Cambridge Cranfield High Performance Computing Facility
Introductory User Guide

Version: 1.3

April 26, 2004

Contents

1 Introduction

The Cambridge Cranfield High Performance Computing Facility (CCHPCF) began in the early 90's by the efforts of a consortium of scientists from different departments who found themselves unable to obtain sufficient computing resources either locally or nationally.

The consortium was successful in obtaining financial support from Cambridge University, the Newton Trust and grant applications including bids to JREI. The CCHPCF is not a computing service but is a facility that is maintained by users for users to supply flexible high performance computing resources to researchers in both Universities and their teams. It is not intended to service jobs that could be run on departmental workstations.

1.1 About this Guide

This guide gives a basic overview of the CCHPCF systems and covers topics such as logging into the machines and submitting jobs. It is assumed that you have experience with simple Unix commands and access to precompiled binaries suitable for the CCHPCF machines. The University Computing Service runs regular courses on basic use of Unix. (see http://www.cam.ac.uk/cs/)

Information on compilation, debugging and tuning can be found at http://www.hpcf.cam.ac.uk.

1.2 Code of Conduct

The CCHPCF machines are multi-user systems. Although we try to set them up in such a way that everyone has fair access to the resources available, we depend to a very large extent on the goodwill and cooperation of all the users.

- The CCHPCF is intended to be a very open system, with users helping each other and checking for accidents and inefficiencies. If you notice a problem with the system you will probably need to do the initial checking yourself. However report all problems to support@hpcf.cam.ac.uk
- Most files will be readable to all users. Please use this appropriately, and respect the confidentiality of other people's files and the work that they are doing.
- Users are encouraged to keep well within their disk quota on their homespace and to avoid filling the temporary disks.
- We suggest that users limit themselves to no more than five simultaneous jobs on each machine (either running or queued, but excluding the shorter test and development queues). Users who need to submit significantly more than this should discuss their

requirements with HPCF support (Email: `support@hpcf.cam.ac.uk`). It is reasonable to submit a few more before a long break (e.g. Christmas), but it is essential to check that you are not obstructing other people from using the system.

- However you work, please do **NOT** submit many jobs to the test and development queues, have a job on those queues that automatically submits its successor, or have a script that submits multiple jobs to them. Those queues are for testing and development, and must not be used to run production jobs ahead of ones on the production queues.

We remind you of the undertaking you have given when signing up to use the facility:

I have read the Rules and understand that allocations of computing resources are made and may only be used subject to the Rules issued from time to time by the University of Cambridge Information Technology Syndicate and the Management Committee for the Cambridge-Cranfield HPCF. I agree to abide by such Rules. I understand that National and International networks may be used for academic purposes only.

Data Protection Act: I will not hold or process personal data without first seeking permission from the Management Committee for the Cambridge Cranfield HPCF. Please note that and infringement of this undertaking would lead to disbarment from using the CCHPCF.

1.3 Publications

It is important that we keep a record of all publications arising from work carried out on the CCHPCF. Please send details of any relavent publications to the CCHPCF administrator email admin@hpcf.cam.ac.uk. When apropriate the CCHPCF should be acknowledged in any published work, eg *Computational resources were provided by the Cambridge-Cranfield High Performance Computing Facility.*

2 Overview of Machines

The Cambridge-Cranfield HPCF consists of two supercomputers. Franklin, a 972 processor Sun and Hartree, a 72 processor IBM SP2-Power4.

Franklin, the Cambridge-Cranfield HPCF Sun Galaxy-class configuration supercomputer is based on the SunFire F15K node, nine of which are installed at Cambridge with 100 CPUs and one at Cranfield with 72 CPUs (UltraSPARC III).

Each node has 288 GBytes of main memory. The CPUs are 1.20 GHz Ultra Sparc III with a peak of 2.4 GFlops. Each node is a shared memory system (SMP) allowing each CPU access to all the available memory on the node using fast intranode fabric. The

nodes are connected by the new SunFire Link interconnect which enables system to scale exceptionally well up to eight nodes. The Cambridge systems will initially be linked in 3 groups of three with the new FireLink technology. So the maximum job size supported initially is 24 hours at a realistic performance of 300 GFlops with 150 GB/sec intranode memory bandwidth communicating with 800 GBytes of memory. The available disk space is about 3 TBytes.

Hartree is an IBM SP parallel supercomputer which was commissioned in 2001. It has 8 computation nodes and 2 I/O nodes (mainly for system processes and interactive use.) Each computation node contains 8 Power4 processors running at 1.1GHz (peak 4.4GFlops), and 16GB of memory. Each I/O node has 4 Power3 processor running at 375MHz and 8Gb of memory. The nodes are connected via IBM's high performance SP Switch2.

3 Getting Started

3.1 Help and Advice

The CCHPCF webpages http://www.hpcf.cam.ac.uk contain the latest version of this document as well as pointers to more detailed information.

If you have a problem with the CCHPCF machines the first person to ask is probably another user in your department / research group.

Failing this, please direct any questions to the system managers, email: support@hpcf.cam.ac.uk.

The operation of the CCHPCF is managed by the Management Committee. Each department has a representative and users are encouraged to pass on comments and suggestions. Contact details for department reps can be found on the CCHPCF webpages.

3.2 Getting an Account

Resources on the computers are allocated to projects, each of which has a named project leader. Application forms to add a new user to an existing project, or to set up a new project, can be downloaded from the CCHPCF website (www.hpcf.cam.ac.uk). Forms can also be obtained from the director of the CCHPCF

```
Dr. Ron Horgan
CMS
Wilberforce Road
Cambridge
CB3 0WA
```

Telephone (01223) 337839
E-mail: r.r.horgan@damtp.cam.ac.uk

3.3 Logging in

The names of the machines are

- **franklin.hpcf.cam.ac.uk** - Sun
- **hartree.hpcf.cam.ac.uk** - IBM

Users can log into a machine using ssh. E.g.

```
[my_workstation] ssh  spqr1@franklin.hpcf.cam.ac.uk
spqr1@franklin.hpcf.cam.ac.uk's password:
Last login: Sat Feb 22 17:24:29 2003 from virgo.cus.cam.a
franklin-1 [1]
```

3.4 Shells

The standard shells are available (sh, csh, tcsh, ksh, bash). The command `chsh` can be used to change your default shell.

```
franklin-1[1] chsh
The supported shells are sh, csh, bash, ksh and tcsh; please type the
name of the one you want as your login shell:
```

3.5 File Space

Your home directory is visible on all machines. It physically exists on a fileserver called perutz. An overview of the CCHPCF network is shown in Figure 1.

There are large temporary (scratch) filespaces attached to each machine, symbolic links in your home directory point to these (franklin-tmp, hartree-tmp). Access to franklin-tmp from franklin is much faster than accessing your own home directory from franklin. Thus jobs on franklin requiring significant file I/O should first copy the files they need to franklin-tmp and then run there. The same applies to Hartree.

The temporary filespace is not backed up. Files which have not been accessed for two months are automatically deleted. Users should remember to delete large unwanted files periodically and copy important files to their homespace or local workstation.

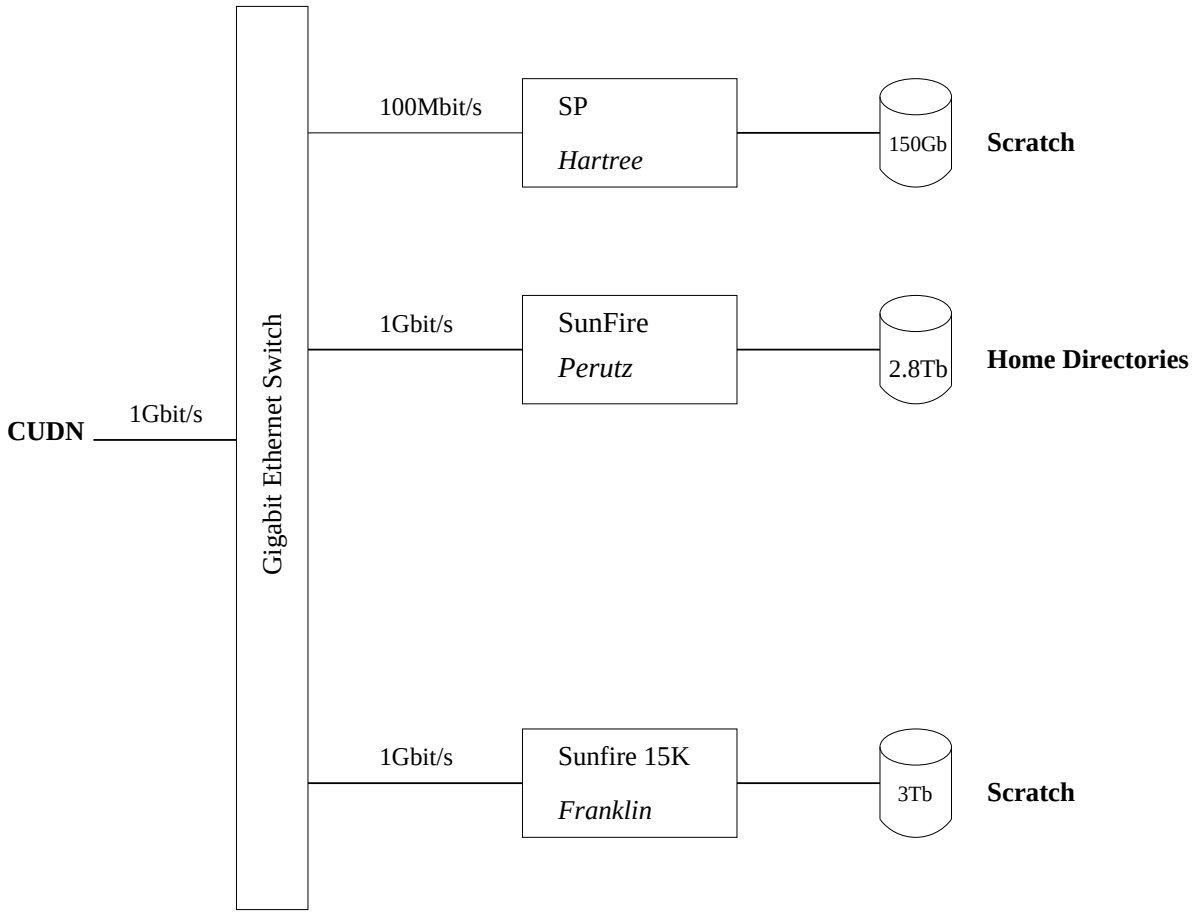

Figure 1: CCHPCF Network (Schematic)

3.6 File Transfer

The recommended means of transferring files into and out of the CCHPCF is `scp`. Eg.

```
my_workstation[1] scp file.dat spqr1@franklin.hpcf.cam.ac.uk:
Warning: Permanently added 'franklin.hpcf.cam.ac.uk'
(DSA) to the list of known hosts.
sprq1@franklin.hpcf.cam.ac.uk's password:
file.dat    100\% |**************************| 15452       00:00
my_workstation[2]
```

3.7 Editors

There are two editors available on the machines: Emacs and vi.

4 Running Jobs

The CCHPCF machines are intended primarily for processing batch jobs. However each system can run small jobs interactively for testing purposes.

The following sections outline the batch queuing systems for each machine and give the recommended procedures for job submission. Please do not use any features not documented here. (doing so might cause your jobs not to run correctly, or worse, interfere with other users' jobs). If you think you need a facility not documented here please contact support@hpcf.cam.ac.uk. It is not intended to disable features for which there is a genuine academic requirement.

5 Franklin: Running Jobs

When logging into Franklin you will actually get connected to the front-end machine, franklin-1. The frontend can be used for short interactive jobs and compilation. The frontend has 8 UltraSPARC IIICu 900MHz processors and 16Gb memory. Batch jobs are submitted to the queuing system and processed on the computation nodes. There should be no need to log into the computation nodes directly.

5.1 Interactive Usage

Small test jobs, both serial and parallel, can be run on the front-end machine. There is a five minute limit on interactive jobs. To run an MPI job on 2 CPU's in the background, whilst redirecting output to the file out.dat, use;

```
franklin-1> mprun -n -np 4 ./mycode.mpi > out.dat &
franklin-1>
```

5.2 Batch Jobs

Franklin uses the Sun Grid Engine Batch queuing system. There are queues $s < n >$, $t < n >$ and $u < n >$, where $< n >$ is the number of processors and may be 4, 6, 8, 12, 24, 32, 48, 64, 84 or 96. The "s" queues have a 24 hour real-time limit, the "t" ones a 2 hour limit and the "u" ones a 10 minute limit. (The "s" queues are for production runs, "t" and "u" are for testing and development)

Generally, the s96 logical queue is the best one for production work, but users with heavily memory- and communication-bound jobs may get better throughput out of the s64 logical queue.

The actual number of CPUs allocated one of 6, 12, 24, 48 or 96. Logical queues of other sizes will reserve 6*N CPU's but execute on a subset of these CPU's. For example a job in the t8 queue will reserve 12 CPU's. It should be noted that your resource usage is calculated from the number of CPU's reserved, so a 2hour 64 CPU job will "cost" the same as a 2 hour 96 CPU job (as it is not possible to use the remaining 28 CPU's).

If your program is CPU bound, you should use the maximum number of CPU's (ie a queue of size 6*N). The logical queues with 4, 8, 16, 32 and 64 CPUs use only the 'main' CPUs and may provide 50% more memory bandwidth and memory per CPU; if your program is memory bound, you should use a queue of size 4*N. If in doubt, try both and use the one with the smaller wall-clock time, or ask for advice.

The x84 logical queues are mainly for people doing performance analysis, as they have exactly the same number of CPUs on each board, where the x96 ones (and, to some extent, the x64 ones) don't.

To submit a job, type `qsub -Q queue test.job` on one of the frontend machines, where the test.job may be a script or executable. It will be scheduled and run in the directory (and most of the environment) that it was submitted from and NOT as if it were a fresh login. Note that the `-Q` option is a local feature. To run a program using MPI on 12 processors a suitable job submission script is

```
franklin-1 [1] cat test.job
#!/bin/sh
date
mpirun -np 12 myprog.mpi
```

and this can be submitted to the 24hr queue with

```
franklin [2] qsub -Q s12 job.go
your job 2352 ("job.go") has been submitted
```

The command `qstatu` can be used to monitor the status of the job

```
franklin [3] qstatu
job-ID prior name       user    state  submit/start at    queue  master
-----------------------------------------------------------------------
2352    0   job.go      spqr1    r    03/15/2003 15:52:42 b-s96  MASTER
```

The state of a job is usually one of (r)unning, (q)ueued, (d)eleted or (t)hreaded (ie about to be executed).

The command `qstatj job-ID` gives more information about the state of a particular job. The `qstat` command also gives detailed information, consult its man page for more details.

To delete a job use the `qdel` command

```
franklin [4] qdel 2352
spqr1 has deleted job 2352
```

Both stdout and stderr are returned in files such as `job.go.o2352` and `job.go.e2352`.

5.3 Job Scheduling and Priority Access

Franklin runs Sun's Grid Engine queuing system. Grid Engine schedules jobs in an attempt to share the available resources between different users and projects. The command `qstats` provides information on job scheduling and a user's resources.

```
franklin-1> qstats
The pts is the percentage share of the resources the project should get
The pas is the percentage share of the resources the project has had
The usage is a time decayed past cpu usage of the user
JID   user   project  state sub/start at shares  Q pts   pas    usage
2611  qwer1   hpcf2-pay r   1/3  13:18:24 2470 s96 0.08  6.99  4154154
2583  sprq1   hpcf2     r  29/2  15:02:27    5 s96 1.42 25.96  1244804
2498  asdf2   hpcf10    qw 20/2  17:24:29  500 s96 0.19  2.42  1244804
2614  spqr1   hpcf2     qw  1/3  13:57:05  476 s96 1.42 25.96  6951406
```

Projects with the suffix "pay" are granted a larger share of the computing resources under the priority access scheme (generally these are projects which have contributed towards the running costs of the CCHPCF). Users should contact their project leader for more details.

Users who belong to the priority access scheme may use the local options to `qsub`, `-priority` or `-nopriority` to select or deselect the priority mechanism for their jobs. The default will usually be "priority". As an example to submit a priority job to the 96 processor production queue;

```
franklin [2] qsub -priority -Q s96 job.go
your job 2353 ("job.go") has been submitted
```

Users can check their monthly charged and uncharged usage using the command `myusage`

```
franklin [3]  myusage
Charged         76987.2 CPUh Pounds:10778.21
```

The -l option to myusage will report the usage from the previous month,

```
franklin [3]  myusage -l
Last month
Not Charged      5719.9 CPUh
```

There is further information for project leaders on the CCHPCF website.

5.4 Compilation

We give a quick example of fortran programming with MPI. Further details will be found on the CCHPCF website.

On the CCHPCF, the relevant commands have been surrounded by wrapper scripts, so that the default is to provide a reasonable set of defaults for the CCHPCF systems and some detection of potential pitfalls. This is optional, and can be disabled entirely, but you are advised not to do that unless you know what you are doing.

By default the environment variable HPCF_MODE is set to "yes". This will set a good level of optimisation and link to the Sun Performance Libraries, which contain optimised versions of BLAS, LAPACK, FFT routines etc.

To compile an MPI program you should set the environment variable HPCF_MPI to "yes". For (ba)sh

```
franklin-1> export HPCF_MPI=yes
```

This will set up the correct paths and libraries. A complete example of compiling a Fortran MPI program is given below:

```
franklin-1> cat hello.f90
program hello

  include 'mpif.h'
  integer npe,mype,ierr

  call mpi_init(ierr)
  if (ierr.ne.0) stop 'MPI initialisation error'

  call mpi_comm_rank(mpi_comm_world, mype, ierr)
  call mpi_comm_size(mpi_comm_world, npe, ierr)
```

```
  write(*,101) mype,npe
101 format(' Hello parallel world, I am process ',I3,' out of ',I3)

  call mpi_finalize(ierr)
end program hello

franklin-1> export HPCF_MPI=yes
franklin-1> mpf90 hello.f90
franklin-1>
franklin-1> mprun -np 4 ./a.out
 Hello parallel world, I am process   3 out of   4
 Hello parallel world, I am process   1 out of   4
 Hello parallel world, I am process   2 out of   4
 Hello parallel world, I am process   0 out of   4
franklin-1>
```

6 Hartree: Running Jobs

When logging into Hartree you will actually get connected to one of the I/O nodes, hartree_a. The I/O nodes can be used for short interactive jobs and compilation. Batch jobs are submitted to the queuing system and processed on the computation nodes. There should be no need to log into the computation nodes directly.

6.1 Interactive Usage

Short serial jobs can be run on the I/O nodes. To run an MPI exectuable for testing or debugging, use the command `poe` eg

```
hartree_a [1] poe prog.x -procs 2
```

Your current directory must contain a `host.list` file for node allocation, with as many lines as processors requested eg,

```
hartree_a [2]cat host.list
hartree_a
hartree_a
```

Note that executables compiled with certain Power4 optimisations may not run interactively on the I/O nodes. See the CCHPCF webpages on compilation for more details

6.2 Batch Jobs

On Hartree the LoadLeveller scheduler is used. The queues are named $[u \mid t \mid s][2 \mid 4 \mid 8 \mid 16 \mid 32]$: `u` for 10 minutes, `t` for 2 hours, and `s` for 12 hours. (The "s" queues are for production runs, "t" and "u" are for testing and development). All queues of 16 or more processors have exclusive use of the node(s) they run on. A single node contains eight 1.1GHz Power4 processors and 16GB of shared memory.

For communication intensive codes (eg any FFT based codes) it is recommended to use the 8 processor queues. Jobs requiring larger resources should use Franklin

To run an MPI program on 8 processors in the 2hr queue a suitable job submission script is

```
hartree_a [6] cat test.job
#!/bin/sh
#@ class = t8
#@queue

date
./myprog.mpi
```

Note that the mpriun command is not used. The first three lines are required by the Loadleveller scheduler, only the class should be altered to reflect the queue you wish to submit to.

This script can be submitted with

```
hartree_a [7] llsubmit test.job
stderr will be sent to LoadL.err.$(jobid).$(stepid)
stdout will be sent to LoadL.out.$(jobid).$(stepid)
llsubmit: Processed .. Submit Filter: "/var/loadl/home/prefilter".
llsubmit: The job "hartree_a.823" has been submitted.
```

The llq command is used to monitor the status of the job.

```
hartree_a [8] llq
Id                   Owner   Submitted   ST PRI Class  Running On
-------------------- ------- ----------- -- --- ------ -----------
hartree_e.14120.0    spqr1    2/23 10:43 ST 50  t8    hartree_b
```

The llcancel command is used to delete jobs.

```
hartree_a [9] llcancel  hartree_e.14120.0
llcancel: Cancel command has been sent to the central manager.
```

Feedback

We would be grateful for any comments, corrections and suggestions for improvements.
Email: webmaster@hpcf.cam.ac.uk