HPC User Environment 2

Oleg N. Starovoytov
HPC User Services
LSU HPC / LONI
sys-help@loni.org

Louisiana State University, Baton Rouge
February 4th, 2026

Outlines

- **HPC User Environment 1**

  1. Introduction to HPC
  2. Getting Started
  3. Introduction to the Cluster
  4. Software Environment & Modules

- **HPC User Environment 2**

  1. Basic Concepts
  2. Preparing My Job
  3. Submitting My Job
  4. Managing My Job

Outlines

- **HPC User Environment 2**

1. **Basic concepts**
   1) Review of HPC User Environment 1
   2) Job and job scheduler
2. **Preparing my job**
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queue and partitions
3. **Submitting my job**
   1) Interactive job
   2) Batch job
4. **Managing my job**
   1) Useful commands
   2) Monitoring job health

Outlines

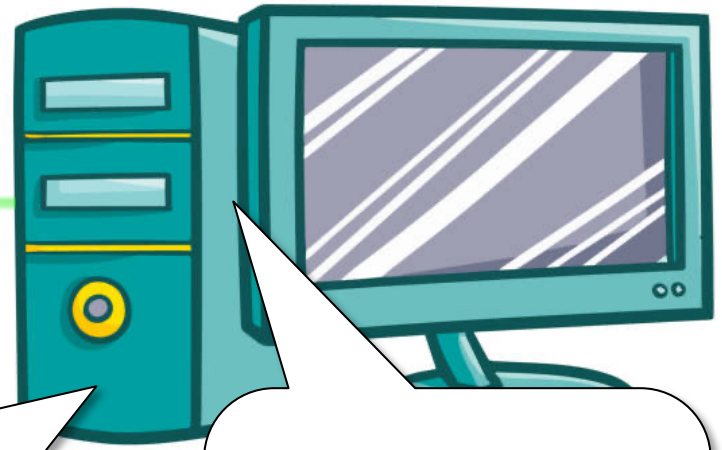
- **HPC User Environment 2**

1. Basic concepts
   1) Review of HPC User Environment 1
   2) Job and job scheduler
2. **Preparing my job**
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queue and partitions
3. **Submitting my job**
   1) Interactive job
   2) Batch job
4. **Managing my jobs**
   1) Useful commands
   2) Monitoring job health

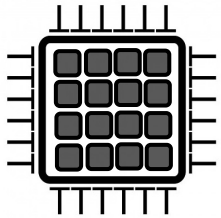
1) Review of HPC User Environment 1

Run my code on my machine as long as I would like to and until it is finished.

You do not share your resources with other people at home. That is your personal machine.

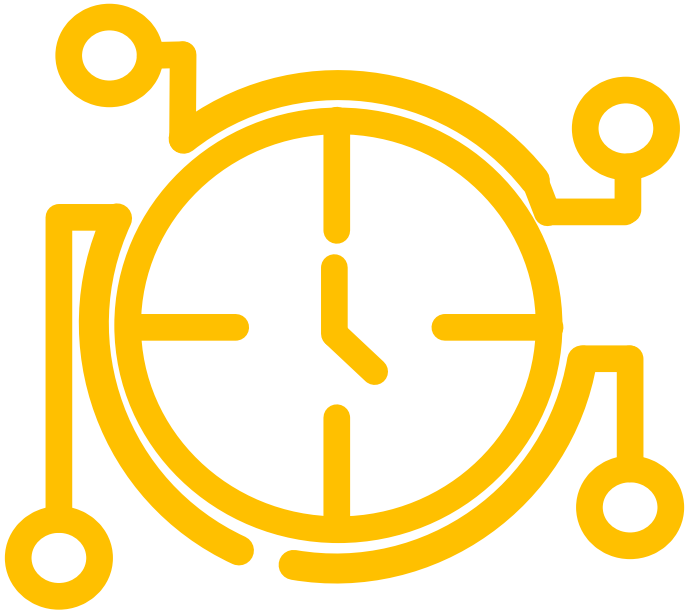


You can also install any software packages you want without restrictions using your root credentials with package management tools like yum, dnf, apt-get, etc.



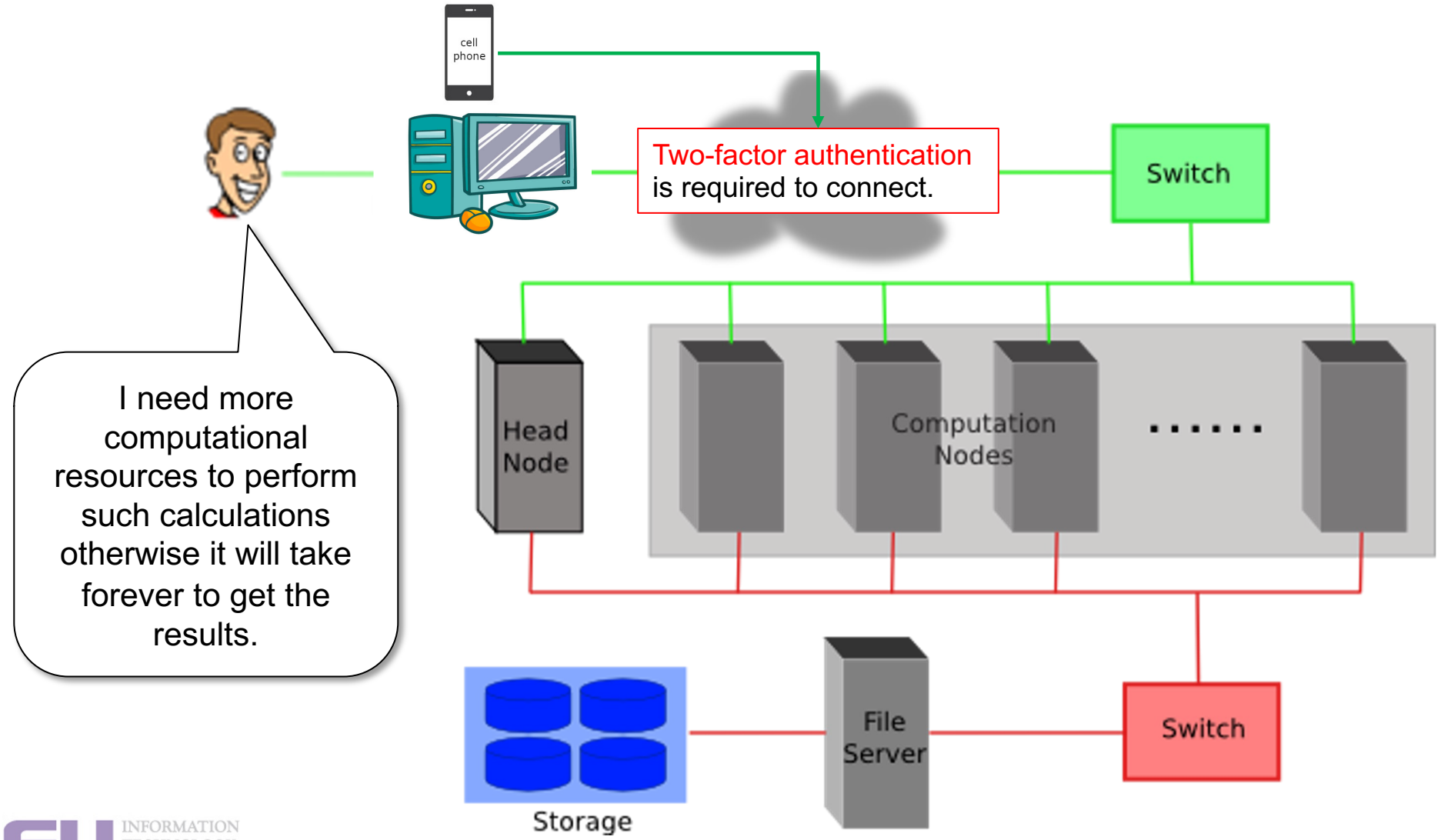
Multicore processor

Intel ® Core™ i7 ~ 4.0 GHz



You will be limited by the clock speed of your personal machine and a number of available cores.

1) Review of HPC User Environment 1



How do I connect, request resources, and run heavy calculations on the cluster?

What should I do about that?

1) Review of HPC User Environment 1

1) Account

Account

- An **individual** who is authorized to access a computing cluster based on their affiliation, credentials, and permissions.

2) Allocation

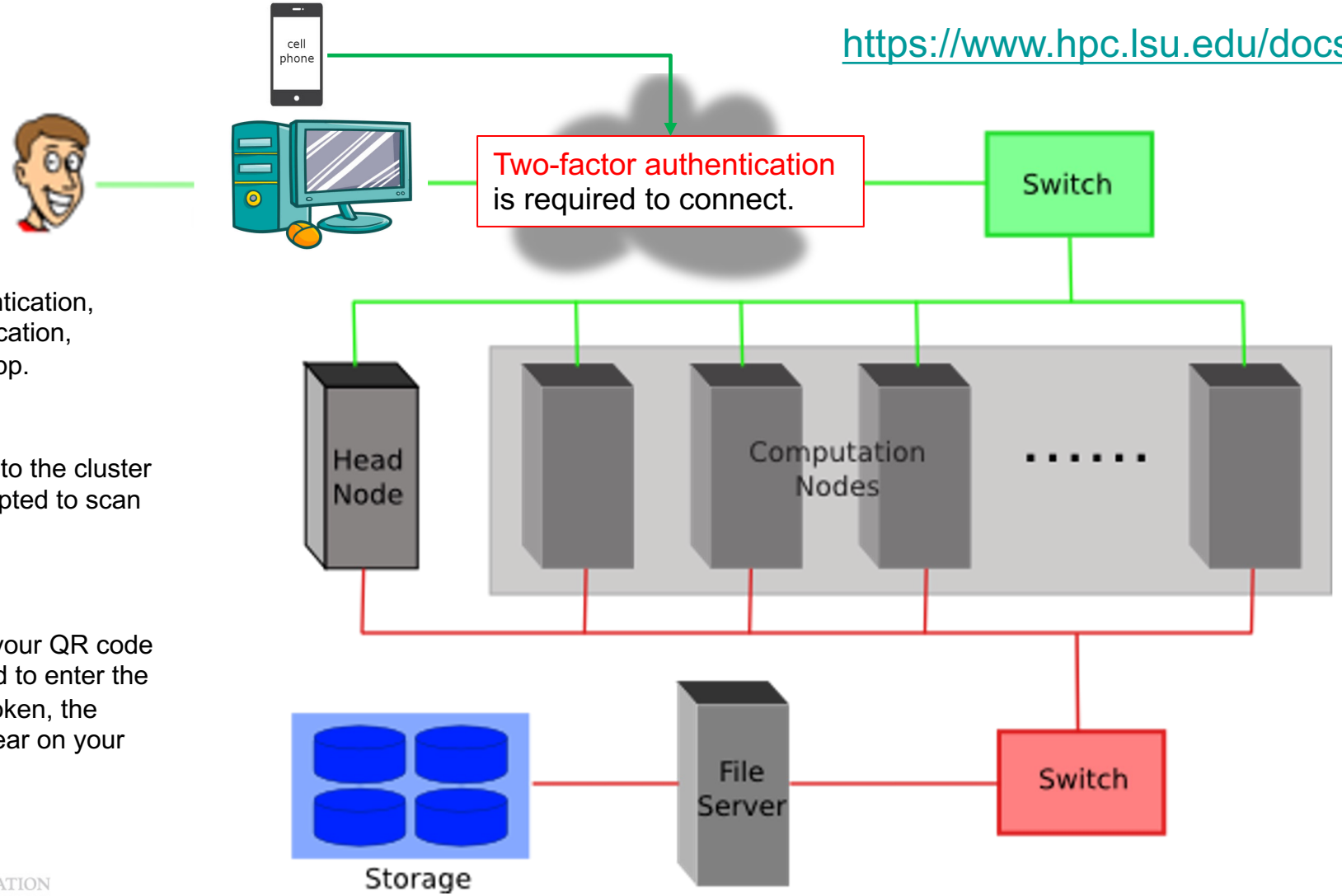
Allocation

- A share of computing **resources** (like CPU cores or GPU hours) granted to an account for a specific period.

http://www.hpc.lsu.edu/links.php

1) Review of HPC User Environment 1

https://www.hpc.lsu.edu/docs/systems/twofa.php



Step 1:
Microsoft Authentication, Google Authentication, or Mobile Due app.

Step 2:
While you log in to the cluster you will be prompted to scan your QR code.

Step 3:
Once you scan your QR code you will be asked to enter the Authentication token, the number will appear on your app.

Step 4:
If you did not scan the QR code the first time, you should be able to log in to the cluster in another way.

Outlines

- **HPC User Environment 2**

1. Basic concepts
    1) Review of HPC User Environment 1
    2) Job and job scheduler
2. Preparing my job
    1) Basic principles
    2) Job duration (wall time)
    3) Number of nodes and cores
    4) Job queues and partitions
3. Submitting my job
    1) Interactive job
    2) Batch job
4. Managing my jobs
    1) Useful commands
    2) Monitoring job health

2) Job and job scheduler

Job Components

Job

A computational task submifted to the clusier for execution.

Job Script

A file containing all instructions and parameters needed to run the job, written in shell scr pting language. it specifies the command resource. requirements, and job specific cnfigurations.

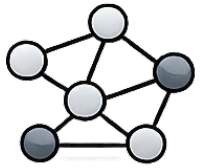
Executable

The program or application that performs the actu- computation.

2) Job and job scheduler

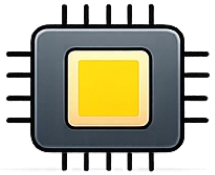
Partition type

-p



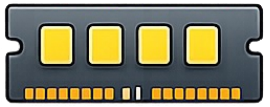
Number of nodes

-N



Number of cores

-n



Amount of memory

Not supported?

Time limit

-t

2) Job and job scheduler

Problem:

You are given a **startup allocation of 150,000 Service Units (SUs)** to use on an HPC cluster. Each compute node contains **64 cores**. Service Units are charged based on the **number of cores used multiplied by the number of hours the job runs**. (CORES x HOURS)

You plan to run a job that uses **two nodes (128 cores)** and is expected to run for **10 hours**.

Your task is to **calculate how many Service Units this job will consume** and then **determine how many Service Units will remain** in your allocation after the job finishes.

1 SU Unit = 1 CORE / hour

2) Job and job scheduler

Problem:

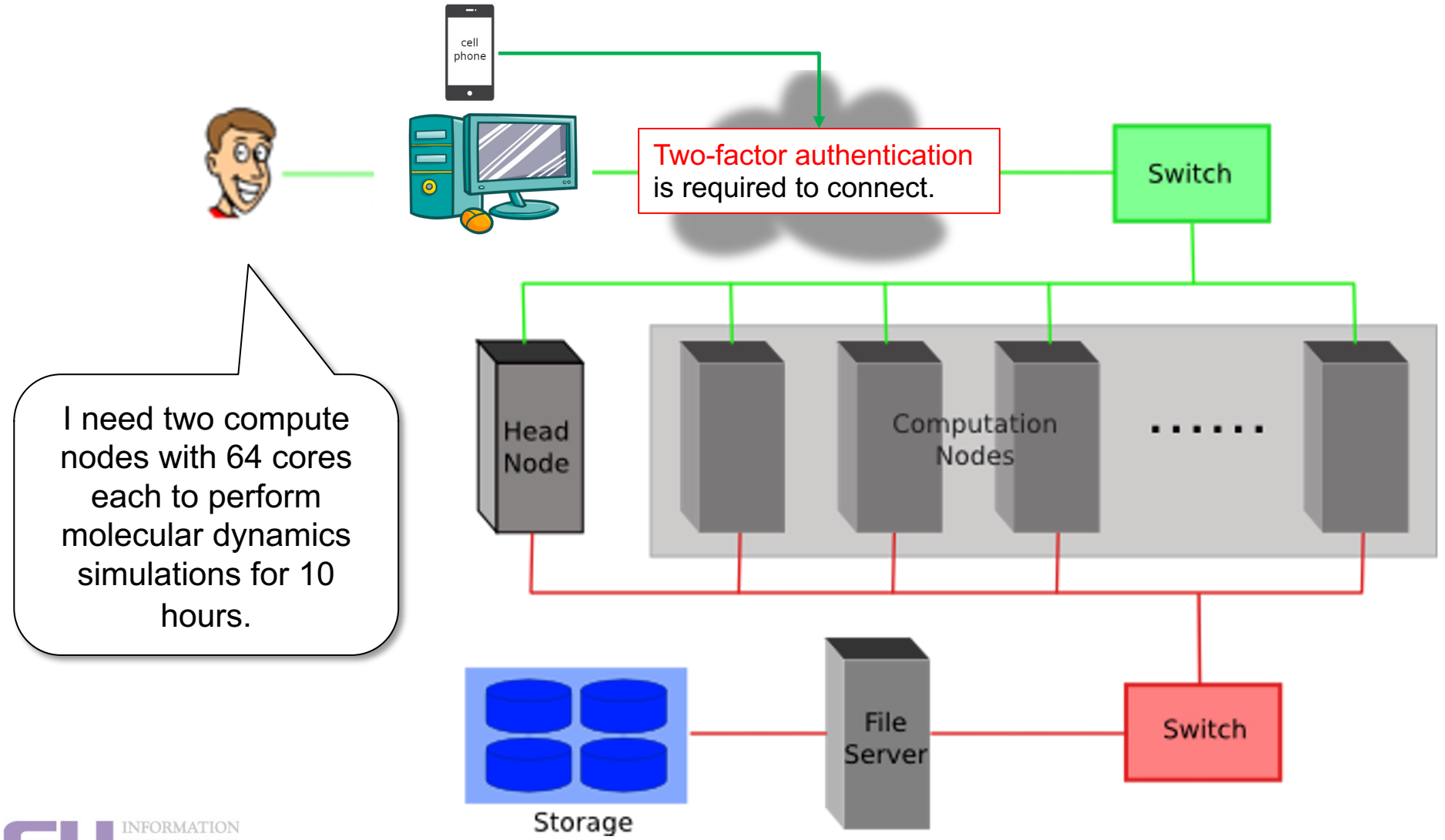
You are given a **startup allocation of 150,000 Service Units (SUs)** to use on an HPC cluster. Each compute node contains **64 cores**. Service Units are charged based on the **number of cores used multiplied by the number of hours the job runs**. (CORES x HOURS)

You plan to run a job that uses **two nodes (128 cores)** and is expected to run for **10 hours**.

Your task is to **calculate how many Service Units this job will consume** and then **determine how many Service Units will remain** in your allocation after the job finishes.

1 SU Unit = 1 CORE / hour

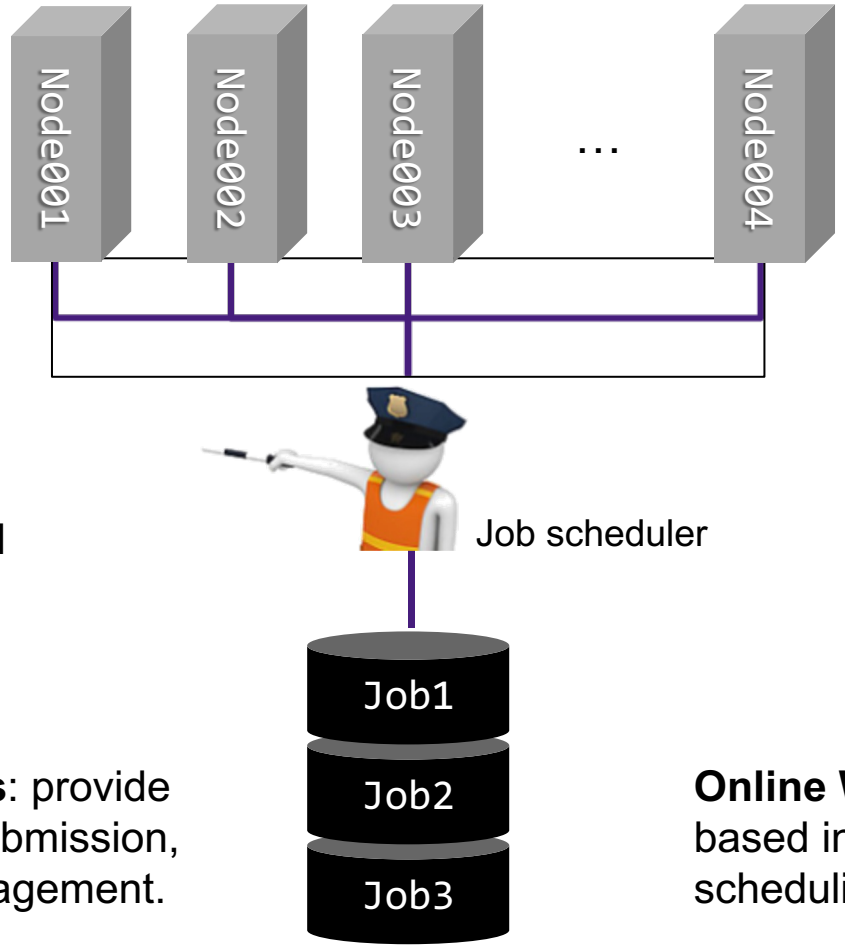
2) Job and job scheduler



2) Job and job scheduler

Allocation of resources: queue management and allocation of the resources based on job requirements.

Priority scheduling: Assigns priority to jobs based on required computational resources and job duration. It also checks for fair resource distribution.



Monitoring and reporting: monitoring of the job progress, resource usage, and job status.

Fault tolerance and recovery: Supports job checkpointing and recovery. It also handles job rescheduling when a compute node fails.

Command line tools: provide users with the tool submission, monitoring, and management.

Online Web tools: support web-based interfaces for easier job scheduling and management.

2) Job and job scheduler

Job Scheduler functions	HPC User responsibilities
• Allocation of computational resources • Priority job scheduling • Monitoring of computational resources • Recovery and failures • Support command line tools and web tools	• Decide a job's size and duration • Understand the job queuing system and policies • Submit/monitor/cancel jobs • Monitoring job status

2) Job and job scheduler

	LSU HPC	LONI
	Deep Bayou SuperMike III SuperMIC	QB3 (QBC) QB4 (QBD)

2) Summary

https://www.ibm.com/docs/en/spectrum-lsf/10.1.0

LSF (Load Sharing Facility)

https://slurm.schedmd.com/documentation.html

SLURM (Simple Linux Utility for Resource Management)

https://github.com/openpbs/openpbs

PBS (Portable Batch System)

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Review of HPC User Environment 1
   2) Job and job scheduler
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queues and partitions
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

Guidelines for Requesting HPC Resources

Request sufficient resources

Large enough to meet computational needs.

Avoid overestimation

Small enough to optimize resource usage and access.

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Review of HPC User Environment 1
   2) Job and job scheduler
2. Preparing my job
   1) Basic Principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queues and partitions
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my jobs
   1) Useful commands
   2) Monitoring job health

#SBATCH --time=00-10:10:30

2) Job duration (wall time)

HPC Queue Information

Partition Name	Max Walltime (hours)	Max Nodes (per user)	Allowed Cores per Node
single	168		1 to 64 Cores
workq	72		64 Cores
checkpt	72		64 Cores
gpu2	72	Up to 4	32/64 Cores
gpu4	72	Up to 4	16/32/48/64 Cores
bigmem	72	Up to 4	64 Cores

2) Job duration (wall time)

```
[username@hostname ~]$ sinfo -s
```

```
PARTITION AVAIL  TIMELIMIT   NODES(A/I/O/T) NODELIST
single*      up 7-00:00:00     415/0/65/480 qbd[001-480]
checkpt      up 3-00:00:00     415/0/65/480 qbd[001-480]
workq        up 3-00:00:00     415/0/65/480 qbd[001-480]
bigmem       up 3-00:00:00          5/0/0/5 qbd[481-485]
gpu2         up 3-00:00:00       35/17/0/52 qbd[486-511,517-542]
gpu4         up 3-00:00:00         2/8/0/10 qbd[512-516,543-547]
```

2) Job duration (wall time)

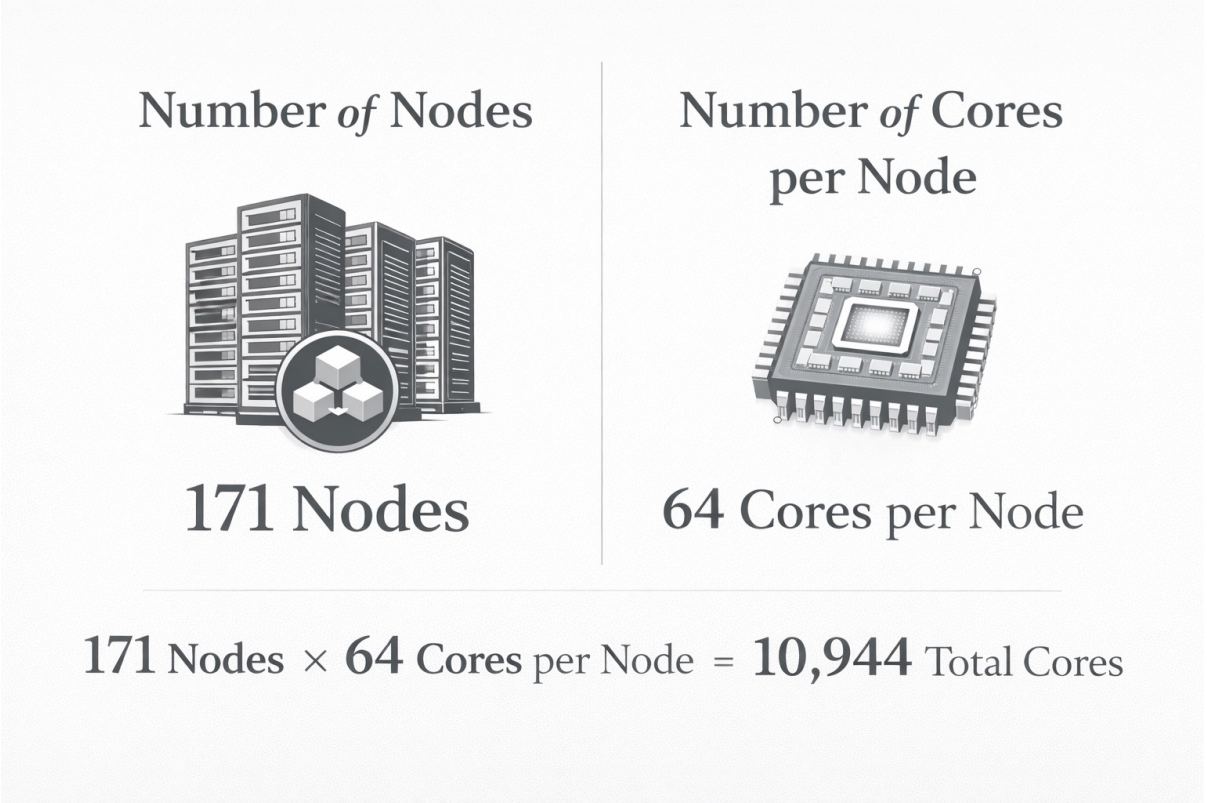
Questions	Answers
• What if my command is still running when the wall time runs out?	• The job is **terminated**. Any running process will be killed.
• What if all my commands in the job finished before the wall time runs out?	• The job will **exit** successfully when all commands finish.
• If my job exits before requested wall time, how many SUs will I be charged?	• You will be charged based on your **actual time used** (if it is less than the requested amount).
• In that case, why don't I just request maximum wall time every time?	• The queuing time might be long.

https://www.hpc.lsu.edu/docs/guides.php?system=QB4

2) Job duration (wall time)

- **HPC User Environment 2**

1. Basic concepts
   1) Review of HPC User Environment 1
   2) Job and job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queues and partitions
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 and cores

Questions	Answers
• 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 and cores

Deep Bayou	
Hostname	db1.hpc.lsu.edu
Peak Performance/TFlops	
Compute nodes	13
Processor/node	2 24-core
Processor Speed	2.4 - 3.0GHz
Processor Type	Intel Cascade Lake Xeon 64bit
Nodes with GPUs	13
Accelerator Type	2 x NVIDIA V100 32GB
OS	RHEL v8
Vendor	Dell
Memory per node	192 GB
Detailed Cluster Description	
User Guide	
Available Software	

SuperMIC	
Hostname	smic.hpc.lsu.edu
Peak Performance/TFlops	1000
Compute nodes	360
Processor/node	2 10-core
Processor Speed	2.8GHz
Processor Type	Intel Xeon 64bit
Nodes with GPUs	4
Accelerator Type	2 x NVIDIA V100 16GB
OS	RHEL v6
Vendor	Dell
Memory per node	64 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 GPUs	8
Accelerator Type	4 x NVIDIA A100 40GB
OS	RHEL v8
Vendor	Dell
Memory per node	256 GB
Detailed Cluster Description	
User Guide	
Available Software	

https://www.hpc.lsu.edu/resources/hpc/index.php#loni

3) Number of nodes and cores

QB2 (QB)	
Hostname	qb.loni.org
Peak Performance/TFlops	1,052
Compute nodes	504
Processor/node	2 10-core
Processor Speed	2.8 GHz
Processor Type	Intel Xeon
Nodes with GPUs	4
Accelerator Type	2 x NVIDIA K40 8GB
OS	RHEL v6
Vendor	Dell
Memory per node	64 GB
Retired in March 2020	

QB3 (QBC)	
Hostname	qbc.loni.org
Peak Performance/TFlops	857
Compute nodes	202
Processor/node	2 24-core
Processor Speed	2.4 GHz
Processor Type	Intel Xeon Cascade Lake
Nodes with GPUs	8
Accelerator Type	2 x NVIDIA V100 32GB
OS	RHEL v8
Vendor	Dell
Memory per node	192 GB
Detailed Cluster Description	
User Guide	
Available Software	

QB4 (QBD)	
Hostname	qbd.loni.org
Peak Performance/TFlops	4,300
Compute nodes	547
Processor/node	2 32-core
Processor Speed	2.6GHz
Processor Type	Intel Xeon Ice Lake
Nodes with GPUs	62
Accelerator Type	2/4 x NVIDIA A100 80GB
OS	RHEL v8
Vendor	Dell
Memory per node	256 GB
Detailed Cluster Description	
User Guide	
Available Software	

https://www.hpc.lsu.edu/resources/hpc/index.php#loni

3) Number of nodes and cores

```
[username@hostname ~]$ ssh –X username@hostname

(username@hostname) Password:

(username@hostname) TOTP Authenticator Token:
```

3) Number of nodes and cores

```
##################################################################################
Send questions and comments to the email ticket system at sys-help@loni.org.
##################################################################################

QB-4 at LONI (Open for general use)

# 02-May-2024

QB-4 is a 4.3 PetaFlop peak performance cluster with 35,008 CPU cores and
144 NVIDIA A100 GPUs, comprised of 547 compute nodes connected by 200 Gbps
Infiniband fabric.

Node Specs:
compute CPUs=64 SocketsPerBoard=2 CoresPerSocket=32 ThreadsPerCore=1 RealMemory=256000 DefMemPerCPU=3920 MB
bigmem  CPUs=64 SocketsPerBoard=2 CoresPerSocket=32 ThreadsPerCore=1 RealMemory=2063000 DefMemPerCPU=32100 MB
gpu2    Gres=gpu:2 CPUs=64 SocketsPerBoard=2 CoresPerSocket=32 ThreadsPerCore=1 RealMemory=514000 DefMemPerGPU=254000 MB
gpu4    Gres=gpu:4 CPUs=64 SocketsPerBoard=2 CoresPerSocket=32 ThreadsPerCore=1 RealMemory=514000 DefMemPerGPU=127000 MB
```

3) Number of nodes and cores

```
Quotas for the /home volume are enabled at 10 GB.  Please do
not use the /home volume for batch job I/O, use the /work volume
instead.  Please limit the number of files per directory to 10,000.
No disk GB quotas are in effect for the /work volume, but files are purged that
are older than 60 days.  There are quotas on file counts.  Manage your files,
but it is against our policy to blatantly circumvent the purge procedure.
Misconfigured jobs that are wasting CPU resources are subject to deletion.

**************************************************************************
Allocations are required. Use -A with sbatch/srun/qsub to specify your allocation.
**************************************************************************

NOTE: The same compute allocation can be used on LONI QB-2, QB-3, and QB-4
NOTE: /project and /work are shared between QB-3 and QB-4

If you have any questions or need assistance migrating your code, please
contact sys-help@loni.org
```

3) Number of nodes and cores

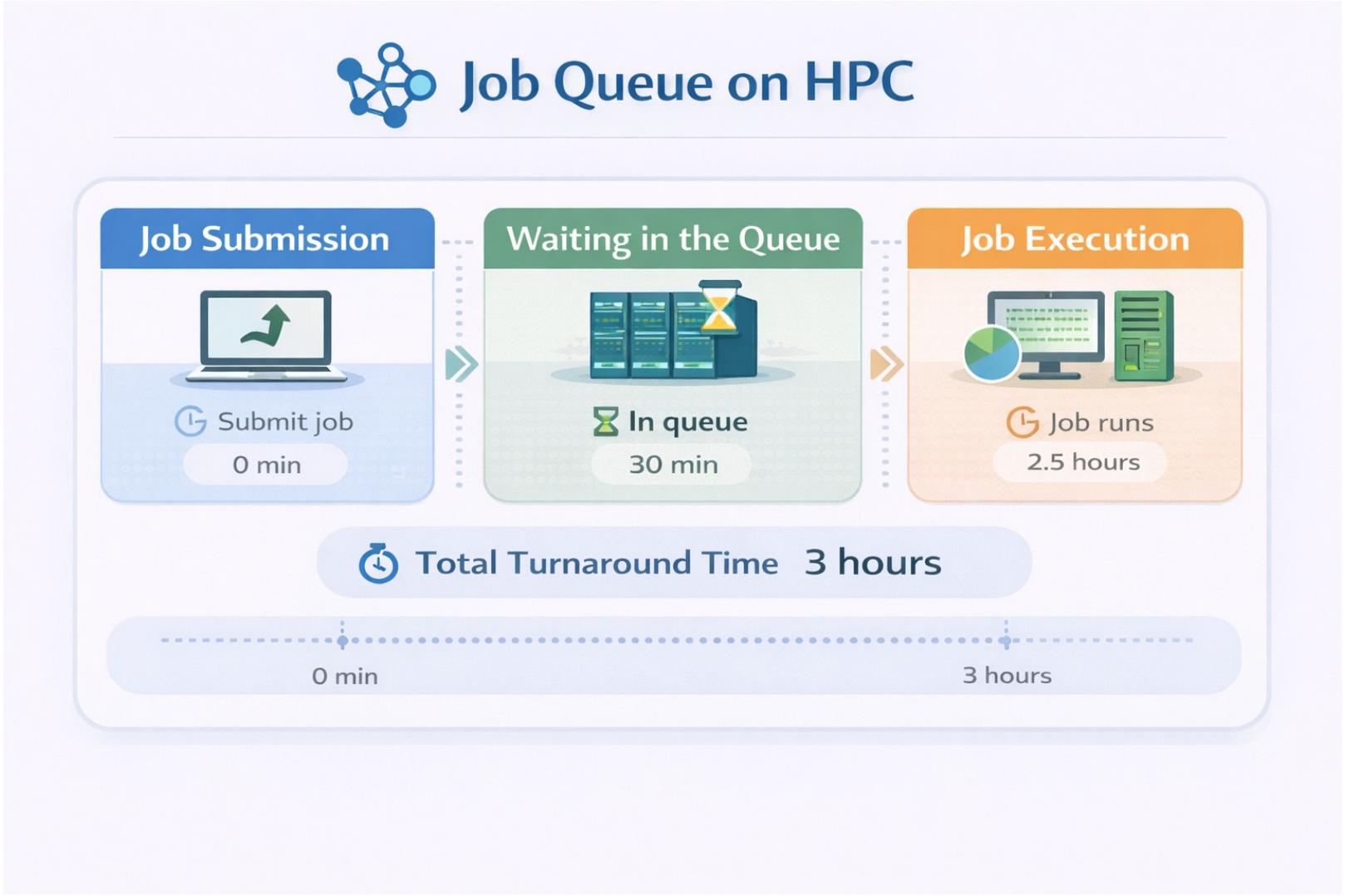
```
---------------- RECENT UPDATES ------------------

For Two-Factor Authentication help see:
https://hpc.loni.org/docs/systems/twofa.php

# 06Jan2026 Upgrades
* GPU Drivers now support CUDA 13.1
* Added hardware information to top of message of the day.
* Upgraded Slurm scheduler version
* Enabled rate limits to prevent abuse of client commands when accidentally looped.
  Do not run client commands like srun, sinfo, squeue, etc in a loop.
```

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Review of HPC User Environment 1
   2) Job and job scheduler
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queues and partitions
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my job
   1) Useful commands
   2) Monitoring job health

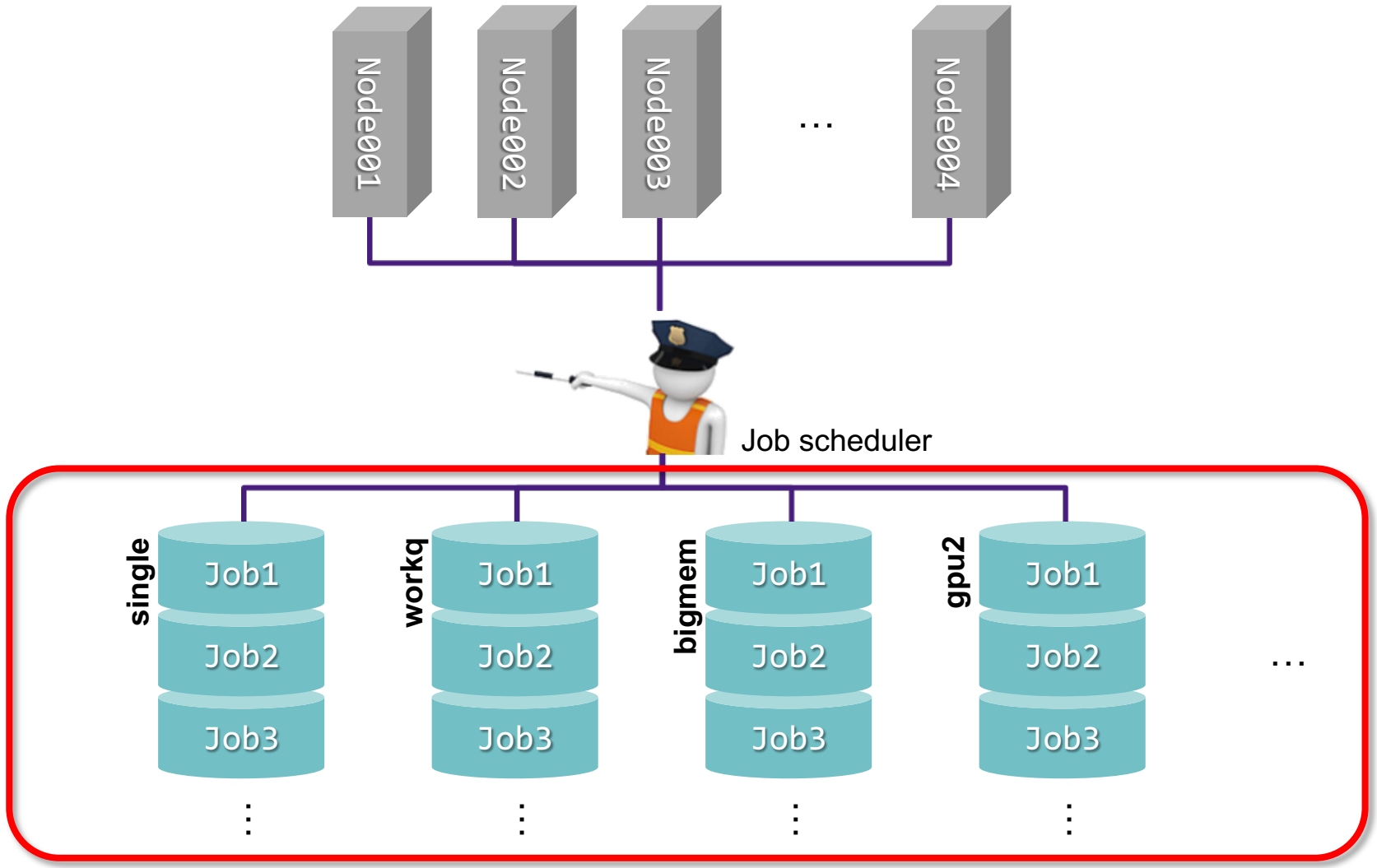
4) Job queue on HPC



4) Job queue on HPC

Queue - a scheduler-defined waiting list of jobs with shared policies

4) Job queue on HPC



4) Job queue on HPC

```
[username@hostname ~]$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME_LIMIT	TIME	CPUS	NODES	NODELIST(REASON)
525028	bigmem	ATAC_cell_type_heatm	ygong	R	3-00:00:00	9:42:02	64	1	qbd481
523481	bigmem	P150GPa	jhinz	R	3-00:00:00	1-03:13:07	128	2	qbd[484-485]
525191	bigmem	AFM	agiri	R	2-00:00:00	8:03:49	128	2	qbd[482-483]
526416	checkpt	batch_cpu_newqb4.sh	sid	PD	3-00:00:00	0:00	1024	16	(Resources)
525418	checkpt	ScAlN-MC-30	hsh012	R	3-00:00:00	7:29:15	256	4	qbd[286-289]
525199	checkpt	ScAlN-MC-50	hsh012	R	3-00:00:00	7:47:08	256	4	qbd[294-297]
525536	checkpt	ScAlN-MC-10	hsh012	R	3-00:00:00	6:18:29	256	4	qbd[350-353]
525523	checkpt	welding	collin	R	3-00:00:00	6:43:48	64	1	qbd053
524943	checkpt	welding	collin	R	3-00:00:00	10:29:00	64	1	qbd020
524931	checkpt	welding	collin	R	3-00:00:00	10:40:08	64	1	qbd015
524926	checkpt	welding	collin	R	3-00:00:00	10:43:11	64	1	qbd003
525882	checkpt	ScAlN-MC-20	hsh012	R	3-00:00:00	5:34:35	256	4	qbd[447-450]

4) Job queue on HPC

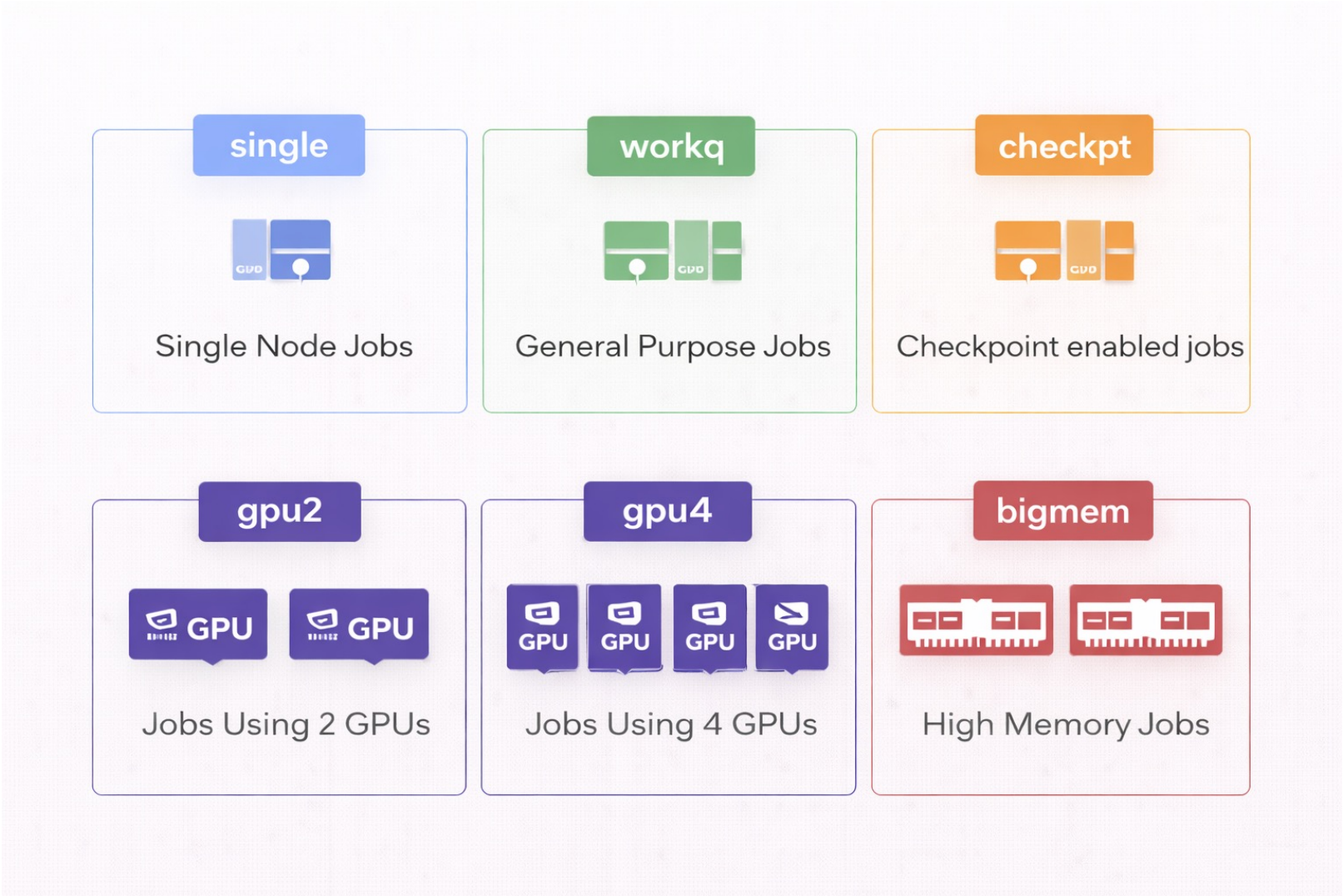
```
[username@hostname ~]$ squeue -l
```

```
JOBID PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
50878    bigmem submit-p  username1  RUNNING 2-06:28:02 3-00:00:00      1 qbd482
51761   checkpt CuFe_Fit  username2  RUNNING       1:29 1-00:00:00      1 qbd081
51257   checkpt GOM1km_e  username3  RUNNING 1-17:35:43 3-00:00:00     25 qbd[275-299]
51254   checkpt GOM1km_e  username3  RUNNING 1-17:43:59 3-00:00:00     25 qbd[431-455]
51252   checkpt GOM1km_e  username3  RUNNING 1-17:58:50 3-00:00:00     25 qbd[056-080]
50872   checkpt Asph50_S  username4  RUNNING 2-07:16:01 3-00:00:00      4 qbd[088-091]
51756      gpu2 interact  username5  RUNNING    1:07:05   12:00:00      1 qbd486
51740      gpu4 submit-1  username6  RUNNING    2:38:09 3-00:00:00      1 qbd545
51731      gpu4 submit-1  username6  RUNNING    2:55:23 3-00:00:00      1 qbd544
51728      gpu4 submit-1  username6  RUNNING    3:14:49 3-00:00:00      1 qbd543
49034     workq dynAB+cp  username7  PENDING       0:00      10:00    200 (MaxNodePerAccount)
50873     workq   qbd.sh  username8  RUNNING 2-07:06:42 3-00:00:00     84 qbd[191-274]
```

```
[username@hostname ~]$ squeue -u username6
```

```
JOBID      PARTITION  NAME              USER      ST  TIME_LIMIT   TIME        CPUS  NODES  NODELIST(REASON)
51740      gpu4       submit-1-4.sh     username6 R   3-00:00:00   2:40:00     16    1      qbd545
51731      gpu4       submit-1-4.sh     username6 R   3-00:00:00   2:57:14     16    1      qbd544
```

4) Partitions



4) Partitions

Partition - a scheduler-defined group of compute nodes.

4) Partitions

```
[username@hostname ~]$ sinfo

PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
single*      up 7-00:00:00      1 drain* qbd422
single*      up 7-00:00:00      1  down* qbd041
single*      up 7-00:00:00    163  alloc qbd[056-080,088-091,191-299,431-455]
single*      up 7-00:00:00    315   idle qbd[001-040,042-055,081-087,092-190,300-421,423-430,456-480]
checkpt      up 3-00:00:00      1 drain* qbd422
checkpt      up 3-00:00:00      1  down* qbd041
checkpt      up 3-00:00:00    163  alloc qbd[056-080,088-091,191-299,431-455]
checkpt      up 3-00:00:00    315   idle qbd[001-040,042-055,081-087,092-190,300-421,423-430,456-480]
workq        up 3-00:00:00      1 drain* qbd422
workq        up 3-00:00:00      1  down* qbd041
workq        up 3-00:00:00    163  alloc qbd[056-080,088-091,191-299,431-455]
workq        up 3-00:00:00    315   idle qbd[001-040,042-055,081-087,092-190,300-421,423-430,456-480]
bigmem       up 3-00:00:00      1  alloc qbd482
bigmem       up 3-00:00:00      4   idle qbd[481,483-485]
gpu2         up 3-00:00:00     52   idle qbd[486-511,517-542]
gpu4         up 3-00:00:00      3    mix qbd[543-545]
gpu4         up 3-00:00:00      6   idle qbd[512-516,546]
gpu-small    up 3-00:00:00      1   idle qbd547
```

4) Partitions

```
[username@hostname ~]$ sinfo -s

PARTITION AVAIL  TIMELIMIT   NODES(A/I/O/T) NODELIST
single*      up 7-00:00:00     408/7/65/480 qbd[001-480]
checkpt      up 3-00:00:00     408/7/65/480 qbd[001-480]
workq        up 3-00:00:00     408/7/65/480 qbd[001-480]
bigmem       up 3-00:00:00          5/0/0/5 qbd[481-485]
gpu2         up 3-00:00:00       22/30/0/52 qbd[486-511,517-542]
gpu4         up 3-00:00:00         1/9/0/10 qbd[512-516,543-547]
```

4) Partitions

```
[username@hostname ~]$ sinfo –O partitionname

PARTITION
single
checkpt
workq
bigmem
gpu2
gpu4
```

4) Partitions

single

Purpose		• Only need a **portion** of one node
Names		• All clusters: **single**
Resource availability	**Nodes**	• **Portion** of one node: 1/2/4/8/16/32/64
	Cores	• **1** ~ **all** cores
	Memory	• A **portion**, **proportional** to the number of requested cores
Max duration		• **168** hours (7 days)

[QB-4]

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

4) Partitions

checkpt / workq

Purpose		• General purposes • Most likely your **default** queue • Difference: **non-preemptible** (workq) vs. **preemptible** (checkpt)
Name		• All clusters: **workq** / **checkpt**
Resource availability	**Nodes**	• **Entire** node(s) • 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) Partitions

bigmem

Purpose		• Need **large memory** (larger than regular computing nodes have)
Names		• All clusters: **bigmem**
Resource availability	**Nodes**	• **Entire** node(s)
	Cores	• **All** cores on the node
	Memory	• **All** memory on the node
Max duration		• **72** hours (3 days)

4) Partitions

gpuX

Purpose		• Need **GPU**	
Name		• QB-3: **gpu2**	• SMIC: **gpu2** • Deep Bayou: **gpu2**, **gpu4** • SuperMike 3: **gpu4** • QB-4: **gpu2**, **gpu4**
Resource availability	**Nodes**	• **Entire** node(s)	• **Portion** or **entire** node(s)
	Cores	• **All** cores on the node(s)	• **Portion** or **all** on the node
	Memory	• **All** memory on the node(s)	• **Portion** or **all** on the node
	GPU	• **All** GPUs on the node(s)	• **1** ~ **all** GPU on the node(
Max duration		• **72** hours (3 days)	

gpuX :
X = [Number of GPUs on one node]

[QBD / gpu4]

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

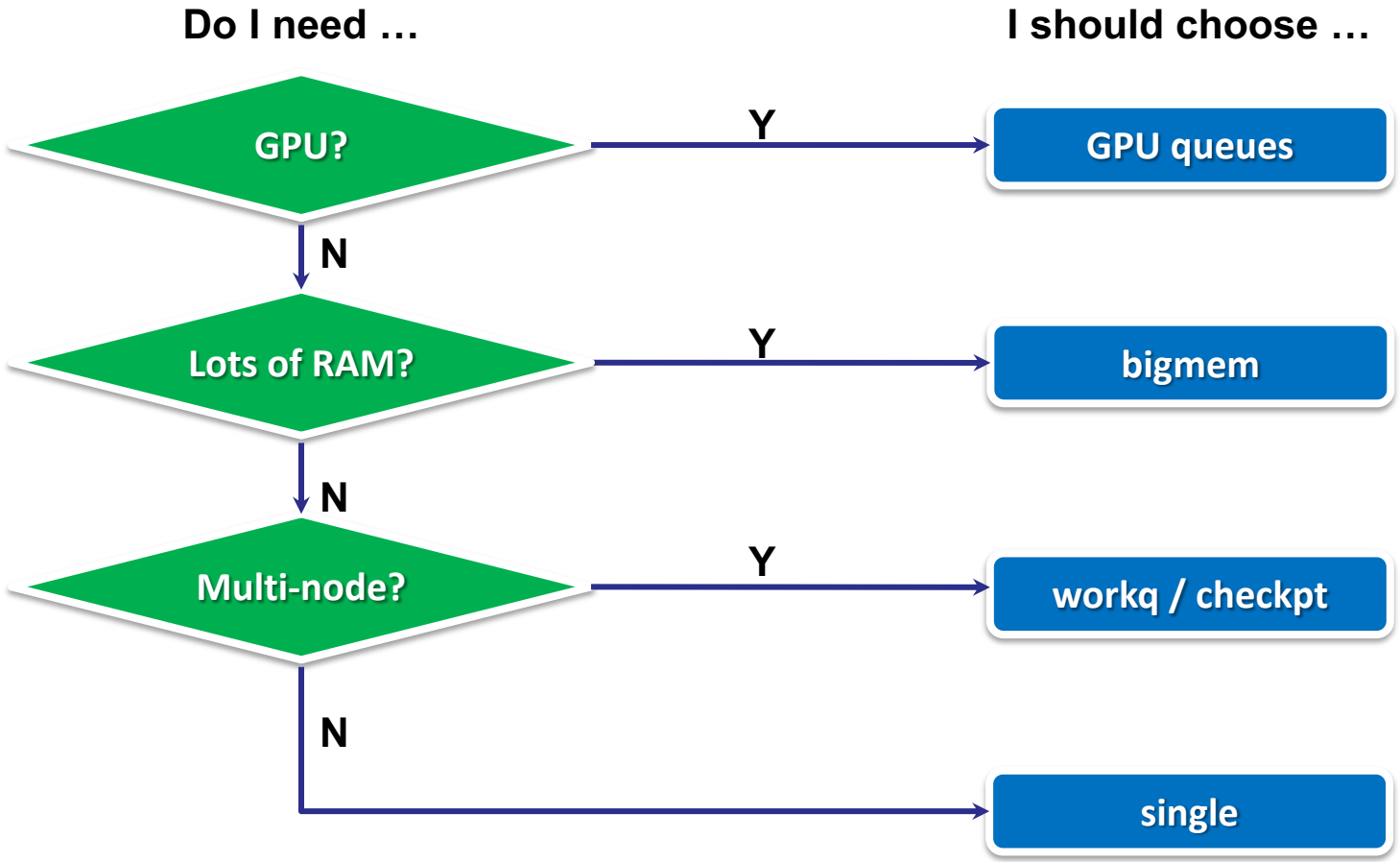
4) Job queues and partitions

LSU HPC Cluster	Partition	Cores per node (ppn)	Max running jobs	Max nodes per user
SuperMIC (smic.hpc.lsu.edu)	workq	20	45 (*global*)	86
	checkpt			
	single	1 ~ 20		
	gpu2	18,36		2
	bigmem	28		3
DeepBayou (db1.hpc.lsu.edu)	gpu2	24,48	-	8
	gpu4	12,24,36,48		2
SuperMike-III (mike.hpc.lsu.edu)	workq	64	32 (*global*)	84
	checkpt			
	single	1 ~ 64		
	gpu4	16,32,48,64		4
	bigmem	64		4

4) Job queues and partitions

LONI HPC Clusters	Partitions	Cores per node (n)	Max running jobs	Max nodes per user
QBC (QB3) (qbc.loni.org)	workq	48	32 (*global*)	48
	checkpt			
	single	1 ~ 48		
	gpu2	48		4
	bigmem	48		2
QBD (QB4) (qbd.loni.org)	workq	64	32 (*global*)	84
	checkpt			
	single	1 ~ 64		
	gpu2	32,64		4
	gpu4	16,32,48,64		4
	bigmem	64		5

4) Job queues and partitions



4) Job queues and partitions

```
[username@hostname ~]$ scontrol show partitions

PartitionName=workq
   AllowGroups=ALL AllowAccounts=ALL AllowQos=ALL
   AllocNodes=ALL Default=NO QoS=normal
   DefaultTime=12:00:00 DisableRootJobs=YES ExclusiveUser=NO ExclusiveTopo=NO GraceTime=0 Hidden=NO
   MaxNodes=UNLIMITED MaxTime=3-00:00:00 MinNodes=0 LLN=NO MaxCPUsPerNode=64 MaxCPUsPerSocket=UNLIMITED
   Nodes=qbd[001-480]
   PriorityJobFactor=1 PriorityTier=1 RootOnly=NO ReqResv=NO OverSubscribe=EXCLUSIVE
   OverTimeLimit=NONE PreemptMode=OFF
   State=UP TotalCPUs=30720 TotalNodes=480 SelectTypeParameters=NONE
   JobDefaults=(null)
   DefMemPerNode=UNLIMITED MaxMemPerNode=UNLIMITED
   TRES=cpu=30720,mem=120000G,node=480,billing=30720
   TRESBillingWeights=CPU=1

PartitionName=bigmem
   AllowGroups=ALL AllowAccounts=ALL AllowQos=ALL
   AllocNodes=ALL Default=NO QoS=normal
   DefaultTime=12:00:00 DisableRootJobs=YES ExclusiveUser=NO ExclusiveTopo=NO GraceTime=0 Hidden=NO
   MaxNodes=UNLIMITED MaxTime=3-00:00:00 MinNodes=0 LLN=NO MaxCPUsPerNode=64 MaxCPUsPerSocket=UNLIMITED
   Nodes=qbd[481-485]
   PriorityJobFactor=1 PriorityTier=1 RootOnly=NO ReqResv=NO OverSubscribe=EXCLUSIVE
   OverTimeLimit=NONE PreemptMode=OFF
   State=UP TotalCPUs=320 TotalNodes=5 SelectTypeParameters=NONE
   JobDefaults=(null)
   DefMemPerNode=UNLIMITED MaxMemPerNode=UNLIMITED
   TRES=cpu=320,mem=10315000M,node=5,billing=320
   TRESBillingWeights=CPU=1
```

4) Job queues and partitions

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	**gpu4** (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)

Summary

- **HPC User Environment 2**

1. Basic concepts
   1) Review of HPC User Environment 1
   2) Job and job scheduler
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queues and partitions
3. Submitting my job
   1) Interactive job
   2) Batch job
4. Managing my job
   1) Useful commands
   2) Monitoring job health

It is time for a break.

See you in 5 minutes!

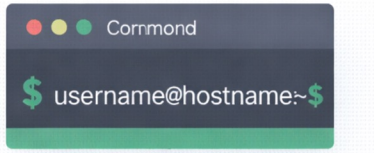
Outlines

- **HPC User Environment 2**

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

3. Submitting my job

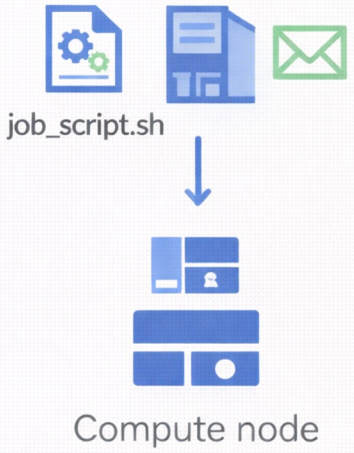
Interactive Jobs



- Runs in **terminal** (just like using a local machine)
- Can **interact** with the job while running, for testing, debugging, and short runs.

for testing, debugging, and short runs

Batch Jobs



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

3. Submitting my job

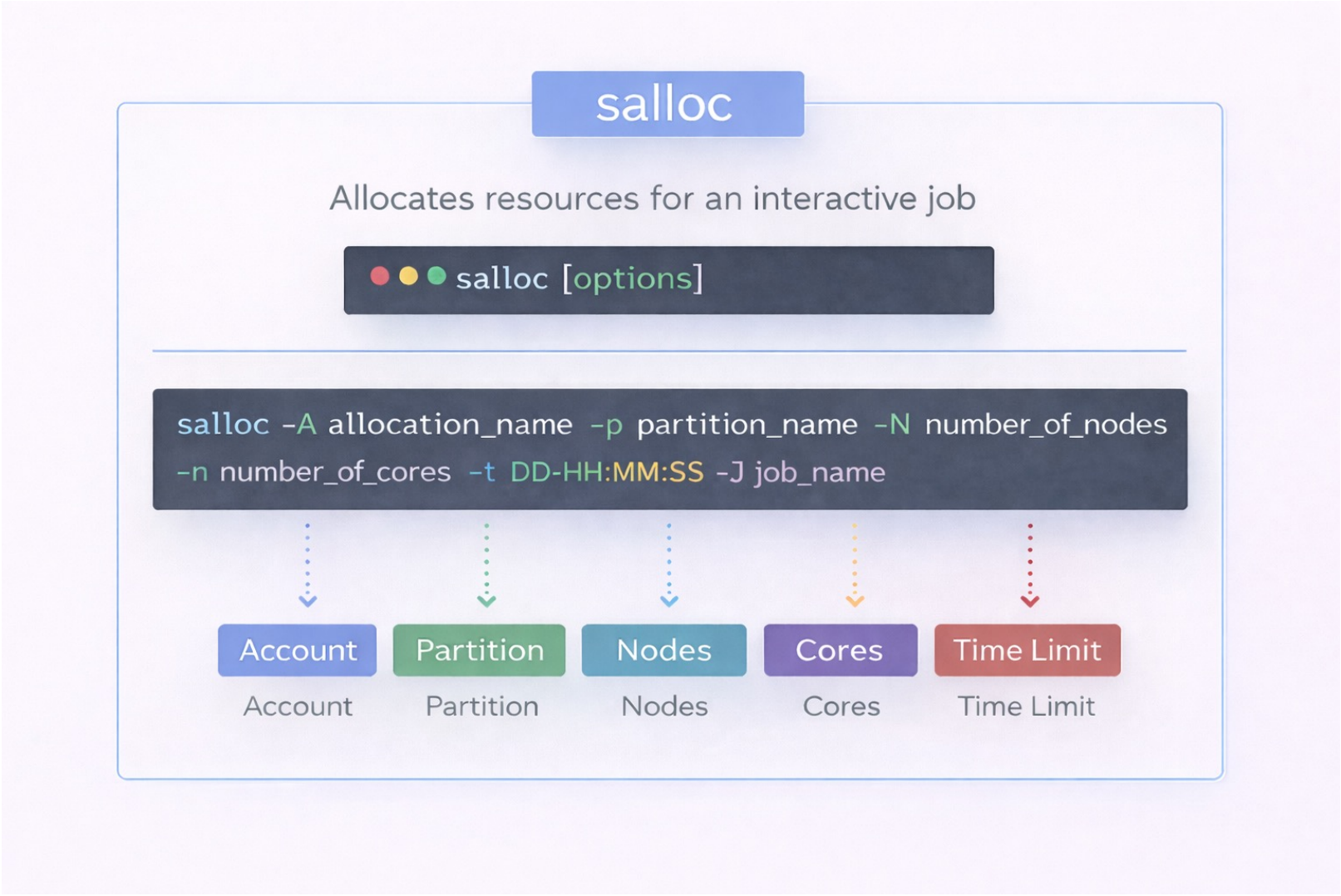
	An interactive job	A batch job
Pros	• Can interact and monitor with job in real time	• Submit and leave it • Repeatable for complicated jobs
Cons	• Waiting for human intervention is the opposite of “high performance”	• Cannot edit or interact with job while running
Ideal for	• Debugging, testing, and compiling	• Production

Outlines

- **HPC User Environment 2**

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

1) Interactive job



1) Interactive job

salloc [options]

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

Allocation name

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

Job duration

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

Job queue

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

Number of nodes

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

Total number of cores

1) Interactive job

```
[username@host ~]$ salloc -A loni_loniadmin1 -p workq -N 1 -n64 --time=01:00:00
      salloc: Pending job allocation 51814
      salloc: lua: Submitted job 51814
      salloc: job 51814 queued and waiting for resources
      salloc: job 51814 has been allocated resources
      salloc: Granted job allocation 51814
      salloc: Waiting for resource configuration
      salloc: Nodes qbd085 are ready for job
```

1) Interactive job

```
[username@host ~]$ salloc -A loni_loniadmin1 -p workq -N 1 -n64 --time=01:00:00
salloc: Pending job allocation 51814
salloc: lua: Submitted job 51814
salloc: job 51814 queued and waiting for resources
salloc: job 51814 has been allocated resources
salloc: Granted job allocation 51814
salloc: Waiting for resource configuration
salloc: Nodes qbd085 are ready for job
```

1) Interactive job

```
[username@qbd1 ~]$ salloc -A loni_loniadmin1 -p workq -N1 -n 64 --time=12:00:00
```

```
[username@qbd085 ~]$
```

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

1) Interactive job

```
[username@qbd1 Tests]$ salloc –A loni_loniadmin1 –p workq –N1 –n64 –t 12:00:00


salloc: Pending job allocation 51860
salloc: lua: Submitted job 51860
salloc: job 51860 queued and waiting for resources
salloc: job 51860 has been allocated resources
salloc: Granted job allocation 51860
salloc: Nodes qbd083 are ready for job
[username@qbd083 Tests]$
```

Job starts in **where the job was submitted**

1) Interactive job

```
[username@qbd1 Tests]$ salloc –A loni_loniadmin1 –p workq –N1 –n64 –t 12:00:00


salloc: Pending job allocation 51860
salloc: lua: Submitted job 51860
salloc: job 51860 queued and waiting for resources
salloc: job 51860 has been allocated resources
salloc: Granted job allocation 51860
salloc: Nodes qbd083 are ready for job
[username@qbd083 Tests]$
```

Once a job starts, **type and run commands** as you normally do.

1) Interactive job

salloc

-A <allocation name>

-t <D-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

Number of total cores

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

-c < number of cores per process>

<# of TOTAL cores>

= <n> * <c>

b) Starting an MPI / OpenMP hybrid job (For those who use it)

```
salloc
-A <allocation name>
-t   <DD-HH:MM:SS>
-p   workq
-N   2
-n   32
-c   4
```

[QB-4 / workq]

- **64** / node
- **2** nodes → **128** cores
- **32** processes
- **4** cores / process

1) Interactive job

```
salloc
-A <allocation name>
-t   <DD-HH:MM:SS>
-p   workq
-N   2
-n   32
-c   4
```

[QB-4 / workq]

- **64** / node
- **2** nodes → **128** cores
- **32** processes
- **4** cores / process

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p workq

-N 2

-n 32

-c 4

[QB-4 / workq]

- **64** / node
- **2** nodes → **128** cores
- **32** processes
- **4** cores / process

1) Interactive job

salloc

-A <allocation name>

-t <DD-HH:MM:SS>

-p workq

-N 2

-n 32

-c 4

[QB-4 / workq]

- **64** / node
- **2** nodes → **128** cores
- **32** processes
- **4** cores / process

1) Interactive job

salloc

-A <allocation name>

-t <D-HH:MM:SS>

-p <queue name>

-N < number of nodes>

-n <number of tasks/cores>

- - gres=gpu:1 < number of GPUs per node>

Number of GPUs **per node**

1) Interactive job

```
salloc
-A <allocation name>
-t   <D-HH:MM:SS>
-p   gpu 4
-N   1
-n   16
- - gres=gpu:1
```

[QB-4 / gpu4]

- **64** cores, **4** GPUs → **16** cores / GPU
- **1** nodes (only need ¼)
- **16** cores (¼ of total)
- **1** GPU (¼ of total)

1) Interactive job

```
salloc
-A <allocation name>
-t   <D-HH:MM:SS>
-p  gpu 4
-N  1
-n  16
- - gres=gpu:1
```

[QB-4 / gpu4]

- **64** cores, **4** GPUs → **16** cores / GPU
- **1** nodes (only need ¼)
- **16** cores (¼ of total)
- **1** GPU (¼ of total)

1) Interactive job

```
salloc
-A <allocation name>
-t   <D-HH:MM:SS>
-p  gpu 4
-N  1
-n  16
- - gres=gpu:1
```

[QB-4 / gpu4]

- **64** cores, **4** GPUs → **16** cores / GPU
- **1** nodes (only need ¼)
- **16** cores (¼ of total)
- **1** GPU (¼ of total)

1) Interactive job

```
salloc
-A <allocation name>
-t   <D-HH:MM:SS>
-p  gpu 4
-N  1
-n  16
- - gres=gpu:1
```

[QB-4 / gpu4]

- **64** cores, **4** GPUs → **16** cores / GPU
- **1** nodes (only need ¼)
- **16** cores (¼ of total)
- **1** GPU (¼ of total)

1) Interactive job

Flag		Description	
`--x11`		Enable x11 forwarding for GUI (exclusive to interactive job)	
`-J`		Job name	
`--mail-type`	`FAIL`	Send email when …	Job aborts / fails
	`BEGIN`		Job begins
	`END`		Job ends
`--mail-user`		Email address (will check against registered institutional email)	

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

1) Interactive job

```
[username@hostname ~]$ salloc -A loni_loniadmin1 -p workq -N1 -n 64 -t 12:00:00

[username@hostname ~]$ salloc -A loni_loniadmin1 -p workq -t 1-00:00:00

[username@hostname ~]$ salloc --account=loni_loniadmin1 --partition=workq
--nodes=1 --ntasks=64 --time=1-00:00:00 --job-name=training
--mail-user=user@mail.address --mail-type=BEGIN,END
```

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

1) Interactive job

The default partition is single

1) Interactive job

Serial (Single-thread)	Parallel (MPI)

• **Run commands as you normally do**
 	
`$ <Executable> [options]`
	• **Method 1 (Recommended)**
	
	`$ srun -N[..] -n[..] -c[..] <mpi_executable> [options]`
	
	• **Method 2**
	
	`$ module load <desired MPI>`
	`$ export OMP_NUM_THREADS=[..]`
	`$ mpirun -np [..] <mpi_executable> [options]`

1) Interactive job

Variable	Description
$SLURM_JOBID	Job ID
$SLURM_SUBMIT_DIR	Job submit directory
$SLURM_JOB_NODELIST	A temp file, contains a list of allocated nodes' names (useful for MPI)
$SLURM_NNODES	Number of allocated nodes
$SLURM_NTASKS	Number of processes (tasks)
…	

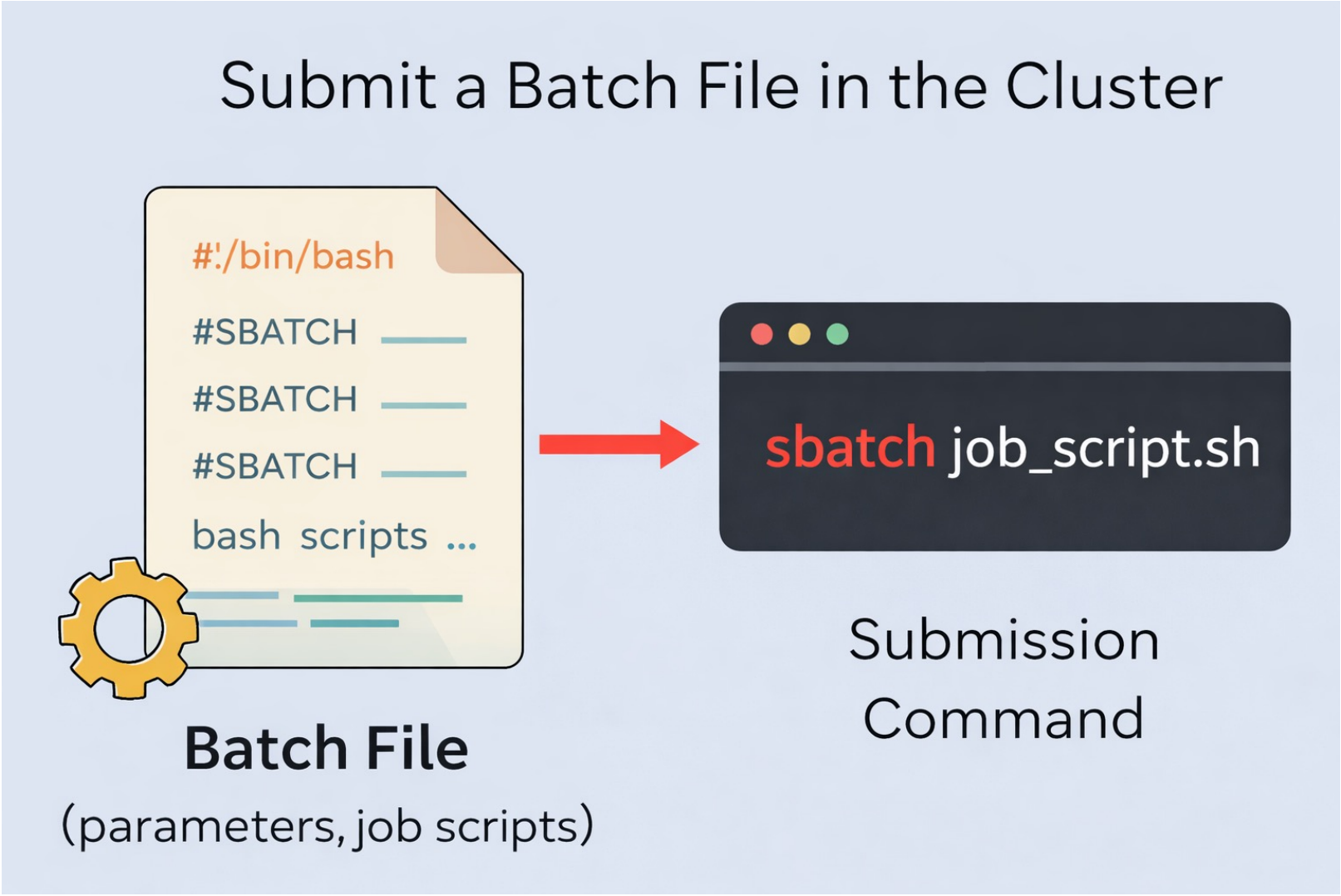
[1] http://www.hpc.lsu.edu/docs/slurm.php

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) Batch job


Submit a Batch File in the Cluster
#!/bin/bash
#SBATCH
#SBATCH
#SBATCH
bash scripts ...
Batch File
(parameters, job scripts)
sbatch job_script.sh
Submission
Command

2) Batch job

```
#!/bin/bash
#SBATCH -A allocation_name
#SBATCH -p partition_name
#SBATCH -N 1
#SBATCH -n 64
#SBATCH -o output_file.out
#SBATCH -e output_file.err
#SBATCH --mail-user=username@institution.edu
#SBATCH --mail-type=END,FAIL
export OMP_NUM_THREADS=1
cd $SLURM_SUBMIT_DIR
srun -N 1 -n 64 -c 1  my_mpi_program
```

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END
```

[Header]

Job parameters

[Body]

Commands to run after job starts

```
module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END
```

[Header]
Job parameters

```
module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

Info for interpreter

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

GPU queue

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tas
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

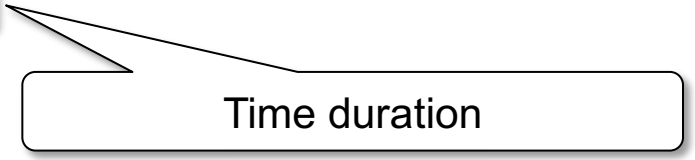
Number of nodes (N)

Number of processes (n)

and the number of cores per process (c)

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```



2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation_name
#SBATCH -J job_name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

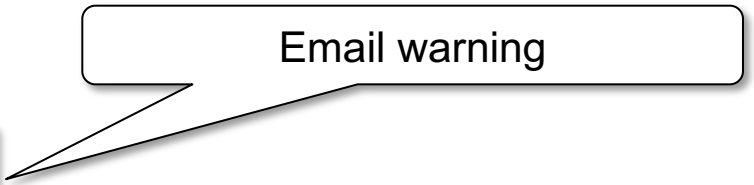
2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

Standard error and output

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END

module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```



2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END

module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

Loading some modules to the environment

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-m    2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

Setting up the number of OpenMP threads for the job

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation name
#SBATCH -J job name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

Switching to the submit directory

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation-name
#SBATCH -J job-name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

Executing the command lmp_mpi

2) Batch job

```
#!/bin/bash
#SBATCH -p gpu4
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32              # Total number of MPI tasks
#SBATCH -c 1               # One CPU core per task
#SBATCH -t HH:MM:SS
#SBATCH -A allocation-name
#SBATCH -J job-name
#SBATCH -o standard-output.%j.out
#SBATCH -e standard-error.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END


module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.

export OMP_NUM_THREADS=1 # One core per task

echo "Current working directory: $SLURM_SUBMIT_DIR"
cd   $SLURM_SUBMIT_DIR

time srun -N 1 -n 32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton off -in lj.in > lj.out
```

Leave one space after the command line to avoid error

2) Batch job

Flag		Description	
`-o`		Standard output file (exclusive to batch job)	
`-e`		Standard error file (exclusive to batch job)	
`-J`		Job name	
`--dependency=afterok:[jobid]`		Dependent job (starts after another job finishes)	
`--mail-type`	`FAIL`	Send email when …	Job aborts / fails
	`BEGIN`		Job begins
	`END`		Job ends
`--mail-user`		Email address (will check against registered institutional email)	

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

2) Batch job

sbatch batch_script.sh

Outlines

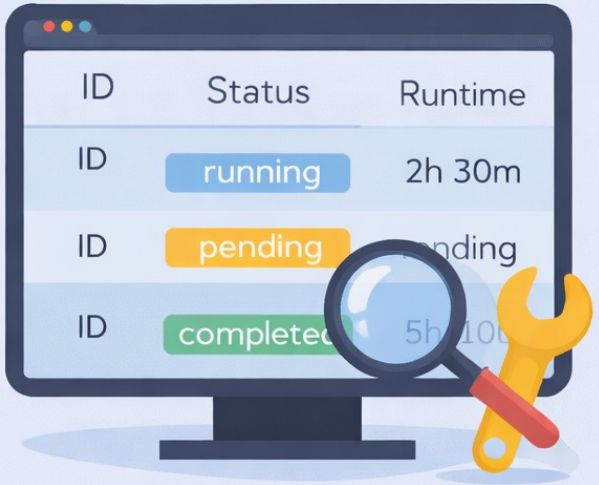
- **HPC User Environment 2**

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

Running Jobs on HPC ≠ "Submit and done"

"Submit and done"

≠



Monitoring & Managing Jobs

Monitoring and managing jobs are part of the work

Outlines

- **HPC User Environment 2**

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

1) Useful commands

Command		Description
squeue		List all jobs
	-j <Job ID>	List the job of specific ID
	-u <Username>	List all jobs belong to a specific user
	-p <Queue name>	List all jobs in a particular queue
	--start	Estimated start time of queuing jobs
scontrol show job <Job ID>		Show job details
scancel <Job ID>		Cancel <Job ID>

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

Outlines

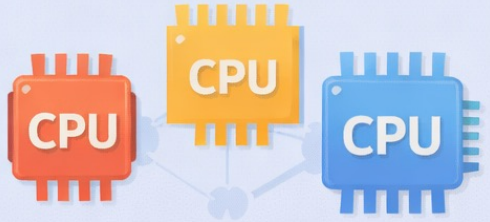
- **HPC User Environment 2**

1. Basic concepts
   1) Review of HPC User Environment 1
   2) Job and job schedulers
2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes and cores
   4) Job queues and partitions
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

Cores Utilization

≠



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=[illegible] 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=603% 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)

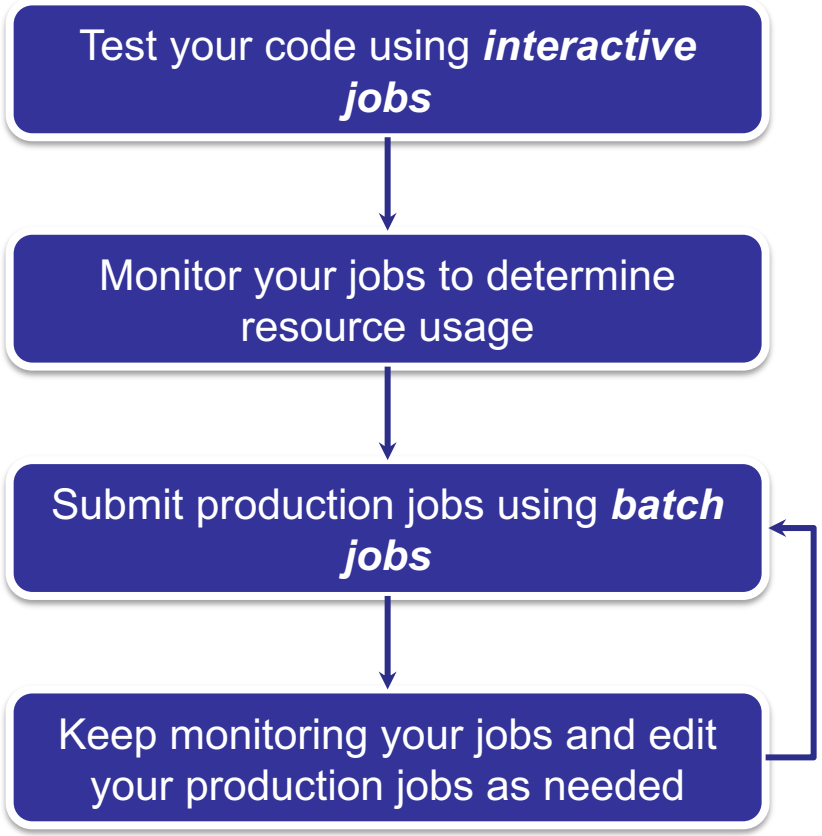
```
(base) [jasonli3@qbc193 ~]$ nvidia-smi
Wed Feb  1 02:38:32 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+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

2) Monitoring job health

Common issues	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


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

Outlines

- **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job and 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 and cores
   4) Job queues and partitions
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**

Auxiliary information

```
salloc -A loni_loniadmin1 -p single --nodelist=qbd042 -t 1-00:00:00

salloc -A loni_loniadmin1 -p single --nodelist=qbd042 -N1 -n64 -t 1-00:00:00

salloc -A loni_loniadmin1 -p workq --nodelist=qbd[046-047] -N2 -n128 -t 1-00:00:00
```

Auxiliary information

Running LAMMPS jobs using 1 GPU

```
#!/bin/bash
#SBATCH --partition=gpu
#SBATCH --gres=gpu:1
#SBATCH --nodes=1
#SBATCH --ntasks=16
#SBATCH --cpus-per-task=1
#SBATCH --time=D-HH:MM:SS
#SBATCH --account=allocation-name
#SBATCH --job-name=test
#SBATCH --output=lammps.%j.out
#SBATCH --error=lammps.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END

module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

echo $SLURM_JOBID
echo $SLURM_NTASKS
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

echo $SLURM_SUBMIT_DIR
cd   $SLURM_SUBMIT_DIR

time srun -N1 -n16 lmp_mpi -sf gpu -pk gpu 1 neigh yes newton
off -in lj.in > lj.out
```

Running LAMMPS jobs using 2 GPUs

```
#!/bin/bash
#SBATCH -p gpu
#SBATCH --gres=gpu:2
#SBATCH -N 1
#SBATCH -n 32
#SBATCH -c 1
#SBATCH -t D-HH:MM:SS
#SBATCH -A hpc_allocation
#SBATCH -J test
#SBATCH -o lammps.%j.out
#SBATCH -e lammps.%j.err
#SBATCH --mail-user=your@email.address
#SBATCH --mail-type=BEGIN,END

module purge
module load lammps/02Aug2023/intel-2021.5.0-cuda-11.6.0-intel-mpi-2021.5.1

echo $SLURM_JOBID
echo $SLURM_NTASKS
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

echo $SLURM_SUBMIT_DIR
cd   $SLURM_SUBMIT_DIR

time srun -N1 -n32 lmp_mpi -sf gpu -pk gpu 2 neigh yes newton
off -in lj.in > lj.out
```

Auxiliary information

SLURM_JOB_ID	- This ID can be used to track the job status and retrieve logs
SLURM_JOB_NAME	- The name of the job as specified by the user in command or script.
SLURM_NTASKS	- **The total number of tasks (processes) allocated for the job.**
SLURM_CPUS_PER_TASK	- **The number of CPUs allocated per task.**
SLURM_NODES	- **A list of nodes allocated for the job.**
SLURM_NODELIST	- A comma-separated list of nodes that are allocated for the job.
SLURM_SUBMIT_DIR	- **The directory from which the job was submitted.**
SLURM_ARRAY_JOB_ID	- The job ID of the array job if the job is part of a job array.
SLURM_ARRAY_TASK_ID	- The ID of the specific task in an array job.
SLURM_JOB_NODELIST	- List of nodes allocated for the job, specifically formatted for easier parsing.
SLURM_JOB_DEPENDENCY.	- Indicating which jobs must complete before this job can start.
SLURM_MEM_PER_CPU	- The amount of memory allocated per CPU core for the job.

Standard error and output

command > output.log 2>&1

In Unix-like operating systems, processes have three standard file descriptors:

1. Standard Input (stdin): File descriptor 0 (used for input).
2. Standard Output (stdout): **File descriptor 1** (used for normal output).
3. Standard Error (stderr): **File descriptor 2** (used for error messages).