# LONI QB3
# User Environment --
## Job Submission using Slurm

Feng Chen
HPC User Services
LSU HPC & LONI
sys-help@loni.org

Louisiana State University
Baton Rouge
October 1, 2020

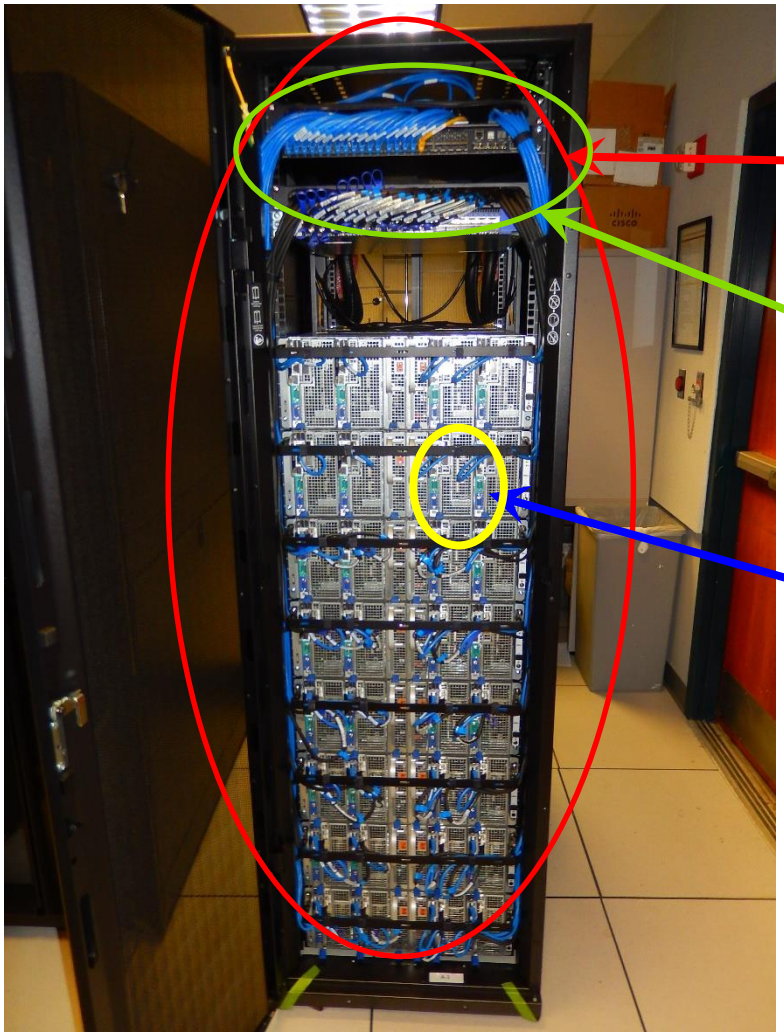
# Outline

- **Things to be covered in this section**
  - QB3 job queues
  - Job submission on QB3
    - Interactive and batch jobs using Slurm
    - Serial and parallel jobs
    - Monitor your jobs

*QB3 HPC User Environment*

# Overview of QB3

# Inside A Cluster Rack

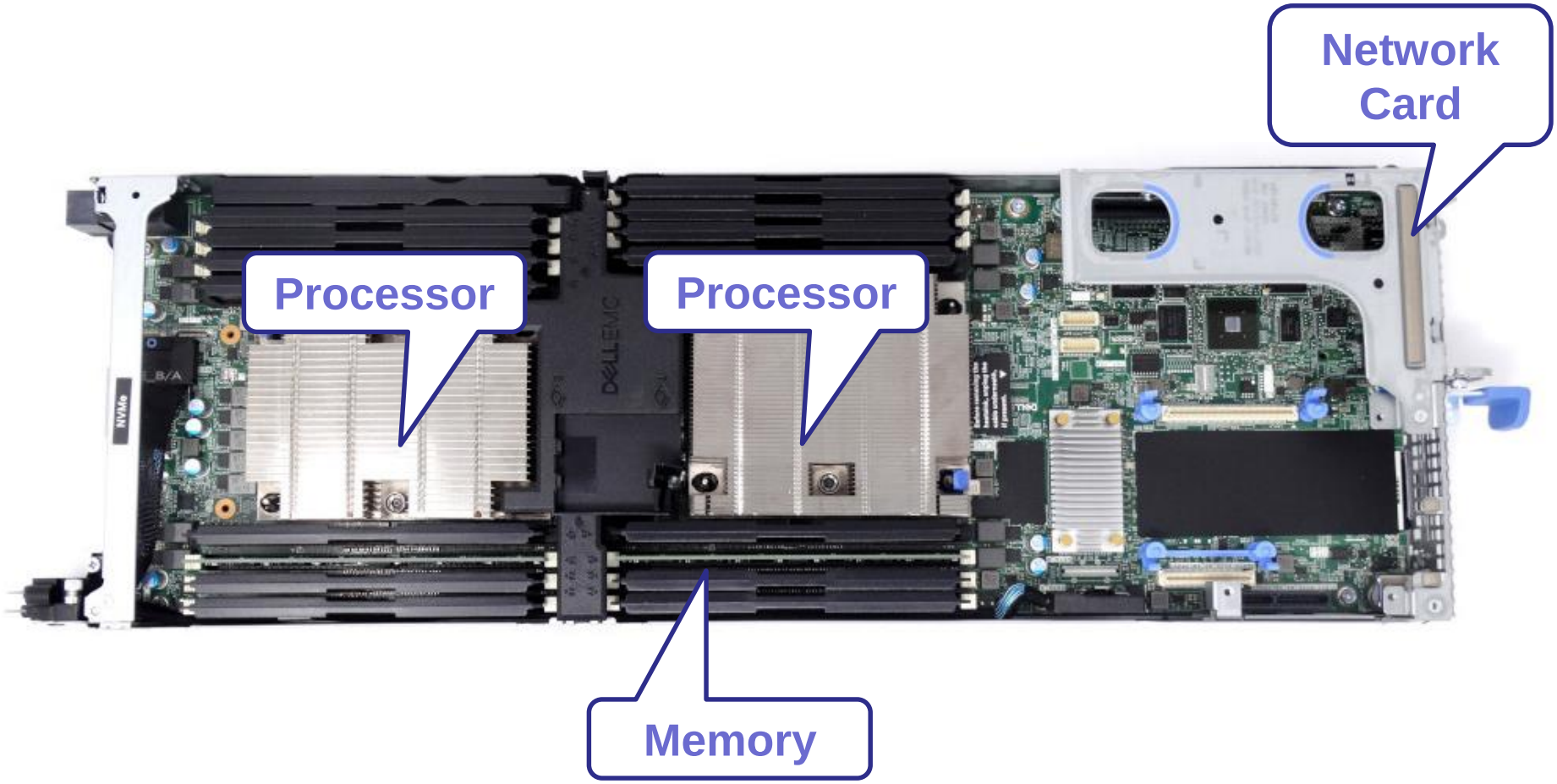


**Rack**

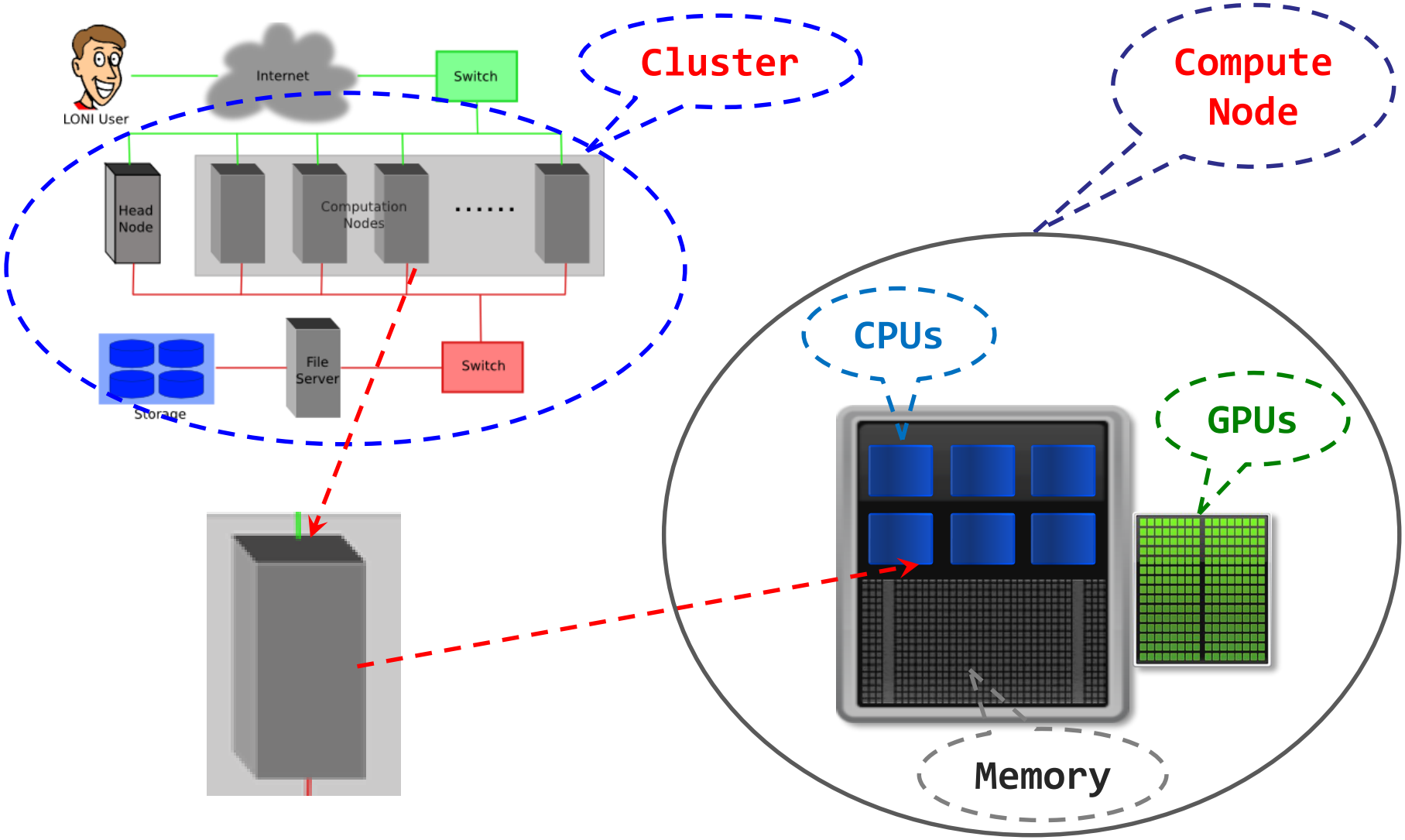
**Infiniband Switch**

**Compute Node**

# Inside A QB3 Compute Node



# Conceptual Relationship



# Summary of QB3

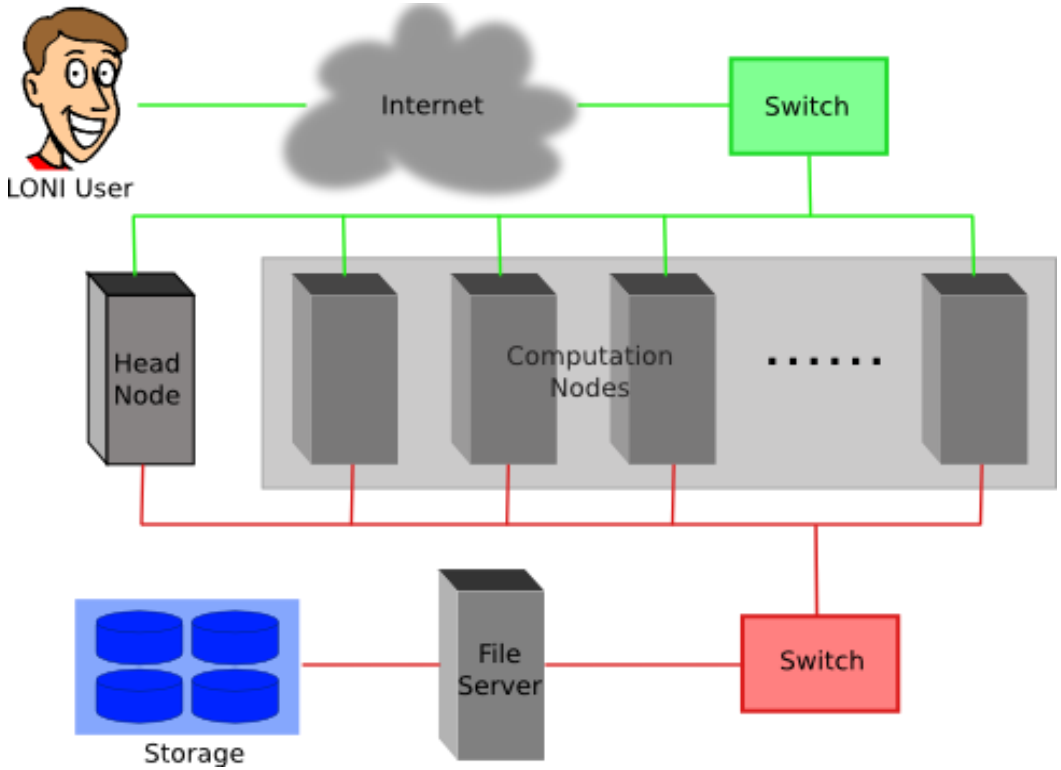
- **QB3 is an 857 TeraFlop peak performance cluster with 9,696 CPU cores, comprised of 202 compute nodes connected by 100 Gbps Infiniband fabric**
  - 192 regular nodes: two 24-core Intel Cascade Lake CPUs, 192 GB RAM
  - 8 GPU nodes: two 24-core Intel Cascade Lake CPUs, 192 GB RAM, two NVIDIA Tesla V100 GPUs
  - 2 bigmem nodes: two 24-core Intel Cascade Lake CPUs, 1.5 TB RAM
- **log in QB3 with your current LONI HPC credentials using**
  - ssh qbc.loni.org
- **Before you submit jobs on QB3, please make sure that you review the user guide here:**
  - http://www.hpc.lsu.edu/docs/guides.php?system=QB3
- **The biggest difference QB2 users would notice on QB3 is that, instead of Torque/Moab (PBS), *Slurm* is employed as the workload and resource manager.**

*QB3 HPC User Environment*

# QB3 Job Queues (Partitions)

# Cluster Environment

- **Multiple compute nodes**
- **Multiple users**
- **Each user may have multiple jobs running simultaneously**
- **Multiple users may share the same node**



# Job submission basics

1. **Find appropriate queue (called “partition” in Slurm)**

2. **Understand the queuing system and your requirements and proceed to submit jobs**

3. **Monitor jobs during execution**

# Job Queues (Partitions)

- **Nodes are organized into queues. Nodes can be shared.**
  - On QB3, the default is the “single” queue
- **Each job queue (partition) differs in**
  - Number of available nodes
  - Max run time
  - Max running jobs per user
  - Nodes may have special characteristics: GPU, Large memory, etc.
- **Jobs need to specify resource requirements**
  - Nodes, time, queue
- **It’s called a queue for a reason, but jobs don’t run on a “First Come First Served” policy,**
  - This will be detailed in later slides

# Queue Characteristics – LONI clusters

| Machine | Queue | Max Runtime (h) | ppn | Max nodes per job |
|---|---|---|---|---|
| QB2 | workq | 72 | 20 | 128 |
| | checkpt | | 20 | 256 |
| | single | 168 | 1,2,4,6,8 | 1 |
| | bigmem | 72 | 48 | 1 |
| **Machine** | **Partition** | **Max Runtime (h)** | **cores per node** | **Max nodes per job** |
| QB3 | workq | 72 | 48 | 96 |
| | checkpt | 72 | 48 | 96 |
| | single | 168 | 1-47 | 1 |
| | gpu | 72 | 48 | 4 |
| | bigmem | 72 | 48 | 1 |

- By default, you job will be submitted to “single” queue (partition).

# Queue Characteristics

- **"sinfo" will give you more info on the queues**

```
[fchen14@qbc2 slurmdoc]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
admin        up   infinite      4    mix qbc[001-003,192]
admin        up   infinite    197   idle qbc[004-190,193-202]
admin        up   infinite      1   down qbc191
single*      up 3-00:00:00      4    mix qbc[001-003,192]
single*      up 3-00:00:00    187   idle qbc[004-190]
single*      up 3-00:00:00      1   down qbc191
checkpt      up 3-00:00:00      4    mix qbc[001-003,192]
checkpt      up 3-00:00:00    187   idle qbc[004-190]
checkpt      up 3-00:00:00      1   down qbc191
workq        up 3-00:00:00      4    mix qbc[001-003,192]
workq        up 3-00:00:00    187   idle qbc[004-190]
workq        up 3-00:00:00      1   down qbc191
gpu          up 3-00:00:00      8   idle qbc[193-200]
bigmem       up 3-00:00:00      2   idle qbc[201-202]
```

# Queue Querying – Linux Clusters

- **Displays information about active, eligible, blocked, and/or recently completed jobs: `showq` command**

```
[fchen14@qbc2 slurmdoc]$ showq
ACTIVE JOBS--------------------
JOBID   JOBNAME    USERNAME  STATE   CORE REMAINING  STARTTIME
================================================================================
37789   qbc_ch4_ti madibi    Running 48   45:45:17  Thu Sep 24 22:06:18
37799   qbc_ch4_ti madibi    Running 48   61:27:15  Fri Sep 25 13:48:16
37830   t1         klhu      Running 48   63:20:02  Fri Sep 25 15:41:03
37845   t10        klhu      Running 48   67:58:08  Fri Sep 25 20:19:09
37846   t11        klhu      Running 48   67:58:15  Fri Sep 25 20:19:16

      5 active jobs

Total Jobs: 5     Active Jobs: 5     Idle Jobs: 0     Blocked Jobs: 0
```

*QB3 HPC User Environment*

# Job Submission Through Slurm

# PBS to Slurm

- **Why Slurm?**
  - Slurm has a more open model
  - Slurm also feels more modern in its design and implementation
  - Slurm scales well, job starts faster, etc.

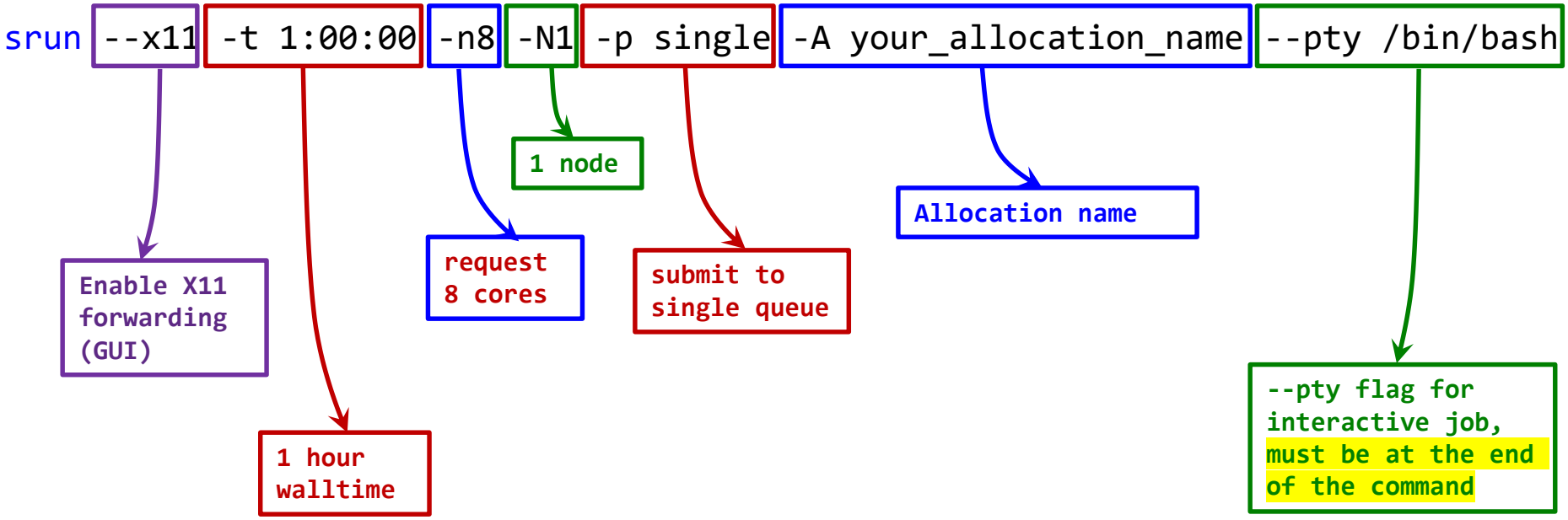
- **What we will cover in this session?**
  - Basic Slurm job submission
    - Interactive job
    - Batch job

# Two Job Types

- **Interactive job**
  - User can interact with the terminal on the compute node.
  - Interactive jobs can be used for ***testing and troubleshooting*** code.
  - Requesting an interactive job will allocate resources and log you into a shell on a compute node.
- **Batch job**
  - Executed without user intervention using a job script
  - Batch jobs are used for production runs.
  - Requesting a batch job will allocate resources and execute the job commands in a non-interactive shell

# Slurm Interactive Job Command

- **To start an interactive job, use the `srun` command like the example below:**



# Check Available Allocations

```
[fchen14.fchen14-t460] ➤ ssh fchen14@qbc.loni.org
Warning: Permanently added 'qbc2.loni.org' (RSA) to the list of known hosts.
Last login: Thu Sep 17 11:41:42 2020 from crimson.its.lsu.edu
##########################################################################
Send questions and comments to the email ticket system at sys-help@loni.org.
##########################################################################
...Message Of The Day...
[fchen14@qbc2 slurmdoc]$ showquota
Hard disk quotas for user fchen14 (uid 32584):
     Filesystem    MB used       quota       files      fquota
     /home            6440       10000      110083           0
     /work           19419           0       48919     4000000

CPU Allocation SUs remaining:
    loni_loniadmin1:       521749.16
    loni_train_2020:        37208.87    50000.00  2021-04-01
```

# Start an Slurm Interactive Job

```
[fchen14@qbc2 ~]$ srun --x11 -t 1:00:00 -n8 -N1 -p single -A loni_train_2020 --pty /bin/bash
[fchen14@qbc192 ~]$ hostname # verify that you are on a compute node
qbc192
[fchen14@qbc192 ~]$ some_job_commands # your own job commands
```

# Slurm Environmental Variables

```
[fchen14@qbc2 slurmdoc]$ srun --x11 -t 1:00:00 -n8 -N1 -p single -A
loni_loniadmin1 --pty /bin/bash
[fchen14@qbc198 slurmdoc]$ echo $SLURM_
$SLURM_CLUSTER_NAME             $SLURM_JOB_NAME
$SLURM_NPROCS                   $SLURM_STEP_NODELIST
$SLURM_CPU_BIND                 $SLURM_JOB_NODELIST
$SLURM_NTASKS                   $SLURM_STEP_NUM_NODES
$SLURM_CPU_BIND_LIST            $SLURM_JOB_NUM_NODES
$SLURM_PRIO_PROCESS             $SLURM_STEP_NUM_TASKS
$SLURM_CPU_BIND_TYPE            $SLURM_JOB_PARTITION
$SLURM_PROCID                   $SLURM_STEP_TASKS_PER_NODE
...
$SLURM_JOB_GID                  $SLURM_NNODES
$SLURM_STEPID                   $SLURM_TOPOLOGY_ADDR_PATTERN
$SLURM_JOBID                    $SLURM_NODEID
$SLURM_STEP_ID                  $SLURM_UMASK
$SLURM_JOB_ID                   $SLURM_NODELIST
$SLURM_STEP_LAUNCHER_PORT       $SLURM_WORKING_CLUSTER
```

# Slurm Batch Job Script

- **To create a batch Slurm script, use your favorite editor (e.g. vi or emacs, nano) to create a text file with both Slurm instructions and commands how to run your job.**
- **All Slurm directives (special instructions) are prefaced by the `#SBATCH`.**

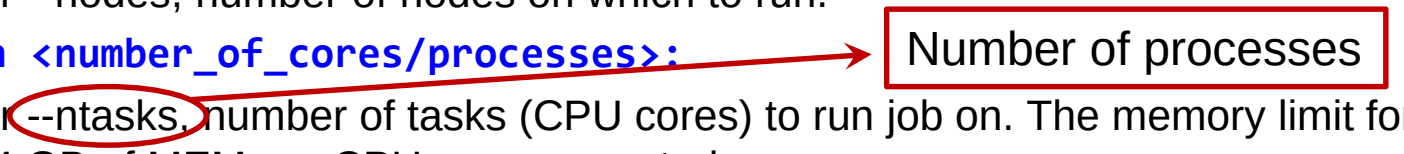
```
#!/bin/bash
#SBATCH -N 1        # request one node
#SBATCH -t 2:00:00  # request two hours
#SBATCH -p single   # in single partition (queue)
#SBATCH -A your_allocation_name
#SBATCH -o %x-%j.out-%N # optional, name of the stdout, using the job number (%j) and the
hostname of the node (%N)
#SBATCH -e %x-%j.err-%N # optional, name of the stderr, using job and hostname values
# below are job commands

date
cd /work/$USER/myjob
./mydemo
# Mark the time it finishes.
date
# exit the job
exit 0
```

Tells the job scheduler how much resource you need.

How will you use the resources?

# Common Slurm Switches

- **`#SBATCH -A allocation_name`:**
  - short for --account, charge jobs to your allocation named allocation_name.
- **`#SBATCH -N <number_of_nodes>:`**
  - short for --nodes, number of nodes on which to run.
- **`#SBATCH -n <number_of_cores/processes>:`**
  - short for --ntasks, number of tasks (CPU cores) to run job on. The memory limit for jobs is 4 GB of MEM per CPU core requested.




- **`#SBATCH -c <cores_per_process>:`**
  - short for --ncpus-per-task, number of threads per process.
- **`#SBATCH -p partition:`**
  - short for --partition, submit job to the partition queue. Allowed values for partition: single, checkpt, workq, gpu, bigmem. Depending on cluster (use `sinfo` command)
- **`#SBATCH -t hh:mm:ss:`**
  - short for --time, request walltime.
- **`#SBATCH -o filename.out:`**
  - short for --output, write standard output.
- **`#SBATCH -e filename.err:`**
  - short for --error, write standard error.
  - Note that by default, Slurm will merge standard error and standard output.

# Submit Slurm Batch Job

- **To submit the above job to the scheduler, save the above script as a text file, e.g., singlenode.sh, then use the `sbatch` command to submit, the output will be something like the below:**

```
[fchen14@qbc2 slurmdoc]$ sbatch singlenode.slm
Submitted batch job 37355 estimates 9 SUs from allocation loni_train_2020.
Estimated remaining SUs: 37352
See running job information with: scontrol show job 37355
```

- **To check the status of your job use the `squeue` command:**

```
[fchen14@qbc2 slurmdoc]$ squeue -u $USER
JOBID PARTITION     QOS      NAME     USER  ACCOUNT       STATE  PRIORITY       TIME
SUBMIT_TIM TIME_LIMI  NODES   CPUS MIN_MEMORY NODELIST(REASON)
             37480   checkpt normal batch.sl  fchen14 loni_tra    RUNNING
1       0:06 2020-09-18   1:00:00      2     96 3958M qbc[161-162]
```

# Common Slurm Commands (1)

- **`squeue` is used to show the partition (queue) status. Useful options:**
  - `-u <username>:` limit output to jobs by username --state=pending: limit output to pending (i.e. queued) jobs --state=running: limit output to running jobs
  - Below is an example to query all jobs submitted by current user (fchen14)

```
[fchen14@qbc2 slurmdoc]$ squeue -u fchen14
JOBID  PARTITION  NAME       USER    ST  TIME_LIMIT TIME CPUS  NODES  NODELIST(REASON)
37876  workq      hybrid_job fchen14 CF  5:00       0:04 96    2      qbc[005-006]
```

# Common Slurm Commands (2)

- **`sinfo` is used to view information about Slurm nodes and partitions.**

```
[fchen14@qbc2 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
admin        up   infinite    201   idle qbc[001-190,192-202]
admin        up   infinite      1   down qbc191
single*      up 3-00:00:00    191   idle qbc[001-190,192]
single*      up 3-00:00:00      1   down qbc191
checkpt      up 3-00:00:00    191   idle qbc[001-190,192]
checkpt      up 3-00:00:00      1   down qbc191
workq        up 3-00:00:00    191   idle qbc[001-190,192]
workq        up 3-00:00:00      1   down qbc191
gpu          up 3-00:00:00      8   idle qbc[193-200]
bigmem       up 3-00:00:00      2   idle qbc[201-202]
```

# Common Slurm Commands (3)

- **scancel is used to signal or cancel jobs. Typical usage with squeue.**

```
[fchen14@qbc1 ~]$ squeue -u fchen14
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
               341   checkpt     bash  fchen14  R       0:13      1 qbc001
               340   checkpt     bash  fchen14  R    1:50:57      1 qbc002
# cancel (delete) job with JOBID 340
[fchen14@qbc1 ~]$ scancel 340
# job status might display a temporary "CG" ("CompletinG") status immediately after
scancel
[fchen14@qbc1 ~]$ squeue -u fchen14
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
               340   checkpt     bash  fchen14 CG    1:51:08      1 qbc002
               341   checkpt     bash  fchen14  R       0:41      1 qbc001
[fchen14@qbc1 ~]$ squeue -u fchen14
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
               341   checkpt     bash  fchen14  R       1:08      1 qbc001
```

# Common Slurm Commands (4)

- **`scontrol` is used to view or modify Slurm configuration and state. Typical usage for the user is to check job status:**

```
[fchen14@qbc1 ~]$ squeue -u fchen14 # show all jobs
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
               341   checkpt     bash  fchen14  R    1:29:20      1 qbc001
[fchen14@qbc1 ~]$ scontrol show job 341
JobId=341 JobName=bash
   UserId=fchen14(32584) GroupId=Admins(10000) MCS_label=N/A
   Priority=1 Nice=0 Account=hpc_hpcadmin6 QOS=normal
   JobState=RUNNING Reason=None Dependency=(null)
   ... some details omitted...
   MinCPUsNode=1 MinMemoryNode=22332M MinTmpDiskNode=0
   Features=(null) DelayBoot=00:00:00
   OverSubscribe=NO Contiguous=0 Licenses=(null) Network=(null)
   Command=/bin/bash
   WorkDir=/home/fchen14/test
   Power=
```

# Serial and Parallel (Multi-Threaded and MPI) Job Templates

- **Serial Jobs**

- **Parallel Jobs**
  - SMP (Shared Memory Parallelism)
    - OpenMP
    - Python's Multiprocessing
    - Pthread
    - R's mcapply
  - MPI
  - Hybrid

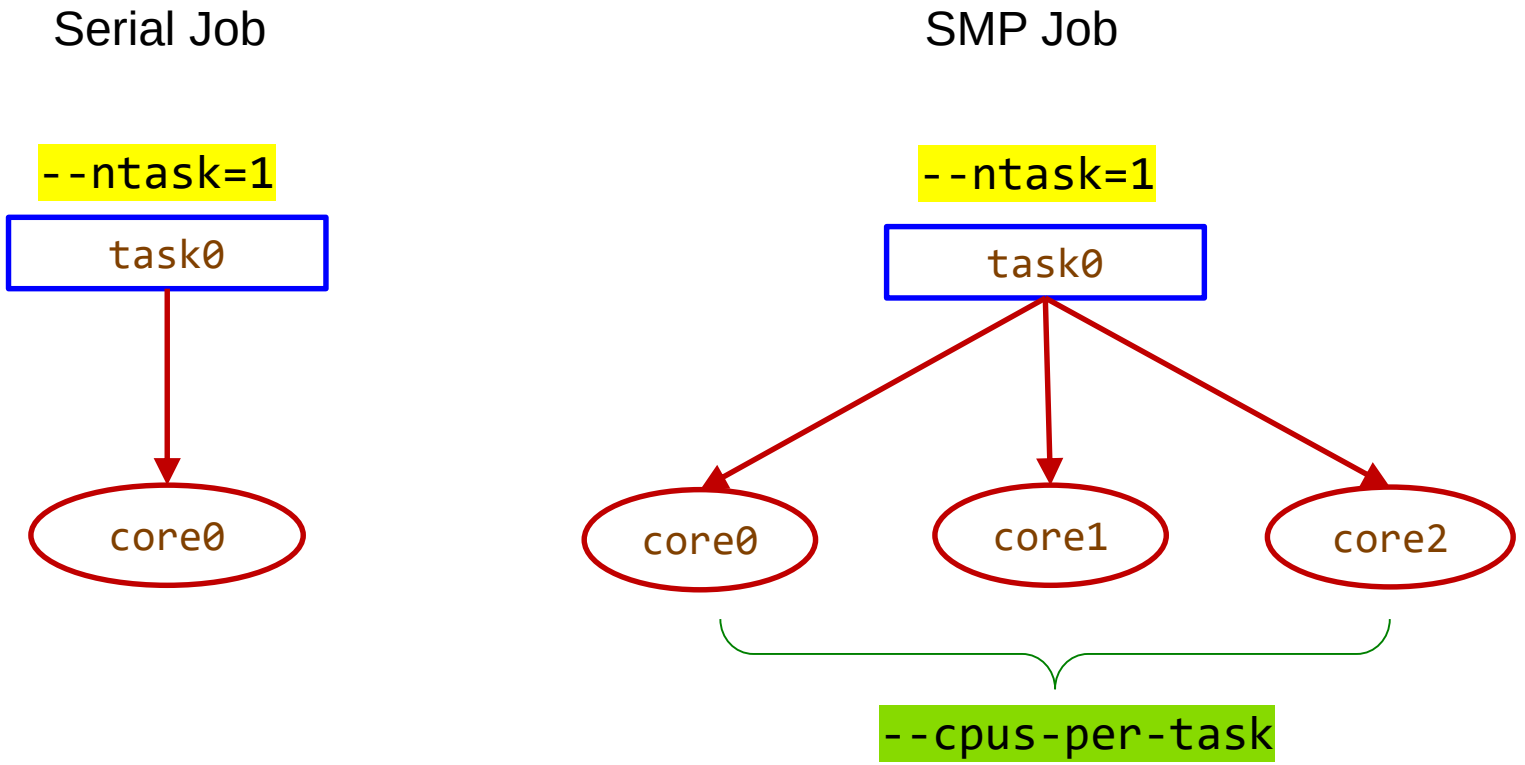
# Serial Job Script Template

```
#!/bin/bash
#SBATCH --job-name=serial_job_test    # Job name
#SBATCH --ntasks=1                    # Using a single core
#SBATCH --time=00:10:00      # Time limit hh:mm:ss
#SBATCH --output=%x_%j.log   # Standard output and error log,
                             # %x: job name
                             # %j: job-id

module load python

echo "Running job on a single CPU core"

date
/home/user/single_core_job.py
date
```

# Shared Memory Parallelism (SMP) Jobs

- **Shared-Memory Parallelism (SMP) is when workload is shared among different CPU cores using multiple threads or processes running within a single compute node and these cores have access to common (shared) memory.**
  - SMP jobs cannot make use of multiple nodes and all the cores must be physically located the same node.
  - When running SMP jobs, you must make the SMP application aware of how many cores to use.
  - How that is done depends on the specific type of application. Typical examples:
    - OpenMP (Open Multi-Processing)
      - First set `--ntasks=1`, and then set OMP_NUM_THREADS to a value less than or equal to the number of cpus-per-task
    - Pthreads
    - Python's multiprocessing module
    - R's mcapply

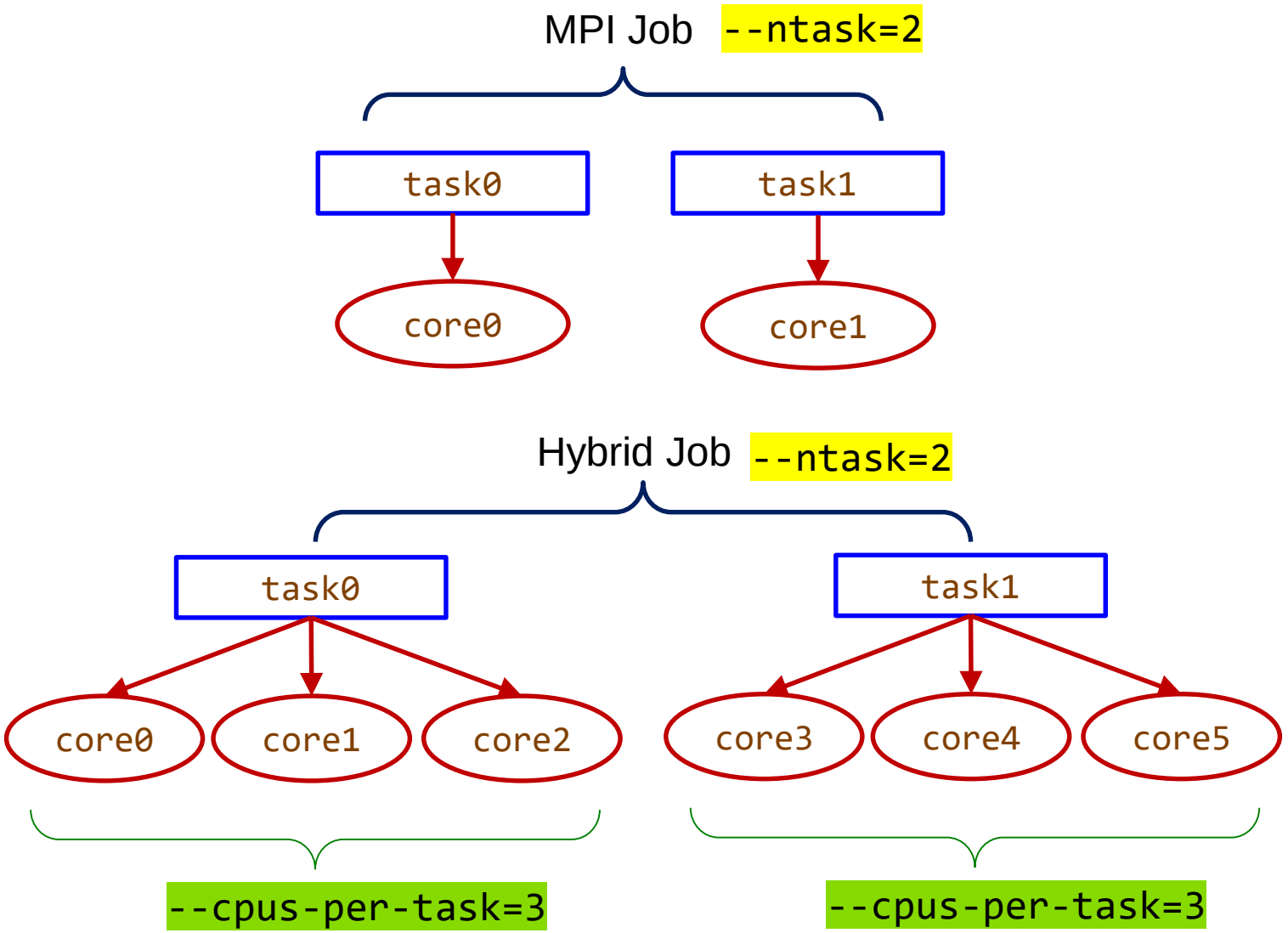
# Conceptual Illustration of Serial vs SMP



# SMP Job Script (OpenMP Based)

```
#!/bin/bash
#SBATCH --job-name=smp_job           # Job name
#SBATCH --nodes=1                    # Run all processes on a single node
#SBATCH --ntasks=1                   # Run a single task
#SBATCH --cpus-per-task=8            # Number of CPU cores per task
#SBATCH --time=00:10:00              # Time limit hh:mm:ss
#SBATCH --output=%x_%j.log           # Standard output and error log

date
# use this line if your job uses OpenMP
export OMP_NUM_THREADS=$Slurm_CPUS_PER_TASK

/home/user/smp_job.out
date
```

# Conceptual Illustration of MPI vs Hybrid



# MPI (Message Passing Interface) Job

- **According to Slurm documentation (https://slurm.schedmd.com/mpi_guide.html), "there are three fundamentally different modes of operation used by various MPI implementation with Slurm:**

  - *“Slurm directly launches the tasks and performs initialization of communications through the PMI2 or PMIx APIs. (Supported by most modern MPI implementations.)*
  - *Use mpirun launches tasks using Slurm's infrastructure (not using PMIx).*
  - *Slurm creates a resource allocation for the job and then mpirun launches tasks using some mechanism other than Slurm." (We do not recommend HPC/LONI users use this method to launch their MPI jobs.)”*

# MPI Job - (PMIx Versions)

- **If you compiled your MPI application using our default mvapich2 libraries (which is compiled with PMIx enabled), you should start the application directly using the `srun` command.**

```
#!/bin/bash
#SBATCH --job-name=mpi_job_test      # Job name
#SBATCH --partition=workq            # For jobs using more than 1 node, submit to workq
#SBATCH --nodes=2                    # Number of nodes to be allocated
#SBATCH --ntasks=96                  # Number of MPI tasks (i.e. processes/cores)
#SBATCH --time=00:05:00              # Wall time limit (hh:mm:ss)
#SBATCH --output=%x_%j.log           # Standard output and error

echo ""
echo "Slurm Nodes Allocated          = $Slurm_JOB_NODELIST"
echo "Number of Nodes Allocated      = $Slurm_JOB_NUM_NODES"
echo "Number of Tasks Allocated      = $Slurm_NTASKS"

module load mvapich2/2.3.3/intel-19.0.5
srun -n $Slurm_NTASKS ./a.out
```

# MPI Job - (Non-PMIx Versions)

- **If your MPI application did not use our default module key mvapich2/2.3.3/intel-19.0.5, you should start the application using the `mpirun` command.**

```
#!/bin/bash
#SBATCH --job-name=mpi_job_test      # Job name
#SBATCH --partition=workq            # For jobs using more than 1 node, submit to workq
#SBATCH --nodes=2                    # Number of nodes to be allocated
#SBATCH --ntasks=96                  # Number of MPI tasks (i.e. processes/cores)
#SBATCH --time=00:05:00              # Wall time limit (hh:mm:ss)
#SBATCH --output=mpi_test_%j.log     # Standard output and error

echo ""
echo "Slurm Nodes Allocated          = $Slurm_JOB_NODELIST"
echo "Number of Nodes Allocated      = $Slurm_JOB_NUM_NODES"
echo "Number of Tasks Allocated      = $Slurm_NTASKS"

module load mvapich2/2.3.3/intel-19.0.5-hydra
mpirun -n $Slurm_NTASKS ./a.out
```

# Hybrid (MPI + SMP) Job

- **Hybrid jobs are MPI applications where each MPI process is multi-threaded and can use multiple cores across multiple nodes. If the MPI implementation is compiled with PMIx enabled, use the `srun` command to start the hybrid job, otherwise, use the `mpirun` command to start it.**

```
#!/bin/bash
#SBATCH --partition=workq            # Need to submit workq for multiple node jobs
#SBATCH --nodes=2                    # Maximum number of nodes to be allocated
#SBATCH --ntasks=4                   # Number of MPI tasks (i.e. processes)
#SBATCH --cpus-per-task=24           # Number of cores per MPI task
#SBATCH --time=00:05:00              # Wall time limit (hh:mm:ss)
#SBATCH --output=hybrid_test_%j.log  # Standard output and error file

echo "Number of Nodes Allocated      = $Slurm_JOB_NUM_NODES"
echo "Number of Tasks Allocated      = $Slurm_NTASKS"
echo "Number of Cores/Task Allocated = $Slurm_CPUS_PER_TASK"

module load mvapich2/2.3.3/intel-19.0.5
# PMIx version
srun -n $Slurm_NTASKS -c $Slurm_CPUS_PER_TASK ./a_pmix.out
# Non-PMIx version
module swap mvapich2/2.3.3/intel-19.0.5 mvapich2/2.3.3/intel-19.0.5-hydra
export OMP_NUM_THREADS=$Slurm_CPUS_PER_TASK
mpirun -n $Slurm_NTASKS ./a_hydra.out
```

# Inspect Job Results

- **Check job standard output/standard error by inspecting the redirected files:**

```
[fchen14@qbc2 slurmdoc]$ cat hybrid_test_37723.log
Date              = Tue Sep 22 15:37:44 CDT 2020
Hostname          = qbc001
Working Directory = /home/fchen14/slurmdoc

Number of Nodes Allocated      = 2
Slurm Nodes Allocated          = qbc[001-002]
Number of Tasks Allocated      = 4
Number of Cores/Task Allocated = 24
Loading mvapich2/2.3.3/intel-19.0.5
  Loading requirement: intel/19.0.5
Autoloading hdf5/1.10.6/intel-19.0.5-mvapich-2.3.3
Autoloading fftw/3.3.8/intel-19.0.5-mvapich-2.3.3
Autoloading netcdf-c/4.7.3/intel-19.0.5-mvapich-2.3.3
```

*QB3 HPC User Environment*

# Job Monitoring

# Job Monitoring on QB3

- **Check details on your job using `qstat`**

```
$ squeue -u $USER  : For quick look at nodes assigned to you
$ scontrol show job <jobid>     : For details on your job
$ scancel jobid          : To delete job
```

- **Check memory usage of your job using `qshow`**

```
$ qshow jobid
```

- **Please pay close attention to the load and the memory consumed by your job!**

# Using the “top” command

- **The top program provides a dynamic real-time view of a running system.**

```
top - 23:30:16 up 51 days, 16:18,  4 users,  load average: 0.16, 0.05, 0.06
Tasks: 692 total,   2 running, 690 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.1 us,  1.0 sy,  0.0 ni, 97.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 19647060+total, 18699553+free,  8677504 used,   797560 buff/cache
KiB Swap: 13421772+total, 13405440+free,   163328 used. 18702988+avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
208754 fchen14   20   0 7731040   5.5g  20108 R 100.0  2.9   0:16.50 lmp
208999 fchen14   20   0  172868   2948   1624 R   0.7  0.0   0:00.07 top
     1 root      20   0  191624   2832   1544 S   0.0  0.0  21:18.21 systemd
     2 root      20   0       0      0      0 S   0.0  0.0   0:04.81 kthreadd
     4 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
     6 root      20   0       0      0      0 S   0.0  0.0   1:06.85 ksoftirqd/0
```

# Check memory Usage for Multi-Node Job

- Check health of your job using `qshow`

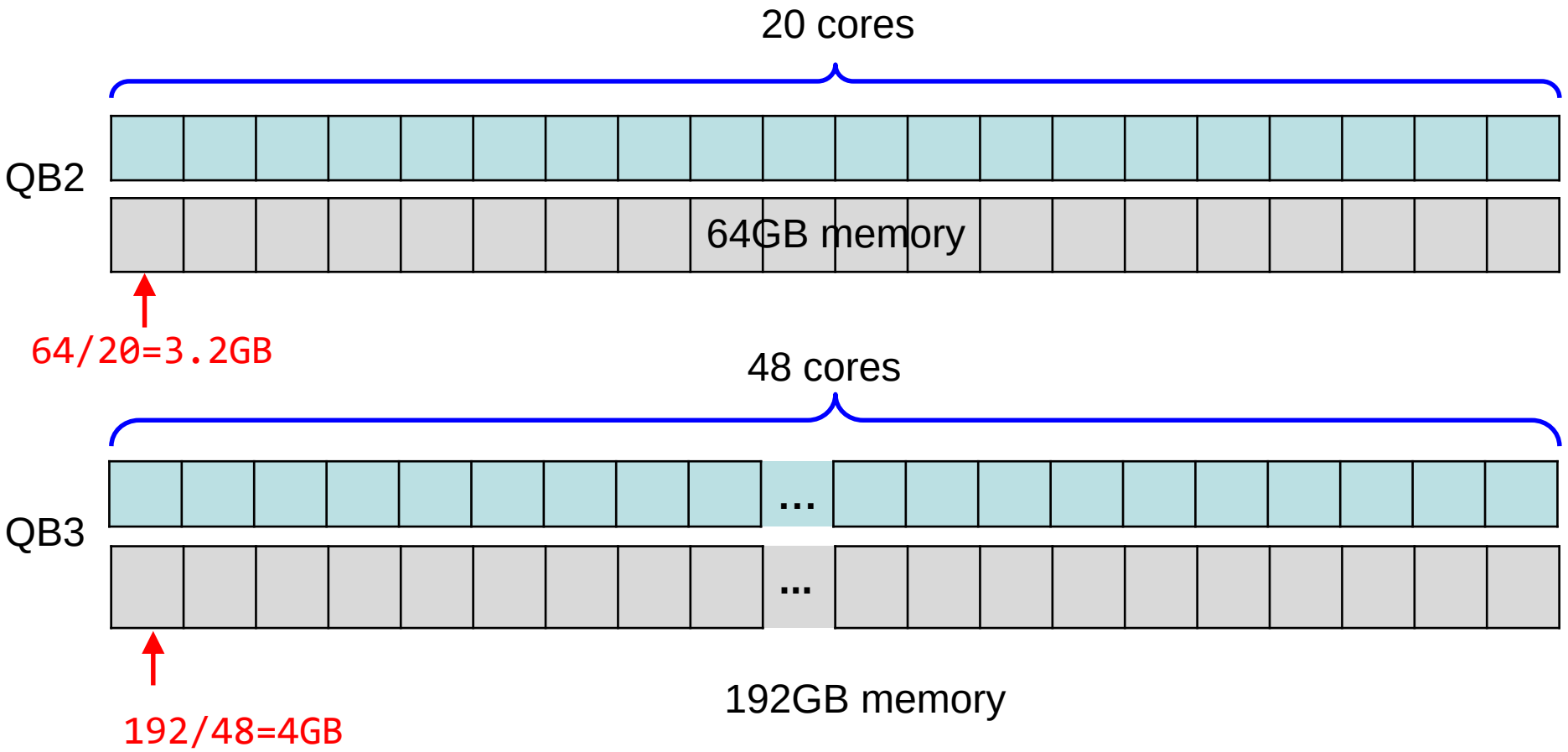
```
$ qshow <jobid>
```

```
[fchen14@qbc2 slurmdoc]$ sbatch ex_lmp_hybrid.sh
Submitted batch job 37888 estimates 8 SUs from allocation loni_loniadmin1.
Estimated remaining SUs: 521696
JOBID      NAME               PARTITION  TIME_LIMIT  ST  CPUS  NODES  REASON
37888      hybrid_job_test    workq      5:00        PD  96    2      None
[fchen14@qbc2 slurmdoc]$ qshow 37888
PBS job: 37888, nodes: 2
Hostname  Days Load CPU U# (User:Process:VirtualMemory:Memory:Hours)
qbc005       0 Autoloading 211  6 fchen14:lmp:5847M:3.2G fchen14:lmp:5846M:3.3G
fchen14:slurm_scr+:113M:2M fchen14:srun:388M:5M fchen14:srun:50M:1M
qbc006       0 Autoloading 216  3 fchen14:lmp:5870M:5.1G fchen14:lmp:4447M:3.3G
PBS_job=37888 user=fchen14 allocation=loni_loniadmin1 queue=workq total_load=0.00
cpu_hours=0.00 wall_hours=0.00 unused_nodes=0 total_nodes=2 ppn=48 avg_load=0.00
avg_cpu=213% avg_mem=7640mb avg_vmem=11746mb
top_proc=fchen14:lmp:qbc006:5870M:5.1G:0.0hr:105%  node_processes=3
```

# Pay attention to single queue usage

- **Single queue - Used for jobs that will only execute on a single node, i.e. `–N1 -n1-47`.**
- **Jobs in the single queue should not use:**
  - More than 3GB memory per core for QB2 (64G/20).
  - More than 4GB memory per core for QB3 (192G/48).
- **If applications require more memory, scale the number of cores `(--ntasks)` to the amount of memory required: i.e. max memory available for jobs in single queue is 16GB for `--ntasks 4` on QB3.**

# Core and Memory in Single queue




***Question:***

On QB2, if my job needs 17GB memory, what `-ppn` value should I use?

On QB3, if my job needs 17GB memory, what `--ntasks (-n)` value should I use?

# Short Summary

- **How to check queue (partition) information in slurm?**
- **How to submit Slurm interactive job?**
- **How to submit Slurm batch job?**
  - SMP (OpenMP)
  - MPI
  - Hybrid
- **Monitor your job?**

❖ ***Questions?***

# Next Sessions

- **11:30 - 12:30**
  - Performance benchmarks and tuning
- **12:30 - 02:00**
  - Lunch break
- **02:00 - 04:00**
  - Q&A + On-ramp sessions (breakout sessions)