Training Practitioners for Effective Use of HPC Systems: Experience from the Offered Courses

Teaching High Throughput Computing

Anja Gerbes

Goethe University of Frankfurt

July 30, 2019

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Center for Scientific Computing Frankfurt

Hessisches Kompetenzzentrum für Hochleistungsrechnen

Activities

- 09/2015 — start of my employment contract at Goethe University Frankfurt & HKHLR
- 5 months
- 02/2016 — start of course sessions
- 20 months
- 10/2017 — Talk on eResearchAustralasia Conference in Australia
- 7 months

- 05/2018 — start membership at HPCCF
- 14 months
- 07/2019 — Workshop: HPC training, education and documentation at RRZ
- 17 months
- 12/2020 — end of contract

User Feedback

Center for Scientific Computing Frankfurt

Questionnaire

Hessisches Kompetenzzentrum für Hochleistungsrechnen

Anja Gerbes | **Introductory Courses in Frankfurt** | Date: 14.02.2019

Evaluation of the Talk

How would you evaluate. . .	very bad					very good
. . . the content and target of the workshop?	☐	☐	☐	☐	☐	☐
actuality	☐	☐	☐	☐	☐	☐
comprehensibility	☐	☐	☐	☐	☐	☐
relevance of content	☐	☐	☐	☐	☐	☐
practical relevance	☐	☐	☐	☐	☐	☐
handout	☐	☐	☐	☐	☐	☐
. . . the professional competence of the course instructor?	☐	☐	☐	☐	☐	☐
. . . the presentation?	☐	☐	☐	☐	☐	☐
. . . the methodical-didactic competence with regard to . . .						
the structure of learning content	☐	☐	☐	☐	☐	☐
and it's presentation?	☐	☐	☐	☐	☐	☐
. . . the participant orientation?	☐	☐	☐	☐	☐	☐
. . . the equipment and environment?	☐	☐	☐	☐	☐	☐
. . . the course length?	☐	☐	☐	☐	☐	☐
. . . the course depth?	☐	☐	☐	☐	☐	☐

Do you wish further courses on this subject?	☐Yes	☐No
Would you recommend this course?	☐Yes	☐No
Are you using the material later on?	☐Yes	☐No

What did you like about the lecture?

Which ideas and suggestions do you have for this lecture?

What content would you have additionally preferred?

How did you like the exercises?

Which course . . . you join?	did	would
UNIX	☐	☐
TOOLS	☐	☐
GIT	☐	☐
SHELL	☐	☐
CLUSTER	☐	☐
PYTHON	☐	☐
CPP	☐	☐
TOTALVIEW	☐	☐
HPC	☐	☐
LIKWID	☐	☐
VAMPIR	☐	☐
MATLAB	☐	☐

Is there any topic missing that you are interested in?

Follow up courses in Python

Would you be interested in a follow-up course about . . .

. . . TDD with Python?

☐ Yes ☐ No

. . . Python project development?

☐ Yes ☐ No

. . . Other python related topic? Which one?

http://csc.uni-frankfurt.de
http://www.hpc-hessen.de

Center for Scientific Computing
Hessisches Kompetenzzentrum für Hochleistungsrechnen

User has to

- evaluate the course session (content, talk)
- choose the course session
- choose follow up session (python, cpp)
- personal anonymous information
- choose your high performance computing experience

Reference Guide

Hessisches Kompetenzzentrum für Hochleistungsrechnen — Version 2.0 February 14, 2019

Cluster | quick reference | **Frankfurt**

Center for Scientific Computing Frankfurt — GOETHE UNIVERSITÄT FRANKFURT AM MAIN

1 Cluster Hardware

Access — Cluster Frankfurt

```
ssh <username>@goethe.hhlr-gu.de                GOETHE-HLR
```

Go to CSC-Website/Access/Goethe-HLR to get an account. The project manager has to send a request to submission@csc.uni-frankfurt.de to get CPU-Time for research projects. Further information at our website.

Architecture & Constraints — Cluster Frankfurt

#nodes	CPU	GHz	# CPUs/Cores	RAM
400	Intel Xeon Skylake Gold 6148	2.10	2/40	192GB
72	Intel Xeon Skylake Gold 6148	2.10	2/40	768GB
198	Intel Xeon E5-2670v2 Ivy Bridge	2.50	2/20	128GB
139	Intel Xeon E5-2640v2 Broadwell		2/20	128GB
50	Intel Xeon E5-2650v2 Ivy Bridge	2.60	2/12	128GB

Intel Xeon E5-2650v2 Ivy Bridge has 2xAMD FirePro S10000 12GB GPUs.

The architecture will be selectable via the '`--constraint`' option,

`ivy` = dual-socket Intel Ivy Bridge CPU nodes,

`broadwell` = dual-socket Intel Broadwell CPU nodes,

File Systems — storage systems

mountpoint	/home	/scratch	/local	/arc[1|2]
size	10GB PU	764 TB	1.4 T	500TB each
access time	slow	fast	fast	slow
system	NFS	FhGFS	ext3	NFS
network	Ethernet	InfiniBand		Ethernet

Contact — HPC Frankfurt

If you have any HPC-questions about SLURM and want help by debugging & optimizing your program, please write to hpc-support@csc.uni-frankfurt.de.
Else, you can contact the system administrators if you need software to be installed: support@csc.uni-. . . .
Detailed documentation on using the cluster can be found at CSC-Website.

Partitions — Cluster Frankfurt

partition	run time	Max Nodes	Max NodesPU	Max JobsPU	Max SubmitPU	Intel Nodes
general1	21d	475	150	40	50	skylake
general2	21d	337	150	40	50	broadwell ivybridge
gpu	21d	50	50	40	50	
test	1h	2-12		10	10	

To view such informations on the cluster, use the command:

```
sacctmgr list QOS partition format=maxnodes,maxnodesperuser
    ,maxjobsperuser,maxsubmitjobsperuser
scontrol show partition
sinfo -p partition
squeue -p partition
```

Per-User Resource Limits — Cluster Frankfurt

limit	description	
MaxNodes	max No of nodes	
MaxNodesPU	max No of nodes to use at the same time	
MaxJobsPU	max No of jobs to run simultaneously	
MaxSubmitPU	max No of jobs in running or pending state	
MaxArraySize	max job array size	1001

2 Cluster Usage

How-To-Compile

C1	install spack
C2	source spack
C3	compile software
C4	prepare module file

How-To-Run

R1	load module file
R2	write bash script
R3	submit job with slurm

Getting Help

Cluster Frankfurt
You will find further information about usable commands on the clusters with `man <command>`.

Spack `https://spack.io/`
You will find further information about usable commands of spack with `spack --help`.

3 Software Handling

Module — setting program environments — R1

Syntax:	`module <command> <modulename>`
`avail`	display all available modules
`list`	display all loaded modules
`load | add <module>`	load a module
`load unstable`	load a deprecated or unstable module
`unload | rm <module>`	unload a module
`switch | swap <old-module> <new-module>`	switch modules
`purge`	unload all currently loaded modules

How-To — use custom modules

1. writing a module file in tcl to set environment variables
2. `module load use.own` enables you to load your own modules
3. `module load ~/privatemodules/modulename`
4. use facilities provided by module

Installation Spack itself — C1

```
1  git clone https://github.com/spack/spack.git
2  cd spack
```

Basic Spack Usage — C2

`. share/spack/setup-env.sh` Has to be made after each login

`echo ". spack/share/spack/setup-env.sh" >> ~/.bash_profile`
If you want this to be permanent

`- /scratch/<your-project-name>/<your-user-name>/spack/tmp`
add to build_stage in `config.yaml`

Managing Modules

`module avail`
installed sofware packages
`spack install lmod`
to clean things up

Spack Workflow

W1	building packages
W2	running binaries
W3	developing software

`http://csc.uni-frankfurt.de`
`http://www.hpc-hessen.de`

Center for Scientific Computing
Hessisches Kompetenzzentrum für Hochleistungsrechnen

It is our duty to offer courses at universities
at regular intervals and to offer variety with new topics.

Cluster Computing Course

Cluster Computing Course

1. Cluster Hardware
   - Access/Organization/Architecture to a Cluster
   - Hardware Resources (#nodes, CPU, #Cores, RAM)
   - Partition (Slurm options: `--partition`, `--constraint`)
   - Hyperthreading on a Cluster (Slurm options: `--extra-node-info`)
   - Filesystem
   - Getting Help (Help, Documentation, Support, How-To-Compile, How-To-Run, How-To-Workflow)

1.2 Software Handling

- Definition: Environments Modules

1.2.1 Basic Packages provided by CSC

- Information: Which Compilers, MPIs and Libraries are installed?

1.2.2 Modules Provided by Spack

- How-To-Spack
- Definition: How-To-Lmod

1.2.3 own Modules

- How-To-Own-Modules
- Usage of Intel Compiler

Cluster Computing Course

3.1 Slurm Basic

- Batch System Concepts (Cluster, Resource Manager, Scheduler, Batch System, Slurm, Batch Processing, Job, Job Steps)
- Definition: Slurm
- Slurm Commands (`sbatch`, `salloc`, `srun`, `scancel`, `sinfo`, `squeue`, `scontrol`, `sacct`, `sacctmgr`)
- Backfilling Scheduling Algorithm
- How-To-Job-Submission
- Slurm options (`--nodes`, `--ntasks`, `--ntasks-per-node`, `--ntasks-per-socket`, `--ntasks-per-core`, `--cpus-per-task`, `--mem`,`--mem-per-cpu`, `--nodelist`, `--time`, `--job`, `--output`, `--error`,`--mail-type`, `--constraint`, `--partition`, `--extra-node-info`)

Cluster Computing Course

3.2 Practical Job Submission

- Job Script Examples:
  - simple job starting one process
  - going parallel
  - going parallel across nodes
  - serial job
- How-To-Creating-Parallel-Job-in-Slurm
- Job Script Example:
  - MPI job HT off vs. MPI job HT on
  - Hybrid Job (MPI + OpenMP)
  - MPI Job Steps, HT off
  - How-To-Job-Array
  - OpenMP Job
- How-To-Submitting-a-Batch-Script with Visualization
  - OpenMP Job
  - MPI Job
  - Hybrid Job (MPI + OpenMP)

3.3 Advanced CPU Management

- Differences: Allocation, Distribution, Core Binding
- Slurm options (`--exclusive`, `--share`, `--account`, `--array`, , `--account` , `--tasks-per-node`, `--mem_bind`, `--mail-user`, `--distribution`, `--bind-to-core`, `--bind-to-socket`, `--bind-to-none`, `--cpus-per-proc`, `--report-bindings`, `--slot-list`)

Cluster Computing Course

2.1 Spack Basic

- Motivation
- automatic installation vs. manual installation
- How-To-Install, How-To-Setup
- Configuration of Folder in config.yaml
- Usage in working groups
- Spack commands:
  - `spack list`
  - `spack spec <package>`
  - `spack install <package>`

Cluster Computing Course

2.2 Introducing Complexity

- Spack commands:
  - `spack list <package>`
  - `spack info <package>`
  - `spack extensions <package>`
  - `spack find [options] <package>`
  - `spack compiler info <package>`
- Reference Box of Spack commands
- Compiler Configurations
- Spack commands:
  - `spack compilers`
  - `spack compiler list`
- Introducion Yaml-Files: compilers.yaml
- How-To-Install with more specification
- How-To-Uninstall
- Novel concretization process
- Introducion Yaml-Files: spec.yaml

Cluster Computing Course

2.3 Advanced Spack Usage

- Introducion Yaml-Files: packages.yaml
- How-To-Installation-Script
- Spack command: `spack location --install-dir`
- How-To-Creating-Own-Spack-Package
- Spack commands:
  - `spack create <my-package>`
  - `spack edit <my-package>`
  - `spack install <my-package>`
- Filesystem Views
- Introducion Yaml-Files: projections.yaml
- Spack options: `--exclude`, `--dependencies`
- Spack command: `spack activate`
- Spack Summary

Cluster Computing Course

2.3 Advanced Spack Usage

- Core Spack Concepts
  - Software Complexity Handling
  - Restricting versions of dependencies
  - Virtual Dependency Handling
  - Novel concretization process (more details)
- Package Manager
  - How-To-Package-Manager
  - Low Level Build System: Make
  - High Level Build System: CMake, Autotools
  - Visualization of How-To-Package-Manager

UNIX Introduction

- Operating System: UNIX, kernel, shell, hardware, utility programs, application programs, system calls, library routines, processes, files
- File System Basic: Directory structure
- Terminal usage: `copy`, `paste`, filename completion
- Getting Help: `man`
- Wildcards
- Paths (Absolute vs. relative paths)
- Elementary Commands and its options: `ls`, `mkdir`, `pwd`, `cd`, `cp`, `mv` , `rm`, `rmdir`
- Redirections
- Displaying content: `echo`, `cat`, `clear`, `head`, `tail`, `more`, `less`, `grep`, `wc`
- Editing content: Introduction into `vi`, `emacs`, `nano`, `mc`

UNIX Introduction

- File Access Permissions: Introduction, `chmod`
- Process Management: `ps`, `top`, `bg`, `fg`, `jobs`, `kill`, several `crtl`-Options
- File System Commands: `file`, `find`, `apropos`, `touch`, `whereis`, `which`, `du`, `df`
- System Information: `printenv`, `history`, `date`, `reset`, `uname`, `tar`
- Password Authentication: `ssh`, `ssh-keygen`
- Authentication using Public Key
- User Configuration (client side vs. server side)
- Data transfer: `scp`, `rsync`

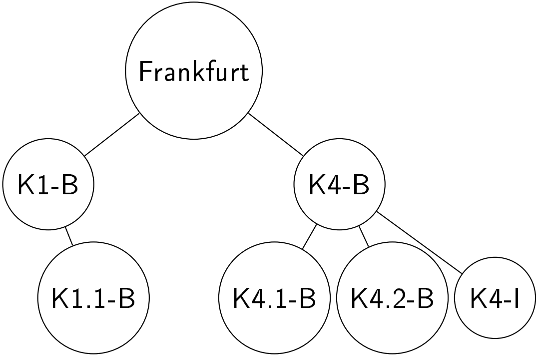
An Introduction to Shell Scripting

- Motivation: Why & What is Shell Scripting?
- Shell: `sh`, `csh`, `ksh`, `bash`
- Example: Writing Bash Scripts
- Command Line, Exit Status and Get Input
- Special files: `/etc/profile`, `/.bash_profile`, `/.bash_login`, `/.profile`, `/.bashrc`
- Filename metacharacters
- Quoting, I/O Redirection, Pipe
- Variable Assignment and Substitution
- Variables: $HOME, $HOSTNAME, $PATH, $PWD, $OLDPWD, $0, $n, $*, $@, $#, $$, $, $?
- History: `history`, line-edit mode, `fc`, `C`-shell-style history
- Control structure: `if`, `for`, `while`, `test`
- Exercises

Introduction to Version Control with GIT

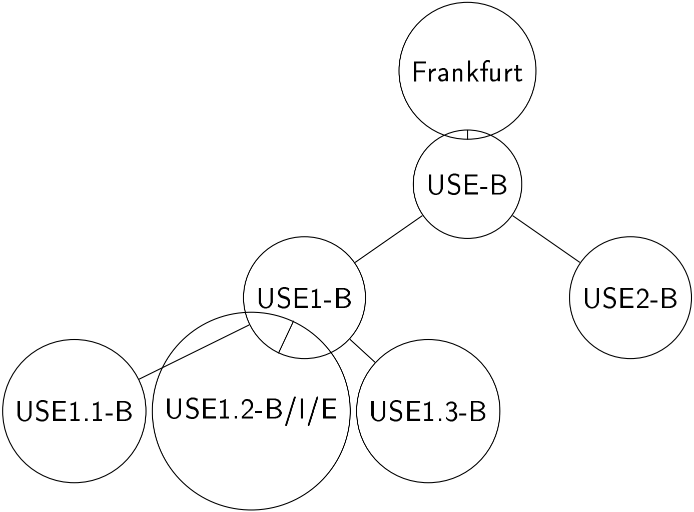
- Explanation of Key Concepts (Snapshots, Commits, Repositories)
- Requirements (`install`, `config`, create repository)
- Basic Commands (`status`, `add`, `commit`, `clone`, `push`, `pull`, `remote`)
- Branching (`create`, `rename`, `delete`, `switch`, `update merge` branches)
- Presentation of different or typical workflow concepts for branching
- Dealing with Merge Conflicts
- Cool Ad-Ons (`stash`, `log`, `gitignore`, `worktree`)

Skill Tree [1]



[1] K1-B: Supercomputers K4-B: Job Scheduling

Skill Tree [2]



[2] USE-B: Use of HPC Environment USE1-B: Cluster Operating System USE2-B: Running of Parallel Program

Start of HPCCF locally

- create concept for lecture *Tools in High Performance Computing* in September/October 2016
  - summer school for everybody
  - each topic, 2 h, 3-5 Credit Points
  - include the module into curriculum
  - understand coding as a language
- Lev Lafayette, australian researcher, visit Germany in October 2016 and attend my Cluster Computing Course
  - get good feedback and start a great collaboration
  - talk on eResearchAustralasia Conference in Australia in October 2017
  - introduced him to Julian Kunkel in October 2017 to start collaboration at HPCCF

Tools in High Performance Computing

1. Unix Basics
2. Software Tools Version Control Git, Subversion, Mercurial
3. Software Tools Emacs, VIM
4. Software Tools Regular Expression, AWK, SED
5. Software Tools Compiler and MAKE
6. Shell Scripting Basics
7. Shell Programming
8. Shell Programming
9. Cluster Usage
10. Batch Usage with SLURM
11. Introduction to HPC
12. Parallel Programming Concepts MPI, OpenMP
13. Debugging Tools Totalview, GDB
14. Profiling TAU, LIKWID
15. HPC Course Summary

The Importance of High Performance/Throughput Computing

- High Performance/Throughput Computing is of increasing importance.
- Distinction between HPC and HTC exists because performance does not always correlate with throughput due to opportunity costs.
- Whilst physical provision must come first, user education is also a necessary component.
- Evidence shows that the provision of training material has a significant effect on HPC usage.

The Situation of HP/TC Researchers

- Most researchers do not have formal teaching in HTC skills prior to need (Linux command line, HPC job submission).
- Only a handful of education institutions include HTC systems utilisation or parallel programming in the science curriculum.
- Fortunately most researchers are competent learners and can pick up new subject-matter quickly if delivered appropriately.
- A day's training is sufficient to introduce researchers to the concepts and practise of command-line Linux and job submission.
- Another day for shell scripting for HTC job submission.
- A day for the core concepts of parallel programming (multithreaded and message passing) and so forth.
- This follows the proposals of the "software carpentry" response to the skills-gap in scientific computing.

Interface Improvements or Skill Improvements

- The main methods for improving eResearch computational ability consist of developing the skillset among users to use the existing tools, or modifying the existing tools to fit the existing skillset.
- The intrinsic level of complexity in the environment and the requirement a grounded understanding of the process limits the capacity for automation and simplification.
- Without comprehension the eResearcher will be caught in an application relearning cycle.
- Grounding requires incorporating the core insights of adult and advanced education, including andragogical education.

The Continuum to Advanced Adult Education

- Adult learner has different characteristics to the child learner:
  1. autonomy of direction in learning
  2. importance of the use of personal experience as a learning resource
  3. emphasis on intrinsic rather than extrinsic motivations.

- These differences should be considered as a continuum with graduated equilibrium.
- In the contemporary environment this is supplemented with the notion of lifelong learning.

The Continuum to Advanced Adult Education

- Content needs to organised in terms of objectives, timed, and revised.
- Content needs to provided in as modular 'structural knowledge'.
- Provide grounding to a concept; facts and reasons provides understanding which allows elaboration by the learner.
- Delivery should make use of discipline-based learning styles.

Adult Education Stages and Institutional Opportunities

- An important insight from several years of using andragogical techniques with advanced computer training is the recognition that adult learner components (i.e., autonomy, personal experience, intrinsic motivation) varies significantly within the general status of advanced adult learner.
- At least part of this can be attributed to age and cultural diversity. Disciplinary diversity is increasingly challenging as researchers may be more familiar with different learning styles.

Lessons Learned and Future Initiatives

- The independent variation in components suggests that a review of researcher's needs prior to attending classes and bespoke content will have the best possible outcome.
- Conducting highly granular course content can contribute significantly in this process.
- Need to expand feedback including learners as partners in learning design, teaching governance, support (peer mentoring), evaluation and evidence, learning environment.

Course Sessions

ACRONYM	Course Title	Slides	Time
own courses			
UNIX	Introduction to UNIX	70	3 h
SHELL	Introduction to Shell Scripting	52	2 h
CLUSTER	Cluster Computing Course	165	4 h
TOOLS	Software Tools for UNIX Systems	208	5 h
GIT	Introduction to GIT	35	1 $^1/_2$ h
courses of HKHLR-colleagues			
PYTHON	Introduction to Python		3 h
PYTHON-H	Beginners Hands-On Python Course		12 h
PYTHON-A	Creating of programming projects with Python		12 h
TOTALVIEW	Introduction to TotalView Debugger		5 h

Course Sessions

ACRONYM	Course Title	Speaker	Days/x
courses with invited speakers			
MPI	Parallelization with MPI and OpenMP	Rabenseifner	3/2
CPP	Introduction to Cpp	Sitzmann	2/3
MATLAB	Cluster Course for Matlab Users	Martynenko	1/1
HPC	Spack: Managing HPC Software Complexity	Becker	1/1
	Clacc: OpenACC Support for Clang/LLVM	Denny	
	TAU: Performance Evaluation	Shende	
NLPE	Node-Level Performance Engineering	Eitzinger Gruber	2/1

Thank you for your attention!

Anja Gerbes
gerbes@csc.uni-frankfurt.de