.globus.org/developer/resource-management.html.

Installation. The main description of Globus Toolkit 2.4 can be found in [6, 7]. To install the Globus Toolkit follow the instructions above. Here will be described only the major steps of installation and some important tips.

Note: *It seems that Globus Toolkit 2.4 have some bugs and there are some difficulties during the installation, so here will be described one of correct ways, with important comments sometimes.*

An example of installation (use the root account):

1. Create a new user "`globus`";

2. Create two new directories for the Globus and the GPT. It will be better to create them in `/usr/local/`. For example:

```
/usr/local/"globus_install_dir"
/usr/local/"gpt_install_dir"
```

3. Change the owner of this directories to `globus`:

```
% chown globus /usr/local/"globus_install_dir"
% chown globus /usr/local/"gpt_install_dir"
```

4. Set environment variables for all users that plan to use Globus Toolkit (at list for `globus` user, and one more client user). In user's configuration file `home/.bashrc` add three strings

```
export GLOBUS_LOCATION="globus_install_dir"
export GPT_LOCATION="gpt_install_dir"
source "globus_install_dir"/etc/globus-user-env.sh
```

(this file doesn't exists now but it will be created later).

5. Download all necessary packages from http://www-unix.globus.org/toolkit/. At least you will need GPT (Grid packaging tool) and Resource Management bundle (SDK, client, server).

Important note: *If you plan to use MPI-G2 you need to download and install source bundles of Resource Management (SDK,client). This is because MPI-G2 uses source codes of the Globus for configuring and compiling.*

6. Install GPT (Grid packaging tool):

```
% gzip -dc gpt-3.0.1-src.tar.gz | tar xf -
% cd gpt-3.0.1
% ./build_gpt
```

7. Install Globus components from sources

```
% $GPT_LOCATION/sbin/gpt-build "bundle" "flavors"
```

or from binaries

```
% $GPT_LOCATION/sbin/gpt-install "bundle"
```

(repeat for every bundle).

Note: `-std-flavors` *compile bundles with all standard flavors. Or you can choose any flavor, for example gcc32, gcc32dbg.*

8. After installation of all bundles run

```
% $GPT_LOCATION/sbin/gpt-postinstall
```

Note: *Before running any client tools you need to run the source*

```
$GLOBUS_LOCATION/etc/globus-user-env.sh
```

or to login again, because we have put this string in `.bashrc` *file.*

9. Security setup. *Note: If you are planning to get certificates from Globus CA you must run* `$GLOBUS_LOCATION/setup/globus/setup-gsi` *and go directly to 11.*

10. Installing your own Globus Simple CA package. *Note: Do not run* `$GLOBUS_LOCATION/setup/globus/setup-gsi` *at this time* (*as you can read in the manual*) *or you can have problems with security later.*

Download Globus Simple CA package from http://www.globus.org/gt2.4/download.html. Install it

```
$GPT_LOCATION/sbin/gpt-build
globus_simple_ca_bundle-latest.tar.gz "flavor type"
$GLOBUS_LOCATION/sbin/gpt-postinstallsbin/gpt-postinstall
```

Set Globus Simple CA package as the default security package with

```
$GLOBUS_LOCATION/bin/grid-default-ca
```

Run now

```
% $GLOBUS_LOCATION/setup/globus_simple_ca"code"/setup-gsi
```

where `"code"` is a code of your CA.

11. Verify the installation

```
% $GPT_LOCATION/sbin/gpt-verify
```

If you install your own Globus Simple CA package your will probably get the warning that your standard Globus CA is not set correct. Never mind, it is ok.

12. Getting certificates:

User Certificate. To make a request for user certificate, login as user and run

```
$GLOBUS_LOCATION/bin/grid-cert-request
```

Host Certificate. To make a request for host certificate, login as root and run

```
% grid-cert-request -service host -host "your-hostname"
```

13. Signing certificates, run as root

```
$GLOBUS_LOCATION/bin/grid-ca-sign
-in $HOME/.globus/usercert_req.pem
-out $HOME/.globus/usercert.pem
```

14. Running Applications.

After the installation, it is possible to use client commands to start and manage jobs. For example, a globus-job-run command allows a user to start new jobs on the grid.

Examples:

```
globus-job-run remote.host.ru -stage script
```

This command will start the "script" module on "remote.host.ru" host with the Globus installed on it. Additionally before starting, the file "script" will be copied from the client's PC to "remote.host.ru" host (-stage option)

```
globus-job-run choeur/jobmanager-pbs -d /home/nikmal/tmp
-np 3 -stage a.out
```

This command will run file a.out on choeur server, using job-manager for pbs queue system with a home directory /home/nikmal/tmp. The program will be started on three (`-np 3`) nodes.

```
mpirun -globusrsl my.rsl
```

This command will use the Globus Toolkit+MPICH-G2 to start a user program. The whole configuration info will be read from `my.rsl` text file.

3. Limitations

The Globus Toolkit and the MPICH-G2 have some important limitations. They do not work behind firewalls because the security mechanisms, employed by the Globus require knowledge of the actual IP address of the host that is being connected to.

4. NumGRID project

The primary goal of the project is to create grids based on several multicomputers for modelling of the large-scale numerical problems such as problem of simulation of the proto-planetary disk evolution [8]. Such problems are characterized by a lot of regular data (a numerical mesh, particles), the long-time calculation, nonlinear dynamics of calculations etc. Summarizing the properties of the class of numerical modeling problems, the requirements for the NumGRID architecture and system software are formulated as follows:

- NumGRID is a union of multicomputers. Each node of multicomputers is an SMP-system. Thus, supercomputers are integrated for the superproblem solution.

- NumGRID is programmed as a single multicomputer. A distributed structure of a grid should not be reflected in the application program code.
- Modifications of the already existing application parallel programs should be minimized.
- There should be an opportunity to execute programs written in C, C++ and Fortran 77, 90
- It is necessary to use the MPI library for interprocess communications.
- It is necessary to provide simplicity of starting and management of tasks. Start and management should look like standard "mpirun" call.
- It is necessary to provide safety and reliability of distributed computations.
- There is no necessity to provide such a property of a grid as addition and removal of computing nodes in the course of computations.

A computer system that satisfies all of these requirements is called the Numerical Grid (NumGRID).

Keeping in mind the above requirements, many grid projects were reviewed. Despite of a variety of grid projects (several hundreds worldwide) and high popularity of grid technologies, none of the projects has been found to satisfy all the above requirements.

The most suitable software project is the Globus Toolkit with MPICH-G2. The basic advantages of Globus Toolkit are:

- An open source code of the project
- Development of new versions and their support.
- One of the main developers is the Argone National laboratory, which takes the major participation in the MPI standard formation and also, is involved in its non-commercial implementation MPICH.
- It is a well-known package for creation of Grid-networks in the world which is de facto standard.
- There are powerful tools for running jobs, management and security in the Globus Toolkit.

Nevertheless, testing the Globus Toolkit software has revealed substantial restrictions, which do not allow the construction of a necessary NumGRID. The main restriction is in the necessity to assign the global IP-address to each node and accessibility of each node of the grid. The usage of the NAT (Network address translation) or similar methods does not help to solve this problem.

Therefore, a concept of the grid creation satisfying the NumGRID specifications has been developed. The concept is based on the analysis of the project specification and studying solutions to similar programs as well as discussions.

The main idea is the following. On each master-node of all clusters, a special program is loaded. It is called the MPI-proxy. This program is actually a proxy-server for the MPI-packets routing. With the help of this router there is an opportunity to receive and to send packets to the internal (private) nodes of clusters. This router transfers the MPI packets from the internal nodes of a cluster (Cluster 1) to the router installed on the master host of the other cluster (Cluster 2). Later, the second router will transfer the received MPI messages to its internal nodes (inside cluster 2).

Another part of the NumGRID software is a special MPI library with an opportunity to send messages not only within a local network, but also through an MPI-proxy server to global networks. On each cluster or on the SMP system any version of the MPI library is always installed. Usually, this version is installed by the hardware vendor (a cluster, an SMP) and is most effective. Therefore it is desirable that application programs use those vendor-MPI libraries, which have been initially already installed. Implementing this idea, the developed MPI library actually represents a "wrapper" for standard MPI functions. That is, inside a component of a grid (clusters and the SMP), all messages are transferred using a pre-established, optimized vendor-MPI library, and all messages "outside" – using the developed library via the MPI-proxy.

It was planned to integrate the current project with the Globus toolkit. In this case, the division of the roles would look as follows. All calculations and transfer of the data would be carried out using the NumGRID system, and all the basic mechanisms, management and safety would be realized using the Globus toolkit. Also, in the sequel it is planned to widen the integration potentialities of the system.

As the project is planned to realize within the framework of the European grid program (http://www.cordis.lu/ist/grids/links.htm), the primary attention was given to the European projects. Recently the confirmation has been received that no similar project exists in the European Community or, at least, they are not declared. The nearest project to the NumGRID from European is PACX-MPI, but it has also a problem of the private IP addresses and impossibility of access to all the nodes.

5. State-of-the-arts of the project

By now an experimental prototype of the NumGRID system has been already developed. For testing the system, there was chosen a program written in C language with the MPI. The main objective of this program is to shift

data across the ring of computers. All the tests were done using L3M cluster and the bi-processor server of the same laboratory (L3M, Institut Meditarraneen de Technologie). To start the test on the created prototype of a grid it was necessary to make the following:

- To change one line in the test program (to replace #include "mpi.h" by #include "grid_mpi.h")
- To link the test program with a special library "grid_mpi" instead of a standard MPICH
- To start a proxy-server and the program at all the nodes.

Now all the operations are manually done, because this is only a test prototype. Further the start procedure will be simplified as much as possible and ideally aspire to procedure of start on a usual cluster. The experiments have shown the serviceability of the proposed method of the grid creation. Falling the productivity is insignificant and does not exceed 15%. Also, the system has not been optimized yet. The first version of the software is due to arrive by the end of 2004. The system will be installed on IMC&MG's 96-processor MVS-1000 and 32-processor cluster T-Forge based on the Opteron.

References

[1] SETI project. – http://setiathome.berkeley.edu/.

[2] Selikhov A., Germain C. CMDE: a channel memory based dynamic environment for fault-tolerant message passing based on MPICH-V architecture // Proc. 7th Int. Conf on Parallel Computing Technologies (PaCT-2003), LNCS, Vol. 2763, Springer-Verlag, Berlin Heidelberg (2003528–537).

[3] The Globus Alliance. – http://www.globus.org/.

[4] MPICH-G2. – http://www.hpclab.niu.edu/mpi/.

[5] PACX-MPI. – http://www.hlrs.de/organization/pds/projects/pacx-mpi/.

[6] Globus Toolkit TM 2.4 Installation Instructions. – http://www.globus.org/gt2.4/install.html.

[7] Globus 2.4 Admin Guide. – http://www.globus.org/gt2.4/admin/.

[8] Kraeva M.A., Malyshkin V.E. Assembly technology for parallel realization of numerical models on MIMD-multicomputers // Special issue of the International Journal on Future Generation Computer Systems devoted to Parallel Computing Technologies. – Elsevier Science, 2001. – Vol. 17, No. 6. – P. 755–765.

[9] Kuksheva E.A., Malyshkin V.E., Nikitin S.A., Snytnikov A.V., Snytnikov V.N., Vshivkov V.A. Numerical Simulation of Self-Organisation in Gravitationally Unstable media on Supercomputers // LNCS. – Springer Verlag, 2003. – Vol. 2763. – P. 354–383.

[10] Sun Grid Engine. – http://wwws.sun.com/software/gridware/sge.html.

[11] Grid based systems. – http://www.cordis.lu/ist/grids/links.htm.

[12] JINR. – http://grid.jinr.ru/projects.asp.

[13] Grid Computing Info Centre. – http://www.gridcomputing.com.