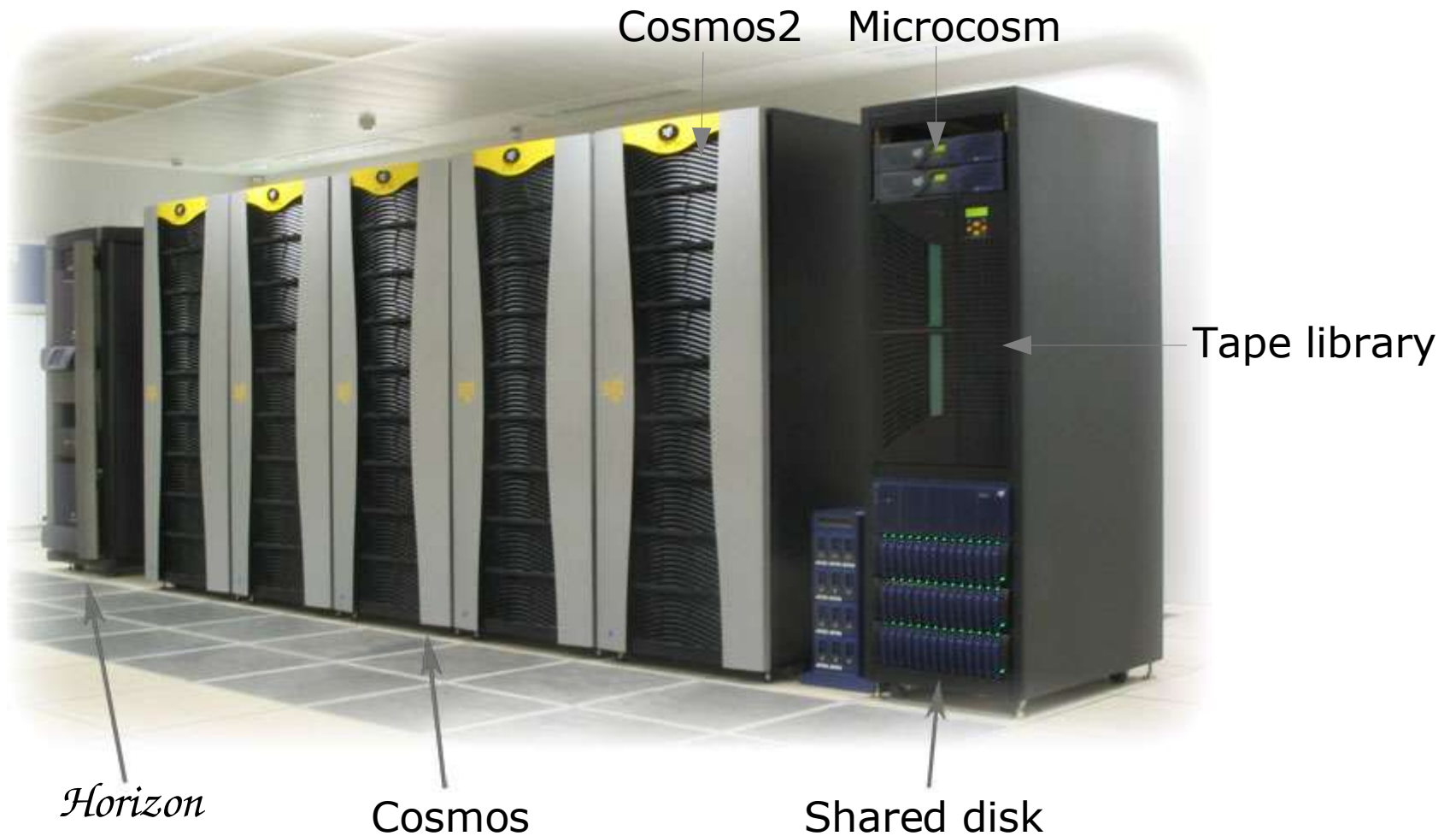


Using COSMOS



Contents

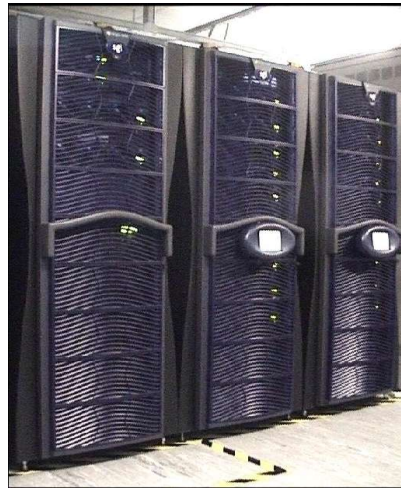
- History
- Web Documentation
- The Platform
- The Basics
- Job Queues
- Storage
- Common Problems
- Email Contact

History



I

IV



- Cosmos I (1997)
- Cosmos II (1998)
- Cosmos III (1999)
- Cosmos IV (2002)
- Cosmos V (2003)
- Cosmos VI (2004)



V

Web Documentation

<http://www.damtp.cam.ac.uk/cosmos/>

<http://www.damtp.cam.ac.uk/cosmos/news.html>

<http://www.damtp.cam.ac.uk/cosmos/User/>

<http://www.damtp.cam.ac.uk/cosmos/faq.html>

<http://www.damtp.cam.ac.uk/cosmos/docs/>

COSMOS – The Platform

- Supercomputer architecture
- Main features
- User's view
- General issues

The Super in Supercomputer

- Overall performance much better, and available memory much larger, than a typical single unit
 - implies parallel use of multiple units (cpus and chunks of memory)
 - for general computing, not just a specialised application
- Processing units are integrated in some way
 - co-operating to perform a **single task**
- Most today are MIMD
 - Many Instruction streams on Many Data streams
 - Vector processors and processor arrays are SIMD
 - single workstation/PC/unit is SISD

Cluster: An Overused Word

- Large numbers of components =>
 - physical extension and non-uniform latencies
 - proliferation of cpu-cpu and cpu-memory traffic
- For small cpu numbers, UMA/SMP may be economical
 - Uniform Memory Architecture/Symmetric MultiProcessor
 - requires expensive switch with designed capacity
 - expense grows rapidly with cpu number
- For large cpu numbers, economics => clustering of smaller SMP nodes via an interconnect
 - inter-node latency > intra-node latency, i.e. non-uniform
 - SGI Altix (ccNUMA) and "commodity cluster" (multicomputer) are both clusters in this sense

Main Features of COSMOS

- 152 Itanium2 (1.3GHz) cpus
- 152 GB memory
- = 76 x (2cpu, 2GB) SMP nodes + fast interconnect
- So memory is in physically distributed chunks
 - distributed memory programming available as in any cluster
 - access to local memory is quicker than to remote memory
- *Plus* special hardware providing:
 - single logical address space
 - cache coherency
 - single system image and shared-memory programming
 - cache-coherent Non-Uniform Memory Architecture (ccNUMA)

User's View

- One monolithic system
 - looks like a large SMP workstation
- Underlying complexity is hidden to first order
 - but relevant for best performance
- Choice of distributed memory (MPI) or shared-memory (OpenMP) programming
- Flexibility:
 - OpenMP jobs using N cpus
 - MPI jobs using M cpus
 - Hybrid jobs using $M \times N$ cpus
 - Farms of N single cpu jobs
 - total memory of the system is available for all sizes of all jobs
 - easier to keep all cpus busy and to administer

General Issues

- Resource allocation
- Protection of your resources from activity of others
- Protection of others' activity from your errors
- Local memory accesses are faster than remote
 - memory allocated by “first touch”
 - ideally want $\leq 1\text{GB}$ memory local to each CPU
 - however O/S may migrate processes to remote CPUs
- Easy to get code to run poorly
 - e.g. with one CPU assigned most work, accessing mostly remote memory

COSMOS – The Basics

- Logging in and availability
- Interactive work and development environment
- Multiprocessor jobs
 - How to start them
 - What they look like
 - Where they run

Logging In

- Logins are allowed from registered hosts only
 - email cosmos_sys@damtp.cam.ac.uk to add or remove registered hosts (FQDN or IP address)
- SSH (ssh/scp/sftp) access only
 - **ssh abc123@cosmos.damtp.cam.ac.uk**
 - say “yes” to accept key the first time, **if** fingerprint matches
 - <http://www.damtp.cam.ac.uk/cosmos/user/index.html#quick>
 - X windows applications (e.g. emacs) started at a cosmos prompt should appear in new windows on your screen
 - if this doesn't work, you may need to try ssh -X or ssh -Y
 - don't try setting DISPLAY or using xauth to set cookies
- Outward connections are unrestricted

Availability

- In general, service is continual (24 hours, 7 days per week, 52 weeks per year) *EXCEPT*:
- Wednesdays 10:00-14:00 is a regular planned maintenance period
 - not always interrupting service, but a “vulnerable period”
 - see emails and the news page
- Special maintenance sessions are sometimes required
- We cannot guarantee instant response out of office hours (but email `cosmos_sys` anyway)
- We reserve the right to withdraw service without notice in an emergency threatening personnel, data or equipment.

Interactive Work - Guidelines

- For code development, job submission and data analysis
- Confined automatically to the first 8 cpus
- As this is small, large tests or production runs **must** be run through the batch queues (**not** interactively)
- *Watchdog* kills interactive processes using excessive cpu time
- Care needed (large jobs can compromise the system if accidentally launched on only 8 cpus)
- Don't use dplace (bad for sharing)
- *express* queue
 - for running test jobs on the first 8 cpus

Interactive Work - Tips

- Linux **top** has a problem on large machines
 - press t then W
- Once logged in you can start other xterms without ssh'ing again (and start other X programs)
- The processor arch is (64-bit) IA64, not the same as (32-bit) i386
 - code from i386 machines must be recompiled
 - i368 binaries will run in software emulation, *slowly*
- IDL 6.1 (6.2 soon) is available, but note is the i386 version
- Mathematica and Matlab are there but also 32-bit
- Web browsers should be avoided (32-bit anyway)

Development Environment

- Much more in next talk
- GCC compiler suite present but:
 - poor Fortran support
 - no OpenMP support
 - Intel compilers preferred for performance on IA64
- Intel compilers:
 - icc, ifort (currently v8.1; use -V for exact version)
 - many alternative versions available via **modules**
 - for full list: **modules whatis**
 - e.g. **module load icomp71**
 - binaries should remember which compiler built them, but in general ensure the same modules are loaded at run-time as at compile-time. Batch queue submission reminds about this.

Multiprocessor Jobs 1

- NB batch queues and dplace modify the following
- Scalar (single threaded) job:

```
cd /home/cosmos-tmp/acb123/jobdir  
./a.out
```

- or

```
cd /home/cosmos-tmp/acb123/jobdir  
./a.out >outfile 2>errfile
```

- OpenMP parallel job (N threads): set this environment variable first, then start like a scalar job:

```
export OMP_NUM_THREADS=N
```

- Make sure this is set explicitly to a low value interactively

Multiprocessor Jobs 2

- MPI parallel job (M threads):

```
cd /home/cosmos-tmp/acb123/jobdir  
mpirun -np M a.out
```

- Note that omitting the space as in `-npM` causes a daft error (*MPI: bad process count*)

- Hybrid parallel job (N OpenMP x M MPI):

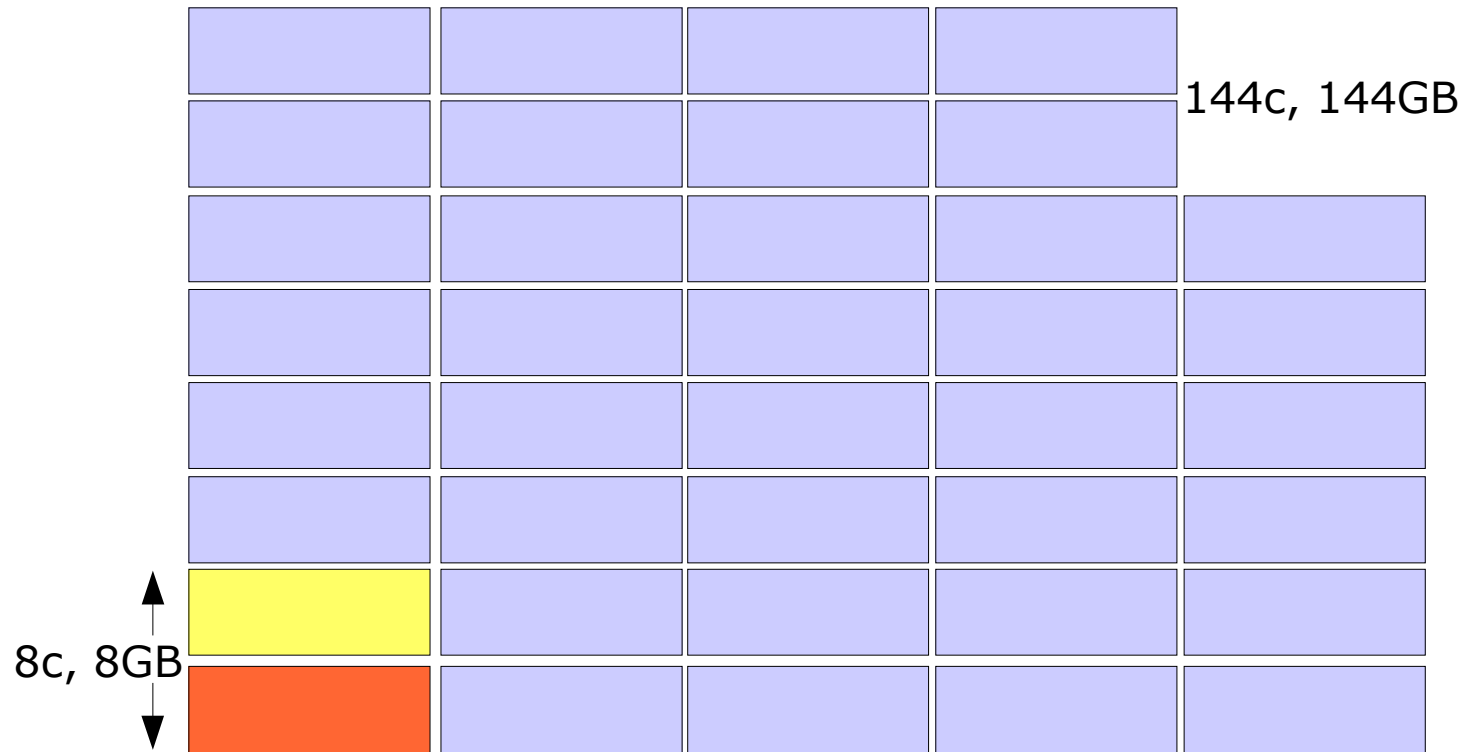
```
export OMP_NUM_THREADS=N  
cd /home/cosmos-tmp/acb123/jobdir  
mpirun -np M cosmomc
```

- The batch queue submission procedure will handle `OMP_NUM_THREADS` unless you prepare submission manually (in which case you need to remember it, as well as any module settings).

Threads vs CPUs

- Threads (\approx processes) are software entities
- Not all threads created by a parallel job do real work
- Use **ps -m** or press **H** in **top** to see OpenMP threads
- O/S runs all runnable threads on the available CPUs
- A single CPU can only execute one thread of anything at a given moment, but may timeshare if there are more threads to run than CPUs available
- Parallel performance requires that each working thread is allocated to a different CPU
- Best performance requires that jobs not overlap
- O/S may move threads among the available CPUs

Available CPUs



- Batch Jobs
- Interactive/Express
- Interactive

- CPUs 0-7 are available to **interactive** processes
- CPUs 8-151 are available to general **batch** jobs
- CPUs 4-151 are available to **express queue** jobs
- Each batch job is allocated a **unique** set of CPUs
 - *exception:* aux queue jobs
 - processes cannot migrate out of the set, but memory use need not be local to the set
- System processes have access to all CPUs

Shepherds

- Auxiliary threads/processes which do negligible work
- Should be ignored when computing CPU number
- Relevant for **dplace**, or for understanding **top** or **ps** output
- Created by multiprocessor jobs:
 - Each *mpirun -np M* contributes 2 shepherds (including mpirun)
 - Each group of OpenMP threads contributes 1 shepherd
 - E.g. hybrid code *cosmomc* with 4 chains, each chain using N OpenMP threads (*mpirun -np 4 cosmomc*) uses 4N worker threads and $2 + (4 \times 1) = 6$ shepherds.

dplace

- Binds processes to particular CPUs in the available set to prevent migration
 - memory allocated locally stays local
 - removes migration overhead
 - performance noticeably improves if done correctly
- Incorrect use can be **disastrous** to performance
- Aux queue jobs must be able to migrate
- Bind only workers, not shepherds

dplace - Examples

- MPI ($M > 1$)
 - **mpirun -np M dplace -s1 a.out**
2 shepherds: mpirun and the first a.out spawned (skip)
- OpenMP ($OMP_NUM_THREADS > 1$)
 - **dplace -x2 ./a.out**
2 = 10 (binary)
1 shepherd: the second process spawned (skip)
- Hybrid ($4 \times OMP_NUM_THREADS$ in size)
 - **mpirun -np 4 dplace -x481 ./a.out**
481 = 111100001 (binary)
6 shepherds: (probably) mpirun, 1st and 6th-9th spawned (skip)

COSMOS – Job Queues

- LSF batch queues
- Submission procedure (*live!*)
- LSF commands
- Watchdog
- Guidelines
- Advanced Usage
- Priorities and fairshares

Batch Queues

- Based on *Platform LSF 6.0*.
- Jobs can be submitted to the queues at any time
- System will allocate time, cpus and memory as efficiently and fairly as possible and start jobs
- By default, system will email at job start and finish
 - NB so we need your current email address!
- Execution order \neq submission order
- At busy times, expect ~ 8 hours waiting in the queue
- During the run, *watchdog* monitors actual resource usage

Which Queue?

- Large
 - 8-32 cpus (default 16), ≤ 64 GB memory, ≤ 8 hours real time
 - Jobs allocated non-overlapping CPU sets
 - dplace recommended
- Small
 - 1-8 cpus (default 4), ≤ 32 GB memory, ≤ 8 hours real time
 - Jobs allocated non-overlapping CPU sets
 - dplace recommended

Which Queue (ctd)?

- Aux
 - 1-16 cpus (default 4), $\leq 512\text{MB}$, ≤ 8 hours real time
 - Low performance, migratable, must have low impact on other jobs
 - dplace **forbidden**, and watchdog may **terminate** if necessary
- Express
 - 1-4 cpus (default 4), $\leq 4\text{GB}$ memory, ≤ 2 hours real time
 - For quick tests
 - Jobs from this queue can use 4 interactive cpus
 - Safer and much less restrictive than running tests interactively
- Other queues exist

How to Submit

1. Select an appropriate queue (e.g. *small*)
2. Create a text file containing job start commands (e.g. called *jobscript*), such as:

```
cd /home/cosmos-tmp/acb123/jobdir
```

```
mpirun -np M a.out >myoutfile 2>myerrfile
```

OMP_NUM_THREADS will be set by the next step (but any setting in *jobscript* will take precedence).

comments and bash-style syntax are allowed.

myoutfile, myerrfile are optional (defaults are outfile and errfile in the submission directory).

Avoid backgrounding jobs with **&** unless the final command of *jobscript* is **wait**. Then this can be used to construct a farm job (multiprocessor job consisting of several smaller sub-jobs).

How to Submit (ctd)

3. Run the command **small jobscript**

Answer each question and finally save the generated submission script (*small.abcd*) and/or submit it immediately to the LSF queueing system.

The **small**, **large**, **aux**, **express** (etc) commands prepend LSF directives (and other settings) to *jobscript* to generate a submission script. The LSF directives encode submission options, e.g.

```
# BSUB -n 8                # No. processors needed overall
# BSUB -W 480/cosmos      # Wall clock run limit
# BSUB -M 8388608         # Maximum physical memory required
# BSUB -R "rusage[mem=1024:duration=15m:decay=1]" # Expected resource consumption
```

This script can be fed manually to LSF via

bsub < small.abcd (note the <)

The BSUB directives have matching bsub options (the latter take precedence).

Submission Tips

- Specify resources as accurately as possible:
 - requesting too much run time or memory may prevent the job starting when opportunity actually exists
 - requesting too little memory will hurt performance globally and may even endanger service!
 - incorrect cpu number will either hobble your job or leave parts of the system idle and unusable by others
- Automatic method handles subtleties – e.g.
 - correct development environment (modules state)
 - occurrence of total memory in **two** different submission directives
- Manual edits of old submission scripts not recommended

LSF Commands

- `bqueues [-l] [queuename]`
 - information about queue *queuename*
- `bjobs [-l] [jobid]`
 - information about job *jobid* (or all your jobs if *jobid* omitted)
- `bjobs -uall`
 - all users' jobs
- `bjobs -lp`
 - list your PENDING jobs (i.e. those waiting in queues) including the PENDING REASONS.
- `bkill, bstop, bresume, bsub, bmod` (see man pages)

Watchdog

- Cron job running every 10 minutes
- Replaces LSF run-time resource monitoring which doesn't work (e.g. true job thread number and memory usage)
- Detects mismatches between submission directives and actual usage (and submission errors)
- Sends *one* email per job after a grace period
- Please take note, and adjust job parameters on future runs if necessary
- Watchdog kills badly-behaved aux queue jobs, and interactive processes consuming excessive cpu time **only**.

Guidelines

- Please read them!
- Most obviously:
 - please be mindful of other users
 - try not to cause a crash
 - don't set up automatically resubmitting jobs which could end up emailing the sysadmin every 2 minutes
 - please read the web documentation.

Advanced Usage 1

- Job dependencies
 - e.g. Job2 to start only after Job1 finishes successfully

Submit Job1 using -J to name it, as in:

```
# BSUB -J Job1
```

Then name Job2 and set a dependency via -w:

```
# BSUB -J Job2
```

```
# BSUB -w 'done("Job1")'
```

Advanced Usage 2

- Checkpointing
 - periodic saving of state so that work interrupted by emergencies and failures, or by a run time limit, can be continued later
 - difficult to do completely generally without much more memory and disk space
 - programmer should write a suitable routine to be triggered by receipt of the signal SIGURG and at regular intervals
 - add extra bsub options so that LSF sends the job SIGURG sufficiently soon prior to the end of the allowed run time:
 - # BSUB -wa 'URG'
 - # BSUB -wt '5' # SIGURG is sent 5 minutes before termination

Priorities and Fairshares

- Jobs are *not* launched in queue order
- Users and subconsortia have initial “shares”
- The most deserving user of the most deserving subconsortium is scheduled next
- If insufficient CPUs are free, those CPUs available are reserved
- Recent activity reduces effective shares (at all levels)
- Larger jobs (in terms of time and CPU number) expend shares faster
- Shares regenerate during inactivity
- **bhpart -r**

COSMOS - Storage

- Filesystems
- DMF and offline tape storage
 - Recalling offline files efficiently

Filesystems 1

- 3 CXFS filesystems (act as local to both cosmos and cosmos2)
 - 200MB/s FC SAN
 - RAID5
- Large read/writes perform better since each metadata transaction is mediated by *microcosm* (the metadata server) over TCP/IP.
- 5TB online disk is supplemented by 3.8TB offline tape
- Online <-> offline migration managed by DMF
 - automatic and manual migration
 - ensure data is online before jobs start

Filesystems 2

- **/home/cosmos** (100 GB)
 - Home directories: code only
 - 1GB quota, backed up nightly
 - Nightly backups recycled weekly, full backups made weekly and retained for 4 weeks
- **/home/cosmos-med** (1.4 TB)
 - Medium-term scratch work
 - 64GB quota, **not backed up**
- **/home/cosmos-tmp** (3.5 TB)
 - Short-term temporary storage (≤ 3 months)
 - No quotas, **not backed up**
 - Files older than 3 months are deleted at the start of each month
 - Files may automigrate to offline tape to recover online space

DMF

- On **/home/cosmos-tmp**, preferentially large and old files will automatically *migrate* to tape via DMF, to maintain 5-15% of disk space free for current jobs
- **dmls -al**
 - REG: normal file not yet managed by DMF (data is *online*)
 - DUL: file's data has been migrated to tape but also remains on disk (*dual state* – data is both *online* and *offline*)
 - OFL: file's data has been migrated to tape and the data blocks on disk have been released (data is now *offline* only)
- Only data is copied; directory structure unchanged
- Disk and tape data are not independent – **not a backup**

Recalling Offline Files

- An offline file is automatically recalled (data copied back to disk from tape) when the **contents** of the file are needed
- Each recall involves tape load/seek/read/unload so should not be left to happen automatically
- **dmget filename**
- Most efficiently for directories (minimize loading):
dmfind mydir -state MIG -o -state OFL | dmget
- REG -> MIG -> DUL -> OFL
- OFL -> UNM -> DUL

Common Problems

- *See also*

<http://www.damtp.cam.ac.uk/cosmos/faq.html>

Common Problems 1

- *Job ends up in the small queue with strange parameters (and probably never runs)*

If LSF cannot understand the job parameters supplied, defaults will be applied (small is the default queue)

most often this happens because you did

bsub submission_script

instead of

bsub < submission_script

Common Problems 2

- *Job doesn't run because "resource requirements not satisfied"*

There may not be sufficient memory to execute the job, but sometimes this occurs because you have manually edited an old submission script and not modified the resource requirement string correctly (the `#BSUB -R` directive).

Suggestion: generate a new submission script using the automatic method for a dummy job with identical CPU and memory requirements, and compare the `#BSUB` directives.

Common Problems 3

- *Usually when I log into cosmos, X applications display without problems on my screen, but now they are failing with "Can't open display."*

This is often a symptom of having exceeded your quota in your home directory – this prevents the addition of the correct "cookie" to your .Xauthority file and X authentication breaks.

Suggestion: check your quota with **quota -v**. Correct the overfull condition, log out and log in again.

Email Contact

- Please use

cosmos_help@damtp.cam.ac.uk

for general help and code issues

cosmos_sys@damtp.cam.ac.uk

for system problems and user administration

- These help us to be more efficient (both currently go to SJR and VT)