

Setting Up Mac OS X 10.4 Server and Clients for Xgrid, Xgrid Enabled OpenMPI, LAM-MPI, and MPICH2

Gergely V. Záruba
zaruba@uta.edu

Version: 2007-09-10

***Abstract** - In this document we outline some simple (and sometimes slightly dirty) methods for setting up a MAC OS X 10.4 cluster for Xgrid (SETI@home style processing) and three popular versions of MPI, i.e., OpenMPI, LAM-MPI, and MPICH2. Our sample setup consists of one MAC OS X 10.4 server and three MAC OS X 10.4 (client) Intel based MAC Mini computers. There are many resources available on the Internet that try to make parts of such installations documented, however many of them do not work on OS X 10.4, do not provide enough information, or provide too much information (thus obfuscating the essence). We will try to make this document a usable reference for an average computer user with some general *nix and/or MAC user interface knowledge. This document is by far not a complete or a comprehensive guide to Mac OS 10.4 and in many cases we will use some tricks to get to our goal. There obviously is no liability assumed for any of the procedures described below and many of them do not consider security to be a major problem as our cluster is sitting behind a firewalled NAT router.*

1. Introduction

We will start our journey with some notes on installing MAC OS X 10.4 Server (from now on referred to as S10.4). Next we describe setting up Xgrid. We then proceed to creating a common nfs directory (readable and writable) that all nodes will see at the same place in their directory structure (necessary for OpenMPI, LAM-MPI, and MPICH2). Setup of OpenMPI will follow. We then show a quick method for installing LAM-MPI and MPICH2 in the next consecutive sections. Finally, we describe the setup process of a simple LDAP Open Directory so the S10.4 serves users and directories for the 10.4 Clients (C10.4).

Our sample physical setup consists of a dual core Intel MAC Mini running the S10.4 and three single core Intel MAC minis running C10.4. They are all connected via Gigabit Ethernet to a NAT router with DHCP capabilities (on a 192.168.x.x domain). The names we will use are: *mmac-srvr00*, *mmac01*, *mmac02*, and *mmac03*. In OS X, computers in the same Ethernet domain can see each other using the “local” domain, thus the computers can ping or connect to each other by the names: *mmac-srvr00.local*, *mmac01.local*, *mmac02.local*, and *mmac03.local*. In our setup we also use a Mac mini look-alike IOGear MiniView KVM switch for simple access to the MAC Minis.

The References section contains links to online documents and resources that were extremely helpful when setting up our cluster. We will try to do our best to reference

them at the right point in the instructions. In addition we have used many man pages and general `**nix`, and MAC OSX resources.

2. Installation Notes for S10.4

Everyone working with OSX 10.4 must have realized that the default user on C10.4 and/or the admin user on S10.4 are not superusers (root). To gain superuser rights in a terminal session, a `“su -”` or `“sudo su -”` command in a terminal window needs to be entered on S10.4 and C10.4 respectively. Alternatively, commands that require root-access maybe invoked with placing `“sudo”` in front of them. The password for root is the same as the administrator password on S10.4 or the single user password on the C10.4.

When installing S10.4 it is recommended (default option) to create an admin (long name: administrator) user. For the computer name we entered: `mmac-srvr00`, which can relatively easily be changed from the “Server Administration” tool. Leave the DHCP settings as they are, i.e., let IPv4 be configured by DHCP but enter the server’s name (`mmac-srvr00`) as the DHCP client id. Add `“local”` to the search domain if possible.

When it comes to “Directory usage”, set the computer to “Open Directory Master” (and if you will not be using any Windows clients, disable the “Windows Primary Domain Controller” checkbox). The software is going to ask you to create an administrative account for the open directory master, for which the name `“diradmin”` is recommended with the same password as that of `“admin”`. You will also need to enable The Xgrid agent, and Xgrid controller. (You can do all of this later as well from the Server Administration tool.)

You will likely be prompted to update your software components, once the server is set up. After finishing installing all updates, you will need to install XCode from the same disc as you installed S10.4 from, i.e., insert the disc, select other installs, XCode, XcodeTools.mpkg, and follow the installation instructions (a default installation is likely going to suffice). One last thing to do (that is, if you like to use the `“locate”` utility to find files on your computer) is to open a terminal window and type: `“sudo /usr/libexec/locate.updatedb”`; this will create the index database required by `“locate”`. It is recommended that you do this last step on all C10.4 computers as well.

3. Setting up Xgrid

We need to mention here, as it is a little known fact, that even a C10.4 can act as an Xgrid controller, by downloading the free “Server Administration” tool from Apple. In this section we use information from [3].

3.1. Xgrid Controller on S10.4

For Xgrid to work, there needs to be a controller that accepts Xgrid agents (other computers) offering to perform jobs. An Xgrid controller does not necessarily need to be an agent as well, but why waste any resources? Thus, we will set up the S10.4 to work as a controller as well as an agent to the controller. For this, start the computer and log in as `“admin”`. Open the “Server Admin” tool (the one that looks like a globe), and log in to it

using the admin credentials. In the leftmost part of the window you will see your servers name with an arrow (well, a triangle) to its left; if this arrow does not point downwards, then click on it. Now you will see all the services that are, or could be provided by your server. If you just installed your S10.4 with the above instructions, then Xgrid should have a green disc next to it, indicating that it is running.

To set up Xgrid, click on Xgrid (highlighting it), then click on settings in the right pane. Click on “Controller” and make sure that the “enable controller service” box is checked. Set the client and agent passwords (e.g., client password: *cls-client*, and agent password: *cls-agent*). The agent password is needed for agents to connect to the controller offering their services, while the client password is needed to submit or monitor jobs. Click on “Save” and you will be offered to restart the service (which offer you cannot refuse). Once done, click on “Agent” and make sure that the “enable agent service” is checked. Select the “use specific controller” option and choose your (already running) server: *mmac-srvr00*. Select “always” for “agent accepts tasks” (if the primary purpose of your computers is to be a cluster). Click on “Save” and take the restart offer. If you click on “Overview” you should see your own server acting as an agent. The easy part is done.

Xgrid Admin is a graphical tool to monitor your Xgrid; you can find it under “Applications→Server→Xgrid Admin” (do yourself a favor and drag it out to your bottom toolbar). When you start it up it will try to connect as a client to your Xgrid, thus you will need to select your controller “*mmac-srvr00*” and enter the client password that you have just set (e.g., *cls-client*). Now you will get a nice meter showing usage of your cluster or you can check its capabilities. With this tool you can get status info on your agents, on your jobs, or remove jobs or agents (removing connected agents will not prevent them to reconnect automatically, for that you will have to stop their offering at their local settings).

There is **one last thing** left (that you will need to do on all computers that you will want to use to submit jobs from), that is to set some environment variables. Here we assume that you are using *bash* (the default shell with OSX 10.4). Open a terminal window, and edit the */etc/profile* file in a superuser mode (e.g., by typing: “`sudo emacs /etc/profile`” you can save the file with the sequence: CTRL-X CTRL-S and exit emacs with a CTRL-X CTRL-C sequence). You need to add the following two lines to */etc/profile*:

```
export XGRID_CONTROLLER_HOSTNAME=mmac-srvr00.local
export XGRID_CONTROLLER_PASSWORD=cls-client
```

obviously, with replacing the server name and the password to your own. From now on when you start a terminal window these variables will all be set and you can easily submit Xgrid jobs. (Note: 1. you can also put the above environmental variable definitions in *~/profile* or *~/bashrc* which will only set them for the current user, and 2. once you are editing */etc/profile*, you may want to add “`:/usr/local/bin:.`” To your PATH). Close the terminal window and open a new one (so your new variables are loaded). Type: “`xgrid -grid list`”; if your output is not something like: “`{gridList = (0);}`” then check that you have entered the above XGRID_* variables correctly (by editing the corresponding file again, or by typing: “`set | grep XGRID`”). You can now submit jobs using the command line tool , e.g., “`xgrid -job submit prog_name`”.

3.2. Xgrid Agents on C10.4

Starting Xgrid agents on additional C10.4 machines is very easy. Click on “System Preferences” (the light switch icon) and select “Sharing” (you may need to open the lock at the bottom of the window if it is closed by clicking on it and entering an administrator password). Highlight “Xgrid” from among the list and click on “Configure”. Use “specific controller” and select your server (*mmac-srvr00*). (Thanks to *Bonjour*, a service discovery protocol, all active Xgrid controllers will show up to be selected). Select “Agent accepts tasks: always” (see note at the previous section) and enter your agent password (e.g., *cls-agent*) for “Password”. Start, or restart (stop and start) the Xgrid agent.

If you want to submit jobs from this computer as well, see the “**one last thing**” note at the previous section.

4. Setting up a Common NFS Shared Directory

Since OpenMPI, LAM-MPI, and MPICH2 need the user’s executable to be available on all nodes, the easiest (and due to the nature of some file-opening MPI programs) and best way to do that is to have a directory on all nodes that is shared and has the same absolute path. As we had problems with automatic (boot-time) sharing setup with AFP (Apple Filing Protocol) we are going to describe a straightforward (however not globally secure) way of sharing a common directory with nfs. The name and absolute path of this directory is going to be “*/NetCommon*”. We are going to use some of the information collected from [7][10].

4.1. NFS Sharing on S10.4 server

Open a terminal window and for the sake of simplicity obtain root access (“`su -`”). Create the directory to be shared and give it a generous access right: “`mkdir /NetCommon; chmod 777 /NetCommon`”. Start “Workgroup Manager” and log in to it as “*diradmin*”. Click on “Sharing” on top and then click “All”, select the “MacintoshHD” and then highlight the “NetCommon” folder. On the left, click on “General” and check “Share this item”, click on “Save”. Click on Protocols and go through all offered protocols unchecking and then saving each an every one (Windows, FTP, Apple File Sharing) except for NFS which you should enable. While the NFS protocol is showing, uncheck “read-only” but leave both “mapping” related boxes checked. To make things simple, select “World” next to “Export this item to”; when done click on save. Since now you have at least one NFS shared folder, your NFS service should have been automatically enabled by the OS. To check on that, click “Admin” on top (make sure that the arrow on the left of your server’s name points downwards) and see if the dot next to NFS is green (which it should be). Additionally, the settings can be checked from a terminal window, by typing: “`nidump -r /exports /`”.

4.2. NFS Import on C10.4 client

Open “NetInfo Manager” (found under “Applications→Utilities”), and unlock the lock at the bottom by clicking on it and entering your admin password. If you do not have

a “*mounts*” entry, you will need to create one by clicking on the top tool bar “Directory→New Subdirectory” and rename the new directory to *mounts* (by changing the value of the “name” attribute in the bottom part of the window); click on “Domain→Save changes”. Click on the newly created “*mounts*”, then on “Edit→Add” and then three times on “Directory→New property”. With these commands, we have created a sub-entry under *mounts* and three additional properties for this new entry. We have to change now these new property entries to:

Property:	Value:
name	mmac-srvr00.local:/NetCommon
dir	/NetCommon
type	Nfs
opts	-P

After having entered the above entries, click on “Domain→Save changes”. Now, *automount* is told to mount the “/NetCommon” folder from *mmac-srvr00.local* using *nfs* to the local mount point “/NetCommon”. All there is left is to create the local mount point using a terminal window, entering the commands:

```
sudo mkdir /NetCommon
sudo chmod 777 /NetCommon"
```

The easiest thing to do now is to restart the computer; alternatively, restarting the *automount* daemon may work too: “`sudo kill -1 `cat /var/run/automount.pid``” (note the backward quotes around *cat*, which will put the results of the enclosed command in its place).

5. Setting up OpenMPI

OpenMPI can be downloaded in an executable form for OS X from the OpenMPI website, however these binaries do not have Fortran support, a language is still used by many, in scientific computing. Additionally we would want an OpenMPI compilation that uses the Xgrid infrastructure. Thus we are going to download sources and compile them into binaries, which we will then distribute over all computers. To compile these set of instructions we have used resources [1][5][8][11][12].

5.1. OpenMpi on S10.4

Download the OSX version of the *gfortran* binaries from [13]; once downloaded, install them. The *gfortran* libraries will be installed in */usr/local/gfortran*, thus we will need to extend our search path. Edit the */etc/profile* file (e.g., “`sudo emacs /etc/profile`”) and add “`:/usr/local/gfortran/bin`” to the end of the *PATH* variable (close the terminal window, open a new one and see that typing “*gfortran*” invokes the compiler).

Download OpenMPI sources from [8]. Open a terminal window and become superuser “`su -`”. Move the source tar file to a suitable directory (e.g., */var/private/root*) and unpack it (“`tar -xvf openmpi*.tar`”); change into the directory containing the source (e.g., “`cd openmpi-1.2.3`”). Create a directory for your build and change to it, configure the compilation, build it, and install it:

```
mkdir build
```

```

cd build
./configure --prefix /usr/local/openmpi --with-fortran
--enable-shared --with-xgrid
make
make install

```

This will install the binaries into `/usr/local/openmpi`, thus we need to add them to the search path by appending `"/usr/local/openmpi/bin"` to `PATH` in `/etc/profile` (e.g., by `"emacs /etc/profile"`).

In order to work with OpenMPI easily, there needs to be a common shared directory over all nodes; this was one of the main reasons we have set up the NFS shared `/NetCommon` directory in Section 4. Create a "tgz" archive from the binaries and place it into the common directory:

```

cd /usr/local/openmpi
tar -cvzf openmpibinaries.tgz /usr/local/openmpi
mv openmpibinaries.tgz /NetCommon

```

5.2. OpenMPI on all C10.4-s

For OpenMPI to work, it needs to be installed on all cluster nodes; thus, use the archive in the common directory for this install. Open a terminal window and become superuser (`"sudo su -"`). Install the binaries:

```

cp /NetCommon/openmpibinaries.tgz /
cd /
tar -xvzf openmpibinaries.tgz
rm /openmpibinaries.tgz

```

Do not forget to add OpenMpi to the search path by appending `"/usr/local/openmpi/bin"` to `PATH` in `/etc/profile` (e.g., by `"emacs /etc/profile"`).

5.3. Running OpenMPI Programs

You can compile OpenMPI programs on the server. To run OpenMPI enabled programs, copy them into (a subdirectory of) `/NetCommon` and execute binaries using `"/usr/local/openmpi/bin/mpiexec -n nr_procs /NetCommon/your_executable"`. You can monitor their progress with the Xgrid Admin tool. A sample ping-pong program (one, that sends "ping" MPI messages among nodes) slightly modified from [11] can be downloaded from [12] (compile with `make`, use only even number of processors when running).

6. Setting up LAM-MPI

In this set of instructions we are going to make LAM-MPI work quickly but relatively dirty. LAM-MPI is not undergoing any more development as they have joined forces with OpenMPI; however, they are still fixing bugs. Unfortunately, LAM-MPI does not come with support for Xgrid thus we will need to use another method to make the nodes form an MPI cluster. For that we will need to use `ssh`. The first thing to do is to make sure remote login is enabled on all machines (use "System preferences→Sharing" to enable remote login). For LAM-MPI to work, `ssh` needs to be configured so that the user starting it up does not need a password to log into any of the other machines. Thus, we

recommend (and are going to assume this from now on) that an *admin* (local) account is available on all machines. (You can use the “System preferences→Accounts” to set up an admin user on the clients and use “System preferences→Sharing” to enable remote login). Make sure that all nodes have “local” in their network domain search by adding “local” into the domain search in “System preferences→Network” We are using references [3] and [6] for this description.

We will not be able to easily partition these instructions into client and server part as commands rely on each other among all nodes.

1. On the S10.4: download LAM-MPI from [14], get the OS X installer version (unfortunately it comes without Fortran support). Install it (click on it once downloaded), and then copy the installer package to `/NetCommon` so it can be installed on all other computers easily.
2. On all the C10.4-s: install LAM-MPI from `/NetCommon`
3. On all nodes: when a non-login interactive shell is invoked (something that LAM-MPI does while booting) then `/etc/profile` is not parsed by bash but `~/.bashrc` is. Thus we will need to create or modify `~/.bashrc` (e.g., using “`emacs ~/.bashrc`”) to include the following line (make sure admin account is used):
`export PATH="$PATH:/usr/local/bin"`
as LAM-MPI was directly installed into `/usr/local`.
4. On the S10.4: create a “`hostfile.lam`” file (e.g., “`emacs hostfile.lam`”) with each line containing a name of a node (extended by `cpu=2` if it has two or a dual core CPU, `cpu=4` for a quad-core). For example in our infrastructure’s `hostfile.lam` is:
`mamac-servr00.local cpu=2`
`mamac01.local`
`mamac02.local`
`mamac03.local`
5. On the S10.4: open a terminal as admin; type: “`ssh-keygen -t rsa`”; use the default key-pair name (`id_rsa`) and enter a password of your choosing. This has created a key-pair in the “`~admin/.ssh`” directory. Copy `id_rsa.pub` from `.ssh` to `/NetCommon` as `authorized_keys2`:
“`cp ~/.ssh/id_rsa.pub /NetCommon/authorized_keys2`”.
6. On all the C10.4-s: open a terminal window as admin (or log in from the server using ssh: “`ssh admin@cms1-mmac01`”). Make sure you have a `.ssh` directory, if you do not then use ssh to log into the same computer: (“`ssh localhost`”). Copy `authorized_keys2` into `.ssh`:
“`cp /NetCommon/authorized_keys2 ~/.ssh`”. With the above steps we have shared a public/private key-pair among nodes that is going to be essential a little later.
7. (S10.4) To be able to boot LAM-MPI you will need to have an ssh-agent running that can tell your login ssh to read the private key that we have created above. Thus (and each time you need to set up LAM-MPI), open a terminal window and type: “`ssh-agent -s`”. This will output two lines of

settings; copy its output and paste it into your terminal window (you can do this easily with the following commands:

```
ssh-agent -s >ssh_ebbe.del
chmod 700 ssh_ebbe.del
./ssh_ebbe.del
rm ssh_ebbe.del
```

This will set environmental variables for *ssh-agent* to work correctly. Now we need to tell *ssh-agent* to use the private key, thus type: “ssh-add” (and type your password when asked). Now you can try to log in to each of the client nodes by e.g., “ssh mmac0x.local” and the system should not ask you for a password (you may need to do that twice so the respective system’s key is added to you “known hosts” file).

8. (S10.4) The last thing to do is to start LAM-MPI, type: “lamboot -v -ssi boot rsh hostfile.lam”. This will start your LAM-MPI (*lamd* daemons) on all the machines included in *hostfile.lam*.
9. (S10.4) Place and start your LAM-MPI programs from */NetCommon* just like you did with Open-MPI. Make sure files and directories have proper access rights.

If the S10.4 computer has been rebooted, you will need to repeat from Step 7 to get LAMMPI working again.

7. Setting up MPICH2

Setting up MPICH2 is similar to setting up LAM-MPI. Make sure you have *gcc* and *gofortran* (you should if you have followed instructions in the OpenMPI installation section). Just like in the previous section we cannot easily separate instructions into two separate subsections, thus in the beginning of each step we will say which computers they need to be applied on. Please read the introduction of Section 6 carefully.

1. On the S10.4: download the MPICH2 source from [8]. Become superuser (“su -”) and untar the source into its own directory. Configure, make, install, and create an archive of mpich2:

```
mkdir /usr/local/mpich2/
cd mpich2-*
./configure --enable-f90 --prefix=/usr/local/mpich2 | tee
    configure.log
make | tee make.log
make install
cd /usr/local/mpich2
tar -cvzf mpich2binaries.tgz /usr/local/mpich2
mv mpich2binaries.tgz /NetCommon
```

2. On all C10.4-s: copy and extract the archive as root:

```
sudo su -
cp /NetCommon/mpich2binaries.tgz /
cd /
tar -xvzf mpich2binaries.tgz
rm mpich2binaries.tgz
```

Add the binaries into the local search path similarly to that in Step 3 of Section 6, except use “/usr/local/mpich2/bin” as the path.

3. On the all computers (S10.4 and C10.4-s): open a terminal window as admin. Create a file `.mpd.conf` in `~admin` (e.g., `emacs ~/.mpd.conf`); alternatively you can create the file: `/etc/mpd.conf` containing:


```
secretword=my_secret_word
```

make sure `.mpd.conf` is hidden from others (otherwise `mpich` will not start up): `chmod 600 .mpd.conf`. You can try the local installation with the following commands: `mpd & mpdtrace; mpdallexit`
4. On the S10.4: open a terminal window as admin, and create a `“mpd.hosts”` file (e.g., `emacs mpd.hosts`) with each line containing a name of a node. For example in our infrastructure, the `mpd.hosts` is:


```
mamac-srvr00.local
mamac01.local
mamac02.local
mamac03.local
```

A hint: put computers first in this list that have more than one CPU, as MPICH2 will establish a ring, and it is more likely then to run processes on machines with more CPUs.
5. (S10.4) If you have not done so, follow Step 5 of Section 6.
6. (C10.4) If you have not done so, follow Step 6 of Section 6.
7. (S10.4) If you have not done so, follow Step 7 of Section 6.
8. On the S10.4: you can now start up the ring by opening a terminal window as admin and typing: `“mpdboot -n 4”`. You should replace the `“4”` by the number of entries in `“mpd.hosts”` as this will determine how many computers are involved in the MPICH2 ring.
9. On the S10.4: `“mpdtrace”` and/or `“mpdringtest 10”` will test the MPICH2 MPI ring; `“mpdallexit”` will tear down the ring. Place processes below the `“/NetCommon”` directory and run them using:


```
/usr/local/mpich2/bin/mpiexec -n nr_proc /NetCommon/proc_name
```

If the S10.4 computer has been rebooted, you will need to repeat from Step 7 to get MPICH2 working again.

To run a simulator like FDS (Fire Dynamics Simulator) [15], the executables need to be placed into a common directory (e.g., `/NetCommon`). The following sample command will start off an FDS example with five meshes on five CPUs:

```
usr/local/mpich2/bin/mpiexec -np 5 /NetCommon/FDS/fds5_osx_intel_mpi
/NetCommon/FDS/Examples/Verification/hallways.fds
```

8. Setting up Open Directory (LDAP) Users

This step is not really necessary for the Xgrid, OpenMPI, LAM-MPI setup however it will enable us to “share” users and their homes between machines with the one caveat that the home directories of users are not statically mounted but dynamically when they log in at the computer terminal (thus we could not use this feature with LAM-MPI). We are using reference [2] for this section.

8.1. Open Directory Master on S10.4

- Click on “Server Admin” (and make sure that the arrow on the left of the server name points downwards).
- Click on “Open Directory”
- Click on “Settings”
- Click on “General”, “Role” should be “Open Directory Master”
- Click on “Protocols”
- LDAP settings should remain untouched, i.e., “search base” should be “dc=mmac-srvr00” (or whatever your server’s name is. Database should be: “/var/db/openldap/openldap-data”. Set “policies” and “replicas” as needed (or ignore).
- Start or restart Open Directory services.
- Click “Overview”, you should see “lookup”, “LDAP”, “password” “running”, “Netinfo” being “local” and “Kerberos” possibly “stopped” (unless you have enabled it).
- Make sure Apple Filing Protocol (AFP) is running (green disc next to its name)
- Open a terminal window and create a common user directory */NetUsers*:

```
sudo mkdir /NetUsers
sudo chmod 775 /NetUsers
```
- Open “Workgroup Manager”, click on “Sharing” then on “All”. Click on “MacintoshHD”, then on “NetUsers”. Under the “General” settings, check “Share this item and its contents”.

Adding users:

- Open “Workgroup Manager” and log in as “diradmin”. Click “New User”. Enter User’s name and password into the appropriate boxes. Click on “Home” and then on “+”; Share point should be: “*afp://mmac-srvr00.local/NetUsers*”; Path should be the user name you have picked; Home should be: “/NetUsers/username” substituting username for the user name you have picked. Click on “OK”, “Create home now” and “Save”.
- Open terminal window and type:

```
sudo mkdir /Netusers/username
sudo chown username /NetUsers/username
```

substituting “username” for the user name you have picked.

8.2. Open Directory Clients on C10.4

- Open “System preferences” and click on accounts (open lock if necessary). Under login options uncheck “automatic login” and enable fast user switching (for convenience).
- Go to “Finder” and click on “Applications→Utilities→Directory Access” (open lock if necessary). Disable “SLP”, “SMB”, “Active Directory”, “BSD”, and “Netinfo”; enable: “LDAP”, “Apple talk”, and “Bonjour”. Configure LDAP, show options (but do not add DHCP). Click “New” and use for “authentication” and “contacts”; enter server name: “mmac-srvr00.local”, click on Continue and on “OK”.

- Under “Authentication”, now a “custom path” should be showing with a directory domain: “LDAPv3/mmac-srvr00.local”. Click “Apply” and enter password.
- Restart the computer.

9. References

- [1] Apple, inc., “Introduction to MPI distributed Programming on MAC OS X,” <http://developer.apple.com/hardwaredrivers/hpc/mpionmacosx.html>
- [2] Apple, inc., “Open Directory Server Admin 10.4 Help,” <http://docs.info.apple.com/article.html?path=ServerAdmin/10.4/en/c5od.html>
- [3] Arunchandar Vasan, “SSH Without Password: Micro HOW-TO”, <http://www.cs.umd.edu/~arun/misc/ssh.html>
- [4] Drew McCormack, “Distributed Tiger: Xgrid Comes of Age,” <http://www.macdevcenter.com/pub/a/mac/2005/08/23/xgrid.html?page=1>
- [5] Drew McCormack, “Getting Started with OpenMPI and Xgrid,” http://www.macresearch.org/getting_started_with_openmpi_and_xgrid
- [6] “LAM-MPI Parallel Computing,” <http://www.lam-mpi.org/>
- [7] MacTechNotes, “MAC OS X as an NFS Server,” <http://mactechnotes.blogspot.com/2005/09/mac-os-x-as-nfs-server.html>
- [8] “MPICH2 Home Page,” <http://www-unix.mcs.anl.gov/mpi/mpich2/>
- [9] “Open MPI: Open Source High Performance Computing,” <http://www.open-mpi.org/>
- [10] “Using automount on MAC OS X,” <http://sial.org/howto/osx/automount/>
- [11] http://www.lrr.in.tum.de/archiv/smile/tutorial/ss2001/smi_ex.tgz
- [12] <http://crystal.uta.edu/~zaruba/pingpong.tgz>
- [13] <http://gcc.gnu.org/wiki/GFortranBinaries>
- [14] <http://www.lam-mpi.org>
- [15] NIST, “Fire Dynamics Simulator” <http://www.fds-smv.net/>