Grid Computing
Harnessing Unused CPU Cycles

by
Nicholas A. Senedzuk

A MASTERS PROJECT
Submitted in partial fulfillment of the requirements
For the degree of Master of Science in
The Department of Computer Science in
The Graduate Program of
Montclair State University

May 2007

MONTCLAIR STATE UNIVERSITY

Grid Computing
Harnessing Unused CPU Cycles

by

Nicholas A. Senedzuk

A Master's Project Submitted to the Faculty of

Montclair State University

In Partial Fulfillment of the Requirements

For the Degree of

Master of Science

May 2007

School College of Science and Mathematics
Department Computer Science

Acknowledgements

I would like to thank my advisor, Dr. Stefan Robila, for allowing me the opportunity to work on this project. He has been able to provide me support and guidance throughout the entire project. Without his help this project would have been not been possible.

I would also like to thank the graduate committee members: Drs. Benham, Antoniou, Jenq and Wang for their time and effort that they have provided me with.

I would like to give special thanks to Jim Ceravolo for allowing me to use the computer systems in the mediated classrooms. Without the use of these machines this project would not have been possible.

Special thanks are also in order for James Ragucci and David Puliti. Without their help my Masters would have been a lot harder. I am not sure how I would have made it through some of my classes without them.

Finally I would like to thank my parents, brother and fiancée, Christine. They have provided me with an overwhelming amount of encouragement and support throughout my Masters at Montclair State University.

Abstract

In this project, we present the deployment of a computing grid that allows for distributed processing across a parallel infrastructure. The implementation of the computing grid takes advantage of the Sun N1 Grid Engine product and unused CPU cycles. The Sun N1 Grid Engine is a highly flexible environment allowing the computing grid to be implemented on several different operating systems and on different hardware architectures such as Unix, Linux and Windows. Such flexibility permits a more efficient deployment of the grid than other currently available options and allows for a greater number of systems to be used in the implementation. The programs that are submitted to the grid are written in Java in order to leverage the languages platform independence. In this grid architecture, the same programs will be distributed to any system within the grid and the number of software and hardware dependencies of the systems are minimized.

The project involved the design and implementation of the computing grid as a way to harness unused CPU cycles from systems within the College of Science and Mathematics (CASM) at Montclair State University (MSU). Given the available resources, up to 16 systems were available for use in the grid at any given time. These included research dedicated Sun workstations and servers, as well as classroom-mediated systems running the Windows OS. Beyond the elegant deployment of the grid environment, two test applications (one performing prime number generation and one computing optimal spectral distance computation) were designed, implemented and tested. Experimental results show the chosen architecture to be easily manageable with the execution times reflecting a significant speedup and suggesting grids as an approach in solving computationally intensive problems otherwise not tractable in regular computing environments.

Contents

List of Figures

List of Tables

1. Introduction

The amount of data to be processed in the world is rapidly increasing today. While this is not a problem in itself, the processing of large data sets often leads to computational costs prohibitive to regular computer systems.

While distributed clusters or regular parallel computing is often found as a solution, many of the current configurations are still unable to reach the computational complexity level that is required for exhaustive search solutions. These parallel computing systems often come with a large price tag and may require special skill sets to maintain. This may prevent smaller research facilities such as some universities from being able to afford a parallel computing system. In an environment such as this, grid computing can become a viable alternative to costly systems [1].

Grid computing is an emerging computing model that distributes processing across a parallel infrastructure. A grid system enables the sharing, selection, and aggregation of resources distributed across "multiple" administrative domains based on their (resources) availability, capability, performance, cost, and users' quality-of-service requirements [2]. The throughput of a grid is increased by networking many heterogeneous resources across administrative domains; also know as boundaries, to model a virtual computer architecture [3]. The administrative boundaries are set by the implementor of the grid and can change if and when the grid needs to grow. These boundaries consist of any number of systems and locations such as a college campus or offices located throughout the world [4]. The systems that are part of the grid do not need to be dedicated to the grid. This allows for the cost of implementing a grid to be rather inexpensive or even free, as compared to distributed clusters or parallel machines, since they can consist of computer systems that are underutilized and have extra CPU cycles. Any system that is part of the grid is required to run a special piece of software known as middleware. This middleware allows the systems to communicate with each other and allow them to appear and function as on large computer system [4].

This project involves the design and implementation of a computing grid within the College of Science and Mathematics (CSAM) at Montclair State University (MSU). The computing grid was built using several computer systems within CSAM using the Sun N1 Grid Engine Software. The software is available from Sun Microsystems at no cost. The systems that are being used consist of underutilized research computer systems, mediated classroom computers systems and lab computers systems. These systems included the SPARC and x86 architectures running the Windows XP and Solaris Operating Systems. Using both OSs provides more flexibility when choosing systems to be part of the grid and allows for a greater number of systems to be part of the grid. The operation of systems within the mediated classrooms and computer labs continued to operate normally throughout the day as grid jobs were not dispatched to these systems during these hours. Jobs were dispatched to these systems during the night while these systems go unused.

In addition to the grid deployment, we have also implemented two applications that run on the grid environment. The first application being prime number generation and the second being an application to determine minimum distance components for vectors. Both of these applications are known to take a great deal of time to run when large data sets are used. By breaking the large data sets into smaller ones and distributing these smaller data sets to multiple systems running the application within the grid we will cut down on the amount of time that is needed to process one large data set.

2. Grid Design

2.1. Background on Grid Computing

The idea of sharing computer resources, now know as distributed computing, originated in the early days of computer systems. Back in the late 1950s and early 1960 researches realized that computer systems costing tens and even hundreds of thousands of dollars were not very efficient. Most computer systems spent much of their time idle waiting for human input. It was for this reason that researches believed that if the systems unused resources could be shared among multiple users the systems would be more efficient. Now 35 years later computers are significantly cheaper but they are still underutilized [5].

In 1989 Ian Foster arrived at Argonne National Laboratory. He had earned a doctorate in the field of Computer Science at New Zealand's University of Canterbury after switching his course of study away from Math and Chemistry before coming to Argonne. When he arrived at Argonne he began work on programming specialized languages for computing chemistry codes using parallel networks. In 1994 Foster refocused his research to distributed computing. Working along side Steven Tuecke, currently the lead software architect in Argonne's Distributed Systems Laboratory, and Carl Kesselman, currently the director of the Center for Grid Technologies at the University of Southern California's Information Sciences Institute, he begin the Globus Project, a software system for international scientific collaboration. They envisioned creating a common language and set of tools that would link sites together into a "virtual organization," with standardized methods to authenticate identities, authorize specific activities, and control data movement [5].

At a supercomputing conference is 1995 this concept was quickly put to use. Rick Stevens, who directs Argonne's math and computer-science division, and Thomas A. DeFanti, director of the University of Illinois–Chicago's Electronic Visualization Lab, headed a prototype project called I-WAY (Information Wide Area Year). I-WAY was an ATM network connecting supercomputers, mass storage systems, and advanced visualization devices at 17 different sites within North America [6]. Forster and his team created the software that tied the site together into a single virtual machine. This allowed uses to log in once, locate resources, reserve time, load application code and monitor the codes execution. The virtual machine was used by over 60 application groups for experiments in high-performance computing, collaborative design, and the coupling of remote supercomputers and databases into local environments [6]. It performed computationally complicated simulations such as colliding neutron stars and moving cloud patterns around the planet. This experience had inspired a great deal of enthusiasm and funding. The Globus Project was awarded $800,000 a year for 3 years by the U.S Defense Advanced Research Projects Agency. In 1997 the Globus Project released the first version of the Globus Toolkit, the software that ties the resources together. The code was developed using an open-source approach. Allowing the software to be distributed freely and the code to be read and modified [7]. It was shortly after this that the National Science Foundation, NASA and the Energy Department began grid projects with the Globus Toolkit as there underlying software. In 1998 Foster and his colleagues formed the Global Grid Forum. The forum is a group that meets 3 times a year to adopt basic language and infrastructure standards for grids [5].

Within the last few years the United States, Europe and Japan have seen an increase in the number of gird projects that have appeared. Most of these projects are collaborations among scientific researchers at national laboratories and universities on projects like climate modeling, high-energy physics, genetic research, earthquake simulations and brain research [7]. Even more recently projects such as "WISDOM" have begun to appear. WISDOM, which stands for World-wide In Silico Docking On Malaria, ran between 1 October 2006 and 31 January 2007. The project used computers from EGEE, Enabling Grids for E-sciencE, and provided scientists with the ability to analyze an average of 80,000 drug compounds each hour, in search for a drug that will combat malaria. This computer grid allowed for the equivalent of 420 years worth of work on a single PC to be completed in only 4 months [1].

2.2. Software

The initial step in the design of the computing grid was to determine what grid software product was to be used. Three of the possible choices were

- The Globus Toolkit by The Globus Alliance [8]
- Sun N1 Grid Engine by Sun Microsystems [9, 10]
- Condor by the University of Wisconsin-Madison [11]

The Globus Toolkit is an open source software toolkit used for building Grid systems and applications. The Globus Toolkit is developed by the Globus Alliance and many others all over the world [8].

The Sun N1 Grid Engine is an enterprise level piece of software used for building grid systems. The Sun N1 Grid Engine software is developed and supported by Sun Microsystems. It is a commercial version of the Sun Grid Engine software that is supported by Sun Microsystems and the Open Source Community through the Grid Engine Project [9, 10].

Condor is the software of the Condor Research Project at the University of Wisconsin-Madison. The Condor Project is a project to develop, implement, deploy, and evaluate mechanisms and policies that support High Throughput Computing (HTC) on large collections of distributively owned computing resources. Condor is developed by the University of Wisconsin-Madison Academic Staff [11].

After performing some research on these products the Sun N1 Grid Engine 6.0 Product, N1GE, was chosen. The main reasons for choosing the N1GE product over the other products were grid engine's scalability, cost and the proven track record of the software. Sun N1 Grid Engine natively runs on most UNIX Operating Systems without the need for 3rd party libraries or additional software packages. On the Windows OS, N1GE requires Microsoft Services for UNIX 3.5, which provides tools and libraries to integrate Windows with UNIX. Microsoft Services for UNIX (SFU) is available for no license fee and is supported by Microsoft. [12]. N1GE also allows for integration with the Globus Toolkit using 3rd party tools available from Sun Partners. N1GE is an enterprise class piece of software that is available for use without any cost. The only cost associated with the product is for support and training, both available from Sun, if required. Support can also be obtained for free via grid engine user forums on the Internet. The N1GE product has a proven track record in the technology industry and has been deployed at companies and universities such as Ford Motor Company, Motorola, Ohio Supercomputer Center and the University of Houston [13].

2.3. Requirements

The two main requirements for N1 Grid Engine are a supported operating system and a supported hardware platform. These requirements are only for N1GE itself and not the additional tools and software packages such as ARCo, the Accounting and Reporting Consol, and GEMM, the Grid Engine Management Module, that come with N1GE or Microsoft Services for UNIX which is required to install N1GE on a Windows host. These tools and software packages each have addition requirements. The N1 Grid Engine product is supported on several operating systems and several hardware platforms. The following are the supported operating systems [10]:

- Solaris 10, 9, 8, and 7 Operating Systems (SPARC Platform Edition)

- Solaris 10, 9, and 8 Operating Systems (x86 Platform Edition)
- Solaris 10 Operating System (x64 Platform Edition)
- Apple Macintosh OS/X 10.2.x or later
- Hewlett Packard HP-UX 11.x
- IBM AIX 4.3, 5.1
- Linux x86, kernel 2.4x, 2.6x, glibc >= 2.2[1]
- Linux AMD64 (Opteron), kernel 2.4x, 2.6x, glibc >= 2.2
- Silicon Graphics IRIX 6.5
- Microsoft Windows Server 2003[2, 3]
- Microsoft Windows XP Professional with Service Pack 1 or later[2, 3]
- Microsoft Windows 2000 Server with Service Pack 3 or later[2, 3]
- Microsoft Windows 2000 Professional with Service Pack 3 or later[2, 3]

[1] Includes Linux Red Hat release 4.0

[2] The following Microsoft operating systems are NOT supported:

- Windows 95
- Windows 98
- Windows Millennium Edition
- Windows XP Home Edition
- Windows NT Workstation
- Windows NT Server

[3] Requires Microsoft Windows Services for UNIX 3.5

The following are the supported hardware platform architectures:

- SPARC

- Intel and AMD x86
- AMD64
- PA-RISC
- PowerPC and POWER4

The support of such a large number of OSs and hardware architectures, allows for N1GE to run on almost any computer system, which provides for more scalability when incorporating systems into the grid. The systems that are to be incorporated into the grid must also meet minimum system requirements to have the N1GE software installed. The minimum requirements for N1 Grid Engine 6 are [14]:

- Master host (qmaster host)
  - 100 MB of free memory minimum
  - 500 MB of free disk space minimum
- Execution host
  - 20 MB of free memory minimum
  - 500 MB of free disk space minimum if the host is running a supported Microsoft Windows OS
  - 100 MB of free disk space minimum if the host is running a supported UNIX OS

The current recommended and supported version of Microsoft Services for UNIX, SFU, to use with N1GE is 3.5. The system requirements for SFU are as follows [6].

- Operating Systems Supported
  - Windows 2000
  - Windows 2000 Service Pack 3
  - Windows 2000 Service Pack 4

  - Windows Server 2003
  - Windows XP
- Amount of disk space required can vary depending what components are to be installed. *Please refer to Appendix D for required components and settings for SFU to function properly with N1GE.*
  - Maximum disk space required ~ 275MB
  - Minimum disk space required ~ 19MB

Before the installation of N1 Grid Engine can occur a grid engine user, *sgeadmin*, needs to be created locally on each host, the NFS shared files systems need to be mounted and the ports that N1GE communicates over need to be added to the services file. These operations require root access on a UNIX system and Administrator access on a Windows system. It is important that when the grid engine user is added to a UNIX system to ensure that the UID for the user is identical to the UID of the grid engine user on the qmaster host. If the UID is not identical the grid engine user will not be able to install N1GE correctly. On Windows this step is accomplished by mapping the Windows sgeadmin user with the UNIX sgeadmin user via the User Name Mapping Service within Windows. Along with the other tasks mentioned above there may need to be additional tasks completed depending on the environment and its configuration. Appendices C, E and F provide detailed documentation on what needs to be completed prior to installing N1GE and how to install N1GE on both UNIX and Windows systems.

It is important to note that N1GE is configured to have an execution, administrative or submit host installation be completed by the sgeadmin user and not by any other user including the root or Administrator user. Attempting to install N1GE as another user may cause errors and cause the installation to fail.

2.4. Limitations

Along with the requirements of the N1 Grid Engine there are limitations as of to how the software can be deployed and configured. In the following I discuss only the limitations relevant to this project. In addition one can encounter significant limitations with the relationship between N1GE and other versions of UNIX such as AIX, HP-UX and Linux. There may also be limitations with other tools and software packages such as ARCo and GEMM [15].

- Fixed IP Address

A systems IP address must not change for it to remain as part of the grid. Systems that have their ip address configured via DHCP need to have their hostname and ip address entered into the host file on both the local system and the qmaster host. N1GE cannot install unless it can resolve the system hostname on both the local system and the qmaster host. If a system that obtains its ip 0address from DHCP receives a new ip address for some reason the host file on both the local system and the qmaster host will need to be manually updated. *Please refer to Appendix E for details on how to manually add an ip address and hostname to the local systems host file.*

- Non Windows Qmaster Server

A Windows system cannot act as a qmaster host. Windows systems can only be an administrative host, submit host or execution host.

- Systems Must Stay Alive

All systems that are to be part of the grid as execution hosts must remain on during the hours that jobs are to be dispatched and executed on them. The only effect if a system is shut off before any jobs are dispatched to the system is that the gird will contain one less system. If a system is shut off while a job is being executed on the system the job will be lost. If the lost job was submitted to the grid with rerun option set to “yes” then the job will be resubmitted to the queue and rerun.

- *Need for Custom Installation Scripts for Windows Systems*

During the Windows install of Sun N1 Grid Engine there is no startup script put into place for the execution host daemon. The script must be manually put into place and linked to the correct run level. *Please refer to Appendix G for information on how the script is installed and linked to the correct run level.*

- *No Auto Installation for Windows*

A Windows execution host cannot be installed with the auto installation procedure. Windows execution hosts must be installed using the interactive method.

- *Additional Windows Configurations*

Username mapping, NFS mounts, and hostname resolving in SFU require special attention to successfully install the Grid Engine execution daemon, submit host functionality and the integration of Windows hosts into a N1 Grid Engine. If any of these three services are not functioning properly then N1GE cannot be install onto the Windows host. *Please refer to Appendix G for the scripts that configure these services.*

3. Grid Implementation

3.1. Grid Layout

The design for the deployed grid environment is summarized in Figure 3.1.

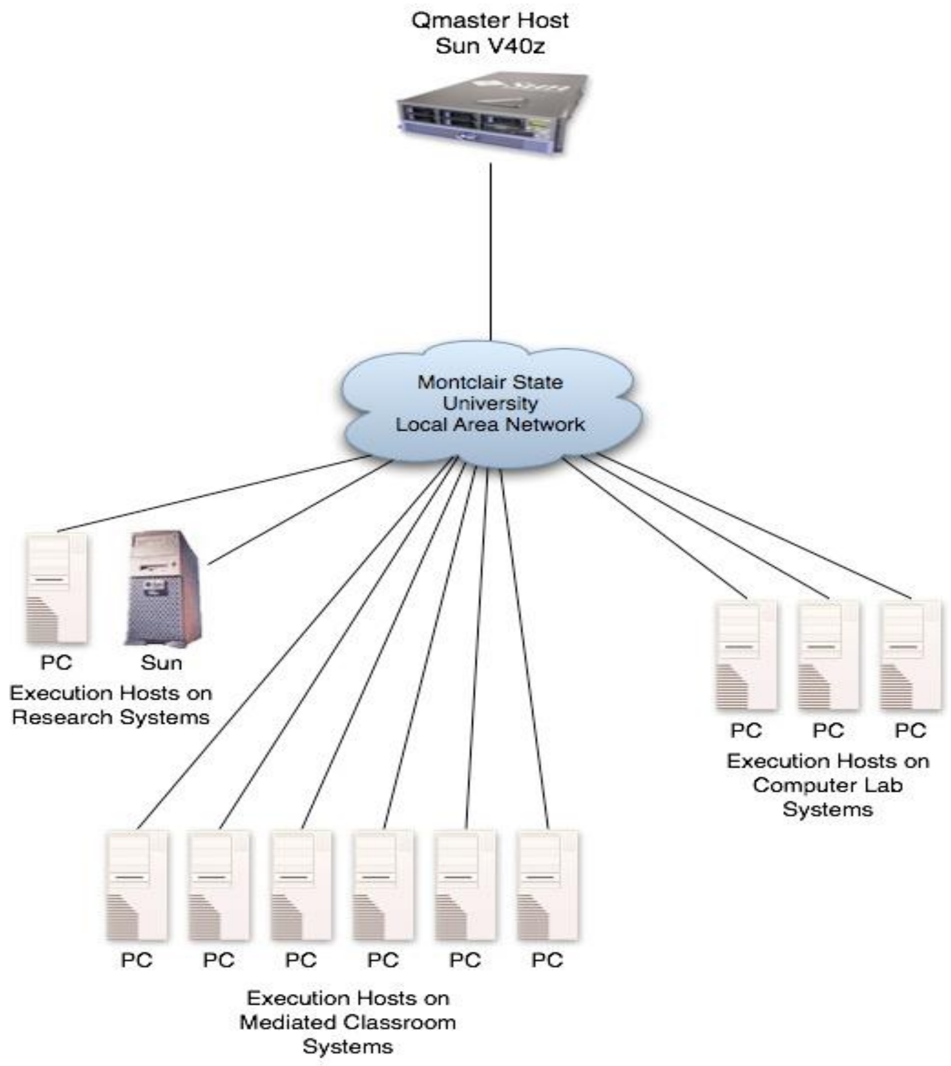

Qmaster Host
Sun V40z
Montclair State University Local Area Network
PC
Sun
Execution Hosts on Research Systems
PC
PC
PC
Execution Hosts on Computer Lab Systems
PC
PC
PC
PC
PC
PC
Execution Hosts on Mediated Classroom Systems

Figure 3.1 – Computing Grid at Montclair State University.

The master host server also known as the qmaster host resides on a Sun Fire V40z system running the Solaris 10 x64 Operating System. The master hosts configuration consists of a 4 CPU single-core AMD system with 16GB of RAM and 2 73GB hard drives. The 2 hard drives are set up in a mirrored configuration using the internal LSI Logic SCSI Controller. This allows for higher availability on the system in the even of a hard drive failure. The master host is connected to the University's Local Area Network, LAN, using an Ethernet connection. The (Q)master host is used to control the scheduling and dispatching of jobs on the grid making it the main administrative system on the grid. The master host also acts as a submit host allowing it to be used to submit jobs to the grid and has also been configured as an execution host allowing it to process jobs. Configuring the master host as an execution host is not recommended if the host is a very slow system host or if your cluster is significantly large [16].

The execution hosts consist of a variety of different systems located in mediated classrooms, computer labs and research areas. These systems include both x86 and SPARC based systems. These systems are running both the Windows XP and Solaris OS. The configuration of these systems such as RAM, hard drives and CPU speeds varies between systems. Theses systems are connected to the University's LAN using Ethernet connections. These systems' main purpose is to process jobs in the grid but they can also be used to submit jobs to the grid since execution hosts also need to be configured as submit hosts.

3.2. Software Installation

Both the Qmaster and execution host software must be installed. The grid engine software installation can be preformed by several different methods [16]:

- Interactive
- Interactive, with increased security
- Automated, using the *inst_sge* script and a configuration file
- Upgrade

For this grid deployment the master host and the execution hosts were installed interactively. This is the recommended method to install the master host since it is only installed once. As for the execution hosts this method was chosen since automated installs are not possible with Windows execution hosts. The increased security method was not used since it added a level of complexity to the installation and the time constraints of this project did not allow for the extra needed time to add the increased security options to the installs. Before the installation of the grid engine software can be completed the software packages must be installed onto the master host. The installation of the packages include the Solaris binaries along with the documentation and architecture independent files. These packages were installed on the master host using the Solaris *pkgadd* command. For the installation of the other supported operating systems the *tar* method must be used [17]. For the *tar* method the architecture dependent files are copied to the *SGE_ROOT* directory and then de-archived. Once this is completed for all needed architectures the installation of the software can begin.

The installation of the qmaster host is the first installation to be completed. The qmaster host is installed by executing the *install_qmaster* script from the *SGE_ROOT* directory. During the install you are asked several questions in regards to configuring the qmaster host. Once this installation is completed execution hosts can then be installed. While the installation methods are the same for both Windows and UNIX execution hosts an additional software package, MS Services for UNIX (SFU), needs to be installed on the Windows hosts before N1GE can be installed. SFU allows for Windows to be integrated into existing UNIX-based environments. Once SFU is installed and configured on the Windows hosts the installation of N1GE is similar to that of a UNIX host. For both operating systems the NFS file systems that are shared out from the qmaster host need to be mounted. On a UNIX host the NFS file systems are mounted using a "hard mount" which requires entries for the mounted file systems in the */etc/vfstab* file on the host. For a Windows system the NFS file systems are mounted using the "automounter" supplied with SFU. Although the automounter method is not the preferred method for mounting the NFS file systems it works better on the Windows hosts and is easier to configure. Once the NFS file systems are mounted an admin user for N1GE must be created locally on each system. Once the N1GE admin user has been created and the NFS file systems are mounted the installation of the execution host can be completed. The execution host is installed by executing the *install_execd* script or the *inst_sge* script with the *–x* option as the N1GE admin user. These scripts are located in the *SGE_ROOT* directory. *For detailed documentation and instructions on how to install N1GE and other required software please refer to Appendices A, B, C, D, E and F.*

3.3. Software Configuration

The basic cluster configuration is the collection of information that is configured to reflect site dependences and influence the grid engine system behavior. The site dependences include valid paths for programs that the grid engine may require access to such as *mailx,* the interactive message processing system, and *xterm,* the terminal emulator for X Windows. A global configuration is provided for the master host as well as for every other host in the grid. Along with the global configuration a local configuration can be configured on each host to override particular settings in the global configuration. It is important to note that not all global configuration settings can be overridden with a local configuration. During the installation of the master host you are prompted to set some of the global configuration settings such as the administrative e-mail address that should receive e-mails.

In the current environment only the global configuration was being configured and used and no local configurations were created. The global configuration that is being used contains only one modification; the value of the variable "reschedule_unknown" has been configured to a time of 00:05:00. This variable determines whether jobs on hosts in an unknown state are rescheduled and thus sent to other hosts. The value of 00:05:00 indicates that the jobs are to be rescheduled 5 minutes after the grid engine system has determined that the host the job is running on is in an unknown state. The "reschedule_unkown" variable works in conjunction with the "max_unheard" variable. The "max_unheard" variable determines how long the qmaster host will wait from the last time it contacted or was contacted by the execution daemon of a host before setting the status of all queues residing on a the particular host to a status of unknown. For a job to have the ability to be rescheduled it needs to be submitted with the *"-r"* option. This option identifies the ability of a job to be rerun. If the value of *-r* is *"yes"* the job will be rerun if the job was aborted without leaving a consistent exit state. This option can also be set after a job has already been submitted to the grid using the *qalter* command. *Please refer to Appendix B for the global configuration [18].*

3.4. Host Group Configuration

A host group is a group of hosts that can be treated collectively as identical. Host groups enable the use of a single name to refer to multiple hosts allowing for the ability to manage multiple hosts by the means of a single host group configuration. A host group can include other host groups as well as multiple individual hosts. It is at the discretion of the grid administrator on how to configure the host groups [19]. In this project the host groups have been configured as follows:

- *lab* - computer systems that reside in the computer labs
- *mediated_class* - computer systems that reside in the mediated class rooms
- *research* - computer systems that are used for research

3.5. Calendar Configuration

Queue calendars define the availability of queues according to the day of the year, the day of the week, or the time of day. Calendars allow queues to change their status at specified times of the day. The status of a queue can be changed to one of the following [19]:

- disabled – close the queue, but do not affect the running jobs
- enabled – open the queue
- suspended – stop execution of all jobs running in the queue and close the queue
- resumed (unsuspended) – unsuspend the queue, and then open it

Each queue can attach to a single calendar, thereby adopting the availability profile defined in the attached calendar. In this project two calendars have been defined as follows.

- calendar *"night"*

calendar_name	night
year	NONE
week	7-23=suspend

- calendar *"24-hour"*

calendar_name	24-hour
year	NONE
week	NONE

The *"night"* calendar will allow queues associated with it to be enabled from the hours of 11:00pm to 7:00am. Between the hours of 7:00am and 11:00pm the queues and all running jobs will be suspended. The *"24-hour"* calendar will allow queues associated with it to be enabled at all times.

3.6. Queue Configuration

Queues are containers for different categories of jobs. Queues provide the corresponding resources for concurrent execution of multiple jobs that belong to the same category. A queue can be associated with one or multiple execution hosts. Queues that span multiple hosts are known as cluster queues. A cluster queue allows for the configuration of a cluster of execution hosts via one cluster queue configuration file. Each host that is associated with a cluster queue will receive an instance of that cluster queue. This allows for systems to be part of more then one queue. A cluster queue can have any combination of the following host objects associated with it [19]:

- one execution host
- a list of separate execution hosts
- one or more host groups

It is important to note that when a system that contains more then one CPU is made part of a queue N1GE sees each CPU as a different system and will dispatch jobs to each of the CPUs on that system. This default behavior can be changed by changing the "slot" parameter in the queue configuration.

In this project each cluster queue has only host groups associated with them. Each cluster queue will have like systems associated with the queue. The cluster queues have been configured as follows.

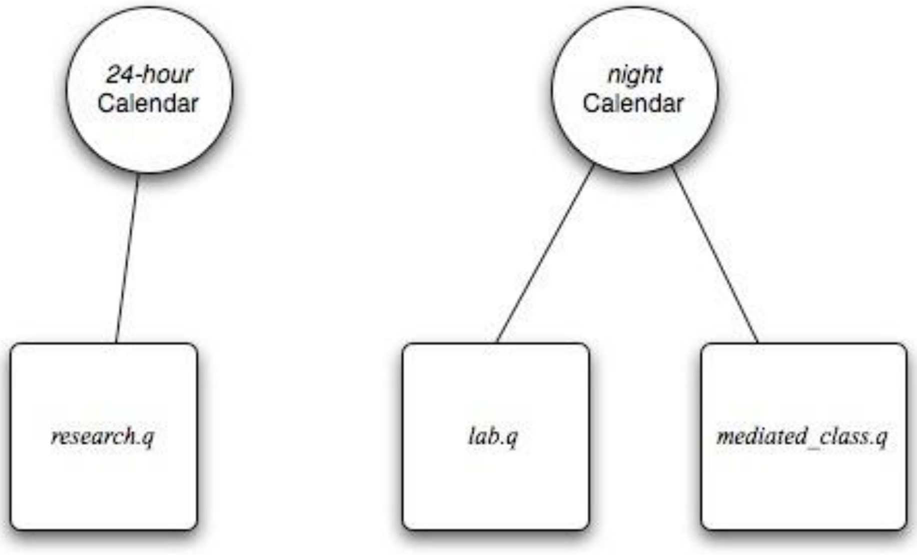


Figure 3.2 – Calendar and queue configuration.

• *lab.q*	systems that are part of the lab host group, attached to "*night*" calendar
• *mediated_class.q*	systems that are part of the mediated_class host group, attached to "night" calendar
• *research.q*	systems that are part of the research host group, attached to "24-hour" calendar

By configuring the queues in this manner it will allow the systems associated with the *lab.q* and *mediated_class.q* to only be accessed by the grid between the hours of 23:00 and 07:00, 7 days a week. Between 07:00 and 23:00 all work on these queues will be suspended and any running jobs will be stopped. This will allow for normal operation of the systems during the day. The systems that are associated with the research.q queue are not used on a regular basis and can be used throughout the day without affecting their normal operations. The *research.q* queue will allow all systems associated with it to be accessed 24 hours a day, 7 days a week. If needed a queue instance can be suspended if a grid job is interfering with operations of a system while it is being used for another purpose.

3.7. Scheduler Configuration

The scheduler is a process that runs on the qmaster that the grid engine system uses to determine which jobs are to be dispatched to which host and what order the jobs are to be dispatched in. The scheduler can schedule jobs based off of the following criteria:

- *the cluster's current load*
- *the jobs' relative importance*
- *the hosts' relative performance*
- *the jobs' resource requirements, for example, CPU, memory, and I/O bandwidth*

In this project the default scheduler configuration was used. At this point in time there was no need to make configuration changes to the default configuration. The default scheduling policy is a *first-in-first-out* policy meaning the first job submitted is the first job that is dispatched to a queue. It is also important to note that no priority policy has been configured. By default the grid engine system will use the *first-in-first-out* policy to assign priorities to submitted jobs [19]. Figure 3.2 summarizes how the queues and scheduler control the flow of jobs that are submitted to the grid.

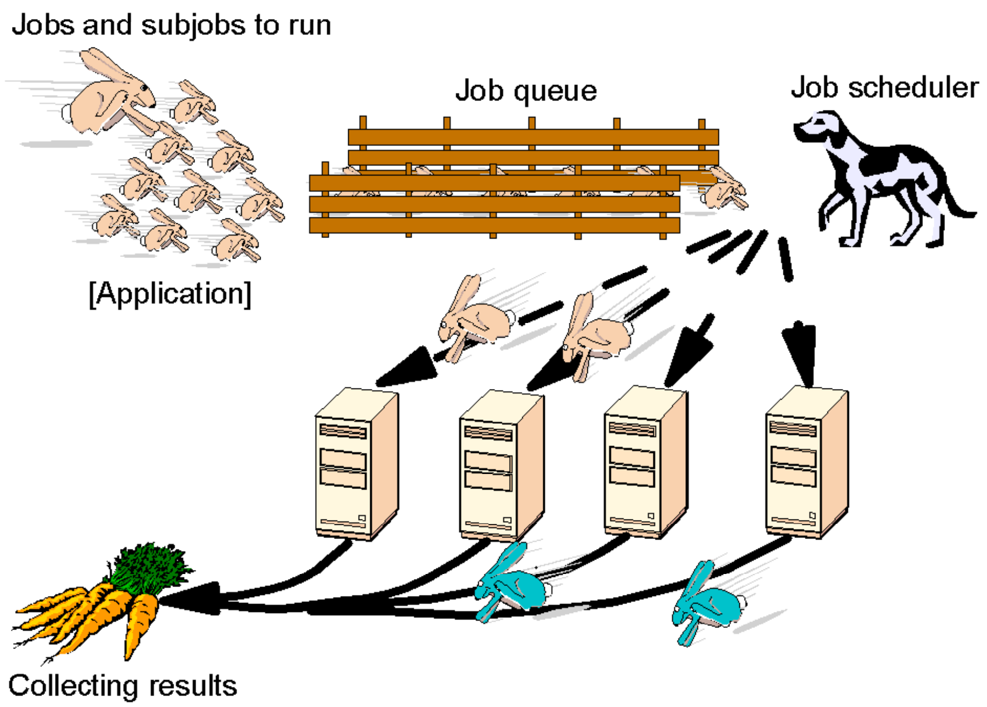


Figure 3.3 - An application is one or more jobs that are scheduled to run on machines in the grid. The results are collected and assembled to produce the answer [20].

3.8. Backup Configuration

After installing and configuring all the components of a computing grid it is important to make sure the configuration of the qmaster host is backed up. The backups can be used to restore the configuration of the qmaster in the event of data loss. It is important to make sure a copy of the backup resides on another host besides the qmaster incase all data on the qmaster host is lost. An automated backup facility is supplied with N1GE to perform a backup of the grid engine system configuration. For the environment that has been configured in this project a backup is run daily at 08:00. The backup is then copied over to the host pegasus.montclair.edu as a secondary backup copy in event of a catastrophic failure on the qmaster host. The backup and file copy are preformed via a script that is run via *cron. Cron* is the UNIX clock daemon that starts a process that executes commands at specified dates and times [21]. *Refer to Appendix H for detailed information on the configuration of backups [19].*

4. Grid Applications

Once the grid environment was deployed, we have designed and implemented two applications that would benefit from the increased computational power now available. While the chosen grid architecture allows for execution of any command line launchable application, in order to ensure flexibility for deployment we have decided to employ Java [22, 23, 24]. Through this, we have aimed to maintain any of the grid jobs as independent as possible. We note that however, some of the settings remain specific to the machines (such as building the path to the files). In this case, scripts were used to provide the correct parameters.

4.1. Prime Number Generation

A prime number is a natural number that has exactly two distinct natural number divisors, which are 1 and the prime number itself. Many algorithms have been developed, some faster then others, to find prime numbers but determining prime numbers can still be an exhaustive task. These algorithms include the trial division method, wheel factorization method and the Sieve of Eratosthenes method.

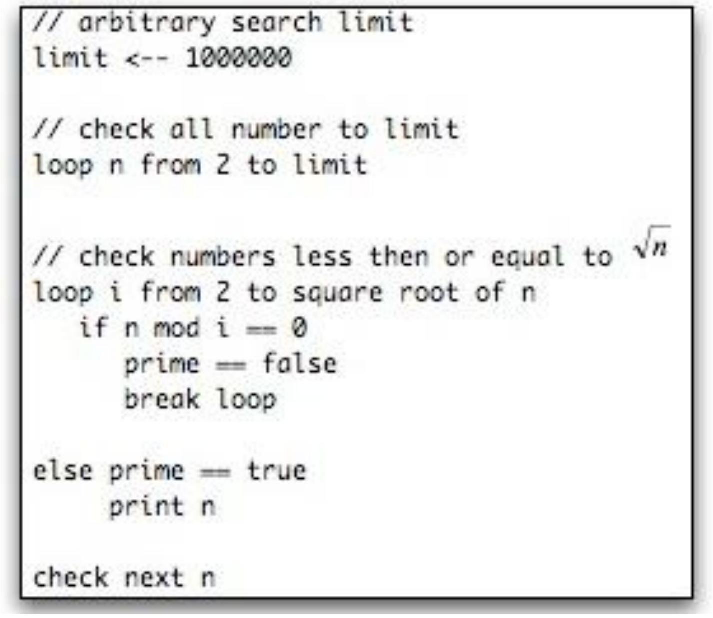
Prime numbers and their properties were first studied extensively by the ancient Greek mathematicians. The mathematicians were interested in the prime numbers for their mystical and numerological properties [25]. It was in 300 BC that the Greek Mathematician Euclid proved the theorem that there are an infinite number of prime numbers [26]. Around 200 BC the Greek Mathematician Eratosthenes devised a prime number generation algorithm for calculating prime numbers [26]. The algorithm known as the Sieve of Eratosthenes is one of the simplest and fastest algorithms know to find prime numbers [27]. The next important development was made at the beginning of the 17th Century but Fermat, French lawyer and government official most remembered for his work in number theory. His theory, which he had proved, is known as

Fermat's Little Theorem. The theorem stats that if p is a prime and a is a natural number, then for any integer a we have $a^p = a$ (mod p) [28]. Fermat also corresponded with other mathematicians and worked particularly with the monk Marin Mersenne. Fermat believed that the numbers $2^n + 1$ were always prime if n is a power of 2. He have verified his belief with $n = 1$, 2, 4, 8 and 16 and if n were not a power of 2 then the result failed. It was not until more then 100 years later that Euler showed that the case $2^{32} + 1 = 4294967297$ is divisible by 641 and so is not prime [26]. The number of the form 2^n -1 also attracted attention because it is easy to show that if unless n is prime these numbers must be composite. It is important to note that not all numbers with the form 2^n - 1 with n prime are prime. For example 211 - 1 = 2047 = 23 * 89 is composite, though this was first noted as late as 1536. Prime numbers of this form became know as Mersenne Prime Numbers since Mersenne often studied them. In 1952 the Mersenne prime numbers M_{521}, M_{607}, M_{1279}, M_{2203} and M_{2281} were proved to be prime numbers using an early computer and the electronic age had begun [26]. By September of 2006 a total of 44 Mersenne Prime Numbers have been found. The largest being $M_{32582657}$ which is 9,808,358 digits long[29].

One of the largest uses of prime numbers is in cryptography. Cryptography algorithms, such as the RSA algorithm for public-key encryption, are based on large prime numbers. The RSA algorithm is based on the idea that it is very simple to multiply numbers together but can be very difficult to factor numbers. The algorithm uses the product of 2 huge random prime numbers in the creation of the public and private keys. The 2 prime numbers, referenced as p and q, are kept in the private key and their product, referenced as N, is put into the public key. It is relatively simple to find N by multiplying p and q but is basically impossible to find p and q from N [30].

When attempting to find a large amount of prime numbers using a single computer system waiting for the results can be time consuming. If multiple computer systems were used to process the same numbers but process smaller intervals of numbers over several jobs waiting for the results can be less time consuming. Using the grid engine we will do just this. The prime number generation program will be run on multiple computer systems with intervals of numbers of a reasonable size. The prime number generation program will be run several different times with different intervals of numbers and with a different number of machines. Running the program with different intervals of numbers will allow us to see what effect decreasing the interval of numbers and increasing the number of jobs has on the speed of calculating the prime numbers. We will also see what effect running the program using a different number of machines will have on the speed of calculating the prime numbers.

The Java code that is being used to generate the prime numbers is using a variant of the division method algorithm. In this algorithm a number is determined to be a prime number by taking the modulo of all numbers that are less than or equal to the square root of the number. If at any point the module of the number equals 0 then the number is not a prime. The function that is performing the computation will then exit marking that number as not being a prime number and then move onto the next number. The number is prime if the module of the number does not equal 0 and all numbers up to one less then the number have been checked. The function that is performing the calculation will make this number as a prime number and then move onto the next number. The Java code for the application can be found in appendix I. We note that for every number *n* checked, the algorithm will have to perform up to $\sqrt{n}$ operations. While this is not a very efficient system (compared to the traditional sieve method) it works well within the distributed environment since no further communication is necessary between the Qmaster and the execution hosts until the end of the job.



```
// arbitrary search limit
limit <-- 1000000

// check all number to limit
loop n from 2 to limit


// check numbers less then or equal to $\sqrt{n}$
loop i from 2 to square root of n
   if n mod i == 0
      prime == false
      break loop

else prime == true
     print n

check next n
```

Figure 4.1 – Prime number algorithm.

When the code is compiled and executed it requires 2 parameters. The first parameter being the lower bound and the second parameter being the upper bound for the prime number generation. The program is submitted to the grid using the wrapper script *prime.sh*. The wrapper script specifics where Java binary is, what shell to use, that the job is re-runable if it was aborted without leaving a consistent exit state, sets up need path information and executes the program with the submitted parameters. The *prime.sh* script can be found in appendix I. The jobs are submitted to the grid with the following command line string, changing the interval for each job submitted.

```
"sgeadmin@panic:/export/home/sgeadmin/jobs $ qsub prime.sh 1 2500000"
```

The prime number generation application was run a total of 8 different times. The application was run with a beginning parameter of 1 and an ending parameter of 1000000000 for every run. The program was first run 4 different times on a total of 8 CPUs. The first run consisted of 50 jobs, second run consisted of 100 jobs, third run consisted of 200 jobs and the fourth run consisted of 400 jobs. The program was then run another 4 times with a different number of jobs being submitted each time on a total of 14 CPUs. Again the first run consisted of 50 jobs, second run consisted of 100 jobs, third run consisted of 200 jobs and the fourth run consisted of 400 jobs. These 8 application runs produced the following results.

Total Time of 14 CPUs in minutes	Total Time of 8 CPUs in minutes	# of jobs
34	54	50
27	41	100
29	43	200
30	46	400

Table 4.1 – Prime Number Execution Times.

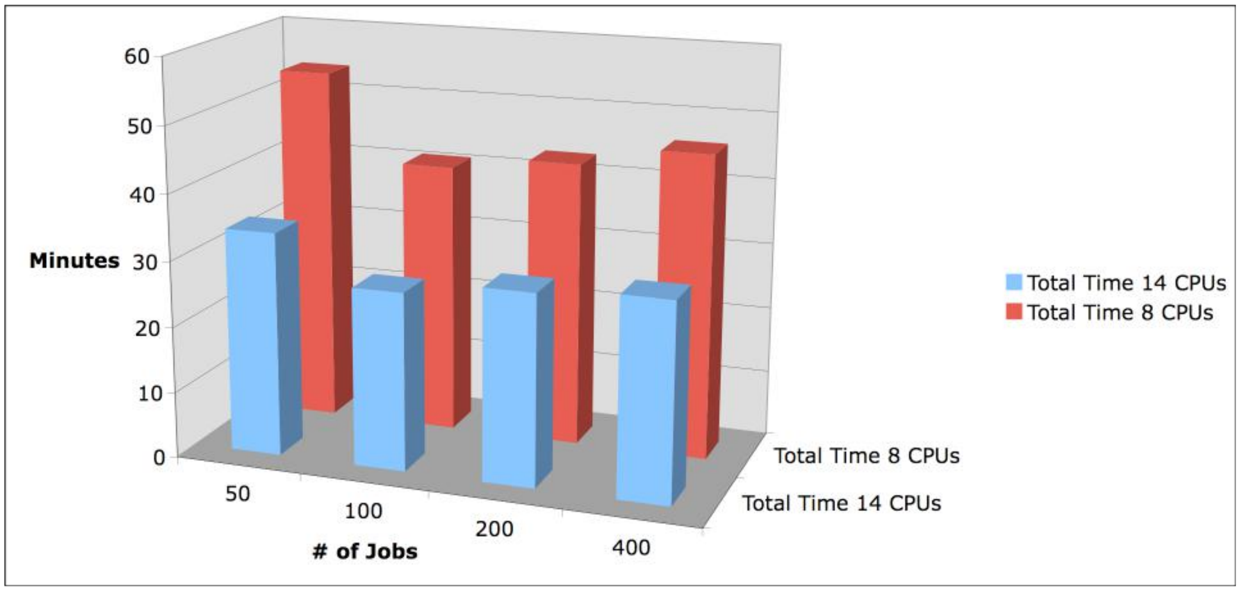


Figure 4.2 – Time comparison of number of jobs executed and number of CPUs.

These results show that by increasing the number of CPUs used to process the jobs there is a significant decrease in the amount of time it takes to process the data. The same does not always hold true when the number of jobs is increased. As we can see there is a decrease in processing time when the number of jobs increased from 50 to 100 when both 8 and 14 CPUs were used. For the next increase of jobs from 100 to 200 we do not see a decrease in processing time but see a slight increase in processing time. The same thing is seen when the number of jobs increase from 200 to 400. This again holds true for when both 8 and 14 CPUs are used. This increase in processing time is most likely due to an increase in overhead within the grid system. This shows that by increasing the number of jobs submitted to the grid for a single application that there may be an increase in processing time but this may not hold true for every instance. This indicates that for every application that is run on the grid that there is an optimal number of jobs that should be submitted. These results show for the 8 application runs that were preformed that the optimal number of jobs to be submitted to the grid is 100.

As a limitation of our application we note that as the jobs increase in the minimum and maximum number to be checked so do the number of operations for each number within the interval. As future development we suggest an investigation of a dynamic creation of the intervals with larger ones for small numbers and small ones for large numbers.

4.2. Determining Minimum Distance Components for Vectors

A vector is an ordered collection of n elements, which are called components, in a one-dimensional array. Given two *n*-dimensional vectors **x** and **y** with the components $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$, one can design measures that compute the 'distance' between **x** and **y** as a function on the vectors' components:

$$d(\mathbf{x},\mathbf{y}) = f(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) \tag{4.1}$$

Distance measures play an important role in many areas that employ multidimensional data analysis. One such area is hyperspectral image processing. Hyperspectral images are based on recording the scene's reflected light in bands associated to narrow intervals of spectrum wavelengths (**see Figure 4.3**). Each element from the image (pixel) is associated with a certain area of the surveyed scene and with its spectral response [31]. In this case, differences in the spectral intervals will lead to differentiation of the pixel values of distinct materials [32]. Note how different vegetation types reflect differently under the infrared light in the plant arrangement displayed in the Figure 4.3.

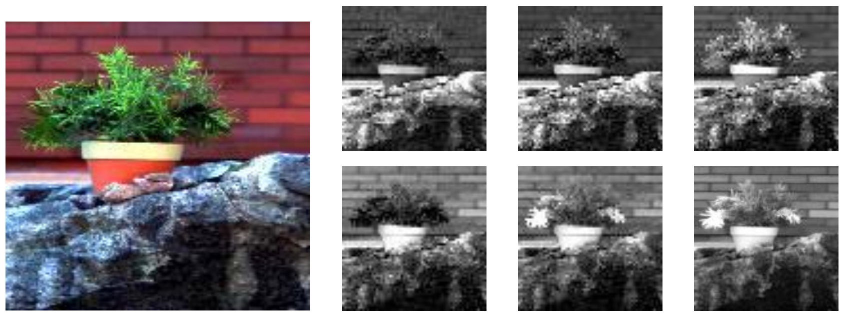
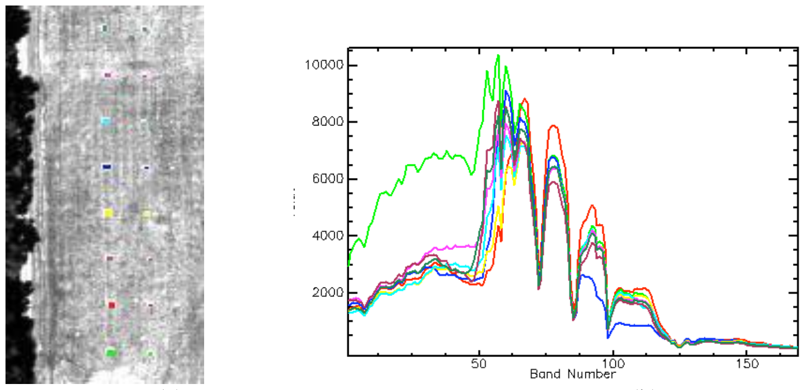

(a) (b)

Figure 4.3 - Hyperspectral images are formed by combining a large number of narrow spectrum bands. a) color composite image b) spectral bands ranging from visible (blue, green, red – top row) to near infrared (bottom row).

Spectra, (also called spectral vectors) are obtained by collecting the pixel values for the same location throughout the image stack. In figure 4.4, we display a hyperspectral image with various panels placed on the ground. The corresponding average spectra are plotted.



(a) (b)

Figure 4.4 a) Hyperspectral data displaying rows of panels marked by squares, b) Average spectra for the eight panel categories in the scene

In hyperspectral data the idea of spectra separability is at the basis of many problems. If a material spectrum is easily separable from the background, then the material will be easily detected in the image [33]. If two adjoining materials have significantly dissimilar spectra, then the edge between them will also be easily detected. Unfortunately, clear separability cannot be easily achieved. Each spectrum is formed of tens to hundreds of values collected within narrow adjacent wavelength intervals and often exposes strong local correlation [34]. Solutions to target separability are often found in the literature. One such popular distance is the spectral angle [35].

Given two vectors of the same dimension **x** and **y**, the *spectral angle* is defined as the arccosine of their dot product [35]:

$$SA(x,y) = ar\cos\left(\frac{\langle x,y\rangle}{\|x\|\|y\|}\right) \tag{4.2}$$

where <.,.> represents the dot product of the vectors and ||**.**|| the Euclidean norm.

Since many of the vector components are strongly related to each other, it is of interest to design methods that would select only those bands that increase (or decrease the separability). Given two spectra **x** and **y** with values over a set of spectra bands **B**, a spectral distance β the goal of band screening is to find the subset of bands $\mathbf{B_1}$ such that:

$$\beta(x,y,B_1) = \min_{B_1 \subseteq B}(\beta(x,y,B) \tag{4.3}$$

where by $\beta(\mathbf{x}, \mathbf{y}, \mathbf{B_s})$ we refer to the value of the distance measure β computed between the two vectors but taking into consideration only the bands included in the subset $\mathbf{B_s}$.

The goal in band screening is to find the subset of bands that maximize the distance between the two spectra [32]. While finding the optimum subset is always possible through exhaustive search, the complexity of such operation is prohibitive. Given a hyperspectral image of *n* bands, and assuming the B_1 can have any size (between 1 and 150), the number of band combinations to be tried is roughly equivalent to the number of possible mappings:

$$f : \{1,2,3,\ldots,n\} \to \{0,1\} \tag{4.4}$$

This is because, each subset $\mathbf{B_s}$ of **B** can be seen as an *n*-uple of 0's and 1's where, for each position, one indicating that the corresponding band is in the subset and zero indicating the absence of the band. The above equation leads to 2^n possible mappings. Using the deployed grid we decided to break these calculations up into smaller intervals and spread the calculations over several computer systems making the wait for results less time consuming.

To conceptually test our approach we have selected four 38-dimensional pixel vectors associated to the first row of panels and computed the average spectral vectors. We then exhaustively searched for the minimum distance between the average vectors and the original vectors by trying all the possible band combinations. The application was run twice using a different interval of numbers for each run. This allowed us to submit a different number of jobs to the grid each time the application is run. Since it will take longer then 8 hours for all the application jobs to complete we will not be able to run a set of jobs on an increased number of CPUs. If we were able to deploy a larger number of CPUs for the grid running such a test might have been feasible. Since the application jobs will not finish processing in less then 8 hours the jobs will be process on 8 CPUs between the hours of 07:00 and 23:00. Between the hours of 23:00 and 07:00 the jobs will process on 14 CPUs. Any jobs that are running on a queue that becomes suspended at 07:00 will also become suspended. If a system that contains a suspended queue is restarted while there is a suspended job in the queue the job will be rescheduled and placed in the queue to run again.

The Java code that is being used to determine the minimum distance components for vectors will compute the best bands that minimize the distance between one vector and a group of vectors. When the code is compiled and executed it requires 3 parameters. The first parameter being the file containing the vectors, the second parameter being the lower bound of the vector and the third parameter being the upper bound of the vector. The file containing the vectors will be submitted as a normal text file. The 2 parameters that make up the lower and upper bounds will be submitted as 38 bit binary numbers. The program is submitted to the grid using the wrapper script *bestDistance.sh*. The wrapper script specifics where Java binary is, what shell to use, that the job is re-runable if it was aborted without leaving a consistent exit state, sets up need path information and executes the program with the submitted parameters. The Java code for the application and *bestDistance.sh* script can be found in appendix I.

A second Java application was written to produce the binary number intervals that would make up the lower and upper bound parameters. When the code is compiled and executed it requires 3 parameters. The first parameter being the text file containing the vectors, the second parameter being the number of vectors and the third parameter being the interval. The Java code for the application can be found in appendix I. The jobs are submitted to the grid with the following command line string, changing the binary interval number for each jobs submitted.

```
“sgeadmin@panic:/export/home/sgeadmin/jobs $ qsub bestDistance.sh
spectra_1a.txt 00000000000000000000000000000000000000
00000000000010000000000000000000000000”
```

The application to determine the minimum distance components for vectors was run a total of 2 different times. The application was run with beginning parameter of 00000000000000000000000000000000000000 and an ending parameter of 11111111111110000000000000000000000000. The program was first run with 4,095 jobs and a variable amount of CPUs since the jobs took longer then 8 hours to process. The program was then run with 8,191 jobs and a variable amount of CPUs. The 2 application runs produced the following results.

Average Minute for Each Job	# of Jobs
7.287464131	4095
3.894372525	8191

Table 4.2 - Average Minute per Job.

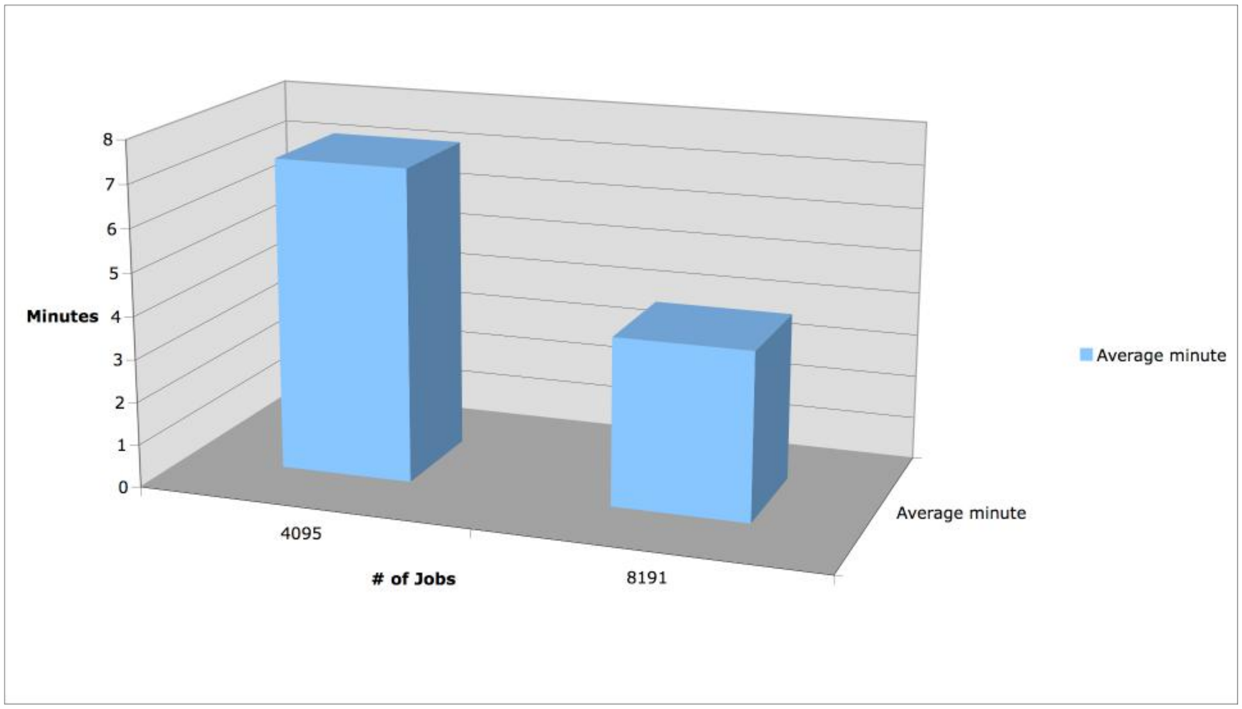


Figure 4.5 – Average time per job in minutes.

These results show that by shorting the interval for each job and increasing the number of jobs to process that the average time to process a job is significantly decreased. This initial analysis may lead to the assumption that by increasing the number of jobs that the total time for processing the jobs will decrease. This assumption turns out to be false. To determine the total time to process all jobs simply multiply the number of jobs by the average time it took to complete a job. This would indicate the total time it would take to process all the jobs if the jobs were run in a serial fashion. This is the best way to determine the total run time since the number of active CPUs within the grid changes over a 24-hour period. The results for total processing time are as follows.

Average Minute for Each Job	Total Time Based on Average Minute	# of Jobs
7.287464131	29813.01576	4095
3.894372525	31898.80535	8191

Table 4.3 – Average Time per the Number of Jobs.

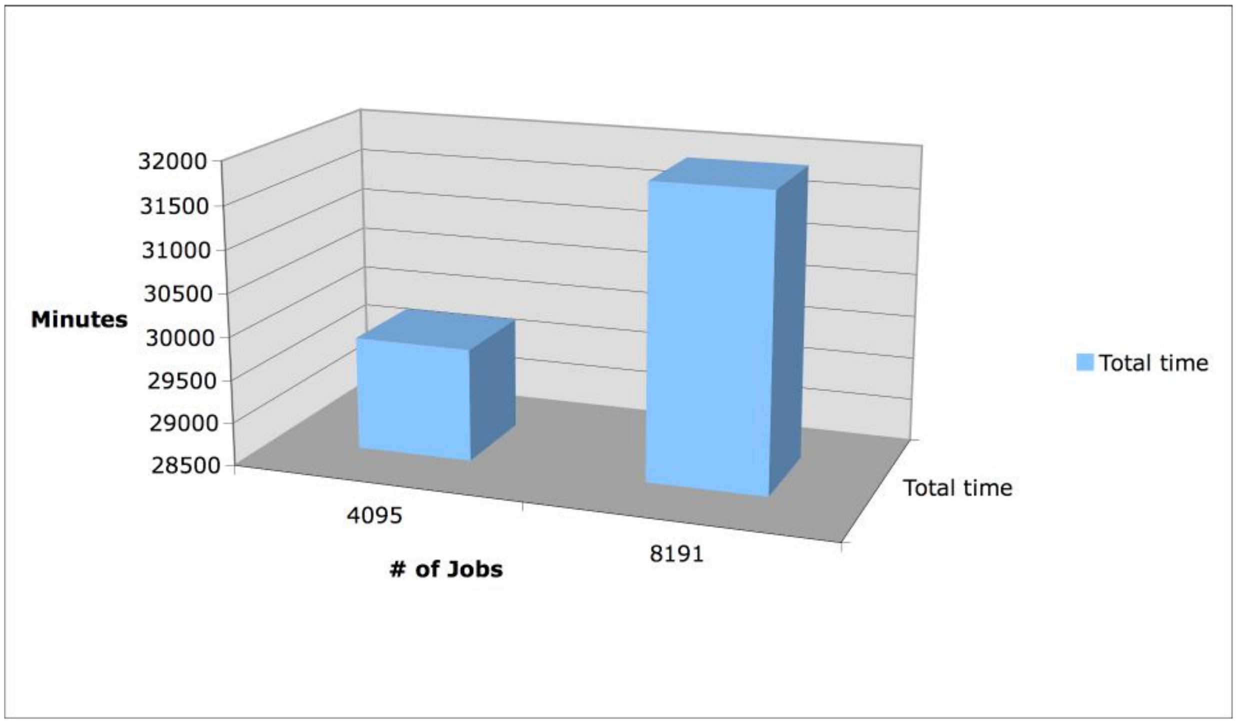


Figure 4.6 – Total time to run the jobs in a serial fashion based off the average time per job.

The results shown in figure 4.6 show that by increasing the number of jobs the total time to process the jobs has increased a significant amount. This increase is most likely due to an increase in overhead within the grid system. The increase shows that an increase in the number of jobs for this application will result in longer wait times for results. This application would most likely benefit from having a greater number of systems available to process the jobs. This would allow for a greater number of jobs to be processed at once.

5. Conclusion

Distributed computing has started to influence the speed at which data is processed. This increased speed of processing has reduced the amount of time wasted while waiting for data. Distributed computing systems do not require special application code to be written unlike regular parallel computing systems, which require special application code to be written to process the application on the system. This is one of many features that has made distributed computing an attractive option for processing large amounts of data.

In this project, we have designed and implemented a computing grid that allows for distributed processing across a parallel infrastructure. The main advantage of the computing grid is that it allows for the use of already purchased computer systems which have unused CPU cycles to be used without interrupting the intended purpose of the systems. This has allowed for large amounts of data to be processed in a shorter amount of time then it would have taken to process the data on a single computer system. Though the grid we have been able to shorten the wait time for collecting data reducing the amount of wasted time.

REFERENCES

[1] Tay, L. “Grid computes 420 years worth of data in four months”, Computerworld, 16 February 2007. 29 March 2007 <http://www.computerworld.com.au/index.php/id;446136154;fp;16;fpid;1>.

[2] IBM, “IBM Solutions Grid for Business Partners Helping IBM Business Partners to Grid-enable applications for the next phase of e-business on demand”, < http://www-304.ibm.com/jct09002c/isv/marketing/emerging/grid_wp.pdf>.

[3] Wikipedia. “Grid computing.” 12 January 2007 <http://en.wikipedia.org/wiki/Grid_computing>.

[4] The EDUCAUSE Learning Initiative. “7 Things You Should Know About Grid Computing.” Educause. January 2006. <http://www.educause.edu/LibraryDetailPage/666?ID=ELI7010>

[5] A. M. Braverman, “Father of the Grid”, *The University of Chicago Magzine,* May 2004, <http://magazine.uchicago.edu/0404/features/index.shtml>.

[6] I. Foster, J. Geisler, W. Nickless, W. Smith, S. Tuecke., “Software Infrastructure for the I-WAY High Performance Distributed Computing Experiment”, *Proc. 5th IEEE Symposium on High Performance Distributed Computing*, pp. 562-571, 1997.

[7] S. Lohr, “Teaching Computers To Work In Unison.” New York Times 15 July 2003

[8] The Globus Alliance, *The Globus Alliance*, 19 February 2007 <http://www.globus.org>.

[9] SunSource.net, *gridengine Project home*, 12 January 2007 <http://gridengine.sunsource.net>.

[10] Sun Microsystems, *Sun N1 Grid Engine 6*, 12 January 2007 <http://www.sun.com/software/gridware>.

[11] Condor High Throughput Computing, *Condor Project Homepage*, 14 February 2007 <http://www.cs.wisc.edu/condor>.

[12] Microsoft. *Microsoft Services for UNIX*, 13 January 2007 <http://www.microsoft.com/downloads/details.aspx?familyid=896C9688-601B-44F1-81A4 02878FF11778&displaylang=en>.

[13] Sun Microsystems, *Grid Technology - Case Studies,* 17 January 2007 <http://www.sun.com/software/grid/case_studies.xml>.

[14] Sun Microsystems, *Sun N1 Grid Engine 6 Data Sheet*, 27 January 2007 <http://www.sun.com/software/gridware/datasheet.xml>.

[15] Sun Microsystems. *N1 Grid Engine 6 Release Notes*. May 2005.

[16] Sun Microsystems. *N1 Grid Engine Installation Guide.* May 2005.

[17] Indiana University, "In Unix, what is tar, and how do I use it?", *IU Knowledge Base*, <http://kb.iu.edu/data/acfi.html>.

[18] Sun Microsystems. *sge_conf Manual Page.* November 2005.

[19] Sun Microsystems. *N1 Grid Engine Administration Guide.* May 2005.

[20] V. Berstis. "Fundamentals of Grid Computing." *Redbooks Paper.* 2002. 13 January 2007. <http://www.redbooks.ibm.com/abstracts/redp3613.html?Open>.

[21] Sun Microsystems. *cron Manual Page.* May 2006.

[22] Sun Microsystems, *Java Technology*, 11 April 2007, <http://java.sun.com/>.

[23] J. Lewis, W. Loftus, *Java Software Solutions – Foundations of Program Design*, Addison Wesley, 2006.

[24] Torres, Mario A. “Developing Scalable Distributed Applications.” Dr. Dobb’s Journal 328 (2001): 21-25.

[25] J. J. O'Connor and E. F. Robertson, “Prime Numbers”, 18 April 2007 <http://www-groups.dcs.st-and.ac.uk/~history/HistTopics/Prime_numbers.html>.

[26] C. Caldwell, “Euclid's Proof of the Infinitude of Primes (c. 300 BC)”, 18 April 2007 <http://primes.utm.edu/notes/proofs/infinite/euclids.html>.

[27] Wikipedia. “Prime Number.” 12 January 2007 <http://en.wikipedia.org/wiki/Prime_number>.

[28] Weisstein, E. W. "Fermat's Little Theorem." *MathWorld--A Wolfram Web Resource*, 18 April 2007 <http://mathworld.wolfram.com/FermatsLittleTheorem.html>.

[29] “GIMPS – The Great Internet Mersenne Prime Search”, *GIMPS Home Page*, 18 April 2007 <http://www.mersenne.org>.

[30] Z. Stankova-Frenkel, “RSA Encryption”, 18 April 2007 <http://mathcircle.berkeley.edu/BMC3/rsa/node4.html>.

[31] Lillesand, T. M. and R.W. Kiefer (2000). *Remote sensing and image interpretation*, John Wiley and Sons, New York.

[32] N. Keshava N., “Distance metrics and band selection in hyperspectral processing with applications to material identification and spectral libraries”, *IEEE Trans. Geosci. Remote Sensing*, 2004, vol. 42, no.7, 1552- 1565, 2004.

[33] S. A. Robila, A. Gershman, “Spectral Matching Accuracy in Processing Hyperspectral Data”, *IEEE ISSCS,* pp. 163-166, 2005.

[34] S. A. Robila, "New Developments in Target Detection in Hyperspectral Imagery Using Spectral Metrics and Spectra Extraction", *Proceedings ASPRS National Conference*, May 2007, 11 pgs on CD.

[35] F.A Kruse, Lekoff A.B., Boardman J.W., Heidebrecht K.B., Shapiro A.T., Barloon P.J., and A.F.H. Goetz, "The Spectral Image Processing System (SIPS) – interactive visualization and analysis of imaging spectometer data", Remote Sensing of the Environment, 44, 145-163, 1993.

Appendix A - Installing Sun N1 Grid Engine Software on Server

Download the N1 Grid Engine Software from *http://www.sun.com*

Once the file is downloaded unzip the file

```
-  root@one:/var/tmp # unzip n1ge6u9.zip
```

Go to the directory in which the software was unzipped to and unzip the following packages for the SPARC and x86 Solaris installs. These packages also include architecture independent files and documentation for all versions of the grid engine software.

```
-  root@one:/var/tmp/n1ge6_0u9 # n1ge-6_0u9-bin-solaris-
   sparcv9.zip
-  root@one:/var/tmp/n1ge6_0u9 # n1ge-6_0u9-bin-solaris-x64.zip
-  root@one:/var/tmp/n1ge6_0u9 # n1ge-6_0u9-bin-solaris.zip
-  root@one:/var/tmp/n1ge6_0u9 # n1ge-6_0u9-common.zip
-  root@one:/var/tmp/n1ge6_0u9 # n1ge-6_0u9-doc.zip
```

Once these files are unzipped choose the directory in which the software will be installed and create the directory.

```
-  root@one:/var/tmp/n1ge6_0u9 # mkdir -p /opt/gridware/sge
```

From the directory in which the software was unzipped into run the pkgadd command to install the software.

```
-  root@one:/var/tmp/n1ge6_0u9 # pkgadd -d .
```

```
The following packages are available:
  1  SUNWsgee       N1 Grid Engine - Solaris 32 bit binaries
                    (sparc) 6.0,REV=2004.06.03.16.00
  2  SUNWsgeeax     N1 Grid Engine - Solaris x64 bit binaries
                    (x64) 6.0u4,REV=2005.05.06.11.30
  3  SUNWsgeec      N1 Grid Engine - architecture independent files
                    (all) 6.0,REV=2004.06.03.16.00
  4  SUNWsgeed      N1 Grid Engine - Documentation
                    (all) 6.0,REV=2004.06.03.16.00
  5  SUNWsgeex      N1 Grid Engine - Solaris 64 bit binaries
                    (sparc) 6.0,REV=2004.06.03.16.00

Select package(s) you wish to process (or 'all' to process
all packages). (default: all) [?,??,q]:

Processing package instance <SUNWsgee> from </var/tmp/n1ge6_0u9>

N1 Grid Engine - Solaris 32 bit binaries(sparc) 6.0,REV=2004.06.03.16.00
Copyright 2004 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Where should N1GE 6.0 be installed [default /gridware/sge, ? for help] [?,q]
/opt/gridware/sge
```

```
Using </opt/gridware/sge> as the package base directory.
## Processing package information.
## Processing system information.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

The following files are being installed with setuid and/or setgid
permissions:
  /opt/gridware/sge/utilbin/sol-sparc/rlogin <setuid root>
  /opt/gridware/sge/utilbin/sol-sparc/rsh <setuid root>
  /opt/gridware/sge/utilbin/sol-sparc/testsuidroot <setuid root>

Do you want to install these as setuid/setgid files [y,n,?,q] y

Installing N1 Grid Engine - Solaris 32 bit binaries as <SUNWsgee>

## Installing part 1 of 1.
/opt/gridware/sge/bin/sol-sparc/qacct
/opt/gridware/sge/bin/sol-sparc/qalter
/opt/gridware/sge/bin/sol-sparc/qconf
.
.
.
/opt/gridware/sge/utilbin/sol-sparc/spoolinit
/opt/gridware/sge/utilbin/sol-sparc/testsuidroot
/opt/gridware/sge/utilbin/sol-sparc/uidgid
[ verifying class <none> ]

Installation of <SUNWsgee> was successful.

Processing package instance <SUNWsgeeax> from </var/tmp/n1ge6_0u9>

N1 Grid Engine - Solaris x64 bit binaries(x64) 6.0u4,REV=2005.05.06.11.30
Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Where should N1GE 6.0u4 be installed [default /gridware/sge, ? for help]
[?,q]
/opt/gridware/sge
Using </opt/gridware/sge> as the package base directory.
## Processing package information.
## Processing system information.
   5 package pathnames are already properly installed.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

The following files are being installed with setuid and/or setgid
permissions:
  /opt/gridware/sge/utilbin/sol-amd64/rlogin <setuid root>
  /opt/gridware/sge/utilbin/sol-amd64/rsh <setuid root>
  /opt/gridware/sge/utilbin/sol-amd64/testsuidroot <setuid root>

Do you want to install these as setuid/setgid files [y,n,?,q] y

Installing N1 Grid Engine - Solaris x64 bit binaries as <SUNWsgeeax>
```

```
## Installing part 1 of 1.
/opt/gridware/sge/bin/sol-amd64/qacct
/opt/gridware/sge/bin/sol-amd64/qalter
/opt/gridware/sge/bin/sol-amd64/qconf
.
.
.
/opt/gridware/sge/utilbin/sol-amd64/spoolinit
/opt/gridware/sge/utilbin/sol-amd64/testsuidroot
/opt/gridware/sge/utilbin/sol-amd64/uidgid
[ verifying class <none> ]

Installation of <SUNWsgeeax> was successful.

Processing package instance <SUNWsgeec> from </var/tmp/n1ge6_0u9>

N1 Grid Engine - architecture independent files(all) 6.0,REV=2004.06.03.16.00
Copyright 2004 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Where should N1GE 6.0 be installed [default /gridware/sge, ? for help] [?,q]
/opt/gridware/sge
Using </opt/gridware/sge> as the package base directory.
## Processing package information.
## Processing system information.
   2 package pathnames are already properly installed.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

Installing N1 Grid Engine - architecture independent files as <SUNWsgeec>

## Installing part 1 of 1.
/opt/gridware/sge/3rd_party/THIRDPARTYLICENSEREADME.txt
/opt/gridware/sge/3rd_party/qmon/copyrights
/opt/gridware/sge/3rd_party/qmon/ltree_changed.tar.gz
.
.
.
/opt/gridware/sge/util/sge_log_tee
/opt/gridware/sge/util/sge_request
/opt/gridware/sge/util/sgeremoterun
[ verifying class <none> ]

Installation of <SUNWsgeec> was successful.

Processing package instance <SUNWsgeed> from </var/tmp/n1ge6_0u9>

N1 Grid Engine - Documentation(all) 6.0,REV=2004.06.03.16.00
Copyright 2004 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Where should N1GE 6.0 be installed [default /gridware/sge, ? for help] [?,q]
/opt/gridware/sge
Using </opt/gridware/sge> as the package base directory.
## Processing package information.
## Processing system information.
   1 package pathname is already properly installed.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

Installing N1 Grid Engine - Documentation as <SUNWsgeed>

## Installing part 1 of 1.
/opt/gridware/sge/doc/N1GE6Update4_Administration_Guide.pdf
/opt/gridware/sge/doc/N1GE6Update4_GEMM_Guide.pdf
/opt/gridware/sge/doc/N1GE6Update4_Installation_Guide.pdf
.
.
.
/opt/gridware/sge/doc/htmldocs/N1GE6Update4_User_Guide/toc-glossary-1.html
/opt/gridware/sge/doc/htmldocs/N1GE6Update4_User_Guide/toc-preface-1.html
/opt/gridware/sge/doc/htmldocs/N1GE6Update4_User_Guide/toc.html
[ verifying class <none> ]

Installation of <SUNWsgeed> was successful.

Processing package instance <SUNWsgeex> from </var/tmp/n1ge6_0u9>

N1 Grid Engine - Solaris 64 bit binaries(sparc) 6.0,REV=2004.06.03.16.00
Copyright 2004 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Where should N1GE 6.0 be installed [default /gridware/sge, ? for help] [?,q]
/opt/gridware/sge
Using </opt/gridware/sge> as the package base directory.
## Processing package information.
## Processing system information.
   5 package pathnames are already properly installed.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

The following files are being installed with setuid and/or setgid
permissions:
  /opt/gridware/sge/utilbin/sol-sparc64/rlogin <setuid root>
  /opt/gridware/sge/utilbin/sol-sparc64/rsh <setuid root>
  /opt/gridware/sge/utilbin/sol-sparc64/testsuidroot <setuid root>

Do you want to install these as setuid/setgid files [y,n,?,q] y

Installing N1 Grid Engine - Solaris 64 bit binaries as <SUNWsgeex>

## Installing part 1 of 1.
/opt/gridware/sge/bin/sol-sparc64/qacct
/opt/gridware/sge/bin/sol-sparc64/qalter
/opt/gridware/sge/bin/sol-sparc64/qconf
.
.
.
/opt/gridware/sge/utilbin/sol-sparc64/spoolinit
/opt/gridware/sge/utilbin/sol-sparc64/testsuidroot
```

```
/opt/gridware/sge/utilbin/sol-sparc64/uidgid
[ verifying class <none> ]

Installation of <SUNWsgeex> was successful.
root@one:/var/tmp/n1ge6_0u9 #
```

The N1 Grid Engine Software is now installed on the server for the SPARC and x86 Solaris Architectures along with some architecture independent files and documentation.

To install additional architectures of the software go to the directory in which the software was unzipped and copy the file for the architecture that you need to the installation directory of /opt/gridware/sge. Then uncompress and untar the file in the installation directory.

- `root@one:/var/tmp/n1ge6_0u9 # cp n1ge-6_0u9-bin-windows-x86.tar.gz /opt/gridware/sge/`
- `root@one:/optgridware/sge # gzip -cd n1ge-6_0u9-bin-windows-x86.tar.gz |tar xvf -`

```
x bin/win32-x86, 0 bytes, 0 tape blocks
x bin/win32-x86/sge_execd, 1974378 bytes, 3857 tape blocks
x bin/win32-x86/sge_shepherd, 1678011 bytes, 3278 tape blocks
.
.
.
x lib/win32-x86/libcrypto.so symbolic link to libcrypto.so.0.9.7
x lib/win32-x86/libssl.so.0.9.7, 305954 bytes, 598 tape blocks
x lib/win32-x86/libssl.so symbolic link to libssl.so.0.9.7
root@one:/opt/gridware/sge #
```

This architecture is now installed on the server. You can remove the compress file that was copied over for the install.

- `root@one:/optgridware/sge # rm n1ge-6_0u9-bin-windows-x86.tar.gz`

Repeat these steps for any other architecture that you may wish to install.

Appendix B - Installing a UNIX Master Server

B.1. Log in to the master host as root.

B.2. Ensure that you have set the $SGE_ROOT environment variable by typing:

```
# echo $SGE_ROOT
```

If the $SGE_ROOT environment variable is not set, set it now, by typing:

```
# SGE_ROOT=sge-root; export SGE_ROOT
```

B.3. Change to the installation directory.

B.4. Type the install_qmaster command.

This command starts the master host installation procedure. You will be asked several questions and may be required to run some administrative actions.

```
one:/opt/gridware/sge# ./install_qmaster
Welcome to the Grid Engine installation
---------------------------------------
Grid Engine qmaster host installation
-------------------------------------
Before you continue with the installation please read these hints:

- Your terminal window should have a size of at least
80x24 characters

- The INTR character is often bound to the key Ctrl-C.
The term >Ctrl-C< is used during the installation if you
have the possibility to abort the installation

The qmaster installation procedure will take approximately 5-10 minutes.

Hit <RETURN> to continue >>
```

B.5. Choose an administrative account owner.

```
Choosing Grid Engine admin user account
---------------------------------------

You may install Grid Engine that all files are created with the user id of an
unprivileged user.

This will make it possible to install and run Grid Engine in directories
where user >root< has no permissions to create and write files and
directories.

- Grid Engine still has to be started by user >root<

- this directory should be owned by the Grid Engine administrator

Do you want to install Grid Engine
under an user id other than >root< (y/n) [y] >> y


Choosing a Grid Engine admin user name
```

```
--------------------------------------
Please enter a valid user name >> sgeadmin

Installing Grid Engine as admin user >sgeadmin<

Hit <RETURN> to continue >>
```

B. 6. Verify the *sge-root* directory setting.

```
Checking $SGE_ROOT directory
----------------------------

The Grid Engine root directory is:

$SGE_ROOT = /opt/gridware/sge

If this directory is not correct (e.g. it may contain an automounter
prefix) enter the correct path to this directory or hit <RETURN>
to use default [/opt/n1ge6] >> /opt/gridware/sge

Your $SGE_ROOT directory: /opt/gridware/sge

Hit <RETURN> to continue >>
```

B. 7. Set up the TCP/IP services for the grid engine software.

```
Grid Engine TCP/IP service >sge_qmaster<
----------------------------------------------------
Using the service

sge_ qmaster

for communication with Grid Engine.

Hit <RETURN> to continue >>


Grid Engine TCP/IP service >sge_execd<
--------------------------------------

Using the service

sge_execd

for communication with Grid Engine.

Hit <RETURN> to continue >>
```

B. 8. Enter the name of your cell.

```
Grid Engine cells
-----------------
Grid Engine supports multiple cells.

If you are not planning to run multiple Grid Engine clusters or if you don't
know yet what is a Grid Engine cell it is safe to keep the default cell name

default
```

```
If you want to install multiple cells you can enter a cell name now.

The environment variable

$SGE_CELL=<your_cell_name>

will be set for all further Grid Engine commands.

Enter cell name [default] >> default

Using cell >default<.

Hit <RETURN> to continue >>
```

B. 9. Specify a spool directory.

```
Grid Engine qmaster spool directory
-----------------------------------
The qmaster spool directory is the place where the qmaster daemon stores
the configuration and the state of the queuing system.

The admin user >sgeadmin< must have read/write access
to the qmaster spool directory.

If you will install shadow master hosts or if you want to be able to start
the qmaster daemon on other hosts (see the corresponding section in the
Grid Engine Installation and Administration Manual for details) the account
on the shadow master hosts also needs read/write access to this directory.
The following directory

[/opt/n1ge6/default/spool/qmaster]

will be used as qmaster spool directory by default!

Do you want to select another qmaster spool directory (y/n) [n] >> n
```

B. 10. Windows-based execution hosts.

```
Windows Execution Host Support
------------------------------

Are you going to install Windows Execution Hosts? (y/n) [n] >> y

Windows Domain User Access
--------------------------

Do you want to use Windows Domain Users (answer: y)
or are you going to use local Windows Users (answer: n) (y/n) [y] >> n
```

B. 11. Verify or set the correct file permissions. We installed using pkgadd so file permissions are correct.

```
Verifying and setting file permissions
--------------------------------------

Did you install this version with >pkgadd< or did you already
```

```
verify and set the file permissions of your distribution (enter: y)

In some cases, eg: the binaries are stored on a NTFS or on any other
filesystem, which provides additional file permissions, the UNIX file
permissions can be wrong. In this case we would advise to verify and
to set the file permissions (enter: n) (y/n) [n] >> y

We do not verify file permissions. Hit <RETURN> to continue >>
```

B. 12. Specify whether all of your grid engine system hosts are located in a single DNS domain.

```
Select default Grid Engine hostname resolving method
----------------------------------------------------

Are all hosts of your cluster in one DNS domain? If this is
the case the hostnames

>hostA< and >hostA.foo.com<

would be treated as eqal, because the DNS domain name >foo.com<
is ignored when comparing hostnames.

Are all hosts of your cluster in a single DNS domain (y/n) [y] >> y

Ignoring domainname when comparing hostnames.

Hit <RETURN> to continue >>
```

B. 13. Making needed directories.

```
Making directories
------------------

creating directory: default
creating directory: default/common
creating directory: /opt/gridware/sge/default/spool/qmaster
creating directory: /opt/gridware/sge/default/spool/qmaster/job_scripts
Hit <RETURN> to continue >>
```

B. 14. Specify whether you want to use classic spooling or Berkeley DB.

```
Setup spooling
--------------

Your SGE binaries are compiled to link the spooling libraries
during runtime (dynamically). So you can choose between Berkeley DB
spooling and Classic spooling method.
Please choose a spooling method (berkeleydb|classic) [berkeleydb] >>
berkeleydb

The Berkeley DB spooling method provides two configurations!

Local spooling:
The Berkeley DB spools into a local directory on this host (qmaster host)
This setup is faster, but you can't setup a shadow master host
```

```
Berkeley DB Spooling Server:
If you want to setup a shadow master host, you need to use
Berkeley DB Spooling Server!
In this case you have to choose a host with a configured RPC service.
The qmaster host connects via RPC to the Berkeley DB. This setup is more
failsafe, but results in a clear potential security hole. RPC communication
(as used by Berkeley DB) can be easily compromised. Please only use this
alternative if your site is secure or if you are not concerned about
security. Check the installation guide for further advice on how to achieve
failsafety without compromising security.

Do you want to use a Berkeley DB Spooling Server? (y/n) [n] >> n

Hit <RETURN> to continue >>

Berkeley Database spooling parameters
-------------------------------------

Please enter the Database Directory now, even if you want to spool locally
it is necessary to enter this Database Directory.

Default: [/opt/gridware/sge/default/spool/spooldb] >>
/opt/gridware/sge/default/spool/spooldb

creating directory: /opt/n1ge6/default/spool/spooldb
Dumping bootstrapping information
Initializing spooling database

Hit <RETURN> to continue >>
```

B. 15. Enter a group ID range

Grid Engine group id range

```
When jobs are started under the control of Grid Engine an additional group id
is set on platforms which do not support jobs. This is done to provide
maximum
control for Grid Engine jobs.

This additional UNIX group id range must be unused group id's in your system.
Each job will be assigned a unique id during the time it is running.
Therefore you need to provide a range of id's which will be assigned
dynamically for jobs.

The range must be big enough to provide enough numbers for the maximum number
of Grid Engine jobs running at a single moment on a single host. E.g. a range
like >20000-20100< means, that Grid Engine will use the group ids from
20000-20100 and provides a range for 100 Grid Engine jobs at the same time
on a single host.

You can change at any time the group id range in your cluster configuration.

Please enter a range >> 20000-40000

Using >20000-40000 < as gid range. Hit <RETURN> to continue >>
```

B. 16. Verify the spooling directory for the execution daemon.

```
Grid Engine cluster configuration
---------------------------------

Please give the basic configuration parameters of your Grid Engine
installation:

<execd_spool_dir>

The pathname of the spool directory of the execution hosts. User >sgeadmin<
must have the right to create this directory and to write into it.

Default: [/opt/gridware/sge/default/spool] >> /opt/gridware/sge/default/spool
```

B. 17. Enter the email address of the user who should receive problem reports.

```
Grid Engine cluster configuration (continued)
---------------------------------------------

<administator_mail>
The email address of the administrator to whom problem reports are sent.

It's is recommended to configure this parameter. You may use >none<
if you do not wish to receive administrator mail.

Please enter an email address in the form >user@foo.com<.
Default: [none] >>
```

B. 18. Verify the configuration parameters.

```
The following parameters for the cluster configuration were configured:

execd_spool_dir /opt/gridware/sge/default/spool
administrator_mail none

Do you want to change the configuration parameters (y/n) [n] >> n

Creating local configuration
----------------------------

Creating >act_qmaster< file
Adding default complex attributes
Reading in complex attributes.
Adding default parallel environments (PE)
Reading in parallel environments:
PE "make".
Adding SGE default usersets
Reading in usersets:
Userset "deadlineusers".
Userset "defaultdepartment".
Adding >sge_aliases< path aliases file
Adding >qtask< qtcsh sample default request file
Adding >sge_request< default submit options file
Creating >sgemaster< script
Creating >sgeexecd< script
Creating settings files for >.profile/.cshrc<
```

```
Hit <RETURN> to continue >>
```

B. 19. Specify whether you want the daemons to start when the system is booted.

```
qmaster/scheduler startup script
--------------------------------

We can install the startup script that will
start qmaster/scheduler at machine boot (y/n) [y] >> y

Installing startup script /etc/rc2.d/S95sgemaster and /etc/rc2.d/K03sgemaster

Hit <RETURN> to continue >>


Grid Engine qmaster and scheduler startup
-----------------------------------------

Starting qmaster and scheduler daemon. Please wait ...
   starting sge_qmaster
   starting sge_schedd
Hit <RETURN> to continue >>
```

B. 20. Add the Windows Administrator name to the SGE manager list.

```
Windows Administrator Name
--------------------------
For a later execution host installation it is recommended to add the
Windows Administrator name to the SGE manager list

Please, enter the Windows Administrator name [Default: Administrator] >>
Administrator
root@one added "Administrator" to manager list
Hit <RETURN> to continue >>
```

B. 21. Identify the hosts that you will later install as execution hosts.

```
Adding Grid Engine hosts
------------------------
Please now add the list of hosts, where you will later install your execution
daemons. These hosts will be also added as valid submit hosts.

Please enter a blank separated list of your execution hosts. You may
press <RETURN> if the line is getting too long. Once you are finished
simply press <RETURN> without entering a name.

You also may prepare a file with the hostnames of the machines where you plan
to install Grid Engine. This may be convenient if you are installing Grid
Engine on many hosts.

Do you want to use a file which contains the list of hosts (y/n) [n] >> n

Adding admin and submit hosts
-----------------------------
Please enter a blank separated list of hosts.
Stop by entering <RETURN>. You may repeat this step until you are
```

```
entering an empty list. You will see messages from Grid Engine
when the hosts are added.
Host(s):
Finished adding hosts.  Hit <RETURN> to continue >>

Creating the default <all.q> queue and <allhosts> hostgroup
-------------------------------------------------------------

root@one added "@allhosts" to host group list
root@one added "all.q" to cluster queue list

Hit <RETURN> to continue >>
```

B. 22. Select a scheduler profile.

```
Scheduler Tuning
----------------

The details on the different options are described in the manual.
Configurations
--------------

1) Normal
      Fixed interval scheduling, report scheduling information,
      actual + assumed load

2) High
      Fixed interval scheduling, report limited scheduling information,
      actual load

3) Max
      Scheduling on demand, report no scheduling information,
      actual load

Enter the number of your preferred configuration and hit <RETURN>!
Default configuration is [1] >> 2

We're configuring the scheduler with >High< settings!
Do you agree? (y/n) [y] >> y

changed scheduler configuration


Using Grid Engine
-----------------

You should now enter the command:

   source /opt/gridware/sge/default/common/settings.csh

if you are a csh/tcsh user or

   # . /opt/gridware/sge/default/common/settings.sh

if you are a sh/ksh user.

This will set or expand the following environment variables:
   - $SGE_ROOT         (always necessary)
   - $SGE_CELL         (if you are using a cell other than >default<)
   - $SGE_QMASTER_PORT (if you haven't added the service >sge_qmaster<)
   - $SGE_EXECD_PORT   (if you haven't added the service >sge_execd<)
   - $PATH/$path       (to find the Grid Engine binaries)
   - $MANPATH          (to access the manual pages)

Hit <RETURN> to see where Grid Engine logs messages >>



Grid Engine messages
--------------------

Grid Engine messages can be found at:

   /tmp/qmaster_messages (during qmaster startup)
   /tmp/execd_messages   (during execution daemon startup)

After startup the daemons log their messages in their spool directories.

   Qmaster:     /opt/gridware/sge/default/spool/qmaster/messages
   Exec daemon: <execd_spool_dir>/<hostname>/messages

Grid Engine startup scripts
---------------------------

Grid Engine startup scripts can be found at:

   /opt/gridware/sge/default/common/sgemaster (qmaster and scheduler)
   /opt/gridware/sge/default/common/sgeexecd (execd)

Do you want to see previous screen about using Grid Engine again (y/n) [n] >>
n



Your Grid Engine qmaster installation is now completed
------------------------------------------------------

Please now login to all hosts where you want to run an execution daemon
and start the execution host installation procedure.

If you want to run an execution daemon on this host, please do not forget
to make the execution host installation in this host as well.

All execution hosts must be administrative hosts during the installation.
All hosts which you added to the list of administrative hosts during this
installation procedure can now be installed.

You may verify your administrative hosts with the command

   # qconf -sh

and you may add new administrative hosts with the command
```

```
   # qconf -ah <hostname>

Please hit <RETURN> >>
```

B. 23. Create the environment variables for use with the grid engine software.
Note – If no cell name was specified during installation, the value of *cell* is default.

If you are using a C shell, type the following command:

```
% source sge-root/cell/common/settings.csh
```

If you are using a Bourne shell or Korn shell, type the following command:

```
$ . sge-root/cell/common/settings.sh
```

Appendix C - Adding a UNIX execution host

On the Qmaster run the following commands in this order to add the new host as an administrative, submit and execution host. The hostname “foster” will be used in the examples.

```
-   root@panic:/ # qconf -ah foster (adds host as an
                             administrative host, needed to be an
                             execution host.)
-   root@panic:/ # qconf -as foster (adds host as an submit host,
                             needed to be an execution host.)
-   root@panic:/ # qconf -ae   This will bring up a template that
                           needs to be edited.
```

```
hostname              template
load_scaling          NONE
complex_values        NONE
user_lists            NONE
xuser_lists           NONE
projects              NONE
xprojects             NONE
usage_scaling         NONE
report_variables      NONE
```

Change the first line from *template* to *foster* (you are currently in the default OS editor, most likely vi)

```
hostname              foster
load_scaling          NONE
complex_values        NONE
user_lists            NONE
xuser_lists           NONE
projects              NONE
xprojects             NONE
usage_scaling         NONE
report_variables      NONE
```

Save the file and the host is added as an execution host.

Add the ip and hostname to the /etc/hosts file on the Qmaster.

On the node that is being added to the grid do the following as root.

Add the following to /etc/services:

```
sge_qmaster     536/tcp                        # Grid Engine qmaster
sge_execd       537/tcp                        # Grid Engine execd
```

Add the user sgeadmin to the system:

First check to see if another user has the uid of 1000. If another user does then that user will need to be migrated to another uid before you can continue. Also be sure to set a password for the sgeadmin user.

```
- useradd -u 1000 -g 14 -d /export/home/sgeadmin -s /bin/ksh -c
  "Grid Engine" -m sgeadmin
- passwd sgeadmin
```

/etc/passwd should have an entry for the sgeadmin user that looks like this

```
sgeadmin:x:1000:14:Grid Engine:/export/home/sgeadmin:/bin/ksh
```

** Note: It is important that the sgeadmin user has uid of 1000, the home directory of /export/home/sgeadmin and the shell of ksh.*

Make the directory /opt/gridware/sge

```
- mkdir –p /opt/gridware/sge
```

Add the following lines to /etc/vfstab:

```
panic:/export   -       /export nfs     1       yes     bg
panic:/opt/gridware/sge -       /opt/gridware/sge       nfs     1       yes
bg
```

Make sure there is a tab between each of the columns. Also make sure that the nfs client is running on the system and then run a mountall from the command prompt. You will see some errors for file system that are already mounted. Do a *df –k* to make sure that the 2 file systems are mounted. You should see something like this

```
.
.
.
panic:/opt/gridware/sge
                     51918952 5703510 45696253    12%    /opt/gridware/sge
panic:/export
                     51918952 5703510 45696253    12%    /export
.
.
.
```

Check to make sure that the Grid Engine Environment Variables are set:

```
- env |grep SGE
```

This should return *SGE_ROOT=/opt/gridware/sge*

If not then you need configure the environment variables.

```
- root@foster:/ # . /opt/gridware/sge/default/common/settings.sh
```

Install the software:

```
- root@foster:/ # cd /opt/gridware/sge
- root@foster:/ # ./inst_sge -x
```

```
Welcome to the Grid Engine execution host installation
------------------------------------------------------

If you haven't installed the Grid Engine qmaster host yet, you must
execute
this step (with >install_qmaster<) prior the execution host
installation.

For a sucessfull installation you need a running Grid Engine qmaster.
It is
also neccesary that this host is an administrative host.

You can verify your current list of administrative hosts with
the command:

   # qconf -sh

You can add an administrative host with the command:

   # qconf -ah <hostname>

The execution host installation will take approximately 5 minutes.

Hit <RETURN> to continue >>


Checking $SGE_ROOT directory
----------------------------

The Grid Engine root directory is:

   $SGE_ROOT = /opt/gridware/sge

If this directory is not correct (e.g. it may contain an automounter
prefix) enter the correct path to this directory or hit <RETURN>
to use default [/opt/gridware/sge] >> /opt/gridware/sge

Your $SGE_ROOT directory: /opt/gridware/sge

Hit <RETURN> to continue >>

Grid Engine cells
-----------------
```

```
Please enter cell name which you used for the qmaster
installation or press <RETURN> to use [default] >> default

Using cell: >default<

Hit <RETURN> to continue >>


Checking hostname resolving
---------------------------

This hostname is known at qmaster as an administrative host.

Hit <RETURN> to continue >>


Local execd spool directory configuration
-----------------------------------------

During the qmaster installation you've already entered a global
execd spool directory. This is used, if no local spool directory is
configured.

Now you can enter a local spool directory for this host.

Do you want to configure a local spool directory
for this host (y/n) [n] >> n


Creating local configuration
----------------------------
sgeadmin@foster.montclair.edu modified "foster.montclair.edu" in
configuration list
Local configuration for host >foster.montclair.edu< created.

Hit <RETURN> to continue >>


execd startup script
--------------------

We can install the startup script that will
start execd at machine boot (y/n) [y] >> y

Installing startup script /etc/rc2.d/S96sgeexecd and
/etc/rc2.d/K02sgeexecd

Hit <RETURN> to continue >>




Grid Engine execution daemon startup
------------------------------------

Starting execution daemon. Please wait ...
```

```
   starting sge_execd

Hit <RETURN> to continue >>


Adding a queue for this host
----------------------------

We can now add a queue instance for this host:

   - it is added to the >allhosts< hostgroup
   - the queue provides 1 slot(s) for jobs in all queues
     referencing the >allhosts< hostgroup

You do not need to add this host now, but before running jobs on this
host it must be added to at least one queue.

Do you want to add a default queue instance for this host (y/n) [y] >>
y

root@foster.montclair.edu modified "@allhosts" in host group list
root@foster.montclair.edu modified "all.q" in cluster queue list

Hit <RETURN> to continue >>


Using Grid Engine
-----------------

You should now enter the command:

   source /opt/gridware/sge/default/common/settings.csh

if you are a csh/tcsh user or

   # . /opt/gridware/sge/default/common/settings.sh

if you are a sh/ksh user.

This will set or expand the following environment variables:

   - $SGE_ROOT         (always necessary)
   - $SGE_CELL         (if you are using a cell other than >default<)
   - $SGE_QMASTER_PORT (if you haven't added the service >sge_qmaster<)
   - $SGE_EXECD_PORT   (if you haven't added the service >sge_execd<)
   - $PATH/$path       (to find the Grid Engine binaries)
   - $MANPATH          (to access the manual pages)

Hit <RETURN> to see where Grid Engine logs messages >>




Grid Engine messages
--------------------

Grid Engine messages can be found at:
   /tmp/qmaster_messages (during qmaster startup)
   /tmp/execd_messages   (during execution daemon startup)

After startup the daemons log their messages in their spool
directories.

   Qmaster:     /opt/gridware/sge/default/spool/qmaster/messages
   Exec daemon: <execd_spool_dir>/<hostname>/messages


Grid Engine startup scripts
---------------------------

Grid Engine startup scripts can be found at:

   /opt/gridware/sge/default/common/sgemaster (qmaster and scheduler)
   /opt/gridware/sge/default/common/sgeexecd (execd)

Do you want to see previous screen about using Grid Engine again (y/n)
[n] >> n

Your execution daemon installation is now completed.
```

Appendix D - Installing Microsoft Services for UNIX

To install N1 Grid Engine onto a Windows system, Microsoft Service for UNIX, SFU, needs to be installed.

This document will walk you through installing SFU with custom install scripts that will create the need user for Grid Engine, open the firewall and enable the needed services. If you are installing SFU manually you will need to follow the other steps listed in the other appendices to fully configure everything that is needed for Grid Engine.

From the Windows PC map the network drive that contains the SFU Software.

Right click on my computer and click on map network drive.

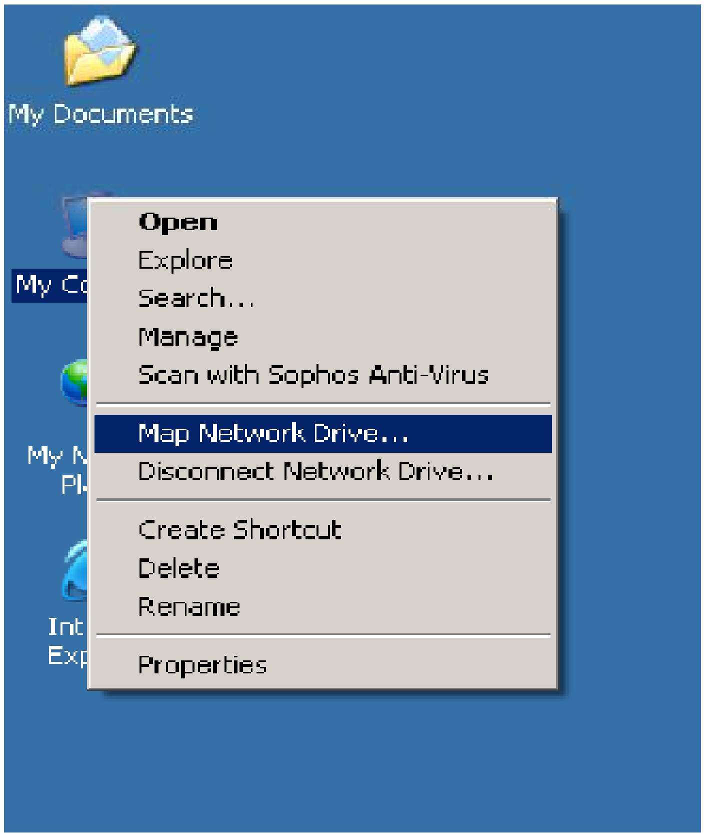
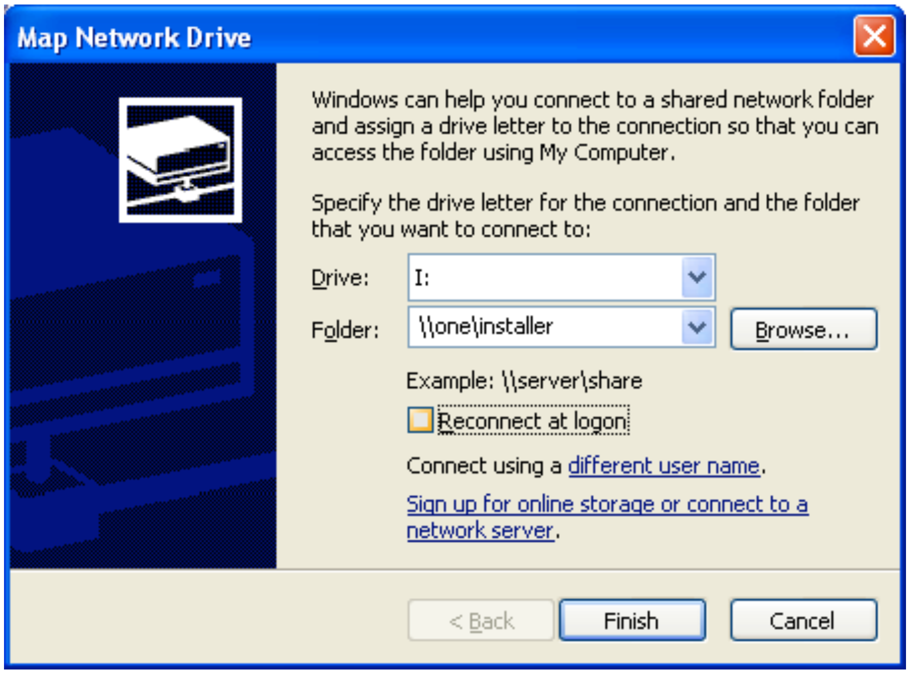


Figure D.1 – Mapping network drive.

Enter the Drive letter you want to map to and the folder that you are mapping.
**NOTE: The folder that you should be mounting is \\panic\installer. If this does not work then try mounting \\130.68.20.236\installer.*


Map Network Drive

Windows can help you connect to a shared network folder and assign a drive letter to the connection so that you can access the folder using My Computer.

Specify the drive letter for the connection and the folder that you want to connect to:

Drive: I:

Folder: \\one\installer

Example: \\server\share

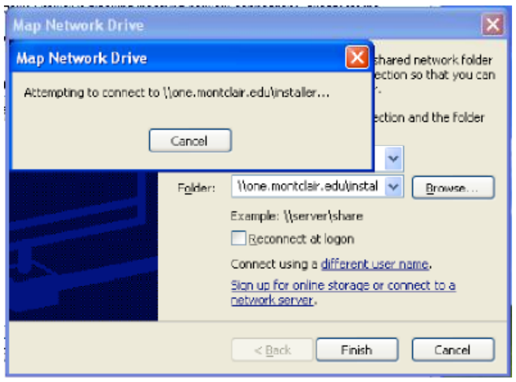
Reconnect at logon

Connect using a different user name.

Sign up for online storage or connect to a network server.

Figure D.2 – Network drive parameters.

The attempt mount the drive.


Map Network Drive

Attempting to connect to \\one.montclair.edu\installer...

Folder: \\one.montclair.edu\instal

Example: \\server\share

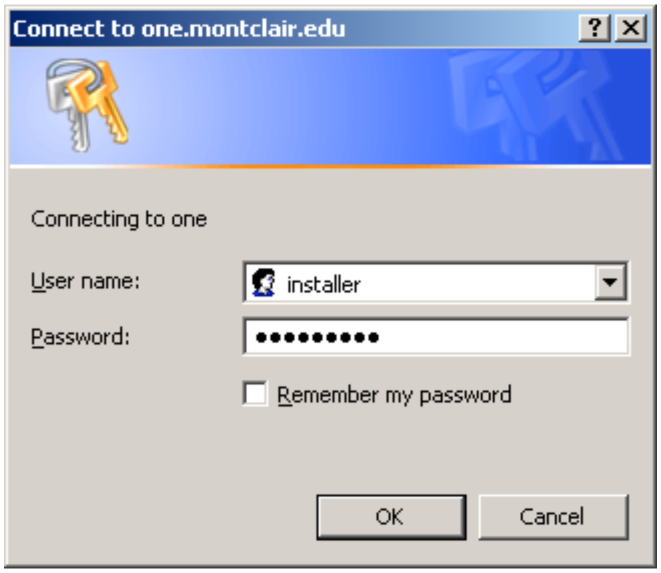
Reconnect at logon

Connect using a different user name.

Sign up for online storage or connect to a network server.

Figure D.3 – Network drive mounting.

You will be then be prompted for the username and password. Enter the username installer and the password.


Connect to one.montclair.edu

Connecting to one

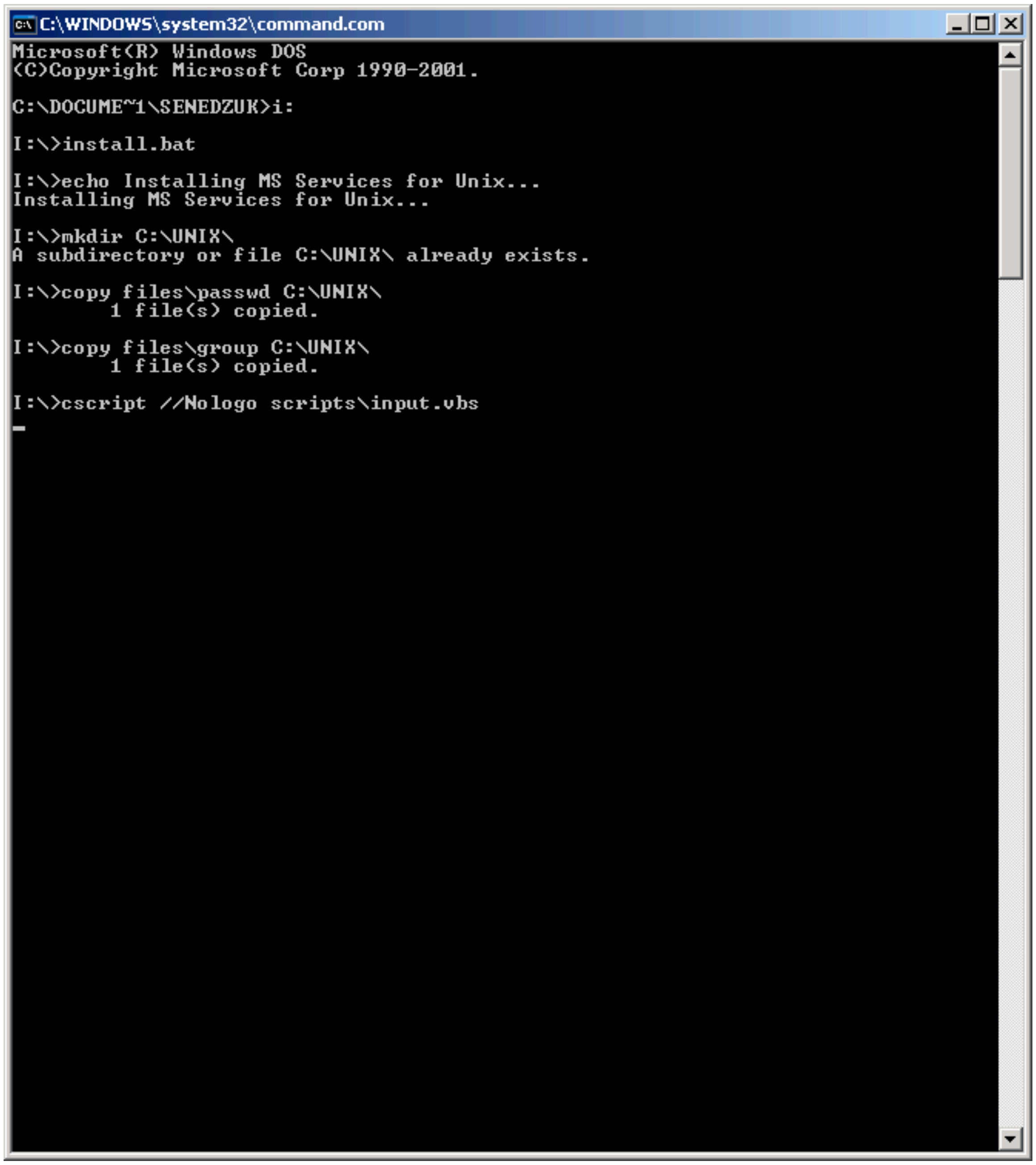
User name: installer

Password: ●●●●●●●●●●

☐ Remember my password

Figure D.4 – Network drive username and password prompt.

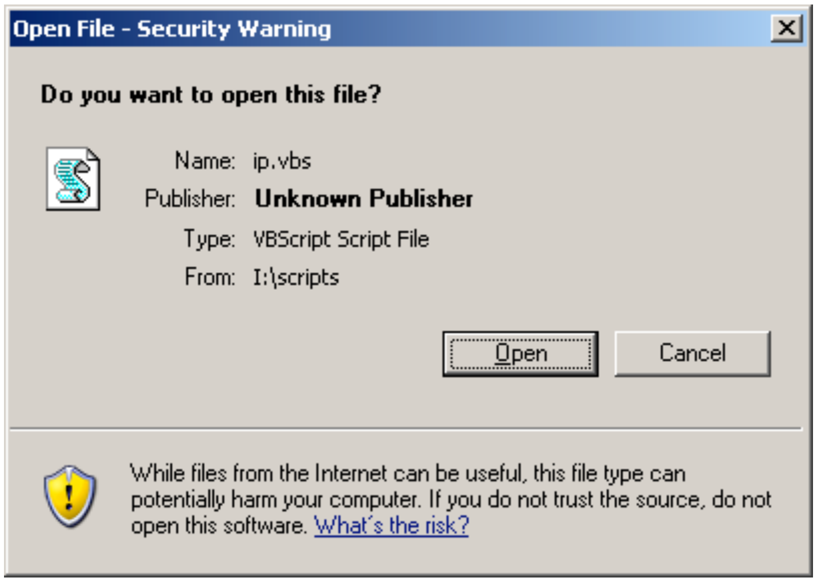
Once the network drive is mapped open the command prompt and go to the network drive. Then from the command prompt run the install.bat script. This will create the needed user, configure the firewall and then start the SFU installation.



```
Microsoft(R) Windows DOS
(C)Copyright Microsoft Corp 1990-2001.

C:\DOCUME~1\SENEDZUK>i:

I:\>install.bat

I:\>echo Installing MS Services for Unix...
Installing MS Services for Unix...

I:\>mkdir C:\UNIX\
A subdirectory or file C:\UNIX\ already exists.

I:\>copy files\passwd C:\UNIX\
        1 file(s) copied.

I:\>copy files\group C:\UNIX\
        1 file(s) copied.

I:\>cscript //Nologo scripts\input.vbs
_
```

Figure D.5 – DOS prompt showing the install.bat script running.

While the install script is running you may get a Security Warning about opening a file such as the one below. If you do receive this warning you must click on "Open" to allow the installation to complete properly. This must be done for every warning that is received for the software to install and function correctly. If "Cancel" is clicked on instead rerun the install script after if completes.


Open File - Security Warning

Do you want to open this file?

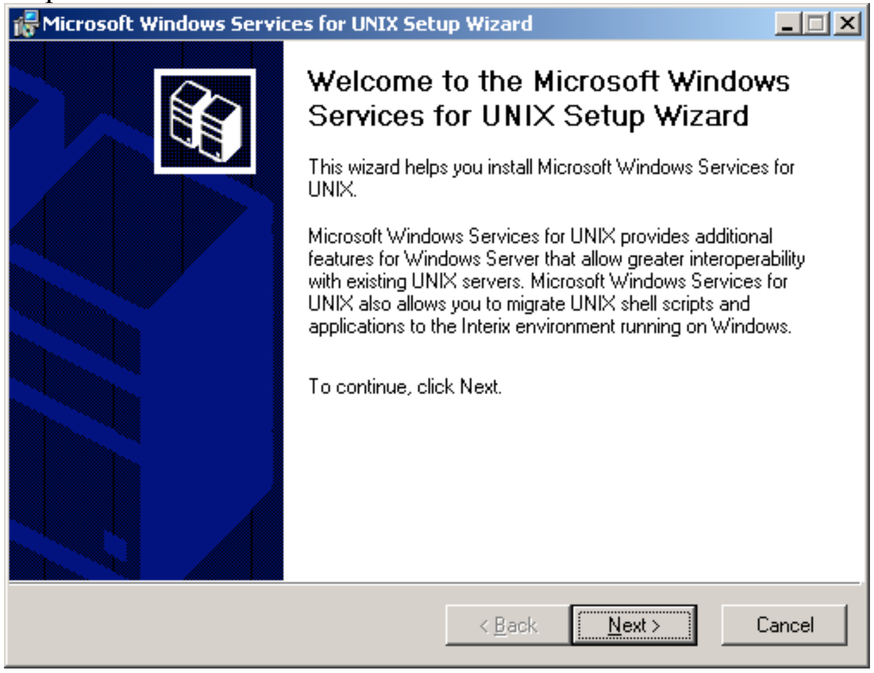
Name: ip.vbs
Publisher: **Unknown Publisher**
Type: VBScript Script File
From: I:\scripts

While files from the Internet can be useful, this file type can potentially harm your computer. If you do not trust the source, do not open this software. What's the risk?

Figure D.6 – Install script security warning.

The following screen shots will provide the settings that are needed for SFU.

Step 1: Click Next


Microsoft Windows Services for UNIX Setup Wizard

Welcome to the Microsoft Windows Services for UNIX Setup Wizard

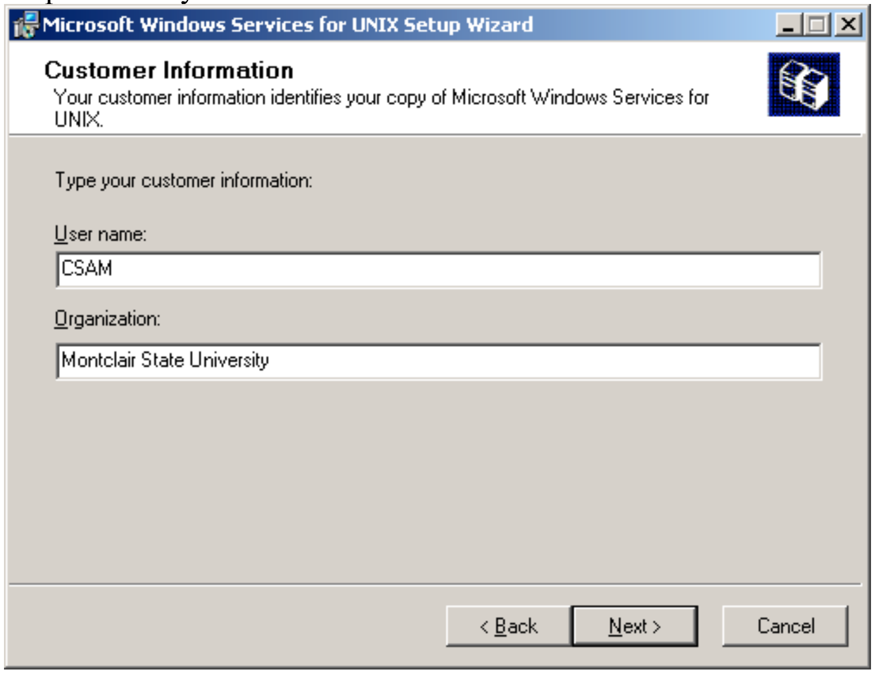
This wizard helps you install Microsoft Windows Services for UNIX.

Microsoft Windows Services for UNIX provides additional features for Windows Server that allow greater interoperability with existing UNIX servers. Microsoft Windows Services for UNIX also allows you to migrate UNIX shell scripts and applications to the Interix environment running on Windows.

To continue, click Next.

Figure D.7 – Welcome screen for SFU install.

Step 2: Enter your information



Microsoft Windows Services for UNIX Setup Wizard

Customer Information

Your customer information identifies your copy of Microsoft Windows Services for UNIX.

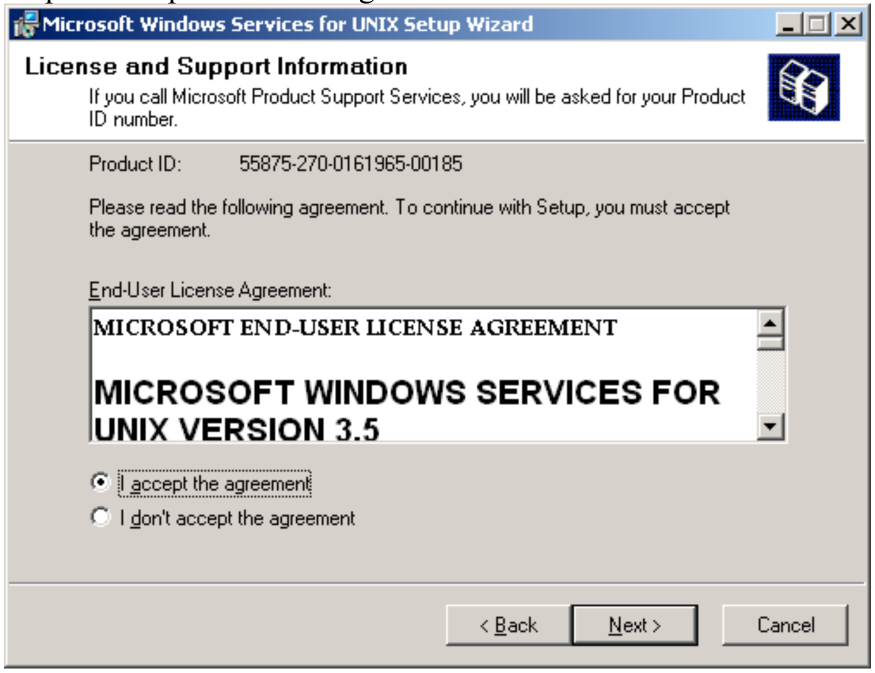
Type your customer information:

User name: CSAM

Organization: Montclair State University

Figure D.8 – SFU Customer information prompt.

Step 3: Accept the license agreement



Microsoft Windows Services for UNIX Setup Wizard

License and Support Information

If you call Microsoft Product Support Services, you will be asked for your Product ID number.

Product ID: 55875-270-0161965-00185

Please read the following agreement. To continue with Setup, you must accept the agreement.

End-User License Agreement:

MICROSOFT END-USER LICENSE AGREEMENT

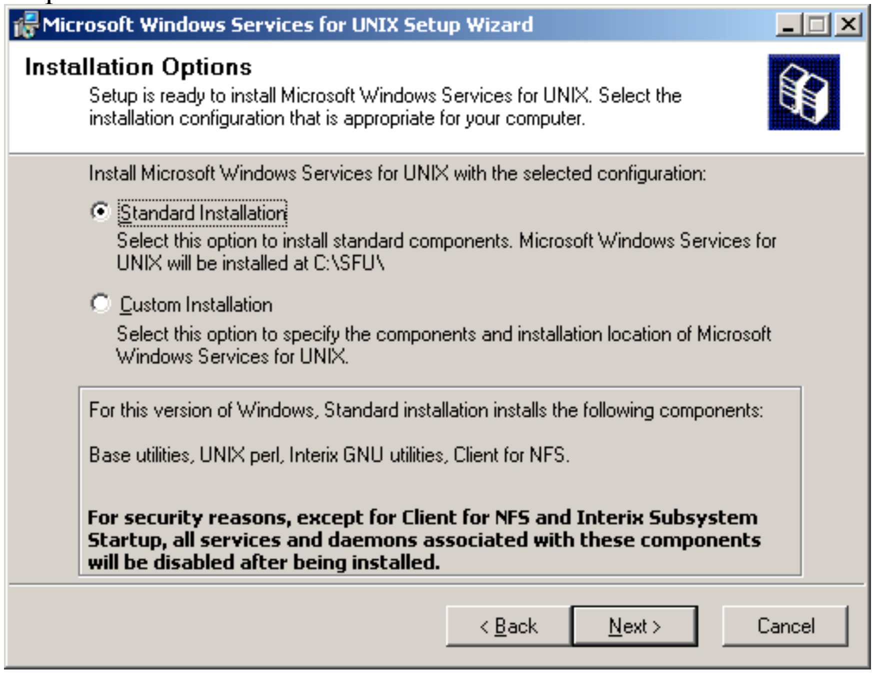
MICROSOFT WINDOWS SERVICES FOR UNIX VERSION 3.5

[✓ I accept the agreement]

I don't accept the agreement

Figure D.9 – SFU license agreement.

Step 4: Select the "Standard Installation"



Installation Options

Setup is ready to install Microsoft Windows Services for UNIX. Select the installation configuration that is appropriate for your computer.

Install Microsoft Windows Services for UNIX with the selected configuration:

[✓ Standard Installation]
Select this option to install standard components. Microsoft Windows Services for UNIX will be installed at C:\SFU\

Custom Installation
Select this option to specify the components and installation location of Microsoft Windows Services for UNIX.

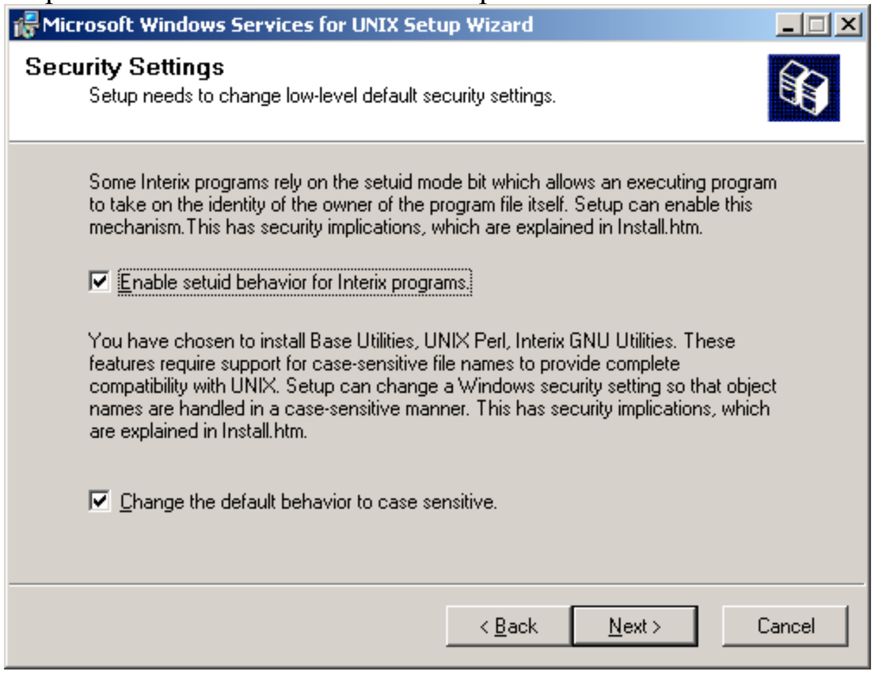
For this version of Windows, Standard installation installs the following components:

Base utilities, UNIX perl, Interix GNU utilities, Client for NFS.

For security reasons, except for Client for NFS and Interix Subsystem Startup, all services and daemons associated with these components will be disabled after being installed.

Figure D.10 – SFU installation options.

Step 5: Place a check mark on both options.



Security Settings

Setup needs to change low-level default security settings.

Some Interix programs rely on the setuid mode bit which allows an executing program to take on the identity of the owner of the program file itself. Setup can enable this mechanism. This has security implications, which are explained in Install.htm.

[✓] Enable setuid behavior for Interix programs.

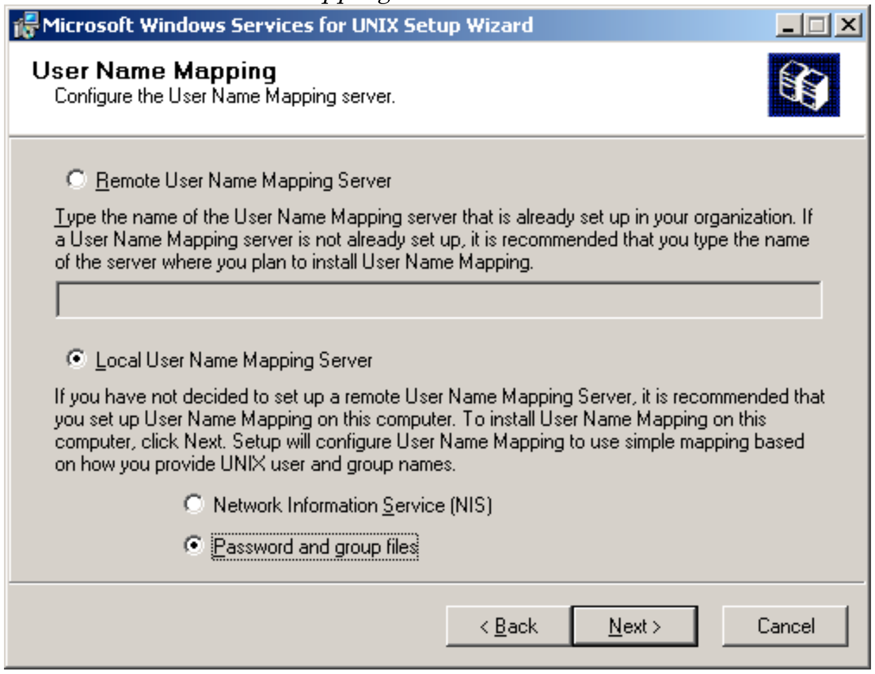
You have chosen to install Base Utilities, UNIX Perl, Interix GNU Utilities. These features require support for case-sensitive file names to provide complete compatibility with UNIX. Setup can change a Windows security setting so that object names are handled in a case-sensitive manner. This has security implications, which are explained in Install.htm.

[✓] Change the default behavior to case sensitive.

Figure D.11 – SFU security settings.

Step 6: Choose "Local User Name Mapping Server" and "Password and group file"

** Note: This configuration is for a local password and group file. If you are using a naming service such as Active Directory then refer to the SFU Documentation for the configuration of the "User Name Mapping".*



Microsoft Windows Services for UNIX Setup Wizard

User Name Mapping

Configure the User Name Mapping server.

( ) Remote User Name Mapping Server

Type the name of the User Name Mapping server that is already set up in your organization. If a User Name Mapping server is not already set up, it is recommended that you type the name of the server where you plan to install User Name Mapping.

(●) Local User Name Mapping Server

If you have not decided to set up a remote User Name Mapping Server, it is recommended that you set up User Name Mapping on this computer. To install User Name Mapping on this computer, click Next. Setup will configure User Name Mapping to use simple mapping based on how you provide UNIX user and group names.

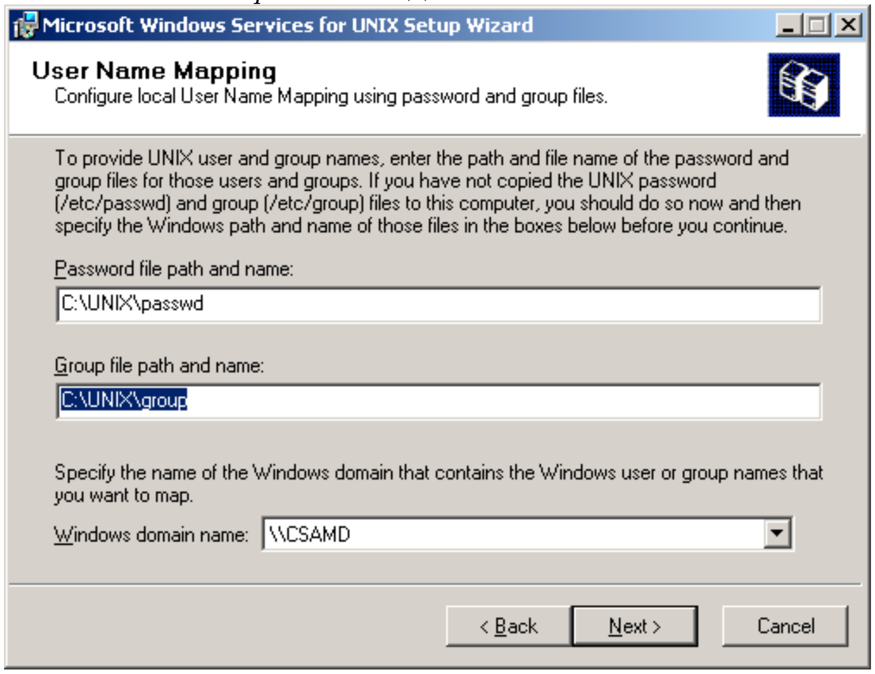
( ) Network Information Service (NIS)

(●) Password and group files

Figure D.12 – SFU username mapping.

Step 7: Set the paths for the password and group files.

** Note: These files are copied over when you run install.bat. If you are installing SFU without the install.bat script you will then manually create the directory and files before you can proceed with this step. See Note (1) at the end of this document.*



Microsoft Windows Services for UNIX Setup Wizard

User Name Mapping

Configure local User Name Mapping using password and group files.

To provide UNIX user and group names, enter the path and file name of the password and group files for those users and groups. If you have not copied the UNIX password (/etc/passwd) and group (/etc/group) files to this computer, you should do so now and then specify the Windows path and name of those files in the boxes below before you continue.

Password file path and name: C:\UNIX\passwd

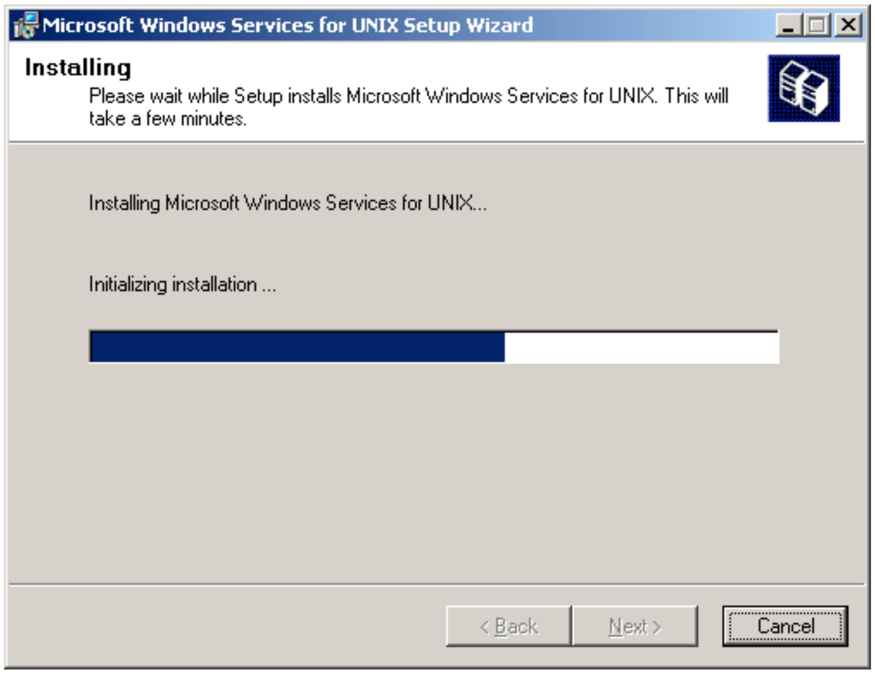
Group file path and name: C:\UNIX\group

Specify the name of the Windows domain that contains the Windows user or group names that you want to map.

Windows domain name: \\CSAMD

Figure D.13 – SFU username mapping configuration.

Step 8: Installation in progress. Depending on the speed of the computer the install may take some time.



Microsoft Windows Services for UNIX Setup Wizard

Installing

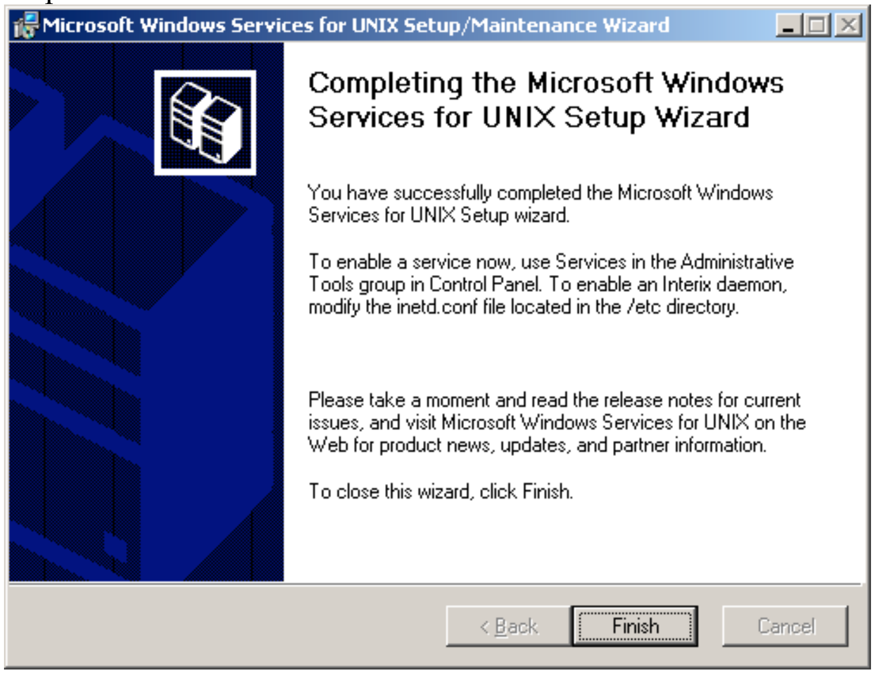
Please wait while Setup installs Microsoft Windows Services for UNIX. This will take a few minutes.

Installing Microsoft Windows Services for UNIX...

Initializing installation ...

Figure D.14 – Mapping network drive.

Step 9: Once the install is done click finish.



Microsoft Windows Services for UNIX Setup/Maintenance Wizard

Completing the Microsoft Windows Services for UNIX Setup Wizard

You have successfully completed the Microsoft Windows Services for UNIX Setup wizard.

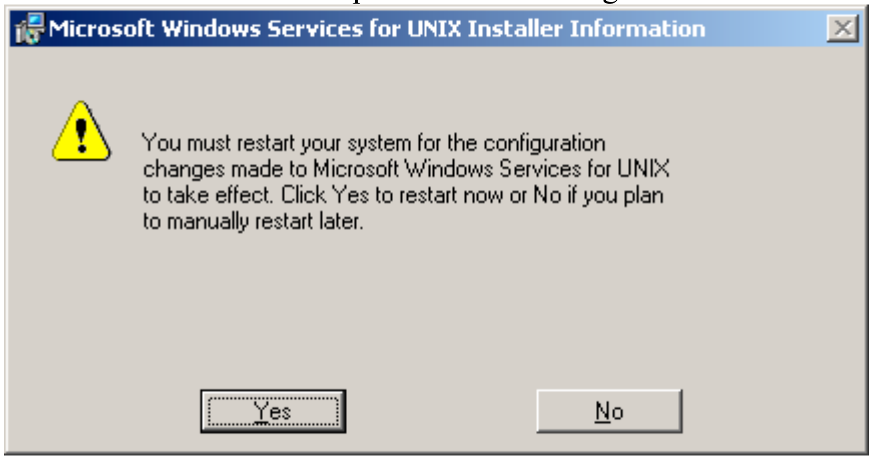
To enable a service now, use Services in the Administrative Tools group in Control Panel. To enable an Interix daemon, modify the inetd.conf file located in the /etc directory.

Please take a moment and read the release notes for current issues, and visit Microsoft Windows Services for UNIX on the Web for product news, updates, and partner information.

To close this wizard, click Finish.

Figure D.15 – SFU installation finishes.

Step 10: You will be prompted to restart the machine now that the install has finished. Choose "No" so that the install script can finish running.


Microsoft Windows Services for UNIX Installer Information

You must restart your system for the configuration changes made to Microsoft Windows Services for UNIX to take effect. Click Yes to restart now or No if you plan to manually restart later.

Figure D.16 – SFU request to reboot system.

The install.bat script should finish running now that the SFU install is complete. If not hit the "Enter" key and the script should continue. The install.bat script finishes by creating some needed directories and copying some files into place. The system will now reboot to complete the install.

NOTE (1)
If you are installing SFU by hand and are using a local password and group files, you must first create a directory to hold the files. It is recommended to create a "UNIX" directory on the root drive of the Windows install. The directory can be placed in any location that you would like if a different location is preferred. Then create 2 text files in the directory, passwd and group. Place the following entries in the files.

passwd
```
root:x:0:0:Super-User:/:/sbin/sh
sgeadmin:x:1000:14:Grid Engine:/export/home/sgeadmin:/bin/ksh
```

group
```
sysadmin::14:
```

Appendix E - Adding a Windows execution host

* *NOTE: Please read the notes section of this document if you have no installed Service for UNIX via the install.bat script.*

On the Qmaster run the following commands in this order to add the new host as an administrative, submit and execution host. The hostname "ri117" will be used in the examples.

```
-  root@panic:/ # qconf -ah ri117 (adds host as an administrative
                                  host, needed to be an execution host.)
-  root@panic:/ # qconf -as ri117 (adds host as an submit host,
                                  needed to be an execution host.)
-  root@panic:/ # qconf -ae   This will bring up a template that
                              needs to be edited.
```

```
hostname              template
load_scaling          NONE
complex_values        NONE
user_lists            NONE
xuser_lists           NONE
projects              NONE
xprojects             NONE
usage_scaling         NONE
report_variables      NONE
```

Change the first line from *template* to *ri117* (you are currently in the default OS editor, most likely vi)

```
hostname              ri117
load_scaling          NONE
complex_values        NONE
user_lists            NONE
xuser_lists           NONE
projects              NONE
xprojects             NONE
usage_scaling         NONE
report_variables      NONE
```

Save the file and the host is added as an execution host.

Add the ip and hostname to the /etc/hosts file on the Qmaster.

*_NOTE: If you are using a pc that does not have a static ip address then you must add the ip address of the qmaster and the pc itself into the hosts file on the pc. The hosts file can be found in the %SystemRoot%\system32\drivers\etc. This has already been done if you installed SFU using the install.bat file._

On the node that is being added to the grid do the following as an Administrator user.

Add the following to *%SystemRoot%\system32\drivers\etc*\services:

```
sge_qmaster     536/tcp                          # Grid Engine qmaster
sge_execd       537/tcp                          # Grid Engine execd
```

**NOTE: This has already been done if you have installed SFU using the install.bat script.*

Now log into the Korn Shell as the sgeadmin user. This can be done 2 different ways.

1) Start → Programs → Windows Service for UNIX → Korn Shell
2) telnet to the system

If the sgeadmin user does not exist on the system see NOTE (2) at the end of this document.

Check to make sure that the Grid Engine Environment Variables are set:

```
            -  env |grep SGE
```

This should return *SGE_ROOT=/opt/gridware/sge*

If not then you need to configure the environment variables.

```
$ . /opt/gridware/sge/default/common/settings.sh
```

Install the software:

```
            -  $ cd /opt/gridware/sge
            -  $ ./inst_sge -x
```

```
Welcome to the Grid Engine execution host installation
------------------------------------------------------

If you haven't installed the Grid Engine qmaster host yet, you must execute
this step (with >install_qmaster<) prior the execution host installation.

For a successful installation you need a running Grid Engine qmaster. It is
also necessary that this host is an administrative host.

You can verify your current list of administrative hosts with
the command:
   # qconf -sh

You can add an administrative host with the command:

   # qconf -ah <hostname>

The execution host installation will take approximately 5 minutes.

Hit <RETURN> to continue >>


Checking $SGE_ROOT directory
----------------------------

The Grid Engine root directory is:

   $SGE_ROOT = /opt/gridware/sge
```

```
If this directory is not correct (e.g. it may contain an automounter
prefix) enter the correct path to this directory or hit <RETURN>
to use default [/opt/gridware/sge] >> /opt/gridware/sge

Your $SGE_ROOT directory: /opt/gridware/sge

Hit <RETURN> to continue >>


Grid Engine cells
-----------------

Please enter cell name which you used for the qmaster
installation or press <RETURN> to use [default] >> default

Using cell: >default<

Hit <RETURN> to continue >>


Checking hostname resolving
---------------------------

This hostname is known at qmaster as an administrative host.

Hit <RETURN> to continue >>


Local execd spool directory configuration
-----------------------------------------

During the qmaster installation you've already entered a global
execd spool directory. This is used, if no local spool directory is
configured.

Now you can enter a local spool directory for this host.

Do you want to configure a local spool directory
for this host (y/n) [n] >> n


Creating local configuration
----------------------------
guser@ri117.montclair.edu modified "ri117.montclair.edu" in configuration
list
Local configuration for host >ri117< created.

Hit <RETURN> to continue >>


Hit <RETURN> to continue >>

----------------------------------------

If you're going to run Windows job's using GUI support, you have
to install the Windows Helper Service
```

```
Do you want to install the Windows Helper Service? (y/n) [n] >> n


Grid Engine execution daemon startup
------------------------------------

Starting execution daemon. Please wait ...
   starting sge_execd

Hit <RETURN> to continue >>


Adding a queue for this host
----------------------------

We can now add a queue instance for this host:

   - it is added to the >allhosts< hostgroup
   - the queue provides 1 slot(s) for jobs in all queues
     referencing the >allhosts< hostgroup

You do not need to add this host now, but before running jobs on this host
it must be added to at least one queue.

Do you want to add a default queue instance for this host (y/n) [y] >>

guser@ri117.montclair.edu modified "@allhosts" in host group list
guser@ri117.montclair.edu modified "all.q" in cluster queue list

Hit <RETURN> to continue >>



Using Grid Engine
-----------------

You should now enter the command:

   source /opt/gridware/sge/default/common/settings.csh
if you are a csh/tcsh user or

   # . /opt/gridware/sge/default/common/settings.sh

if you are a sh/ksh user.

This will set or expand the following environment variables:

   - $SGE_ROOT         (always necessary)
   - $SGE_CELL         (if you are using a cell other than >default<)
   - $SGE_QMASTER_PORT (if you haven't added the service >sge_qmaster<)
   - $SGE_EXECD_PORT   (if you haven't added the service >sge_execd<)
   - $PATH/$path       (to find the Grid Engine binaries)
   - $MANPATH          (to access the manual pages)

Hit <RETURN> to see where Grid Engine logs messages >>
```

```
Grid Engine messages
--------------------

Grid Engine messages can be found at:

   /tmp/qmaster_messages (during qmaster startup)
   /tmp/execd_messages   (during execution daemon startup)

After startup the daemons log their messages in their spool directories.

   Qmaster:     /opt/gridware/sge/default/spool/qmaster/messages
   Exec daemon: <execd_spool_dir>/<hostname>/messages


Grid Engine startup scripts
---------------------------

Grid Engine startup scripts can be found at:

   /opt/gridware/sge/default/common/sgemaster (qmaster and scheduler)
   /opt/gridware/sge/default/common/sgeexecd (execd)

Do you want to see previous screen about using Grid Engine again (y/n) [n] >>
n
```

Your execution daemon installation is now completed.

NOTE (1)
If you did not install SFU via the install.bat script you must complete the following before you can install Sun Grid Engine on the Windows system. The steps must be performed by an Administrator account.

1) Map the Windows Administrator user to the UNIX root user. First you will need to check to make sure that username mapping is turned on. To do this go to Start → Settings → Control Panel → Administrative Tools → Services. Then go to "User Name Mapping" and it should show the status as started and startup type as automatic. If it does not right click on "User Name Mapping" → Properties. Set the startup type to automatic and click start to start the service and then choose "OK". Then run the command to map the users from the Windows command prompt.
   - C:\SFU\common\mapadmin add -wu \\%USERDOMAIN%\Administrator -uu pcnfs\root
2) Open up the Korn Shell by going to Start → Programs → Windows Service for UNIX → Korn Shell. Create the home and Grid Engine directories and then create links from the qmaster's NFS share to them. Then copy the startup script for Grid Engine onto the system and create a link to the script.

   - mkdir /export/home
   - ln -s /net/panic/export/home /export/home
   - mkdir /opt/gridware
   - chmod 755 /opt/gridware
   - ln -s /net/panic/opt/gridware/sge /opt/gridware/sge
   - cp /opt/gridware/sge/default/common/sgeexecd /etc/init.d/sgeexecd
   - ln -s /etc/init.d/sgeexecd /etc/rc2.d/S39sgeexecd

3) While still in the Korn Shell edit the following files.
   - rm /etc/motd
   - uncomment the telnet line in /etc/inetd.conf
   - comment out the lines “echo "DISPLAY=$DISPLAY"” and “else echo "Welcome to the Interix UNIX utilities."”

NOTE (2)

If you did not install SFU via the install.bat script you must add the sgeadmin user to the windows system before installing Sun Grid Engine. The steps must be preformed by an Administrator account.

1) Mount the shared home directories to the system by right clicking on My Computer → Map Network Drive → Browse → NFS Network → Default LAN → panic → /export/home then click “OK”. Make sure that the drive is set to “H:” then click finish. You will be asked if you want to mount the drive, click “Yes”.

2) Right click on My Computer → Manage → Local Users and Groups → Users. Then click on the Actions menu → New User.
   - User name: sgeadmin
   - Full name: Grid Engine Admin
   - Uncheck “User must change password at next logon”
   - Check “Password never expires” and set a password
   - Click “Create”

3) Now right click on the sgeadmin user and click “Properties”. Then click on the “Profile” tab.
   - Check “Connect:” and set the drive letter to “H:”
   - To: \\panic\export\home\sgeadmin
   - Click “OK”

Appendix F - Windows Firewall Settings

**Note: The Windows Firewall is configured during the installation of Services for UNIX via the sge_firewall.vbs and nfs_firewall.vbs script that runs when the install.bat installation script is executed. The steps that are shown here are just for documentation purposes. If SFU was installed manually then these setting need to be configured. Configure these firewall rules after SFU has been installed.*

The Windows Firewall can be configured by first opening the Control Panel and then double clicking on the Windows Firewall.

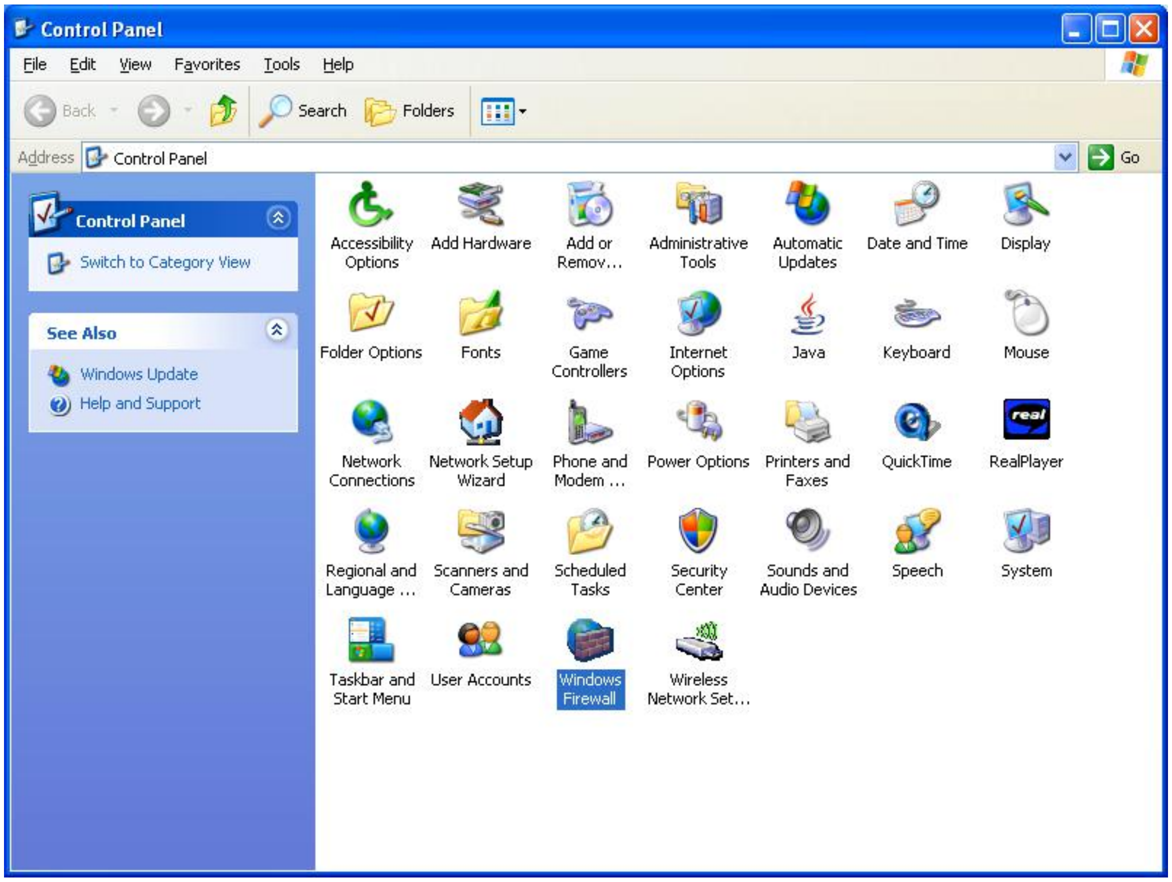
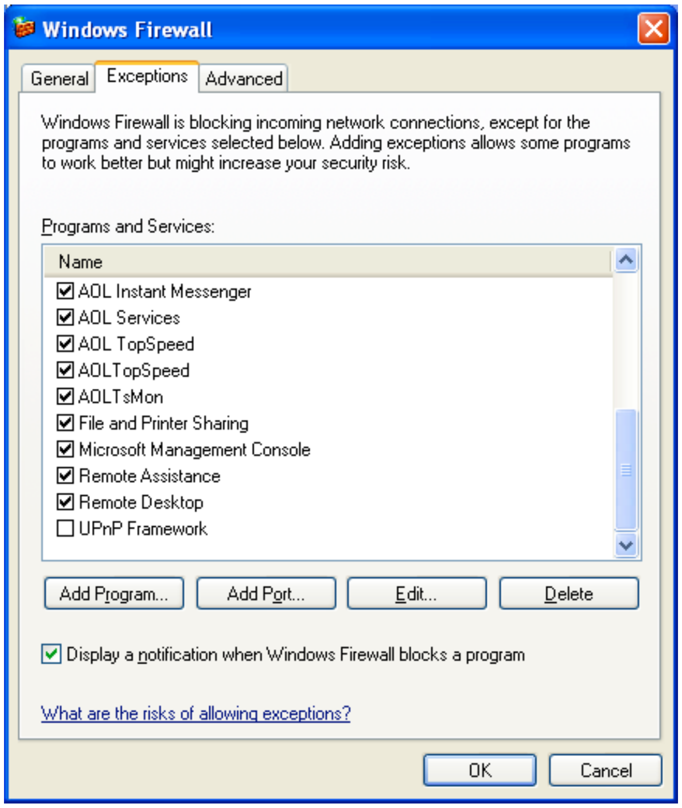


Figure F.1 – Windows control panels window.

This will open up the Windows Firewall Control Panel. Click on the "Exceptions" tab and then on the "Add Port" button.



Windows Firewall is blocking incoming network connections, except for the programs and services selected below. Adding exceptions allows some programs to work better but might increase your security risk.

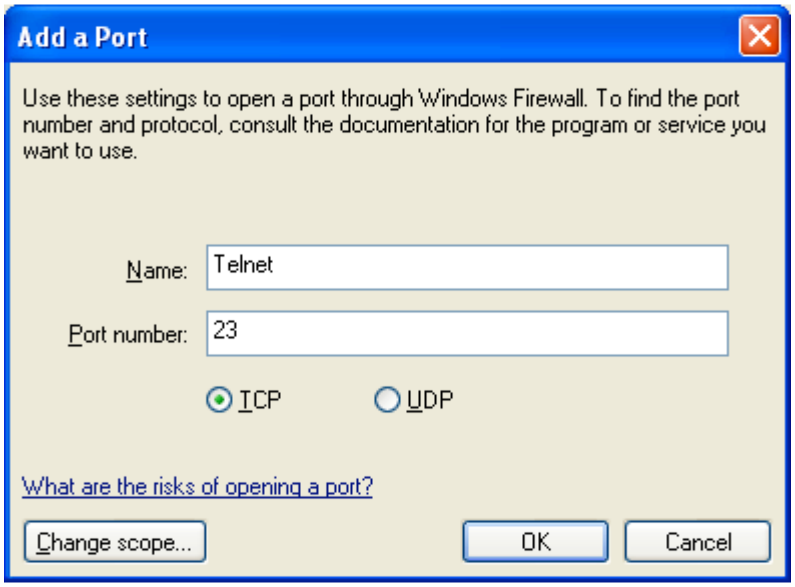
Programs and Services:

Name
☑ AOL Instant Messenger
☑ AOL Services
☑ AOL TopSpeed
☑ AOLTopSpeed
☑ AOLTsMon
☑ File and Printer Sharing
☑ Microsoft Management Console
☑ Remote Assistance
☑ Remote Desktop
☐ UPnP Framework

☑ Display a notification when Windows Firewall blocks a program

Figure F.2 – Windows firewall control panel.

You will need to add 3 different ports to the firewall. These ports are port 23, telnet, port 536, sge_qmaster daemon, and port 537, sge_execd daemon. All ports use TCP.



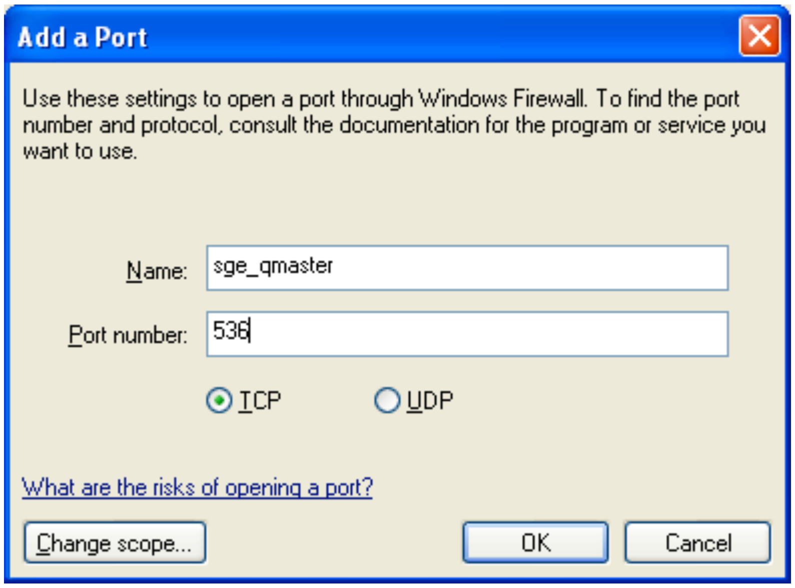
Use these settings to open a port through Windows Firewall. To find the port number and protocol, consult the documentation for the program or service you want to use.

Name: Telnet

Port number: 23

[✓ TCP] UDP

Figure F.3 – Adding telnet to the firewall.


Add a Port

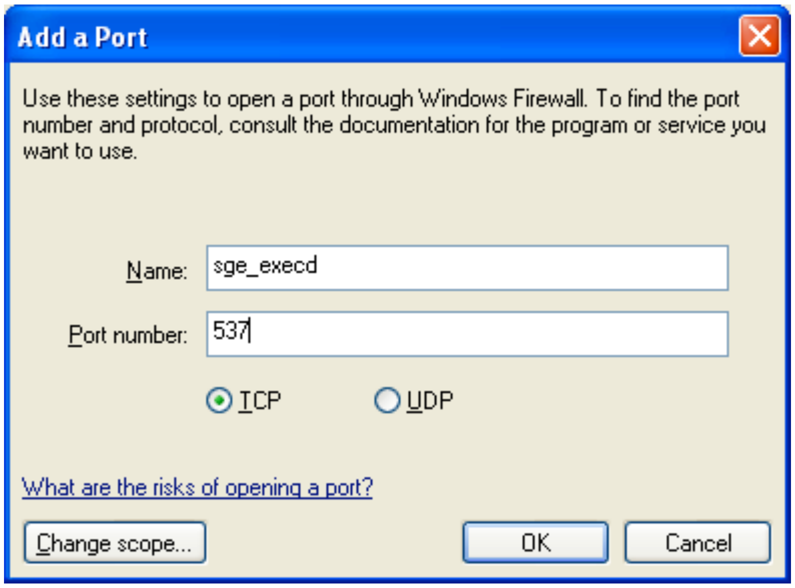
Use these settings to open a port through Windows Firewall. To find the port number and protocol, consult the documentation for the program or service you want to use.

Name: sge_qmaster

Port number: 536

[✓ TCP] UDP

Figure F.4 – Adding sge_qmaster to the firewall.


Add a Port

Use these settings to open a port through Windows Firewall. To find the port number and protocol, consult the documentation for the program or service you want to use.

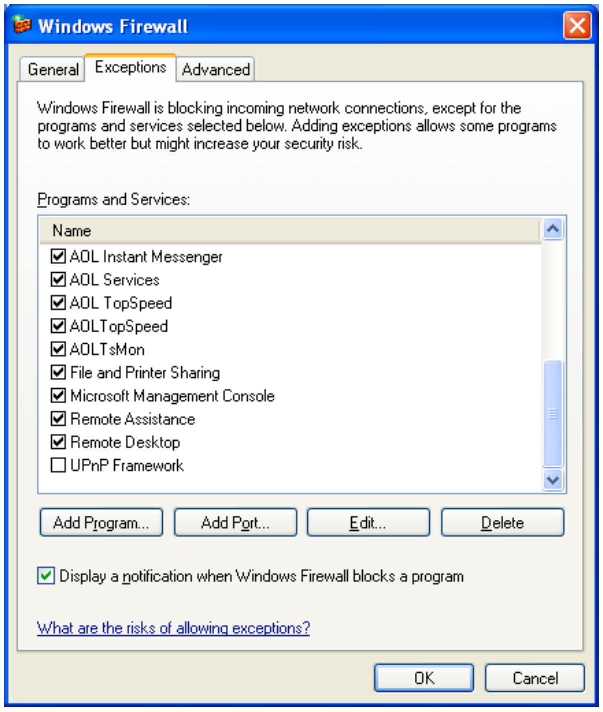
Name: sge_execd

Port number: 537

[✓ TCP] UDP

Figure F.5 – Adding sge_execd to the firewall.

You will also need to add the Client for NFS Program to the firewall. To add the program click on the "Exceptions" tab in the Windows Firewall Control Panel, then on the "Add Program" button.



Windows Firewall

General | Exceptions | Advanced

Windows Firewall is blocking incoming network connections, except for the programs and services selected below. Adding exceptions allows some programs to work better but might increase your security risk.

Programs and Services:

Name
☑ AOL Instant Messenger
☑ AOL Services
☑ AOL TopSpeed
☑ AOLTopSpeed
☑ AOLTsMon
☑ File and Printer Sharing
☑ Microsoft Management Console
☑ Remote Assistance
☑ Remote Desktop
☐ UPnP Framework

Add Program... | Add Port... | Edit... | Delete

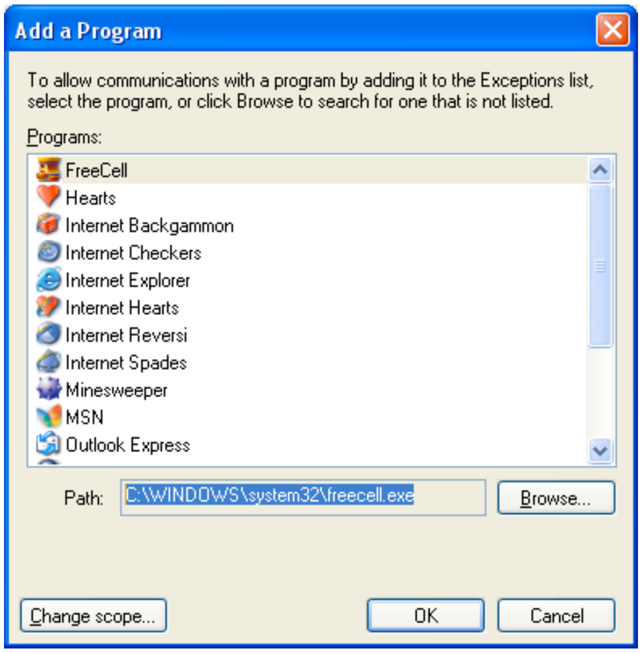
☑ Display a notification when Windows Firewall blocks a program

What are the risks of allowing exceptions?

OK | Cancel

Figure F.6 – Add program via the firewall control panel.

This will bring up a window that will allow you to choose a program from a list or search for a program to be added. Click on the "Browse" button.



Add a Program

To allow communications with a program by adding it to the Exceptions list, select the program, or click Browse to search for one that is not listed.

Programs:

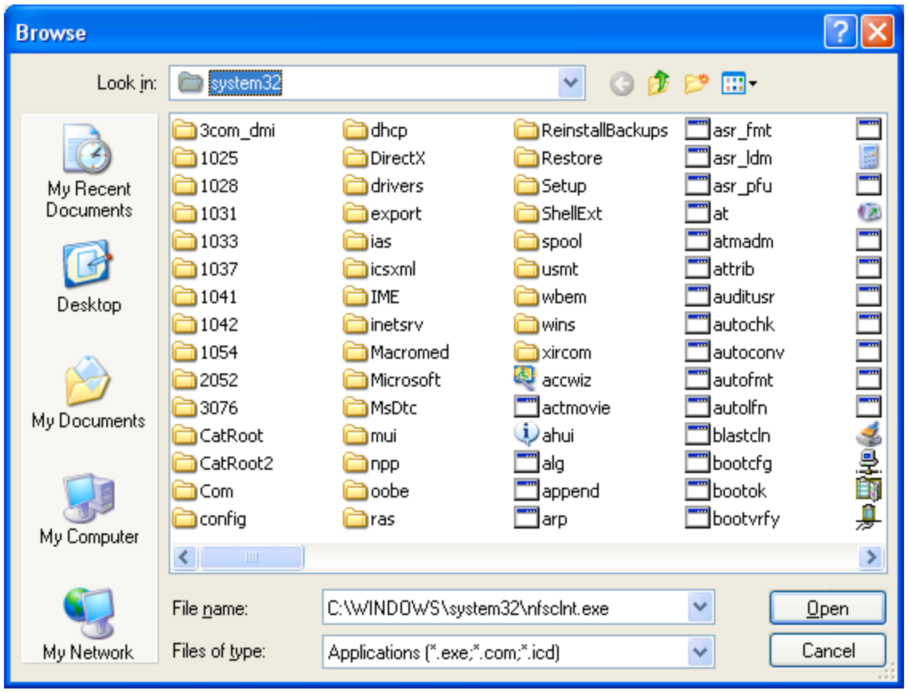
- FreeCell
- Hearts
- Internet Backgammon
- Internet Checkers
- Internet Explorer
- Internet Hearts
- Internet Reversi
- Internet Spades
- Minesweeper
- MSN
- Outlook Express

Path: C:\WINDOWS\system32\freecell.exe | Browse...

Change scope... | OK | Cancel

Figure F.7 – Adding a program to the firewall.

This will bring up a window that will allow you to choose the Client for NFS Program. In the "File Name: " box type *C:\WINDOWS\system32\nfsclnt.exe* and the click "Open"

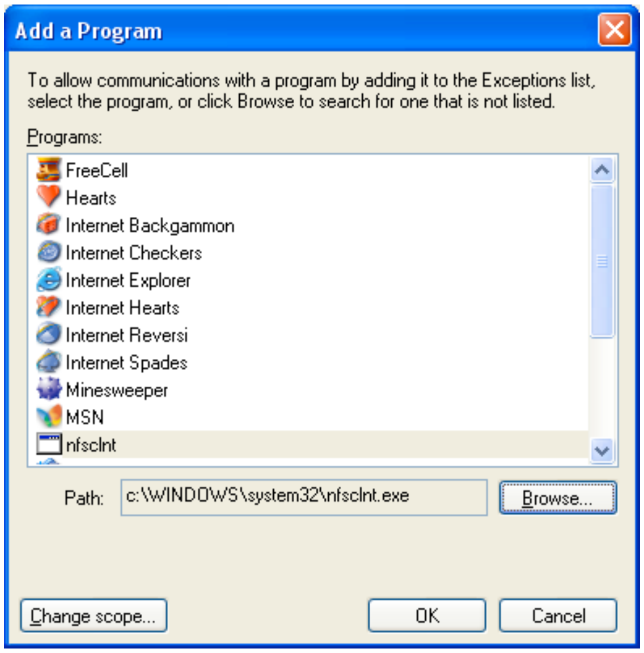


File name: C:\WINDOWS\system32\nfsclnt.exe

Files of type: Applications (*.exe;*.com;*.icd)

Figure F.8 – Browsing Windows for the program to add to the firewall.

This will add the program to the list of programs that can be added to the firewall configuration. Click the "OK" button.



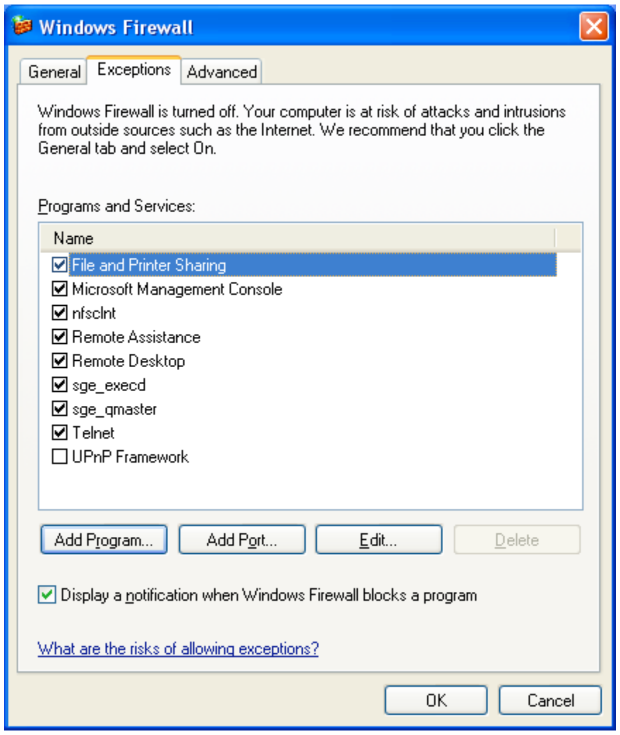
To allow communications with a program by adding it to the Exceptions list, select the program, or click Browse to search for one that is not listed.

Programs: nfsclnt

Path: c:\WINDOWS\system32\nfsclnt.exe

Figure F.9 – Program added to the firewall.

The firewall configuration should now look something like this. It will list the program and ports that were just added.



Windows Firewall is turned off. Your computer is at risk of attacks and intrusions from outside sources such as the Internet. We recommend that you click the General tab and select On.

Programs and Services:

Name
☑ File and Printer Sharing
☑ Microsoft Management Console
☑ nfsclnt
☑ Remote Assistance
☑ Remote Desktop
☑ sge_execd
☑ sge_qmaster
☑ Telnet
☐ UPnP Framework

☑ Display a notification when Windows Firewall blocks a program

What are the risks of allowing exceptions?

Figure F.10 – Updated firewall control panel.

Click “OK” to close the Windows Firewall configuration.

Appendix G - N1GE Backup Configuration

Configuring the automated backup system for N1 Grid Engine.

Configure the backup configuration file with the correct settings for you environment. An example file can be found in $SGE_ROOT/util/install_modules/backup_template.conf. Do not modify this file but make a copy of the file naming it backup.conf in the $SGE_ROOT/util/install_modules directory.

Here is a copy of the backup.conf file after it has been configured with the correct settings.
**NOTE: These are settings that work in the current environment and may not be the optimal settings for all environments.*

backup.conf

```
##################################################
# Autobackup Configuration File
##################################################

# Please, enter your SGE_ROOT here (mandatory)
SGE_ROOT="/opt/gridware/sge"

# Please, enter your SGE_CELL here (mandatory)
SGE_CELL="default"

# Please, enter your Backup Directory here
# After backup you will find your backup files here (mandatory)
# The autobackup will add a time /date combination to this dirname
# to prevent an overwriting!
BACKUP_DIR="/opt/admin/BACKUP/sge_backup"

# Please, enter true to get a tar/gz package
# and false to copy the files only (mandatory)
TAR="true"

# Please, enter the backup file name here. (mandatory)
BACKUP_FILE="backup.tar"
```

In this configuration it was chosen to use /opt/admin/BACKUP/sge_backup as the backup directory. Make sure that the /opt/admin/BACKUP directory exists and is owned by the sgeadmin user.

As the sgeadmin user, go to the $SGE_ROOT directory and run the following command to start an automatic backup,

```
./inst_sge -bup -auto $SGE_ROOT/util/install_modules/backup.conf
```

This command must be run from within the $SGE_ROOT directory as the sgeadmin user otherwise the backup will fail.

The following is a script that will complete a backup of the grid engine configuration and then copy the backup file to another host. The script should be run from cron.

sge_backup.sh

```
#!/bin/sh
#Script to back up the SGE config

. /opt/gridware/sge/default/common/settings.sh

SGE_ROOT=/opt/gridware/sge
TAR=/usr/sbin/tar
SCP=/usr/bin/scp
DATE=`date '+%Y-%m-%d_%H%M%S'`
BACKUP_DIR=/opt/admin/BACKUP
BACKUP_CMD=./inst_sge
BACKUP_CONF=$SGE_ROOT/util/install_modules/backup.conf
LOG_DIR=$BACKUP_DIR/log


cd $SGE_ROOT
$BACKUP_CMD -bup -auto $BACKUP_CONF >> $LOG_DIR/backup_log.$DATE

cd $BACKUP_DIR
$TAR cvf backup_$DATE.tar `ls -lat |head -2 |grep sge |awk {'print $9'}` >>
$LOG_DIR/backup_log.$DATE


$SCP backup_$DATE.tar
senedzuk@pegasus.montclair.edu:/home/students/senedzuk/BACKUP >>
$LOG_DIR/backup_log.$DATE
```

Appendix H - Install Scripts

install.bat

```
echo Installing MS Services for Unix...
mkdir C:\UNIX\

copy files\passwd C:\UNIX\
copy files\group C:\UNIX\

cscript //Nologo scripts\ip.vbs

scripts\useradd.vbs
scripts\sge_firewall.vbs

SFU\setup.exe

scripts\enable_user_name_mapping.vbs

copy files\inetd.conf C:\SFU\etc
copy files\profile.lcl C:\SFU\etc
copy files\sgeexecd C:\SFU\etc\init.d
copy files\S01ln C:\SFU\etc\rc2.d
del C:\SFU\etc\motd

echo sge_qmaster       536/tcp                         # Grid Engine qmaster
>> %SystemRoot%\s
ystem32\drivers\etc\services
echo sge_execd         537/tcp                         # Grid Engine execd >>
%SystemRoot%\sys
tem32\drivers\etc\services

C:\SFU\common\mapadmin add -wu \\%USERDOMAIN%\Administrator -uu pcnfs\root

scripts\nfs_firewall.vbs

mkdir C:\SFU\export
mkdir C:\SFU\opt\gridware

Tools\subinacl /noverbose /file C:\SFU\etc\rc2.d\S01ln
/setowner=Administrator /setprimarygroup=Administrators
/grant=Administrator=F /grant=Administrators=F /grant=everyone=E

shutdown -r -f -t 10
```

S01ln

```
#!/bin/sh

if [ ! -f /etc/rc2.d/S39sgeexecd ]; then
        /bin/ln -s ../init.d/sgeexecd /etc/rc2.d/S39sgeexecd
        /bin/chmod 755 /etc/init.d/sgeexecd
fi

if [ ! -f /export/home ]; then
        /bin/ln -s /net/panic/export/home /export/home
```

```
fi

if [ ! -f /opt/gridware/sge ]; then
        /bin/chmod 755 /opt/gridware
        /bin/ln -s /net/panic/opt/gridware/sge /opt/gridware/sge
fi
```

<u>useradd.vbs</u>

```
On Error Resume Next
Dim cmpname, passwd, username, fullname, objstr
Dim i
Dim checkuser
Const ADS_UF_DONT_EXPIRE_PASSWD = &h10000

Set WshNetwork = CreateObject("WScript.Network")
cmpname = WshNetwork.ComputerName 'getting the machine name

username="sgeadmin"
passwd="password_has_been_removed"
fullname="Grid Engine Admin"

checkuser = 1
Set objComputer = GetObject("WinNT://" & cmpname)
objComputer.Filter = Array("User")
For Each objUser In objComputer
'To check whether the same user exists or not by traversing through all the
existing users
  If (objUser.Name = username) And (checkuser = 1) Then
  checkuser = 0
  End If
Next
' Add the user and also include it into the Users group, if the user doesn't
exist
If checkuser = 1 Then
  Set colaccounts = GetObject("WinNT://" & cmpname & ",computer")
  Set objUser = colaccounts.Create("user", username)
  objUser.FullName = fullname
  objUser.SetPassword passwd
  objUser.Put "homeDirDrive", "H"
  objUser.Put "homeDirectory", "\\panic\export\home\sgeadmin"
  objUser.SetInfo

  Set objUser = GetObject("WinNT://" & cmpname & "/" & username & ",User")
  intUAC = objUser.Get("UserFlags")
  objUser.Put "UserFlags", intUAC XOR ADS_UF_DONT_EXPIRE_PASSWD
  objUser.SetInfo


  objstr = "/" & username & ",user"
  Set objGroup = GetObject("WinNT://" & cmpname & "/Users,group")
  Set objUser2 = GetObject("WinNT://" & cmpname & objstr)
  objGroup.Add (objUser2.ADsPath)
End If
wscript.quit
```

sge_firewall.vbs

```
Set objFirewall = CreateObject("HNetCfg.FwMgr")
Set objPolicy = objFirewall.LocalPolicy.CurrentProfile

Set objPort = CreateObject("HNetCfg.FwOpenPort")
objPort.Port = 23
objPort.Name = "Telnet"
objPort.Enabled = TRUE
Set colPorts = objPolicy.GloballyOpenPorts
errReturn = colPorts.Add(objPort)

Set objPort = CreateObject("HNetCfg.FwOpenPort")
objPort.Port = 536
objPort.Name = "sge_qmaster"
objPort.Enabled = TRUE
Set colPorts = objPolicy.GloballyOpenPorts
errReturn = colPorts.Add(objPort)


Set objPort = CreateObject("HNetCfg.FwOpenPort")
objPort.Port = 537
objPort.Name = "sge_execd"
objPort.Enabled = TRUE
Set colPorts = objPolicy.GloballyOpenPorts
errReturn = colPorts.Add(objPort)
```

enable_user_name_mapping.vbs

```
strComputer = "."
'Set WshNetwork = CreateObject("WScript.Network")
'strComputer = WshNetwork.ComputerName 'getting the machine name
Set objWMIService = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")

Set colServiceList = objWMIService.ExecQuery _
    ("Select * from Win32_Service where Name = 'mapsvc'")

For Each objService in colServiceList
        errReturnCode = objService.ChangeStartMode("Automatic")
        objService.StartService()
        Wscript.Sleep 5000
Next
```

nfs_firewall.vbs

```
Set objFirewall = CreateObject("HNetCfg.FwMgr")
Set objPolicy = objFirewall.LocalPolicy.CurrentProfile

Set objApplication = CreateObject("HNetCfg.FwAuthorizedApplication")
objApplication.Name = "Client for NFS"
objApplication.IPVersion = 2
objApplication.ProcessImageFileName = "%SystemRoot%\system32\nfsclnt.exe"
objApplication.RemoteAddresses = "*"
objApplication.Scope = 0
objApplication.Enabled = True
Set colApplications = objPolicy.AuthorizedApplications
colApplications.Add(objApplication)
```

Appendix I - Application source code

prime.java

```
//#Org: WWW
//#Modified: Nick Senedzuk, 2.11.07 - made all numbers double
//#Modified: Dave Puliti, 2.28.07 - added command line args

public class prime
{
  // This method tests whether a given number is prime or not.
  public static boolean isPrime ( double num )
  {
    boolean prime = true;
    double limit = (double) Math.sqrt ( num );

    for ( double i = 2; i <= limit; i++ )
    {
      if ( num % i == 0 )
      {
        prime = false;
        break;
      }
    }

    return prime;
  }

  public static void main ( String[] args )
  {

    double MIN = 2;
    double MAX = 1000000;

    if (( args.length == 2 ) && ( Double.parseDouble( args[0]) <
Double.parseDouble( args[1]) ))
    {
          MIN = Double.parseDouble( args[0] );
          MAX = Double.parseDouble( args[1] );
    }
    else
    {
       System.out.println( "Usage: <class_name> <min> <max>");
       return;
    }

    for ( double i = MIN; i <= MAX; i++ )
    {
      if ( isPrime ( i ) )
        System.out.println ( i );
    }
  }

}
```

<u>prime.sh</u>

```
#!/bin/sh
#
#
# request "bin/sh" as shell for job
#$ -S /bin/sh
#
#
# make job restartable
#$ -r y

if [ -x /usr/bin/java ]; then

   JAVA=/usr/bin/java

elif [ -x /dev/fs/C/WINDOWS/system32/java.exe ]; then

   JAVA=/dev/fs/C/WINDOWS/system32/java.exe

elif [ -x /dev/fs/C/WINDOWS/SYSTEM32/java.exe ]; then

   JAVA=/dev/fs/C/WINDOWS/SYSTEM32/java.exe

else

   JAVA=/dev/fs/C/WINNT/system32/java.exe

fi

HOME=/export/home/sgeadmin
CLASSPATH=./src

cd $HOME

$JAVA  -classpath $CLASSPATH prime $1 $2
```

<u>generateQsubDistance.java</u>

```
/**
*
*       generateQsubDistance.java
*
*       Application that computes the best bands that minimize the distance
*       between one vector and a group of vectors (i.e. between the ground
truth
*       target and the sampled truth vectors
*
*@author Stefan A. Robila
*@version 1.0
*@date 4/05/2007
*/


public class generateQsubDistance
{
  public static int vecsize=0;
  public static int jobs=0;
  public static String filename="Blah";


  //note that jobs is the power of two for jobs
  public static void generateJobs() {
    String tail_of_zeros="";

        int i=vecsize-jobs;
        while (i>0) {
                tail_of_zeros+='0';
                i--;
        }

        //create the intervals
        String start_all="";
        String end_all="";
        i=jobs;
        while (i>0) {
                start_all+='0';
                end_all+='1';
                i--;
        }

        String cStart=start_all;
        while (!cStart.equals(end_all)) {


                //get the next band combination
                char cdata [] = cStart.toCharArray();
                int j = cStart.length()-1;
                boolean done = false;
                while (j>=0 && !done) {
                         if (cdata[j] == '1'){
                                cdata [j] = '0';
                                j--;
                     }
                         else {
                           cdata[j] = '1';
                                done = true;
                         }
                 }
                 String cEnd = new String(cdata);

                 System.out.println("qsub bestDistance.sh "+filename+"
"+(cStart+tail_of_zeros)+" "+(cEnd+tail_of_zeros));
                 cStart = cEnd;
        }


  }



  /**
  *
  * The main portion of the application
  *
  **/
  public static void main ( String[] args )
  {


    //The application is started by three arguments
         // filename and string interval
    if ( args.length == 3 )
    {
            filename = args[0];
            try {
             vecsize = Integer.parseInt(args[1]);
             jobs = Integer.parseInt(args[2]);
                 } catch (NumberFormatException e) {
                       System.err.println("Caught NumberException: "
                       + e.getMessage()+": not an integer number");
                               return;
                 }

                  System.out.println("Size: "+vecsize+", jobs: "+jobs);
                  if (vecsize < jobs) {
                     System.out.println("Incorrect use, num bands smaller
than power num intervals");
                         return;
                  }
                  generateJobs();
    }
    else
    {
       System.out.println( "Usage: <class_name> <file_name> (147) <num_bands>
<num_jobs>");
    }
  }


}
```

bestDistance.java

```
/**
*
*       bestDistance.java
*
*       Application that computes the best bands that minimize the distance
*       between one vector and a group of vectors (i.e. between the ground
truth
*       target and the sampled truth vectors
*
*@author Stefan A. Robila
*@version 1.0
*@date 4/05/2007
*/


import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.io.Reader;
import java.io.StreamTokenizer;
import java.util.Date;

public class bestDistance
{
  public static final int VECSIZE=38;
  public static final int NUMVECTORS=5;

  private static double [][] x;



  public static void computeBestDistance(String  st, String en) {
    int i,j;
         String current = st;
         double min_distance = 10000;
         String best_distance_bands="";
         double current_distance=0;
         Date time = new Date();

         long t1=System.currentTimeMillis();
         while (en.compareTo(current)>=0) {
            int numbands=0;

       //compute the norms (used in the lower part of the distance)
                 for (j=0; j<VECSIZE; j++) {
                                       if (current.charAt(j)=='1')
                                          numbands++;
                 }
                // System.out.println(numbands);

                 if (numbands >= 2)
                 {
                    double [] distances = new double [NUMVECTORS];
                         double [] norms        = new double [NUMVECTORS];

                 for (i=0; i<NUMVECTORS; i++){
                                       norms[i]=0;
                                       distances[i]=0;
                         }

                         //compute the norms (used in the lower part of the
distance)
                    for (j=0; j<VECSIZE; j++) {
                                       if (current.charAt(j)=='1'){
                                          norms[0] += x[0][j]*x[0][j];
                                       for (i=1; i<NUMVECTORS; i++){
                                               norms[i] += x[i][j]*x[i][j];
                                                       distances[i] +=
x[0][j]*x[i][j];
                                               }
                                       }
                    }
```

```
                         current_distance = 0;
                         norms[0]=Math.sqrt(norms[0]);
                 for (i=1; i<NUMVECTORS; i++){
                                norms[i]=Math.sqrt(norms[i]);

distances[i]=distances[i]/(norms[0]*norms[i]);
                                current_distance += Math.acos(distances[i]);
                         }
                         current_distance /= (NUMVECTORS-1);

                        // see if this is a better band combination
                        if (current_distance < min_distance){
                                min_distance = current_distance;
                                best_distance_bands = current;
                        }

                 }
                        //get the next band combination
                        char cdata [] = current.toCharArray();
                        j = VECSIZE-1;
                        boolean done = false;
                        while (j>=0 && !done) {
                         if (cdata[j] == '1'){
                                cdata [j] = '0';
                                j--;
                         }
                         else {
                           cdata[j] = '1';
                                done = true;
                         }
                   }
                   current = new String(cdata);
         }
         long t2=System.currentTimeMillis();
         System.out.println(min_distance+" "+best_distance_bands);
         System.out.println("Elapsed: "+(t2-t1));
        }


  /**
  *
  * The main portion of the application
  *
  **/
  public static void main ( String[] args )
  {

    //The application is started by three arguments
         // filename and string interval
    if ( args.length == 3 )
    {
            String filename = args[0];
        String st=args[1];
        String en=args[2];


                  if (st.length() != VECSIZE || en.length() != VECSIZE){
```

```
                                System.err.println("The size of the bit
strings must be EXACTLY "+VECSIZE+" "+ st.length()+" "+en.length());
                                return;
                  }


                  //initialize the vectors

                  x = new double[NUMVECTORS][VECSIZE];



                  //test if the search interval is correct
        if (en.compareTo(st)>=0)
        {
                                //Need to read vector data here
                                try {
                           readfileinVectors(filename);
                                } catch (IOException e) {
                                        System.err.println("Caught
IOException: "
                        + e.getMessage()+": "+en);
                                }
            computeBestDistance(st,en);
        }
        else
        {
            System.out.println( "First string must be less than the
second!");
        }
    }
    else
    {
       System.out.println( "Usage: <class_name> <file_name> <min> <max>");
    }
  }


    static void readfileinVectors(String filename) throws IOException {
                        Reader r = new BufferedReader(new
FileReader(filename));
                StreamTokenizer stok = new StreamTokenizer(r);
               stok.parseNumbers();
                double sum = 0;
                stok.nextToken();

                        int count=0;
                        int v = 1;
                        boolean done=false;
                        //read data
                while (stok.ttype != StreamTokenizer.TT_EOF && !done) {
                if (stok.ttype == StreamTokenizer.TT_NUMBER) {
                                x[v][count] = stok.nval;
                                        count=(count+1)%VECSIZE;
                                        if (count==0){
                                                v++;
                                                if (v >= NUMVECTORS)
```

```
                                                        done=true;
                                          }
                                }
                                else
                                System.out.println("Nonnumber: " +
stok.sval);
                stok.nextToken();
                }
                        for (count = 0; count<VECSIZE; count++)
                        {
                           x[0][count]=0;
                                for (v=1;v<NUMVECTORS;v++)
                                  x[0][count]+=x[v][count];
                                x[0][count]/=NUMVECTORS;
                        }
        }
}
```

bestDistance.sh

```
#!/bin/sh
#
#
# request "bin/sh" as shell for job
#$ -S /bin/sh
#
#
# make job restartable
#$ -r y

if [ -x /usr/bin/java ]; then

   JAVA=/usr/bin/java

elif [ -x /dev/fs/C/WINDOWS/system32/java.exe ]; then

   JAVA=/dev/fs/C/WINDOWS/system32/java.exe

elif [ -x /dev/fs/C/WINDOWS/SYSTEM32/java.exe ]; then

   JAVA=/dev/fs/C/WINDOWS/SYSTEM32/java.exe

else

   JAVA=/dev/fs/C/WINNT/system32/java.exe

fi

HOME=/export/home/sgeadmin
CLASSPATH=./distance

PROJECT_DIR=./distance

cd $HOME

$JAVA  -classpath $CLASSPATH bestDistance $PROJECT_DIR/$1 $2 $3
```