Slurm Workload Management for GPU Systems

Morris Jette
SchedMD

GTC DC 2018

Outline

- Overview of Slurm design and functionality
- New GPU options

What is Slurm?

- Historically Slurm was an acronym standing for
  - **S**imple **L**inux **U**tility for **R**esource **M**anagement
- Development started in 2002 at Lawrence Livermore National Laboratory as a resource manager for Linux clusters
- Sophisticated scheduling plugins added in 2008
- Over 500,000 lines of C code (plus test suite and documentation)
- Used on many of the world's largest computers (6 of top 10)

Slurm Design Goals

- Highly scalable (managing 10.6 million core TaihuLight in China)
- Open source (GPL version 2, available on Github)
- System administrator friendly
- Secure
- Fault-tolerant (no single point of failure)
- Portable - targeting POSIX2008.1 and C99
- Scalable HTC environment (14k jobs/minute sustained)

Plugins

- Dynamically linked objects loaded at run time based upon configuration file and/or user options
- 86 plugins of 31 different varieties currently included with distribution
  - Network topology: 3D torus, tree, dragonfly, etc.
  - Energy accounting: Cray IBM AEM, IPMI, RAPL, etc.
  - Easily extensible for customizations

Slurm Kernel (65% of code)				
File System Acct Plugin	Energy Acct Plugin	ProcTrack Plugin	Topology Plugin	Accounting Storage Plugin
Lustre	IPMI	cgroup	Tree	SlurmDBD

Job Queues (Slurm Partitions)

- Resource allocation requests (jobs) are placed in priority-ordered queues
- Resources (compute nodes) can be in one or more queues
- Dozens of limits available on a queue, both per-job and aggregate
- Jobs can be submitted to multiple queues at the same time

Database Use

- Job accounting information
- Quality of Service (QOS) definitions
- Fair-share resource allocations
- Configuring limits (max job count, max job size, etc.)
  - Per Job limits (e.g. MaxNodes)
  - Aggregate limits by user, account or QOS (e.g. GrpJobs)
- Based upon hierarchical accounts
  - Limits by user AND by accounts
- Information pushed out live to scheduler daemons

And More ...

- Job steps: Sub-allocation of resources allocated to jobs
- Job dependencies
- Fine-grained control over task layout
- Wrappers for other workload manager commands
- Burst buffers management
- Job arrays
- KNL support
  - Automatic reboot to change NUMA/MCDRAM mode

New Slurm Enhancements for GPUs

- SchedMD and NVIDIA working together to make GPUs as easy to use and manage in Slurm as CPUs are today
- A major development effort
- Thanks to NVIDIA for its support!

New Job Submit Options

Same options apply to salloc, sbatch and srun commands

- --cpus-per-gpu= CPUs required per allocated GPU
- -G/--gpus= GPU count across entire job allocation
- --gpu-bind= Task/GPU binding option
- --gpu-freq= Specify GPU freq, memory freq, voltage
- --gpus-per-node= Works like “--gres=gpu:#” option today
- --gpus-per-socket= GPUs per allocated socket
- --gpus-per-task= GPUs per spawned task
- --mem-per-gpu= Memory per allocated GPU

New Configuration Parameters

Parameters available globally and on per-partition basis. The command line options override these default values.

- DefCpusPerGPU= Default CPUs count per allocated GPU
- DefMemPerGPU= Default memory size per allocated GPU

New select/cons_tres Plugin

- “cons_tres” represents “Consumable TRES”
- “TRES” represents “Trackable RESources”
- All functionality provided by “cons_res” plugin is also supported by “cons_tres” (e.g. CR_LLN, CR_PACK_NODES, CR_SOCKET, etc.)
- New “gpu” job options only supported by the cons_tres plugin
  - No other select plugin recognizes the new GPU options
- Available in Slurm version 19.05

Examples of Use (1 of 2)

$ sbatch --gpus=16 --gpu-freq=low,verbose --gpu-bind=closest --nodes=2 my.bash

$ sbatch --ntasks=16 --gpus-per-task=2 my.bash

$ sbatch --ntasks=8 --ntasks-per-socket=2 --gpus-per-socket=tesla:4 my.bash

$ sbatch --gpus=gtx1080:8,gtx1060:2 --nodes=1 my.bash

Examples of Use (2 of 2)

Allocation of resources to job steps also supports these GPU options:

```
$ cat my.bash
#!/bin/bash
srun --gpus=1 --ntasks=1 --nnodes=1 app1 &
srun --gpus=1 --ntasks=1 --nnodes=1 app2 &
srun --gpus=2 --ntasks=1 --nnodes=1 app3 &
srun --gpus=2 --ntasks=1 --nnodes=1 app4 &
Wait

$ sbatch --gpus=2 my.bash
```

Conflicting Options

- Given the multitude of options, it is possible to submit a job with conflicting options
  - Many conflicting options are possible
  - In most cases the job will be rejected

```
$ sbatch --gpus-per-task=1 --gpus-per-node=2  --ntasks-per-node=1  ...
```

Implicitly sets tasks-per-node to 2

Explicitly sets tasks-per-node to 1

Additional Enhancements for GPUs

- Concurrent selection of co-located CPUs and GPUs for improved locality/performance
- Consideration of GPUs with NVLink (high speed communications between GPUs and GPUs/CPUs) to select preferred GPUs for co-scheduling

Deployment Schedule of Version 19.05

25 October 2018 - Pre-release of Slurm version 19.05 with new GPU option support mostly functional, unsupported

May 2019 - Slurm version 19.05 with full functionality and support

Downloads: https://www.schedmd.com/downloads.php

Sources: git://github.com/SchedMD/slurm.git

Questions?