TECHNISCHE UNIVERSITÄT DRESDEN

Center for Information Services and High Performance Computing (ZIH)

Introduction to HPC at ZIH

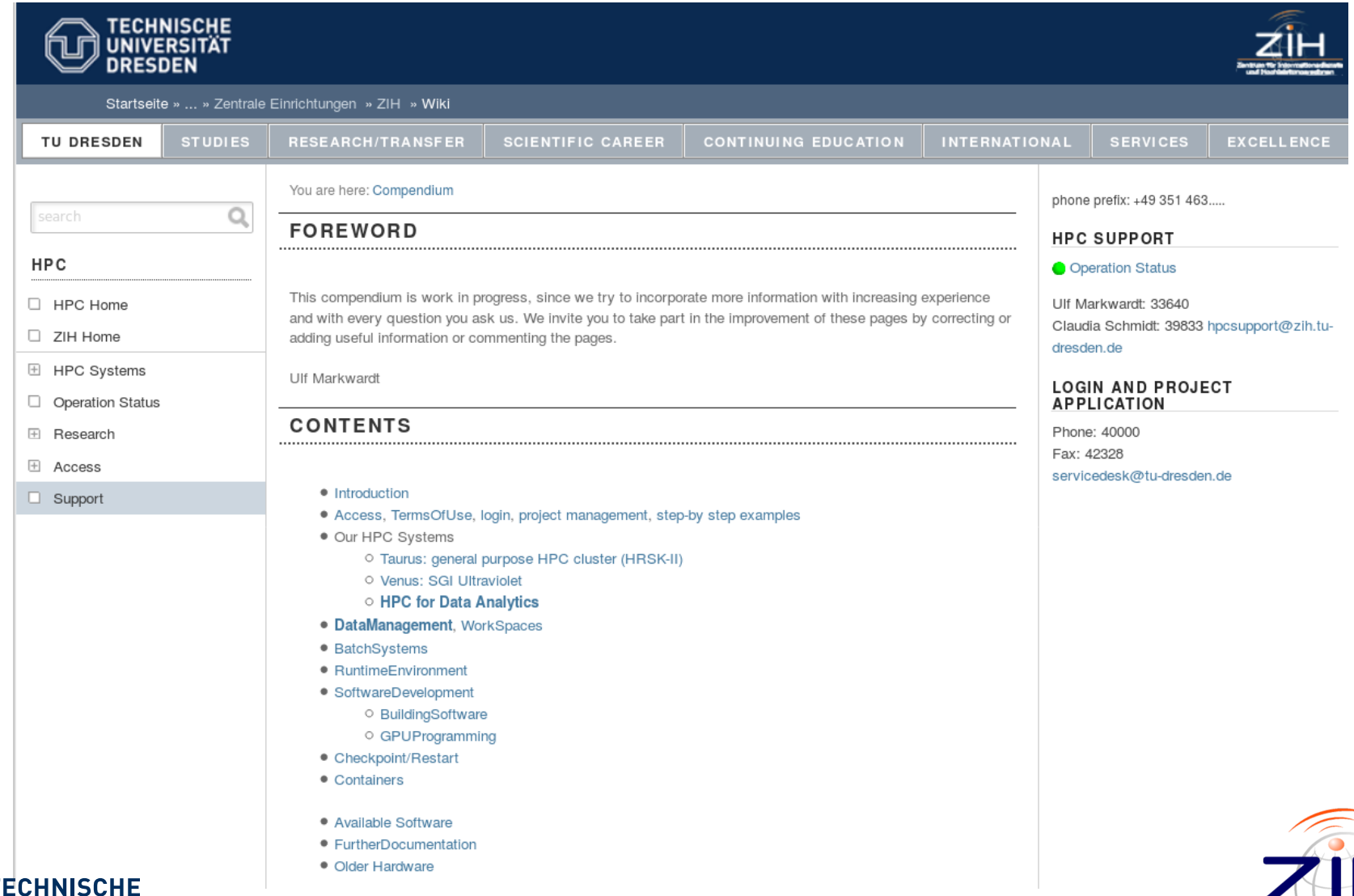
19 September 2019

Dr. Ulf Markwardt
hpcsupport@zih.tu-dresden.de

ZIH
Center for Information Services & High Performance Computing

HPC wiki has the answer

Please check our HPC wiki at `https://doc.zih.tu-dresden.de`



Agenda

1. Linux from the command line
2. HPC Environment at ZIH
   - Access to HPC systems at ZIH
   - Compute hardware
   - HPC file systems
   - Software environment at ZIH
3. Batch System
   - General
   - Slurm examples
4. Software Development at ZIH's HPC systems
   - Compiling
   - Tools
5. HPC Support
   - Management of HPC projects
   - Channels of communication
   - Kinds of support
   - Beyond support

General

- first version 1991, Linus Torvalds
- hardware-independent operating system
- 'Linux' is the name of the kernel as well as of the whole operating system
- since 1993 under GNU public license (GNU/Linux)
- various distributions for all purposes (OpenSuSE, SLES, Ubuntu, Debian, Fedora, RedHat,...) `http://www.distrowatch.com`

Tools for SSH access

Tools to access HPC systems at ZIH from Windows systems
(see `https://doc.zih..../Login`)

- command line login: PuTTY, Secure Shell
- file transfer: WinSCP, Secure Shell
- GUI transfer (XMing, XMing-Mesa, X-Win32)

- integrated solution: MobaXterm

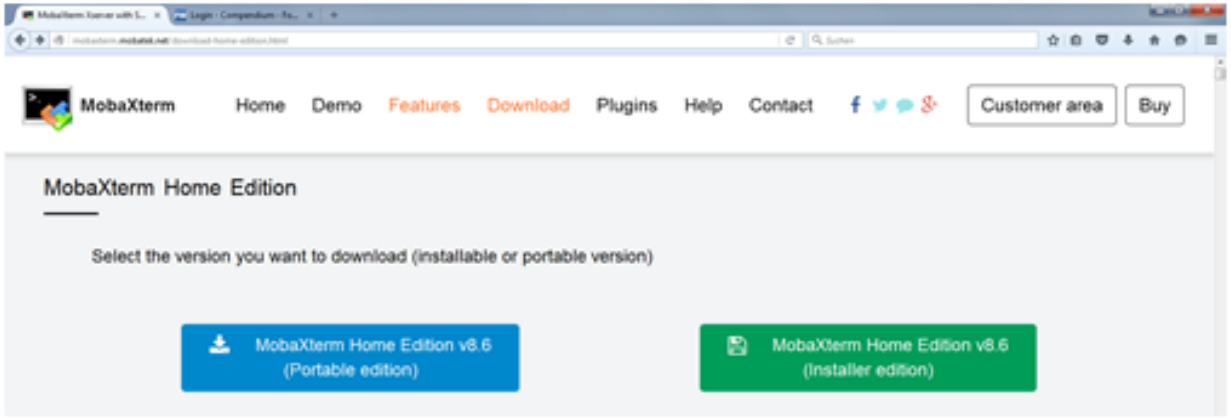
MobaXterm step-by-step instructions

at `https://doc.zih..../MobaXterm`

MOBAXTERM

Installation

- Follow this link to download MobaXTerm: http://mobaxterm.mobatek.net/download.html
- Choose the „Free Version“ by clicking „Download now“
- Then choose the green “Installer Edition”-button



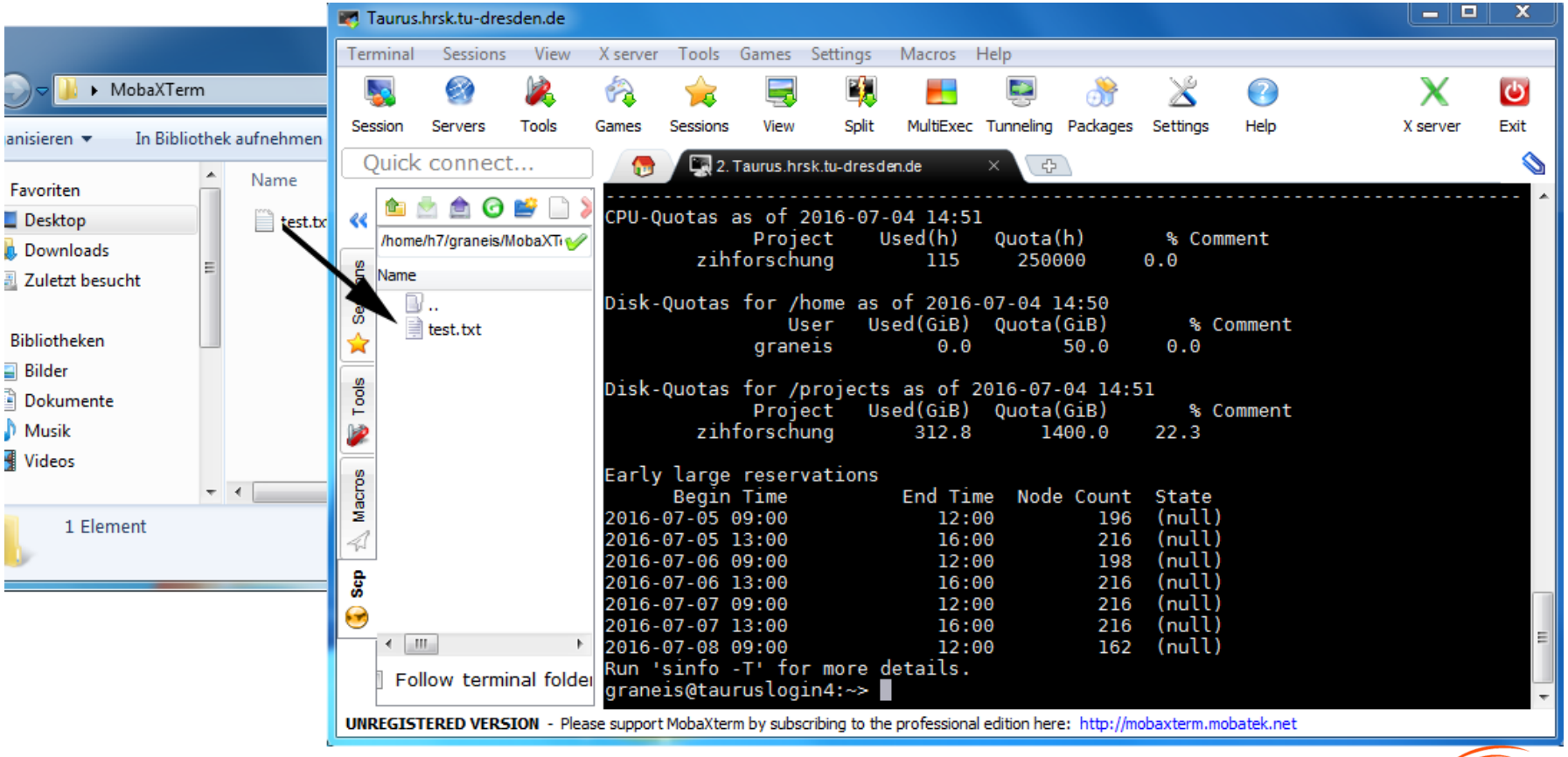
- After download you only have to choose the buttons for going on with the installation

Configuration

- If you have an icon of MobaXTerm on your desktop now open the program
- If you don't have this icon, search the program on your PC and open it

MobaXterm

- console to HPC systems (including X11 forwarding)
- transfer files to and from the HPC systems
- browse through the HPC file systems



Execution of a program

"*Bash is a command processor [...] where the user types commands that cause actions.*" (Wikipedia)

- The shell tries to locate a program from an absolute or relative path like `/sbin/ifconfig` or `./myprog` or `bin/myprog`
- If this fails, it uses the search path to find the file: `/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:.`
- Program execution is controlled by command line options.

```
mark ~/devel> ls
a.out  test1  ulf.c
mark ~/devel> ls -la
total 40
drwxr-xr-x  3 mark zih  4096 Mar 29 09:22 .
drwxr-xr-x 58 mark zih 12288 Jun 18 11:21 ..
-rwxr-xr-x  1 mark zih 13739 Mar 29 09:22 a.out
drwxr-xr-x  8 mark zih  4096 Mar 28 23:17 test1
-rw-r--r--  1 mark zih  2426 Mar 29 09:18 ulf.c
```

Command shell - bash

"*Today, many end users rarely, if ever, use command-line interfaces and instead rely upon graphical user interfaces and menu-driven interactions. However, many software developers, system administrators and* ***advanced users*** *still rely heavily on command-line interfaces to perform tasks more efficiently...*" (Wikipedia)

- large set of environment variables (`printenv [NAME]`) like...

`PATH`	search path for binaries
`LD_LIBRARY_PATH`	search path for dynamic libraries
`HOME`	path to user's home directory

- locate executables and libraries
- expand file names like `ls error*.txt`

bash - language and environment

Mighty scripting language (search the Web for “bash examples”):

```
#! /bin/bash
FILES="*.tgz"
for f in "$FILES"
do
  tar -xzf $f # expand all files
done
```

Set the environment with `NAME=VALUE`.

- access variable with `$NAME`
- will not be set for child processes
- use `export NAME=VALUE` to make the variable valid for other commands. Like `export PATH=$PATH:~/bin`

```
> TEST=Hello
> echo $TEST
Hello
>
```

```
> TEST=Hello
> bash  #sub shell
> echo $TEST

>
```

```
> export TEST=Hello
> bash  #sub shell
> echo $TEST
Hello
>
```

Basic commands

`pwd`	print work directory
`ls`	list directory (`ls -ltrs bin`)
`cd`	change directory (`cd` = `cd $HOME`)
`mkdir`	create directory (`mkdir -p child/grandchild`)
`rm`	remove file/directory **Caution: No trash bin!** (`rm -rf tmp/*.err`)
`rmdir`	remove directory
`cp`	copy file/directory (`cp -r results ~/projectXY/`)
`mv`	move/rename file/directory (`mv results ~/projectXY/`)
`chmod`	change access properties (`chmod a+r readme.txt`)
`find`	find a file (`find . -name "*.c"`)
	or `find . -name "core*" -exec rm {} \;`

Basic commands (cont'd)

`echo`	display text to stdout `echo $PATH`
`cat`	display contents of a file `cat > newfile.txt`
`less`, `more`	pagewise display (`less README`)
`grep`	search for words/text (`grep result out.res`)
`file`	determine type of a file
`ps`	display running processes (`ps -axuf`)
`kill`	kill a process (`kill -9 12813`)
`top`	display table of processes (interactive per default)
`ssh`	secure shell to a remote machine (`ssh -X mark@taurus.hrsk.tu-dresden.de`)

Basic commands (cont'd)

`echo`	display text to stdout `echo $PATH`
`cat`	display contents of a file `cat > newfile.txt`
`less`, `more`	pagewise display (`less README`)
`grep`	search for words/text (`grep result out.res`)
`file`	determine type of a file
`ps`	display running processes (`ps -axuf`)
`kill`	kill a process (`kill -9 12813`)
`top`	display table of processes (interactive per default)
`ssh`	secure shell to a remote machine (`ssh -X mark@taurus.hrsk.tu-dresden.de`)

Editors:

- `vi` - a cryptic, non-intuitive, powerful, universal editor. The web has several "cheat sheets" of vi.
- `emacs` - a cryptic, non-intuitive, powerful, universal editor. But it comes with an X11 GUI.
- `nedit` - an inituitve editor with an X11 GUI. (`module load nedit`)

Help at the command line

Every Linux command comes with detailed manual pages. The command `man <program>` is the first aid kit for Linux questions.

```
CHMOD(1)                        User Commands                        CHMOD(1)

NAME
       chmod - change file mode bits

SYNOPSIS
       chmod [OPTION]... MODE[,MODE]... FILE...
       chmod [OPTION]... OCTAL-MODE FILE...
       chmod [OPTION]... --reference=RFILE FILE...

DESCRIPTION
       This manual page documents the GNU version of chmod.  chmod changes the file
       mode bits of each given file according to mode, which can be either  a  sym-
       bolic representation of changes to make, or an octal number representing the
       bit pattern for the new mode bits.

       The format of a symbolic mode is [ugoa...][[+-=][perms...]...], where  perms
       is  either zero or more letters from the set rwxXst, or a single letter from
       the set ugo.  Multiple symbolic modes can be given, separated by commas.

       A combination of the letters ugoa controls which users' access to  the  file
       will  be  changed:  the user who owns it (u), other users in the file's group
       (g), other users not in the file's group (o), or all users (a).  If none  of
       these  are given, the effect is as if a were given, but bits that are set in
Manual page chmod(1) line 1
```

Linux file systems

- after mounting, remote file systems can be accessed like local resources.
- names are **case sensitiv**
- there is no “drive letter” like C:

Linux file systems

Typical directory structure:

- system programs in `/bin`, `/usr/bin`
- ...
- third party applications, libraries and tools, special software trees e.g
  - normally in `/opt`
  - ZIH's HPC systems in `/sw`
- every user has her own home directory
  - `/home/<login>`
  - e.g. `/home/mark`

Special directories:

- ~ = home directory (`cd ~` or `cd $HOME`)
- `.` = current directory
- `..`=parent directory

File properties

Every file or directory has its access properties:

- 3 levels of access: **u**ser, **g**roup, **o**ther
- 3 properties per level: **r**ead, **w**rite, e**x**ecute (for directories: execute = enter)
- list directory `ls -l .`

```
-rwx-wx-r-x  1 mark zih        9828 Apr 22 13:19 omp
-rw-r--r--   1 mark staff       521 Apr 22 13:19 omp.c
-rw-------   1 mark zih    310288384 May  7 19:01 p1s055.30880.core
-rw-r--r--   1 mark root   116007687 Apr 12 12:56 pluk.tgz
drwx--x-r-x  4 mark staff      4096 Mar 18 16:44 projekte
```

dir/link user group other

Default: User has all access rights in her `$HOME`-directory.

Change file properties

Usage of the `chmod` command:

- indirectly: Which access rights shall be added/removed
  - set a file readable for all: `chmod a+r readme.txt`
  - remove all rights for the group: `chmod g-rwx readme.txt`
- directly: `chmod 700 a.out` Consult the man page: `man chmod`

Redirection of I/O

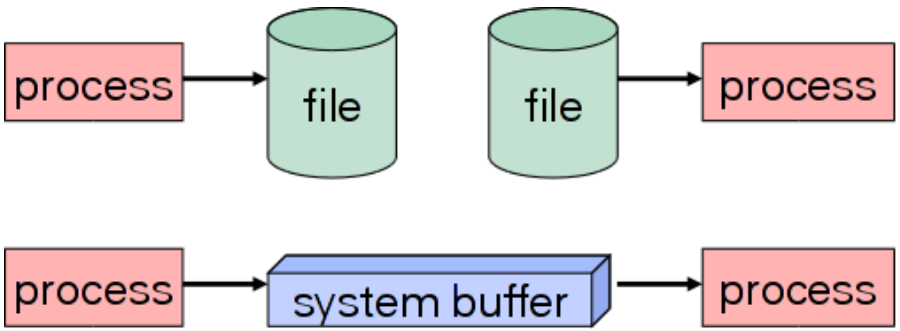
Linux is a text-oriented operating system. Input and output is 'streamable'.

- standard streams are: stdin, stdout, stderr
- streams can be redirected from/to files
  e.g. `myprog <in.txt >out.txt`
- error messages (warnings) are separated from normal program output
  e.g. `myprog 2>error.txt >out.txt`
- merge error messages and output: `myprog 2>&1 out_err.txt`

Attention:
The '`>`' operator will always empty an existing output file. For appending a stream output to a file use the '`>>`' operator. e.g. `myprog >>all_outs.txt`.

Command pipelines

Inputs and outputs can also be other programs.




```
ls -la | sort | more
echo 'Have fun!' | sed -s 's/fun/a break/g'
```

Versatility of Linux (and Linux like operating systems) comes from

- command line controlled program execution
- combining multiple programs in a pipelined execution
- mightful scripting, parsing, and little helper tools (shell, awk, sed, perl, grep, sort)

Nelle's Pipeline I (https://software-carpentry.org)

Nelle Nemo, a marine biologist, has just returned from a six-month survey of the North Pacific Gyre, where she has been sampling gelatinous marine life in the Great Pacific Garbage Patch. She has **1520 samples** in all and now needs to:

- Run each sample through an assay machine that will measure the relative abundance of 300 different proteins. The machine's output for a single sample is a file with one line for each protein.
- Calculate statistics for each of the proteins separately using a program her supervisor wrote called *goostats*.
- Write up results. Her supervisor would really like her to do this by the end of the month so that her paper can appear in an upcoming special issue of *Aquatic Goo Letters*.

It takes about **half an hour** for the assay machine to process **each sample**. The good news is that it only takes two minutes to set each one up. Since her lab has eight assay machines that she can use in parallel, this step will "only" take about two weeks.

Nelle's Pipeline II (https://software-carpentry.org)

The bad news is that if she has to run *goostats* by hand using a **GUI**, she'll have to **select a file using an open file dialog 1520 times**. At 30 seconds per sample, the whole process will take **more than 12 hours** (and that's assuming the best-case scenario where she is ready to select the next file as soon as the previous sample analysis has finished) [...].
The next few lessons will explore what she should do instead. More specifically, they explain how she can **use a command shell** to run the goostats program, using **loops to automate** the repetitive steps e.g. entering file names, so that her **computer can work 24 hours a day** while she writes her paper.

As a bonus, once she has put a processing **pipeline** together, she will be able to use it again whenever she collects more data.

Hands-on training

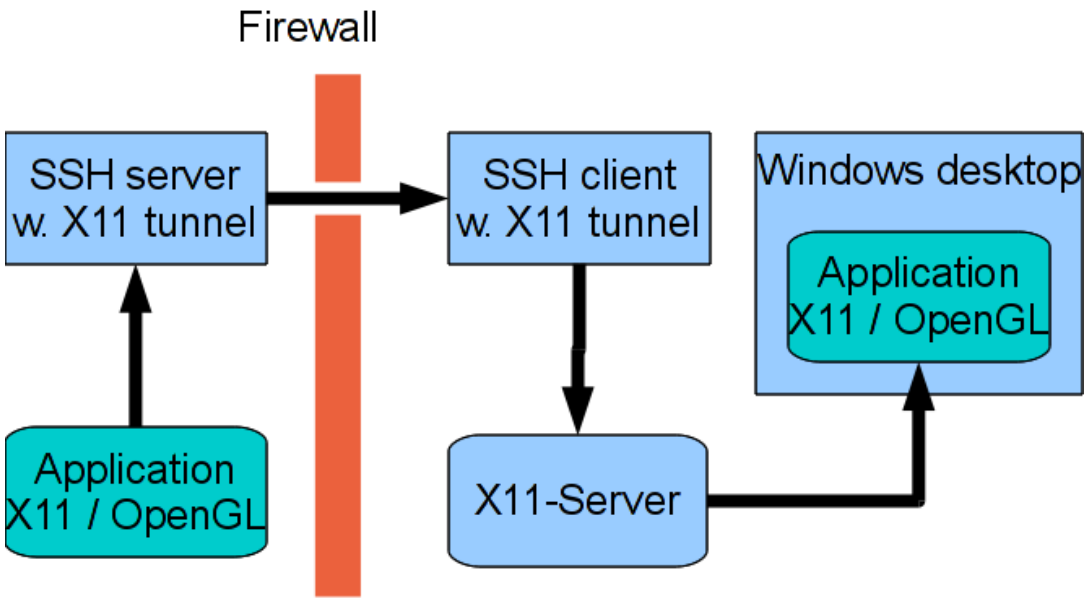
Recommended online material:
`http://swcarpentry.github.io/shell-novice`

Introducing the Shell	What is a command shell and why would I use one?
Navigating Files and Directories	How can I move around on my computer? How can I see what files and directories I have? How can I specify the location of a file or directory on my computer?
Working With Files and Directories	How can I create, copy, and delete files and directories? How can I edit files?
Pipes and Filters	How can I combine existing commands to do new things?
Loops	How can I perform the same actions on many different files?
Shell Scripts	How can I save and re-use commands?
Finding Things	How can I find files? How can I find things in files?

X11 tunnel

Why do we need it?

- redirect graphic contents (GUI or images) to personal desktop system
- only SSH connections are allowed with HPC systems
- at desktop: X11 server to handle graphic input (mouse, keyboard) and output (window contents)

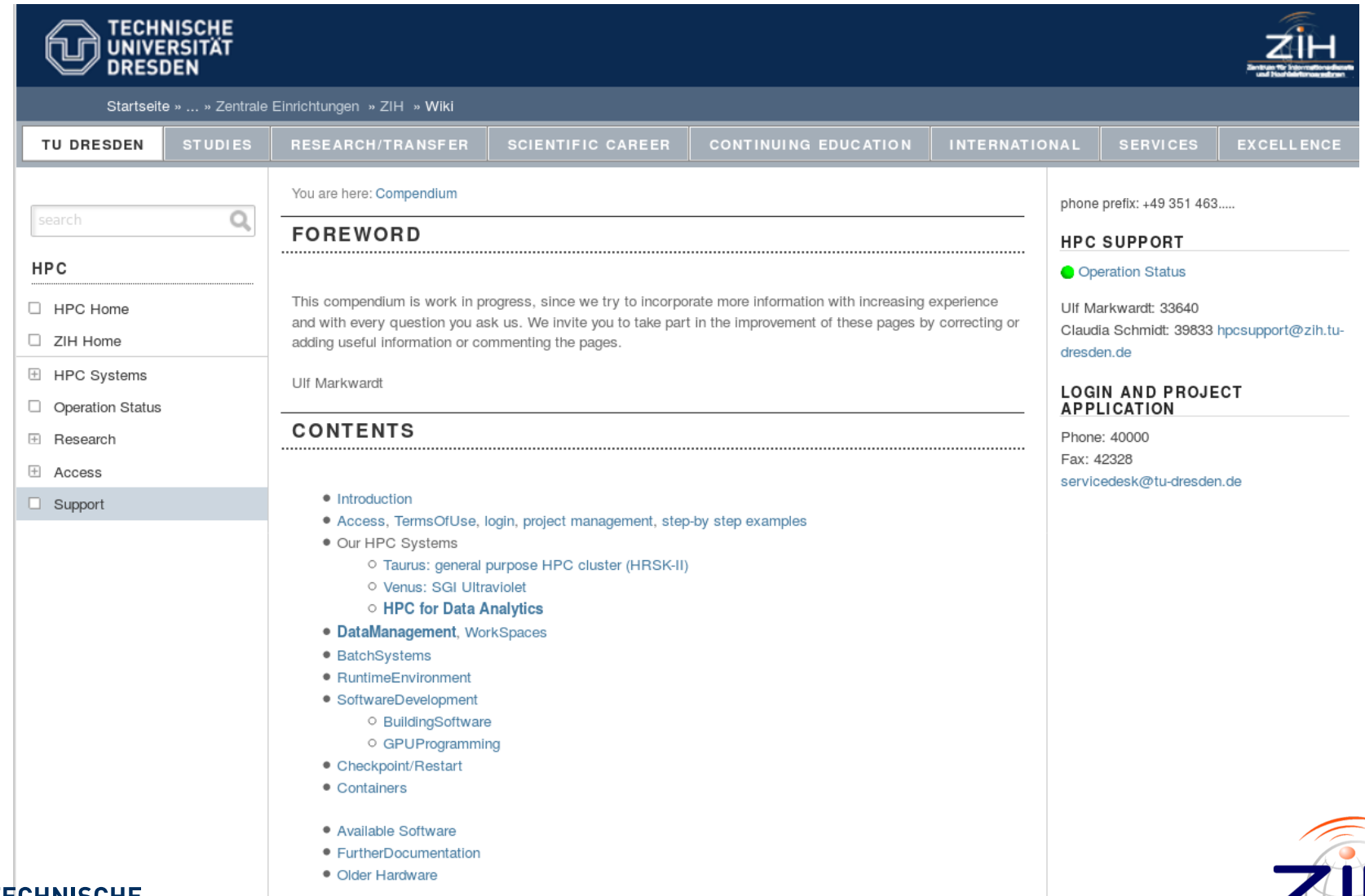


X11 forwarding

- Linux: `ssh -X ...`
- Mac OS X: `http://www.apple.com/downloads/macosx/apple/macosx_updates/x11formacosx.html`
- Windows:
  - Public Domain tool Xming/Xming-mesa: `http://www.straightrunning.com/XmingNotes` or similar product.
  - enable X11 forwarding in the SSH tool
  - integrated solution in MobaXterm
- OpenGL might be an issue

HPC wiki has the answer

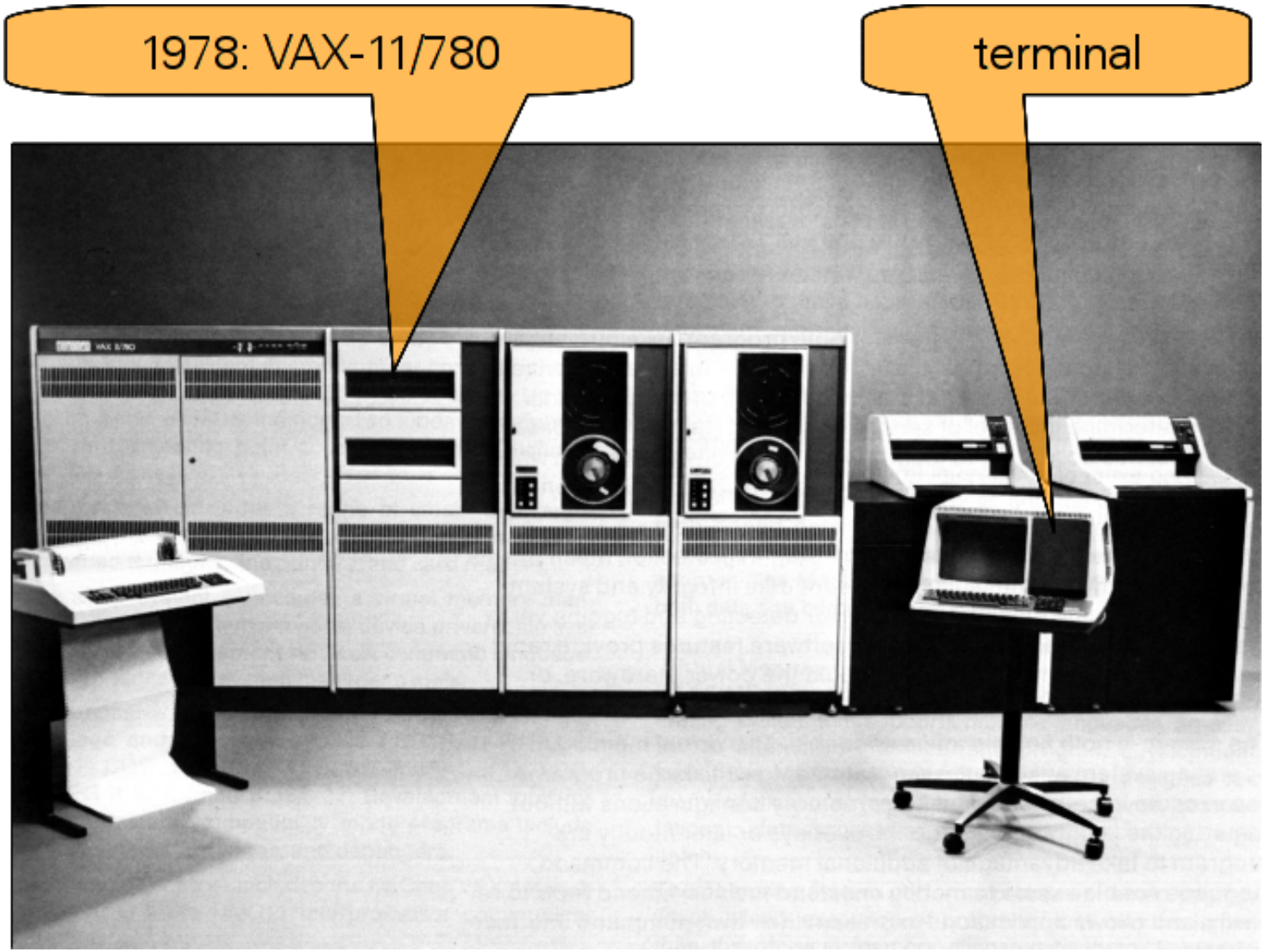
Please check our HPC wiki at `https://doc.zih.tu-dresden.de`



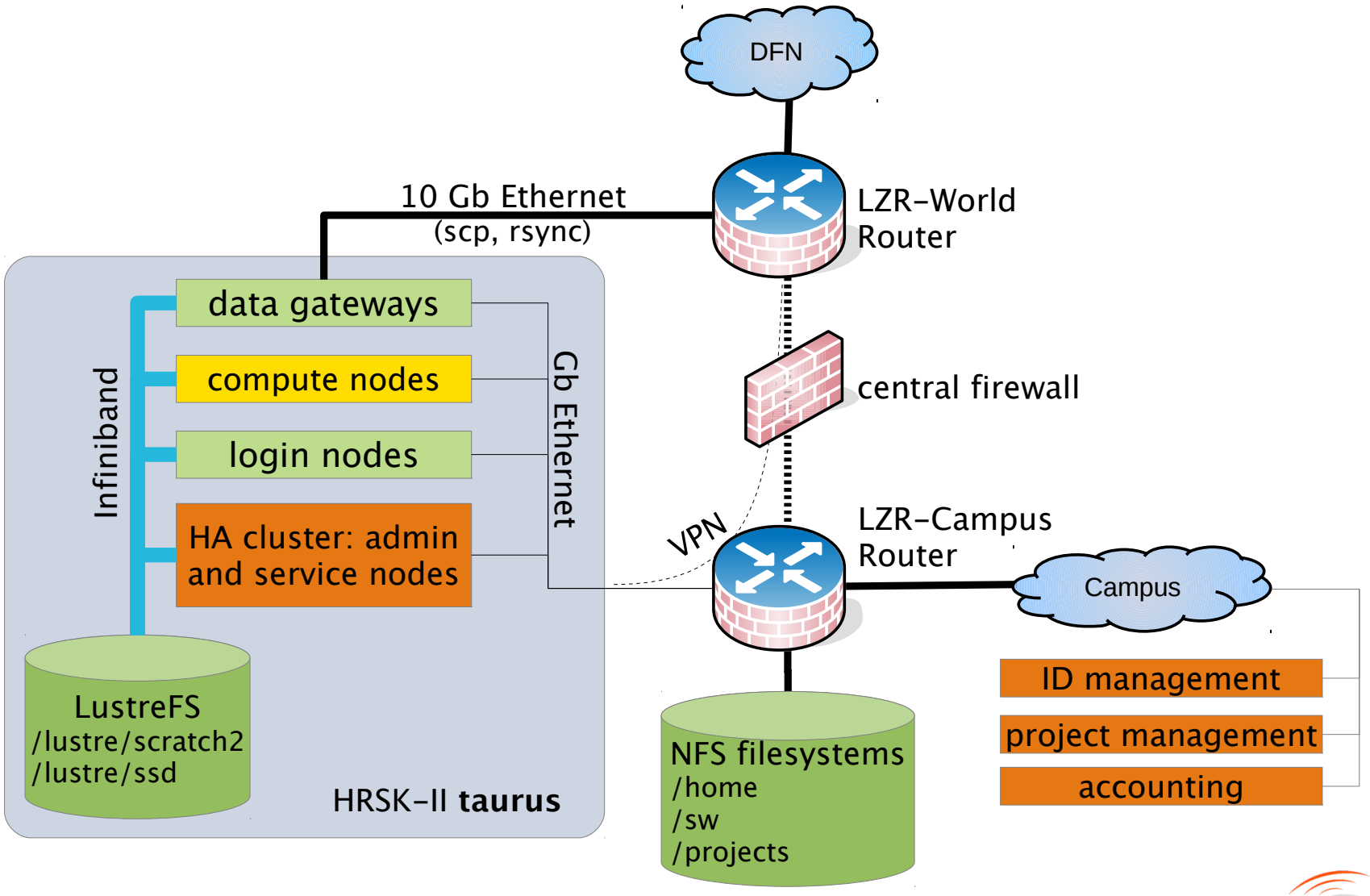
Agenda

1. Linux from the command line
2. HPC Environment at ZIH
   - Access to HPC systems at ZIH
   - Compute hardware
   - HPC file systems
   - Software environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support

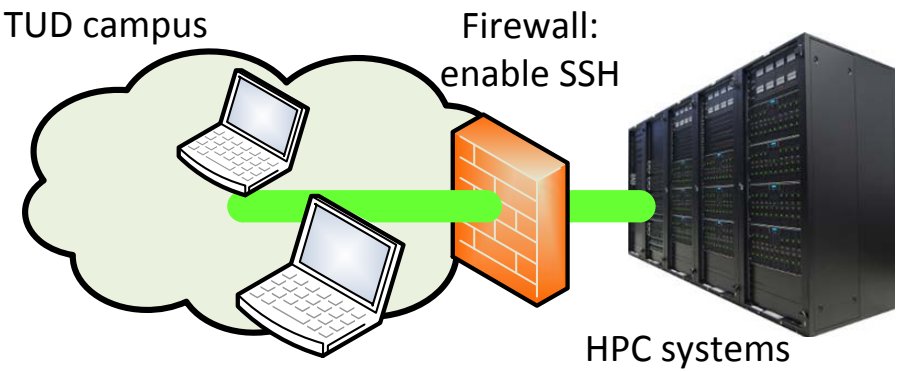
Computer and terminal



Access to the HPC systems



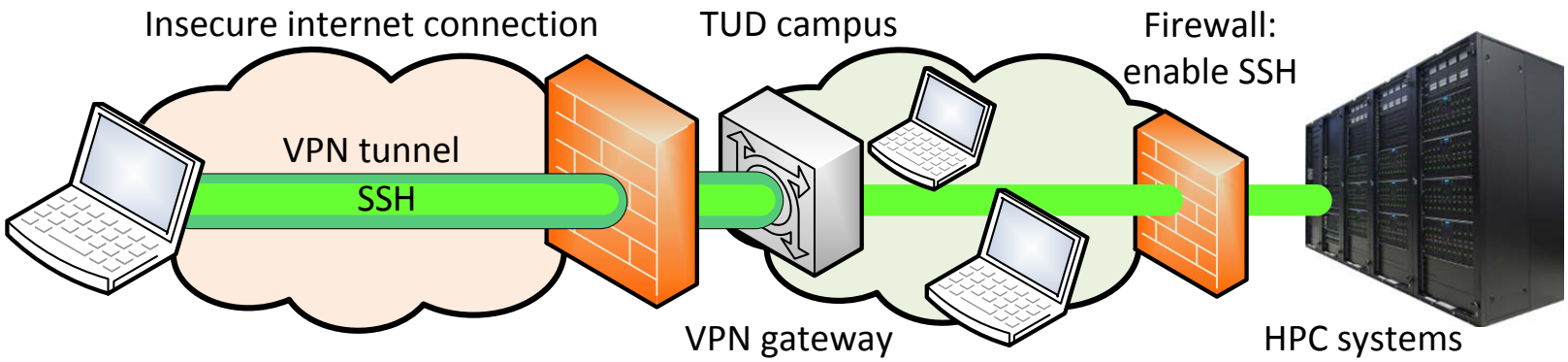
Firewall around HPC systems



The only access to ZIH's HPC systems is

- from within the TU Dresden campus
- from acknowledged IP ranges, e.g.
  TU Freiberg 139.20.0.0/16
  TU Chemnitz 134.109.0.0/16
  Uni Leipzig 139.18.0.0/16
- via secure shell (ssh).

Firewall around HPC systems



The only access to ZIH's HPC systems is

- from within the TU Dresden campus
- from acknowledged IP ranges, e.g.
  TU Freiberg 139.20.0.0/16
  TU Chemnitz 134.109.0.0/16
  Uni Leipzig 139.18.2.0/24
- via secure shell (ssh).

From other IP ranges: **V**irtual **P**rivate **N**etwork

VPN for external users

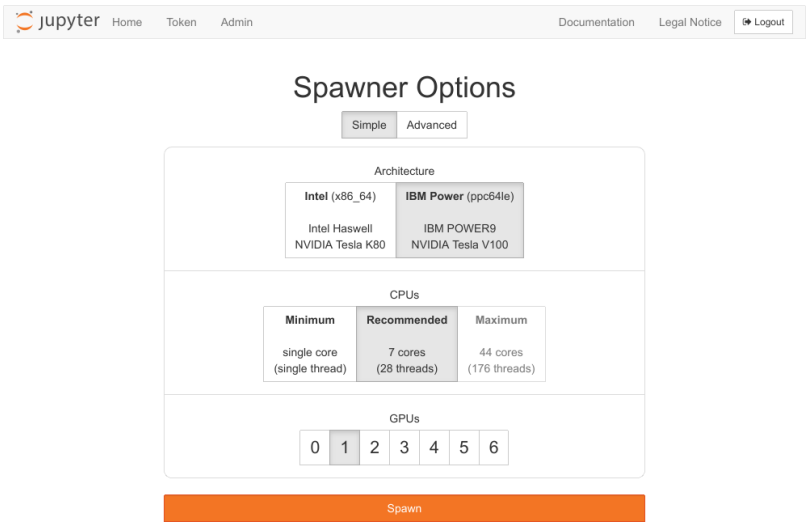
How-To for Linux, Windows, Mac can be found here:
https://tu-dresden.de/zih/dienste/service-katalog/arbeitsumgebung/zugang_datennetz/vpn

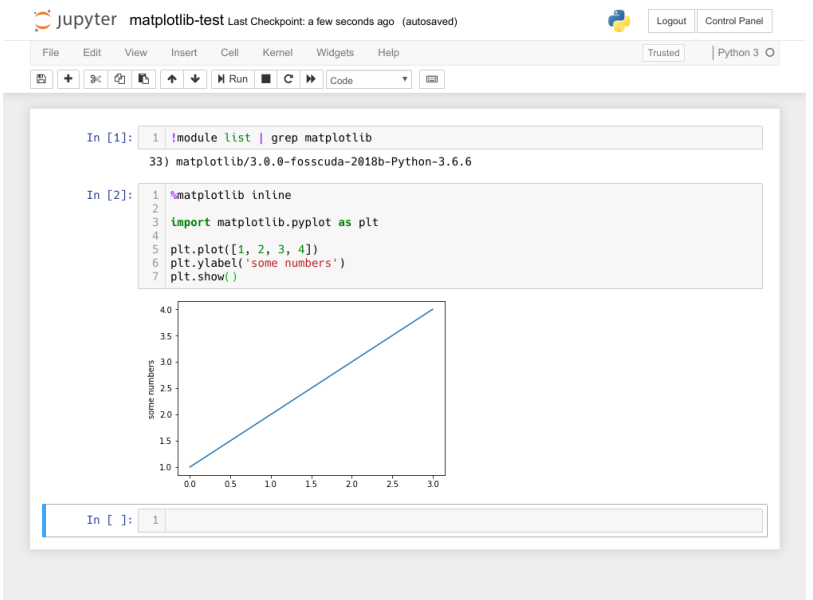
- install VPN tool at your local machine
  - OpenConnect (`http://www.infradead.org/openconnect`)
  - Cisco Anyconnect
- configuration

gateway	vpn2.zih.tu-dresden.de
group	TUD-vpn-all
username	ZIH-LOGIN@tu-dresden.de
password	ZIH-PASSWORD

Access to HPC

Unleash the HPC power with `ssh -X taurus.hrsk.tu-dresden.de` !
Or use a GUI from your Web browser → JupyterHub.





Detailled documentation can be found at `https://doc.zih..../JupyterHub` .

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
   - Access to HPC systems at ZIH
   - Compute hardware
   - HPC file systems
   - Software environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support

HPC Infrastructure at ZIH

HPC at ZIH

- state's computing center for HPC in Saxony
- HPC systems are funded by BMBF and SMWK
- services free of charge to
  - all universities in Saxony,
  - all listed research institutes (e.g. Leibniz, Max Planck, Fraunhofer institutes)
- active projects outside TUD: MPI-CBG, HZDR, IFW, Uni Leipzig, TUBAF)

National competence center for data analytics "ScaDS" (with Universität Leipzig)

- hardware extensions
  - NVMe nodes (block storage over Infiniband),
  - nodes for machine learning,
  - "warm archive" for research data, VM images...
- new methods to access systems complementary to "classical" HPC mode

Taurus

Overview

- General purpose cluster from Bull/Atos for highly parallel HPC applications (2013/2015)
- running with RHEL 7.4
- 1,029.9 TFlop/s total peak performance (rank 66 in top500, 06/2015)
- GPU partition with 128 dual GPUs + 88 single GPUs
- all compute nodes have local SSD
- 4 PB scratch file system
- 40 TB High IOPS file system based on SSDs
- sample energy consumption with 1 kHz

Taurus

Heterogenous compute resources

- Normal compute nodes
  - 270 nodes Intel Sandy Bridge (2 x 8 cores, AVX), 2,4,8 GB/core
  - 1456 nodes Intel Haswell, (2 x 12 cores), 64,128,256 GB/node
  - 32 nodes Intel Broadwell, (2 x 14 cores), 64 GB/node
- Large SMP nodes
  - 2 nodes with 1 TB RAM, Intel Sandy Bridge (4 x 8 cores)
  - 5 nodes with 2 TB RAM, Intel Haswell (4 x 14 cores)
- Accelerator and manycore nodes
  - 44 nodes with 2 x NVidia K20x, Intel Sandy Bridge (2 x 8 cores)
  - 64 nodes with 2 x NVidia K80, Intel Haswell (2 x 12 cores)
  - 32 nodes with 6 x NVidia V100-SXM2, IBM Power9 (2 x 22 cores)
- **Storage for data analytics**
  - 90 nodes with 8 NVMe cards (2.9 TB each)
  - warm archive powered by Quobyte
    - total of 13 PB
    - accessible as filesystem inside Taurus
    - S3 object store

Venus

Large shared-memory System (SGI Ultraviolet) for memory-intensive computing (2013)

- 8 TB shared memory
- 10,6 TFlop/s peak performance
- 512 cores Intel E5-4650L (Sandy Bridge) 2,6 GHz
- SuSE Linux Enterprise Server 11
- batch system Slurm

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
   - Access to HPC systems at ZIH
   - Compute hardware
   - HPC file systems
   - Software environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support

Overview

Hierarchy of file systems: **speed** vs. **size** vs. **duration**:

- local SSD `/tmp`
- HPC global `/ssd`
- HPC global `/scratch`
- HPC global `/projects`, `/home`
- warm archive `/warm_archive`
- TUD global intermediate archive
- TUD global long term storage

The **number of files** (millions) is critical for all file systems.

Local disk

Recommended at Taurus:

- SSD: best option for lots of small I/O operations, limited size ($\sim$ 100GB),
- ephemeral: data will be deleted automatically after finishing the job,
- Each node has its own local disk. Attention: Multiple processes on the same node share their local disk,
- path to the local disk is `/tmp`

High-IOPS file system

Fastest parallel file systems (IOPS) at each HPC machine:

- large parallel file system for high number of I/O operations,
- data may be deleted after 30 days,
- management via workspaces,
- All HPC nodes share this file system.

Attention: Data might get lost.

Scratch file system

Fastest parallel file systems (streaming) at each HPC machine:

- large parallel file system for high bandwidth,
- data may be deleted after 100 days,
- management via workspaces,
- All HPC nodes share this file system.

Attention: Data might get lost. Probably not.

Permanent file systems

Common file system for all ZIH's HPC machines:

- Very slow and small, but with multiple backups.
- Deleted files are accessible via the logical `.snapshot` directory. This directory contains weekly, daily, and hourly snapshots. Copy your file to where you need it.
- Paths to permanent storage are
  - `/home/<login>` (20 GB !) and
  - `/projects/<projectname>` - mounted read-only on compute nodes!

  with different access rights (cf. Terms of Use).
- All HPC systems of ZIH share these file systems.

Warm archive

Large storage at each HPC machine:

- large parallel file system for moderately high bandwidth,
- management via workspaces,
- all HPC nodes share this file system,
- mounted read-only on compute nodes

Archive

Common tape based file system:

- really slow and large,
- expected storage time of data: about 3 years,
- access under user's control.

Best practice:

- "Low" file count is important.
- Tar and zip your files. (Use datamover nodes.)
- LTO-6 tapes have a capacity of 2.5 TB. Please ask before you plan to archive files larger than 200 GB.

Data management

Automated workflows vs. manual control

- A set of rules specifies how and when data is moved between storage systems.
- Who defines these rules? User or administrator?
- When are actions triggered?

- User moves her own data.
- User knows when data can be stored away or have to be retrieved for next processing steps.

In general, users are responsible for their data.
Admins care for usability and data integrity.

Workspaces

Tool for users to manage their storage demands

- In HPC, projects (and data) have limited lifetime.
- User creates a workspace with defined expiration date.
- User can get an email (or calender entry) before expiration.
- Data is deleted automatically (cf. comment).
- Life-span can be extended twice.

Maximum initial lifetime depends on file system:

Storage system	**Duration**	**Remarks**
ssd	30 days	High-IOPS file system, SDDs.
scratch	100 days	High streaming bandwidth, disks.
warm_archive	1 year	Capacity file system, disks.

Workspace - examples

```
mark@taurus login3:~> ws_find -l
available filesystems:
warm_archive
scratch
ssd
```

Allocation

```
mark@taurus login3:~> ws_allocate -F ssd SPECint
Info: creating workspace.
/lustre/ssd/ws/mark-SPECint
remaining extensions  : 2
remaining time in days: 5
```

Notification:

```
mark@taurus login3:~> ws_send_ical -m ulf.markwardt@tu-dresden.de \
-F ssd SPECint
Sent reminder for workspace SPECint to ulf.markwardt@tu-dresden.de
please do not forget to accept invitation
```

→Invitation: “Workspace SPECint will be deleted on host Taurus”

Workspace - examples

List all allocated workspaces

```
mark@tauruslogin3:~> ws_list
id: SPECint
     workspace directory  : /lustre/ssd/ws/mark-SPECint
     remaining time       : 4 days 23 hours
     creation time        : Wed Sep 18 09:41:08 2019
     expiration date      : Mon Sep 23 09:41:08 2019
     filesystem name      : ssd
     available extensions : 2
```

Extend the live-time of a workspace

```
mark@tauruslogin3:~> ws_extend -F ssd SPECint 10
Info: extending workspace.
/lustre/ssd/ws/mark-SPECint
remaining extensions  : 1
remaining time in days: 10
```

Attention: Extension starts **now**, not at the end of the life-time

```
mark@tauruslogin3:~> ws_list -F ssd
id: SPECint
     workspace directory  : /lustre/ssd/ws/mark-SPECint
     remaining time       : 9 days 23 hours
     creation time        : Wed Sep 18 09:43:01 2019
     expiration date      : Sat Sep 28 09:43:01 2019
     filesystem name      : ssd
     available extensions : 1
```

Workspace -examples

```
#!/bin/bash
#SBATCH --partition=haswell
...
COMPUTE_DIR=gaussian_$SLURM_JOB_ID
ws_allocate -F ssd $COMPUTE_DIR 7
export GAUSS_SCRDIR=/ssd/ws/$USER-$COMPUTE_DIR

srun g16 inputfile.gjf logfile.log

#Delete ASAP!
test -d $GAUSS_SCRDIR && rm -rf $GAUSS_SCRDIR/*

ws_release -F ssd $COMPUTE_DIR
```

Expired workspaces are moved automatically to another location. They remain there for another 30 days (warm_archive: 60 days) before final deletion. → Delete as soon as possible (using “rm”)!

```
mark@taruslogin3:~> ws_release -F ssd SPECint
```

Data transfer

Special data transfer nodes are running in batch mode to comfortably transfer large data between different file systems:

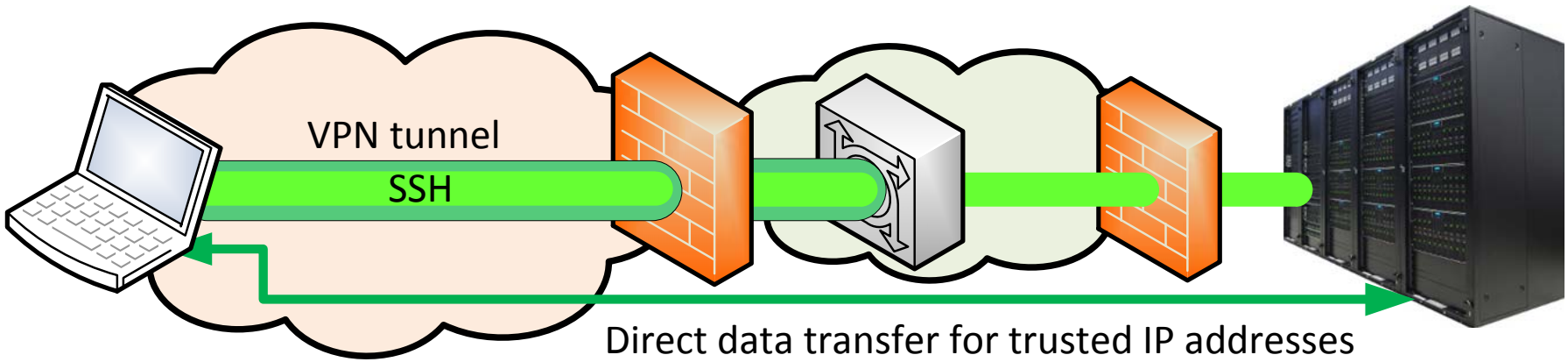
- Commands for data transfer are available on all HPC systems with prefix **dt**: dtcp, dtls, dtmv, dtrm, dtrsync, dttar.
- The transfer job is then created, queued, and processed automatically.
- User gets an email after completion of the job.
- Aditional commands: dtinfo, dtqueue.

Very simple usage like

```
dttar -czf /warm_archive/ws/jurenz-sim/results_20190820.tgz \
          /scratch/ws/jurenz-sim21/results
```

External data transfer

The nodes taurusexport.hrsk.tu-dresden.de allow access with high bandwidth bypassing firewalls



Restrictions

- trusted IP addresses only
- protocols: sftp, rsync

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
   - Access to HPC systems at ZIH
   - Compute hardware
   - HPC file systems
   - Software environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support

Modules

Installed software is organized in modules.

A module is a user interface, that:

- allows you to easly switch between different versions of software
- dynamically sets up user's environment (`PATH`, `LD_LIBRARY_PATH`, ...) and loads dependencies.

Private modules files are possible (e.g. group-wide installed software).

Installed Software

Installed software (appr. 2000 modules) can be found at...

`https://doc.zih..../SoftwareModulesList` (daily update)

abaqus, abinit, ace, adolc, afni, amber, ansys, ansysem, asm, autoconf, automake, autotools, AVS, bazel, bison, boost, bullxde, bullxmpi, casita, ceph, cereal, cg, clFFT, cmake, collectl, comsol, conn, cp2k, ctool, cube, cuda, cusp, cython, dalton, darshan, dash, dataheap, ddt, dftb+, dmtcp, doxygen, dune, dyninst, eigen, eman2, ensight, extrae, fftw, flex, fme, freecad, freeglut, freesurfer, fsl, ga, gamess, gams, gaussian, gautomatch, gcc, gcl, gcovr, gctf, gdb, gdk, geany, ghc, git, glib, gmock, gnuplot, gpaw, gperftools, gpi2, gpudevkit, grid, gromacs, gsl, gulp, gurobi, h5utils, haskell, hdeem, hdf5, hoomd, hyperdex, imagemagick, intel, intelmpi, iotop, iotrack, java, julia, knime, lammps, lbfgsb, libnbc, liggghts, llvm, lo2s, ls, ls-dyna, lumerical, m4, map, mathematica, matlab, maxima, med, meep, mercurial, metis, mkl, modenv, motioncor2, mpb, mpi4py, mpirt, mumps, must, mvapich2, mxm, mysql, namd, nedit, netcdf, netlogo, numeca, nwchem, octave, octopus, opencl, openems, openfoam, openmpi, opentelemac, orca, oscar, otf2, papi, paraview, parmetis, pathscale, pdt, petsc, pgi, pigz, protobuf, pycuda, pyslurm, python, q, qt, quantumespresso, r, redis, relion, ripgrep, root, ruby, samrai, scala, scalasca, scons, scorep, sftp, shifter, siesta, singularity, sionlib, siox, spm, spm12, spparks, sqlite3, stack, star, suitesparse, superlu, svn, swig, swipl, tcl, tcltk, tecplot360, tesseract, texinfo, theodore, tiff, tinker, tmux, totalview, trace, trilinos, turbomole, valgrind, vampir, vampirtrace, vasp, visit, vmd, vtk, wannier90, wget, wxwidgets, zlib

Module environments

Two different module environments:

- scs5 - for software built from “recipes” with EasyBuild (default)
- classic - for manually installed software (older modules)

```
 $ module load modenv/scs5
The following have been reloaded with a version change:
  1) modenv/classic => modenv/scs5

$ module load modenv/classic
The following have been reloaded with a version change:
  1) modenv/scs5 => modenv/classic
```

Module usage

- Check `https://doc.zih..../SoftwareModulesList`
- Identify module and version

```
$module spider CP2K
----------------------------------------------------------------------------
  CP2K:
----------------------------------------------------------------------------
    Description:
      CP2K is a freely available (GPL) program[...]

     Versions:
        CP2K/5.1-intel-2018a
        CP2K/6.1-intel-2018a
     Other possible modules matches:
        cp2k
```

Module commands

`module avail` - lists all available modules (in the current module environment)

`module spider` - lists all available modules (across all module environments)

`module list` - lists all currently loaded modules

`module show <modname>` - display informations about `<modname>`

`module load <modname>` - loads module `modname`

`module save` - saves the current modules, to be reloaded at the next login

`module rm <modname>` - unloads module `modname`

`module purge` - unloads all modules

Modules for HPC applications

Loading compiler, MPI, and numeric library (MKL)

```
~> module load intel
Module intel/2018b and 8 dependencies loaded.

~> module list
Currently Loaded Modules:
  1) modenv/scs5                          (S)    6) iccifort/2018.3.222-GCC-7
  2) GCCcore/7.3.0                               7) impi/2018.3.222-iccifort-
  3) binutils/2.30-GCCcore-7.3.0                 8) iimpi/2018b
  4) icc/2018.3.222-GCC-7.3.0-2.30               9) imkl/2018.3.222-iimpi-201
  5) ifort/2018.3.222-GCC-7.3.0-2.30            10) intel/2018b

~> mpicc -show
icc -I/sw/installed/impi/2018.3.222-iccifort-2018.3.222-GCC-7.3.0-2.30/

~> mpicc hello.c

~> srun -n 4 -t 1 -N 1 --mem-per-cpu=500 ./a.out
```

Remarks

Commercial codes requiring licenses (Matlab, Ansys)

- basic principle: do not uses thes extensively, we have only a limited number of licenses!
- Matlab: use the Matlab compiler (`https://doc.zih..../Mathematics`)

Containers

- `Singularity` as container environment on Taurus
- Docker containers can easily be converted
- more information at `https://doc.zih..../Container`

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
   - General
   - Slurm examples
4. Software Development at ZIH's HPC systems
5. HPC Support

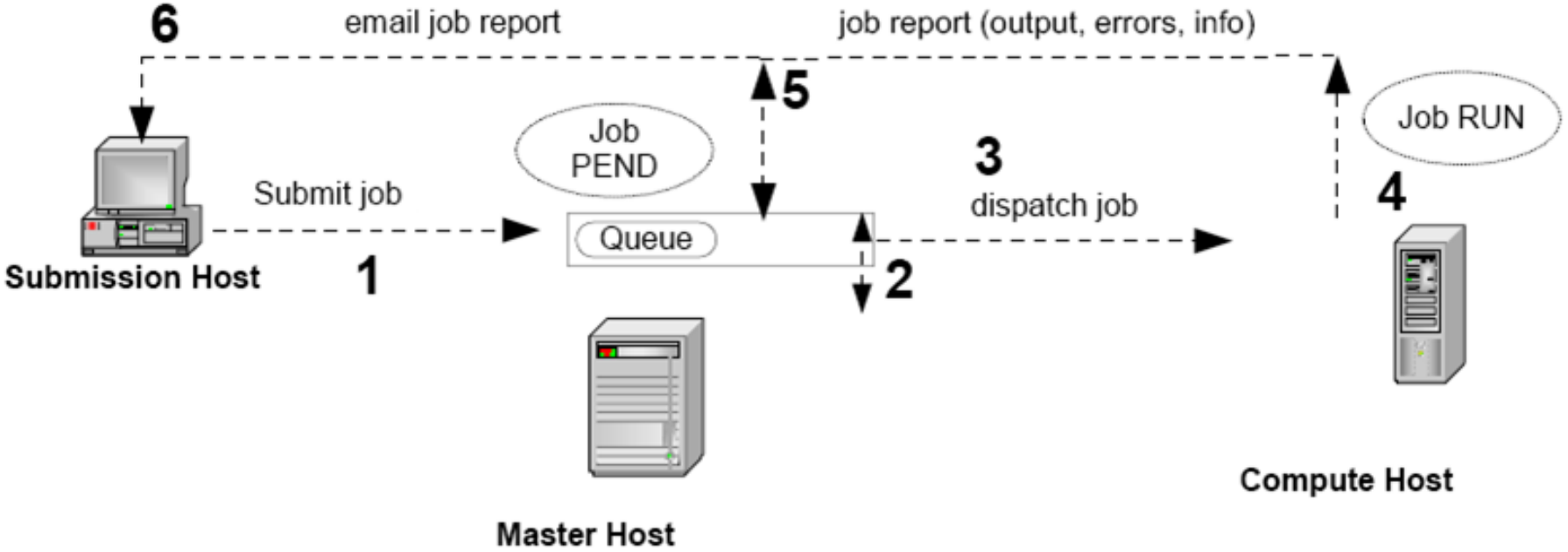
Overview

Why do we need a batchsystem?

- Find an adequate compute system (partition/island) for our needs.
- All resources in use? - The batch system organizes the queueing and messaging for us.
- Allocate the resource for us.
- Connect to the resource, transfer run-time environment, start the job.

Workflow of a batch system

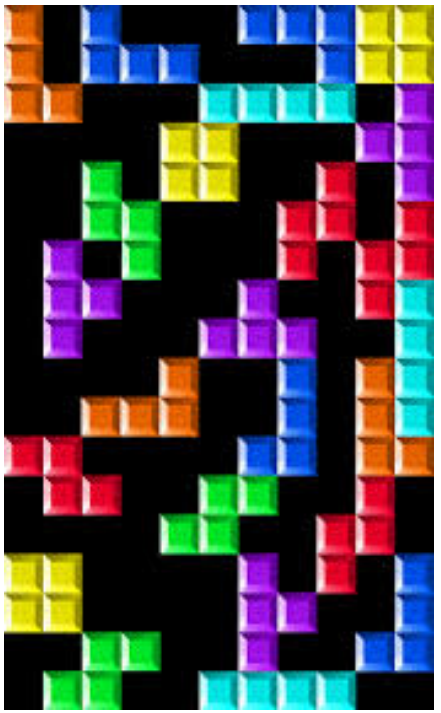
Agreed, we need a batchsystem.



Multi-dimensional optimizations

Optimization goals:

- **Users want short waiting time.**
- Queueing priorities according to:
  - waiting time of the job (+),
  - used CPU time in the last 2 weeks (-),
  - remaining CPU time for the HPC project (+),
  - duration of the job (-)
- Limited resources require efficient job placement:
  - number of compute cores / compute nodes,
  - required memory per core for the job,
  - maximum wall clock time for the job



Optimization is NP-hard $\rightarrow$ heuristics allowed.

Useful functions of a batchsystem

Basic user functions:

- submit a job,
- monitor the status of my job (notifications),
- cancel my job

Additional functions:

- check the status of the job queue,
- handle job dependencies,
- handle job arrays

Job submission: required information

In order to allow the scheduler an efficient job placement it needs these specifications:

- requirements: cores, memory per core, (nodes), additional resources (GPU)
- maximum run-time,
- HPC project (normally use primary group which gives `id`),
- who gets an email on which occasion,

... to run the job:

- executable with path and command-line,
- environment is normally taken from the submit shell.

Queueing order

Factors that determine the position in the queue:

- **Total share of the project:**
  remaining CPU quota, new project starts with 100% (updated daily)
- **Share within the project:**
  balance equal chances between users of one project
- **Age:**
  the longer a job waits the higher becomes its priority
- **Recent usage:**
  the more CPU time a user has consumed recently the lower becomes her priority,
- **Quality of Service:**
  additional control factors, e.g. to restrict the number of long running large jobs

Pre-factors are subject to adaptations by the batch system administrators.

Overview Slurm

submit a job script	`sbatch`
run interactive job	`srun --pty ...`
monitor a job status	`squeue` - Not permanently!
kill a job	`scancel`
cluster status	`sinfo` - Not permanently!
host status	`sinfo -N`
max job time	`-t <[hh:]mm:ss>`
number of processes	`-n <N>`
number of nodes	`-N <N>`
MB per core	`--mem-per-cpu`
output file	`--output=result_%j.txt`
error file	`--error=error_%j.txt`
notification (TUD)	`--mail-user <email>`
notification reason	`--mail-type ALL`

Overview Slurm

job array	`--array 3-8`
job ID	`$SLURM_ARRAY_JOB_ID`
array idx	`$SLURM_ARRAY_TASK_ID`
Interactive jobs	`--pty`
X11 forwarding	`--x11=first`

Examples for parameters for our batch systems can be found at `https://doc.zih..../Slurm`

- job arrays,
- job dependecies,
- multi-threaded jobs

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
   - General
   - Slurm examples
4. Software Development at ZIH's HPC systems
5. HPC Support

Slurm examples

Slurm interactive example:

```
srun --ntasks=1 --cpus-per-task=1 --time=1:00:00 \
     --mem-per-cpu=1000 --pty bash
```

Slurm X11 example:

```
module load matlab
srun --ntasks=1 --cpus-per-task=8 --time=1:00:00 \
     --mem-per-cpu=1000 --pty --x11=first matlab
```

Remarks:

- default partition Taurus: `-p haswell,sandy`
- normally: shared usage of resources
- if a job asks for more memory it will be canceled by Slurm automatically
- a job is confined to its requested CPUs

Slurm examples

Normal MPI parallel job `sbatch <myjobfile>`

```
#SBATCH --partition=haswell,sandy,west
#SBATCH --time=8:00:00
#SBATCH --ntasks=64
#SBATCH --mem-per-cpu=780
#SBATCH --mail-type=end
#SBATCH --mail-user=ulf.markwardt@tu-dresden.de
#SBATCH -o output_%j.txt
#SBATCH -e stderr_%j.txt
srun ./path/to/binary
```

Remark: The batch system is responsible to minimize number of nodes.

Slurm examples

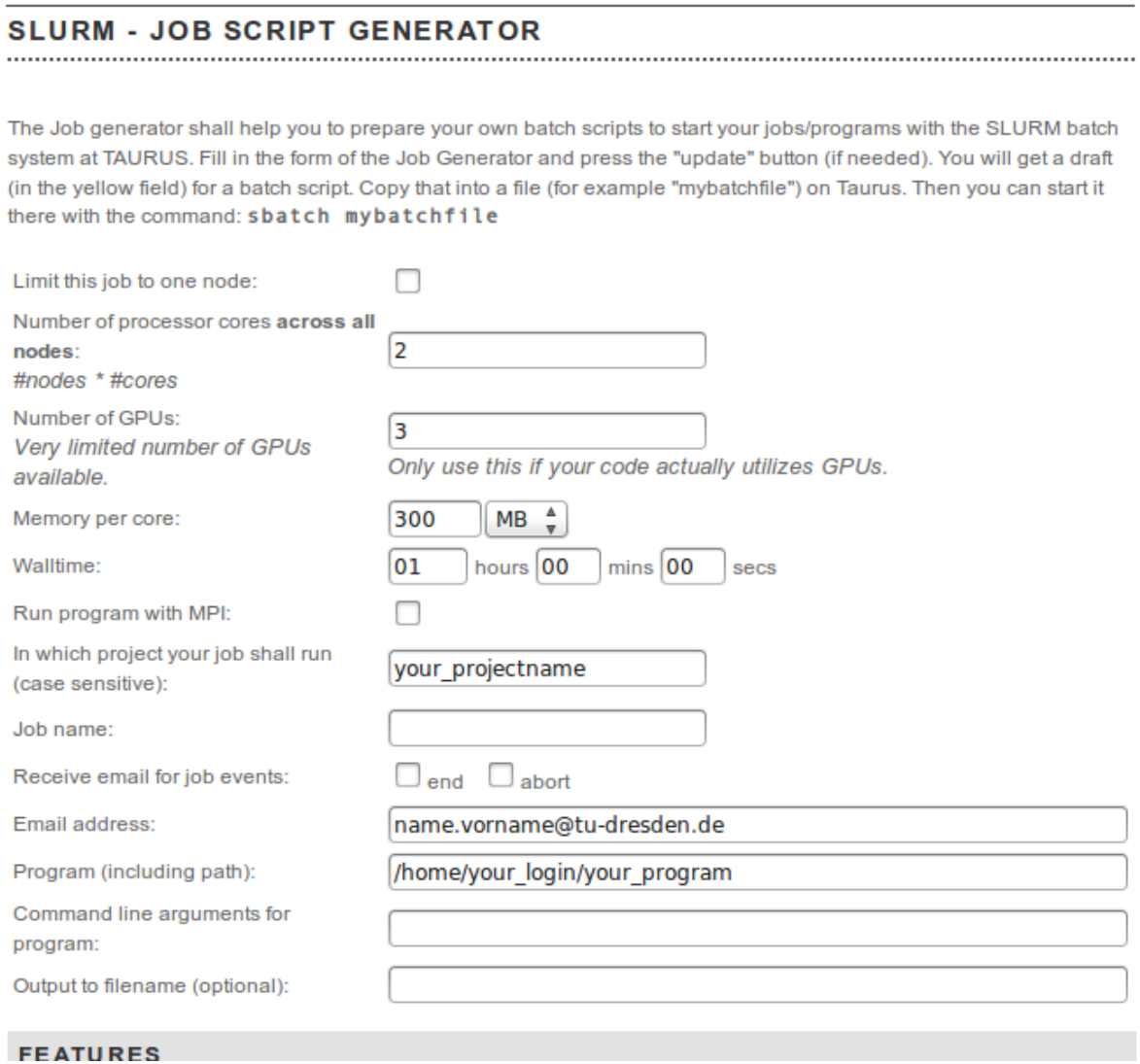
Requesting multiple GPU cards

```
#SBATCH --partition=gpu
#SBATCH --time=4:00:00
#SBATCH --job-name=MyGPUJob
#SBATCH --nodes=16
#SBATCH --ntasks-per-node=2
#SBATCH --cpus-per-task=8
#SBATCH --gres=gpu:2
#SBATCH --mem-per-cpu=3014
#SBATCH --mail-type=END
#SBATCH --mail-user=ulf.markwardt@tu-dresden.de

#SBATCH -o stdout
#SBATCH -e stderr
echo 'Running program...'
```

Slurm generator

Good starting point: `https://doc.zih..../Slurmgenerator`



SLURM - JOB SCRIPT GENERATOR

The Job generator shall help you to prepare your own batch scripts to start your jobs/programs with the SLURM batch system at TAURUS. Fill in the form of the Job Generator and press the "update" button (if needed). You will get a draft (in the yellow field) for a batch script. Copy that into a file (for example "mybatchfile") on Taurus. Then you can start it there with the command: `sbatch mybatchfile`

Limit this job to one node:	☐
Number of processor cores **across all nodes**: *#nodes * #cores*	2
Number of GPUs: *Very limited number of GPUs available.*	3 *Only use this if your code actually utilizes GPUs.*
Memory per core:	300 MB
Walltime:	01 hours 00 mins 00 secs
Run program with MPI:	☐
In which project your job shall run (case sensitive):	your_projectname
Job name:	
Receive email for job events:	☐ end ☐ abort
Email address:	name.vorname@tu-dresden.de
Program (including path):	/home/your_login/your_program
Command line arguments for program:	
Output to filename (optional):	

FEATURES

Slurm: Job monitoring

Basic question: Why does my job not start? Try: `whypending <jobid>`

```
> whypending 3402037

SLURM Reason: Priority
Reason 'Priority' means that the job can run as soon as
resources free up *and* the higher priority jobs start.

Position in queue: 86
Your priority might be low because your FairShare value
is low (0.000000).
Your raw usage during the last weeks was 16789 seconds.

Estimated start time: 2014-11-02T16:51:00

Job 3402037 is requesting:
       Time Limit: 08:00:00
            Nodes: 5
            Cores: 5
  Memory per node: 0 MB
     Total Memory: 0 MB
         Features:
              QOS: normal
```

Slurm: Fair share monitoring

Is my fair share really so low???

```
> sshare -u mark -A swtest
Accounts requested:      : swtest
Account  User  Raw Shares  Norm Shares  Raw Usage  Effectv Usage  FairShare
-------  ----  ----------  -----------  ---------  -------------  ---------
swtest                 0     0.000000     680889       0.000033   0.000000
swtest   mark      parent     0.000000      16789       0.000001   0.000000
```

Project information

Look at the login screen. Or `showquota`

```
CPU-Quotas as of 2013-10-31
   Project     Used(h)    Quota(h)      % Comment
    swtest       12241       10000 122.4 Limit reached (SOFT) *
* Job priority is minimal for this project

Disk-Quotas as of 2013-10-31
   Project    Used(GB)  Quota(GB)      % Comment
    swtest         7.1       100.0    7.1
```

As soon as a project reaches its CPU limit the share drops to 0.

As soon as a project reaches its DISK limit submission is blocked.
→ Clean up first!

What is fair...?

Fair share of a project is based on

- leftover CPU quota of the current month: *RawShare* $\rightarrow$ *NormShares*
- used resources "during the last few days" *RawUsage* $\rightarrow$ *EffektvUsage*
  CPUs usage is summed up with an exponential decay
  (half-value period 1 day)

Account	RawShares	NormShares	RawUsage	EffectvUsage	FairShare
p_abc	369	0.001355	123069773	0.034009	0.030841
p_def	342	0.001256	1962604	0.000546	0.941520

$$FairShare = 2^{\frac{-EffektvUsage}{d \cdot NormShares}}$$ (dampening factor $d = 5$).

See: `https://slurm.schedmd.com/priority_multifactor.html`

System information

Look at the login screen. Or `nodestat`

```
> nodestat
-------------------------------------------------------------------------
nodes available:     2139/2179    nodes unavailable:   40/2179
gpus  available:     499/519      gpus  unavailable:   20/519
-------------------------------------+-----------------------------------
jobs running:        3634           |   cores in use:         26355
jobs pending:        23824          |   cores unavailable:    850
jobs suspend:        0          |   gpus  in use:         231
jobs damaged:        7          |
-------------------------------------------------------------------------
                                CORES / GPUS
                      free  |   resv  |   down  | total
                     -------+---------+---------+-------
Westmere 48GB:         460  |     60  |     24  |  1656   (mem-per-cpu <= 3875
                     -------+---------+---------+-------
Sandy Bridge 32GB:     171  |      0  |    128  |  3648   (mem-per-cpu <= 1875
Sandy Bridge 64GB:       0  |     16  |      0  |   448   (mem-per-cpu <= 3875
Sandy Bridge 128GB:    110  |      0  |      0  |   224   (mem-per-cpu <= 7875
                     -------+---------+---------+-------
Haswell 64GB:          101  |  14952  |    552  | 31680   (mem-per-cpu <= 2583
Haswell 128GB:         674  |      0  |      0  |  2016   (mem-per-cpu <= 5250
Haswell 256GB:          97  |      0  |      0  |  1056   (mem-per-cpu <= 1058
[...]
```

See also `sinfo -T`.

Simple job monitoring

Detailled job information

```
> /home/mark sjob 6121181
JobId=6121181 UserId=mark(19423) Account=hpcsupport JobName=bash
   TimeLimit=00:05:00 NumNodes=11 NumCPUs=256
   TRES=cpu=256,mem=194560,node=1 Shared=0
   Partition=haswell64,sandy32
   JobState=PENDING Reason=Resources Dependency=(null)
   Priority=692 QOS=normal
   StartTime=Unknown SubmitTime=2016-03-29T16:11:30
```

Detailled job monitoring

Detailled job information

```
> /home/mark scontrol show job=6121181
JobId=6121181 JobName=Slurmtest
   UserId=mark(19423) GroupId=hpcsupport(50245)
   Priority=692 Nice=0 Account=hpcsupport QOS=normal
   JobState=PENDING Reason=None Dependency=(null)
   Requeue=1 Restarts=0 BatchFlag=0 Reboot=0 ExitCode=0:0
   RunTime=00:00:00 TimeLimit=00:05:00 TimeMin=N/A
   SubmitTime=2016-03-29T16:11:30 EligibleTime=2016-03-29T16:11:30
   StartTime=Unknown EndTime=Unknown
   PreemptTime=None SuspendTime=None SecsPreSuspend=0
   Partition=haswell64,sandy32 AllocNode:Sid=taruslogin4:18475
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=(null)
   NumNodes=11 NumCPUs=256 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
   TRES=cpu=256,mem=194560,node=1
   Socks/Node=* NtasksPerN:B:S:C=0:0:*:1 CoreSpec=*
   MinCPUsNode=1 MinMemoryCPU=760M MinTmpDiskNode=0
   Features=(null) Gres=(null) Reservation=(null)
   Shared=0 Contiguous=0 Licenses=(null) Network=(null)
   Command=bash
   WorkDir=/home/h3/mark
   Comment=<<<ZIH_JOB_STATS__REMOVE_HDF5>>>
   Switches=1@12:00:00
```

More Slurm tools

More tools for Slurm:

- `squeue` - show current queue jobs
- `sprio` - show priorities of current queue jobs
- efficiently distribute/collect data files to/from compute nodes: `sbcast`, `sgather`

See man pages or documentation at `http://slurm.schedmd.com`

Still... not starting

The system looks empty, but no job starts. Especially not mine!

- Maybe a reservation prevents my job from starting (`sinfo -T`)
- Maybe an older large job is scheduled and waits for resources:

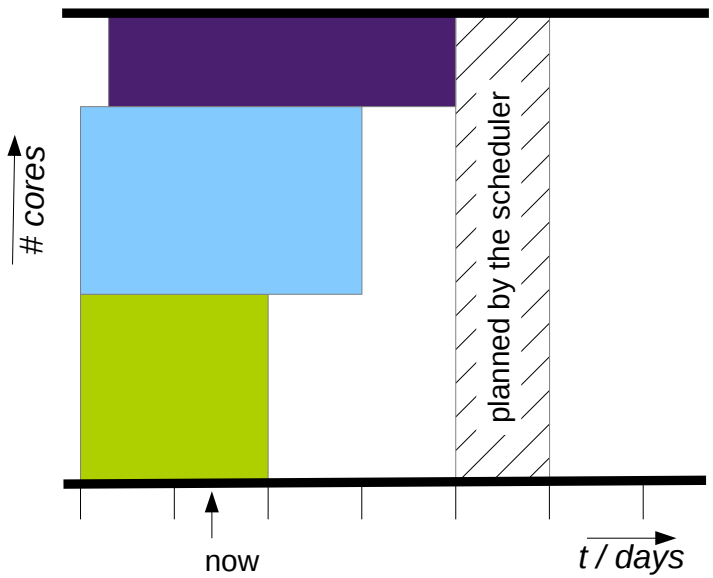
```
sprio|head
  JOBID    PRIORITY          AGE  FAIRSHARE     JOBSIZE          QOS
3402372         686          685          0           1            0
3402373         686          685          0           1            0
3402374         686          685          0           1            0

sprio|sort -k 2 -n -r |head -n 2
3404912      100186          153     100000          34            0
3406363       84432            1      84423           8            0
```

  Here is job 3404912 with a very high priority, scheduled for a certain time (see `scontrol show job=3404912`) . If my job would finish before that one it could be backfilled.

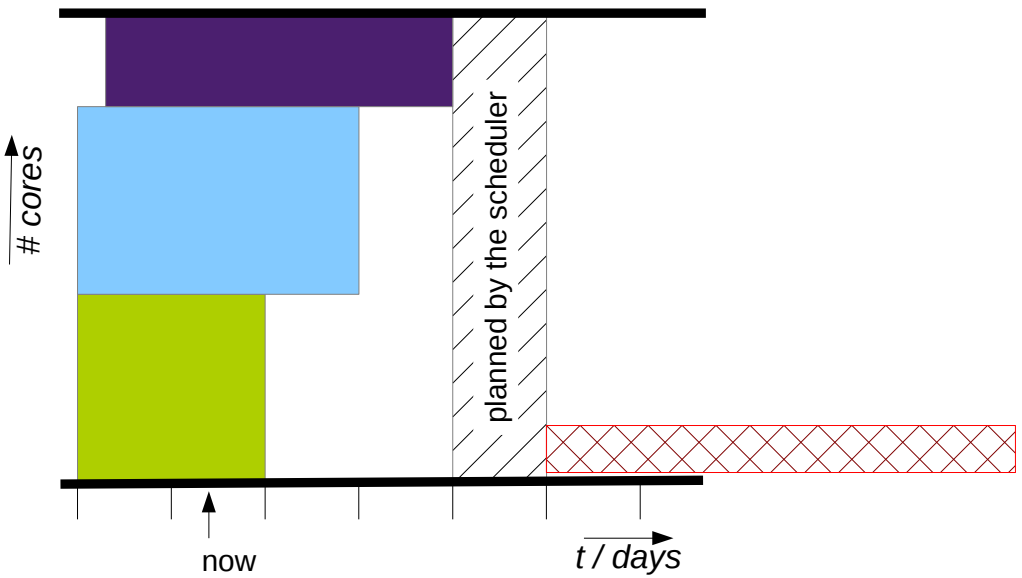
- Maybe fragmentation would be too high.

Backfilling

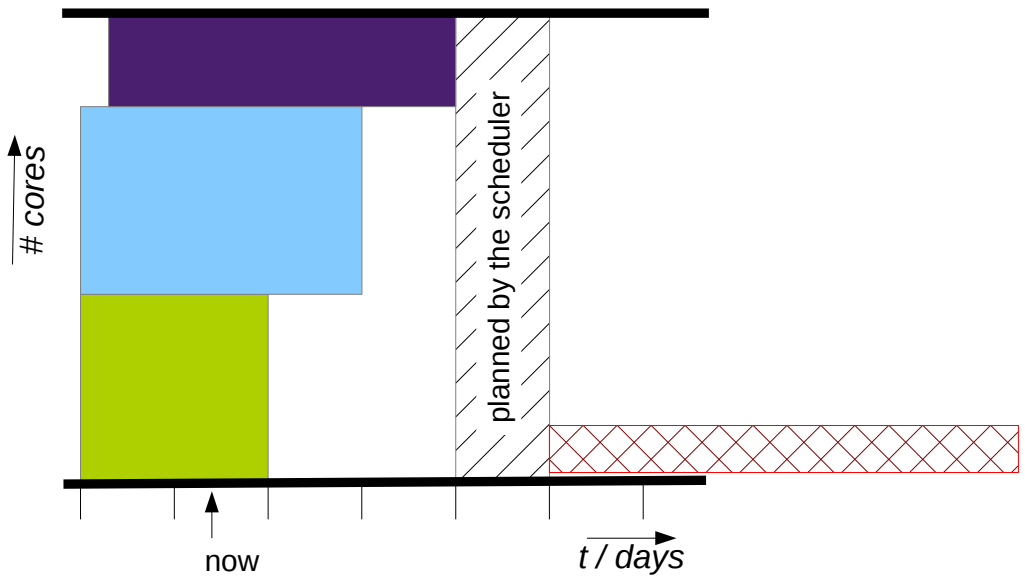


My job to be placed:

Backfilling

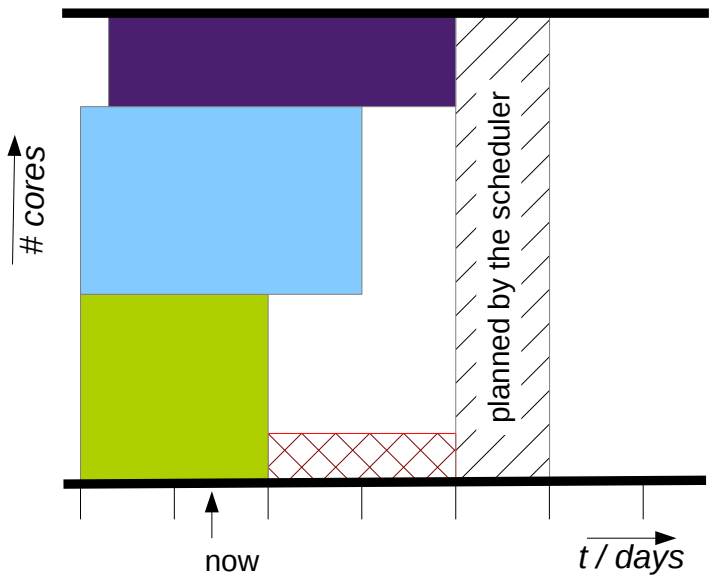


Backfilling



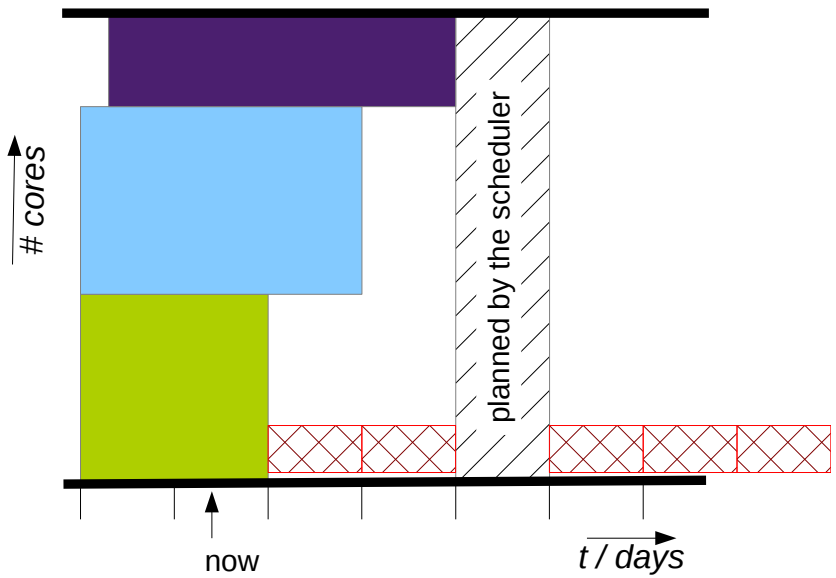
I know my job better:

Backfilling



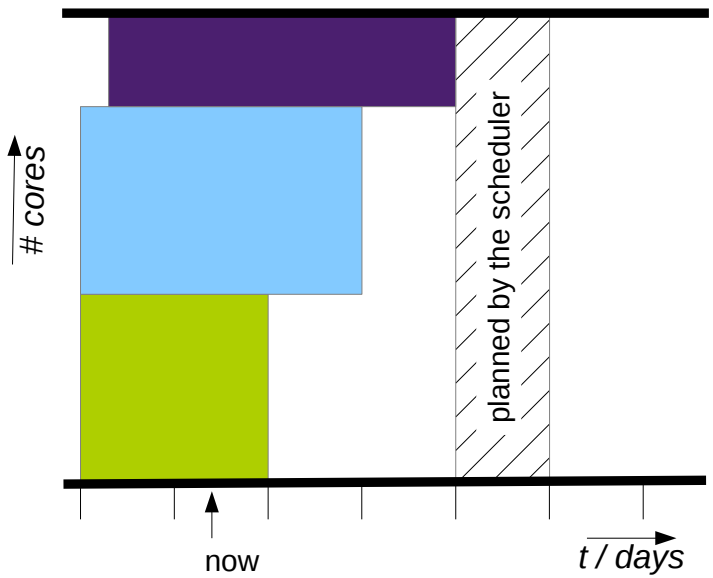
Estimate the maximum run-time of your job!

Backfilling



Try to use shorter jobs!

Backfilling



Allow checkpointing:

Checkpoint / restart

Self-developed code:

- identify best moment to dump "all" data to the file system
- implement data export and import
- implement restart

Commercial or community software

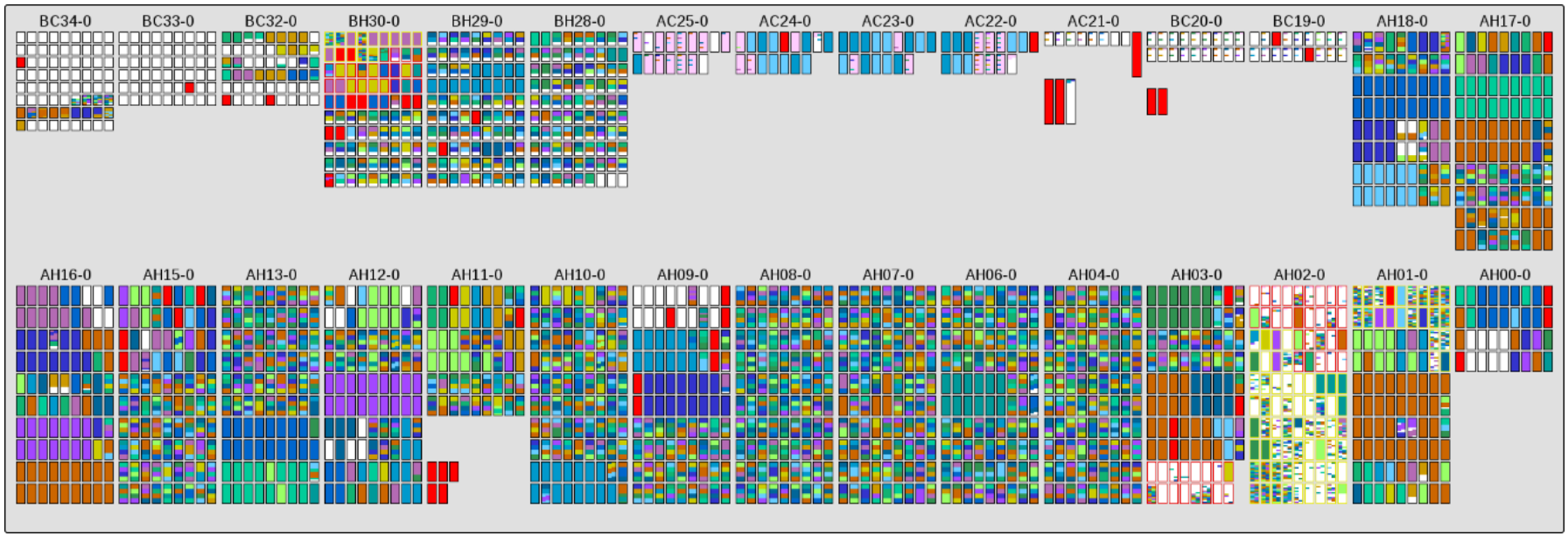
- Check if you can use built-in CR-capabilities of your application: (e.g. Abaqus, Amber, Gaussian, GROMACS, LAMMPS, NAMD, NWChem, Quantum Espresso, STAR-CCM+, VASP)
- If application does not support checkpointing:
  1. `module load dmtcp`
  2. modify your batch script like this:
     `srun dmtcp_launch --ib --rm ./my-mpi-application`
  3. run the modified script like `dmtcp_sbatch -i 28000,800 mybatch.sh`
     This creates chain jobs of length 28000 s, planning 800 s for I/O
- more details at `https://doc.zih..../CheckpointRestart`

Efficient use of resources

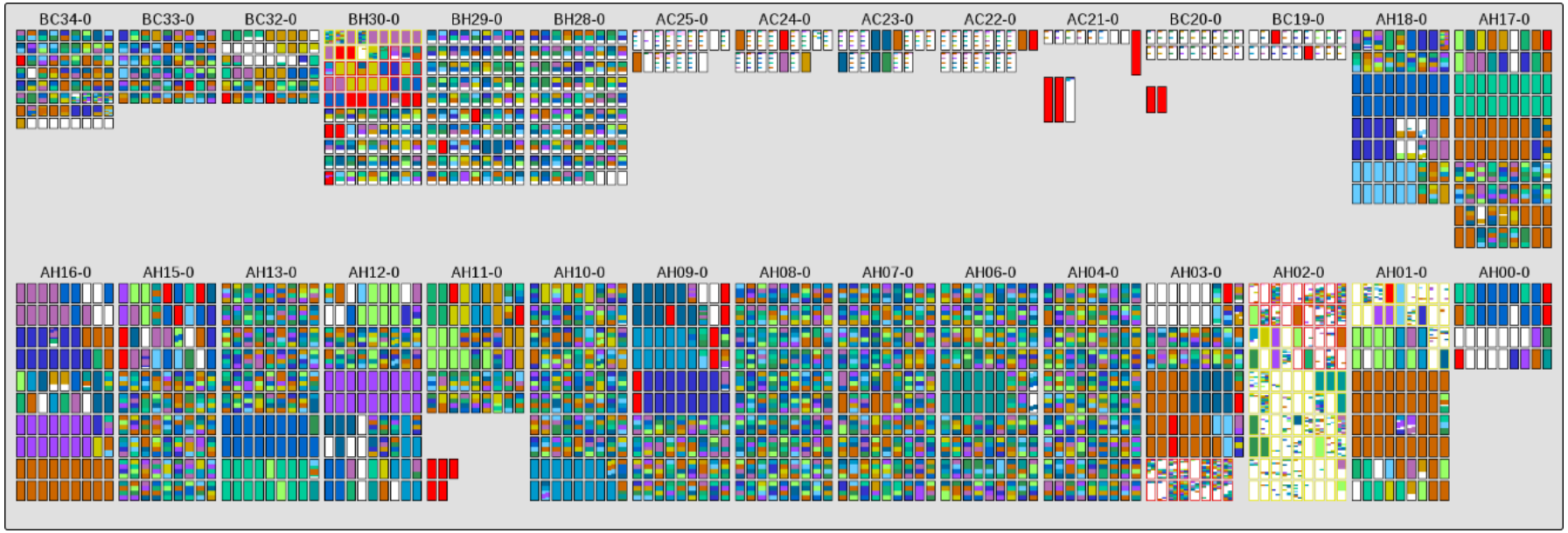
Make use of heterogeneity of the system

- number of cores per node differ (24, 32, 56, ...)
- memory per core available to the application is less then installed memory (OS needs RAM, too). Stay below the limits to increase the number of potential compute nodes for your job!
- Current numbers for Taurus:
  - 85% of the nodes have 2 GiB RAM per core. Slurm: 1875
  - 10% of the nodes have 4 GiB RAM per core. Slurm: 3995
  - 5% of the nodes have 8 GiB RAM per core. Slurm: 7942
  - 2 large SMP nodes have 32 cores, 1 TiB. Slurm: 31875
  - 5 large SMP nodes have 56 cores, 2 TiB. Slurm: 36500
  - GPU nodes: 3/2.6 GiB. Slurm: 3000/2538

Efficient use of resources



Efficient use of resources



Let Taurus work!

The batch system (Slurm) manages resources (heterogeneity) and job requirements (cores, RAM, runtime) to optimally use the system.

Normal jobs

- run without interaction (everything prepared in input data and scripts)
- start whenever resources for the particular jobs are available (+ priority)
- can run over hundreds of cores in parallel
- can run as a job array with thousands of independent single core jobs

Run-time considerations

- the larger a system the higher the chance of hitting a problem
- maximum run time: 7 days (today)
- use checkpoint / restart and chain jobs for longer computations
  - controlled by the application
  - controlled by Slurm + additional helper scripts

Working on Taurus

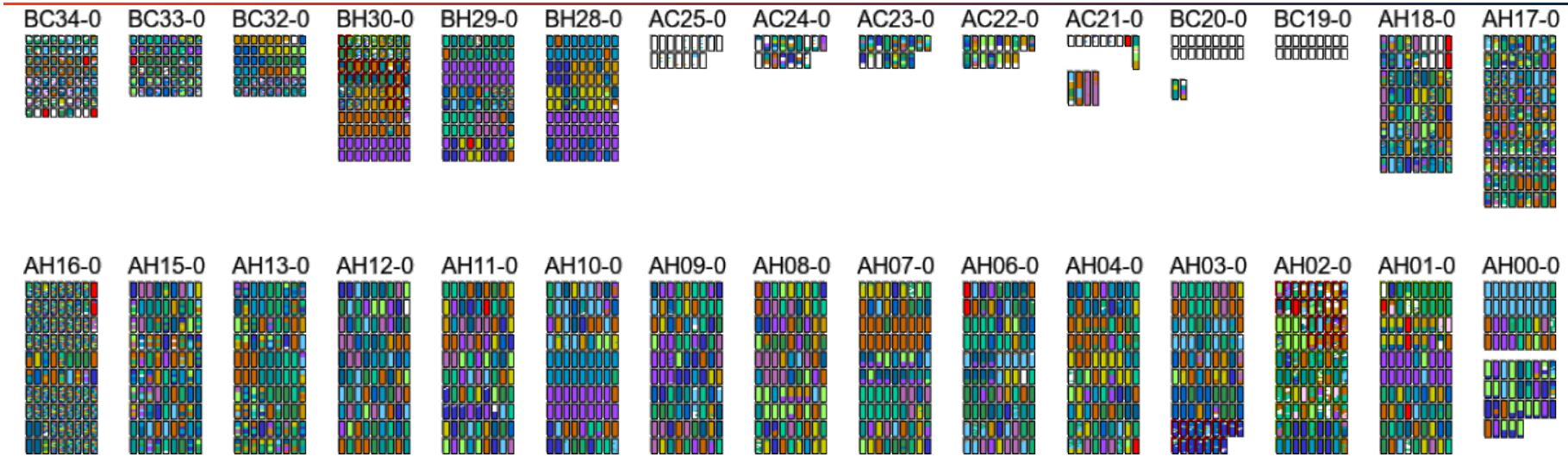
Interactive jobs

- for pre- or post- processing, compiling and testing / developmemt
- can use terminal or GUI via X11
- partition “interactive” (6 nodes) is reserved for these jobs.

Remote rendering with Nice Desktop Cloud Virtualization (DCV)

- licensed product installed on Taurus
- documentation in (`https://doc.zih..../DesktopCloudVisualization`)
- e.g. rendering with ParaView using GPUs

Availability



High utilization - good for “us” - bad for the users?

- short jobs lead to higher fluctuation (limits 1/2/7 days)
- interactive partition is nearly always empty
  - restricted to one job per user
  - default time 30 min, maximum time 8h
- plan resources in advance (publication deadline) - reserve nodes

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
   - Compiling
   - Tools
5. HPC Support

Available compilers

Which compilers are installed?

- Starting point: `https://doc.zih..../Compilers`
- Up-to-date information: `module av` at the target system

Available compilers

Which compilers are installed?

- Starting point: `https://doc.zih..../Compilers`
- Up-to-date information: `module av` at the target system

Which one is “the best”?

- Newer versions are better adapted to modern hardware.
- Newer versions implement more features (e.g. OpenMP 4.0, C++11, Fortran 2010).
- GNU compilers are most portable.
- Listen to hardware vendors. (But not always.)

Available compilers

Which compilers are installed?

- Starting point: `https://doc.zih..../Compilers`
- Up-to-date information: `module av` at the target system

Which one is “the best”?

- Newer versions are better adapted to modern hardware.
- Newer versions implement more features (e.g. OpenMP 4.0, C++11, Fortran 2010).
- GNU compilers are most portable.
- Listen to hardware vendors. (But not always.)

→ There is no such thing as “best compiler for all codes”.

Expensive operations

Time consuming operations in scientific computing:

- division, power, trigonometric and exponential functions,
- un-cached memory operations (bandwidth, latency)

Expensive operations

Time consuming operations in scientific computing:

- division, power, trigonometric and exponential functions,
- un-cached memory operations (bandwidth, latency)

How to find performance bottlenecks?

- Tools available at ZIH systems (perf, hpctoolkit, Vampir, PAPI counters),
- `https://doc.zih..../PerformanceTools`
- experience...
- Ask ZIH staff about your performance issues!

Low hanging fruits

What is the needed floating point precision?
32 bit vs. 64 bit impacts on

- memory footprint,
- computing speed.

Low hanging fruits

What is the needed floating point precision?
32 bit vs. 64 bit impacts on

- memory footprint,
- computing speed.

What is the needed floating point accuracy?

- very strict (replicable),
- slightly relaxed (numerical stability),
- very relaxed (aggressive optimizations)

Low hanging fruits

What is the needed floating point precision?
32 bit vs. 64 bit impacts on

- memory footprint,
- computing speed.

What is the needed floating point accuracy?

- very strict (replicable),
- slightly relaxed (numerical stability),
- very relaxed (aggressive optimizations)

→ see man pages!

Default Intel options (Taurus): "-axavx -msse4.2" generate code for SSE 4.2 and optional optimizations for AVX (vectorization).

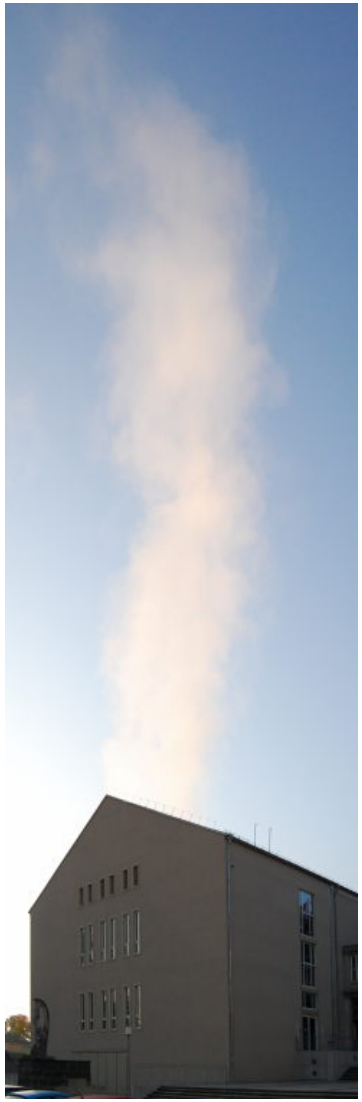
Intel training course: `https://doc.zih..../SoftwareDevelopment`

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
   - Compiling
   - Tools
5. HPC Support

On HPC systems: Efficient code is essential!

- the same code is running for several 1000 CPUh
- use of multiple CPUs sometimes does not help (wrong parallelization or job placement)
- parallel scalability



Profiling

. . . is a form of *dynamic program analysis*.

Profiling allows you to learn

- . . . where your (?) program has spent its time . . .
- . . . which functions have called which other functions . . .
- . . . how often each function is called . . .

while it was executing.

$\rightarrow$ Identify slow code – redesign it!

Profiling

. . . is a form of *dynamic program analysis*.

Profiling allows you to learn

- . . . where your (?) program has spent its time . . .
- . . . which functions have called which other functions . . .
- . . . how often each function is called . . .

while it was executing.

→ Identify slow code – redesign it!

Profiling has an impact on performance, but relative performance should be consistent.

Using GNU's gprof

part of GCC available on most unix systems

- compiling and linking (`-pg`):

```
g++ -pg my_prog.cpp -o my_prog.out
```

- execute to produce profiling information:

```
./my_prog.cpp
```

- get human readable information:

```
gprof my_prog.out gmon.out > analysis.txt
```

- analysis: `vi analysis.txt`

```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  s/call   s/call  name
 34.70     16.42    16.42        1    16.42    16.42  func3
 33.52     32.29    15.86        1    15.86    15.86  func2
 26.97     45.05    12.76        1    12.76    29.19  func1
  0.13     45.11     0.06                             main
```

Comment: see also Intel slides.

PerfTools

Support for sampling applications and reading performance counters.

- available on all systems via `module load gperf`
- perf consists of two parts
  - kernel space implementation
  - user space tools: `perf stat`, `perf record`, `perf report`

PerfTools

`perf stat`

- general performance statistics for a program
- attach to a running (own) process or monitor a new process

```
 Performance counter stats for 'ls':
  1,726551 task-clock                #    0,141 CPUs utilized
         4 context-switches          #    0,002 M/sec
         0 cpu-migrations            #    0,000 K/sec
       260 page-faults               #    0,151 M/sec
 5.573.264 cycles                    #    3,228 GHz
 3.778.153 stalled-cycles-frontend   #   67,79% frontend
                                     #          cycles idle
 2.675.646 stalled-cycles-backend    #   48,01% backend
                                     #          cycles idle
 3.671.699 instructions              #    0,66  insns per cycle
                                     #    1,03  stalled
                                     #          cycles per insn
   736.259 branches                  #  426,433 M/sec
    19.670 branch-misses             #    2,67% of all branches

 0,012276627 seconds time elapsed
```

PerfTools

`perf record`

- sample an application (or a system)
- records CPU, instruction pointer and call graph (`--call-graph`)
- compile your code with debug symbols

`perf report`

- analyze sampling result with `perf report`

Example:

```
perf record --call-graph ./my_prog
perf report perf.data
```

→ Useful documentation `perf --help` and at
`https://doc.zih..../PerfTools` .

SLURM profiling with HDF5 (on Taurus)

SLURM offers the option to gather profiling data from every task/node of the job.

- task data, i.e. CPU frequency, CPU utilization, memory consumption, I/O
- energy consumption of the nodes - subject of HDEEM research project
- Infiniband data (currently deactivated)
- Lustre filesystem data (currently deactivated)

The aggregated data is stored in an HDF5 file in `/scratch/profiling/${USER}`.

SLURM profiling with HDF5 (on Taurus)

SLURM offers the option to gather profiling data from every task/node of the job.

- task data, i.e. CPU frequency, CPU utilization, memory consumption, I/O
- energy consumption of the nodes - subject of HDEEM research project
- Infiniband data (currently deactivated)
- Lustre filesystem data (currently deactivated)

The aggregated data is stored in an HDF5 file in `/scratch/profiling/${USER}`.

Caution:

- Profiling data may be quite large. Please use `/scratch` or `/tmp`, not HOME.
- Don't forget to remove the `--profile` option for production runs!
  Penalty is a round of ice cream with strawberries for the support team.

SLURM profiling with HDF5

Example

- Create task profiling data:

```
srun -t 20 --profile=Task --mem-per-cpu=2001 \
  --acctg-freq=5,task=5 \
  ./memco-sleep --min 100 --max 2000 --threads 1 --steps 2
```

- Merge the node local files (in `/scratch/profiling/${USER}`) to a single file (maybe time-consuming):
  - login node: `sh5util -j <JOBID> -o profile.h5`
  - in jobscripts:
    `sh5util -j ${SLURM_JOBID} -o /scratch/ws/mark-prof/profile.h5`

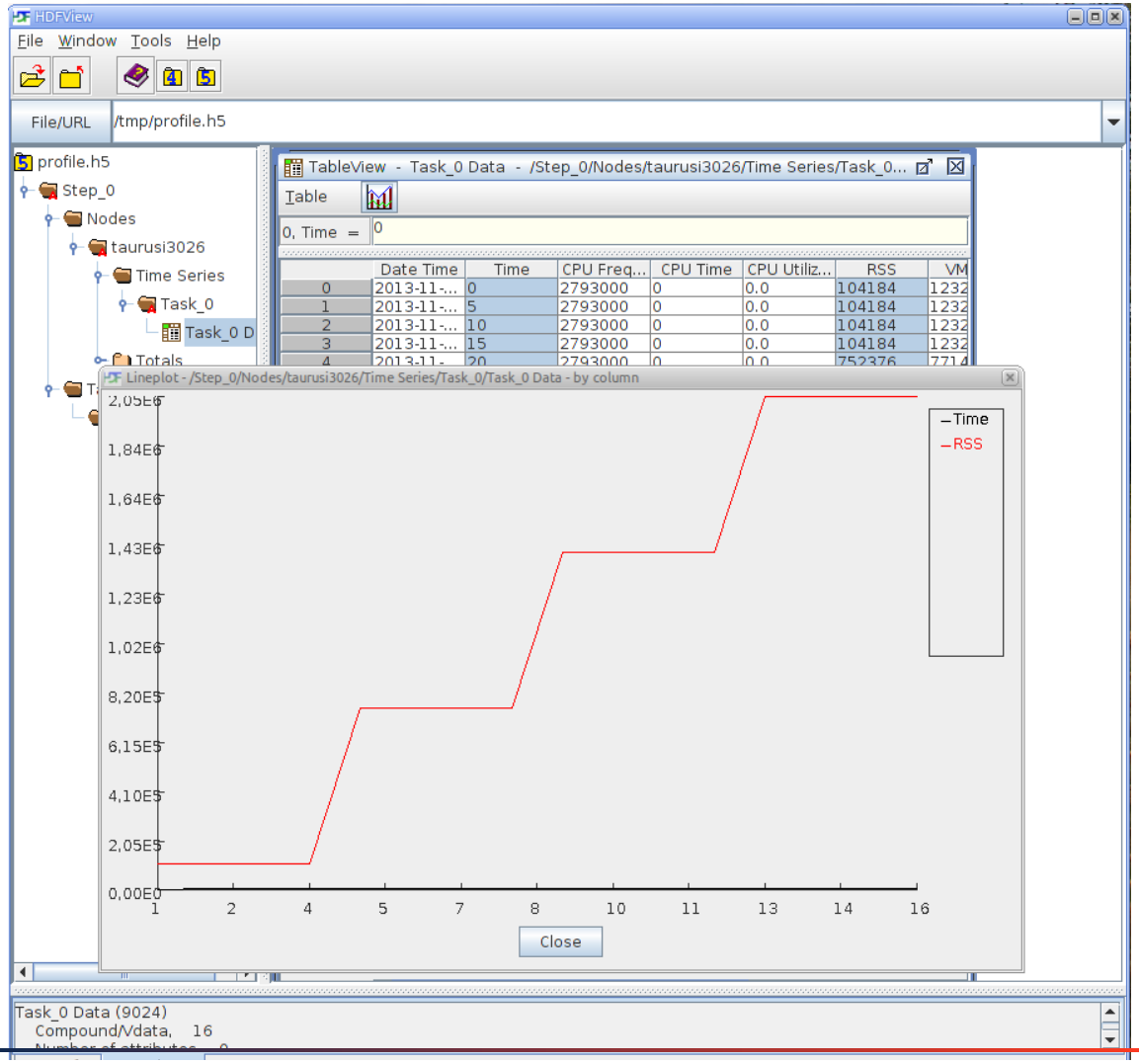
External information:

`http://slurm.schedmd.com/hdf5_profile_user_guide.html`
`http://slurm.schedmd.com/sh5util.html`

SLURM profiling with HDF5

View example data

```
module load hdf5/hdfview; hdfview.sh /scratch/ws/mark-prof/profile.h5
```



Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support
   - Management of HPC projects
   - Channels of communication
   - Kinds of support
   - Beyond support

Start a new project

Two steps for project application:

1. online application form
   - with or without existing ZIH login (select institute)
   - head of the project (universities: chair)
   - needed resources (CPUh per month, permanent disk storage...)
   - abstract

   After a technical review the project will be enabled for testing and benchmarking with up to 3500 CPUh/month.

Start a new project

Two steps for project application:

1. online application form
   - with or without existing ZIH login (select institute)
   - head of the project (universities: chair)
   - needed resources (CPUh per month, permanent disk storage...)
   - abstract
2. full application (3-4 pages pdf):
   - scientific description of the project
   - prelimiary work, state of the art...
   - objectives, used methods
   - software, estimation of needed resources and scalability

Management of HPC projects

Who...

- project leader (normally chair of institute) → accountable
- project administrator (needs HPC login) → responsible

What...

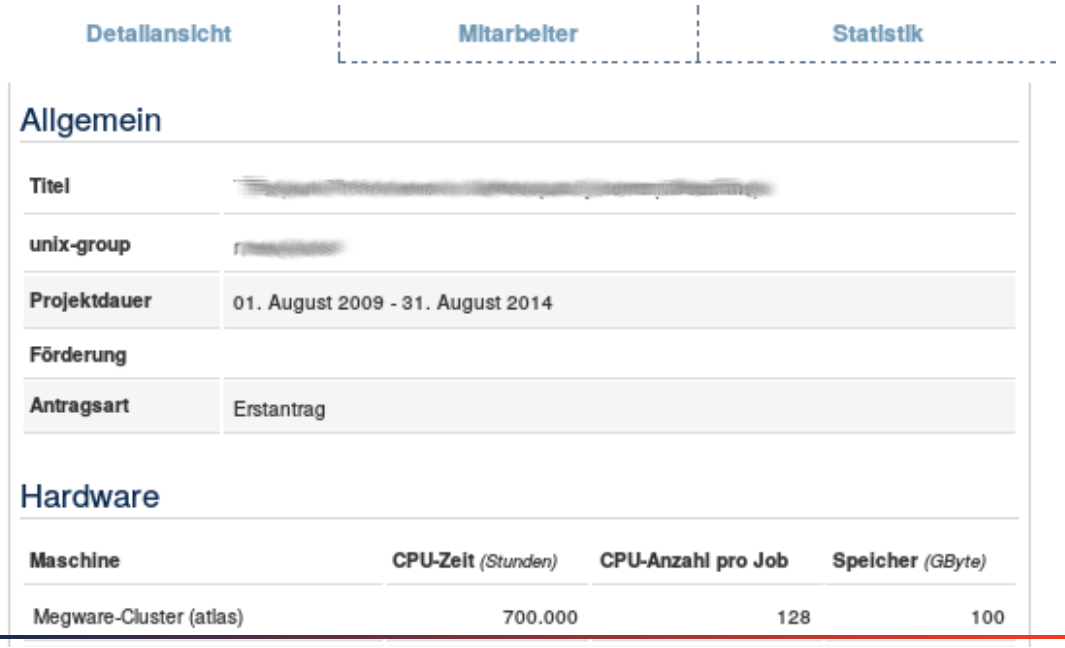
- manage members of the project (add + remove)
  (remark: external users need login..)
- check storage consumption within the project,
- retrieve data of retiring members
- contact for ZIH

Online project management

Web access: `https://hpcprojekte.zih.tu-dresden.de/managers`

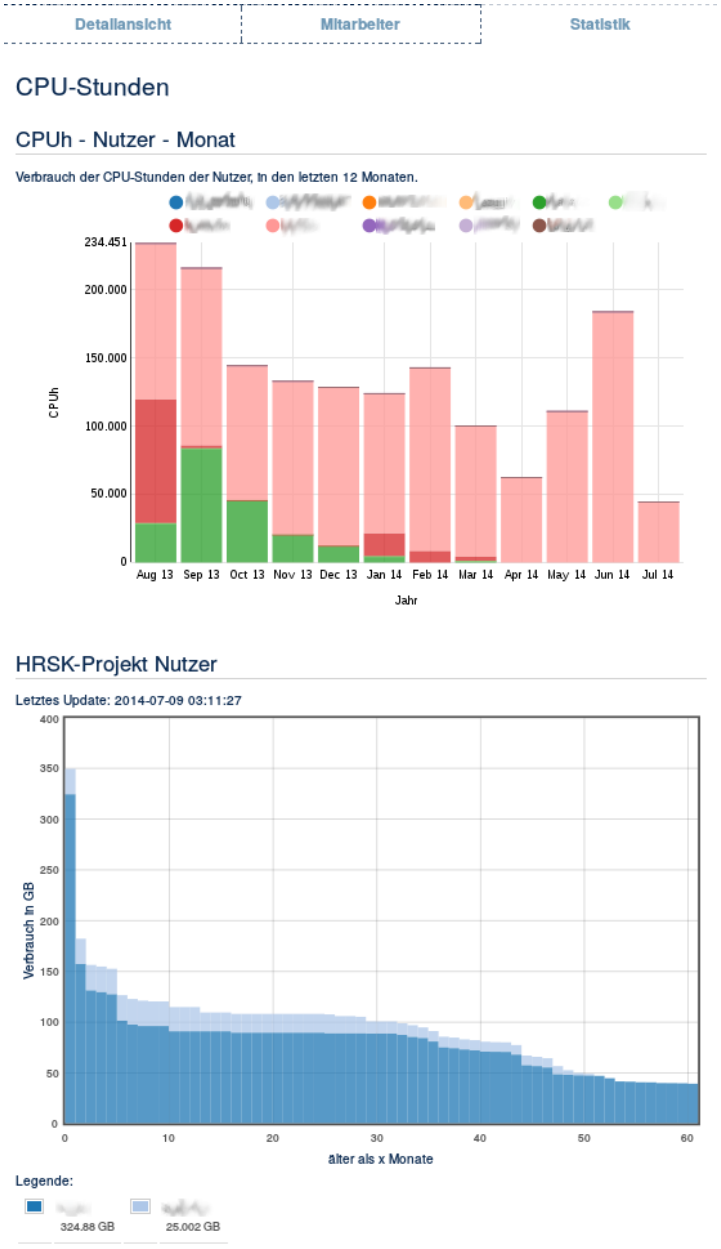
The front-end to the HPC project database enables the project leader and the project administrator to

- add and remove users from the project,
- define a technical administrator,
- view statistics (resource consumption),
- file a new HPC proposal,
- file results of the HPC project.



Online project management

Online project management



Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support
   - Management of HPC projects
   - Channels of communication
   - Kinds of support
   - Beyond support

Channels of communication

ZIH → users:

- training course "Introduction to HPC at ZIH"
- HPC wiki: `https://doc.zih.tu-dresden.de`
  - link to the operation status,
  - knowledge base for all our systems, howtos, tutorials, examples...
- mass notifications per signed email from the sender "[ZIH] HPC Support" to your address `...@mailbox.tu-dresden.de` or `...@tu-dresden.de` for:
  - problems with the HPC systems,
  - new features interesting for all HPC users,
  - training courses
- email, phone - in case of requests or emergencies (e.g. user stops the file system).

Channels of communication

User → ZIH

- If the machine feels "completely unavailable" please check the operation status first. (Support is notified automatically in case a machine/file system/batch system goes down.)
- Trouble ticket system:
  - advantages
    - reach group of supporters (independent of personal availability),
    - issues are handled according to our internal processes,
  - entry points
    - email: `servicedesk@tu-dresden.de` or `hpcsupport@zih.tu-dresden.de`
      please: use your `...@tu-dresden` address as sender and voluntarily include: name of HPC system, job ID...
    - phone: service desk (0351) 463 40000
    - planned: self service portal
- personal contact
  - phone call, email, talk at the Mensa
  - socializing is fine... but: risk of forgetting

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support
   - Management of HPC projects
   - Channels of communication
   - Kinds of support
   - Beyond support

Kinds of support

HPC management topics:

- HPC project proposal,
- login,
- quota, accounting etc.

HPC usage requests:

- Why does my job not start? - and other questions concerning the batch system
- Why does my job crash?
- How can I ...

Kinds of support

HPC Software questions:

- help with the compiling of a new software
- installation of new applications, libraries, tools
- update to a newer / different version

→ restrictions of this support:

- only if several user groups need this
- no support for a particular software
- allow for some time

Kinds of support

Performance issues

- joint analysis of a piece of SW
- discussion of performance problems
- detailed inspection of self-developed code
- in the long run: help users to help themselves

Storage and workflow issues

- joint analysis of storage capacity needs
- joint development of a storage strategy
- joint design of workflows

Kinds of support

Scalable Data Services and Solutions – Dresden-Leipzig
ScaDS support for data analytics:

- data analysis tools (parallel R/Python, RStudio, Jupyter, etc.)
- Big Data Frameworks (Apache Hadoop, Spark, Flink, etc.)
- software for Deep Learning (TensorFlow, Keras, etc.)
- survey of performance optimization of the mentioned software

`https://www.scads.de/services` or `services@scads.de`

HPC Support Team

HPC support group

- Claudia Schmidt (project management)
- Matthias Kräußlein (accounting and project infrastructure)
- Maik Schmidt, Michael Müller, Etienne Keller (technical support)
- Danny Rotscher (Slurm, technical support)
- Ulf Markwardt (Slurm, technical support... head of the group)
- Jörg Weller (Venus and file systems)

Agenda

1. Linux from the command line
2. HPC Environment at ZIH
3. Batch System
4. Software Development at ZIH's HPC systems
5. HPC Support
   - Management of HPC projects
   - Channels of communication
   - Kinds of support
   - Beyond support

Beyond support

ZIH is state computing centre for HPC

- hardware funded by DFG and SMWK
- collaboration between (non-IT) scientists and computer scientists
- special focus on data-intensive computing

Joint research projects

- funded by BMBF or BMWi
- ScaDS Dresden Leipzig
- Nvidia CCoE (GPU), IPCC (Xeon Phi)

Research topics

Scalable software tools to support the optimization of applications for HPC systems

- Data intensive computing and data life cycle
- Performance and energy efficiency analysis for innovative computer architectures
- Distributed computing and cloud computing
- Data analysis, methods and modeling in life sciences
- Parallel programming, algorithms and methods

You can help

If you plan to publish a paper with results based on the used CPU hours of our machines please acknowledge ZIH like...

The computations were performed on an HPC system at the Center for Information Services and High Performance Computing (ZIH) at TU Dresden.

We thank the Center for Information Services and High Performance Computing (ZIH) at TU Dresden for generous allocations of compute resources.

Recapitulation

Most important topics:

- Use the correct file system.
- Hand over the requirements of your application to the batch system.
- Plan your needed resources in advance.

- You are responsible for your application and your data.
  We can help you.
- Please acknowledge ZIH and send us the publication.