SLURM Version 1.3
May 2008

Morris Jette (jette1@llnl.gov)

Danny Auble (auble1@llnl.gov)

S&T Principal Directorate - Computation Directorate

UCRL-PRES-402832

Disclaimer

Major Changes in Slurm Version 1.3 Include

- Major changes in user commands
- Job accounting logic largely re-written and integrated with a database
- Major enhancements to job scheduling including support for gang scheduling (time-sharing for parallel jobs)
- See RELEASE_NOTES for a more complete description of changes

Command Changes

- *srun*'s *--allocate*, *--attach*, and *--batch* options removed. Use *salloc, sattach* and *sbatch* commands instead. Most options are consistent across commands
- *slaunch* command removed. Use *srun* command instead
- *srun --exclusive* option added for job steps
  - Provides resource management within job allocation for multiple concurrent job steps
- Feature counts added for job constraints
  - *srun --nodes=16 --constraint=graphics*4 …*

Command Changes (continued)

- *srun --pty* option added for pseudo-terminal support
- Time specification is more flexible:
  - <minutes> OR
  - <minutes>:<seconds> OR
  - <hours>:<minutes>:<seconds> OR
  - <days>-<hours>:<minutes>:<seconds>
- Much richer job dependency support:
  - Each job can be dependent upon many other jobs
  - Several dependency types added: Wait for other job to begin, complete successfully (exit code of zero), fail or complete (any exit status)

Accounting Changes

- Job accounting split into two components
  - JobAcctGatherType: Linux or AIX
  - AccountingStorage: SlurmDBD, MySQL, PostgreSQL, or text file
- New SlurmDBD (SLURM DataBase Daemon) securely manages accounting data for multiple clusters
- User/bank account database can be easily integrated with Moab Cluster Suite
- New tool, *sacctmgr,* available for managing the data
- Web tools are under development

Sample Accounting and Workload Scheduling Architecture

sacctmgr
sacct
(admin tools)

Moab
Cluster
Suite

SLURM
(cluster 1)

SLURM
(cluster 2)

Job and step
details

SlurmDBD

MySQL

User and account limits

User account and limit info
Job and step details

Scheduling Changes

- Backfill scheduler plugin re-written to support allconfigurations and job options
- Partitions have *Priority* parameter
  - Partitions can have overlapping nodes, but differing user, time, and size limits so they are really queues
- Partitions have a count of how many jobs can share an allocated resource (node, socket, core, etc. depending upon *SelectType* and *SelectTypeParameters*)

```
# Example of configuration with three priority levels and 2x oversubscription of standby
PartitionName=DEFAULT Nodes=tux[0-127] State=UP Shared=NO
PartitionName=express Priority=8 MaxNodes=8  MaxTime=30:00
PartitionName=normal   Priority=4 MaxNodes=64 MaxTime=8:00:00 Default=Yes
PartitionName=standby  Priority=1 MaxNodes=64 MaxTime=8:00:00 Shared=FORCE:2
```

Scheduling Changes (continued)

- Added support for cluster-wide consumable resources (e.g. licenses, added in v1.3.1)

- Many enhancements for Moab and Maui schedulers
  - New job and node state information managed
  - Slurm partitions and their jobs can be scheduled without Moab or Maui interaction for better responsiveness without scheduling policy support)

Gang Scheduling Support Added

- Gang scheduling support added to time-slice parallel jobs for improved responsiveness and utilization
- Jobs in the same partition sharing resources will alternately be suspended and resumed so all jobs make progress
- Jobs in lower priority partitions can be preempted (suspended) to execute jobs in higher priority partitions. Suspended jobs will automatically be resumed when idle resources are available
- Options and configuration parameters added to avoid memory over-subscription

Gang Scheduling Example

Time	Node 0	Node 1	Node 2	Node 3
0	Job A	Job A	Job A	Job A
1	Job B	Job B	Job C	Job C
2	Job D	Job D	Job D	Job E

All jobs make progress, but at 1/3 of the dedicated resource rate
Jobs can be started without having to wait for resources to be idle

Other Recent Changes

- Added support for periodic node health check (see *HealthCheckInterval* and *HealthCheckProgram*)
- Added response logic for non-killable processes (see *UnkillableStepTimeout* and *UnkillableStepProgram*)
- Configurable default job behavior on node failure (requeue or kill, see *JobRequeue*)
- Perl APIs and PBS/Torque command wrappers added (in v1.2.13)
- Event trigger support added (e.g. run some script when specific or any nodes goes DOWN, added in v1.2.2)