General Overview
Slurm Training15

Alfred Gil & Jordi Blasco (HPCNow!)
About Slurm
About Slurm

- **Slurm** was an acronym for Simple Linux Utility for Resource Management.
- Development started in 2002 at LLNL.
- It evolved into a capable job scheduler through use of optional plugins.
- On 2010 the Slurm developers founded SchedMD.
- It has about 500,000 lines of C code. Not Simple anymore.
- Now is called: Slurm Workload Manager.
- Supports AIX, Linux, Solaris, other Unix variants.
- Used on many of the world’s largest computers.
- Official website: http://slurm.schedmd.com
Support

Community Support through email list slurm-dev@schedmd.com

We strongly recommend 3rd-Level Support provided by SchedMD
Key Features Available in Slurm

<table>
<thead>
<tr>
<th>Feature</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full control over CPU usage</td>
<td>Topology awareness</td>
</tr>
<tr>
<td>Full control Memory usage</td>
<td>Job Preemption support</td>
</tr>
<tr>
<td>Scheduling techniques &amp; performance</td>
<td>Job requeue mechanism</td>
</tr>
<tr>
<td>Job Array support</td>
<td>Suspension and Resume capabilities</td>
</tr>
<tr>
<td>Integration with MPI</td>
<td>Kernel Level Checkpointing &amp; Restart</td>
</tr>
<tr>
<td>Interactive sessions support</td>
<td>Job Migration</td>
</tr>
<tr>
<td>High Availability</td>
<td>Job profiling</td>
</tr>
<tr>
<td>Debugger friendly</td>
<td></td>
</tr>
</tbody>
</table>
Extending Slurm

Extending Slurm features

- Write a plug-in
- C API slurm.h
- Perl API
- Python API

Contributing back to Slurm

1. Fork the Github Slurm branch.
2. Clone the Github Slurm branch.
3. Upload your changes to your github repo.
4. Create a pull request to SchedMD Github.
Resource Management

Manage resources within a single cluster

- **Nodes**
  - state (up/down/idle/allocated/mix/drained)
  - access

- **Jobs**
  - queued and running
  - start/stop
  - suspended
  - resizing
  - completing
Figure: Job States and job flow. Source SchedMD
Definitions of Socket, Core & Thread

**Socket**
Receptacle on the motherboard for one physically packaged processor (each of which can contain one or more cores).

**Core**
A complete private set of registers, execution units, and retirement queues needed to execute programs.

**Threads**
One or more hardware contexts within a single core. Each thread has attributes of one core; managed & scheduled as a single logical processor by the OS.

**Figure:** Definitions of Socket, Core & Thread. Source SchedMD

Alfred Gil & Jordi Blasco (HPCNow!)
Slurm daemons

- **slurmctld**
  - Central management daemon
  - Master/Slave
- **slurmdbd (optional)**
  - Accounting database system
- **slurmd**
  - On every compute node
- **slurmstepd**
  - Started by slurmd for every job step
Running a job

Job/step allocation

- sbatch – submits a script job.
- salloc – creates an allocation.
- srun – runs a command across nodes.
- srun_cr – runs a command across nodes with C&R.
- sattach – Connect stdin/out/err for an existing job or job step.
- scancel – cancels a running or pending job.
- sbcast – Transfer file to a compute nodes allocated to a job.
Running a job

**sr**un : Simple way to manage MPI, OpenMP, pthreads & serial jobs

- Slurm provides a single command line to manage all the MPI flavors, OpenMP, Pthreads and serial applications
- Users don’t need to worry about MPI flags and options for each MPI implementation mpirun/mpiexec/mpiexec.hydra
- The tool is called **sr**un and it should be mandatory for submitting jobs in the cluster.
- It provides key features like:
  - memory affinity
  - CPU binding
  - cgroups integration
Examples

sbatch

~ $ vim testjob.sl
~ $ cat testjob.sl
#!/bin/bash
#SBATCH --nodes=10
srun echo "running on : $(hostname)"
srun echo "allocation : $SLURM_NODELIST"

~ $ sbatch testjob.sl
Submitted batch job 11109

~ $ cat slurm-11109.out
running on : hsw001
allocation : hsw[001-010]
Examples

sbatch

~ $ vim testjob.sl
~ $ cat testjob.sl
#!/bin/bash
#SBATCH --nodes=10
srun echo "running on : $(hostname)"
srun echo "allocation : $SLURM_NODELIST"

~ $ sbatch testjob.sl
Submitted batch job 11109

~ $ cat slurm-11109.out
running on : hsw001
allocation : hsw[001-010]
sbatch

```
~ $ vim testjob.sl
~ $ cat testjob.sl
#!/bin/bash
#SBATCH --nodes=10
srun echo "running on : $(hostname)"
srun echo "allocation : $SLURM_NODELIST"
```

```
~ $ sbatch testjob.sl
Submitted batch job 11109
```

```
~ $ cat slurm-11109.out
running on : hsw001
allocation : hsw[001-010]
```
Submitting a Job

Job Description Example: SMP

```bash
#!/bin/bash
#SBATCH -o blastn-%j.%N.out
#SBATCH -D /share/src/blast+/example
#SBATCH -J BLAST
#SBATCH --cpus-per-task=4
#SBATCH --mail-type=end
#SBATCH --mail-user=youremail@yourdomain.org
#SBATCH --time=08:00:00
ml blast+
export OMP_NUM_THREADS=4
srun blastn
```
Submitting a Job

Job Description Example: MPI

```bash
#!/bin/bash
#SBATCH -o migrate-%j.%N.out
#SBATCH -D /share/src/migrate/example
#SBATCH -J MIGRATE-MPI
#SBATCH --ntasks=96
#SBATCH --mail-type=end
#SBATCH --mail-user=youremail@yourdomain.org
#SBATCH --time=08:00:00
ml migrate/3.4.4-ompi
srun migrate-n-mpi parmfile.testml -nomenu
```
Submitting a Job

Parametric Job

To submit 10,000 element job array will take less than 1 second!
```
sbatch -array=1-10000 jobarray.sh
```
In order to identify the input files or the parameters you can use environment variable as array index:
```
SLURM_ARRAY_TASK_ID
```
Submitting a Job

Parametric Job Example: jobarray.sh

```bash
#!/bin/bash
#SBATCH -J JobArray
#SBATCH --time=01:00:00
#SBATCH --cpus-per-task=4
#SBATCH --mem-per-cpu=1024
PARAMS=$(sed -n ${SLURM_ARRAY_TASK_ID}p params.dat)
srun your_binary $PARAMS
```

params.dat example

```
-file your_input_file -x 1 -y 1 -z 1
-file your_input_file -x 1 -y 1 -z 2
-file your_input_file -x 1 -y 1 -z 3
```
Submitting a Job

Parametric Job

The management is really easy:

```bash
squeue

 JOBID PARTITION PRIOR NAME USER STATE TIME  TIMELIMIT NODES  CPUS GRES
2142564_[20-1000] high 24544 JobArray jordi PD 0:00 1:00:00 2 64 (null) 2015-01-14T17:47:59 (Resources)
2142564_1 high 24544 JobArray jordi R 0:02 1:00:00 25 64 (null) 2015-01-14T16:47:59 hsw-[001,006,008,011-013}
2142564_2 high 24544 JobArray jordi R 0:02 1:00:00 13 64 (null) 2015-01-14T16:47:59 hsw-[005,010,013,017,026,032,045]

$ scancel 204258_[400-500]

$ scontrol hold 204258
```

Alfred Gil & Jordi Blasco (HPCNow!)
Realistic example

#!/bin/bash
#SBATCH -J GPU_JOB
#SBATCH --time=01:00:00  # Walltime
#SBATCH -A uoa99999     # Project Account
#SBATCH --ntasks=4      # number of tasks
#SBATCH --ntasks-per-node=2  # number of tasks per node
#SBATCH --mem-per-cpu=8132 # memory/core (in MB)
#SBATCH --cpus-per-task=4 # 4 OpenMP Threads
#SBATCH --gres=gpu:2     # GPUs per node
#SBATCH -C kepler
ml GROMACS/4.6.5-goolfc-2.6.10-hybrid
srun mdrun_mpi --------
System Information

- `squeue` – shows the status of jobs.
- `sinfo` – provides information on partitions and nodes.
- `sview` – GUI to view job, node and partition information.
- `smap` – CLI to view job, node and partition information.
### System Information

**squeue**

Show jobid and allocated nodes for running jobs of the user jordi:

```
[4925]  ~$ squeue -u jordi

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST(REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>24258</td>
<td>high</td>
<td>Migrate-N</td>
<td>jordi</td>
<td>PD</td>
<td>0:00</td>
<td>4</td>
<td>(Resources)</td>
</tr>
<tr>
<td>24259</td>
<td>high</td>
<td>Migrate-N</td>
<td>jordi</td>
<td>PD</td>
<td>0:00</td>
<td>4</td>
<td>(Priority)</td>
</tr>
<tr>
<td>24257</td>
<td>high</td>
<td>Migrate-N</td>
<td>jordi</td>
<td>R</td>
<td>0:27</td>
<td>512</td>
<td>hsw[1-512]</td>
</tr>
</tbody>
</table>
```
System Information

svview

![Job list screenshot]

Alfred Gil & Jordi Blasco (HPCNow!)
Slurm Commands

Accounting

- `sacct` – Report accounting information by individual job and job step
- `sstat` – Report accounting information about currently running jobs and job steps (more detailed than `sacct`)
- `sreport` – Report resources usage by cluster, partition, user, account, etc.
Slurm Commands

Scheduling

- `sacctmgr` – Database management tool
  Add/delete clusters, accounts, users, etc.
  Get/set resource limits, fair-share allocations, etc.
- `sprio` – View factors comprising a job’s priority
- `sshare` – View current hierarchical fair-share information
- `sdiag` – View statistics about scheduling module operations
  (execution time, queue length, etc.)
Slurm Commands

**Slurm Administration Tool**

**scontrol** - Administrator tool to view and/or update system, job, step, partition or reservation status. The most popular are:

- checkpoint
- create
- delete
- hold
- reconfigure
- release
- requeue
- resume
- suspend

More information: [http://slurm.schedmd.com/scontrol.html](http://slurm.schedmd.com/scontrol.html)
Enjoy Slurm

IT'S HIGHLY ADDICTIVE!!!