SLURM Version 2.3 and Beyond

Morris Jette
jette@schedmd.com

SchedMD LLC

Outline

- SchedMD and SLURM
- Contents of version 2.3
- Plans for future releases

SchedMD and SLURM

- Moe Jette and Danny Auble founded SchedMD LLC in 2010 in order to satisfy requests from the user community for SLURM development, while both maintained full-time employment at LLNL
- Conflicts of interest and demands upon our time made work at LLNL and SchedMD incompatible, so we left LLNL

Impact upon SLURM

- SLURM remains freely available under the GPL version 2 license
- We have no plans for a proprietary version of SLURM
- All development work by SchedMD has gone into the publicly available version of SLURM
- SLURM remains under active development by many companies and organizations
- More options now available for SLURM development and support

SLURM Version 2.3

- Released September 9, 2011
- New systems supported
  - Cray XE and XT systems
  - IBM BlueGene/Q systems (partial support)

SLURM Version 2.3

- Support added for multiple front-end nodes
  - Improves fault-tolerance for Cray and BlueGene systems
- Added ability to set default and maximum memory limits per partition instead of one value for the entire cluster
  - Provides better gang scheduling control (e.g. time-slice some partitions and not others)
- Added *GraceTime* to Partition and QOS data structures for job preemption
  - Gives job opportunity to gracefully stop
- Only current job dependencies are displayed
  - Satisfied dependencies are hidden for easier use

SLURM Version 2.3

- Better estimates of pending job's start time
- Support for Linux cgroups (containers)
    - Eventually can be used to manage job's memory allocation and device files (e.g. access to specific GPUs)
- Added ability to expand job sizes
    - Requires submission of new job that merges its resources into another job's resources

Job Expansion

```
$ salloc -N1 bash
salloc: Granted job allocation 65542
$ srun hostname
icrm1
```

Create original job allocation

```
$ salloc -N1 --dependency=expand:$SLURM_JOBID bash
salloc: Granted job allocation 65543
```

Create allocation for expanding original job

```
$ scontrol update jobid=$SLURM_JOBID NumNodes=0
To reset SLURM environment variables, execute
  For bash or sh shells:  . ./slurm_job_65543_resize.sh
  For csh shells:         source ./slurm_job_65543_resize.csh
$ exit
exit
salloc: Relinquishing job allocation 65543
```

Transfer additional resources to original job

```
$ scontrol update jobid=$SLURM_JOBID NumNodes=ALL
To reset SLURM environment variables, execute
  For bash or sh shells:  . ./slurm_job_65542_resize.sh
  For csh shells:         source ./slurm_job_65542_resize.csh
$ . ./slurm_job_$SLURM_JOBID_resize.sh
```

Update original job's environment variables (node count, node list, etc.)

```
$ srun hostname
icrm1
icrm2
$ exit
exit
salloc: Relinquishing job allocation 65542
```

Use expanded allocation

SLURM Version 2.4 Plans

- Available 2nd quarter 2012
- Complete SLURM port to IBM BlueGene/Q
- Wrappers for IBM's LoadLeveler commands
- Cloud Bursting: Move overflow work to the cloud
  - User would have to specify this is acceptable option
    - Application might start sooner
    - Application performance would likely suffer
  - Allocate, boot and start SLURM daemons in cloud
  - Add resources on demand, release idle resources

DOE Exascale Initiative

- SchedMD submitted a proposal for work we believe is essential for SLURM operation at Exascale
  - Power management
  - Heat management
  - Failure management
- None of this work is funded, but we wanted to discuss these ideas with a broader audience

Power Management Issues

- Power cost are likely to represent a significant cost of Exascale computing
  - Users will need to recognize the cost in order to adjust behavior accordingly
- Under some workloads, an Exascale computer's power demands may exceed power availability
  - The scheduler should optimize throughput within the available power envelope(s)
  - Power limits could effect multiple levels of resources
    - Entire computer center, cluster, set of racks, etc.

Application Power Management

- Collecting power use data about applications would be the first step
  - Add a SLURM plugin to collect power use information from various mechanisms to optimize flexibility
    - CPU/core frequency
    - Motherboard
    - Power monitors at the node, rack, and/or other level
    - Multiple plugins might be used on a single cluster
  - Different levels of precision are available from different mechanisms

Application Power Management

- Record job power use in accounting database along with a measure of precision
- Power use could be a factor in accounting
- Resource selection for jobs might be influenced to optimize precision of data collected
  - Large jobs allocated whole racks with power monitors
  - Smaller jobs allocated nodes with power monitors
  - Extrapolate as needed to get more precise data for entire job

Power Aware Scheduling

- Consider power envelopes in scheduling resources (tunable factor)
  - Use accounting records to estimate power needs of pending jobs
  - Coschedule high-power and low power jobs
  - Distribute high-power jobs through machine room
  - Schedule large high-power jobs at night when more power is available
  - Throttle jobs as needed (uniformly across all resources allocated to the job)
    - Add SLURM plugin for flexible control mechanism

Power Aware Scheduling

- Add job power control options
  - Get user guidance concerning application power/performance characteristics
- Gang scheduling (if used) would need to save/restore power configuration between jobs
  - Collection of power use would also need to be synchronized with gang scheduling

Heat Management

- Consider heat load of machine room as another facet of job scheduling decision process
  - Packing high-power job into a single rack may yield optimal communication performance, but generate too much heat
  - Nodes higher within a rack could be exposed to more heat and thus have lower performance characteristics
  - Need to begin collecting temperature data and develop scheduling algorithms to manage heat
  - May need to decrease job performance to address excess heat using similar logic to power management

Factors in Resource Selection

- Network topology (available today)
- Power management (future)
  - Optimized power usage data precision
  - Optimized overall power use
- Heat management (future)

Failure Management

- Add plugin to interface with RAS
  - Record SLURM failures and get information from other systems
  - Interface with CiFTS* and vendor-specific systems
- Expand failure management options for jobs and steps
  - Already have good mechanism for jobs to recognize and continue execution after failures
  - Cluster-wide hot-spare nodes
    - Replacement for job-specific spares as done today
  - Better checkpoint/restart support

* Coordinated Infrastructure for Fault Tolerant Systems
http://www.mcs.anl.gov/research/cifts/

Open Discussion

- Status of work at other sites
- Problems
- Requirements