

“Amdahl” Cluster Quick Start

Preston M. Smith
psmith@physics.purdue.edu

September 16, 2002

1 Introduction

This document is intended as a “Quick Start” manual for a user wanting to get a computation running quickly on the “Amdahl” Linux Cluster.

It is assumed that a user reading this document is familiar with UNIX compilers (`cc`, `f77`, `f90`) and possesses basic UNIX skills. (Can navigate UNIX filesystems, edit files, run programs, etc.)

This document will provide a step-by-step guide to compiling and running a computation on the cluster. Examples will involve an example C program (`program.c`), a Fortran 77 program (`program.f`), a Fortran 90 program (`program.f90`), their input data (`inputdata.dat`), and a small subset of test data (`testdata.dat`).

2 Accessing the Cluster

In order to access the cluster, a user should contact PCN at `staff@physics.purdue.edu`, and request that their account be activated on the Amdahl cluster. The account will have the same username and password as regular PCN accounts, such as on *bohr*, *curie*, or *heisenberg*.

When your account is enabled on *amdahl*, log into the system with `ssh`. A new user will quickly notice that their home directory is empty! As the cluster systems are connected to a private network, it has home directories that are separate from those that exist for your usual account.

Now that you are logged in, you need to prepare your program for compilation and execution. If, for example, your program is in your *bohr* home directory, you can copy it to your cluster home directory easily:

```
cp /Home/username/program.f /home/username
```

This will copy the file `program.f` from your PCN home directory (Available on the master node as `/Home/username`, with a capital H), to the cluster home directory (`/home/username`). You could repeat similar commands to copy `program.c`, `program.f90`, `testdata.dat`, and `inputdata.dat` into your cluster home directory.

3 Compile and test your programs

Now that your programs are in your home directory, they are ready for compilation.

3.1 GNU Compilers

First, with the GNU compilers:

```
gcc -o program program.c
```

```
g77 -o program program.f
```

will compile `program.c` and `program.f` into “program”. The source for `program.f90` cannot be compiled with the GNU compilers, as GNU does not offer a Fortran 90 compiler.

Options for optimizing programs with GNU compilers can be detailed in the `gcc(1)` and `g77(1)` manual pages.

3.2 Portland Group Compilers

Alternatively, the *amdahl* cluster has commercial compilers installed from the Portland Group. The Portland compilers offer increased optimizability over GCC 2.95, as well as a Fortran 90 compiler.

```
pgcc -o program program.c
```

```
pgf77 -o program program.f
```

```
pgf90 -o program program.f90
```

Optimization information and general compiler information can be found on the WWW at <http://amdahl.physics.purdue.edu/pgi/>.

3.3 Run and Test

Now, you can try and run a test of your program. Take a small subset of your input data, and use it as input for your program to verify your program’s execution.

```
./program < testdata.dat.
```

If the program successfully runs, you can use a program like “time” to test optimization settings:

- Run the program, with `time ./program < testdata.dat`
- Recompile the program, with new optimization settings.
- Re-run the program, with “time”, to see if the optimization settings improved execution.

If the program doesn’t run, (if it dumps core or otherwise exits irregularly), you can use “gdb” or “ddd” to debug it.

4 Submit Your Job For Execution

Now that your program is compiled and tested, you can submit it to the cluster for execution. Job submission and scheduling is handled by PBS (Portable Batch System), which is a

commonly used system in place at many high-performance computing sites, as well as on Purdue's IBM SP.

First, create a file that will be used as a batch script. In this example, we will call it `job.pbs`.

Place the following lines into "`job.pbs`", substituting an appropriate name for "`jobname`" and the name of your executable for "`program`":

```
#PBS -N jobname # Set jobname
#PBS -e jobname.err
#PBS -o jobname.log # Set filenames to log standard error and out
#PBS -m abe
#PBS -M username@physics.purdue.edu
### Email this username when the job starts, ends, or aborts
#PBS -l nodes=1 #PBS -l walltime=02:00:00
### Request one node, for two hours
#PBS -W qos=1

### Run the program
time ./program < inputdata.dat
### END OF PBS SCRIPT
```

Save your file, and submit it to the scheduling system with the command:
`qsub job.pbs`

5 Get the Results

Now just wait for your computation to complete!

The cluster will email you upon completion. When the job is done, you can examine the `jobname.err` and `jobname.log` files specified with `-e` and `-o` for output generated by your program.

6 For More Information

Now that you've submitted your first computation to the cluster, you may want to do more advanced things. Information on advanced tricks with PBS, running parallel programs, and the complete PCN cluster user's manual can be found at <http://amdahl.physics.purdue.edu>.

If you would like to do something not covered in this quick-start guide or the user manual, please contact PCN staff at staff@physics.purdue.edu.