HPC User Environment 2

Jason Li
HPC User Services
LSU HPC / LONI
sys-help@loni.org

Louisiana State University, Baton Rouge
Feb 7, 2024

Outlines

- **HPC User Environment 1**

  1. Intro to HPC
  2. Getting started
  3. Into the cluster
  4. Software environment (modules)

- **HPC User Environment 2**

  1. Basic concepts
  2. Preparing my job
  3. Submitting my job
  4. Managing my jobs

Outlines

- **HPC User Environment 2**

1. Basic concepts
2. Preparing my job
3. Submitting my job
4. Managing my jobs

Outlines

- **HPC User Environment 2**

1. Basic concepts
    1) Previously on HPC User Environment 1…
    2) Job & Job schedulers
2. Preparing my job
    1) Basic principles
    2) Job duration (wall time)
    3) Number of nodes & cores
    4) Job queues
3. Submitting my job
    1) Interactive job
    2) Batch job
4. Managing my jobs
    1) Useful commands
    2) Monitoring job health

Outlines

- **HPC User Environment 2**

1. Basic concepts
    1) Previously on HPC User Environment 1…
    2) Job & Job schedulers
2. Preparing my job
    1) Basic principles
    2) Job duration (wall time)
    3) Number of nodes & cores
    4) Job queues
3. Submitting my job
    1) Interactive job
    2) Batch job
4. Managing my jobs
    1) Useful commands
    2) Monitoring job health

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

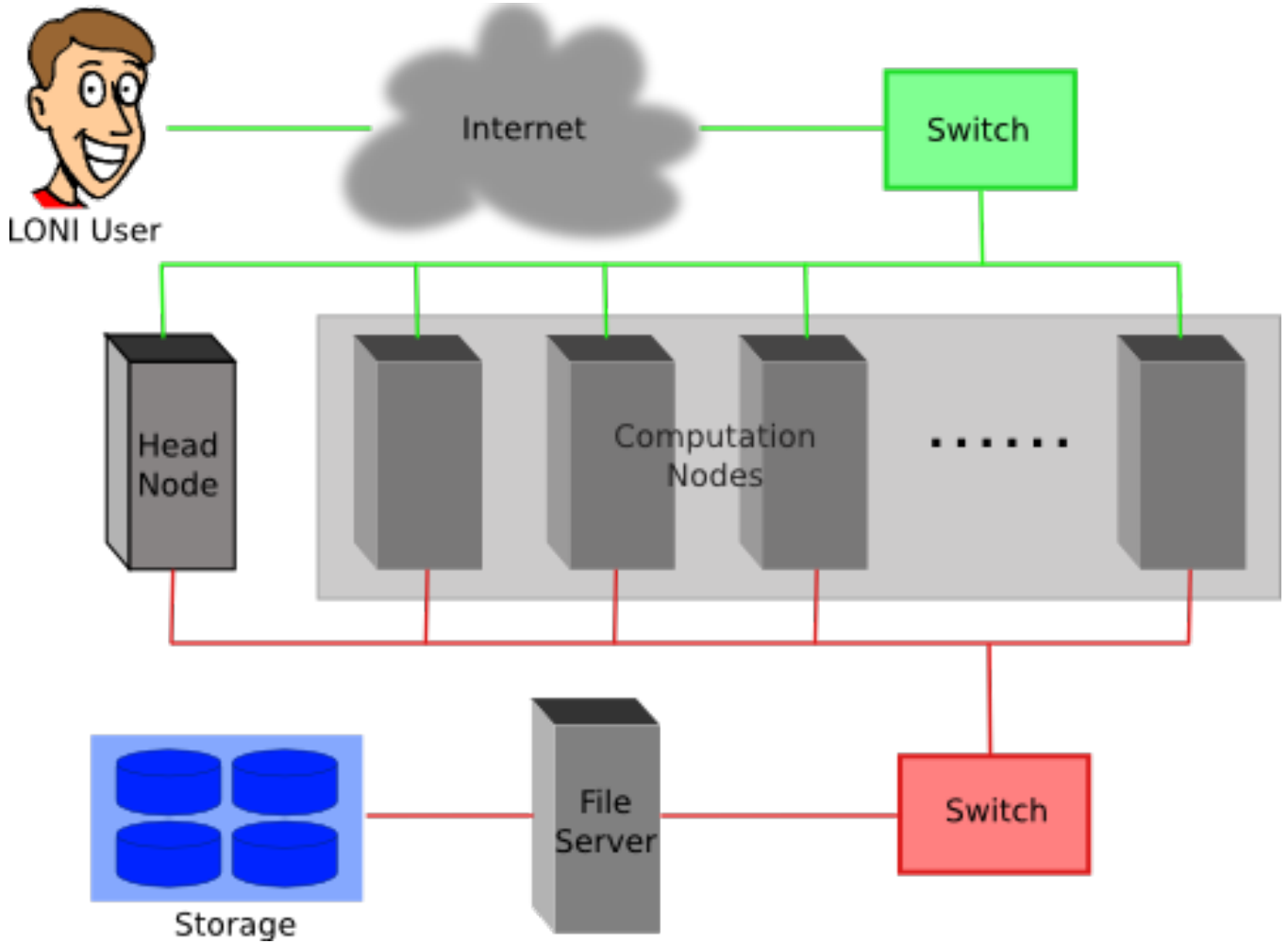
1) Previously on HPC User Environment 1…

Two things needed to run jobs on our clusters:

1) Account

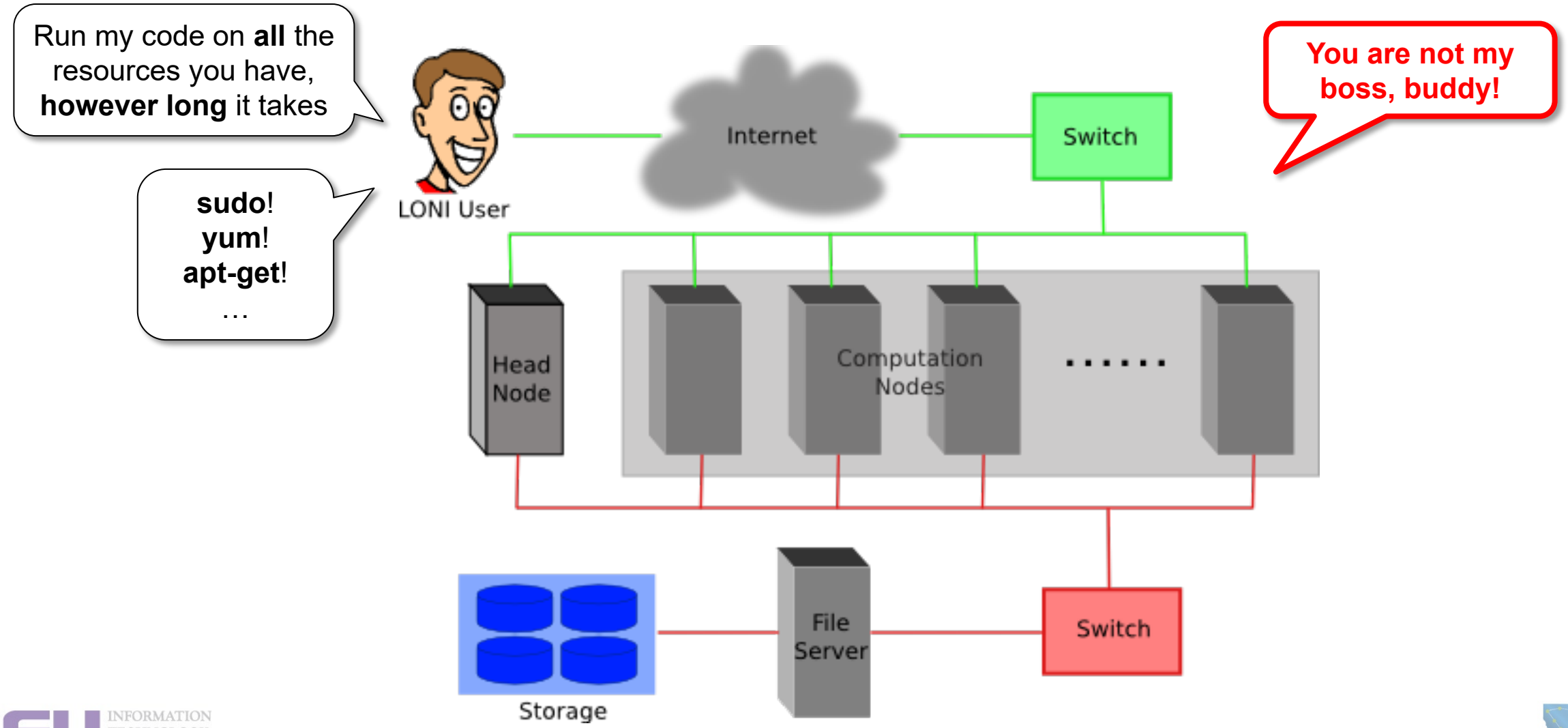
2) Allocation

1) Previously on HPC User Environment 1…



1) Previously on HPC User Environment 1…

1) Previously on HPC User Environment 1…

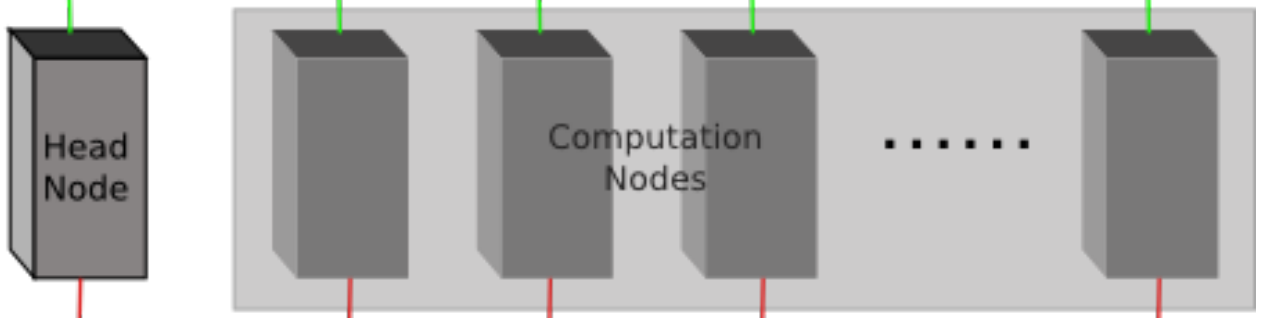


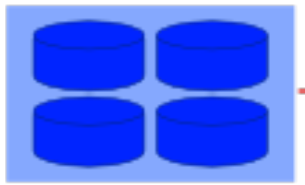
1) Previously on HPC User Environment 1…

I will ask nicely. Please grant me the use of **24 cores** for **10 hours** to run my code.

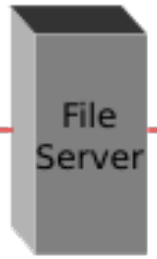
LONI User

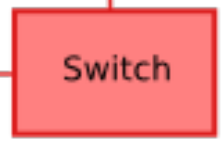
Now we are talking. Let me schedule it for you.





Storage





LSU INFORMATION TECHNOLOGY SERVICES

LONI

Outlines

- **HPC User Environment 2**

1. Basic concepts
    1) Previously on HPC User Environment 1…
    2) Job & Job schedulers
2. Preparing my job
    1) Basic principles
    2) Job duration (wall time)
    3) Number of nodes & cores
    4) Job queues
3. Submitting my job
    1) Interactive job
    2) Batch job
4. Managing my jobs
    1) Useful commands
    2) Monitoring job health

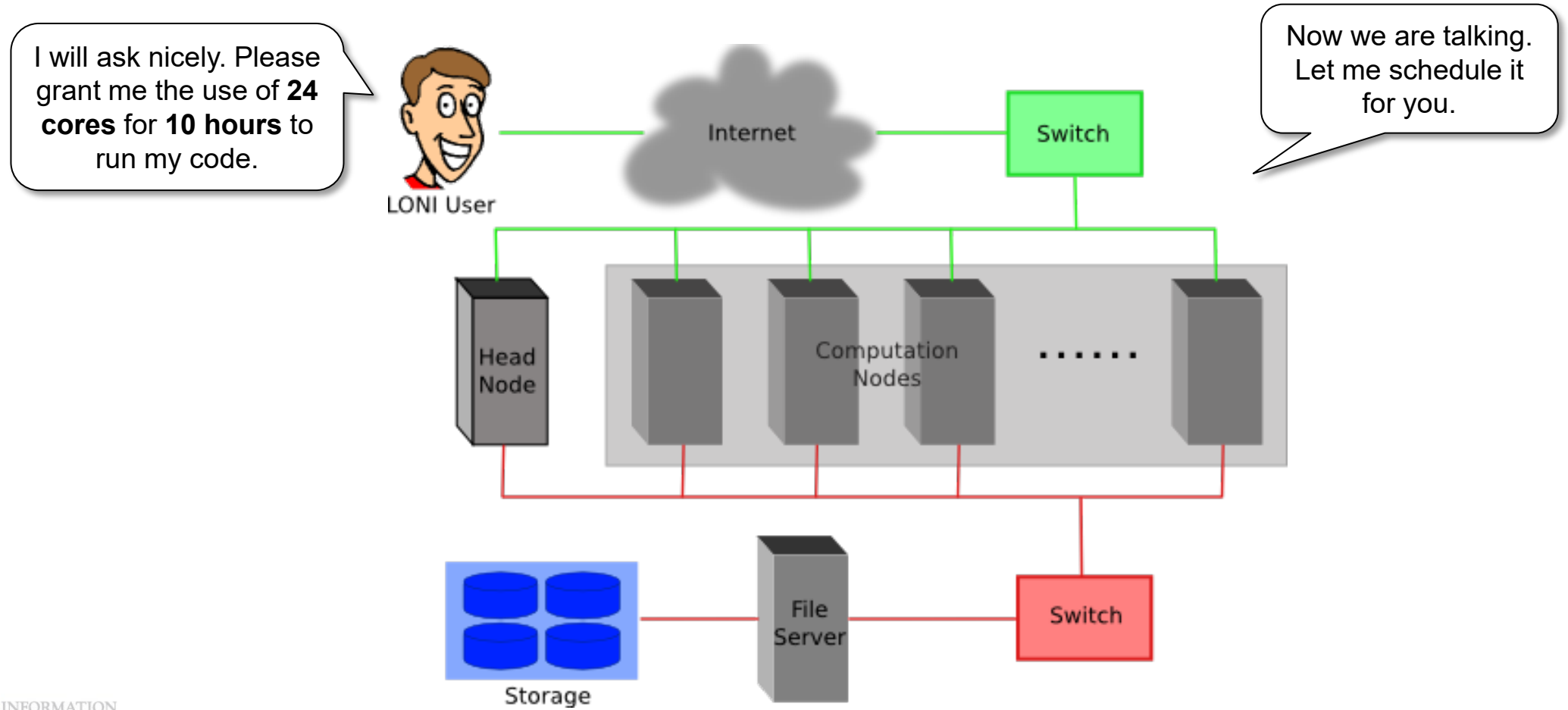
2) Job & Job scheduler

a) What's a "job"?

- A user's request to use **a number of nodes/cores** for **a certain amount of time** on a cluster.
- Calculation **MUST** be done via jobs (**NO** heavy calculation on head nodes!!)
- SUs deducted from allocations based on actual usage of each job.
  - Example:
    - My allocation: 50,000 SU
    - Running a job: 24 core * 10 hours = 240 SU
    - Balance: 49,760 SU

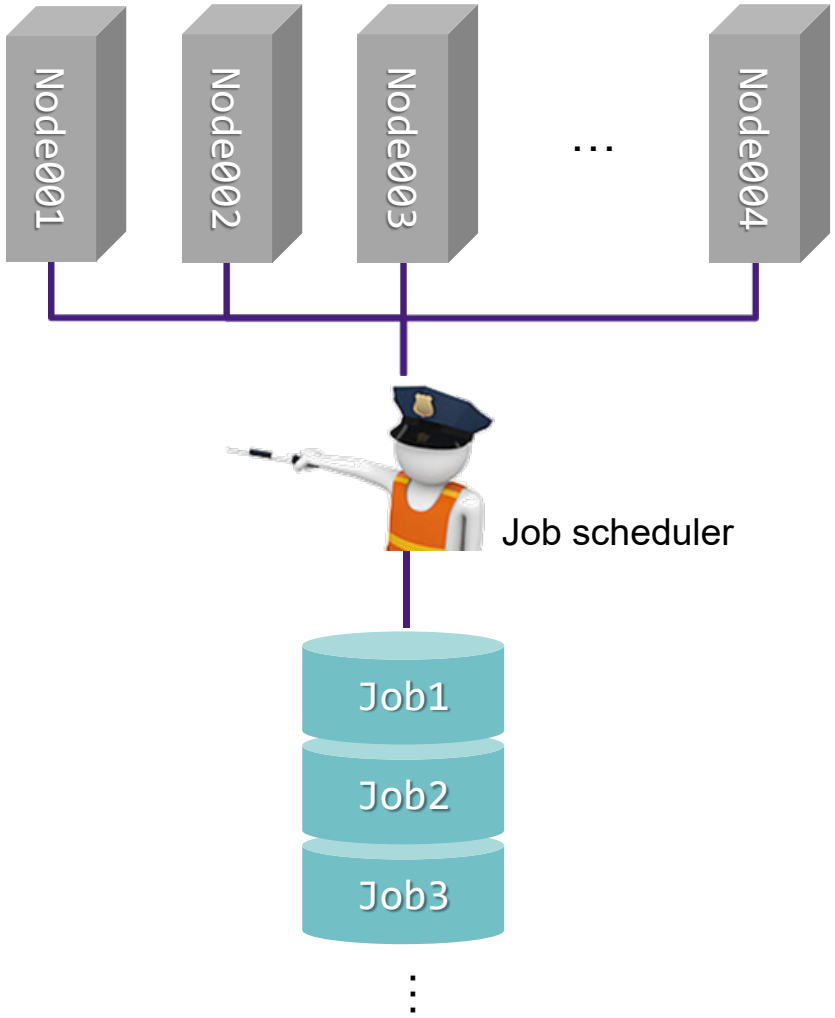
2) Job & Job scheduler

b) What's a "job scheduler"?



2) Job & Job scheduler

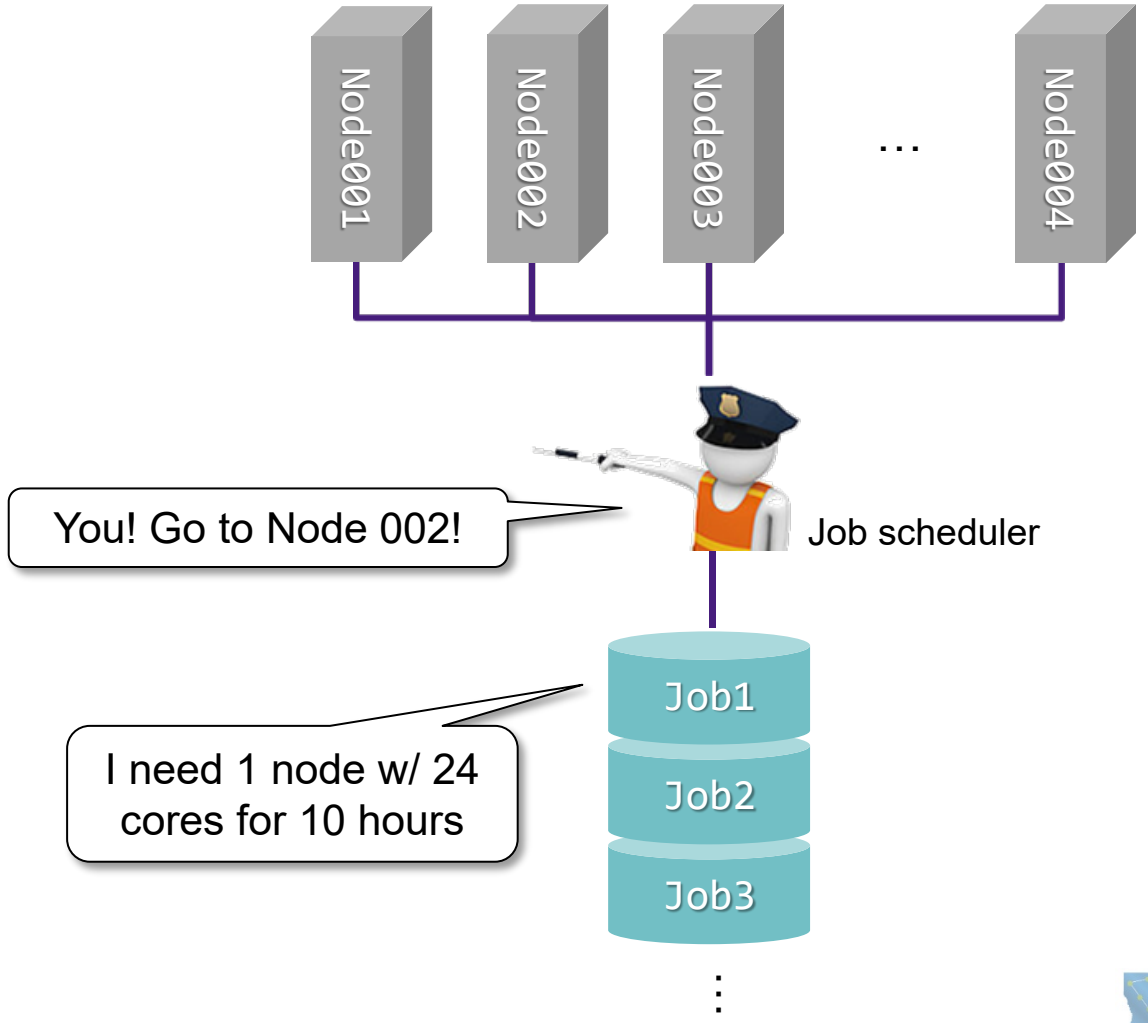
b) What's a "job scheduler"?



2) Job & Job scheduler

b) What's a "job scheduler"?

i. Decides which job runs when and where

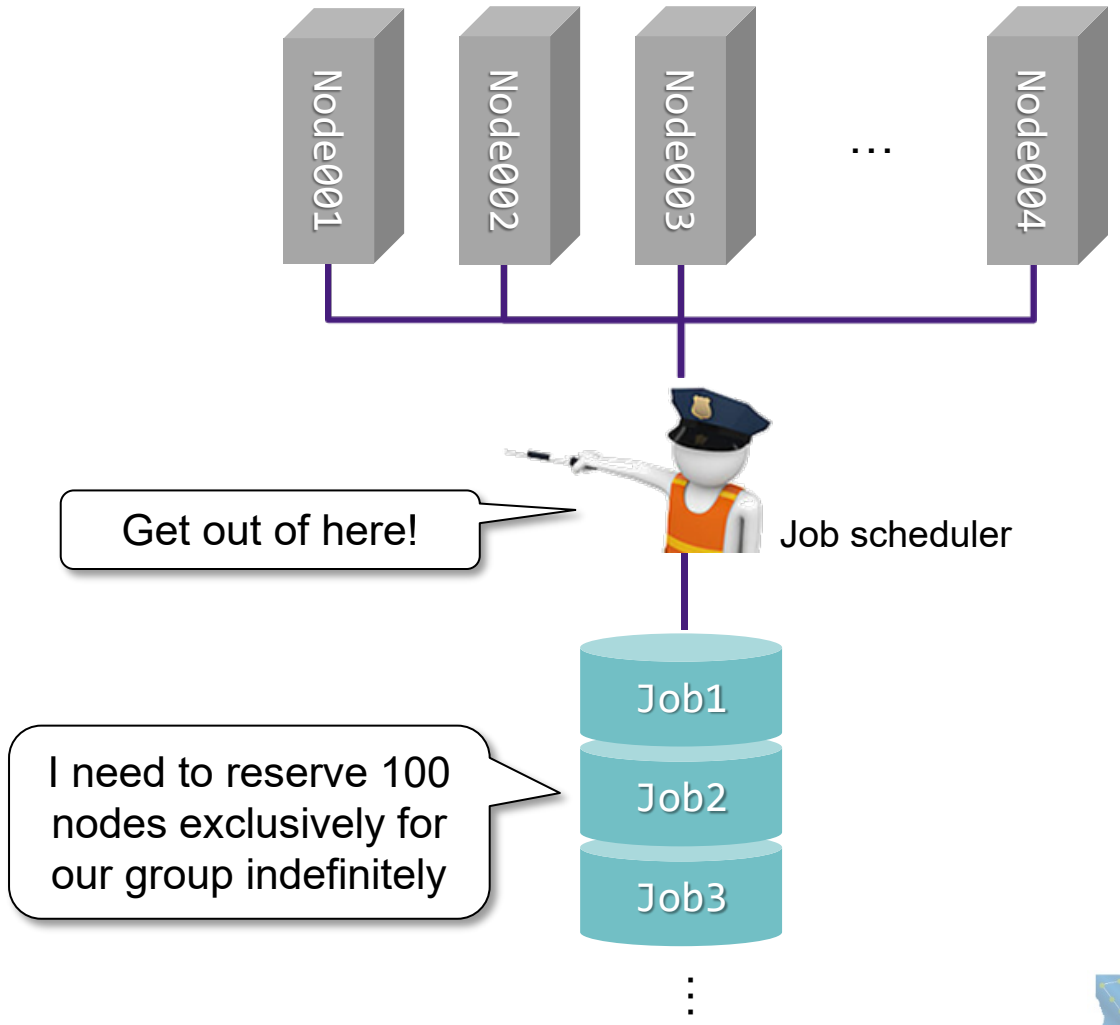


2) Job & Job scheduler

b) What's a "job scheduler"?

i. Decides which job runs when and where

ii. Enforces job policies



b) What's a "job scheduler"?

Job scheduler's responsibilities	Your responsibilities
• Decides which job runs when and where • Enforces job policies	• Decide a job's size and duration • Understand the job queuing system and policies • Submit/monitor/cancel jobs • Diagnose job health

b) What's a "job scheduler"?

i) PBS

b) What's a "job scheduler"?

i) PBS

ii) Slurm

2) Job & Job scheduler

b) What's a "job scheduler"?

	LSU HPC	LONI
i) PBS	SMIC	QB2
ii) Slurm	Deep Bayou SuperMike III	QB3

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

1) Basic principles

- **Two basic principles of requesting resources**
  - Number of nodes / cores, RAM size, job duration, …

Large enough …

Small enough …

1) Basic principles

- **Two basic principles of requesting resources**
  - Number of nodes / cores, RAM size, job duration, …

Large enough …	Small enough …
• To successfully complete your job	• To ensure quick turnaround • Not to waste resources for other users

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

2) Job duration (wall time)

- **What is it?**
  - **Real-world** (**wall**) time taken from the start to the end
  - Must tell job scheduler how long you want your job to run
  - There is a **maximum** wall time you may request (see later)

2) Job duration (wall time)

- **FAQ**

Q	A
• What if my command is still running when the wall time runs out?	• Job **terminated**, any running process **killed**
• What if all my commands in the job finished before the wall time runs out?	• Job **exits** successfully when all commands finished
• If my job exits before requested wall time, how many SUs will I be charged?	• You will be charged based on your **actual time used** (if less than requested)
• In that case, why don't I just request maximum wall time every time?	• Your queuing time may be long…

2) Job duration (wall time)

- **Back to basic principles…**

Large enough …	Small enough …
• To successfully complete your job	• To ensure quick turnaround • Not to waste resources for other users

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

3) Number of nodes & cores

- **Previously in HPC User Environment 1 …**

SuperMIC	
Hostname	smic.hpc.lsu.edu
Peak Performance/TFlops	925
Compute nodes	360
Processor/node	2 10-core
Processor Speed	2.8 GHz
Processor Type	Intel Xeon 64bit
Nodes with Accelerators	360
Accelerator Type	Xeon Phi 7120P
OS	RHEL v6
Vendor	
Memory per node	64 GB
Detailed Cluster Description	
User Guide	
Available Software	

Deep Bayou	
Hostname	db1.lsu.edu
Peak Performance/TFlops	257
Compute nodes	13
Processor/node	2 24-core
Processor Speed	2.4 GHz
Processor Type	Intel Cascade Lake Xeon 64bit
Nodes with Accelerators	13
Accelerator Type	2 x NVIDIA Volta V100S
OS	RHEL v7
Vendor	Dell
Memory per node	192 GB
Detailed Cluster Description	
User Guide	
Available Software	

SuperMike III	
Hostname	mike.hpc.lsu.edu
Peak Performance/TFlops	1,285
Compute nodes	183
Processor/node	2 32-core
Processor Speed	2.6GHz
Processor Type	Intel Xeon Ice Lake
Nodes with Accelerators	8
Accelerator Type	4 NVIDIA A100
OS	RHEL v8
Vendor	Dell
Memory per node	256/2048 GB
Detailed Cluster Description	
User Guide	
Available Software	

3) Number of nodes & cores

- **When submitting you job…**
  - Must tell job scheduler the number of nodes & cores you need

3) Number of nodes & cores

- **FAQ**

Q	A
• My code runs slow. Can I request more nodes / cores to make it faster?	• Not quite! Your code most likely is **NOT** using multiple nodes / cores, if: - You **do not know** if it is using multiple nodes / cores - You **did not tell** it to use multiple nodes / cores - You are **not familiar** with names like “MPI” / “OpenMP” • **Underutilization** is THE most common warning received on our clusters
• How many nodes / cores should I request?	• In short: We can’t answer that • Each code / job is different. You must test to determine

3) Number of nodes & cores

- **Back to basic principles…**

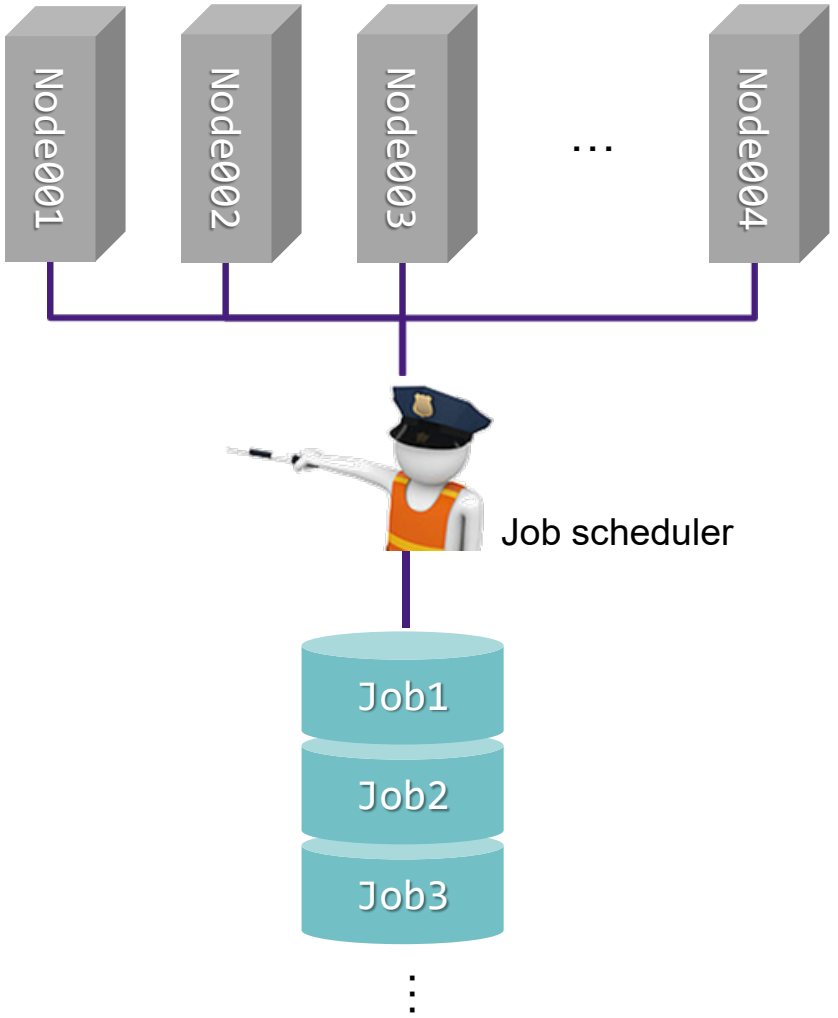
Large enough …	Small enough …
• To successfully complete your job	• To ensure quick turnaround • Not to waste resources for other users

Outlines

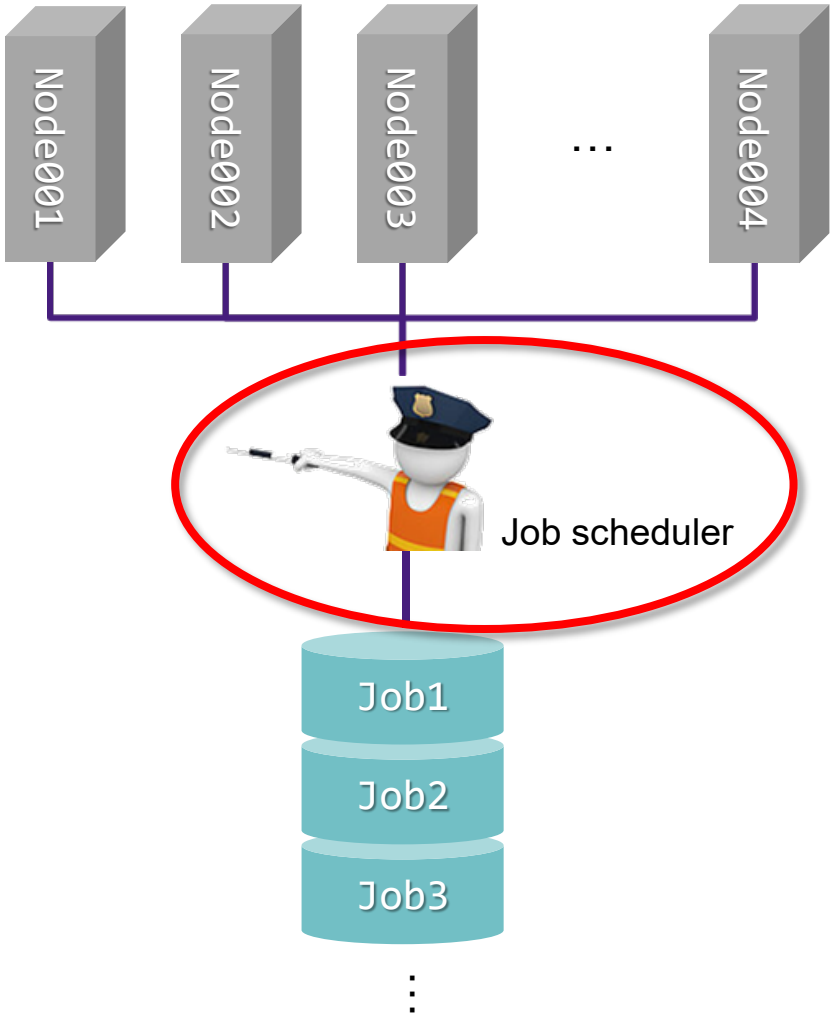
- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

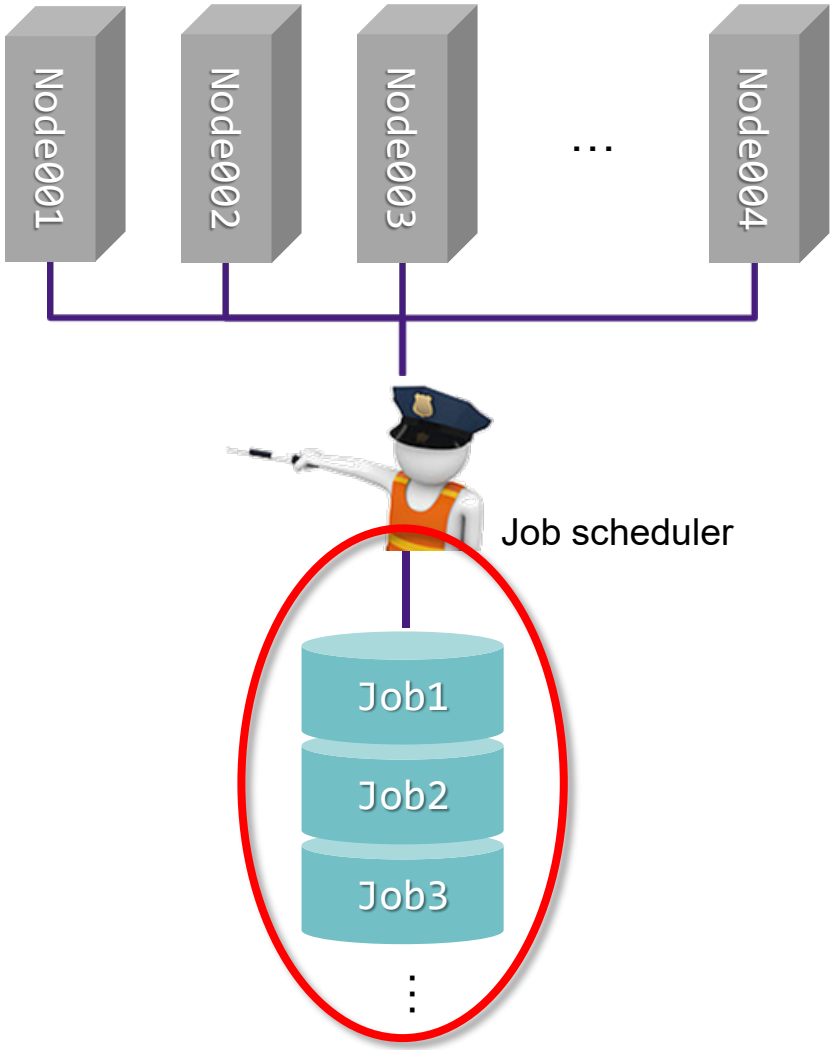
4) Job queues


Node001
Node002
Node003
...
Node004
Job scheduler
Job1
Job2
Job3

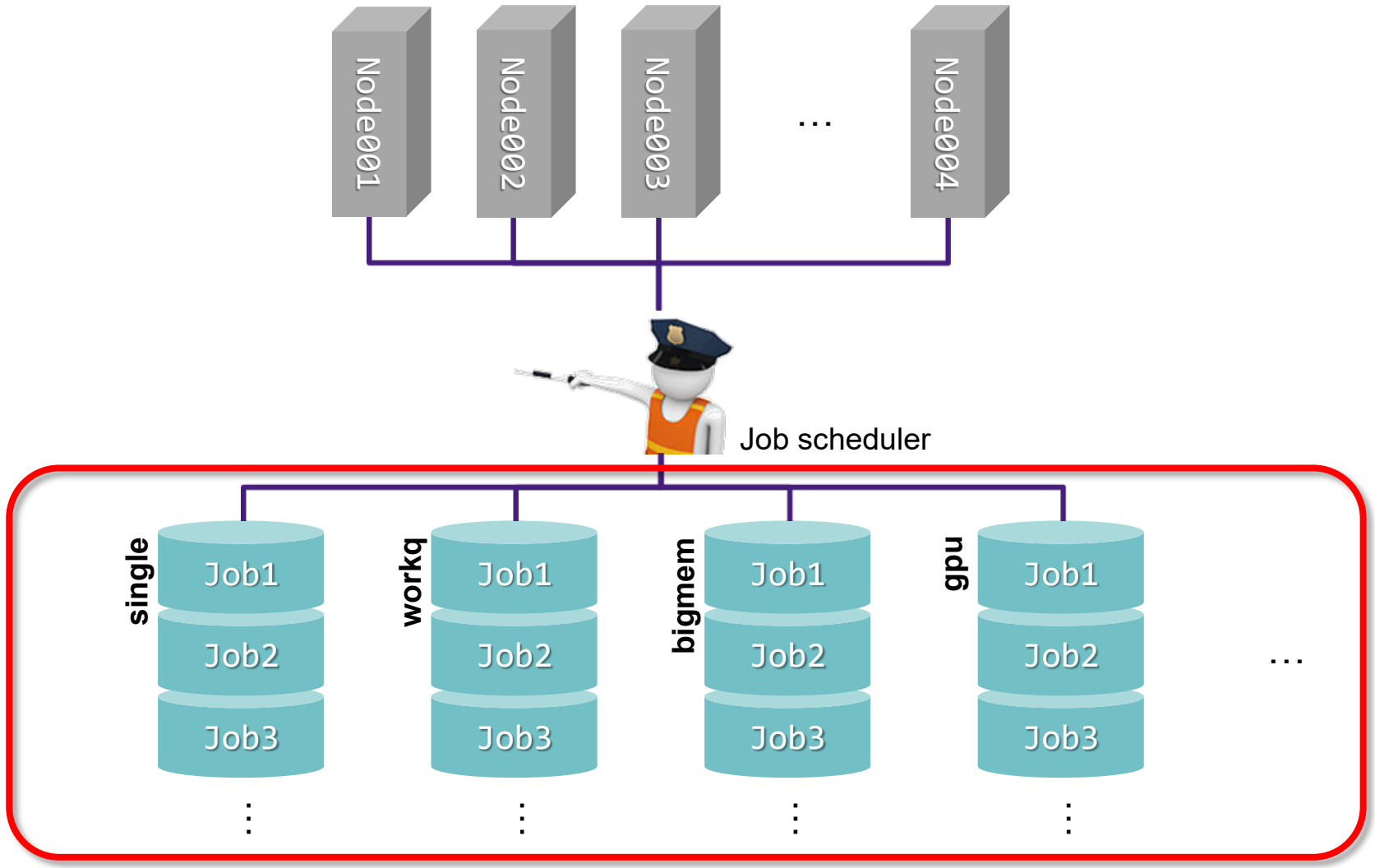
4) Job queues


Node001
Node002
Node003
...
Node004
Job scheduler
Job1
Job2
Job3

4) Job queues


Node001
Node002
Node003
...
Node004
Job scheduler
Job1
Job2
Job3
⋮

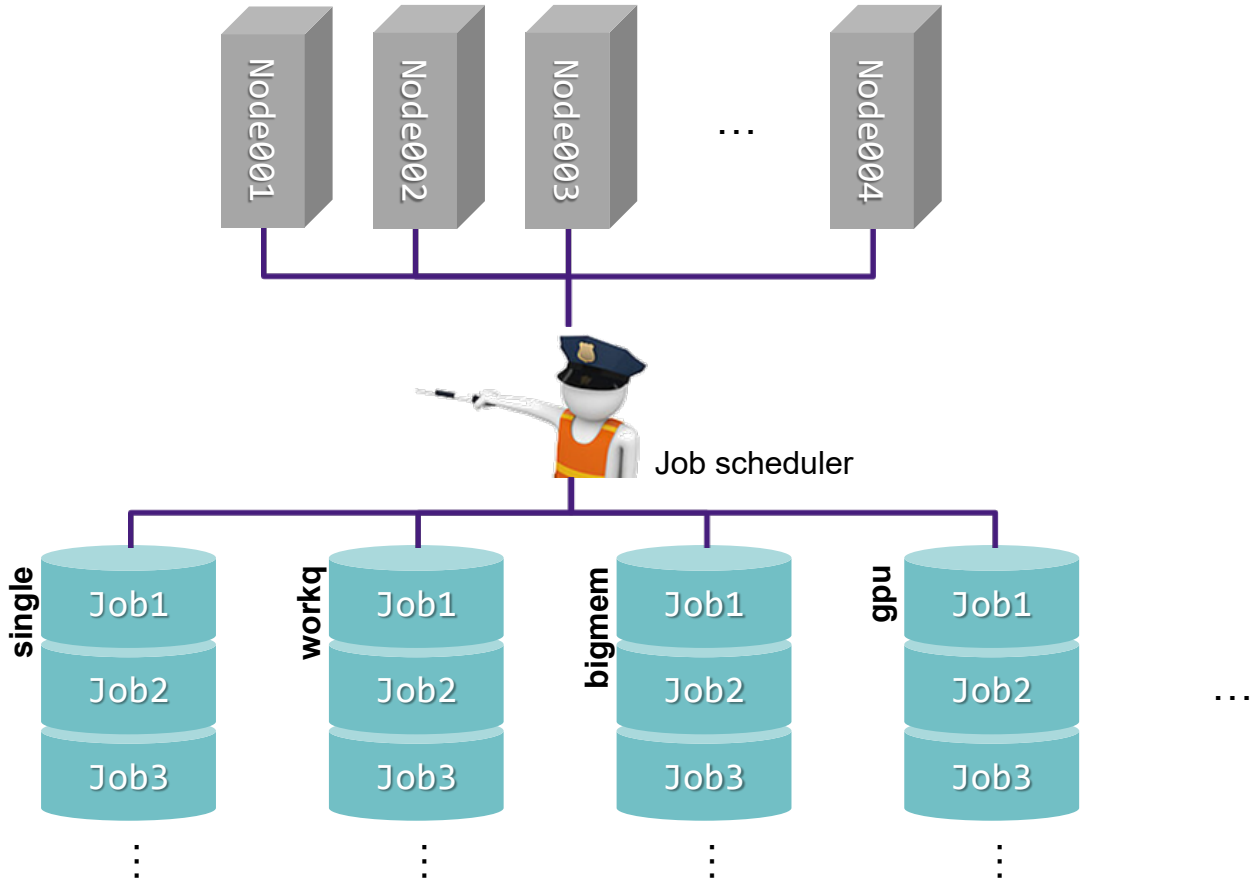
4) Job queues


Node001
Node002
Node003
...
Node004
Job scheduler
single
Job1
Job2
Job3
workq
Job1
Job2
Job3
bigmem
Job1
Job2
Job3
gpu
Job1
Job2
Job3
...

4) Job queues

a) Definition

- Different groups / lines where jobs are being grouped into
- Must pick one queue to submit job
- Goal: Use the resources more efficiently



4) Job queues

a) Definition

4) Job queues

b) Available queues

i. workq / checkpt

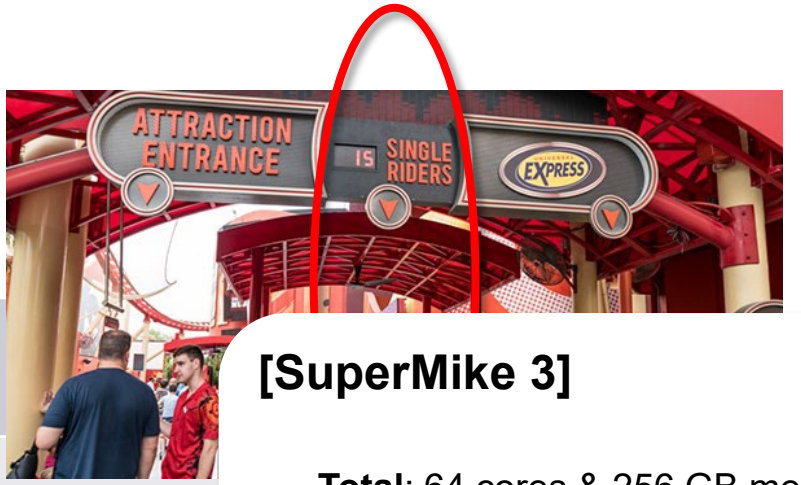
Description		• General purposes • Most likely your **default** queue • Difference: **non-preemptable** (workq) vs. **preemptable** (checkpt)
Names		• All clusters: **workq** / **checkpt**
Resource availability	**Nodes**	• **One** or **multiple** • Up to a maximum
	Cores	• **All** cores on the node(s)
	Memory	• **All** memory on the node(s)
Max duration		• **72** hours (3 days)

4) Job queues

b) Available queues

ii. single

Description		• Only need a **portion** of one node
Names		• All clusters: **single**
Resource availability	**Nodes**	• A **portion** of one node
	Cores	• PBS: **1**/**2**/**4**/**6**/**8** • Slurm: **1** ~ **all** cores
	Memory	• A **portion**, **proportional** to the number of requested cores
Max duration		• **168** hours (7 days)



[SuperMike 3]

- **Total**: 64 cores & 256 GB memory
  → 4 GB / core
- **Request**: 10 cores
  → 40 GB memory

b) Available queues

iii. bigmem

Description		• Your job needs **large memory**
Names		• All clusters: **bigmem**
Resource availability	**Nodes**	• **One** or **multiple**
	Cores	• **All** cores on the node
	Memory	• **All** memory on the node
Max duration		• **72** hours (3 days)

4) Job queues

b) Available queues

iv. GPU

Description		• Your job needs **GPU**	
Names		• SMIC: **v100** • Deep Bayou (*): **gpu**, **nvlink** • SuperMike 3 (*): **gpu** • QB3: **gpu**	• Deep Bayou (* • SuperMike 3 (*): **gp**
Resource availability	**Nodes**	• **One** or **multiple**	• **Portion** of one node
	Cores	• **All** cores on the node	• **Portion** of one node
	Memory	• **All** memory on the node	• **Portion** of one node
	GPU	• **All** GPU devices on the node	• **1** ~ **all** GPU devices
Max duration		• **72** hours (3 days)	

[SuperMike 3]

- **Total**: 64 cores & 4 GPUs
  → 16 cores / GPU
- **Request**: 3 GPUs
  → 48 cores

4) Job queues

c) Queues by clusters (LSU HPC)

Cluster	Queue	Cores per node (ppn)	Max running jobs	Max nodes per user
SuperMIC	workq	20	45 (*global*)	86
	checkpt			
	single	1,2,4,6,8,16		
	v100	36		2
	bigmem	28		3
DeepBayou	gpu	24,48	4 (*global*)	4
	nvlink	12,24,36,48		2
SuperMike3	workq	64	32 (*global*)	96
	checkpt			
	single	1 ~ 64		
	gpu	16,32,48,64		4
	bigmem	64		4

4) Job queues

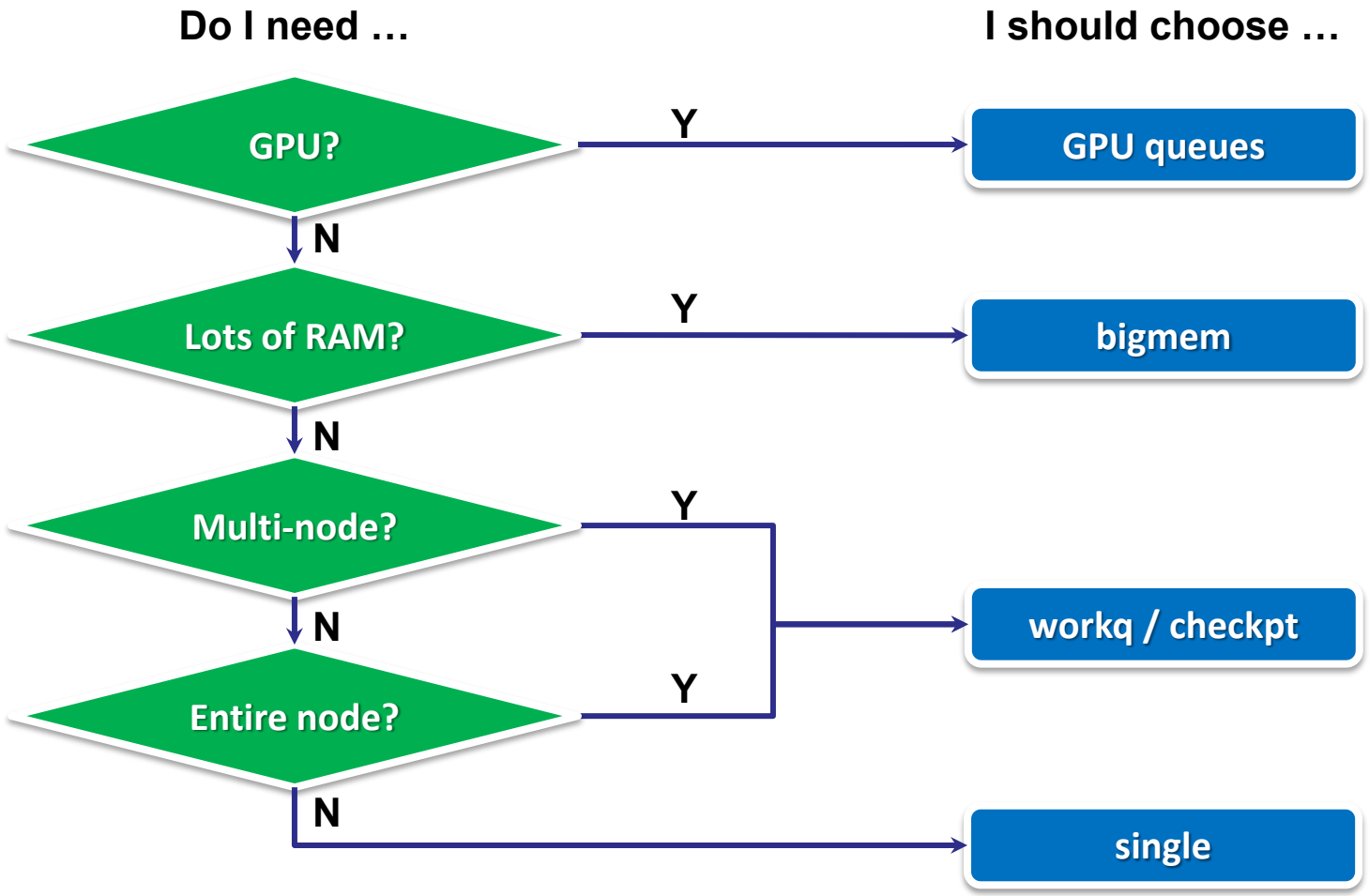
c) Queues by clusters (LONI)

Cluster	Queue	Cores per node (ppn)	Max running jobs	Max nodes per user
QB-2	workq	20	32 (*global*)	128
	checkpt			
	single	1,2,4,6,8		
	bigmem	48		1
QB-3	workq	48	32 (*global*)	96
	checkpt			
	single	1 ~ 48		
	gpu	48		4
	bigmem	48		2

d) Choose your queue

Large enough …	Small enough …
• To successfully complete your job	• To ensure quick turnaround • Not to waste resources for other users

d) Choose your queue



d) Choose your queue

Test

My job …	Queue choice? (include number of nodes / cores)
• **SMIC** • MPI code, needs 100 CPU cores - Hint: SMIC has 20 cores / node	**workq** / **checkpt** (5 nodes, 20 cores per node)
• **SuperMike 3** • Uses 3 GPUs to train a neural network - Hint: SuperMike 3 has 64 cores / node, 4 GPUs / node → 16 cores / GPU	**gpu** (1 node, 48 cores per node)
• **QB-3** • Single-core serial code • Needs to store and process 30 GB data in RAM - Hint: QB-3 has 192 GB RAM / node, 4 GB RAM / core	**single** (1 node, 8 cores per node)

4) Job queues

e) Useful commands to check queues

i. qstat –q : All queue information

```
(base) [jasonli3@mike2 ~]$ qstat -q
Queue             Memory CPU Time Walltime Node  Run Que Lm State
---------------- ------ -------- -------- ----  --- --- -- -----
admin               --      --       --    --    0   0 --  E R
single              --      --    168:00:00   1    0   0 --  E R
checkpt             --      --    72:00:00   --    3   0 --  E R
workq               --      --    72:00:00   --   12   0 --  E R
bigmem              --      --    72:00:00   --    0   0 --  E R
gpu                 --      --    72:00:00   --    0   0 --  E R
                                                 ----- -----
                                                   15     0
```

4) Job queues

e) Useful commands to check queues

ii. qfree : Free nodes in each queue

```
(base) [jasonli3@mike2 ~]$ qfree
PBS total nodes: 183,  free: 120,  busy: 58,  down: 2,  use: 31%
PBS workq nodes: 171,  free: 108,  busy: 54,  queued: 0
PBS single nodes: 171,  free: 108,  busy: 0,  queued: 0
PBS checkpt nodes: 171,  free: 108,  busy: 4,  queued: 0
PBS bigmem nodes: 4,  free: 4,  busy: 0,  queued: 0
PBS gpu nodes: 8,  free: 8,  busy: 0,  queued: 0
```

4) Job queues

e) Useful commands to check queues

iii. sinfo (Slurm only) : Detailed node health information of all queues

```
(base) [jasonli3@mike2 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
single*      up 7-00:00:00      2  inval mike[035,138]
single*      up 7-00:00:00      1   comp mike144
single*      up 7-00:00:00     58  alloc mike[008-026,031-034,036-044,046-050,141-143,148-162,167-169]
single*      up 7-00:00:00    108   idle mike[001-007,027-030,045,051-137,139,145-146,163-166,170-171]
single*      up 7-00:00:00      2   down mike[140,147]
checkpt      up 3-00:00:00      2  inval mike[035,138]
checkpt      up 3-00:00:00      1   comp mike144
checkpt      up 3-00:00:00     58  alloc mike[008-026,031-034,036-044,046-050,141-143,148-162,167-169]
checkpt      up 3-00:00:00    108   idle mike[001-007,027-030,045,051-137,139,145-146,163-166,170-171]
checkpt      up 3-00:00:00      2   down mike[140,147]
workq        up 3-00:00:00      2  inval mike[035,138]
workq        up 3-00:00:00      1   comp mike144
workq        up 3-00:00:00     58  alloc mike[008-026,031-034,036-044,046-050,141-143,148-162,167-169]
workq        up 3-00:00:00    108   idle mike[001-007,027-030,045,051-137,139,145-146,163-166,170-171]
workq        up 3-00:00:00      2   down mike[140,147]
bigmem       up 3-00:00:00      4   idle mike[172-175]
gpu          up 3-00:00:00      8   idle mike[176-183]
```

Summary

1. **Basic concepts**
   a) How job works on clusters
   b) Job scheduler and how it works
2. **Preparing my job**
   a) Basic principles
      - "**large enough**" and "**small enough**"
   b) Information you need to tell job scheduler:
      - Duration
      - Number of nodes & cores
      - Job queue

Break

1) **Have your terminal open and ready to connect to HPC**
2) **Download our testing code (π calculation) to your /home directory**
   - http://www.hpc.lsu.edu/training/weekly-materials/Downloads/pi_Jason.tar.gz
   - Hint: use *wget* command

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

3. Submitting a job

- **Two types of jobs:**

1) Interactive job

- Runs **in terminal** (just like using a local machine)
- **Can interact** with the job while running

2) Batch job

- Submit to server and runs **by itself**, until finished or error
- **Cannot interact** with the job while running

3. Submitting a job

- **Two types of jobs:**

	1) Interactive job	2) Batch job
Pros	• Can interact and monitor with job in real time	• Submit and leave it
Cons	• Waiting for human intervention is the opposite of “high performance”	• Cannot edit or interact with job while running
Ideal for	• Debugging and testing • Large compilation	• Production

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

1) Interactive job

a) Command

PBS	Slurm

1) Interactive job

a) Command

PBS	Slurm
`qsub -I [options]`	`srun [options] --pty bash` (Or any other shell of your preference)

1) Interactive job

a) Command

PBS	Slurm

`qsub -I \`
`  -X \`
`  -A <Allocation name> \`
`  -q <Queue name> \`
`  -l walltime=<HH:MM:SS>,nodes=<# of nodes>:ppn=<# of cores PER NODE>`
	`srun \`
	`  --x11 \`
	`  -A <Allocation name> \`
	`  -p <Queue name> \`
	`  -t <HH:MM:SS> \`
	`  -N <# of nodes> \`
	`  -n <# of TOTAL cores> \`
	`  --pty bash`

1) Interactive job

a) Command

PBS	Slurm

`qsub -I \`
`  -X \`
`  -A <Allocation name> \`
`  -q <Queue name> \`
`  -l walltime=<HH:MM:SS>,nodes=<# of nodes>:ppn=<# of cores PER NODE>`
	`srun \`
	`  --x11 \`
	`  -A <Allocation name> \`
	`  -p <Queue name> \`
	`  -t <HH:MM:SS> \`
	`  -N <# of nodes> \`
	`  -n <# of TOTAL cores> \`
	`  --pty bash`

Enable X11 forwarding

1) Interactive job

a) Command

PBS	Slurm

`qsub -I \`
`  -X \`
`  -A <Allocation name> \`
`  -q <Queue name> \`
`  -l walltime=<HH:MM:SS>,nodes=<# of nodes>:ppn=<# of cores PER NODE>`
	`srun \`
	`  --x11 \`
	`  -A <Allocation name> \`
	`  -p <Queue name> \`
	`  -t <HH:MM:SS> \`
	`  -N <# of nodes> \`
	`  -n <# of TOTAL cores> \`
	`  --pty bash`

Allocation name

1) Interactive job

a) Command

PBS	Slurm

`qsub -I \`
`  -X \`
`  -A <Allocation name> \`
`  -q <Queue name> \`
`  -l walltime=<HH:MM:SS>,nodes=<# of nodes>:ppn=<# of cores PER NODE>`
	`srun \`
	`  --x11 \`
	`  -A <Allocation name> \`
	`  -p <Queue name> \`
	`  -t <HH:MM:SS> \`
	`  -N <# of nodes> \`
	`  -n <# of TOTAL cores> \`
	`  --pty bash`

Queue name

1) Interactive job

a) Command

PBS	Slurm

`qsub -I \`
`  -X \`
`  -A <Allocation name> \`
`  -q <Queue name> \`
`  -l walltime=<HH:MM:SS>,nodes=<# of nodes>:ppn=<# of cores PER NODE>`
	`srun \`
	`  --x11 \`
	`  -A <Allocation name> \`
	`  -p <Queue name> \`
	`  -t <HH:MM:SS> \`
	`  -N <# of nodes> \`
	`  -n <# of TOTAL cores> \`
	`  --pty bash`

Wall time, number of nodes, number of cores

1) Interactive job

a) Command

PBS	Slurm

`qsub -I \`
`  -X \`
`  -A <Allocation name> \`
`  -q <Queue name> \`
`  -l walltime=<HH:MM:SS>,nodes=<# of nodes>:ppn=<# of cores PER NODE>`
	`srun \`
	`  --x11 \`
	`  -A <Allocation name> \`
	`  -p <Queue name> \`
	`  -t <HH:MM:SS> \`
	`  -N <# of nodes> \`
	`  -n <# of TOTAL cores> \`
	`  --pty bash`

Does not change with # of nodes

Scales proportionally with # of nodes

1) Interactive job

a) Command

PBS	Slurm

`qsub -I \`
`-X \`
`-A <A`
`-q <Q`
`-l wa[illegible]`
`nodes>:ppn=<# of cores PER NODE>`
	`run \`
	`--x11 \`
	`-A <Allocation name> \`
	`-p <Queue name> \`
	`-t <HH:MM:SS> \`
	`-N <# of nodes> \`
	`-n <# of TOTAL cores> \`
	`--pty bash`

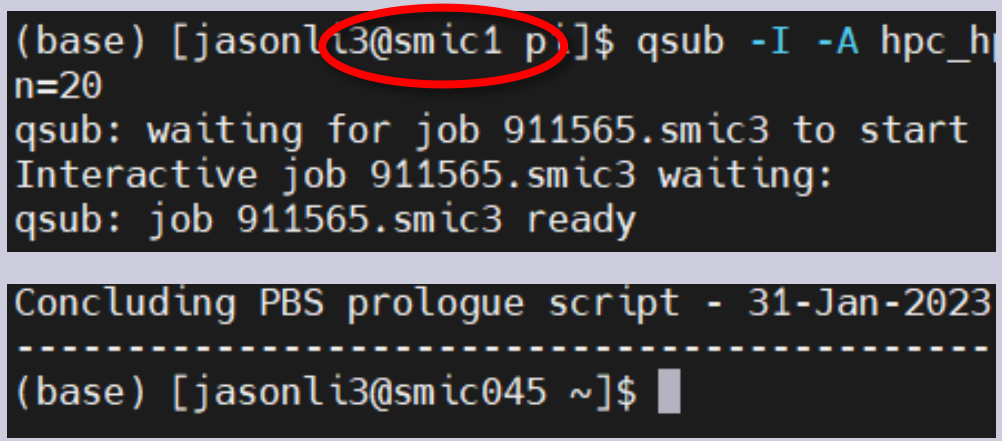
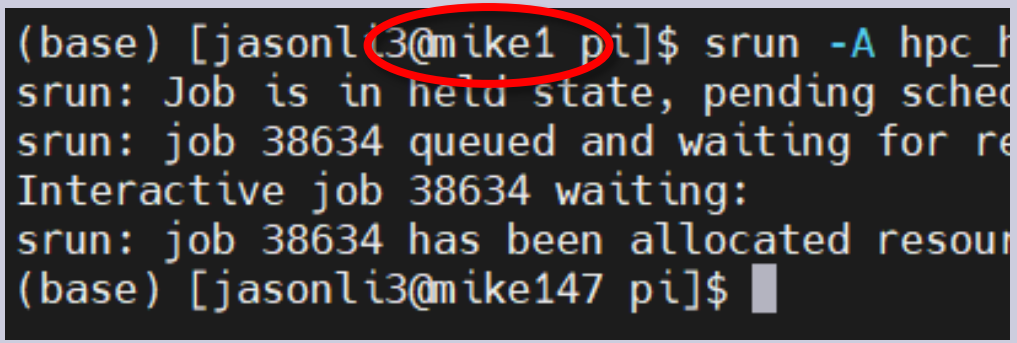
For those who run MPI / OpenMP hybrid –

-n <# of tasks>
-c <# of cores per task>

⟹ <n> * <c> = <# of TOTAL cores>

1) Interactive job

b) Starting an interactive job

PBS	Slurm
	

1) Interactive job

b) Starting an interactive job

PBS	Slurm

```
(base) [jasonli3@smic1 pi]$ qsub -I -A hpc_h
n=20
qsub: waiting for job 911565.smic3 to start
Interactive job 911565.smic3 waiting:
qsub: job 911565.smic3 ready

Concluding PBS prologue script - 31-Jan-2023
--------------------------------------------
(base) [jasonli3@smic045 ~]$
```

```
(base) [jasonli3@mike1 pi]$ srun -A hpc_h
srun: Job is in held state, pending schec
srun: job 38634 queued and waiting for re
Interactive job 38634 waiting:
srun: job 38634 has been allocated resour
(base) [jasonli3@mike147 pi]$
```

Successfully started: on a computing node (**3-digit** number)

1) Interactive job

b) Starting an interactive job

PBS	Slurm

```
(base) [jasonli3@smic1 pi]$ qsub -I -A hpc_h
n=20
qsub: waiting for job 911565.smic3 to start
Interactive job 911565.smic3 waiting:
qsub: job 911565.smic3 ready

Concluding PBS prologue script - 31-Jan-2023
----------------------------------------
(base) [jasonli3@smic045 ~]$
```

```
(base) [jasonli3@mike1 pi]$ srun -A hpc_h
srun: Job is in held state, pending schec
srun: job 38634 queued and waiting for re
Interactive job 38634 waiting:
srun: job 38634 has been allocated resour
(base) [jasonli3@mike147 pi]$
```

PBS: Job starts in **/home** directory
Slurm: Job starts in **where the job was submitted**

1) Interactive job

c) One more thing about GPU jobs …

PBS	Slurm

```
srun \
     --x11 \
     -A <Allocation name> \
     -p <Queue name> \
     -t <HH:MM:SS> \
     -N1 \
     -n16 \
     --gres=gpu:1 \
     --pty bash
```

[**GPU**] Request **1** out of 4 GPUs on SuperMike 3

d) Running an interactive job

i. Serial (single-thread)

ii. Parallel (MPI)

*** Slurm + interactive + MPI:**

$ srun <mpi_executable> → **Will hang**

$ srun **--overlap** <mpi_executable> → **Will run**

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

2) Batch job

- **What do you need?**

i. A **batch file** (containing job parameters and bash scripts)

ii. Run a **submission command** to submit this batch file

2) Batch job

a) Batch file

PBS	Slurm

2) Batch job

a) Batch file

PBS	Slurm

`#!/bin/bash`
`#PBS -A <Allocation name>`
`#PBS -q workq`
`#PBS -l walltime=1:00:00`
`#PBS -l nodes=1:ppn=20`
	`#!/bin/bash`
	`#SBATCH -A <Allocation name>`
	`#SBATCH -p workq`
	`#SBATCH -t 1:00:00`
	`#SBATCH -N 1`
	`#SBATCH -n 64`

`module load python`
 	
`cd $PBS_O_WORKDIR`
`./pi_serial.out 100000000`
	`module load python`
	
	`cd $SLURM_SUBMIT_DIR`
	`./pi_serial.out 100000000`

[Header]
Job parameters

[Body]
Commands to run after job starts

2) Batch job

a) Batch file

PBS

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20


module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

Slurm

```
#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -t 1:00
#SBATCH -N 1
#SBATCH -n 64

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

Shell type ("***shebang***")

2) Batch job

a) Batch file

PBS

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20


module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

Slurm

Allocation name

```
#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -t 1:00:00
#SBATCH -N 1
#SBATCH -n 64

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

2) Batch job

a) Batch file

PBS

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20


module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

Slurm

Queue name

```
#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -t 1:00:00
#SBATCH -N 1
#SBATCH -n 64

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

2) Batch job

a) Batch file

PBS	Slurm

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20


module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

```
#!/bin/bash
#SBATCH -A <All
#SBATCH -p workq
#SBATCH -t 1:00:00
#SBATCH -N 1
#SBATCH -n 64

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

Wall time

2) Batch job

a) Batch file

PBS	Slurm

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20

module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

```
#!/bin/bash
#SBATCH -A <All
#SBATCH -p work
#SBATCH -t 1:00:00
#SBATCH -N 1
#SBATCH -n 64

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

Number of nodes & cores

a) Batch file

PBS[1]		Slurm[2]		Description	
#PBS -A		#SBATCH -A		Allocation name	
#PBS -q		#SBATCH -p		Queue name	
#PBS -l		#SBATCH -t		Resource request	Wall time
		#SBATCH -N			Number of nodes
		#SBATCH -n			Number of tasks
		#SBATCH -c			Number of cores per task
#PBS -o		#SBATCH -o		Standard output file	
#PBS -e		#SBATCH -e		Standard error file	
#PBS -m	a	#SBATCH --mail-type	FAIL	Send email when	Job aborts / fails
	b		BEGIN		Job begins
	e		END		Job ends
#PBS -M		#SBATCH --mail-user		Email address	
#PBS -N		#SBATCH -J		Job name	

LSU INFORMATION TECHNOLOGY SERVICES

[1] http://www.hpc.lsu.edu/docs/pbs.php
[2] http://www.hpc.lsu.edu/docs/slurm.php

2) Batch job

a) Batch file

PBS	Slurm

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20

module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

```
#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -t 1
#SBATCH -N 1
#SBATCH -n 6

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

[Body]
Commands to run after job starts

2) Batch job

a) Batch file

PBS	Slurm

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20

module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

```
#!/bin/bash
#SBATCH -A <
#SBATCH -p w
#SBATCH -t 1
#SBATCH -N 1
#SBATCH -n 64

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

[Recommended] Explicitly load modules here if needed!

2) Batch job

a) Batch file

PBS	Slurm

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20

module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

```
#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -t 1:00:00
#SBATCH -N 1
#SBATCH -n 6

module load python

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000
```

Whatever commands you need to run your jobs

2) Batch job

a) Batch file

PBS

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20


module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000

```

Slurm

```
#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -t 1:00:00
#SBATCH -N 1
#SBATCH -n 64

module load python

cd $SLURM_S
./pi_serial.out 100000000

```

An empty line (avoid error)

b) Command

PBS	Slurm
`qsub <batch file name>`	`sbatch <batch file name>`

c) Useful environmental variables

PBS[1]	Slurm[2]	Description
$PBS_JOBID	$SLURM_JOBID	Job ID
$PBS_O_WORKDIR	$SLURM_SUBMIT_DIR	Job submit directory
$PBS_NODEFILE	$SLURM_JOB_NODELIST	A temp file, contains a list of allocated nodes' names (for MPI)
$PBS_NUM_NODES	$SLURM_NNODES	Number of allocated nodes
$PBS_NP	$SLURM_NTASKS	Number of allocated cores (tasks)
…	…	

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=1:00:00
#PBS -l nodes=1:ppn=20

module load python

cd $PBS_O_WORKDIR
./pi_serial.out 100000000
```

[1] http://www.hpc.lsu.edu/docs/pbs.php
[2] http://www.hpc.lsu.edu/docs/slurm.php

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

4. Manage jobs

- **Running jobs on HPC ≠ "Submit and done"**
  - Monitoring and managing jobs are part of the work

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

1) Useful commands

PBS[1]	Slurm[2]	Description
`qstat`	`squeue`	List all jobs

[1] http://www.hpc.lsu.edu/docs/pbs.php
[2] http://www.hpc.lsu.edu/docs/slurm.php

1) Useful commands

PBS[1]		Slurm[2]		Description
qstat		squeue		List all jobs
	<Job ID>		-j <Job ID>	List the job of specific ID
	-u <Username>		-u <Username>	List all jobs belong to a specific user
	<Queue name>		-p <Queue name>	List all jobs in a particular queue
qstat -n <Job ID>		scontrol show job <Job ID>		Show job details
		squeue --start		Estimated start time of queuing jobs
qdel <Job ID>		scancel <Job ID>		Cancel <Job ID>

Alter jobs after submission? → NOT allowed!

[1] http://www.hpc.lsu.edu/docs/pbs.php
[2] http://www.hpc.lsu.edu/docs/slurm.php

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

2) Monitoring job health

A job requesting n cores ≠ A job utilizing n cores

- **Goal**
  - Use the allocated resources (CPU cores, RAM, time, …) **as fully and efficiently as possible**
  - **No serious underutilizing**
  - **No serious overutilizing**

- **Things to check**
  - CPU / GPU load
  - Memory usage

2) Monitoring job health

a) **Method 1: qshow** <Job ID>

- Displays diagnostic information of a **running job**
- Can be run on **head node**

2) Monitoring job health

a) Method 1: qshow <Job ID>

```
(base) [jasonli3@mike4 ~]$ qshow 38581
PBS job: 38581, nodes: 1
Hostname  Days Load CPU U# (User:Process:VirtualMemory:Memory:Hours)
mike145    278 64.12 6033 68 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:533M:107M:13.5 yxan:lmp_mik+:748M:128M:13.5
yxan:lmp_mik+:738M:124M:13.5 yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:587M:109M:13.5 yxan:lmp_mik+:743M:128M:13.5 yxan:lmp_mik+:696M:118M:13.5
yxan:lmp_mik+:528M:101M:13.5 yxan:lmp_mik+:578M:108M:13.5 yxan:lmp_mik+:528M:105M:13.5 yxan:lmp_mik+:528M:106M:13.5 yxan:lmp_mik+:520M:105M:13.5
yxan:lmp_mik+:561M:106M:13.5 yxan:lmp_mik+:583M:109M:13.5 yxan:lmp_mik+:520M:103M:13.5 yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:738M:125M:13.5
yxan:lmp_mik+:709M:119M:13.5 yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:574M:107M:13.5 yxan:lmp_mik+:697M:121M:13.5 yxan:lmp_mik+:658M:115M:13.5
yxan:lmp_mik+:528M:102M:13.5 yxan:lmp_mik+:557M:108M:13.5 yxan:lmp_mik+:524M:105M:13.5 yxan:lmp_mik+:524M:105M:13.5 yxan:lmp_mik+:515M:102M:13.5
yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:567M:108M:13.5 yxan:lmp_mik+:566M:108M:13.5 yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:536M:105M:13.5
yxan:lmp_mik+:519M:104M:13.5 yxan:lmp_mik+:528M:103M:13.5 yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5
yxan:lmp_mik+:528M:104M:13.5 yxan:lmp_mik+:516M:101M:13.5 yxan:lmp_mik+:515M:101M:13.5 yxan:lmp_mik+:515M:104M:13.5 yxan:lmp_mik+:520M:101M:13.5
yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:520M:101M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:516M:102M:13.5 yxan:lmp_mik+:587M:110M:13.5
yxan:lmp_mik+:558M:108M:13.5 yxan:lmp_mik+:524M:102M:13.5 yxan:lmp_mik+:537M:103M:13.5 yxan:lmp_mik+:572M:109M:13.5 yxan:lmp_mik+:549M:104M:13.5
yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:528M:104M:13.5 yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:515M:103M:13.5
yxan:lmp_mik+:520M:105M:13.5 yxan:lmp_mik+:528M:105M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:515M:104M:13.5 yxan:lmp_mik+:515M:104M:13.5
yxan:slurm_s+:12M:3M yxan:srun:324M:8M yxan:srun:53M:1M
PBS_job=38581 user=yxan allocation=hpc_lipidhpre queue=checkpt total_load=64.12 cpu_hours=866.08 wall_hours=13.21 unused_nodes=0 total_nodes=1 pp
n=64 avg_load=64.12 avg_cpu=6033% avg_mem=6852mb avg_vmem=36176mb top_proc=yxan:lmp_mik+:mike145:524M:104M:13.5hr:100% toppm=yxan:lmp_mikeCpu:mik
e145:730M:125M node_processes=68
```

What to look at …	Normal behavior …	You should be concerned if …

2) Monitoring job health

a) Method 1: qshow <Job ID>

```
(base) [jasonli3@mike4 ~]$ qshow 38581
PBS job: 38581, nodes: 1
Hostname  Days Load CPU U# (User:Process:VirtualMemory:Memory:Hours)
mike145    278 64.12 6033 68 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:533M:107M:13.5 yxan:lmp_mik+:748M:128M:13.5
yxan:lmp_mik+:738M:124M:13.5 yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:587M:109M:13.5 yxan:lmp_mik+:743M:128M:13.5 yxan:lmp_mik+:696M:118M:13.5
yxan:lmp_mik+:528M:101M:13.5 yxan:lmp_mik+:578M:108M:13.5 yxan:lmp_mik+:528M:105M:13.5 yxan:lmp_mik+:528M:106M:13.5 yxan:lmp_mik+:520M:105M:13.5
yxan:lmp_mik+:561M:106M:13.5 yxan:lmp_mik+:583M:109M:13.5 yxan:lmp_mik+:520M:103M:13.5 yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:738M:125M:13.5
yxan:lmp_mik+:709M:119M:13.5 yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:574M:107M:13.5 yxan:lmp_mik+:697M:121M:13.5 yxan:lmp_mik+:658M:115M:13.5
yxan:lmp_mik+:528M:102M:13.5 yxan:lmp_mik+:557M:108M:13.5 yxan:lmp_mik+:524M:105M:13.5 yxan:lmp_mik+:524M:105M:13.5 yxan:lmp_mik+:515M:102M:13.5
yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:567M:108M:13.5 yxan:lmp_mik+:566M:108M:13.5 yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:536M:105M:13.5
yxan:lmp_mik+:519M:104M:13.5 yxan:lmp_mik+:528M:103M:13.5 yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5
yxan:lmp_mik+:528M:104M:13.5 yxan:lmp_mik+:516M:101M:13.5 yxan:lmp_mik+:515M:101M:13.5 yxan:lmp_mik+:515M:104M:13.5 yxan:lmp_mik+:520M:101M:13.5
yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:520M:101M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:516M:102M:13.5 yxan:lmp_mik+:587M:110M:13.5
yxan:lmp_mik+:558M:108M:13.5 yxan:lmp_mik+:524M:102M:13.5 yxan:lmp_mik+:537M:103M:13.5 yxan:lmp_mik+:572M:109M:13.5 yxan:lmp_mik+:549M:104M:13.5
yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:528M:104M:13.5 yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:515M:103M:13.5
yxan:lmp_mik+:520M:105M:13.5 yxan:lmp_mik+:528M:105M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:515M:104M:13.5 yxan:lmp_mik+:515M:104M:13.5
yxan:slurm_s+:12M:3M yxan:srun:324M:8M yxan:srun:53M:1M
PBS job=38581 user=yxan allocation=hpc_lipidhpre queue=checkpt total_load=64.12 cpu_hours=866.08 wall_hours=13.21 unused_nodes=0 total_nodes=1 pp
n=64 avg_load=64.12 avg_cpu=6033% avg_mem=6852mb avg_vmem=36176mb top_proc=yxan:lmp_mik+:mike145:524M:104M:13.5hr:100% toppm=yxan:lmp_mikeCpu:mik
e145:730M:125M node_processes=68
```

What to look at …	Normal behavior …	You should be concerned if …
avg_load	Close to allocated number of cores on the node	Consistently too low or too high

2) Monitoring job health

a) Method 1: qshow <Job ID>

```
(base) [jasonli3@mike4 ~]$ qshow 38581
PBS job: 38581, nodes: 1
Hostname  Days Load CPU U# (User:Process:VirtualMemory:Memory:Hours)
mike145    278 64.12 6033 68 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:533M:107M:13.5 yxan:lmp_mik+:748M:128M:13.5
yxan:lmp_mik+:738M:124M:13.5 yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:587M:109M:13.5 yxan:lmp_mik+:743M:128M:13.5 yxan:lmp_mik+:696M:118M:13.5
yxan:lmp_mik+:528M:101M:13.5 yxan:lmp_mik+:578M:108M:13.5 yxan:lmp_mik+:528M:105M:13.5 yxan:lmp_mik+:528M:106M:13.5 yxan:lmp_mik+:520M:105M:13.5
yxan:lmp_mik+:561M:106M:13.5 yxan:lmp_mik+:583M:109M:13.5 yxan:lmp_mik+:520M:103M:13.5 yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:738M:125M:13.5
yxan:lmp_mik+:709M:119M:13.5 yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:574M:107M:13.5 yxan:lmp_mik+:697M:121M:13.5 yxan:lmp_mik+:658M:115M:13.5
yxan:lmp_mik+:528M:102M:13.5 yxan:lmp_mik+:557M:108M:13.5 yxan:lmp_mik+:524M:105M:13.5 yxan:lmp_mik+:524M:105M:13.5 yxan:lmp_mik+:515M:102M:13.5
yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:567M:108M:13.5 yxan:lmp_mik+:566M:108M:13.5 yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:536M:105M:13.5
yxan:lmp_mik+:519M:104M:13.5 yxan:lmp_mik+:528M:103M:13.5 yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5
yxan:lmp_mik+:528M:104M:13.5 yxan:lmp_mik+:516M:101M:13.5 yxan:lmp_mik+:515M:101M:13.5 yxan:lmp_mik+:515M:104M:13.5 yxan:lmp_mik+:520M:101M:13.5
yxan:lmp_mik+:524M:103M:13.5 yxan:lmp_mik+:520M:101M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:516M:102M:13.5 yxan:lmp_mik+:587M:110M:13.5
yxan:lmp_mik+:558M:108M:13.5 yxan:lmp_mik+:524M:102M:13.5 yxan:lmp_mik+:537M:103M:13.5 yxan:lmp_mik+:572M:109M:13.5 yxan:lmp_mik+:549M:104M:13.5
yxan:lmp_mik+:519M:103M:13.5 yxan:lmp_mik+:528M:104M:13.5 yxan:lmp_mik+:520M:104M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:515M:103M:13.5
yxan:lmp_mik+:520M:105M:13.5 yxan:lmp_mik+:528M:105M:13.5 yxan:lmp_mik+:515M:103M:13.5 yxan:lmp_mik+:515M:104M:13.5 yxan:lmp_mik+:515M:104M:13.5
yxan:slurm_s+:12M:3M yxan:srun:324M:8M yxan:srun:53M:1M
PBS_job=38581 user=yxan allocation=hpc_lipidhpc queue=checkpt total_load=64.12 cpu_hours=866.08 wall_hours=13.21 unused_nodes=0 total_nodes=1 pp
n=64 avg_load=64.12 avg_cpu=6033% avg_mem=6852mb avg_vmem=36176mb top_proc=yxan:lmp_mik+:mike145:524M:104M:13.5hr:100% toppm=yxan:lmp_mikeCpu:mik
e145:730M:125M node_processes=68
```

What to look at …	Normal behavior …	You should be concerned if …
avg_load	Close to allocated number of cores on the node	Consistently too low or too high
ave_mem	Does not exceed total allocated memory	Exceeds total allocated memory

2) Monitoring job health

b) Method 2: top

- Displays dynamic real-time view of a **computing node**
- Must run on **computing nodes** !

  * ssh to computing nodes while job running (cannot ssh if you do not have jobs on it)

2) Monitoring job health

b) Method 2: top

```
top - 02:23:58 up 278 days, 19:17,  2 users,  load average: 63.63, 39.81, 17.49
Tasks: 981 total,  65 running, 916 sleeping,   0 stopped,   0 zombie
%Cpu(s): 90.2 us,  9.2 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.5 hi,  0.0 si,  0.0 st
MiB Mem : 257004.8 total, 211261.0 free,  41926.9 used,   3816.9 buff/cache
MiB Swap:  16641.0 total,  16580.7 free,     60.2 used. 212737.8 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
2701318 jasonli3  20   0  595668 582356   2568 R 100.0   0.2   4:08.94 TDSE_np3_e0
2701342 jasonli3  20   0  595668 581944   2616 R 100.0   0.2   4:08.90 TDSE_np3_e0
2701249 jasonli3  20   0  595668 581792   2464 R  99.7   0.2   4:08.97 TDSE_np3_e0
2701252 jasonli3  20   0  595668 514684   2520 R  99.7   0.2   4:09.00 TDSE_np3_e0
2701261 jasonli3  20   0  595668 393828   2616 R  99.7   0.1   4:08.97 TDSE_np3_e0
2701264 jasonli3  20   0  595668 581856   2532 R  99.7   0.2   4:08.92 TDSE_np3_e0
2701270 jasonli3  20   0  595668 582480   2432 R  99.7   0.2   4:08.95 TDSE_np3_e0
2701273 jasonli3  20   0  595668 581776   2448 R  99.7   0.2   4:08.81 TDSE_np3_e0
2701276 jasonli3  20   0  595668 582160   2568 R  99.7   0.2   4:08.98 TDSE_np3_e0
```

What to look at …	Normal behavior …	You should be concerned if …

2) Monitoring job health

b) Method 2: top

```
top - 02:23:58 up 278 days, 19:17,  2 users,  load average: 63.63, 39.81, 17.49
Tasks: 981 total,  65 running, 916 sleeping,   0 stopped,   0 zombie
%Cpu(s): 90.2 us,  9.2 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.5 hi,  0.0 si,  0.0 st
MiB Mem : 257004.8 total, 211261.0 free,  41926.9 used,   3816.9 buff/cache
MiB Swap:  16641.0 total,  16580.7 free,     60.2 used. 212737.8 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
2701318 jasonli3  20   0  595668 582356   2568 R 100.0   0.2   4:08.94 TDSE_np3_e0
2701342 jasonli3  20   0  595668 581944   2616 R 100.0   0.2   4:08.90 TDSE_np3_e0
2701249 jasonli3  20   0  595668 581792   2464 R  99.7   0.2   4:08.97 TDSE_np3_e0
2701252 jasonli3  20   0  595668 514684   2520 R  99.7   0.2   4:09.00 TDSE_np3_e0
2701261 jasonli3  20   0  595668 393828   2616 R  99.7   0.1   4:08.97 TDSE_np3_e0
2701264 jasonli3  20   0  595668 581856   2532 R  99.7   0.2   4:08.92 TDSE_np3_e0
2701270 jasonli3  20   0  595668 582480   2432 R  99.7   0.2   4:08.95 TDSE_np3_e0
2701273 jasonli3  20   0  595668 581776   2448 R  99.7   0.2   4:08.81 TDSE_np3_e0
2701276 jasonli3  20   0  595668 582160   2568 R  99.7   0.2   4:08.98 TDSE_np3_e0
```

What to look at …	Normal behavior …	You should be concerned if …
Load average	Close to allocated number of cores on the node	Consistently too low or too high

2) Monitoring job health

b) Method 2: top

```
top - 02:23:58 up 278 days, 19:17,  2 users,  load average: 63.63, 39.81, 17.49
Tasks: 981 total,  65 running, 916 sleeping,   0 stopped,   0 zombie
%Cpu(s): 90.2 us,  9.2 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.5 hi,  0.0 si,  0.0 st
MiB Mem : 257004.8 total, 211261.0 free,  41926.9 used,   3816.9 buff/cache
MiB Swap:  16641.0 total,  16580.7 free,     60.2 used. 212737.8 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
2701318 jasonli3  20   0  595668 582356   2568 R 100.0   0.2   4:08.94 TDSE_np3_e0
2701342 jasonli3  20   0  595668 581944   2616 R 100.0   0.2   4:08.90 TDSE_np3_e0
2701249 jasonli3  20   0  595668 581792   2464 R  99.7   0.2   4:08.97 TDSE_np3_e0
2701252 jasonli3  20   0  595668 514684   2520 R  99.7   0.2   4:09.00 TDSE_np3_e0
2701261 jasonli3  20   0  595668 393828   2616 R  99.7   0.1   4:08.97 TDSE_np3_e0
2701264 jasonli3  20   0  595668 581856   2532 R  99.7   0.2   4:08.92 TDSE_np3_e0
2701270 jasonli3  20   0  595668 582480   2432 R  99.7   0.2   4:08.95 TDSE_np3_e0
2701273 jasonli3  20   0  595668 581776   2448 R  99.7   0.2   4:08.81 TDSE_np3_e0
2701276 jasonli3  20   0  595668 582160   2568 R  99.7   0.2   4:08.98 TDSE_np3_e0
```

What to look at …	Normal behavior …	You should be concerned if …
Load average	Close to allocated number of cores on the node	Consistently too low or too high
Memory usage (not virtual memory)	Does not exceed total allocated memory	Exceeds total allocated memory

2) Monitoring job health

c) **Method 3: nvidia-smi** (for GPU only)

```
(base) [jasonli3@qbc193 ~]$ nvidia-smi
Wed Feb  1 02:38:32 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-PCIE...  On   | 00000000:3B:00.0 Off |                  Off |
| N/A   36C    P0    54W / 250W |   4155MiB / 32768MiB |     72%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-PCIE...  On   | 00000000:AF:00.0 Off |                  Off |
| N/A   36C    P0    52W / 250W |   4155MiB / 32768MiB |     78%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A    259491      C   ...che/TeraChem/bin/terachem     4147MiB |
|    1   N/A  N/A    259491      C   ...che/TeraChem/bin/terachem     4147MiB |
+-----------------------------------------------------------------------------+
```

What to look at …	Normal behavior …	You should be concerned if …

2) Monitoring job health

c) Method 3: nvidia-smi (for GPU only)

```
(base) [jasonli3@qbc193 ~]$ nvidia-smi
Wed Feb  1 02:38:32 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-PCIE...  On   | 00000000:3B:00.0 Off |                    Off |
| N/A   36C    P0    54W / 250W |   4155MiB / 32768MiB |     72%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-PCIE...  On   | 00000000:AF:00.0 Off |                    Off |
| N/A   36C    P0    52W / 250W |   4155MiB / 32768MiB |     78%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A    259491      C   ...che/TeraChem/bin/terachem     4147MiB |
|    1   N/A  N/A    259491      C   ...che/TeraChem/bin/terachem     4147MiB |
+-----------------------------------------------------------------------------+
```

What to look at …	Normal behavior …	You should be concerned if …
GPU usage	Close to 100%	Consistently too low

2) Monitoring job health

c) Method 3: nvidia-smi (for GPU only)

```
(base) [jasonli3@qbc193 ~]$ nvidia-smi
Wed Feb  1 02:38:32 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-PCIE...  On   | 00000000:3B:00.0 Off |                  Off |
| N/A   36C    P0    54W / 250W |   4155MiB / 32768MiB |     72%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-PCIE...  On   | 00000000:AF:00.0 Off |                  Off |
| N/A   36C    P0    52W / 250W |   4155MiB / 32768MiB |     78%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A    259491      C   ...che/TeraChem/bin/terachem     4147MiB |
|    1   N/A  N/A    259491      C   ...che/TeraChem/bin/terachem     4147MiB |
+-----------------------------------------------------------------------------+
```

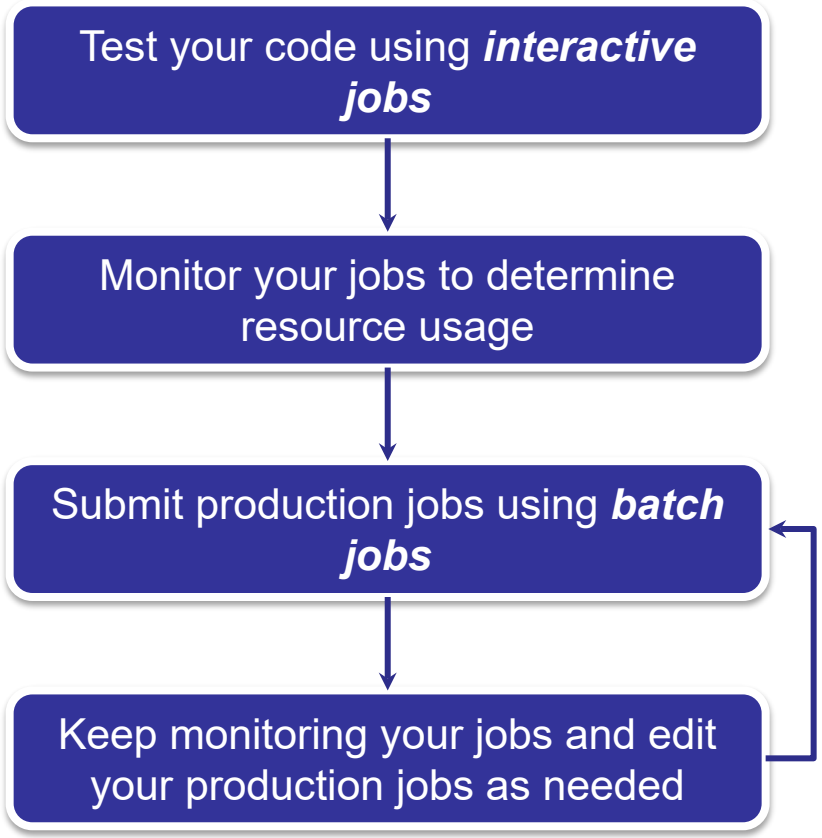
What to look at …	Normal behavior …	You should be concerned if …
GPU usage	Close to 100%	Consistently too low
Memory usage (not virtual memory)	Not used up	Used up

d) Common issues

Issue	What would happen
Exceeded memory allocation (e.g., using more memory than allocated w/ single queue)	Terminated. Receive email notice.
Exceeded ppn/core allocation (e.g., using more cores than allocated w/ single queue)	Terminated. Receive email notice.
Seriously underutilize node CPU cores / unused nodes (e.g., Requested multiple nodes but only runs on one node)	Receive email warning. (* Killed if ***completely idle*** for a long time)
Submitting to bigmem but only using little memory	Receive email warning.
Running intensive calculation on head nodes	Terminated. Receive email notice.
Submitting too many (i.e., hundreds of) single-thread jobs	Poor parallelization and bad for server. We may reach out to you to help. (Better yet, reach out to us first)

Summary

- **A typical workflow --**



Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers → **All calculation must be submitted as jobs**
2. Preparing my job
   1) Basic principles → **Large enough & small enough**
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues
3. Submitting my job
   1) Interactive job → **Good for testing and debugging**
   2) Batch job → **Good for production**
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health → **How to monitor jobs health, and how to create health jobs**

Next week

- **Basic Shell Scripting**

Contact us

- **Contact user services**
  - Email Help Ticket: sys-help@loni.org
  - Telephone Help Desk: +1 (225) 578-0900