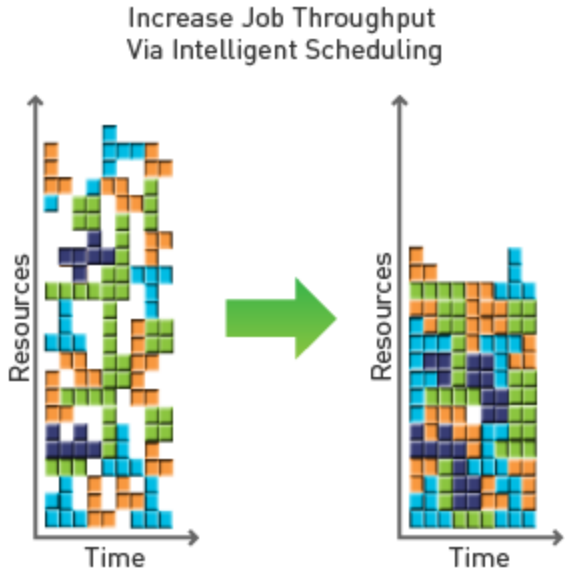
SLURM Primer

SLURM on Discovery Cluster



Northeastern University Research Computing: Nilay K Roy, MS Computer Science, Ph.D Computational Physics

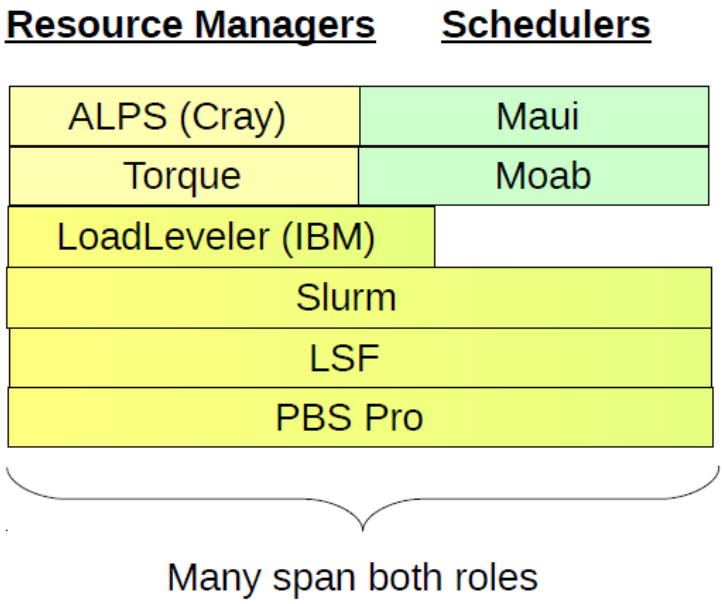
SLURM – Some background

- SLURM = SIMPLE LINUX UTILITY FOR RESOURCE MANAGEMENT
- http://slurm.schedmd.com – Free and Open Source
- Others: SGE now OGE, Torque, Condor, Platform LSF
- SLURM (hilbert curve scheduling – top500.org – on 1st fastest supercomputer – China's Tianhe-2 – 33.86 PETAFLOPS –16,000 nodes, each with two Intel Xeon IvyBridge processors and three Xeon Phi processors for a total of 3.1 million cores – 17.8MW) – November 2015

Roles of resource manager and job scheduler

Allocate resources within a cluster

- Nodes (typically a unique IP address)
  - NUMA boards
- Sockets
  - Cores
- Hyperthreads
- Memory
  - Interconnect/switch resources
  - Generic resources (e.g. GPUs)
- Licenses
- Launch and otherwise manage jobs



Can require extensive knowledge about the hardware and system software (e.g. to alter network routing or manage switch window)

Slurm started as a resource manager (the “rm” in “slurm) and added scheduling logic later

Simple Linux Utility for Resource Management

- Development started in 2002 at Lawrence Livermore National Laboratory as a resource manager for Linux clusters
- Sophisticated scheduling plugins added in 2008
- About 500,000 lines of C code today
- Supports AIX, FreeBSD, Linux, Solaris, other Unix variants
- Used on many of the world's largest computers
- Active global development community

Highly scalable (managing 3.1 million core Tianhe-2, tested to much larger systems using emulation)

- Open source (GPL version 2, available on Github)
- System administrator friendly
- Secure
- Fault-tolerant (no single point of failure)
- Portable

Dynamically linked objects loaded at run time based upon configuration file and/or user options

- 100+ plugins of 26 different varieties currently available
- Network topology: 3D-torus, tree, etc
- MPI: OpenMPI, MPICH1, MVAPICH, MPICH2, etc.
- External sensors: Temperature, power consumption, lustre usage,

Slurm Kernel (65% of code)				
Authentication Plugin	MPI Plugin	Checkpoint Plugin	Topology Plugin	Accounting Storage Plugin
Munge	mvapich	BLCR	Tree	MySQL

```
[nilay.roy@discovery4 ~]$ sacctmgr -p show association where user=nilay.roy
Cluster|Account|User|Partition|Share|GrpJobs|GrpNodes|GrpCPUs|GrpMem|GrpSubmit|GrpWall|GrpCPUMins|MaxJobs|MaxNodes|MaxCPUs|MaxSubmit|MaxWall|MaxCPUMins|QOS|Def QOS|GrpCPURunMins|
discoverycluster|generalqueues|nilay.roy|ser-par-10g-2|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|ser-par-10g-3|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|ser-par-10g-4|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|par-gpu|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|par-gpu-2|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|interactive-10g|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|interactive-ib|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|ht-10g|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|largemem-10g|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|hadoop-10g|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|ser-par-10g|1|||||||||||normal|||
discoverycluster|generalqueues|nilay.roy|parallel-ib|1|||||||||||normal|||
[nilay.roy@discovery4 ~]$
```

Slurm Entities

- Jobs: Resource allocation requests
- Job steps: Set of (typically parallel) tasks
- Typically an MPI, UPC and/or multi-threaded application program
- Allocated resources from the job's allocation
- A job can contain multiple job steps which can execute sequentially or concurrently
- Use cases with thousands of job steps are common
- Partitions: Job queues with limits and access control

Node State Information

- Baseboards, Sockets, Cores, Threads
- CPUs (Core or thread count depending upon configuration)
- Memory size
- Generic resources (with names and counts)
- Features (arbitrary string, e.g. OS version or CPU type)
- State (e.g. drain, down, etc.)
- Reason, time and user ID

(e.g. “Bad PDU [operator@12:40:10T12/20/2013]”)

Numeric suffix with comma separated numbers or ranges

- Used in all commands and configuration files to make managing large clusters easier
- Bracketed value be at end of name (with optional range elsewhere in the name)

```
[nilay.roy@compute-0-004 ~]$
[1]+  Done                    sview
[nilay.roy@compute-0-004 ~]$ exit
logout
Connection to compute-0-004 closed.
[nilay.roy@discovery4 ~]$ exit
exit
salloc: Relinquishing job allocation 120
salloc: Job allocation 120 has been revoked.
[nilay.roy@discovery4 ~]$ sinfo -Nle
Tue Apr 12 11:12:30 2016
NODELIST                   NODES       PARTITION       STATE CPUS    S:C:T MEMORY TMP_DISK WEIGHT FEATURES REASON
compute-0-[000-003,008-063]    60    ser-par-10g*        idle   16    2:8:1      1        0      1   (null) none
compute-0-[000-003]             4 interactive-10g        idle   16    2:8:1      1        0      1   (null) none
compute-0-[004-007]             4          ht-10g        idle   32   2:16:1      1        0      1   (null) none
compute-0-[064-065]             2   ondrechen-10g        idle   40   2:10:2      1        0      1   (null) none
compute-0-[066-095]            30   ser-par-10g-2        idle   40   2:10:2      1        0      1   (null) none
compute-0-[096-143]            48   ser-par-10g-3        idle   40   2:10:2      1        0      1   (null) none
compute-0-[144-327]           184   ser-par-10g-4        idle   48   2:12:2      1        0      1   (null) none
compute-1-[064-127]            64     parallel-ib        idle   16    2:8:1      1        0      1   (null) none
compute-1-[064-067]             4  interactive-ib        idle   16    2:8:1      1        0      1   (null) none
compute-2-[000-003]             4    largemem-10g        idle   32    2:8:2      1        0      1   (null) none
compute-2-[004-006]             3      hadoop-10g        idle   40   2:10:2      1        0      1   (null) none
compute-2-008                   1             suh        idle   32    2:8:2      1        0      1   (null) none
compute-2-[128-159]            32         par-gpu        idle   32    2:8:2      1        0      1   (null) none
compute-2-[160-175]            16       par-gpu-2        idle   48   2:12:2      1        0      1   (null) none
compute-3-[000-031]            32         redwood        idle   20   2:10:1      1        0      1   (null) none
compute-3-[032-039]             8         hanchen        idle   40   2:10:2      1        0      1   (null) none
compute-3-040                   1    westlargemem        idle   40   2:10:2      1        0      1   (null) none
compute-3-[041-047]             7            west        idle   40   2:10:2      1        0      1   (null) none
compute-3-[048-063]            16              fu        idle   40   2:10:2      1        0      1   (null) none
compute-3-[064-079]            16         maloney        idle   40   2:10:2      1        0      1   (null) none
compute-3-080                   1             suh        idle   40   2:10:2      1        0      1   (null) none
compute-3-081                   1         mcguire        idle   40   2:10:2      1        0      1   (null) none
compute-3-[082-084,086]         4         feiguin        idle   40   2:10:2      1        0      1   (null) none
compute-3-085                   1           pizer        idle   40   2:10:2      1        0      1   (null) none
compute-3-087                   1      briesacher        idle   40   2:10:2      1        0      1   (null) none
compute-3-[088-091]             4         griffin        idle   40   2:10:2      1        0      1   (null) none
compute-3-[096-103]             8       ioannidis        idle   40   2:10:2      1        0      1   (null) none
compute-4-000                   1        krioukov        idle   80   4:10:2      1        0      1   (null) none
compute-4-[001-004]             4        lhct3alv        idle    8    2:2:2      1        0      1   (null) none
compute-4-[017-020]             4    krioukov_gpu        idle   48   2:12:2      1        0      1   (null) none
[nilay.roy@discovery4 ~]$
```

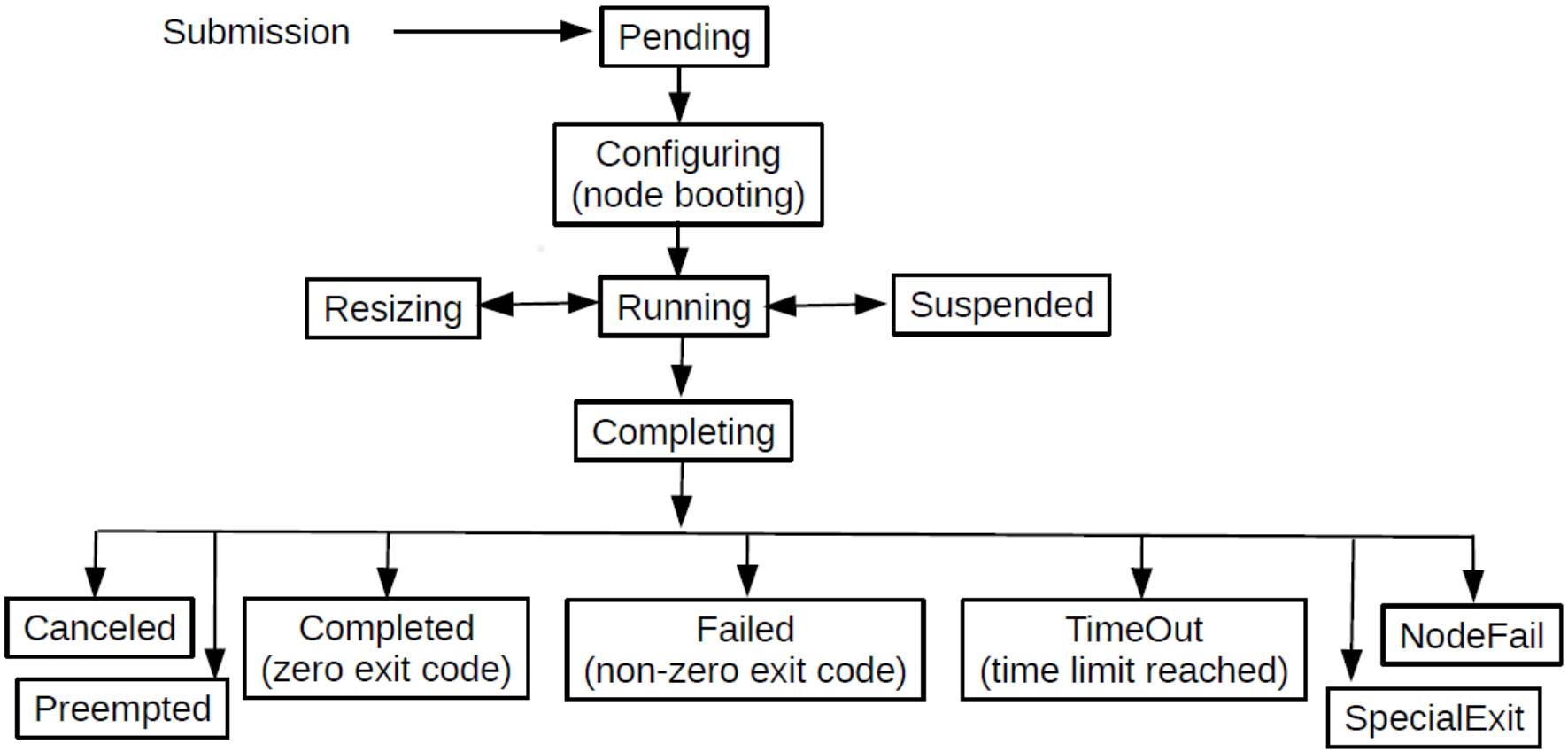
Queue/Partition State Information

- Associated with specific set of nodes
- Nodes can be in more than one partition
- Job size and time limits (e.g. small size and time limits for some partition and larger limits for others)
- Access control list (by bank account, Quality Of Service or Linux group)
- Preemption rules
- State information (e.g. up, down, drain, etc.)
- Over-subscription and gang scheduling rules

Job State Information

- ID (a number)
- Name
- Time limit (minimum and/or maximum)
- Size specification (minimum and/or maximum; nodes, CPUs, sockets, cores, and/or threads)
- Specific node names to include or exclude in allocation
- Node features required in allocation
- Dependency
- Account name
- Quality Of Service (QOS)
- State (Pending, Running, Suspended, Canceled, Failed, etc.)

Job States


Submission
Pending
Configuring
(node booting)
Resizing
Running
Suspended
Completing
Canceled
Preempted
Completed
(zero exit code)
Failed
(non-zero exit code)
TimeOut
(time limit reached)
NodeFail
SpecialExit

Step State Information

ID (a number): <jobid>.<stepid>
- Name
- Time limit (maximum)
- Size specification (minimum and/or maximum; nodes, CPUs, sockets, cores, and/or threads)
- Specific node names to include or exclude in allocation
- Node features required in allocation

Job is submitted to a Slurm queue/partition
- Job is allocated resources (cores, memory, etc.)
- Job steps execute applications using the job's resources

Daemons

slurmctld – Central controller (typically one per cluster)

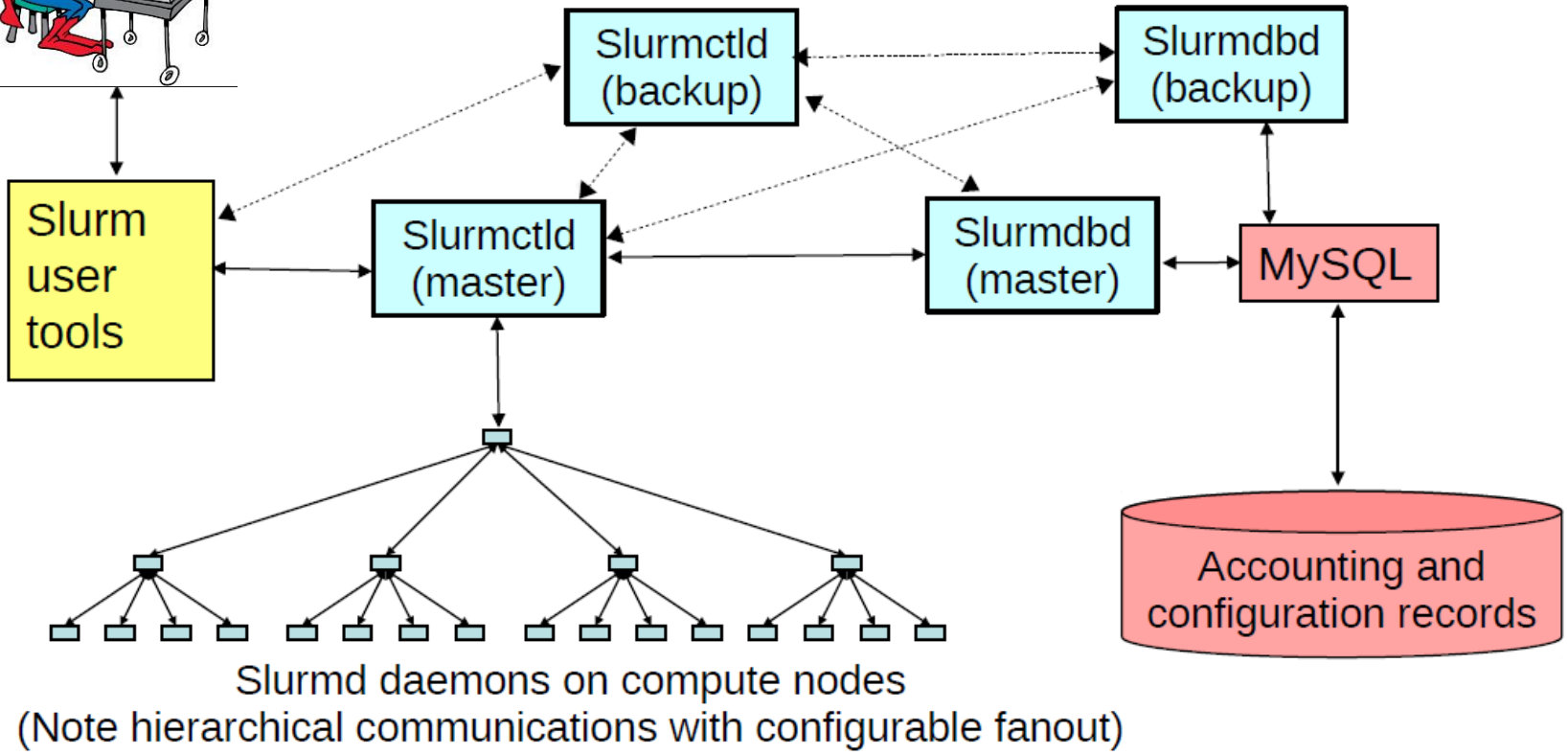
- Monitors state of resources
- Manages job queues
- Allocates resources

slurmdbd – Database daemon (typically one per enterprise)

- Collects accounting information
- Uploads configuration information (limits, fair-share, etc.) to slurmctld

slurmd – Compute node daemon (typically one per compute node, one or more on front-end nodes)

- Launches and manages tasks
- Small and very light-weight (low memory and CPU use)
- Quiescent after launch except for optional accounting
- Supports hierarchical communications with configurable fanout



Slurmd daemons on compute nodes
(Note hierarchical communications with configurable fanout)

SLURM Commands: Job/step Allocation

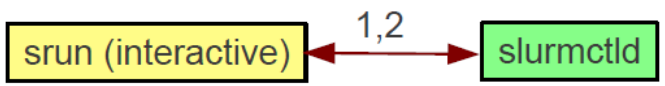
- sbatch – Submit script for later execution (batch mode)
- salloc – Create job allocation and start a shell to use it (interactive mode)
- srun – Create a job allocation (if needed) and launch a job step (typically an MPI job)
- sattach – Connect stdin/out/err for an existing job or job step

MPI Support on Discovery Cluster

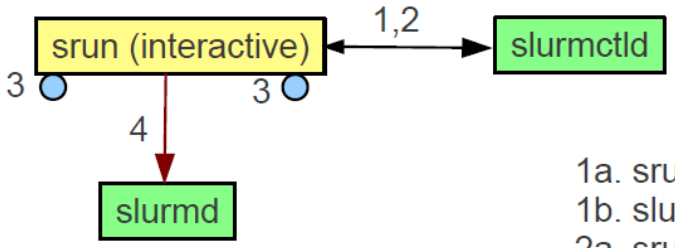
Many different MPI implementations are supported:

- IBM Platform MPI, ANL's MPICH, and OpenMPI are currently supported
- Many use srun to launch the tasks directly
- Some use "mpirun" or another tool within an existing SLURM allocation (they reference SLURM environment variables to determine what resources are allocated to the job)

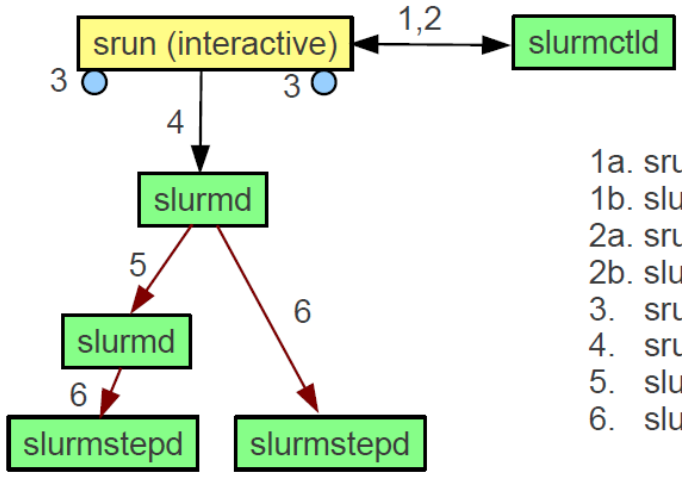
Linux Job Launch Sequence



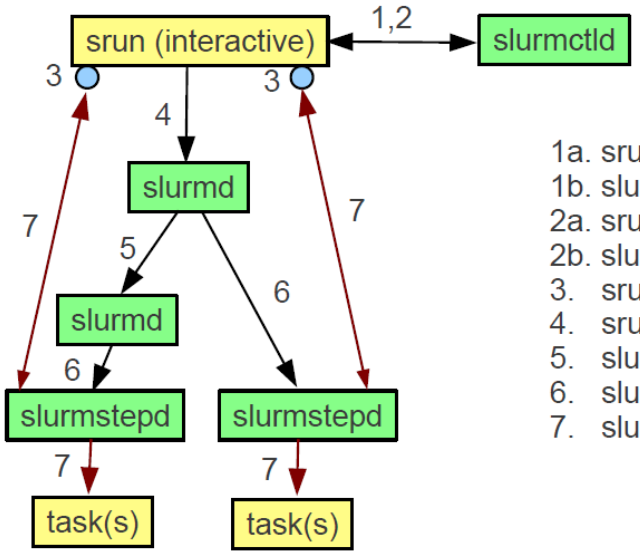
1a. srun sends job allocation request to slurmctld
1b. slurmctld grant allocation and returns details
2a. srun sends step create request to slurmctld
2b. slurmctld responds with step credential



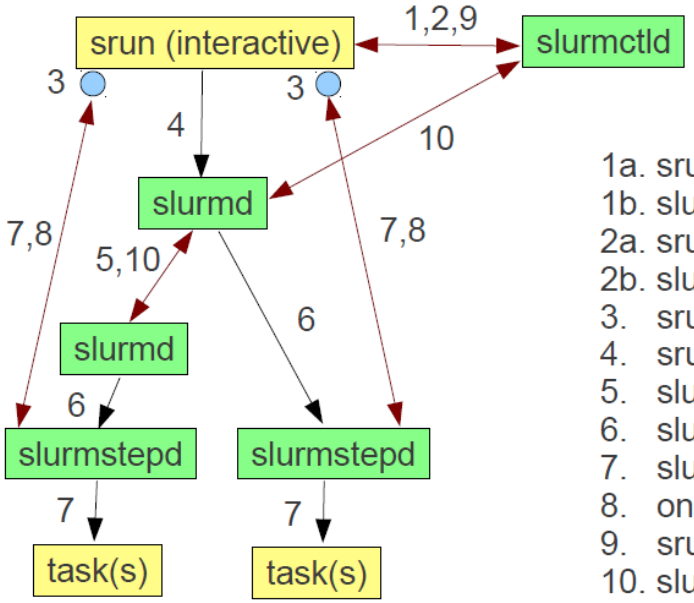
1a. srun sends job allocation request to slurmctld
1b. slurmctld grant allocation and returns details
2a. srun sends step create reqeust to slurmctld
2b. slurmctld responds with step credential
3. srun opens sockets for I/O
4. srun forwards credential with task info to slurmd



1a. srun sends job allocation request to slurmctld
1b. slurmctld grant allocation and returns details
2a. srun sends step create request to slurmctld
2b. slurmctld responds with step credential
3. srun opens sockets for I/O
4. srun forwards credential with task info to slurmd
5. slurmd forward request as needed (per fanout)
6. slurmd forks/execs slurmstepd



1a. srun sends job allocation request to slurmctld
1b. slurmctld grant allocation and returns details
2a. srun sends step create request to slurmctld
2b. slurmctld responds with step credential
3. srun opens sockets for I/O
4. srun forwards credential with task info to slurmd
5. slurmd forward request as needed (per fanout)
6. slurmd forks/execs slurmstepd
7. slurmstepd connects I/O to run & launches tasks



1a. srun sends job allocation request to slurmctld
1b. slurmctld grant allocation and returns details
2a. srun sends step create request to slurmctld
2b. slurmctld responds with step credential
3. srun opens sockets for I/O
4. srun forwards credential with task info to slurmd
5. slurmd forward request as needed (per fanout)
6. slurmd forks/execs slurmstepd
7. slurmstepd connects I/O to run & launches tasks
8. on task termination, slurmstepd notifies srun
9. srun notifies slurmctld of job termination
10. slurmctld verifies termination of all processes via slurmd and releases resources for next job

SLURM Commands: System Information

- sinfo – Report system status (nodes, queues, etc.)
- squeue – Report job and job step status
- smap – Report system, job or step status with topology (curses-based GUI), less functionality than sview
- sview – Report and/or update system, job, step, partition or reservation status with topology (GTKbased GUI)
- scontrol – Administrator tool to view and/or update system, job, step, partition or reservation status

sinfo Command

- Reports status of nodes or partitions
  - Partition-oriented format is the default
- Almost complete control over filtering, sorting and output format is available

```
> sinfo --Node (report status in node-oriented form)
NODELIST    NODES  PARTITION  STATE
tux[000-099]      100   batch       idle
tux[100-127]       28   debug       idle

> sinfo -p debug (report status of nodes in partition “debug”)
PARTITION AVAIL TIMELIMIT NODES NODELIST
debug             up        60:00         28 tux[100-127]

> sinfo -i60   (report status every 60 seconds)
```

```
[nilay.roy@compute-0-004 ~]$
[1]+  Done                    sview
[nilay.roy@compute-0-004 ~]$ exit
logout
Connection to compute-0-004 closed.
[nilay.roy@discovery4 ~]$ exit
exit
salloc: Relinquishing job allocation 120
salloc: Job allocation 120 has been revoked.
[nilay.roy@discovery4 ~]$ sinfo -Nle
Tue Apr 12 11:12:30 2016
NODELIST                     NODES       PARTITION       STATE CPUS    S:C:T MEMORY TMP_DISK WEIGHT FEATURES REASON
compute-0-[000-003,008-063]     60    ser-par-10g*        idle   16    2:8:1      1        0      1   (null) none
compute-0-[000-003]              4 interactive-10g        idle   16    2:8:1      1        0      1   (null) none
compute-0-[004-007]              4          ht-10g        idle   32   2:16:1      1        0      1   (null) none
compute-0-[064-065]              2   ondrechen-10g        idle   40   2:10:2      1        0      1   (null) none
compute-0-[066-095]             30   ser-par-10g-2        idle   40   2:10:2      1        0      1   (null) none
compute-0-[096-143]             48   ser-par-10g-3        idle   40   2:10:2      1        0      1   (null) none
compute-0-[144-327]            184   ser-par-10g-4        idle   48   2:12:2      1        0      1   (null) none
compute-1-[064-127]             64     parallel-ib        idle   16    2:8:1      1        0      1   (null) none
compute-1-[064-067]              4  interactive-ib        idle   16    2:8:1      1        0      1   (null) none
compute-2-[000-003]              4    largemem-10g        idle   32    2:8:2      1        0      1   (null) none
compute-2-[004-006]              3      hadoop-10g        idle   40   2:10:2      1        0      1   (null) none
compute-2-008                    1             suh        idle   32    2:8:2      1        0      1   (null) none
compute-2-[128-159]             32         par-gpu        idle   32    2:8:2      1        0      1   (null) none
compute-2-[160-175]             16       par-gpu-2        idle   48   2:12:2      1        0      1   (null) none
compute-3-[000-031]             32         redwood        idle   20   2:10:1      1        0      1   (null) none
compute-3-[032-039]              8         hanchen        idle   40   2:10:2      1        0      1   (null) none
compute-3-040                    1    westlargemem        idle   40   2:10:2      1        0      1   (null) none
compute-3-[041-047]              7            west        idle   40   2:10:2      1        0      1   (null) none
compute-3-[048-063]             16              fu        idle   40   2:10:2      1        0      1   (null) none
compute-3-[064-079]             16         maloney        idle   40   2:10:2      1        0      1   (null) none
compute-3-080                    1             suh        idle   40   2:10:2      1        0      1   (null) none
compute-3-081                    1         mcguire        idle   40   2:10:2      1        0      1   (null) none
compute-3-[082-084,086]          4         feiguin        idle   40   2:10:2      1        0      1   (null) none
compute-3-085                    1           pizer        idle   40   2:10:2      1        0      1   (null) none
compute-3-087                    1      briesacher        idle   40   2:10:2      1        0      1   (null) none
compute-3-[088-091]              4         griffin        idle   40   2:10:2      1        0      1   (null) none
compute-3-[096-103]              8       ioannidis        idle   40   2:10:2      1        0      1   (null) none
compute-4-000                    1        krioukov        idle   80   4:10:2      1        0      1   (null) none
compute-4-[001-004]              4        lhct3alv        idle    8    2:2:2      1        0      1   (null) none
compute-4-[017-020]              4    krioukov_gpu        idle   48   2:12:2      1        0      1   (null) none
[nilay.roy@discovery4 ~]$
```

squeue Command

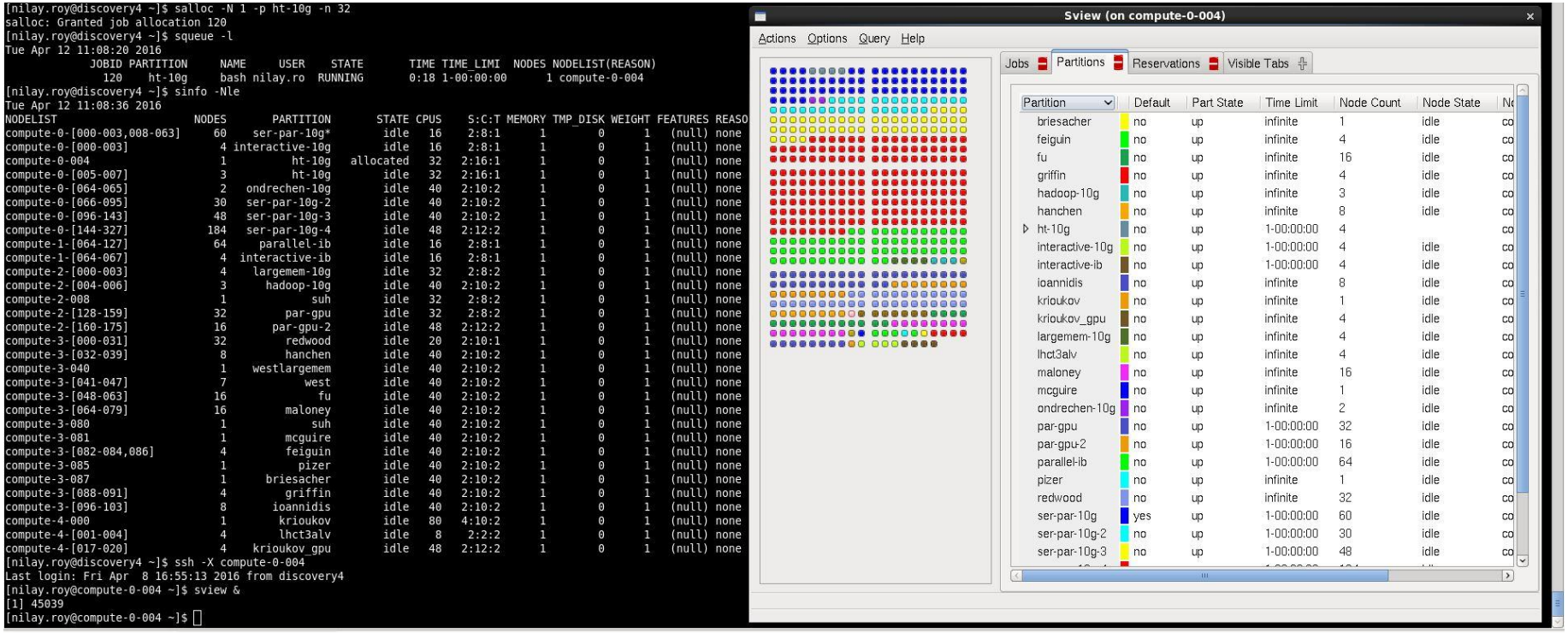
- Reports status of jobs and/or steps in slurmctld daemon's records (recent job's only, older information available in accounting records only)
- Almost complete control over filtering, sorting and output format is available

```
> squeue -u alec -t all (report jobs for user “alec” in any state)
JOBID PARTITION  NAME  USER  ST  TIME  NODES NODELIST(REASON)
45124  debug          a.out    alec     CD   0:12            1 tux123

> squeue -s -p debug  (report steps in partition “debug”);
STEPID PARTITION NAME USER TIME NODELIST
45144.0 debug          a.out    moe   12:18 tux[100-115]

> squeue -i60   (report currently active jobs every 60 seconds)
```

sview


[nilay.roy@discovery4 ~]$ salloc -N 1 -p ht-10g -n 32
salloc: Granted job allocation 120
[nilay.roy@discovery4 ~]$ squeue -l
Tue Apr 12 11:08:20 2016
JOBID PARTITION NAME USER STATE TIME TIME_LIMI NODES NODELIST(REASON)
120 ht-10g bash nilay.ro RUNNING 0:18 1-00:00:00 1 compute-0-004
[nilay.roy@discovery4 ~]$ sinfo -Nle
Tue Apr 12 11:08:36 2016
NODELIST NODES PARTITION STATE CPUS S:C:T MEMORY TMP_DISK WEIGHT FEATURES REASO
compute-0-[000-003,008-063] 60 ser-par-10g* idle 16 2:8:1 1 0 1 (null) none
compute-0-[000-003] 4 interactive-10g idle 16 2:8:1 1 0 1 (null) none
compute-0-004 1 ht-10g allocated 32 2:16:1 1 0 1 (null) none
compute-0-[005-007] 3 ht-10g idle 32 2:16:1 1 0 1 (null) none
compute-0-[064-065] 2 ondrechen-10g idle 40 2:10:2 1 0 1 (null) none
compute-0-[066-095] 30 ser-par-10g-2 idle 40 2:10:2 1 0 1 (null) none
compute-0-[096-143] 48 ser-par-10g-3 idle 40 2:10:2 1 0 1 (null) none
compute-0-[144-327] 184 ser-par-10g-4 idle 48 2:12:2 1 0 1 (null) none
compute-1-[064-127] 64 parallel-ib idle 16 2:8:1 1 0 1 (null) none
compute-1-[064-067] 4 interactive-ib idle 16 2:8:1 1 0 1 (null) none
compute-2-[000-003] 4 largemem-10g idle 32 2:8:2 1 0 1 (null) none
compute-2-[004-006] 3 hadoop-10g idle 40 2:10:2 1 0 1 (null) none
compute-2-008 1 suh idle 32 2:8:2 1 0 1 (null) none
compute-2-[128-159] 32 par-gpu idle 32 2:8:2 1 0 1 (null) none
compute-2-[160-175] 16 par-gpu-2 idle 48 2:12:2 1 0 1 (null) none
compute-3-[000-031] 32 redwood idle 20 2:10:1 1 0 1 (null) none
compute-3-[032-039] 8 hanchen idle 40 2:10:2 1 0 1 (null) none
compute-3-040 1 westlargemem idle 40 2:10:2 1 0 1 (null) none
compute-3-[041-047] 7 west idle 40 2:10:2 1 0 1 (null) none
compute-3-[048-063] 16 fu idle 40 2:10:2 1 0 1 (null) none
compute-3-[064-079] 16 maloney idle 40 2:10:2 1 0 1 (null) none
compute-3-080 1 suh idle 40 2:10:2 1 0 1 (null) none
compute-3-081 1 mcguire idle 40 2:10:2 1 0 1 (null) none
compute-3-[082-084,086] 4 feiguin idle 40 2:10:2 1 0 1 (null) none
compute-3-085 1 pizer idle 40 2:10:2 1 0 1 (null) none
compute-3-087 1 briesacher idle 40 2:10:2 1 0 1 (null) none
compute-3-[088-091] 4 griffin idle 40 2:10:2 1 0 1 (null) none
compute-3-[096-103] 8 ioannidis idle 40 2:10:2 1 0 1 (null) none
compute-4-000 1 krioukov idle 80 4:10:2 1 0 1 (null) none
compute-4-[001-004] 4 lhct3alv idle 8 2:2:2 1 0 1 (null) none
compute-4-[017-020] 4 krioukov_gpu idle 48 2:12:2 1 0 1 (null) none
[nilay.roy@discovery4 ~]$ ssh -X compute-0-004
Last login: Fri Apr 8 16:55:13 2016 from discovery4
[nilay.roy@compute-0-004 ~]$ sview &
[1] 45039
[nilay.roy@compute-0-004 ~]$
Sview (on compute-0-004)
Actions Options Query Help
Jobs Partitions Reservations Visible Tabs
Partition Default Part State Time Limit Node Count Node State
briesacher no up infinite 1 idle
feiguin no up infinite 4 idle
fu no up infinite 16 idle
griffin no up infinite 4 idle
hadoop-10g no up infinite 3 idle
hanchen no up infinite 8 idle
ht-10g no up 1-00:00:00 4
interactive-10g no up 1-00:00:00 4 idle
interactive-ib no up 1-00:00:00 4 idle
ioannidis no up infinite 8 idle
krioukov no up infinite 1 idle
krioukov_gpu no up infinite 4 idle
largemem-10g no up infinite 4 idle
lhct3alv no up infinite 4 idle
maloney no up infinite 16 idle
mcguire no up infinite 1 idle
ondrechen-10g no up infinite 2 idle
par-gpu no up 1-00:00:00 32 idle
par-gpu-2 no up 1-00:00:00 16 idle
parallel-ib no up 1-00:00:00 64 idle
pizer no up infinite 1 idle
redwood no up infinite 32 idle
ser-par-10g yes up 1-00:00:00 60 idle
ser-par-10g-2 no up 1-00:00:00 30 idle
ser-par-10g-3 no up 1-00:00:00 48 idle

scontrol Command

Designed for system administrator use
- Shows all available fields, but no filtering, sorting or formatting options
- Many fields can be modified

```
[nilay.roy@discovery4 ~]$ scontrol show config
Configuration data as of 2016-04-13T17:04:41
AccountingStorageBackupHost = (null)
AccountingStorageEnforce = associations,limits,qos
AccountingStorageHost   = discovery3
AccountingStorageLoc    = N/A
AccountingStoragePort   = 7032
AccountingStorageType   = accounting_storage/slurmdbd
AccountingStorageUser   = N/A
AccountingStoreJobComment = YES
AcctGatherEnergyType    = acct_gather_energy/none
AcctGatherFilesystemType = acct_gather_filesystem/none
AcctGatherInfinibandType = acct_gather_infiniband/none
AcctGatherNodeFreq      = 0 sec
AcctGatherProfileType   = acct_gather_profile/none
AllowSpecResourcesUsage = 0
AuthInfo             = (null)
AuthType             = auth/munge
BackupAddr           = 10.100.8.21
BackupController     = discovery1
BatchStartTimeout    = 10 sec
BOOT_TIME            = 2016-03-22T13:49:36
CacheGroups          = 0
CheckpointType       = checkpoint/none
ChosLoc              = (null)
ClusterName          = discoverycluster
CompleteWait         = 0 sec
ControlAddr          = 10.100.8.23
ControlMachine       = discovery3
CoreSpecPlugin       = core_spec/none
CpuFreqDef           = OnDemand
CryptoType           = crypto/munge
DebugFlags           = (null)
DefMemPerNode        = UNLIMITED
DisableRootJobs      = NO
DynAllocPort         = 0
EnforcePartLimits    = NO
Epilog               = (null)
EpilogMsgTime        = 2000 usec
EpilogSlurmctld      = (null)
ExtSensorsType       = ext_sensors/none
ExtSensorsFreq       = 0 sec
FairShareDampeningFactor = 1
FastSchedule         = 1
FirstJobId           = 1
GetEnvTimeout        = 2 sec
GresTypes            = (null)
GroupUpdateForce     = 0
GroupUpdateTime      = 600 sec
HASH_VAL             = Match
HealthCheckInterval  = 0 sec
HealthCheckNodeState = ANY
HealthCheckProgram   = (null)
InactiveLimit        = 0 sec
JobAcctGatherFrequency  = 30
JobAcctGatherType    = jobacct_gather/linux
JobAcctGatherParams  = (null)
JobCheckpointDir     = /var/slurm/checkpoint
JobCompHost          = discovery3
JobCompLoc           = slurm_acct_db
JobCompPort          = 3306
JobCompType          = jobcomp/mysql
JobCompUser          = SlurmUser
JobContainerType     = job_container/none
JobCredentialPrivateKey = (null)
JobCredentialPublicCertificate = (null)
JobFileAppend        = 0
JobRequeue           = 1
JobSubmitPlugins     = (null)
KeepAliveTime        = SYSTEM_DEFAULT
KillOnBadExit        = 0
KillWait             = 30 sec
LaunchType           = launch/slurm
Layouts              =
Licenses             = MATLAB_Distrib_Comp_Engine:256
LicensesUsed         = MATLAB_Distrib_Comp_Engine:0/256
MailProg             = /bin/mail
MaxArraySize         = 1001
MaxJobCount          = 10000
MaxJobId             = 4294901760
MaxMemPerNode        = UNLIMITED
MaxStepCount         = 40000
MaxTasksPerNode      = 128
MemLimitEnforce      = yes
MessageTimeout       = 100 sec
MinJobAge            = 300 sec
MpiDefault           = none
MpiParams            = (null)
NEXT_JOB_ID          = 138
OverTimeLimit        = 0 min
PluginDir            = /shared/apps/slurm/slurm-14.11.8/INSTALL/lib/slurm
PlugStackConfig      = /shared/apps/slurm/slurm-
14.11.8/INSTALL/etc/plugstack.conf
```

```
PreemptMode           = OFF
PreemptType           = preempt/none
PriorityParameters    = (null)
PriorityDecayHalfLife   = 7-00:00:00
PriorityCalcPeriod    = 00:05:00
PriorityFavorSmall    = 0
PriorityFlags         =
PriorityMaxAge        = 7-00:00:00
PriorityUsageResetPeriod = NONE
PriorityType          = priority/multifactor
PriorityWeightAge     = 0
PriorityWeightFairShare = 0
PriorityWeightJobSize   = 0
PriorityWeightPartition = 0
PriorityWeightQOS     = 0
PrivateData           = none
ProctrackType         = proctrack/cgroup
Prolog                = (null)
PrologSlurmctld       = (null)
PrologFlags           = (null)
PropagatePrioProcess    = 0
PropagateResourceLimits = ALL
PropagateResourceLimitsExcept = (null)
RebootProgram         = (null)
ReconfigFlags         = (null)
RequeueExit           = (null)
RequeueExitHold       = (null)
ResumeProgram         = (null)
ResumeRate            = 300 nodes/min
ResumeTimeout         = 60 sec
ResvEpilog            = (null)
ResvOverRun           = 0 min
ResvProlog            = (null)
ReturnToService       = 1
RoutePlugin           = (null)
SallocDefaultCommand    = (null)
SchedulerParameters     = CR_CORE_MEMORY
SchedulerPort         = 7321
SchedulerRootFilter     = 1
SchedulerTimeSlice      = 30 sec
SchedulerType         = sched/backfill
SelectType            = select/cons_res
SelectTypeParameters    = CR_CPU
SlurmUser             = SlurmUser(510)
SlurmctldDebug        = info
SlurmctldLogFile      = /shared/apps/slurm/slurm-14.11.8/INSTALL/var_log_slurmcrtld
SlurmctldPort         = 111111
SlurmctldTimeout      = 120 sec
SlurmdDebug           = info
SlurmdLogFile         = /shared/apps/slurm/slurm-14.11.8/INSTALL/var_log_slurmd
SlurmdPidFile         = /tmp/slurmd.pid
SlurmdPlugstack       = (null)
SlurmdPort            = 222222
SlurmdSpoolDir        = /shared/apps/slurm/slurm-14.11.8/INSTALL/var_dir/spool
SlurmdTimeout         = 300 sec
SlurmdUser            = root(0)
SlurmSchedLogFile     = (null)
SlurmSchedLogLevel      = 0
SlurmctldPidFile      = /tmp/slurmctld.pid
SlurmctldPlugstack      = (null)
SLURM_CONF            = /shared/apps/slurm/slurm-14.11.8/INSTALL/etc/slurm.conf
SLURM_VERSION         = 14.11.8
SrunEpilog            = (null)
SrunPortRange         = 0-0
SrunProlog            = (null)
StateSaveLocation     = /shared/apps/slurm/slurm-14.11.8/INSTALL/var_dir/spool/statesavelocation
SuspendExcNodes       = (null)
SuspendExcParts       = (null)
SuspendProgram        = (null)
SuspendRate           = 60 nodes/min
SuspendTime           = NONE
SuspendTimeout        = 30 sec
SwitchType            = switch/none
TaskEpilog            = (null)
TaskPlugin            = affinity,cgroup
TaskPluginParam       = (null type)
TaskProlog            = (null)
TmpFS                 = /tmp
TopologyPlugin        = topology/none
TrackWCKey            = 0
TreeWidth             = 50
UsePam                = 0
UnkillableStepProgram   = (null)
UnkillableStepTimeout   = 60 sec
VSizeFactor           = 0 percent
WaitTime              = 0 sec

Slurmctld(primary/backup) at discovery3/discovery1 are UP/UP
[nilay.roy@discovery4 ~]$
```

```
[nilay.roy@discovery4 ~]$ scontrol show licenses
LicenseName=MATLAB_Distrib_Comp_Engine
    Total=256 Used=0 Free=256 Remote=no
[nilay.roy@discovery4 ~]$
```

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.

- Reports accounting information for jobs and steps
- Many filtering and output format options
- Uses accounting file or database (which may not exist depending upon SLURM configuration)

```
> sacct -u joseph (report accounting information for user "joseph")
> sacct -p debug  (report accounting information for partition "debug")
```

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: Other

- scancel – Signal/cancel jobs or job steps
- sbcast – Transfer file to a compute nodes allocated to a job (uses hierarchical communications)
- srun_cr – Wrapper to srun for support of Berkeley checkpoint/restart
- strigger – Event trigger management tools

scancel Command

- Cancel a running or pending job or step
- Can send arbitrary signal to all processes on all nodes associated with a job or step
- Has filtering options (state, user, partition, etc.)
- Has interactive (verify) mode

> scancel 45001.1 (cancel job step 45001.1)
> scancel 45002 (cancel job 45002)
> scancel –user=alec –state=pending (cancel all pending jobs from user “alec”)

sbcast Command

- Copy a file to local disk on allocated nodes
- Execute command after a resource allocation is made
- Data transferred using hierarchical slurmd daemons communications
- May be faster than shared file system

> salloc -N100 bash
salloc: Granted job allocaiton 45201
> sbcast --force my_data /tmp/moe/my_data (overwrite old files)
> srun a.out
> exit (terminate spawned “bash” shell)

TO USE SLURM MUST LOAD SLURM MODULE via .bashrc OR in slurm batch submit scripts

```
[nilay.roy@discovery4 slurm_test]$ module whatis slurm-14.11.8
slurm-14.11.8        : loads the modules environment for SLURM 14.11.8 (http://slurm.schedmd.com/) executibles, libraries, and include files.

Needs the following modules to be loaded as prerequisites

module load gnu-4.4-compilers
module load fftw-3.3.3
module load perl-5.20.0

Put these module load commands in your .bashrc file that is found in your /home/<user-id> directory.



[nilay.roy@discovery4 slurm_test]$ module list
Currently Loaded Modulefiles:
  1) gnu-4.4-compilers   2) fftw-3.3.3          3) perl-5.20.0         4) slurm-14.11.8       5) platform-mpi
[nilay.roy@discovery4 slurm_test]$
```

INTERACTIVE RUNS

STEPS:

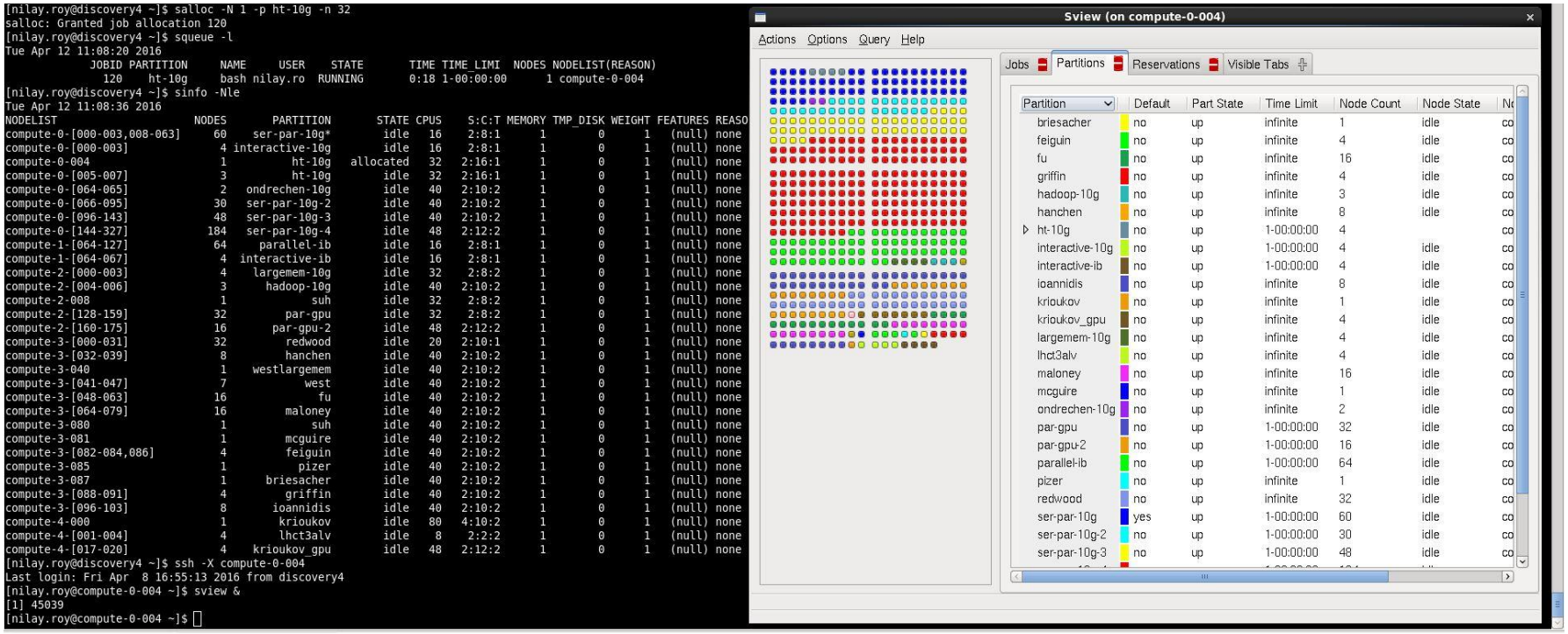
>>sallocate (Run with options to get an allocation from a login node – discovery2/4

>>Find the node the allocation is using “squeue –l”

>>“ssh –X” to that node

>>When done logout of that node, and then “exit” from the allocation

>>Check allocation is released



```
[nilay.roy@compute-0-004 ~]$
[1]+  Done                    sview
[nilay.roy@compute-0-004 ~]$ exit
logout
Connection to compute-0-004 closed.
[nilay.roy@discovery4 ~]$ exit
exit
salloc: Relinquishing job allocation 120
salloc: Job allocation 120 has been revoked.
[nilay.roy@discovery4 ~]$ sinfo -Nle
Tue Apr 12 11:12:30 2016
NODELIST                    NODES       PARTITION       STATE CPUS    S:C:T MEMORY TMP_DISK WEIGHT FEATURES REASON
compute-0-[000-003,008-063]     60    ser-par-10g*        idle   16    2:8:1      1        0      1   (null) none
compute-0-[000-003]              4 interactive-10g        idle   16    2:8:1      1        0      1   (null) none
compute-0-[004-007]              4          ht-10g        idle   32   2:16:1      1        0      1   (null) none
compute-0-[064-065]              2   ondrechen-10g        idle   40   2:10:2      1        0      1   (null) none
compute-0-[066-095]             30   ser-par-10g-2        idle   40   2:10:2      1        0      1   (null) none
compute-0-[096-143]             48   ser-par-10g-3        idle   40   2:10:2      1        0      1   (null) none
compute-0-[144-327]            184   ser-par-10g-4        idle   48   2:12:2      1        0      1   (null) none
compute-1-[064-127]             64     parallel-ib        idle   16    2:8:1      1        0      1   (null) none
compute-1-[064-067]              4  interactive-ib        idle   16    2:8:1      1        0      1   (null) none
compute-2-[000-003]              4    largemem-10g        idle   32    2:8:2      1        0      1   (null) none
compute-2-[004-006]              3      hadoop-10g        idle   40   2:10:2      1        0      1   (null) none
compute-2-008                    1             suh        idle   32    2:8:2      1        0      1   (null) none
compute-2-[128-159]             32         par-gpu        idle   32    2:8:2      1        0      1   (null) none
compute-2-[160-175]             16       par-gpu-2        idle   48   2:12:2      1        0      1   (null) none
compute-3-[000-031]             32         redwood        idle   20   2:10:1      1        0      1   (null) none
compute-3-[032-039]              8         hanchen        idle   40   2:10:2      1        0      1   (null) none
compute-3-040                    1    westlargemem        idle   40   2:10:2      1        0      1   (null) none
compute-3-[041-047]              7            west        idle   40   2:10:2      1        0      1   (null) none
compute-3-[048-063]             16              fu        idle   40   2:10:2      1        0      1   (null) none
compute-3-[064-079]             16         maloney        idle   40   2:10:2      1        0      1   (null) none
compute-3-080                    1             suh        idle   40   2:10:2      1        0      1   (null) none
compute-3-081                    1         mcguire        idle   40   2:10:2      1        0      1   (null) none
compute-3-[082-084,086]          4         feiguin        idle   40   2:10:2      1        0      1   (null) none
compute-3-085                    1           pizer        idle   40   2:10:2      1        0      1   (null) none
compute-3-087                    1      briesacher        idle   40   2:10:2      1        0      1   (null) none
compute-3-[088-091]              4         griffin        idle   40   2:10:2      1        0      1   (null) none
compute-3-[096-103]              8       ioannidis        idle   40   2:10:2      1        0      1   (null) none
compute-4-000                    1        krioukov        idle   80   4:10:2      1        0      1   (null) none
compute-4-[001-004]              4        lhct3alv        idle    8    2:2:2      1        0      1   (null) none
compute-4-[017-020]              4    krioukov_gpu        idle   48   2:12:2      1        0      1   (null) none
[nilay.roy@discovery4 ~]$
```

BATCH RUNS

SUBMIT USING SUBMIT SCRIPT

```
[nilay.roy@discovery4 slurm_test]$ sbatch slurm_submit.sbatch
Submitted batch job 137
[nilay.roy@discovery4 slurm_test]$ squeue -l
Tue Apr 12 17:41:31 2016
             JOBID PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
               137 ser-par-1     run1 nilay.ro  RUNNING       0:03     20:00      2 compute-0-[002-003]
[nilay.roy@discovery4 slurm_test]$
```

ANALYSIS OF A TYPICAL SUBMIT SCRIPT – MPI USE CASE IS IBM PLATFORM MPI

```
[nilay.roy@discovery4 slurm_test]$ cat slurm_submit.sbatch
#!/bin/bash
#set a job name
#SBATCH --job-name=run1
#################
#a file for job output, you can check job progress
#SBATCH --output=run1.out
#################
# a file for errors from the job
#SBATCH --error=run1.err
#################
#time you think you need; default is one day
#in minutes in this case, hh:mm:ss
#SBATCH --time=20:00
#################
#number of tasks you are requesting
#SBATCH -n 32
#SBATCH --exclusive
#################
#partition to use
#SBATCH --partition=ser-par-10g
#################
#number of nodes to distribute n tasks across
#SBATCH -N 2
#################

work=/gss_gpfs_scratch/nroy/slurm_test

cd $work

mpirun -prot -srun  ./mpi_mm
[nilay.roy@discovery4 slurm_test]$
```

FIND PARTITION AND NODE INFORMATION TO DECIDE WHICH QUEUE TO USE

```
[nilay.roy@discovery4 slurm_test]$ sinfo -l -p ser-par-10g-4
Tue Apr 12 17:35:27 2016
PARTITION     AVAIL  TIMELIMIT   JOB_SIZE ROOT    SHARE     GROUPS  NODES       STATE NODELIST
ser-par-10g-4    up 1-00:00:00       1-90   no       NO        all    184        idle compute-0-[144-327]
[nilay.roy@discovery4 slurm_test]$ sinfo -l -p ser-par-10g-3
Tue Apr 12 17:35:31 2016
PARTITION     AVAIL  TIMELIMIT   JOB_SIZE ROOT    SHARE     GROUPS  NODES       STATE NODELIST
ser-par-10g-3    up 1-00:00:00       1-22   no       NO        all     48        idle compute-0-[096-143]
[nilay.roy@discovery4 slurm_test]$ sinfo -l -p ser-par-10g-2
Tue Apr 12 17:35:35 2016
PARTITION     AVAIL  TIMELIMIT   JOB_SIZE ROOT    SHARE     GROUPS  NODES       STATE NODELIST
ser-par-10g-2    up 1-00:00:00       1-14   no       NO        all     30        idle compute-0-[066-095]
[nilay.roy@discovery4 slurm_test]$ sinfo -l -p ser-par-10g
Tue Apr 12 17:35:37 2016
PARTITION    AVAIL  TIMELIMIT   JOB_SIZE ROOT    SHARE     GROUPS  NODES       STATE NODELIST
ser-par-10g*    up 1-00:00:00       1-32   no       NO        all     60        idle compute-0-[000-003,008-063]
[nilay.roy@discovery4 slurm_test]$ sinfo -l -p ht-10g
Tue Apr 12 17:35:43 2016
PARTITION AVAIL  TIMELIMIT   JOB_SIZE ROOT    SHARE     GROUPS  NODES       STATE NODELIST
ht-10g       up 1-00:00:00          1   no       NO        all      4        idle compute-0-[004-007]
[nilay.roy@discovery4 slurm_test]$
```

Check RUN results

```
[nilay.roy@discovery4 slurm_test]$ ls -la
total 2048
drwxr-xr-x 2 nilay.roy GID_nilay.roy   4096 Apr 12 17:41 .
drwxr-xr-x 3 nilay.roy GID_nilay.roy   4096 Apr 12 17:39 ..
-rw-r--r-- 1 nilay.roy GID_nilay.roy    135 Apr 12 10:50 Makefile
-rwxr-xr-x 1 nilay.roy GID_nilay.roy  10060 Apr 12 10:50 mpi_mm
-rw-r--r-- 1 nilay.roy GID_nilay.roy   4018 Apr 12 10:50 mpi_mm.c
-rw-r--r-- 1 nilay.roy GID_nilay.roy      0 Apr 12 17:43 run1.err
-rw-r--r-- 1 nilay.roy GID_nilay.roy 710759 Apr 12 17:43 run1.out
-rwxr--r-- 1 nilay.roy GID_nilay.roy    677 Apr 12 17:12 slurm_submit.sbatch
[nilay.roy@discovery4 slurm_test]$
```

```
[nilay.roy@discovery4 slurm_test]$ cat Makefile
CC          =  mpicc
OBJ         =  mpi_mm
SRC         =  mpi_mm.c
LIBS        =

$(OBJ):  $(SRC)
        $(CC) $(SRC) -o $(OBJ) $(LIBS)

[nilay.roy@discovery4 slurm_test]$ module list
Currently Loaded Modulefiles:
  1) gnu-4.4-compilers   2) fftw-3.3.3         3) perl-5.20.0        4) slurm-14.11.8      5) platform-mpi
[nilay.roy@discovery4 slurm_test]$
```

SOURCE CODE ON NEXT SLIDE

```
[nilay.roy@discovery4 slurm_test]$ head -74 run1.out
Host 0 -- ip 10.100.8.42 -- ranks 0 - 15
task ID = 16
task ID = 18
task ID = 29
task ID = 31
task ID = 26
task ID = 24
task ID = 28
task ID = 27
task ID = 25
task ID = 30
task ID = 19
task ID = 17
task ID = 23
task ID = 21
task ID = 22
task ID = 20
Host 1 -- ip 10.100.8.43 -- ranks 16 - 31

 host | 0    1
======|==========
    0 : SHM  TCP
    1 : TCP  SHM

 Prot -  All Intra-node communication is: SHM
 Prot -  All Inter-node communication is: TCP

task ID = 10
task ID = 9
task ID = 11
task ID = 8
task ID = 5
task ID = 7
task ID = 13
task ID = 3
task ID = 15
task ID = 1
task ID = 14
task ID = 12
task ID = 2
task ID = 4
task ID = 6
task ID = 0
Number of worker tasks = 31
   sending 17 rows to task 1
   sending 17 rows to task 2
   sending 17 rows to task 3
   sending 17 rows to task 4
   sending 16 rows to task 5
   sending 16 rows to task 6
   sending 16 rows to task 7
   sending 16 rows to task 8
   sending 16 rows to task 9
   sending 16 rows to task 10
   sending 16 rows to task 11
   sending 16 rows to task 12
   sending 16 rows to task 13
   sending 16 rows to task 14
   sending 16 rows to task 15
   sending 16 rows to task 16
   sending 16 rows to task 17
   sending 16 rows to task 18
   sending 16 rows to task 19
   sending 16 rows to task 20
   sending 16 rows to task 21
   sending 16 rows to task 22
   sending 16 rows to task 23
   sending 16 rows to task 24
   sending 16 rows to task 25
   sending 16 rows to task 26
   sending 16 rows to task 27
   sending 16 rows to task 28
   sending 16 rows to task 29
   sending 16 rows to task 30
[nilay.roy@discovery4 slurm_test]$
```

```
[nilay.roy@discovery4 slurm_test]$ cat mpi_mm.c
#include "mpi.h"
#include <stdio.h>
#define NRA 500                 /* number of rows in matrix A */
#define NCA 100                 /* number of columns in matrix A */
#define NCB 100                 /* number of columns in matrix B */
#define MASTER 0                /* taskid of first task */
#define FROM_MASTER 1           /* setting a message type */
#define FROM_WORKER 2           /* setting a message type */

int main(argc,argv)
int argc;
char *argv[];
{
int                     numtasks,           /* number of tasks in partition */
                        taskid,             /* a task identifier */
                        numworkers,         /* number of worker tasks */
                        source,             /* task id of message source */
                        dest,               /* task id of message destination */
                        mtype,              /* message type */
                        rows,               /* rows of matrix A sent to each worker */
                        averow, extra, offset, /* used to determine rows sent to each worker */
                        i, j, k, rc;        /* misc */
double                  a[NRA][NCA],        /* matrix A to be multiplied */
                        b[NCA][NCB],        /* matrix B to be multiplied */
                        c[NRA][NCB];        /* result matrix C */
MPI_Status status;

rc = MPI_Init(&argc,&argv);
  rc|= MPI_Comm_size(MPI_COMM_WORLD,&numtasks);
  rc|= MPI_Comm_rank(MPI_COMM_WORLD,&taskid);
  if (rc != 0)
    printf ("error initializing MPI and obtaining task ID information\n");
  else
    printf ("task ID = %d\n", taskid);
  numworkers = numtasks-1;
/************************** master task ************************************/
  if (taskid == MASTER)
  {
    printf("Number of worker tasks = %d\n",numworkers);
    for (i=0; i<NRA; i++)
      for (j=0; j<NCA; j++)
        a[i][j]= i+j;
    for (i=0; i<NCA; i++)
      for (j=0; j<NCB; j++)
        b[i][j]= i*j;
    /* send matrix data to the worker tasks */
    averow = NRA/numworkers;
    extra = NRA%numworkers;
    offset = 0;
    mtype = FROM_MASTER;

for (dest=1; dest<=numworkers; dest++)
    {
rows = (dest <= extra) ? averow+1 : averow;
      printf("   sending %d rows to task %d\n",rows,dest);
      MPI_Send(&offset, 1, MPI_INT, dest, mtype, MPI_COMM_WORLD);
      MPI_Send(&rows, 1, MPI_INT, dest, mtype, MPI_COMM_WORLD);
      MPI_Send(&a[offset][0], rows*NCA, MPI_DOUBLE, dest, mtype,
               MPI_COMM_WORLD);
      MPI_Send(&b, NCA*NCB, MPI_DOUBLE, dest, mtype, MPI_COMM_WORLD);
      offset = offset + rows;
    }
    /* wait for results from all worker tasks */
    mtype = FROM_WORKER;
    for (i=1; i<=numworkers; i++)
    {
      source = i;
      MPI_Recv(&offset, 1, MPI_INT, source, mtype, MPI_COMM_WORLD, &status);
      MPI_Recv(&rows, 1, MPI_INT, source, mtype, MPI_COMM_WORLD, &status);
      MPI_Recv(&c[offset][0], rows*NCB, MPI_DOUBLE, source, mtype, MPI_COMM_WORLD, &status);
    }

    /* print results */
    printf("Here is the result matrix\n");
    for (i=0; i<NRA; i++)
    {
      printf("\n");
      for (j=0; j<NCB; j++)
        printf("%6.2f   ", c[i][j]);
    }
    printf ("\n");
  }
/************************** worker task ************************************/
  if (taskid > MASTER)
  {
    mtype = FROM_MASTER;
    MPI_Recv(&offset, 1, MPI_INT, MASTER, mtype, MPI_COMM_WORLD, &status);
    MPI_Recv(&rows, 1, MPI_INT, MASTER, mtype, MPI_COMM_WORLD, &status);
    MPI_Recv(&a, rows*NCA, MPI_DOUBLE, MASTER, mtype, MPI_COMM_WORLD, &status);
    MPI_Recv(&b, NCA*NCB, MPI_DOUBLE, MASTER, mtype, MPI_COMM_WORLD, &status);

    for (k=0; k<NCB; k++)
      for (i=0; i<rows; i++)
      {
        c[i][k] = 0.0;
        for (j=0; j<NCA; j++)
          c[i][k] = c[i][k] + a[i][j] * b[j][k];
      }
    mtype = FROM_WORKER;
    MPI_Send(&offset, 1, MPI_INT, MASTER, mtype, MPI_COMM_WORLD);
    MPI_Send(&rows, 1, MPI_INT, MASTER, mtype, MPI_COMM_WORLD);
    MPI_Send(&c, rows*NCB, MPI_DOUBLE, MASTER, mtype, MPI_COMM_WORLD);
  }
  MPI_Finalize();
}

[nilay.roy@discovery4 slurm_test]$
```

SOME OTHER TIPS

How do I know to which partition I should submit my job so that it starts as early as possible?

Simply submit the job to all partitions you are considering, by listing them in the `--partition` argument:

```
#SBATCH --partition=partition1,partition2
```

The job will submitted to the partition which offers the earliest allocation according to your job parameters and priority.

How do I use the local scratch space ?

Slurm offer the sbcast command that propagates a file to the local file systems of the nodes that were allocated to the job. However, sbcast works one file at a time. It is therefore unsuited for copying entire data directories for instance.

One neat way is to use a construction like

```
srun cp
```

For instance, in the script below

```
#!/bin/bash
#SBATCH -N 2
#SBATCH -o output.txt
SCRATCH=/scratch/$USER/$SLURM_JOB_ID

echo Creating temp dir $SCRATCH
srun mkdir -p $SCRATCH || exit $?
echo Coping files. srun cp is equivalent to loop over each node + scp
srun cp -r $SLURM_SUBMIT_DIR/*  $SCRATCH || exit $?
```

the data are copied from the home to the local scratch (assuming the home directory is mounted on each compute node). A directory is created with the login and the job ID.

If each result file has a distinct name, we can simply `srun cp` from the scratch to the home.

At the end, make sure to clean the scratch space.

```
echo Removing $SCRATCH
srun rm -rf $SCRATCH || exit $?
```

How do I know which slots exactly are assigned to my job ?

The command `scontrol show -d job jobid` gives very detailed information about jobs.

When will my job start ?

The squeue --start command gives an estimation of the date and time a job is supposed to start but beware that the estimation is based on the environment at a given time. Slurm cannot anticipate higher-priority jobs being submitted after yours, or machine downtimes which lead to fewer resources for the jobs, of job crashes which can lead to large jobs starting earlier than expected and smaller jobs scheduled for backfilling to loose that backfilling opportunity.

How do I cancel a job?

Use the `scancel jobid` command with the jobid of the job you want canceled. In the case you want to cancel all your jobs, type `scancel -u login`. You can also cancel all your pending jobs for instance with `scancel -t PD`.

How do I know how much memory my job is using/has used ?

If your job is still running, you can have memory information through `sstat`. If your job is done, the information is provided by `sacct`. Both support the `--format` option so you can run, for instance:

```
sacct --format JobID,jobname,NTasks,nodelist,MaxRSS,MaxVMSize,AveRSS,AveVMSize
```

See the manpages for both utilities

```
man sstat
man sacct
```

Priority computation `priority/multifactor`.

The priority then depends on five elements:

- Job age: how long the job has been waiting in the queue ;
- User fairshare: a measure of past usage of the cluster by the user ;
- Job size: the number of CPUs a job requests ;
- Partition: the partition to which a job is submitted , specified with the `--partition` submission parameter;
- QOS: a quality of service associated with the job, specified with the `--qos` submission parameter.

Note that the job age parameter is bounded so that priority stops increasing when the bound is attained. The job size parameter can be configured to favor small or large jobs, although it is used most of the time to favor large jobs. The faishare parameter has a 'forgetting' parameter that leads to considering only the recent history of the user and not its total use over the time life of the cluster.

All these are combined in a weighted average to form the priority. The weights can be found by running

```
sprio -w
```

A detailed description of how these are computed (including the fairshare), is given in the Slurm documentation for multifactor and for multifactor2.

The precise configuration for a cluster can be found by running the following command:

```
scontrol show config | grep ^Priority
```

Finding a user's current fairshare situation is done with the sshare command.

Getting the priority given to a job can be done either with squeue

```
squeue -o %Q -j jobid
```

or with the sprio command which gives the details of the computation.

THANK YOU

QUESTIONS