DAY 1: SUPERCOMPUTING
Getting started on a supercomputer

2021-02-01 | Stefan Kesselheim | Helmholtz AI @ JSC

JÜLICH
Forschungszentrum

OUTLINE

Supercomputers

How to get in

Queueing Jobs

Tutorial Tasks

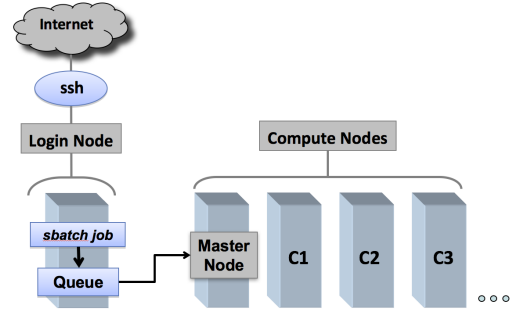
WHAT IS A SUPERCOMPUTER

Compute cluster	Many computers bound together locally
Supercomputer	A damn lot of computers bound together locally
Grid computing	Many clusters working as some kind of supercomputer
Cloud computing	Virtual machines in compute center(s)
High throughput computing	Many independent computations

ANATOMY OF A SUPERCOMPUTER

- Login Nodes: Normal machines, for compilation, data transfer, scripting, etc. No GPUs.
- Compute Nodes: Guess what :-)
- Special Purpose Nodes: Visualization, ...
- Network file system
- Scratch file system accessible from compute nodes
- Key stats:
  - Number of Nodes
  - CPUs, Number of Cores, Single-core Performance
  - RAM
  - Network: Bandwidth, Latency
  - Accelerators

Structure



SUPERCOMPUTER@JSC

JURECA DC

432 Nodes, Dual Socket AMD EPYC with 64 cores@ 2.25 GHz, 512/1025 GiB RAM, InfiniBand HDR 100, 48 nodes with 4x Nivida A100 with 40gb.

JUWELS

2271+240 Nodes, Dual Socket Skylake with 24 cores@ 2.7 GHz, 96, 196 GiB memory, InfiniBand EDR. 56 Nodes with 4x NVidia V100 16gb.

JUWELS Booster

936 Nodes, 2× AMD EPYC Rome 7402 CPU, 2× 24 cores, 2.7 GHz, 512, 1024 GiB memory, Mellanox HDR, 4x NVIDIA A100 with 40gb

DEEP-EST

Modular Prototype. Cluster Module, 50x2x Xeon 12core @3.2 GHz. Extreme Scale Booster, 75x Xeon 8Core@2.5 GHz w 1x Nvidia V100 (compact!). Data Analytics Module with 1xNvidia V100 and 1x Intel FPGA Stratix10.

JUST

- 52 PB total storage
- `$HOME`: Minimal, only dotfiles
- `$SCRATCH`: 9 PB, Working storage, deleted after 90 days.
- `$DATA`: 14 PB, Large scale file storage, not available from compute nodes.
- `$FASTDATA`: 9 PB, Large scale file storage that must be continuously accessed.
- `$PROJECT`: 2.3 PB, Project data .
- Peak bandwidth up to 400 GB/sec.
- Data is shared among all supercomputers.

AUTHENTIFICATION AND AUTHORIZATION

- Create public/private key pair
- Register public key in JuDoor
- Make sure to use that key.


```
ssh -i path/to/keyfile
surname1@juwels.fz-juelich.de
```

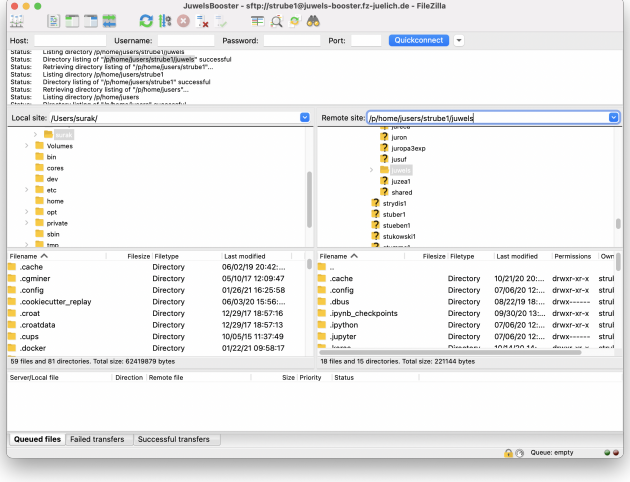
Attention: You will need to restrict IP ranges/DNS areas. Check tutorial.

FILE ACCESS

Copying files to a remote machine

```
scp -i path/to/key <source> <target>
```

Or use a graphical tool, e.g. Filezilla:



FILE ACCESS (CONT.)

Your home directory is tiny.
Project directory: `/p/project/training2004`

```
cd /p/project/training2004
mkdir $USER
cd $USER
git clone https://gitlab.version.fz-juelich.de/MLDL_FZJ/juhaicu/jsc_internal/superhaicu/
    shared_space/teaching/intro_scalable_dl_2021/course-material
```

SUPERCOMPUTER USAGE MODEL

- Using the the supercomputer means submitting a job to a batch system.
- Job scheduling according to priorities. The jobs with the highest priorities will be scheduled next.
- No node-sharing. The smallest allocation for jobs is one compute node (4 GPUs).
- Compute time allocation is based on compute projects. For every compute job, a compute project pays.
- Data projects allocate large amounts of storage, but no compute.
- To enable fair share, only relatively short job runs (24h) are allowed. Please implement checkpointing (or make your code fast enough).
- Solution for long-running tasks: Job arrays.

Query Project quotas:

```
jutil project show -project training2004
q_cpuquota
```

SLURM I: EXAMPLE SUBMISSION SCRIPT

```
#!/bin/bash
#SBATCH --nodes=1                # How many nodes?
#SBATCH -A training2004          # Who pays for it?
#SBATCH --partition booster      # Where does the code run?
#SBATCH --gres gpu               # Not required on booster
#SBATCH --time=00:15:00          # How long?
#SBATCH -o output.txt
#SBATCH -e error.txt

source /p/project/training2004/course2021_working_environment/activate.sh
cd /p/project/training2004/${USER}
srun --ntasks-per-node=4 python my_script.py
```

```
sbatch my_script.sh
```

SLURM II: RESOURCES, COMPUTE BUDGET

Partition

Every node belongs to at least one partition of the cluster.

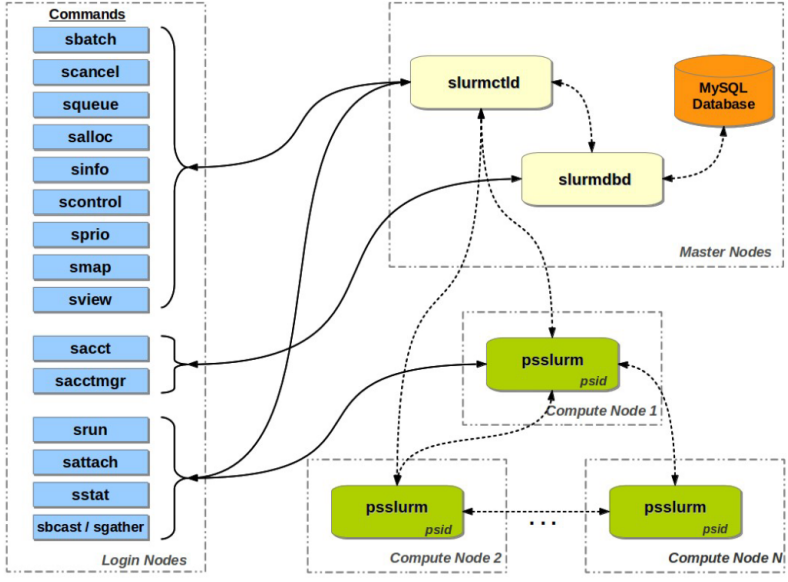
Queue

Jobs wait in a queue. Queues are executed in one or several partitions.

Generic Resources (GRES)

Jobs can request generic resources. One option is GPUs. RAM requirements or software licenses are also possible

SLURM III


Commands
sbatch
scancel
squeue
salloc
sinfo
scontrol
sprio
smap
sview
sacct
sacctmgr
srun
sattach
sstat
sbcast / sgather
Login Nodes
slurmctld
MySQL Database
slurmdbd
Master Nodes
psslurm
psid
Compute Node 1
psslurm
psid
Compute Node 2
psslurm
psid
Compute Node N

MODULE SYSTEM

All kind of software is already installed in modules.

Compiler/GCCcore/9.3.0

Autotools/20200321	Nsight-Compute/2020.3.0	h5py/2.10.0-serial-Python-3.8.5
Bazel/3.6.0	Nsight-Systems/2020.3.1	hwloc/2.2.0
CFITSIO/3.490	Nsight-Systems/2020.4.1	jemalloc/5.2.1
CMake/3.18.0	Nsight-Systems/2020.5.1	libvpx/1.9.0
CVS/1.11.23	Octave/6.1.0-nompi	likwid/5.0.2
Cling/0.7	OpenCV/4.5.0-Python-3.8.5	magma/2.5.4
CubeGUI/4.5	OpenEXR/2.5.2	meld/3.21.0-Python-3.8.5
Doxygen/1.8.18	OpenGL/2020	memkind/1.10.1
Eigen/3.3.7	OpenJPEG/2.3.1	nano/5.5
Emacs/27.1	PAPI/6.0.0	netCDF-C++4/4.3.1-serial
FriBidi/1.0.9	PCRE2/10.34	netCDF-Fortran/4.5.3-serial
GDB/10.1	Perl/5.32.0	netCDF/4.7.4-serial
GEOS/3.8.1-Python-3.8.5	Pillow-SIMD/7.0.0.post3-Python-3.8.5	netcdf4-python/1.5.4-serial-Python-3.8.5
GMP/6.2.0	PostgreSQL/12.3	numba/0.51.1-Python-3.8.5
GPicView/0.2.5	PyOpenCL/2020.2.2-Python-3.8.5	parallel/20201122
GSL/2.6	PyQt5/5.15.1-Python-3.8.5	pyqrcr/2.6.1.post1-Python-3.8.5
Grace/5.1.25	PyTorch/1.7.0-Python-3.8.5	re2c/1.3
Graphviz/2.44.1	Python-Neuroimaging/2020-Python-3.8.5	scikit/2020-Python-3.8.5
HDF5/1.10.6-serial	Python/3.8.5	tbb/2020.3
ICA-AROMA/0.4.4-beta-Python-3.8.5	R/4.0.2-nompi	tcsh/6.22.02
ITK/5.1.2-nompi-Python-3.8.5	Ruby/2.7.1	texlive/20200406
ImageMagick/7.0.10-25	Rust/1.47.0	tmux/3.1b
Julia.CUDA/2.0.2	SciPy-Stack/2020-Python-3.8.5	torchvision/0.8.1-Python-3.8.5
Julia/1.5.2	Shapely/1.7.1-Python-3.8.5	trimesh/3.8.11-Python-3.8.5
Jupyter/2020.2.5-Python-3.8.5	Singularity-Tools/2020-Python-3.8.5	unzip/6.0
JupyterCollection/2020.2.5	Subversion/1.14.0	xpra/4.0.4-Python-3.8.5
LLVM/10.0.1	Tcl/8.6.10	zsh/5.8
MPFR/4.1.0	TensorFlow/2.3.1-Python-3.8.5	
Mercurial/5.5.2-Python-3.8.5	TotalView/2020.1.13	
Meson/0.55.0-Python-3.8.5	VirtualGL/2.6.4	
NCCL/2.8.3-1-CUDA-11.0	Voro++/0.4.6	
NSPR/4.25	X11/20200222	
NSS/3.51	XServer/1.20.9	
Ninja/1.10.0	ZeroMQ/4.3.3	
Nsight-Compute/2020.1.2		
Nsight-Compute/2020.2.0		

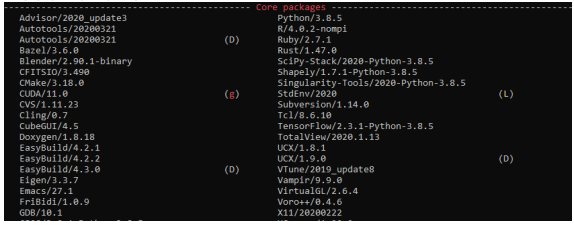
https://apps.fz-juelich.de/jsc/llview/juwels_modules_booster/

MODULE SYSTEM

Commands

- `module avail`
- `module purge # unload everything`
- `module load`
- `module spider`

`module avail`



`module spider nano`

```
[kesselheim1@jwlogin07 ~]$ module spider nano

----------------------------------------------------------------------------
  nano: nano/5.5
----------------------------------------------------------------------------
    Description:
      GNU nano is a small and friendly text editor. Besides basic text editing, nano offers features like
      undo/redo, syntax coloring, interactive search-and-replace, auto-indentation, line numbers, word
      completion, file locking, backup files, and internationalization support.
```

`module load nano`

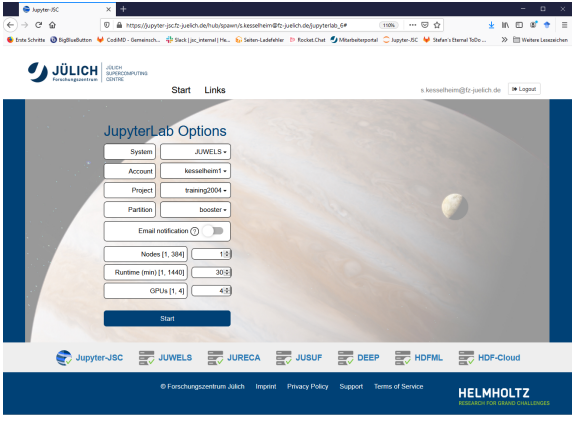
For this Training:

`source /p/project/training2004/course2021_working_environment/activate.sh`

`https://gitlab.version.fz-juelich.de/MLDL_FZJ/juhaicu/jsc_internal/superhaicu/shared_space/teaching/intro_scalable_dl_2021/course2021_working_environment`

JUPYTER-JSC

`https://jupyter-jsc.fz-juelich.de/`



- Add JupyterLab
  - Machine (JUWELS)
  - Compute Project (training2004)
  - Partition (Login or Booster)
  - Number of Nodes
  - Runtime
  - Start Jupyterlab
  - Create Symlink: In a terminal, type `ln -s /p/project/training2004/ ~/training2004`
  - Have fun

TUTORIAL TASKS

1. Connect to Juwels Booster by SSH
2. Run your first interactive Slurm Job
3. Queue your first batch job
4. Load the course compute environment
5. Bonus: Login into Jupyter-JSC