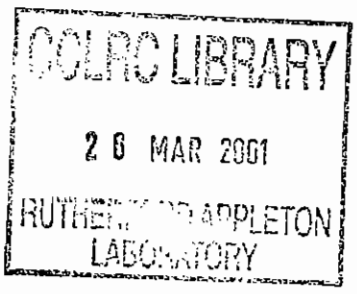
Technical Report
RAL-TR-2001-019

Experiences with Globus and MPICH-G

R F Fowler and C Greenough



22nd March 2001

COUNCIL FOR THE CENTRAL LABORATORY OF THE RESEARCH COUNCILS

The Central Laboratory of the Research Councils
Library and Information Services
Rutherford Appleton Laboratory
Chilton
Didcot
Oxfordshire
OX11 0QX
Tel: 01235 445384 Fax: 01235 446403
E-mail library@rl.ac.uk

ISSN 1358-6254

Experiences with Globus and MPICH-G

R.F. Fowler and C. Greenough

Computational Science and Engineering Department
CLRC Rutherford Appleton Laboratory,
Chilton, Didcot OX11 0QX, UK

Email: r.f.fowler@rl.ac.uk or c.greenough@rl.ac.uk

This report is available from http://www.cse.clrc.ac.uk/Activity/HPCI

November 10, 2000

Abstract

The Globus project is one of several efforts currently underway to provide easy and reliable access to the computational power of networked computers and efficient management of the associated data requirements of such systems. This report discusses our experiences in installing the Globus toolkit and in some basic use of the tools and the related MPICH-G toolkit.

Keywords: globus, grid based computing, MPI

Contents

1 Introduction

A great deal of research is currently underway into ways in which distributed computing resources can be utilised most effectively. To make the best use of all the computational power and data storage facilities within existing and future networks requires software developments in many areas such as:

- Secure access to remote machines
- Efficient scheduling of work across available resources
- Easy transport of data, executables and source code between machines
- Use of heterogeneous communications networks

Many projects are seeking to address these problems and provide a general framework to allow easy access to grid resources. These include UNICORE [1], Condor [2], Globus [3] and many others.

This report covers our initial experiences with installation and use of the Globus toolkit on two systems within CLRC. A large amount of effort is been devoted to this system by the developers in the USA. There is also considerable interest in it from European organisations such as CERN, who have vast amounts of experimental data to distribute and which requires a great deal of computational analysis. In the following sections we give a brief overview of our experiences in installing the Globus software on two machines at CLRC and on running some test examples including use of the Globus aware version of MPI, MPICH-G.

2 Installing the Globus Toolkit

2.1 Globus 1.1.1

The Globus software system is currently in a state of fast development. One illustration of this rapid change is the fact that in just over three months the systems has moved from version 1.1.1 to version 1.1.3, which is the current release at the time of writing.

One unfortunate feature of these releases is that they are offered only as complete systems - it is not an option to install a few patches and recompiling just those files, the whole system has to be recompiled from scratch. This can be a very time consuming task, particularly on older computer systems. For example a complete build on a Sun SPARC-10 system takes several hours. In addition, if any error occurs in the default build, all the work files are deleted by default, making it necessary to start from scratch again.

While systems such as the SPARC-10 are not likely to provide significant compute power within the Grid, they can be useful as platforms from which to launch jobs to other systems. A set to binary distributions of the grid components would greatly ease the installation process. However, a significant amount of work would be needed to support the wide range of computer systems on which Globus can run. Manchester University has recently announced an alpha release of a binary packaging of the latest Globus release for RedHat Linux systems. The Globus development team have also indicated that they are considering providing binary distributions in the future at least for some architectures.

Apart from the Globus software itself, it is also necessary to install packages for OpenLDAP and SSLeay which provide interfaces to the Light Weight Directory Access Protocol and Secure

Socket Layers. Special care has to be taken to get the correct versions of these libraries and to build them with the correct patches and options for Globus. The Globus team have patched the OpenLDAP 1.2.7 release used with Globus and this has changed between versions 1.1.1 and 1.1.3, so care is needed to get the latest copy from the Globus site. Again it would be nice if pre-compiled binaries of these libraries were available.

Another aspect of installing Globus Version 1.1.1 is the need to register systems and obtain security certificates from the controlling site in the USA. For 1.1.1 it was first necessary to register the organisation followed by the individual machines and finally the users. The data about machines within the grid was held on a few servers located in the USA, known as Meta Data Servers (MDS). In normal usage of the grid software, each basic operation, such as job submission would actually have to contact the MDS to perform operations such as mapping the hostname of a compute resource onto the full "contact string" which allows the job to be submitted to the remote host. The success of Globus meant that these servers became very heavily loaded with poor response time. On at least two occasions the servers were unavailable for several days due to system crashes preventing any normal access to the Globus software. In addition separate MDS servers were not kept up to date, so for example, machines at Manchester University were not visible to CLRC machines as they were not listed in our MDS.

A method was provided to set up local MDS servers that would allow a site to be independent of the connection to the USA and ensure that all relevant resources were listed. However, this required use of commercial software (Netscape LDAP server) and was an additional level of complication. The situation has changed significantly in the latest release of Globus, as outlined in the following section.

2.2 Globus 1.1.3

The 1.1.2 release of Globus was only available for a short period before being replaced with 1.1.3, which is the version we have now moved to. As mentioned above, each release currently requires a complete rebuild of the software from scratch which is time consuming. The initial release notes provided with 1.1.3 were very brief considering the significant changes that were made to the software at this release. A more detailed instruction guide has just been released, though it is still often necessary to check with the e-mail based discussion list to clarify some aspects of installation.

The major change in 1.1.3 is the removal of the dependency of the system on a single MDS site such as `mds.globus.org`. Instead each organisation is expected (but not required) to deploy one or more GIIS (Grid Index Information Server) which performs many of the original functions of the previous MDS sites. However it only contains data about resources that choose to register with it. In addition each computational resource runs its own GRIS (Grid Resource Information Server) which provides information about just that computational resource. When a request is made to a GIIS about available resources the GIIS may make a request to the appropriate GRIS to find the data unless it is already available in the local cache.

This change avoids the dependence on a single remote site which we found to be often slow to respond and unavailable for significant periods of time. The current 1.1.3 installation process consists of the following steps:

- Installation of the secure sockets package, SSLeay 0.9.0. For some architectures it is necessary to apply Globus specific patches. This is the same version used as with 1.1.1 and 1.1.2 releases of Globus.
- Installation of the Light Weight Directory Access Protocol package OpenLDAP 1.2.7.

Though this is the same version number as that used for earlier Globus releases some further changes have been mode, so a fresh version must be built.

- Registration of the site (organisation) name with the Globus authority. This is only need for the first installation of Globus at a site. This is done by email and may take two days.
- Installation of the Globus software. This involves three steps:
  - Building the software. This can take some hours on slower machines. By default threaded and non-threaded versions both with debugging enabled are built. Optimised versions require a separate build.
  - The setup step. This is now optional under 1.1.3. However, if an organisation wide GIIS is to be used then the setup step must be performed to define the host which will act as the GIIS.
  - The final deployment step copies the relevant executables to a non-NFS disk partition on the selected resource. A secure file system is needed to ensure the integrity of the system.
- After deployment it is necessary to request a certificate for the Globus gatekeeper. This again requires an email request to the Globus site and takes up to two days, though the 1.1.1 certificate can be reused in 1.1.3.
- As the Globus gatekeeper requires root privileges to map remote requests to named local users, several system files need to be updated to permit this.

2.3 Current Status

Globus 1.1.3 has been deployed on two hosts within our group. These are the four processor SPARC-10 machine (`hornbeam.inf.rl.ac.uk`) and the single processor Pentium workstation (`jericho.cis.rl.ac.uk`). Both machines have been set up with the default "fork" jobmanager.

A GIIS server has also been configured on jericho and both machines now register with this service. Two problems were encountered with setting up the GIIS:

- The documentation does not give a recommendation for the Unix port number that should be used by a GIIS. The GRIS on each machine defaults to port 2135 and it seems unlikely that this could also be used by the GIIS. Hence we followed the example of a site at Manchester University and used port 2136.
- By default the globus-setup command will use a Distinguished Name (DN) for the local resource based on the DNS name, with the actual hostname removed. However this gives two different DNs for these machines as one is in domain `inf.rl.ac.uk` and the other in `cis.rl.ac.uk`. This name difference seems to prevent the GRIS on hornbeam registering with the GIIS on jericho. This problem was overcome by changing the DN (in globus-setup) on jericho to be the same as that on hornbeam.

Globus jobs can be submitted between these two resources or to other Globus hosts which have added our user data to their local gridmap file. We have access to the ITD Beowulf cluster "wulfgar" on which Globus 1.1.3 has also been installed. This is a cluster of AMD Athlon processors running RedHat Linux. ITD have configured this machine to run jobmanagers for both standard fork jobs, which just run on the master host, and for job submission to the PBS batch system which allows access to all the slave processors. Globus has all been installed on two resources at Daresbury Laboratory by staff there. These include a power PC cluster using

the LoadLeveler jobmanager and another Beowulf cluster with PBS, though this is currently only running version 1.1.1.

3 Using the Globus 1.1.3 toolkit

The Globus toolkit provides a number of utilities to submit jobs, transfer files and query the state of remote systems. A few observations are given on some of these in the following sections.

3.1 Globus information services

By default, each Globus host should run a GRIS information server. These can be configured so that they register with one or more GIIS. These services store data in the LDAP (Lightweight Directory Access Protocol) format and can be queried remotely. In Globus 1.1.3, if a request to the GIIS asks for information about a host then the GIIS will query the appropriate GRIS and cache the results in case they is needed again in the near future. The 1.1.1 model was based on the resources constant sending data to the central MDS server.

At present there is no security on these services, so anyone can access them if they know the machine on which they run. However the data available presently is quite limited, and there seems little risk in making it publicly available.

One query that can be made to either a GRIS or a GIIS is to find the current load average of the host computer. This can be done with a command such as:

```
grid-info-search "objectclass=GlobusComputeResource" dn cpuload1 \\
                 cpuload5 cpuload15
```

This command queries the GIIS server which on our system might return:

```
hn=hornbeam.inf.rl.ac.uk, dc=inf, dc=rl, dc=ac, dc=uk, o=Grid
cpuload1=1.82
cpuload5=0.47
cpuload15=0.18
hn=jericho.cis.rl.ac.uk, dc=inf, dc=rl, dc=ac, dc=uk, o=Grid
cpuload1=3.02
cpuload5=1.12
cpuload15=0.52
```

This can provide a useful guide to system load on SMP machines, though it is of little help for systems such as Beowulf clusters where the load of the host machine is not necessarily any guide to the number of jobs running on the slave nodes. In these cases it would be more useful to query the number of jobs in the batch queues, though this data would be more complex to interpret as there may be several queues with different priorities. It appears that batch queue information is not automatically published via the GRIS/GIIS, though it is possible to enable this feature.

The current implementation of this information service is done mainly via shell scripts and this leads to performance problems on older computers. With a Pentium P75 machine as the GIIS server, the above query takes over one minute to complete and only slightly fast response is seen with the SPARC-10 machine. When the data is in the cache, the query is answered in under 10

seconds, but this is still very much longer than, for example, a remote shell command to find the load average.

3.2 File transfer facilities

As part of the move towards the data-grid, rather than just a computational grid, the Globus developers are enhancing the file transfer capabilities of the software. One aspect of this is the Globus enabled version of ftp that is now available and makes use of standard security components within the toolkit. The current version is actually based on the University of Washington's wuftp package. An extended ftp package is under development within Globus, which will replace this version eventually. Some simple tests have been made on the file transfer tools that are part of the standard Globus distribution and using gsi-ftp.

One of the easiest to use tools for file transfer is `globus-rcp` which is very similar to the well known remote copy `rcp` but making use of Globus security features. Thus the user must first get a valid proxy for the machine on which he wishes to use the command, and once he has this he can transfer to and from any Globus resource which has his name in the gridmap file.

This tool works as expected, though the observed speed of transfer between hosts was less than obtained with the standard (insecure) rcp command or the safer scp copy. Some timings are shown in Table 1 for fetching files from hornbeam (SPARC10) to wulfgar (AMD650). These times are the lowest of three measurements and subject to significant variations so the comparison should be treated with some caution. The SPARC10 has 10Mbit/s Ethernet connection which limits the peak transfer rate.

Command	Elapsed Time		
	1.1Mbytes	2.2Mbytes	4.9Mbytes
rcp	1.95	2.94	5.38
scp	5.48	8.59	16.24
globus-rcp	14.42	17.88	25.92
globus-url-copy	1.41	2.66	5.53
globus-url-copy (ns)	1.12	2.18	4.77

Table 1: Timings for file transfer between hornbeam (SPARC10) and wulfgar (Linux AMD 650). Times are in seconds and "(ns)" indicates the use of an insecure globus-gass-server.

The graph in Figure 1 shows the data transfer rates as a function of file size for the various methods. This shows that, at least with this particular combination of processors and network cards, that `globus-rcp` is not very efficient. Using a `globus-gass-server` with `globus-url-copy` is significantly faster than secure copy `scp`, though the later command does not require a server to be set up in advance.

We suspect that the main bottleneck in these transfers is the slow SPARC10 system, and that the results could be rather different between two faster machines. In such a case the network speed would become the limiting factor and the different transfer methods may have less performance spread.

The above results refer to two machines that are connected via a local area network in a single laboratory. Similar results are also obtained between two machines at separate sites connected across a Wide Area Network, though the differences are not quite so significant. Some results between wulfgar, at RAL, and tci18, a PowerPC machine at Daresbury Laboratory, are shown in Table 2.

In this case the insecure rcp command is slightly faster than the secure globus-url-copy. For

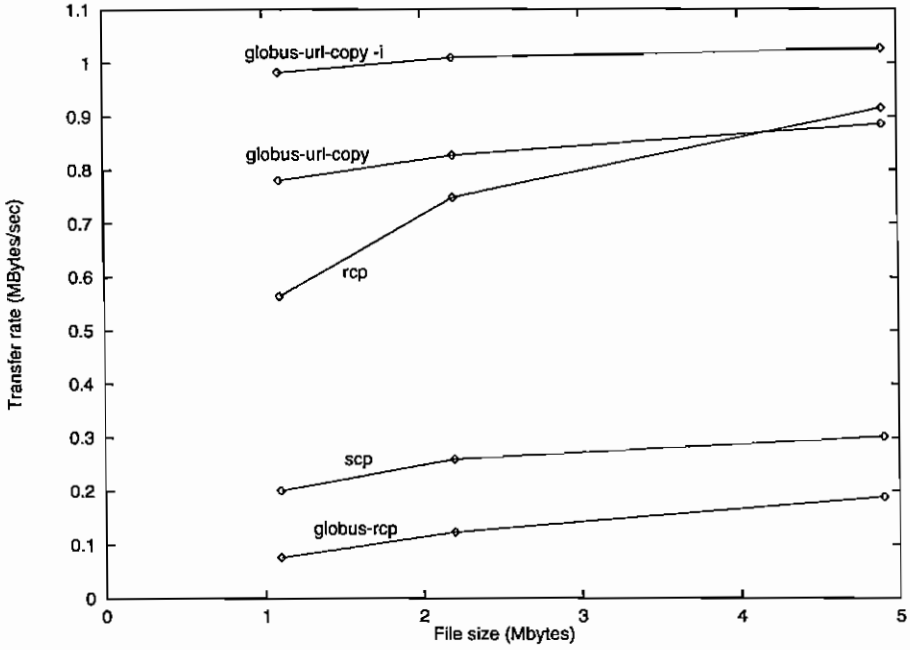


Figure 1: Data transfer rates between wulfgar and hornbeam using a selection of methods.

Command	Elapsed Time		
	1.1Mbytes	2.2Mbytes	4.9Mbytes
rcp	5	9	21
globus-rcp	15	20	28
globus-url-copy	6	11	24

Table 2: Timings for file transfer between wulfgar (Linux AMD 650, at RAL) and tci18 (PowerPC at DL). Varying network load means these results are more variable than those just over the local network. Times are in seconds, taking the lowest of three measurements.

large files even the globus-rcp command is not too inefficient compared with the other methods. The overheads associated with globus-rcp will of course vary with the processor speed of the two machines involved and in both the examples listed here one of the machines has been relatively slow by current standards.

During these tests the globus-url-copy command failed for one transfer with an unexplained segmentation fault. This error has also been seen before in use of this command and when it occurs it renders the globus-gass-server inoperable and a fresh one has to be started. This error does not occur often or in a repeatable manner, but it is slightly worrying.

The Globus aware version of the ftp protocol, gsi-ftp, has recently been installed with the server on jericho and the client software on wulfgar. With this software installed it is possible to transfer files without entering a password, as long as the user has a valid proxy. Some timings for file transfers between these two systems (measured on wulfgar, fetching files from jericho) are shown in Table 3. These were made using the gisncftpget command line utility.

Command	Elapsed Time		
	1.1Mbytes	2.2Mbytes	4.9Mbytes
rcp	1.44	2.45	4.96
globus-url-copy	2.18	4.15	8.93
gsi-ftp	2.98	4.14	6.47
gsi-ssh (scp)	5.04	9.15	19.02

Table 3: Timings for file transfer between wulfgar (Linux AMD 650, at RAL) and jericho (Linux P75). Times are in seconds, taking the lowest of three measurements.

These results show that the gsi-ftp gives a very similar data transfer rate to the rcp program (approximately 1Mbyte/s, which is the limit of the network card on jericho). The Globus security check adds an overhead of about 2 seconds to a file transfer compared to 0.4s using rcp and perhaps 0.2s with globus-url-copy. The slow processor on jericho is responsible for these large latencies. The fact that the globus-url-copy (the secure version) only transfers data at about half the speed of either gsi-ftp or rcp suggests that the data is not encrypted by the gsi-ftp program. This overhead would also be reduced when running on a faster machine.

Also shown in the above table are times using the Globus aware version of ssh, gsi-ssh (the scp command). The figures shown are very similar to those obtained using the standard version of this command. The data transfer rate is about 0.25Mbytes/s, lower than any of the other methods.

3.3 Job submission

Globus jobs can be dispatched using commands such as globus-job-run, with output to the terminal, or globus-job-submit, which spools job output for later viewing with the command globus-job-get-output.

The security overhead of submitting jobs means that even using the fork jobmanager there is some delay in running even simple commands. On a fast processor such as wulfgar the command

```
globus-job-run wulfgar.cis.rl.ac.uk /bin/pwd
```

takes almost 5 seconds to complete. On slower machines delays of up to 14 seconds are possible, much longer than using either rsh or ssh commands. Some typical figures are shown in Table 4.

Method	Time (secs)
globus-job-run	11.0
rsh	1.1
ssh	2.5

Table 4: Timings for running a trivial command (/bin/pwd) on hornbeam (SPARC10) from wulfgar (AMD650) using globus-job-run, rsh and ssh.

Obviously globus-job-run is not designed to run trivial commands in this way, but without fast response there is still a need for remote access via secure shell or similar. The Globus aware version of secure shell should address this requirement.

4 Globus enabled MPI: MPICH-G

Grid software such as the Globus toolkit enables one to access many different computational resources through a uniform interface and a single authentication mechanism. Another desirable feature of particular importance to users with large computational requirements is to be able to use several resources simultaneously. One way in which Globus seeks to enable this is by providing a Globus aware version of the public domain message passing toolkit MPICH [4].

The standard version of MPICH provides a full implementation of the MPI 1.0 standard with low level communication using an underlining library for data exchange between hosts. This can be the p4 communication package, shared memory or other perhaps vendor specific libraries. The Globus enabled version currently makes use of the Nexus communication layer [5], though there is some suggestion it may latter change to using the globus_io package instead.

The Nexus library permits messages to be passed between any two hosts that are able to at least support a TCP/IP connection. The software is also able to support more efficient and faster non-TCP/IP communication methods on certain machines such as the IBM SP2. Some benchmark figures on such a system are given at `http://www.globus.org/mpi/benchmark.html`.

4.1 Comparison of MPICH-G communication performance

A simple MPI ping-pong style communication benchmark has been run between some Globus hosts to measure the performance of MPICH-G. In Figure 2 we show the measured communication performance as a function of message size between two nodes of a SPARC-10 SMP. The results are compared with those using MPICH using the p4 communication library. In both cases the shared memory option has not been used, so data is passed via sockets.

For small messages (less than about 3Kbytes) the Globus MPI is roughly 10% slower than the p4 based version. For larger messages the MPICH-G overhead can be 30 to 50%. In both cases the default build of the MPICH library was used which includes some debugging features. The Globus software, including Nexus, was built without debugging.

Another set of results is shown in Figure 3 for the performance of MPICH-G on the Linux Beowulf cluster wulfgar. The slave nodes on this cluster are AMD 850MHz CPUs. The nodes are connected via fast Ethernet connections and also via a Myrinet high performance network. As far as we know, Nexus does not yet support Myrinet communications and hence is limited to using the Ethernet connection. Two other versions of MPI are available on the cluster, LAM-MPI, using normal Ethernet communication, and a version of MPICH adapted to use Myrinet.

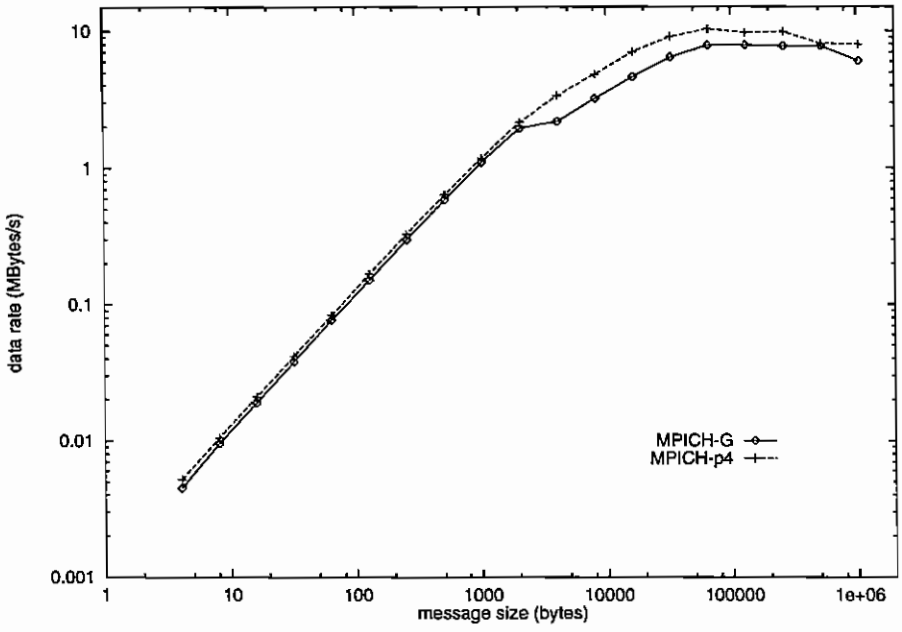


Figure 2: MPICH-G data transfer rate between processors as a function of message length on a Sun SPARC10 SMP machine (solid line). Also shown are the results obtained using MPICH with the p4 communication system (broken line).

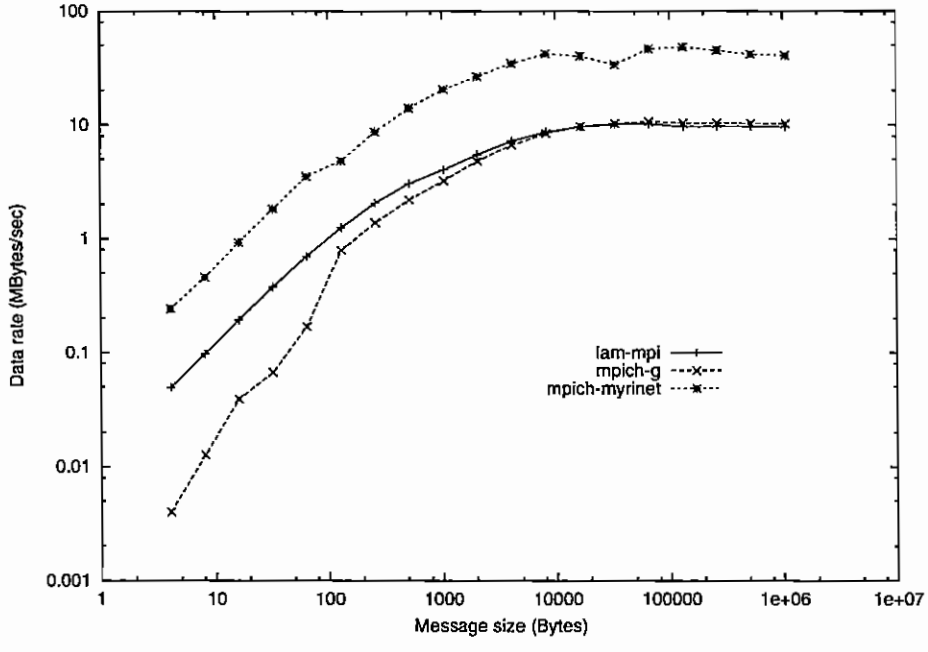


Figure 3: Data transfer rate between slave processors on wulfgar as a function of message length. Results for MPICH-G, LAM-MPI and MPICH-Myrinet are shown.

Results for all three versions of MPI are shown in Figure 3. The Myrinet version of MPICH gives significantly superior performance than both LAM-MPI and MPICH-G as is expected since the latter are limited to the Ethernet connection. MPICH-G slightly out performs LAM MPI for large message size, above about 50 Kbytes. For short messages LAM-MPI is faster by up to an order of magnitude.

These results show that if Globus is to be used to submit a job to a single resource, such as a Beowulf cluster, then it will often be better to use the native MPI for that system rather than the current MPICH-G implementation. Of course if one is trying to link two separate systems then the advantages of having a single MPI address space across the systems may be more important.

4.2 Running a CFD application with MPICH-G

As a simple test of MPICH-G and Globus for real applications a parallel computational fluid dynamics has been updated to use this software. The code is an explicit solver used for turbine blade performance prediction. Only exchange of boundary data is used in the solution process.

The code solves the flow equations on an unstructured tetrahedral mesh. This is first partitioned and the appropriate sections sent to each processor. Results are gathered to the master processor and then written to disk.

Tests to date have involved running the master process and partitioning on the SPARC10 machine hornbeam with the actual solution been performed on the slave nodes of the Beowulf cluster wulfgar. Since the computational speed of the Beowulf nodes is an order of magnitude faster than those of the SPARC10, there is little to be gained in using the SPARC10 in the solution phase.

MPICH-G can use a file called `machines` to specify which hosts should be used in the run. For hosts which have at least a correctly configured GRIS on the standard port, this file can just contain the fully qualified host name followed by the number of processes to run on that machine. Not all machines have a GRIS running, for example it was found that some IBM AIX machines, by default, do not allow sufficient user processes for the GRIS to answer queries. In these cases a work around is to use the full contact instead of just the hostname. An example machines file the use of both methods is:

```
jericho.cis.rl.ac.uk 1
hornbeam.inf.rl.ac.uk 4
"wulfgar.cis.rl.ac.uk:2119/jobmanager-pbs:/C=US/O=Globus/O=\
Central Laboratory of the Research Councils/OU=Rutherford\
 Appleton Laboratory/OU=Scientific Computing Services/CN=\
wulfgar.cis.rl.ac.uk" 2
```

Under MPICH-G the mpirun command can be used to directly submit the required executable or to generate the RSL (Resource Specification Language) which would be used to run the job. The options "`-globusargs dumprsl`" are used to list the RSL. This latter option is particularly useful if you wish to alter things such as the default directory for the remote jobs. It also makes it easy to have different executable names and even to set local environment variables if required. The modified RSL file can then be submitted using:

```
mpirun -globusrsl file.rsl
```

Using the above method it has been possible to run the CFD application with a single master process running on hornbeam and three computational slaves on the wulfgar nodes. The master

process is responsible for reading and partitioning the data between the slaves. It also gathers the results from them and saves these to disk. As mentioned above, it is not useful to include the SPARC processor in the computations, though it is capable of dealing with the pre and post processing stages.

In these tests we have used the PBS batch jobmanager on wulfgar and the standard fork jobmanager on hornbeam. This worked well, because the wulfgar batch system was unloaded at the time of the tests, so jobs could run immediately. If the batch system was busy, then the jobs would be delayed until PBS chose to run the queued job. This is not a great problem when the other jobs are to be run via fork. Jobs have also been successfully run across two separate batch systems, wulfgar (PBS) and tci18 (LoadLeveler), but again only when the batch systems were empty beforehand.

Though PBS batch jobs have been run with MPICH-G and PBS, the system is not yet properly configured. The problem is that if multiple processes are requested on the PBS resource in the "machines" file, then a batch job is run requesting the correct number of nodes, but the executables all run on just one of these processors. This can be avoided by instead listing the resource several times, each with a count of 1, in the machines file. While this works, it actually means many one processor jobs are sent to the batch queue and there is a danger that these will not all run at the same time. This problem has been raised on one of the globus e-mail lists, and the solution may require updates to the PBS system to make it explicitly aware of MPICH-G batch jobs.

Another problem with MPICH-G is that all nodes in the cluster must be able to make direct TCP/IP connections to all other nodes. Some Beowulf clusters are configured so that only the master node is available to external TCP/IP connections, while the slave nodes can only connect to other nodes in the cluster. There appears to be no way for the cluster to route communication through the master node.

5 Conclusions

Globus provides a wide range of tools to allow simple and secure access to remote computing resources. The software is still in its early stages and this means that significant changes are occurring quite frequently. It would be useful if these software could be packaged in a way that makes updating to new versions easier.

The performance of some Globus tools, such as globus-rcp, can be rather disappointing, particularly on older computing systems. The performance of the grid aware version of MPICH is reasonable, but unless the underlining Nexus layer has been compiled to take advantage of system specific communication hardware, vendor supplied versions of MPI will usually give better performance. MPICH-G does of course offer uniform MPI connectivity between separate grid aware systems, which the vendor MPIs cannot do.

References

[1] J. Almond, D.Snelling, "UNICORE: uniform access to supercomputing as an element of electronic commerce", FGCS Volume 15 (1999), Numbers 5-6, pp. 539-548, October 1999.

[2] M. Litzkow, M. Livny, and M. W. Mutka, "Condor - A Hunter of Idle Workstations", Proceedings of the 8th International Conference of Distributed Computing Systems, pp. 104-111, June, 1988.

[3] I. Foster, C. Kesselman, "The Globus Project: A Status Report", Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop, pg. 4-18, 1998.

[4] I. Foster, N. Karonis, "A Grid-Enabled MPI: Message Passing in Heterogeneous Distributed Computing Systems", Proc. '98 SC Conference, 1998.

[5] I. Foster, J. Geisler, C. Kesselman, S. Tuecke, "Managing Multiple Communication Methods in High-Performance Networked Computing Systems", J. Parallel and Distributed Computing, 40:35-48, 1997.