

PBS Queues

Christopher Holmes

September 2006

GEOS-Chem and some other programs (e.g. long IDL scripts) require extensive computing power. In order to keep the login machines (i.e. Sol, Europa, Callisto) running quickly and prevent routine work from slowing the fast machines (i.e. Terra, Altix, Titan, Amalthea), queues keep the login and fast servers separate. The PBS system allows one to submit and execute a job or program on a fast server without logging in to that server.

Basic commands

```
qstat -a @amalthea @europa @altix @terra @titan
```

Find out what's running on all servers. Only query Altix, Titan, and Terra from Sol.

```
submit {queue} {job} [{log}]
```

```
qsub -q {queue} {job}
```

Submit program {job} to queue {queue}, [with output directed to {log}].

Do not use `qsub` to altix or terra from europa.

`submit` is safer since it will not hang a server.

```
qdel {number} . {server}
```

Halt execution of a job. Find the number and server by using `qstat -a`.

```
trun
```

Testrun is a Perl script written by bmy to simplify compiling and executing GEOS-Chem.

See the webpage:

<http://www-as.harvard.edu/chemistry/trop/geos/doc/testrun/index.html>

Simple Job Scripts

Any job submitted via `submit` or `qsub` must be a shell script. Below are the file contents of two scripts that compile and execute GEOS-Chem, respectively. In practice, you may want more complex scripts, which you can copy from other people.

```
#!/bin/csh          (first line specifies the shell used by the Operating System)
make                (commands...)
```

```
#!/bin/csh
geos > log
```

Server & Queue descriptions (subject to change)

Terra (Sun 4100) -- fast machine, should only use for "long" runs (esp. 2x2.5)

Compile w/ Makefile.sparc

cmp_amd64		-- compile
mp2t	(t2x1)	-- 2 processor test queue (20 min)
mp256t	(t4x14)	-- 4 processors (14 hrs)

Altix & Titan (SGI Altix)

Compile w/ Makefile.ifort

cmp_intel	(intel)	-- compile
sp1a	(a1x20m)	-- 1 processor test queue (20 min)
sp36a	(a1x12)	-- 1 processor (12 hrs)
mp2a	(a2x1)	-- 2 processor test queue (32 min)
mp48a	(a4x4)	-- 4 processor (4 hrs) <i>large memory demands only</i>
mp128a	(a4x11)	-- 4 processors (11 hrs)
mp240a	(a4x20)	-- 4 processors (20 hrs)

Amalthea (SGI Origin)

Compile w/ Makefile.sgi

cmp_mips		-- compile
sp1	(q1x1)	-- 1 processor test queue (1 hr)
sp12	(q1x12)	-- 1 processor (12 hrs)
sp36	(q1x36)	-- 1 processor (36 hrs)
mp4	(q4x1)	-- 4 processors (1 hr)
mp32	(q4x8)	-- 4 processors (8 hrs)
mp64	(q4x16)	-- 4 processors (16 hrs)
mp128	(q4x32)	-- 4 processors (32 hrs)
mp240	(q4x60)	-- 4 processors (60 hrs)

Use Altix queues for most runs and Terra for the longest runs. The SGI Origins are slower, but generally underutilized.

Queue names begin with "sp", "mp", or "cmp", which signify "single processor", "multi-processor", and "compilation". The final character if any, designates the machine type on which the job will execute - "a" for Altix, "t" for Terra --- or on which it will compile - "mips" for SGI Origin, "intel" for Altix, "amd64" for Terra.

The number in the queue name gives the number of processor-hours (# of processors * maximum allowed time) that a SGI Origin 2000 would require to complete the longest allowed job (for historical consistency). So mp32 = 4 processors * 8 hrs. The computing units used in the queue names allow one to compare computing resources, so computations submitted to mp128 and mp128a will accomplish approximately the same amount, although the mp128a job will finish in 1/3 the time. A job submitted to mp256t will get more done in 14 hours than a job submitted to mp240a will get done in 20 hours.

