ForWind
Center for Wind Energy Research

Introduction to HPC

Dr. Stefan Albensoeder

Contact: Stefan.Albensoeder@uni-oldenburg.de

Leibniz Universität Hannover

CARL VON OSSIETZKY universität OLDENBURG

Universität Bremen

Introduction to HPC

MOTIVATION

HPC (High-Performance-Computing)

- Computing at the limit of computational resources
- Why HPC?
  - Enable to solve computational intensive problems in an optimal way
  - To increase energy efficiency
- Requirements
  - Understanding of the used computational architecture
  - Identifying the bottlenecks of the resources
  - Optimization of the algorithms w.r.t. the bottlenecks
- Typical computational resources for HPC:
  - Resources which are close to each other
    → HPC-Cluster (e.g. FLOW, HERO)
  - Distributed resources (e.g. Laptops, PC's)
    → Grid-Computing

Parallelization

- Why parallelization?
  - Enabler to solve computational intensive problems (e.g. large linear equation systems)
    - problems with high memory consumption
    - time consuming calculations

- Simple case
  - Computation of small independent problems for a huge parameter space (low memory/time consumption per parameter set)
    - Processes are independent
      → can be executed in parallel by starting them N_{proc} –times
      → e.g. 1 process per CPU core
    - Reduce simulation wall clock times
      → at best by factor 1/ N_{proc}

Parallelization

- Complex case
  - Large problem which needs parallelization, e.g. large linear equation systems (high memory/time consumption for each computation)
    - N_{proc} processes work together
      → e.g. 1 process per CPU core
    - Decrease memory usage per core
      → at best by factor 1/ N_{proc}
  - Reduce simulation wall clock times
    → at best by factor 1/ N_{proc}
  - Overhead
    - exchange/management of data
    - organization of processes
    - load imbalances

Introduction to HPC

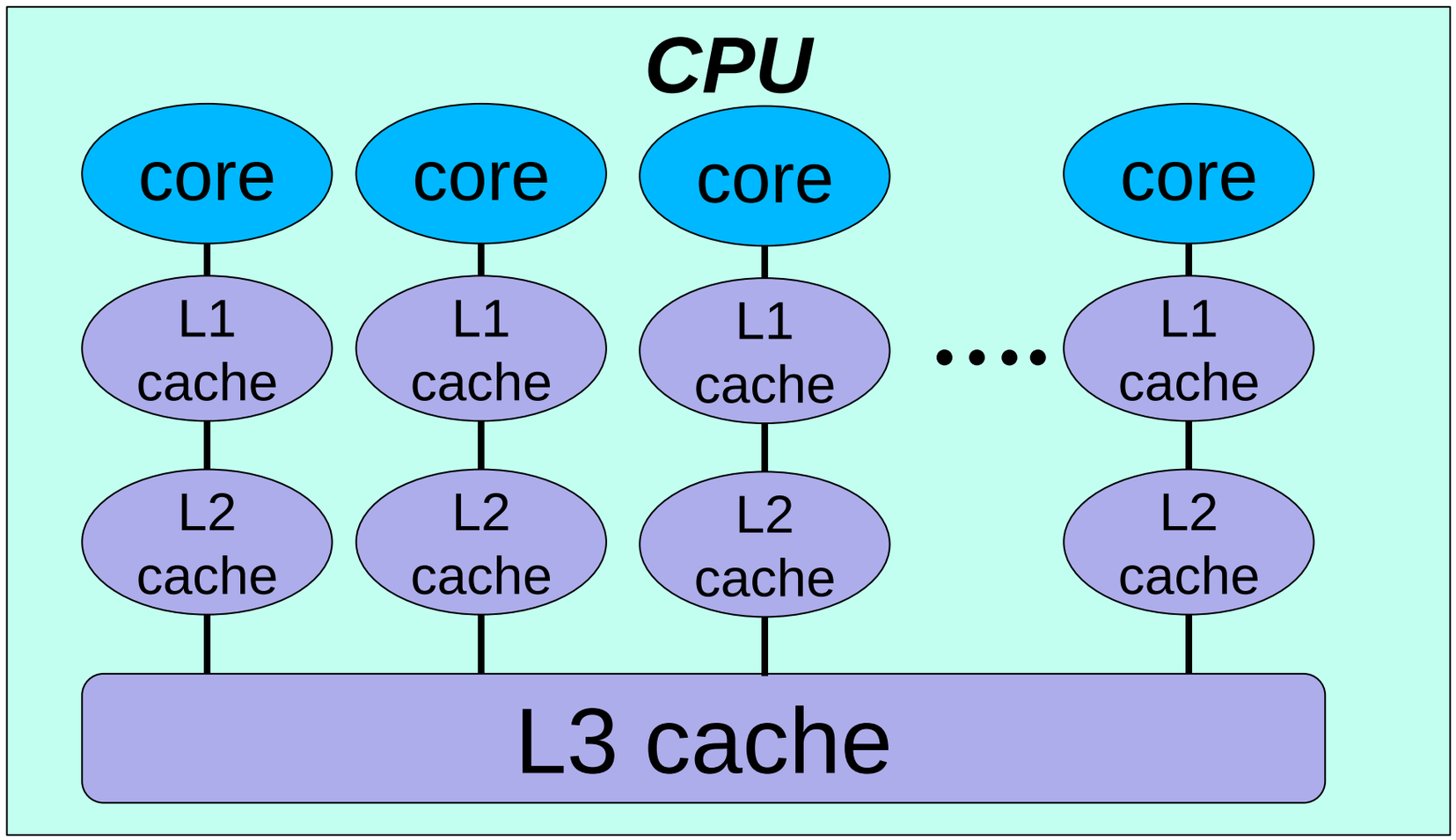
ARCHITECTURES

Architectures

- Why is knowledge about hardware architecture important?
  - To write efficient code:
    - Algorithm should fit to the architecture
      → Increase performance
  - To know the limits/bottlenecks

Central Processing Unit - CPU

- CPU (chip on a socket) contains several cores
- Cores connected to caches for fast memory access, low latency
  → O(10) faster than direct memory access
- Cache coherence



Westmere X5650

6 cores, 2,66GHz

64Kb

256Kb

12Mb

Central Processing Unit - CPU

Todays CPU (chip on a socket)

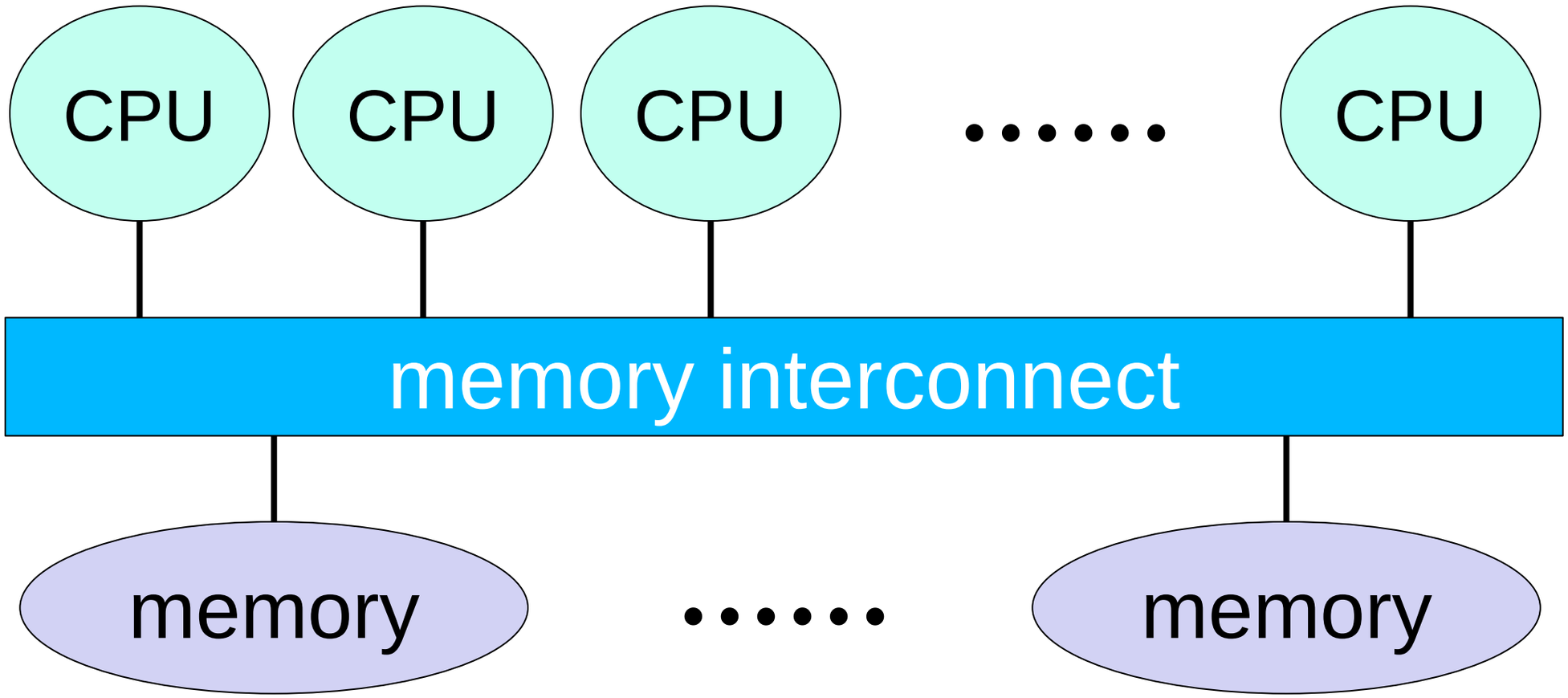
- contains several cores
  → number of cores increases, todays driver for performance increase
- several levels of caches (high transfer rates, low latency)
- execute instructions in parallel (pipelines)
- *Single Instruction, Multiple Data* (SIMD) features, e.g. SSE, AVX, AVX2,... instruction sets
  → multiple floating point/integer operations within on cycle
- clock frequency stagnates/decrease due to
  - energy efficiency
  - thermal effects (heating, leakage current,...)

Main limitations

- memory bandwidth (e.g. floating point operations faster then getting the number out of memory)
- limit in instruction per cycle (to many work)
- parallelization of algorithms

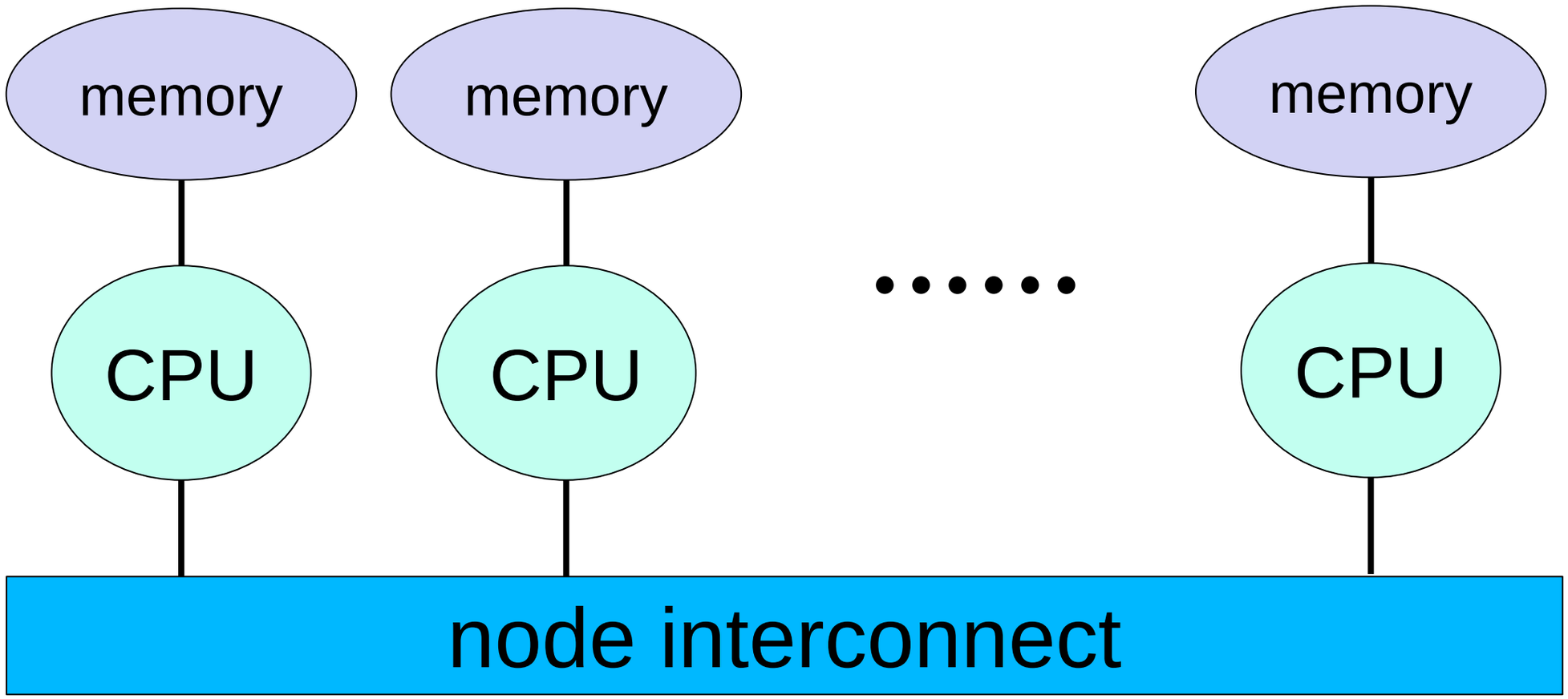
Shared memory system

- Symmetric multi-processing (smp)
- Uniform memory access (uma)
  - Same access time from every CPU



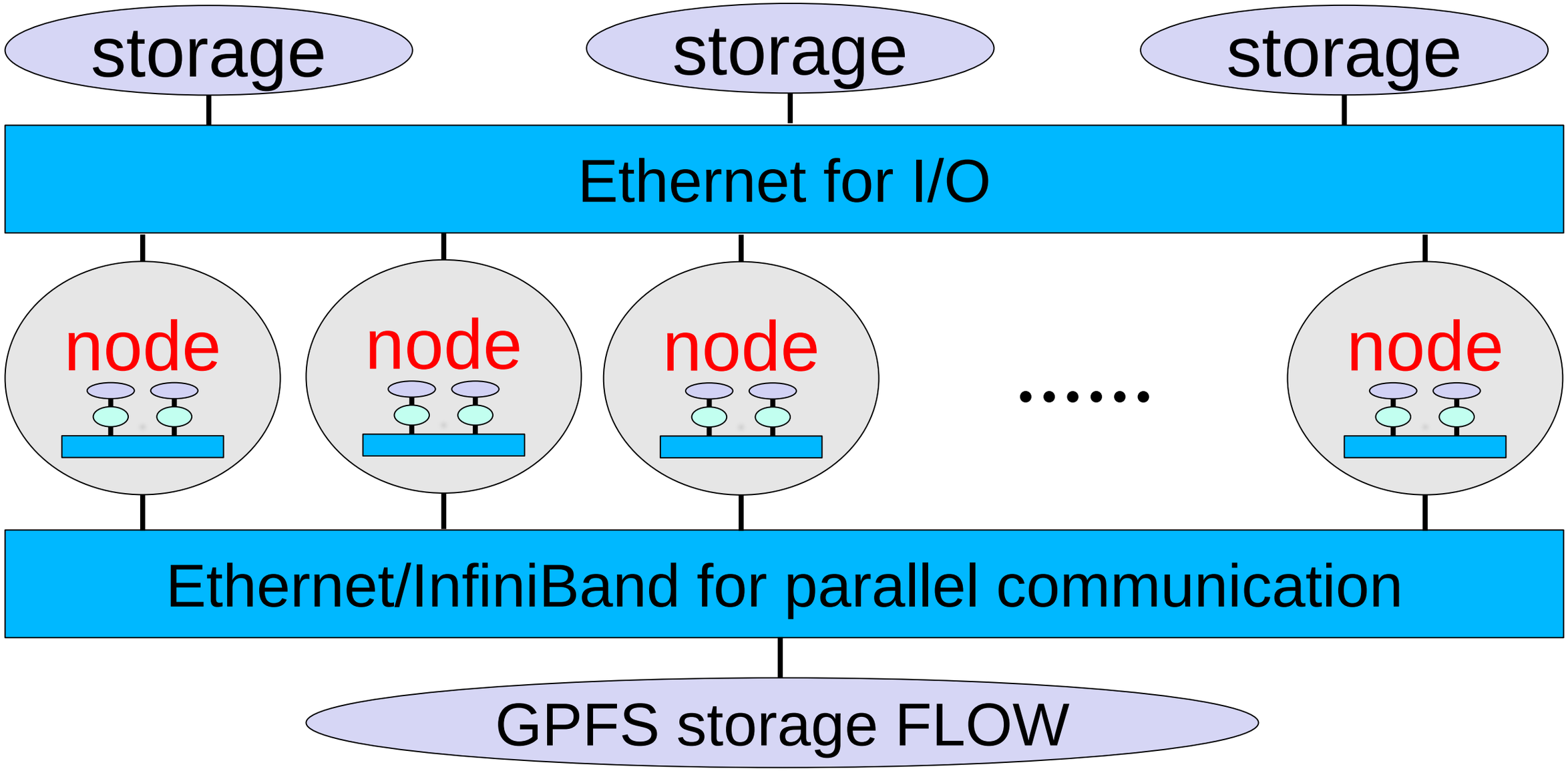
Shared memory system

- Today architecture of a node: Non-uniform memory access (NUMA)
  - Fast access to own memory, slow access to other memory
  - Cache coherence → ccNUMA
- Nodes on HERO and FLOW have ccNUMA architecture (except UV100)



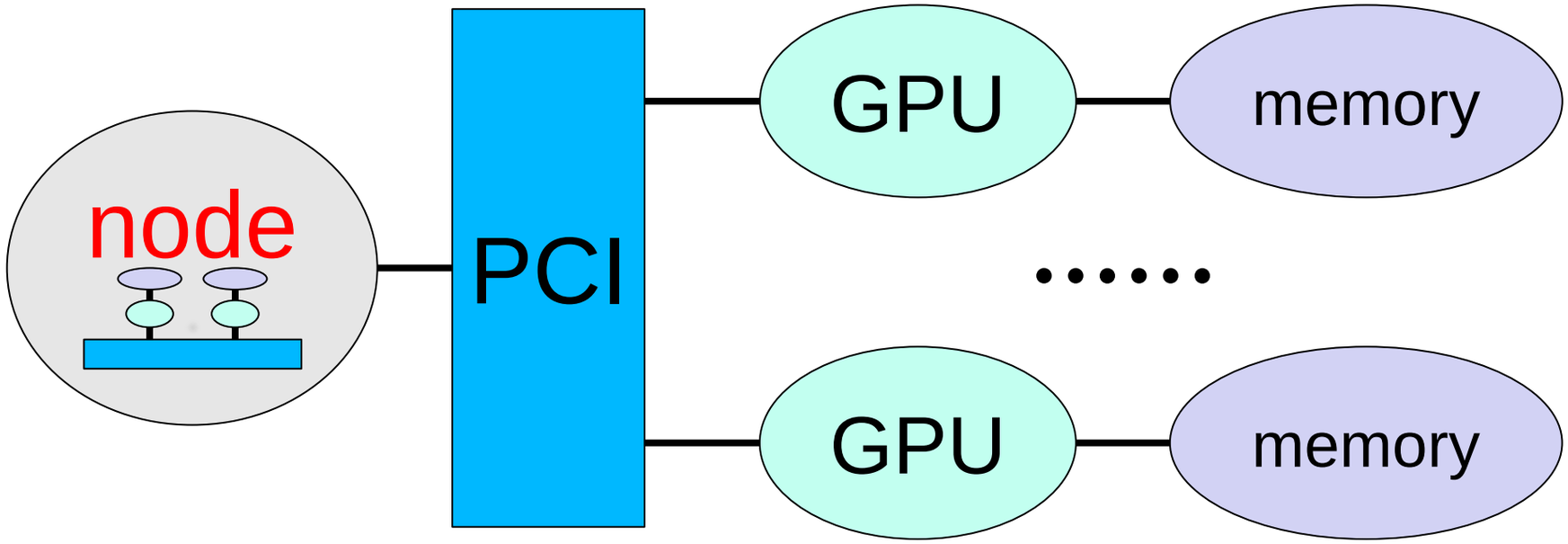
HPC-Cluster FLOW/HERO

- O(100) NUMA nodes connected by fast interconnect



Accelerators (many cores)

- Todays typical accelerators
  - General-Purpose computing on Graphics Processing Units (GPGPU), e.g. Graphic card chips with O(1000) cores
  - Intel Phi (~60 Pentium cores)



- Other
  - Field-Programmable Gate Arrays (FPGA)
  - ...

ForWind
Center for Wind Energy Research

Introduction to HPC

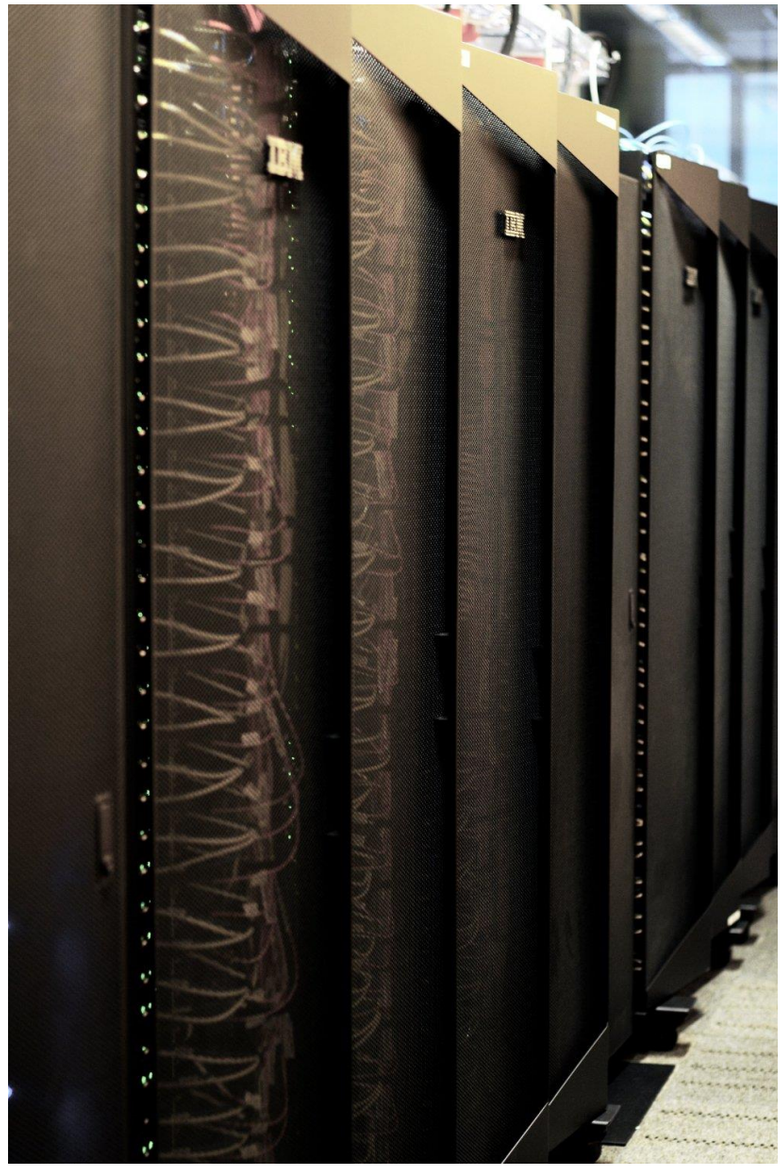
HPC AT THE UNIVERSITY OF OLDENBURG

Leibniz Universität Hannover
CARL VON OSSIETZKY universität OLDENBURG
Universität Bremen

ForWind
Center for Wind Energy Research

FLOW –
Facility for Large-Scale Computations in Wind Energy Research

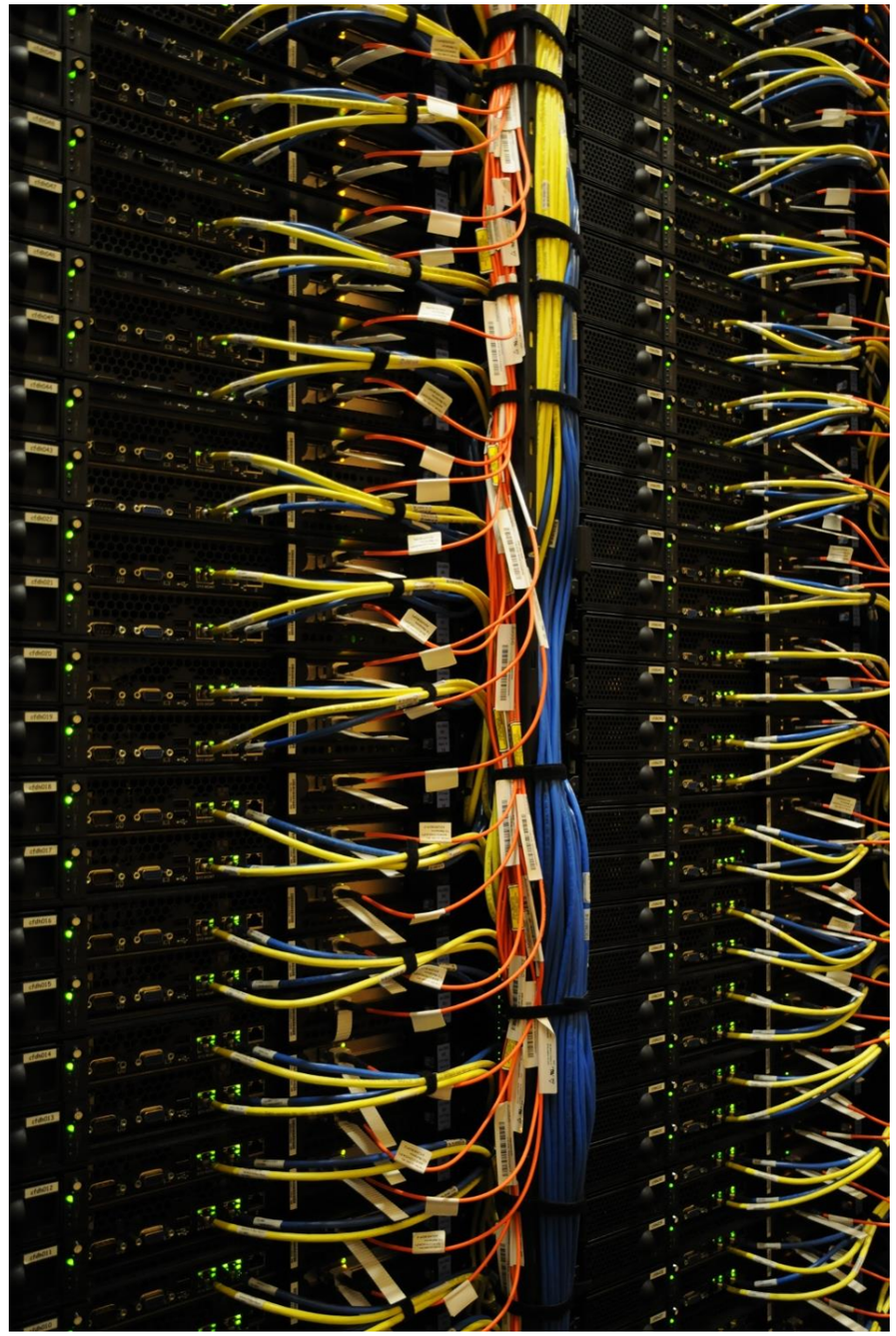
- 122 "low-memory" compute nodes: 2x6 cores per node, 24Gb, diskless (host names cfdl001..cfdl122)
- 64 "high-memory" compute nodes: 2x6 cores per node, 48Gb, diskless (host names cfdh001..cfdh064)
- 7 compute nodes: 2x4 cores per node, 32Gb (host names cfdx001..cfdx007)
- QDR InfiniBand interconnect (fully non-blocking)
- Gigabit Ethernet for File-I/O etc.
- High-performance IBM GPFS storage system, 130TB connected by InfiniBand
- 160 TB NAS storage shared with HERO
- Theoretical peak performance: 24 TFlop/s (Flop/s – Floating Point Operations per second)



Leibniz Universität Hannover
CARL VON OSSIETZKY universität OLDENBURG
Universität Bremen

HERO - High-End Computing Resource Oldenburg

- 130 "standard" nodes: 2x6 cores, 24 GB, 1 TB disk (host names mpcs001..mpcs130)
- 20 "big" nodes: 2x6 cores, 48 GB, RAID 8 x 300 GB SAS (host names mpcb001..mpcb020)
- Gigabit Ethernet II for communication of parallel jobs
- Second, independent Gigabit Ethernet for File-I/O
- SGI Altix UV 100 shared-memory system: 10x6 cores, 640 GB, RAID 20 x 600 GB SAS (host uv100)
- 160 TB NAS storage shared with FLOW
- Theoretical peak performance: 19.2 TFlop/s (Flop/s – Floating Point Operations per second)



ForWind
Center for Wind Energy Research

HPC CLUSTER USAGE

Login

- Login nodes of FLOW and HERO
  - `flow.hpc.uni-oldenburg.de` (to access `flow01` or `flow02`)
  - `hero.hpc.uni-oldenburg.de` (to access `hero01` or `hero02`)
- Login from terminal by ssh, e.g.
  ```
  ssh -XY abcd1234@flow.hpc.uni-oldenburg.de
  ```
- Terminal programs
  - Windows
    - Putty (`http://www.putty.org/` , no X-Window support)
    - MobaXterm (`http://mobaxterm.mobatek.net/`, including X-Window support)
  - Linux/Mac: Terminal included
- Copy of data sftp/scp, e.g. from your host
  ```
  scp FILENAME abcd1234@flow.hpc.uni-oldenburg.de:TARGET_DIR
  ```
- Note: Direct login on computational nodes is forbidden
- To login from outside or University WLAN you need a special VPN connection (see HPC-WIKI)

File systems

- two directories for storage
  - home directory `/user/<user_group>/<user_name>`
    - place for long term storage
    - fully backed up
    - limited size (usually 110 Gb per user)
    - usage can be determined by the command iquota
  - working directory `/data/work/<user_group>/<user_name>`
    - place for data used or produced during the computations
    - not backed up
    - huge (limited) size (usually 3.5 Tb per user)
    - size of usage weekly per mail
- both no parallel file systems (temporal delays possible, max. 30s files, max. 60s directories)
- both mountable via CIFS on your workstation (HPC-Wiki)

File systems

Home directory has *snapshot* functionality

- hidden directory `.snapshot` exists (not visible with `ls`, but `cd .snapshot` possible)
- within that directory a list of different snapshots exist

```
hpc_user_daily_2013-06-28_01-00
hpc_user_daily_2013-06-29_01-00
...
hpc_user_daily_recent
hpc_user_hourly_2013-07-03_20-00
...
hpc_user_hourly_recent
hpc_user_weekly_2013-06-10_02-00
...
hpc_user_weekly_recent
```

- Within these subdirectories the file-structure of the current parent directory was stored (at that point in time where the particular snapshot was taken at)

File systems

Additional high performance GPFS file system on FLOW

- `/data/work/gpfs/<user_group>/<user_name>`
- place for data used or produced during the computations
- high transfer rates (about 4 times higher than the other file system)
- parallel file system
- not backed up
- huge size, up to now no quota (110Tb)
- only mountable via sshfs (SFTP) on your workstation (see HPC Wiki)

User environment

- Most of the programs (e.g. compilers, MPI,...) not available per default
- Use command `module` to load the needed environment
  - Show available modules
    ```
    module av [NAME]
    ```
    (e.g. `module av intel` to see all Intel products)
  - Module name convention: lower case letters
  - More information about a module
    ```
    module help MODULE_NAME
    module show MODULE_NAME
    ```
- Load module (make Software available)
  ```
  module load MODULE_NAME
  ```
  (e.g. `module load ics/2013_sp1.3.174/64` to load Intel Cluster Studio 2013 SP1
- Unload module (disable Software)
  ```
  module unload MODULE_NAME
  ```

Available development tools

- Compiler (C, C++,Fortran)
  - Intel Cluster Studio
  - GNU compiler
  - PGI Accelerator Suite
  - Open64
  - CLang
- MPI implementation
  - OpenMPI
  - Intel MPI
- Libraries
  - BLAS/LAPACK in MKL
  - LEDA
  - FFTW
  - NetCDF, HDF5
  - NAG
  - ...
- Other languages
  - Python
  - R
  - Matlab, Octave

and many more....

Thanks!

Please visit our HPC-Wiki

http://wiki.hpcuser.uni-oldenburg.de