



UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386



Tools 2026 – Slurm Cluster System

Shigehiko Schamoni
11.06.2026

Overview

- 1 Introduction
 - Overview
 - Resource Management
- 2 Slurm Workload Manager
 - Slurm Basics
 - Slurm Workflow
- 3 Slurm Usage
 - Accessing Slurm
 - Command Overview
 - Job Execution
 - Interactive Mode
 - Array Jobs
 - Tips and Tricks
- 4 bwUniCLuster
 - bwUniCLuster
- 5 Final Comments
- 6 Exercise

What is a Cluster?

A *Computer Cluster* is:

- A set of computers that works together.
- Can be viewed as a single system.
- Tasks are run on several nodes in parallel.
- Controlled and scheduled by software.
- Properties of a cluster:
 - Sharing of resources
 - Management software s.t. different tasks don't interfere
 - Loads of RAM, CPU cores, possibly GPU resources, etc.

Why to Use a Compute Cluster

- Processes require more resources than a single server or a workstation provides
- Parallel processing of large amounts of data
- Huge models that require distributed computation (e.g. simulations)
- Tuning of model parameters (e.g. machine learning)
- Modern use-cases:
 - Cloud-computing (e.g. AWS)
 - Cloud services (e.g. heiBOX)

Clusters for ZITI

- bwUniCluster
 - Accessible to university members after registration
 - <https://wiki.bwhpc.de/e/BwUniCluster2.0>
 - <https://wiki.bwhpc.de/e/BwUniCluster3.0>
- bwForCluster
 - More complex application process: needs an entitlement
 - <https://wiki.bwhpc.de/e/Registration/bwForCluster>
- ZITI's research groups that have their own clusters:
 - HAWAII-Cluster
 - ASC-Cluster
 - NCA-Cluster

Resource Management

- A cluster is used by multiple users at the same time.
- Efficient resource allocation is crucial:
 - Which processes are running where?
 - How much resources does a process need?
 - Which processes need exclusive access to resources (compute accelerators, graphics cards, etc.)?
- Common to all ZTI- and BW-clusters:
 - Rely on Slurm batch system
 - <https://slurm.schedmd.com/>

Our Tools Test Cluster

- We have a small test-cluster for the Tools 2026 lecture.
- The Tools2026 test-cluster is managed by Slurm.
- To use the cluster, log in to the login-node:

```
$ ssh UniID@ziti-peac.ziti.uni-heidelberg.de
```

- ZITI-pool's home-folder is available.
 - Additional software in `/opt/tools`
- The login node (`ziti-peac`) is for managing tasks only,
never run compute intensive tasks there!

What is Slurm?

- Slurm stands for:
Simple Linux Utility for Resource Management
- Slurm is a *workload manager* to administer a cluster's resources
- Slurm Terminology
 - Node : a single computer entity
 - Partition : a group of nodes
 - Job : set of resources assigned to a user for a specific task
 - Job step : single process that is part of a job
- More Information: <https://slurm.schedmd.com/>

Slurm Entities Example

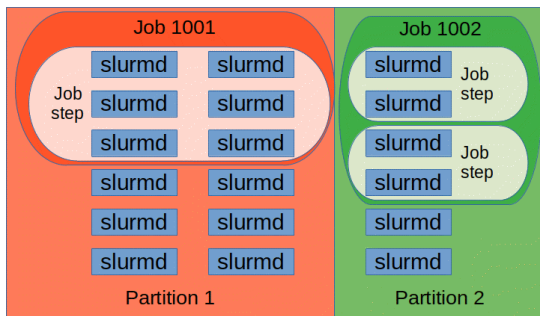


Figure: <https://slurm.schedmd.com/overview.html>

- `slurmd`: a daemon running on a compute node
- We have multiple partitions.
- Job steps might run on several nodes.

Slurm Architecture in Detail

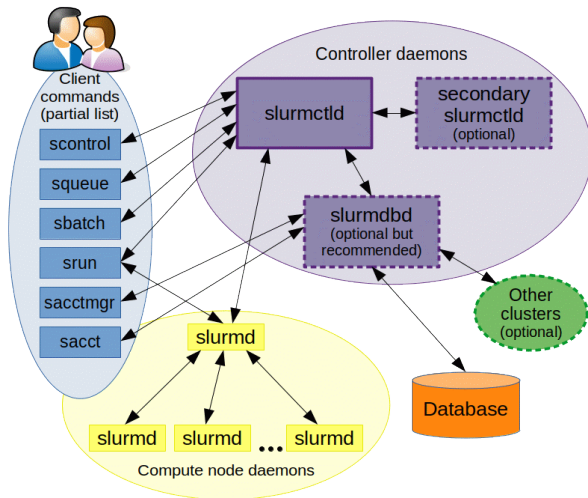


Figure: <https://slurm.schedmd.com/overview.html>

Slurm Workflow

- 1 User requests resources from a certain partition for a job
- 2 Resources are allocated to the user's job
 - Every partition has a queue.
 - If the required resources are occupied we have to wait until they are freed.
- 3 A job can be divided in multiple *job steps*:
 - Steps use allocated resources.
 - Job steps typically run in parallel.
- 4 Allocated resources are freed:
 - When all steps were processed.
 - When the age of the job exceeds a time limit.

Login via SSH (I)

- Secure shell (SSH) is the most common way to connect to remote machines.
- Linux, Mac, and BSD users have the command `ssh` installed by default.
- Windows users can use WSL or PuTTY¹
- The command `ssh` expects a username and a hostname separated by an @-sign:

```
$ ssh <user>@zitigate.ziti.uni-heidelberg.de
```
- The example above creates an ssh-connection to a gate server which enables access to ZITI's compute resources.
- The ziti-gate is very restricted and should thus only be used as a *jump host*.

¹<https://www.chiark.greenend.org.uk/~sgtatham/putty/>

Login via SSH (II)

- To jump to a different server, run `ssh` on the gate:
`ssh <user>@<hostname>` If `hostname` is part of the ZITI domain, you don't need the fully qualified domain name (FQDN).
- The jump host configuration can be automated by editing the client's config file `$HOME/.ssh/config`:

```
Host zitigate
    HostName zitigate.ziti.uni-heidelberg.de
    User <UniID>
Host headnode
    HostName ziti-peac
    User <UniID>
    ProxyJump zitigate
```

- Now one can directly connect to `ziti-peac` by
`$ ssh headnode`
- Notes: add an ssh-key to avoid being locked out by mistyped passwords.
- When connecting from the ZITI-pool, the `ProxyJump`

Basic Slurm Commands

- `sbatch <batch-script>`: Submits a batch script to Slurm for processing.
- `squeue`: Shows information about your job(s) and all other jobs queue.
- `srun <parameters>`: Run jobs interactively on the cluster.
- `scancel <job-id>`: Cancels a (queued) job.
- `sacct`: Shows information about current and previous jobs.
- `sinfo`: Returns information about the resources on available nodes of the cluster.

Preparation

- We have a small test-cluster for the Tools2026 lecture!
- To use this cluster, log in to the login-node:

```
$ ssh UniID@ziti-peac.ziti.uni-heidelberg.de
```

- PC-pool's home-folder is available
 - Additional software in `/opt/tools`
- Necessary steps for using the Tools2026 cluster:
 - 1 Add your public key to your `.ssh/authorized_keys` file:

```
$ cd ~/.ssh
```

```
$ cat id_ed25519.pub >> authorized_keys
```

- 2 Acquire the host keys of each compute node:

```
$ ssh UniID@ziti-peac
```

```
$ for i in {01..04}; do ssh ziti-peac$i hostname; done
```

On the question if you want to continue connecting answer “yes”.

- Note: The login-node `ziti-peac` is for managing tasks only, *never* run compute intensive tasks there!

Slurm Scripts

- Jobs are usually run using a batch script file:

```
#!/usr/bin/env bash

#SBATCH --partition=tools1
#SBATCH --ntasks=1
#SBATCH --job-name=YOUR_JOB_NAME
#SBATCH --output=%x_%j.txt
#SBATCH --mail-user=YOUR_UNIID@stud.uni-heidelberg.de
#SBATCH --mail-type=ALL

srun hostname
```

- A Slurm script consist of several components:
 - *Shebang*: This is always the first line of your script and defines the interpreter (`bash`).
 - *Resource Request* : The `#SBATCH` directives inform Slurm about the name of the job, resource requirements, and other parameters to be used for processing the job.
 - *Dependencies*: Software that the project depends on has to be loaded (e.g. a certain virtual environment for Python).
 - *Job Steps*: The list of tasks to be carried out.
- The script file can have any name, e.g. `first.sh`.

Execution (I)

- Before running a job, it is often useful to get an overview on the cluster.
- Use the `sinfo` command to get the cluster's current state:

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
all*      up    30-00:00:0    1  drain ziti-peac01
all*      up    30-00:00:0    1   mix  ziti-peac02
all*      up    30-00:00:0    2  idle  ziti-peac[03-04]
tools1    up    30-00:00:0    1  drain ziti-peac01
tools1    up    30-00:00:0    1   mix  ziti-peac02
tools1    up    30-00:00:0    1  idle  ziti-peac03
tools2    up    30-00:00:0    2  idle  ziti-peac[03-04]
```

- `idle`: the node is empty.
- `mix`: there are jobs running on this node.
- `drain`: there is a problem with this node. Already running tasks on this node will still execute, however, new tasks are not scheduled on this node until the issue is resolved.
- `down`: the node is not available.

Execution (II)

- Once your script is ready, add it to the queue:

```
$ sbatch first.sh
```

- This will return a job id and the job will be put into the queue corresponding to the specified partition.
- You can retrieve runtime information on your job(s) with:

```
$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	ODELIST(REASON)
32	tools1	SLURM_TE	vo001	R	0:02	1	ziti-peac02

- If you need to cancel a job (running or queued) use:

```
$ scancel <job-id>
```

Slurm Generic RESources (GRES)

- A mechanism to define and schedule arbitrary Generic RESources (GRES)
- Special built-in features for specific GRES types are available
 - Graphics Processing Units (GPUs)
 - CUDA Multi-Process Service (MPS) devices
 - ...
- Jobs will not allocate any generic resources unless requested:
 - `--gres:<specification>` : Generic resources required per node
 - `--gpus:<count>` : GPUs required per job
- Example:
 - `--gres=gpu:2` : Requests two gpus of any type for the job.
- There are no special GRES defined on the Tools2026 test cluster, all jobs are CPU only.

More SBATCH options

- There are a lot of possible `sbatch` command line options:
<https://slurm.schedmd.com/sbatch.html>
- All options can be added as `#SBATCH` directives in scripts.
- Some frequently used options:
 - `--nodelist={node01,node02,...}` : Run job only on specified nodes. Make sure your `--nodelist` and `--partition` options fit together!
 - `--nodes=N` : Requests a minimum of N nodes to be allocated to this job.
 - `--time=<days>-<hours>:<minutes>:<seconds>` : Specifies the maximum run time of your job, after which it will be killed.
 - `--mem=<size>[units]` : Specifies the real memory required per node. Units can be [K|M|G|T].

Interactive Mode

- During development, it is often necessary to execute quick experiments.
- Slurm provides an *interactive mode* for this:
 - You first run `salloc` to allocate resources as you would normally do with `#SBATCH` directives.
 - Once the command returns, you can use `srun` to execute commands on the allocated nodes.
- Example:

```
$ salloc --partition=tools2 --time=01:00:00 --ntasks=1  
salloc: Granted job allocation 123
```

Execute code on the allocated node(s) with `srun`:

```
$ srun hostname  
ziti-peac03
```

Return requested resources with `scancel`:

```
$ scancel 123  
salloc: Job allocation 123 has been revoked.
```

Slurm Array Jobs (I)

- Array jobs are convenient in case of data parallelism:
 - Large amounts of input data in the same format.
 - Processing is independent of previous data.
- Array Jobs are a simple mechanism to distribute work to several nodes:
 - Data is divided into numbered partitions (files).
 - Job is defined by an `sbatch` script.
 - The job is scheduled with the `--array` option.
 - Slurm starts the script multiple times.

Slurm Array Jobs (II)

- Example call of a Slurm Array Job:

```
$ sbatch --array 1-10 second.sh
```

- Requires `sbatch -script second.sh`
- Runs the given script for each number between 1 and 10 (10 times)
- The environment variable `$SLURM_ARRAY_TASK_ID` is set and can be accessed within the script.

Slurm Array Jobs (III)

- Assuming we have our data split into files with the following naming scheme:

- data-01.dat
- data-02.dat
- data-03.dat

- And we want to process the data using a binary named `/opt/tools/bin/dataprocessor`.
- Each data split analysis is independent of the others.
- A possible batch script example:

```
#!/bin/env bash
# SBATCH --ntasks=1
# SBATCH --array=1-3
srun /opt/tools/bin/dataprocessor data-${SLURM_ARRAY_TASK_ID}.dat
```

- The array definition can be provided on the command line with `sbatch` or within the script (example above).

Port forwarding for Jupyter Notebooks (I)

- Several applications provide web interfaces (e.g., Jupyter Notebooks or TensorBoard).
- Usually cannot be directly accessed (e.g. firewall restrictions).
- SSH Port Forwarding can solve this:
 - Run the application on a specific compute node.
 - Create an ssh-tunnel to the compute node.
 - Access the port on the remote node on your local machine.

Port forwarding for Jupyter Notebooks (II)

- Example sbatch-script:

```
#!/usr/bin/env bash

#SBATCH --partition=tools1
#SBATCH --odelist=ziti-peac01

#SBATCH --ntasks=1

#SBATCH --job-name=jupyter_test
#SBATCH --output=%x_%j.txt
#SBATCH --mail-user=UniID@stud.uni-heidelberg.de
#SBATCH --mail-type=ALL

srun hostname
srun jupyter notebook --no-browser --ip 0.0.0.0 --port 8888
```

The script print out the hostname as soon as it starts.

- Open the ssh-tunnel on your local machine:

```
$ ssh -L 8888:ziti-peac01:8888 UniID@ziti-peac -N
```

- Then, open `http://localhost:8888` in a browser window.
- The tunnel is open until the ssh-session is terminated.

Unbuffered Output

- Sometimes you need to monitor the current out of your job.
- However, the file specified by the `--output` parameter is empty although the job has started.
- This is often due to buffering of the output of `stdout`.
- Switching off buffering helps:
 - Activate unbuffered output on Slurm's side: `srun -u`
 - Additionally force unbuffered output in the application's side, e.g. use `python -u`.

More Useful Tips

- Show Slurm configuration:

```
$ scontrol show config
```

Documentation of all parameters at

<https://slurm.schedmd.com/slurm.conf.html>

- The official Quickstart guide:

<https://slurm.schedmd.com/quickstart.html>

- A very good introduction to Slurm commands:

<https://hpc.nmsu.edu/discovery/slurm/commands/>

A Real Computer Cluster: bwUniCluster



Figure: <https://wiki.bwhpc.de/e/BwUniCluster3.0>

- The *bwUniCluster 3.0+KIT-GFA-HPC 3* is a high-performance computer cluster system.
- Jointly run by Baden-Württemberg's Universities and Universities of Applied Sciences.
- Available for general purpose and teaching.
- Located at the Scientific Computing Center (SCC) at Karlsruhe Institute of Technology (KIT).

Access to the bwUniCluster

- You need an entitlement from Heidelberg University.
- This entitlement is required to get access to the bwUniCluster.
- Log in with your UniID at:
`https://heiservices.uni-heidelberg.de/entitlement/my-entitlement/`
- Fill out the form.
- Your supervisor will get a notification to approve the application.

Entitlement Application

The screenshot shows a web application for requesting HPC entitlements. The header includes the 'heiservices' logo, a navigation menu, and user information for Luca Mustermensch. The main content area is titled 'Meine bwHPC-Entitlements' and displays a form for a request titled 'Antrag für die bwHPC-Entitlements'. The form fields are as follows:

Field	Value
Antrag für die bwHPC-Entitlements	Akronym: et25_7875 Status: ✓ Aktiv
Antragsteller:in	Luca Mustermensch
Betreuer:in	Prof. Dr. Daniel Düsentrieb
Titel	Running AI LLMs
Beschreibung	I want to run LLMs for my Computational Linguistics master
Ablaufdatum	30.09.2026
Gewöhnlicher Aufenthaltsort	

Figure: <https://heiservices.uni-heidelberg.de/entitlement/my-entitlement/>

bwUniCluster Registration (I)

Once you have received the entitlement, you need to register for cluster access:

- Open the url `https://login.bwidm.de/`
 - Select “Universität Heidelberg” and click “Continue”.
 - Enter your University's credentials (UniID and password) and click “Login”
 - Verify the displayed account information. If everything looks correct, click “Continue”.
- Enable 2-Factor-Authentication (2FA) at `https://login.bwidm.de/user/twofa.xhtml`
- Click on “Register” below the section “bwUniCluster 3.0” service.
- You need to set a service-password to access the bwUniCluster. Do not lose this password, it is required to access the cluster.
- You might upload an SSH-key for authentication:
`https://login.bwidm.de/user/ssh-keys.xhtml`

bwUniCluster Registration (II)



bwIDM

Impressum Datenschutz KIT Deutsch

Übersicht Registrierte Dienste Dienste

bwIDM - Federiertes Identitätsmanagement - Dienste

Sie haben sich bereits bei den folgenden Diensten registriert:

Noch keine Dienste registriert

Folgende Dienste stehen zur Verfügung:

bwUniCluster 3.0

Der am Scientific Computing Center (SCC) des Karlsruher Institut für Technologie (KIT) betriebene bwUniCluster 3.0 ist eines von mehreren zentralen Systemen für eine flächendeckende Grundversorgung der Baden-Württembergischen Universitäten und Hochschulen mit Hochleistungsrechnerkapazität.

[Dienstbeschreibung](#) →

[Registrieren](#) →

- After successful registration you need to fill out a survey within 14 days:
https://zas.bwhpc.de/shib/en/bwunicluster_survey.php
- If you do not fill out the survey your account will get deactivated!

bwUniCluster Access

- Login via SSH (with service-password and 2FA or SSH-key):
`$ ssh hd_<IHRE UNI-ID>@bwunicluster.scc.kit.edu`
- List of available cluster resources:
https://wiki.bwhpc.de/e/BwUniCluster3.0/Hardware_and_Architecture

Final Comments

- A cluster is typically used by many people.
- They often need results as fast as possible.
- However, too many users lead to congestion and long queues.
- Thus, try to be resource conserving when running jobs:
 - Only request GRES if you really use them.
 - Do some testing to determine the required resources of your job (e.g. in interactive mode).
 - Add some headroom, but try to keep resource requests as low as possible.
 - Remember: smaller jobs are scheduled faster!
- When using interactive mode, don't forget to `scancel` your job afterwards.
- Always specify a `--time` parameter in case you forgot about the job.

Exercise

- Cluster Exercise