TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

DIPLOMARBEIT

University Campus Grid Computing

ausgeführt am Institut für
Analysis und Scientific Computing
der Technischen Universität Wien

unter der Anleitung von
Prof. Dr. Christoph Überhuber

durch

Philipp Kolmann
Matrikelnummer: 9825804

Untere Feldstraße 80
A-2823 Pitten.

Pitten, am 25. August 2005 _____

Vorwort

Die meisten Universitäten bieten heute ihren Studenten Arbeitsplätze an, an denen sie ihre tägliche Arbeit machen können. Die meisten Computer stehen in der Nacht, am Wochenende und in Ferialzeiten still. Das Ziel dieser Diplomarbeit ist die Evaluierung der Möglichkeiten diese Ressourcen zu verwenden und welche Software dafür in Frage kommt.

Diese Diplomarbeit präsentiert eine Auswahl an vorhandener Gridsoftware welche frei im Internet verfügbar ist. Die meisten Pakete sind quelloffene Software oder es gibt zumindest die Software für Linux.

Der Hauptteil dieser Diplomarbeit liefert eine Beschreibung der Installation eines „Remote Boot" Servers mit Debian GNU/Linux. Von diesem Server bekommen die PCs ihr Betriebssystem und können dann als Rechenknoten in der Gridumgebung dienen.

Durch diese Umgebung ist es erstmals möglich richtiges Grid-Computing an der TU Wien durchzuführen, da es keine Voraussagen über die Verfügbarkeit der Maschinen gibt und die Hardwarekonfiguration der Maschinen untereinander sehr unterschiedlich sein kann. Daher werden sonst ungenutzte Ressourcen im Leistungsbereich eines Supercomputers verfügbar gemacht, welche ohne ein solches Setup ungenutzt wären.

Alle in dieser Diplomarbeit beschriebenen Techniken wurden erfolgreich in einer Gridumgebung an der Technischen Universität Wien – WINZIG – eingebaut und werden von einigen wissenschaftlichen Gruppen als zusätzliche Rechenkapazität genutzt.

Diese Umgebung kann als Vorbild für andere österreichische und ausländische Universitäten dienen, um derzeit noch brach liegende Ressourcen zu aktivieren.

Preface

Today most universities provide their students with access to computers for their daily work. Most of these computers are idle during night time, on weekends, and during holidays. The aim of this diploma thesis is to check if there are possibilities to make use of the resources and the software in question.

This thesis presents a selection of available grid software which is freely available on the internet. Most of these packages are open source or at least available for the Linux operating system.

The main part of this thesis provides a description of how to set up a remote boot server using Debian GNU/Linux that can handle PCs obtaining their operating system from this machine and then serve as computation nodes in a grid environment.

For the first time this environment makes true grid computing possible at the Vienna University of Technology since no predictions about the availability of nodes can be made and the hardware configuration of the nodes may differ. Thus, unused resources in the range of a super computer are made accessible which wouldn't be possible without such a setup.

All techniques described in this thesis have been successfully implemented in a grid environment that runs at the Vienna University of Technology—called WINZIG—and serves several scientific groups as an additional computational resource.

This environment can serve as an example and prototype to be used by other Austrian and foreign universities to utilize their idle resources.

Acknowledgements

First of all, I would like to express my deepest thankfulness to my family—my mother, my father, and my aunt—for always supporting me and for giving me guidance.

I want to deeply thank the advisor of this thesis, Christoph Überhuber, for his encouragement and support.

I want to thank Gerhard Schmitt for providing me with the opportunity to work with the Information Technology Services (ZID).

I want to thank Peter Berger and Martin Rathmayer for always supporting my work and giving me all the opportunities to make this work possible.

I want to thank Wolfgang Kleinert and Johannes Demel for their cooperation and support.

I want to say special thanks to my girlfriend Helga who encouraged me to finish this thesis and did a lot of proof reading.

Also, I want to thank my colleague Irmgard Husinsky for her perfect proof reading and helping me with the graphics.

Last, but not least, I want to thank my friends for their friendship and support.

PHILIPP KOLMANN

Contents

Chapter 1

Introduction

Inspired by the electrical power grids pervasiveness, ease of use, and reliability, computer scientists in the mid-1990s began exploring the design and development of an analogous infrastructure in information technology called the computational (power) grid [9] for (wide-area) parallel and distributed computing. The motivation for computational grids was initially driven by large scale, resource (computational and data) intensive scientific applications that require more resources than single computers (PCs, workstations, supercomputers, or clusters) could provide in a single administrative domain.

A computational grid enables the sharing, selection, and aggregation of a wide variety of geographically distributed resources including supercomputers, storage systems, data sources, and specialized devices owned by different organizations for solving large-scale resource intensive problems in science, engineering, and commerce [6].

This diploma thesis deals with the usability of workstation equipment to do medium to high performance computing. The Vienna University of Technology (TU Wien) provides several computer labs for students that are mostly unused during the night, on weekends, and during holidays. This thesis describes a way to use these resources during their unused time as a computational grid. The main difference to a computational cluster is that this setup is not dedicated to a homogeneous cluster computing but derived from several different PCs. Also in this setup it is not ensured that every time the same PCs are available.

Prerequisites. This thesis deals intensively with the Linux operating system. Accordingly, a certain knowledge and familiarity with Linux or Unix is strongly recommended. No references to the basic functions of a Unix system (like copying, moving, and editing files) are included.

Document Conventions. This document provides information through short `bash` shell commands. Here are the conventions used:

```
# command in root account
$ command in user account
... description of action
```

These shell command examples use `PS2=" "`.

Chapter 2

Computational Grids

> The current status of computation is analogous in some respects to that of electricity around 1910. At that time, electric power generation was possible, and new devices were being devised that depended on electric power, but the need for each user to build and operate a new generator hindered use. The truly revolutionary development was not, in fact, electricity, but the electric power grid and the associated transmission and distribution technologies. Together, these developments provided reliable, low-cost access to a standardized service, with the result that power which for most of human history has been accessible only in crude and not especially portable forms (human effort, horses, water power, steam engines, candles) became universally accessible. By allowing both individuals and industries to take for granted reliable power, the electric power grid made possible new industries that manufactured them.
>
> Foster, Kesselman [9], Chapter 2

By analogy, the term computational grid is adopted from the power grid and describes the infrastructure that will enable increases in computation power. To achieve these increases the hardware and software infrastructure must be provided.

Buyya, Abramson, Venugopal [6] state that in grid environments the *supplier* (computer center) and the *consumer* of resources have different goals, objectives, strategies, and supply-and-demand patterns how they want to use the system. Consumers want to get whatever resources they can get and don't want to pay much for them. On the other hand the suppliers have fixed costs as well as costs to buy new equipment. Therefore they need to make money if they sell the computing cycles to others. Also if you take into account only non-dedicated machines like desktop computers, which have different day-to-day duties, you still need to calculate on extra costs. It is also not possible to block a PC completely so that the actual owner cannot get any computer power. To achieve the balance between the needs of the customers and the suppliers you have to have some policies. They can either be system-centric or user-centric .

System-centric is the traditional approach in many time-sharing operating systems where the scheduler in the system decides which tasks are executed in which order. *User-centric* on the other hand concentrates more on the users needs and contracts. It depends on the Quality

of Service (QoS) agreement the user has, how much computation power he gets. Enforcing QoS requires a system which knows about penalty and rewards.

Grids could also be distinguished by their propagation. Wolfgang Gentzsch [15] writes about three different types of grids:

Cluster Grid: The simplest grid is a local cluster of computers, interconnected through a network. Cluster grids are mostly used for heavy throughput computing, distributing many compute jobs to the processors.

Examples include:

- Entertainment: rendering of thousands of frames to produce a movie
- Electronic Design: design and test simulations to build VLSI chips, and
- Bioinformatics: scanning hundreds of thousands of genomics and proteomics sequences

Cluster grids are usually in a single administrative domain, used by one owner (one group of users, one project, or one department).

Campus Grids and Global Grids: Campus and global grids consist of several clusters located in multiple administrative domains, in departments, enterprises, or even distributed globally over the Internet. Complexity of these grids depends on many factors, but most importantly on the number of grid resource owners, on the number of different grid sites, or on the number of layers or tiers in a hierarchical architecture. Complexity of these grids depends on many factors, but most importantly on the number of grid resource owners, and on the number of different grid sites.

With the history of the development of computer networking in mind there is considerable evidence that such an evolution will also occur to distributed computing and especially grid computing. The capabilities of both computers and the underlying network continue to increase dramatically. All these parameters will lead to a network of computers which need to be connected. As in the early days when time-sharing computers came up one will again share computers but in a grid fashion.

The time is ripe for a transition to a more integrated computational grid with dependable and pervasive computational capabilities and consistent interfaces. In such grids, today's distributed applications will be routine, and programmers will be able to explore a new generation of yet more interesting applications that leverage teraFLOP computers and petabyte storage systems interconnected by gigabit networks.

Here is an example to illustrate how grid functionality may transform different aspects of our lives:

A finance software that you can use at home uses only your own CPU for forecasts that it generates from collected data. Depending on your Internet connection you will be able to get more or less data to analyze to make the decision. The actual computations performed on this data are relatively simple. In a grid environment an individual may make a stock-purchasing decision on the basis of detailed Monte Carlo analysis of future asset value, performed on remote teraFLOP computers. This would provide three orders of magnitude more computing power than today.

2.1 Grid Applications

The applications in need of the computing power of grid environments can be divided into five major application classes [9]:

Category	Examples	Characteristics
Distributed supercomputing	Distributed interactive simulation Stellar dynamics Ab initio chemistry	Very large problems needing lots of CPU, memory, etc.
High throughput computing	Chip design Parameter studies Cryptographic problems	Harnessing many otherwise idle resources to increase aggregate throughput
On demand computing	Medical instrumentation Network-enabled solvers Cloud detection	Remote resources integrated with local computation, often for a limited amount of time
Data intensive computing	Sky survey Physics data Data assimilation	Synthesis of new information from many or large data sources
Collaborative computing	Collaborative design Data exploration Education	Support communication or collaborative work between multiple participants

2.1.1 Distributed Supercomputing

Distributed supercomputing applications use grids to aggregate substantial computational resources in order to tackle problems that cannot be solved on a single system. Depending on the grid on which you are working, these aggregated resources might comprise the majority of the supercomputers in the country or simply all of the workstations within a company.

An example may be Distributed Interactive Simulation (DIS) which is a technique used for training in the military. Realistic scenarios may involve hundreds of thousands of entities. Each of them may have a potentially complex behavior pattern.

The challenging issues for a grid architecture are the need to co-scedule scarce and expensive resources, scalability of protocols and algorithms to many nodes and achieving and maintaining high levels of performance across heterogeneous systems.

2.1.2 High Throughput Computing

In high throughput computing, the grid is used to process many independent tasks on unused processors (often from idle workstations). The result may be the focussing of available resources on a single problem. Here are two examples:

- The Condor system of the University of Wisconsin is used to manage pools of hundreds of workstations at universities and laboratories around the world. These resources have

been used for studies as diverse as molecular simulations of liquid crystals, studies of ground-penetrating radar, and the design of diesel engines.

- More loosely organized efforts are around that harness the CPU power from tens of thousands of computers worldwide:
  - distributed.net:

    > distributed.net was the Internet's first general-purpose distributed computing project. Founded in 1997, our network has grown to include thousands of users around the world donating the power of their home computers to academic research and public-interest projects.
    >
    > http://www.distributed.net/

  - SETI@home:

    > SETI@home is a scientific experiment that uses Internet-connected computers in the Search for Extraterrestrial Intelligence (SETI). You can participate by running a free program that downloads and analyzes radio telescope data.
    >
    > http://setiathome.ssl.berkeley.edu/

2.1.3 On-Demand Computing

On-demand applications use grid capabilities to meet short-term requirements for resources that cannot be cost-effectively or conveniently located locally. These resources may be computation, software, data repositories, specialized sensors, and so on. In contrast to distributed supercomputing applications, these applications are often driven by cost-performance concerns rather than absolute performance.

Over the last decade, biologists have developed a community code called *MCell*[1], which is a general simulator for cellular microphysiology (the study of the physiological phenomena occurring at the microscopic level in living cells). MCell uses Monte Carlo diffusion and chemical reaction algorithms in 3D to simulate complex biochemical interactions of molecules inside and outside cells. Grid technologies have enabled the deployment of large-scale MCell runs on a wide variety of target resources.

2.1.4 Data Intensive Computing

In data-intensive applications, the focus is on synthesizing new information from data that is maintained in geographically distributed repositories, digital libraries, and databases. The grid will be used to collect, store and analyze data and information, as well as to synthesize knowledge from data.

This synthesis process is often computationally and communication intensive as well.

2.1.5 Collaborative Computing

Collaborative applications are concerned primarily with enabling and enhancing human-to-human interactions. Such applications are often structured in terms of a virtual shared space.

[1] http://www.mcell.cnl.salk.edu/

Many collaborative applications are concerned with enabling the shared use of computational resources such as data archives and simulations; in this case, they also have characteristics of the other application classes just described.

The astronomy community has targeted the grid as a means of successfully collecting, sharing and mining critical data about the universe. The National Virtual Observatory Project (NVO) in the United States, the EU AVO project and the UK AstroGrid project are working together to provide scientists and educators with an unprecedented amount of accessible information about the heavens.

2.2 The Grid Architecture

A decade of focused research and experimentation has produced a considerable consensus on the requirements and architecture of grid technology. Standard protocols, which define the content and sequence of message exchanges, and standardized application programming interfaces (APIs), which define standard interfaces to code libraries and allow code components to be reused, have emerged.

Grid architecture can be thought of as a series of layers. The lowest level, the fabric, provides the physical devices or resources that grid users want to share and access, including computers and storage systems.

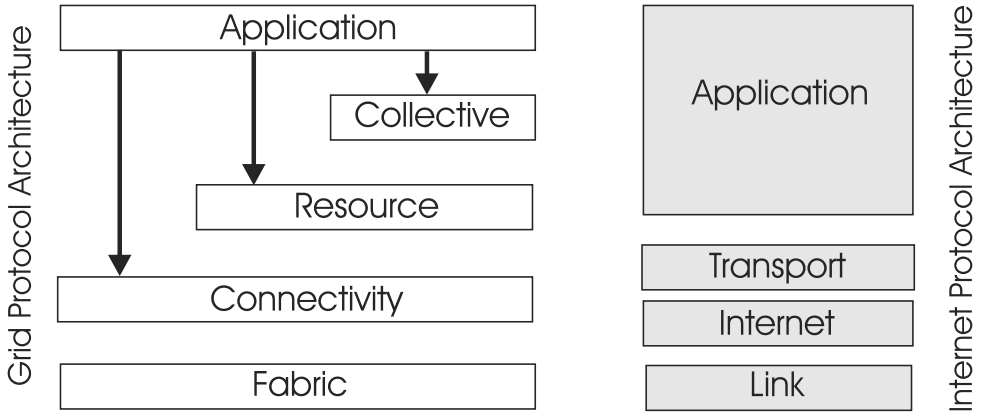

Figure 2.1: Grid protocol architecture [14].

Above the fabric there are the connectivity and resource layers. The protocols in these layers must be implemented everywhere and, therefore, must be relatively small in number. The connectivity layer contains the core communication and authentication protocols. Communication protocols enable the exchange of data between resources, whereas authentication protocols build ontop of these communication protocols to provide cryptographically secure mechanisms for verifying the identity of users.

The collective layer contains the needed protocols and APIs that implement interactions across collections of resources. Examples of collective services include directory services for resource discovery, monitoring services and membership and policy services for keeping track of who in a community is allowed to access resources.

At the top of any grid system there are the user applications, which depend on the components of the other layers. Every application needs to

- *obtain* necessary authentication credentials (connectivity layer protocols)
- *query* an information system to determine the availability of computers (collective services)
- *submit* requests to appropriate computers (resource protocols) and
- *monitor* the progress of the various computations.

Many of these functions can be carried out by tools that automate the more complex tasks.

Chapter 3

HPC at the TU Wien

Most universities provide their students with computer infrastructure. At the Vienna University of Technology (TU Wien), for instance, the Information Technology Services (”Zentraler Informatikdienst”, ZID) provides about 250 PCs located all over the university buildings near the Vienna Karlsplatz. These PCs—being a considerable computational resource—are idle during night-time and times without lectures (holidays etc.).

These PCs are currently embedded in a remote boot environment. This setup is required because the PCs do not have hard-disks and cannot boot locally. For this purpose an environment is set up to push all needed files over the network to the nodes. In order to utilize this computing power in a grid context it is necessary that the nodes are provided with the needed software. The remote boot environment simplifies this process so that a special boot image can be provided that has all the needed grid tools on board and therefore enables the nodes to participate in the grid environment.

It is the objective of this thesis to make these idle resources available for research projects which are in desperate need of computer power during times of peak demand or even permanently. Specific examples of such projects at the Vienna University of Technology are located at

(i) the Institute for Microelectronics (running, for instance, their MINIMOS code),

(ii) the Photonics Institute (with time-consuming MCTDHF calculations), and

(iii) the Institute for Materials Chemistry (with their WIEN2K materials science code).

These examples demonstrate that there is definitely interest in using such environments.

On the other hand it has to be pointed out that such a setup is not a global toolkit to solve any problem around. These PCs have some disadvantages one has to keep in mind when thinking about such a setup. For example, the computers currently available have a maximum of 512 MB RAM. Many scientific applications require more memory. Also the lack of a hard disk makes swapping impracticable.

Everyone interested in such a setup needs to study the requirements and then has to decide whether the effort setting it up is legitimate.

3.1 Partners in this Project

Currently the partners of the project described in this thesis are running their software on their own equipment exclusively. The goal is to give them access to the unused resources.

Following a description is given of the three departments: their current situation, their needs, and their wishes.

3.1.1 The Institute for Microelectronics (IuE)

The main research field of the IuE is the development of software tools for the simulation of semiconductor devices and technological process steps. These tools are applied either to the improvement of existing ULSI semiconductor devices or to support the development of new technologies.

The IuE has currently the following soft- and hardware:

- AIX Nodes:
  IBM 4 Nodes with 8 64bit Power+ processors (about 1.7 GHz) each, 2 with 64 GB and 2 with 32 GB RAM, 1.5 TB Storage
- Linux Nodes:
  about 50 Nodes with Intel PIV, 1.5-2.8 GHz, 1-2GB RAM and about 10-15 Nodes 500-1000 MHz, 256-512 MB RAM
- Tru64 Nodes:
  4 Nodes with Alpha 21164 64bit about 300 MHz, about 500 MB RAM

There is a scheduler in place which queues the simulation jobs on the AIX Cluster and the Linux workstations, which are used in a student lab during the day and for computation during the night.

The grid environment of this thesis has been tested with the software of the IuE and has successfully finished all test cases.

3.1.2 The Photonics Institute Theory Group

The need for high power computing at this group is based on their research into correlated multi-electron dynamics in laser fields (MCTDHF) [28].

Currently a cluster of Linux PCs is in use using Condor as queueing and scheduling software.

The group is planning to use the grid environment from time to time to overcome peak load situations.

3.1.3 The Institute of Materials Chemistry

The program package WIEN2k allows to perform electronic structure calculations of solids using density functional theory (DFT). It is based on the full-potential (linearized) augmented plane-wave ((L)APW) + local orbitals (lo) method, one among the most accurate schemes for band structure calculations. In DFT the local (spin)

> density approximation (LDA) or the improved version of the generalized gradient approximation (GGA) can be used. WIEN2k is an all-electron scheme including relativistic effects and has many features.
>
> http://www.wien2k.at/

There are plans to use the WIEN2k program in the grid environment and the tests were very promising.

3.2 Public Compute Resources at the TU Wien

There are several institutions which provide their students with workstations for their education. The Information Technology Services (ZID) is the institution providing the largest number of PCs for their internet rooms. But there are other computer labs as well, which might join the grid environment:

- ZID Internet-Räume: There are about 250 PCs all over the TU Wien:
  - Freihaus: 4 Rooms, 87 PCs
  - Operngasse: 3 Rooms, 35 PCs
  - Gußhaus: 34 PCs
  - Getreidemarkt: 20 PCs
  - Karlsplatz: 18 PCs
  - other small locations: 29 PCs
- Computer Science Lab in Favoritenstraße:
  There are 74 PCs available. In their normal setup they are behind a firewall. They are completely idle during holidays and weekends.
- Computerlab for architects:
  68 PCs running Windows from Disk. The location is open from 08:00 until 22:00 hours.

There are several other labs which are usually unused during holidays. If more computer resources are needed these labs might be added to the environment.

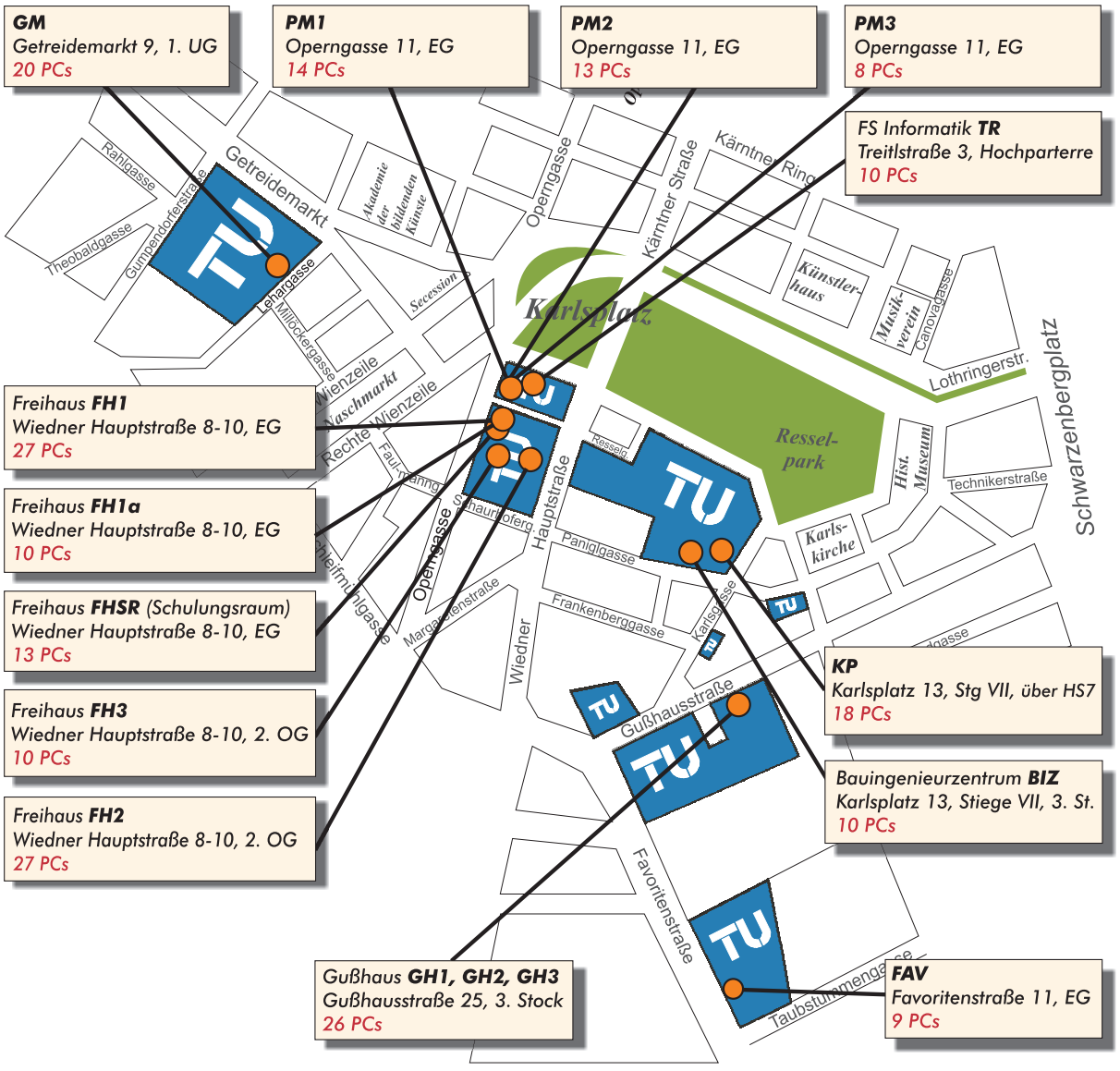

Figure 3.1: ZID Internet-Räume.

Chapter 4

Available Grid Software

In this section I want to discuss some of the available grid toolkits in their functionality, complexity, installation, and use. For this purpose a small setup has been made with three computers to test the installation and gather knowledge about how the software works and performs.

The following software products were tested:

- Sun Grid Engine 6.0,
- NetSolve/GridSolve-2.0,
- Globus Toolkit 4.0,
- IBM Grid Toolbox V3 for Multiplatforms, and
- Condor.

4.1 The Sun Grid Engine 6.0 (SGE)

The SGE is an open source initiative from Sun Microsystems and available to the public as source and pre-built binaries for several platforms.

Sun Microsystems acquired gridware, a private developer of Distributed Resource Management (DRM) software, in July 2000.

With this acquisition Sun was able to facilitate the deployment of compute farms, the basic building blocks of grid computing, by releasing Sun Grid Engine as a free downloadable binary for Solaris and for Linux. In the first nine months following the product's introduction Sun had over 9,000 free downloads of Sun Grid Engine software, and interest in the technology grew.

4.1.1 Grid Engine Project Objectives

The Grid Engine open source project further extends Sun's commitment to enable the grid computing model. The project has the following objectives:

Technology Evolution and Adoption. Grid computing is about making large amounts of compute power available for applications and users. Collaborative development of Grid

Engine technology provides the proper development framework to ensure that Grid Engine technology meets the requirements of the largest number of users. And as Bill Joy of Sun said, "Innovation happens elsewhere". Open source invites participation and ideas, and the collaborative nature of open source development is a proven approach to building high quality code which addresses user needs.

Open Standards. In the absence of standards, Independent Software Vendors (ISVs) have been understandably reluctant to directly interface their applications to DRM software. The Grid Engine project will foster the creation of new open standards for DRM, focusing initially on a community effort to develop a standard API for application integration. ISVs would only have to 'write once' to be able to use any compatible DRM system, and integration tasks would be eased for the end user.

Market Infrastructure. A strong base of channel participants for service and support can ensure that everyone will benefit from the grid computing model. Today, many organizations use internal staff to deploy and maintain compute farms. As the technology is gaining acceptance, the need for service providers in the market is increasing. Open source provides a franchise to all market participants unlike any other business agreement or license mechanism. All open source licensees have full and equal participation in the project and in the community, and will have all the resources necessary to fully utilize Grid Engine in implementing integrated solutions.

4.1.2 Installation

The installation was quite straightforward. The documentation is very good and there was just one little issue with the pre-built Linux binaries, which was that a binary was linked with a four year old library, which is not installed by default on common systems anymore. But once this lib was installed, everything worked nicely.

4.1.3 Configuration

For the configuration there is on one hand a graphical interface with which you can tune every parameter of the system. On the other hand there is also the possibility to use command-line tools to configure the SGE.

4.1.4 Working with the SGE

For a queueing system the SGE is a nice tool which has all necessary stuff on board to be used in a cluster environment.

4.1.5 Pros

- The SGE has a nice user interface for job submission and configuration of the cluster.
- It can be used quite nicely as a queueing system for existing software.

- Grid Engine 6.0u1 supports the following Platforms:
  - Apple Mac OS/X
  - Compaq Tru64 Unix 5.0, 5.1
  - Hewlett Packard HP-UX 11.x
  - IBM AIX 4.3, 5.1
  - Linux x86, kernel 2.4, glibc $\geq$ 2.2
  - Linux AMD64 (Opteron), kernel 2.4, glibc $\geq$ 2.2
  - Silicon Graphics IRIX 6.5
  - Sun Microsystems Solaris (Sparc) 7 and higher 32-bit
  - Sun Microsystems Solaris (Sparc) 7 and higher 64-bit
  - Sun Microsystems Solaris (x86) 8 and higher
- Checkpointing: SGE has a possibility to use checkpointing software. There are bindings to use available checkpointing software on Linux (not tested).

4.1.6 Cons

- There is no obvious way to get the stdout/stderr of running programs back to the submission host. These files always remain on the exec host.
- Every execution host needs complete administrative rights over the SGE. If there are users on these hosts they can change settings. This is the main show-stopper for SGE in the current setup.

4.2 GridSolve/NetSolve

This software has been developed at the University of Tennessee, Knoxville Tennessee, by Jack Dongarra and his team. It is an open source system aimed to bring together disparate computational resources connected by computer networks. It is a RPC based client/agent/server system that allows one to remotely access both hardware and software components.

The project overview at http://icl.cs.utk.edu/netsolve/ cites:

> The purpose of GridSolve is to create the middleware necessary to provide a seamless bridge between the simple, standard programming interfaces and desktop Scientific Computing Environments (SCEs) that dominate the work of computational scientists and the rich supply of services supported by the emerging grid architecture, so that the users of the former can easily access and reap the benefits (shared processing, storage, software, data resources, etc.) of using the latter.
>
> This vision of the broad community of scientists, engineers, research professionals and students, working with the powerful and flexible tool set provided by their familiar desktop SCEs, and yet able to easily draw on the vast, shared resources of the grid for unique or exceptional resource needs, or to collaborate intensively with colleagues in other organizations and locations, is the vision that GridSolve will be designed to realize.

4.2.1 Installation

The installation is a straight forward compilation and installation with very common autoconf/automake scripts. It compiles in a recent Linux environment without problems and the tests are executed without problems.

4.2.2 Configuration

The basic configuration is quite straightforward. Just set the remote machine which is the agent for this cluster and you are ready. Same procedure for the clients and the servers.

4.2.3 Pros

Easy to handle software with the ability to add your own problem solvers.

4.2.4 Cons

To use GridSolve you need to adapt your software to use the hooks provided by GridSolve. Since in our primary setup we only want to use ready made software to run this is the main show-stopper for GridSolve.

4.3 The Globus Toolkit Version 4.0

> The open source Globus Toolkit is a fundamental enabling technology for the "grid," letting people share computing power, databases, and other tools securely online across corporate, institutional, and geographic boundaries without sacrificing local autonomy. The toolkit includes software services and libraries for resource monitoring, discovery, and management, plus security and file management. [...]
>
> The toolkit includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability. It is packaged as a set of components that can be used either independently or together to develop applications. Every organization has unique modes of operation, and collaboration between multiple organizations is hindered by incompatibility of resources such as data archives, computers, and networks. The Globus Toolkit was conceived to remove obstacles that prevent seamless collaboration. Its core services, interfaces and protocols allow users to access remote resources as if they were located within their own machine room while simultaneously preserving local control over who can use resources and when.
>
> http://globus.org/toolkit/about.html

To be able to easily use the current grid environment for computation of ready available problems the Globus Toolkit is probably not the best choice. It is obviously just a level above a scheduler.

4.4 The IBM Grid Toolbox V3 for Multiplatforms

> The IBM Grid Toolbox V3 for Multiplatforms is a comprehensive, integrated toolkit for creating and hosting grid services. This product includes material developed by the Globus Alliance (http://www.globus.org/), as well as a set of APIs and development tools to create and deploy new grid services and grid applications. It also includes a limited integrated hosting and development environment, capable of running grid services and sharing them with other grid participants, such as grid service providers and grid service consumers. It also provides a set of tools to manage and administer grid services and the grid hosting environment, including a Web-based interface, the Grid Services Manager.
>
> http://www-1.ibm.com/grid/solutions/grid_toolbox.shtml

The IBM Grid Toolbox has been evaluated in this project but it was very hard to suite the requirements of this software:

- Red Hat Enterprise Linux Advanced Server for Intel 2.1.
- SUSE LINUX Enterprise Server 8.0—Powered by UnitedLinux v1.0.

The IBM Grid Toolbox was with a special setup of Red Hat Enterprise Linux Advanced Server 2.1. That was the only version of the operating system where the package would extract itself. With RHAS 3.0, which was the latest then, it would not even extract.

In general the IBM software is not easy to handle. After the installation, the examples did not work at all. Errors occurred often so that in the end testing this software was stopped because it did not work.

4.5 The Condor Workload Management System

> The goal of the Condor Project is to develop, implement, deploy, and evaluate mechanisms and policies that support High Throughput Computing (HTC) on large collections of distributively owned computing resources. Guided by both the technological and sociological challenges of such a computing environment, the Condor Team has been building software tools that enable scientists and engineers to increase their computing throughput.
>
> http://www.cs.wisc.edu/condor/

Condor is a specialized workload management system for compute-intensive jobs. Like other full-featured batch systems, Condor provides a job queueing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. Users submit their serial or parallel jobs to Condor, Condor places them into a queue, chooses when and where to run the jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion.

Condor features some different execution environments (in Condor terms ”Universes”). The ”Standard Universe” uses specially linked binaries to be able to checkpoint and migrate

jobs to other machines if one becomes unavailable. In the ”Vanilla Universe” regular binaries are being executed. They lack the ability to be checkpointed.

In the MPI Universe you can even execute MPI jobs without the need of an mpid or shared keys. Until Condor 6.7.8 it is only possible to use MPICH 1.2.4.

4.5.1 Pros

- May be used as a queueing system: In the ”Vanilla” universe condor can queue and execute any job which also runs as a native process on one node.
- MPI: Condor can handle MPI jobs by itself without the need of rsh or ssh.
- Console redirection: During the execution of a job all messages sent to *stdout* or *stderr* can be saved in a file.
- Coupling of different Condor pools: It is possible to share condor pools between interested parties (Flocking).

During the evaluation of Condor it appeared that it would suite all our needs perfectly. Condor has been in use for over 6 months now and we have had no problems.

Chapter 5

WINZIG

The TU Wien ZID Grid

Before going into the details of the installation of the necessary servers I want to present our setup to show you how everything works here at the Vienna University of Technology.

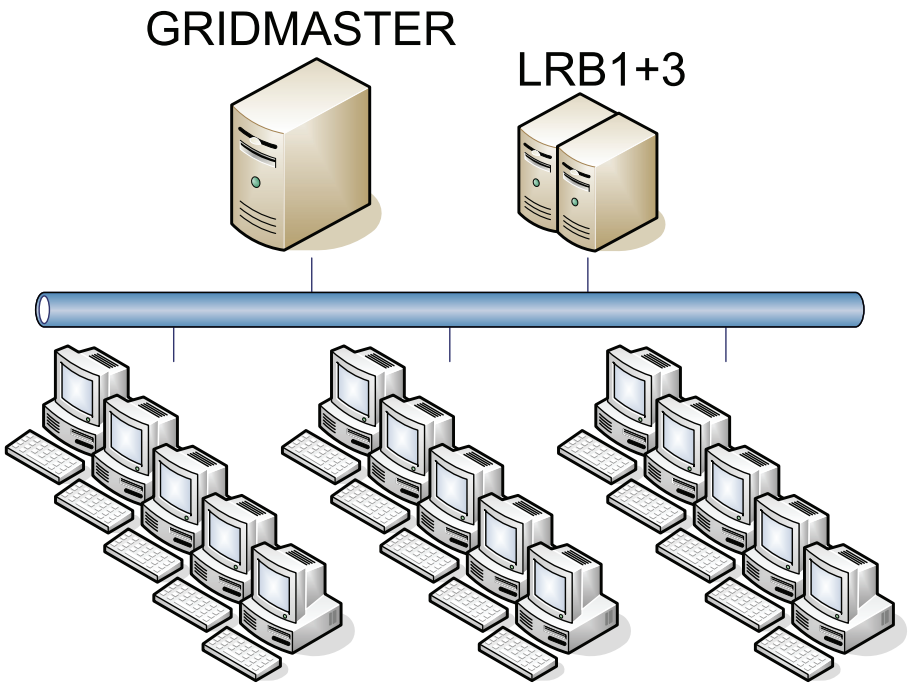

Figure 5.1: ZID Internet-Räume.

As you can see in Fig 5.1 all PCs in the Information Technology Services Internet rooms are connected in one LAN. There are two boot servers for the normal student service (LRB1 and LRB3). These two servers are also used for the grid service as DHCP servers. They change the DHCP configuration automatically in the evening to the grid setup and back to student service in the morning using cron jobs.

- Student Service:
  During regular student service hours LRB1 and LRB3 provide all necessary services to the clients (DHCP, TFTP, and all needed Filesystems via NFS).
- Grid Service:
  After the rooms are closed for the public, the LRBs are reconfigured for grid service providing only DHCP service to the clients. TFTP and NFS service is now provided by gridmaster.

The server housing gridmaster is currently serving its services to over 200 clients without problems. The hardware configuration can be taken from the Table below:

Motherboard	Supermicro P4SCE
CPU	Intel® Pentium® 4 CPU 2.80 GHz
Memory	2 GB DDR RAM
Disk space	2 x 250 GB IDE Drive in Software RAID1
Network	2 x Gigabit Ethernet (Etherchannel)
OS	Debian GNU/Linux 3.1 Sarge

Table 5.1: Hardware of Gridmaster.

Chapter 6

Debian Basics

In the next chapters I am going to describe the installation of the remote boot server using Debian GNU/Linux [1]. After that I will guide you through the process of setting up the remote boot system and to finish I will show the installation of the grid software onto the server and the clients.

A remote boot environment is needed in our setup because our lab PCs have no hard-disk and can't be booted locally. For this purpose an environment is set up to push all needed files over the network to the node.

This environment could also be used for workstations that usually use a hard-disk and boot into the needed environment, but also want to be added to the cluster. For this purpose we could deliver a floppy disk, where the needed information is stored to be able to contact the remote boot server and start the image from there. With this ability one can add a node very easily to the cluster and doesn't risk to damage their own installation.

I have chosen Debian because in my opinion Debian is the best GNU/Linux distribution around. I know that many people think that Debian is hard to use, but once you are over the first troubles it's only downhill.

6.1 What is GNU/Linux?

Linux is an operating system: a series of programs that let you interact with your computer and run other programs.

An operating system consists of various fundamental programs which are needed by your computer so that it can communicate and receive instructions from users; read and write data to hard disks, tapes, and printers; control the use of memory; and run other software. The most important part of an operating system is the kernel. In a GNU/Linux system, Linux is the kernel component. The rest of the system consists of other programs, many of which were written by or for the GNU Project. Because the Linux kernel alone does not form a working operating system, we prefer to use the term "GNU/Linux" to refer to systems that many people casually refer to as "Linux".

Linux is modelled on the Unix operating system. From the start, Linux was designed to be a multi-tasking, multi-user system. These facts are enough to make Linux different from

[1] http://www.debian.org

other well-known operating systems. However, Linux is even more different than you might imagine. In contrast to other operating systems, nobody owns Linux. Much of its development is done by unpaid volunteers.

Development of what later became Linux began in 1984, when the Free Software Foundation[2] began development of a free Unix-like operating system called GNU.

The GNU Project has developed a comprehensive set of free software tools for use with Unix™ and Unix-like operating systems such as Linux. These tools enable users to perform tasks ranging from the mundane (such as copying or removing files from the system) to the arcane (such as writing and compiling programs or doing sophisticated editing in a variety of document formats).

While many groups and individuals have contributed to Linux, the largest single contributor is still the Free Software Foundation, which created not only most of the tools used in Linux, but also the philosophy and the community that made Linux possible.

Linux kernel[3] first appeared in 1991, when a Finnish computing science student named Linus Torvalds announced an early version of a replacement kernel for Minix to the Usenet newsgroup `comp.os.minix`. See Linux Internationale's Linux History Page[4].

Linus Torvalds continues to coordinate the work of several hundred developers with the help of a few trusty deputies. An excellent weekly summary of discussions on the `linux-kernel` mailing list is Kernel Traffic[5].

Linux users have immense freedom of choice in their software. For example, Linux users can choose from a dozen different command line shells and several graphical desktops. This selection is often bewildering to users of other operating systems, who are not used to thinking of the command line or desktop as something that they can change.

Linux is also less likely to crash, better able to run more than one program at the same time, and more secure than many operating systems. With these advantages, Linux is the fastest growing operating system in the server market. Linux has begun to be popular among home and business users as well.

6.2 What is Debian?

Debian is an all-volunteer organization dedicated to developing free software and promoting the ideals of the Free Software Foundation. The Debian Project began in 1993, when Ian Murdock issued an open invitation to software developers to contribute to a complete and coherent software distribution based on the relatively new Linux kernel. That relatively small band of dedicated enthusiasts, originally funded by the Free Software Foundation[6] and influenced by the GNU[7] philosophy, has grown over the years into an organization of around 900 *Debian Developers*.

[2] http://www.gnu.org/
[3] http://www.kernel.org/
[4] http://www.li.org/linuxhistory.php
[5] http://www.kerneltraffic.org/kernel-traffic/index.html
[6] http://www.fsf.org/fsf/fsf.html
[7] http://www.gnu.org/gnu/the-gnu-project.html

6.3 Basics of the Debian Distributions

Debian maintains three different distributions simultaneously. These are:

- `stable` — Most useful for a production server since it is only updated with security fixes.
- `testing` — The preferred distribution for a workstation since it contains recent releases of desktop software which have received a bit of testing.
- `unstable` — Cutting edge. The choice of Debian developers.

When packages in `unstable` have no release-critical (RC) bugs filed against them after the first week or so, they are automatically promoted to `testing`.

Debian distributions also have code names[8]. Before Woody was released in August 2002, the three distributions were, respectively, Potato, Woody, and Sid. After Woody was released the three distributions were, respectively, Woody, Sarge, and Sid. When Sarge was released, the `stable` and `unstable` distributions were Sarge and Sid; a new `testing` distribution was then created (initially as a copy of `stable`) and given the new code name Etch.

For more detailed information on Debian read the very detailed Reference Guide by Osamu Aoki[3].

[8]Codenames that have already been used in the past are:
"Buzz" for release 1.1, "Rex" for release 1.2, "Bo" for releases 1.3.x, "Hamm" for release 2.0, "Slink" for release 2.1, "Potato" for release 2.2, "Woody" for release 3.0, and "Sarge" for release 3.1.

Chapter 7

Installation of the Remote Boot Server

7.1 Getting the Installation Media

To be able to install Debian on your hardware, I suggest to get a small ISO image of the installation CD and afterwards download all missing packages from a mirror close to you. This saves a lot of bandwidth comparing to downloading about 13 CD ISOs or one DVD ISO.

In this document I will describe the installation using the latest stable release named Sarge.

At http://www.debian.org/CD/netinst/ you will find links to the ”Official netinst images for the "stable" release”. Choose your architecture (most probably i386) and download the `.iso` image file. After downloading you have to burn this image on a CD and now you are ready to get into Debian!

7.2 Booting the Installation Media

After you have booted from the CD you will be presented a welcome screen. If you have fairly new hardware I would recommend that you start with a kernel from the 2.6 series. To do this, just issue the command “linux26” on the command prompt. If you just press “ENTER” the installation will be done using 2.4 series kernel. For more options you can flip through help pages using the “F-Keys” (F1 - F10).

After the kernel has booted you will be presented a welcome screen where you can choose the language of your installation. I always use English Unices and therefore I will describe everything using English here.

If you need or want to have another language installed you are free to decide here or later using the `locales` setting.

On the next screens the location of the server and the keyboard layout are determined. Afterwards the installed hardware is detected. If this scan is successful the installer starts to unpack itself and starts with the network setup. If you have a DHCP Server in your network it will contact it and request an IP address. If not you need to do this manually. Please check with your local administrator for free IP addresses. You will need at least one in each subnet where you want to remote boot a machine.

Next you need to name your server. In my documentation the Server is called gridmaster. But you are free to choose anything else. But be careful that you may need to substitute the

Figure 7.1: Boot Screen of the Debian Installer.

name in some screen or scripts later on. If your IP was assigned using DHCP you will be presented with your domain name on the next screen. If not, please fill in your domain.

After finishing with the network section, you need to partition your disk(s). This is one of the trickiest parts of any installation.

7.3 Partitioning the System

I prefer to use different partitions for different directory trees to limit damage upon system crash, e.g.

```
/          == (/ + /boot + /bin + /sbin + /var +
               /usr + /tmp + /remote)
           == 10GB+
/grid      == (/grid + /grid/home)
           == 50GB+
```

The size of the `/` directory depends on how many packages and applications you want to install on the gridmaster and for the clients. You could also split off the `/remote` directory onto a separate partition. The more you have on different partitions the smaller the partitions get and the sooner one partition will be too small. On the other hand, one big chunk is also not very wise. If this partition crashes all data is lost.

For example, the current status of my gridmaster machine is as follows:

```
Filesystem            Size  Used Avail Use% Mounted on
/dev/md0               37G  4.2G   31G  12% /
tmpfs                1015M  4.0K 1015M   1% /dev/shm
/dev/md2              189G  2.3G  178G   2% /grid
none                  5.0M  2.7M  2.4M  54% /dev
```

(The file system mounted on `/dev` is a virtual dev filesystem. This manages all dev-nodes on the fly, when a kernel module is inserted or removed. This is very handy because you have a very tidy `/dev` directory. For 2.6.x kernels this is achieved with the package udev.)

As you can see I have set up RAID devices for the server. Since hardware prices dropped a lot in recent years, it is not very expensive to buy two IDE hard drives of the same capacity. I have e.g. bought 2 250 GB IDE drives and am running a software RAID1 on those drives. This gives me an assurance against hardware fault: If one drive has a defect, I will insert a new one, and rebuild the raidsets onto the new drive. Then both drives have the same information on them again and if now the second drive fails, I still have the new drive with all information.

The RAID is only optional for the grid setup. If you only have one disk and good backup and restore capability it may be enough for you to have one drive and a regular backup to cover with disk failure.

The next steps of the setup procedure are dedicated to non-RAID setup. I will point out the changes you will need to take, if you have two or more disks to go on with ”Partitioning with RAID” on the next page.

While partitioning the disk be sure to set the partition with the boot loader to be bootable. Don’t forget a swap partition.

7.3.1 Partitioning without RAID1

When the partitioner starts up you are asked if you want to use the whole disk or if you want to set up the partitions yourself. I recommend that you do it yourself since only then you have the full control over the partitions (choose ”Manually edit partition table” on the screen).

On the next screen choose your hard drive and press Enter. If this is a brand new disk, you will be asked if you want to ”create a new empty partition table on this device”. Say ”Yes”.

Now you choose ”FREE SPACE” and ”Create New Partition”. Now you have to choose how much space of your disk should go to which partition. For the root partition settings I recommend:

```
Partition settings:
    Use as:             Eat3 journaling file system
    Mount point:        /
    Mount options:      defaults
    Label:              /
    Reserved blocks:    5%
    Typical usage:      standard
    Bootable flag:      on
```

If you are satisfied with this partition then choose ”Done setting up the partition” and go on with the rest of the ”FREE SPACE”.

For the swap partition the recommended settings are:

```
Partition settings:
    Use as:             swap area
    Bootable flag:      off
```

The last partition I recommend is a home/grid partition. The settings should be (to enter /grid as mount point just choose ”Enter manually” in the Mount point chooser):

```
Partition settings:
    Use as:             Ext3 journaling file system
    Mount point:        /grid
    Mount options:      defaults
    Label:              /grid
    Reserved blocks:    5%
    Typical usage:      standard
    Bootable flag:      off
```

If you want to distribute the partitions over several disks just feel free to do so. You will have each disk in the menu and can assign the partitions on your own will.

With this you are set to ”Finish partitioning and write changes to disk” and continue with the installation of the packages.

7.3.2 Partitioning with RAID1

When the partitioner starts up you are asked if you want to use the whole disk or if you want to set up the partitions yourself. You need do it yourself to get RAID support (choose ”Manually edit partition table” on the screen).

On the next screen you see the possibility to ”configure a software RAID”. But before we can configure it, we need the partitions set up. So choose the first hard drive and press Enter. If this is a brand new disk you will be asked, if you want to ”create a new empty partition table on this device”. Say ”Yes”.

Now you choose ”FREE SPACE” and ”Create New Partition”. Now you have to choose how much space of your disk should go to which partition. For the root partition settings I recommend:

```
Partition settings:
    Use as:             physical volume for RAID
    Bootable flag:      on
```

If you are satisfied with this partition then choose ”Done setting up the partition” and go on with the rest of the ”FREE SPACE”.

Do the same for the swap and grid/home partition but this time with the bootable flag set to ”OFF”. I also recommend to put the swap partition on a RAID partition because if one disk fails and the kernel wants to fetch something from the swap area which is located on the failed

disk, the machine will crash. If you have the swap also on both drives you are safe here again. In old kernels this was not possible but with both 2.4.x and 2.6.x Kernels this is no problem.

Also remember the exact sizes you used for the partitions. You will need them right away again when you partition the second drive exactly as the first one (See figure 7.2 on this page).

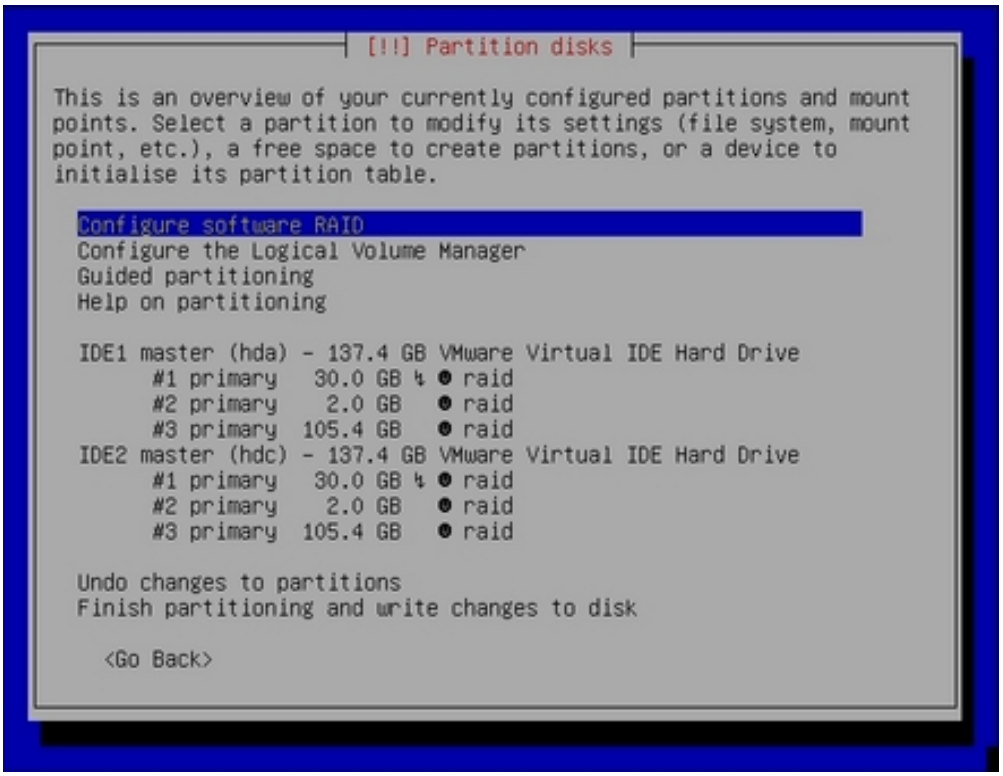

Figure 7.2: Partitions prepared for RAID setup.

After you have configured the partitions you can ”Configure software RAID”. You are requested to write the changes to the disks. Now you can ”Create (a) MD device” as ”RAID1” with 2 active devices and 0 spare devices.

For the root device take the two partitions on the disks which you have prepared for this purpose (probably part1). Select them with Space and press Tab to get to ”<Continue>”. Now do the same with the other partitions.

When you are finished with the setup choose ”Finish” and now you will see new RAID1 partitions below the other partitions (See figure 7.3 on the next page).

These partitions need to be set up to be usable by the system. Select the dedicated / partition and change the parameters correctly:

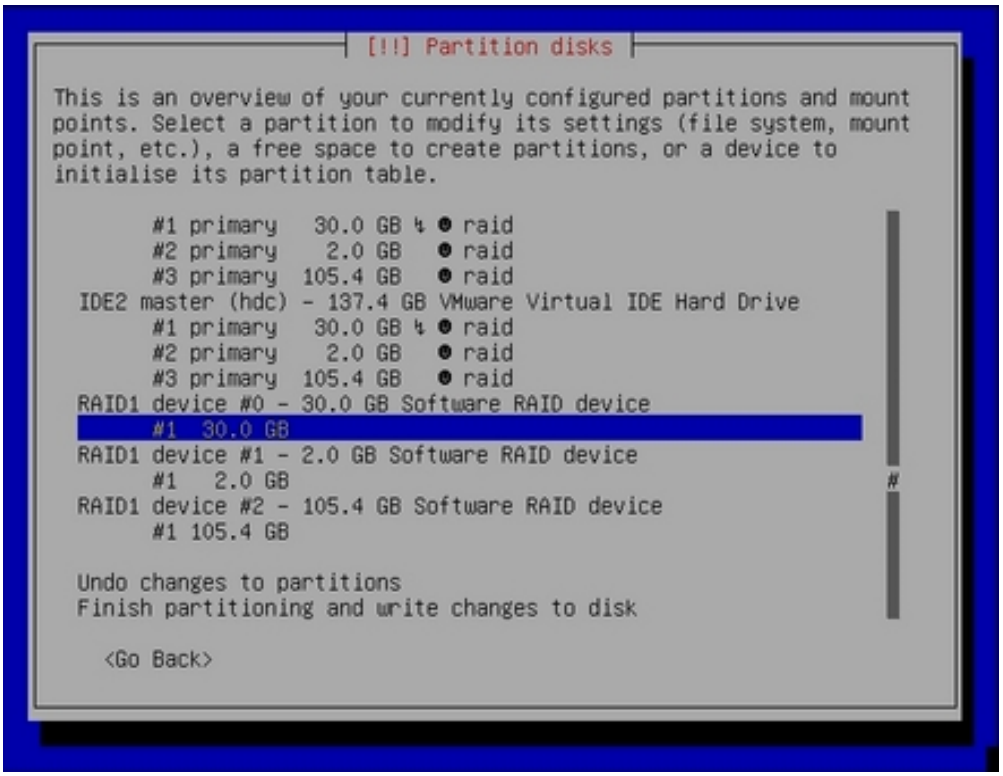

Figure 7.3: RAID partitions.

```
Partition settings:
    Use as:             Ext3 journaling file system
    Mount point:        /
    Mount options:      defaults
    Label:              /
    Reserved blocks:    5%
    Typical usage:      standard
```

Do the same with the swap partition and . . .

```
Partition settings:
    Use as:             swap area
```

the home/grid partition:

```
Partition settings:
    Use as:             Ext3 journaling file system
    Mount point:        /grid
    Mount options:      defaults
    Label:              /grid
    Reserved blocks:    5%
    Typical usage:      standard
```

After the setup the layout should be something like figure 7.4 on this page.

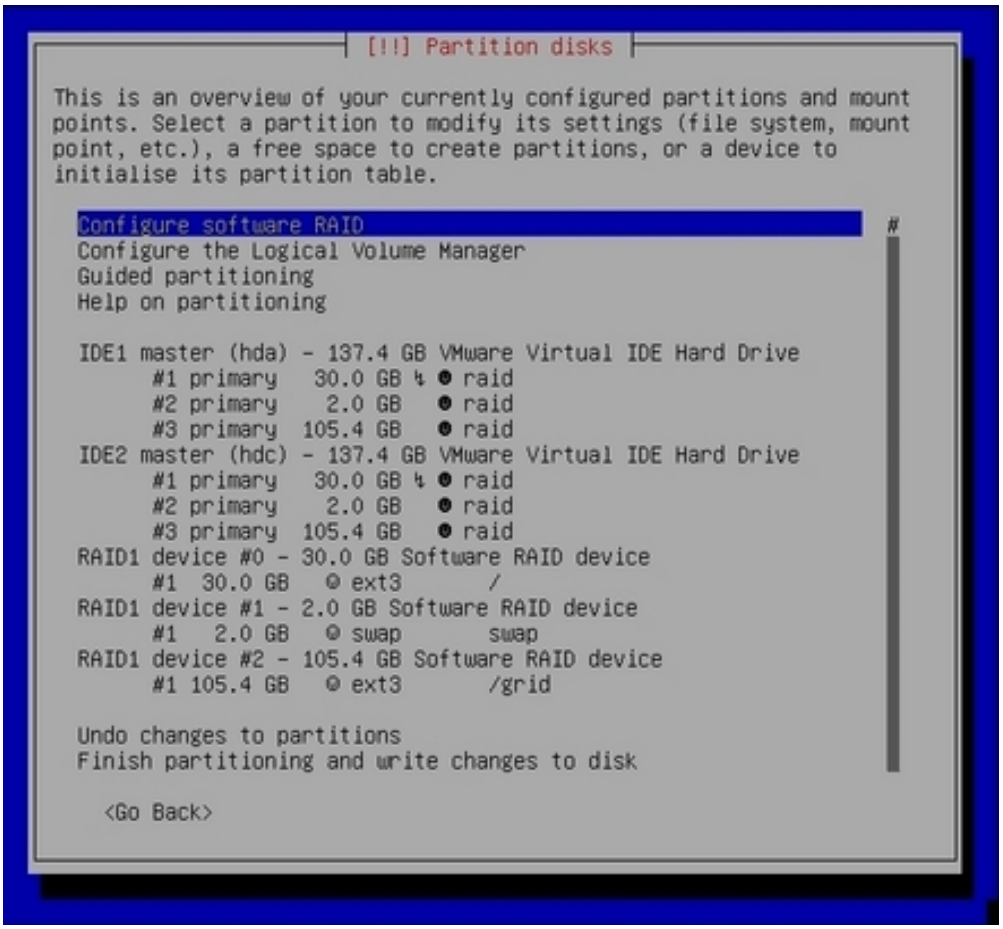

Figure 7.4: Partitions prepared for RAID setup.

With this you are set to ”Finish partitioning and write changes to disk” and continue with the installation of the packages.

7.4 Base Install

After formating the partitions the system is installing the base system. Just wait and watch or get yourself a cup of coffee! ;-)

7.5 Installing the Bootloader

The default bootloader in Debian is GRUB. It is a very easy to use bootloader. Just let the installer install it and the system should boot without problems.

I recommend that you change the bootloader to LILO if you have RAID1 for your root partition since only LILO supports booting from both partitions in case of a disk failure. Detailed

description can be found in section 7.8 on the next page.

7.6 Configuring the Base System

After the reboot of your new Debian installation your will be presented a ”Welcome Screen”.

Next you have to decide if ”the hardware clock is set to GMT”? Since your gridmaster will be a dedicated Linux Server your can say ”Yes” here. If you don’t want to, it is also safe to have the local time as your hardware time.

If you entered your location correctly before, the time zone should be set correctly. If not, please specify the right zone.

Next step is the root (or administrator) password. Make this password a safe one or your server may be compromised soon.

Since you should not work as root all the time it is time to set up a user for you. Enter the required parameters (your name, a user name and a password) for your account. The account will now be created and you should be able to login using this account after the configuration has finished.

Next you will be asked how to retrieve packages. This is one of the main advantages of Debian. All packages can be retrieved using standard HTTP or FTP mechanisms.

I recommend HTTP since I believe it is faster. Choose your country and then a mirror near you. If you are not satisfied with the performance of the mirror, your can choose another one later on as well.

If your mirror is reachable some packages will be retrieved. Next you are prompted to ”choose software to install”. Don’t choose any task here since we can install the needed software later on. Just tab to ”<OK>” and the needed packages will be installed.

After the package installation you will be prompted to configure your mail settings. Decide which config-item suits your setup most.

If everything goes well, you are set with the base installation.

From now on it’s fine-tuning.

7.7 Installing Additional Packages

If you need additional software installed I recommend to search if the needed software was already packaged. There are over 20.000 packages available where I believe the needed software can be found.

To search for a package use `apt-cache`:

```
# apt-cache search foo
```

You will be presented a list of packages. If you found your software you can install it with:

```
# apt-get install foo-package
```

If there are dependencies you will be presented a screen which additional packages will be installed. The software is fetched from a nearby mirror and installed to your system.

7.8 Changing GRUB to LILO with Root on RAID1

In setups with the root partition on a RAID1 I have achieved better performance with LILO as boot loader compared to GRUB. The problem with GRUB is that if the primary RAID disk fails and you try to start the machine from the secondary disk GRUB can not boot anymore.

Therefore I recommend to change the bootloader to LILO.

First install LILO using:

```
# apt-get install lilo
```

Next you need to do a first time configuration using

```
# liloconfig
```

This should generate a `/etc/lilo.conf` file for you. Now you need to edit it to include a section for the RAID setup:

```
raid-extra-boot=/dev/hda,/dev/hdc
```

Now just run LILO and everything should be set:

```
# lilo
Added linux *
Added Linux-old
The boot record of  /dev/md0  has been updated.
Warning: /dev/hda is not on the first disk
The boot record of  /dev/hda  has been updated.
The boot record of  /dev/hdc  has been updated.
```

The complete `/etc/lilo.conf` file can be found on page 80.

7.9 Kernel Update

If you want to use a newer/other kernel than the shipped 2.6.8 kernel from Sarge I recommend to take a recent kernel from the testing release and install the `kernel-image-*.deb` file using

```
# dpkg -i kernel-image-{YOUR VERSION}.deb
```

This should also trigger the bootloader to have the new kernel as the default entry. Reboot your server and make sure everything works as expected. After the upgrade your could also update the kernel for the remote boot clients. See Section 9.14 on page 50.

Chapter 8

Configuration of the Boot Server

To get the needed files to the clients a special setup is used at the DHCP [8] Server. If you have already a DHCP Server running for the subnet check with the administrator if it is possible to make the necessary adjustments. This setup is only describing how to configure a ISC DHCP Version 3 Server [1]. For other DHCP Server please consult their manual and/or producer if they are suitable for our purpose.

Furthermore you need a tftp [25] server capable of distributing the boot files to your client. I have chosen the atftpd package because it has all required features, especially the `tsize` option which is needed for `PXE` (Section 10.1 on page 51).

Last but not least you also need `NFS`[19] support on your server. See on page 35 for further details.

8.1 dhcpd

As mentioned above I am using ISC DHCP Version 3 Server. To install this software be sure you are in the gridmaster tree (and not the Remote Boot tree) and issue

```
# apt-get install dhcp3-server
```

During automatic package configuration you will be asked to which interfaces the DHCP Server should listen. If you are not sure then state all interfaces here on which you want to serve remote boot clients.

Additionally you need to configure the main config file `/etc/dhcp3/dhcpd.conf`.

There are some more important sections you need to take care of and some optional settings. I will walk you through the important settings.

The beginning of the config file specifies some global parameters:

- `option domain-name`
  This gives the local domain to the clients. If your clients are all within one domain you can specify this in the global section. Otherwise you can also specify this parameter in each `subnet` section.

[1] http://www.isc.org/index.pl?/sw/dhcp/

- `option domain-name-servers`
  The name server(s) of your organization. Without this setting no name resolution will work and you have to address each host by their IP address.
- `default-lease-time`
  How long the DHCP offer is valid. His parameter specifies at which interval the client should check back at the server if the issued IP address is still valid. For fixed clients the value can be 86400 (24 hours in seconds).
- `authoritative`
  If the DHCP Server is the only DHCP Server on the local network, then activate this directive. If there are other DHCP Servers around on the same network for other hosts, then either omit or comment this.
- `subnet`
  This parameter specifies the actual behavior of the DHCP server. Here you specify which IP range the clients are assigned to and many other things. It is subnet based and the syntax is:

```
subnet 10.254.239.0 netmask 255.255.255.0 {
  option routers                  10.254.239.1;
  option subnet-mask              255.255.255.0;
  option broadcast-address        10.254.239.255;
  option domain-name              "grid.test.tld";
  option domain-name-servers      10.254.239.1;
  use-host-decl-names             on;
}
```

There are 2 different possibilities to issue the clients their IP addresses. You can either provide a pool and let all clients be issued an address by the first-come-first-serve principle or specify each host exactly by its MAC address.

I suggest to provide MAC bound IP addresses because if your cluster gets bigger, problems will arise if you are not sure which machine failed.

Therefore I will only explain how to use the host based approach. If you want to use the pool, please check with the DHCP manual and/or man-page.

The following parameters are all valid within a `subnet` parameter:

- `option routers`
  This parameter is needed to specify the router (or gateway) for the subnet. Without this your machines won't get out into the internet or to other hosts on other subnets.
- `option subnet-mask`, `option broadcast-address`
  These parameters specify the layout of the network.
- `option domain-name`
  Specify the domain name of your subnet.

- `option domain-name-servers`
  Specify the name servers for your subnet.
- `use-host-decl-names`
  If the use-host-decl-names parameter is true in a given scope, then for every host declaration within that scope, the name provided for the host declaration will be supplied to the client as its hostname.

Within the `subnet` section an entry for every host is needed:

```
host pkcl4 {
  hardware ethernet       00:E0:18:A8:67:67;
  fixed-address           10.254.239.123;
  option root-path       "10.254.239.44:/remote";
  filename                "/remote/boot/pxelinux.0";

  next-server             10.254.239.44;
}
```

- `hardware ethernet`
  Specify here the MAC address of the network interface of the remote boot client. This is used to assign a specified IP address to a certain host.
- `fixed-address`
  The IP address to assign.
- `option root-path`
  This is passed to the dhcp-client and from there to the setup-script. It is used to mount the NFS-root filesystem on the client.
- `filename`
  Path to the initial boot file. This file is requested via TFTP and after transmission executed. It can be either a PXE executable or the kernel. How to set up these files is described in section 9.11 on page 47.
- `next-server`
  If the DHCP and the TFTP server are not on the same machine you can specify the address of the TFTP server with this option. Otherwise you do not need this option.

Such a section is now needed for every client you have in your pool.
An example of the `/etc/dhcp3/dhcpd.conf` file can be found on page 82.

8.2 tftp

For the initial file transfer a tftp server is needed. I can recommend the `atftpd` package:

```
# apt-get install atftpd
```

During configuration you should be set with the default answers. Important is to activate the `tsize` and the `block size` option. Furthermore set the `Base directory` to `/remote/boot`.

8.3 nfs

As NFS Server I suggest installing the `nfs-kernel-server`:

```
# apt-get install nfs-kernel-server
```

The kernel-mode NFS server is faster than the user-mode server (which can also be installed via `apt-get install nfs-user-server`).

After the installation you need to specify which directories you want to export. This is specified in the file `/etc/exports`:

```
# /etc/exports: the access control list for filesystems which may
#               be exported to NFS clients.  See exports(5).
/remote/bin      193.170.74.0/255.255.254.0(ro,sync)
/remote/lib      193.170.74.0/255.255.254.0(ro,sync)
/remote/root     193.170.74.0/255.255.254.0(rw,sync,no_root_squash)
/remote/sbin     193.170.74.0/255.255.254.0(ro,sync)
/remote/usr      193.170.74.0/255.255.254.0(ro,sync)
/remote/etc      193.170.74.0/255.255.254.0(ro,sync)
/remote/setup    193.170.74.0/255.255.254.0(ro,sync)

/root            193.170.74.0/255.255.254.0(rw,sync,no_root_squash)
/opt             193.170.74.0/255.255.254.0(rw,sync)
/grid            193.170.74.0/255.255.254.0(rw,sync)
/grid/condor     193.170.74.0/255.255.254.0(rw,sync)
/grid/condor/hosts  193.170.74.0/255.255.254.0(rw,sync)
/usr/lib/mpich   193.170.74.0/255.255.254.0(ro,sync)
```

After the configuration you need to restart the server by

```
# /etc/init.d/nfs-kernel-server restart
```

With these steps accomplished you should be set.

Remember: Whenever you have troubles, check the file `/var/log/syslog`. There should be all relevant error messages to debug the setup.

8.4 Wake on LAN

To be able to send the required Magic Packet [16] to the clients to wake them from a hibernated state you need to install a software capable of generating such packages.

I recommend the package `wakeonlan`:

```
# apt-get install wakeonlan
```

This program can be used with user rights and generates the needed packets to wake the clients.

The usage is really easy. Only specify the MAC address and the right packet is generated:

```
wakeonlan 01:02:03:04:05:06
```

Chapter 9

Installation of the Remote Boot Environment

To be able to start Linux on clients which don't have hard disks or where you don't want to change the current hard disk setup we create a new installation in a chroot[1] environment of our server. This kind of setup is quite handy, since you can have two different setups of Linux and still be able to maintain both in a very easy way using the Debian tools.

There is also the possibility to have more than one chroot environment to be able to have different setups for different machines. Sometimes you may want to have a special setup for a certain pool of machines. Then you can copy the whole chroot environment to a new directory and change there whatever you want. The clients booting from the new environment will have the changes and the other clients won't be touched.

For such a second environment you need to export the second paths like the first ones (see section 8.3 on page 35) and specify another *root-path* in the dhcp config (see section 8.1 on page 32).

9.1 Installing Debian in a chroot Environment

After our server has been installed and is running nicely, we need the environment for the clients. Debian comes with a script that makes the work simple for us:

`debootstrap`.

First step for our client setup is a new directory, where all client related stuff will go. In my setup it is called `/remote`. This directory will be the root to all clients.

9.2 Bootstrapping the chroot Directory

Now we are going to fill the new root directory with life:

```
# debootstrap sarge /remote http://ftp.at.debian.org/debian
```

[1] chroot: change root directory, means that the root of the Linux installation is in another directory

Now it's time again to get a coffee. Depending on your Internet connection it takes some time to get all the necessary files downloaded and installed. The files are extracted and installed into the chroot directory. Do not get alarmed by some warnings. They are normal, since in the beginning some tools are missing.

The program should finish with a line like:

```
I: Base system installed successfully.
```

Then you have made the first step. Now our new system needs some hand tuning.

9.3 Setting up Files in /remote/etc

Now you have to fill

- /remote/etc/hosts,
- /remote/etc/resolv.conf and
- /remote/etc/fstab

with adequate values. You can use and modify the files from your running installation in /etc.

Below you find some example values. Please adjust them to your settings:

/remote/etc/hosts:

```
# /etc/hosts
127.0.0.1      localhost
192.168.1.2   gridmaster.ben.tuwien.ac.at    gridmaster
```

/remote/etc/resolv.conf:

```
# /etc/resolv.conf
search ben.tuwien.ac.at       # put your domain here
nameserver 128.130.2.3        # primary name server
nameserver 128.130.3.131      # secondary name server
```

/remote/etc/fstab:

```
# /etc/fstab: static file system information.
#
# <filesystem> <mountpoint> <type> <options> <dump> <pass>
  proc         /proc        proc   defaults  0      0
  none         /sys         sysfs  defaults  0      0
```

9.4 Configuring the Base System

Now time has come to set up this instance. chroot into the environment via:

```
# chroot /remote/ su -
```

Make sure that the `/proc` file system is mounted with

```
# mount /proc
```

and issue the command to start the configuration:

```
# base-config
```

Just go through the parameters and enter the correct value. If you are asked if you want to "Use a PPP connection to install the system" then you want to answer "NO".

Next is to configure apt (the Debian package management tool). I recommend to use http or ftp to gather the updates (http is the faster though). Security updates from security.debian.org are not a bad idea to enable as well. If you have at least one mirror then you can continue and will be presented what kind of installation you want to do. I recommend to unselect all entries (no asterisks in the menu) and continue. You get the chance to make a much detailed decision which packages you want.

Following is a download of needed packages and the setup. Read through the questions the system asks you and answer them.

For the Exim configuration (Mail Server) I recommend that you choose "no configuration this time" and do the configuration by hand later on (see "Mail server setup" on page 49).

9.5 Setting the Default Keyboard Layout

Default is an English keyboard layout. To change this you can issue:

```
# dpkg-reconfigure console-data
```

Choose "Select keymap from arch list" and then the preferred keyboard layout.

9.6 Installation of Additionally Needed Software

For the remote system to work some additional software is needed. The software can be installed easily by issuing the command

```
# apt-get install PACKAGE-NAME
```

You can either put all packages on one line or do it step by step.

The following packages are needed:

- `busybox`
  BusyBox combines tiny versions of many common UNIX utilities into a single small executable. It provides minimalist replacements for the most common utilities you would usually find on your desktop system (i.e., ls, cp, mv, mount, tar, etc.). The utilities in BusyBox generally have fewer options than their full-featured GNU cousins; however, the options that are included provide the expected functionality and behave very much like their GNU counterparts.

- `cramfsprogs`
  This package contains tools that let you construct a CramFs (Compressed ROM File System) image from the contents of a given directory, as well as checking a constructed CramFs image and extracting its contents.

  Cram file systems are used for Debian INITRD images.

- `initrd-tools`
  This package contains tools needed to generate an initrd image suitable for booting a prepackaged Linux kernel image (as shipped with the Debian main distribution). It does not cater for other uses of initrd at the moment.

- `mknbi`
  With mknbi you can create tagged images for Etherboot. Tagged images are data files, which contain the necessary files for booting up (kernel+root for Linux, kernel+minifs for dos, ...) bundled together with a special format.

  These tagged images are downloaded and understood by Etherboot and Netboot during the boot process.

- `ntpdate`
  NTP, the Network Time Protocol, is used to keep computer clocks accurate over the Internet, or by following an accurate hardware receiver which interprets GPS, DCF-77, NIST or similar time signals.

  ntpdate is a simple NTP client which allows a system's clock to be set to match the time obtained by communicating with one or more servers.

- `udhcpc`
  DHCP is a protocol like BOOTP (actually dhcpd includes much of the functionality of BOOTPD!). It assigns IP addresses to clients based on lease times. This package is primarily geared towards embedded systems. It does however, strive to be fully functional, and RFC compliant.

- `syslinux`
  SYSLINUX is a boot loader for the Linux/i386 operating system which operates off an MS-DOS/Windows FAT filesystem. It is intended to simplify first-time installation of Linux, and for creation of rescue and other special-purpose boot disks.

  It can also be used as a PXE bootloader during network boots.

- `discover`
  Discover is a hardware identification system based on the libdiscover1 library. Discover provides a flexible interface that programs can use to report a wide range of information about the hardware that is installed on a Linux system. In addition to reporting information, discover includes support for doing hardware detection at boot time. Detection occurs in two stages: The first stage, which runs from an initial ramdisk (initrd), loads just the drivers needed to mount the root file system, and the second stage loads the rest (ethernet cards, sound cards, etc.).

- `discover-data`
  The Discover hardware detection library uses XML data files to describe software interfaces to various ATA, PCI, PMCMIA, SCSI, and USB devices. While the Discover library can retrieve data from anywhere on the net, it is often convenient to have a set of Discover XML data files on one's system; thus, this package.

- `hotplug`
  This package contains the scripts necessary for hotplug Linux support, and lets you plug in new devices and use them immediately. It includes support for PCI, Cardbus (PCMCIA), USB and Firewire devices and can automatically configure network interfaces.

- `dash`
  "dash" is a POSIX compliant shell that is much smaller than "bash". We take advantage of that by making it the shell on the installation root floppy, where space is at a premium.

  It can be usefully installed as /bin/sh (because it executes scripts somewhat faster than "bash"), or as the default shell either of root or of a second user with a userid of 0 (because it depends on fewer libraries, and is therefore less likely to be affected by an upgrade problem or a disk failure). It is also useful for checking that a script uses only POSIX syntax.

  "bash" is a better shell for most users, since it has some nice features absent from "dash", and is a required part of the system.

- `udev`
  udev is a program which dynamically creates and removes device nodes from /dev/. It responds to /sbin/hotplug device events and requires a 2.6 kernel.

- `vim` (optional but recommended) and `vim-common`
  Vim is an almost compatible version of the UNIX editor Vi. Many new features have been added: multi level undo, syntax highlighting, command line history, on-line help, filename completion, block operations, folding, Unicode support, etc.

After the installation you can purge the following packages with

```
# dpkg --purge logrotate mdetect
```

- `logrotate`
  The logrotate utility is designed to simplify the administration of log files on a system which generates a lot of log files. Logrotate allows for the automatic rotation compression, removal and mailing of log files. Logrotate can be set to handle a log file daily,

weekly, monthly or when the log file gets to a certain size. Normally, logrotate runs as a daily cron job.

- `mdetect`
  mdetect is a tool for autoconfiguring mice; it is typically used as the backend to some user-friendly frontend code. mdetect writes the autodetected mouse device and protocol (as used by gpm) to standard output. It can be invoked so as to produce output appropriate for XFree86 X server configuration files.

9.7 Preparing `/remote/etc`

To be able to have all needed files from `/remote/etc` in the client environment you need to prepare the chrooted `/etc`:

- `/etc/ssh`:
  To be able to copy the host-keys you need to make the files `ssh_host_dsa_key` and `ssh_host_rsa_key` readable for everyone.

- `/etc/shadow` and `/etc/gshadow`:
  This file should not be made readable by default. Therefore I have set up a cronjob that copies the `/etc/shadow` to `/etc/shadow.copy`. The second file has read permission for the setup script. There it is specially copied to `/etc/shadow` and the permissions are set correctly again.

  The same is done with the file `/etc/gshadow`.

  The entries from root's crontab:

  ```
  # Copy every hour the password files to the remote image.
  0 * * * * cp /etc/passwd /remote/etc/passwd
  0 * * * * cp /etc/shadow /remote/etc/shadow.copy
  ```

- `/etc/fstab`:
  Only put mountpoints for `/proc` and `/sys` in `/etc/fstab`:

  ```
  #/etc/fstab
  proc            /proc   proc    defaults        0       0
  none            /sys    sysfs   defaults        0       0
  ```

- `/etc/syslog.conf`:
  Make syslog log everything to the gridmaster machine instead of locally:

  ```
  #  /etc/syslog.conf     Configuration file for syslogd.
  #
  #  For more information see syslog.conf(5)
  #  manpage.
  *.*     @193.170.74.44
  ```

9.8 Installing the "Right" Kernel

I would recommend that you install the same kernel image in your remote boot environment as you have currently running on your gridmaster boot server. It makes a lot of work easier (e.g. the initrd image generation).

To check which kernel is currently running you can issue the command

```
# uname -a
```

or you still know which kernel you have installed previously. If you still have the `kernel-*.deb` file around, you can install it in the chroot environment.

9.9 Configuration of the initrd Image

The `/boot/initrd.img` contains the needed configuration and programs to mount the root filesystem after the kernel has booted the system. Usually it is used to provide the needed kernel modules to mount a local partition. In our case it is used to mount a NFS exported partition where the root filesystem resides.

9.9.1 `/remote/etc/mkinitrd/modules`

For this purpose it is important that you know which network interface cards (NICs) are in your PCs which you want to boot remotely. Especially you need to know which kernel module supports the cards. These modules need to be added to the `/remote/etc/mkinitrd/modules` file.

Below is an example of the `/remote/etc/mkinitrd/modules` file used by my environment:

```
# /etc/mkinitrd/modules: Kernel modules to load for initrd.
#
# This file should contain the names of kernel modules and
# their arguments (if any) that are needed to mount the root
# file system, one per line.
# Comments begin with a '#', and everything on the line after
# them is ignored.
#
# You must run mkinitrd(8) to effect this change.

af_packet
mii
e100
eepro100
8139too
nfs
acpi
apm
```

9.9.2 /remote/etc/mkinitrd/mkinitrd.conf

This is the config file for the generation of the image file. Make sure that busybox is included here to make the nfs mount script working:

```
# /etc/mkinitrd/mkinitrd.conf:
#  Configuration file for mkinitrd(8).  See mkinitrd.conf(5).
#
# This file is meant to be parsed as a shell script.

# What modules to install.
MODULES=dep

# The length (in seconds) of the startup delay during which
# linuxrc may be interrupted.
DELAY=0

# If this is set to probe mkinitrd will try to figure out
# what's needed to mount the root file system.  This is
# equivalent to the old PROBE=on setting.
ROOT=/dev/hda

# This controls the permission of the resulting initrd image.
UMASK=022

# Command to generate the initrd image.
MKIMAGE='mkcramfs %s %s > /dev/null'

# Set this to yes if you want to use busybox(1).
BUSYBOX=yes

# Set this to no if you want to disable
# /usr/share/initrd-tools/scripts.
PKGSCRIPTS=yes

# This is the value for LD_LIBRARY_PATH when deciding what
# goes onto the image.
INITRD_LD_LIBRARY_PATH=$LD_LIBRARY_PATH
```

9.9.3 NFS Mount Script

To actually mount the nfs root filesystem some scripts are needed. These scripts are included into the initrd image file, which is distributed to the clients with the kernel.

To generate the needed scripts a "Generator Script" is used which I will describe now. I will present some portions of the script here. The whole script can be found at Appendix A.3 on page 83.

- required modules: In the first step the required modules are calculated. It can be done automatically but this didn't suite the heterogeneous environment. So I decided to go with a fixed modules list in `/etc/mkinitrd/modules`. Every module which is in this file is included into the image.
- create various mount points: Next mountpoints are created to be able to hook in the mounted shares from the server. This is needed since the initrd filesystem is generally cramfs, which is read-only.
- copy required modules and files: The needed modules from Step 1 are now copied into the image. Also some needed binaries are copied:
  - udhcpc: Compact dhcp client.
  - shutdown: Binary used to get into sleep mode on shutdown.
  - bash, `/lib/libncurses.so.5`, `/lib/tls/libdl.so.2`: bash for better debugging and the needed libs for bash. These files can be omitted to save space in the image.
- create initrd config scripts: Here we (i) configure networking and fetch nfs parameters via dhcp client and (ii) mount the nfs root filesystem.

  The mkinitrd init runs the files in `/scripts/*` (sourcing .sh files in a subshell). These scripts may in turn call others, e.g. the DHCP client script. (Note that due to the initrd fs being read-only, it's difficult to pass parameters back to the caller (i.e. by writing a state file to be sourced by others). Thus, a writable `/tmp` filesystem is required.)

  i) the DHCP client (`scripts/40networking`) calls `/etc/udhcpc-script` when a lease is acquired (or not acquired—see udhcpc(8)). This client script gets passed the DHCP parameters as env variables and configures the network.

  ii) `45mountnfs` uses the DHCP-supplied 'filename' as the nfs export to use as `/setup`. There is a safeguard implemented in the file to make sure the clients don't hang if something goes wrong. If a mount doesn't work or produces an error, the client is halted immediately (using the `trap` mechanism).

  At the end of `45mountnfs` `/setup/setup_path` is called which generates the rest of the environment. This file is an extra file because whenever you want to change something in the NFS mount script you need to regenerate the initrd image. A description of the `/setup/setup_path` file can be found at Section 9.10 on the following page.

9.9.4 Generating the initrd.img File

After all this setting up we need to generate the image. This is done by issuing (in the chroot environment):

```
# mkinitrd -o /boot/initrd.img
```

when the appropriate kernel is installed in the chroot environment and the same kernel is running on the server. If the kernels differ then you can still generate the proper image by issuing

```
# mkinitrd -o /boot/initrd.img <kernel version>
```

9.10 The Setup Script

After the initial mount of the `/setup` directory (as described in Section 9.9.3 on the page before) the `/setup/setup_path` script is called. Within this script the other necessary directories (`/sbin`, `/bin`, `/lib`, `/root`, `/usr`, `/opt`, `/grid` and `/home` on `/grid/home`) are mounted. `/sbin` is mounted first to be able to set the connection speed of the network interface to 100 MBit Full Duplex.

Furthermore, the `/etc` directory is mounted to `/setup/etc`. This is done with special care, since some files in `/etc` need some special attention on each client.

There are two files (`/setup/etc.ignore` and `/setup/etc.copy`) which instruct the script how to handle the files from `/etc`. Files which are in `/setup/etc.ignore` are ignored completely (e.g. `mtab`) whereas files in `/setup/etc.copy` are copied to the client (e.g. `fstab`). All other files in the `/etc` tree are symlinked to `/setup/etc`:

```
ls -l /setup/etc/ | grep -v "^total" | mawk 'print($9)' |
while read line
do
   erg=‘grep "$line" /setup/etc.ignore‘
   if [ "X$erg" = "X" ]
   then
       erg=‘grep "$line" /setup/etc.copy‘
       if [ "X$erg" = "X" ]
       then
           ln -s "/setup/etc/$line" /etc
       else
           cp -r -p "/setup/etc/$line" /etc
       fi
   fi
done
```

The next step is to treat some special files. In Section 9.7 on page 42 we have prepared `/etc/shadow.copy`, `/etc/gshadow.copy` and `/etc/ssh/*key`.

These files are now used to configure the client properly:

```
# Set up /etc
mv /etc/shadow.copy /etc/shadow
mv /etc/gshadow.copy /etc/gshadow
chown root:shadow /etc/shadow
chown root:shadow /etc/gshadow
chmod 640 /etc/shadow
chmod 640 /etc/gshadow
chmod 600 /etc/ssh/*key
echo "$hostname" > /etc/hostname
```

Additionally the `/var` file structure is created completely in memory on each node. There are several directories needed. Additionally the Mailserver *Exim4* is configured for the node.

After all configuration for the operating system there is some setup necessary for Condor. This is done with an extra condor setup script. See Section 11.1.6 on page 58 regarding the "Condor Client Configuration".

9.11 Preparing the Kernel for Remote Boot

Now it is time to prepare the kernel for remote boot to test our hard work. It depends on the setup of the clients if you need to perform both steps or just one or the other:

9.11.1 Etherboot Setup

For the Etherboot to work you need to generate one image file which contains both kernel and initrd image. Compiled is this file using the `mkelf-linux` binary:

```
# mkelf-linux --append="ks= ramdisk_size=12000" \
  /boot/vmlinuz-2.6.x-y-686 /boot/initrd.img-2.6.x-y-686 > \
  /boot/etherboot.img
```

9.11.2 PXE Setup

For the PXE to work a little more work must be done. Firstly you need to copy the PXE loader into the `/boot` directory:

```
# cp /usr/lib/syslinux/pxelinux.0 /boot
```

PXE also requires a directory `pxelinux.cfg` where it will look for config files. Because more than one system may be booted from the same server, the configuration file name depends on the IP address of the booting machine. PXELINUX will search for its config file on the boot server in the following way:

First, it will search for the config file using its own IP address in upper case hexadecimal, e.g. 192.0.2.91 -> C000025B (you can use the included program "gethostip" to compute the hexadecimal IP address for any host.)

If that file is not found, it will remove one hex digit and try again. Ultimately, it will try looking for a file named "default" (in lower case).

As an example, for 192.0.2.91, it will try C000025B, C000025, C00002, C0000, C000, C00, C0, C, and default, in that order.

You should note that all filename references are relative to the directory of `pxelinux.0` (`/boot`). PXELINUX generally requires that filenames (including any relative path) are 127 characters or shorter in length.

In my setup I have created the `pxelinux.cfg` where a symlink points to the config file in the `/boot` directory:

```
-rw-r--r--  1 root root  pxeconfig
lrwxrwxrwx  1 root root  pxelinux.cfg/default -> ../pxeconfig
```

The `pxeconfig` file holds the necessary components to boot the client:

```
DEFAULT winzig

LABEL winzig
    KERNEL vmlinuz
    APPEND initrd=initrd.img
    IPAPPEND 1
```

You are now set to try and boot a client.

9.11.3 Script for Kernel and initrd.img Preparation

Since setting up the proper Kernel links and the initrd.img file is done quite often I have put together a script which does everything for you (`/remote/usr/local/sbin/gen_initrd`):

```
#!/bin/sh

if [ -f /boot/etherboot ]; then
  rm /boot/etherboot
fi
if [ -f /boot/initrd.img ]; then
  rm /boot/initrd.img
fi
if [ -f /boot/vmlinuz ]; then
  rm /boot/vmlinuz
fi
```

```
mkinitrd -o /boot/initrd.img-‘uname -r‘
mkelf-linux --append="ramdisk_size=15000" \
        /boot/vmlinuz-‘uname -r‘ \
        /boot/initrd.img-‘uname -r‘ > \
        /boot/etherboot-‘uname -r‘


ln -s etherboot-‘uname -r‘ /boot/etherboot
ln -s initrd.img-‘uname -r‘ /boot/initrd.img
ln -s vmlinuz-‘uname -r‘ /boot/vmlinuz
```

9.12 Mail Server Setup

At the basic configuration you have told the installer to keep the mailserver unconfigured. We want to change this now. Edit the file `/etc/exim4/update-exim4.conf.conf` to have suitable values:

```
# /etc/exim4/update-exim4.conf.conf
#
# Edit this file and /etc/mailname by hand and execute
# update-exim4.conf yourself or use
# 'dpkg-reconfigure exim4-config'

dc_eximconfig_configtype='none'
dc_other_hostnames='GRIDMASTER'
dc_local_interfaces='127.0.0.1'
dc_readhost=''
dc_relay_domains=''
dc_minimaldns='false'
dc_relay_nets=''
dc_smarthost='mr.tuwien.ac.at'
CFILEMODE='644'
dc_use_split_config='false'
dc_hide_mailname='false'
dc_mailname_in_oh='true'
```

With this setup you should be able to use the mailserver on all clients. In the client setup script each client is configured during boot.

9.13 Filesystem Layout for Remote Boot Clients

Default directories:

```
/bin
/boot
/dev
/etc
/home
/initrd
/lib
/media
/mnt
/opt
/proc
/root
/sbin
/srv
/sys
/tmp
/usr
/var
```

Special directories dedicated to our setup.

```
/grid
/setup
```

9.14 Kernel Update

I have experienced some ACPI troubles with the standard 2.6.8 kernel that ships with Sarge. Therefore I have updated to the latest kernel from unstable.

First update the Kernel on the Server (see 7.9 on page 31). Reboot the server using the new kernel (Make sure it is the active kernel in your bootloader).

After your server runs fine again, chroot into the remote boot environment, make sure that /proc and /sys (for kernels > 2.6) are mounted and then update the kernel image. While the kernel package generated the initrd image it will also include the nfs stuff for the remote boot. The only thing left for you is to generate the etherboot image which can be done using the generator script described on page 48.

Chapter 10

Configuration of the Remote Grid Nodes

In order to be able to boot the PCs from a remote server the network cards must support either Etherboot or PXE. If your NIC doesn't support it natively you can also do it with a floppy.

On many modern mainboards with on-board network interfaces you can enable the "Network Book Option" in the BIOS. If you have an additional interface card it can be possible to obtain a boot ROM for the cards. Please check with the manufacturer of your motherboard / network card.

10.1 Etherboot, PXE

If your BIOS has support for network boot you have to check if the board follows the Etherboot[1] or PXE[2] (Preboot Execution Environment) standard. The easiest way to do this is to enable network boot and start up the machine. After the BIOS has finished booting the machine the network boot code will kick in. Usually it prompts whether it is a PXE or Etherboot code.

According to this information you need to setup the DHCP server as described on page 32.

10.2 Floppy

If your system is not able to boot from your network card you can check with http://rom-o-matic.net/ if your NIC is supported there. They have over 250 cards supported.

If you have found your card then the simplest way is to get a "Floppy bootable ROM Image". This file you can copy to a disk and then set your computer's BIOS to boot from the floppy.

[1] http://www.etherboot.org/

[2] http://syslinux.zytor.com/pxe.php

10.3 Wake on LAN

To be able to start the clients automatically you can use the wake on LAN capability of most modern network cards. You need to enable the feature in your BIOS.

With the previously installed `wakeonlan` program (Section 8.4 on page 36) you should test the client now, if it really wakes on your request.

If the client doesn't wake make sure you are on the same subnet and no router or firewall is blocking your packages.

Chapter 11

Installation of Condor

I have decided to use Condor [1] as a queueing system. I will provide a short description on how I installed condor on my pool. For detailed installation instructions consult the "Condor® Version 6.7.8 Manual" ([20]).

11.1 First Time Installation

11.1.1 Adding a Condor User

For all Condor related software it is recommended to have a separate user. First add the new user:

```
# useradd -u 5000 -d /grid/condor -m condor
```

and set a password for the new user:

```
# passwd condor
```

The same procedure needs to be done in the chroot environment:

```
# chroot /remote su -
# useradd -u 5000 -d /grid/condor -m condor
# passwd condor
# exit
```

11.1.2 Getting the Software

The Condor software can be found at http://www.cs.wisc.edu/condor/downloads/. I recommend the latest package from the current development Version (V6.7). It has many new features as well as support for the 2.6 kernel series. Also support for "Globus V4" was added recently. (If there is already a new stable version V6.8, try it).

You have to accept the "CONDOR PUBLIC LICENSE" by giving your name and e-mail address. You can also request to be subscribed to the Condor-World mailing list, on which

[1] http://www.cs.wisc.edu/condor/

updates are announced. The Condor-Users mailing list is for discussion between users. If you are stuck you could try your luck there. On the next page you are presented a list of "Currently Available Condor v6.7 Binaries". Scroll down to "Debian Linux 3.1 (sarge)" and download the "tar.gz" binary to the gridmaster server.

I recommend that you dedicate a directory to your condor downloads. I have chosen `/root/SW` as download directory. All following examples are using this path. If you choose another path, your have to substitute the examples accordingly.

11.1.3 Installing the Software

After downloading the package extract the software (Currently version 6.7.8 is the latest):

```
# tar xvfz condor-6.7.8-linux-x86-glibc23-dynamic.tar.gz
```

The Condor installer is extracted to a new directory `condor-6.7.8`. Change to this directory. There you will have a `release.tar` file which contains all necessary files and a `condor_configure` file which is the new installer. There is also an old installer (`condor_install`) but it is recommended to use the new one. The following documentation is only using the new installer:

```
# ./condor_configure \
  --install=/root/SW/condor-6.7.8/release.tar \
  --install-dir=/grid/condor \
  --type=manager,submit \
  --verbose \
  --owner=condor

Condor will be run as user: condor
WARNING: Multiple network interfaces detected. Condor might not
work properly until you set NETWORK_INTERFACE = <interface IP>

Setting CONDOR_ADMIN to "root@gridmaster.ben.tuwien.ac.at"
If this is not your preferred email address, please modify
CONDOR_ADMIN in the configuration file
Setting mail path to: /usr/bin/mail
Setting FILESYSTEM_DOMAIN and UID_DOMAIN to ben.tuwien.ac.at
Unable to find a valid Java installation
Java Universe will not work properly until the JAVA (and
JAVA_MAXHEAP_ARGUMENT) parameters are set in the configuration
file!
Install directory: /grid/condor
Main config file: /grid/condor/etc/condor_config
Local directory: /grid/condor/local.gridmaster
Local config file:
     /grid/condor/local.gridmaster/condor_config.local

Writing settings to file:
```

```
     /grid/condor/local.gridmaster/condor_config.local
CONDOR_HOST=gridmaster.ben.tuwien.ac.at
RELEASE_DIR=/grid/condor
LOCAL_DIR=/grid/condor/local.$(HOSTNAME)
CONDOR_ADMIN=root@gridmaster.ben.tuwien.ac.at
MAIL=/usr/bin/mail
UID_DOMAIN=ben.tuwien.ac.at
FILESYSTEM_DOMAIN=ben.tuwien.ac.at
COLLECTOR_NAME=Personal Condor at gridmaster.ben.tuwien.ac.at
CONDOR_IDS=5000.100
LOCK=/tmp/condor-lock.$(HOSTNAME)0.874316378104599
START=TRUE
SUSPEND=FALSE
PREEMPT=FALSE
KILL=FALSE
DAEMON_LIST=COLLECTOR, MASTER, NEGOTIATOR, SCHEDD

Writing settings to file: /grid/condor/etc/condor_config
CONDOR_HOST=
LOCAL_CONFIG_FILE=
      /grid/condor/local.gridmaster/condor_config.local

Condor has been installed into:
    /grid/condor

In order for Condor to work properly you must set your
CONDOR_CONFIG environment variable to point to your
Condor configuration file:
    /grid/condor/etc/condor_config
before running Condor commands/daemons.
```

When the setup has finished the procedure Condor is installed correctly. Now it needs some more hand-tuning to achieve best usability.

First task is to generate a symbolic link to `etc/condor_config` in the condor home directory.

```
# cd ~condor
# ln -s etc/condor_config
```

Next we need to make condor start every time the gridmaster starts:

```
# cp /grid/condor/etc/examples/condor.boot /etc/init.d/condor
# update-rc.d condor defaults 99 1
```

... and also when a client boots:

```
# cp /etc/init.d/condor /remote/system/etc/init.d/
# chroot /remote su -
  # update-rc.d condor defaults 99 1
  # exit
```

11.1.4 The Global Configuration

Go through the main config file `/grid/condor/etc/condor_config` and check that everything is O.K.

Set some default values:

- `CONDOR_HOST`:
  What machine is your central manager?

  ```
  CONDOR_HOST=gridmaster.your.domain
  ```

- `LOCAL_DIR_HOST`:
  Where is the local condor directory for each host?
  This is where the local config file(s), logs and spool/execute directories are located

  ```
  LOCAL_DIR               = $(RELEASE_DIR)/hosts/$(HOSTNAME)
  ```

- `LOCAL_CONFIG_FILE`:
  What machine is your central manager?

  ```
  LOCAL_CONFIG_FILE      = $(LOCAL_DIR)/condor_config.local
  ```

- `CONDOR_ADMIN`:
  When something goes wrong with condor at your site, who should get the email?

  ```
  CONDOR_ADMIN           = your-address@your.domain
  ```

- `UID_DOMAIN`:
  Internet domain of machines sharing a common UID space. If your machines don't share a common UID space, set it to UID_DOMAIN = $(FULL_HOSTNAME) to specify that each machine has its own UID space.

  ```
  UID_DOMAIN             = your.domain
  ```

- `FILESYSTEM_DOMAIN`:
  Internet domain of machines sharing a common file system. If your machines don't use a network file system, set it to FILESYSTEM_DOMAIN = $(FULL_HOSTNAME) to specify that each machine has its own file system.

  ```
  FILESYSTEM_DOMAIN      = your.domain
  ```

- COLLECTOR_NAME:
  This macro is used to specify a short description of your pool. It should be about 20 characters long.

  ```
  COLLECTOR_NAME                  = My Pool
  ```

- CONDOR_IDS:
  The user/group ID <uid>.<gid> of the "Condor" user.

  ```
  CONDOR_IDS=5000.100
  ```

- USE_NFS:
  Do you want to use NFS for file access instead of remote system calls?

  ```
  USE_NFS = True
  ```

- Setup for MPI jobs:
  Setup for Running Dedicated and Opportunistic Jobs (see "Condor Manual" [20] Section 3.10.10).

  ```
  START                   = True
  CONTINUE                = True
  SUSPEND                 = FALSE
  PREEMPT                 = FALSE
  KILL                    = FALSE

  MPI_CONDOR_RSH_PATH     = $(LIBEXEC)
  WANT_SUSPEND            = False
  WANT_VACATE             = False
  STARTD_EXPRS = $(STARTD_EXPRS),WANT_SUSPEND,WANT_VACATE
  ```

  For details about this file look at section "3.3 Configuration" of the Condor Manual [20]. The full condor_config file is attached at appendix B.2 on page 94.

11.1.5 Gridmaster Configuration

Since we have configured that all host config files are in a new directory called `/grid/condor/hosts` we need to create this directory and put the local configuration file for gridmaster in there:

```
# mkdir hosts
# mv local.gridmaster/ hosts/gridmaster
```

Next you need to edit the local Condor configfile for gridmaster `hosts/gridmaster/condor_config.local`. Since most of the definitions are already done in the global config file you can remove them.

Actually the whole file only has the following definitions:

```
##  condor_master
##  Daemons you want the master to keep running for you:

DAEMON_LIST = MASTER,COLLECTOR,NEGOTIATOR,SCHEDD,CKPT_SERVER

## If the dedicated scheduler has resources claimed, but nothing to
## use them for (no MPI jobs in the queue that could use them), how
## long should it hold onto them before releasing them back to the
## regular Condor pool? Specified in seconds. Default is 10
## minutes.
## If you define this to '0', the schedd will never release claims
## (unless the schedd is shutdown).  If your dedicated resources
## are configured to only run jobs, you should probably set this
## attribute to '0'
UNUSED_CLAIM_TIMEOUT = 600
```

Furthermore, if you want to be able to compile binaries for the condor standard universe you need to exchange the linker (see Condor Manual [20] Section 3.10.3):

```
# cp /usr/bin/ld /usr/bin/ld.real
# cp ~condor/lib/ld  /usr/bin/ld
# chown root /usr/bin/ld
# chmod 755 /usr/bin/ld
```

11.1.6 Client Configuration

Since every client needs its own `/grid/condor/hosts` directory I have made a setup script (`/grid/condor/hosts/setup_host`) which is executed every time a host boots.

The script not only checks for the directory and in case it doesn't exist creates it, but also sets some parameters for the hosts. Those parameters can be used to specify special needs. Currently the following parameters are set:

- `WINZIG_CPU_SSE2`
- `WINZIG_CPU_SPEED`
- `WINZIG_CPU_ROOM`

For the host directory I have put together a `SKELETON` directory:

```
root@gridmaster:/grid/condor/hosts# ls -Al SKELETON/
-rw-r--r-- 1 root root   72 2005-02-22 14:00 condor_config.local
drwxr-xr-x 2 root root 4096 2005-02-09 10:54 execute
drwxr-xr-x 2 root root 4096 2005-02-09 10:54 log
drwxr-xr-x 2 root root 4096 2005-02-09 10:54 spool
```

The `execute`, `log` and `spool` directories are completly empty. The `condor_config.local` file has only two directives in it:

```
DAEMON_LIST = MASTER, STARTD
CONDOR_HOST = gridmaster.ben.tuwien.ac.at
```

The `setup_host` script can be found at appendix B.1 on page 93. It is called by the `setup_path` script (see appendix A.4 on page 89) using the user condor:

```
su - condor -c "/grid/condor/hosts/setup_host"
```

11.2 Update Procedure

The Condor Team is working all the time on updates and improvements to their software. Therefore at some time you should think about updating the condor software. This procedure is not very troublesome and described below.

First you need to get the new software (see 11.1.2 on page 53).

The next step is to extract the new version:

```
# tar xvfz condor-6.7.9-linux-x86-glibc23-dynamic.tar.gz
```

... stop Condor:

```
# /etc/init.d/condor stop
```

... and install the software:

```
root@gridmaster:~/SW/condor-6.7.9# condor_configure \
    --install=/root/SW/condor-6.7.9/release.tar \
    --install-dir=/grid/condor--verbose --owner=condor
Condor will be run as user: condor

This is an upgrade installation. Will not modify config files.

Install directory: /grid/condor
Main config file: /grid/condor/etc/condor_config
Local directory: /grid/condor/hosts/gridmaster
Local config file: /grid/condor/hosts/gridmaster/\
                                          condor_config.local

Condor has been installed into:
    /grid/condor

In order for Condor to work properly you must set your
CONDOR_CONFIG environment variable to point to your
Condor configuration file:
    /grid/condor/etc/condor_config
before running Condor commands/daemons.
```

After the installation has finished you should be set to start Condor again:

```
# /etc/init.d/condor start
```

If your cluster was still running, then the nodes should show up again. This can take some time.

If a node restarts then also the new version of Condor is started. No installation needs to be done in the remote client tree since there is only one Condor installation.

11.3 Checkpoint Server

If you want to enable the Condor Checkpoint mechanism (see Condor Manual [20] Section 4.2) you need to install an additional package and configure Condor accordingly.

First you need to get the software. This can be achieved as described in Section 11.1.2 on page 53. On the download page choose "Checkpoint Server, PVM, Condor View Server and Client, Condor Analyze" and after the Condor License Page download the "Checkpoint Server" for "Debian Linux 3.1 (sarge)".

After you downloaded the `ckpt_server-6.7.0-linux-x86-glibc23-dynamic.tar.gz` file unpack it. In the new directory you will find a `ckpt_server.tar` file. Go to your condor installation directory and extract the file there:

```
# cd ~condor
# tar xvf /path/to/ckpt_server.tar

# add condor_config.local.ckpt.server to etc/condor_config
# add condor_config.local.root.ckpt.server to
    hosts/gridmaster/condor_config.local
```

After you restart Condor the checkpoint server should also run.

Chapter 12

Installing Additional Intel Compiler Suites and the Intel Math Kernel Library (MKL)

To achieve top performance on Intel systems it is necessary to have the best compilers you can get. The GCC suite is very stable and achieves a very good performance but to go to the limit the Intel Compilers are your choice. I have done some comparisons in my experiments which show that the installation of the Intel Compiler Suite is recommended.

Additionally there is a Intel Math Kernel Library (MKL) available which pushes performance again.

The Intel Compiler Suites (C++ and Fortran) and MKL are shipped using RPM packages. These packages need some special attention to get Debian packages from them. See Section 12.2 on the current page.

12.1 Getting the Software

For non-commercial purposes Intel provides free Compilers. There is also a 30 day evaluation license available.

To obtain the software go to the Intel homepage at http://www.intel.com and follow the path "Software Development", "Software Products", "Compilers" to "Intel®C++ Compiler for Linux" or "Intel®Fortran Compiler for Linux".

The MKL can also be found at http://www.intel.com, "Software Development", "Software Products", "Performance Libraries", "Intel®Math Kernel Library".

12.2 How to Generate the .deb Files from Intel's .rpm Files

After you have downloaded the packages you will have a `.tar.gz` or `.tar` file. Unpack it (`tar xvfz filename`) and change into the new directory. There you will have several `.rpm` files.

Davor Ocelic has put together some notes on the net[1]. I will state the main steps here. For

[1] http://colt.projectgamma.com/intel/ixc-debian.html

the complete reference please visit his page.

If you are doing this on a (still) standard 32-bit PC, it's safe to just delete all files with '64' in their name (rm -rf *64*), as they won't be of use to you. (or the other way around, delete *32* if you have a 64bit machine). Then, use the alien (apt-get install alien) program to convert remaining `.rpm` files to `.deb`; here is a "session transcript":

```
# tar zxf l_cc_p_8.0.055.tar.gz
# cd l_cc_p_8.0.055
# rm -rf *64*
# alien *.rpm
# rm *.rpm
```

So far so good. Now we need to modify the *postinst* script of **only** the main `.deb` package (`intel-icc8_8.0-45_i386.deb` in this case), and rebuild the `.deb`. Let's first extract the `.deb` to a temporary location (`tmp/`) and add one line to the *postinst* script:

```
# mkdir tmp
# dpkg-deb -e intel-icc8_8.0-45_i386.deb tmp/DEBIAN
# dpkg-deb -x intel-icc8_8.0-45_i386.deb tmp/
# echo DESTINATION=/opt/‘ls tmp/opt/‘/ >> \
             tmp/DEBIAN/postinst
```

Now open the original *install.sh* script (should be in your current directory), and search for string "for FILE" in it. You should see a few "for FILE" blocks listed one by one in a row (all basically performing various substitutions on files in the DESTINATION directory, if you understand a little shell or sed syntax). Those few blocks are the ones you need to copy and paste at the end of `tmp/DEBIAN/postinst`.

```
for FILE in $(find $DESTINATION/bin/ -regex \
    '.*[ei](cc|fort|fc|cpc)$\|.*cfg$\|
               .*pcl$\|.*vars[^/]*.c?sh$' \
     2> /dev/null) ; do
  sed s@$DESTINATION@g $FILE > $FILE.abs
  mv $FILE.abs $FILE
  chmod 755 $FILE
done

for FILE in $(find $DESTINATION/bin/ -regex '.*[ei]cc' \
     2> /dev/null) ; do
  sed s@$DESTINATION@g $FILE > $FILE.abs
  mv $FILE.abs $FILE
  chmod 755 $FILE
```

```
done

for FILE in $(find $DESTINATION/bin/ -regex '.*[ei]cpc' \
     2> /dev/null) ; do
  sed s@$DESTINATION@g $FILE > $FILE.abs
  mv $FILE.abs $FILE
  chmod 755 $FILE
done

for FILE in $(find $DESTINATION/bin/ -regex '.*[ei]fort' \
     2> /dev/null) ; do
  sed s@$DESTINATION@g $FILE > $FILE.abs
  mv $FILE.abs $FILE
  chmod 755 $FILE
done

for FILE in $(find $DESTINATION/bin/ -regex '.*[ei]fc' \
     2> /dev/null) ; do
  sed s@$DESTINATION@g $FILE > $FILE.abs
  mv $FILE.abs $FILE
  chmod 755 $FILE
done
```

But do not directly copy this, do open your *install.sh* script and find this piece yourself since it could change at any time.

Now just rebuild the *.deb* and install it (first this "main" deb, then the rest of them):

```
# dpkg-deb -b tmp intel-icc8_8.0-45_i386.deb
# dpkg -i intel-icc8_8.0-45_i386.deb
# dpkg -i intel-iidb7_7.3.1-86_i386.deb
# dpkg -i --force-overwrite intel-isubh8_8.0-45_i386.deb
```

And copy the license file you received in your email to the appropriate place:

```
# cp /where-ever/it/is/l_cpp_XXXXXXXX.lic \
     /opt/intel_cc_80/licenses/
```

To setup your environment, run on the command line or put in startup scripts (like /.bash_profile):

```
. /opt/intel_cc_80/bin/iccvars.sh
```

The same steps need to be done if you want to use the Fortran suite. It looks like quite a lot of work but it actually is not.

Chapter 13

Condor Usage

In this chapter you will find a short manual how to use the Condor system. The commands used to control condor can be found on this page.

The complete Condor manual can be found on the Condor homepage:
http://www.cs.wisc.edu/condor/manual/

13.1 User Configuration

For easier access to all Condor programs it is recommended to add the path to your Condor installation to your environment.

Depending on your login shell you need to add a path variable to the config file.

For BASH you need to edit the file `.bashrc` located in your home directory and add the following to the end of this file:

```
PATH="$PATH":/grid/condor/bin
export PATH
```

After you login again you should be able to call `condor_version` and the program should be found.

13.2 Condor Commands

The following commands are the main commands you will come accoss using Condor:

- `condor_submit`:
  is used to submit a new job to the Condor system using a submission file. Details about this file can be found on the next page.
- `condor_q`:
  prints out the current jobs in the Queue with their current status. If you have troubles with a job not starting to execute try `condor_q-analyze`. You will be presented a list of possibilities why the job doesn't start.

- `condor_reschedule`:
  calls the Condor scheduler daemon to reschedule all idle jobs.
- `condor_status`:
  shows the current state of your cluster.
- `condor_rm`:
  If a job doesn't execute or has some other error, you can remove it from the queue with `condor_rm`.
- `condor_history`:
  Shows the history of the jobs executed on the cluster.
- `condor_compile`:
  Used to compile a binary for the Standard Universe.

13.3 Submitting a Job

A job is submitted for execution to Condor using the *condor_submit* command. *condor_submit* as an argument the name of a file called a submit description file. This file contains commands and keywords to direct the queuing of jobs. In the submit description file, Condor finds everything it needs to know about the job. Items such as the name of the executable to run, the initial working directory, and command-line arguments to the program all go into the submit description file.

The following examples are taken from [20], Section 2.5:

13.3.1 Example 1

Example 1 is the simplest submit description file possible. It queues up one copy of the program *foo* (which had been created by *condor_compile*) for execution by Condor. Since no platform is specified, Condor will use its default, which is to run the job on a machine which has the same architecture and operating system as the machine from which it was submitted. No `input`, `output`, and `error` commands are given in the submit description file, so the files `stdin`, `stdout`, and `stderr` will all refer to `/dev/null`. The program may produce output by explicitly opening a file and writing to it. A log file, `foo.log`, will also be produced that contains events the job had during its lifetime inside of Condor. When the job finishes, its exit conditions will be noted in the log file. It is recommended that you always have a log file so you know what happened to your jobs.

```
####################
#
# Example 1
# Simple condor job description file
#
####################
Executable = foo
Log = foo.log
Queue
```

13.3.2 Example 2

Example 2 queues two copies of the program *mathematica*. A requirement is that the program gets executed on a host that has the SSE2 CPU extention. The first copy will run in directory `run_1`, and the second will run in directory `run_2`. For both queued copies, `stdin` will be `test.data`, `stdout` will be `loop.out`, and `stderr` will be `loop.error`. There will be two sets of files written, as the files are each written to their own directories. This is a convenient way to organize data if you have a large group of Condor jobs to run. The example file shows program submission of *mathematica* as a vanilla universe job. This may be necessary if the source and/or object code to program *mathematica* is not available.

```
####################
#
# Example 2: demonstrate use of multiple
# directories for data organization.
#
####################
Executable = mathematica
Universe = vanilla
Requirements = WINZIG_CPU_SSE2
input = test.data
output = loop.out
error = loop.error
Log = loop.log

Initialdir = run_1
Queue
Initialdir = run_2
Queue
```

More details can be found in the Condor User's Manual ([20]).

Chapter 14

Experiments

To test the performance of the WINZIG Grid I did some experiments. I have tested the software developed by IuE with the test cases they provided. They all ran without problems.

Further more we did some linpack benchmarking. I evaluated the performance of the whole grid but there the bottleneck was the underlying Gigabit Network in the backbone through which the different rooms are connected. This experiment was actually already harmful to the network since all bandwidth was consumed.

Therefore we later decided to only make room-wise tests with the linpack suite.

To be able to compare the results we have benchmarked a recent cluster bought at the Vienna University of Technology. I will provide these results for comparison as well.

14.1 IuE: Minimos NT

> The main research field [of the Institute for Microelectronics at the Vienna University of Technology] is the development of software tools for the simulation of semiconductor devices and technological process steps. These tools are applied either to the improvement of existing ULSI semiconductor devices or to support the development of new technologies. As the institute does not have any direct access to fabrication facilities the successful research strongly relies on intensive co-operation with national, European, and international industrial partners. Furthermore, the institute is involved in Austrian and European research programs.
>
> http://www.iue.tuwien.ac.at/research.0.html

14.1.1 Installation

The IuE kindly provided their software for testing. After downloading, install the software with the provided installer into `/opt/minimos-nt`.

14.1.2 Job Submission

To be able to process the jobs properly the environment needs to be set up properly. There is a setup script in the Minimos distribution. It only needs to be sourced on login. This can be achieved by putting it to your `.bashrc` file:

```
. /opt/minimos-nt/bashrc_input
```

If you login again, you should see the Minimos setup:

```
Last login: Sun Jul 10 11:21:17 2005 from some.host
VISTA setup done
iue@gridmaster:~$
```

With this setup done copy the example files to a new directory `data/` and setup a Condor submission file (see Section 13.3 on page 65 for details):

```
Executable = /opt/minimos-nt/bin/mmnt
Universe = vanilla

initialdir = /grid/home/iue/data

output = logs/condor.$(cluster).$(process).out
error =  logs/condor.$(cluster).$(process).err
Log =    logs/log

environment = HOME=$ENV(HOME);VISTA=$ENV(VISTA);PATH=$ENV(PATH)

notification =  complete
notify_user  = your.name@your.domain

arguments  = amp-hydro.ipd
Queue
arguments  = amp.ipd
Queue
arguments  = bjt-hydro.ipd
Queue
arguments  = bjt.ipd
Queue
```

Make sure the `logs/` directory exists or Condor won't start the job. After the submission of the job `condor_submitfile` the job starts executing and you will receive an email upon completion.

14.1.3 Results

The testjobs did all execute and produced the expected results. As examples some results from the test runs:

- `amp-hydro.ipd`:

```
Real Time:           0 00:01:35
Virtual Image Size:  27628 Kilobytes
```

  The complete Results can be found at Appendix C.1 on page 118.

- `amp.ipd`:

```
Real Time:           0 00:14:21
Virtual Image Size:  25388 Kilobytes
```

- `bjt-hydro.ipd`:

```
Real Time:           0 00:00:12
Virtual Image Size:  22404 Kilobytes
```

- `bjt.ipd`:

```
Real Time:           0 00:00:14
Virtual Image Size:  6 Kilobytes
```

14.2 Benchmarking the Cluster with High Performance Computing Linpack Benchmark (HPL)

The goal of this experiment was to be able to tell what performance can be achieved by such a setup. To state up front, the setup is not a high performance cluster! But there is still some computing power available especially as there are many nodes. Still I wanted to have hard numbers to be able to compare my setup to other clusters.

14.2.1 Preparation

Condor

To be able to run MPI[1] jobs in the Condor environment you need to setup Condor as mentioned in Section 11.1.4 "Setup for MPI jobs" (on page 57).

MPICH

Additionally the MPI software needs to be installed. Currently Condor only supports Version 1.2.4 of MPICH[2]. There are still Debian packages around. They cannot be found in the distri-

[1] MPI is a library specification for message-passing, proposed as a standard by a broadly based committee of vendors, implementors, and users.
See http://www-unix.mcs.anl.gov/mpi/

[2] MPICH is a freely available, portable implementation of MPI.
See http://www-unix.mcs.anl.gov/mpi/mpich/

bution but on the Internet. If you can't find them anymore it is not very hard to compile them yourself.

To use the Intel compiler with mpicc you need a config file for mpicc:
`/usr/lib/mpich/etc/mpicc-icc.conf`

```
#! /bin/sh
# icc compiler configuration
# Philipp Kolmann
# kolmann@zid.tuwien.ac.at
# 2005-04-06
#
# Directory locations: Fixed for any MPI implementation
prefix=/usr/lib/mpich
exec_prefix=$prefix
sysconfdir=$exec_prefix/etc
includedir=$prefix/include
libdir=$exec_prefix/lib
#
CC="icc"
CLINKER="icc"
USER_CFLAGS="-fpic "
#
```

HPL

The High Performance Computing Linpack Benchmark software can be obtained from the net[3]:

> HPL is a software package that solves a (random) dense linear system in double precision (64 bits) arithmetic on distributed-memory computers. It can thus be regarded as a portable as well as freely available implementation of the High Performance Computing Linpack Benchmark.
>
> The algorithm used by HPL can be summarized by the following keywords: Two-dimensional block-cyclic data distribution - Right-looking variant of the LU factorization with row partial pivoting featuring multiple look-ahead depths - Recursive panel factorization with pivot search and column broadcast combined - Various virtual panel broadcast topologies - bandwidth reducing swap-broadcast algorithm - backward substitution with look-ahead of depth 1.
>
> The HPL package provides a testing and timing program to quantify the accuracy of the obtained solution as well as the time it took to compute it. The best performance achievable by this software on your system depends on a large variety of factors. Nonetheless, with some restrictive assumptions on the interconnection network, the algorithm described here and its attached implementation are scalable in the sense that their parallel efficiency is maintained constant with respect to the per processor memory usage.

[3] http://www.netlib.org/benchmark/hpl/

The HPL software package requires the availability on your system of an implementation of the Message Passing Interface MPI (1.1 compliant). An implementation of either the Basic Linear Algebra Subprograms BLAS or the Vector Signal Image Processing Library VSIPL is also needed. Machine-specific as well as generic implementations of MPI, the BLAS and VSIPL are available for a large variety of systems.

ATLAS

For the gcc tests a BLAS implementation is needed. The ATLAS package[4] provides the needed BLAS subroutines.

Install the ATLAS Package suitable for your system (here for SSE2 CPUs):

```
# apt-get install atlas3-sse2 atlas3-sse2-dev
```

14.2.2 Compilation

To compile the HPL benchmark you need machine specific make files:

Compilation Using gcc and ATLAS

Special care needs to be given to the "Linear Algebra library" section of the Makefile:

```
# ----------------------------------------------------------------
# - Linear Algebra library (BLAS or VSIPL) -----------------------
# ----------------------------------------------------------------
# LAinc tells the  C  compiler where to find the Linear Algebra
# library header files,  LAlib  is defined  to be the name of  the
# library to be used. The variable LAdir is only used for defining
# LAinc and LAlib.
#
LAdir        = /usr/lib/sse2
LAinc        =
LAlib        = $(LAdir)/libcblas.a $(LAdir)/libatlas.a
```

The complete file to be put in the root of the HPL suite can be found in Appendix C.2 on page 125.

To compile just issue

```
$ make arch=Linux_WINZIG_mpi_gcc
```

After compilation the binary can be found in `bin/Linux_WINZIG_mpi_gcc/xhpl`.

[4] Automatically Tuned Linear Algebra Software, SSE2 shared ATLAS is an approach for the automatic generation and optimization of numerical software. Currently ATLAS supplies optimized versions for the complete set of linear algebra kernels known as the Basic Linear Algebra Subroutines (BLAS), and a subset of the linear algebra routines in the LAPACK library.
See: http://math-atlas.sourceforge.net/

Compilation Using icc and Intel MKL

For compilation with the Intel icc and MKL library you need to set the "Linear Algebra library" include to the MKL:

```
# ----------------------------------------------------------------
# - Linear Algebra library (BLAS or VSIPL) -----------------------
# ----------------------------------------------------------------
# LAinc tells the  C  compiler where to find the Linear Algebra
# library header files,  LAlib  is defined  to be the name of  the
# library to be used. The variable LAdir is only used for defining
# LAinc and LAlib.
#
LAdir        = /opt/intel/mkl72/lib/32
LAinc        = -I$(LAdir) -I/usr/include \
               -I/opt/intel_cc_80/include \
               -I/usr/lib/gcc-lib/i486-linux/3.3.5/include
LAlib        = -L$(LAdir) -lpthread -lmkl -lguide
```

and instruct the compiler to use icc:

```
CC           = $(MPdir)/bin/mpicc -config=icc
LINKER       = $(MPdir)/bin/mpicc -config=icc
```

The complete file to be put in the root of the HPL suite can be found in Appendix C.3 on page 128.

To compile just issue

```
$ make arch=Linux_WINZIG_mpi_icc
```

After compilation the binary can be found in `bin/Linux_WINZIG_mpi_icc/xhpl`.

14.2.3 Submission

The benchmark is submitted using the following submission file:

```
######################################
## MPI example submit description file
######################################
universe = MPI

executable = xhpl_Linux_WINZIG_mpi_icc

log = log/logfile
output = log/outfile.$(cluster)
error = log/errfile.$(cluster)
```

```
notification =  complete
notify_user  = kolmann@zid.tuwien.ac.at

Requirements = (WINZIG_CPU_SSE2 == TRUE) && \
        ((WINZIG_ROOM == "FH1") || (WINZIG_ROOM == "FH1a"))

environment = P4_GLOBMEMSIZE=200000000;\
              LD_LIBRARY_PATH=$ENV(LD_LIBRARY_PATH);\
              PATH=$ENV(PATH)

machine_count = 32
queue
```

14.2.4 Results and Comparison to cluster04

The following results are taken from the HPL benchmark running on 32 nodes of the WINZIG Grid using the 100MBit Ethernet as MPI transport layer. The best results were made using a binary compiled with Intel's icc and MKL.

The other results were taken from a Opteron Cluster using different means of MPI transportation. The Opteron nodes have 2 CPUs each so there are also differences using 32 nodes with only one CPU per node and using 16 nodes with both CPUs per node.

We have done the benchmark using 100 MBit and Gigabit Ethernet as well as Myrinet and IP-over-Myrinet. Below you can find the results:

		partitioning blocking factor				
		180	190	200	210	220
WINZIG	100MBit 32 Nodes (gcc, ATLAS)	27.26	26.90	26.27	25.79	28.22
	100MBit 32 Nodes (icc, MKL)	29.71	29.59	29.37	29.08	29.07
Opteron Cluster	100MBit 16 Nodes	29.18	28.98	28.67	28.95	28.49
	100MBit 32 Nodes	29.10	28.97	28.75	29.04	28.58
	Gbit 16 Nodes	79.59	79.88	79.45	78.74	75.99
	Gbit 32 Nodes	85.77	84.49	85.84	83.70	83.83
	Myrinet 16 Nodes	98.26	94.83	97.41	94.87	94.64
	Myrinet 32 Nodes	98.40	96.73	97.50	96.97	94.04
	IP-over-Myrinet 32 Nodes	90.76	89.89	89.11	89.01	87.80

Table 14.1: Comparison results in Gflop/s.

As you can see the main bottleneck of the setup is the 100 MBit network. But still there is quite some computing power available.

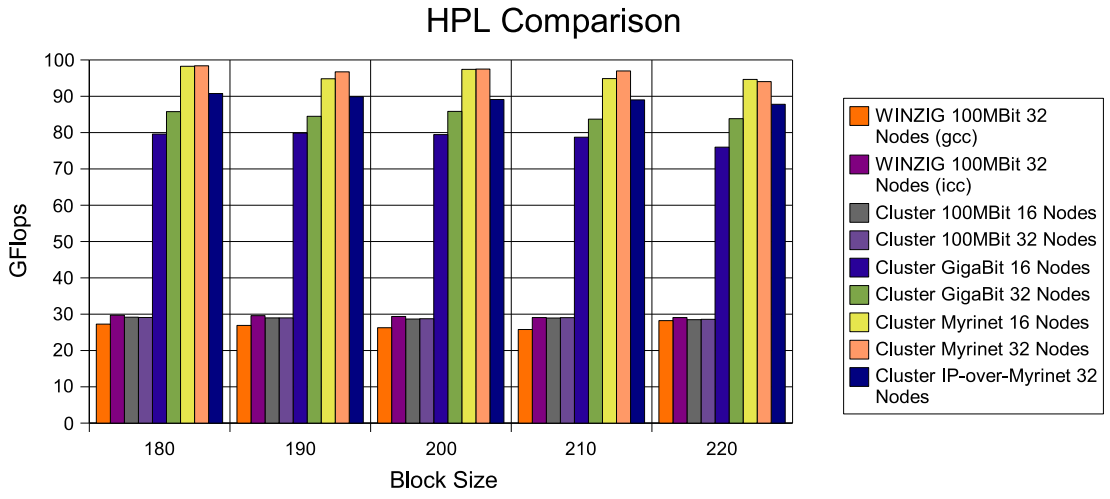

Figure 14.1: Comparison Chart of HPL.

Chapter 15

Conclusio Prospectusque

In this thesis several possible software packages were evaluated to see if they could fit the needs to gather the idle computers of the Information Technology Services (ZID) into a computational grid.

The current setup of the TU Wien ZID Grid—WINZIG—is already actively used and the results look quite promising. Since there is no guarantee that every node is up every day there are fluctuations every night. But the average number of nodes is well above 100.

In the future collaboration with other grid projects will be sought and more researchers will be encouraged to use WINZIG.

On the software side a collaboration with the Austrian Grid Initiative is intended.

Just recently a new version of Condor was released that features a new parallel universe which supports MPI versions greater that 1.2.4. This is certainly a version that needs to be tested.

Bibliography

[1] Grid Architecture.
http://gridcafe.web.cern.ch/gridcafe/gridatwork/architecture.html.

[2] R.J. Allan and M. Ashworth. A Survey of Distributed Computing, Computational Grid, Meta-Computing and Network Information Tools. 2001.
http://www.ukhec.ac.uk/publications/reports/survey.pdf.

[3] Osamu Aoki et al. Debian Reference. 2005.
http://qref.sourceforge.net/.

[4] Thomas Bauer. Das Condor-Batchsystem im Rahmen des Morfeus-Projektes. 2004.
http://www.uni-muenster.de/IVVNWZ/Morfeus/CondorDoku.pdf.

[5] Fran Berman, Geoffrey C. Fox, and Anthony J. G. Hey. *Grid Computing - Making the Global Infrastructure a Reality*. John Wiley & Sons Ltd., 2003.

[6] Rajkumar Buyya, David Abramson, and Srikumar Venugopal. The Grid Economy. 2004.
http://ieeexplore.ieee.org/iel5/5/30407/01398022.pdf.

[7] Charlie Catlett. Standards for Grid Computing: Global Grid Forum. *Journal of Grid Computing*, pages 3–7, 2003.

[8] R. Droms. Dynamic Host Configuration Protocol. 1997.
http://www.ietf.org/rfc/rfc2131.txt.

[9] I. Foster and C. Kesselman. *The Grid: Blueprint for a Future Computing Infrastructure*. San Mateo, CA: Morgan Kaufmann, 1999.

[10] Ian Foster. The Grid: Computing without Bounds. *Scientific American*, April 2003:60–67.

[11] Ian Foster. The Grid: A New Infrastructure for 21st Century Science. *Physics Today*, 2002.
http://www.aip.org/pt/vol-55/iss-2/p42.html.

[12] Ian Foster. What is the Grid? A Three Point Checklist. 2002.
http://www-fp.mcs.anl.gov/ foster/Articles/WhatIsTheGrid.pdf.

[13] Ian Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke. *The Physiology of the Grid*, An Open Grid Services Architecture for Distributed Systems Integration.
http://www.globus.org/alliance/publications/papers/ogsa.pdf.

[14] Ian Foster, Carl Kesselman, and Steven Tuecke. The Anatomy of the Grid. 2001. http://www-unix.globus.org/alliance/publications/papers/anatomy.pdf.

[15] Wolfgang Gentzsch. Grid Computing, A Vendor's Vision. 2002. http://ieeexplore.ieee.org/iel5/7937/21887/01017148.pdf.

[16] Advanced Micro Devices Inc. Magic Packet Technology. 1998. http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/20213.pdf.

[17] H.N. Lim Choi Keung, J.R.D. Dyson, S.A. Jarvis, and G.R. Nudd. Performance Evaluation of a Grid Resource Monitoring and Discovery Service. 2003. http://ieeexplore.ieee.org/iel5/5658/27809/01240222.pdf.

[18] Zsolt Németh and Vaidy Sunderam. Characterizing Grids: Attributes, Definitions, and Formalisms. 2003. http://www.lpds.sztaki.hu/ zsnemeth/pub/jogc03.pdf.

[19] Bill Nowicki. NFS: Network File System Protocol Specification. 1989. http://www.ietf.org/rfc/rfc1094.txt.

[20] Condor Team University of Wisconsin-Madison. *Condor® Version 6.7.8 Manual*. Condor Team University of Wisconsin-Madison, 2005. http://www.cs.wisc.edu/condor/manual/.

[21] Liang Peng, Simon See, Jie Song, Appie Stoelwinder, and Hoon Kang Neo. Benchmark Performance on Cluster Grid with NGB. 2004. http://www.lpds.sztaki.hu/ zsnemeth/pub/jogc03.pdf.

[22] Daniel A. Reed and Celso L. Mendes. Intelligent Monitoring for Adaptation in Grid Applications. 2005. http://ieeexplore.ieee.org/iel5/5/30187/01386660.pdf.

[23] David De Roure, Mark A. Baker, Nicholas R. Jennings, and Nigel R. Shadbolt. The Evolution of the Grid. http://www.semanticgrid.org/documents/evolution/evolution.pdf.

[24] Keith Seymour, Asim YarKhan, Sudesh Agrawal, and Jack Dongarra. NetSolve: Grid Enabling Scientific Computing Environments. 2005. http://icl.cs.utk.edu/projectsfiles/rib/pubs/netsolve-grandinetti-book.pdf.

[25] K. Sollins. The TFTP Protocol (Revision 2). 1992. http://www.ietf.org/rfc/rfc1350.txt.

[26] Sathish S. Vadhiyar and Jack J. Dongarra. A Performance Oriented Migration Framework For The Grid. 2001. http://icl.cs.utk.edu/news_pub/submissions/vadhiyar-migration.pdf.

[27] Sathish S. Vadhiyar and Jack J. Dongarra. GrADSolve - a Grid-based RPC system for Parallel Computing with Application-Level Scheduling. *Journal of Parallel and Distributed Computing*, 2003.

[28] J. Zanghellini, M. Kitzler, C. Fabian, T. Brabec, and A. Scrinzi. An MCTDHF Approach to Multielectron Dynamics in Laser Fields. 2002.
http://eos.photonik.tuwien.ac.at/physics/LSPH1064.pdf.

Appendix A

Remote Setup Scripts

This appendix contains some scripts and config files which have been written or adopted for the WINZIG project.

A.1 /etc/lilo.conf

```
# Generated by liloconfig

raid-extra-boot=/dev/hda,/dev/hdc

# This allows booting from any partition on disks with more than 1024
# cylinders.
lba32

# Specifies the boot device
boot=/dev/md0

# Specifies the device that should be mounted as root.
# If the special name CURRENT is used, the root device is set to the
# device on which the root file system is currently mounted. If the root
# has been changed with  -r , the respective device is used. If the
# variable ROOT is omitted, the root device setting contained in the
# kernel image is used. It can be changed with the rdev program.
root=/dev/md0

# Bitmap configuration for /boot/sarge.bmp
#bitmap=/boot/sarge.bmp
#bmp-colors=1,,0,2,,0
#bmp-table=120p,173p,1,15,17
#bmp-timer=254p,432p,1,0,0

# Enables map compaction:
# Tries to merge read requests for adjacent sectors into a single
# read request. This drastically reduces load time and keeps the map
# smaller. Using COMPACT is especially recommended when booting from a
# floppy disk.
# compact

# Install the specified file as the new boot sector.
# LILO supports built in boot sectory, you only need
# to specify the type, choose one from 'text', 'menu' or 'bitmap'.
# new: install=bmp      old: install=/boot/boot-bmp.b
# new: install=text     old: install=/boot/boot-text.b
# new: install=menu     old: install=/boot/boot-menu.b or boot.b
# default: 'menu' is default, unless you have a bitmap= line
# Note: install=bmp must be used to see the bitmap menu.
install=menu
#install=bmp

# Specifies the number of _tenths_ of a second LILO should
# wait before booting the first image.  LILO
# doesn't wait if DELAY is omitted or if DELAY is set to zero.
# delay=20

# Prompt to use certaing image. If prompt is specified without timeout,
# boot will not take place unless you hit RETURN
prompt
timeout=50

# Specifies the location of the map file. If MAP is
# omitted, a file /boot/map is used.
map=/boot/map

# Specifies the VGA text mode that should be selected when
# booting. The following values are recognized (case is ignored):
#   NORMAL  select normal 80x25 text mode.
#   EXTENDED  select 80x50 text mode. The word EXTENDED can be
#     abbreviated to EXT.
```

```
#   ASK  stop and ask for user input (at boot time).
#   <number>  use the corresponding text mode. A list of available modes
#     can be obtained by booting with  vga=ask  and pressing [Enter].
vga=normal

# These images were automagically added. You may need to edit something.

image=/vmlinuz
        label="linux"
        initrd=/initrd.img
        read-only

image=/vmlinuz.old
        label="Linux-old"
        initrd=/initrd.img.old
        read-only


# If you have another OS on this machine (say DOS),
# you can boot if by uncommenting the following lines
# (Of course, change /dev/hda2 to wherever your DOS partition is.)
# other=/dev/hda2
#   label=dos
```

A.2 /etc/dhcp3/dhcpd.conf

```
default-lease-time      1314000;
max-lease-time          1314000;

subnet 193.170.72.0 netmask 255.255.252.0
        option routers                  193.170.72.1;
        option subnet-mask              255.255.252.0;
        option broadcast-address        193.170.75.255;
        option domain-name              "ben.tuwien.ac.at";
        option domain-name-servers      128.130.2.3,128.130.3.131;
        use-host-decl-names             on;

        host pkcl2
                hardware ethernet       00:E0:18:A8:67:4B;
                fixed-address           193.170.74.39;
                filename                "/remote/boot/etherboot";
                option vendor-encapsulated-options
                                        3c:09:45:74:68:65:72:62:6f:6f:74:ff;
                option option-128       e4:45:74:68:0:1;
                option option-184       "^[[J";
                option option-160       "timeout=0";
                next-server             193.170.74.44;
                option log-servers      193.170.74.44;
                option root-path        "193.170.74.44:/remote";


        host pkcl3
                hardware ethernet       00:E0:18:A8:65:68;
                fixed-address           193.170.74.42;
                filename                "/remote/boot/etherboot";
                option vendor-encapsulated-options
                                        3c:09:45:74:68:65:72:62:6f:6f:74:ff;
                option option-128       e4:45:74:68:0:1;
                option option-184       "^[[J";
                option option-160       "timeout=0";
                next-server             193.170.74.44;
                option log-servers      193.170.74.44;
                option root-path        "193.170.74.44:/remote";


        host pkcl4
                hardware ethernet       00:0E:A6:B4:5D:A7;
                fixed-address           193.170.74.43;
                next-server             193.170.74.44;
                option log-servers      193.170.74.44;
                option root-path        "193.170.74.44:/remote";
                filename                "/remote/boot/pxelinux.0";
```

A.3 /remote/etc/mkinitrd/scripts/nfsroot

```
#!/bin/sh
#
# Based on http://lists.debian.org/debian-devel/2002/10/msg02376.html
# Changes by Philipp Kolmann - 2004-12-27
#
#
# mkinitrd initrd-module-and-script-generator
# - copies required modules and creates a set of initrd scripts so as
#   to mount an nfs root when booting a diskless box
# i.e. this script is run when *generating* the initrd image,
# and is meant to figure out what modules need to be included
# in the initrd image; it further generates script(s) to be run by the
# initrd's init to mount the nfs root at boot time
#
# env vars set by mkinitrd:
# INITRDDIR - initrd assembly area (e.g. /tmp/mkinitrd.4553/initrd)
# MODULEDIR - module source directory (e.g. /lib/modules/2.4.18)
#
# works with initrd-tools (0.1.33), udhcpc (0.9.7-4)
#
# BDL Jul02
# Changelog:
# - removed pivot_root hack, now done by mkinitrd's init (initrd-tools 0.1.26)

# initrd required programs:
# mkinitrd conveniently handles including the required binaries
# (and supporting libraries) listed in /etc/mkinitrd/exe
# - i.e. make sure mount, pivot_root etc are listed there
# - inbuilt exe list is: /bin/sleep /sbin/modprobe /sbin/insmod /bin/ash
#    (ref. /usr/sbin/mkinitrd)
#
# required modules:
# mkinitrd can probe to determine which modules need to be
# included in the initrd image for local devices (eg ide-disk),
# but can't grok nfs
# - need to manually copy them into the initrd image (e.g below)
# - or, just say "MODULES=all" in mkinitrd.conf (at 100Mb/s, what's an
#   extra MB or two? :-)
#
# The DHCP config should specify the root-path parameter
# to the client for the NFS server and export, e.g. for dhcp3:
#
#    option root-path "193.170.74.38:/remote/system";

###################################
# required modules
# - dependencies are automatically determined (below)

## mkinitrd.conf MODULES=all
#MODULES=

# only include those which are going to be loaded
MODULES=`grep -v '^#' /etc/mkinitrd/modules`


###################################

if ! [ "$INITRDDIR" -a "$MODULEDIR" ]; then
echo "usage: INITRDDIR=<initrd dir> MODULEDIR=<module dir> $0"
exit 1
fi
```

```
###############################################################################
# 1. create various mount points (in initrd filesystem)

# create var ramfs mount point (mainly for dhcp params)
# - initrd fs will generally be cramfs, which is ro (should be called cromfs!)
[ -d "$INITRDDIR/var" ] || mkdir $INITRDDIR/var

# hey, can't hurt... (right? :-)
[ -d "$INITRDDIR/proc" ] || mkdir $INITRDDIR/proc

[ -d "$INITRDDIR/.dev" ]  || mkdir $INITRDDIR/.dev
[ -d "$INITRDDIR/bin" ]  || mkdir $INITRDDIR/bin
[ -d "$INITRDDIR/home" ] || mkdir $INITRDDIR/home
[ -d "$INITRDDIR/lib" ]  || mkdir $INITRDDIR/lib
[ -d "$INITRDDIR/root" ] || mkdir $INITRDDIR/root
[ -d "$INITRDDIR/sbin" ] || mkdir $INITRDDIR/sbin
[ -d "$INITRDDIR/usr" ]  || mkdir $INITRDDIR/usr
[ -d "$INITRDDIR/setup" ]  || mkdir $INITRDDIR/setup
[ -d "$INITRDDIR/opt" ]  || mkdir $INITRDDIR/opt
[ -d "$INITRDDIR/grid" ]  || mkdir $INITRDDIR/grid
[ -d "$INITRDDIR/grid/condor" ]  || mkdir $INITRDDIR/grid/condor
[ -d "$INITRDDIR/grid/SGE" ]  || mkdir $INITRDDIR/grid/SGE

###############################################################################
# 2. copy required modules
# a simple modprobe -v -n <module> does pretty much what we want
# (lists modules which would be loaded, i.e. the dependancies, of
# <module>), though we need to tell it to look in the new moduledir,
# not the running kernel's

modulesconf=`mktemp -p $INITRDDIR/..` || exit 1

# fabricate a bare modules.conf for the target module tree
# (basically a stripped down /etc/modules.conf; ref. modules.conf(5))
# - must quote special chars (e.g. ` => ')
cat > "$modulesconf" <<-EOF
depfile=$MODULEDIR/modules.dep
generic_stringfile=$MODULEDIR/modules.generic_string
pcimapfile=$MODULEDIR/modules.pcimap
isapnpmapfile=$MODULEDIR/modules.isapnpmap
usbmapfile=$MODULEDIR/modules.usbmap
parportmapfile=$MODULEDIR/modules.parportmap
ieee1394mapfile=$MODULEDIR/modules.ieee1394map
pnpbiosmapfile=$MODULEDIR/modules.pnpbiosmap

path[toplevel]=$MODULEDIR
EOF

# determine full paths of desired modules and cp into INITRDDIR
# - cpio invocation is as per mkinitrd
(
for m in $MODULES; do
# look for "insmod <module>", one for each dependancy + the target
#modprobe -C "$modulesconf" -v -n $m | awk '$1 ~ /insmod/  print $2 '
modprobe -v -n $m | awk '$1 ~ /insmod/  print $2 '

# alas, in '-n' mode, modprobe doesn't return correct exit status
if [ $? -ne 0 ]; then
echo "modprobe failed, aborting" >&2
exit 1
fi
done
) | cpio -pLd --quiet $INITRDDIR/ 2>/dev/null

rm "$modulesconf"
```

```
# copy needed files to the initrd
cp ‘which udhcpc‘ $INITRDDIR/bin2/
cp ‘which halt‘ $INITRDDIR/bin2/
cp ‘which bash‘ $INITRDDIR/bin2/pkbash
cp /lib/libncurses.so.5 $INITRDDIR/lib/
cp /lib/tls/libdl.so.2 $INITRDDIR/lib/

# create /dev/initctl to be able to halt when something happens
mknod -m 600 $INITRDDIR/dev/initctl p



##############################################################################
# 3. create initrd config scripts:
#    i) configure networking and fetch nfs params via dhcp client
#   ii) mount nfs fs
#
# The mkinitrd init runs the files in /scripts/* (sourcing .sh files in a
# subshell).
# These scripts may in turn call others, e.g. the DHCP client script.
# - note that due to the initrd fs being read-only, it’s difficult to pass
#   parameters back to the caller (i.e. by writing a state file to be
#   sourced by others)
#   - thus, a writeable tmp filesystem is required
#
#  i) the DHCP client (scripts/40networking) calls /etc/udhcpc-script when
#     a lease is acquired (or not acquired - ref udhcpc(8))
#     - this client script gets passed the DHCP parameters as env variables
#       - the nfs-network is configured
# ii) 45mountnfs
#     - uses the DHCP-supplied ’filename’ as the nfs export to use as /
#


########################################
# dhcp guff

UDHCPCSCRIPT=/etc/udhcpc-script
UDHCPENV=/var/dhcp.env
cat > $INITRDDIR/$UDHCPCSCRIPT <<EOF
#!/bin/sh
# udhcpc client script
# - udhcp will have set various env variables according to the new state of
#   the interface
# - NOTE: udhcpc doesn’t actually configure the interface
# - NOTE: some options are very similar (cf. bootfile, boot_file !)
#   - udhcpc only requests a select few options: option-1,3,6,12,51,53,54
#     = subnet mask, router, dns, hostname, dhcp msg type, serverid, lease t.
#       see http://www.freesoft.org/CIE/RFC/2131/8.htm for bootp/dhcp format
#   - ISC DHCP options => udhcp script env variables:
#     - filename => boot_file
#     - dhcp-server-identifier => serverid (i.e. DHCP server address)
#     - next-server => siaddr (NOTE: v0.9.6 has a bug siaddr==yiaddr)
#

case "$1" in

deconfig) echo "$0: $interface: deconfig"

# i/f up, but deconfigured
ifconfig "$interface" 0.0.0.0
;;

bound) echo "$0: $interface: bound to $ip"
```

```
ifconfig "$interface" "$ip" netmask "$subnet"  $broadcast:+broadcast $broadcast

route add default gw "$router"

# set hostname, may be echoed back to the server in the
# client's dhclient script (so as to get added to lease file)
[ "$hostname" ] && hostname "$hostname"

# udhcp 0.9.6 bug: siaddr incorrectly == yiaddr
siaddr="$serverid"

# ok, write env to file for others to source...
set | grep -v PPID > $UDHCPENV
;;

renew) echo "$0: $interface: renew $ip";
;;

nak) echo "$0: $interface: nak: $message"
;;

*) echo "$0: huh? unknown udhcpc action: [$1]"
;;
esac

EOF

chmod 755 $INITRDDIR/$UDHCPCSCRIPT



########################################
# scripts/


cat > $INITRDDIR/scripts/05mounts <<'EOF'
#!/bin/sh
trap '/bin/echo "FAILED [$0]"; /bin2/halt -n -d -f -h -p' 0
[ -d /proc/1 ] || mount -n -t proc proc /proc
mount -t ramfs none /var
trap - 0
EOF
chmod 755 $INITRDDIR/scripts/05mounts


# note: heredoc not quoted-literal, watch interpolation
cat > $INITRDDIR/scripts/40networking <<EOF
#!/bin/sh

trap '/bin/echo "FAILED [$0]"; /bin2/halt -n -d -f -h -p' 0
ifconfig lo 127.0.0.1 netmask 255.0.0.0
trap - 0

# udhcpc: lightweight dhcp client
# -f = foreground (don't fork)
# -n = "now" (exit if lease cannot be obtained)
# -q = quit after (not) obtaining lease
# -s = script to run on dhcp events (e.g. getting a lease)
# UDHCPCSCRIPT manages the interface config and writes the DHCP parameters
# (options) to UDHCPENV, as shell environment variables
sleep 11

echo -n "Getting IP via udhcp"
udhcpc --foreground --now --quit --script=$UDHCPCSCRIPT
```

```
if [ $? -ne 0 ]; then
    echo " First try failed"
    echo -n "Still trying"
    sleep 5
    udhcpc --foreground --now --quit --script=$UDHCPCSCRIPT
    if [ $? -ne 0 ]; then
        echo " Second try failed"
        echo -n "Still trying"
        sleep 5
        udhcpc --foreground --now --quit --script=$UDHCPCSCRIPT
        if [ $? -ne 0 ]; then
    /bin2/halt -n -d -f -h -p
fi
    fi
fi
echo "."

EOF
chmod 755 $INITRDDIR/scripts/40networking


# note: heredoc not quoted-literal, watch interpolation
cat > $INITRDDIR/scripts/45mountnfs <<EOF
#!/bin/sh
trap '/bin/echo "FAILED [$0]"; /bin2/halt -n -d -f -h -p' 0
. $UDHCPENV

# could/should mount w/ 'lock'?
echo -n "Mounting /setup!"
mount -o nolock,ro,rsize=8192,wsize=8192 "$rootpath/setup" /setup
# Signal kernel that root is mounted properly
echo 256 > /proc/sys/kernel/real-root-dev
echo "."
trap - 0

echo "Begin setup path"
/setup/setup_path
echo "END setup_path"

echo "Check for /etc/inittab"
if [ ! -f /etc/inittab ]
then
    sleep 10
    /bin2/halt -n -d -f -h -p
fi

echo "Check for /etc/init.d"
if [ ! -d /etc/init.d ]
then
    sleep 10
    /bin2/halt -n -d -f -h -p
fi

echo "Check for /etc/rcS.d"
if [ ! -d /etc/rcS.d ]
then
    sleep 10
    /bin2/halt -n -d -f -h -p
fi

echo "Check for /etc/rc2.d"
if [ ! -d /etc/rc2.d ]
then
    sleep 10
    /bin2/halt -n -d -f -h -p
```

```
fi

EOF
chmod 755 $INITRDDIR/scripts/45mountnfs
```

A.4 /remote/setup/setup_path

```
#!/bin/sh

# Client Setup Script
# (c) Philipp Kolmann, 2005

# Used to setup the client nodes for the WINZIG Grid at TU Wien

trap '/bin/echo "FAILED [$0]"; /bin2/halt -n -d -f -h -p' 0

# Source the DHCP environment
. /var/dhcp.env

echo -n "Mounting sbin"
mount -o nolock,ro,rsize=8192,wsize=8192 $rootpath/sbin /sbin
echo "."

echo -n "Setting 100Mbit Full duplex"
# Set 100Mbit Full duplex
mii-tool -F 100baseTx-FD
echo "."

echo -n "Mounting /bin"
mount -o nolock,ro,rsize=8192,wsize=8192 $rootpath/bin /bin
echo "."
echo -n "Mounting /lib"
mount -o nolock,ro,rsize=8192,wsize=8192 $rootpath/lib /lib
echo "."
echo -n "Mounting /root"
mount -o nolock,rw,rsize=8192,wsize=8192 $rootpath/root /root
echo "."
echo -n "Mounting /usr"
mount -o nolock,ro,rsize=8192,wsize=8192 $rootpath/usr /usr
echo "."
echo -n "Mounting /setup/etc"
mount -o nolock,ro,rsize=8192,wsize=8192 $rootpath/etc /setup/etc
echo "."

# Call awk down here, since busybox doesn't have awk on board.
ROOTIP=`echo $rootpath | /usr/bin/mawk -F: 'print($1)''

echo -n "Mounting /opt"
mount -o nolock,ro,rsize=8192,wsize=8192 $ROOTIP:/opt /opt
echo "."
echo -n "Mounting /grid"
mount -o nolock,rw,rsize=8192,wsize=8192 $ROOTIP:/grid /grid
echo "."
echo -n "Mounting /home"
mount -o nolock,rw,rsize=8192,wsize=8192 $ROOTIP:/grid/home /home
echo "."

echo -n "Set up /tmp"
# Set up /tmp
mount -t ramfs none /tmp
chmod 1777 /tmp
echo "."

# Everything is mounted now
# Disable trap
trap - 0

echo -n "Set up /etc"
```

```
# copy stuff from setup/etc to etc
mount -t ramfs none /etc
echo "."

echo -n "Setting 100Mbit Full duplex"
# Set 100Mbit Full duplex
mii-tool -F 100baseTx-FD
echo "."


echo -n "Populating /etc"
ls -l /setup/etc/ | grep -v "^total" | mawk 'print($9)' |
while read line
do
erg=`grep "$line" /setup/etc.ignore`
if [ "X$erg" = "X" ]
then
    erg=`grep "$line" /setup/etc.copy`
    if [ "X$erg" = "X" ]
    then
ln -s "/setup/etc/$line" /etc
    else
cp -r -p "/setup/etc/$line" /etc
    fi
fi
done

# Set up /etc
mv /etc/shadow.copy /etc/shadow
mv /etc/gshadow.copy /etc/gshadow
chown root:shadow /etc/shadow
chown root:shadow /etc/gshadow
chmod 640 /etc/shadow
chmod 640 /etc/gshadow
chmod 600 /etc/ssh/*key
echo "$hostname" > /etc/hostname

echo "."

echo -n "Create /var structure"
[ -d /var/backups ]  || mkdir /var/backups
[ -d /var/lib ]  || mkdir /var/lib
[ -d /var/lib/nfs ]  || mkdir /var/lib/nfs
[ -d /var/lib/urandom ]  || mkdir /var/lib/urandom
[ -d /var/run ]  || mkdir /var/run
[ -d /var/spool ]  || mkdir /var/spool
[ -d /var/spool/cron ]  || mkdir /var/spool/cron
[ -d /var/spool/cron/crontabs ]  || mkdir /var/spool/cron/crontabs
[ -d /var/state ]  || mkdir /var/state
[ -d /var/tmp ]  || mkdir /var/tmp

[ -d /var/log ]  || mkdir /var/log
[ -f /var/log/lastlog ]  || touch /var/log/lastlog
[ -f /var/log/wtmp ]  || touch /var/log/wtmp
chown root:utmp /var/log/wtmp
[ -f /var/log/btmp ]  || touch /var/log/btmp
chown root:utmp /var/log/btmp

# Make cron.daily happy:
[ -d /var/lib/dpkg ]  || mkdir /var/lib/dpkg
[ -f /var/lib/dpkg/status ] || touch /var/lib/dpkg/status

# Exim 4 Setup
[ -d /var/lib/exim4 ]  || mkdir /var/lib/exim4
[ -d /var/log/exim4 ]  || mkdir /var/log/exim4
```

```
[ -d /var/spool/exim4 ]  || mkdir /var/spool/exim4
[ -d /var/spool/exim4/db ]  || mkdir /var/spool/exim4/db
[ -d /var/spool/exim4/input ]  || mkdir /var/spool/exim4/input
[ -d /var/spool/exim4/msglog ]  || mkdir /var/spool/exim4/msglog
chown Debian-exim:adm /var/log/exim4
chown -R Debian-exim:Debian-exim /var/spool/exim4
[ -d /var/mail ]  || mkdir /var/mail
chmod 3775 /var/mail
chown root:mail /var/mail
echo "$hostname.$domain" > /etc/mailname
/usr/sbin/update-exim4.conf

echo "."


# Condor Setup
# Check if /grid/condor/hosts/$hostname exists
su - condor -c "/grid/condor/hosts/setup_host"


# Add a direct route the the gridmaster to communicate directly without a
# router. (Only needed if more then one subnet
erg=‘echo $ip | grep 75‘
if [ "X$erg" != "X" ]
then
    sed s/74/75/ /etc/syslog.conf > /etc/syslog.conf_new
    mv /etc/syslog.conf_new /etc/syslog.conf

    # add route to be able to connect directly
    route add gridmaster gw 193.170.75.44
fi
```

Appendix B

Condor Config Files

This appendix contains the client script for Condor setup and the main Condor config file.

B.1 /grid/condor/hosts/setup_host

```
#!/bin/sh
#
# Setup the condor_config.local file for this host

. /var/dhcp.env

# Check if /grid/condor/hosts/$hostname exists
if [ ! -d "/grid/condor/hosts/$hostname" ]
then
    cp -r -p /grid/condor/hosts/SKELETON /grid/condor/hosts/$hostname
fi

# Clean up config file
grep -v WINZIG /grid/condor/hosts/$hostname/condor_config.local | \
    egrep -v '^$' | grep -v DedicatedScheduler > \
    /grid/condor/hosts/$hostname/condor_config.local.back

cp -p /grid/condor/hosts/$hostname/condor_config.local.back \
    /grid/condor/hosts/$hostname/condor_config.local

# Set up WINZIG Special Parameters new each boot:
echo >> /grid/condor/hosts/$hostname/condor_config.local
erg=‘grep sse2 /proc/cpuinfo‘
if [ "X$erg" = "X" ]
then
    echo "WINZIG_CPU_SSE2 = FALSE"  >> \
        /grid/condor/hosts/$hostname/condor_config.local
else
    echo "WINZIG_CPU_SSE2 = TRUE"  >> \
            /grid/condor/hosts/$hostname/condor_config.local
fi
echo "STARTD_EXPRS = $(STARTD_EXPRS), WINZIG_CPU_SSE2">> \
        /grid/condor/hosts/$hostname/condor_config.local

# Set up the Dedicated Maschines (Condor Manual 3.10.10
echo >> /grid/condor/hosts/$hostname/condor_config.local
echo -n "DedicatedScheduler = " >> \
        /grid/condor/hosts/$hostname/condor_config.local
echo "D̈edicatedScheduler@gridmaster.ben.tuwien.ac.at" >> \
        /grid/condor/hosts/$hostname/condor_config.local
echo "STARTD_EXPRS = $(STARTD_EXPRS), DedicatedScheduler">> \
        /grid/condor/hosts/$hostname/condor_config.local


# Set the CPU Speed
echo >> /grid/condor/hosts/$hostname/condor_config.local
erg=‘grep MHz /proc/cpuinfo |awk -F: 'print($2)' |awk -F. 'print($1)''
echo "WINZIG_CPU_SPEED = $erg"  >> \
    /grid/condor/hosts/$hostname/condor_config.local
echo "STARTD_EXPRS = $(STARTD_EXPRS),WINZIG_CPU_SPEED">> \
    /grid/condor/hosts/$hostname/condor_config.local

room=‘GET http://studman.ben.tuwien.ac.at/__winzig/room.php?pc=$hostname‘
if [ "$room" != "ERROR" ]
then
    echo >> /grid/condor/hosts/$hostname/condor_config.local
    echo "WINZIG_ROOM = $̈room"  >> \
        /grid/condor/hosts/$hostname/condor_config.local
    echo "STARTD_EXPRS = $(STARTD_EXPRS),WINZIG_ROOM">> \
        /grid/condor/hosts/$hostname/condor_config.local
fi
```

B.2 /grid/condor/etc/condor_config

```
##########################################################################
##
##  condor_config
##
##  This is the global configuration file for condor.
##
##  The file is divided into four main parts:
##  Part 1:  Settings you MUST customize
##  Part 2:  Settings you may want to customize
##  Part 3:  Settings that control the policy of when condor will
##           start and stop jobs on your machines
##  Part 4:  Settings you should probably leave alone (unless you
##  know what you're doing)
##
##  Please read the INSTALL file (or the Install chapter in the
##  Condor Administrator's Manual) for detailed explanations of the
##  various settings in here and possible ways to configure your
##  pool.
##
##  If you are installing Condor as root and then handing over the
##  administration of this file to a person you do not trust with
##  root access, please read the Installation chapter paying careful
##  note to the condor_config.root entries.
##
##  Unless otherwise specified, settings that are commented out show
##  the defaults that are used if you don't define a value.  Settings
##  that are defined here MUST BE DEFINED since they have no default
##  value.
##
##  Unless otherwise indicated, all settings which specify a time are
##  defined in seconds.
##
##########################################################################

##########################################################################
##########################################################################
##
##  ######                                      #
##  #     #    ##    #####    #####           ##
##  #     #   #  #   #    #     #            # #
##  ######   #    #  #    #     #              #
##  #        ######  #####      #              #
##  #        #    #  #   #      #              #
##  #        #    #  #    #     #            #####
##
##  Part 1:  Settings you must customize:
##########################################################################
##########################################################################

##  What machine is your central manager?
#CONDOR_HOST = central-manager-hostname.your.domain
CONDOR_HOST=gridmaster.ben.tuwien.ac.at

##--------------------------------------------------------------------
##  Pathnames:
##--------------------------------------------------------------------
##  Where have you installed the bin, sbin and lib condor directories?
RELEASE_DIR = /grid/condor

##  Where is the local condor directory for each host?
##  This is where the local config file(s), logs and
```

```
##  spool/execute directories are located
#LOCAL_DIR = $(TILDE)
LOCAL_DIR = $(RELEASE_DIR)/hosts/$(HOSTNAME)

##  Where is the machine-specific local config file for each host?
LOCAL_CONFIG_FILE = $(LOCAL_DIR)/condor_config.local

## If the local config file is not present, is it an error?
## WARNING: This is a potential security issue.
## If not specified, the default is True
#REQUIRE_LOCAL_CONFIG_FILE = TRUE

##--------------------------------------------------------------------
##  Mail parameters:
##--------------------------------------------------------------------
##  When something goes wrong with condor at your site, who should get
##  the email?
CONDOR_ADMIN = kolmann@zid.tuwien.ac.at

##  Full path to a mail delivery program that understands that "-s"
##  means you want to specify a subject:
MAIL = /usr/bin/mail

##--------------------------------------------------------------------
##  Network domain parameters:
##--------------------------------------------------------------------
##  Internet domain of machines sharing a common UID space.  If your
##  machines don't share a common UID space, set it to
##  UID_DOMAIN = $(FULL_HOSTNAME)
##  to specify that each machine has its own UID space.
UID_DOMAIN = ben.tuwien.ac.at

##  Internet domain of machines sharing a common file system.
##  If your machines don't use a network file system, set it to
##  FILESYSTEM_DOMAIN = $(FULL_HOSTNAME)
##  to specify that each machine has its own file system.
FILESYSTEM_DOMAIN = ben.tuwien.ac.at

##  This macro is used to specify a short description of your pool.
##  It should be about 20 characters long. For example, the name of
##  the UW-Madison Computer Science Condor Pool is ''UW-Madison CS''.
COLLECTOR_NAME  = WINZIG

##########################################################################
##########################################################################
##
##  ######                                    #####
##  #     #    ##    #####    #####         #     #
##  #     #   #  #   #    #     #                 #
##  ######   #    #  #    #     #            #####
##  #        ######  #####      #           #
##  #        #    #  #   #      #           #
##  #        #    #  #    #     #           #######
##
##  Part 2:  Settings you may want to customize:
##  (it is generally safe to leave these untouched)
##########################################################################
##########################################################################

##
##  The user/group ID <uid>.<gid> of the "Condor" user.
##  (this can also be specified in the environment)
#CONDOR_IDS=x.x
CONDOR_IDS=5000.100
```

```
##--------------------------------------------------------------------
##  Flocking: Submitting jobs to more than one pool
##--------------------------------------------------------------------
##  Flocking allows you to run your jobs in other pools, or lets
##  others run jobs in your pool.
##
##  To let others flock to you, define FLOCK_FROM.
##
##  To flock to others, define FLOCK_TO.

##  FLOCK_FROM defines the machines where you would like to grant
##  people access to your pool via flocking. (i.e. you are granting
##  access to these machines to join your pool).
FLOCK_FROM =
##  An example of this is:
#FLOCK_FROM = somehost.friendly.domain, anotherhost.friendly.domain

##  FLOCK_TO defines the central managers of the pools that you want
##  to flock to. (i.e. you are specifying the machines that you
##  want your jobs to be negotiated at -- thereby specifying the
##  pools they will run in.)
FLOCK_TO =
##  An example of this is:
#FLOCK_TO = central_manager.friendly.domain, condor.cs.wisc.edu

##  FLOCK_COLLECTOR_HOSTS should almost always be the same as
##  FLOCK_NEGOTIATOR_HOSTS (as shown below).  The only reason it would be
##  different is if the collector and negotiator in the pool that you are
##  flocking too are running on different machines (not recommended).
##  The collectors must be specified in the same corresponding order as
##  the FLOCK_NEGOTIATOR_HOSTS list.
FLOCK_NEGOTIATOR_HOSTS = $(FLOCK_TO)
FLOCK_COLLECTOR_HOSTS = $(FLOCK_TO)
## An example of having the negotiator and the collector on different
## machines is:
#FLOCK_NEGOTIATOR_HOSTS = condor.cs.wisc.edu, condor-negotiator.friendly.domain
#FLOCK_COLLECTOR_HOSTS =  condor.cs.wisc.edu, condor-collector.friendly.domain

##--------------------------------------------------------------------
##  Host/IP access levels
##--------------------------------------------------------------------
##  Please see the administrator's manual for details on these
##  settings, what they're for, and how to use them.

##  What machines have administrative rights for your pool?  This
##  defaults to your central manager.  You should set it to the
##  machine(s) where whoever is the condor administrator(s) works
##  (assuming you trust all the users who log into that/those
##  machine(s), since this is machine-wide access you're granting).
HOSTALLOW_ADMINISTRATOR = $(CONDOR_HOST)

##  If there are no machines that should have administrative access
##  to your pool (for example, there's no machine where only trusted
##  users have accounts), you can uncomment this setting.
##  Unfortunately, this will mean that administering your pool will
##  be more difficult.
#HOSTDENY_ADMINISTRATOR = *

##  What machines should have "owner" access to your machines, meaning
##  they can issue commands that a machine owner should be able to
##  issue to their own machine (like condor_vacate).  This defaults to
##  machines with administrator access, and the local machine.  This
##  is probably what you want.
HOSTALLOW_OWNER = $(FULL_HOSTNAME), $(HOSTALLOW_ADMINISTRATOR)
```

```
##  Read access.  Machines listed as allow (and/or not listed as deny)
##  can view the status of your pool, but cannot join your pool
##  or run jobs.
##  NOTE: By default, without these entries customized, you
##  are granting read access to the whole world.  You may want to
##  restrict that to hosts in your domain.  If possible, please also
##  grant read access to "*.cs.wisc.edu", so the Condor developers
##  will be able to view the status of your pool and more easily help
##  you install, configure or debug your Condor installation.
##  It is important to have this defined.
#HOSTALLOW_READ = *
#HOSTALLOW_READ = *.your.domain, *.cs.wisc.edu
#HOSTDENY_READ = *.bad.subnet, bad-machine.your.domain, 144.77.88.*
HOSTALLOW_READ= *.tuwien.ac.at

##  Write access.  Machines listed here can join your pool, submit
##  jobs, etc.  Note: Any machine which has WRITE access must
##  also be granted READ access.  Granting WRITE access below does
##  not also automatically grant READ access; you must change
##  HOSTALLOW_READ above as well.
##  If you leave it as it is, it will be unspecified, and effectively
##  it will be allowing anyone to write to your pool.
#HOSTALLOW_WRITE = *
#HOSTALLOW_WRITE = *.your.domain, your-friend's-machine.other.domain
#HOSTDENY_WRITE = bad-machine.your.domain
HOSTALLOW_WRITE = *.ben.tuwien.ac.at

##  Negotiator access.  Machines listed here are trusted central
##  managers.  You should normally not have to change this.
HOSTALLOW_NEGOTIATOR = $(NEGOTIATOR_HOST)
##  Now, with flocking we need to let the SCHEDD trust the other
##  negotiators we are flocking with as well.  You should normally
##  not have to change this either.
HOSTALLOW_NEGOTIATOR_SCHEDD = $(NEGOTIATOR_HOST), $(FLOCK_NEGOTIATOR_HOSTS)

##  Config access.  Machines listed here can use the condor_config_val
##  tool to modify all daemon configurations except those specified in
##  the condor_config.root file.  This level of host-wide access
##  should only be granted with extreme caution.  By default, config
##  access is denied from all hosts.
#HOSTALLOW_CONFIG = trusted-host.your.domain

##  Flocking Configs.  These are the real things that Condor looks at,
##  but we set them from the FLOCK_FROM/TO macros above.  It is safe
##  to leave these unchanged.
HOSTALLOW_WRITE_COLLECTOR = $(HOSTALLOW_WRITE), $(FLOCK_FROM)
HOSTALLOW_WRITE_STARTD    = $(HOSTALLOW_WRITE), $(FLOCK_FROM)
HOSTALLOW_READ_COLLECTOR  = $(HOSTALLOW_READ), $(FLOCK_FROM)
HOSTALLOW_READ_STARTD     = $(HOSTALLOW_READ), $(FLOCK_FROM)

##--------------------------------------------------------------------
##  Security parameters for setting configuration values remotely:
##--------------------------------------------------------------------
##  These parameters define the list of attributes that can be set
##  remotely with condor_config_val for the security access levels
##  defined above (for example, WRITE, ADMINISTRATOR, CONFIG, etc).
##  Please see the administrator's manual for futher details on these
##  settings, what they're for, and how to use them.  There are no
##  default values for any of these settings.  If they are not
##  defined, no attributes can be set with condor_config_val.

##  Attributes that can be set by hosts with "CONFIG" permission (as
##  defined with HOSTALLOW_CONFIG and HOSTDENY_CONFIG above).
##  The commented-out value here was the default behavior of Condor
##  prior to version 6.3.3.  If you don't need this behavior, you
```

```
##  should leave this commented out.
#SETTABLE_ATTRS_CONFIG = *

##  Attributes that can be set by hosts with "ADMINISTRATOR"
##  permission (as defined above)
#SETTABLE_ATTRS_ADMINISTRATOR = *_DEBUG, MAX_*_LOG

##  Attributes that can be set by hosts with "OWNER" permission (as
##  defined above) NOTE: any Condor job running on a given host will
##  have OWNER permission on that host by default.  If you grant this
##  kind of access, Condor jobs will be able to modify any attributes
##  you list below on the machine where they are running.  This has
##  obvious security implications, so only grant this kind of
##  permission for custom attributes that you define for your own use
##  at your pool (custom attributes about your machines that are
##  published with the STARTD_EXPRS setting, for example).
#SETTABLE_ATTRS_OWNER = your_custom_attribute, another_custom_attr

##  You can also define daemon-specific versions of each of these
##  settings.  For example, to define settings that can only be
##  changed in the condor_startd's configuration by hosts with OWNER
##  permission, you would use:
#STARTD_SETTABLE_ATTRS_OWNER = your_custom_attribute_name

##--------------------------------------------------------------------
##  Network filesystem parameters:
##--------------------------------------------------------------------
##  Do you want to use NFS for file access instead of remote system
##  calls?
#USE_NFS = False
USE_NFS = True

##  Do you want to use AFS for file access instead of remote system
##  calls?
#USE_AFS = False

##--------------------------------------------------------------------
##  Checkpoint server:
##--------------------------------------------------------------------
##  Do you want to use a checkpoint server if one is available?  If a
##  checkpoint server isn't available or USE_CKPT_SERVER is set to
##  False, checkpoints will be written to the local SPOOL directory on
##  the submission machine.
#USE_CKPT_SERVER = True

##  What's the hostname of this machine's nearest checkpoint server?
#CKPT_SERVER_HOST = checkpoint-server-hostname.your.domain

##  Do you want the starter on the execute machine to choose the
##  checkpoint server?  If False, the CKPT_SERVER_HOST set on
##  the submit machine is used.  Otherwise, the CKPT_SERVER_HOST set
##  on the execute machine is used.  The default is true.
#STARTER_CHOOSES_CKPT_SERVER = True

##--------------------------------------------------------------------
##  Miscellaneous:
##--------------------------------------------------------------------
##  Try to save this much swap space by not starting new shadows.
##  Specified in megabytes.
#RESERVED_SWAP = 5

##  What's the maximum number of jobs you want a single submit machine
##  to spawn shadows for?
#MAX_JOBS_RUNNING = 200
```

```
##  Condor needs to create a few lock files to synchronize access to
##  various log files.  Because of problems we've had with network
##  filesystems and file locking over the years, we HIGHLY recommend
##  that you put these lock files on a local partition on each
##  machine.  If you don't have your LOCAL_DIR on a local partition,
##  be sure to change this entry.  Whatever user (or group) condor is
##  running as needs to have write access to this directory.  If
##  you're not running as root, this is whatever user you started up
##  the condor_master as.  If you are running as root, and there's a
##  condor account, it's probably condor.  Otherwise, it's whatever
##  you've set in the CONDOR_IDS environment variable.  See the Admin
##  manual for details on this.
LOCK = $(LOG)

##  If you don't use a fully qualified name in your /etc/hosts file
##  (or NIS, etc.) for either your official hostname or as an alias,
##  Condor wouldn't normally be able to use fully qualified names in
##  places that it'd like to.  You can set this parameter to the
##  domain you'd like appended to your hostname, if changing your host
##  information isn't a good option.  This parameter must be set in
##  the global config file (not the LOCAL_CONFIG_FILE from above).
#DEFAULT_DOMAIN_NAME = your.domain.name

##  Condor can be told whether or not you want the Condor daemons to
##  create a core file if something really bad happens.  This just
##  sets the resource limit for the size of a core file.  By default,
##  we don't do anything, and leave in place whatever limit was in
##  effect when you started the Condor daemons.  If this parameter is
##  set and "True", we increase the limit to as large as it gets.  If
##  it's set to "False", we set the limit at 0 (which means that no
##  core files are even created).  Core files greatly help the Condor
##  developers debug any problems you might be having.
#CREATE_CORE_FILES = True

##  Condor Glidein downloads binaries from a remote server for the
##  machines into which you're gliding. This saves you from manually
##  downloading and installing binaries for every architecture you
##  might want to glidein to. The default server is one maintained at
##  The University of Wisconsin. If you don't want to use the UW
##  server, you can set up your own and change the following values to
##  point to it, instead.
GLIDEIN_SERVER_URLS = \
  http://www.cs.wisc.edu/condor/glidein/binaries \
  gsiftp://gridftp.cs.wisc.edu/p/condor/public/binaries/glidein

##  If your site needs to use UID_DOMAIN settings (defined above) that
##  are not real Internet domains that match the hostnames, you can
##  tell Condor to trust whatever UID_DOMAIN a submit machine gives to
##  the execute machine and just make sure the two strings match.  The
##  default for this setting is False, since it is more secure this
##  way.
#TRUST_UID_DOMAIN = False

## If you would like to be informed in near real-time via condor_q when
## a vanilla/standard/java job is in a suspension state, set this attribute to
## TRUE. However, this real-time update of the condor_schedd by the shadows
## could cause performance issues if there are thousands of concurrently
## running vanilla/standard/java jobs under a single condor_schedd and they are
## allowed to suspend and resume.
#REAL_TIME_JOB_SUSPEND_UPDATES = False


##--------------------------------------------------------------------
##  Settings that control the daemon's debugging output:
##--------------------------------------------------------------------
```

```
##
## The flags given in ALL_DEBUG are shared between all daemons.
##

ALL_DEBUG               =

MAX_COLLECTOR_LOG = 1000000
COLLECTOR_DEBUG =

MAX_KBDD_LOG = 1000000
KBDD_DEBUG =

MAX_NEGOTIATOR_LOG = 1000000
#NEGOTIATOR_DEBUG = D_MATCH
NEGOTIATOR_DEBUG = D_FULLDEBUG
MAX_NEGOTIATOR_MATCH_LOG = 1000000

MAX_SCHEDD_LOG = 1000000
#SCHEDD_DEBUG = D_COMMAND
SCHEDD_DEBUG = D_FULLDEBUG

MAX_SHADOW_LOG = 1000000
SHADOW_DEBUG =

MAX_STARTD_LOG = 1000000
STARTD_DEBUG = D_COMMAND

MAX_STARTER_LOG = 1000000
STARTER_DEBUG = D_NODATE

MAX_MASTER_LOG = 1000000
MASTER_DEBUG = D_COMMAND
##  When the master starts up, should it truncate it's log file?
#TRUNC_MASTER_LOG_ON_OPEN        = False


##########################################################################
##########################################################################
##
##  ######                                      #####
##  #     #    ##    #####    #####         #     #
##  #     #   #  #   #    #     #                 #
##  ######   #    #  #    #     #             #####
##  #        ######  #####      #                  #
##  #        #    #  #   #      #           #     #
##  #        #    #  #    #     #            #####
##
##  Part 3:  Settings control the policy for running, stopping, and
##  periodically checkpointing condor jobs:
##########################################################################
##########################################################################

##  This section contains macros are here to help write legible
##  expressions:
MINUTE = 60
HOUR = (60 * $(MINUTE))
StateTimer = (CurrentTime - EnteredCurrentState)
ActivityTimer = (CurrentTime - EnteredCurrentActivity)
ActivationTimer = (CurrentTime - JobStart)
LastCkpt = (CurrentTime - LastPeriodicCheckpoint)

##  The JobUniverse attribute is just an int.  These macros can be
##  used to specify the universe in a human-readable way:
STANDARD = 1
```

```
PVM = 4
VANILLA = 5
MPI = 8
IsPVM           = (TARGET.JobUniversity == $(PVM))
IsMPI           = (TARGET.JobUniversity == $(MPI))
IsVanilla       = (TARGET.JobUniversity == $(VANILLA))
IsStandard      = (TARGET.JobUniversity == $(STANDARD))

NonCondorLoadAvg = (LoadAvg - CondorLoadAvg)
BackgroundLoad = 0.3
HighLoad = 0.5
#StartIdleTime = 15 * $(MINUTE)
StartIdleTime =  0
ContinueIdleTime =  5 * $(MINUTE)
MaxSuspendTime = 10 * $(MINUTE)
MaxVacateTime = 10 * $(MINUTE)

KeyboardBusy = (KeyboardIdle < $(MINUTE))
ConsoleBusy = (ConsoleIdle  < $(MINUTE))
CPUIdle = ($(NonCondorLoadAvg) <= $(BackgroundLoad))
CPUBusy = ($(NonCondorLoadAvg) >= $(HighLoad))
KeyboardNotBusy = ($(KeyboardBusy) == False)

BigJob = (TARGET.ImageSize >= (50 * 1024))
MediumJob = (TARGET.ImageSize >= (15 * 1024) && TARGET.ImageSize < (50 * 1024))
SmallJob = (TARGET.ImageSize <  (15 * 1024))

JustCPU = ($(CPUBusy) && ($(KeyboardBusy) == False))
MachineBusy = ($(CPUBusy) || $(KeyboardBusy))

##  The RANK expression controls which jobs this machine prefers to
##  run over others.  Some examples from the manual include:
##    RANK = TARGET.ImageSize
##    RANK = (Owner == "coltrane") + (Owner == "tyner") \
##                 + ((Owner == "garrison") * 10) + (Owner == "jones")
##  By default, RANK is always 0, meaning that all jobs have an equal
##  ranking.
#RANK = 0


##########################################################################
##  This where you choose the configuration that you would like to
##  use.  It has no defaults so it must be defined.  We start this
##  file off with the UWCS_* policy.
##########################################################################

##  Also here is what is referred to as the TESTINGMODE_*, which is
##  a quick hardwired way to test Condor.
##  Replace UWCS_* with TESTINGMODE_* if you wish to do testing mode.
##  For example:
##  WANT_SUSPEND  = $(UWCS_WANT_SUSPEND)
##  becomes
##  WANT_SUSPEND  = $(TESTINGMODE_WANT_SUSPEND)

#WANT_SUSPEND  = $(UWCS_WANT_SUSPEND)
#WANT_VACATE = $(UWCS_WANT_VACATE)

###  When is this machine willing to start a job?
#START = $(UWCS_START)
#
###  When to suspend a job?
#SUSPEND = $(UWCS_SUSPEND)
#
###  When to resume a suspended job?
#CONTINUE = $(UWCS_CONTINUE)
```

```
#
###  When to nicely stop a job?
###  (as opposed to killing it instantaneously)
#PREEMPT = $(UWCS_PREEMPT)
#
###  When to instantaneously kill a preempting job
###  (e.g. if a job is in the pre-empting stage for too long)
#KILL = $(UWCS_KILL)

# MPI:
# Setup for Running Dedicated and Opportunistic Jobs (Condor Manual 3.10.10)
START = True
CONTINUE = True
SUSPEND = FALSE
PREEMPT = FALSE
KILL = FALSE

PERIODIC_CHECKPOINT = $(UWCS_PERIODIC_CHECKPOINT)
PREEMPTION_REQUIREMENTS = $(UWCS_PREEMPTION_REQUIREMENTS)
PREEMPTION_RANK = $(UWCS_PREEMPTION_RANK)
NEGOTIATOR_PRE_JOB_RANK = $(UWCS_NEGOTIATOR_PRE_JOB_RANK)
NEGOTIATOR_POST_JOB_RANK = $(UWCS_NEGOTIATOR_POST_JOB_RANK)
MaxJobRetirementTime    = $(UWCS_MaxJobRetirementTime)

##########################################################################
## This is the UWisc - CS Department Configuration.
##########################################################################
UWCS_WANT_SUSPEND = ( $(SmallJob) || $(KeyboardNotBusy) \
                           || $(IsPVM) || $(IsVanilla) )
UWCS_WANT_VACATE  = ( $(ActivationTimer) > 10 * $(MINUTE) \
                           || $(IsPVM) || $(IsVanilla) )

# Only start jobs if:
# 1) the keyboard has been idle long enough, AND
# 2) the load average is low enough OR the machine is currently
#    running a Condor job
# (NOTE: Condor will only run 1 job at a time on a given resource.
# The reasons Condor might consider running a different job while
# already running one are machine Rank (defined above), and user
# priorities.)
UWCS_START = ( (KeyboardIdle > $(StartIdleTime)) \
                  && ( $(CPUIdle) || \
                       (State != "Unclaimed" && State != "Owner")) )

# Suspend jobs if:
# 1) the keyboard has been touched, OR
# 2a) The cpu has been busy for more than 2 minutes, AND
# 2b) the job has been running for more than 90 seconds
UWCS_SUSPEND = ( $(KeyboardBusy) || \
                 ( (CpuBusyTime > 2 * $(MINUTE)) \
                   && $(ActivationTimer) > 90 ) )

# Continue jobs if:
# 1) the cpu is idle, AND
# 2) we've been suspended more than 10 seconds, AND
# 3) the keyboard hasn't been touched in a while
UWCS_CONTINUE = ( $(CPUIdle) && ($(ActivityTimer) > 10) \
                  && (KeyboardIdle > $(ContinueIdleTime)) )

# Preempt jobs if:
# 1) The job is suspended and has been suspended longer than we want
# 2) OR, we don't want to suspend this job, but the conditions to
#    suspend jobs have been met (someone is using the machine)
UWCS_PREEMPT = ( ((Activity == "Suspended") && \
                  ($(ActivityTimer) > $(MaxSuspendTime))) \
```

```
 || (SUSPEND && (WANT_SUSPEND == False)) )

# Maximum time (in seconds) to wait for a job to finish before kicking
# it off (due to PREEMPT, a higher priority claim, or the startd
# gracefully shutting down).  This is computed from the time the job
# was started, minus any suspension time.  Once the retirement time runs
# out, the usual preemption process will take place.  The job may
# self-limit the retirement time to _less_ than what is given here.
# By default, nice user jobs and standard universe jobs set their
# MaxJobRetirementTime to 0, so they will usually not wait in retirement.

UWCS_MaxJobRetirementTime = 0

# Kill jobs if they have taken too long to vacate gracefully
UWCS_KILL = $(ActivityTimer) > $(MaxVacateTime)

##  Only define vanilla versions of these if you want to make them
##  different from the above settings.
#SUSPEND_VANILLA  = ( $(KeyboardBusy) || \
#       ((CpuBusyTime > 2 * $(MINUTE)) && $(ActivationTimer) > 90) )
#CONTINUE_VANILLA = ( $(CPUIdle) && ($(ActivityTimer) > 10) \
#                     && (KeyboardIdle > $(ContinueIdleTime)) )
#PREEMPT_VANILLA  = ( ((Activity == "Suspended") && \
#                     ($(ActivityTimer) > $(MaxSuspendTime))) \
#                     || (SUSPEND_VANILLA && (WANT_SUSPEND == False)) )
#KILL_VANILLA    = $(ActivityTimer) > $(MaxVacateTime)

##  We use a simple Periodic checkpointing mechanism, but then
##  again we have a very fast network.
UWCS_PERIODIC_CHECKPOINT = $(LastCkpt) > (3 * $(HOUR))

##  You might want to checkpoint a little less often.  A good
##  example of this is below.  For jobs smaller than 60 megabytes, we
##  periodic checkpoint every 6 hours.  For larger jobs, we only
##  checkpoint every 12 hours.
#UWCS_PERIODIC_CHECKPOINT = ( (TARGET.ImageSize < 60000) && \
#     ($(LastCkpt) > (6 * $(HOUR))) ) || \
#   ( $(LastCkpt) > (12 * $(HOUR)) )

##  The rank expressions used by the negotiator are configured below.
##  This is the order in which ranks are applied by the negotiator:
##    1. NEGOTIATOR_PRE_JOB_RANK
##    2. rank in job ClassAd
##    3. NEGOTIATOR_POST_JOB_RANK
##    4. cause of preemption (0=user priority,1=startd rank,2=no preemption)
##    5. PREEMPTION_RANK

##  The NEGOTIATOR_PRE_JOB_RANK expression overrides all other ranks
##  that are used to pick a match from the set of possibilities.
##  The following expression matches jobs to unclaimed resources
##  whenever possible, regardless of the job-supplied rank.
UWCS_NEGOTIATOR_PRE_JOB_RANK = RemoteOwner =?= UNDEFINED

##  The NEGOTIATOR_POST_JOB_RANK expression chooses between
##  resources that are equally preferred by the job.
##  The following example expression steers jobs toward
##  faster machines and tends to fill a cluster of multi-processors
##  breadth-first instead of depth-first.  In this example,
##  the expression is chosen to have no effect when preemption
##  would take place, allowing control to pass on to
##  PREEMPTION_RANK.
#UWCS_NEGOTIATOR_POST_JOB_RANK = \
# (RemoteOwner =?= UNDEFINED) * (KFlops - VirtualMachineID)

##  The negotiator will not preempt a job running on a given machine
```

```
##  unless the PREEMPTION_REQUIREMENTS expression evaluates to true
##  and the owner of the idle job has a better priority than the owner
##  of the running job.  This expression defaults to true.
UWCS_PREEMPTION_REQUIREMENTS = $(StateTimer) > (1 * $(HOUR)) && \
RemoteUserPrio > SubmittorPrio * 1.2

##  The PREEMPTION_RANK expression is used in a case where preemption
##  is the only option and all other negotiation ranks are equal.  For
##  example, if the job has no preference, it is usually preferable to
##  preempt a job with a small ImageSize instead of a job with a large
##  ImageSize.  The default is to rank all preemptable matches the
##  same.  However, the negotiator will always prefer to match the job
##  with an idle machine over a preemptable machine, if all other
##  negotiation ranks are equal.
UWCS_PREEMPTION_RANK = (RemoteUserPrio * 1000000) - TARGET.ImageSize

##########################################################################
##  This is a Configuration that will cause your Condor jobs to
##  always run.  This is intended for testing only.
##########################################################################

##  This mode will cause your jobs to start on a machine an will let
##  them run to completion.  Condor will ignore all of what is going
##  on in the machine (load average, keyboard activity, etc.)

TESTINGMODE_WANT_SUSPEND = False
TESTINGMODE_WANT_VACATE = False
TESTINGMODE_START = True
TESTINGMODE_SUSPEND = False
TESTINGMODE_CONTINUE = True
TESTINGMODE_PREEMPT = False
TESTINGMODE_KILL = False
TESTINGMODE_PERIODIC_CHECKPOINT = False
TESTINGMODE_PREEMPTION_REQUIREMENTS = False
TESTINGMODE_PREEMPTION_RANK = 0


##########################################################################
##########################################################################
##
##  ######                                  #
##  #     #    ##    #####    #####         #    #
##  #     #   #  #   #    #     #           #    #
##  ######   #    #  #    #     #           #    #
##  #        ######  #####      #           #######
##  #        #    #  #   #      #                #
##  #        #    #  #    #     #                #
##
##  Part 4:  Settings you should probably leave alone:
##  (unless you know what you're doing)
##########################################################################
##########################################################################

##########################################################################
##  Daemon-wide settings:
##########################################################################

##  Pathnames
LOG = $(LOCAL_DIR)/log
SPOOL = $(LOCAL_DIR)/spool
EXECUTE = $(LOCAL_DIR)/execute
BIN = $(RELEASE_DIR)/bin
LIB = $(RELEASE_DIR)/lib
INCLUDE = $(RELEASE_DIR)/include
SBIN = $(RELEASE_DIR)/sbin
```

```
LIBEXEC = $(RELEASE_DIR)/libexec

## If you leave HISTORY undefined (comment it out), no history file
## will be created.
HISTORY = $(SPOOL)/history

##  Log files
COLLECTOR_LOG = $(LOG)/CollectorLog
KBDD_LOG = $(LOG)/KbdLog
MASTER_LOG = $(LOG)/MasterLog
NEGOTIATOR_LOG = $(LOG)/NegotiatorLog
NEGOTIATOR_MATCH_LOG = $(LOG)/MatchLog
SCHEDD_LOG = $(LOG)/SchedLog
SHADOW_LOG = $(LOG)/ShadowLog
STARTD_LOG = $(LOG)/StartLog
STARTER_LOG = $(LOG)/StarterLog

##  Lock files
SHADOW_LOCK = $(LOCK)/ShadowLock

##  In most cases, your condor_collector and condor_negotiator are
##  going to run on the same machine.  If for some reason, this isn't
##  the case, here's where you'd change it. Also, you can change the
##  port that the collector and negotiator run on. By default, the
##  collector uses port 9618 and the negotiator uses port 9614, but
##  you can set the port with a ":port", such as:
##  COLLECTOR_HOST = $(CONDOR_HOST):1234
COLLECTOR_HOST  = $(CONDOR_HOST)
NEGOTIATOR_HOST = $(CONDOR_HOST)

##  How long are you willing to let daemons try their graceful
##  shutdown methods before they do a hard shutdown? (30 minutes)
#SHUTDOWN_GRACEFUL_TIMEOUT = 1800

##  How much disk space would you like reserved from Condor?  In
##  places where Condor is computing the free disk space on various
##  partitions, it subtracts the amount it really finds by this
##  many megabytes.  (If undefined, defaults to 0).
RESERVED_DISK = 5

##  If your machine is running AFS and the AFS cache lives on the same
##  partition as the other Condor directories, and you want Condor to
##  reserve the space that your AFS cache is configured to use, set
##  this to true.
#RESERVE_AFS_CACHE = False

##  By default, if a user does not specify "notify_user" in the submit
##  description file, any email Condor sends about that job will go to
##  "username@UID_DOMAIN".  If your machines all share a common UID
##  domain (so that you would set UID_DOMAIN to be the same across all
##  machines in your pool), *BUT* email to user@UID_DOMAIN is *NOT*
##  the right place for Condor to send email for your site, you can
##  define the default domain to use for email.  A common example
##  would be to set EMAIL_DOMAIN to the fully qualified hostname of
##  each machine in your pool, so users submitting jobs from a
##  specific machine would get email sent to user@machine.your.domain,
##  instead of user@your.domain.  In general, you should leave this
##  setting commented out unless two things are true: 1) UID_DOMAIN is
##  set to your domain, not $(FULL_HOSTNAME), and 2) email to
##  user@UID_DOMAIN won't work.
#EMAIL_DOMAIN = $(FULL_HOSTNAME)

##  If your site needs to use TCP updates to the collector, instead of
##  UDP, you can enable this feature.  HOWEVER, WE DO NOT RECOMMEND
##  THIS FOR MOST SITES!  In general, the only sites that might want
```

```
##  this feature are pools made up of machines connected via a
##  wide-area network where UDP packets are frequently or always
##  dropped.  If you enable this feature, you *MUST* turn on the
##  COLLECTOR_SOCKET_CACHE_SIZE setting at your collector, and each
##  entry in the socket cache uses another file descriptor.  If not
##  defined, this feature is disabled by default.
#UPDATE_COLLECTOR_WITH_TCP = True

## HIGHPORT and LOWPORT let you set the range of ports that Condor
## will use. This may be useful if you are behind a firewall. By
## default, Condor uses port 9618 for the collector, 9614 for the
## negotiator, and system-assigned (apparently random) ports for
## everything else. HIGHPORT and LOWPORT only affect these
## system-assigned ports, but will restrict them to the range you
## specify here. If you want to change the well-known ports for the
## collector or negotiator, see COLLECTOR_HOST or NEGOTIATOR_HOST.
## Note that both LOWPORT and HIGHPORT must be at least 1024.
#HIGHPORT = 9700
#LOWPORT = 9600

##########################################################################
##  Daemon-specific settings:
##########################################################################

##--------------------------------------------------------------------
##  condor_master
##--------------------------------------------------------------------
##  Daemons you want the master to keep running for you:
DAEMON_LIST = MASTER, STARTD, SCHEDD

##  Which daemons use the Condor DaemonCore library (i.e., not the
##  checkpoint server or custom user daemons)?
#DC_DAEMON_LIST = MASTER, STARTD, SCHEDD, KBDD, COLLECTOR, NEGOTIATOR, EVENTD

##  Where are the binaries for these daemons?
MASTER = $(SBIN)/condor_master
STARTD = $(SBIN)/condor_startd
SCHEDD = $(SBIN)/condor_schedd
KBDD = $(SBIN)/condor_kbdd
NEGOTIATOR = $(SBIN)/condor_negotiator
COLLECTOR = $(SBIN)/condor_collector
GRID_MONITOR = $(SBIN)/grid_monitor.sh

##  When the master starts up, it can place it's address (IP and port)
##  into a file.  This way, tools running on the local machine don't
##  need to query the central manager to find the master.  This
##  feature can be turned off by commenting out this setting.
MASTER_ADDRESS_FILE = $(LOG)/.master_address

##  Where should the master find the condor_preen binary? If you don't
##  want preen to run at all, just comment out this setting.
PREEN = $(SBIN)/condor_preen

##  How do you want preen to behave?  The "-m" means you want email
##  about files preen finds that it thinks it should remove.  The "-r"
##  means you want preen to actually remove these files.  If you don't
##  want either of those things to happen, just remove the appropriate
##  one from this setting.
PREEN_ARGS = -m -r

##  How often should the master start up condor_preen? (once a day)
#PREEN_INTERVAL = 86400

##  If a daemon dies an unnatural death, do you want email about it?
#PUBLISH_OBITUARIES = True
```

```
##  If you're getting obituaries, how many lines of the end of that
##  daemon's log file do you want included in the obituary?
#OBITUARY_LOG_LENGTH = 20

##  Should the master run?
#START_MASTER = True

##  Should the master start up the daemons you want it to?
#START_DAEMONS = True

##  How often do you want the master to send an update to the central
##  manager?
#MASTER_UPDATE_INTERVAL = 300

##  How often do you want the master to check the timestamps of the
##  daemons it's running?  If any daemons have been modified, the
##  master restarts them.
#MASTER_CHECK_NEW_EXEC_INTERVAL = 300

##  Once you notice new binaries, how long should you wait before you
##  try to execute them?
#MASTER_NEW_BINARY_DELAY = 120

##  What's the maximum amount of time you're willing to give the
##  daemons to quickly shutdown before you just kill them outright?
#SHUTDOWN_FAST_TIMEOUT = 120

######
##  Exponential backoff settings:
######
##  When a daemon keeps crashing, we use "exponential backoff" so we
##  wait longer and longer before restarting it.  This is the base of
##  the exponent used to determine how long to wait before starting
##  the daemon again:
#MASTER_BACKOFF_FACTOR = 2.0

##  What's the maximum amount of time you want the master to wait
##  between attempts to start a given daemon?  (With 2.0 as the
##  MASTER_BACKOFF_FACTOR, you'd hit 1 hour in 12 restarts...)
#MASTER_BACKOFF_CEILING = 3600

##  How long should a daemon run without crashing before we consider
##  it "recovered".  Once a daemon has recovered, we reset the number
##  of restarts so the exponential backoff stuff goes back to normal.
#MASTER_RECOVER_FACTOR = 300


##--------------------------------------------------------------------
##  condor_startd
##--------------------------------------------------------------------
##  Where are the various condor_starter binaries installed?
STARTER_LIST = STARTER, STARTER_PVM, STARTER_STANDARD
STARTER = $(SBIN)/condor_starter
STARTER_PVM = $(SBIN)/condor_starter.pvm
STARTER_STANDARD = $(SBIN)/condor_starter.std

##  When the startd starts up, it can place it's address (IP and port)
##  into a file.  This way, tools running on the local machine don't
##  need to query the central manager to find the startd.  This
##  feature can be turned off by commenting out this setting.
STARTD_ADDRESS_FILE = $(LOG)/.startd_address

##  When a machine is claimed, how often should we poll the state of
##  the machine to see if we need to evict/suspend the job, etc?
```

```
#POLLING_INTERVAL        = 5

##  How often should the startd send updates to the central manager?
#UPDATE_INTERVAL         = 300
UPDATE_INTERVAL         = 120

##  How long is the startd willing to stay in the "matched" state?
#MATCH_TIMEOUT = 300

##  How long is the startd willing to stay in the preempting/killing
##  state before it just kills the starter directly?
#KILLING_TIMEOUT = 30

##  When a machine unclaimed, when should it run benchmarks?
##  LastBenchmark is initialized to 0, so this expression says as soon
##  as we're unclaimed, run the benchmarks.  Thereafter, if we're
##  unclaimed and it's been at least 4 hours since we ran the last
##  benchmarks, run them again.  The startd keeps a weighted average
##  of the benchmark results to provide more accurate values.
##  Note, if you don't want any benchmarks run at all, either comment
##  RunBenchmarks out, or set it to "False".
BenchmarkTimer = (CurrentTime - LastBenchmark)
RunBenchmarks : (LastBenchmark == 0 ) || ($(BenchmarkTimer) >= (4 * $(HOUR)))
#RunBenchmarks : False

##  Normally, when the startd is computing the idle time of all the
##  users of the machine (both local and remote), it checks the utmp
##  file to find all the currently active ttys, and only checks access
##  time of the devices associated with active logins.  Unfortunately,
##  on some systems, utmp is unreliable, and the startd might miss
##  keyboard activity by doing this.  So, if your utmp is unreliable,
##  set this setting to True and the startd will check the access time
##  on all tty and pty devices.
#STARTD_HAS_BAD_UTMP = False

##  This entry allows the startd to monitor console (keyboard and
##  mouse) activity by checking the access times on special files in
##  /dev.  Activity on these files shows up as "ConsoleIdle" time in
##  the startd's ClassAd.  Just give a comma-separated list of the
##  names of devices you want considered the console, without the
##  "/dev/" portion of the pathname.
CONSOLE_DEVICES = input/mice, console

##  The STARTD_EXPRS entry allows you to have the startd advertise
##  arbitrary expressions from the config file in its ClassAd.  Give
##  the comma-separated list of entries from the config file you want
##  in the startd ClassAd.
##  Note: because of the different syntax of the config file and
##  ClassAds, you might have to do a little extra work to get a given
##  entry into the ClassAd.  In particular, ClassAds require "'s
##  around your strings.  Numeric values can go in directly, as can
##  boolean expressions.  For example, if you wanted the startd to
##  advertise its list of console devices, when it's configured to run
##  benchmarks, and how often it sends updates to the central manager,
##  you'd have to define the following helper macro:
#MY_CONSOLE_DEVICES = "$(CONSOLE_DEVICES)"
##  Note: this must come before you define STARTD_EXPRS because macros
##  must be defined before you use them in other macros or
##  expressions.
##  Then, you'd set the STARTD_EXPRS setting to this:
#STARTD_EXPRS = MY_CONSOLE_DEVICES, RunBenchmarks, UPDATE_INTERVAL

COLLECTOR_HOST_STRING = "$(COLLECTOR_HOST)"
STARTD_EXPRS = COLLECTOR_HOST_STRING
```

```
##  When the startd is claimed by a remote user, it can also advertise
##  arbitrary attributes from the ClassAd of the job its working on.
##  Just list the attribute names you want advertised.
##  Note: since this is already a ClassAd, you don't have to do
##  anything funny with strings, etc.  This feature can be turned off
##  by commenting out this setting (there is no default).
STARTD_JOB_EXPRS = ImageSize, ExecutableSize, JobUniverse, NiceUser

##  If you want to "lie" to Condor about how many CPUs your machine
##  has, you can use this setting to override Condor's automatic
##  computation.  If you modify this, you must restart the startd for
##  the change to take effect (a simple condor_reconfig will not do).
##  Please read the section on "condor_startd Configuration File
##  Macros" in the Condor Administrators Manual for a further
##  discussion of this setting.  Its use is not recommended.  This
##  must be an integer ("N" isn't a valid setting, that's just used to
##  represent the default).
#NUM_CPUS = N

##  Normally, Condor will automatically detect the amount of physical
##  memory available on your machine.  Define MEMORY to tell Condor
##  how much physical memory (in MB) your machine has, overriding the
##  value Condor computes automatically.  For example:
#MEMORY = 128

##  How much memory would you like reserved from Condor?  By default,
##  Condor considers all the physical memory of your machine as
##  available to be used by Condor jobs.  If RESERVED_MEMORY is
##  defined, Condor subtracts it from the amount of memory it
##  advertises as available.
#RESERVED_MEMORY = 0

######
##  SMP startd settings
##
##  By default, Condor will evenly divide the resources in an SMP
##  machine (such as RAM, swap space and disk space) among all the
##  CPUs, and advertise each CPU as its own "virtual machine" with an
##  even share of the system resources.  If you want something other
##  than this, there are a few options available to you.  Please read
##  the section on "Configuring The Startd for SMP Machines" in the
##  Condor Administrator's Manual for full details.  The various
##  settings are only briefly listed and described here.
######

##  The maximum number of different virtual machine types.
#MAX_VIRTUAL_MACHINE_TYPES = 10

##  Use this setting to define your own virtual machine types.  This
##  allows you to divide system resources unevenly among your CPUs.
##  You must use a different setting for each different type you
##  define.  The "<N>" in the name of the macro listed below must be
##  an integer from 1 to MAX_VIRTUAL_MACHINE_TYPES (defined above),
##  and you use this number to refer to your type.  There are many
##  different formats these settings can take, so be sure to refer to
##  the section on "Configuring The Startd for SMP Machines" in the
##  Condor Administrator's Manual for full details.  In particular,
##  read the section titled "Defining Virtual Machine Types" to help
##  understand this setting.  If you modify any of these settings, you
##  must restart the condor_start for the change to take effect.
#VIRTUAL_MACHINE_TYPE_<N> = 1/4
#VIRTUAL_MACHINE_TYPE_<N> = cpus=1, ram=25%, swap=1/4, disk=1/4
#  For example:
#VIRTUAL_MACHINE_TYPE_1 = 1/8
#VIRTUAL_MACHINE_TYPE_2 = 1/4
```

```
##  If you define your own virtual machine types, you must specify how
##  many virtual machines of each type you wish to advertise.  You do
##  this with the setting below, replacing the "<N>" with the
##  corresponding integer you used to define the type above.  You can
##  change the number of a given type being advertised at run-time,
##  with a simple condor_reconfig.
#NUM_VIRTUAL_MACHINES_TYPE_<N> = M
#  For example:
#NUM_VIRTUAL_MACHINES_TYPE_1 = 6
#NUM_VIRTUAL_MACHINES_TYPE_2 = 1

##  The number of evenly-divided virtual machines you want Condor to
##  report to your pool (if less than the total number of CPUs).  This
##  setting is only considered if the "type" settings described above
##  are not in use.  By default, all CPUs are reported.  This setting
##  must be an integer ("N" isn't a valid setting, that's just used to
##  represent the default).
#NUM_VIRTUAL_MACHINES = N

##  How many of the virtual machines the startd is representing should
##  be "connected" to the console (in other words, notice when there's
##  console activity)?  This defaults to all virtual machines (N in a
##  machine with N CPUs).  This must be an integer ("N" isn't a valid
##  setting, that's just used to represent the default).
#VIRTUAL_MACHINES_CONNECTED_TO_CONSOLE = N

##  How many of the virtual machines the startd is representing should
##  be "connected" to the keyboard (for remote tty activity, as well
##  as console activity).  Defaults to 1.
#VIRTUAL_MACHINES_CONNECTED_TO_KEYBOARD = 1

##  If there are virtual machines that aren't connected to the
##  keyboard or the console (see the above two settings), the
##  corresponding idle time reported will be the time since the startd
##  was spawned, plus the value of this parameter.  It defaults to 20
##  minutes.  We do this because, if the virtual machine is configured
##  not to care about keyboard activity, we want it to be available to
##  Condor jobs as soon as the startd starts up, instead of having to
##  wait for 15 minutes or more (which is the default time a machine
##  must be idle before Condor will start a job).  If you don't want
##  this boost, just set the value to 0.  If you change your START
##  expression to require more than 15 minutes before a job starts,
##  but you still want jobs to start right away on some of your SMP
##  nodes, just increase this parameter.
#DISCONNECTED_KEYBOARD_IDLE_BOOST = 1200

######
##  Settings for computing optional resource availability statistics:
######
##  If STARTD_COMPUTE_AVAIL_STATS = True, the startd will compute
##  statistics about resource availability to be included in the
##  classad(s) sent to the collector describing the resource(s) the
##  startd manages.  The following attributes will always be included
##  in the resource classad(s) if STARTD_COMPUTE_AVAIL_STATS = True:
##    AvailTime = What proportion of the time (between 0.0 and 1.0)
##      has this resource been in a state other than "Owner"?
##    LastAvailInterval = What was the duration (in seconds) of the
##      last period between "Owner" states?
##  The following attributes will also be included if the resource is
##  not in the "Owner" state:
##    AvailSince = At what time did the resource last leave the
##      "Owner" state?  Measured in the number of seconds since the
##      epoch (00:00:00 UTC, Jan 1, 1970).
##    AvailTimeEstimate = Based on past history, this is an estimate
```

```
##      of how long the current period between "Owner" states will
##      last.
#STARTD_COMPUTE_AVAIL_STATS = False

##  If STARTD_COMPUTE_AVAIL_STATS = True, STARTD_AVAIL_CONFIDENCE sets
##  the confidence level of the AvailTimeEstimate.  By default, the
##  estimate is based on the 80th percentile of past values.
#STARTD_AVAIL_CONFIDENCE = 0.8

##  STARTD_MAX_AVAIL_PERIOD_SAMPLES limits the number of samples of
##  past available intervals stored by the startd to limit memory and
##  disk consumption.  Each sample requires 4 bytes of memory and
##  approximately 10 bytes of disk space.
#STARTD_MAX_AVAIL_PERIOD_SAMPLES = 100

##--------------------------------------------------------------------
##  condor_schedd
##--------------------------------------------------------------------
##  Where are the various shadow binaries installed?
SHADOW_LIST = SHADOW, SHADOW_PVM, SHADOW_STANDARD
SHADOW = $(SBIN)/condor_shadow
SHADOW_PVM = $(SBIN)/condor_shadow.pvm
SHADOW_STANDARD = $(SBIN)/condor_shadow.std

##  When the schedd starts up, it can place it's address (IP and port)
##  into a file.  This way, tools running on the local machine don't
##  need to query the central manager to find the schedd.  This
##  feature can be turned off by commenting out this setting.
SCHEDD_ADDRESS_FILE = $(LOG)/.schedd_address

##  How often should the schedd send an update to the central manager?
#SCHEDD_INTERVAL = 300

##  How long should the schedd wait between spawning each shadow?
#JOB_START_DELAY = 2

##  How often should the schedd send a keep alive message to any
##  startds it has claimed?  (5 minutes)
#ALIVE_INTERVAL = 300

##  This setting controls the maximum number of times that a
##  condor_shadow processes can have a fatal error (exception) before
##  the condor_schedd will simply relinquish the match associated with
##  the dying shadow.
#MAX_SHADOW_EXCEPTIONS = 5

##  Estimated virtual memory size of each condor_shadow process.
##  Specified in kilobytes.
SHADOW_SIZE_ESTIMATE = 1800

##  The condor_schedd can renice the condor_shadow processes on your
##  submit machines.  How how "nice" do you want the shadows? (1-19).
##  The higher the number, the lower priority the shadows have.
##  This feature can be disabled entirely by commenting it out.
SHADOW_RENICE_INCREMENT = 10

##  By default, when the schedd fails to start an idle job, it will
##  not try to start any other idle jobs in the same cluster during
##  that negotiation cycle.  This makes negotiation much more
##  efficient for large job clusters.  However, in some cases other
##  jobs in the cluster can be started even though an earlier job
##  can't.  For example, the jobs' requirements may differ, because of
##  different disk space, memory, or operating system requirements.
##  Or, machines may be willing to run only some jobs in the cluster,
##  because their requirements reference the jobs' virtual memory size
```

```
##  or other attribute.  Setting NEGOTIATE_ALL_JOBS_IN_CLUSTER to True
##  will force the schedd to try to start all idle jobs in each
##  negotiation cycle.  This will make negotiation cycles last longer,
##  but it will ensure that all jobs that can be started will be
##  started.
#NEGOTIATE_ALL_JOBS_IN_CLUSTER = False

## This setting controls how often, in seconds, the schedd considers
## periodic job actions given by the user in the submit file.
## (Currently, these are periodic_hold, periodic_release, and periodic_remove.)
PERIODIC_EXPR_INTERVAL = 60

######
## Queue management settings:
######
##  How often should the schedd truncate it's job queue transaction
##  log?  (Specified in seconds, once a day is the default.)
#QUEUE_CLEAN_INTERVAL = 86400

##  How often should the schedd commit "wall clock" run time for jobs
##  to the queue, so run time statistics remain accurate when the
##  schedd crashes?  (Specified in seconds, once per hour is the
##  default.  Set to 0 to disable.)
#WALL_CLOCK_CKPT_INTERVAL = 3600

##  What users do you want to grant super user access to this job
##  queue?  (These users will be able to remove other user's jobs).
##  By default, this only includes root.
QUEUE_SUPER_USERS = root, condor


##--------------------------------------------------------------------
##  condor_shadow
##--------------------------------------------------------------------
##  If the shadow is unable to read a checkpoint file from the
##  checkpoint server, it keeps trying only if the job has accumulated
##  more than MAX_DISCARDED_RUN_TIME seconds of CPU usage.  Otherwise,
##  the job is started from scratch.  Defaults to 1 hour.  This
##  setting is only used if USE_CKPT_SERVER (from above) is True.
#MAX_DISCARDED_RUN_TIME = 3600

##  Should periodic checkpoints be compressed?
#COMPRESS_PERIODIC_CKPT = False

##  Should vacate checkpoints be compressed?
#COMPRESS_VACATE_CKPT = False

##  Should we commit the application's dirty memory pages to swap
##  space during a periodic checkpoint?
#PERIODIC_MEMORY_SYNC = False

##  Should we write vacate checkpoints slowly?  If nonzero, this
##  parameter specifies the speed at which vacate checkpoints should
##  be written, in kilobytes per second.
#SLOW_CKPT_SPEED = 0


##--------------------------------------------------------------------
##  condor_shadow.pvm
##--------------------------------------------------------------------
##  Where is the condor pvm daemon installed?
PVMD = $(SBIN)/condor_pvmd

##  Where is the condor pvm group server daemon installed?
PVMGS = $(SBIN)/condor_pvmgs
```

```
##--------------------------------------------------------------------
##  condor_starter
##--------------------------------------------------------------------
##  The condor_starter can renice the processes from remote Condor
##  jobs on your execute machines.  If you want this, uncomment the
##  following entry and set it to how "nice" do you want the user
##  jobs. (1-19)  The larger the number, the lower priority the
##  process gets on your machines.
#JOB_RENICE_INCREMENT = 10

##  Should the starter do local logging to its own log file, or send
##  debug information back to the condor_shadow where it will end up
##  in the ShadowLog?
#STARTER_LOCAL_LOGGING = TRUE

##  If the UID_DOMAIN settings match on both the execute and submit
##  machines, but the UID of the user who submitted the job isn't in
##  the passwd file of the execute machine, the starter will normally
##  exit with an error.  Do you want the starter to just start up the
##  job with the specified UID, even if it's not in the passwd file?
#SOFT_UID_DOMAIN = FALSE


##--------------------------------------------------------------------
##  condor_submit
##--------------------------------------------------------------------
##  If you want condor_submit to automatically append an expression to
##  the Requirements expression or Rank expression of jobs at your
##  site, uncomment these entries.
#APPEND_REQUIREMENTS = (expression to append job requirements)
#APPEND_RANK = (expression to append job rank)

##  If you want expressions only appended for either standard or
##  vanilla universe jobs, you can uncomment these entries.  If any of
##  them are defined, they are used for the given universe, instead of
##  the generic entries above.
#APPEND_REQ_VANILLA = (expression to append to vanilla job requirements)
#APPEND_REQ_STANDARD = (expression to append to standard job requirements)
#APPEND_RANK_STANDARD = (expression to append to vanilla job rank)
#APPEND_RANK_VANILLA = (expression to append to standard job rank)

##  This can be used to define a default value for the rank expression
##  if one is not specified in the submit file.
#DEFAULT_RANK         = (default rank expression for all jobs)

##  If you want universe-specific defaults, you can use the following
##  entries:
#DEFAULT_RANK_VANILLA = (default rank expression for vanilla jobs)
#DEFAULT_RANK_STANDARD = (default rank expression for standard jobs)

##  If you want condor_submit to automatically append expressions to
##  the job ClassAds it creates, you can uncomment and define the
##  SUBMIT_EXPRS setting.  It works just like the STARTD_EXPRS
##  described above with respect to ClassAd vs. config file syntax,
##  strings, etc.  One common use would be to have the full hostname
##  of the machine where a job was submitted placed in the job
##  ClassAd.  You would do this by uncommenting the following lines:
#MACHINE = "$(FULL_HOSTNAME)"
#SUBMIT_EXPRS = MACHINE

## Condor keeps a buffer of recently-used data for each file an
## application opens.  This macro specifies the default maximum number
## of bytes to be buffered for each open file at the executing
```

```
## machine.
#DEFAULT_IO_BUFFER_SIZE = 524288

## Condor will attempt to consolidate small read and write operations
## into large blocks.  This macro specifies the default block size
## Condor will use.
#DEFAULT_IO_BUFFER_BLOCK_SIZE = 32768

##--------------------------------------------------------------------
##  condor_preen
##--------------------------------------------------------------------
##  Who should condor_preen send email to?
#PREEN_ADMIN = $(CONDOR_ADMIN)

##  What files should condor_preen leave in the spool directory?
VALID_SPOOL_FILES = job_queue.log, job_queue.log.tmp, history, \
                        Accountant.log, Accountantnew.log

##  What files should condor_preen remove from the log directory?
INVALID_LOG_FILES = core

##--------------------------------------------------------------------
##  Java parameters:
##--------------------------------------------------------------------
##  If you would like this machine to be able to run Java jobs,
##  then set JAVA to the path of your JVM binary.  If you are not
##  interested in Java, there is no harm in leaving this entry
##  empty or incorrect.

JAVA = /usr/bin/java

##  Some JVMs need to be told the maximum amount of heap memory
##  to offer to the process.  If your JVM supports this, give
##  the argument here, and Condor will fill in the memory amount.
##  If left blank, your JVM will choose some default value,
##  typically 64 MB.  The default (-Xmx) works with the Sun JVM.

JAVA_MAXHEAP_ARGUMENT = -Xmx

## JAVA_CLASSPATH_DEFAULT gives the default set of paths in which
## Java classes are to be found.  Each path is separated by spaces.
## If your JVM needs to be informed of additional directories, add
## them here.  However, do not remove the existing entries, as Condor
## needs them.

JAVA_CLASSPATH_DEFAULT = $(LIB) $(LIB)/scimark2lib.jar .

##  JAVA_CLASSPATH_ARGUMENT describes the command-line parameter
##  used to introduce a new classpath:

JAVA_CLASSPATH_ARGUMENT = -classpath

##  JAVA_CLASSPATH_SEPARATOR describes the character used to mark
##  one path element from another:

JAVA_CLASSPATH_SEPARATOR = :

##  JAVA_BENCHMARK_TIME describes the number of seconds for which
##  to run Java benchmarks.  A longer time yields a more accurate
##  benchmark, but consumes more otherwise useful CPU time.
##  If this time is zero or undefined, no Java benchmarks will be run.

JAVA_BENCHMARK_TIME = 2

##  If your JVM requires any special arguments not mentioned in
```

```
##  the options above, then give them here.

JAVA_EXTRA_ARGUMENTS =

##
##--------------------------------------------------------------------
##  Condor-G settings
##--------------------------------------------------------------------
##  Where is the GridManager binary installed?

GRIDMANAGER = $(SBIN)/condor_gridmanager
GAHP = $(SBIN)/gahp_server

##--------------------------------------------------------------------
##  Settings that control the daemon's debugging output:
##--------------------------------------------------------------------
##
## Note that the Gridmanager runs as the User, not a Condor daemon, so
## all users must have write permssion to the directory that the
## Gridmanager will use for it's logfile. Our suggestion is to create a
## directory called GridLogs in $(LOG) with UNIX permissions 1777
## (just like /tmp )
##  Another option is to use /tmp as the location of the GridManager log.
##

MAX_GRIDMANAGER_LOG = 1000000
GRIDMANAGER_DEBUG = D_COMMAND

#GRIDMANAGER_LOG = $(LOG)/GridLogs/GridmanagerLog.$(USERNAME)
GRIDMANAGER_LOG = /tmp/GridmanagerLog.$(USERNAME)

##--------------------------------------------------------------------
##  Various other settings that the Condor-G can use.
##--------------------------------------------------------------------

## If we're talking to a Globus 2.0 resource, Condor-G will use the new
## version of the GRAM protocol. The first option is how often to check the
## proxy on the submit site of things. If the GridManager discovers a new
## proxy, it will restart itself and use the new proxy for all future
## jobs launched. In seconds,  and defaults to 10 minutes
#GRIDMANAGER_CHECKPROXY_INTERVAL = 600

## The GridManager will shut things down 3 minutes before loosing Contact
## because of an expired proxy.
## In seconds, and defaults to 3 minutes
#GRDIMANAGER_MINIMUM_PROXY_TIME  = 180

## Condor requires that each submitted job be designated to run under a
## particular "universe". Condor-G is active when jobs are as marked as
## "GLOBUS" universe jobs. The universe of a job is set in the submit file
## with the 'universe = GLOBUS' line.
##
## If no universe is specificed in the submit file, Condor must pick one
## for the job to use. By default, it chooses the "standard" universe.
## The default can be overridden in the config file with the DEFAULT_UNIVERSE
## setting, which is a string to insert into a job submit description if the
## job does not try and define it's own universe
##
#DEFAULT_UNIVERSE = globus

#
# The Cred_min_time_left is the first-pass at making sure that Condor-G
# does not submit your job without it having enough time left for the
# job to finish. For example, if you have a job that runs for 20 minutes, and
# you might spend 40 minutes in the queue, it's a bad idea to submit with less
```

```
# than an hour left before your proxy expires.
# 2 hours seemed like a reasonable default.
#
CRED_MIN_TIME_LEFT = 120

##
## The location of the wrapper for invoking
## GT3 GAHP server
##
GT3_GAHP = $(SBIN)/gt3_gahp

##
## The location of GT3 files. This should normally be lib/gt3
##
GT3_LOCATION = $(LIB)/gt3



## PK: MPI

MPI_CONDOR_RSH_PATH = $(LIBEXEC)
WANT_SUSPEND        = False
WANT_VACATE         = False
STARTD_EXPRS = $(STARTD_EXPRS),WANT_SUSPEND,WANT_VACATE

## Condor Checkpoint

##  In what directory should the checkpoint server store checkpoint
##  files?
CKPT_SERVER_DIR = /grid/condor/ckpt_server

##########################################################################
##########################################################################
##  Settings you may want to customize:
##  (it is generally safe to leave these untouched)
##########################################################################
##########################################################################

##  The checkpoint server creates a child process for each active file
##  transfer.  What is the maximum number of processes it should
##  create?  It will deny any requests when at the maximum.  This is
##  set to 50 processes by default.
#CKPT_SERVER_MAX_PROCESSES = 50

##  You can also control the maximum number of processes for
##  checkpoint restores vs. checkpoint stores.  You may want to set a
##  lower maximum for checkpoint restores, so a large number of
##  restores can't starve all checkpoint stores.  The default maximum
##  for both is 50.
#CKPT_SERVER_MAX_STORE_PROCESSES = 50
#CKPT_SERVER_MAX_RESTORE_PROCESSES = 50


##########################################################################
##########################################################################
##  Settings you should probably leave alone:
##  (unless you know what you're doing)
##########################################################################
##########################################################################

CKPT_SERVER_LOG = $(LOG)/CkptServerLog
MAX_CKPT_SERVER_LOG = 1000000
CKPT_SERVER_DEBUG = D_ALWAYS
```

Appendix C

Experiment Scripts

This appendix contains the files used in the experiments and some result files.

C.1 amp-hydro Results

```
------------------------------------------------------------------------------
Minimos-NT 2.1                                    (c) IuE/TU-Vienna 1992-2004
------------------------------------------------------------------------------
Stepping: * [  20]  time =  0 s

                          Stepping Information:
==============================================================================
name       type   slot/pri  log  prev/post        range                  unit
------------------------------------------------------------------------------
time       delta    -1/  0  lin  0/2e-06          0 to 2e-06 [1e-07]      s
------------------------------------------------------------------------------

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD_DDPre-         3927    1  20T  3.0e-01,P 0  1.2e+04  1.2e+04  1.3e+00     1.0
HD_DDPre-         3927    2  12T  3.0e-01,P 0  3.0e+03  7.3e+16  1.3e+00     1.3
HD_DDPre-         3927    3  12T  3.0e-01,P 0  2.4e+03  2.4e+18  8.4e-01     1.7
HD_DDPre-         3927    4  17T  3.0e-01,P 0  1.6e+03  2.1e+21  5.1e-01     2.0
HD_DDPre-         3927    5  22T  3.2e-01,P 0  9.8e+02  2.2e+06  3.3e-01     2.5
HD_DDPre-         3927    6  37T  3.2e-01,P 0  7.1e+02  8.1e+21  2.1e-01     3.0
HD_DDPre-         3927    7  14T  3.3e-01,P 0  5.5e+02  6.0e+22  1.4e-01     3.3
HD_DDPre-         3927    8  11T  3.9e-01,P 0  4.2e+02  4.2e+22  9.6e-02     3.6
HD_DDPre-         3927    9  16T  5.1e-01,P 0  2.6e+02  6.3e+18  5.8e-02     4.0
HD_DDPre-         3927   10  19T  6.9e-01,P 0  1.5e+02  4.8e+22  2.9e-02     4.3
HD_DDPre-         3927   11  23T  4.3e-01,P 0  1.1e+02  1.6e+25  9.0e-03     4.7
HD_DDPre-         3927   12  16T  6.5e-01,P 0  5.8e+01  7.1e+06  5.1e-03     5.0
HD_DDPre-         3927   13  18T  8.9e-01,P 0  2.1e+01  2.8e+01  1.8e-03     5.4
HD_DDPre-         3927   14  18T  1.0e+00,P 0  2.5e+00  3.8e+06  1.9e-04     5.7
HD_DDPre-         3927   15  20T  1.0e+00,P 0  1.7e-02  8.7e+05  2.7e-09     6.0
HD_DDPre-         3927   16  22T  1.0e+00,P 0  2.8e-06  5.1e-01  1.7e-13     6.3
HD_DDPre-         3927   17  15T  1.0e+00,P 0  3.1e-12  7.0e-10  5.0e-16     6.6
HD_DD-            3927   18  16T  1.0e+00,P 0  4.1e-13  5.2e-13  3.9e-16     7.6
HD-               6391   19   7T  1.0e+00,P 0  3.9e-13  8.4e+02  4.0e-05     8.0
HD-               6391   20  24T  9.9e-01,P 0  2.5e+01  8.9e+01  2.6e-07     9.1
HD-               6391   21  17T  9.7e-01,P 0  3.4e+01  1.7e+04  2.6e-07     9.7
HD-               6391   22  18T  1.0e+00,P 0  1.4e+01  7.4e+01  3.7e-07    10.4
HD-               6391   23  18T  1.0e+00,P 0  1.5e+01  5.6e+09  2.4e-07    11.0
HD-               6391   24  23T  1.0e+00,P 0  2.4e+00  1.2e+01  6.6e-08    11.7
HD-               6391   25  22T  1.0e+00,P 0  6.1e+00  1.2e+01  3.9e-08    12.4
HD-               6391   26  22T  1.0e+00,P 0  9.9e-01  3.6e+00  2.5e-08    13.1
HD-               6391   27  19T  1.0e+00,P 0  2.5e-01  1.6e+00  1.1e-08    13.7
HD-               6391   28  21T  1.0e+00,P 0  2.0e-01  6.8e-01  4.6e-09    14.4
HD-               6391   29  26T  1.0e+00,P 0  2.3e-01  4.0e-01  2.0e-09    15.4
HD-               6391   30  30T  1.0e+00,P 0  3.4e-04  2.9e-03  1.7e-11    16.5
Rank:  6391(8056) Iterations: 30
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  19]  time =  1e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   5T  8.9e-01,P 0  6.1e+01  6.1e+01  3.0e-05     0.5
HD-               6391    2   5T  9.6e-01,P 0  3.6e+01  3.6e+01  3.4e-06     0.9
HD-               6391    3   5T  1.0e-00,P 0  1.8e+01  1.8e+01  1.5e-07     1.4
HD-               6391    4   6T  1.0e+00,P 0  7.6e+00  7.7e+00  1.8e-08     1.8
HD-               6391    5   6T  1.0e+00,P 0  2.6e+00  2.7e+00  3.5e-09     2.2
HD-               6391    6   7T  1.0e+00,P 0  5.8e-01  5.8e-01  1.2e-09     2.5
HD-               6391    7   9T  1.0e+00,P 0  2.0e+00  2.0e+00  6.2e-10     3.1
HD-               6391    8   6T  1.0e+00,P 0  3.6e-04  2.8e-02  2.3e-10     3.4
```

```
HD-               6391    9   9T  1.0e+00,P 0  4.4e-04  2.8e-02  2.8e-12     3.9
HD-               6391   10  12T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     4.4
Rank:  6391(8056) Iterations: 10
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  18]  time =  2e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
--------------------------------------------------------------------------------
HD-               6391    1   8T  9.4e-01,P 0  4.2e+01  4.3e+01  1.8e-05     0.3
HD-               6391    2   7T  9.9e-01,P 0  2.4e+01  2.5e+01  1.1e-06     0.7
HD-               6391    3   7T  1.0e+00,P 0  1.2e+01  1.2e+01  3.9e-08     1.0
HD-               6391    4   7T  1.0e+00,P 0  5.0e+00  5.0e+00  8.5e-09     1.4
HD-               6391    5   7T  1.0e+00,P 0  1.7e+00  1.7e+00  1.8e-09     1.7
HD-               6391    6   7T  1.0e+00,P 0  3.6e-01  3.7e-01  8.2e-10     2.1
HD-               6391    7  10T  1.0e+00,P 0  1.4e+00  1.4e+00  4.3e-10     2.5
HD-               6391    8   7T  1.0e+00,P 0  1.6e-04  2.8e-02  1.3e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  2.0e-04  2.8e-02  2.8e-12     3.3
HD-               6391   10  11T  1.0e+00,P 0  3.1e-07  8.8e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  17]  time =  3e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
--------------------------------------------------------------------------------
HD-               6391    1   7T  1.0e+00,P 0  3.2e+00  3.2e+00  2.8e-10     0.3
HD-               6391    2   7T  1.0e+00,P 0  1.8e+00  1.8e+00  6.7e-10     0.7
HD-               6391    3   8T  1.0e+00,P 0  8.5e-01  8.6e-01  2.6e-10     1.0
HD-               6391    4  10T  1.0e+00,P 0  4.0e-01  4.0e-01  1.3e-10     1.5
HD-               6391    5  10T  1.0e+00,P 0  1.5e-05  4.2e-05  7.1e-12     1.9
Rank:  6391(8056) Iterations: 5
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  16]  time =  4e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
--------------------------------------------------------------------------------
HD-               6391    1   8T  9.6e-01,P 0  3.8e+01  3.8e+01  1.8e-05     0.3
HD-               6391    2   8T  1.0e-00,P 0  2.2e+01  2.2e+01  8.2e-07     0.7
HD-               6391    3   8T  1.0e+00,P 0  1.0e+01  1.0e+01  2.8e-08     1.0
HD-               6391    4   8T  1.0e+00,P 0  4.2e+00  4.2e+00  7.1e-09     1.4
HD-               6391    5   8T  1.0e+00,P 0  1.4e+00  1.4e+00  1.6e-09     1.7
HD-               6391    6   7T  1.0e+00,P 0  2.6e-01  2.7e-01  7.2e-10     2.1
HD-               6391    7  12T  1.0e+00,P 0  1.3e+00  1.3e+00  3.7e-10     2.6
HD-               6391    8   7T  1.0e+00,P 0  1.1e-04  2.8e-02  1.0e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  1.3e-04  2.8e-02  2.8e-12     3.4
HD-               6391   10  13T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  15]  time =  5e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
--------------------------------------------------------------------------------
HD-               6391    1   8T  8.8e-01,P 0  6.4e+01  6.5e+01  3.0e-05     0.4
HD-               6391    2   8T  9.5e-01,P 0  3.8e+01  3.9e+01  3.7e-06     0.7
HD-               6391    3   8T  1.0e-00,P 0  1.9e+01  1.9e+01  1.9e-07     1.1
HD-               6391    4   8T  1.0e+00,P 0  7.9e+00  7.9e+00  2.3e-08     1.4
HD-               6391    5   8T  1.0e+00,P 0  2.7e+00  2.7e+00  4.6e-09     1.8
HD-               6391    6   9T  1.0e+00,P 0  5.4e-01  5.4e-01  1.3e-09     2.1
```

```
HD-               6391    7  10T  1.0e+00,P 0  2.1e+00  2.1e+00  6.4e-10     2.6
HD-               6391    8   7T  1.0e+00,P 0  3.7e-04  2.8e-02  2.5e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  3.9e-04  2.8e-02  2.8e-12     3.4
HD-               6391   10  11T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     3.9
Rank:  6391(8056) Iterations: 10
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  14]  time =  6e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   7T  8.7e-01,P 0  6.6e+01  6.6e+01  3.0e-05     0.4
HD-               6391    2   8T  9.5e-01,P 0  3.8e+01  3.9e+01  3.8e-06     0.7
HD-               6391    3   7T  1.0e-00,P 0  1.9e+01  1.9e+01  1.9e-07     1.1
HD-               6391    4   8T  1.0e+00,P 0  7.5e+00  7.5e+00  2.0e-08     1.4
HD-               6391    5   8T  1.0e+00,P 0  2.4e+00  2.4e+00  4.0e-09     1.8
HD-               6391    6   9T  1.0e+00,P 0  3.7e-01  3.7e-01  1.2e-09     2.1
HD-               6391    7  10T  1.0e+00,P 0  2.1e+00  2.1e+00  6.0e-10     2.6
HD-               6391    8   7T  1.0e+00,P 0  4.1e-04  2.8e-02  2.1e-10     2.9
HD-               6391    9  11T  1.0e+00,P 0  1.0e-04  2.8e-02  2.9e-12     3.4
HD-               6391   10  10T  1.0e+00,P 0  3.1e-07  9.0e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  13]  time =  7e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   8T  9.4e-01,P 0  4.2e+01  4.2e+01  1.8e-05     0.3
HD-               6391    2   7T  9.9e-01,P 0  2.3e+01  2.3e+01  1.1e-06     0.7
HD-               6391    3   7T  1.0e+00,P 0  1.0e+01  1.0e+01  2.5e-08     1.0
HD-               6391    4   7T  1.0e+00,P 0  3.6e+00  3.7e+00  6.0e-09     1.4
HD-               6391    5   8T  1.0e+00,P 0  9.7e-01  9.8e-01  1.4e-09     1.7
HD-               6391    6  15T  1.0e+00,P 0  1.3e+00  1.3e+00  5.7e-10     2.2
HD-               6391    7   7T  1.0e+00,P 0  2.4e-04  2.8e-02  7.2e-11     2.6
HD-               6391    8  10T  1.0e+00,P 0  6.5e-05  2.9e-02  3.1e-12     3.0
HD-               6391    9  12T  1.0e+00,P 0  3.0e-07  9.1e-06  5.3e-12     3.5
Rank:  6391(8056) Iterations: 9
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  12]  time =  8e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   7T  1.0e+00,P 0  2.9e+00  3.0e+00  2.6e-10     0.3
HD-               6391    2   8T  1.0e+00,P 0  1.5e+00  1.5e+00  6.3e-10     0.7
HD-               6391    3   8T  1.0e+00,P 0  6.3e-01  6.3e-01  2.8e-10     1.1
HD-               6391    4  12T  1.0e+00,P 0  2.0e-01  2.1e-01  1.4e-10     1.5
HD-               6391    5  10T  1.0e+00,P 0  8.3e-06  2.6e-05  5.4e-12     2.0
Rank:  6391(8056) Iterations: 5
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  11]  time =  9e-07  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   9T  9.6e-01,P 0  3.7e+01  3.7e+01  1.8e-05     0.4
HD-               6391    2   7T  1.0e-00,P 0  1.9e+01  1.9e+01  8.2e-07     0.7
HD-               6391    3   7T  1.0e+00,P 0  8.7e+00  8.7e+00  1.8e-08     1.1
HD-               6391    4   8T  1.0e+00,P 0  3.3e+00  3.3e+00  3.6e-09     1.4
HD-               6391    5   8T  1.0e+00,P 0  9.2e-01  9.3e-01  9.8e-10     1.8
```

```
HD-               6391    6  10T  1.0e+00,P 0  1.0e+00  1.1e+00  4.8e-10     2.2
HD-               6391    7   7T  1.0e+00,P 0  1.1e-04  2.8e-02  5.4e-11     2.6
HD-               6391    8  10T  1.0e+00,P 0  1.2e-04  2.8e-02  2.8e-12     3.0
HD-               6391    9  12T  1.0e+00,P 0  3.1e-07  9.0e-06  5.3e-12     3.5
Rank:  6391(8056) Iterations: 9
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [  10]  time =  1e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
--------------------------------------------------------------------------------
HD-               6391    1   8T  8.8e-01,P 0  6.3e+01  6.3e+01  3.0e-05     0.3
HD-               6391    2   7T  9.6e-01,P 0  3.6e+01  3.6e+01  3.6e-06     0.7
HD-               6391    3   7T  1.0e-00,P 0  1.8e+01  1.8e+01  1.6e-07     1.0
HD-               6391    4   7T  1.0e+00,P 0  7.4e+00  7.4e+00  1.6e-08     1.4
HD-               6391    5   8T  1.0e+00,P 0  2.4e+00  2.4e+00  3.1e-09     1.7
HD-               6391    6   7T  1.0e+00,P 0  4.6e-01  4.6e-01  1.1e-09     2.1
HD-               6391    7  10T  1.0e+00,P 0  1.9e+00  2.0e+00  6.0e-10     2.5
HD-               6391    8   7T  1.0e+00,P 0  3.0e-04  2.8e-02  2.1e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  3.0e-04  2.8e-02  2.8e-12     3.3
HD-               6391   10  10T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   9]  time =  1.1e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
--------------------------------------------------------------------------------
HD-               6391    1   8T  8.7e-01,P 0  6.5e+01  6.6e+01  3.0e-05     0.4
HD-               6391    2   8T  9.5e-01,P 0  3.9e+01  3.9e+01  3.8e-06     0.7
HD-               6391    3   7T  1.0e-00,P 0  2.0e+01  2.0e+01  2.0e-07     1.1
HD-               6391    4   8T  1.0e+00,P 0  8.4e+00  8.5e+00  2.2e-08     1.4
HD-               6391    5   7T  1.0e+00,P 0  3.0e+00  3.0e+00  4.1e-09     1.7
HD-               6391    6   7T  1.0e+00,P 0  6.7e-01  6.8e-01  1.3e-09     2.1
HD-               6391    7  10T  1.0e+00,P 0  2.1e+00  2.1e+00  6.8e-10     2.6
HD-               6391    8   7T  1.0e+00,P 0  4.2e-04  2.8e-02  2.6e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  5.3e-04  2.8e-02  2.8e-12     3.3
HD-               6391   10  11T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   8]  time =  1.2e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
--------------------------------------------------------------------------------
HD-               6391    1   8T  9.4e-01,P 0  4.3e+01  4.3e+01  1.8e-05     0.3
HD-               6391    2   7T  9.9e-01,P 0  2.5e+01  2.5e+01  1.1e-06     0.7
HD-               6391    3   8T  1.0e+00,P 0  1.2e+01  1.2e+01  4.0e-08     1.0
HD-               6391    4   7T  1.0e+00,P 0  5.0e+00  5.0e+00  8.6e-09     1.4
HD-               6391    5   7T  1.0e+00,P 0  1.7e+00  1.7e+00  1.9e-09     1.7
HD-               6391    6   7T  1.0e+00,P 0  3.7e-01  3.7e-01  8.2e-10     2.1
HD-               6391    7  10T  1.0e+00,P 0  1.4e+00  1.4e+00  4.3e-10     2.5
HD-               6391    8   7T  1.0e+00,P 0  1.7e-04  2.8e-02  1.3e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  2.1e-04  2.8e-02  2.8e-12     3.3
HD-               6391   10  11T  1.0e+00,P 0  3.1e-07  8.8e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
--------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   7]  time =  1.3e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
```

```
------------------------------------------------------------------------------
HD-               6391    1   7T  1.0e+00,P 0  3.2e+00  3.3e+00  2.8e-10     0.3
HD-               6391    2   7T  1.0e+00,P 0  1.8e+00  1.8e+00  6.8e-10     0.7
HD-               6391    3   8T  1.0e+00,P 0  8.6e-01  8.6e-01  2.6e-10     1.0
HD-               6391    4  10T  1.0e+00,P 0  4.0e-01  4.1e-01  1.3e-10     1.5
HD-               6391    5  10T  1.0e+00,P 0  1.5e-05  4.3e-05  7.2e-12     1.9
Rank:  6391(8056) Iterations: 5
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   6]  time =  1.4e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   8T  9.6e-01,P 0  3.8e+01  3.8e+01  1.8e-05     0.3
HD-               6391    2   7T  1.0e-00,P 0  2.2e+01  2.2e+01  8.2e-07     0.7
HD-               6391    3   8T  1.0e+00,P 0  1.0e+01  1.0e+01  2.8e-08     1.0
HD-               6391    4   8T  1.0e+00,P 0  4.2e+00  4.2e+00  7.1e-09     1.4
HD-               6391    5   8T  1.0e+00,P 0  1.4e+00  1.4e+00  1.6e-09     1.7
HD-               6391    6   7T  1.0e+00,P 0  2.6e-01  2.7e-01  7.2e-10     2.1
HD-               6391    7  12T  1.0e+00,P 0  1.3e+00  1.3e+00  3.7e-10     2.6
HD-               6391    8   7T  1.0e+00,P 0  1.1e-04  2.8e-02  1.0e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  1.3e-04  2.8e-02  2.8e-12     3.4
HD-               6391   10  12T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   5]  time =  1.5e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   8T  8.8e-01,P 0  6.4e+01  6.5e+01  3.0e-05     0.3
HD-               6391    2   8T  9.5e-01,P 0  3.8e+01  3.9e+01  3.7e-06     0.7
HD-               6391    3   8T  1.0e-00,P 0  1.9e+01  1.9e+01  1.9e-07     1.1
HD-               6391    4   8T  1.0e+00,P 0  7.9e+00  7.9e+00  2.3e-08     1.4
HD-               6391    5   8T  1.0e+00,P 0  2.7e+00  2.7e+00  4.6e-09     1.8
HD-               6391    6   9T  1.0e+00,P 0  5.4e-01  5.4e-01  1.3e-09     2.1
HD-               6391    7  10T  1.0e+00,P 0  2.1e+00  2.1e+00  6.4e-10     2.6
HD-               6391    8   7T  1.0e+00,P 0  3.7e-04  2.8e-02  2.5e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  3.9e-04  2.8e-02  2.8e-12     3.4
HD-               6391   10  12T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     3.9
Rank:  6391(8056) Iterations: 10
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   4]  time =  1.6e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
------------------------------------------------------------------------------
HD-               6391    1   7T  8.7e-01,P 0  6.6e+01  6.6e+01  3.0e-05     0.4
HD-               6391    2   8T  9.5e-01,P 0  3.8e+01  3.9e+01  3.8e-06     0.7
HD-               6391    3   7T  1.0e-00,P 0  1.9e+01  1.9e+01  1.9e-07     1.0
HD-               6391    4   8T  1.0e+00,P 0  7.5e+00  7.5e+00  2.0e-08     1.4
HD-               6391    5   8T  1.0e+00,P 0  2.4e+00  2.4e+00  4.0e-09     1.8
HD-               6391    6   9T  1.0e+00,P 0  3.7e-01  3.7e-01  1.2e-09     2.1
HD-               6391    7  10T  1.0e+00,P 0  2.1e+00  2.1e+00  6.0e-10     2.6
HD-               6391    8   7T  1.0e+00,P 0  4.1e-04  2.8e-02  2.1e-10     2.9
HD-               6391    9  11T  1.0e+00,P 0  1.0e-04  2.8e-02  2.9e-12     3.4
HD-               6391   10  10T  1.0e+00,P 0  3.1e-07  9.0e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   3]  time =  1.7e-06  s
```

```
MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
-------------------------------------------------------------------------------
HD-               6391    1   8T  9.4e-01,P 0  4.2e+01  4.2e+01  1.8e-05     0.3
HD-               6391    2   7T  9.9e-01,P 0  2.3e+01  2.3e+01  1.1e-06     0.7
HD-               6391    3   7T  1.0e+00,P 0  1.0e+01  1.0e+01  2.5e-08     1.0
HD-               6391    4   7T  1.0e+00,P 0  3.6e+00  3.7e+00  6.0e-09     1.4
HD-               6391    5   9T  1.0e+00,P 0  9.7e-01  9.8e-01  1.4e-09     1.7
HD-               6391    6  15T  1.0e+00,P 0  1.3e+00  1.3e+00  5.7e-10     2.2
HD-               6391    7   7T  1.0e+00,P 0  2.4e-04  2.8e-02  7.2e-11     2.6
HD-               6391    8  10T  1.0e+00,P 0  6.5e-05  2.9e-02  3.1e-12     3.1
HD-               6391    9  10T  1.0e+00,P 0  3.0e-07  9.1e-06  5.3e-12     3.5
Rank:  6391(8056) Iterations: 9
-------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   2]  time =  1.8e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
-------------------------------------------------------------------------------
HD-               6391    1   7T  1.0e+00,P 0  2.9e+00  3.0e+00  2.6e-10     0.3
HD-               6391    2   8T  1.0e+00,P 0  1.5e+00  1.5e+00  6.3e-10     0.7
HD-               6391    3   8T  1.0e+00,P 0  6.3e-01  6.3e-01  2.8e-10     1.1
HD-               6391    4  12T  1.0e+00,P 0  2.0e-01  2.1e-01  1.4e-10     1.5
HD-               6391    5  10T  1.0e+00,P 0  8.3e-06  2.6e-05  5.4e-12     2.0
Rank:  6391(8056) Iterations: 5
-------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   1]  time =  1.9e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
-------------------------------------------------------------------------------
HD-               6391    1   8T  9.6e-01,P 0  3.7e+01  3.7e+01  1.8e-05     0.4
HD-               6391    2   7T  1.0e-00,P 0  1.9e+01  1.9e+01  8.2e-07     0.7
HD-               6391    3   7T  1.0e+00,P 0  8.7e+00  8.7e+00  1.8e-08     1.1
HD-               6391    4   8T  1.0e+00,P 0  3.3e+00  3.3e+00  3.6e-09     1.4
HD-               6391    5   8T  1.0e+00,P 0  9.2e-01  9.3e-01  9.8e-10     1.8
HD-               6391    6  10T  1.0e+00,P 0  1.0e+00  1.1e+00  4.8e-10     2.2
HD-               6391    7   7T  1.0e+00,P 0  1.1e-04  2.8e-02  5.4e-11     2.6
HD-               6391    8  10T  1.0e+00,P 0  1.2e-04  2.8e-02  2.8e-12     3.0
HD-               6391    9  10T  1.0e+00,P 0  3.1e-07  9.0e-06  5.3e-12     3.5
Rank:  6391(8056) Iterations: 9
-------------------------------------------------------------------------------
SteppingControl: Constant algorithm : new delta=1e-07
Stepping: * [   0]  time =  2e-06  s

MixedHD          Matrix It   Sol  Damp         Pot      Total    RHS         CPU
Blockname         Size  Num       d,Scheme it  Update   Update   Norm          s
-------------------------------------------------------------------------------
HD-               6391    1   8T  8.8e-01,P 0  6.3e+01  6.3e+01  3.0e-05     0.3
HD-               6391    2   7T  9.6e-01,P 0  3.6e+01  3.6e+01  3.6e-06     0.7
HD-               6391    3   7T  1.0e-00,P 0  1.8e+01  1.8e+01  1.6e-07     1.0
HD-               6391    4   7T  1.0e+00,P 0  7.4e+00  7.4e+00  1.6e-08     1.4
HD-               6391    5   8T  1.0e+00,P 0  2.4e+00  2.4e+00  3.1e-09     1.7
HD-               6391    6   8T  1.0e+00,P 0  4.6e-01  4.6e-01  1.1e-09     2.1
HD-               6391    7  10T  1.0e+00,P 0  1.9e+00  2.0e+00  6.0e-10     2.5
HD-               6391    8   7T  1.0e+00,P 0  3.0e-04  2.8e-02  2.1e-10     2.9
HD-               6391    9  10T  1.0e+00,P 0  3.0e-04  2.8e-02  2.8e-12     3.3
HD-               6391   10  11T  1.0e+00,P 0  3.1e-07  8.9e-06  5.3e-12     3.8
Rank:  6391(8056) Iterations: 10
-------------------------------------------------------------------------------
==================================
       Constant Node Voltage
 ---------------------------------
```

```
 gnd           =  0.0000000000e+00 V
 ----------------------------------
        Fixed Node Voltage
 ----------------------------------
 B             =  2.8339136415e+00 V
 C             =  6.5669394905e+00 V
 E             =  2.1129744918e+00 V
 ----------------------------------
           Node Voltage
 ----------------------------------
 Vcc           =  1.5000000000e+01 V
 in            = -4.0425562203e-18 V
 ----------------------------------
          Branch Current
 ----------------------------------
 Vcc.I         =  2.1951657442e-04 A
 Vin.I         =  3.6773164294e-07 A
==================================

Wed Aug 10 17:17:55 2005: TERMINATING mmnt, CPU: 87.65, exit code 0
```

C.2 Make.Linux_WINZIG_mpi_gcc

```
#
# ----------------------------------------------------------------------
# - shell --------------------------------------------------------------
# ----------------------------------------------------------------------
#
SHELL        = /bin/sh
#
CD           = cd
CP           = cp
LN_S         = ln -s
MKDIR        = mkdir
RM           = /bin/rm -f
TOUCH        = touch
#
# ----------------------------------------------------------------------
# - Platform identifier ------------------------------------------------
# ----------------------------------------------------------------------
#
ARCH         = Linux_WINZIG_mpi_gcc
#
# ----------------------------------------------------------------------
# - HPL Directory Structure / HPL library ------------------------------
# ----------------------------------------------------------------------
#
TOPdir       = $(HOME)/hpl
INCdir       = $(TOPdir)/include
BINdir       = $(TOPdir)/bin/$(ARCH)
LIBdir       = $(TOPdir)/lib/$(ARCH)
#
HPLlib       = $(LIBdir)/libhpl.a
#
# ----------------------------------------------------------------------
# - Message Passing library (MPI) --------------------------------------
# ----------------------------------------------------------------------
# MPinc tells the  C  compiler where to find the Message Passing library
# header files,  MPlib  is defined  to be the name of  the library to be
# used. The variable MPdir is only used for defining MPinc and MPlib.
#
MPdir        = /usr/lib/mpich
MPinc        = -I$(MPdir)/include
MPlib        = $(MPdir)/lib/libmpich.a
#
# ----------------------------------------------------------------------
# - Linear Algebra library (BLAS or VSIPL) -----------------------------
# ----------------------------------------------------------------------
# LAinc tells the  C  compiler where to find the Linear Algebra  library
# header files,  LAlib  is defined  to be the name of  the library to be
# used. The variable LAdir is only used for defining LAinc and LAlib.
#
LAdir        = /usr/lib/sse2
LAinc        =
LAlib        = $(LAdir)/libcblas.a $(LAdir)/libatlas.a
#
# ----------------------------------------------------------------------
# - F77 / C interface --------------------------------------------------
# ----------------------------------------------------------------------
# You can skip this section  if and only if  you are not planning to use
# a  BLAS  library featuring a Fortran 77 interface.  Otherwise,  it  is
# necessary  to  fill out the  F2CDEFS  variable  with  the  appropriate
# options.  **One and only one**  option should be chosen in **each** of
# the 3 following categories:
#
```

```
# 1) name space (How C calls a Fortran 77 routine)
#
# -DAdd_              : all lower case and a suffixed underscore  (Suns,
#                       Intel, ...),                           [default]
# -DNoChange          : all lower case (IBM RS6000),
# -DUpCase            : all upper case (Cray),
# -DAdd__             : the FORTRAN compiler in use is f2c.
#
# 2) C and Fortran 77 integer mapping
#
# -DF77_INTEGER=int   : Fortran 77 INTEGER is a C int,         [default]
# -DF77_INTEGER=long  : Fortran 77 INTEGER is a C long,
# -DF77_INTEGER=short : Fortran 77 INTEGER is a C short.
#
# 3) Fortran 77 string handling
#
# -DStringSunStyle    : The string address is passed at the string loca-
#                       tion on the stack, and the string length is then
#                       passed as  an  F77_INTEGER  after  all  explicit
#                       stack arguments,                       [default]
# -DStringStructPtr   : The address  of  a  structure  is  passed  by  a
#                       Fortran 77  string,  and the structure is of the
#                       form: struct char *cp; F77_INTEGER len;,
# -DStringStructVal   : A structure is passed by value for each  Fortran
#                       77 string,  and  the  structure is  of the form:
#                       struct char *cp; F77_INTEGER len;,
# -DStringCrayStyle   : Special option for  Cray  machines,  which  uses
#                       Cray  fcd  (fortran  character  descriptor)  for
#                       interoperation.
#
F2CDEFS      =
#
# ----------------------------------------------------------------------
# - HPL includes / libraries / specifics -------------------------------
# ----------------------------------------------------------------------
#
HPL_INCLUDES = -I$(INCdir) -I$(INCdir)/$(ARCH) $(LAinc) $(MPinc)
HPL_LIBS     = $(HPLlib) $(LAlib) $(MPlib)
#
# - Compile time options -----------------------------------------------
#
# -DHPL_COPY_L           force the copy of the panel L before bcast;
# -DHPL_CALL_CBLAS       call the cblas interface;
# -DHPL_CALL_VSIPL       call the vsip  library;
# -DHPL_DETAILED_TIMING  enable detailed timers;
#
# By default HPL will:
#    *) not copy L before broadcast,
#    *) call the BLAS Fortran 77 interface,
#    *) not display detailed timing information.
#
HPL_OPTS     = -DHPL_CALL_CBLAS
#
# ----------------------------------------------------------------------
#
HPL_DEFS     = $(F2CDEFS) $(HPL_OPTS) $(HPL_INCLUDES)
#
# ----------------------------------------------------------------------
# - Compilers / linkers - Optimization flags ---------------------------
# ----------------------------------------------------------------------
#
CC           = $(MPdir)/bin/mpicc
CCNOOPT      = $(HPL_DEFS)
CCFLAGS      = $(HPL_DEFS) -O3 -fomit-frame-pointer -funroll-loops
#
```

```
# On some platforms,  it is necessary  to use the Fortran linker to find
# the Fortran internals used in the BLAS library.
#
LINKER       = $(MPdir)/bin/mpicc
#LINKER       = $(MPdir)/bin/mpicc -mpitrace
LINKFLAGS    = $(CCFLAGS)
#
ARCHIVER     = ar
ARFLAGS      = r
RANLIB       = echo
#
# ----------------------------------------------------------------------
```

C.3 Make.Linux_WINZIG_mpi_icc

```
#
# ----------------------------------------------------------------------
# - shell --------------------------------------------------------------
# ----------------------------------------------------------------------
#
SHELL        = /bin/sh
#
CD           = cd
CP           = cp
LN_S         = ln -s
MKDIR        = mkdir
RM           = /bin/rm -f
TOUCH        = touch
#
# ----------------------------------------------------------------------
# - Platform identifier ------------------------------------------------
# ----------------------------------------------------------------------
#
ARCH         = Linux_WINZIG_mpi_icc
#
# ----------------------------------------------------------------------
# - HPL Directory Structure / HPL library ------------------------------
# ----------------------------------------------------------------------
#
TOPdir       = $(HOME)/hpl
INCdir       = $(TOPdir)/include
BINdir       = $(TOPdir)/bin/$(ARCH)
LIBdir       = $(TOPdir)/lib/$(ARCH)
#
HPLlib       = $(LIBdir)/libhpl.a
#
# ----------------------------------------------------------------------
# - Message Passing library (MPI) --------------------------------------
# ----------------------------------------------------------------------
# MPinc tells the  C  compiler where to find the Message Passing library
# header files,  MPlib  is defined  to be the name of  the library to be
# used. The variable MPdir is only used for defining MPinc and MPlib.
#
MPdir        = /usr/lib/mpich
MPinc        = -I$(MPdir)/include
MPlib        = $(MPdir)/lib/libmpich.a
#
# ----------------------------------------------------------------------
# - Linear Algebra library (BLAS or VSIPL) -----------------------------
# ----------------------------------------------------------------------
# LAinc tells the  C  compiler where to find the Linear Algebra  library
# header files,  LAlib  is defined  to be the name of  the library to be
# used. The variable LAdir is only used for defining LAinc and LAlib.
#
LAdir        = /opt/intel/mkl72/lib/32
LAinc        = -I$(LAdir) -I/usr/include \
               -I/opt/intel_cc_80/include \
               -I/usr/lib/gcc-lib/i486-linux/3.3.5/include
LAlib        = -L$(LAdir) -lpthread -lmkl -lguide
#
# ----------------------------------------------------------------------
# - F77 / C interface --------------------------------------------------
# ----------------------------------------------------------------------
# You can skip this section  if and only if  you are not planning to use
# a  BLAS  library featuring a Fortran 77 interface.  Otherwise,  it  is
# necessary  to  fill out the  F2CDEFS  variable  with  the  appropriate
# options.  **One and only one**  option should be chosen in **each** of
```

```
# the 3 following categories:
#
# 1) name space (How C calls a Fortran 77 routine)
#
# -DAdd_              : all lower case and a suffixed underscore  (Suns,
#                       Intel, ...),                           [default]
# -DNoChange          : all lower case (IBM RS6000),
# -DUpCase            : all upper case (Cray),
# -DAdd__             : the FORTRAN compiler in use is f2c.
#
# 2) C and Fortran 77 integer mapping
#
# -DF77_INTEGER=int   : Fortran 77 INTEGER is a C int,         [default]
# -DF77_INTEGER=long  : Fortran 77 INTEGER is a C long,
# -DF77_INTEGER=short : Fortran 77 INTEGER is a C short.
#
# 3) Fortran 77 string handling
#
# -DStringSunStyle    : The string address is passed at the string loca-
#                       tion on the stack, and the string length is then
#                       passed as  an  F77_INTEGER  after  all  explicit
#                       stack arguments,                       [default]
# -DStringStructPtr   : The address  of  a  structure  is  passed  by  a
#                       Fortran 77  string,  and the structure is of the
#                       form: struct char *cp; F77_INTEGER len;,
# -DStringStructVal   : A structure is passed by value for each  Fortran
#                       77 string,  and  the  structure is  of the form:
#                       struct char *cp; F77_INTEGER len;,
# -DStringCrayStyle   : Special option for  Cray  machines,  which  uses
#                       Cray  fcd  (fortran  character  descriptor)  for
#                       interoperation.
#
F2CDEFS      =
#
# ----------------------------------------------------------------------
# - HPL includes / libraries / specifics -------------------------------
# ----------------------------------------------------------------------
#
HPL_INCLUDES = -I$(INCdir) -I$(INCdir)/$(ARCH) $(LAinc) $(MPinc)
HPL_LIBS     = $(HPLlib) $(LAlib) $(MPlib)
#
# - Compile time options -----------------------------------------------
#
# -DHPL_COPY_L           force the copy of the panel L before bcast;
# -DHPL_CALL_CBLAS       call the cblas interface;
# -DHPL_CALL_VSIPL       call the vsip  library;
# -DHPL_DETAILED_TIMING  enable detailed timers;
#
# By default HPL will:
#    *) not copy L before broadcast,
#    *) call the BLAS Fortran 77 interface,
#    *) not display detailed timing information.
#
HPL_OPTS     = -DHPL_CALL_CBLAS
#
# ----------------------------------------------------------------------
#
HPL_DEFS     = $(F2CDEFS) $(HPL_OPTS) $(HPL_INCLUDES)
#
# ----------------------------------------------------------------------
# - Compilers / linkers - Optimization flags ---------------------------
# ----------------------------------------------------------------------
#
CC           = $(MPdir)/bin/mpicc -config=icc
CCNOOPT      = $(HPL_DEFS)
```

```
CCFLAGS      = $(HPL_DEFS) -O3
#
# On some platforms,  it is necessary  to use the Fortran linker to find
# the Fortran internals used in the BLAS library.
#
LINKER       = $(MPdir)/bin/mpicc -config=icc
#LINKER       = $(MPdir)/bin/mpicc -config=icc -mpitrace
LINKFLAGS    = $(CCFLAGS)
#
ARCHIVER     = ar
ARFLAGS      = r
RANLIB       = echo
#
# ----------------------------------------------------------------------
```

C.4 HPL.dat

```
HPLinpack benchmark input file
Innovative Computing Laboratory, University of Tennessee
HPL.out      output file name (if any)
6            device out (6=stdout,7=stderr,file)
1            # of problems sizes (N)
22000
5            # of NBs
180 190 200 210 220
0            PMAP process mapping (0=Row-,1=Column-major)
1            # of process grids (P x Q)
4 Ps
8      Qs
16.0         threshold
1            # of panel fact
0 1 2        PFACTs (0=left, 1=Crout, 2=Right)
1            # of recursive stopping criterium
2 4 8          NBMINs (>= 1)
1            # of panels in recursion
2 4          NDIVs
1            # of recursive panel fact.
0 1 2        RFACTs (0=left, 1=Crout, 2=Right)
1            # of broadcast
0 1 2 3 4 5  BCASTs (0=1rg,1=1rM,2=2rg,3=2rM,4=Lng,5=LnM)
1            # of lookahead depth
0            DEPTHs (>=0)
2            SWAP (0=bin-exch,1=long,2=mix)
64           swapping threshold
0            L1 in (0=transposed,1=no-transposed) form
0            U  in (0=transposed,1=no-transposed) form
0            Equilibration (0=no,1=yes)
8            memory alignment in double (> 0)
```

C.5 HPL Result File

```
================================================================================
HPLinpack 1.0a  --  High-Performance Linpack benchmark  --   January 20, 2004
Written by A. Petitet and R. Clint Whaley,  Innovative Computing Labs.,  UTK
================================================================================

An explanation of the input/output parameters follows:
T/V    : Wall time / encoded variant.
N      : The order of the coefficient matrix A.
NB     : The partitioning blocking factor.
P      : The number of process rows.
Q      : The number of process columns.
Time   : Time in seconds to solve the linear system.
Gflops : Rate of execution for solving the linear system.

The following parameter values will be used:

N      :   22000
NB     :     180      190      200      210      220
PMAP   : Row-major process mapping
P      :       4
Q      :       8
PFACT  :    Left
NBMIN  :       2
NDIV   :       2
RFACT  :    Left
BCAST  :   1ring
DEPTH  :       0
SWAP   : Mix (threshold = 64)
L1     : transposed form
U      : transposed form
EQUIL  : no
ALIGN  : 8 double precision words

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

- The matrix A is randomly generated for each test.
- The following scaled residual checks will be computed:
   1) ||Ax-b||_oo / ( eps * ||A||_1  * N        )
   2) ||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1  )
   3) ||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo )
- The relative machine precision (eps) is taken to be          2.220446e-16
- Computational tests pass if scaled residuals are less than           16.0

================================================================================
T/V                N    NB     P     Q               Time             Gflops
--------------------------------------------------------------------------------
WR00L2L2       22000   180     4     8             238.94          2.971e+01
--------------------------------------------------------------------------------
||Ax-b||_oo / ( eps * ||A||_1  * N        ) =        0.0139836 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1  ) =        0.0066454 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) =        0.0013102 ...... PASSED
================================================================================
T/V                N    NB     P     Q               Time             Gflops
--------------------------------------------------------------------------------
WR00L2L2       22000   190     4     8             239.92          2.959e+01
--------------------------------------------------------------------------------
||Ax-b||_oo / ( eps * ||A||_1  * N        ) =        0.0154894 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1  ) =        0.0073611 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) =        0.0014513 ...... PASSED
================================================================================
T/V                N    NB     P     Q               Time             Gflops
--------------------------------------------------------------------------------
```

```
WR00L2L2       22000   200     4     8             241.75          2.937e+01
--------------------------------------------------------------------------------
||Ax-b||_oo / ( eps * ||A||_1  * N        ) =        0.0174208 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1  ) =        0.0082790 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) =        0.0016322 ...... PASSED
================================================================================
T/V                N    NB     P     Q               Time             Gflops
--------------------------------------------------------------------------------
WR00L2L2       22000   210     4     8             244.10          2.908e+01
--------------------------------------------------------------------------------
||Ax-b||_oo / ( eps * ||A||_1  * N        ) =        0.0150298 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1  ) =        0.0071427 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) =        0.0014082 ...... PASSED
================================================================================
T/V                N    NB     P     Q               Time             Gflops
--------------------------------------------------------------------------------
WR00L2L2       22000   220     4     8             244.21          2.907e+01
--------------------------------------------------------------------------------
||Ax-b||_oo / ( eps * ||A||_1  * N        ) =        0.0145396 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1  ) =        0.0069097 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) =        0.0013623 ...... PASSED
================================================================================

Finished      5 tests with the following results:
              5 tests completed and passed residual checks,
              0 tests completed and failed residual checks,
              0 tests skipped because of illegal input values.
--------------------------------------------------------------------------------

End of Tests.
================================================================================
```