HPC User Environment 2

Jason Li

HPC User Services

LSU HPC / LONI

sys-help@loni.org

Louisiana State University

Baton Rouge

Feb 1, 2023

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. How jobs are handled
  3. Submitting a job
  4. Manage my job

Outlines

- **HPC User Environment 2**

  1. Basic concepts
  2. How jobs are handled
  3. Submitting a job
  4. Manage my job

Outlines

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage jobs
   1) Useful commands
   2) Monitoring job health

Outlines

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage jobs
   1) Useful commands
   2) Monitoring job health

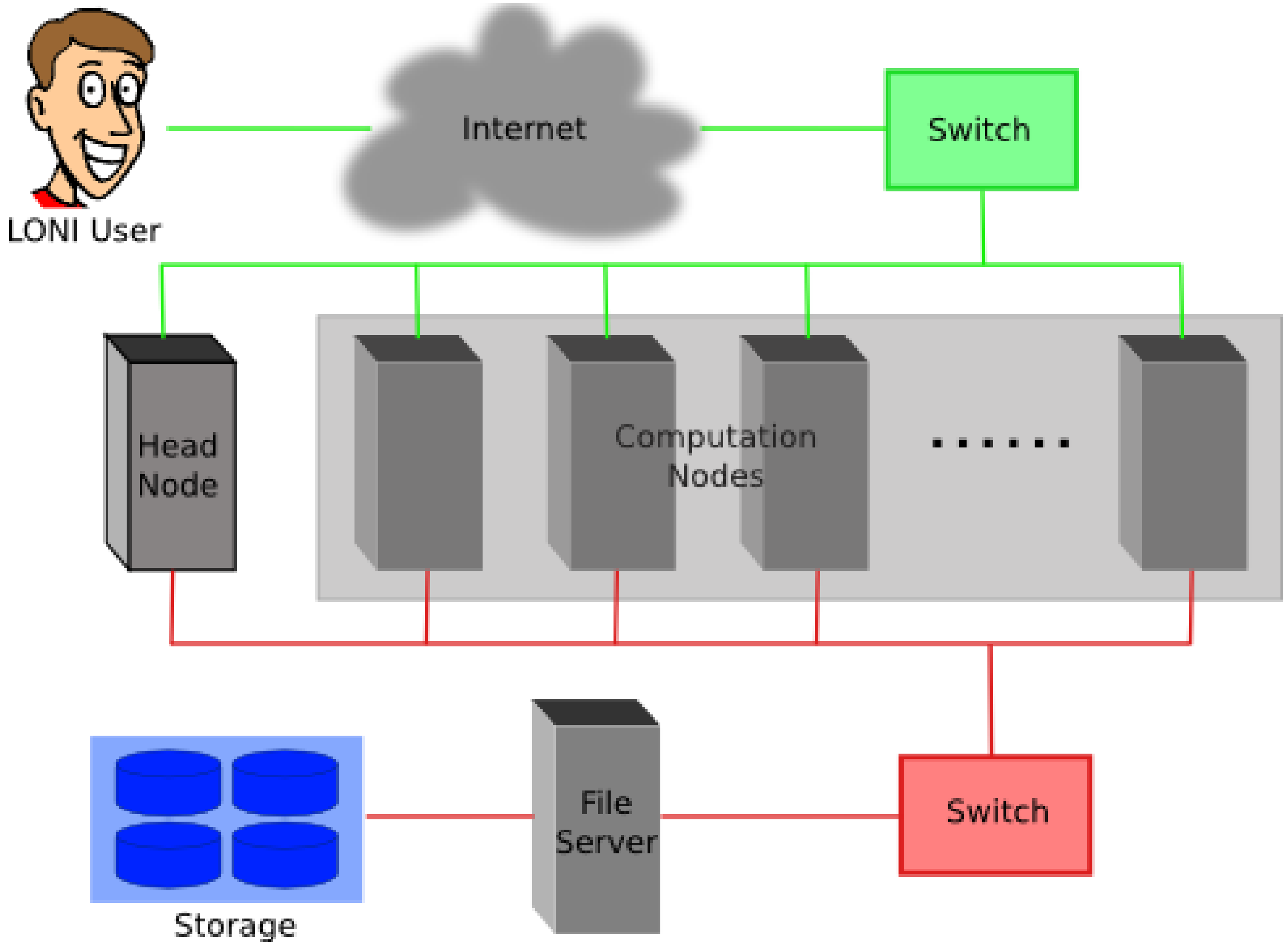
1) Previously on HPC User Environment 1…

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…

Run my code on **all** the resources you have, **however long** it takes

sudo! **yum**! **apt-get**! …

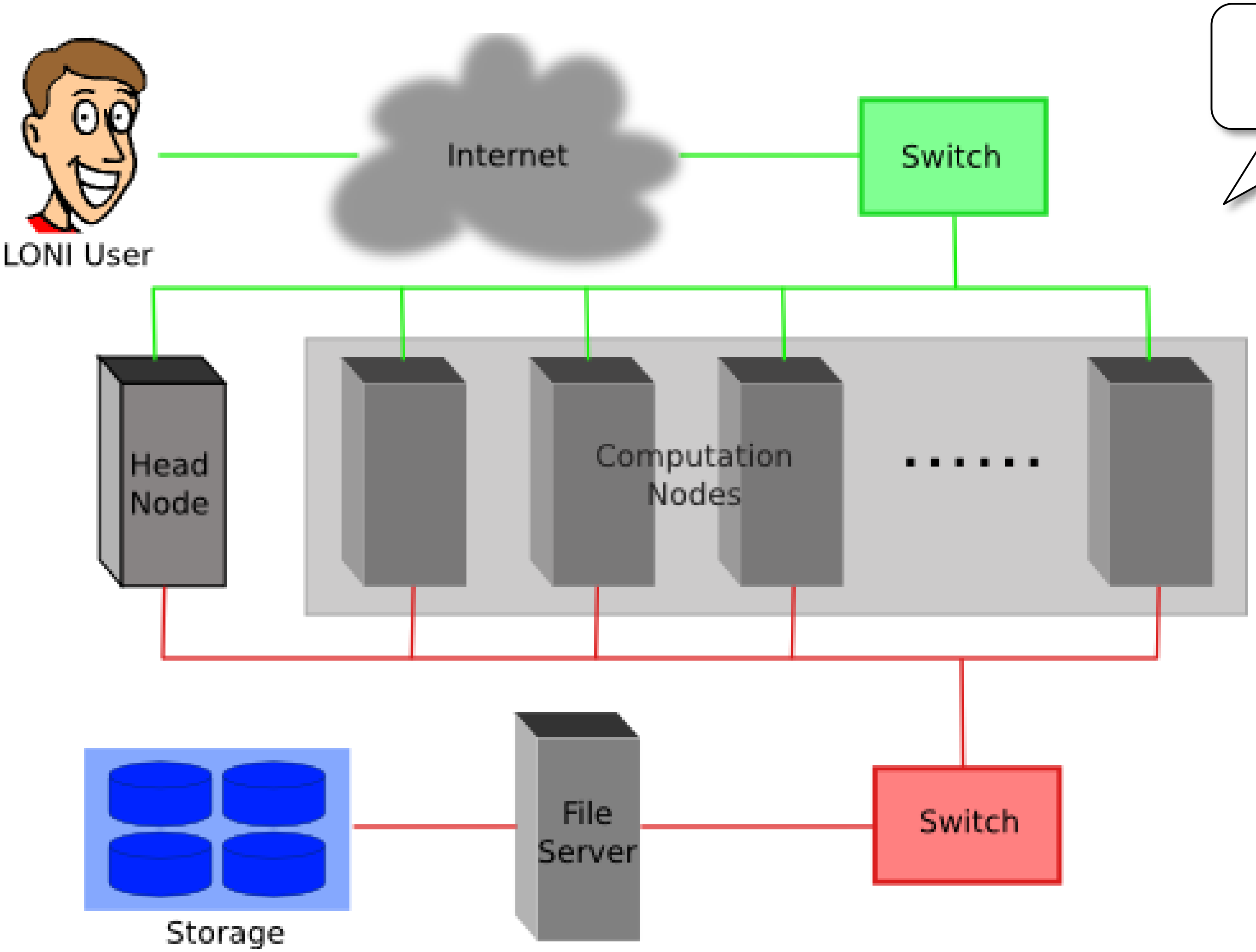
Yes, my master!

1) Previously on HPC User Environment 1…

Run my code on **all** the resources you have, **however long** it takes

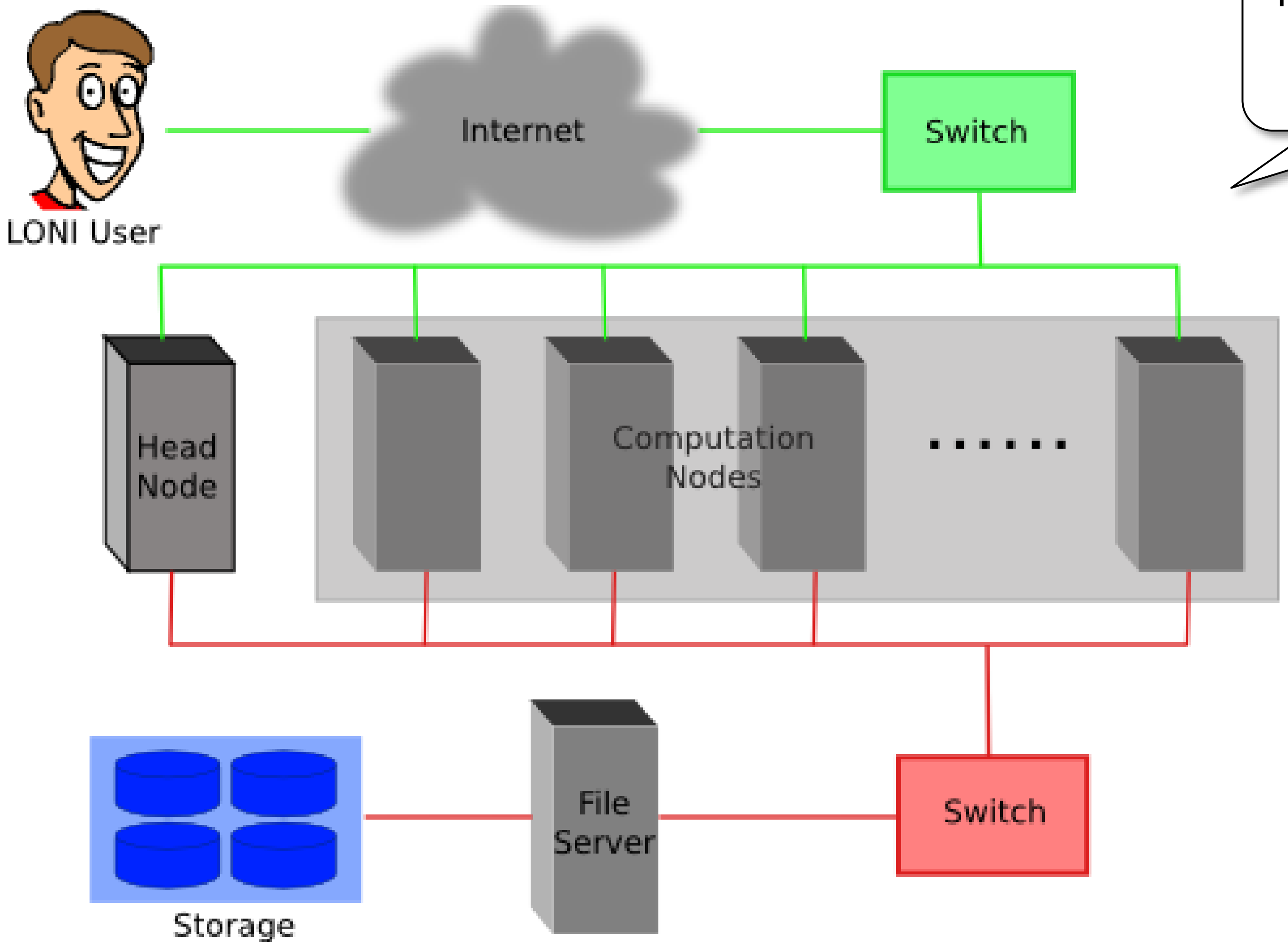
sudo! **yum**! **apt-get**! …

You are not my master, buddy!



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.



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

2) What is a “job”?

- **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

Outlines

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage jobs
   1) Useful commands
   2) Monitoring job health

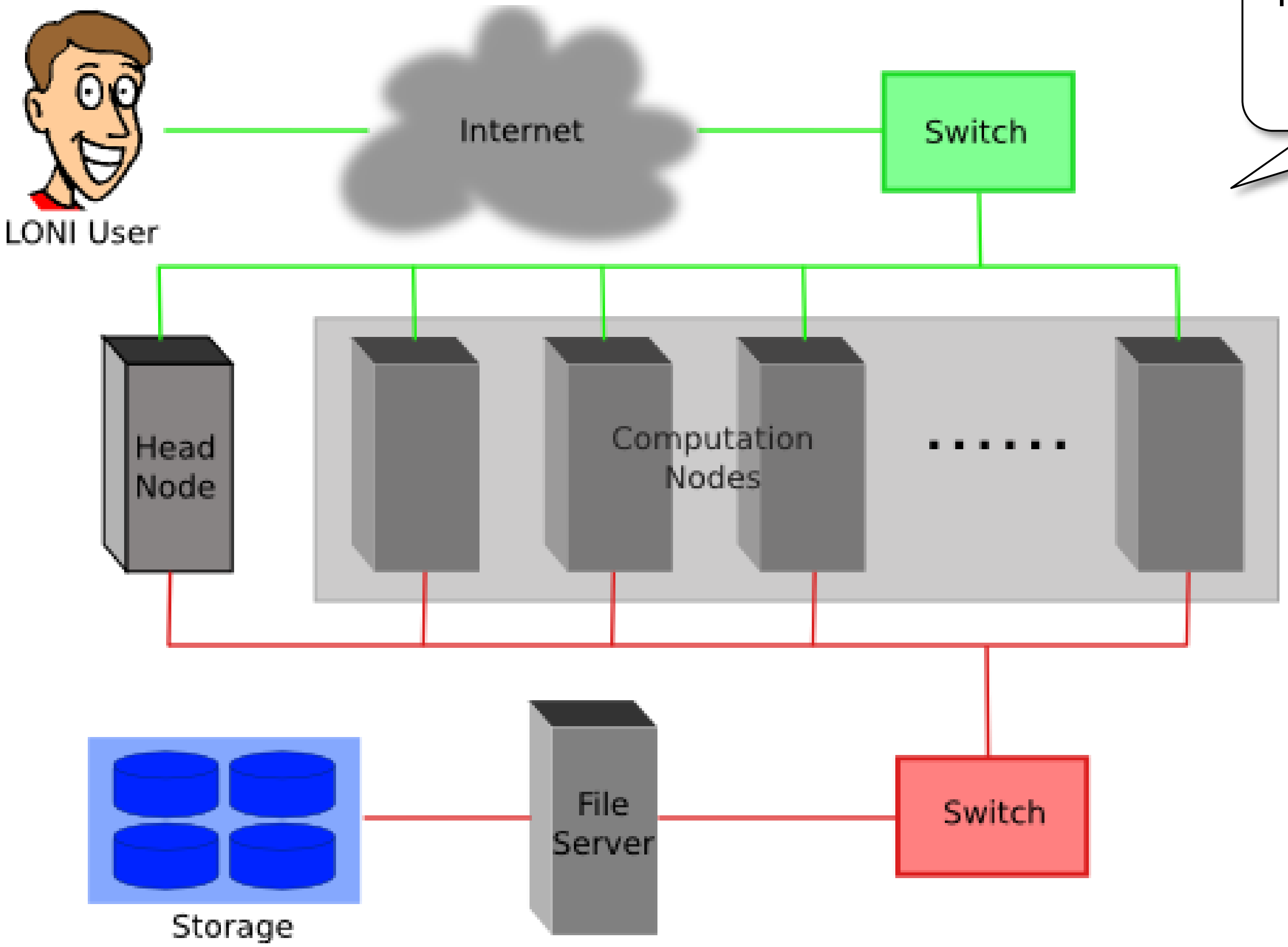
1) Job schedulers

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
    1) Job schedulers
    2) Job queues
    3) Choose your queue
3. Submitting a job
    1) Interactive job
    2) Batch job
    3) Cheat sheets
4. Manage jobs
    1) Useful commands
    2) Monitoring job health

1) Job schedulers

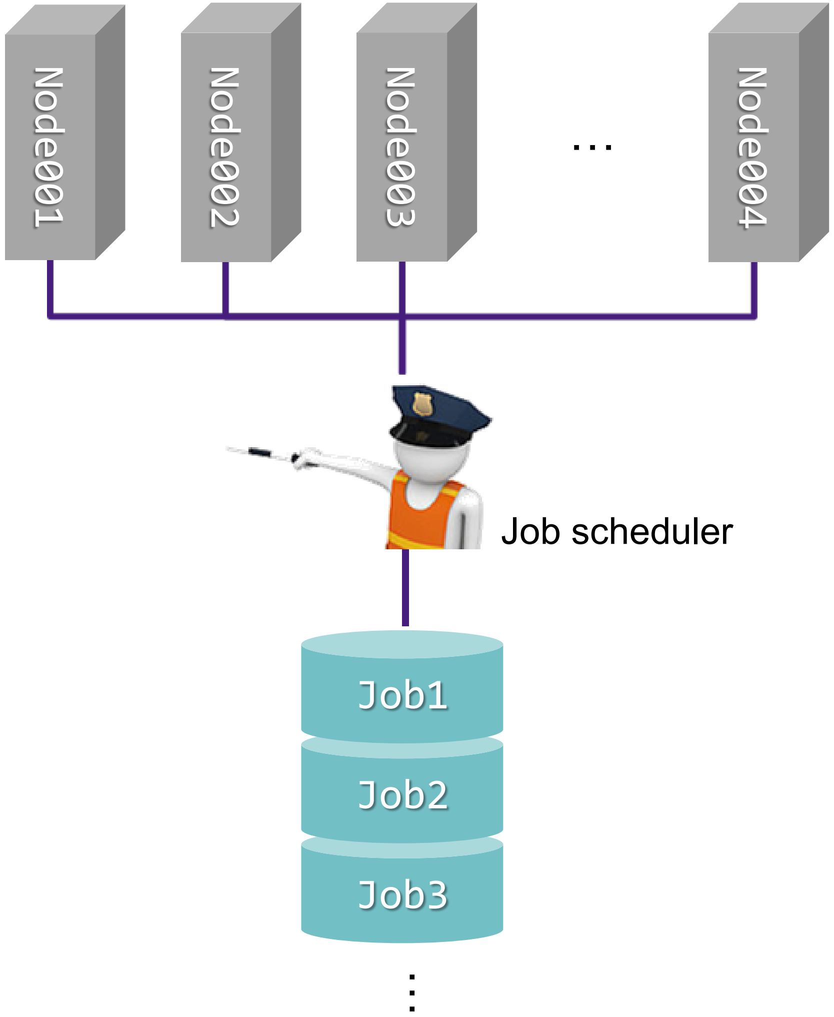
I will ask nicely. Please grant me the use of **24 cores** for **10 hours** to run my code.



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

1) Job schedulers

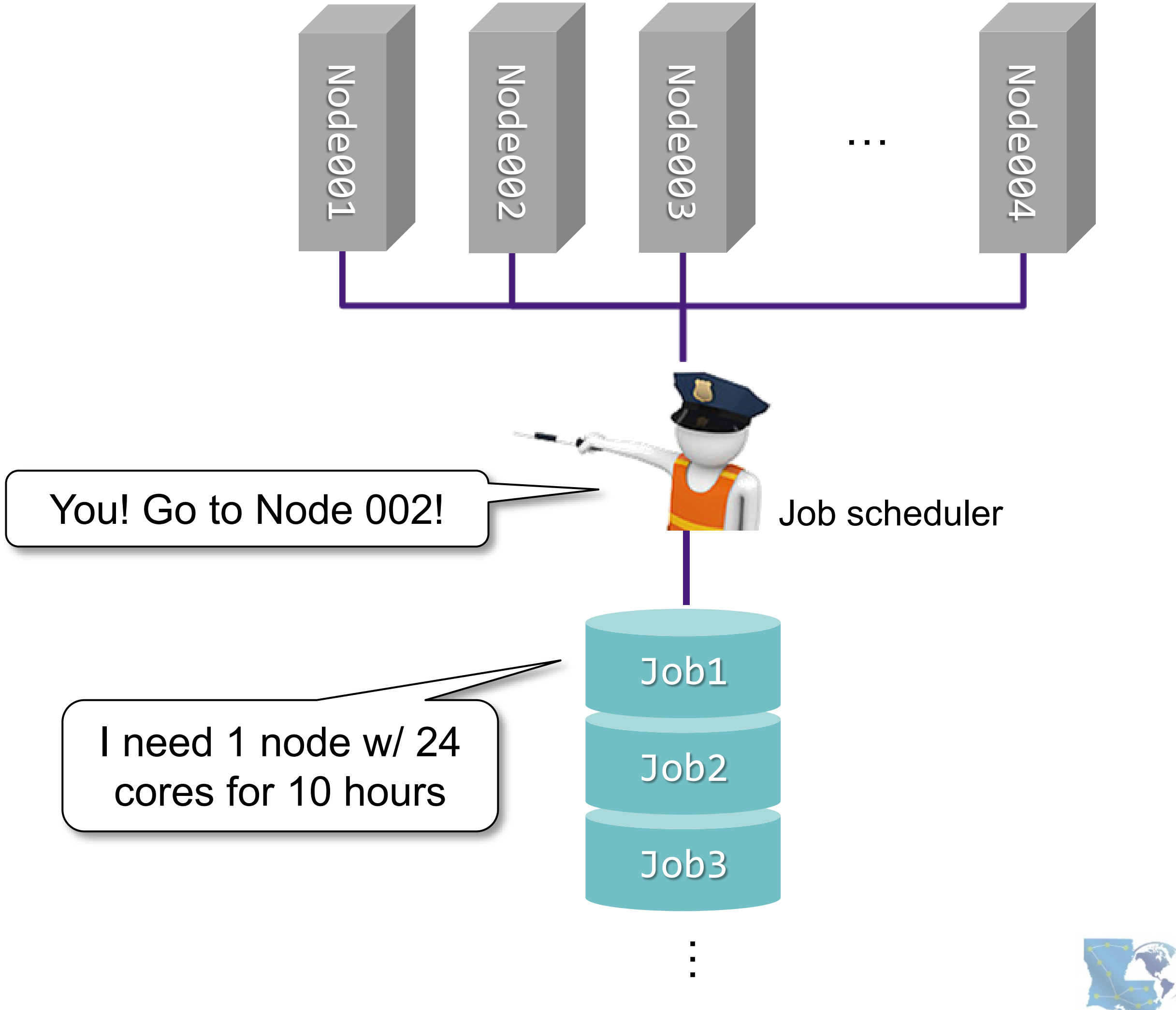
- **Job scheduler**



1) Job schedulers

- **Job scheduler**

   a) Decides which job runs when and where

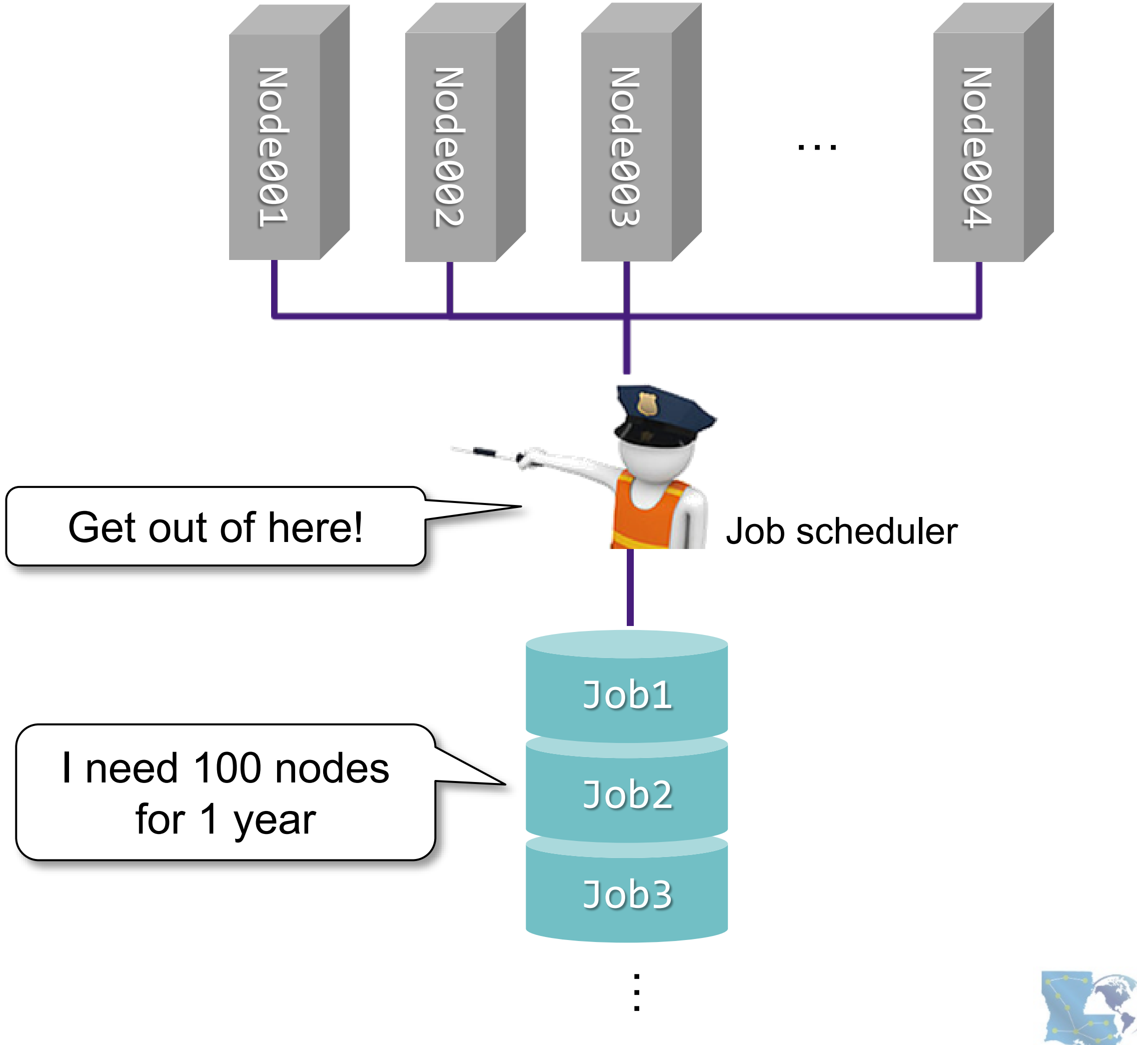


1) Job schedulers

- **Job scheduler**

a) Decides which job runs when and where

b) Enforces job policies



1) Job schedulers

- **Job scheduler**

Job scheduler's responsibilities	
• Decides which job runs when and where • Enforces job policies	

1) Job schedulers

- **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

1) Job schedulers

- **Job scheduler**

i) PBS

1) Job schedulers

- **Job scheduler**

i) PBS

ii) Slurm

1) Job schedulers

- **Job scheduler**

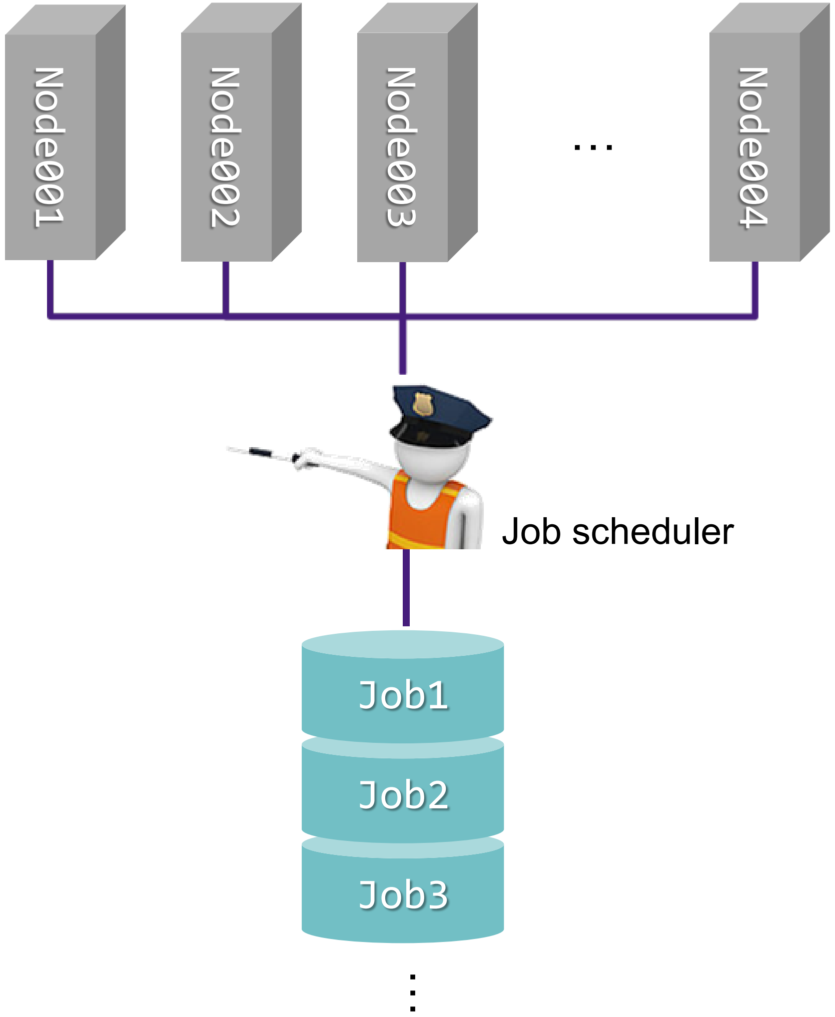
	LSU HPC	LONI
i) PBS	SMIC	QB2
ii) Slurm	Deep Bayou SuperMike III	QB3

2) Job queues

- **HPC User Environment 2**

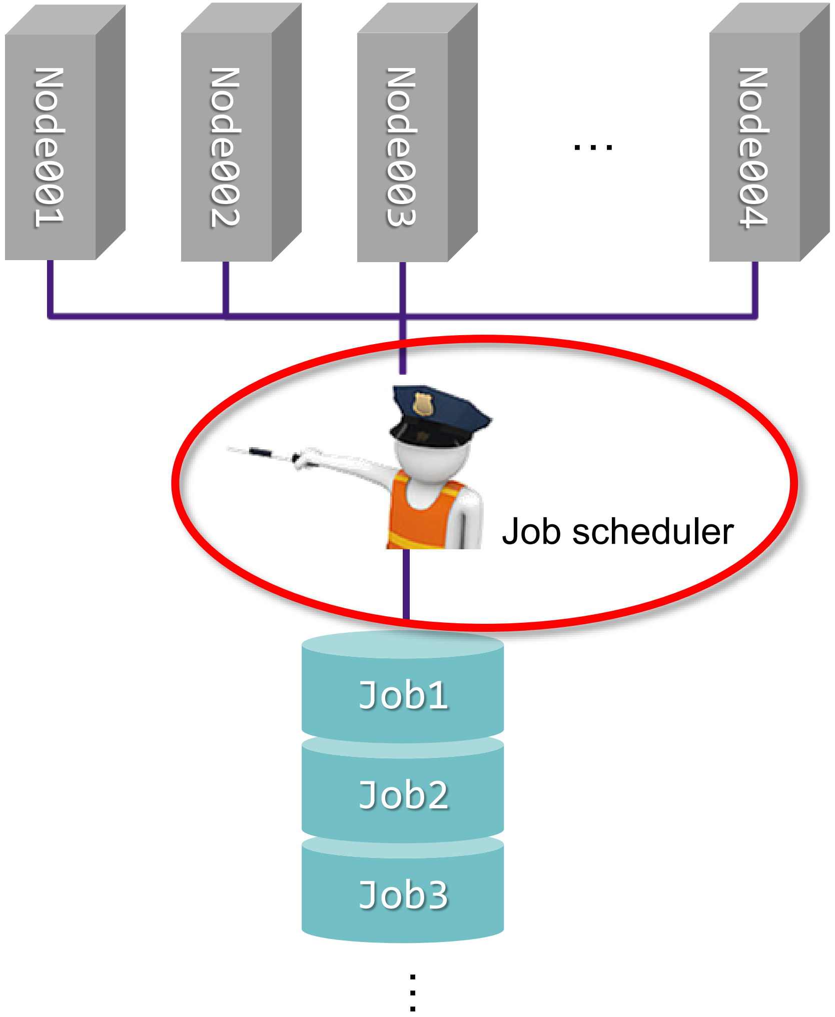
1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage jobs
   1) Useful commands
   2) Monitoring job health

2) Job queues

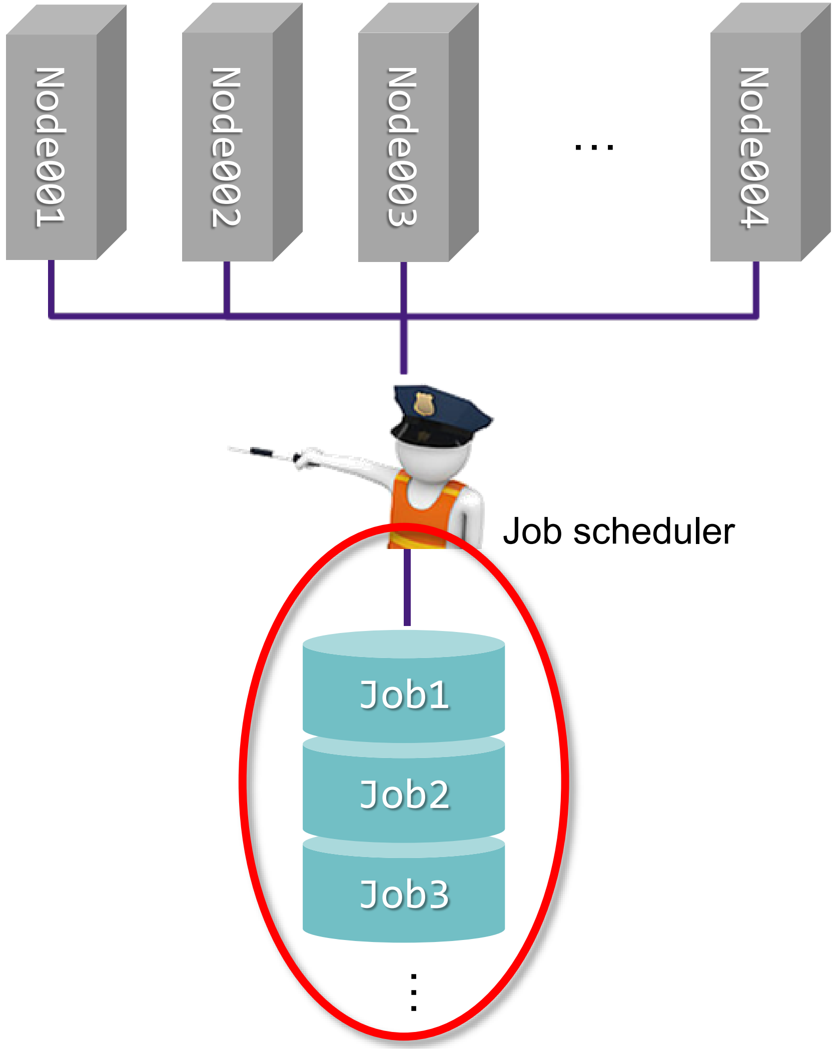


Node001
Node002
Node003
…
Node004
Job scheduler
Job1
Job2
Job3
⋮

2) Job queues

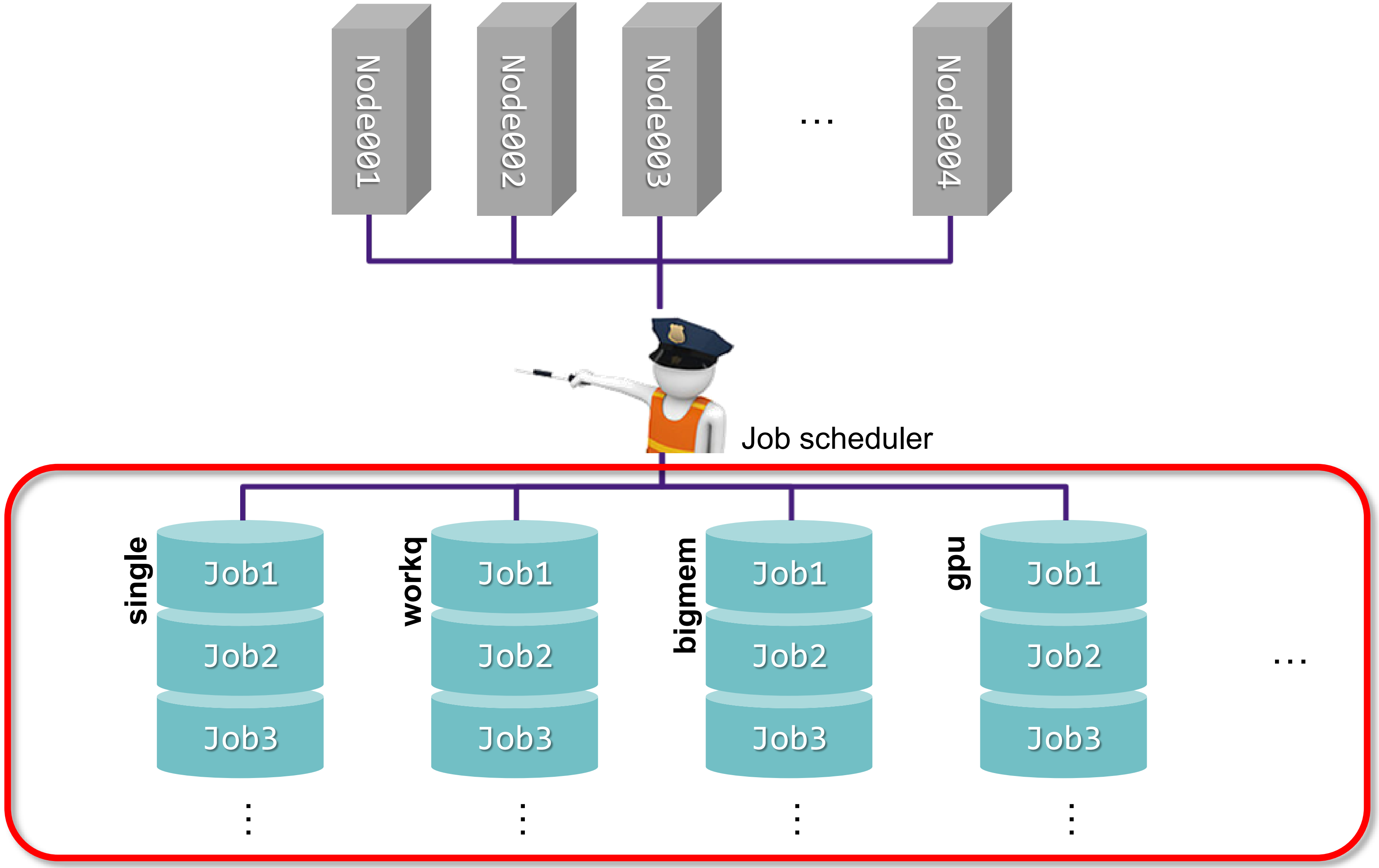

Node001
Node002
Node003
…
Node004
Job scheduler
Job1
Job2
Job3
⋮

2) Job queues



Node001
Node002
Node003
…
Node004
Job scheduler
Job1
Job2
Job3
⋮

2) Job queues

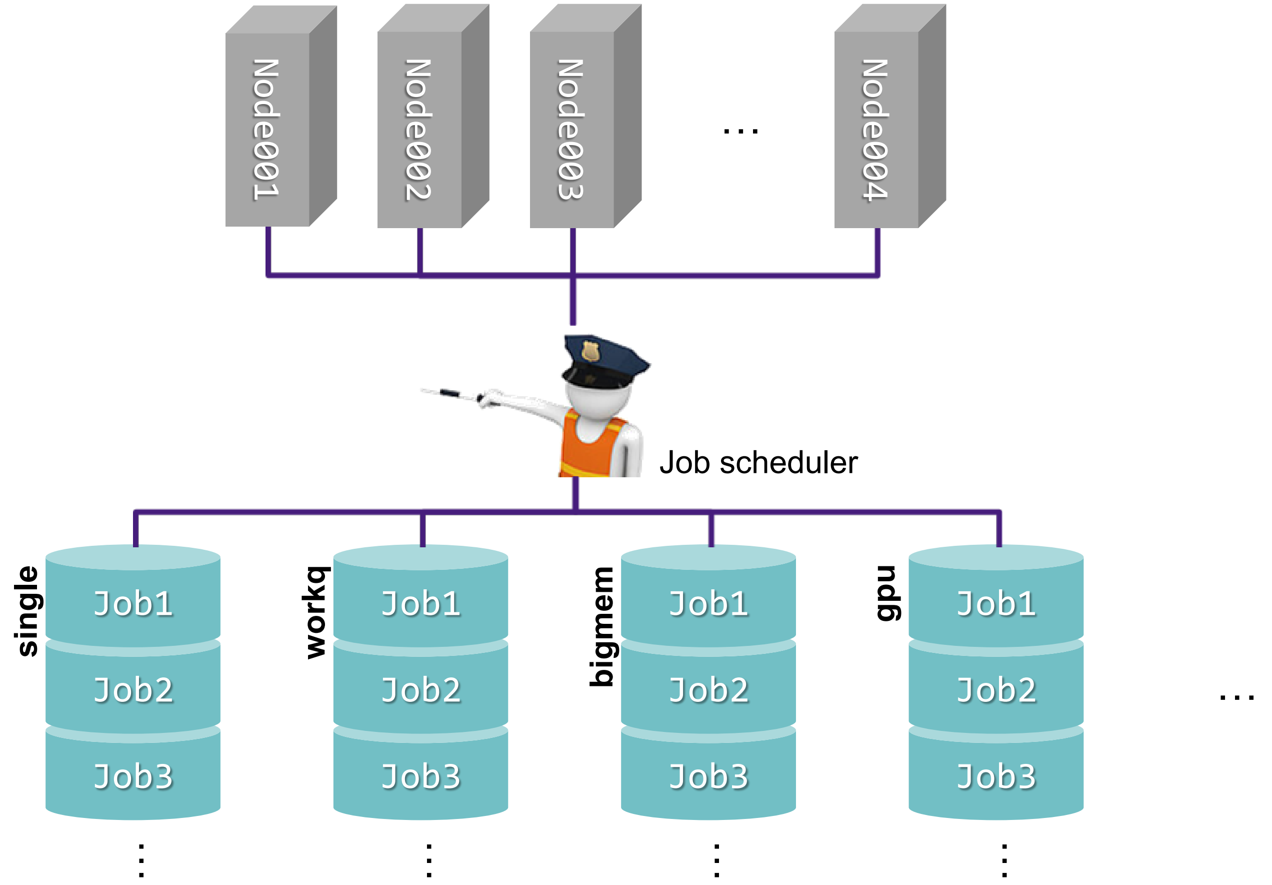

Node001
Node002
Node003
…
Node004
Job scheduler
single
workq
bigmem
gpu
Job1
Job2
Job3
⋮
…

2) Job queues

a) Definition

- Different queues / lines where jobs are being organized into

- Must pick one queue to submit job



2) Job queues

2) Job queues

b) Available queues

Queue	Feature	Allowed number of cores (ppn)	Available RAM	Max duration

2) Job queues

b) Available queues

Queue	Feature	Allowed number of cores (ppn)	Available RAM	Max duration
workq **checkpt**	-	N	All	3 days

2) Job queues

b) Available queues

Queue	Feature	Allowed number of cores (ppn)	Available RAM	Max duration
workq **checkpt**	-	N	All	3 days
single	-	[PBS] 1/2/4/6/8 [Slurm] N-1	(RAM/core) * ppn	7 days

[SuperMike 3]

- Each node: 256 GB RAM, 64 cores
  → 4 GB RAM / core
- Request ppn=10
  → 40 GB RAM

2) Job queues

b) Available queues

Queue	Feature	Allowed number of cores (ppn)	Available RAM	Max duration
workq **checkpt**	-	N	All	3 days
single	-	[PBS] 1/2/4/6/8 [Slurm] N-1	(RAM/core) * ppn	7 days
gpu **v100** **nvlink**	GPU	N	All	3 days

2) Job queues

b) Available queues

Queue	Feature	Allowed number of cores (ppn)	Available RAM	Max duration
workq **checkpt**	-	N	All	3 days
single	-	[PBS] 1/2/4/6/8 [Slurm] N-1	(RAM/core) * ppn	7 days
gpu **v100** **nvlink**	GPU	N	All	3 days
bigmem	Large RAM	N	All	3 days

2) Job queues

c) Queues by clusters (LSU HPC)

2) Job queues

c) Queues by clusters (LSU HPC)

Cluster	Queue	ppn	Max running jobs	Max nodes per job

2) Job queues

c) Queues by clusters (LSU HPC)

Cluster	Queue	ppn	Max running jobs	Max nodes per job
SuperMIC	workq	20	34	128
	checkpt			200
	single	1,2,4,6,8		1
	v100	36		2
	bigmem	28		3

2) Job queues

c) Queues by clusters (LSU HPC)

Cluster	Queue	ppn	Max running jobs	Max nodes per job
SuperMIC	workq	20	34	128
	checkpt			200
	single	1,2,4,6,8		1
	v100	36		2
	bigmem	28		3
DeepBayou	checkpt	48	4	4
	single	1 to 47		1
	nvlink	48		1

2) Job queues

c) Queues by clusters (LSU HPC)

Cluster	Queue	ppn	Max running jobs	Max nodes per job
SuperMIC	workq	20	34	128
	checkpt			200
	single	1,2,4,6,8		1
	v100	36		2
	bigmem	28		3
DeepBayou	checkpt	48	4	4
	single	1 to 47		1
	nvlink	48		1
SuperMike3	workq	64	32	84
	checkpt			
	single	1 to 63		1
	gpu	64		4
	bigmem	64		4

2) Job queues

c) Queues by clusters (LONI)

Cluster	Queue	ppn	Max running jobs	Max nodes per job

2) Job queues

c) Queues by clusters (LONI)

Cluster	Queue	ppn	Max running jobs	Max nodes per job
QB-2	workq	20	64	128
	checkpt			
	single	1,2,4,6,8		1
	bigmem	48		1

2) Job queues

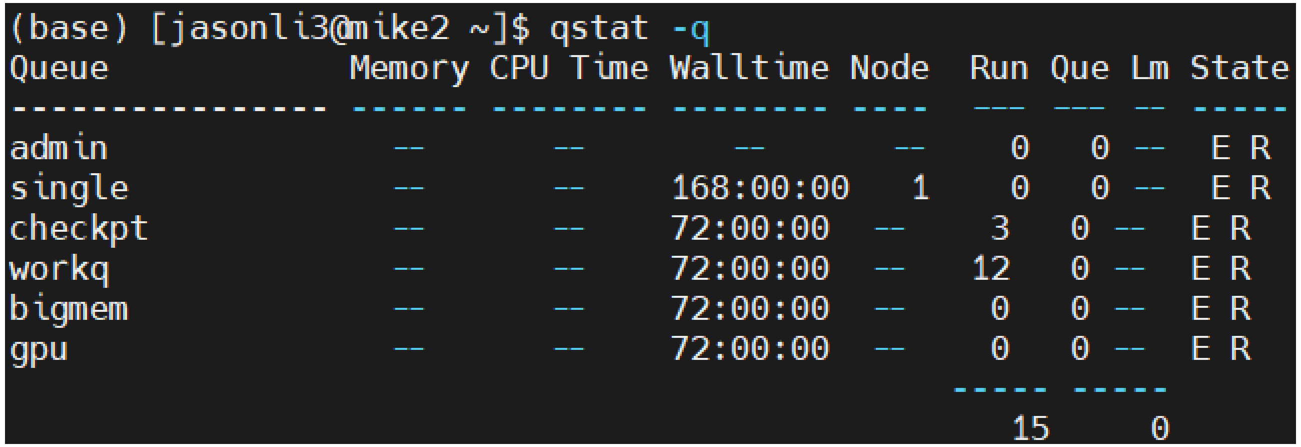
c) Queues by clusters (LONI)

Cluster	Queue	ppn	Max running jobs	Max nodes per job
QB-2	workq	20	64	128
	checkpt			
	single	1,2,4,6,8		1
	bigmem	48		1
QB-3	workq	48	32	96
	checkpt			
	single	1-47		1
	gpu	48		4
	bigmem	48		1

2) Job queues

d) Useful commands to check queues

i. **qstat –q** : All queue information

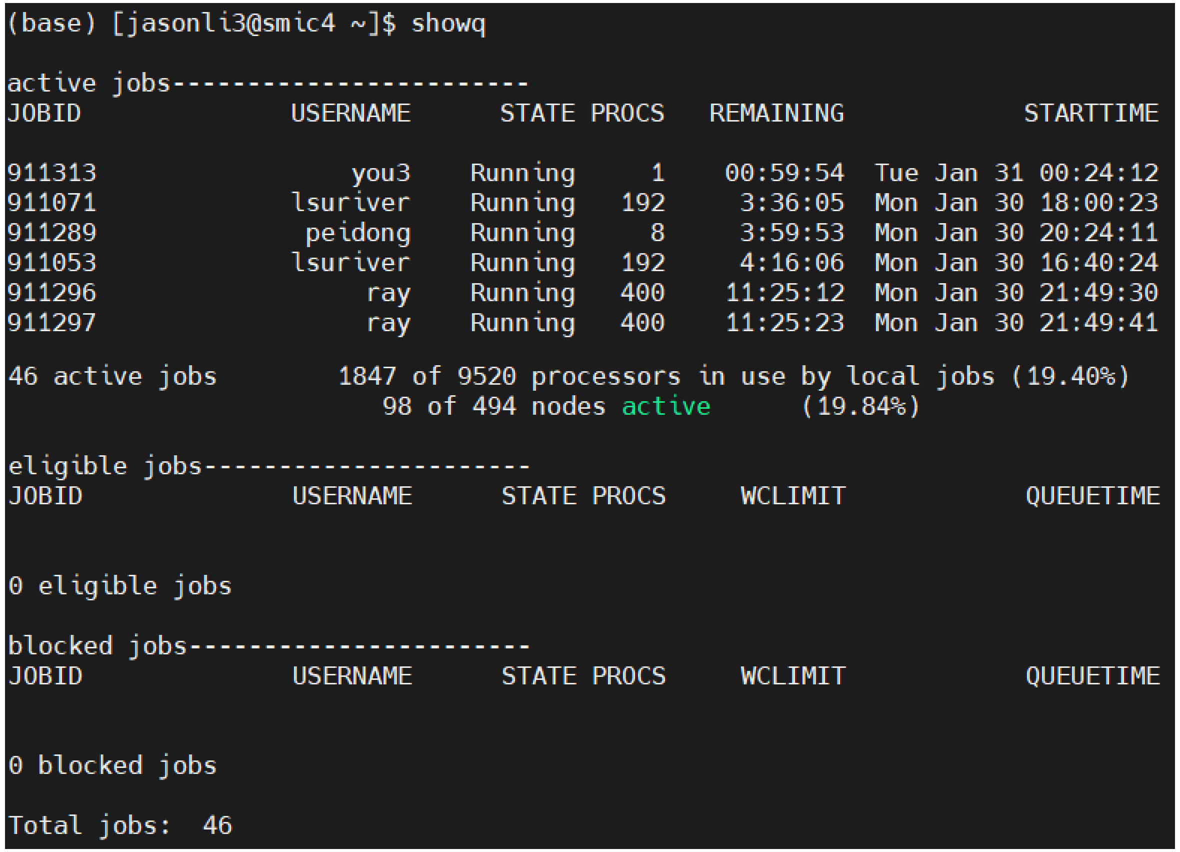


```
(base) [jasonli3@mike2 ~]$ qstat -q
Queue            Memory CPU Time Walltime Node  Run Que Lm State
---------------- ------ -------- -------- ----  --- --- -- -----
admin              --      --       --       --    0   0 -- E R
single             --      --    168:00:00    1    0   0 -- E R
checkpt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

2) Job queues

d) Useful commands to check queues

ii. **showq** : All active, eligible, blocked, and/or recently completed jobs

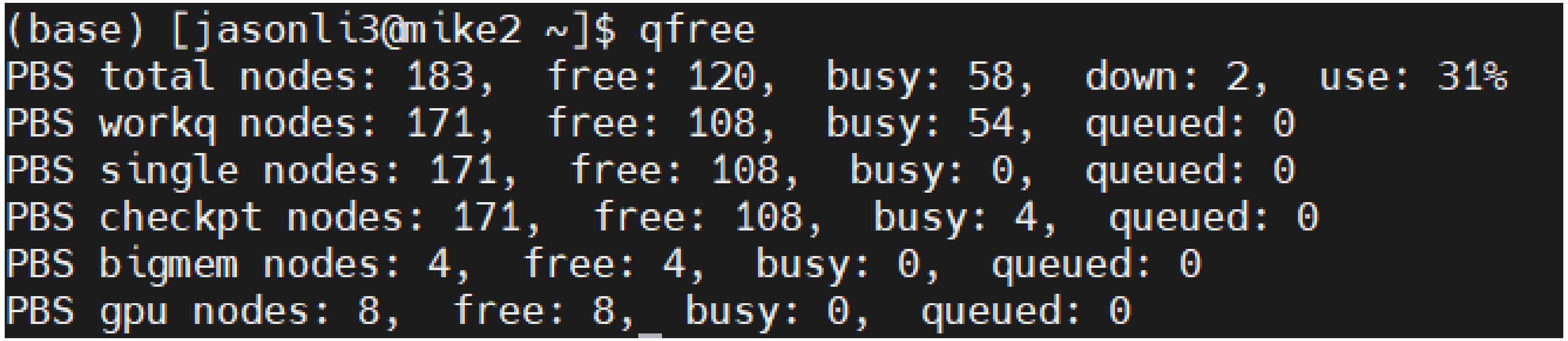


```
(base) [jasonli3@smic4 ~]$ showq

active jobs------------------------
JOBID              USERNAME      STATE PROCS   REMAINING            STARTTIME

911313                 you3    Running     1    00:59:54  Tue Jan 31 00:24:12
911071             lsuriver    Running   192     3:36:05  Mon Jan 30 18:00:23
911289              peidong    Running     8     3:59:53  Mon Jan 30 20:24:11
911053             lsuriver    Running   192     4:16:06  Mon Jan 30 16:40:24
911296                  ray    Running   400    11:25:12  Mon Jan 30 21:49:30
911297                  ray    Running   400    11:25:23  Mon Jan 30 21:49:41

46 active jobs         1847 of 9520 processors in use by local jobs (19.40%)
                          98 of 494 nodes active      (19.84%)

eligible jobs----------------------
JOBID              USERNAME      STATE PROCS     WCLIMIT            QUEUETIME


0 eligible jobs

blocked jobs-----------------------
JOBID              USERNAME      STATE PROCS     WCLIMIT            QUEUETIME


0 blocked jobs

Total jobs:  46
```

2) Job queues

d) Useful commands to check queues

iii. **qfree** : Free nodes in each queue

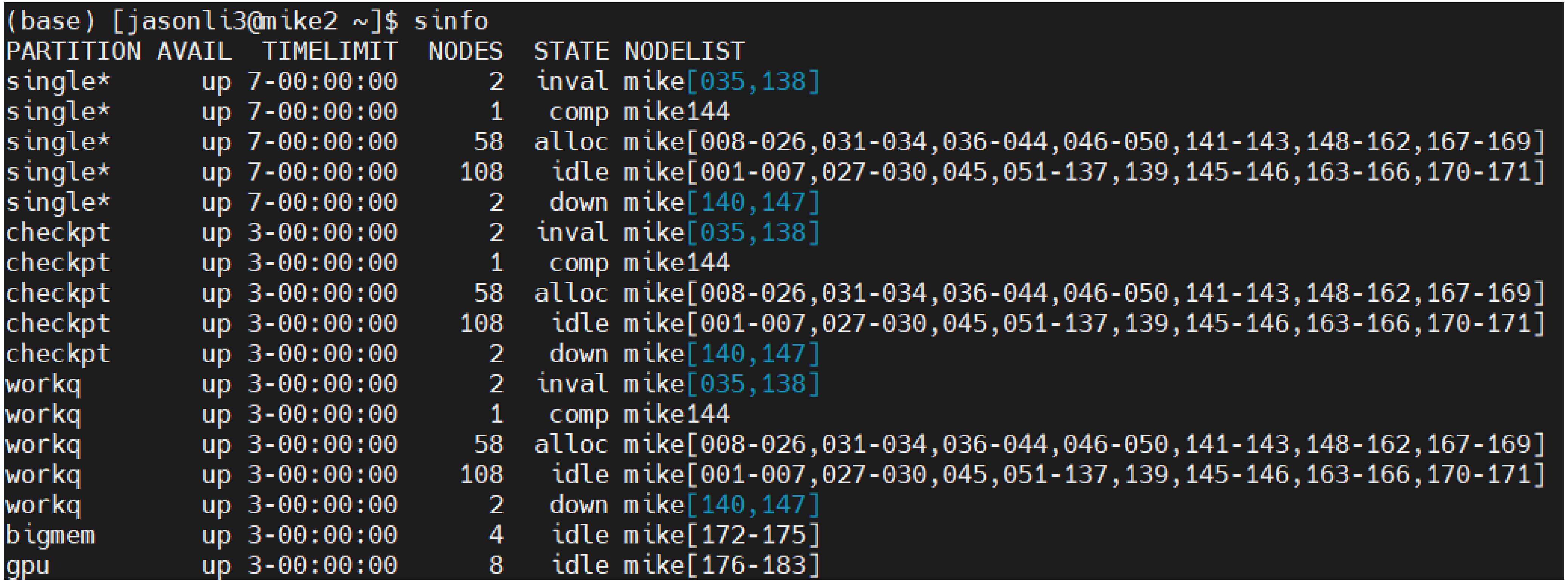


```
(base) [jasonli3@mike2 ~]$ qfree
PBS total nodes: 183,  free: 120,  busy: 58,  down: 2,  use: 31%
PBS workq nodes: 171,  free: 108,  busy: 54,  queued: 0
PBS single nodes: 171,  free: 108,  busy: 0,  queued: 0
PBS checkpt nodes: 171,  free: 108,  busy: 4,  queued: 0
PBS bigmem nodes: 4,  free: 4,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

2) Job queues

d) Useful commands to check queues

iv. **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]
```

3) Choose your queue

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
    1) Job schedulers
    2) Job queues
    3) Choose your queue
3. Submitting a job
    1) Interactive job
    2) Batch job
    3) Cheat sheets
4. Manage jobs
    1) Useful commands
    2) Monitoring job health

3) Choose your queue

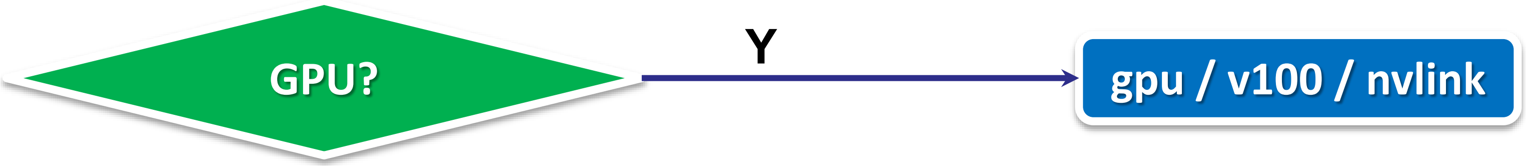
Do I need …

I should choose …

3) Choose your queue

Do I need …

I should choose …

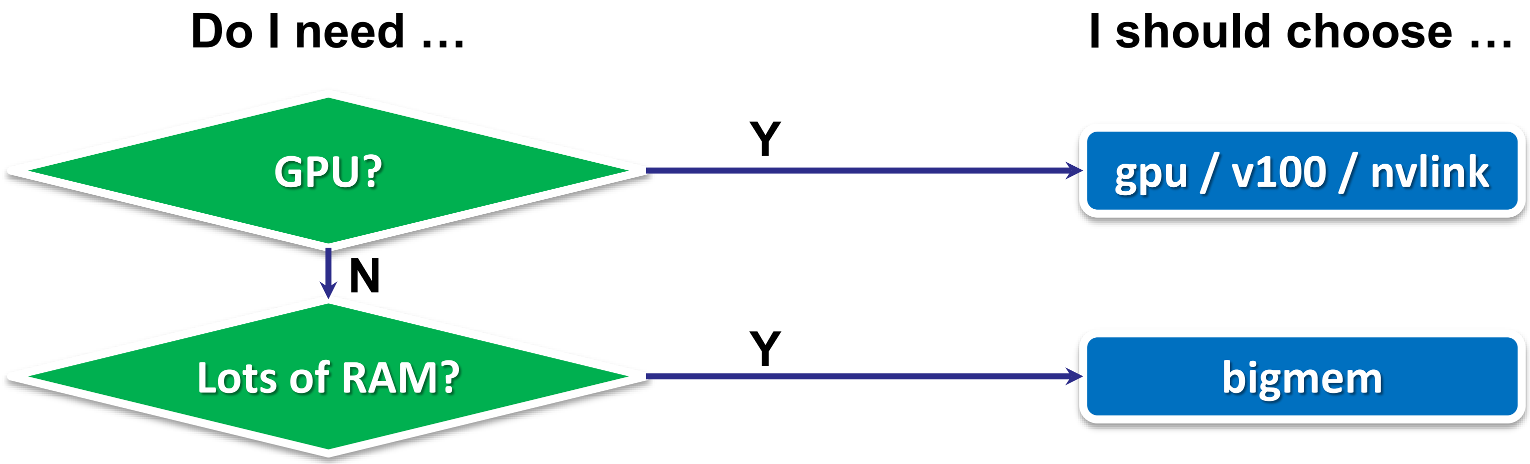


GPU?

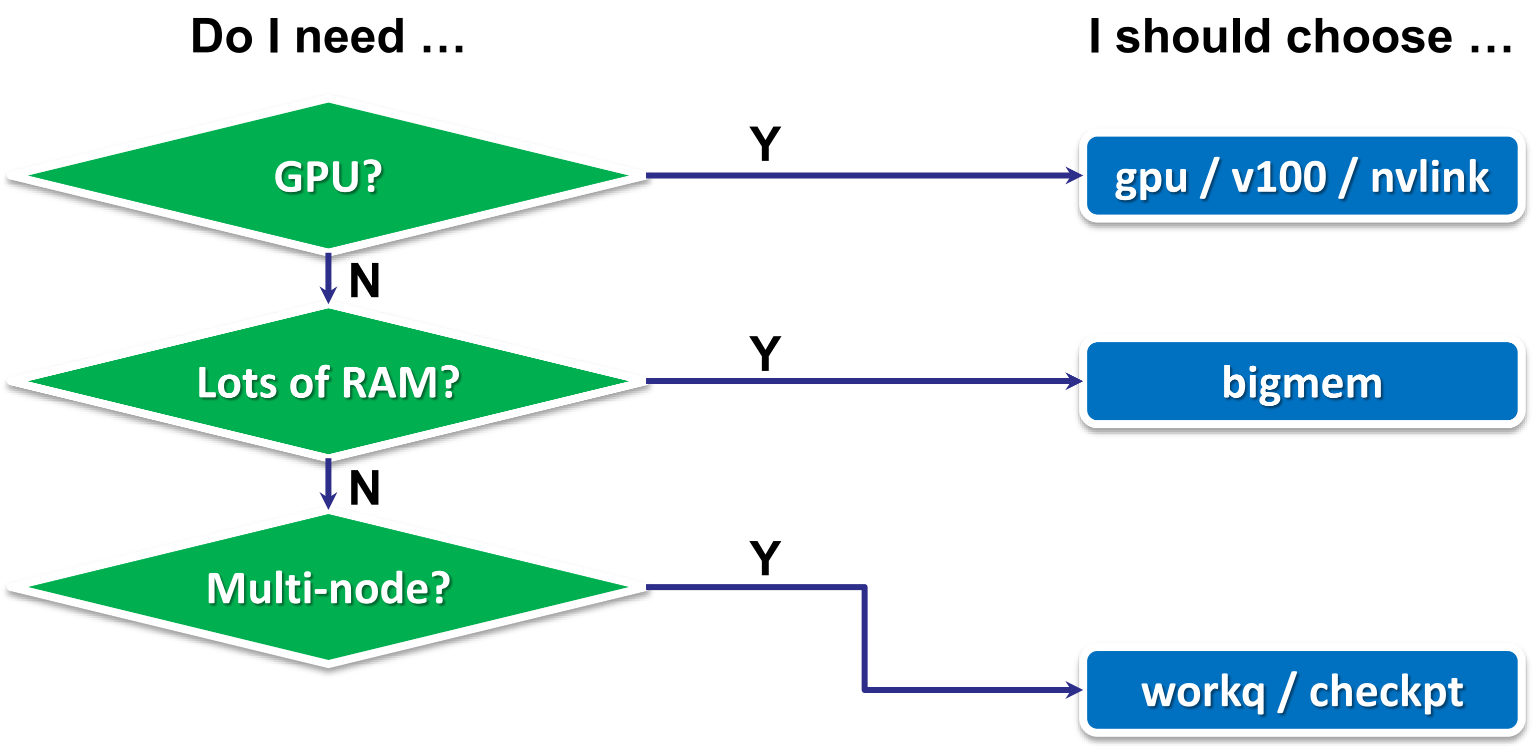
Y

gpu / v100 / nvlink

3) Choose your queue



3) Choose your queue



Do I need …

I should choose …

GPU?

Y

gpu / v100 / nvlink

N

Lots of RAM?

Y

bigmem

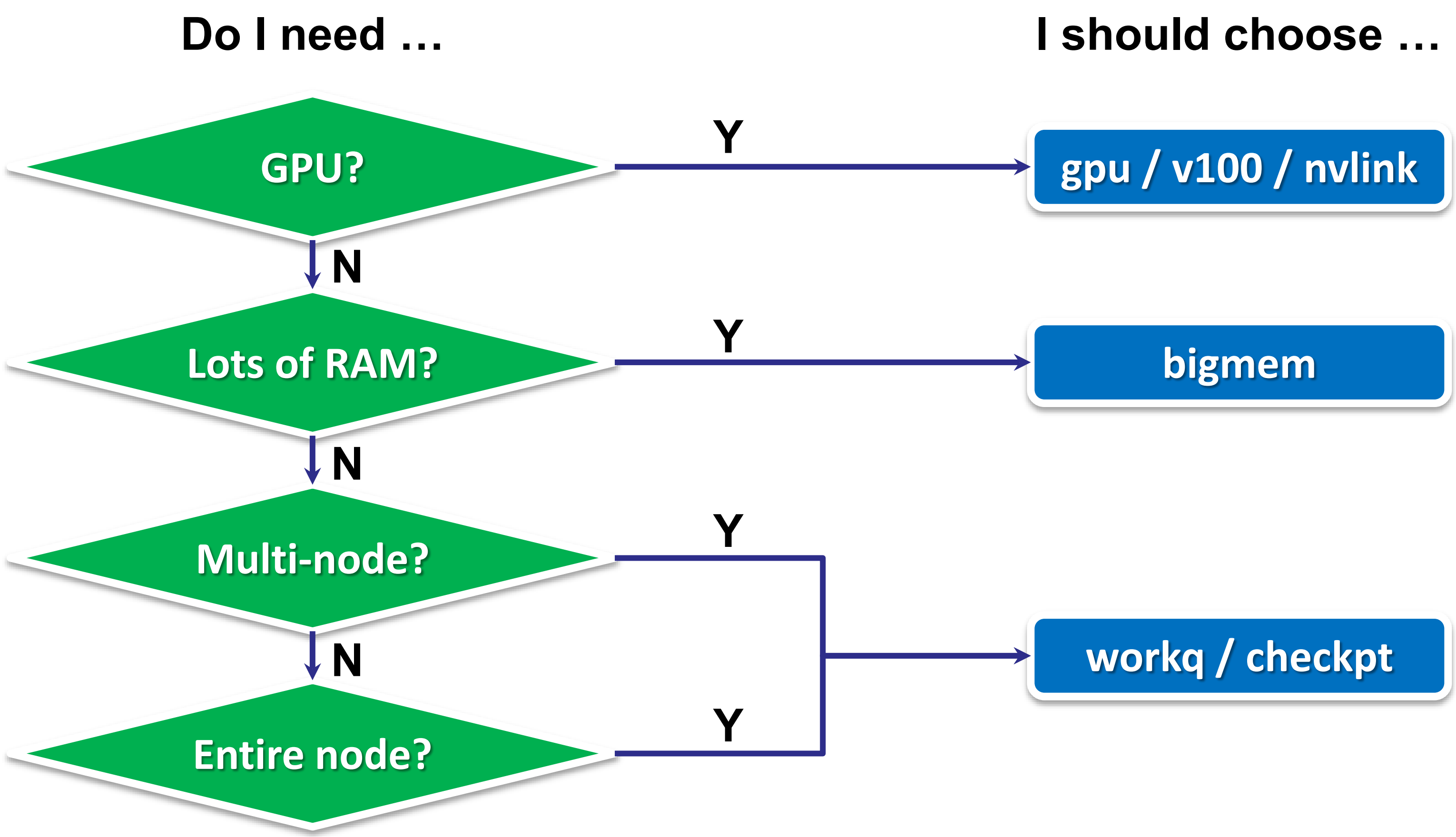
N

Multi-node?

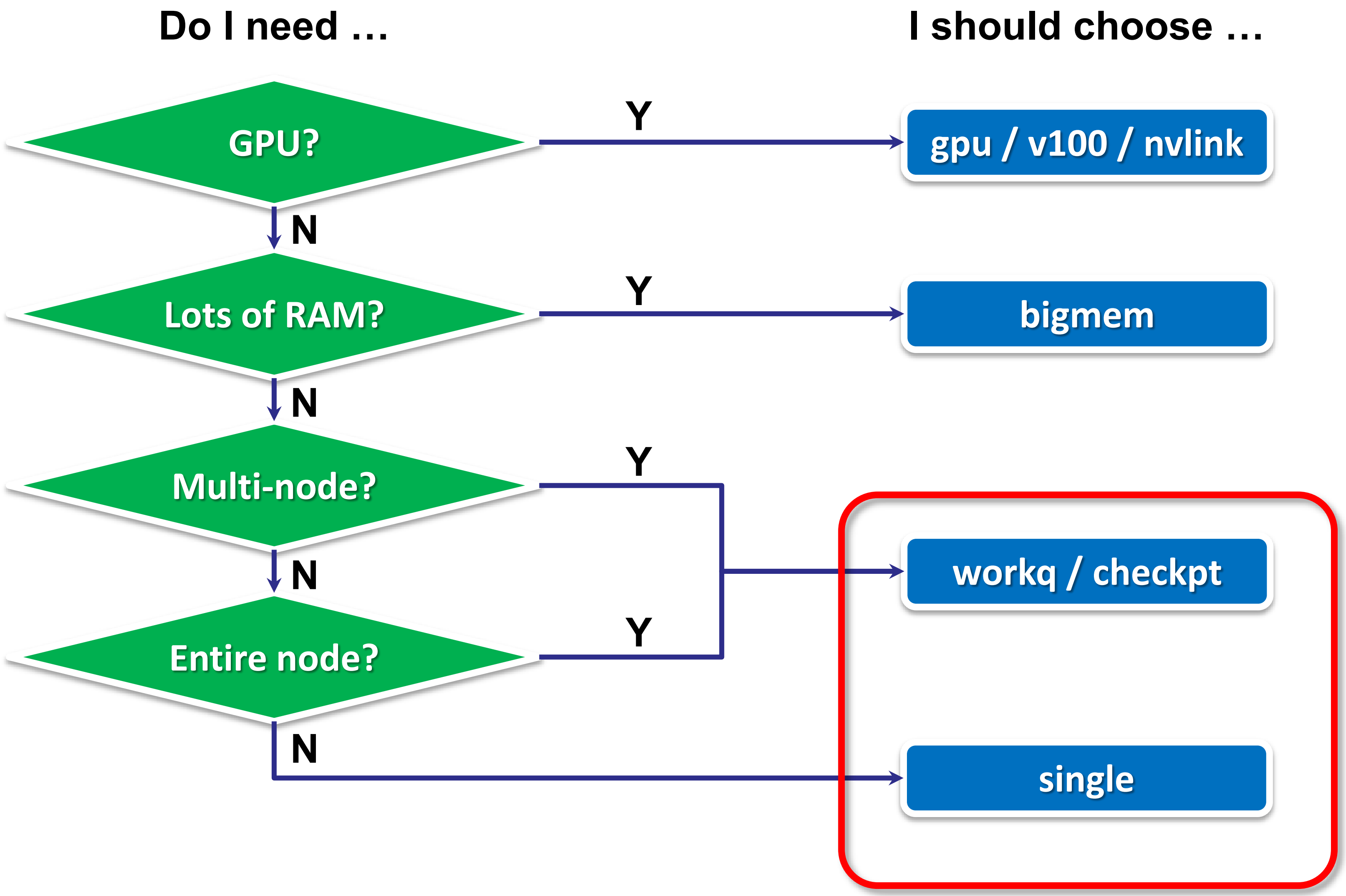
Y

workq / checkpt

3) Choose your queue



3) Choose your queue



3) Choose your queue

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

Large enough …	Small enough …

3) Choose your queue

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

Large enough …	Small enough …
• To successfully complete your job	

3) Choose your queue

- **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

3) Choose your queue

Test

My job …	Queue choice? (include *ppn* if choose single)

3) Choose your queue

Test

My job …	Queue choice? (include *ppn* if choose single)
- Runs on SMIC - MPI code, needs 100 CPU cores, not memory heavy - Hint: SMIC has 20 cores per node	

3) Choose your queue

Test

My job …	Queue choice? (include *ppn* if choose single)
• Runs on SMIC • MPI code, needs 100 CPU cores, not memory heavy - Hint: SMIC has 20 cores per node	**workq** / **checkpt**

3) Choose your queue

Test

My job …	Queue choice? (include *ppn* if choose single)
• Runs on SMIC • MPI code, needs 100 CPU cores, not memory heavy - Hint: SMIC has 20 cores per node	**workq** / **checkpt**
• Runs on QB-3 • Trains neural network with GPU • CPU portion of the code needs one core	

3) Choose your queue

Test

My job …	Queue choice? (include *ppn* if choose single)
• Runs on SMIC • MPI code, needs 100 CPU cores, not memory heavy - Hint: SMIC has 20 cores per node	**workq** / **checkpt**
• Runs on QB-3 • Trains neural network with GPU • CPU portion of the code needs one core	**gpu**

3) Choose your queue

Test

My job …	Queue choice? (include *ppn* if choose single)
• Runs on SMIC • MPI code, needs 100 CPU cores, not memory heavy - Hint: SMIC has 20 cores per node	workq / checkpt
• Runs on QB-3 • Trains neural network with GPU • CPU portion of the code needs one core	gpu
• Runs on SuperMike 3 • Single-core serial code • Needs to store and process 30 GB data in RAM - Hint: SuperMike 3 has 256 GB RAM per node, 4 GB RAM per core	

3) Choose your queue

Test

My job …	Queue choice? (include *ppn* if choose single)
- Runs on SMIC - MPI code, needs 100 CPU cores, not memory heavy - Hint: SMIC has 20 cores per node	**workq** / **checkpt**
- Runs on QB-3 - Trains neural network with GPU - CPU portion of the code needs one core	**gpu**
- Runs on SuperMike 3 - Single-core serial code - Needs to store and process 30 GB data in RAM - Hint: SuperMike 3 has 256 GB RAM per node, 4 GB RAM per core	**single** (ppn = **8**)

Summary

1) **Job scheduler and how it works**
2) **Job queues**
   a) What is job queue
   b) Job queues on our cluster
   c) Useful commands to check job queues
3) **How to choose job queue**
   a) Flowchart
   b) 2 basic principles - “large enough” and “small enough”

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
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage 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

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		
Cons		
Ideal for		

3. Submitting a job

- **Two types of jobs:**

	1) Interactive job	2) Batch job
Pros	• Can interact and monitor with job in real time	
Cons		
Ideal for		

3. Submitting a job

- **Two types of jobs:**

	1) Interactive job	2) Batch job
Pros	• Can interact and monitor with job in real time	
Cons	• Waiting for human intervention is the opposite of "high performance"	
Ideal for		

3. Submitting a job

- **Two types of jobs:**

	1) Interactive job	2) Batch job
Pros	• Can interact and monitor with job in real time	
Cons	• Waiting for human intervention is the opposite of "high performance"	
Ideal for	• Debugging and testing • Large compilation	

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”	
Ideal for	• Debugging and testing • Large compilation	

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	

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

1) Interactive job

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
    1) Job schedulers
    2) Job queues
    3) Choose your queue
3. Submitting a job
    1) Interactive job
    2) Batch job
    3) Cheat sheets
4. Manage 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 -p <Queue name> \ -t <HH:MM:SS> \ -N <# of nodes> \ -n <# of TOTAL cores> \ --pty bash

Walltime, 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 <`
`-q <`
`-l walltime=<HH:MM:SS>,node`
`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`

A little more technical –

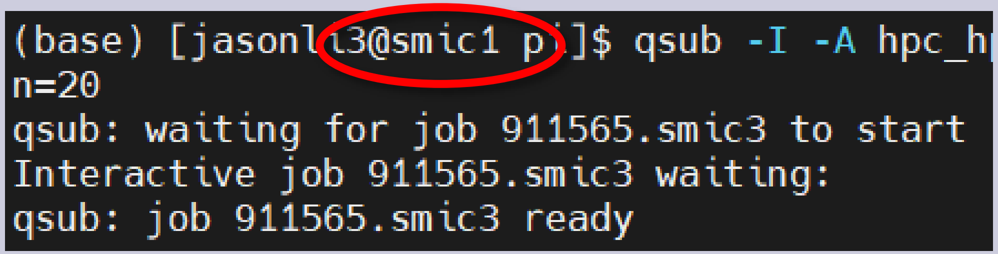
-n <# of tasks>
-c <# of cores per task>

→ <n> * <c> = <# of TOTAL cores>

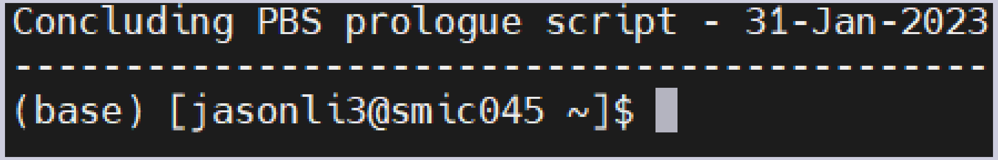
1) Interactive job

b) Starting an interactive job

PBS

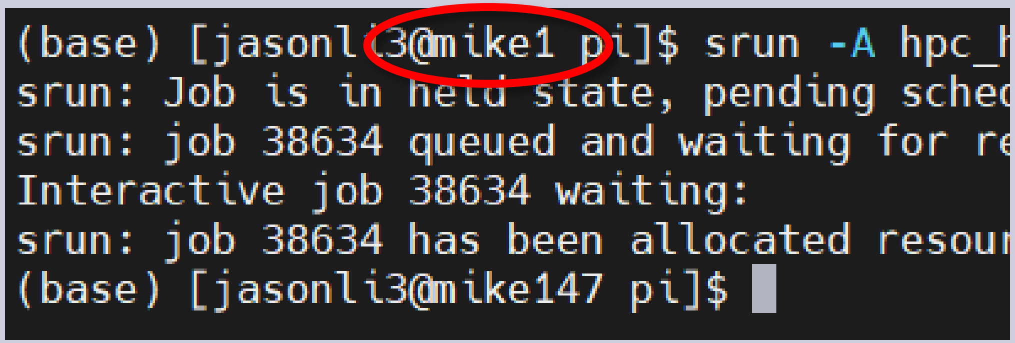


```
(base) [jasonli3@smic1 pi]$ qsub -I -A hpc_h
n=20
qsub: waiting for job 911565.smic3 to start
Interactive job 911565.smic3 waiting:
qsub: job 911565.smic3 ready
```



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

Slurm

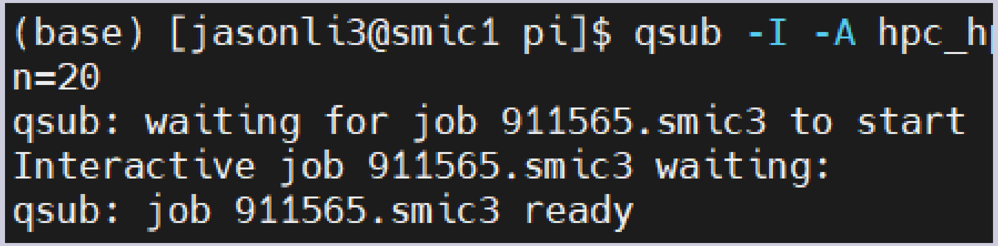


```
(base) [jasonli3@mike1 pi]$ srun -A hpc_h
srun: Job is in held state, pending sched
srun: job 38634 queued and waiting for re
Interactive job 38634 waiting:
srun: job 38634 has been allocated resour
(base) [jasonli3@mike147 pi]$
```

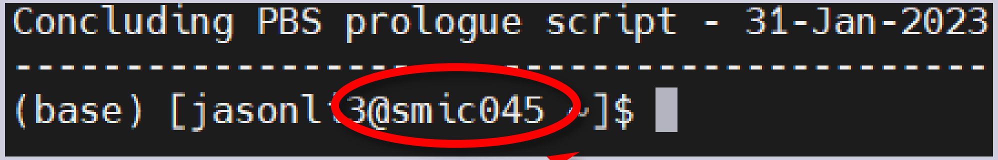
1) Interactive job

b) Starting an interactive job

PBS

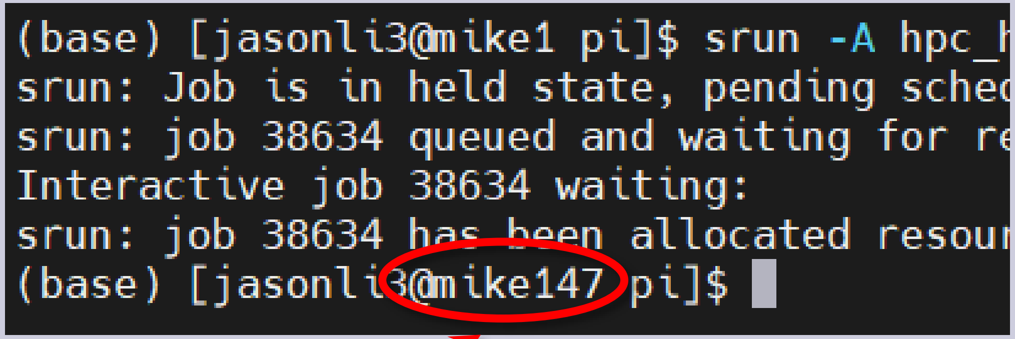


```
(base) [jasonli3@smic1 pi]$ qsub -I -A hpc_h
n=20
qsub: waiting for job 911565.smic3 to start
Interactive job 911565.smic3 waiting:
qsub: job 911565.smic3 ready
```



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

Slurm



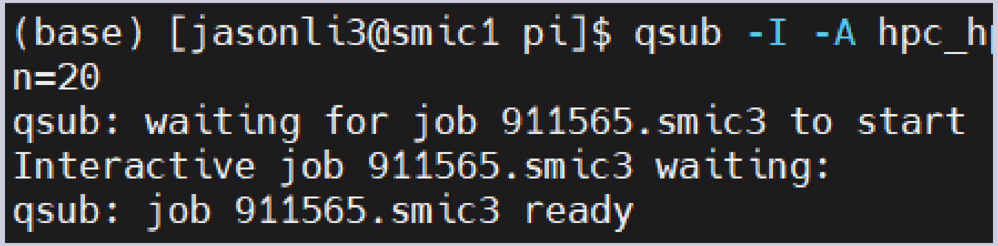
```
(base) [jasonli3@mike1 pi]$ srun -A hpc_h
srun: Job is in held state, pending sched
srun: job 38634 queued and waiting for re
Interactive job 38634 waiting:
srun: job 38634 has been allocated resour
(base) [jasonli3@mike147 pi]$
```

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

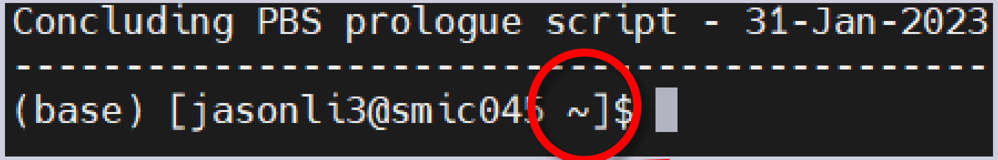
1) Interactive job

b) Starting an interactive job

PBS

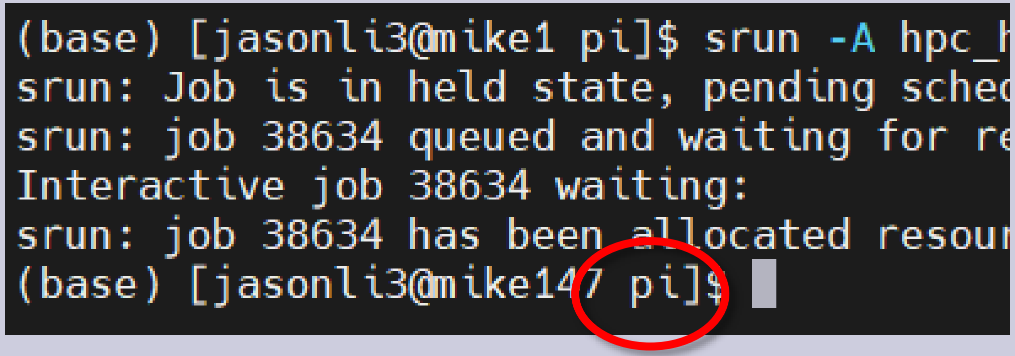


```
(base) [jasonli3@smic1 pi]$ qsub -I -A hpc_h
n=20
qsub: waiting for job 911565.smic3 to start
Interactive job 911565.smic3 waiting:
qsub: job 911565.smic3 ready
```



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

Slurm



```
(base) [jasonli3@mike1 pi]$ srun -A hpc_h
srun: Job is in held state, pending sched
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) 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

2) Batch job

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage 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

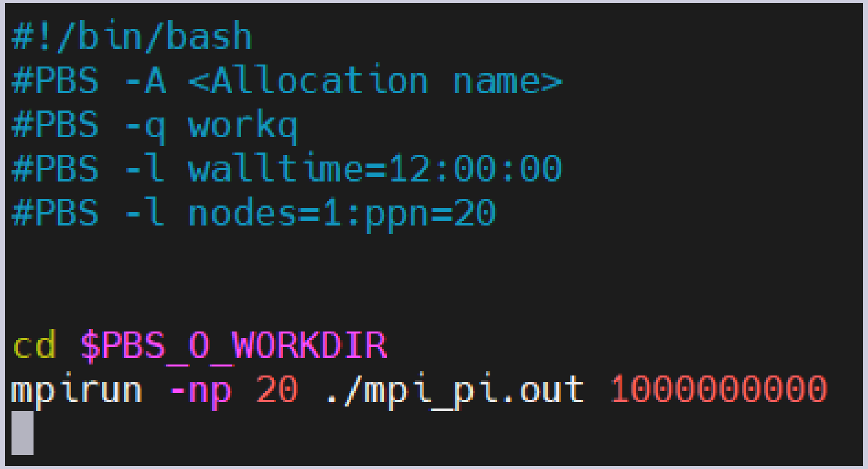
a) Batch file

PBS	Slurm

2) Batch job

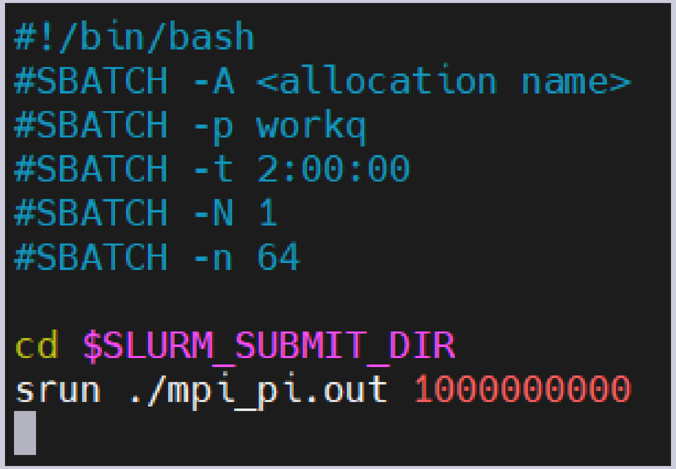
a) Batch file

PBS



```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20


cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi_pi.out 1000000000
```

Slurm

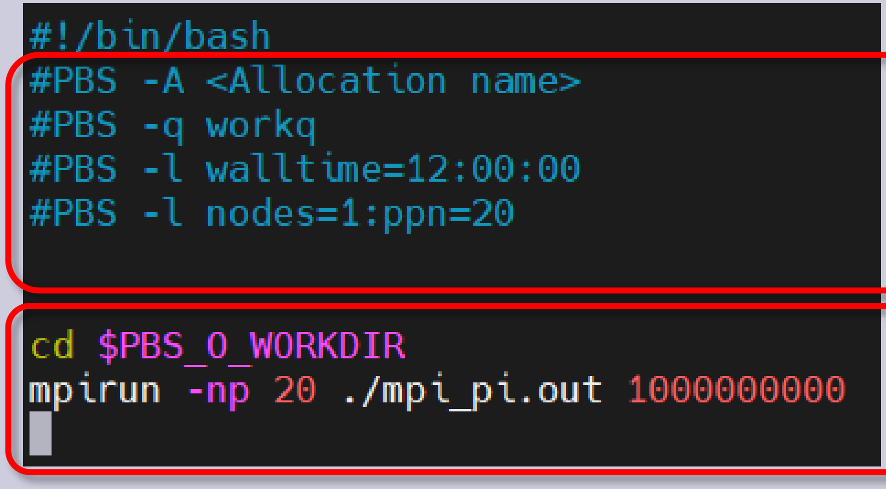
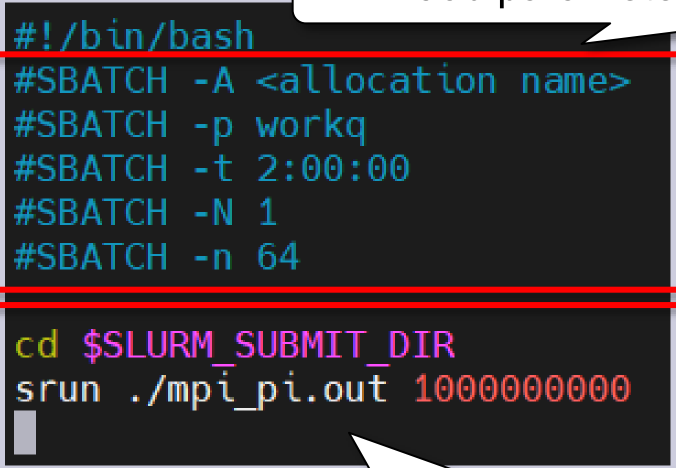


```
#!/bin/bash
#SBATCH -A <allocation name>
#SBATCH -p workq
#SBATCH -t 2:00:00
#SBATCH -N 1
#SBATCH -n 64

cd $SLURM_SUBMIT_DIR
srun ./mpi_pi.out 1000000000
```

2) Batch job

a) Batch file

PBS	Slurm

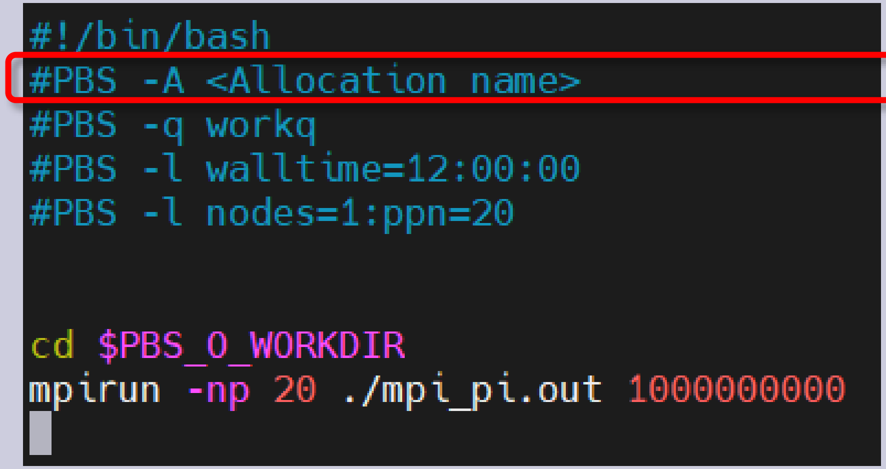
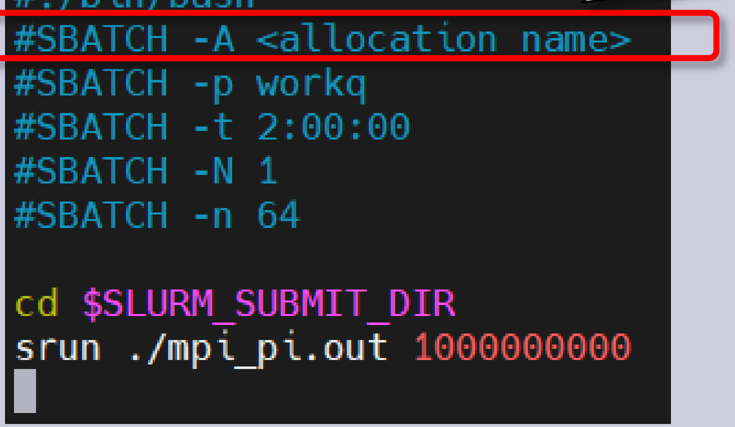

`#!/bin/bash`
`#PBS -A <Allocation name>`
`#PBS -q workq`
`#PBS -l walltime=12:00:00`
`#PBS -l nodes=1:ppn=20`
 	
`cd $PBS_O_WORKDIR`
`mpirun -np 20 ./mpi_pi.out 1000000000`
	
	`#!/bin/bash`
	`#SBATCH -A <allocation name>`
	`#SBATCH -p workq`
	`#SBATCH -t 2:00:00`
	`#SBATCH -N 1`
	`#SBATCH -n 64`
	
	`cd $SLURM_SUBMIT_DIR`
	`srun ./mpi_pi.out 1000000000`

Job parameters

Commands to execute after job starts

2) Batch job

a) Batch file

PBS	Slurm
	

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20


cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi_pi.out 1000000000
```

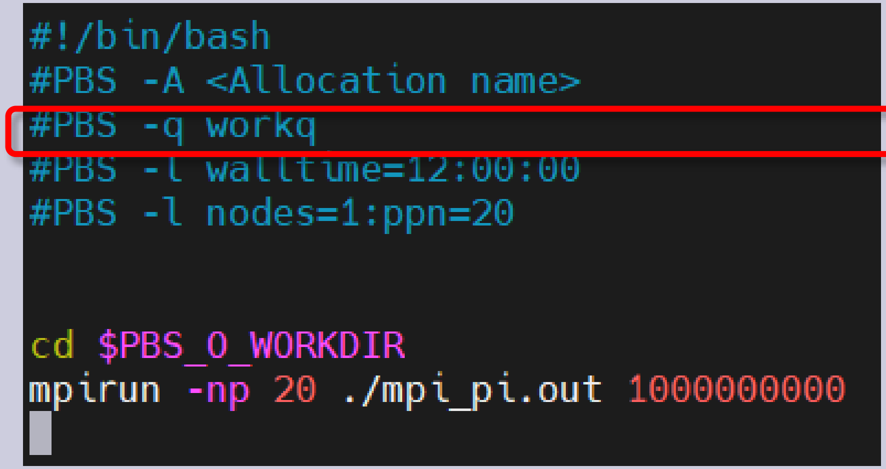
Allocation name

```
#!/bin/bash
#SBATCH -A <allocation name>
#SBATCH -p workq
#SBATCH -t 2:00:00
#SBATCH -N 1
#SBATCH -n 64

cd $SLURM_SUBMIT_DIR
srun ./mpi_pi.out 1000000000
```

2) Batch job

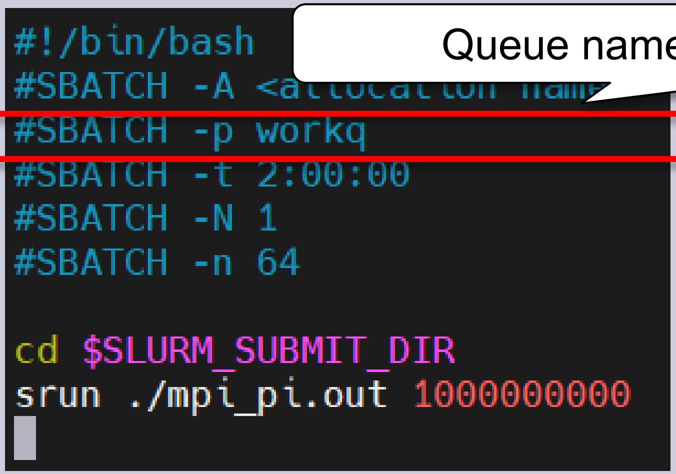
a) Batch file

PBS



```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20


cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi_pi.out 1000000000
```

Slurm



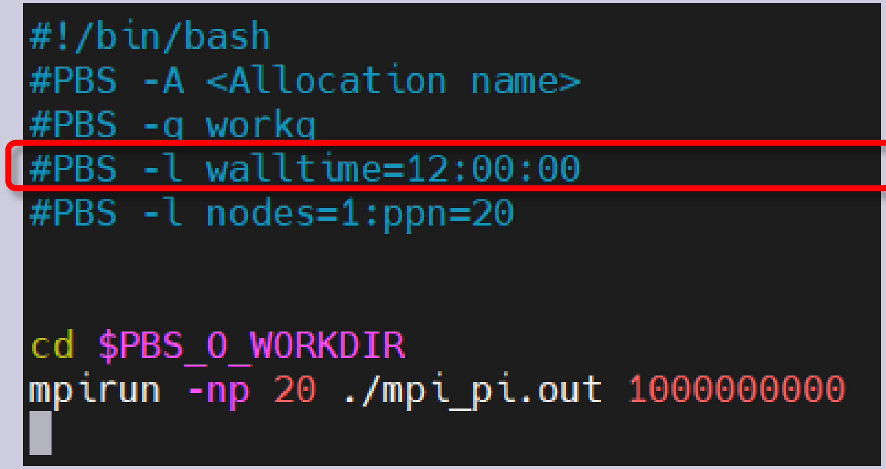
```
#!/bin/bash
#SBATCH -A <allocation name>
#SBATCH -p workq
#SBATCH -t 2:00:00
#SBATCH -N 1
#SBATCH -n 64

cd $SLURM_SUBMIT_DIR
srun ./mpi_pi.out 1000000000
```

Queue name

2) Batch job

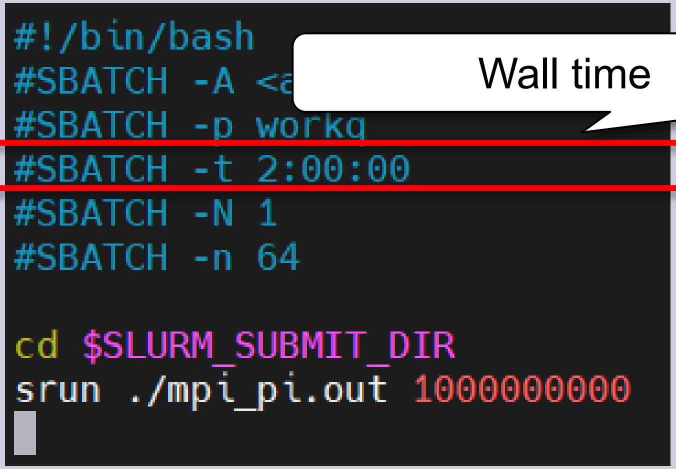
a) Batch file

PBS



```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20


cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi_pi.out 1000000000
```

Slurm



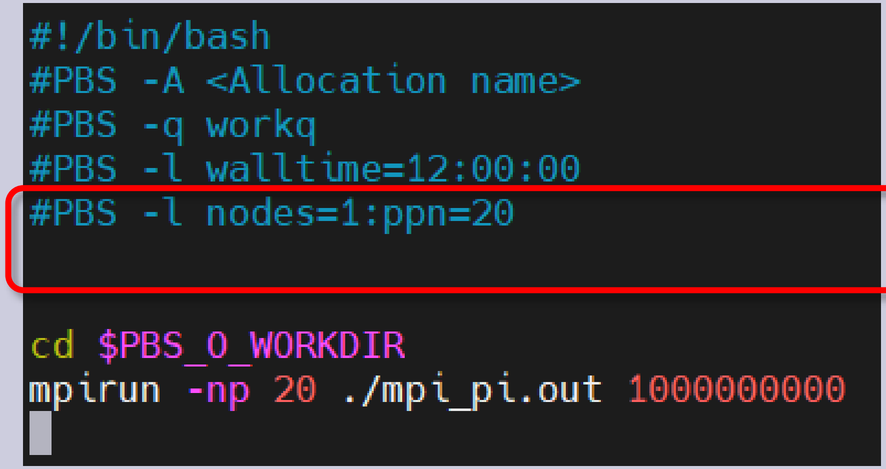
```
#!/bin/bash
#SBATCH -A <a
#SBATCH -p workq
#SBATCH -t 2:00:00
#SBATCH -N 1
#SBATCH -n 64

cd $SLURM_SUBMIT_DIR
srun ./mpi_pi.out 1000000000
```

Wall time

2) Batch job

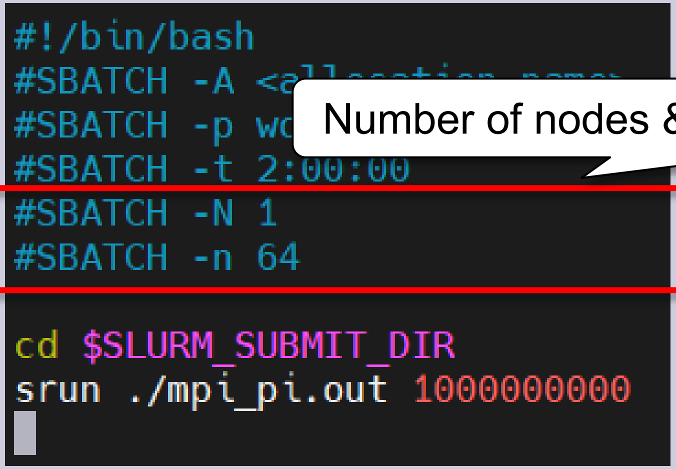
a) Batch file

PBS



```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20

cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi_pi.out 1000000000
```

Slurm

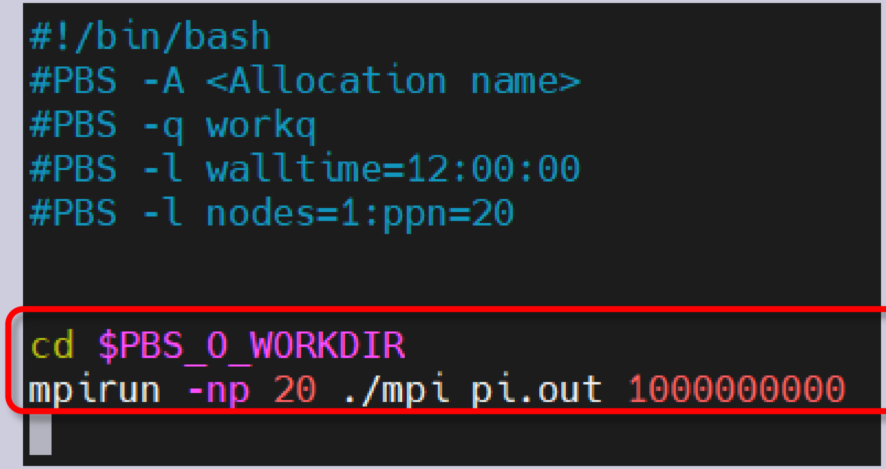
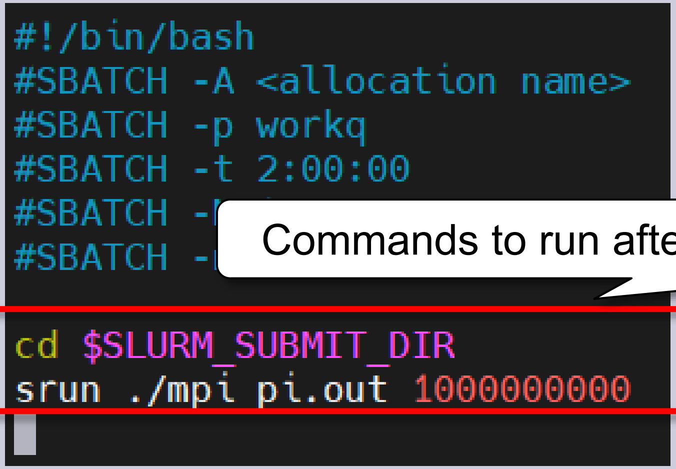


```
#!/bin/bash
#SBATCH -A <a
#SBATCH -p wo
#SBATCH -t 2:00:00
#SBATCH -N 1
#SBATCH -n 64

cd $SLURM_SUBMIT_DIR
srun ./mpi_pi.out 1000000000
```

Number of nodes & cores

2) Batch job

a) Batch file

PBS	Slurm
	

PBS:

```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20

cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi pi.out 1000000000
```

Slurm:

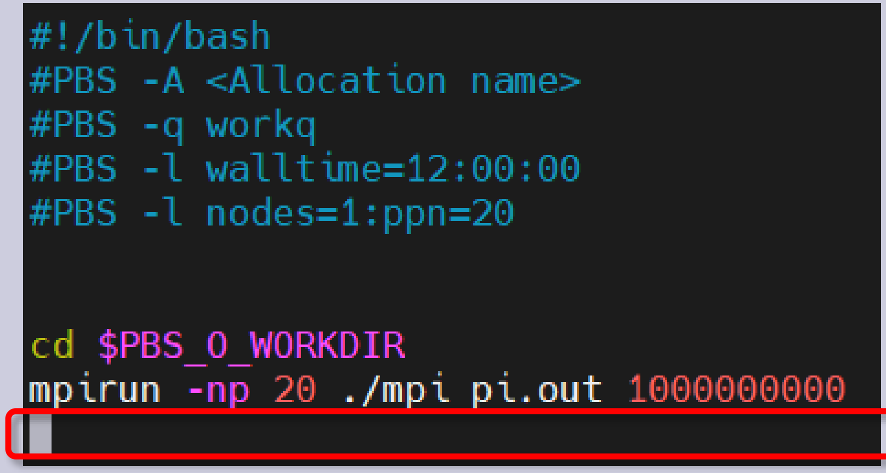
```
#!/bin/bash
#SBATCH -A <allocation name>
#SBATCH -p workq
#SBATCH -t 2:00:00
#SBATCH -
#SBATCH -

cd $SLURM_SUBMIT_DIR
srun ./mpi pi.out 1000000000
```

Commands to run after job starts

2) Batch job

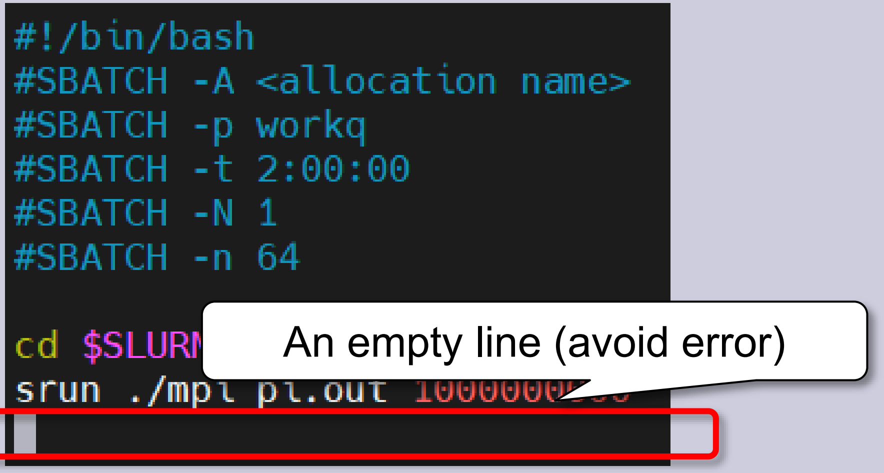
a) Batch file

PBS



```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20


cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi pi.out 1000000000

```

Slurm



```
#!/bin/bash
#SBATCH -A <allocation name>
#SBATCH -p workq
#SBATCH -t 2:00:00
#SBATCH -N 1
#SBATCH -n 64

cd $SLURM
srun ./mpi pi.out 10000000

```

An empty line (avoid error)

2) Batch job

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

[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[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	

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

2) Batch job

b) Command

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

3) Cheat sheets

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage jobs
   1) Useful commands
   2) Monitoring job health

3) Cheat sheets

a) Useful PBS / Slurm options

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	

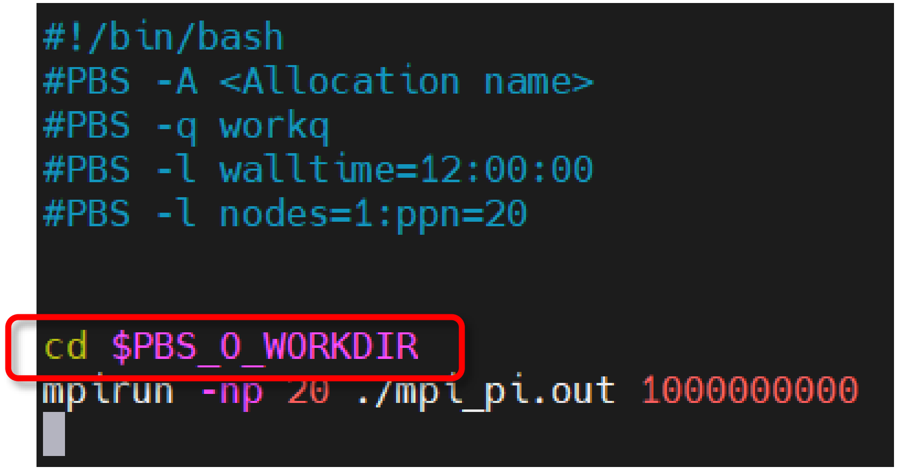
[1] http://www.hpc.lsu.edu/docs/pbs.php

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

3) Cheat sheets

b) 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		er of allocated cores (tasks)
…		



```
#!/bin/bash
#PBS -A <Allocation name>
#PBS -q workq
#PBS -l walltime=12:00:00
#PBS -l nodes=1:ppn=20


cd $PBS_O_WORKDIR
mpirun -np 20 ./mpi_pi.out 1000000000
```

[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
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage 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

1) Useful commands

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage jobs
   1) Useful commands
   2) Monitoring job health

1) Useful commands

PBS[1]		Slurm[2]		Description
`qstat`		`squeue`		List all jobs
	`-n`			List job details
	`-u <Username>`		`-u <Username>`	List all jobs belong to <Username>
`qdel <Job ID>`		`scancel <Job ID>`		Cancel <Job ID>
`checkjob <Job ID>`		`scontrol show job <Job ID>`		Show job details (running or recently finished)

[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
	`-n`			List job details
	`-u <Username>`		`-u <Username>`	List all jobs belong to <Username>
`qdel <Job ID>`		`scancel <Job ID>`		Cancel <Job ID>
`checkjob <Job ID>`		`scontrol show job <Job ID>`		Show job details (running or recently finished)

Alter jobs after submission? → NOT allowed!

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

2) Monitoring job health

- **HPC User Environment 2**

1. Basic concepts
2. How jobs are handled
   1) Job schedulers
   2) Job queues
   3) Choose your queue
3. Submitting a job
   1) Interactive job
   2) Batch job
   3) Cheat sheets
4. Manage 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**
  - Number of processes on each node
  - CPU load
  - RAM usage

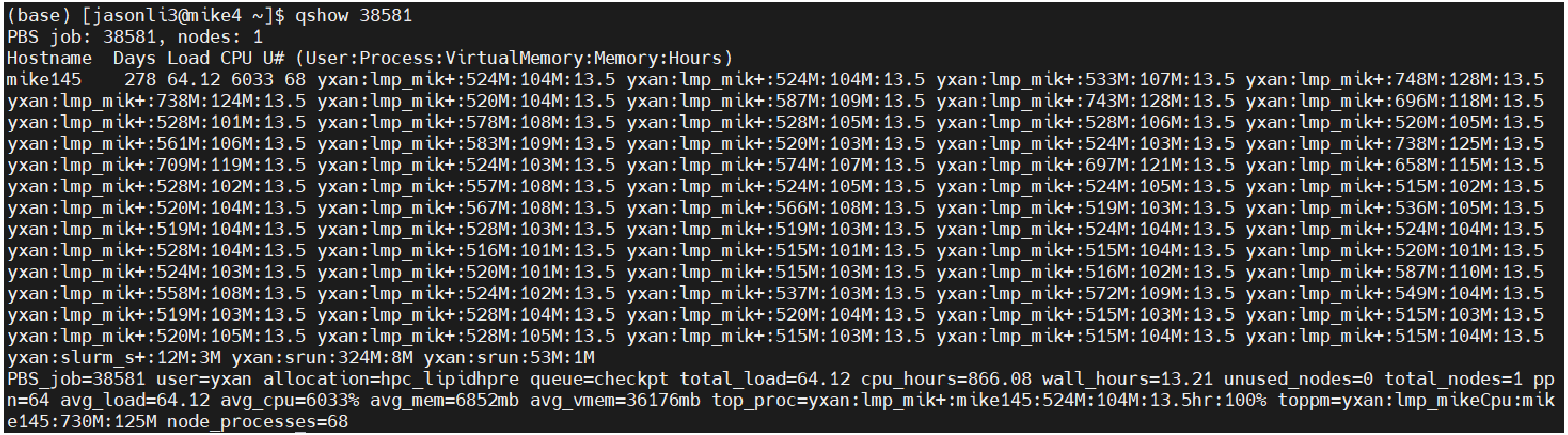
2) Monitoring job health

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>

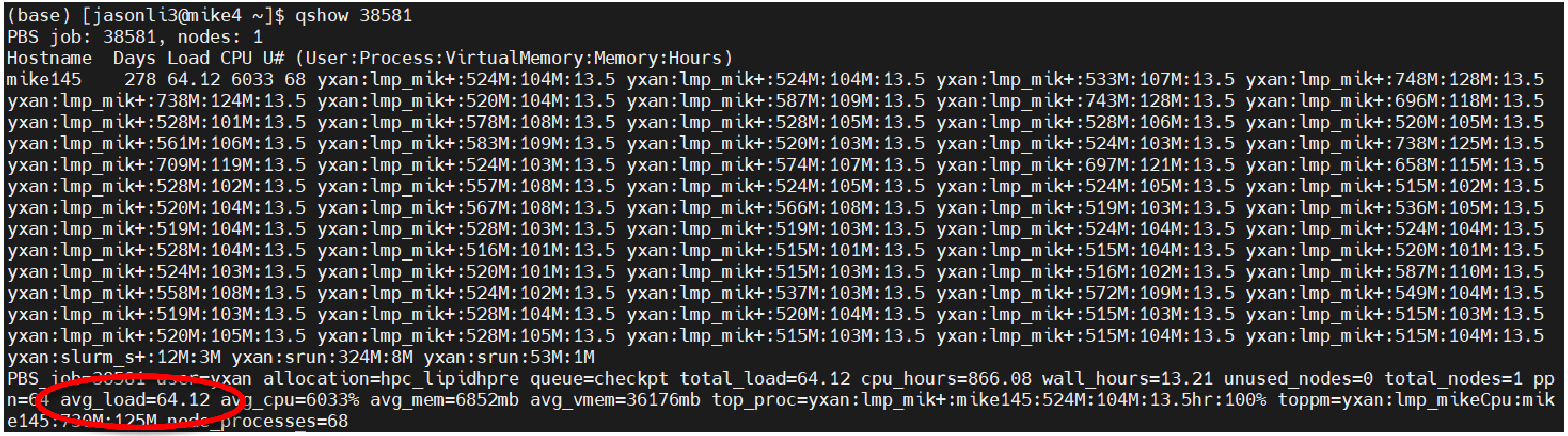


```
(base) [jasonli3@mike4 ~]$ qshow 38581
PBS job: 38581, nodes: 1
Hostname  Days Load CPU U# (User:Process:VirtualMemory:Memory:Hours)
mike145    278 64.12 6033 68 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:533M:107M:13.5 yxan:lmp_mik+:748M:128M:13.5
yxan:lmp_mik+:738M:124M:13.5 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>

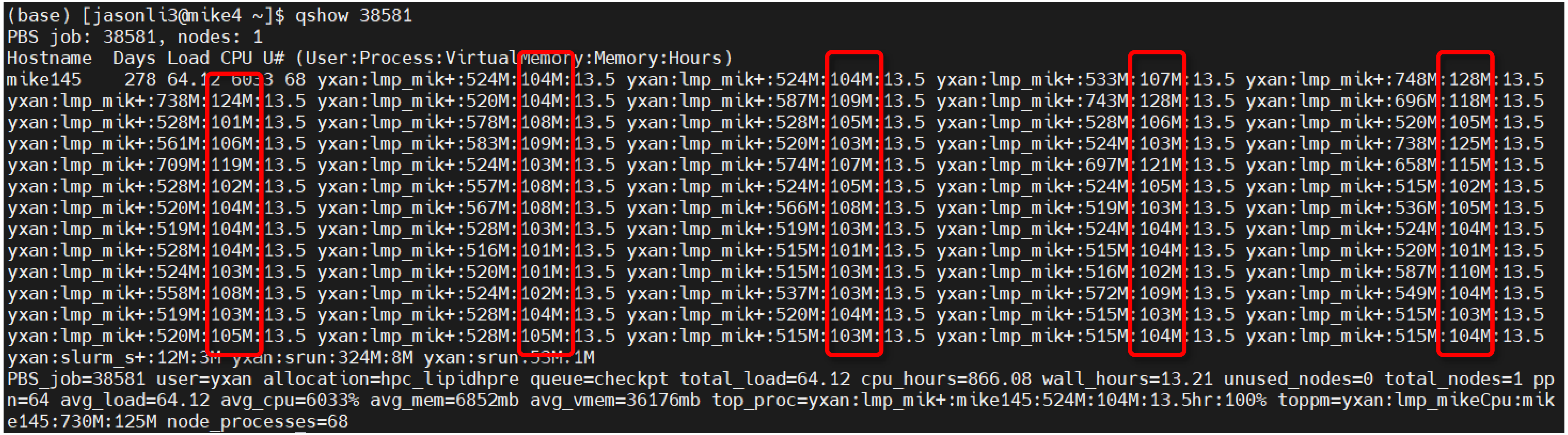


```
(base) [jasonli3@mike4 ~]$ qshow 38581
PBS job: 38581, nodes: 1
Hostname  Days Load CPU U# (User:Process:VirtualMemory:Memory:Hours)
mike145    278 64.12 6033 68 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:524M:104M:13.5 yxan:lmp_mik+:533M:107M:13.5 yxan:lmp_mik+:748M:128M:13.5
yxan:lmp_mik+:738M:124M:13.5 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 requested ppn	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_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 requested ppn	Consistently too low or too high
Memory usage (not virtual memory)	Do not exceed the per core value	Exceeds the per core value

2) Monitoring job health

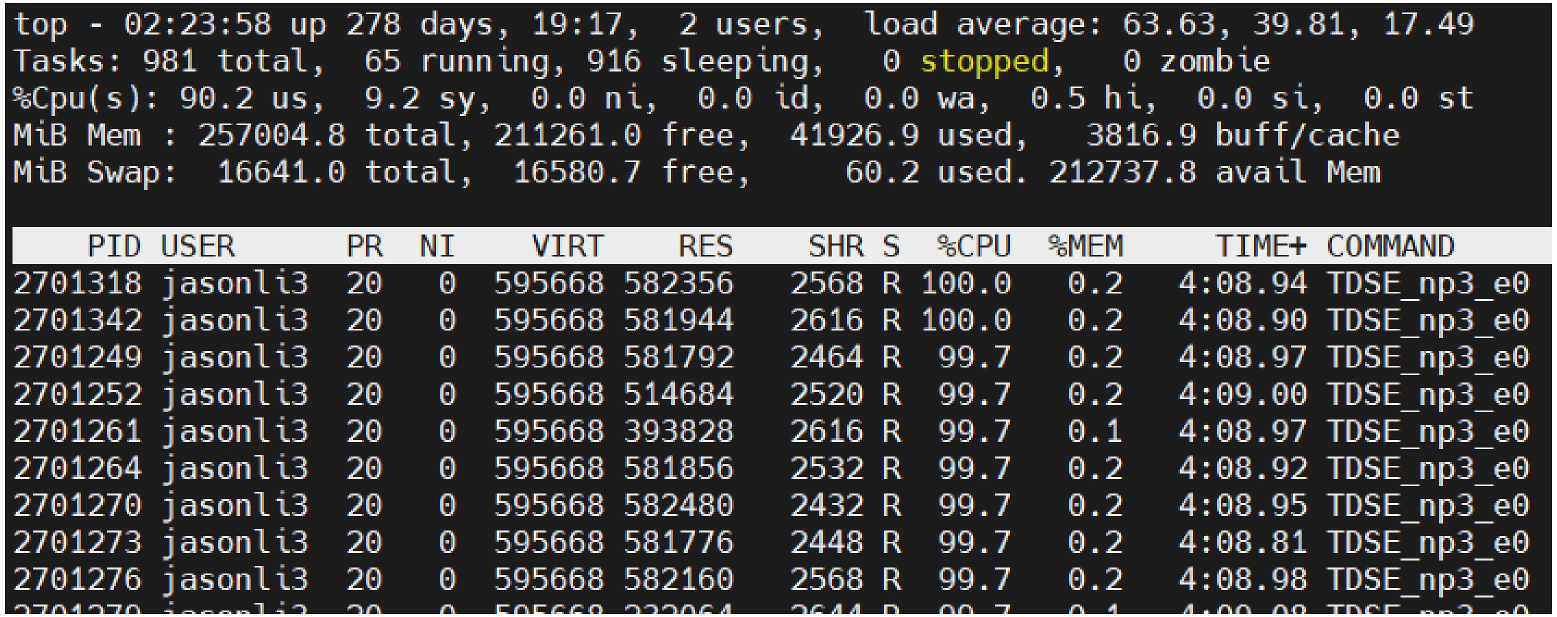
b) Method 2: top

- 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

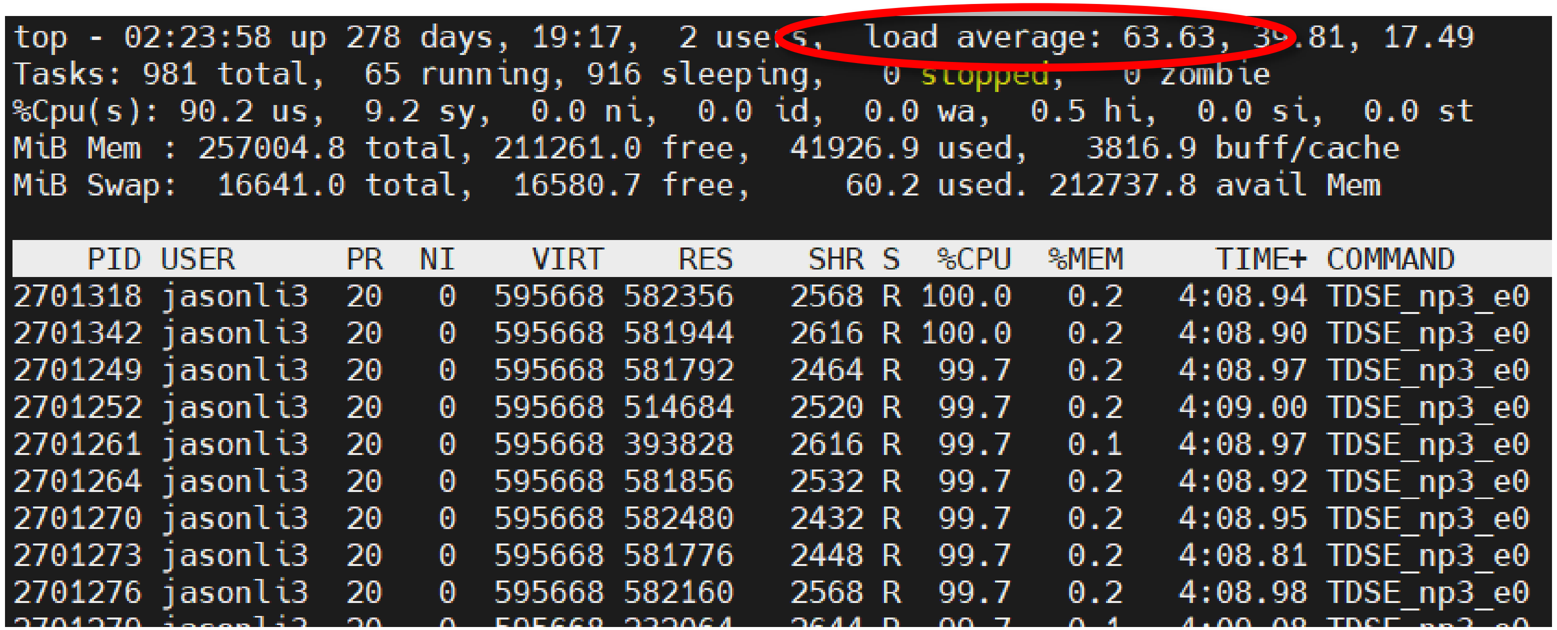


```
top - 02:23:58 up 278 days, 19:17,  2 users,  load average: 63.63, 39.81, 17.49
Tasks: 981 total,  65 running, 916 sleeping,   0 stopped,   0 zombie
%Cpu(s): 90.2 us,  9.2 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.5 hi,  0.0 si,  0.0 st
MiB Mem : 257004.8 total, 211261.0 free,  41926.9 used,   3816.9 buff/cache
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
2701279 jasonli3  20   0  595668 232064   2644 R  99.7   0.1   4:09.08 TDSE_np3_e0
```

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

2) Monitoring job health

b) Method 2: top

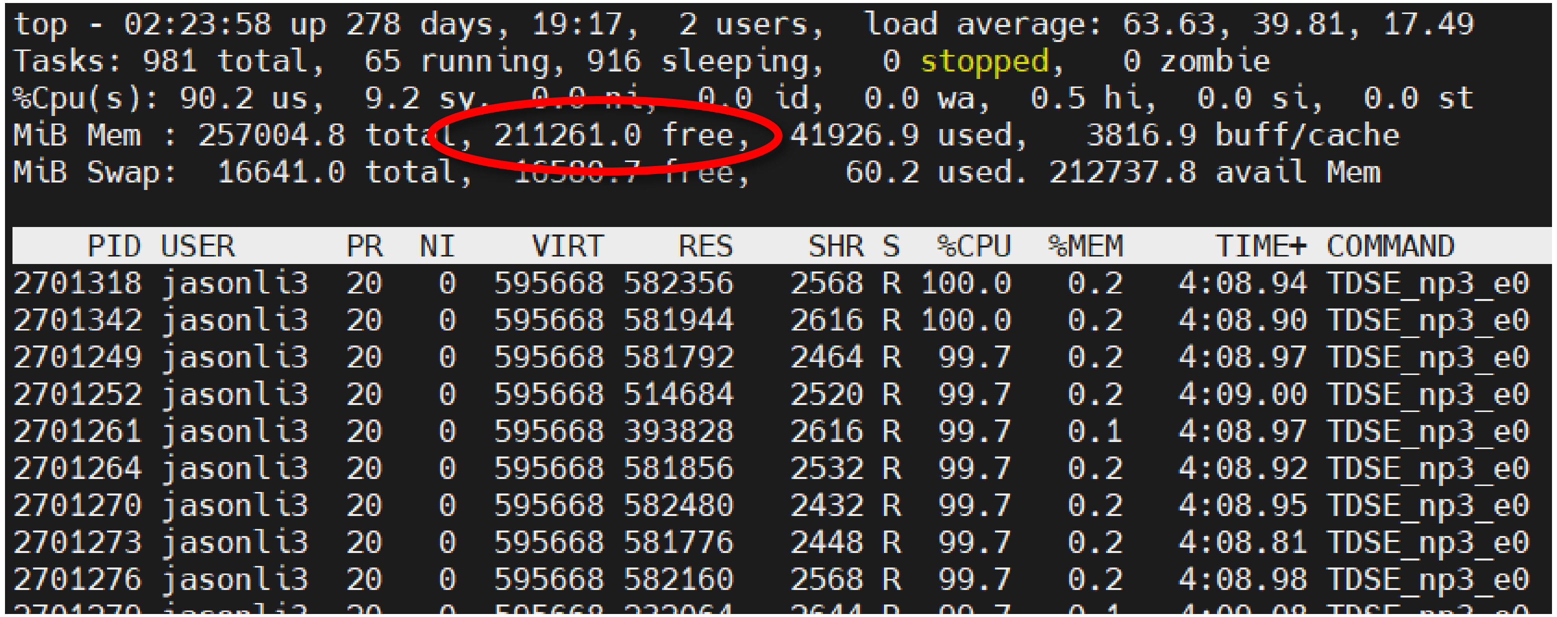


```
top - 02:23:58 up 278 days, 19:17,  2 users,  load average: 63.63, 39.81, 17.49
Tasks: 981 total,  65 running, 916 sleeping,   0 stopped,   0 zombie
%Cpu(s): 90.2 us,  9.2 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.5 hi,  0.0 si,  0.0 st
MiB Mem : 257004.8 total, 211261.0 free,  41926.9 used,   3816.9 buff/cache
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 number of cores or ppn	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
2701279 jasonli3  20   0  595668 333064   2644 R  99.7   0.1   4:09.08 TDSE_np3_e0
```

What to look at …	Normal behavior …	You should be concerned if …
Load average	Close to number of cores or ppn	Consistently too low or too high
Memory usage (not virtual memory)	Not used up	Used up

2) Monitoring job health

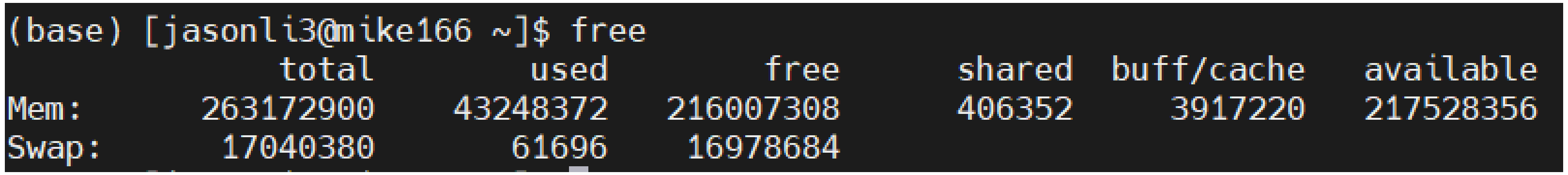
c) Method 3: free

- Displays free and used **physical and swap memory** in the system
- 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

c) Method 3: free

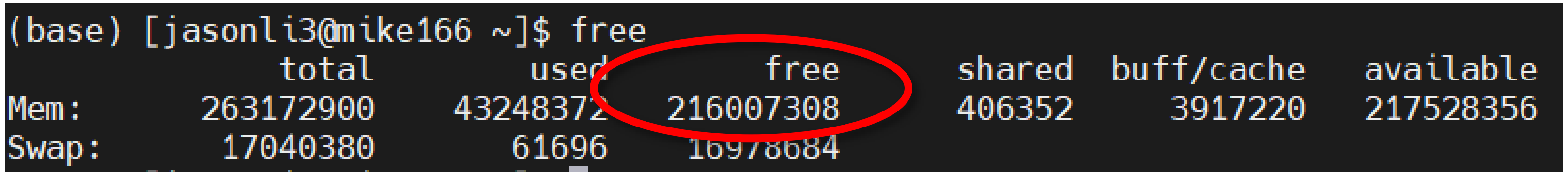


```
(base) [jasonli3@mike166 ~]$ free
              total        used        free      shared  buff/cache   available
Mem:      263172900    43248372   216007308      406352     3917220   217528356
Swap:      17040380       61696    16978684
```

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

2) Monitoring job health

c) Method 3: free



```
(base) [jasonli3@mike166 ~]$ free
              total        used        free      shared  buff/cache   available
Mem:      263172900    43248372   216007308      406352     3917220   217528356
Swap:      17040380       61696    16978684
```

What to look at …	Normal behavior …	You should be concerned if …
Memory usage (not virtual memory)	Not used up	Used up

2) Monitoring job health

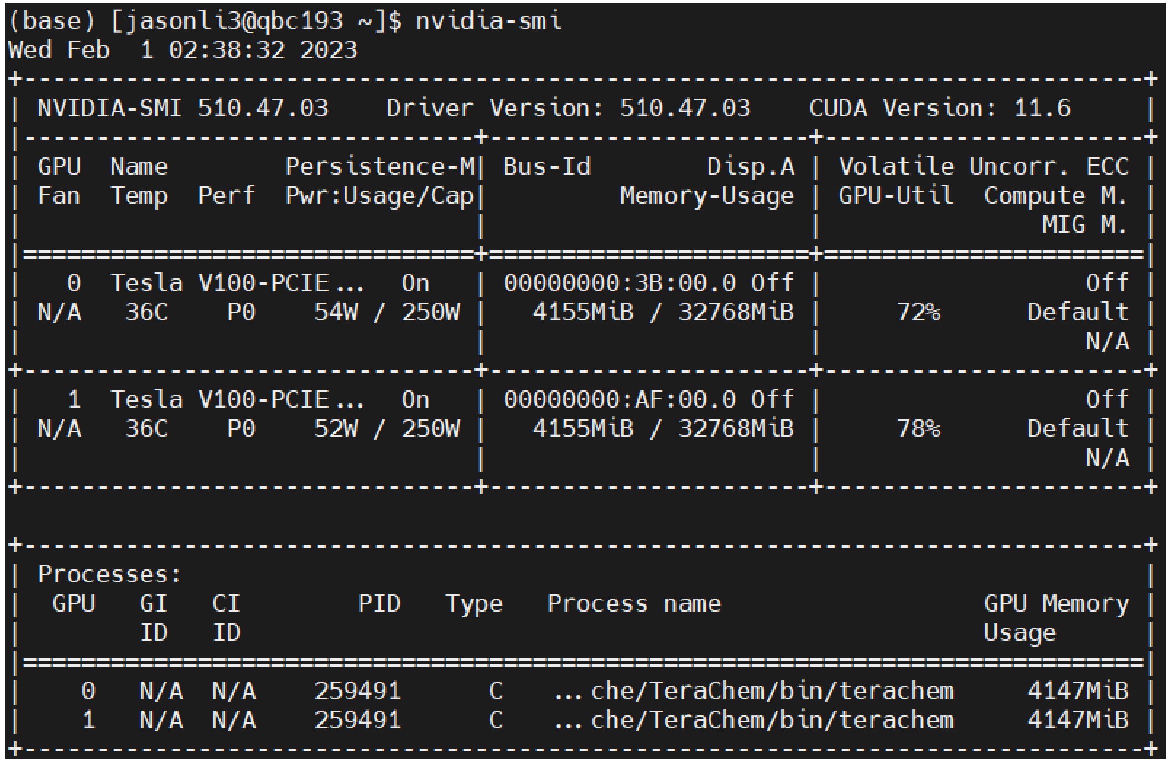
d) Method 4: nvidia-smi (for GPU only)

- Displays diagnostic information of GPUs
- Must run on **GPU nodes** !

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

2) Monitoring job health

d) **Method 4: nvidia-smi** (for GPU only)

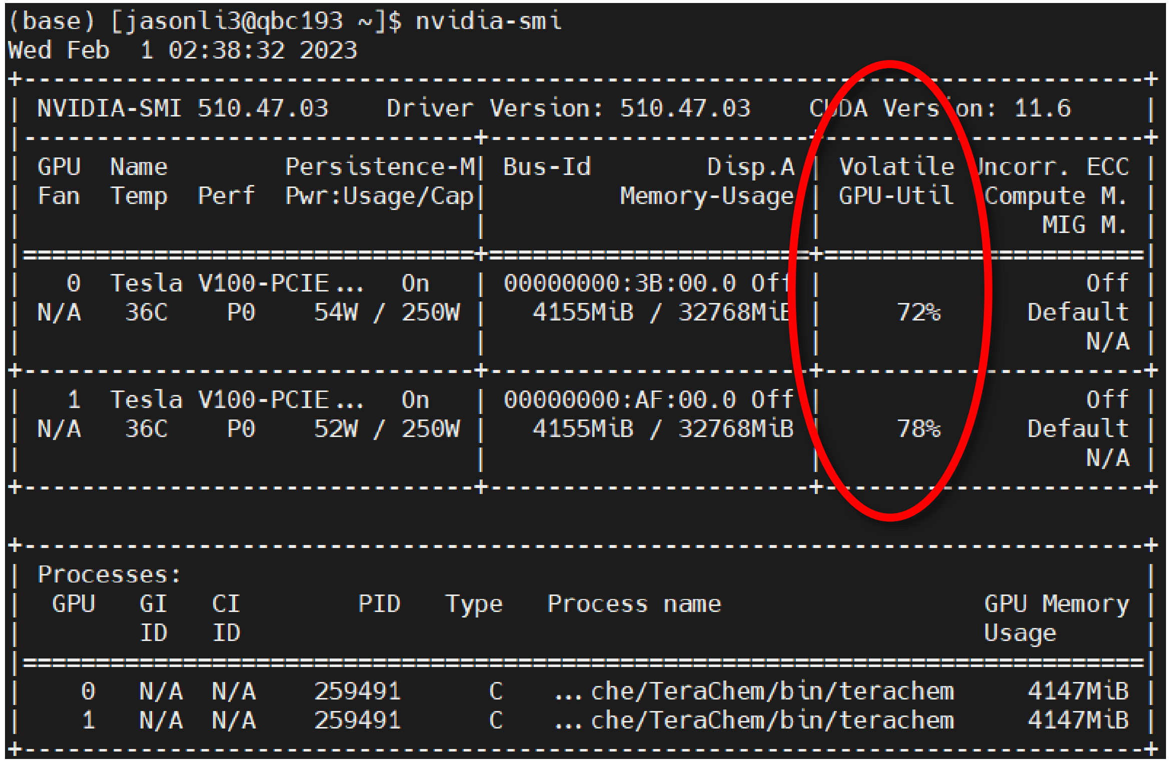


```
(base) [jasonli3@qbc193 ~]$ nvidia-smi
Wed Feb  1 02:38:32 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| 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

d) **Method 4: nvidia-smi** (for GPU only)

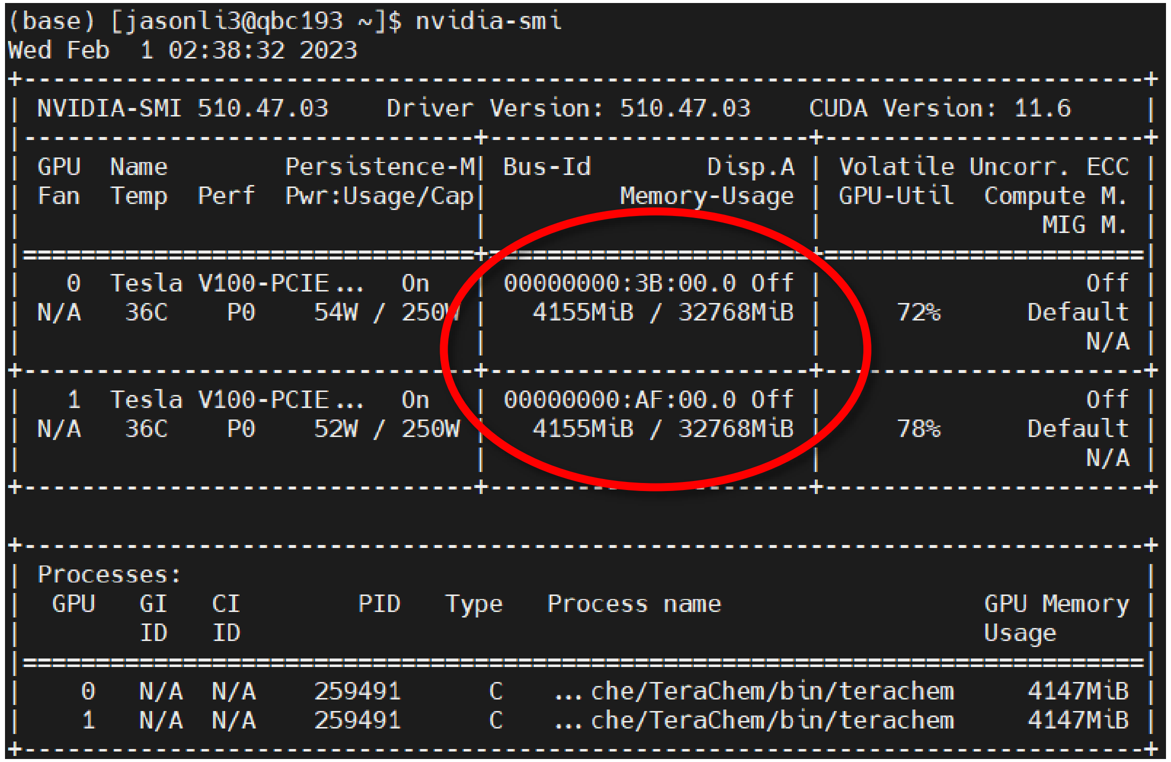


```
(base) [jasonli3@qbc193 ~]$ nvidia-smi
Wed Feb  1 02:38:32 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| 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

d) **Method 4: 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

2) Monitoring job health

e) 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 (e.g., Requested multiple nodes but only runs on one node)	Receive email warning.
Submitting to bigmem but only using little memory	Nothing. Just not nice.
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)

2) Monitoring job health

e) Common issues

Issue	What would happen

2) Monitoring job health

e) Common issues

Issue	What would happen
Exceeded memory allocation (e.g., using more memory than allocated w/ single queue)	Terminated. Receive email notice.

2) Monitoring job health

e) 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.

2) Monitoring job health

e) 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 (e.g., Requested multiple nodes but only runs on one node)	Receive email warning.

2) Monitoring job health

e) 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 (e.g., Requested multiple nodes but only runs on one node)	Receive email warning.
Submitting to bigmem but only using little memory	Nothing. Just not nice.

2) Monitoring job health

e) 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 (e.g., Requested multiple nodes but only runs on one node)	Receive email warning.
Submitting to bigmem but only using little memory	Nothing. Just not nice.
Running intensive calculation on head nodes	Terminated. Receive email notice.

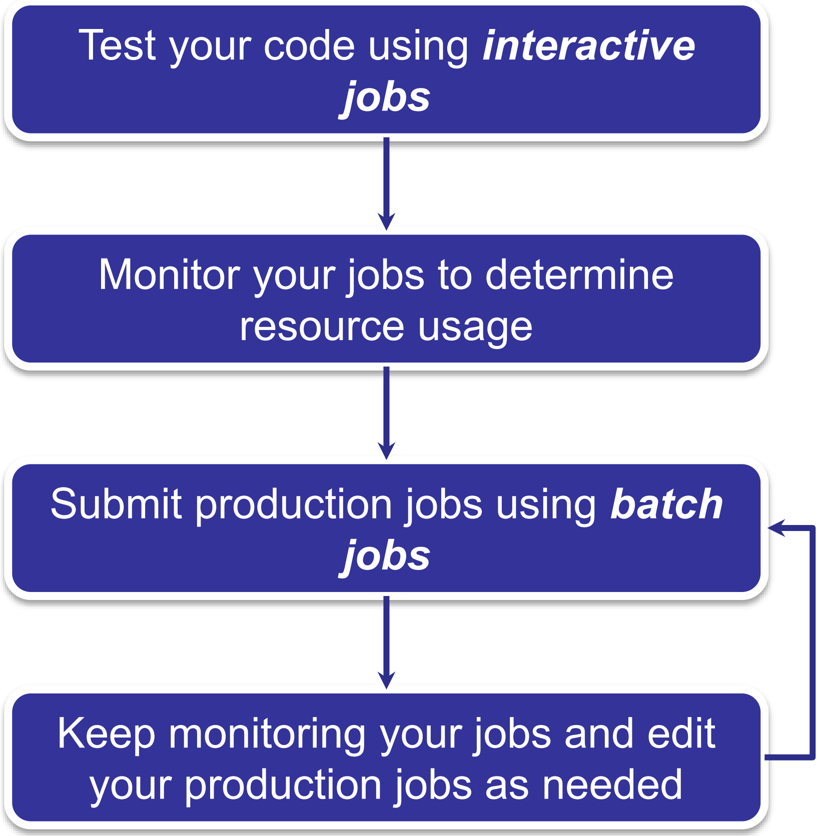
2) Monitoring job health

e) 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 (e.g., Requested multiple nodes but only runs on one node)	Receive email warning.
Submitting to bigmem but only using little memory	Nothing. Just not nice.
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 --**



Test your code using ***interactive jobs***

Monitor your jobs to determine resource usage

Submit production jobs using ***batch jobs***

Keep monitoring your jobs and edit your production jobs as needed

Take home message

- **HPC User Environment 2**

1. Basic concepts → **All calculation must be submitted as jobs**
2. How jobs are handled
   1) Job schedulers → **A “traffic police” to schedule user jobs**
   2) Job queues
   3) Choose your queue → **Get to know different queues and how to choose queues**
3. Submitting a job
   1) Interactive job → **Good for testing and debugging**
   2) Batch job → **Good for production**
   3) Cheat sheets
4. Manage 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