ASPIRE 2A INTRODUCTORY WORKSHOP

National Supercomputing Centre (NSCC) Singapore

A national research infrastructure providing supercomputing solutions for all

By NSCC Singapore Managed Services Team

NSCC.SG

Updated on : 02 July 2026

Contents

A. Introduction to NSCC & HPC

B. ASPIRE 2A

National Supercomputing Centre

A. Introduction to NSCC & HPC

All terms of use are subjected to the prevailing

NSCC Acceptable Use Policy https://help.nscc.sg/aspire2a/aup/

Introduction Video

https://help.nscc.sg/wp-content/uploads/Intro-to-NSCC-HPC_20260525.mp4

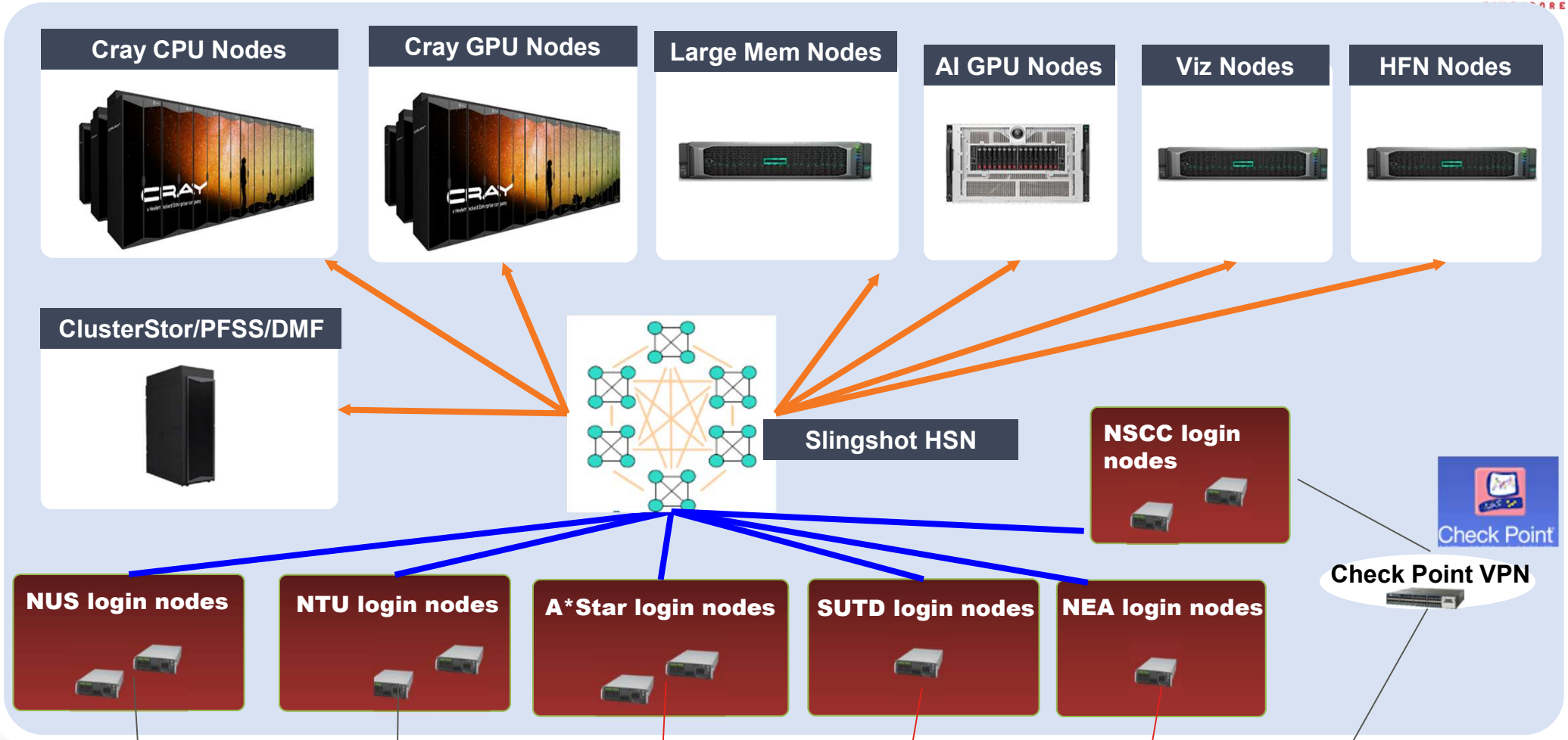
National Supercomputing Centre

B. ASPIRE 2A

National Supercomputing Centre

1. Architecture

ASPIRE2A Architecture Overview



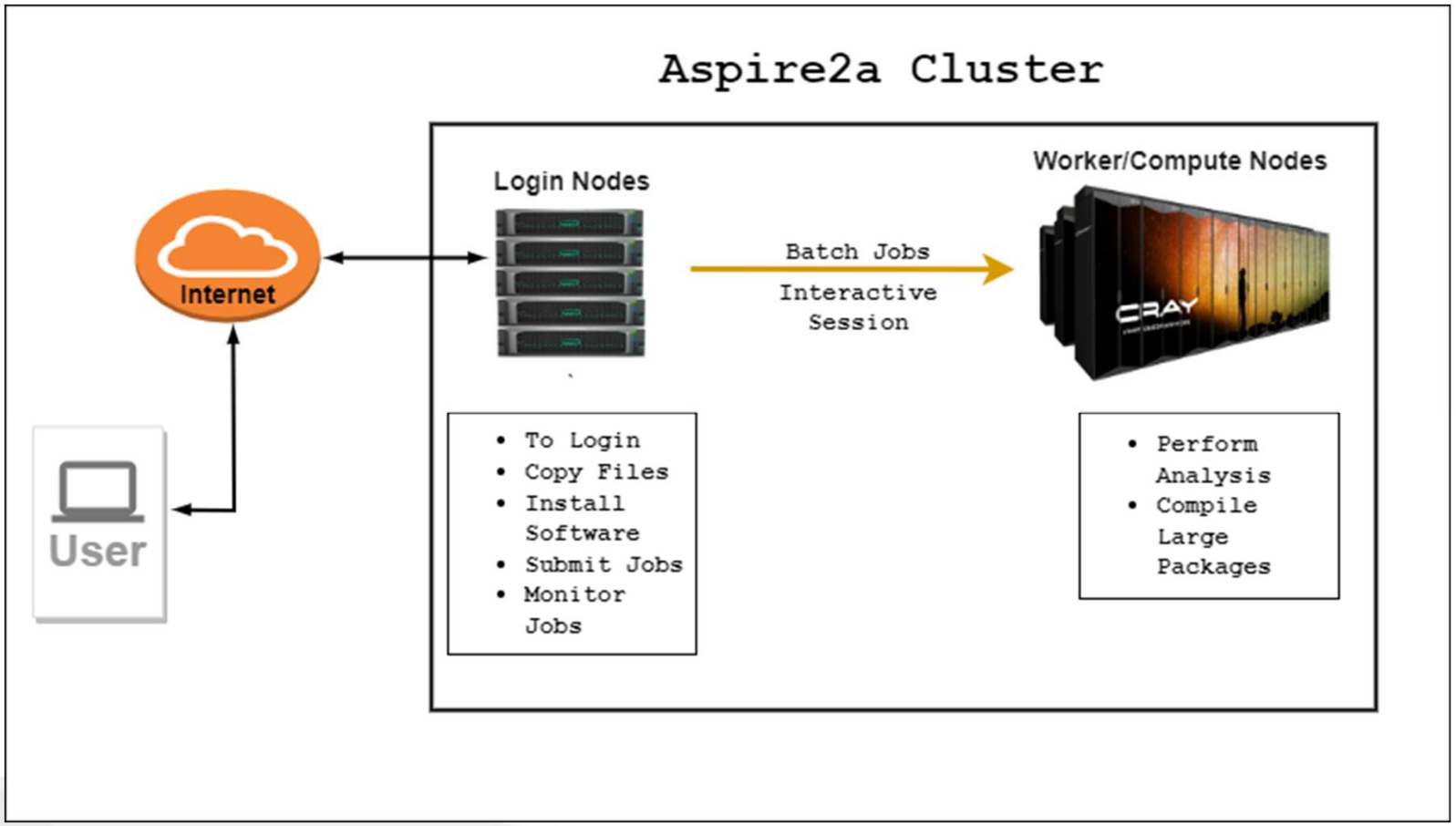
ASPIRE 2A Data Centre Location

National Supercomputing Centre

2. Login Nodes

Purpose Of Login Nodes

- NSCC login nodes are the entry point for users to access the **ASPIRE 2A HPC system**.
- NSCC login nodes are designed for lightweight workload.



ASPIRE 2A Login Nodes

Direct Access

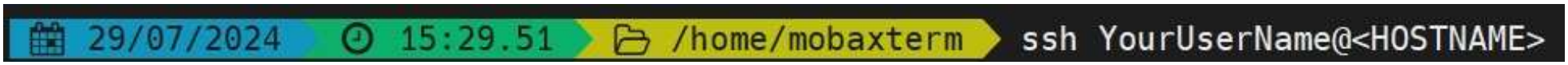
Organisations	Login Hostname FQDN
NUS	aspire2a.nus.edu.sg
NTU	aspire2antu.nscc.sg **Note:- NTU users will need to request NTU jumphost to access ASPIRE2A via aspire2antu.nscc.sg** Please email your jumphost access request to hpcsupport@ntu.edu.sg **How to access jump host:-** Using-NTU-JumpHost-to-NSCC-ASPIRE-2A
ASTAR	aspire2a.a-star.edu.sg
SUTD	aspire2a.sutd.edu.sg

VPN Access

Hostname FQDN
aspire2a.nscc.sg
login.asp2a.nscc.sg

Accessing Login Nodes

- To access a login node, users typically use an SSH client to connect to the login node's hostname or IP address
  - Windows ssh client : MobaXterm, putty
  - Mac ssh client : Mac terminal
  - Linux ssh client : Linux terminal



Example : ssh to remote host using MobaXterm

Note : Users who are not connected to NUS, NTU, A*STAR, or SUTD network must ensure they are connected to their organisation's VPN (default option) or NSCC VPN (for selected users only).

National Supercomputing Centre

3. Storage

ASPIRE 2A Storage

File System	Mount Point	Quota Per user	Data Retention Policy
Lustre	$HOME/scratch	100TB **[Fixed]**	30 Days Purge Policy
GPFS	/home/users/<your_org>/<userid> $HOME	50GB **[Fixed]**	1 Year From Account Expiry
GPFS	/home/project/<project-id>	Based on project	Based On Project Expiry**
Local NVME	/raid (Only accessible on AI nodes)	NA	After Job completion

****Project data will be archived 30 days after the project's expiry date.**

Note : Data Management and Retention Policy

User and project storage allocated will be purged according to AUP

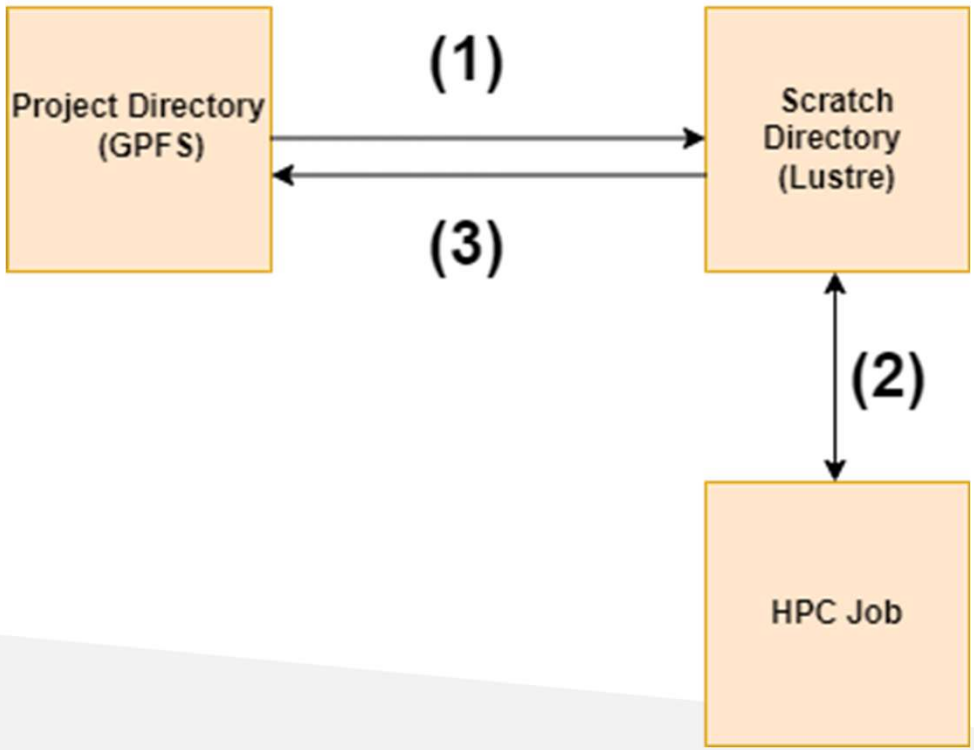
https://help.nscc.sg/aspire2a/aup/

Best Practices For Using Scratch and Project Directories

Step 1 : Transfer input data from the project directory to the scratch directory.

Step 2 : Execute HPC job that read/write from/on the scratch directory, utilizing Lustre IO performance.

Step 3 : Transfer important data back to project directory from scratch directory.



Check Storage Quota

- **Check Home and Scratch quota**

```
]$ myquota
```

- **Check Project quota**

```
]$ myquota -p <project-id>
```

National Supercomputing Centre

4. File Transfer

Transfer Files From Local To ASPIRE 2A

Filezilla

- **User-friendly**: Easy graphical user interface.

SCP (Secure Copy Protocol)

- **Simple & Secure**: Use SSH for secure transfers.

```
]$ scp /path/to/sourcefile username@destination:/path/to/destination
```

Rsync

```
]$ rsync -avz /path/to/source username@destination:/path/to/destination
```

Transfer Files Within ASPIRE 2A File System

How to navigate and move/copy data from/to GPFS to/from Lustre

STEP 1 : Start a Screen Session

```
]$ screen -S <screen_name>
```

STEP 2 : Submit an Interactive Job

```
]$ qsub  -I -l select=1:ncpus=16:ompthreads=1:mem=50GB -P <yourProject> -l
walltime=02:10:00
```

STEP 3 : Initiate the Transfer

```
]$ cp –rv /path/to/source_file /path/to/destination_directory/
```

- How to share my folder with others ?
- Refer to FAQ 10.2 https://help.nscc.sg/aspire2a/faqs/

Introduction to Screen Command

What is Screen?

Screen is a terminal multiplexer that allows you to:

- Start a terminal session and keep it running even if you disconnect.
- Reconnect to the session at a later time.
- Run multiple terminal sessions within a single window.

Basic Screen Commands

- Starting a New Screen Session with Screen Name

```
]$ screen -S <screen_name>
```

- Listing All Screen Sessions

```
]$ screen -ls
```

- Reattaching to a Session

```
]$ screen -r session_name
```

- Detaching from a Session

```
]$ Ctrl + A, then D
```

National Supercomputing Centre

5. HPC Software Stack

HPC Software Stack on ASPIRE 2A

System

Workload Manager
- Altair PBS Pro

Remote Visualization
- Altair Access

Container
- Singularity
- *Kubernetes

Parallel File System
- GPFS
- Lustre

Operating System
- Red Hat Enterprise Linux 8

Development

Cray Programming Environment
- AMD AOCC
- GNU GCC
- HPE CCE
- Intel oneAPI
- NVHPC

Programming
- Nvidia CUDA
- Go
- Python
- Java
- Julia
- Octave
- R

Library

MPI Libraries
- Cray MPICH
- OpenMPI

Scientific Libraries
- AMD AOCL
- BLIS
- Cray LibSci
- FFTW
- GSL
- Intel MKL
- PETSc

I/O Libraries
- HDF5
- NetCDF

Profiling & Debugging

- ARM Forge
- Cray Stat
- gdb4hpc
- Perftools
- Valgrind4hpc

Bio Tools
- ABySS
- BEAST
- bedtools
- BWA
- bowtie2
- GATK
- BLAST+
- SAMtools

Workflow
- Cylc
- Rose
- Nextflow

Application

AI
- Tensorflow
- PyTorch

Simulation
- BerkeleyGW
- GROMACS
- LAMMPS
- Mumax3
- OpenFOAM
- Nek5000
- Quantum ESPRESSO
- Nektar++
- WRF

Applications and Tools
- Miniforge3
- FCM
- GNU Parallel
- FFmpeg
- MRtrix3
- NCL
- RELION
- Ncview
- Tmod
- Velvet

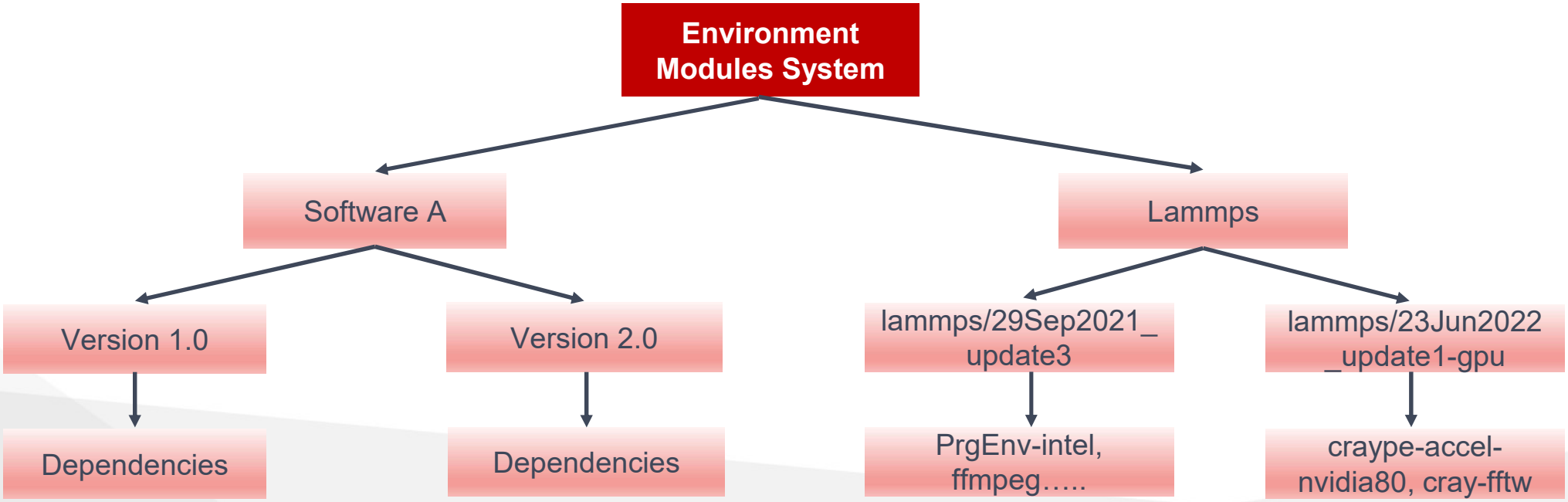
*Software list may be outdated, see module avail for latest versions

Environment Modules System

- Environment modules are a tool for dynamically modifying the user environment via modulefiles.
- They are commonly used in HPC systems to manage different software versions and their dependencies.

Benefits of Using Modules

- **Simplifies Environment Management**: Easily switch between different software versions.
- **Version Control** : Different versions of the same software can coexist, allowing users to select the appropriate version of their task.
- **Simplifies Dependency Management**



Module Commands

Command	Description
module list	List the loaded modules
module avail	List all module
module load <package name>	Load module
module rm/unload <package name>	Unload module **Please read the note below
module swap modulefile/1 modulefile/2	Swap loaded modulefile/1 with modulefile/2
module -l avail 2>&1 | egrep -i "name"	Searching for module
module show <package name>	Show the configure information of module

**** Unloading module or module purge can cause software incompatibility.**

Cray Programming Environment

Compiler	Module Name
Cray Environment (**default**)	PrgEnv-cray
GNU Environment	PrgEnv-gnu
Intel Environment	PrgEnv-intel
AMD Environment	PrgEnv-aocc
NVHPC Environment	PrgEnv-nvhpc
CUDA Environment	craype-accel-nvidia80

Compiler Wrappers

The Cray Compiling Environment (CCE) provides the compiler wrappers for **Fortran, C and C++ compilers** via **ftn, cc and CC**

- Use `**cc**` to compile c source code with the C compilers after you have swapped to the corresponding programming environment (GCC, Intel, Cray, AOCC, NVHPC).

cc -> ~~gcc,icc/icx,cc,clang,nvc, mpicc~~

- Use `**CC**` to compile c++ source code with the C++ compilers after you have swapped to the corresponding programming environment (GCC, Intel, Cray, AOCC, NVHPC).

CC -> ~~C++/g++,icpc/icpx, CC, clang++, nvc++, mpiicc~~

- Use `**ftn**` to compile Fortran source code with the Fortran compilers after you have swapped to the corresponding programming environment (GCC, Intel, Cray, AOCC, NVHPC).

ftn -> ~~gfortan, ifort/ifx, flang, nvfortran, mpiifort~~

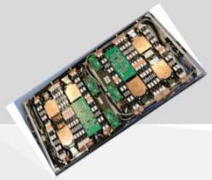
National Supercomputing Centre

6. System Specification

Type Of Compute Nodes On ASPIRE 2A

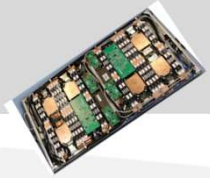
CRAY EX CPU NODES

- **768 nodes**
- Dual socket AMD EPYC™ 7713 CPUs, 64 cores per socket
- 128 cores per node
- 512GB DDR4 ECC RAM per node.
- 1x100G HSN (Slingshot)
- PBS queue : normal



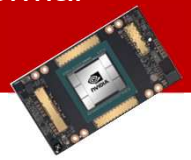
LARGE MEMORY NODES

- **16 nodes**
- Dual socket AMD EPYC™ 7713 CPUs, 64 cores per socket
- 12 Node x 2TB
- 4 Node x 4TB
- 1x100G HSN (Slingshot)
- PBS queue : normal



CRAY EX GPU NODES

- **64 nodes**
- Single socket AMD EPYC™ 7713 CPUs, 64 cores
- 4x Nvidia A100
- (40GB per GPU)
- 512GB DDR4 ECC RAM per node.
- 2x100G HSN (Slingshot)
- PBS queue : normal



AI NODES

- **12 nodes** with 4x Nvidia A100 (40GB), single AMD 7713, 12 TB nvme, 512GB RAM.
- **6 nodes** with 8x Nvidia A100 (40GB), dual socket AMD 7713, 14TB nvme, 1TB RAM
- 2x100G HSN
- PBS queue : ai

One Centre, Many Applications

HIGH FREQUENCY NODES

- **16 nodes**
- Dual socket AMD EPYC™ 75F3 CPUs, 32 cores per socket
- 64 cores per node
- 512GB DDR4 ECC RAM per node.
- 1x100G HSN (Slingshot)
- Kubernetes cluster

VISUALIZATION NODES

- **8 nodes**
- Dual socket AMD EPYC™ 7713 CPUs, 128 cores per socket,
- 256GB DDR4 RAM per node
- NVIDIA A40 GPU (48GB)
- PBS portal : visual.nscc.sg
- Applications : [Paraview,VMD]

PARALLEL FILE SYSTEMS

- I/O bandwidth up to 500GB/s
- GPFS and Lustre File System

SLINGSHOT INTERCONNECT

- Dragonfly Topology
- CPU nodes – 1x100G Interconnect
- GPU nodes – 2x100G Interconnect

National Supercomputing Centre

7. Service Unit Allocation

Service Unit Allocation

- PBS scheduler resource allocation measure in service unit (SU)
- SU is the currency to utilize resources on ASPIRE2A system.
- SU is charged based on the requested CPU and GPU resources as follows:
  - **For CPU only jobs:** 1 SU for 1 CPU core hour.
  - **For GPU jobs** : 64 SU for 1 GPU card hour (no charge for CPU core hour.)

- All users get one-time personal quota upon account creation
  - 100k SU if they are from - NUS, SUTD, NTU, A*STAR
  - 10k SU if they are from - SIT, SMU, SUSS, NP, NYP, RP, SP, TP
  - The personal SU quota is **fixed, non-transferable and cannot be topped up**.

- Once the personal quota is depleted, you can only submit a job through an approved project.

Job Submission Types and SU Usage

For CPU Only Jobs

Usage: 1 SU for 1 CPU core hour.

- **Example: # Request 128 CPUs (One node) for 2 hours**
- **Total SU usage : `1 node * 128 ncpus * 2 Hours = 256 SUs`**

```
#PBS -l select=1:ncpus=128:mem=440G -l walltime=02:00:00 -q normal -P <Project-id>
```

For GPU Jobs

Usage: 64 SU for 1 GPU card hour (no charge for ncpus).

- **Example: # Request 8 GPUs (Across two nodes) for 3 hours**
- **Total SU usage : `2 nodes * 4 ngpus * 3 Hours * 64 = 1536 SUs`**

```
#PBS -l select=2:ngpus=4 -l walltime=03:00:00 -q normal -P <project-id>
```

Notes:

- Users can check SU utilization with "**myusage**" command.
- Project ID must be specified.
- Users should not request unusual amounts of resources.
- Scheduler will block the requested amount of service unit (SU) from the project once the job is submitted.
- Once the job is completed, the unused SUs will be refunded to the project.

Call for Projects

SU allocation are based on approved projects

- Please check our website for more information on the available Call for Projects: https://help.nscc.sg/aspire2a/nscc-call-for-project/
- Project are allocated based on approval.
- Project ID will be allocated once the project is approved.
- Project period is created based on the project allocation cycles.
- Project member update can be requested at any time by project PI by sending request to projects-admin@nscc.sg

More information in FAQs under https://help.nscc.sg

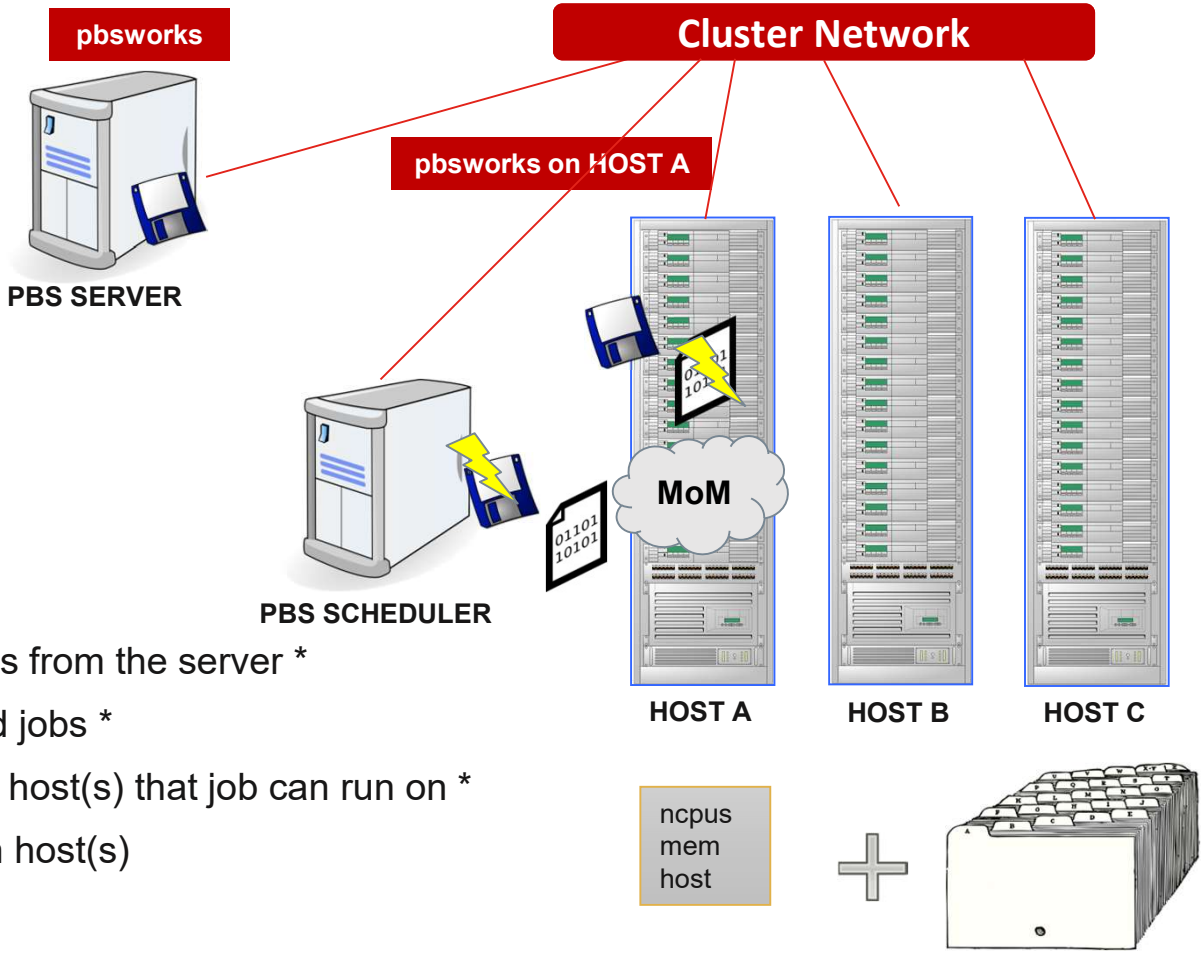
National Supercomputing Centre

8. PBS Professional (Job Scheduler)

What is PBS Pro Job Scheduler ?

- PBS (Portable Batch System) is a job scheduler.
- Commonly used in high-performance computing environments to manage and schedule computing resources.
- It allows users to submit batch and Interactive jobs.
- Scheduler manages the allocation of resources.
- **Some key features of PBS include:**
  - Job scheduling.
  - Resource allocation and budgeting.
  - Job monitoring.

Process Flow Of a PBS Job



1. User submits job
2. PBS server returns a job ID
3. PBS scheduler requests a list of resources from the server *
4. PBS scheduler sorts all the resources and jobs *
5. PBS scheduler informs PBS server which host(s) that job can run on *
6. PBS server pushes job script to execution host(s)
7. PBS MoM executes job script
8. PBS MoM periodically reports resource usage back to PBS server *
9. When job is completed PBS MoM copies output and error files
10. Job execution completed/user notification sent

Note: * This information is for debugging purposes only. It may change in future releases.

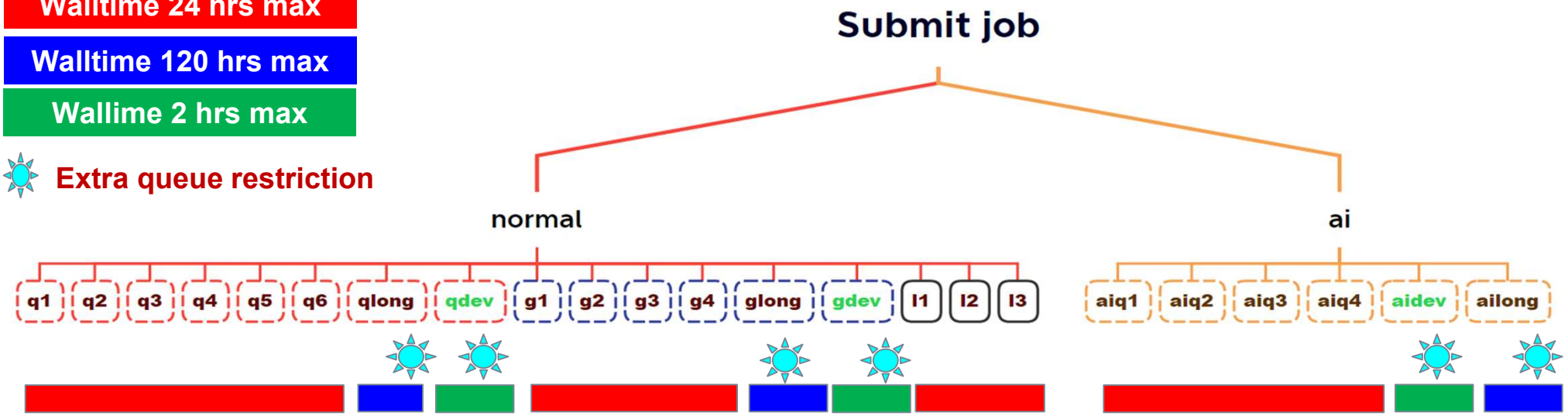
Job Queues & Scheduling Policies

Walltime 24 hrs max

Walltime 120 hrs max

Wallime 2 hrs max

Extra queue restriction



Cray EX CPU Nodes

CPU/Memory Limit:
0 < ncpus ≤ 98,304
mem ≤ 440gb / node

Cray EX GPU Nodes

GPU Limit:
0 < ngpus ≤ 256

Default ratio:-
16 CPU cores / GPU
110GB mem /GPU

Large Memory Nodes

Memory Limit :
440GB < mem < 3.9TB

AI Nodes

GPU Limit:
0 < ngpus ≤ 96

Default ratio:-
16 CPU cores / GPU
110GB mem /GPU

Types of Job Submission

1. Interactive Jobs

- Allows users to use compute node interactively.
- Useful for debugging, running short tests, transfer files and compile/build applications.

```
$] qsub -I -l select=1:ncpus=16:mem=48G -l walltime=01:00:00 -q normal -P
<Project-id>
```

- -I : Interactive job
- -l select=1:ncpus=16:mem=48G : Request for 1 node with 16 CPU cores and 48G RAM
- -l walltime=01:00:00 : Interactive job will be terminated after one hour

2. Batch Jobs

- Batch jobs are submitted using job scripts, which specify tasks, resource requirements, and execution commands.
- The scheduler manages the job script execution, allocating resources as requested.
- Users don't need to stay logged in to monitor the job.

PBS Sample Batch Job File

```
#!/bin/bash

#PBS -N Tensorflow                                          → Job Name
#PBS -l select=1:ncpus=128:mpiprocs=128:ompthreads=1:mem=440GB
                                                            → Resource Allocation
#PBS -q normal                                              → Queue Name
#PBS -l walltime=24:00:00                                   → Max hours job can run
#PBS -j oe                                                  → Join STDOUT and STDERR
#PBS -P <Project-ID>                                        → Project ID

cd $PBS_O_WORKDIR || exit $?                                → Job Submission Directory

mpirun -np 128 ./mpihello                                   → Program Executable
```

PBS Basic Commands

PBS commands are used to submit, manage, and modify jobs in a queue.

- **Submit batch Jobs**
  - `qsub <job-script>`
- **Check Job Status**
  - `qstat –answ`
  - `myqstat`
- **Delete Jobs**
  - `qdel <job-id>`
- **Hold and Release Jobs**
  - `qhold <job-id>`
  - `qrls <job-id>`

Extra Queue Restrictions

Queue	Execution Queue	Restriction
normal **(Default)**	**qlong**	• Maximum of 2 running jobs or 128 ncpus in use, whichever comes first.
	qdev	• Maximum of 2 running jobs per user • Total 256 ncpus available in this queue.
	gdev	• Maximum of 4 ngpus and 440 gb memory per job. • Maximum of 2 running jobs • Total 16 ngpus available in this queue.
	glong	• Maximum of 64 ngpus per job • Maximum 1 running jobs per user
ai	**ailong**	• Maximum of 1 running job per user • Maximum up to 4 ngpus per user
	aidev	• Maximum 1 node per user

NOTE 1: There is a system wide restriction of maximum of 100 Jobs per user, unless otherwise specified at queue level.

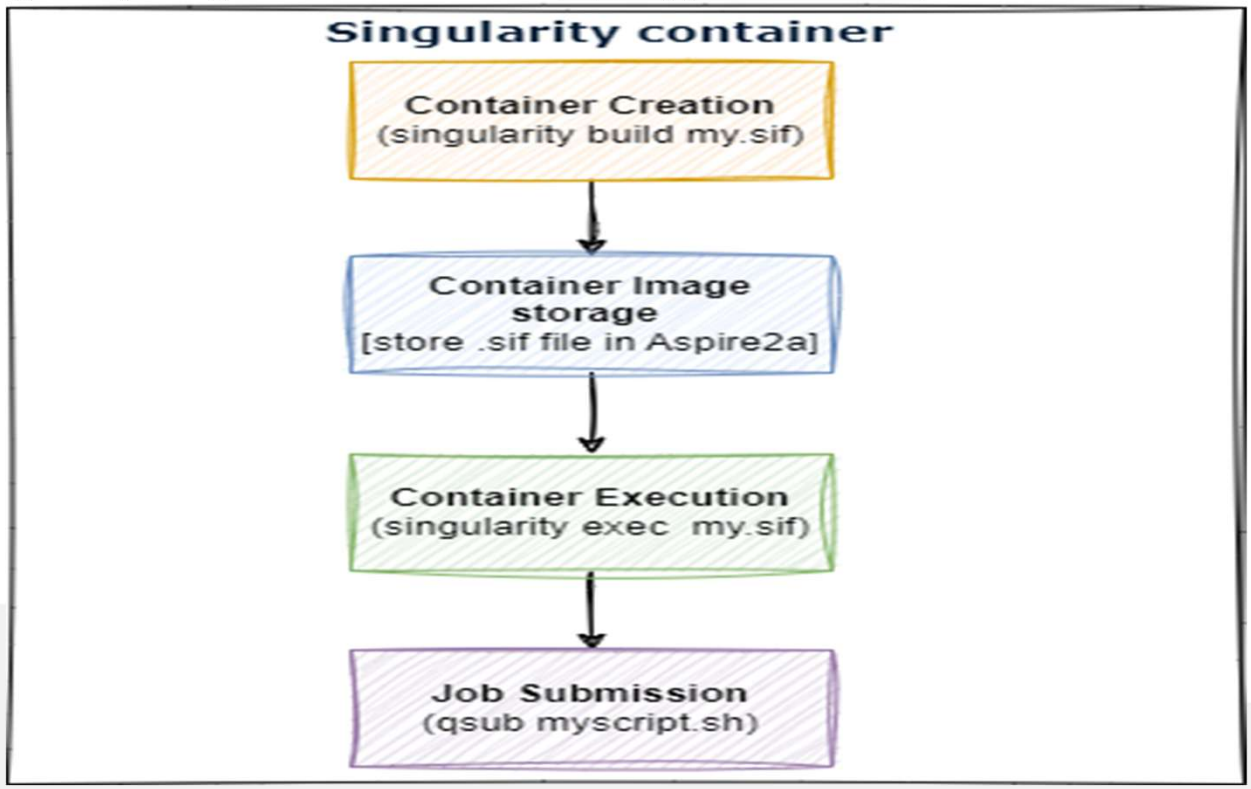
NOTE 2: Users specify only the queue name, i.e., "normal" or "ai". Execution queues are internal and beyond users control.

National Supercomputing Centre

9. Containers and Miniforge

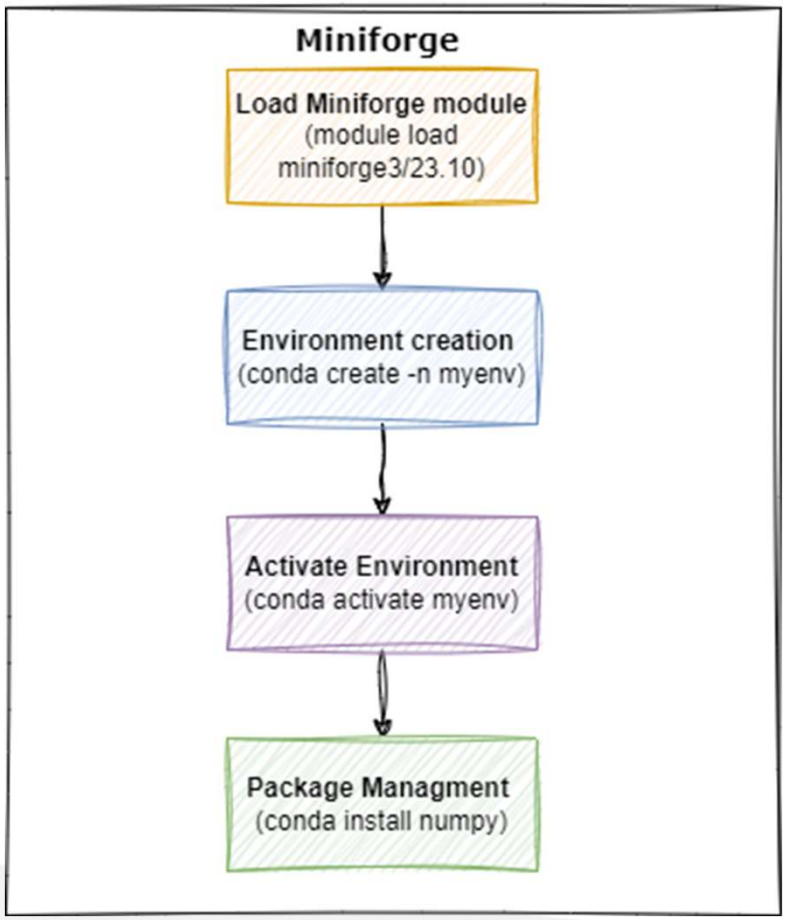
Containers (Singularity)

- ASPIRE2A supports Singularity containers.
- **Singularity** is a container platform designed for use on HPC systems.
- No sudo or root privileges are provided due to security reason.
- Containers built elsewhere can be copied to NSCC for use.
- Docker containers are not supported in ASPIRE2A, as a workaround user need to convert Docker image to Singularity image and run as Singularity container.
- On ASPIRE 2A, Singularity images are available at "/app/apps/containers/"



Miniforge

- **Miniforge** is a Lightweight Conda installer with community-driven package support
- Ideal for creating and managing isolated environments in HPC systems.



National Supercomputing Centre

10. Usage Best Practices

Resource Request Considerations

Optimize Resource Allocation

- **Distribute Workloads:** Ensure that workloads are evenly distributed across the CPU cores, memory and GPU cards to avoid bottlenecks.

- **Cray CPU System Specifications:**
  - **128 CPU core per node.**
  - **440GB memory per node.**

Example:- Core and Memory distribution [1:4]

- 1 Core Configuration : 4gb * 1 core
- 64 Cores Configuration : 256GB * 64 cores
- 128 Cores Configuration : 440gb * 128 cores

Cont'd

- **Cray GPU System Specifications**
  - 4 GPU card per node
  - 64 CPU core per node
  - 440GB memory per node

Example: GPU, Memory and core distribution

- 1 GPU configuration : 110gb * 16ncpus * 1GPU
- 4 GPU configuration : 440gb * 64ncpus * 4GPU's

Optimising and Scaling Your Computational Workloads

- **ALWAYS Start Small (1 CPU node or 1 GPU card)**
  - **Efficient Utilisation:** Ensure your application can utilise close to 100% of the compute capacity on CPU node or GPU card before scaling your job.
  - **Usage Monitoring:** Always monitor your workload using `top` and `nvidia-smi` for CPU and GPU workload respectively.
- **Scale to Multiple CPU Nodes or GPU Cards**
  - **Incremental Scaling:** Gradually increase the number of CPU nodes or GPU cards e.g., 2, 4, 8, 16, 32 measuring performance for each configuration
  - **Parallel Efficiency:** Study the workload performance for each configuration and choose the one the offers the best parallel efficiency for future workloads.
- **CPU Workload**
  - **CPU Binding:** Use option "`--cpu-bind = depth`" to bind each process to each physical core. (Please refer to `man mpiexec`)
  - **Hybrid Approach:** Always consider using an OpenMP/MPI hybrid approach and take NUMA regions into account.
- **GPU Workload (**Make sure to install or load GPU version of your application**)**
  - **Read Documentation**: Running workloads on GPUs requires using the correct application options. Ensure you are using the optimised options.

Do's and Don'ts

Do's

- Install applications in Home or Project directory.
- Share files through Project directory.
- Regular housekeeping in Home, Project and Scratch directories.
- Use Scratch directory for temporary files.
  - ➢ Move essential files to project directory after job completion. Note : Files in scratch are subjected to purging.
- Use striping for large files on scratch directory by using 'lfs setstripe' command.
- Execute data transfer on compute nodes for optimal performance.
- Utilize the 'find' and 'rm' commands for efficient file management.

Don'ts

- Avoid running any computational workload on login nodes.
- Avoid installing applications on Scratch directory.
- Avoid copying & pasting job scripts from web or Microsoft Windows OS.
- Avoid static settings in .bashrc, instead make use of environment modules.
- Avoid setting world readable/writable files & directories on Home, Scratch and Project directories.
- Avoid creating/accessing large number of small files in scratch directory.
- Avoid deleting large number of files using "rm –rf *".

Contact Us

National Supercomputing Centre

Website : https://nscc.sg

Self Service Portal : https://help.nscc.sg/

Helpdesk : https://keris.service-now.com/csm

Email : help@nscc.sg

Contact : +65 6645 3412

Event Survey

Your feedback is valuable to us!

EVENT SURVEY

(Pls scan the QR code to participate)

National Supercomputing Centre

Email : help@nscc.sg

National Supercomputing Centre

Thank You

NSCC.SG