Kleines Tutorial für die Nutzung des bwUniClusters

Sven Wehner
Angela Cho

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Programm

UNI FREIBURG

- Hintergrund bwUniCluster & Zugang
- Dateiensystem
- Skripte starten, Jobs ausführen, Logging
- Shell-Skript
- Hilfreiche Shortcuts & Tipps
- Häufige Fehler und Links

Konvention

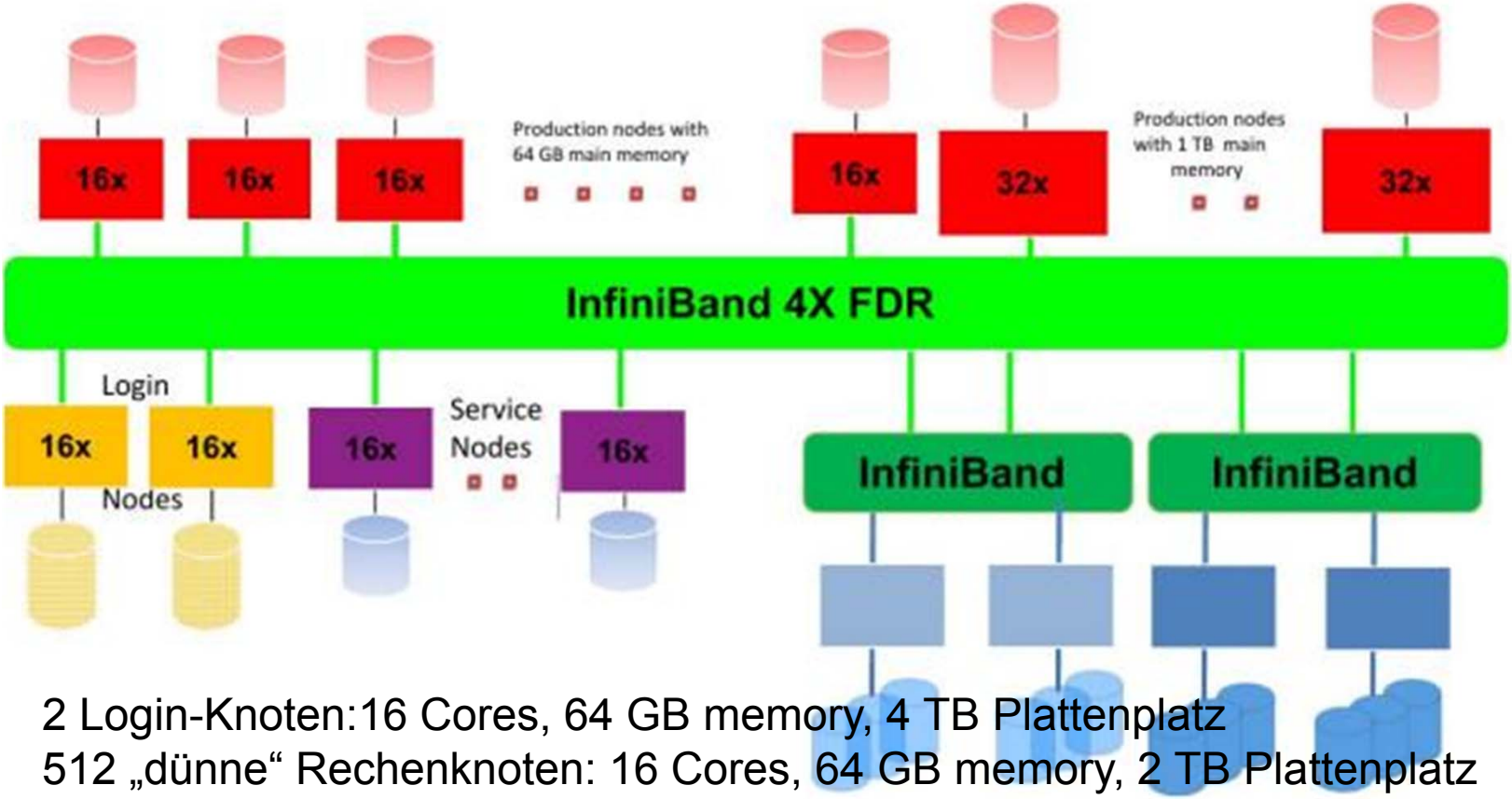
- Befehle: **command**
  - <benötigte Argumente>
  - [Optionen]

Hintergrund bwHPC-C5 Projekt

UNI FREIBURG

- Gestartet am 01.07.2013, Laufzeit 30 Monate
- Getragen durch:
  - Uni Freiburg, Heidelberg, Hohenheim, Konstanz, Mannheim, Stuttgart, Tübingen und Ulm
  - Karlsruher Institut für Technologie
  - Hochschule Esslingen und Stuttgart
- Für Universitäten: bwUniCluster am SCC in Karlsruhe seit Ende Januar 2014 online, unterstützt vom Land Baden-Württemberg und der DFG

Architektur



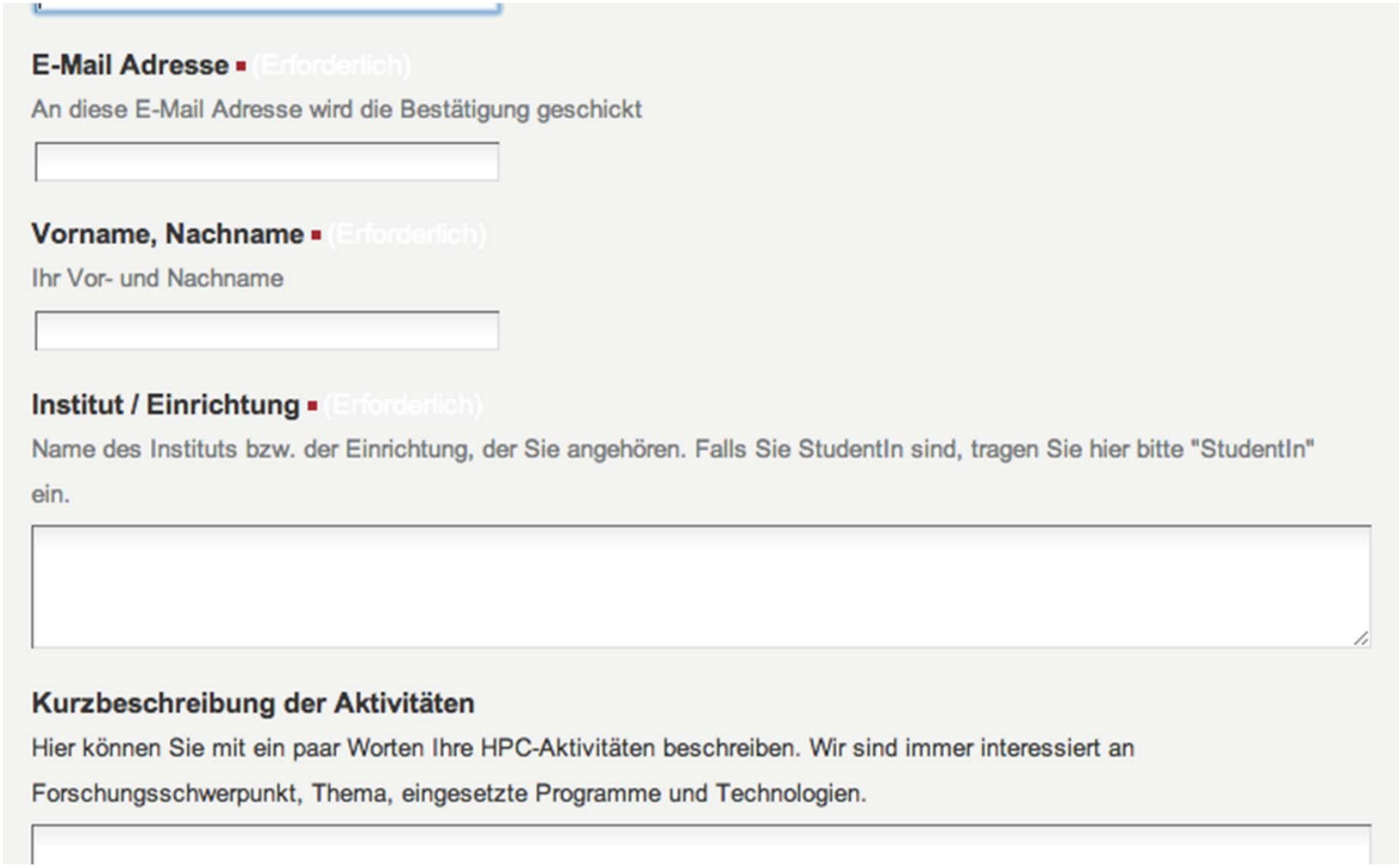
- 2 Login-Knoten:16 Cores, 64 GB memory, 4 TB Plattenplatz
- 512 „dünne“ Rechenknoten: 16 Cores, 64 GB memory, 2 TB Plattenplatz
- 8 „fette“ Rechenknoten: 32 Cores, 1 TB memory, 7 TB Plattenplatz
- 64 bit

Zugang

UNI FREIBURG

- Jeder mit einem RZ-Account der Uni Freiburg (auch Studierende) können kostenlos einen Zugang beantragen
- An der Uni Freiburg sich für den Cluster Dienst registrieren unter https://www.bwhpc-c5.uni-freiburg.de/bwunicluster/freischaltung-uni-account

Zugang



E-Mail Adresse ▪ (Erforderlich)

An diese E-Mail Adresse wird die Bestätigung geschickt

Vorname, Nachname ▪ (Erforderlich)

Ihr Vor- und Nachname

Institut / Einrichtung ▪ (Erforderlich)

Name des Instituts bzw. der Einrichtung, der Sie angehören. Falls Sie StudentIn sind, tragen Sie hier bitte "StudentIn" ein.

Kurzbeschreibung der Aktivitäten

Hier können Sie mit ein paar Worten Ihre HPC-Aktivitäten beschreiben. Wir sind immer interessiert an Forschungsschwerpunkt, Thema, eingesetzte Programme und Technologien.

Zugang

- Sobald man eine Bestätigungs-Email erhalten hat, muss man sich beim SCC am KIT für den bwUniCluster registrieren unter https://bwidm.scc.kit.edu/ und ein Passwort festlegen.
- Benutzername für Freiburger:

  fr_[RZ-Kürzel]@bwunicluster.scc.kit.edu

Zugang

UNI FREIBURG

- Zugriff auf den Cluster geht mit einem SSH Client
  - Windows: z.B. SSHSecureShellClient oder PuTTY
  - Mac und Linux: Vorhandenes Terminal
    - **ssh [Benutzer-Name]@bwunicluster.scc.kit.edu**
- Datei-Management auf dem Home-Verzeichnis */home/fr/fr_fr/fr_[RZ-Kürzel]* mit einem FTP Client
  - Windows: z.B. SSHSecureShellClient oder WinSCP
  - Mac und Linux: Vorhandenes Datei-Management System oder Programme wie z.B. Fetch

Dateisystem (1)

- Verzeichnisse wechseln (change directory): **cd <Pfad>**
  - Absolute Pfade: **cd /usr/include**
  - Relative Pfade: **cd data** bzw. **cd ./data**
  - In übergeordnete Verzeichnisse: **cd ../..**
- Linux ist „case sensitive“
- Verzeichnisse werden mit „/“ getrennt
  - „\“ Escape-Zeichen
- „Root“-Verzeichnis „/“
- „Home“-Verzeichnis „~/“

Dateisystem (2)

- Aktuelles Verzeichnis anzeigen: **pwd**
- Verzeichnis-Inhalt auflisten: **ls [Pfad]**
  - Alias: **ll [Pfad]** für **ls -l [Pfad]**
- Ordner erstellen: **mkdir <Ordner>**
- Dateien/Ordner kopieren (**cp**) bzw. verschieben (**mv**)
  - Kopieren: **cp [Optionen] <Quelle> <Ziel>**
  - Verschieben: **mv [Optionen] <Quelle> <Ziel>**
  - „**-R**“ für Verzeichnisse (rekursiv)
  - Überschreibt ohne Nachfrage („**-i**“ Option)
- Dateien/Ordner löschen
  - **rm <Datei>**
  - **rm -R <Ordner>**
  - Löscht ohne Nachfrage („**-i**“ Option)

Hilfe

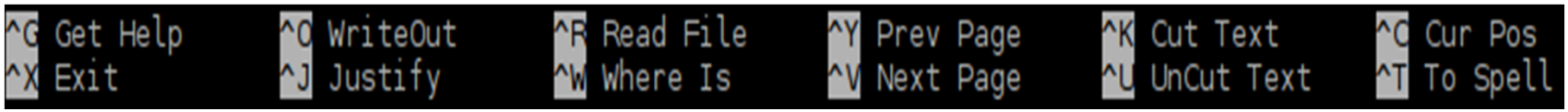
- **man** (wie Manual) zeigt Hilfstexte an
  - **man <Befehl>**
- Alternative 1: **info**
  - **info <Befehl>**
- Alternative 2: „**--help**“-Option
  - **<Befehl> --help**
  - (manchmal „**<Befehl> -h**“)

Eigene Skripte starten

- Programm/Skript starten:
  - Interpreter aufrufen
    - **bash <Datei>**
    - **python <Datei>**
    - …
  - Direkt aufrufen: **<Programm-Pfad>**
    - Z. B.: „**/bin/ls**“, „**./script.sh**“, „**~/app.py**“
    - Berechtigungen müssen ausreichend sein!
    - „eXecute“-Berechtigung hinzufügen mit **chmod**
      - **chmod +x ./script.sh**
    - Datei-Endungen haben keine Bedeutung

Text-Editor

- **nano <Datei>**: Einfacher Editor
  - Kommandos mit Strg



- **vim**: Vielseitiger Editor
  - Schwierig am Anfang
  - Später sehr mächtig
  - Tutorials:
    - http://www.openvim.com/
    - http://linuxconfig.org/vim-tutorial

Cluster-System: MOAB

- Job starten
  - Auf dem Login-Node
  - Job in Queue einfügen
- Job ausführen
  - Auf einem Cluster-Node
  - Initialisierung des eigentlichen Programms
  - Ausführung des Programms

- Skripte für beides erhältlich unter:
  - https://github.com/SvenWe/bwunicluster

Submit Skript – Jobs starten

```
1  #!/bin/bash
2  msub "run_job.sh" \
3       -l nodes=1:ppn=2,walltime=1:00:00:00 \
4       -q "singlenode" \
5       -m "ea" \
6       -v "SEED=42"
```

msub-Optionen

- Queue-Klasse: **-q <Queue>**
  - Definiert maximal-verfügbare Ressourcen
  - Queue-Klassen:
    - **develop**: Laufzeit 30 min, 1 Node, 16 Prozessoren
    - **singlenode**: Laufzeit 3 Tage, 1 Node, 16 Prozessoren
    - **multinode**: Laufzeit 2 Tage, 8 Nodes
    - **verylong**: Laufzeit 6 Tage, 1 Node, 16 Prozessoren
    - **fat**: Laufzeit 1 Tag, 1 Fat-Node, 32 Prozessoren
- Ressourcen: **-l**
  - „**-l nodes=1:ppn=8**“: Anzahl der Nodes und der Prozessoren pro Node
    - Mehrere Nodes machen das Programm nicht per se schneller, MPI o.ä. notwendig
  - „**-l walltime=2:00:00:00**“: Maximale Laufzeit (Format: DD:HH:MM:SS)
  - „**-l mem=1000mb**“ Maximaler Arbeitsspeicher (Einheiten: kb, mb, gb)
- Variablen definieren: **-v <Variable1>=<Wert>,<Variable2>=<Wert>**
- Benachrichtigungen via E-Mail: **-m** und **-M**
  - Wann: „**b**“ (begin), „**e**“ (end), „**a**“ (abort)
  - Beispiel: **msub […] -m ea -M person@psychologie.uni-freiburg.de […]**

MOAB Software Befehle

- **msub** *Shellscript*
  - **l** *resources; z.B. nodes=2:ppn=8, walltime=01:00:00, pmem=1000mb*
  - **m** *message z.B. bae*
  - **N** *name*
  - **o** *filename for output*
  - **q** *queue; z.B. fat, singlenode, multinode, verylong*
  - **v** *variable=arg*
- **checkjob** *Job-ID*
- **showq**
- **canceljob** *Job-ID*

Jobs ausführen

- Auf einem Cluster-Node wird ein Programm/Skript gestartet
- Empfehlung: Run-Skript verwenden
  - Module laden
  - Programm-Start vorbereiten
  - Programm starten

Run-Skript – Jobs ausführen


```
1  #!/bin/bash
2
3  # Module laden
4  module load devel/python/3.3.3
5
6  # Programm starten
7  ./program
8  python program.py
```

Weitere MOAB-Befehle

- Jobs anzeigen: **showq**
  - Listet eigene Jobs auf (aktiv, wartend, geblockt)
- Cluster-Belegung anzeigen: **showstate**
  - Listet die Jobs aller Teilnehmer auf allen Nodes auf
- Informationen über einen Job: **checkjob <Job-ID>**
  - Gibt Gründe, warum Job nicht gestartet wurde
- Job vorzeitig beenden: **canceljob <Job-ID>**
  - Alle Jobs abbrechen: **canceljob r:.***
    - Regulärer Ausdruck: „**r:**“; Muster: „**.***“
  - Alternativ: **mjobctl -c -w user=$(whoami)**

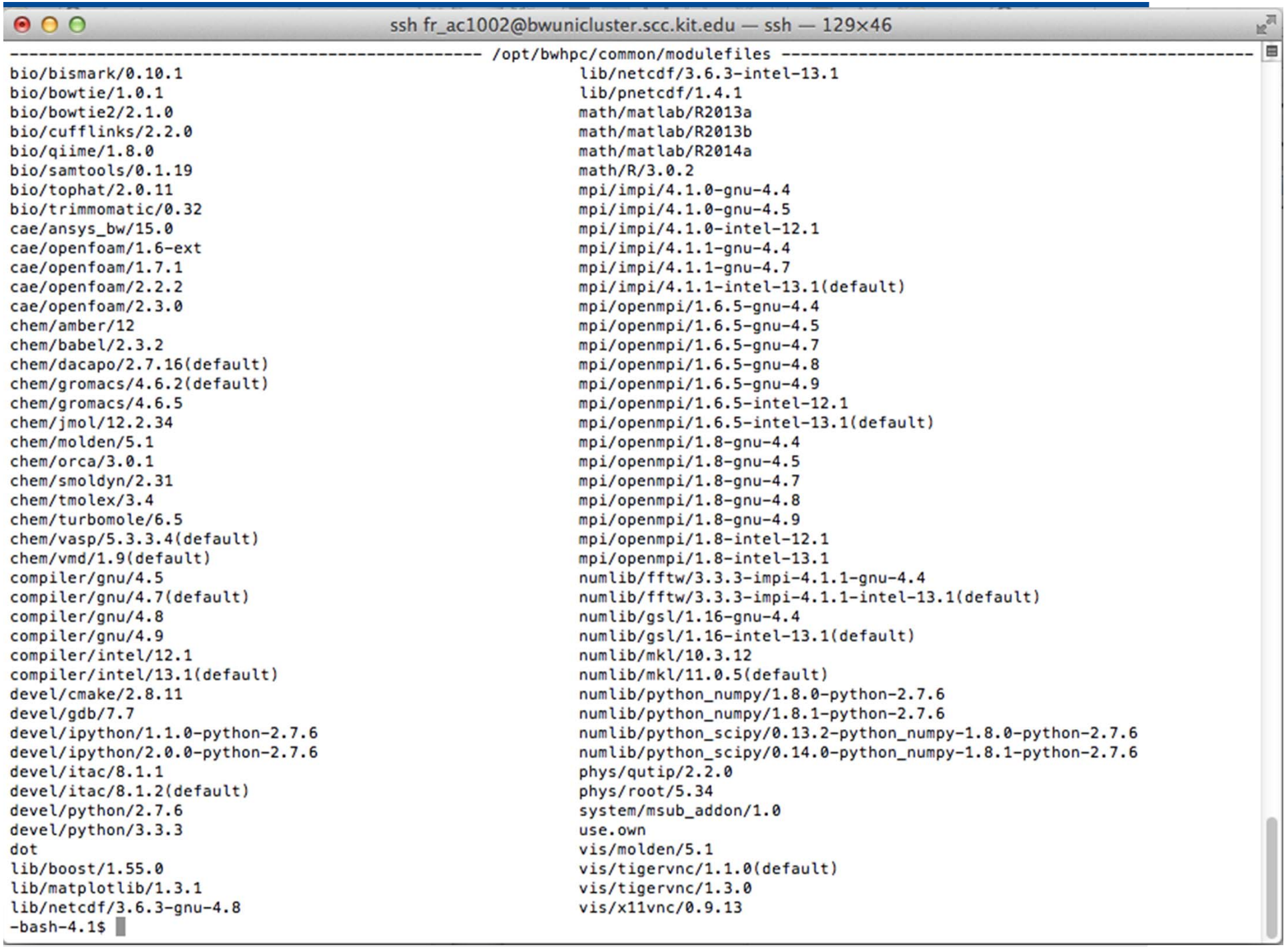
Dateisystem & Logging

- Login-Node und Cluster-Nodes teilen sich Dateisystem
  - über Netzwerk eingebunden
  - kein Umkopieren nötig
  - relativ langsamer Zugriff
  - Achtung: gegenseitiges Überschreiben!
- Ausgabe des Run-Scripts wird in Datei gespeichert
  - Alles was auf Konsole angezeigt würde, wird in Datei geschrieben
  - Ausgabe-Datei: *job_[Job-ID].out*

Environment Modules

UNI FREIBURG

- Viele Programme bereits als *modulefiles* vorhanden, müssen nicht installiert werden!
- Wird auf die einzelnen Rechennodes geladen mit dem Befehl
  - **module load [Kategorie]/[Name]/[Version]**
  - R: **module load math/R/3.0.2**
  - Fortran oder C++: **module load compiler/gnu/4.7**
  - Python: **module load devel/python/3.3.3**
  - Liste aller verfügbaren modulefiles mit **module avail**


ssh fr_ac1002@bwunicluster.scc.kit.edu — ssh — 129×46
/opt/bwhpc/common/modulefiles
bio/bismark/0.10.1
bio/bowtie/1.0.1
bio/bowtie2/2.1.0
bio/cufflinks/2.2.0
bio/qiime/1.8.0
bio/samtools/0.1.19
bio/tophat/2.0.11
bio/trimmomatic/0.32
cae/ansys_bw/15.0
cae/openfoam/1.6-ext
cae/openfoam/1.7.1
cae/openfoam/2.2.2
cae/openfoam/2.3.0
chem/amber/12
chem/babel/2.3.2
chem/dacapo/2.7.16(default)
chem/gromacs/4.6.2(default)
chem/gromacs/4.6.5
chem/jmol/12.2.34
chem/molden/5.1
chem/orca/3.0.1
chem/smoldyn/2.31
chem/tmolex/3.4
chem/turbomole/6.5
chem/vasp/5.3.3.4(default)
chem/vmd/1.9(default)
compiler/gnu/4.5
compiler/gnu/4.7(default)
compiler/gnu/4.8
compiler/gnu/4.9
compiler/intel/12.1
compiler/intel/13.1(default)
devel/cmake/2.8.11
devel/gdb/7.7
devel/ipython/1.1.0-python-2.7.6
devel/ipython/2.0.0-python-2.7.6
devel/itac/8.1.1
devel/itac/8.1.2(default)
devel/python/2.7.6
devel/python/3.3.3
dot
lib/boost/1.55.0
lib/matplotlib/1.3.1
lib/netcdf/3.6.3-gnu-4.8
lib/netcdf/3.6.3-intel-13.1
lib/pnetcdf/1.4.1
math/matlab/R2013a
math/matlab/R2013b
math/matlab/R2014a
math/R/3.0.2
mpi/impi/4.1.0-gnu-4.4
mpi/impi/4.1.0-gnu-4.5
mpi/impi/4.1.0-intel-12.1
mpi/impi/4.1.1-gnu-4.4
mpi/impi/4.1.1-gnu-4.7
mpi/impi/4.1.1-intel-13.1(default)
mpi/openmpi/1.6.5-gnu-4.4
mpi/openmpi/1.6.5-gnu-4.5
mpi/openmpi/1.6.5-gnu-4.7
mpi/openmpi/1.6.5-gnu-4.8
mpi/openmpi/1.6.5-gnu-4.9
mpi/openmpi/1.6.5-intel-12.1
mpi/openmpi/1.6.5-intel-13.1(default)
mpi/openmpi/1.8-gnu-4.4
mpi/openmpi/1.8-gnu-4.5
mpi/openmpi/1.8-gnu-4.7
mpi/openmpi/1.8-gnu-4.8
mpi/openmpi/1.8-gnu-4.9
mpi/openmpi/1.8-intel-12.1
mpi/openmpi/1.8-intel-13.1
numlib/fftw/3.3.3-impi-4.1.1-gnu-4.4
numlib/fftw/3.3.3-impi-4.1.1-intel-13.1(default)
numlib/gsl/1.16-gnu-4.4
numlib/gsl/1.16-intel-13.1(default)
numlib/mkl/10.3.12
numlib/mkl/11.0.5(default)
numlib/python_numpy/1.8.0-python-2.7.6
numlib/python_numpy/1.8.1-python-2.7.6
numlib/python_scipy/0.13.2-python_numpy-1.8.0-python-2.7.6
numlib/python_scipy/0.14.0-python_numpy-1.8.1-python-2.7.6
phys/qutip/2.2.0
phys/root/5.34
system/msub_addon/1.0
use.own
vis/molden/5.1
vis/tigervnc/1.1.0(default)
vis/tigervnc/1.3.0
vis/x11vnc/0.9.13
-bash-4.1$

Shell-Skript

- Was ist ein Shell-Skript?
- Erste Zeile „Shebang“
  - Definiert wie Skript auszuführen ist
  - **#!<Interpreter>**
    - **#!/bin/bash**
    - **#!/usr/bin/env python**
- Kommentare: #
  - bis zum Ende der Zeile

Shell-Skript: Variablen

- Variablen anlegen:
  - **<Variablenname>=<Wert>**
  - Keine Leerzeichen!
- Variablen benutzen:
  - **${<Variablenname>}**

```
1  #!/bin/bash
2  text="Executing job ${MOAB_JOBID}!"
3  echo ${text}
```

Umgebungsvariablen

- Job-ID: MOAB_JOBID
- Job-Name: MOAB_JOBNAME
- Anzahl der Nodes: MOAB_NODECOUNT
- Anzahl der Prozessoren: MOAB_PROCCOUNT
- Verzeichnis: MOAB_SUBMITDIR
- Benutzer-Name: MOAB_USER

MOAB Software Variablen

UNI FREIBURG

- MOAB_JOBID
- MOAB_NODECOUNT *oder*
  SLURM_JOB_NUM_NODES
- MOAB_PROCCOUNT *oder*
  SLURM_JOB_CPUS_PER_NODE
- MOAB_USER

- SLURM_JOB_CPUS_PER_NODE
- SLURM_JOB_NODELIST
- SLURM_MEM_PER_NODE
- SLURM_NPROCS

Pipes

- Ausgabe von Befehlen in andere Befehle füttern: „|“
  - z. B. **cat longtext.txt | grep Error | sort | less**
  - **grep <Suchtext>**: Suche nach Text in Eingabe
    - (oder Dateien: **grep <Suchtext> <Datei>**)
    - (oder Ordnern: **grep -R <Suchtext> <Ordner>**)
  - **sort**: Sortiert in der Ausgabe die Zeilen
  - **less**: Ausgabe zwischenspeichern → Anzeigen: scrollbar, durchsuchbar etc.
- Dateien als Eingabe (**<**) bzw. Ausgabe (**>**)
  - **cat < readme.txt**
  - **grep -R foo . > searchresult.txt**
- Ausgabe von Befehlen direkt verwenden
  - **variable="$(hostname)"**
  - **echo "Du bist als $(whoami) eingeloggt."**

Suchen

- Datei nach Namen suchen: **find**
  - **find <Pfad> -iname *<Suchwort>***
- Dateien nach Inhalt durchsuchen: **grep**
  - **grep -R <Suchwort> <Verzeichnis>**

Wichtige Shortcuts

- Strg+C: Cancel
- Strg+D: End of Input
- Strg+Z: aktuellen Befehl schlafen legen
  - Mit **fg** („foreground“) bzw. **bg** („background“) reaktivieren
- Tabulator: oft Auto-Vervollständigung
  - 2x Tabulator (in bash): Möglichkeiten anzeigen
- Eingabe-Verlauf durchwandern: Pfeiltasten (↑, ↓)
- Strg+R: Suche in Eingabe-Verlauf

Hilfreiche Tricks (1)

- Terminalmultiplexer: **screen**
  - Hält die Sitzungen offen
    - Abbruch der Internet-Verbindung o.ä.
  - Erlaubt mehrere Sitzungen parallel zu verwalten
  - Ctrl+A, dann
    - C: „create“
    - N: „next“; P: „previous“
  - Zu alter Sitzung verbinden („reattach“): **screen -r**
  - Sitzungen anzeigen („list“): **screen -ls**

Hilfreiche Tricks (2)

- ~/bin/
  - Ordner ist in PATH
  - Programme/Skripte in diesem Ordner können direkt ausgeführt werden
    - An Ausführ-Rechte denken

Beispiel für R

UNI FREIBURG

- R im login-Node als modulefile laden und interaktiv starten
- Benötigte packages installieren

Beispiel für R

```
-bash-4.1$ Last login: Wed May 14 16:13:35 2014 from sopc52.psychologie.uni-freiburg.de
*******************************************************************************
*                                                                             *
*         Universal HPC cluster of Baden-Wuerttemberg's universities:         *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
*                                                                             *
*                      (KITE 2.0/RHEL6.5/Lustre 2.5.1)                        *
*                                                                             *
*       https://www.bwhpc-c5.de/wiki/index.php/bwUniCluster_User_Guide        *
*                email: bwunicluster-hotline@lists.kit.edu                    *
*                                                                             *
*******************************************************************************
*                                                                             *
*                     LOCAL bwUniCluster INFORMATION                          *
* General infos and wiki  : http://www.bwhpc-c5.de/wiki/index.php/Main_Page   *
* Local web site and help : http://www.bwhpc-c5.uni-freiburg.de/bwunicluster  *
*                                                                             *
*******************************************************************************
-bash-4.1$ module load math/R
Loading module dependency 'compiler/intel/13.1'.
Loading module dependency 'numlib/mkl/11.0.5'.
-bash-4.1$ R

R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-unknown-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

>
```

Beispiel für R

- R submit

```
-bash-4.1$ vi anstehen.moab
#!/bin/bash
script_path="gapper.moab"
res_nodes="nodes=1:ppn=16"
res_time="walltime=00:30:00"

for j in {1..5..1}
        do
                job_variable="j=${j}"
                job_name="JOB_NUMBER_${j}"
                msub -v "${job_variable}" -N "${job_name}" -l "${res_nodes}","${res_time}" -m bae \
                "${script_path}"
done
```

Beispiel für R

- R run

```
-bash-4.1$ cat gapper.moab
#!/bin/bash
R_script_path="gapper.R"
R_output="gapper.R_${MOAB_JOBID}.log"
echo "Working Directory:                    $PWD"
echo "Running on host                       $HOSTNAME"
echo "Job id:                               $MOAB_JOBID"
echo "Job name:                             $MOAB_JOBNAME"
echo "Number of nodes allocated to job:     $MOAB_NODECOUNT"
echo "Number of cores allocated to job:     $MOAB_PROCCOUNT"
module load math/R/3.0.2
R CMD BATCH \
        "--args j=${j}" \
         "${R_script_path}" \
         "${R_output}"

-bash-4.1$
```

Beispiel für R

UNI FREIBURG

oder **bash anstehen.moab**

```
-bash-4.1$ msub anstehen.moab

uc1.343282
-bash-4.1$ showq

active jobs------------------------
JOBID              USERNAME      STATE PROCS   REMAINING            STARTTIME


0 active jobs            0 of 7888 processors in use by local jobs (0.00%)
                       477 of 486 nodes active      (98.15%)

eligible jobs----------------------
JOBID              USERNAME      STATE PROCS     WCLIMIT             QUEUETIME

uc1.343282         fr_ac100       Idle     1    00:10:00  Thu May 15 11:21:31

1 eligible job

blocked jobs-----------------------
JOBID              USERNAME      STATE PROCS     WCLIMIT             QUEUETIME


0 blocked jobs

Total job:  1

-bash-4.1$
```

Beispiel für R

UNI FREIBURG

```
-bash-4.1$ showq

active jobs------------------------
JOBID              USERNAME      STATE PROCS   REMAINING            STARTTIME


0 active jobs            0 of 7888 processors in use by local jobs (0.00%)
                       477 of 486 nodes active      (98.15%)

eligible jobs----------------------
JOBID              USERNAME      STATE PROCS     WCLIMIT            QUEUETIME

uc1.343284         fr_ac100       Idle    16    00:30:00  Thu May 15 11:21:44
uc1.343286         fr_ac100       Idle    16    00:30:00  Thu May 15 11:21:47
uc1.343287         fr_ac100       Idle    16    00:30:00  Thu May 15 11:21:47
uc1.343285         fr_ac100       Idle    16    00:30:00  Thu May 15 11:21:45
uc1.343288         fr_ac100       Idle    16    00:30:00  Thu May 15 11:21:47

5 eligible jobs

blocked jobs-----------------------
JOBID              USERNAME      STATE PROCS     WCLIMIT            QUEUETIME


0 blocked jobs

Total jobs:  5

-bash-4.1$
```

Beispiel für R

```
library(doParallel)
library(MBESS)
library(MASS)
library(BBmisc)

procs <- as.numeric(Sys.getenv("MOAB_PROCCOUNT"))
registerDoParallel(cores=procs)

args=(commandArgs(TRUE))
print(args)
if(length(args)==0){
  print("j not there")
}else{
  for(i in 1:length(args)){
    eval(parse(text=args[[i]]))
  }
}
```

- foreach(…) %dopar% {…}

Tipps für C++

- Microsoft Visual Studio C++?
  - Bitte STL benutzen!
- Build-System: **cmake**
  - **module load devel/cmake compiler/gnu**
  - „CMakeLists.txt“ erstellen
  - **mkdir build && cd build**
  - **cmake ../ && make**

Häufige Fehler

UNI FREIBURG

- Auf dem Login-Node rechnen statt auf den Slaves
- Software auf Login-Node installieren, statt von den module files runter zu laden
- Interaktive Jobs statt batch mode
- „Nonsense-Jobs“ laufen lassen, Clusterplatz vergeuden
- Mehrere nodes beanspruchen aber nicht nutzen

Links

- bwUniCluster-Wiki:
  - http://www.bwhpc-c5.de/wiki/
    - BwHPC Best Practices Repository
- Beispiele, Tipps und Tricks:
  - https://github.com/SvenWe/bwunicluster
- Linux-/Shell-Tutorials:
  - http://ryanstutorials.net/linuxtutorial/
  - http://linuxcommand.org/
- Generell:
  - http://stackoverflow.com/