An Introduction to Grid Engine

aka:
Sun Grid Engine (SGE),
Oracle Grid Engine (OGE),
Open Grid Schedular(OGS)

Christian Bolliger

University of Zurich IT-Services

Overview

1. Main features of GE
2. Setup at UZH
3. Parallel Jobs
4. Open Source vs. Supported Version
5. Future?

Key Features

The Grid Engine (now owned by Univa) is a distributed resource managment system.
Key features:

- Highly Scalable
- Dynamic Resource Management
- Resource oriented
- Flexible
- Fine grained policies possible
- Advanced Reservation
- Usable for parallel jobs

Highlights

What I like

- Stability (except 6.2u3)
- Scriptability (xml output possible)
- Array Jobs
- Policy Implementation
- Backfillling
- Open Source (up to 6.2.u5), Sun Industry Standards Source License

Darker Spots

My worries

- Adaption necessary for parallel jobs
- Priorities difficult to understand for users

- Future of the open source version

Darker Spots

My worries

- Adaption necessary for parallel jobs
- Priorities difficult to understand for users – especially for jealous users
- Future of the open source version

CLI commands

Important User commands

- `qsub` (`qresub`)
- `qstat`
- `qalter`
- `qrsh` (`qlogin`)
- `qacct`
- `qconf` (read only for users)

CLI commands

Important User commands

- `qsub` (`qresub`)
- `qstat`
- `qalter`
- `qrsh` (`qlogin`)
- `qacct` – especially for jealous users
- `qconf` (read only for users)

CLI output (1)

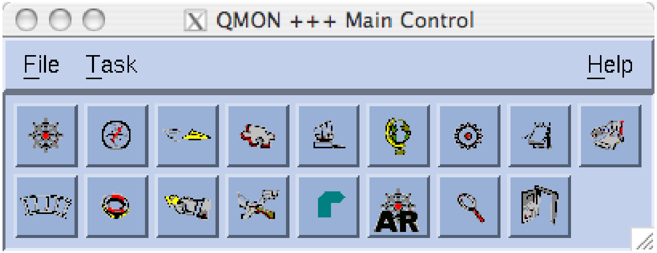
```
chribo@login1:~> qstat -u \*

job-ID  prior   name       user         state submit/start at     queue                    s
--------------------------------------------------------------------------------------------
2084013 0.50345 latr_2     srenneba     r     05/17/2011 23:23:22 long.q@r01c03b11n02.ften.e
2085384 0.51414 ramses3d   ableuler     r     05/15/2011 14:18:30 long.q@r06c03b06n02.ften.e
2085416 0.60000 sr2        dpotter      r     05/13/2011 13:31:31 iftp.q@r03c04b02n02.ften.e
2085538 0.52413 ramses3d   ableuler     r     05/13/2011 15:11:23 iftp.q@r02c02b10n01.ften.e
2085547 0.50345 i1vii.p22. marchand     Rr    05/16/2011 17:11:09 long.q@r01c03b09n01.ften.e
:                :                                 :
2085557 0.50345 ts.1vii.p2 marchand     r     05/17/2011 14:09:47 med.q@r06c04b05n01.ften.es
2085558 0.50345 ts.1vii.p2 marchand     r     05/17/2011 18:22:20 long.q@r01c03b12n01.ften.e
2086044 0.51414 ramses3d   ableuler     r     05/17/2011 15:19:00 med.q@r08c03b03n02.ften.es
2086057 0.51414 ramses3d   ableuler     r     05/17/2011 18:12:46 long.q@r06c03b07n01.ften.e
2086118 0.50078 rank-int.s murri        r     05/17/2011 00:58:02 short.q@r07c02b10n01.ften
2086118 0.50078 rank-int.s murri        r     05/17/2011 01:45:52 short.q@r07c01b06n02.ften
2086118 0.50078 rank-int.s murri        r     05/17/2011 03:03:02 short.q@r07c02b04n02.ften
```

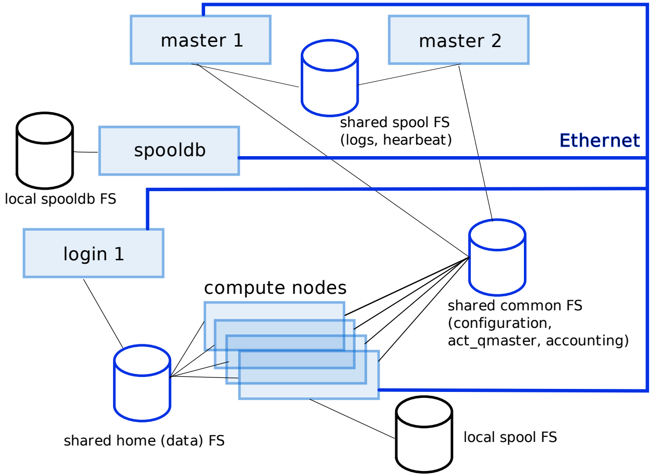
CLI output (2)

```
queue                            slots ja-task-ID
---------------------------------------------
long.q@r01c03b11n02.ften.es.hp    32
long.q@r06c03b06n02.ften.es.hp   128
iftp.q@r03c04b02n02.ften.es.hp  1328
iftp.q@r02c02b10n01.ften.es.hp   128
long.q@r01c03b09n01.ften.es.hp    16
        :                :
med.q@r06c04b05n01.ften.es.hpc    16
long.q@r01c03b12n01.ften.es.hp    16
med.q@r08c03b03n02.ften.es.hpc   128
long.q@r06c03b07n01.ften.es.hp   128
short.q@r07c02b10n01.ften.es.h     8 83
short.q@r07c01b06n02.ften.es.h     8 84
short.q@r07c02b04n02.ften.es.h     8 85
```

GUI - qmon



Setup at UZH


master 1
master 2
shared spool FS
(logs, hearbeat)
Ethernet
spooldb
local spooldb FS
login 1
compute nodes
shared common FS
(configuration,
act_qmaster, accounting)
shared home (data) FS
local spool FS

Resources and Queues

Grid Engine is resource orientated. That means the user decides how much compute time, memory etc. his job will need and GE will find an appropiate queue to run that job. Queues on Schroedinger:

- long.q 72 h (480 cores)
- med.q 48 h (960 cores)
- short.q 24 h (2304 cores)
- very-short.q 30 min, interactive access possible (96 cores)

Resources and Queues

Grid Engine is resource orientated. That means the user decides how much compute time, memory etc. his job will need and GE will find an appropiate queue to run that job.
Queues on Schroedinger:

- long.q 72 h (480 cores)
- med.q 48 h (960 cores)
- short.q 24 h (2304 cores)
- very-short.q 30 min, interactive access possible (96 cores)

Resources and Queues

Grid Engine is resource orientated. That means the user decides how much compute time, memory etc. his job will need and GE will find an appropiate queue to run that job.
Queues on Schroedinger:

- long.q 72 h (480 cores)
- med.q 48 h (960 cores)
- short.q 24 h (2304 cores)
- very-short.q 30 min, interactive access possible (96 cores)

UZH: **Nodes are given exclusively to one job!**

Scheduling Policy

- All users are treated equally, but this can changed easily (sharetree policy used).
- Fair use among groups is reached. Fair use within groups not wanted (just 5% weight).
- Aging of the usage: halftime of usage decay 28 days.
- Usage: 80% CPU, 20% Memory (default)
- Advanced Reservation active, Deadline tickets not used.

Scheduling Policy

- All users are treated equally, but this can changed easily (sharetree policy used).
- Fair use among groups is reached. Fair use within groups not wanted (just 5% weight).
- Aging of the usage: halftime of usage decay 28 days.
- Usage: 80% CPU, 20% Memory (default)
- Advanced Reservation active, Deadline tickets not used.

Scheduling Policy

- All users are treated equally, but this can changed easily (sharetree policy used).
- Fair use among groups is reached. Fair use within groups not wanted (just 5% weight).
- Aging of the usage: halftime of usage decay 28 days.
- Usage: 80% CPU, 20% Memory (default)
- Advanced Reservation active, Deadline tickets not used.

Scheduling Policy

- All users are treated equally, but this can changed easily (sharetree policy used).
- Fair use among groups is reached. Fair use within groups not wanted (just 5% weight).
- Aging of the usage: halftime of usage decay 28 days.
- Usage: 80% CPU, 20% Memory (default)
- Advanced Reservation active, Deadline tickets not used.

Scheduling Policy

- All users are treated equally, but this can changed easily (sharetree policy used).
- Fair use among groups is reached. Fair use within groups not wanted (just 5% weight).
- Aging of the usage: halftime of usage decay 28 days.
- Usage: 80% CPU, 20% Memory (default)
- Advanced Reservation active, Deadline tickets not used.

Tight Integration vs. Loose Integration

- Loose integration means that only the master job of a parallel job is controlled by the Grid Engine. The parallel library has to ensure that no zombies are left. No proper accounting available.
- Tight integration is what you expect from a cluster scheduler. The jobs are fully controlled by the Grid Engine, proper accountig is done. Out of the box tight integration is only available for *Open Mpi* when it is build with the `--with-sge` switch.
- For all other parallel libraries you have to craft a parallel environment which suits the needs of the library and eventually you have to adapt the code of the library.

Tight Integration vs. Loose Integration

- Loose integration means that only the master job of a parallel job is controlled by the Grid Engine. The parallel library has to ensure that no zombies are left. No proper accounting available.
- Tight integration is what you expect from a cluster scheduler. The jobs are fully controlled by the Grid Engine, proper accountig is done. Out of the box tight integration is only available for *Open Mpi* when it is build with the `--with-sge` switch.
- For all other parallel libraries you have to craft a parallel environment which suits the needs of the library and eventually you have to adapt the code of the library.

Tight Integration vs. Loose Integration

- Loose integration means that only the master job of a parallel job is controlled by the Grid Engine. The parallel library has to ensure that no zombies are left. No proper accounting available.
- Tight integration is what you expect from a cluster scheduler. The jobs are fully controlled by the Grid Engine, proper accountig is done. Out of the box tight integration is only available for *Open Mpi* when it is build with the `--with-sge` switch.
- For all other parallel libraries you have to craft a parallel environment which suits the needs of the library and eventually you have to adapt the code of the library.

The Shepherd Guards the Sheep

```
sge   4263     1 /gridware/sge/bin/lx24-amd64/sge_execd
root 8899  4263 \_ /usr/bin/perl ...load_sensor
sge   8850  4263 \_ sge_shepherd-2086368 -bg
user 8851  8850   \_ /gridware/sge/utilbin/lx24-amd64/qrsh_starter\
                     /gridware/sge/schroedinger/spool\
                     /r06c04b07n01/active_jobs/2086368.1/1.r06c04b07n01
user 8858  8851     \_ orted -mca ess env\
                      -mca orte_ess_jobid 3104702464\
                      -mca orte_ess_vpid 10 -mca orte_ess_num_procs 16\
                      --hnp-uri 3104702464.0;tcp://10.129.84.121:51387
user 8859  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
user 8860  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
user 8861  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
user 8862  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
user 8863  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
user 8864  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
user 8865  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
user 8866  8858        \_ /home/pci/user/.../cp2k.popt -in md.in
```

An Ugly but Effective Workarround

- The official way:
  Use a script: `/gridware/sge/mpi/startmpi.sh`.
  This script copys a launcher in to `/tmp` on the node.
  The launcher is a wrapper script around `qrsh` which behaves like `rsh`.
- Some libraries (e.g. mvapich) have `/usr/bin/rsh` hard encoded!
- The ugly, efficient workarround:
  Replace `/usr/bin/rsh` on all nodes by the wrapper script which is essentially:
  `qrsh -inherit -V <host> <cmd>`

An Ugly but Effective Workarround

- The official way:
  Use a script: `/gridware/sge/mpi/startmpi.sh`.
  This script copys a launcher in to `/tmp` on the node.
  The launcher is a wrapper script around `qrsh` which behaves like `rsh`.
- Some libraries (e.g. mvapich) have `/usr/bin/rsh` hard encoded!
- The ugly, efficient workarround:
  Replace `/usr/bin/rsh` on all nodes by the wrapper script which is essentially:
  `qrsh -inherit -V <host> <cmd>`

An Ugly but Effective Workarround

- The official way:
  Use a script: `/gridware/sge/mpi/startmpi.sh`.
  This script copys a launcher in to `/tmp` on the node.
  The launcher is a wrapper script around `qrsh` which behaves like `rsh`.
- Some libraries (e.g. mvapich) have `/usr/bin/rsh` hard encoded!
- The ugly, efficient workarround:
  Replace `/usr/bin/rsh` on all nodes by the wrapper script which is essentially:

```
qrsh -inherit -V <host> <cmd>
```

Open Source vs. Supported Version

- The community support is at least as good as the official support. The code is very well written, so that debugging on code level is possible but rarerly required.
- Compiling the code with all available features is not straight forward but worth to do.
- The intergration of the parallel libraries is not better under the supported version.
- By buying the supported version you eventually support the furture developement.

Future?

- Sun has been bought by Oracle which didn't show much interest in Open Source Software after the overtake.
- Oracle announced that the future developement will be properterian. There is no source available for version 6.2u7.
- The Open Source version has been forked on the base of version 6.2u5.
- Oracle sold the assets to Univa which is marketing now Grid Engine 8.0 (code base 6.2u5) and has employed key persons from the former SGE team.
- It is not sure yet if Univa will cooperate with the OSS maintainers.

Q & A, discussion

? ! ?