JURECA
An Overview

2017-11-23 | Philipp Thörning | HPS group @ JSC
JURECA

- Jülich Research on Exascale Cluster Architectures
- Project partners: T-Platforms, ParTec
- FZJ next-generation general purpose production system
  - NIC, VSR and commercial projects
  - Replaces the decommissioned JUROPA system
- Intended for mixed capacity and capability workloads
  - Designed with big-data science needs in mind
- Cluster architecture
  - Commodity hardware
  - Largely based on a open-source software stack
JURECA hardware overview

- Dual-socket Intel Xeon **E5-2680 v3** Haswell nodes
  - 24 cores @ 2.5 GHz
- NVIDIA K40 and K80 GPUs
- 128/256/512 GiB memory per node (DDR4 @ 2133 MHz)
- 1884 compute nodes ⇒ 45,216 cores
  - **1800** TFps + 430 TFps peak performance
- InfiniBand **EDR** (100 Gbps per link and direction)
  - Full fat tree topology
- 100 GiBps I/O bandwidth to central GPFS storage cluster
JURECA software overview

- **Operating system:** CentOS 7.X
- **Batch system** based on **Slurm/Parastation**
  - Workload management and UI ⇒ Slurm
  - Resource management ⇒ Parastation (**psid** + **psslurm**)
- **Programming environment:**
  - GNU Compilers, Intel Professional Fortran, C/C++ Compilers, OpenMP (Intel, GNU)
  - CUDA
  - Parastation MPI (based on **MPICH3**), Intel MPI, MVAPICH2-GDR
  - Optimized mathematical libraries (Intel Math Kernel Library, etc.) and applications (**/usr/local**)
JURECA node types

- **Login nodes**
  - 256 GiB memory
  - Intended for interactive work: development, compilation, interactive pre- and post-processing
  - CPU time limits (2 hours)

- **Standard/slim nodes**
  - 128 GiB memory
  - Default for batch jobs *(batch partition)*
  - Smallest allocation is one node, charge based on wall-clock time
  - No direct login ⇒ Interactive sessions with `salloc` and `srun --forward-x --pty`
JURECA node types (2)

- Fat (type 1): 256 GiB memory
  - \(--\text{gres}=\text{mem256}\)
  - Included in \textit{batch}

- Fat (type 2): 512 GiB memory
  - \(-p \text{ mem512} --\text{gres}=\text{mem512}\)
  - In a separate \texttt{mem512} partition due to lower node performance

- Fat (type 3): 1 TiB memory
  - \(-p \text{ mem1024} --\text{gres}=\text{mem1024}\)
  - Intended for memory-intense, lowly scalable pre- and post-processing tasks
JURECA node types (3)

- **Visualization nodes**
  - ≥512 GiB memory (2 nodes with 1 TiB), 2× NVIDIA K40
  - `-p vis --gres=gpu:[1-2]`
  - `--gres=mem1024` for large memory nodes
  - Client-server visualization requires `ssh` tunneling

- **GPU nodes**
  - 128 GiB memory, 2× NVIDIA K80 (4 visible GPUs per host)
  - `-p gpus --gres=gpu:[1-4]`
# JURECA node quantities

<table>
<thead>
<tr>
<th>Node type</th>
<th>#</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard/Slim</td>
<td>1605</td>
<td>24 cores, 128 GiB</td>
</tr>
<tr>
<td>Fat (type 1)</td>
<td>128</td>
<td>24 cores, 256 GiB</td>
</tr>
<tr>
<td>Fat (type 2)</td>
<td>64</td>
<td>24 cores, 512 GiB</td>
</tr>
<tr>
<td>Accelerated</td>
<td>75</td>
<td>24 cores, 128 GiB, 2× K80</td>
</tr>
<tr>
<td>Login</td>
<td>12</td>
<td>24 cores, 256 GiB</td>
</tr>
<tr>
<td>Visualization (type 1)</td>
<td>10</td>
<td>24 cores, 512 GiB, 2× K40</td>
</tr>
<tr>
<td>Visualization (type 2)</td>
<td>2</td>
<td>24 cores, 1 TiB, 2× K40</td>
</tr>
</tbody>
</table>
JURECA: Accessing the system

- Access with SSH keys
  - Recommendation: 2048 bit RSA
    \texttt{(ssh-keygen -t rsa -b 2048)}
  - Protection of private key with non-trivial pass phrase is mandatory!
- CPU time limits apply
  - Soft limit: 2 hours
JURECA: Accessing software (hierarchical modules)

1. List available toolchains
   
   \$ module avail

2. Load the desired compiler and MPI
   
   \$ module load <Compiler> <MPI>

3. List available packages based on current list of modules
   
   \$ module avail

4. Load additional applications/libraries
   
   \$ module load <module name>

Search for an application/library

\$ module spider <name>
JURECA: Filesystems

- All user filesystems mounted from the central GPFS fileserver Jülich Storage Cluster (JUST)
  - Exception: Node local /tmp filesystem (ext4), $O(10$ GiB)
- $\$HOME$
- $\$WORK$
- $\$ARCH$
JURECA: Filesystems (\texttt{$\$HOME$})

- **Purposes**
  - Storage of regularly used files and applications
  - Storage of smaller files used for current computation
- **Daily backup**
- **Quota:** Max. 10 TiB disk space and max. 3 mio. inodes per group

```
$ q_dataquota [-l]
```

- Careful with \texttt{chmod -R}!
  - **Safer alternative:** Access control lists (ACL)
JURECA: Filesystems ($\texttt{WORK}$)

- **Purpose**
  - Storage of large files used or generated by the current computation
- **Scratch filesystem with highest performance**
- **No backup**
- **Files will be deleted 90 days after last usage!**
  - \texttt{atime} is not updated for performance reasons
- **Quota:** Max. 30 TiB disk space and max. 4 mio. inodes per group

\begin{verbatim}
$\texttt{q_dataquota } [-l]
\end{verbatim}

- Copy important files to $\texttt{HOME}$ or $\texttt{ARCH}$
JURECA: Filesystems (\$ARCH)

- **Purpose**
  - Storage of large, not recently used, files
- Not available on compute nodes!
- Daily backup
- Files migrated to tapes
- **Quota**: No space quota and max. 2 mio. inodes per group
- **Usage recommendations**
  - `tar/zip` many small files
  - Do not touch/move files
JURECA: Sketch
JURECA: Fat-tree InfiniBand topology
JURECA: NUMA architecture
JURECA: Multicore

Core 0  Core 1  Core 2  Core 3
Core 4  Core 5  Core 6  Core 7
Core 8  Core 9  Core 10 Core 11
JURECA: Hyper-Threading Technology

Diagram showing the layout of HWTs 0 to 23.
JURECA: AVX 2.0 ISA extension

- AVX 2.0 ISA extension ⇒ Two 256-bit wide multiply-adds per cycle!
Further information

- **motd**: Message of the day
  - Information about preventive and emergency maintenances
  - Information about system configuration changes

- On-line documentation
  - [http://www.fz-juelich.de/ias/jsf/jureca](http://www.fz-juelich.de/ias/jsf/jureca)

- User support at FZJ
  - sc@fz-juelich.de
  - Phone: 02461 61-2828