HPC User Environment 2

Jason Li
HPC User Services
LSU HPC / LONI
sys-help@loni.org
Louisiana State University
Baton Rouge
Sep 20, 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. Preparing my job
  3. Submitting my job
  4. Managing my jobs
Outlines

- HPC User Environment 2
  1. Basic concepts
  2. Preparing my job
  3. Submitting my job
  4. Managing my jobs
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
• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
Outlines

• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
Two things needed to run jobs on our clusters:

1) Account

2) Allocation
1) Previously on HPC User Environment 1…
Run my code on all the resources you have, however long it takes.

Yes, my master!

sudo!
yum!
apt-get!

...
1) Previously on HPC User Environment 1…

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

sudo!
yum!
ant-get!
…

You are not my boss, buddy!
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.
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
a) What’s a “job”? 

– A user's request to use a number of nodes/cores for a certain amount of time on a cluster.

– Calculation MUST be done via jobs (NO heavy calculation on head nodes!!)

– SUs deducted from allocations based on actual usage of each job.

• Example:
  – My allocation: 50,000 SU
  – Running a job: 24 core * 10 hours = 240 SU
  – Balance: 49,760 SU
2) Job & Job scheduler

b) What’s a “job scheduler”?

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.
b) What’s a “job scheduler”?

![Diagram of a job scheduler]

Node001, Node002, Node003, ..., Node004 are nodes that connect to the job scheduler. The job scheduler manages and schedules jobs such as Job1, Job2, Job3, and so on.
b) What’s a “job scheduler”?  

i. Decides which job runs when and where

You! Go to Node 002!

I need 1 node w/ 24 cores for 10 hours
b) What’s a “job scheduler”?

i. Decides which job runs when and where

ii. Enforces job policies

I need to reserve 100 nodes exclusively for our group indefinitely
b) What’s a “job scheduler”? 

<table>
<thead>
<tr>
<th>Job scheduler’s responsibilities</th>
<th>Your responsibilities</th>
</tr>
</thead>
</table>
| • Decides which job runs when and where  
• Enforces job policies | • Decide a job’s size and duration  
• Understand the job queuing system and policies  
• Submit/monitor/cancel jobs  
• Diagnose job health |
b) What’s a “job scheduler”? 

i) PBS
2) Job & Job scheduler

b) What’s a “job scheduler”?

i) PBS

ii) Slurm
### 2) Job & Job scheduler

b) What’s a “job scheduler”?

<table>
<thead>
<tr>
<th></th>
<th>LSU HPC</th>
<th>LONI</th>
</tr>
</thead>
<tbody>
<tr>
<td>i) PBS</td>
<td>SMIC</td>
<td>QB2</td>
</tr>
<tr>
<td>ii) Slurm</td>
<td>Deep Bayou SuperMike III</td>
<td>QB3</td>
</tr>
</tbody>
</table>
Outlines

• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
Outlines

• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
1) Basic principles

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

Large enough … Small enough …
1) Basic principles

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

<table>
<thead>
<tr>
<th>Large enough …</th>
<th>Small enough …</th>
</tr>
</thead>
</table>
| • To successfully complete your job | • To ensure quick turnaround  
  • Not to waste resources for other users |
HPC User Environment 2

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

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

• **FAQ**

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

• Back to basic principles…

<table>
<thead>
<tr>
<th>Large enough …</th>
<th>Small enough …</th>
</tr>
</thead>
</table>
| • To successfully complete your job | • To ensure quick turnaround  
|  | • Not to waste resources for other users |
Outlines

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
3) Number of nodes & cores

<table>
<thead>
<tr>
<th>SuperMIC</th>
<th>Deep Bayou</th>
<th>SuperMike III</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hostname</strong></td>
<td>smic.hpc.lsu.edu</td>
<td>db1.lsu.edu</td>
</tr>
<tr>
<td><strong>Peak Performance/TFlops</strong></td>
<td>925</td>
<td>257</td>
</tr>
<tr>
<td><strong>Compute nodes</strong></td>
<td>360</td>
<td>13</td>
</tr>
<tr>
<td><strong>Processor/node</strong></td>
<td>2 10-core</td>
<td>2 24-core</td>
</tr>
<tr>
<td><strong>Processor Speed</strong></td>
<td>2.8 GHz</td>
<td>2.4 GHz</td>
</tr>
<tr>
<td><strong>Processor Type</strong></td>
<td>Intel Xeon 64bit</td>
<td>Intel Cascade Lake Xeon 64bit</td>
</tr>
<tr>
<td><strong>Nodes with Accelerators</strong></td>
<td>360</td>
<td>13</td>
</tr>
<tr>
<td><strong>Accelerator Type</strong></td>
<td>Xeon Phi 7120P</td>
<td>2 x NVIDIA Volta V100S</td>
</tr>
<tr>
<td><strong>OS</strong></td>
<td>RHEL v6</td>
<td>RHEL v7</td>
</tr>
<tr>
<td><strong>Vendor</strong></td>
<td></td>
<td>Dell</td>
</tr>
<tr>
<td><strong>Memory per node</strong></td>
<td>64 GB</td>
<td>192 GB</td>
</tr>
<tr>
<td><strong>Detailed Cluster Description</strong></td>
<td>User Guide</td>
<td>User Guide</td>
</tr>
<tr>
<td></td>
<td>Available Software</td>
<td>Available Software</td>
</tr>
<tr>
<td><strong>Hostname</strong></td>
<td>mike.hpc.lsu.edu</td>
<td></td>
</tr>
<tr>
<td><strong>Peak Performance/TFlops</strong></td>
<td>1,285</td>
<td></td>
</tr>
<tr>
<td><strong>Compute nodes</strong></td>
<td></td>
<td>183</td>
</tr>
<tr>
<td><strong>Processor/node</strong></td>
<td>2 32-core</td>
<td></td>
</tr>
<tr>
<td><strong>Processor Speed</strong></td>
<td>2.6 GHz</td>
<td></td>
</tr>
<tr>
<td><strong>Processor Type</strong></td>
<td>Intel Xeon Ice Lake</td>
<td></td>
</tr>
<tr>
<td><strong>Nodes with Accelerators</strong></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td><strong>Accelerator Type</strong></td>
<td>4 NVIDIA A100</td>
<td></td>
</tr>
<tr>
<td><strong>OS</strong></td>
<td>RHEL v8</td>
<td></td>
</tr>
<tr>
<td><strong>Vendor</strong></td>
<td>Dell</td>
<td></td>
</tr>
<tr>
<td><strong>Memory per node</strong></td>
<td>256/2048 GB</td>
<td></td>
</tr>
<tr>
<td><strong>Detailed Cluster Description</strong></td>
<td>User Guide</td>
<td>User Guide</td>
</tr>
<tr>
<td></td>
<td>Available Software</td>
<td>Available Software</td>
</tr>
</tbody>
</table>
3) Number of nodes & cores

• When submitting your job…
  – Must tell job scheduler the number of nodes & cores you need
### FAQ

<table>
<thead>
<tr>
<th>Q</th>
<th>A</th>
</tr>
</thead>
</table>
| **My code runs slow. Can I request more nodes / cores to make it faster?** | **Not quite! Your code most likely is **NOT** using multiple nodes / cores, if:**
- You *do not know* if it is using multiple nodes / cores
- You *did not tell* it to use multiple nodes / cores
- You are *not familiar* with names like “MPI” / “OpenMP”

**Underutilization** is THE most common warning received on our clusters |
| **How many nodes / cores should I request?** | **In short: We can't answer that**
- Each code / job is different. You must test to determine |
### 3) Number of nodes & cores

- Back to basic principles…

<table>
<thead>
<tr>
<th>Large enough …</th>
<th>Small enough …</th>
</tr>
</thead>
<tbody>
<tr>
<td>- To successfully complete your job</td>
<td>- To ensure quick turnaround</td>
</tr>
<tr>
<td></td>
<td>- Not to waste resources for other users</td>
</tr>
</tbody>
</table>
Outlines

• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
4) Job queues

Node001  Node002  Node003  Node004

Job scheduler

Job1
Job2
Job3

...
4) Job queues

- Node001
- Node002
- Node003
- Node004

Job scheduler

- Job1
- Job2
- Job3

...
4) Job queues

Job scheduler

Node001
Node002
Node003
Node004

...
4) Job queues

- Node001
- Node002
- Node003
- Node004

Job scheduler

- single
  - Job1
  - Job2
  - Job3

- workq
  - Job1
  - Job2
  - Job3

- bigmem
  - Job1
  - Job2
  - Job3

- gpu
  - Job1
  - Job2
  - Job3

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

a) Definition

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

a) Definition
4) Job queues

b) Available queues

i. **workq / checkpt**

<table>
<thead>
<tr>
<th>Description</th>
<th>Nodes</th>
<th>Cores</th>
<th>Memory</th>
<th>Max duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• General purposes</td>
<td>• One or multiple</td>
<td>• All cores on the node(s)</td>
<td>• All memory on the node(s)</td>
<td>• 72 hours (3 days)</td>
</tr>
<tr>
<td>• Most likely your <strong>default</strong> queue</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Difference: <strong>non-preemptable</strong> (workq) vs. <strong>preemptable</strong> (checkpt)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Names                           |       |         |              |              |
| • All clusters: **workq / checkpt** |       |         |              |              |
## 4) Job queues

### b) Available queues

#### ii. single

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

---

**[SuperMike 3]**

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

### b) Available queues

#### iii. bigmem

<table>
<thead>
<tr>
<th>Description</th>
<th>• Your job needs <strong>large memory</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Names</strong></td>
<td>• All clusters: <strong>bigmem</strong></td>
</tr>
<tr>
<td><strong>Resource availability</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Nodes</strong></td>
<td>• <strong>One</strong> or <strong>multiple</strong></td>
</tr>
<tr>
<td><strong>Cores</strong></td>
<td>• <strong>All</strong> cores on the node</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>• <strong>All</strong> memory on the node</td>
</tr>
<tr>
<td><strong>Max duration</strong></td>
<td>• <strong>72</strong> hours (3 days)</td>
</tr>
</tbody>
</table>
### b) Available queues

#### iv. GPU

<table>
<thead>
<tr>
<th>Description</th>
<th>Names</th>
<th>Resource availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Your job needs GPU</td>
<td>• SMIC:</td>
<td>• Nodes: One or multiple</td>
</tr>
<tr>
<td></td>
<td>• Deep Bayou (*): v100</td>
<td>• Cores: All cores on the node</td>
</tr>
<tr>
<td></td>
<td>• SuperMike 3 (*): GPU</td>
<td>• Memory: All memory on the node</td>
</tr>
<tr>
<td></td>
<td>• QB3: v100</td>
<td>• GPU: All GPU devices on the node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Max duration: 72 hours (3 days)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Deep Bayou (*): gpu, nvlink</td>
<td>• Nodes: Portion of one node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cores: Portion of one node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Memory: Portion of one node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GPU: 1 ~ all GPU devices</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• SuperMike 3 (*): gpu</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 4. Managing my jobs

2. Preparing my job

3. Submitting my job

4. Managing my jobs

#### 4) Job queues

c) Queues by clusters (LSU HPC)

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Queue</th>
<th>Cores per node (ppn)</th>
<th>Max running jobs</th>
<th>Max nodes per user</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SuperMIC</strong></td>
<td>workq</td>
<td>20</td>
<td>45 (global)</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>checkpt</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>single</td>
<td>1,2,4,6,8,16</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>v100</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bigmem</td>
<td>28</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DeepBayou</strong></td>
<td>gpu</td>
<td>24,48</td>
<td>4 (global)</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>nvlink</td>
<td>12,24,36,48</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td><strong>SuperMike3</strong></td>
<td>workq</td>
<td>64</td>
<td>32 (global)</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>checkpt</td>
<td>64</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>single</td>
<td>1 ~ 64</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>gpu</td>
<td>16,32,48,64</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>bigmem</td>
<td>64</td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>
### c) Queues by clusters (LONI)

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Queue</th>
<th>Cores per node (ppn)</th>
<th>Max running jobs</th>
<th>Max nodes per user</th>
</tr>
</thead>
<tbody>
<tr>
<td>QB-2</td>
<td>workq</td>
<td>20</td>
<td>32 (global)</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>checkpoint</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>single</td>
<td>1,2,4,6,8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bigmem</td>
<td>48</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QB-3</td>
<td>workq</td>
<td>48</td>
<td>32 (global)</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>checkpoint</td>
<td>48</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>single</td>
<td>1 ~ 48</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>gpu</td>
<td>48</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>bigmem</td>
<td>48</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 4) Job queues

**d) Choose your queue**

<table>
<thead>
<tr>
<th>Large enough …</th>
<th>Small enough …</th>
</tr>
</thead>
</table>
| • To successfully complete your job | • To ensure quick turnaround  
| | • Not to waste resources for other users |
4) Job queues

d) Choose your queue

- **GPU?**
  - Y → **GPU queues**
  - N

- **Lots of RAM?**
  - Y → **bigmem**
  - N

- **Multi-node?**
  - Y → **workq / checkpoint**
  - N

- **Entire node?**
  - Y → **single**
  - N
d) Choose your queue

<table>
<thead>
<tr>
<th>My job …</th>
<th>Queue choice? (include number of nodes / cores)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>• SMIC</strong>&lt;br&gt;• MPI code, needs 100 CPU cores&lt;br&gt;  - Hint: SMIC has 20 cores / node</td>
<td><strong>workq / checkpt</strong>&lt;br&gt;  (5 nodes, 20 cores per node)</td>
</tr>
<tr>
<td><strong>• SuperMike 3</strong>&lt;br&gt;• Uses 2 GPUs to train a neural network&lt;br&gt;  - Hint: SuperMike 3 has 64 cores / node, 4 GPUs / node → 16 cores / GPU</td>
<td><strong>gpu</strong>&lt;br&gt;  (1 node, 32 cores per node)</td>
</tr>
<tr>
<td><strong>• QB-3</strong>&lt;br&gt;• Single-core serial code&lt;br&gt;• Needs to store and process 30 GB data in RAM&lt;br&gt;  - Hint: QB-3 has 192 GB RAM / node, 4 GB RAM / core</td>
<td><strong>single</strong>&lt;br&gt;  (1 node, 8 cores per node)</td>
</tr>
</tbody>
</table>
4) Job queues

e) Useful commands to check queues

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

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


e) Useful commands to check queues

ii. `qfree`: Free nodes in each queue

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

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

```
(base) [jasonli3@mike2 ~]$ sinfo
PARTITION  AVAIL   TIMELIMIT  NODES  STATE      NODELIST
single*   up 7-00:00:00  2       inval mikel[035,138]
single*   up 7-00:00:00  1       comp mikel144
single*   up 7-00:00:00  58      alloc mikel[008-026,031-034,036-044,046-050,141-143,148-162,167-169]
single    up 7-00:00:00  108     idle mikel[001-007,027-030,045,051-137,139,145-146,163-166,170-171]
single*   up 7-00:00:00  2       down mikel[140,147]
checkpt   up 3-00:00:00  2       inval mikel[035,138]
checkpt   up 3-00:00:00  1       comp mikel144
checkpt   up 3-00:00:00  58      alloc mikel[008-026,031-034,036-044,046-050,141-143,148-162,167-169]
checkpt   up 3-00:00:00  108     idle mikel[001-007,027-030,045,051-137,139,145-146,163-166,170-171]
checkpt   up 3-00:00:00  2       down mikel[140,147]
workq    up 3-00:00:00  2       inval mikel[035,138]
workq    up 3-00:00:00  1       comp mikel144
workq    up 3-00:00:00  58      alloc mikel[008-026,031-034,036-044,046-050,141-143,148-162,167-169]
workq    up 3-00:00:00  108     idle mikel[001-007,027-030,045,051-137,139,145-146,163-166,170-171]
workq    up 3-00:00:00  2       down mikel[140,147]
bigmem   up 3-00:00:00  4       idle mikel[172-175]
gpu      up 3-00:00:00  8       idle mikel[176-183]
```
Summary

1. Basic concepts
   a) How job works on clusters
   b) Job scheduler and how it works

2. Preparing my job
   a) Basic principles
      • “large enough” and “small enough”
   b) Information you need to tell job scheduler:
      • Duration
      • Number of nodes & cores
      • Job queue
1) Have your terminal open and ready to connect to HPC
2) Download our testing code (π calculation) to your /home directory
   – http://www.hpc.lsu.edu/training/weekly-materials/Downloads/pi_Jason.tar.gz
   – Hint: use wget command
Outlines

- **HPC User Environment 2**

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

- Two types of jobs:

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

  2) Batch job
     - Submit to server and runs by itself, until finished or error
     - Cannot interact with the job while running
### 3. Submitting a job

- **Two types of jobs:**

<table>
<thead>
<tr>
<th>Pros</th>
<th>1) Interactive job</th>
<th>2) Batch job</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Can interact and monitor with job in real time</td>
<td>Submit and leave it</td>
</tr>
<tr>
<td>Cons</td>
<td>Waiting for human intervention is the opposite of “high performance”</td>
<td>Cannot edit or interact with job while running</td>
</tr>
<tr>
<td>Ideal for</td>
<td>Debugging and testing, Large compilation</td>
<td>Production</td>
</tr>
</tbody>
</table>
• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
1) Interactive job

a) Command

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 1) Interactive job

### a) Command

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>qsub -I [options]</code></td>
<td><code>srun [options] --pty bash</code></td>
</tr>
</tbody>
</table>

(Or any other shell of your preference)
1) Interactive job

a) Command

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
</table>
| `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

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>qsub</strong> -I \</td>
<td><strong>srun</strong> \</td>
</tr>
<tr>
<td>-X \</td>
<td>--x11 \</td>
</tr>
<tr>
<td>-A &lt;Allocation name&gt; \</td>
<td>-A &lt;Allocation name&gt; \</td>
</tr>
<tr>
<td>-q &lt;Queue name&gt; \</td>
<td>-p &lt;Queue name&gt; \</td>
</tr>
<tr>
<td>-l walltime=<a href="">HH:MM:SS</a>,nodes=&lt;# of</td>
<td>-t <a href="">HH:MM:SS</a> \</td>
</tr>
<tr>
<td>nodes&gt;:ppn=&lt;# of cores PER NODE&gt;</td>
<td>-N &lt;# of nodes&gt; \</td>
</tr>
<tr>
<td></td>
<td>-n &lt;# of TOTAL cores&gt; \</td>
</tr>
<tr>
<td></td>
<td><strong>--pty bash</strong></td>
</tr>
</tbody>
</table>

Enable X11 forwarding
### 1) Interactive job

#### a) Command

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```bash
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

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
</table>
| qsub -I \  
-X \  
-A <Allocation name> \  
-q <Queue name> \  
-1 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

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>qsub -I</code> \ <code>-X</code> \ <code>-A &lt;Allocation name&gt;</code> \ <code>-q &lt;Queue name&gt;</code> \ <code>-l walltime=&lt;HH:MM:SS&gt;,nodes=&lt;# of nodes&gt;:ppn=&lt;# of cores PER NODE&gt;</code></td>
<td><code>srun</code> \ <code>--x11</code> \ <code>-A &lt;Allocation name&gt;</code> \ <code>-p &lt;Queue name&gt;</code> \ <code>-t &lt;HH:MM:SS&gt;</code> \ <code>-N &lt;# of nodes&gt;</code> \ <code>-n &lt;# of TOTAL cores&gt;</code> \ <code>--pty bash</code></td>
</tr>
</tbody>
</table>

Wall time, number of nodes, number of cores
### 1) Interactive job

#### a) Command

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td>`qsub -I \</td>
<td>`srun \</td>
</tr>
<tr>
<td>-X \</td>
<td>--x11 \</td>
</tr>
<tr>
<td>-A &lt;Allocation name&gt; \</td>
<td>-A &lt;Allocation name&gt; \</td>
</tr>
<tr>
<td>-q &lt;Queue name&gt; \</td>
<td>-p &lt;Queue name&gt; \</td>
</tr>
<tr>
<td>-l walltime=HH:MM:SS, nodes=&lt;# of nodes&gt;:ppn=&lt;# of cores PER NODE&gt;</td>
<td>-t HH:MM:SS \</td>
</tr>
<tr>
<td></td>
<td>-N &lt;# of nodes&gt; \</td>
</tr>
<tr>
<td></td>
<td>-n &lt;# of TOTAL cores&gt; \</td>
</tr>
<tr>
<td></td>
<td>--pty bash</td>
</tr>
</tbody>
</table>

- **PBS**
  - `qsub -I` starts an interactive job.
  - `--x11` is used for a graphical user interface.
  - `-A <Allocation name>` specifies an allocation.
  - `-q <Queue name>` specifies a queue.
  - `-l walltime=HH:MM:SS, nodes=<# of nodes>:ppn=<# of cores PER NODE>` sets the walltime and cores per node.

- **Slurm**
  - `srun` is used for an interactive job.
  - `--x11` is used for a graphical user interface.
  - `-A <Allocation name>` specifies an allocation.
  - `-p <Queue name>` specifies a queue.
  - `-t HH:MM:SS` sets the walltime.
  - `-N <# of nodes>` specifies the number of nodes.
  - `-n <# of TOTAL cores>` specifies the number of cores.
  - `--pty bash` runs a bash shell.

- **Notes**
  - Does not change with # of nodes
  - Scales proportionally with # of nodes
## 1. Basic concepts

### 1. Interactive job

#### a) Command

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>qsub -I</code></td>
<td><code>run</code></td>
</tr>
<tr>
<td><code>-X</code></td>
<td><code>--x11</code></td>
</tr>
<tr>
<td><code>-A &lt;Allocation name&gt;</code></td>
<td><code>-A &lt;Allocation name&gt;</code></td>
</tr>
<tr>
<td><code>-q &lt;Queue name&gt;</code></td>
<td><code>-p &lt;Queue name&gt;</code></td>
</tr>
<tr>
<td><code>-l walltime=&lt;HH:MM:SS&gt;,nodes=&lt;# of nodes&gt;:ppn=&lt;# of cores PER NODE&gt;</code></td>
<td><code>-t &lt;HH:MM:SS&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>-N &lt;# of nodes&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>--pty bash</code></td>
</tr>
</tbody>
</table>

For those who run MPI / OpenMP hybrid –

```
-n <# of tasks>
-c <# of cores per task>
⇒ <n> * <c> = <# of TOTAL cores>
```

- `n` <# of TOTAL cores>
### 1) Interactive job

#### b) Starting an interactive job

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>(base) [jasonli3@smic1 pi]$ qsub -I -A hpc_hn n=20</code>&lt;br&gt;<code>qsub: waiting for job 911565.smic3 to start</code>&lt;br&gt;<code>Interactive job 911565.smic3 waiting:</code>&lt;br&gt;<code>qsub: job 911565.smic3 ready</code>&lt;br&gt;<code>Concluding PBS prologue script - 31-Jan-2023</code></td>
<td><code>(base) [jasonli3@mike1 pi]$ srun -A hpc_hn</code>&lt;br&gt;<code>srun: Job is in held state, pending schedule</code>&lt;br&gt;<code>srun: job 38634 queued and waiting for re</code>&lt;br&gt;<code>Interactive job 38634 waiting:</code>&lt;br&gt;<code>srun: job 38634 has been allocated resources</code></td>
</tr>
</tbody>
</table>
### 1) Interactive job

#### b) Starting an interactive job

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
</table>
| ```
(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 pi]$ 
``` | ```
(base) [jasonli3@mike1 pi]$ srun -A hpc_h
srun: Job is in held state, pending scheduler
srun: job 38634 queued and waiting for resources
Interactive job 38634 waiting:
srun: job 38634 has been allocated resources
(base) [jasonli3@mike147 pi]$ 
``` |

**Successfully started:** on a computing node (3-digit number)
## 1) Interactive job

### b) Starting an interactive job

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
</table>
| (base) [jasonli3@smic1 pi]$ qsub -I -A hpc_hn n=20  
qsub: waiting for job 911565.smic3 to start  
Interactive job 911565.smic3 waiting:  
qsub: job 911565.smic3 ready  
Concluding PBS prologue script - 31-Jan-2023  
(base) [jasonli3@smic045 ~]$ | (base) [jasonli3@mike1 pi]$ srun -A hpc_hn  
srun: Job is in held state, pending scheduled  
srun: job 38634 queued and waiting for re  
Interactive job 38634 waiting:  
srun: job 38634 has been allocated resources  
(base) [jasonli3@mike147 pi]$ |

**PBS:** Job starts in `/home` directory  
**Slurm:** Job starts in *where the job was submitted*
### 1) Interactive job

#### c) One more thing about GPU jobs ...

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
</table>
| `srun` \  
  --x11 \  
  -A `<Allocation name>` \  
  -p `<Queue name>` \  
  -t `<HH:MM:SS>` \  
  -N1 \  
  -n16 \  
  --gres=gpu:1 \  
  --pty bash | [GPU] Request 1 out of 4 GPUs on SuperMike 3 |
1) Interactive job

d) Running an interactive job

i. Serial (single-thread)

ii. Parallel (MPI)

* Slurm + interactive + MPI:

$ srun <mpi_executable>

$ srun --overlap <mpi_executable>

Will hang

Will run
• **HPC User Environment 2**

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
2) Batch job

• What do you need?

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

  ii. Run a **submission command** to submit this batch file
### 2) Batch job

#### a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2) Batch job

a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#!/bin/bash</td>
<td>#!/bin/bash</td>
</tr>
<tr>
<td>#PBS -A &lt;Allocation name&gt;</td>
<td>#SBATCH -A &lt;Allocation name&gt;</td>
</tr>
<tr>
<td>#PBS -q workq</td>
<td>#SBATCH -p workq</td>
</tr>
<tr>
<td>#PBS -l walltime=1:00:00</td>
<td>#SBATCH -t 1:00:00</td>
</tr>
<tr>
<td>#PBS -l nodes=1:ppn=20</td>
<td>#SBATCH -N 1</td>
</tr>
<tr>
<td></td>
<td>#SBATCH -n 64</td>
</tr>
<tr>
<td>cd $PBS_O_WORKDIR</td>
<td>cd $SLURM_SUBMIT_DIR</td>
</tr>
<tr>
<td>./pi_serial.out 100000000</td>
<td>./pi_serial.out 100000000</td>
</tr>
</tbody>
</table>

## 2) Batch job

### a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#!/bin/bash</td>
<td>#!/bin/bash</td>
</tr>
<tr>
<td>#PBS -A &lt;Allocation name&gt;</td>
<td>#SBATCH -A &lt;Allocation name&gt;</td>
</tr>
<tr>
<td>#PBS -q workq</td>
<td>#SBATCH -p workq</td>
</tr>
<tr>
<td>#PBS -l walltime=1:00:00</td>
<td>#SBATCH -t 1:00:00</td>
</tr>
<tr>
<td>#PBS -l nodes=1:ppn=20</td>
<td>#SBATCH -N 1</td>
</tr>
<tr>
<td>cd $PBS_O_WORKDIR</td>
<td>cd $SLURM_SUBMIT_DIR</td>
</tr>
<tr>
<td>./pi_serial.out 100000000</td>
<td>./pi_serial.out 100000000</td>
</tr>
</tbody>
</table>

- **Job parameters**
- **Commands to execute after job starts**
## 2) Batch job

### a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#!/bin/bash</code></td>
<td><code>#!/bin/bash</code></td>
</tr>
<tr>
<td><code>#PBS -A &lt;Allocation name&gt;</code></td>
<td><code>#SBATCH -A &lt;Allocation name&gt;</code></td>
</tr>
<tr>
<td><code>#PBS -q workq</code></td>
<td><code>#SBATCH -p workq</code></td>
</tr>
<tr>
<td><code>#PBS -l walltime=1:00:00</code></td>
<td><code>#SBATCH -t 1:00:00</code></td>
</tr>
<tr>
<td><code>#PBS -l nodes=1:ppn=20</code></td>
<td><code>#SBATCH -N 1</code></td>
</tr>
<tr>
<td><code>#SBATCH -n 64</code></td>
<td><code>#SBATCH -n 64</code></td>
</tr>
<tr>
<td><code>cd $PBS_O_WORKDIR</code></td>
<td><code>cd $SLURM_SUBMIT_DIR</code></td>
</tr>
<tr>
<td><code>./pi_serial.out 1000000000</code></td>
<td><code>./pi_serial.out 1000000000</code></td>
</tr>
</tbody>
</table>

Allocation name
## 2) Batch job

### a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#!/bin/bash</code></td>
<td><code>#!/bin/bash</code></td>
</tr>
<tr>
<td><code>#PBS -A &lt;Allocation name&gt;</code></td>
<td><code>#SBATCH -A &lt;Allocation name&gt;</code></td>
</tr>
<tr>
<td><code>#PBS -q workq</code></td>
<td><code>#SBATCH -p workq</code></td>
</tr>
<tr>
<td><code>#PBS -l walltime=1:00:00</code></td>
<td><code>#SBATCH -t 1:00:00</code></td>
</tr>
<tr>
<td><code>#PBS -l nodes=1:ppn=20</code></td>
<td><code>#SBATCH -N 1</code></td>
</tr>
<tr>
<td><code>cd $PBS_O_WORKDIR</code></td>
<td><code>cd $SLURM_SUBMIT_DIR</code></td>
</tr>
<tr>
<td><code>.pi_serial.out 100000000</code></td>
<td><code>.pi_serial.out 100000000</code></td>
</tr>
</tbody>
</table>

Queue name: `workq`
2) Batch job

a) Batch file

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

cd $PBS_O_WORKDIR
./pi_serial.out 100000000` | `#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -t 1:00:00
#SBATCH -N 1
#SBATCH -n 64

cd $SLURM_SUBMIT_DIR
./pi_serial.out 100000000`

Wall time
2) Batch job

a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#!/bin/bash</td>
<td>#!/bin/bash</td>
</tr>
<tr>
<td>#PBS -A &lt;Allocation name&gt;</td>
<td>#SBATCH -A &lt;Allocation&gt;</td>
</tr>
<tr>
<td>#PBS -q workq</td>
<td>#SBATCH -p workq</td>
</tr>
<tr>
<td>#PBS -l walltime=1:00:00</td>
<td>#SBATCH -t 1:00:00</td>
</tr>
<tr>
<td>#PBS -l nodes=1:ppn=20</td>
<td>#SBATCH -N 1</td>
</tr>
<tr>
<td>cd $PBS_O_WORKDIR ./pi_serial.out 1000000000</td>
<td>cd $SLURM_SUBMIT_DIR ./pi_serial.out 100000000</td>
</tr>
<tr>
<td><strong>Number of nodes &amp; cores</strong></td>
<td><strong>Number of nodes &amp; cores</strong></td>
</tr>
</tbody>
</table>

2) Batch job

a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#!/bin/bash</td>
<td>#!/bin/bash</td>
</tr>
<tr>
<td>#PBS -A &lt;Allocation name&gt;</td>
<td>#SBATCH -A &lt;Allocation name&gt;</td>
</tr>
<tr>
<td>#PBS -q workq</td>
<td>#SBATCH -p workq</td>
</tr>
<tr>
<td>#PBS -l walltime=1:00:00</td>
<td>#SBATCH -t 1:00:00</td>
</tr>
<tr>
<td>#PBS -l nodes=1:ppn=20</td>
<td>#SBATCH -N 1</td>
</tr>
<tr>
<td></td>
<td>#SBATCH -n 64</td>
</tr>
<tr>
<td>cd $PBS_O_WORKDIR ./pi_serial.out 100000000</td>
<td>cd $SLURM_SUBMIT_DIR ./pi_serial.out 100000000</td>
</tr>
</tbody>
</table>

Commands to run after job starts
## 2) Batch job

### a) Batch file

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#!/bin/bash</code>&lt;br&gt;<code>#PBS -A &lt;Allocation name&gt;</code>&lt;br&gt;<code>#PBS -q workq</code>&lt;br&gt;<code>#PBS  -l walltime=1:00:00</code>&lt;br&gt;<code>#PBS -l nodes=1:ppn=20</code>&lt;br&gt;<code>cd $PBS_O_WORKDIR</code>&lt;br&gt;<code>./pi_serial.out 100000000</code></td>
<td><code>#!/bin/bash</code>&lt;br&gt;<code>#SBATCH -A &lt;Allocation name&gt;</code>&lt;br&gt;<code>#SBATCH -p workq</code>&lt;br&gt;<code>#SBATCH -t 1:00:00</code>&lt;br&gt;<code>#SBATCH -N 1</code>&lt;br&gt;<code>#SBATCH -n 64</code>&lt;br&gt;<code>cd $SLURM_SUBmitted生肖</code>&lt;br&gt;<code>./pi_serial.out 100000000</code></td>
</tr>
</tbody>
</table>
2) Batch job

### a) Batch file

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#PBS –A</td>
<td>#SBATCH -A</td>
<td>Allocation name</td>
</tr>
<tr>
<td>#PBS –q</td>
<td>#SBATCH -p</td>
<td>Queue name</td>
</tr>
<tr>
<td>#PBS –l</td>
<td>#SBATCH –t</td>
<td>Wall time</td>
</tr>
<tr>
<td>#SBATCH -N</td>
<td>Resource request</td>
<td>Number of nodes</td>
</tr>
<tr>
<td>#SBATCH –n</td>
<td></td>
<td>Number of tasks</td>
</tr>
<tr>
<td>#SBATCH –c</td>
<td></td>
<td>Number of cores per task</td>
</tr>
<tr>
<td>#PBS –o</td>
<td>#SBATCH -o</td>
<td>Standard output file</td>
</tr>
<tr>
<td>#PBS –e</td>
<td>#SBATCH -e</td>
<td>Standard error file</td>
</tr>
<tr>
<td>#PBS –m</td>
<td>#SBATCH --mail-type</td>
<td>Send email when</td>
</tr>
<tr>
<td>a</td>
<td>FAIL</td>
<td>Job aborts / fails</td>
</tr>
<tr>
<td>b</td>
<td>BEGIN</td>
<td>Job begins</td>
</tr>
<tr>
<td>e</td>
<td>END</td>
<td>Job ends</td>
</tr>
<tr>
<td>#SBATCH --mail-user</td>
<td>Email address</td>
<td></td>
</tr>
<tr>
<td>#PBS –M</td>
<td>#SBATCH -J</td>
<td>Job name</td>
</tr>
</tbody>
</table>

2) Batch job

b) Command

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>qsub &lt;batch file name&gt;</code></td>
<td><code>sbatch &lt;batch file name&gt;</code></td>
</tr>
</tbody>
</table>
## 2) Batch job

### c) Useful environmental variables

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$PBS_JOBID</td>
<td>$SLURM_JOBID</td>
<td>Job ID</td>
</tr>
<tr>
<td>$PBS_O_WORKDIR</td>
<td>$SLURM_SUBMIT_DIR</td>
<td>Job submit directory</td>
</tr>
<tr>
<td>$PBS_NODEFILE</td>
<td>$SLURM_JOB_NODELIST</td>
<td>A temp file, contains a list of allocated nodes’ names (for MPI)</td>
</tr>
<tr>
<td>$PBS_NUM_NODES</td>
<td>$SLURM_NNODES</td>
<td>Number of allocated nodes</td>
</tr>
<tr>
<td>$PBS_NP</td>
<td>$SLURM_NTASKS</td>
<td>Number of allocated cores (tasks)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

```bash
#!/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
```

Outlines

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
4. Manage jobs

- Running jobs on HPC ≠ “Submit and done”
  - Monitoring and managing jobs are part of the work
Outlines

• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
## 1. Basic concepts

### 1) Useful commands

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>qstat</td>
<td>squeue</td>
<td>List all jobs</td>
</tr>
<tr>
<td>-n</td>
<td>-u &lt;Username&gt;</td>
<td>List job details</td>
</tr>
<tr>
<td>-u &lt;Username&gt;</td>
<td>-u &lt;Username&gt;</td>
<td>List all jobs belong to &lt;Username&gt;</td>
</tr>
<tr>
<td>qdel &lt;Job ID&gt;</td>
<td>scancel &lt;Job ID&gt;</td>
<td>Cancel &lt;Job ID&gt;</td>
</tr>
<tr>
<td>checkjob &lt;Job ID&gt;</td>
<td>scontrol show job &lt;Job ID&gt;</td>
<td>Show job details (running or recently finished)</td>
</tr>
</tbody>
</table>

Alter jobs after submission? → NOT allowed!

---

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

• HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers

2. Preparing my job
   1) Basic principles
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job
   2) Batch job

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health
2) Monitoring job health

A job requesting n cores ≠ A job utilizing n cores

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

• Things to check
  – CPU 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>**

<table>
<thead>
<tr>
<th>What to look at ...</th>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Job ID&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```bash
$ qshow 38581
```

**Example output:**

- **Hostname:**
  - mke1s
- **Days Load CPU U**
  - 666.08
- **User:**
  - xyan
- **Process:**
  - virtual memory:
    - 6852mb
  - average virtual mem:
    - 685mb
  - top proc:
- **Node processes:**
  - 68

**Legend:**
- **Normal behavior**
  - You should be concerned if...
## 2. Preparing my job

### Monitoring job health

#### a) Method 1: `qshow <Job ID>`

<table>
<thead>
<tr>
<th>What to look at ...</th>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>avg_load</code></td>
<td>Close to allocated number of cores on the node</td>
<td>Consistently too low or too high</td>
</tr>
</tbody>
</table>
2) Monitoring job health

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

<table>
<thead>
<tr>
<th>What to look at ...</th>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>avg_load</td>
<td>Close to allocated number of cores on the node</td>
<td>Consistently too low or too high</td>
</tr>
<tr>
<td>ave_mem</td>
<td>Does not exceed total allocated memory</td>
<td>Exceeds total allocated memory</td>
</tr>
</tbody>
</table>
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`

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>2701318</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>582356</td>
<td>2568</td>
<td>R</td>
<td>100.0</td>
<td>0.2</td>
<td>4:08:94</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701342</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581944</td>
<td>2616</td>
<td>R</td>
<td>100.0</td>
<td>0.2</td>
<td>4:08:90</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701249</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581792</td>
<td>2464</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:97</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701252</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>514684</td>
<td>2520</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:09:00</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701261</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>393828</td>
<td>2616</td>
<td>R</td>
<td>99.7</td>
<td>0.1</td>
<td>4:08:97</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701264</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581856</td>
<td>2532</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:92</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701270</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>582480</td>
<td>2432</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:95</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701273</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581776</td>
<td>2448</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:81</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701276</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>582160</td>
<td>2568</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:98</td>
<td>TDSE_np3_e0</td>
</tr>
</tbody>
</table>

What to look at ...  | Normal behavior ...  | You should be concerned if ...
2) Monitoring job health

b) Method 2: `top`

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>2701318</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>582356</td>
<td>2568</td>
<td>R</td>
<td>100.0</td>
<td>0.2</td>
<td>4:00.94</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701342</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581944</td>
<td>2616</td>
<td>R</td>
<td>100.0</td>
<td>0.2</td>
<td>4:00.90</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701249</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581792</td>
<td>2464</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:00.97</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701252</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>514684</td>
<td>2520</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:00.90</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701261</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>393828</td>
<td>2616</td>
<td>R</td>
<td>99.7</td>
<td>0.1</td>
<td>4:08.97</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701264</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581856</td>
<td>2532</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08.92</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701270</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>582480</td>
<td>2432</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08.95</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701273</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581776</td>
<td>2448</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08.81</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701276</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>582160</td>
<td>2568</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08.98</td>
<td>TDSE_np3_e0</td>
</tr>
</tbody>
</table>

What to look at ...

<table>
<thead>
<tr>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load average</td>
<td>Close to allocated number of cores on the node</td>
</tr>
</tbody>
</table>
2) Monitoring job health

b) Method 2: top

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>2701318</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>562356</td>
<td>2568</td>
<td>R</td>
<td>100.0</td>
<td>0.2</td>
<td>4:08:94</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701342</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581944</td>
<td>2616</td>
<td>R</td>
<td>100.0</td>
<td>0.2</td>
<td>4:08:90</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701249</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581792</td>
<td>2464</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:97</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701252</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>514684</td>
<td>2520</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:09:00</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701261</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>393828</td>
<td>2616</td>
<td>R</td>
<td>99.7</td>
<td>0.1</td>
<td>4:08:97</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701264</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581856</td>
<td>2532</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:92</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701270</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>582480</td>
<td>2432</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:95</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701273</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>581776</td>
<td>2448</td>
<td>R</td>
<td>99.7</td>
<td>0.2</td>
<td>4:08:81</td>
<td>TDSE_np3_e0</td>
</tr>
<tr>
<td>2701276</td>
<td>jasonli3</td>
<td>20</td>
<td>0</td>
<td>595668</td>
<td>573952</td>
<td>2544</td>
<td>R</td>
<td>99.7</td>
<td>0.1</td>
<td>4:08:98</td>
<td>TDSE_np3_e0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>What to look at ...</th>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load average</td>
<td>Close to allocated number of cores on the node</td>
<td>Consistently too low or too high</td>
</tr>
<tr>
<td>Memory usage (not virtual memory)</td>
<td>Does not exceed total allocated memory</td>
<td>Exceeds total allocated memory</td>
</tr>
</tbody>
</table>
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
```

<table>
<thead>
<tr>
<th>What to look at ...</th>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mem: 263172900</td>
<td>Mem: 216007308</td>
<td>Mem: &lt; 216007308</td>
</tr>
<tr>
<td>Swap: 17040380</td>
<td>Swap: 16978684</td>
<td>Swap: &lt; 16978684</td>
</tr>
</tbody>
</table>
2) Monitoring job health

c) Method 3: \textit{free}

What to look at ... | Normal behavior ... | You should be concerned if ...
---|---|---
Memory usage (not virtual memory) | Does not exceed total allocated memory | Exceeds total allocated memory
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)

What to look at ... | Normal behavior ... | You should be concerned if ...

```
(base) [jasonli@qc193 ~]$ nvidia-smi

Mon Feb 6 02:38:32 2023
NVIDIA-SMI 510.47.03 Driver Version: 510.47.03 CUDA Version: 11.6
GPU Name Persistence-M
Fan Temp Perf Pwr:Usage/Cap
Displ A Memory-Usage Volatile Uncorr. ECC GPU-Util Compute M.
MIG M.

9 Tesla V100-PCIE... On 36C P0 54W / 250W 80800000:38:00.0 Off 4155MiB / 32768MiB 72% Default N/A

1 Tesla V100-PCIE... On 36C P0 52W / 250W 80800000:AF:00.0 Off 4155MiB / 32768MiB 78% Default N/A

Processes:
GPU GI CI PID Type Process name GPU Memory 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
```
2) Monitoring job health

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

<table>
<thead>
<tr>
<th>What to look at ...</th>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU usage</td>
<td>Close to 100%</td>
<td>Consistently too low</td>
</tr>
</tbody>
</table>
### 2) Monitoring job health

<table>
<thead>
<tr>
<th>What to look at ...</th>
<th>Normal behavior ...</th>
<th>You should be concerned if ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU usage</td>
<td>Close to 100%</td>
<td>Consistently too low</td>
</tr>
<tr>
<td>Memory usage (not virtual memory)</td>
<td>Not used up</td>
<td>Used up</td>
</tr>
</tbody>
</table>

**d) Method 4: nvidia-smi** (for GPU only)
## 2) Monitoring job health

e) Common issues

<table>
<thead>
<tr>
<th>Issue</th>
<th>What would happen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exceeded memory allocation (e.g., using more memory than allocated w/ single queue)</td>
<td>Terminated. Receive email notice.</td>
</tr>
<tr>
<td>Exceeded ppn/core allocation (e.g., using more cores than allocated w/ single queue)</td>
<td>Terminated. Receive email notice.</td>
</tr>
<tr>
<td>Seriously underutilize node CPU cores (e.g., Requested multiple nodes but only runs on one node)</td>
<td>Receive email warning.</td>
</tr>
<tr>
<td>Submitting to bigmem but only using little memory</td>
<td>Nothing. Just not nice.</td>
</tr>
<tr>
<td>Running intensive calculation on head nodes</td>
<td>Terminated. Receive email notice.</td>
</tr>
<tr>
<td>Submitting too many (i.e., hundreds of) single-thread jobs</td>
<td>Poor parallelization and bad for server. We may reach out to you to help. (Better yet, reach out to us first)</td>
</tr>
</tbody>
</table>
Summary

• A typical workflow --

1. Test your code using *interactive jobs*
2. Monitor your jobs to determine resource usage
3. Submit production jobs using *batch jobs*
4. Keep monitoring your jobs and edit your production jobs as needed
HPC User Environment 2

1. Basic concepts
   1) Previously on HPC User Environment 1…
   2) Job & Job schedulers → All calculation must be submitted as jobs

2. Preparing my job
   1) Basic principles → Large enough & small enough
   2) Job duration (wall time)
   3) Number of nodes & cores
   4) Job queues

3. Submitting my job
   1) Interactive job → Good for testing and debugging
   2) Batch job → Good for production

4. Managing my jobs
   1) Useful commands
   2) Monitoring job health → How to monitor jobs health, and how to create health jobs
Next week

- Basic Shell Scripting
Contact us

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