Heterogeneous Job Support

Tim Wickberg
SchedMD

SC17

Submitting Jobs

- Multiple independent job specifications identified in command line using “:” separator
- The job specifications are sent to slurmctld daemon as a list in a single RPC
- The entire request is validated and accepted or rejected
- Response is also a list of data (e.g. job IDs)

```
$ salloc -n1 -C haswell : -n256 -C knl bash
```

Batch Jobs

- Job components specified using ":" command line separator OR
- Use "#SBATCH" options in script separating components using "#SBATCH packjob"
- Script runs on first component specified

```
$ echo my.bash
#!/bin/bash
#SBATCH -n1 -C haswell
#SBATCH packjob
#SBATCH -n256 -C knl
…
$ sbatch my.bash
```

Job Data Structure

- Each component of a heterogeneous job has its own job structure entry
- “Head” job has pointers to all components (like job arrays)
- New fields
  - JobID - Unique for each component of the heterogeneous job
  - PackJobID - Common value for all components
  - PackJobOffset - Unique for each component, zero origin
  - PackJobIdSet - List of all job IDs in the heterogeneous job

Sample Job Data

Job ID	Pack Job ID	Pack Job Offset	Pack Job ID Set
123	123	0	123-127
124	123	1	123-127
125	123	2	123-127
126	123	3	123-127
127	123	4	123-127

Job Management

- Standard format ID for managing heterogeneous jobs is "<PackJobID>+<PackJobOffset>"

```
$ squeue --job=93
JOBID PARTITION NAME USER ST TIME NODES  NODELIST
 93+0     debug test adam  R 4:56     1  nid00001
 93+1     debug test adam  R 4:56     2  nid000[10-11]
 93+2     debug test adam  R 4:56     4  nid000[20-23]
```

Job Management Examples

```
# Update all components of a heterogeneous job
$ scontrol update jobid=93 timelimit=1-0

# Update specific component of a heterogeneous job
$ scontrol update jobid=123+1 account=abc

# Cancel all components of a heterogeneous job
$ scancel hold 123

# Get accounting information about all components of a heterogeneous job
$ sacct -j 89

# Get accounting information about specific component of a heterogeneous job
$ sacct -j 89+4
```

Job Steps

- srun launches application only in PackJobOffset=0 by default
- Use --pack-group option to launch step in other components

Job Step Examples

```
$ salloc -N1 : -N2 bash
Granted job allocation 6819
$ squeue
JOBID  PARTITION NAME USER ST TIME NODES  NODELIST
  6819+0     debug test adam  R 0:02     1  nid00001
  6819+1     debug test adam  R 0:02     2  nid0000[2-3]
$ srun hostname
nid00001
$ srun --pack-group=1 hostname
nid00002
nid00003
$ srun --pack-group=0,1 --label hostname
0: nid00001
1: nid00002
2: nid00003
```

MPI Support

- Environment variables and Slurm's MPI plugins establish environment so that entire environment (possibly spanning multiple job allocations) looks like a single job allocation
- Job step can not span job components in Slurm version 17.11
  - More work required for MPI support
    - Only OpenMPI with pmi2 plugin supported today and special Slurm configuration required to enable use (SchedulerParameters=enable_pack_step)
  - Addressed in version 18.08

Environment Variables

- Component specific information identified with "PACK_GROUP_#" suffix
- Otherwise global job information reported

```
SLURM_JOB_ID=6819
SLURM_JOB_ID_PACK_GROUP_0=6819
SLURM_JOB_ID_PACK_GROUP_0=6820
SLURM_JOB_NODELIST=nid0000[1-3]
SLURM_JOB_NODELIST_PACK_GROUP_0=nid00001
SLURM_JOB_NODELIST_PACK_GROUP_1=nid0000[2-3]
```

Burst Buffers

- Tied to specific job ID
- Use persistent burst buffer to access from all components

```
#!/bin/bash
#SBATCH -n1 -C haswell
#BB create_persistent name=alpha capacity=10TB access=striped type=scratch
#DW persistentdw name=alpha
#SBATCH packjob
#SBATCH -n256 -C knl
#DW persistentdw name=alpha
…
```

Scheduling

- Only the backfill scheduler will allocate resources
- All components must be allocated resources at the same time
- Backfill scheduler resource reservations for all components synchronized
- Limits of all job components considered before trying to start any component
- All components must be allocated resources on different nodes (mostly a limitation of MPI API)

Limitations

- Arrays of heterogeneous jobs not supported
- All components must run in same cluster (not across federation)
- Not supported with Cray ALPS
- Limited support for steps spanning heterogeneous jobs until version 18.08