

High Throughput Computing at the IPPP

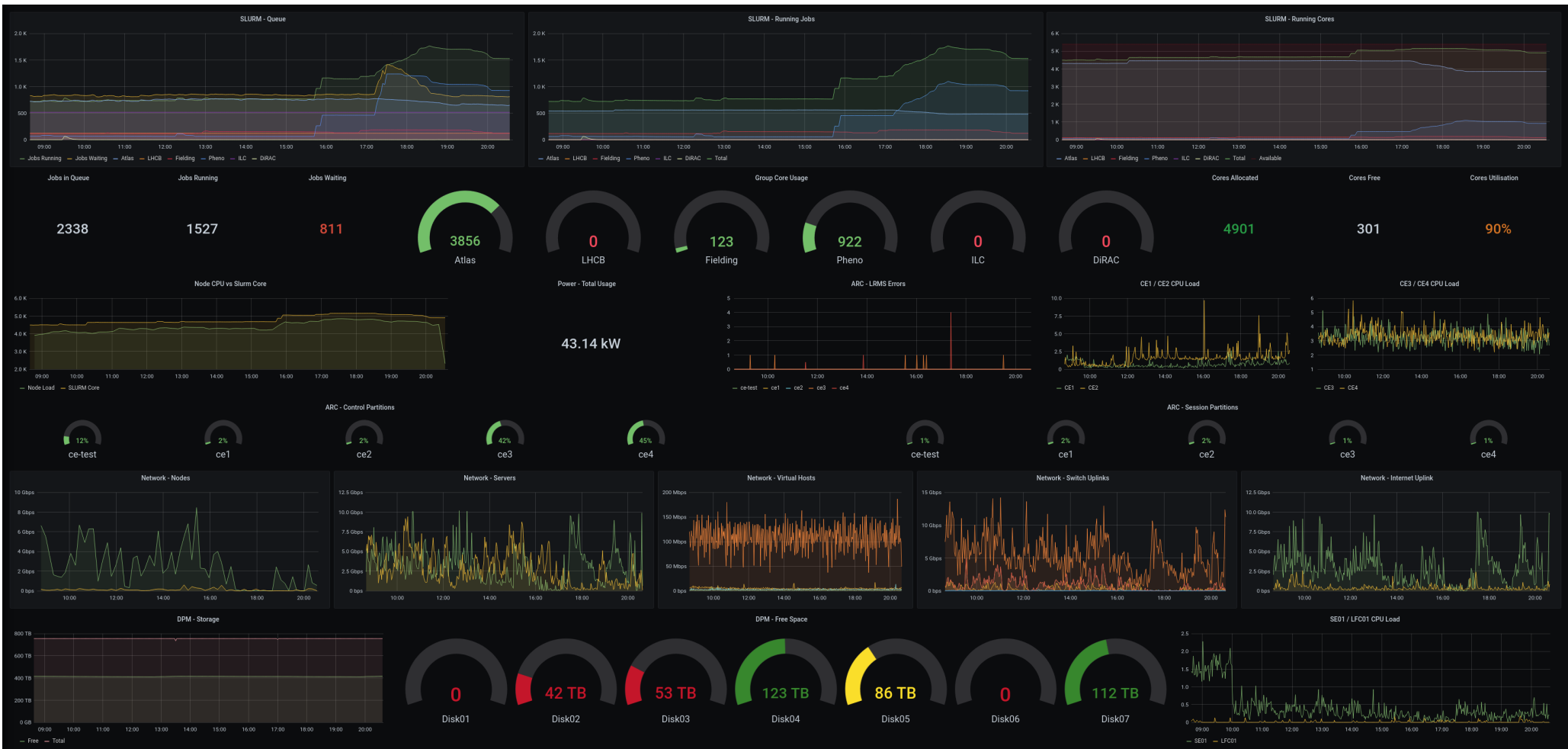
An overview of the resources available* and how to access them.

***) Technically, not just at the IPPP but also elsewhere in the UK, and still available to users at the IPPP.**

Resources available

- **GridPP gives us access to 20 times more CPU power than what we have in Durham. Which is already a lot (>5000 job slots).**
- **GridPP is a smart way for us to gain access to resources elsewhere when we need them, and pay back by allowing ATLAS etc. to run in Durham, when we do not use the CPU**
- **The Durham cluster currently has >5000 job slots (“cores”) and roughly 900TB disk [and has yet more nodes ready to be inserted into queue was the water cooling has been plumbed in]**
- **Special Grid protocols needed for the high-throughput computing**

<https://monitoring.dur.scotgrid.ac.uk/>



The Local Durham setup

Our HTC consists of >170 compute nodes each of 2 CPUs of up to 22 physical cores, and ~3Gb of memory per job slot. Local disks of a few TBs. The cores are further hyperthreaded, but not fully loaded, in order to keep a HEPSPC06 of roughly 13.

Quite typical for grid computing (although we in Durham prioritise fast CPU).

Each node has 2 bonded 1Gbit network connections; the disk servers are on 2x10Gbit links. The one grid Storage Element acts as a front-end to many disk-servers, and automatically balances to load on these, such that it can sustain transfer rates of up to 10 Gbytes/sec.

The network connection to the wider Grid is through a dedicated 10Gbit line. This means that the local SE can be used for disk storage even when jobs are run elsewhere.

Guide for getting you on the grid

Information available at:

<http://ipp.dur.ac.uk/~andersen/GridTutorial/gridtutorial.html>

- **Grid Certificate**
- **Job submission with ARC**
- **Consider resources**
- **Grid storage**
- **CVMFS**

Obtaining or Renewing Grid Certificates

- **A certificate is needed to authenticate yourself when requesting resources for CPU, disk etc.**
- **The certificate is personal, and valid for a year. The renewal process is quick (~1 day), but checks that you are still eligible.**
- **After obtaining and installing a certificate, you can apply for membership at the Virtual Organisation 'Pheno', which is how we as users are known to the rest of the Grid.**
- **Certificates are private. All that goes on in the Grid is monitored and accounted for. When problems are caused, they can be traced to individual certificates.**

Obtaining or Renewing Grid Certificates

<https://portal.ca.grid-support.ac.uk/caportal/>. Further details on what to do at <https://www.ippp.dur.ac.uk/~andersen/GridTutorial/>

Welcome to the UK Certification Authority Portal

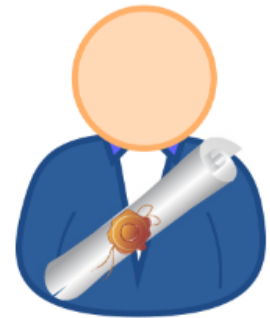


I am a new user

Request New User Certificate

Go To CA Helpdesk

Find My Local RA



I have a grid certificate

Download a Certificate

Login / View My Certificate

Request New Host Certificate

Renew Certificate

Accessing the Grid Resources

- **The Grid resources are accessed through the “Grid User Interfaces”, `gridui1.dur.scotgrid.ac.uk` and `gridui2`. These are machines with the same software environment as that on the compute nodes on the Grid, but with normal user accounts.**
- **Each `gridui` is similar to a single node in the cluster. The GridUI are the “submission interface” to the Grid. The cluster and the `gridUIs` are separate, the GridUIs can be used for submission also to other clusters.**
- **Jobs running on the cluster are disconnected from the user account on the `gridui`.**
- **Technically, the basic environment is dictated by the LHC experiments.**
- **It is of course possible to install another set of compiler, libraries etc. for your jobs - this job can be performed e.g. using a docker container of `cern/cc7-base`**

Working with certificates

- Each job and grid file access needs to be authenticated to verify the rights to get CPU time and disk space.
- Instead of submitting the certificate, the grid authentications work with “proxy certificates”, which are valid for just 12 hours
- Generated with the command
`arcproxy -S pheno`
- It is possible to setup automatic renewals of proxy certificates for longer running jobs (see website). Maximum run-time is one week.

Job description files

Instead of e.g. slurm job scripts, the grid operates with e.g. a Resource Specification Language. Example:

&

(executable = "simple")

(arguments = "input.txt")

(jobName="TestJob")

(inputFiles = ("input.txt" ""))

(outputFiles = ("output.txt" ""))

(stdout = "stdout")

(stderr = "stderr")

(gmlog="testjob.log")

(walltime="240")

(count="1")

(countpernode="1")

Submitting a job

Submit a job to the Durham “compute element”:

```
arcsub --direct -c ce1.dur.scotgrid.ac.uk submit.xrsl
```

Submit to the RAL “compute element”:

```
arcsub -c arc-ce01.gridpp.rl.ac.uk submit.xrsl
```

The ARC information system takes a few minutes to update. List of jobs:

```
arcstat
```

Get the result of the run (gets all job information, including input, outputs, errors):

```
arcget <jobid>
```

Kill all jobs:

```
arckill -a
```

Print output:

```
arccat <jobid>
```

List of compute elements available (requires a proxy certificate):

```
lcg-infosites ce --vo pheno
```

Requesting Multiple Cores for Jobs

**It is possible to request multiple cores for a job by including
(count="N")**

(countpernode="M")

in the job.xrsl ($M \leq N$). Important considerations: Each node has a maximum of ~50 job slots. If one asks for 50 cores, then the job has to wait in the queue until one of the ~170 nodes has finished all of the currently running jobs. Potentially a week. If one asks for 13 cores, then most nodes have to be half empty - and the nodes are generally equally loaded, so effectively half the currently running jobs on the farm have to finish.

It is important to strike a balance between the speed-up gained by running on multiple cores, and the additional wait in the queue before the multiple cores become available. ATLAS currently request 8 cores, which seems a reasonable - generally our local farm needs ~400 free job slots until a 8-core job starts.

The queue

The backend queue operates on a fair-share basis, and every five minutes evaluates the priority of each job waiting to be executed. The priority is based on the historical CPU usage of the owner (calculated with an exponential decay of half time one day) and the time spent in the queue. The queue can be interrogated as e.g.

```
3:26pm griduil 2782 > sprio -o "%.7i %10u %.10Y %.10A %.10F %.10J %.10P %.10Q" | sort -k3nr | cut -c9-18 | uniq -c | head
 37 pheno019
   5 prdatlas03
   1 prdatlas00
   2 prdatlas03
   1 prdatlas02
  10 prdatlas00
   5 prdatlas03
   8 prdatlas02
   3 prdatlas03
   3 prdatlas02
```

The queue

The priority for jobs associated with individual users can be found as

```
3:28pm gridui1 2783 > sshare -a | grep -v " 0 " | egrep "fielding|pheno" | sort -k7n
fielding                2617      0.214315      89240955      0.116977      0.685004
pheno                    7379      0.604291      418186301     0.547933      0.533391
pheno                    pilpheno0+  100      0.008986      67845798      0.095726      0.000621
pheno                    pheno002    500      0.044929      127832877     0.195788      0.048774
pheno                    pheno056    500      0.044929      108197497     0.171972      0.070430
fielding                 jung        1797      0.017742      43635902      0.062128      0.088281
fielding                 hbarlow     2264      0.022353      43712443      0.063504      0.139562
pheno                    pheno019    500      0.044929      60339839      0.113925      0.172458
pheno                    pheno055    500      0.044929      41668858      0.091279      0.244577
pheno                    pheno005    500      0.044929      6083882       0.048118      0.475996
pheno                    pheno012    500      0.044929      2697398       0.044010      0.507136
pheno                    pheno070    500      0.044929      1765177       0.042879      0.516060
pheno                    pheno081    500      0.044929      1304709       0.042321      0.520526
pheno                    pheno031    500      0.044929      435770        0.041267      0.529059
pheno                    pheno082    500      0.044929      14487         0.040756      0.533246
pheno                    pheno099    500      0.044929      3             0.040738      0.533390
fielding                 suzanne     2264      0.022353      1892608       0.014422      0.639408
```

Grid Storage

The grid-connected storage is arranged differently from the usual transparent nfs-mounted drives on your desktop - nfs would not work on a large scale, and the grid protocols focus on load distribution and huge throughput

Ensure you have a directory set aside on the grid disk. You can make a directory for yourself with the command

```
gfal-mkdir
```

```
gsiftp://se01.dur.scotgrid.ac.uk/dpm/dur.scotgrid.ac.uk/home/pheno/  
MyName/
```

The last task of each job is copying the produced output (e.g. outputN.root) to the appropriate directory on the grid disk.

```
gfal-copy `pwd` /filename
```

```
gsiftp://se01.dur.scotgrid.ac.uk/dpm/dur.scotgrid.ac.uk/home/pheno/  
MyName/
```


Grid Storage

You can follow the progress in your web-browser by pointing it to e.g. <http://se01.dur.scotgrid.ac.uk/dpm/dur.scotgrid.ac.uk/home/pheno/MyName/>

The files can be copied from the grid disk with a command like

```
gfal-copy -r  
gsiftp://se01.dur.scotgrid.ac.uk/dpm/dur.scotgrid.ac.uk/home/pheno/  
MyName/OUTPUT file://`pwd`/gridoutput
```

You can also mount the directory directly (read-only) on your laptop or local computer by pointing the file-browser to

```
dav://se01.dur.scotgrid.ac.uk/dpm/dur.scotgrid.ac.uk/home/pheno/  
MyName/
```

Use of Resources

When moving to large-scale grid computing, it is important to consider the various stages in the life-time of a job, and in particular whether they scale, or whether the jobs will hinder each other, when several are started simultaneously on the same machine.

Start the job by copying necessary files from grid disks to local node. Finish the job by copying the required output to grid disk. This is much quicker than relying on downloading the sandbox outputs with the arc job description.

Consider resources as CPU, network and local disk access. Up to ~50 job slots on a node, so in order to reduce the time it takes to start a job, you should reduce the amount of local disk usage and of network traffic. It is no good having many processors, if the network or disk traffic creates a bottle neck.

If the jobs each share a long setup, consider whether multiple jobs can be grouped together (as in e.g. a 8-job submission).

Increase the amount of resources asked for by no more than a factor 10 at a time - 1 test job, 10, 100, 1000,... since new bottlenecks might present themselves at every step.

Use of resources

Be considerate to the servers. Every job leaves a footprint on the servers.

A cluster with 5.000 slots can easily take down any server system. Don't submit 4.000 jobs each taking a few seconds; aim for a run-time of each job of >30min.

CVMFS

The Cern Virtual Machine File System (CernVM-FS or CVMFS) is a read only HTTP based file system which is mounted on the grid UI and on the nodes. It is not meant for distributing data but files that you need to run your program. For example, C++ libraries your program depends on. The smart thing is that the files are cached directly on each node. So pulled only once, even if 1000 jobs are run.

Links to further information

<https://www.ippp.dur.ac.uk/~andersen/GridTutorial/gridtutorial.html>

<https://monitoring.dur.scotgrid.ac.uk/>

<https://www.gridpp.ac.uk/>