

# High Performance Computing in a Nutshell

HPC Services, NHR@FAU

[hpc-support@fau.de](mailto:hpc-support@fau.de)

Which application area do you come from?



# Getting an HPC account

<https://hpc.fau.de/systems-services/documentation-instructions/getting-started/>

# HPC account – How to get one

- You need a separate account (not your IdM account)
- Transition from paper application forms to HPC Portal is currently in progress



- If in doubt, ask your local RRZE contact person for help
- If you change your affiliation or project, you need a new HPC account. Data migration may be required

# HPC account – HPC-portal

- Account via [portal.hpc.fau.de](https://portal.hpc.fau.de)
- Invitation has to be send by project PI/technical contact/contact person at chair
- Login to HPC portal with credentials of home institution via SSO
- No account passwords, cluster login only via SSH keys!
- SSH keys can be managed over HPC portal

The screenshot displays the HPC-Portal interface. At the top, the navigation bar includes the NHR and FAU logos, the text 'HPC-Portal Home', and a 'User' dropdown menu. The main content area is titled 'Your accounts' and features a 'Refresh accounts' button. Below this, there are tabs for 'Active accounts (1)' and 'Archived accounts (0)'. A single account is listed, with a red box around the account icon and another around the project name 'Tier3-Grundversorgung'. A callout box points to the project name with the text: 'project is either: Tier3-Grundversorgung' and 'name of an NHR project'. Below the account list, there is a 'Public SSH keys' section with a '+ Add new SSH key' button. A second callout box points to this button, leading to a modal window for adding a new SSH key. This modal includes a 'Back' button, 'Submit' and 'Reset' buttons, and input fields for 'Alias' and 'Key Content'. A 'Hide advanced options' button is at the bottom of the modal.

# HPC systems at NHR@FAU

<https://hpc.fau.de/systems-services/documentation-instructions/>

# Which cluster should I use?

Does your application run on multiple nodes simultaneously?

Yes

(e.g. by using MPI)

Meggie

Fritz

(Tier3 after application)

No

Only single-node or single-core work

Does your application use GPUs?

Yes

TinyGPU

Alex

(Tier3 after application)

No

Woody

TinyFat

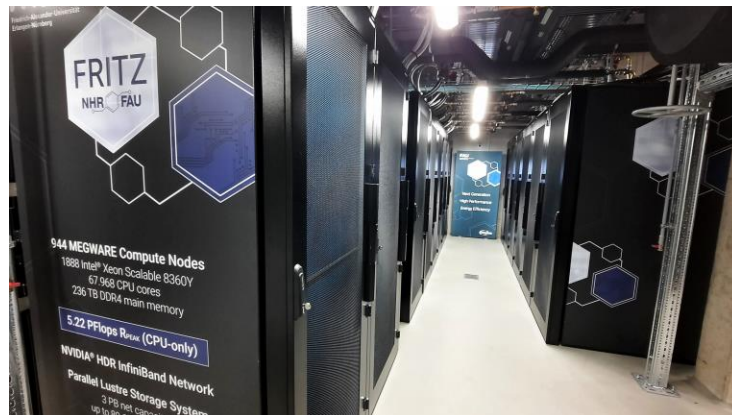
Accessible for projects

- “Tier3-Grundversorgung”
- NHR

# “Fritz” cluster

NHR parallel cluster, open for Tier3 users after application

- 992 compute nodes (71.424 cores)
  - 2 Intel Xeon Platinum 8360Y “Ice Lake” 2.4 GHz (36 cores)
  - 256 GB main memory per node
- 64 huge-memory nodes
  - 2 Intel Xeon Platinum 8470 “Sapphire Rapids” 2.0 GHz (52 cores)
  - 1 or 2 TB of main memory per node
- Blocking (1:4) HDR100 Infiniband network, up to 100 GBit/s
- Parallel file system with 3,5 PB capacity





# “Meggie” cluster

current main cluster for parallel jobs, intended for highly parallel jobs (Tier3)

- 728 Compute nodes (14.560 cores)
  - 2 Intel Xeon E5-2630 v4 (Broadwell) 2.2 GHz (10 cores)
  - 20 cores/node
  - 64 GB main memory
- No local disks
- Intel OmniPath network: Up to 100 Gbit/s



# “Alex” cluster

NHR GPGPU cluster, open for Tier3 users after application

- 44 nodes with
  - 8x NVIDIA A100 (each 40 GB / 80GB HBM2)
  - 2x AMD EPYC 7713 “Milan” 2.0 GHz, 1024 GB / 2048 GB of main memory
  - 14TB local NVMe SSD
  - HDR200 Infiniband network
- 38 nodes with
  - 8x NVIDIA A40 (each with 48 GB DDR6)
  - 2x AMD EPYC 7713 “Milan” 2.0 GHz, 512 GB of main memory
  - 7 TB local NVMe SSD



# “TinyGPU” cluster

for GPU workloads – not all nodes always generally available (Tier3)

- 12 nodes with 2x “Skylake” @ 3.2 GHz, 96 GB RAM, 1.8 TB SSD, 4x RTX 2080Ti
- 4 nodes with 2x “Skylake” @3.2 GHz, 96 GB RAM, 2.9 TB SSD, 4x Tesla V100
- 7 nodes with 2x “Cascade Lake” @2.9 GHz, 384 GB RAM, 3.8 TB SSD, 8x RTX3080
- 8 nodes with 2x AMD Rome 7662 @2.0 GHz, 512 GB RAM, 5.8 TB SSD, 4x Volta A100



# “Woody” cluster

Main workhorse for throughput and single-node jobs (Tier3)

- 176 nodes with 4 cores and high clock frequency (3.5/3.7 GHz) Intel Xeon E3-1240 v? processors
  - 64x Intel Skylake, 32 GB RAM
  - 112x Intel Kaby Lake, 32 GB RAM
- 70 nodes with 2x Intel Xeon Gold 6326 (32 cores total @2.9 GHz, 256 GB RAM)
- at least 960 GB local HDD/SSD
- Gbit network only



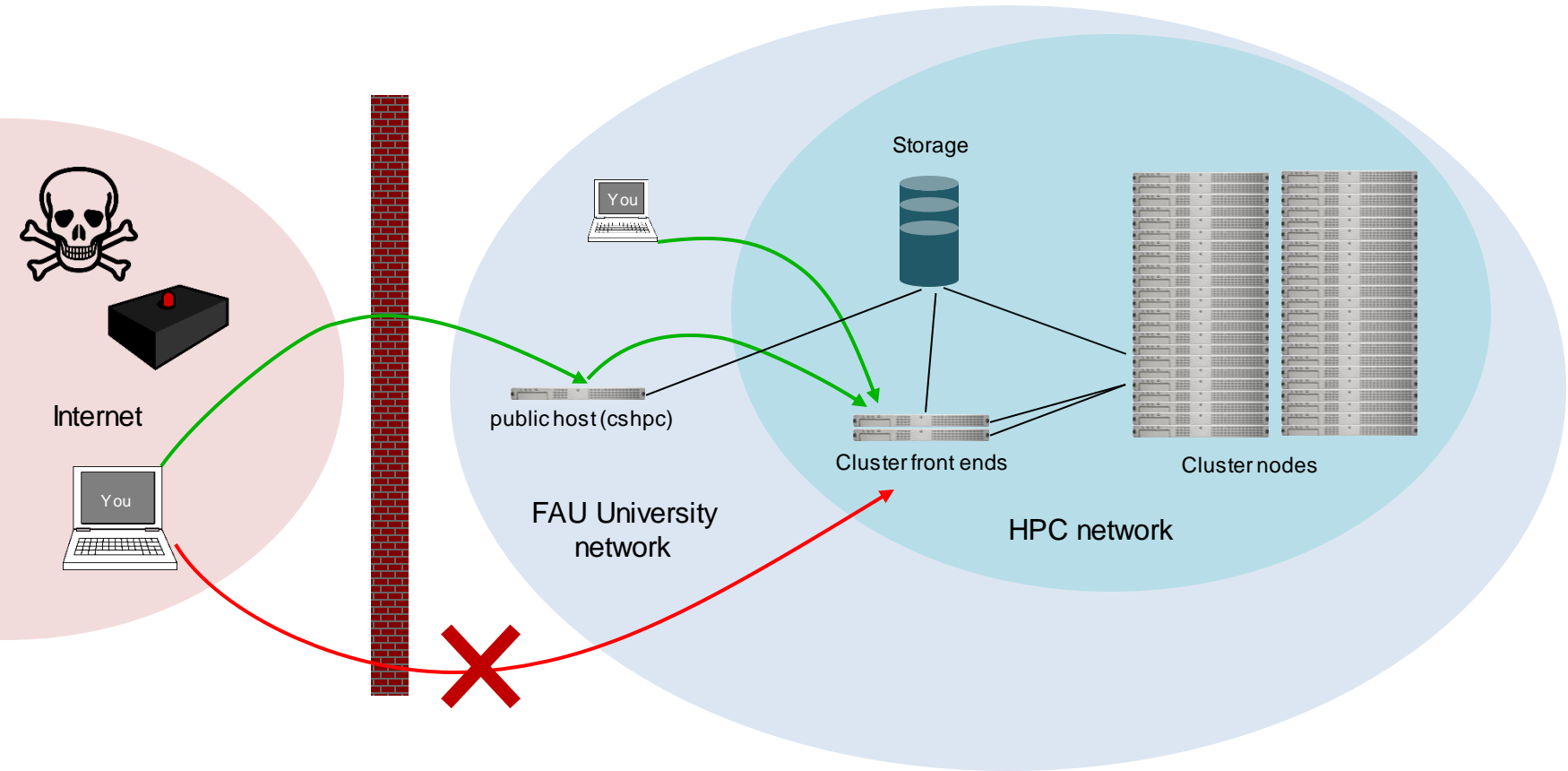
Which cluster(s) are you planning to use?



# Accessing HPC systems

<https://hpc.fau.de/systems-services/documentation-instructions/ssh-secure-shell-access-to-hpc-systems>

# Cluster access



# Cluster access

---

- Cluster frontends: only directly available from within FAU network or via IPv6
  - `meggie.rrze.fau.de`
  - `tinyx.nhr.fau.de` (for TinyGPU/TinyFat)
  - `woody.nhr.fau.de`
  - `alex.nhr.fau.de`
  - `fritz.nhr.fau.de`
- Access from outside FAU network via dialog server
  - `cshpc.rrze.fau.de`
  - The only machine with a public IPv4 address



# SSH – Secure Shell

---

- No GUI is available on the HPC systems. Basic knowledge of file handling, scripting, editing, etc. under Linux is required.

- SSH usage:

```
ssh hpcaccount@cshpc.rrze.fau.de
```

- How to log into HPC systems:

<https://youtu.be/J8PqWUfkCrl>

- In case of problems with login, send output of the following command to [hpc-support@fau.de](mailto:hpc-support@fau.de): `ssh -vv hpcaccount@cshpc.rrze.fau.de`

# Cluster access - HPC portal users

---

- Authentication only with SSH keys uploaded to HPC Portal, no passwords!
- Access to frontends from outside FAU:
  - directly via IPv6
  - by *proxy jump* over dialog server **cs

# hpc

.**rrze**.**fau**.**de****
- General documentation on SSH, how to generate an SSH key and example ssh-config:  
<https://hpc.fau.de/systems-services/documentation-instructions/ssh-secure-shell-access-to-hpc-systems>

# Secure Shell client programs

---

- Linux: OpenSSH available in any distribution
- Mac: OpenSSH available
- Windows
  - MobaXterm (<https://mobaxterm.mobatek.net/>)
  - OpenSSH via Command/PowerShell
  - Linux Subsystem for Windows

# Working with data

<https://hpc.fau.de/systems-services/documentation-instructions/hpc-storage/>

# File systems overview

Available file systems differ in size, redundancy and how they should be used

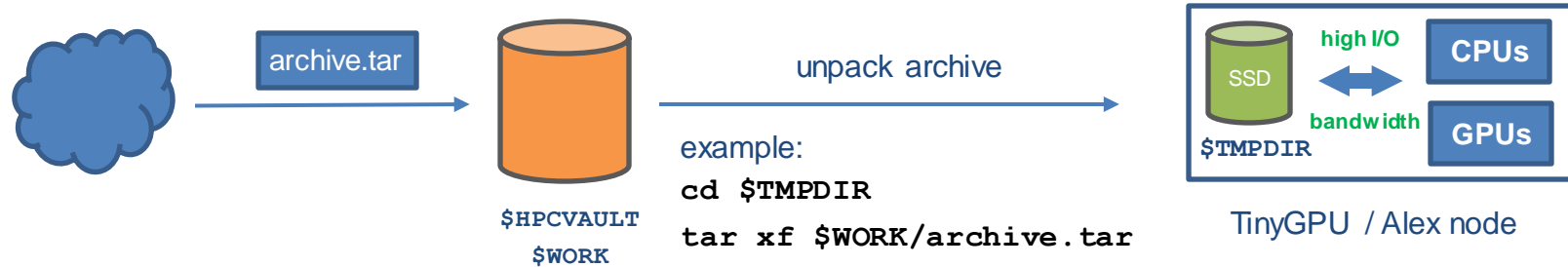
Mount point	Access	Purpose	Technology	Backup	Snapshots	Data lifetime	Quota
/home/hpc	\$HOME	Source, input, important results	NFS	YES	YES	Account lifetime	50 GB
/home/vault	\$HPCVAULT	Mid-/long-term storage	NFS	YES	YES	Account lifetime	500 GB
/home/{woody, saturn, titan, janus, atuin}	\$WORK	General-purpose, log files	NFS	NO	NO	Account lifetime	500 GB (Tier3) NHR project quota
/lxf /lustre	\$FASTTMP (fritz+alex)	High performance parallel I/O	Lustre via InfiniBand	NO	NO	High watermark	Only inodes
/???	\$TMPDIR	Node-local job-specific dir	SSD/ ramdisk	NO	NO	Job runtime	NO

# Working with large datasets containing small files

**Best case:** use a container file format (HDF5, Parquet, ...)

Alternative: pack small files into archive. Do not unpack archive to  
`$HOME/$HPCVAULT/$WORK`

Unpack files to node-local SSDs only and use them from there



More details:

<https://hpc.fau.de/files/2022/01/2022-01-11-hpc-cafe-file-systems.pdf>

<https://www.fau.tv/clip/id/40199>

# File system quotas

- File system may impose quotas on data volume and/or number of files
- Quotas may be set per user or per group (or both)
- Soft quota
  - Can be exceeded temporarily (7 days)
  - Turns into hard quota at end of grace period
- Hard quota:
  - absolute upper limit, cannot be exceeded

```
$ quota -s                # generic command

$ shownicerquota.pl      # only on RRZE systems
  Path                Used      SoftQ    HardQ    Gracetime  Filec    FileQ    FiHaQ    FileGrace
/home/hpc             5.7G    52.5G   104.9G    N/A        72K     500K    1,000K    N/A
/home/woody           112G    333.0G  499.5G    N/A        188K
```

# Data transfer

- `$HOME`, `$HPCVAULT`, `$WORK` are mounted on all HPC systems
- `scp` / `rsync` is used to transfer files from and to the outside

```
scp [options] source destination  
# source/destination: local-path | username@host:remote-path
```

```
# local file to remote  
# -r recurse into directories  
scp -r code unrz55@cshpc.rrze.fau.de:/home/woody/unrz/unrz55  
# remote file to local  
scp unrz55@cshpc.rrze.fau.de:results/output.dat .
```

- Additionally on Windows: WinSCP, MobaXTerm



# Software

<https://hpc.fau.de/systems-services/documentation-instructions/environment/>

What type of software are you using?



# Environment modules

---

- Linux standard distro packages are available on frontends and to some extent on compute nodes, but they might be outdated.
- Additional software is provided via environment modules
  - Compilers, libraries, commercial and open software
  - Installed on central server and available on all cluster nodes
- Environment modules are managed through the **module** command
- All **module** commands affect the current shell only!

# The module command

Show all available modules: `module avail`

```
$ module avail
----- /apps/modules/data/applications -----
amber/20p12-at21p11-impi-gnu                gromacs/2021.5-gcc11.2.0-impi-mkl
amber/20p12-at21p11-impi-intel             gromacs/2022.1-gcc11.2.0-ompi-mkl
amber/20p12-at21p11-openmpi-gnu-cuda11.5   gromacs/2022.1-gcc11.2.0-mkl-cuda
----- /apps/modules/data/compiler -----
gcc/10.3.0 gcc/11.2.0 gcc/12.1.0 intel/2021.4.0 intel/2022.1.0 nvhpc/22.1 nvhpc/22.2
----- /apps/modules/data/development -----
cuda/11.3.1                intelmpi/2021.4.0                openmpi/4.1.2-gcc11.2.0-cuda
cuda/11.4.2                intelmpi/2021.6.0                openmpi/4.1.2-intel2021.4.0-cuda
cuda/11.5.0                openmpi/4.1.2-gcc10.3.0-cuda     openmpi/4.1.2-oneapi2021.4.0-cuda
```

# The `module` command

Load a module: `module load <modulename>`

```
$ module load intel/2021.4.0
$ icc -V
Intel(R) C Intel(R) 64 Compiler Classic for applications running on Intel(R) 64, Version 2021.4.0 Build
20210910_000000
Copyright (C) 1985-2021 Intel Corporation. All rights reserved.
```

Display loaded modules: `module list`

```
$ module load openmpi/4.1.2-intel2021.4.0
$ module list
Currently Loaded Modulefiles:
1) intel/2021.4.0 <aL> 2) openmpi/4.1.2-intel2021.4.0
```

# Environment modules summary

<b>Command</b>	<b>What it does</b>
<code>module avail</code>	List available modules
<code>module whatis</code>	Shows verbose listing of all modules
<code>module list</code>	Shows which modules are currently loaded
<code>module load &lt;pkg&gt;/&lt;version&gt;</code>	Loads specific version of module package, i.e. adjusts environment
<code>module unload &lt;pkg&gt;</code>	Undoes what the load command did
<code>module help &lt;pkg&gt;</code>	Shows a detailed description of package
<code>module show &lt;pkg&gt;</code>	Shows which environment variables are modified and how

<https://hpc.fau.de/systems-services/documentation-instructions/environment/#modules>

# Using Python

- Use anaconda modules instead of system installation

```
$ module avail python
----- /apps/modules/modulefiles/tools -----
python/3.6-anaconda  python/3.7-anaconda(default)  python/3.8-anaconda
```

- Change default package installation path from **\$HOME** to **\$WORK**
- It might be necessary to configure a proxy to access external repositories
- Build packages in an interactive job on the target cluster (especially for GPUs)
- Install packages via conda/pip with **--user** option
- More details: <https://hpc.fau.de/systems-services/documentation-instructions/special-applications-and-tips-tricks/python-and-jupyter/>

# Running jobs

<https://hpc.fau.de/systems-services/documentation-instructions/batch-processing/>



# Interactive work on the front-ends

---

- The cluster frontends are for interactive work
  - Editing, compiling, preparing input,...
  - Front-ends are shared among all users, so be considerate!
  - Amount of compute time per binary is limited by system limits
    - E.g., after 1 hour of CPU time your process will be killed
  - MPI jobs are not allowed on front ends
- Submit computational intensive work to the batch system to be run on the compute nodes!
- Use interactive batch jobs for debugging and testing.

# Batch System

- Users can interact with the resources of the cluster via the “Batch system”
- “Batch jobs” encapsulate:
  - Resource requirements (number of nodes, number of GPUs, ...)
  - Job runtime (usually max. 24 hours)
  - Setup of runtime environment
  - Commands for application run
- Batch system will handle queuing of jobs, resource distribution and allocation
- Job will run when resources become available



# General structure of batch script

```
#!/bin/bash -l
#
#SBATCH --nodes=X
#SBATCH --time=hh:mm:ss
#SBATCH --mail-user=name@xyz.de
#SBATCH --job-name=jobname
#SBATCH --export=none
```

Interpreter directive; -l for initializing modules correctly

Resource requirements (number of nodes, MPI tasks,...)  
Job runtime  
Other options (job name, notifications, ...)

```
unset SLURM_EXPORT_ENV
module load <modulename>
```

Setup job environment, e.g.

- load modules
- stage data
- initialize virtual environment

application run

```
./application
```

# Example: Batch script for Fritz

```
#!/bin/bash -l
#
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=72
#SBATCH --time=06:00:00
#SBATCH --job-name=testjob_cpu
#SBATCH --export=NONE
```

Specifies number of compute nodes and MPI processes (tasks) per node

```
unset SLURM_EXPORT_ENV
module load openmpi
```

```
srun ${HOME}/bin/a.out -i inputfile -o outputfile
```

Launch MPI applications with `srun`

# GPU Jobs on TinyGPU / Alex

---

- Nodes are shared, GPUs are always exclusive
- Granularity is one GPU with a corresponding portion of CPU and main memory
- Request GPUs with **sbatch** option e.g.
  - `--gres=gpu:rtx3080:1` (to request a specific type)
  - `--gres=gpu:a100:1 --partition=a100` (necessary for V100 and A100 GPUs on TinyGPU)
- More details and examples:  
<https://hpc.fau.de/systems-services/systems-documentation-instructions/clusters/tinygpu-cluster>  
<https://hpc.fau.de/systems-services/systems-documentation-instructions/clusters/alex-cluster/>

# Example: Batch script for Alex

```
#!/bin/bash -l
#
#SBATCH --gres=gpu:a40:1
#SBATCH --time=06:00:00
#SBATCH --job-name=testjob_gpu
#SBATCH --export=NONE
```

Specifies number and type of GPU

```
unset SLURM_EXPORT_ENV
module load python
conda activate test-environment
```

```
cd $TMPDIR
tar xzf $WORK/large-archive-with-small-files.tar.gz
```

Unpack data directly to local SSD (\$TMPDIR)

```
python train.py
```

# Slurm batch job submission

```
hpcuser@fritz1$ sbatch script.sh
```

```
Submitted batch job 675329
```

```
hpcuser@fritz1:~ $ squeue -l
```

JOBID	PARTITION	NAME	USER	STATE	TIME	TIME_LIMI	NODES	NODELIST(REASON)
675329	multinode	testjob_cpu	hpcuser	RUNNING	0:06	06:00:00	4	f[0116,0120,0159,0264]

## Specific for TinyFat/TinyGPU:

- All jobs are submitted from the frontend `tinyx.nhr.fau.de`
- Wrapper scripts have to be used for all Slurm commands (e.g. `sbatch.tinygpu`, `sbatch.tinyfat`, `squeue.tinygpu`, ...)

# Interactive batch job with Slurm

- TinyGPU / Alex

```
iww042@tinysx$ salloc.tinygpu --gres=gpu:1 --time=01:00:00
```

```
iww042@alex1$ salloc --gres=gpu:a40:1 --time=01:00:00
```

- Meggie / Fritz:

```
iww042@meggie1$ salloc --nodes=1 --time=01:00:00
```

- Woody / TinyFat

```
iww042@woody$ salloc --ntasks=1 --time=01:00:00
```

```
iww042@tinysx$ salloc.tinyfat --cpus-per-task=10 --time=01:00:00
```



# Slurm documentation

---

- NHR@FAU
  - General: <https://hpc.fau.de/systems-services/systems-documentation-instructions/batch-processing/>
  - Cluster-specific: <https://hpc.fau.de/systems-services/systems-documentation-instructions/clusters/>
  - HPC Café on “Slurm - basics, best practices and advanced usage”:  
<https://hpc.fau.de/files/2022/04/2022-04-12-hpc-cafe-slurm.pdf>,  
<https://www.fau.tv/clip/id/41306>
- Official Slurm documentation
  - Separate documentation for every command and the available options:  
[https://slurm.schedmd.com/man\\_index.html](https://slurm.schedmd.com/man_index.html)
  - Slurm commands and their counterparts in different batch systems:  
<https://slurm.schedmd.com/rosetta.pdf>
  - Slurm tutorials: <https://slurm.schedmd.com/tutorials.html>

# Some Dos and don'ts

# Good practices

---

- Be considerate. Clusters are valuable shared resources that have been paid by the taxpayer.
- Use the appropriate amount of parallelism
  - Most workloads are not highly scalable
  - Best to run scaling experiments to figure out “sweet spot”
- Use the appropriate file system(s)
  - #1 mistake: Overload file servers by doing tiny-size, high-frequency I/O to FS
  - Delete obsolete data

# Good practices

---

- Check your jobs regularly
  - Are the results OK?
  - Does the job actually use the allocated nodes in the intended way? Does it run with the expected performance?
  - Check if your job makes use of the GPUs
    - Connect to compute node with `srun --pty --overlap --jobid <jobid> bash -l`
    - Use e.g. `nvidia-smi` to check GPU utilization
- Job Monitoring: <https://monitoring.nhr.fau.de/>
  - How to use it and what to look out for: <https://hpc.fau.de/files/2023/01/2023-01-10-HPC-Cafe-ClusterCockpit.pdf>

# Good practices

---

- Talk to co-workers who are more experienced cluster users; let them educate you
- Look at tips and tricks for various applications (e.g. example batch scripts):  
<https://hpc.fau.de/systems-services/systems-documentation-instructions/special-applications-and-tips-tricks/>
- Have a look at HPC Café talks from past events:  
<https://hpc.fau.de/systems-services/support/hpc-cafe/>

# Good practices

---

## When reporting a problem to NHR@FAU:

- Mail to [hpc-support@fau.de](mailto:hpc-support@fau.de) – this will immediately open a helpdesk ticket
- Provide as much detail as possible so we know where to look
  - “My jobs always crash” will not do
  - Cluster, JobID, file system, time of event, ...
  - Batch script, output files, ...
  - SSH problems: `ssh -vv <ssh_command>`



THANK YOU.

NHR@FAU

<https://hpc.fau.de>

[hpc-support@fau.de](mailto:hpc-support@fau.de)