

HPC Einführungsveranstaltung

06.07.2023



- **Einleitung**
 - HPC Support Team
 - HPC Dienste
 - Hochleistungsrechner: Definition und Arbeitsweise
- **Vorstellung der verfügbaren HPC-Systeme**
 - Lokale Installationen
 - Linux Cluster (caucluster)
 - NEC HPC-System (nesh)
 - NHR-Verbund
- **Arbeiten auf HPC-Systemen**
 - Accountbeantragung
 - Login und Dateisysteme
 - Software und Batchbetrieb
- **Vortragsfolien:**
 - <https://www.rz.uni-kiel.de/de/angebote/hiperf/hpc-kurse>



- **HPC Support Team**
 - Benutzerberatung
 - Dr. Karsten Balzer
 - Dr. Michael Kisiela
 - Dr. Simone Knief
 - Systemadministration
 - Dr. Holger Naundorf
- **Mailingliste/Ticketsystem**
 - hpcsupport@rz.uni-kiel.de



- **Bereitstellung von Rechenressourcen unterschiedlicher Leistungsklassen**
- **HPC Support**



- **Bereitstellung von Rechenressourcen unterschiedlicher Leistungsklassen**
 - **lokale Hochleistungsrechner**
 - **caucluster**: Linux-Clustersystem
 - **hybrides NEC HPC-System (nesh)**:
 - Linux-Clustersystem
 - NEC SX-Aurora TSUBASA
 - GPU-Komponente (NVIDIA Tesla V100 Grafikkarten)
 - **Rechnersysteme im NHR Verbund**

→ Ressourcen stehen allen CAU- und GEOMAR-Angehörigen zur Verfügung

- **Bereitstellung von Rechenressourcen**
- **HPC Support**
 - Unterstützung bei Nutzung lokaler und dezentraler HPC-Ressourcen
 - Auswahl der geeigneten Rechnerarchitektur
 - Hilfe bei der Portierung und Optimierung von Programmen
 - Auswahl geeigneter Compiler und deren Optionen
 - Optimierung von Programmcodes (Algorithmenauswahl)
 - Hilfe bei Parallelisierung und Vektorisierung von Programmen
 - Installation von Benutzerprogrammen auf unseren Systemen
 - Hilfestellung bei Anfertigung von Projektanträgen für HPC-Rechner im NHR-Verbund



Hochleistungsrechner



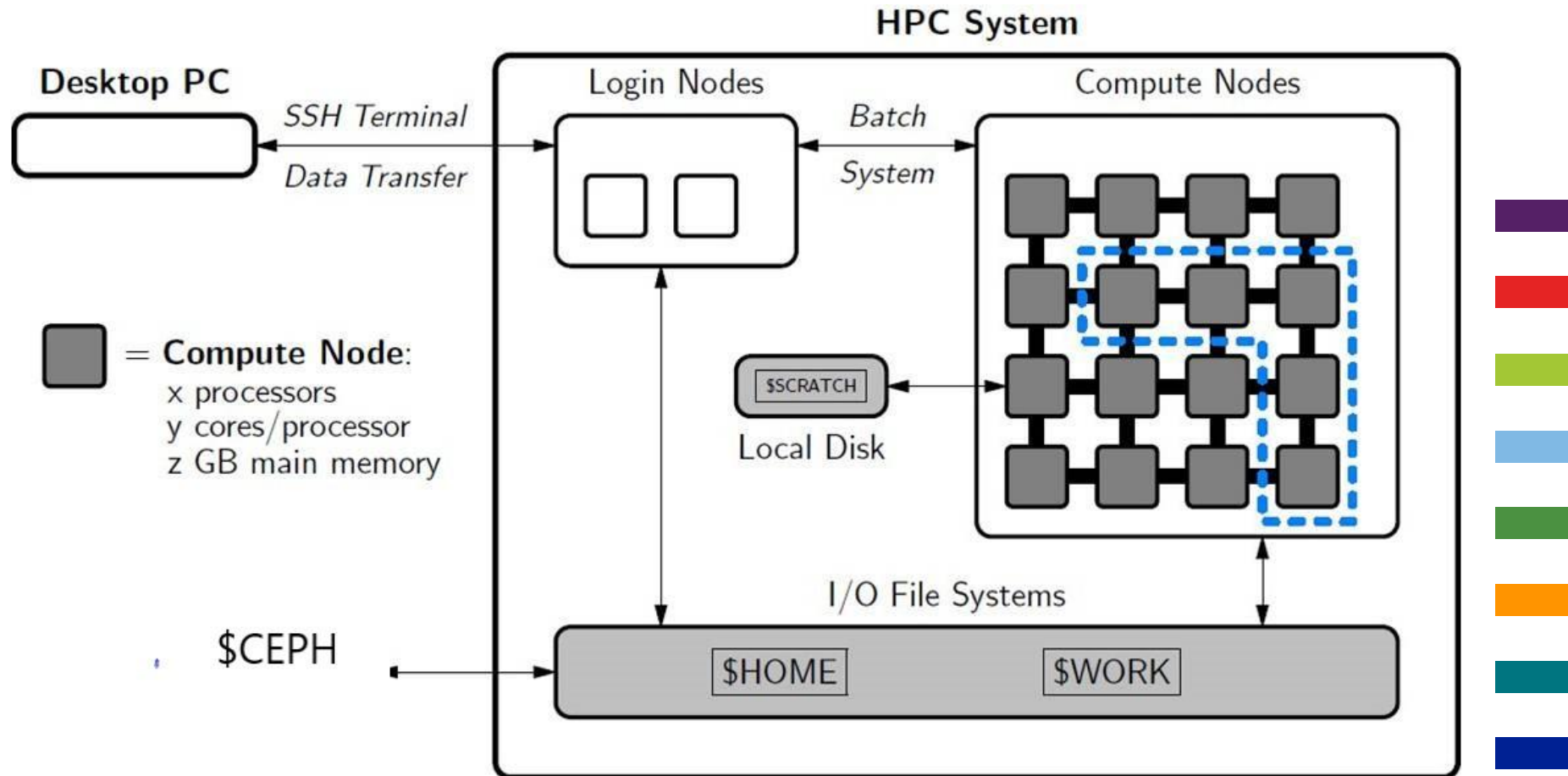
Hochleistungsrechner: Definition

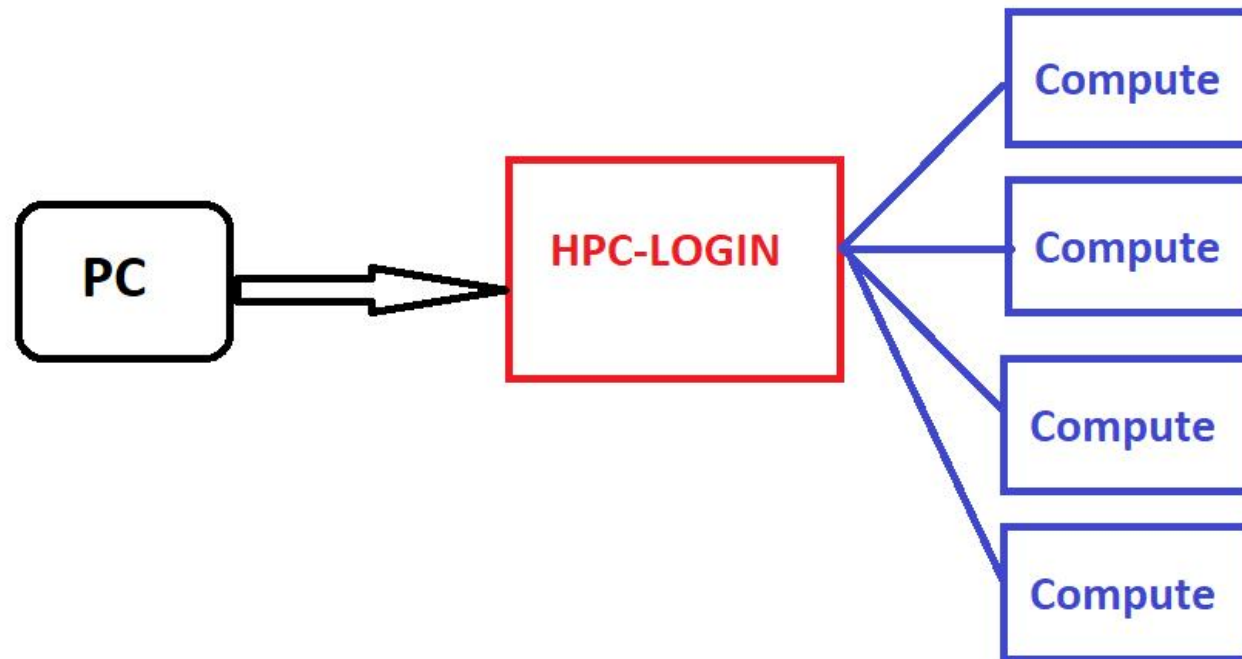
- keine eindeutige Formulierung möglich
- = computer aided computation
- Hochleistungsrechnen ist heute praktisch identisch mit parallelem Rechnen
- Architektur ist auf parallele Verarbeitung ausgerichtet
- aktuelle PCs mit Multi-Core Prozessoren (kleine Parallelrechner)
 - Leistungssteigerung ist jedoch begrenzt

Hochleistungsrechner: Definition

- keine eindeutige Formulierung möglich
- Hochleistungsrechnen ist heute praktisch identisch mit parallelem Rechnen
- Architektur ist auf parallele Verarbeitung ausgerichtet
- aktuelle PCs mit Multi-Core Prozessoren (kleine Parallelrechner)
 - Leistungssteigerung ist jedoch begrenzt
- **Hochleistungsrechner:**
 - hochparallele Rechencluster
 - große Anzahl von Rechenknoten ausgestattet mit Multi-Core CPUs
 - schnelles Verbindungsnetzwerk zwischen Rechenknoten
 - leistungsfähige I/O-Dateisysteme (für Lese- und Schreibprozesse)
 - Rechnersysteme mit einer hohen Hauptspeicherausstattung (>1TB)
 - Rechnersysteme mit spezieller Hardware/Architektur
 - z.B. Vektorrechner, Grafikkarten

HPC-Systeme (Aufbau allgemein)





Linux-Cluster (caucluster)



Linux Cluster (caucluster)

- **Hardware**
 - 2 Login-Knoten
 - **AMD Epyc** 73713 CPUs, 32 Cores und 512GB Memory
 - NVIDIA GPU: Quadro RTX 4000
 - 130 Batchknoten mit 4160 Cores (AMD Epyc CPUs)
 - jeweils 32 Cores, Hauptspeicher: 256GB bis 4TB
 - 16 Knoten 256GB RAM
 - 93 Knoten 512GB RAM
 - 15 Knoten 1TB RAM
 - 5 Knoten 2TB RAM
 - 1 Knoten 4TB RAM



- **Betriebssystem**
 - Rocky Linux 8.6
- **Dateisysteme**
 - HOME-Verzeichnis
 - 85TB für alle Nutzer (ZFS-Dateisystem)
 - WORK-Verzeichnis
 - 1,2PB BeeGFS paralleles Dateisystem
 - Ceph-Verzeichnis
- **Batchsystem**
 - SLURM Workload Manager
 - **S**imple **L**inux **U**tility for **R**esource **m**anagement)
 - Ressourcenverteilung über Fair-Share Algorithmus
 - Fair-Share \neq First In – First Out (FiFO)
 - Berücksichtigung von Core- und Hauptspeichernutzung





- **Hybrides HPC-System**
 - skalarer Linux Cluster mit Intel-CPU's
 - NEC SX-Aurora TSUBASA
 - GPU Subsystem
 - Einheitliche Systemumgebung
 - gemeinsame Vorrechner (*HPC-Login*)
 - globale (gemeinsame) Dateisysteme für \$HOME und \$WORK
 - ein gemeinsames Batchsystem: SLURM
 - Ressourcenverteilung über Fair-Share Algorithmus
- kein Datentransfer für die Nutzung unterschiedlicher Hardwarearchitekturen erforderlich



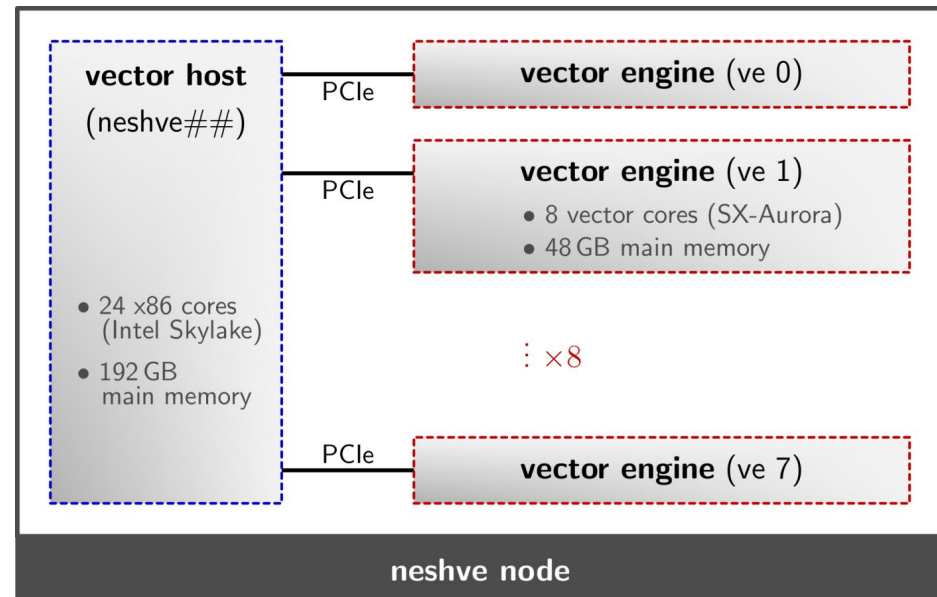
- 4 gemeinsame Vorrechner (*HPC-Login*)
 - Intel Skylake-SP Prozessoren (2,1GHz)
 - 32 Cores pro Knoten und 768GB Hauptspeicher
 - Login-Name: nesh-fe.rz.uni-kiel.de
- interaktives Arbeiten sowie Pre- und Postprocessing

- 464 Knoten mit insgesamt 14848 Cores (*Compute*)
 - 284 Knoten mit jeweils
 - 2 Intel Xeon Gold 6226R (Cascade Lake): 32 Cores pro Knoten und 192 GB Hauptspeicher bzw. 1.5TB (4 Knoten)
 - 180 Knoten mit jeweils
 - 2 Intel Xeon Gold 6130 (Skylake-SP): 32 Cores pro Knoten und 192 GB Hauptspeicher bzw. 384GB (8 Knoten)
- Batchknoten mit schnellem EDR-InfiniBand vernetzt
 - Übertragungsrate: 100Gbit/s
- 2 Data-Mover-Knoten
 - Intel Xeon Silver 4214 (Cascade Lake): 24 cores, 192GB RAM
 - Zugriff auf alle Dateisysteme (\$HOME,\$WORK,\$CEPH) und Internet
 - interaktiv oder über spezielle Batchpartition data
- Betriebssystem: Red Hat Enterprise Linux (Release 8.2)
- Batch-Knoten (*Compute*) nur über Batchsystem erreichbar, kein interaktives Einloggen

- **NEC SX-Aurora TSUBASA**

- 8 Vektorhosts (VH) mit jeweils:
 - 8 Vektor-Engines (VE, Vektor-Karten) mit jeweils 8 Vektor-Cores, 48 GB Speicher, 64 logische Vektor-Register pro Core (Länge 256 x 64 Bits)
 - VE-Anbindung an Vektorhost über PCIe
 - 1,2 TB/s Speicherbandbreite

→ 512 SX-Aurora Vektor-Cores



- 2 Knoten ausgestattet mit jeweils
 - GPU-Host
 - 2 Intel Xeon Gold 6226R (Cascade Lake): 32 Cores, 192GB RAM
 - 4 NVIDIA Tesla V100 GPU-Karten
 - 5120 NVIDIA CUDA Cores und 32GB GPU RAM
 - 900GB/s Speicherbandbreite

- **Betriebssystem**
 - Red Hat Enterprise Linux (Release 8.2)
- **Dateisysteme**
 - HOME-Verzeichnis
 - 300TB NEC GxFS
 - WORK-Verzeichnis
 - 10 PB NEC GxFS (2 Partitionen mit je 5 PB)
 - Zugriff auf ein Ceph-Dateisystem
- **Batchsystem**
 - SLURM Workload Manager
 - **S**imple **L**inux **U**tility for **R**esource **m**anagement
 - Ressourcenverteilung über Fair-Share Algorithmus

- **Betriebssystem**
 - Red Hat Enterprise Linux (Release 8.2)
- **Dateisysteme**
 - HOME-Verzeichnis
 - 300TB NEC GxFS
 - WORK-Verzeichnis
 - 10 PB NEC GxFS (2 Partitionen mit je 5 PB)
 - Zugriff auf ein Ceph-Dateisystem
- **Batchsystem**
 - SLURM Workload Manager
 - **S**imple **L**inux **U**tility for **R**esource **m**anagement
 - Ressourcenverteilung über Fair-Share Algorithmus

→ **Erweiterung/Erneuerung der Systemumgebung 3.Quartal 2023**

- Verbund für Nationales Hochleistungsrechnen
- besteht seit 2021
- Zusammenschluss 9 Hochschulrechenzentren in Deutschland
 - RWTH Aachen, TU Dresden, TU Darmstadt, Uni Erlangen, KIT, Uni Paderborn, NHR Süd-West (Uni Frankfurt/Main, TU Kaiserslautern, Uni Mainz, Uni des Saarlandes)
 - GWDG Göttingen (NHR@GWDG)
 - Zuse-Zentrum Berlin (NHR@ZIB)
- <https://www.nhr-verein.de>



■ Berlin (Lise)

■ Intel-CPU-Partition

- 1.270 Knoten (121.920 Cores)
 - 1.236 Knoten, 384GB RAM
 - 32 Knoten, 768GB RAM
 - 2 Knoten, 1,5TB RAM
 - 2 Intel Cascade Lake CPUs
 - 48 Cores/CPU

■ Nvidia-GPU-Partition

- 42 GPU-Knoten
 - 4 Nvidia Tesla A100 (80GB) Grafikkarten
 - 2 Intel Xeon 8360Y CPUs

■ Göttingen (Emmy)

■ Intel-CPU-Partition

- 974 Knoten (93.504 Cores)
 - 956 Knoten, 384GB RAM
 - 16 Knoten, 768 GB RAM
 - 2 Knoten, 1.5TB RAM
 - 2 Intel Cascade Lake CPUs
 - 48 Cores/CPU
- 448 Knoten (17.920 Cores)
 - 432 Knoten, 192GB RAM
 - 16 Knoten 768GB RAM
 - 2 Intel Skylake CPUs
 - 20 Cores pro CPU

■ Nvidia-GPU-Partition

- 33 GPU Knoten
 - 4 NVIDIA Tesla A100 (40GB) Grafikkarten
 - 2 AMD Epyc CPUs



- **Arbeiten auf unseren lokalen HPC-Systemen**
 - Accountbeantragung
 - Einloggen und Datentransfer
 - Dateisysteme und Datensicherung
 - Compiler und Software
 - Interaktives Arbeiten und Batchbetrieb



- Rechnerzugang kann jederzeit beantragt werden
- Formular 3: Antrag auf Nutzung eines Hochleistungsrechners
 - <https://www.rz.uni-kiel.de/de/ueber-uns/dokumente/formulare/anmeldung/form3.pdf>
 - ausgefüllt an die RZ-Benutzerverwaltung schicken
 - <https://www.rz.uni-kiel.de/de/ueber-uns/dokumente/formulare/anmeldung/rz-antraege-waehrend-corona>
- mit Accounteintragung: Aufnahme in 2 Mailinglisten
 - nesh_user@lists.uni-kiel.de bzw. caucluster_user@lists.uni-kiel.de
 - rechnerpezifische Informationen:
 - Betriebsunterberechnungen, Probleme, neue Software, ...
 - hpc_user@uni-kiel.de
 - allgemeine Informationen
 - Kurse, Infoveranstaltung, Rechnerbeschaffungen, ...

Rechenzentrum der Universität Kiel

Formular 3 / 2019-11

Hochleistungsrechner

(Formular darf selbst ausgedruckt und vervielfältigt werden; mit der Unterschrift wird immer das Original im RZ anerkannt!)

Antrag auf Nutzung eines Hochleistungsrechners

- ☐ **NEC Hochleistungsrechnersystem**
(NEC SX-Aurora TSUBASA und NEC HPC-Linux-Cluster)
- ☐ **Linux-Cluster** (caucluster)

1. Allgemeine Angaben (Bitte leserlich in Druckbuchstaben ausfüllen)

Einrichtung / Abteilung: _____

Führen Sie Ihr Projekt im Rahmen eines Exzellenzclusters durch? : Nein ☐ Ja ☐

Wenn ja, für welches Exzellenzcluster ? : _____

Verantwortliche(r) Leiter/in des Projekts : _____ Tel. : _____

Nutzer/in : _____ Tel. : _____

Nutzerkennzeichen (soweit vorhanden) : _____

E-Mail-Adresse: _____

Aus Drittmitteln finanziertes Projekt; bitte Fördernummer angeben : _____

2. Personenbezogene Daten

Die Verarbeitung **personenbezogener Daten** (Einzelangaben über persönliche oder sachliche Verhältnisse einer bestimmten oder bestimmbarer natürlichen Person) unterliegt den Datenschutzgesetzen des Bundes sowie des Landes Schleswig-Holstein. Die Speicherung oder auch Verarbeitung personenbezogener Daten auf den Hochleistungsrechnern der CAU ist ausschließlich für Forschungszwecke zulässig, wenn die Daten dort bereits in anonymisierter oder pseudonymisierter Form vorliegen. Die Unterzeichnenden bestätigen durch ihre Unterschrift:

- ☐ Mit dem hier beantragten Account werden auf den HPC-Systemen keinerlei personenbezogene Daten gespeichert oder weiter verarbeitet.
- ☐ Mit dem hier beantragten Account werden auf den HPC-Systemen zu Forschungszwecken personenbezogene Daten verarbeitet, die auf separaten Systemen zuvor vollständig anonymisiert wurden. Die Beziehungen zwischen einzelnen Datensätzen und den jeweils zugehörigen, natürlichen Personen wurden vernichtet und können nicht wiederbeschafft werden.
- ☐ Mit dem hier beantragten Account werden auf den HPC-Systemen zu Forschungszwecken personenbezogene Daten verarbeitet, die auf separaten Systemen zuvor vollständig pseudonymisiert wurden. Das komplette Pseudonymisierungsverfahren ist vollständig dokumentiert und der/dem Datenschutzbeauftragten auf Anfrage auszuhändigen. Weder die Pseudonymisierungsschlüssel noch die Dokumentation des Pseudonymisierungsverfahrens selbst sind auf den HPC-Systemen der CAU gespeichert.

3. Kurzbeschreibung des Lehr-/Forschungsprojektes

4. Ressourcen-Anforderungen des Projektes

Welche speziellen Softwareprodukte sind für die Durchführung des Projektes erforderlich?

5. Allgemeine Informationen

- Grundlage für die Nutzung des Rechenzentrums ist die **Benutzungsrahmenordnung** für die Kommunikations- und Datenverarbeitungsinfrastruktur der Christian-Albrechts-Universität zu Kiel (siehe <https://www.rz.uni-kiel.de/de/ueber-uns/dokumente/benutzungsrahmenordnung.pdf>) .
- Die zu bearbeitenden Aufgaben werden finanziert aus Mitteln der Hochschule/angegliederten Einrichtungen oder aus Zuwendungen der DFG. Bei allen anderen Finanzierungen ist die schriftliche Genehmigung des Rechenzentrums erforderlich.
- **Die Nutzungserlaubnis gilt bis 15. Februar des folgenden Jahres. Bei Abmeldung oder bei Nichtverlängerung werden die Daten auf allen Plattenspeichern und Magnetbandkassetten nach 6 Monaten gelöscht.**
- Die in diesem Antrag angegebenen persönlichen Daten werden elektronisch gespeichert und verarbeitet. Mit der Unterschrift wird dazu das Einverständnis erklärt.

Die Unterzeichnenden erkennen die folgenden Auflagen an:

- Die umseitig genannten Rechner gelten als Dual-Use-Technologie. Ihre Nutzung unterliegt Einschränkungen, die sich aus EU-Recht sowie aus dem Außenwirtschaftsgesetz (AWG) und der Außenwirtschaftsverordnung (AWV) ergeben. Die Unterzeichnenden bestätigen durch ihre Unterschrift, dass die diesbezüglich geltenden Exportbeschränkungen des Bundesamtes für Ausfuhrkontrolle eingehalten werden (siehe <http://www.ausfuhrkontrolle.info/ausfuhrkontrolle/de/embargos/uebersicht/index.html>).
- Eine Nutzungsberechtigung beschränkt sich auf die unter "Nutzer/in" genannte Person. Es ist nicht erlaubt anderen, z.B. durch Preisgabe des Passworts, über den zugeteilten Login Namen einen Zugang zu den entsprechenden Rechnern zu ermöglichen.
- Bei Nichtbeachtung dieser Bestimmungen ist der betreffende Benutzer (natürliche oder juristische Person) in vollem Umfang schadenersatzpflichtig.

6. Unterschriften:

	(Institutsstempel)
Kiel, den _____ , _____	(verantwortlicher Leiter/in)
Kiel, den _____ , _____	(Nutzer/in)
Kiel, den _____ , _____	(Geschäftsführende(r) Direktor/in der Einrichtung bzw. bei Anträgen aus dem GEOMAR: Leiter/in RZ GEOMAR)

- Nutzung für alle Mitarbeiter der Universitäten und Fachhochschulen möglich
- Beantragungsweg abhängig vom Rechenzeitbedarf
 - Beantragung von Testprojekten
 - Beantragung eines Großprojekten

- Beantragung ist jederzeit möglich (WWW-Formular)
- <https://zulassung.hlrn.de>
- begrenztes Kontingent an Rechenzeit:
 - 75.000 Core-Stunden pro Quartal (erweiterbar auf 300.000 Core-Std.)

- dient zur Durchführung von größeren Forschungsvorhaben
- <https://www.nhr-verein.de/rechnernutzung>
- Antragsfrist jeweils 3 Monate vor Laufzeitbeginn (1.01., 1.04., 1.07., 1.10.)
- Kontingente werden jeweils für ein Jahr bewilligt und quartalsweise zugeteilt
- 2 Arten von Großprojekten
 - NHR-Normal: 300.000 – 5.000.000 Core-Stunden pro Quartal
 - NHR-Large: > 5.000.000 Core-Stunden pro Quartal

Nutzung unserer lokalen HPC-Systeme



- **interaktives Einloggen nur auf Vorrechner (Login-Knoten) möglich**
 - direkter Zugriff nur über IP-Adressen aus dem CAU- und GEOMAR-Netz bzw. UKSH-Netz
 - Zugriff aus anderen Netzen nur über eine VPN-Verbindung
 - VPN-Verbindung mit su- bzw. smomw-Kennung aufbauen (nicht mit stu-Account)
- **Aufbau einer ssh-Verbindung über ssh-Client vom lokalen PC aus:**
 - MAC und Linux: ssh-Client in Standarddistributionen enthalten
 - Windows-Clients:
 - putty:
 - <http://www.putty.org/>
 - kein Arbeiten mit einer grafischen Benutzeroberfläche möglich
 - Windows ssh command line tool cmd:
 - [Windows+R] Tastenkombination; kein Arbeiten mit GUI
 - MobaXterm:
 - <https://mobaxterm.mobatek.net/> (ssh client mit scp Möglichkeit)
 - SmartTTY:
 - <http://smartty.sysprogs.com/> (ssh client mit scp Möglichkeit)

HPC-Systeme: Zugang

- Zugriff auf [Login-Knoten NEC HPC-System](#)
 - `ssh -X username@nesh-fe.rz.uni-kiel.de`
 - -X Option aktiviert X11-Forwarding
 - Nutzung einer grafischen Benutzeroberfläche
- Zugriff auf [caucluster Login-Knoten](#)
 - `ssh -X username@caucluster.rz.uni-kiel.de`
- *ssh-Kommando auf lokalem PC ausführen*

- **Linux und MAC:**
 - scp-Kommando:
 - `scp dateiname username@nesh-fe.rz.uni-kiel.de:/verzeichnisname`
 - rsync-Kommando:
 - `rsync -av dateiname username@nesh-fe.rz.uni-kiel.de:/verzeichnisname`
- **Windows:**
 - WinSCP (grafischer scp-Client)
 - <https://winscp.net/eng/docs/lang:de>
 - SmarTTY:
 - <http://smartty.sysprogs.com/> (ssh client mit scp Möglichkeit)
 - MobaXterm:
 - <https://mobaxterm.mobatek.net/> (ssh client mit scp Möglichkeit)
- *scp-, rsync-Kommando bzw. WinSCP-Aufruf auf lokalem PC ausführen*

- **Verfügbare Editoren auf HPC-Systemen** (*Vorrechner, HPC-Login*)
 - vi
 - Emacs
 - nedit
 - nano
- **Arbeiten mit Dateien, die unter Windows erzeugt/bearbeitet wurden**
 - nur als reine ASCII-Dateien abspeichern auf lokalem PC
 - vermeidet Fehlermeldung
 - /bin/bash^M: bad interpreter: no such file or directory
 - Kommando `dos2unix` dateiname → entfernt Steuerzeichen
 - dos2unix-Kommando auf Vorrechner (HPC-Login) ausführen

- Benutzerkennung erhalten Sie per E-Mail
- Passwortverwaltung über das CIM-Portal
 - <https://cim.rz.uni-kiel.de/cimportal>
- Setzen eines Wunschkpasswortes vor dem ersten Einloggen
- Passwortänderungen auch über das CIM-Portal



Dateisysteme und Datensicherung



- Vier verschiedene Dateisysteme nutzbar
 - HOME-Verzeichnis
 - WORK-Verzeichnis
 - lokaler Plattenplatz an den Rechenknoten
 - Zugriff auf ein CEPH-Dateisystem

- Login-Verzeichnis
- erreichbar über die Umgebungsvariable \$HOME
- verfügbarer Plattenplatz (für alle Nutzer)
 - NEC HPC-System: 300TB
 - cauccluster: 85TB
- global verfügbar an allen Knoten
- tägliche Datensicherung durch das Rechenzentrum
- Quoten für nutzbaren Plattenplatz pro Benutzer
- Verzeichnis für wichtige Daten, die gesichert werden müssen
 - Skripte, Programme und kleinere Ergebnisdateien
- nicht für die Durchführung von Batchjobs verwenden
 - u.a. langsame Zugriffszeiten

- globales paralleles Dateisystem
- Dateisystem für die Durchführung von Batchberechnungen
- erreichbar über
 - \$WORK-Umgebungsvariable
- keine Datensicherung durch das Rechenzentrum
- verfügbarer Plattenplatz (für alle Nutzer insgesamt)
 - NEC HPC-System: 10 PB GxFS
 - cauccluster: 1,2 PB BeeGFS
- Quoten für nutzbaren Plattenplatz pro Benutzer



\$WORK: Plattenplatzkontingent

- **NEC HPC-System**
 - \$WORK Standardkontingente
 - Nutzbarer Speicherplatz
 - Softlimit: 1.8TB (CAU) bzw. 4.5TB (GEOMAR)
 - Hardlimit: 2.0TB (CAU) bzw. 5.0TB (GEOMAR)
 - \$HOME Standardkontingent
 - Nutzbarer Speicherplatz: 150GB Soft- und 200GB Hardlimit
- **caucluster**
 - \$WORK Standardkontingent:
 - Speicherplatz: 2TB
 - \$HOME Standardkontingent:
 - Speicherplatz: 100 GB Hardlimit
 - Kommando zur Quotenabfrage: **workquota**

Plattenplatzkontingent: workquota I

```
[suabc012@nesh-fe2 ~]$ workquota
```

```
*****
```

```
* Your current quota on $HOME=/gxfs_home/cau/suabc012 *
```

```
*          [used] |      [soft] |      [hard] | [grace] |   [files] | *
```

```
* disk space    10.125G | 150.000G | 200.000G |         |        353 | *
```

```
*****
```

```
* Your current quota on $WORK=/gxfs_work1/cau/suabc012 *
```

```
*          [used] |      [soft] |      [hard] | [grace] |   [files] | *
```

```
* disk space (fs2) 1.100M | 1.953T | 2.148T |         |         34 | *
```

```
*****
```

Use 'workquota --raw' to see native gxfs quota information

Plattenplatzkontingent: workquota II

```
[suabc012@nesh-fe2 ~]$ workquota
```

```
*****
```

```
* Your current quota on $HOME=/gxfs_home/cau/suabc012 *
```

```
*          [used] |      [soft] |      [hard] | [grace] |   [files] | *
```

```
* disk space    10.125G | 150.000G | 200.000G |      |      353 | *
```

```
*****
```

```
* Your current quota on $WORK=/gxfs_work1/cau/suabc012 *
```

```
*          [used] |      [soft] |      [hard] | [grace] |   [files] | *
```

```
* disk space (fs2) 2.010T | 1.953T | 2.148T |      6 |      4524 | *
```

```
*****
```

Use 'workquota --raw' to see native gxfs quota information

Plattenplatzkontingent: workquota II

```
[suabc012@nesh-fe2 ~]$ workquota
```

```
*****
```

```
* Your current quota on $HOME=/gxfs_home/cau/suabc012 *
```

```
*          [used] |      [soft] |      [hard] | [grace] |   [files] | *
```

```
* disk space    10.125G | 150.000G | 200.000G |         |        353 | *
```

```
*****
```

```
* Your current quota on $WORK=/gxfs_work1/cau/suabc012 *
```

```
*          [used] |      [soft] |      [hard] | [grace] |   [files] | *
```

```
* disk space (fs2) 2.300T | 1.953T | 2.148T | expired |        6134 | *
```

```
*****
```

Use 'workquota --raw' to see native gxfs quota information

- erreichbar über Umgebungsvariable
 - \$TMPDIR
- lokaler temporärer Plattenplatz am jeweiligen Batchknoten
- nur innerhalb eines Batchjobs verfügbar
- schnelle Zugriffszeiten: wichtig für I/O intensive Berechnungen (viele Lese- und/oder Schreibprozesse in kurzer Zeit)



- erreichbar über
 - Umgebungsvariable \$CEPH
- Default Quota: 1 TB
- nur lokal auf Vorrechnern verfügbar
- Speicherung von aktuell nicht benötigten Daten
 - erstellen von tar-Dateien

- **HOME-Verzeichnis**
 - tägliche Sicherung
 - Datensicherungen werden 8 Wochen aufbewahrt
 - für die letzten 2 Wochen ist eine tagesaktuelle Wiederherstellung möglich
 - ältere Datensicherungen stehen in Wochenabständen zur Verfügung
- **\$WORK:**
 - keine Sicherung durch das Rechenzentrum
- **\$CEPH:**
 - keine Sicherung durch das Rechenzentrum
- Daten nur verfügbar solange ein Account aktiv ist
- keine Langzeitarchivierung durch das Rechenzentrum

Arbeiten mit den Dateisystemen

- wichtige Daten, die gesichert werden müssen auf \$HOME
- Batchberechnungen
 - starten und Ausführen auf \$WORK
 - Nutzung von \$TMPDIR für I/O intensive Berechnungen
- \$CEPH-Verzeichnis
 - Verzeichnis für aktuell nicht benötigte Daten
 - bevorzugt große Datenpakete abspeichern
- eigenen Datenbestand von Zeit zu Zeit überprüfen

Anwendersoftware und Programmübersetzung



- **Compiler**
 - Gnu-Compiler
 - Intel-Compiler
 - SX-Crosscompiler für Nutzung der NEC SX-Aurora TSUBASA Knoten
 - NVIDIA Cuda Toolkit für GPU Nutzung
 - **MPI-Implementierung**
 - Intel-MPI, OpenMPI (Linux-Cluster)
 - spezielles SX-MPI (für SX-Aurora TSUBASA)
 - **Bibliotheken**
 - netCDF, HDF5, FFTW, MKL, PETSc ...
 - **Anwendersoftware:**
 - Python, Perl, R, Matlab, Gaussian, Turbomole,
- Bereitstellung über ein Module-Konzept

- ermöglicht Nutzung bereits global installierter Software
- Softwareverzeichnis nicht im Standardsuchpfad
- nach dem Laden eines Softwaremodules:
 - Programm ohne weitere Pfadanpassungen nutzbar
- einfaches Setzen/Umsetzen von Suchpfaden:
 - \$PATH, \$LD_LIBRARY_PATH, \$MANPATH



- **module avail**
 - liefert Überblick über bereits installierte Softwarepakete
- **module load *name***
 - lädt Module mit dem Name *name*; alle Einstellungen für die Programmnutzung werden vorgenommen
- **module list**
 - listet alle Module auf, die aktuell geladen sind
- **module unload *name***
 - entfernt das Module *name*, d.h. alle Einstellungen werden rückgängig gemacht
- **module purge**
 - entfernt alle bisher geladenen Module, d.h. module list Kommando liefert keine Einträge mehr
- **module show *name***
 - zeigt Pfadänderung an, die das Module durchführt

Modules (NEC HPC-System): Software I

```
[szzrs112@nesh-fe1 ~]$ module avail
```

```
----- /gxfs_home/sw/modules/cluster-compilers ----
```

```
...
```

```
----- /gxfs_home/sw/modules/cluster-libraries -----
```

```
...
```

```
----- /gxfs_home/sw/modules/cluster-software -----
```

```
....
```

```
----- /gxfs_home/sw/modules/cluster-tools -----
```

```
...
```

```
----- /gxfs_home/sw/modules/gpu -----
```

```
....
```

```
----- /gxfs_home/sw/modules/vector -----
```



```
[szzrs112@nesh-fe1 ~]$ module avail
```

```
----- /gxfs_home/sw/modules/cluster-compilers -----
```

```
gcc/9.3.0                intel/19.0.4      intelmpi/20.0.4    (D)  openmpi-intel20/3.1.6
gcc/10.2.0              (D)  intel/20.0.4    (D)  llvm/11.0.0          openmpi/3.1.6
intel-parallel-studio/20.0.4  intelmpi/19.0.4  openmpi-intel19/3.1.6
```

```
----- /gxfs_home/sw/modules/cluster-libraries -----
```

```
boost-intel/1.74.0  fftw-intel/3.3.9  hdf5-intel/1.10.7  libxml2/2.9.10  netcdf-fortran-
intel/4.5.3
boost/1.74.0        fftw/3.3.9        hdf5-intel/1.12.0 (D)  libxtst/1.2.2   netcdf-fortran/4.5.3
bzip2/1.0.8         glpk/4.65         hdf5/1.10.7test    mpfr/4.1.0      netlib-lapack/3.8.0
curl/7.72.0         gmp/6.2.1         hdf5/1.12.0        (D)  netcdf-c-intel/4.7.4  readline/6.3
eigen-intel/3.3.8   gsl-intel/2.5     libaec/1.0.2       netcdf-c/4.7.4  sbml/5.18.0
eigen/3.3.8         gsl/2.5           libzip/2.1.1       netcdf-cxx4-intel/4.3.1  xz/5.2.5
expat/2.2.10       hdf5-intel/1.10.7test  libxc/5.0.0       netcdf-cxx4/4.3.1  zlib/1.2.11
```

Modules NEC HPC.System: Software III

----- /gxfs_home/sw/modules/[cluster-software](#) -----

R/info	gnuplot/5.2.8	ncview/2.1.8	petsc-intel/3.14.4 (D)	
R/4.0.2	(D) imagemagick/7.0.8-7	octave/5.2.0	python/info	
asymptote/2.68	julia/1.5.2	octopus/6.0	python/2.7.18	
blast-plus/2.9.0	lammps/20181207	octopus/7.3	python/3.7.9	
bowtie2/2.4.2	matlab/2018b	octopus/10.0	(D) python/3.8.6	(D)
cdo/1.9.9	matlab/2019b	openbabel/3.0.0	samtools/1.10	
cp2k/7.1	matlab/2020b	(D) openfoam/8	singularity/3.5.2	
esmf/7.1.0r	matlab_geomar/2018b	openjdk/11.0.2	singularity/3.6.4 (D)	
espresso/6.6	matlab_geomar/2019b	perl/info	tcl/8.6.10	
ferret/7.2	matlab_geomar/2020b (D)	petsc-debug-intel/3.6.1	turbomole/7.5	
flex/2.6.4	miniconda2/4.8.3	petsc-debug-intel/3.7.6	turbomolempi/7.5	
g16/a03	miniconda3/4.9.2	petsc-debug-intel/3.14.4 (D)	turbomolesmp/7.5	
ghostscript/9.50	ncl/6.6.2	petsc-intel/3.6.1		
gmt/6.1.0	nco/4.9.3	petsc-intel/3.7.6		

----- /gxfs_home/sw/modules/**cluster-tools** -----

autoconf/2.69 emacs/27.1 lftp/4.9.2 openssh/8.4p1 texinfo/6.5
automake/1.16.3 gdb/9.2 libtool/2.4.6 parallel/20200822 texlive/20200406
cmake/3.18.4 git-lfs/2.11.0 mc/4.8.23 pigz/2.4
ed/1.4 git/2.29.0 nano/4.9 subversion/1.14.0

----- /gxfs_home/sw/modules/**gpu** -----

cuda/11.1.0 cudnn/8.0.4.30-11.1 cutensor/1.2.2

----- /gxfs_home/sw/modules/**vector** -----

ncc/3.1.0 necmpi/2.11.0 necnlc/2.1.0 szip-ve/2.1.1 vetop/1.0 zlib-ve/1.2.11

Where:

D: Default Module

Use "module spider" to find all possible modules and extensions.

Use "module keyword key1 key2 ..." to search for all possible modules matching any of t

- **Nutzung Programmpaket Matlab**
 - `module load matlab/2020b` (CAU-Nutzer)
 - `module load matlab_geomar/2020b` (GEOMAR-Nutzer)
 - Programmaufruf mit Kommando `matlab –nodisplay` starten
- **Programmübersetzung mit Intel-Compiler**
 - `module load intel/20.0.4` (Compileraufruf: `ifort`, `icc` oder `icpc`)
 - `module load intel/20.0.4 intelmpi/20.0.4`
 - Compileraufruf: `mpiifort`, `mpiicc` oder `miicpc`
- **Programmübersetzung SX-Aurora TSUBASA**
 - `module load ncc/3.1.0` (Compileraufruf: `nfort`, `ncc` oder `n++`)
- **Programmübersetzung GPU NVIDIA Karten**
 - `module load cuda/11.1.0` (compileraufruf: `nvcc`)

```
[szzrs112@caucluster1 ~]$ module avail
```

```
----- /zfshome/sw/modules/default-env -----
```

```
default-env/info  matlab/R2022a  micromamba/1.3.1
```

```
----- /zfshome/sw/modules/compiler-env -----
```

```
compiler-env/info  gcc12-env/12.1.0  gcc8-env/8.5.0  intel2021-  
env/2021.6.0
```

```
[szzrs112@cauccluster1 ~]$ module load gcc12-env/12.1.0
```

```
[szzrs112@cauccluster1 ~]$ module avail
```

```
----- /zfshome/sw/modules/default-env -----
```

```
default-env/info  matlab/R2022a  micromamba/1.3.1
```

```
----- /zfshome/sw/modules/compiler-env -----
```

```
compiler-env/info  gcc12-env/12.1.0 (L)  gcc8-env/8.5.0  intel2021-env/2021.6.0
```

```
----- /zfshome/sw/modules/gcc12-env/12.1.0 -----
```

```
R/4.0.0                git-lfs/3.2.0          openblas/0.3.20
```

```
R/4.2.1-with-intel-mkl-2022.1.0  git/2.38.1          openjdk/11.0.15_10
```

```
R/4.2.1-with-openblas-0.3.20    glpk/4.65            openjdk/16.0.2  
(D)
```

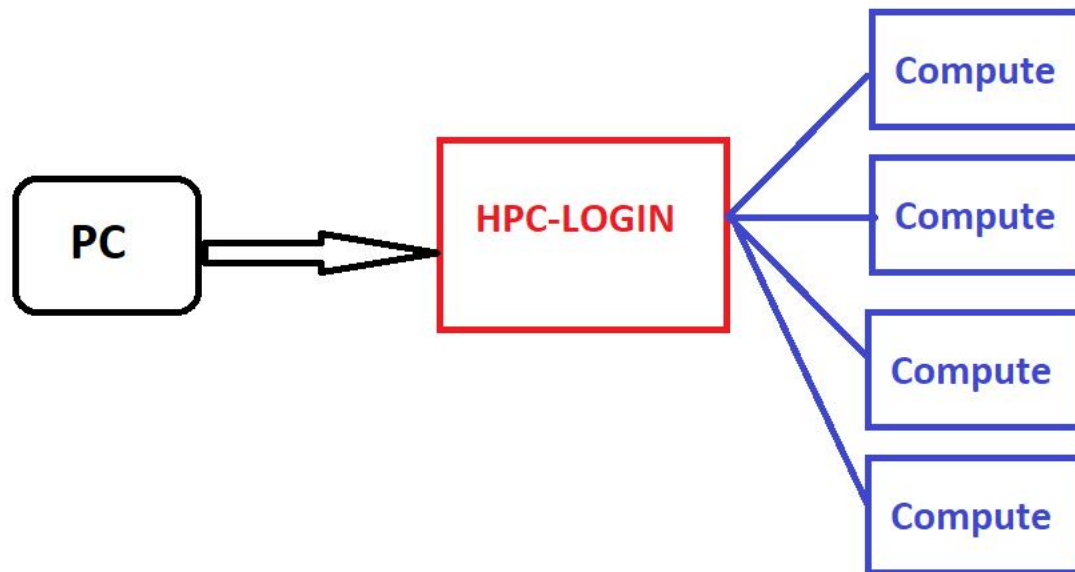
```
R/4.2.1                (D)  gmp/6.2.1          openmpi/4.1.3
```

```
asymptote/2.83          gmt/6.2.0            openssh/9.0p1
```

- **Kommando: `module all`**
 - liefert Überblick über alle bereits installierte Softwarepakete

Interaktives Arbeiten und Batchbetrieb





- **Interaktives Arbeiten**

- nur auf den Login-Knoten (HPC-Login)
- Datentransfer: PC ↔ HPC-Login ↔ CEPH
- Programmübersetzung, kurze Testberechnungen
- Erstellung der Batchskripte
- Abgabe und Kontrolle von Batchjobs
- keine längeren Berechnungen → über Batchsystem

- **Batchbetrieb**

- Berechnungen werden auf den Compute-Knoten ausgeführt
- kein direkter Zugriff auf ausführende Knoten

- **Vorteil**

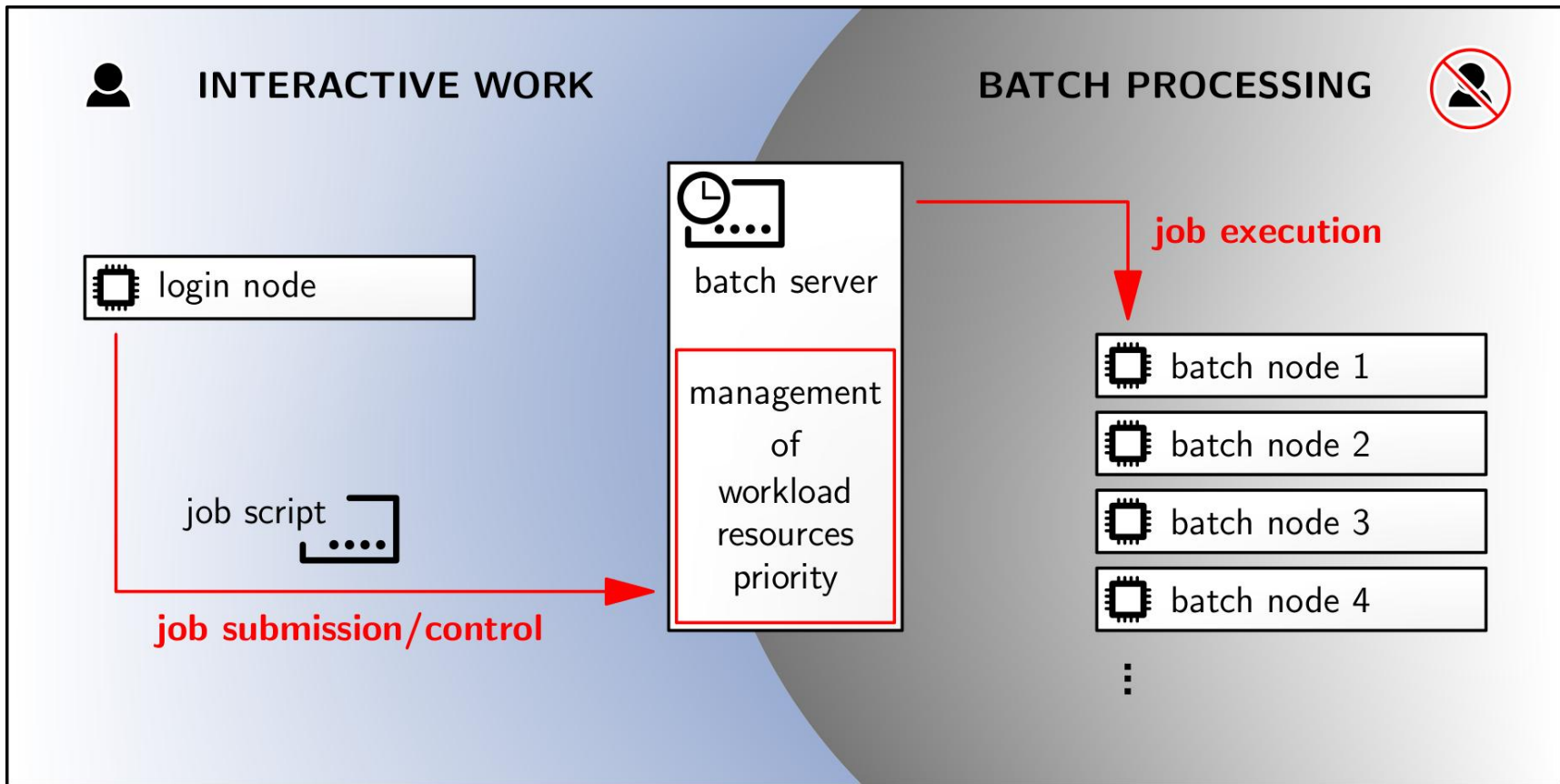
- Rechenressourcen können effektiver genutzt werden
 - höherer Durchsatz
 - gerechte Verteilung auf alle Benutzer
 - jeder Benutzer kann mehrere Berechnungen gleichzeitig ausführen
- angeforderte Ressourcen sind während der kompletten Laufzeit garantiert
- Möglichkeit Jobabhängigkeiten oder komplette Workflows zu definieren

- **Nachteil**

- zunächst etwas ungewohntes Arbeiten



- Benutzer startet sein Programm nicht direkt in der Konsole
- schreibt eine kleine Batchskript-Datei
 - enthält Informationen über die angeforderten/benötigten Ressourcen
 - Core-Anzahl, Hauptspeicherbedarf, Rechenzeit,
 - enthält den eigentlichen Programmaufruf
- Batchskript-Datei wird vom Login-Knoten an den Batchserver übergeben
 - durchsucht alle Knoten nach freien Ressourcen
 - Job startet sofort oder wird in eine Warteschlange gestellt
- Batchsystemsoftware: SLURM



SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

```
#!/bin/bash
```

```
#SBATCH --nodes=1
```

```
#SBATCH --tasks-per-node=1
```

```
#SBATCH --cpus-per-task=1
```

```
#SBATCH --time=01:00:00
```

```
#SBATCH --mem=10000mb
```

```
#SBATCH --partition=cluster
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out
```

```
#SBATCH --error=testjob.err
```

```
#Laden der Softwareumgebung
```

```
module load intel/20.0.4
```

```
# Programmstart
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

`#!/bin/bash` → Shell-Definition

`#SBATCH --nodes=1`

`#SBATCH --tasks-per-node=1`

`#SBATCH --cpus-per-task=1`

`#SBATCH --time=01:00:00`

`#SBATCH --mem=10000mb`

`#SBATCH --partition=cluster`

`#SBATCH --job-name=testjob`

`#SBATCH --output=testjob.out`

`#SBATCH --error=testjob.err`

`#Laden der Softwareumgebung`

`module load intel/20.0.4`

`# Programmstart`

`./prog`

`# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)`

`jobinfo`

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

`#!/bin/bash` → Shell-Definition

`#SBATCH --nodes=1` → Anzahl benötigter Rechenknoten

`#SBATCH --tasks-per-node=1`

`#SBATCH --cpus-per-task=1`

`#SBATCH --time=01:00:00`

`#SBATCH --mem=10000mb`

`#SBATCH --partition=cluster`

`#SBATCH --job-name=testjob`

`#SBATCH --output=testjob.out`

`#SBATCH --error=testjob.err`

`#Laden der Softwareumgebung`

`module load intel/20.0.4`

`# Programmstart`

`./prog`

`# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)`

`jobinfo`

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

```
#!/bin/bash
```

```
#SBATCH --nodes=1
```

```
#SBATCH --tasks-per-node=1 → Anzahl MPI-Prozesse pro Rechenknoten
```

```
#SBATCH --cpus-per-task=1 → Anzahl Tasks pro Knoten (z.B. OpenMP Parallelisierung)
```

```
#SBATCH --time=01:00:00
```

```
#SBATCH --mem=10000mb
```

```
#SBATCH --partition=cluster
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out
```

```
#SBATCH --error=testjob.err
```

```
#Laden der Softwareumgebung
```

```
module load intel/20.0.4
```

```
# Programmstart
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

```
#!/bin/bash
```

```
#SBATCH --nodes=1
```

```
#SBATCH --tasks-per-node=1
```

```
#SBATCH --cpus-per-task=1
```

```
#SBATCH --time=01:00:00 → angeforderte Rechenzeit (Verweilzeit)
```

```
#SBATCH --mem=10000mb
```

```
#SBATCH --partition=cluster
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out
```

```
#SBATCH --error=testjob.err
```

```
#Laden der Softwareumgebung
```

```
module load intel/20.0.4
```

```
# Programmstart
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

```
#!/bin/bash
```

```
#SBATCH --nodes=1
```

```
#SBATCH --tasks-per-node=1
```

```
#SBATCH --cpus-per-task=1
```

```
#SBATCH --time=01:00:00
```

```
#SBATCH --mem=10000mb → max. benötigter Hauptspeicher pro Knoten
```

```
#SBATCH --partition=cluster
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out
```

```
#SBATCH --error=testjob.err
```

```
#Laden der Softwareumgebung
```

```
module load intel/20.0.4
```

```
# Programmstart
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

```
#!/bin/bash
```

```
#SBATCH --nodes=1
```

```
#SBATCH --tasks-per-node=1
```

```
#SBATCH --cpus-per-task=1
```

```
#SBATCH --time=01:00:00
```

```
#SBATCH --mem=10000mb
```

```
#SBATCH --partition=cluster → Partition auf der Job laufen soll (hier x86 Linux cluster)
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out
```

```
#SBATCH --error=testjob.err
```

```
#Laden der Softwareumgebung
```

```
module load intel/20.0.4
```

```
# Programmstart
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=01:00:00
#SBATCH --mem=10000mb
#SBATCH --partition=cluster
#SBATCH --job-name=testjob → Name des Batchjobs
#SBATCH --output=testjob.out
#SBATCH --error=testjob.err
```

#Laden der Softwareumgebung

module load intel/20.0.4

Programmstart

./prog

Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)

jobinfo

SLURM-Batchskript (NEC Linux Cluster) serielle Berechnung

```
#!/bin/bash
```

```
#SBATCH --nodes=1
```

```
#SBATCH --tasks-per-node=1
```

```
#SBATCH --cpus-per-task=1
```

```
#SBATCH --time=01:00:00
```

```
#SBATCH --mem=10000mb
```

```
#SBATCH --partition=cluster
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out → Dateiname für Standardausgabe
```

```
#SBATCH --error=testjob.err → Dateiname für Fehlerausgabe
```

```
#Laden der Softwareumgebung
```

```
module load intel/20.0.4
```

```
# Programmstart
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```

SLURM-Batchskript (NEC Linux Cluster) parallele MPI Berechnung

`#!/bin/bash`

`#SBATCH --nodes=2` → Anforderung von 2 Rechenknoten

`#SBATCH --tasks-per-node=32` → auf jedem Knoten werden 32 Cores benötigt

`#SBATCH --cpus-per-task=1`

`#SBATCH --time=01:00:00`

`#SBATCH --mem=10000mb`

`#SBATCH --partition=cluster`

`#SBATCH --job-name=testjob`

`#SBATCH --output=testjob.out`

`#SBATCH --error=testjob.err`

`#Laden der Softwareumgebung`

`module load intel/20.0.4 intelmpi/20.04`

`# Programmstart`

`mpirun -np 64 ./prog`

`# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)`

`jobinfo`

SLURM-Batchskript (NEC Linux Cluster) parallele OpenMP Berechnung

```
#!/bin/bash
```

```
#SBATCH --nodes=1 → Anforderung von 1 Rechenknoten
```

```
#SBATCH --tasks-per-node=1
```

```
#SBATCH --cpus-per-task=16 → auf dem Knoten sollen 16 OpenMP Threads laufen
```

```
#SBATCH --time=01:00:00
```

```
#SBATCH --mem=10000mb
```

```
#SBATCH --partition=cluster
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out
```

```
#SBATCH --error=testjob.err
```

```
#Laden der Softwareumgebung und Programmstart
```

```
module load intel/20.0.4
```

```
export OMP_NUM_THREADS=16
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --gres=ve:1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=01:00:00
#SBATCH --mem=10000mb
#SBATCH --partition=vector
#SBATCH --job-name=testjob
#SBATCH --output=testjob.out
#SBATCH --error=testjob.err

export VE_PROGINF=DETAIL
export OMP_NUM_THREADS=8
./vector.x
```

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --gres=ve:1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=01:00:00
#SBATCH --mem=10000mb
#SBATCH --partition=vector
#SBATCH --job-name=testjob
#SBATCH --output=testjob.out
#SBATCH --error=testjob.err

export VE_PROGINF=DETAIL
module load necmpi/2.11.0
source necmpivars.sh
mpirun -nn 1 -nnp 8 -ve 0-0 ./vector.x
```

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --gres=ve:8
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=01:00:00
#SBATCH --mem=10000mb
#SBATCH --partition=vector
#SBATCH --job-name=testjob
#SBATCH --output=testjob.out
#SBATCH --error=testjob.err

export VE_PROGINF=DETAIL
module load necmpi/2.11.0
source necmpivars.sh
mpirun -nn 1 -nnp 64 -ve 0-7 ./vector.x
```



SLURM-Batchskript (GPU-Knoten)

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --gpus-pernode=1
#SBATCH --time=01:00:00
#SBATCH --mem=10000mb
#SBATCH --partition=gpu
#SBATCH --job-name=testjob
#SBATCH --output=testjob.out
#SBATCH --error=testjob.err

module load intel/19.0.4
time ./prog
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
jobinfo
```

SLURM-Batchskript (caucluster)

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=01:00:00
#SBATCH --mem=10000mb
#SBATCH --partition=base (--partition=highmem für Knoten mit 2 bzw. 4TB RAM)
#SBATCH --job-name=testjob
#SBATCH --output=testjob.out
#SBATCH --error=testjob.err

module load intel2021-env/2021.6.0
time ./prog

# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
jobinfo
```

SLURM: Abgabe/Kontrolle von Batchjobs

- Kenntnis weniger Kommandos ausreichend
 - **sbatch *jobskript***
 - Abgabe einer Batchberechnung
 - **squeue**
 - Auflistung aller Jobs im System
 - **squeue -u *username* bzw. squeue --me**
 - Auflistungen der eigenen Jobs
 - **scancel *jobid***
 - Beenden eines laufenden bzw. entfernen eines wartenden Jobs
 - **scontrol show job *jobid***
 - liefert Details über einen bestimmten Job
 - **sinfo**
 - Zeigt Informationen über verfügbare Partitionen an

SLURM: weitere Kommandos

- max. Rechenzeit per Default: 48 Stunden
 - `#SBATCH --qos=long` → Rechenzeit bis 10 Tage möglich
 - Für längere Rechenzeiten extra Validierung erforderlich
- benötigte CPU-Art genauer spezifizieren
 - `sinfo --Node -o "%20N %20c %20m %20f %20d"`
 - z.B. NEC-Cluster: Skylake oder Cascade Lake CPU
 - `#SBATCH --constraint=cascade`

SLURM-Batchskript : jobinfo Kommando

```
#!/bin/bash
```

```
#SBATCH --nodes=1
```

```
#SBATCH --tasks-per-node=1
```

```
#SBATCH --cpus-per-task=1
```

```
#SBATCH --time=01:00:00
```

```
#SBATCH --mem=10000mb
```

```
#SBATCH --partition=cluster
```

```
#SBATCH --job-name=testjob
```

```
#SBATCH --output=testjob.out
```

```
#SBATCH --error=testjob.err
```

```
#Laden der Softwareumgebung
```

```
module load intel/20.0.4
```

```
# Programmstart
```

```
./prog
```

```
# Ausgabe der verbrauchten Ressourcen (Rechenzeit, Hauptspeicher des Jobs)
```

```
jobinfo
```


- Requested resources:

Timelimit = 10-00:00:00 (864000s)

MinMemoryNode = 180000M
(180000.000M)

NumNodes = 1

NumCPUs = 32

NumTasks = 1

CPUs/Task = 32

- Used resources:

RunTime = 1-01:29:57 (91797s)

MaxRSS = 31857880K (31111.211M)

=====

- Important conclusions and remarks:

* !!! Please, always check if the number of requested cores and nodes matches the need of your program/code !!!

* !!! Less than 20% of requested walltime used !!! Please, adapt your batch script.

* !!! Less than 20% of requested main memory used !!! Please, adapt your batch script.



- Möglichkeit interaktiv auf Batchknoten zu arbeiten
- cauccluster:
`srun --pty --nodes=1 --cpus-per_task=1 --time=1:00:00 partition=base /bin/bash`
- NEC HPC-System
`srun --pty --nodes=1 --cpus-per-task=1 --mem=1000mb --time=00:10:00 --partition=cluster /bin/bash`
`srun --pty --nodes=1 --cpus-per-task=1 --mem=1000mb --time=00:10:00 --gres=ve:1 --partition=vector /bin/bash`

Arbeiten mit dem Batchsystem

- nicht mehr Knoten und Cores anfordern als benötigt werden
- benötigte Rechenzeit und Hauptspeicher möglichst genau angeben
 - höherer Durchsatz und kürzere Wartezeiten
- Zwischenergebnisse abspeichern (insbes. bei sehr langen Laufzeiten)
- Standardausgabedatei möglichst klein halten
 - Ausgaben evtl. direkt in WORK-Verzeichnis umleiten

- WWW-pages:
 - HPC-Angebote:
 - <https://www.hiperf.rz.uni-kiel.de>
 - NEC HPC-System:
 - <https://www.rz.uni-kiel.de/de/angebote/hiperf/nesh>
 - cauccluster:
 - <http://www.hiperf.rz.uni-kiel.de/cauccluster/>
 - Vortragsfolien:
 - <https://www.rz.uni-kiel.de/de/angebote/hiperf/hpc-kurse>
- **Mailingliste/Ticketsystem**
 - hpcsupport@rz.uni-kiel.de