Hessisches Kompetenzzentrum
für Hochleistungsrechnen

Lichtenberg Cluster | quick reference |

TECHNISCHE
UNIVERSITÄT
DARMSTADT

1 General

Project Proposal — Prerequisites for Access

An application, e.g. in form of a project proposal, is necessary to obtain a login and an allocation of compute time. Allocations are granted in three granularities: **S**mall, **M**edium and **L**arge. All proposals are reviewed regarding technical feasibility and, depending on the requested number of CPU-hours, a scientific review.

	Small	**M**edium	**L**arge
CPU hour quota (per year)	204 k CPUh	6.720 k CPUh	24.480 k CPUh
Technical review	✓	✓	✓
Scientific Review	✗	✓	Peer Review
Detailed project description	✗	✓	✓

Login Access Linux — Lichtenberg Cluster

The HRZ of TU Darmstadt operates multiple login-nodes. Current login-nodes are:

```
lcluster1.hrz.tu-darmstadt.de
lcluster2.hrz.tu-darmstadt.de
...
lcluster12.hrz.tu-darmstadt.de
```

Some of these may be offline due to maitenance. In such a case, simply choose another login-node for remote access. Please use the following command to login to one of these nodes:

Parameter	ssh [-X] <TU-ID@LOGIN-NODE>
`-X`	optional argument to enable X11 graphics forwarding
`TU-ID`	your personal user-identifier at TU Darmstadt
`LOGIN-NODE`	the name of the login-node.

To access these systems from a Windows platform some form of remote SSH software is required. An example for such a software is PuTTY, available at ”http://www.chiark.greenend.org.uk/XXX sgtatham/putty”, that can be used to log-in. WinSCP, available at `http://winscp.net` may be required to transfer data.

File Systems

Three different file-systems are available:

Directory	**Description**
`/home/<TU-ID>/`	Home volume: ”unlimited” lifetime, 10 GB limited space, regular backups, global access
`/work/scratch/<TU-ID>/`	Scratch area : 8 weeks lifetime, 100 TB space, no backup, global access, faster network, environment variable: `$HPC_SCRATCH` (defined at job runtime)
`/work/local/{...}/`	Local disk area: lifetime = job runtime, 100 GB per node, only local access, the fastest access, environment variable: `$HPC_LOCAL` (defined at job runtime)

Getting Help

If you have any questions or require help using the high performance computing resources provided by TU Darmstadt, please don’t hesitate to contact the:

- local administrative staff at `hhlr@hrz.tu-darmstadt.de`, or
- Hesse’s network of high-performance computing at `darmstadt@hpc-hessen.de`

The HPC team of the Lichtenberg High Performance Computer of TU Darmstadt offers local documentation, available at ”http://www.hhlr.tu-darmstadt.de/hhlr/index.en.jsp”.

Software Modules

Software can be made available using the shell command ”module”. At TU Darmstadt a great variety of software is made readily available and accessible using the module system.

Syntax:	module <command> [names]
`avail`	list available versions of modules, empty name argument lists all available modules
`whatis`	short descriptions of modules
`help`	detailed info about modules
`load | add`	load (one or several) modules
`unload | rm`	unload (one or several) modules
`list`	list currently loaded modules
`purge`	unload all modules

A complete list of available software can be queried through the module command: ”`module avail`”.

Lichtenberg Configurations

Different types of machine-configurations are available. Your computations in general must fit a single island. An island here represents a grouped set of compute nodes capable of intercommunication.

	Islands	Nodes	Cores	Memory	Specialty
MPI	15×	32	16	32 GB	-
	2×	32	16	64 GB	-
	1×	162	16	32 GB	-
	1×	84	24	64 GB	-
	16×	32	24	64 GB	-
	1×	23	24	64 GB	-
MEM	1×	4	64	1024 GB	-
	1×	4	60	1024 GB	-
ACC	1×	44	16	32 GB	2×Tesla K20X
	1×	2	24	64 GB	2×Tesla K40m
	1×	1	24	64 GB	2×Tesla K820-Dual

2 Job Control

The Lichtenberg High Performance Computer uses the resource manager Slurm, which manages both interactive and scripted use of compute resources. This system is comprised of a resource specification mechanism, based on comments in the script-file, and basic work management commands, shown below. A more complete documentation is available at the HHLR web-page in the subsection Cluster-usage → Slurm usage. Execution of a job is requires the use of Slurm commands, resource-scripts and runtime environment variables.

sbatch — Submit a new job

The sbatch command is the primary means to tap into the compute capabilities of the Lichtenberg HPC system.

Syntax:	sbatch [options] <script|executable>
`-a,--array=<indices>`	submit a job array
`-A <name>`	select project account for this job
`--mail-user=<address>`	report job changes by mail
`--mail-type=<arguments>`	comma separated list to send mail on startup <BEGIN>, end <END> or failure <FAIL>
`-n,--ntasks=<number>`	specifies the number of tasks required
`-c,--cpus-per-task=<ncpus`	number of cores per task; required for OpenMP
`-t <time>`	walltime-limit for the computation
`-C,--constraint=<feature>`	required features for job execution, e.g. sse, avx, or nvd for CUDA capable accelerators
`-d,--dependency=<list>`	job will only start, after jobs from dependency list have started.
`--exclusive`	exclusive use of resources
`--mem=<memory>`	specifies the total amount of required memory per node
`-J,--job-name=<job-name>`	specify job-name
`-o, --output=<filepath>`	redirect stdout to file
`-e, --error=<filepath>`	redirect stderr to file
`-i, --input=<filepath>`	read file as standard input
`-p <partition>`	submits the job directly to a specific partition

A ”task” is typically a single process; for multi-process computing, e.g. MPI, this represents the number of MPI processes used. Many more options are available. We refer to the man pages of sbatch for more information (”`man sbatch`”).

Slurm Constraints

Important resources available as constraints are:

`sse`	SSE capable machine
`avx`	AVX capable machine
`avx2`	report job changes by mail address
`nvd`	machine with any NVidia capable accelerator
`nvd2`	machine with an Nvidia K20Xm capable accelerator
`nvd4`	machine with an Nvidia K40m capable accelerator
`nvd8`	machine with an Nvidia K80 capable accelerator
`mpi`	machine from the MPI section
`mem1024g`	machine from the MEM section
`acc`	machine from the ACC section
`multi`	special jobs using multiple MPI islands

Slurm Commands — Other important commands

Command:	Function
`sjobs <jobid>`	shows detailed information about all your pending and running jobs or the job with the given ID
`sreport`	shows information about your accounting - separated for each project. The values are given in **core minutes**.

Parameters for Batch scripts

Slurm interprets special comments in the script-file as arguments to the ”sbatch” command. Most options that can used on the command line can be specified using this mechanism. One ore multiple options per line are possible. Example:

1.	`#!/bin/bash`	
2.	`#SBATCH -J helloworld`	name job ”hello world”
3.	`#SBATCH --mail -user=↩ mail@tu-darmstadt.de`	specify mail address
4.	`#SBATCH --mail -type=ALL`	send mail on all events
5.	`#SBATCH -e ~/job.err.%j`	specify error output file
6.	`#SBATCH -o ~/job.out.%j`	specify output file
7.	`#SBATCH --mem -per↩ -cpu=250`	require 250MB of memory per CPU
8.	`#SBATCH -t 00:05:00`	walltime limit of 5 minutes
9.	`#SBATCH -n 4`	require 4 taks
10.	`echo "This is Job ↩ $SLURM_JOB_ID"`	
11.	`module load gcc openmpi/gcc`	
12.	`mpirun hostname`	

Note, that the `%j` is replaced with the JobID by sbatch at submission time.

squeue — view information regarding submitted jobs

Syntax:	squeue [options]
`-i <seconds>`	report requested information
`-j <jobid>`	print information regarding <jobid>
`--start`	report expected start time and resource requirements for pending jobs
`-t <state>`	list all jobs in <state>

scancel — cancel a (running) job

Syntax:	scancel [options]
`<jobid>`	cancel the job with the <jobid>
`-u <username>`	cancel all jobs for an user
`-t PD -u <username>`	cancel all pending jobs of a user