Cluster Eden

training

Michał Kadlof

15 November 2023

Table of content

Table of content

Introduction

Eden is a high-performance computing cluster. It was initially co-funded by:

- Laboratory of Bioinformatics and Computational Genomics, with a supported by grant from the Ministry of Science and Higher Education
- The Faculty of Mathematics and Information Science of the Warsaw University of Technology.

Computing cluster

Set of computers that work togather to achieve a common goal.

Cluster is devoted strictly to scientific research.

Table of content

Description of resources

The cluster consists of:

- 4 computing nodes Nvidia DGX-A100
- 3 computing nodes Lenovo SR665
- 1 computing node Dell PowerEdge C4130
- 2 management servers Lenovo SR645
- DDN disk array with total capacity 1.5 PiB
  - Double controller SS200NV - 24 disk NVMe 13.9 TiB each
  - 2 Expansion enclosures SS9012 - 124 disks SAS 14.5 TiB each
- Fast Mellanox Onyx switches
  - MQM8700 - InfiniBand 200Gb/s
  - MSN2700 - 100GiB Ethernet
- 2 classic switches 1Gbit Ethernet D-Link DGS-3120-48TC
- KVM Console Aten KL1516Ai
- Tape recorder

Fig: Eden-front

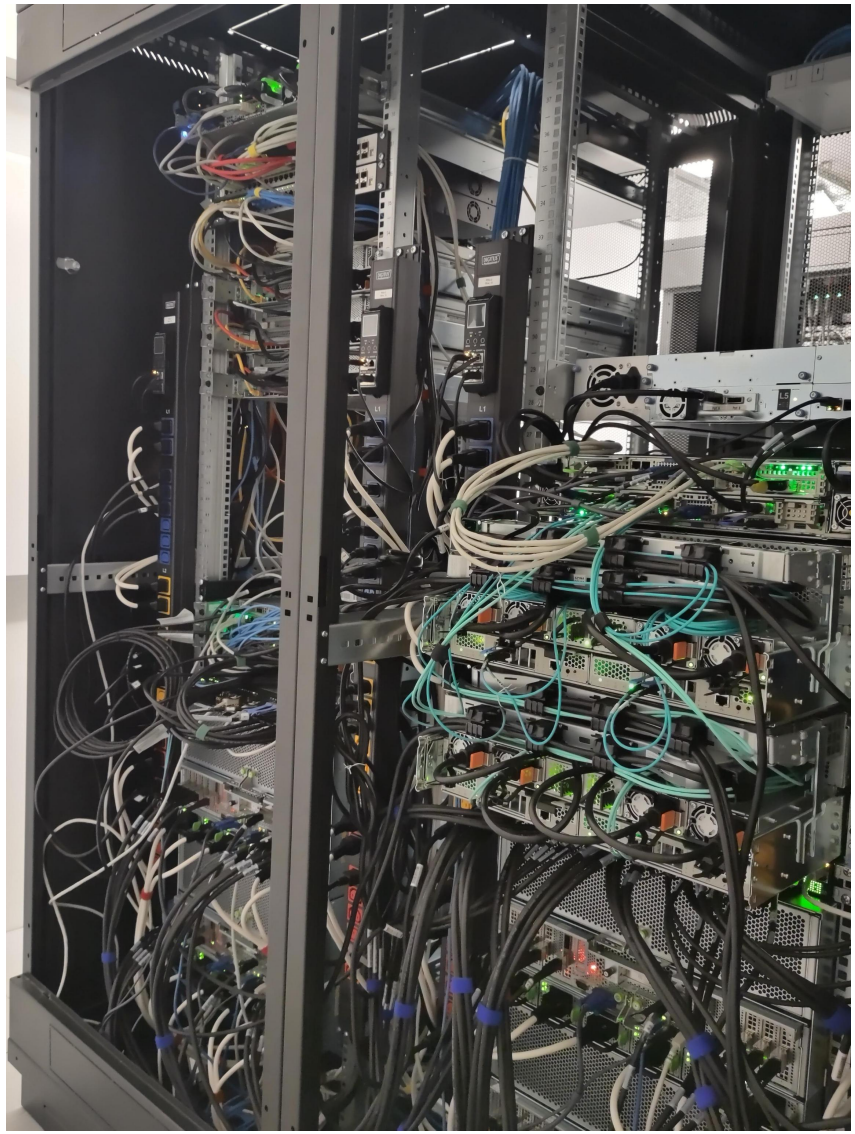

Fig: Eden - rear

NVIDIA DGX-A100

- 2 x processors AMD EPYC 7742
  - 128 physical cores total
- RAM Memory
  - `dgx-1` - 2 TiB
  - `dgx-[2-4]` - 1 TiB
- 8 Graphical units Ampere A100
  - 40 GiB RAM each
- 14 TiB space on internal NVMe drives
- 6 x NVswitch
- 10 x ConnectX-6 200Gb/s network interfaces

Fig: DGX-A100

Pascal

Pascal server

- Dell PowerEdge C4130
- CPU 2x Intel® Xeon® E5-2695 v4 (72 threads)
- GPU 4x Tesla P100 PCIe 16GB
- RAM 251 GiB

Fig: Pascal

Source: Manufacturer's advertising materials

- It is available for students projects.

The Pascal server is at the disposal of Krzysztof Kaczmarski, PhD.

Table of content

Access to Infrastructure

Technical Requirements:

- Access to the internal faculty network
  - On-site within the Mini PW building
  - Remotely via the SSH protocol using the LDAP MINI WUT account

LDAP Account

An email account in the @mini.pw.edu.pl domain is distinct from an LDAP account, although both are created simultaneously for students and employees. If you do not have an account, employees can request its creation for external individuals. Please contact Marcin Borkowski, the head of the computer laboratory, for assistance.

Formal Requirements:

- Being a member of the Research Group lead by one of the scientists employed @ Mini PW.

Research groups

A *Research Group* is a group of people working under the supervision of a *Group Leader*. The group leader must be a person employed at the Mini PW in the group of scientists.
The group leader is responsible for supervising use of the cluster by the group members and is obliged to submit a report on the use of the cluster by the group members once a year.

Terms of use

The most important points:[1]

- The user is responsible for the account entrusted to him and all activities performed through it. In particular, it is forbidden to share the password to the account with other people.
- The user may use the account entrusted to him only for the purpose specified by the group leader.
- If the results of calculations are used in a scientific publication, the group leader is obliged to include the formula in it:

  This research was carried out with the support of the Laboratory of Bioinformatics and Computational Genomics and the High Performance Computing Center of the Faculty of Mathematics and Information Science Warsaw University of Technology.

[1] Full content: `https://hpc.mini.pw.edu.pl/rules/`

Gaining access

If your mentor has already created a group, you can join it. He have to send an e-mail to the HPC Center administrator with the following information:

- First name and last name of new memver
- e-mail address with the @mini.pw.edu.pl domain for people from WUT, and any e-mail address for people from outside the university

If your mentor has not created a group yet, he must first create it. He have to send an e-mail to the HPC Center administrator with the following information:

- Name of the group - the group identifier, which will be used in the Slurm system

An account is usually created within 1–2 days of sending the request. After creating the grant and associated accounts, a welcome e-mail is sent and a password is sent (most often via MS Teams).

Access to the cluster

From now on, you can log in to the access host:

- `eden.mini.pw.edu.pl`

Connecting from outside the faculty

You must log in to the host beforehand `ssh.mini.pw.edu.pl` with a faculty account.[a]

[a] Read more: `https://ww2.mini.pw.edu.pl/laboratorium/uslugi/usluga-ssh/`

Now, you can start working with the cluster!

Fig: login screen

Table of content

Architecture

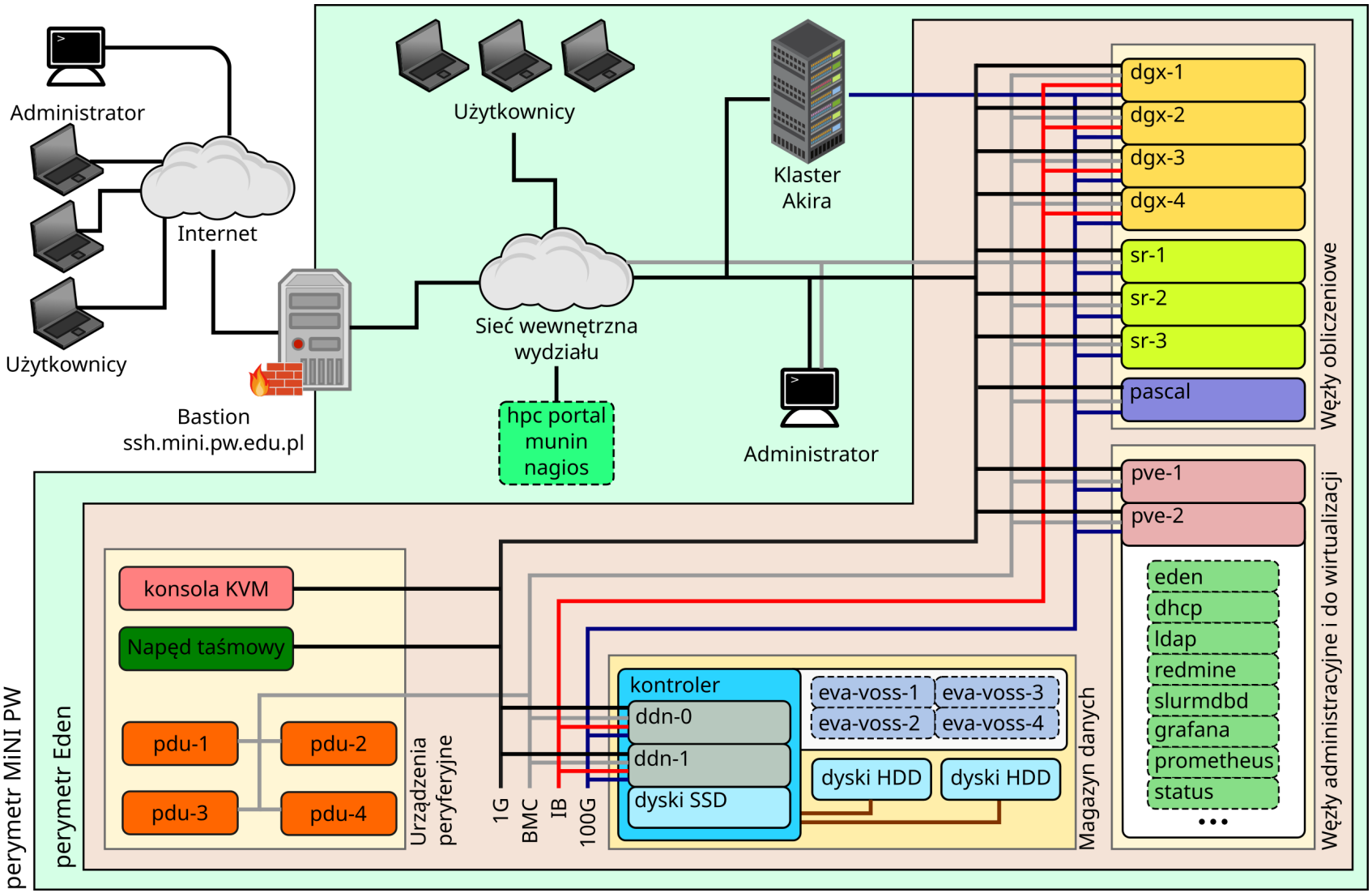

Fig: Eden Architecture

Slurm Queuing System

SLURM - Simple Linux Utility for Resource Management

- Open source (GPL v2)
- Scalable
- Supports approximately 60% of computers from the TOP500 list
- Relatively straightforward to use and administer

Slurm Responsibilities:

- Allocates resources for user tasks for a specified period
- Provides tools to run and monitor tasks on allocated resources
- Maintains accounting of the resources used
- If tasks exceed available resources, it queues them and determines their priorities

SchedMD Logo

Slurm Logo

Slurm Architecture

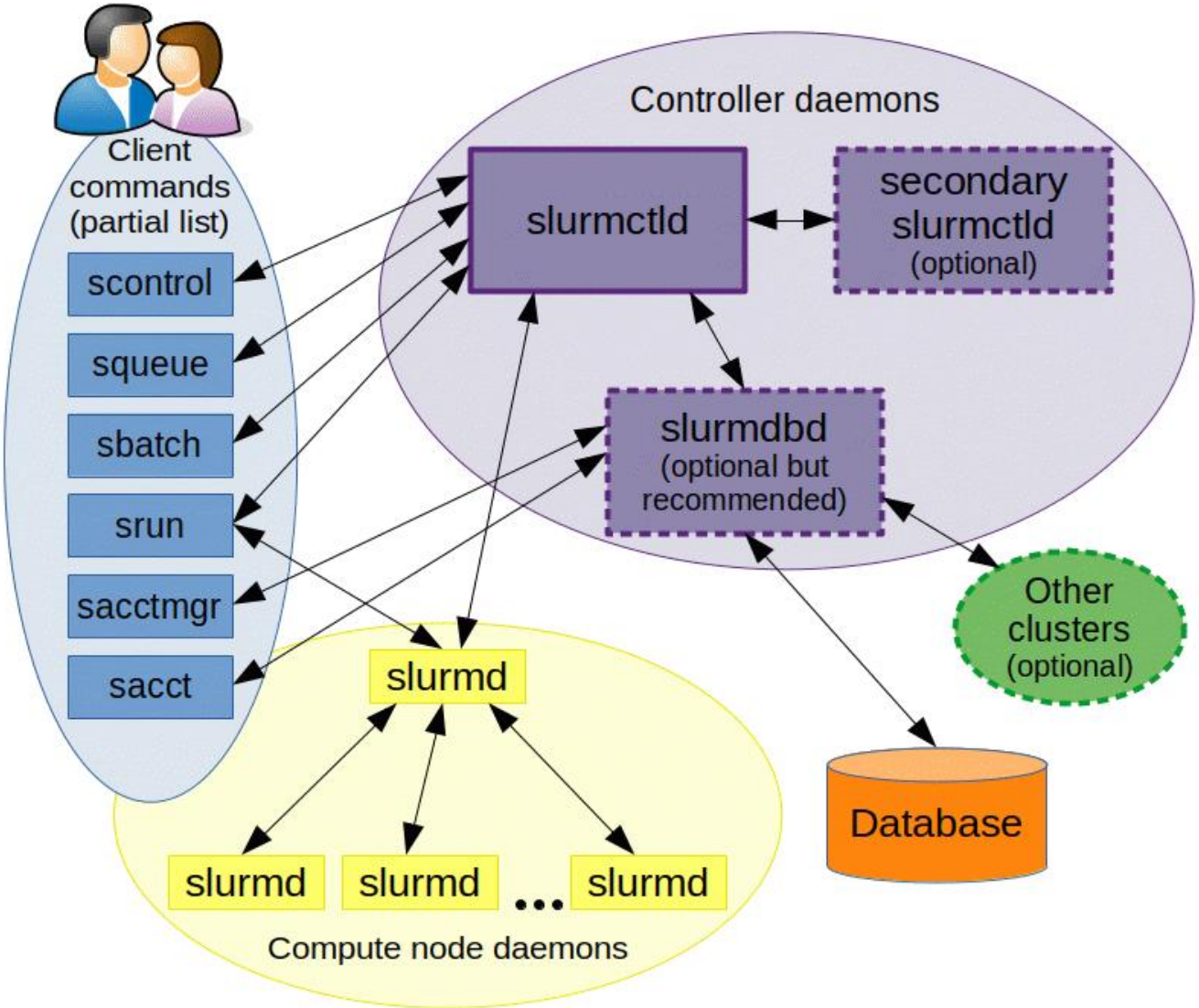

Fig: Slurm Architecture

Source: `https://slurm.schedmd.com/quickstart.html`

Slurm Queues

Queues

A queue (partition) is a logical set of compute nodes. Several queues can group the same nodes

Queues on the Eden cluster:

- `short`
  - default queue
  - the maximum duration of the task is 1 day
  - default task time 1 hour
  - nodes: `dgx-[1-4]`, `sr-[1-3]`
- `long`
  - maximum task duration 10 days
  - default task time 2 days
  - nodes: `dgx-[1-3]`, `sr-[1-3]`
  - `dgx-4` is excluded from the queue
- `experimental`
  - maximum task duration 5 days
  - default task time 1 day
  - Nodes: `pascal`

Cluster state

Usually, the first thing that interests us is the current load on the cluster. Some useful commands:

- `sinfo` - will display information about the queues
- `squeue` - will display information about the jobs in the queue
- `sfree` - informs about free resources
- `pestat` - as above in a slightly different form

sinfo command

```
$ sinfo
PARTITION     AVAIL  TIMELIMIT  NODES  STATE NODELIST
short*           up 1-00:00:00      5    mix dgx-[2-4],sr-[2-3]
short*           up 1-00:00:00      1  alloc dgx-1
short*           up 1-00:00:00      1   idle sr-1
long             up 10-00:00:0      3    mix dgx-[2-3],sr-2
long             up 10-00:00:0      1  alloc dgx-1
long             up 10-00:00:0      1   idle sr-1
experimental     up 5-00:00:00      1   idle pascal
sfglab         down 10-00:00:0      1  alloc dgx-1
```

squeue command

```
$ squeue
 JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
735550      long train_25 mjastrze PD       0:00      1 (ReqNodeNotAvail, Reserved for maintenance)
735616      long train.sh mmalkins PD       0:00      1 (ReqNodeNotAvail, Reserved for maintenance)
735617      long train.sh mmalkins PD       0:00      1 (ReqNodeNotAvail, Reserved for maintenance)
735618      long train.sh mmalkins PD       0:00      1 (ReqNodeNotAvail, Reserved for maintenance)
735619      long train.sh mmalkins PD       0:00      1 (ReqNodeNotAvail, Reserved for maintenance)
735623      long ai_class mjastrze PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:dgx-2)
741129      long train_pl mwojcik2 PD       0:00      1 (AssocGrpGRES)
733606      long nanopoli  sgadakh  R 8-07:56:19      1 dgx-1
735622      long con_prot mgozdera  R   13:48:23      1 dgx-2
735542      long train.sh mmalkins  R 1-13:18:29      1 dgx-2
735540      long train.sh mmalkins  R 1-22:21:45      1 dgx-3
735537      long train.sh mmalkins  R 1-22:21:48      1 dgx-2
735538      long train.sh mmalkins  R 1-22:21:48      1 dgx-3
```

Tip

squeue command allows you to display much more information. The numerous formatting options are difficult to remember and inconvenient to type. It's worth making an alias.

sfree command

```
$ sfree
Node       Free CPUs     Free GPUs    Free MEM [GiB]
=============================================================
dgx-1        0 / 128         0 / 8       1324 / 2016
dgx-2        4 / 128         3 / 8        446 / 1008
dgx-3       44 / 128         0 / 8        582 / 1008
dgx-4      124 / 128         5 / 8        736 / 1008
pascal      36 /  36         4 / 4        504 /  504
sr-1        48 /  48         0 / 0       1008 / 1008
sr-2        27 /  48         0 / 0        802 / 1008
sr-3        40 /  48         0 / 0        908 / 1008
```

pstat command

```
Hostname Partition  Node Num_CPU  CPUload  Memsize  Freemem  Joblist
                 State Use/Tot  (15min)     (MB)     (MB)  JobID User ...
   dgx-1      long alloc 128 128   14.05*  2064063  1518415  733606 sgadakh
   dgx-2     long*   mix 124 128   40.58*  1031878   463958  735622 mgozdera
   dgx-3    long+*   mix  84 128   55.01*  1031877   744029  735540 mmalkinski
   dgx-4    short*   mix   4 128    4.16   1031877   718229  735702 mzelaszczyk
    sr-1    short*  idle   0  48    0.00   1032000   955456
    sr-2    long+*   mix  21  48    0.96*  1032000   807175  735566 mwojcik2
    sr-3    short*   mix   8  48    0.00*  1032000   839286  735674 tbartczak
```

Preview on site

Load plots are available on the website:

`https://eden-status.mini.pw.edu.pl`

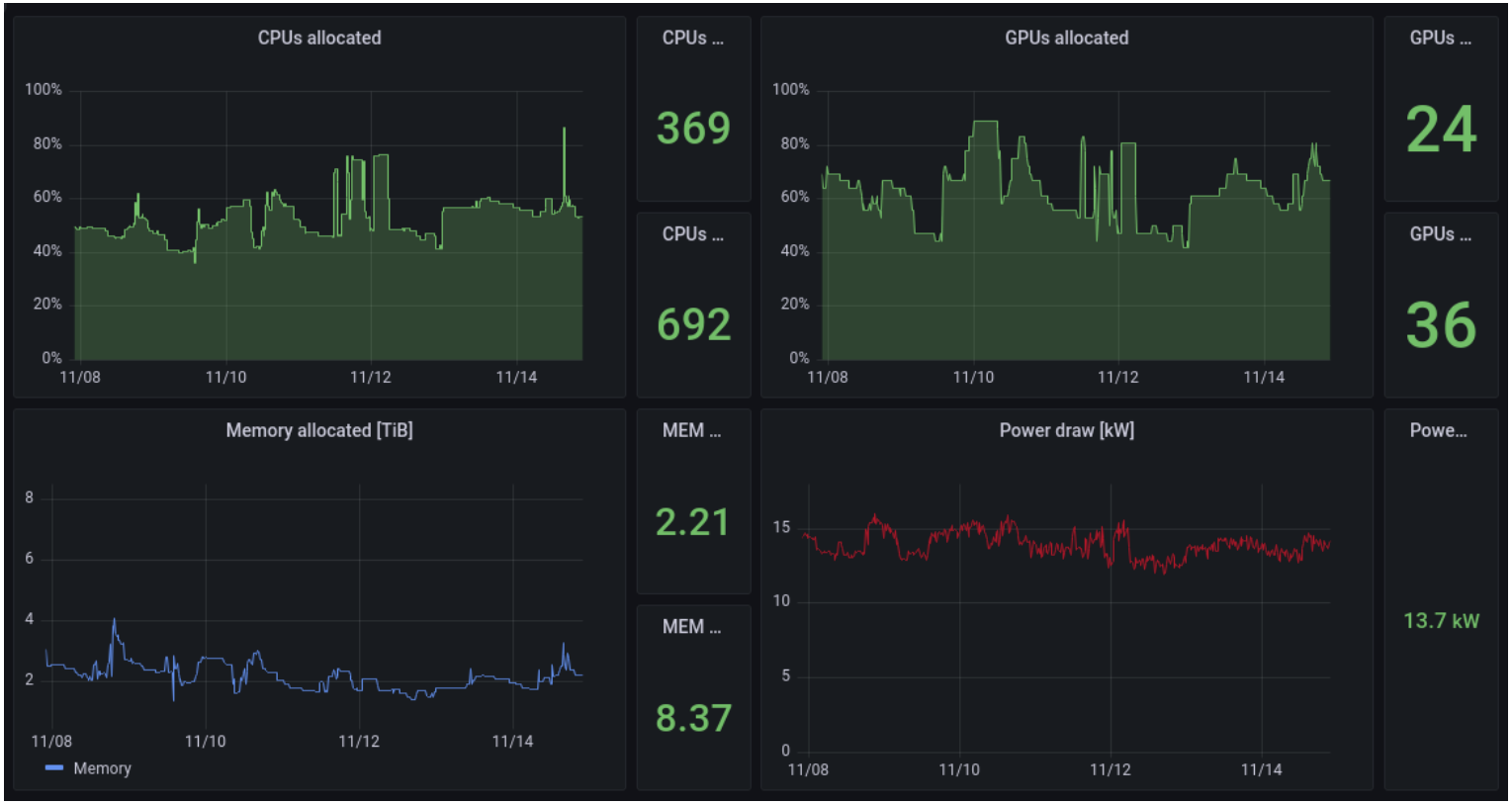

Fig: Grafana plots

Running Tasks

There are two fundamental methods for task execution:

- `srun` - for "normal" launches
- `sbatch` - for launches in "batch" mode

The `srun` method is employed for tasks such as testing, compilation, environment preparation, and interactive work.
On the other hand, the `sbatch` method is utilized for a "submit and forget" approach.

srun command

```
jfikcyjny@eden:~$ srun hostname
srun: job 185415 queued and waiting for resources
srun: job 185415 has been allocated resources
dgx-4
```

Useful flags:

- `-w` determination of hosts
- `-x` determination of unwanted hosts
- `-t` duration time
- `-A` definition of the account (research group)
- `-c` number of cores
- `-G` number of graphics cards
- `-n` number of tasks (for MPI)
- `-p` defining a partition (queue)
- `--test-only` instead of starting the task, it returns information about the estimated start time

srun Command - Interactive Shell

Occasionally, direct access to a shell on a compute node is needed (e.g., for software compilation). Direct SSH login is not an option, but running bash via Slurm is a viable alternative.

Interactive Shell

```
$ srun --pty bash -l
jfikcyjny@dgx-4:~$
```

A common mistake is to add `srun` flags in the end of command instead of before of `bash` word.

sbatch command

sbatch command is used to run tasks in batch mode. It requires the preparation of a file with a batch script.

A batch script is a normal shell script with a few extra special #SBATCH directives.

```
#!/bin/bash
#SBATCH --job-name=serial_job_test    # Job name
#SBATCH --mail-type=END,FAIL          # Mail events
                                      # (NONE, BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=email@pw.edu.pl   # Where to send mail
#SBATCH --ntasks=1                    # Run on a single CPU
#SBATCH --mem=1gb                     # Job memory request
#SBATCH --time=00:05:00               # Time limit days-hrs:min:sec
#SBATCH --output=serial_test_%j.log   # Standard output and error log

pwd; hostname; date
echo "Running plot script on a single CPU core"
python /data/training/SLURM/plot_template.py
date
```

```
$ sbatch test_batch_script.sl
Submitted batch job 185437
```

sbatch command - multithreaded

Example of a multi-threaded job:

```
#!/bin/bash
#SBATCH --job-name=parallel_job       # Job name
#SBATCH --mail-type=END,FAIL          # Mail events
                                       # (NONE, BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=email@pw.edu.pl   # Where to send mail
#SBATCH --nodes=1                     # Run all processes on a single node
#SBATCH --ntasks=1                    # Run a single task
#SBATCH --cpus-per-task=4             # Number of CPU cores per task
#SBATCH --mem=1gb                     # Job memory request
#SBATCH --time=00:05:00               # Time limit hrs:min:sec
#SBATCH --output=parallel_%j.log      # Standard output and error log
pwd; hostname; date
echo "Running prime number generator program on $SLURM_CPUS_ON_NODE \
CPU cores"
/home2/sfglab/jfikcyjny/prime/prime
date
```

sbatch command - multiprocessing

If our python script uses the multiprocess library:

```
#!/bin/bash
#SBATCH --job-name=parallel_job_test # Job name
#SBATCH --mail-type=END,FAIL         # Mail events
                                      # (NONE, BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=email@pw.edu.pl  # Where to send mail
#SBATCH --nodes=1                    # Run all processes on a single node
#SBATCH --ntasks=4                   # Number of processes
#SBATCH --mem=1gb                    # Total memory limit
#SBATCH --time=01:00:00              # Time limit hrs:min:sec
#SBATCH --output=multiprocess_%j.log # Standard output and error log
date;hostname;pwd
python script.py
date
```

Environment Variables

As a gift from Slurm we get numerous environment variables that we can refer to in our scripts.

- SLURMD_NODENAME
- SLURM_CLUSTER_NAME
- SLURM_CONF
- SLURM_CPUS_ON_NODE
- SLURM_CPU_BIND
- SLURM_CPU_BIND_LIST
- SLURM_CPU_BIND_TYPE
- SLURM_CPU_BIND_VERBOSE
- SLURM_GTIDS
- SLURM_JOBID
- SLURM_JOB_ACCOUNT
- SLURM_JOB_CPUS_PER_NODE
- SLURM_JOB_GID
- SLURM_JOB_ID
- SLURM_JOB_NAME
- SLURM_JOB_NODELIST
- SLURM_JOB_NUM_NODES
- SLURM_JOB_PARTITION
- SLURM_JOB_QOS
- SLURM_JOB_UID
- SLURM_JOB_USER
- SLURM_LAUNCH_NODE_IPADDR
- SLURM_LOCALID
- SLURM_NNODES
- SLURM_NODEID
- SLURM_NODELIST
- SLURM_NPROCS
- SLURM_NTASKS
- SLURM_PRIO_PROCESS
- SLURM_PROCID
- SLURM_PTY_PORT
- SLURM_PTY_WIN_COL
- SLURM_PTY_WIN_ROW
- SLURM_SRUN_COMM_HOST
- SLURM_SRUN_COMM_PORT
- SLURM_STEPID
- SLURM_STEP_ID
- SLURM_STEP_LAUNCHER_PORT
- SLURM_STEP_NODELIST
- SLURM_STEP_NUM_NODES
- SLURM_STEP_NUM_TASKS
- SLURM_STEP_TASKS_PER_NODE
- SLURM_SUBMIT_DIR
- SLURM_SUBMIT_HOST
- SLURM_TASKS_PER_NODE
- SLURM_TASK_PID
- SLURM_TOPOLOGY_ADDR
- SLURM_TOPOLOGY_ADDR_PATTERN
- SLURM_UMASK
- SLURM_WORKING_CLUSTER

Descriptions of all variables are available on the manual pages `man sbatch`

Queuing algorithm

After submitting the task, you can relax and wait for the result :)

Queuing algorithm

After submitting the task, you can relax and wait for the result :)
Although, before logging out, it is worth checking the status of your task with the command `squeue`.

Queuing algorithm

After submitting the task, you can relax and wait for the result :)
Although, before logging out, it is worth checking the status of your task with the command `squeue`.

You can get more information about the task by typing the command:
`scontrol show job <job_id>`

Queuing algorithm

After submitting the task, you can relax and wait for the result :)
Although, before logging out, it is worth checking the status of your task with the command `squeue`.

You can get more information about the task by typing the command:
`scontrol show job <job_id>`

Ideally, the task will appear and have a RUNNING status. Sometimes, however, you have to wait. If there are more tasks than resources, the slurm will start queuing the tasks.

Dual Scheduler in Slurm

There are two schedulers in Slurm:

1. Queues jobs according to priority
2. The second attempts to insert small tasks between large tasks, even if the priority order differs

Slurm's scheduler operates like playing multi-dimensional Tetris:

- Tasks serve as building blocks, and dimensions represent resources (time, CPU, GPU, MEM, etc.)
- The smaller the blocks, the easier they are to pack!

Tip!

Occasionally, reducing the number of required cores can lead to a faster completion of the task!

Dual scheduler in Slurm

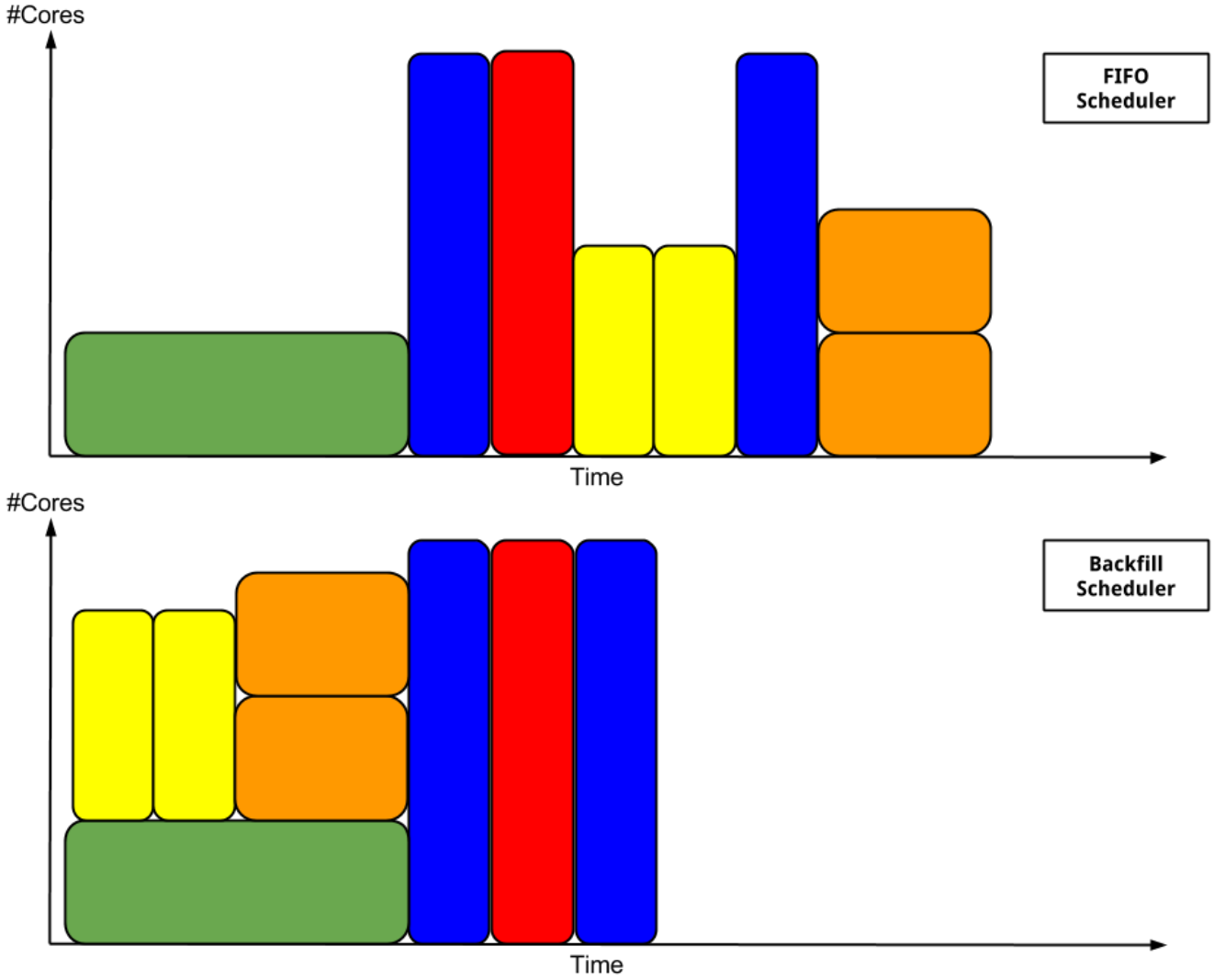

Fig: backfill scheduler
source: https://doc.eresearch.unige.ch/hpc/slurm CC-BY

Queuing algorithm

How does slurm prioritize tasks?

Currently, two factors affect the priority of the task in our cluster:

- job age
- fairshare factor

$$P = f_{age} * 100000 + f_{fairshare} * 200000$$

Factors f_{age} i $f_{fairshare}$ take values from the range [0-1]

Job age

$$P = f_{age} * 100000 + f_{fairshare} * 200000$$

- The task starts with an age factor of 0.
- The longer job waits in queue the more the age factor approaches 1.
- It reaches its maximum value after waiting 7 days for start.

Fairshare factor

$$P = f_{age} * 100000 + f_{fairshare} * 200000$$

- *Fairshare* factor basically is the ratio of used to granted resources.

$$f_{fairshare} = 2^{-U/S}$$

- U - usage
- S - share

$f_{fairshare}$ values:

- 1 - we did not use anything in relation to the other users
- > 0.5 - we used less than we should
- 0.5 - we use exactly as much as we are entitled to
- < 0.5 - we used more than we were entitled to

Usage is a subject of a "half-life decay" with a time constant of 14 days.

Fair tree

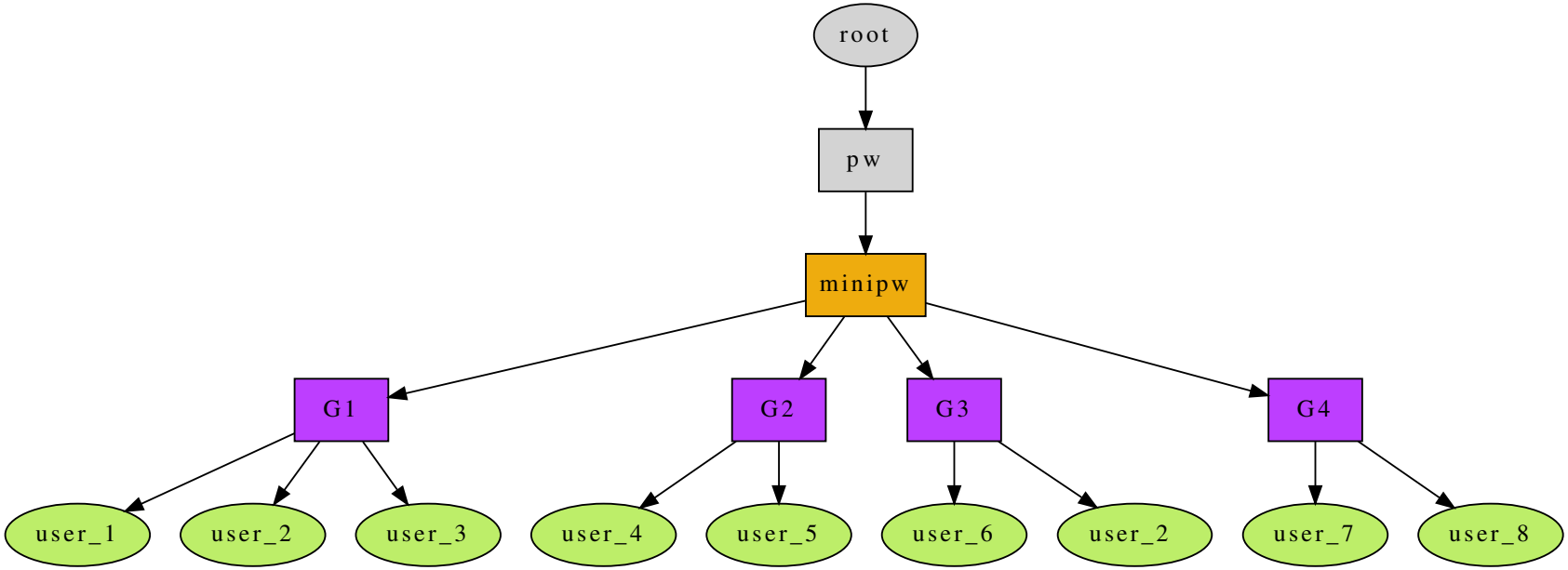

Fig: Fair tree

What to do when a task has low priority?

Most often, you should wait:

- the priority of the job increases with the waiting time in the queue
- *usage* in *fairshare* factor drops in time acording to „half-life" decay
- the more other people use the cluster, the bigger our $f_{fairshare}$ is
- perhaps before the period of increased work, it is worth limiting the number of tasks to "renew" your $f_{fairshare}$
- the required resources should be adapted to real needs
- negotiate individual terms with the HPC Council
- funded groups can increase their share by purchasing equipment

Helpful commands

- `sshare` - examination of fairtree structure and *fairshare* values
- `sprio` - examination of the components of priority

Table of content

Data storage

We usually need access to data for work. Eden cluster users have access to several different data stores.

- Disk array
  - HDD pool
  - NVMe pool
- Internal drives in nodes
- In-memory file systems

Data storage - disk array

The HDD pool is the primary data store

- Capacity 1.5 TiB
  - No limits for now, but quota implementation is planned
- It contains users' home directories
- Lustre file system
- Fast, redundant connections with computing nodes
  - 100Gbit ethernet and InfiniBand
  - native luster client (kernel module)

Path on eden

`/home2/faculty/<username>`

Path on nodes

`/home2/faculty/<username>`

Average write speed		
1 file size 1 GiB	669,2	MiB/s
1000 files size 512 B	721,4	kiB/s

Data storage - NVMe pool

Additional disk space consisting of 24 NVMe disks

- Shared space on computing nodes and the access host
- Total capacity 248.8 TiB
- Theoretically, shorter data access times and faster transfer rates
- Identical connections as for the HDD pool

Path on eden

`/scratch/shared`

Path on dgx

`/scratch/shared`

Average write speed		
1 file size 1 GiB	684,7	MiB/s
1000 files size 512 B	735,5	kiB/s

Data storage - internal DGX drives

DGX-es have two sets of disks:

1. dedicated to the operating system (space inaccessible to users)
2. data storage (available to users)

Workspace:

- 4 NVMe drives in RAID 0
- 14 TiB capacity
  - increased read / write speed in relation to a single disk
  - increased failure rate!
- high performance!

Path on eden

`/mnt/workspace/dgx-[1-4]`

Path on dgx

`/raid`

Average write speed		
1 file size 1 GiB	1,3	GiB/s
1000 files size 512 B	4,1	MiB/s

In-memory file system

It is possible to create a temporary directory in RAM.

- Volatile memory
- Very short access time!
- No data privacy protection!

To use such a space, make a symbolic link to the device in your home directory `/dev/shm`. The link will behave like a normal directory.

```
$ ln -s /dev/shm ./ramdisk
```

The content will be unavailable on `eden` and other hosts.

Average write speed		
1 file size 1 GiB	2,1	GiB/s
1000 files size 512 B	677	MiB/s

Disclaimer

Technical Disclaimer on how to performance measurements was done

The tests were performed with the program dd

```
dd if=/dev/zero of=<target> bs=1G count=1 oflag=sync
dd if=/dev/zero of=<target> bs=512 count=1000 oflag=sync
```

- The test is single-threaded and the results are relative.
- The actual values of the transfer speed depend on many factors and may differ significantly from the measurement.
- The array manufacturer (DDN company) recommends other software to measure the performance.

Methods of Data Transmission

To work with data on the cluster, it must first be transferred to the cluster. Consider the following methods:

- For small volumes (up to several TiB), using SSH is the simplest approach.
- For larger volumes, contemplate physically moving the storage medium.

Transfer rates, as measured:

Medium	**Speed**	
Internet	86	Mbit/s
Faculty Network	252.8	Mbit/s
USB 2.0	346.14	Mbit/s

Tabela: Transfer Rates

Sending approximately 10 TiB from Krakow via the Internet took approximately 10–14 days.

Table of content

Python

Users working with python can install their own python instances by themselves without `root` privileges.
The recommended solution is the `pyenv` environment

```
https://github.com/pyenv/pyenv
```

Pyenv installer

```
https://github.com/pyenv/pyenv-installer
```

Pyenv - installation

1. On any node from the home directory:

```
curl https://pyenv.run | bash
```

2. Add to your `~/.bashrc` file:

```
export PATH="$HOME/.pyenv/bin:$PATH"
eval "$(pyenv init --path)"
eval "$(pyenv virtualenv-init -)"
```

3. Log out and log in again
4. run the `pyenv update` command

After this procedure, a directory will appear in your home directory `.pyenv`, which will hold our python instances.

Using pyenv

To install the selected python instance, enter the command

```
pyenv install <version>.
```

Tip

after word `install` press `<TAB>` twice. A list of available versions and distributions will appear.

Optionally, you can create a virtual environment.

```
pyenv virtualenv <version> <env-name>
```

Using pyenv

In the next step, we need to indicate where we want to use the new instance:

- `pyenv global <version or name>` - everywhere
- `pyenv shell <version or name>` - only in the current shell session
- `pyenv local <version or name>` - in current directory and its subdirectories

The last command will create the `.python-version` file with the name of the selected instance.

Tip

It is safer to use absolute interpreter paths in batch scripts. e.g.:
`/home2/sfglab/mkadlof/.pyenv/versions/3.9.6/bin/python my_job.py`

Other useful commands

- `pyenv versions`
- `pyenv which python`

Jupyter

There is a possibility of interactive work in jupyter.

1. Install jupyter in the local python instance

```
pip install jupyter
```

2. launch the interactive bash shell[2]

```
srun --pty bash -l
```

3. starting the jupyter:

```
jupyter notebook --no-browser --ip 0.0.0.0 --port 9999
```

   If the 9999 port is busy, choose another one in the range 1000–65535

4. Creating an SSH tunnel - local machine in faculty network

```
ssh -NL 8888:dgx-4:9999 eden.mini.pw.edu.pl
```

5. In browser address bar:

```
http://localhost:8888?token=<jupyter-token>
```

[2] Running via `sbatch` is also possible

It is worth remembering that

- interactive jobs usually wait for user instructions most of the time
- during this time the processor is idle
- the task is blocking access to the processor for other users
- consequently the job will charge the user's account and possibly lower the priority of his future jobs

Table of content

Containers

- Problematic docker
- instead of docker singularity[3]

singularity containers

Container system created, with HPC environments in mind. Compatible with docker images.

Simplified workflow:

1. local preparation of the docker image
2. convert to singularity image (single file)
3. image upload to eden
4. launching the container under the control of slurm

[3] More details can be found in:
`https://docs.sylabs.io/guides/3.5/user-guide/introduction.html`

Virtualization

- Users can create their own virtual machines based on the Vagrant + VirtualBox system.
- Inside your own virtual machine you have root privileges (including the ability to run docker).
- No access to the GPU (although theoretically possible).

It is possible to apply for a permanent (not designed for heavy computing) KVM virtual machine or LXC container inside the eden network.

Thank you for your attention!

The presentation can be downloaded from the website

`https://pages.mini.pw.edu.pl/~kadlofm/pages/hpc.html`