# Introduction to high-performance computing

Frédéric Wautelet

CÉCI HPC training 2024

# Who am I

```
$ pinky -l fwautele
Login name: fwautele                    In real life:  Frederic Wautelet
(frederic.wautelet@unamur.be)
Directory: /home/fwautele               Shell:  /bin/bash
```

- Background: Master of CS (UNamur 2002)
- HPC tech lead since 2005
- Based in the Faculty of Sciences

@HPC_UNamur

# Outline

- How to use HPC infrastructure
- How to program on HPC cluster
- Going Parallel
- Data Management

# Outline

## How to use HPC infrastructure

I. Intro, Access, Command line

II. Choosing software, editing files, writing scripts

III. Code and Data Versioning

IV. Submitting jobs, using containers, checkpointing, workflows

# Outline

How to use HPC infrastructure

I. **Intro**, Access, Command line

II. Choosing software, editing files, writing scripts

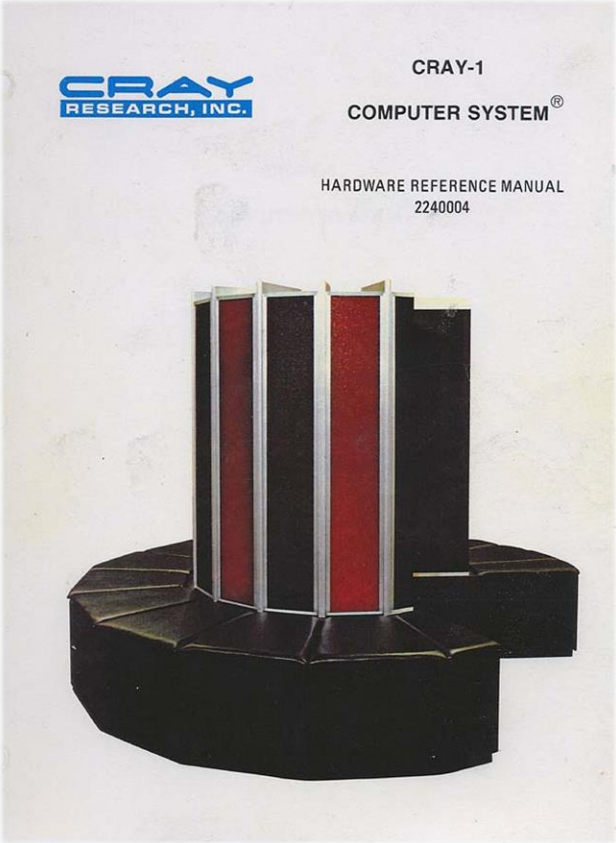
III. Code and Data Versioning

IV. Submitting jobs, using containers, checkpointing, workflows

# High Performance Computing



- High-performance computing (HPC) uses supercomputers and computer clusters to solve advanced computation problems.




**Cray-1a (1977)**
250 MFlops

# Cluster

- A computer cluster is a group of linked computers, working together closely so that in many respects they form a single computer.

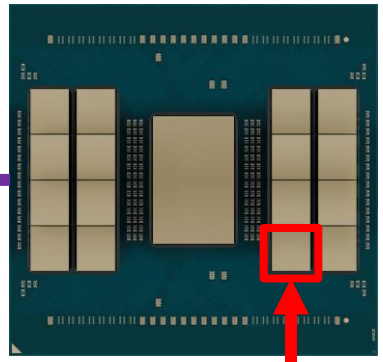
**Fugaku - RIKEN Center, Japan**
440 PFlops

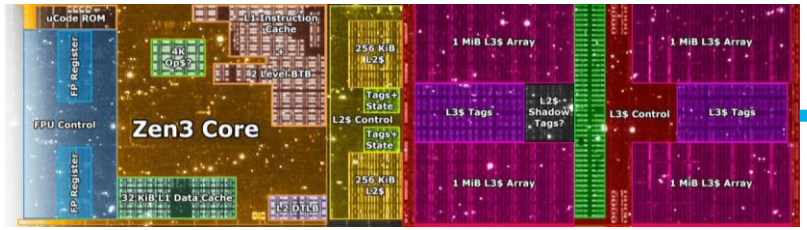
# Nodes and Cores

How to use HPC infrastructure

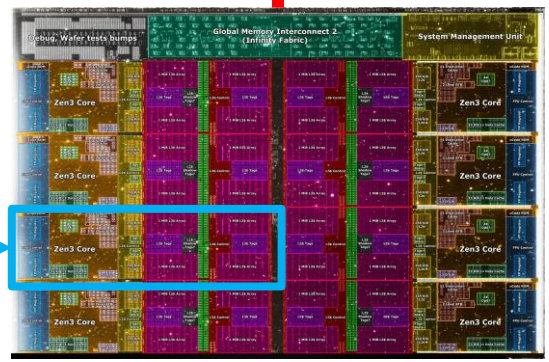
Node

Chip






Core




Chiplet

# Measure supercomputer power

- FLOPS
- **fl**oating-point **o**perations **p**er **s**econd

GigaFLOPS = one billion ($10^9$) floating-point operations per second

TeraFLOPS = one trillion ($10^{12}$) floating-point operations per second

PetaFLOPS = one quadrillion ($10^{15}$) floating-point operations per second

ExaFLOPS = one quintillion ($10^{18}$) floating-point operations per second

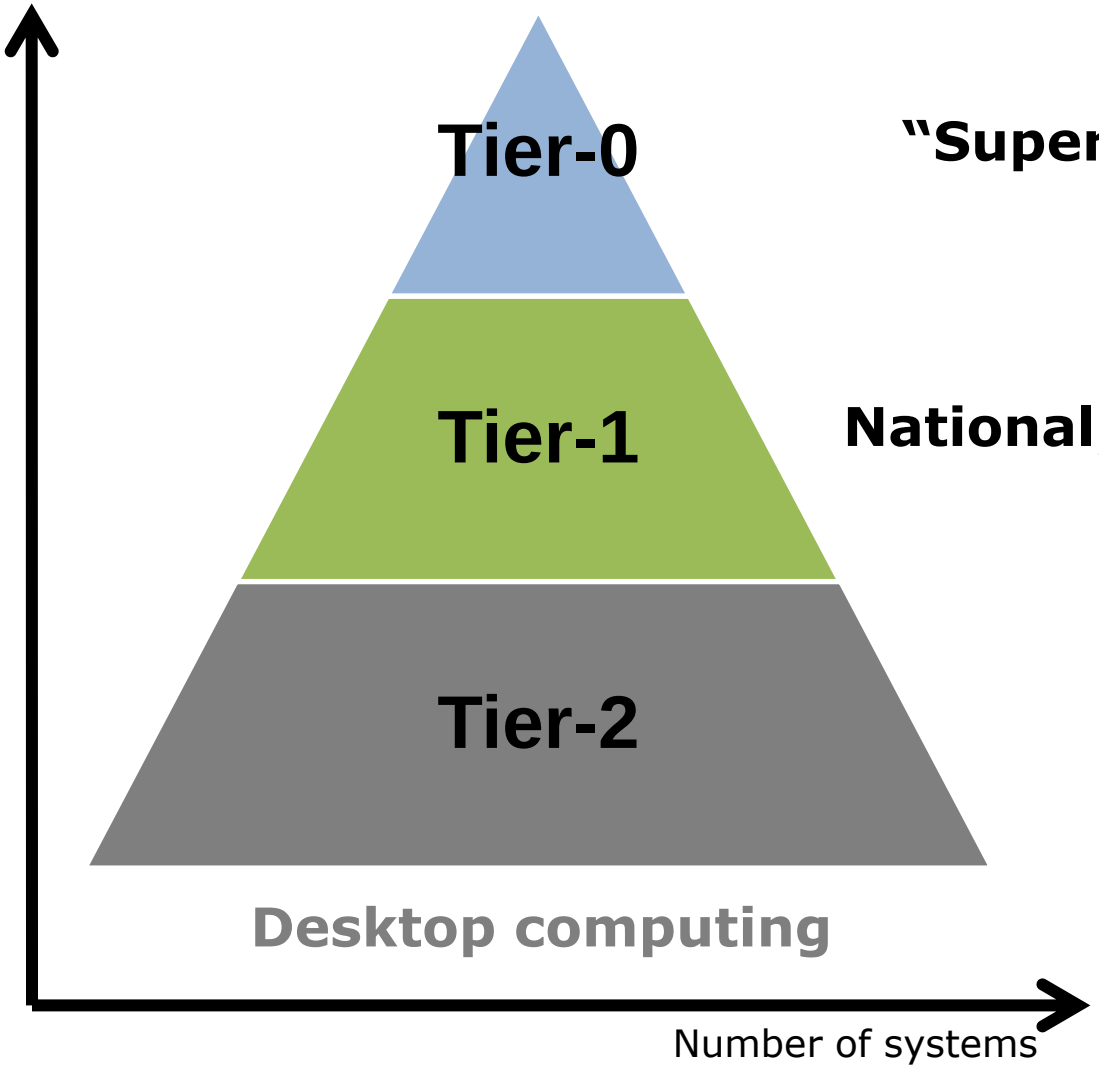
# TOP500



- Frontier
  - Oak Ridge National Laboratory, Tennessee, USA
  - TOP500 #1 (June 2024)
  - 8,700,000 cores
  - First Exascale supercomputer: 1.2 Eflop/s
  - 23 MW
  - $600M+ budget

# HPC ecosystem

How to use HPC infrastructure



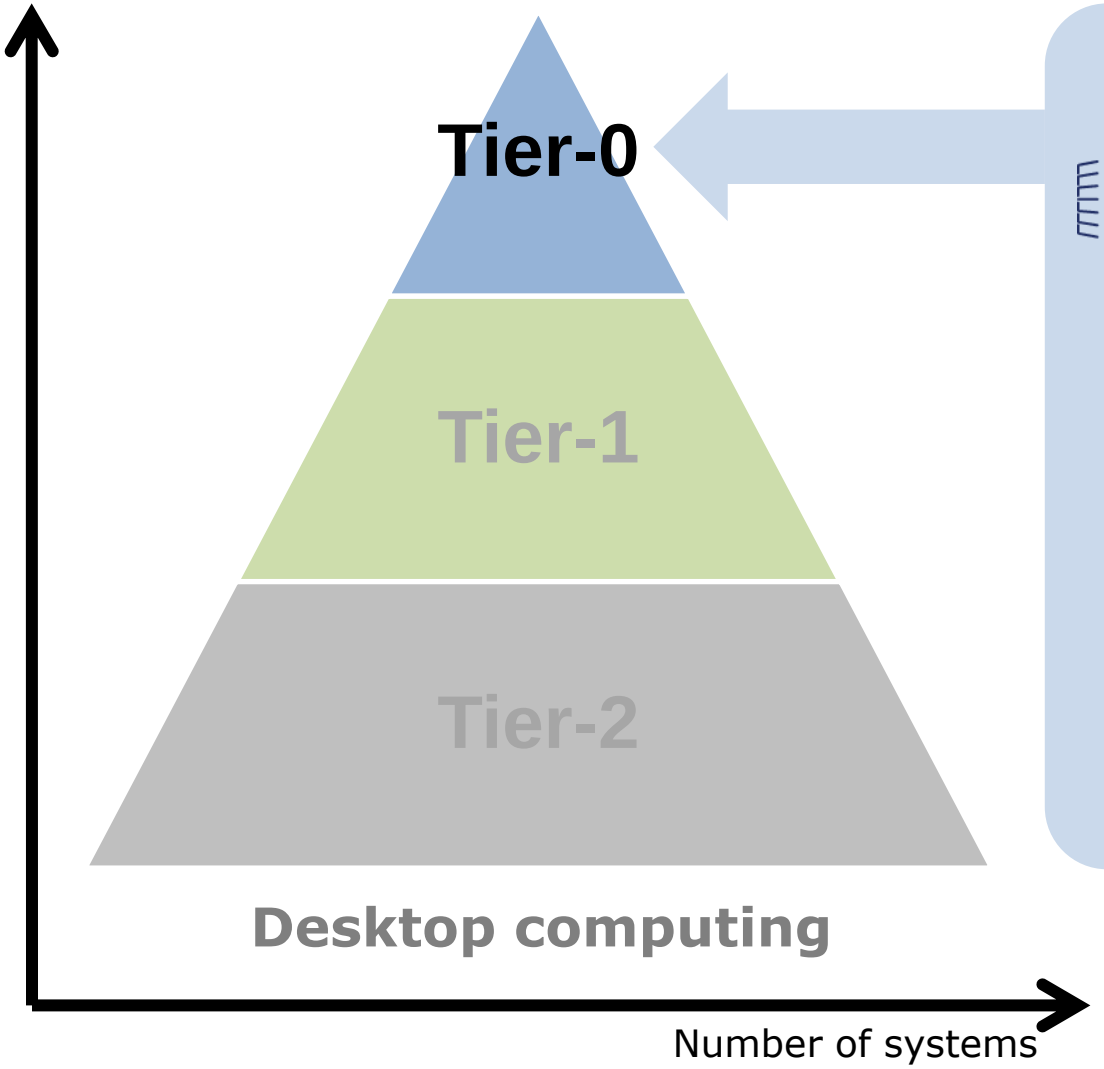

**"Super-supercomputer"**

**National/regional centers**

**Local centers**

# HPC ecosystem

How to use HPC infrastructure




EuroHPC Joint Undertaking

PRACE

# Tier-0: LUMI



2,000,000+ cores

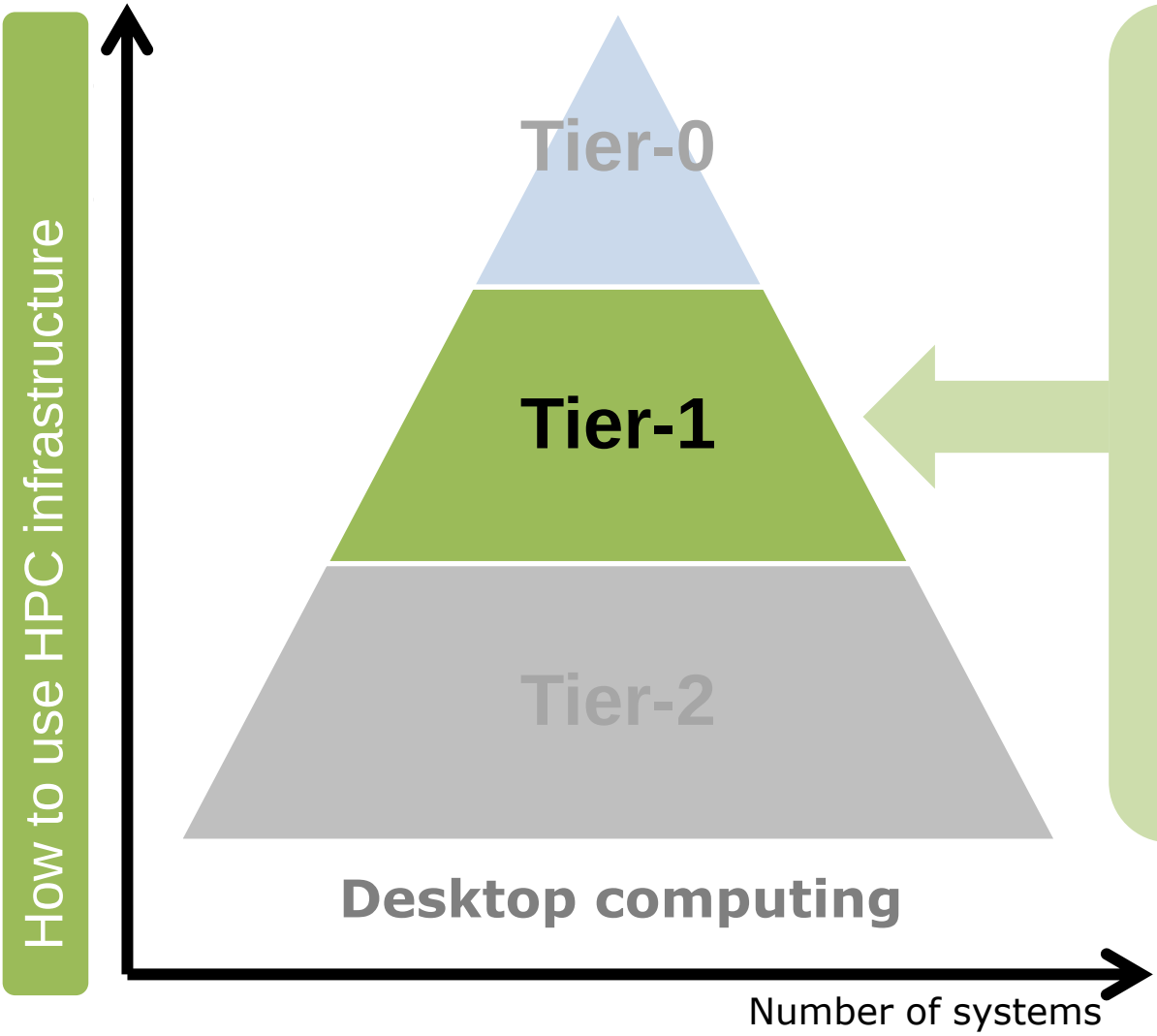
up to 4 TiB memory

20,400+ GPU

110+ PiB storage


Access through call
Contact CÉCI support

# L'écosystème du calcul intensif




Cenaero

Access through CÉCI

# Tier-1 : Lucia

Cenaero

How to use HPC infrastructure

38,000+ cores

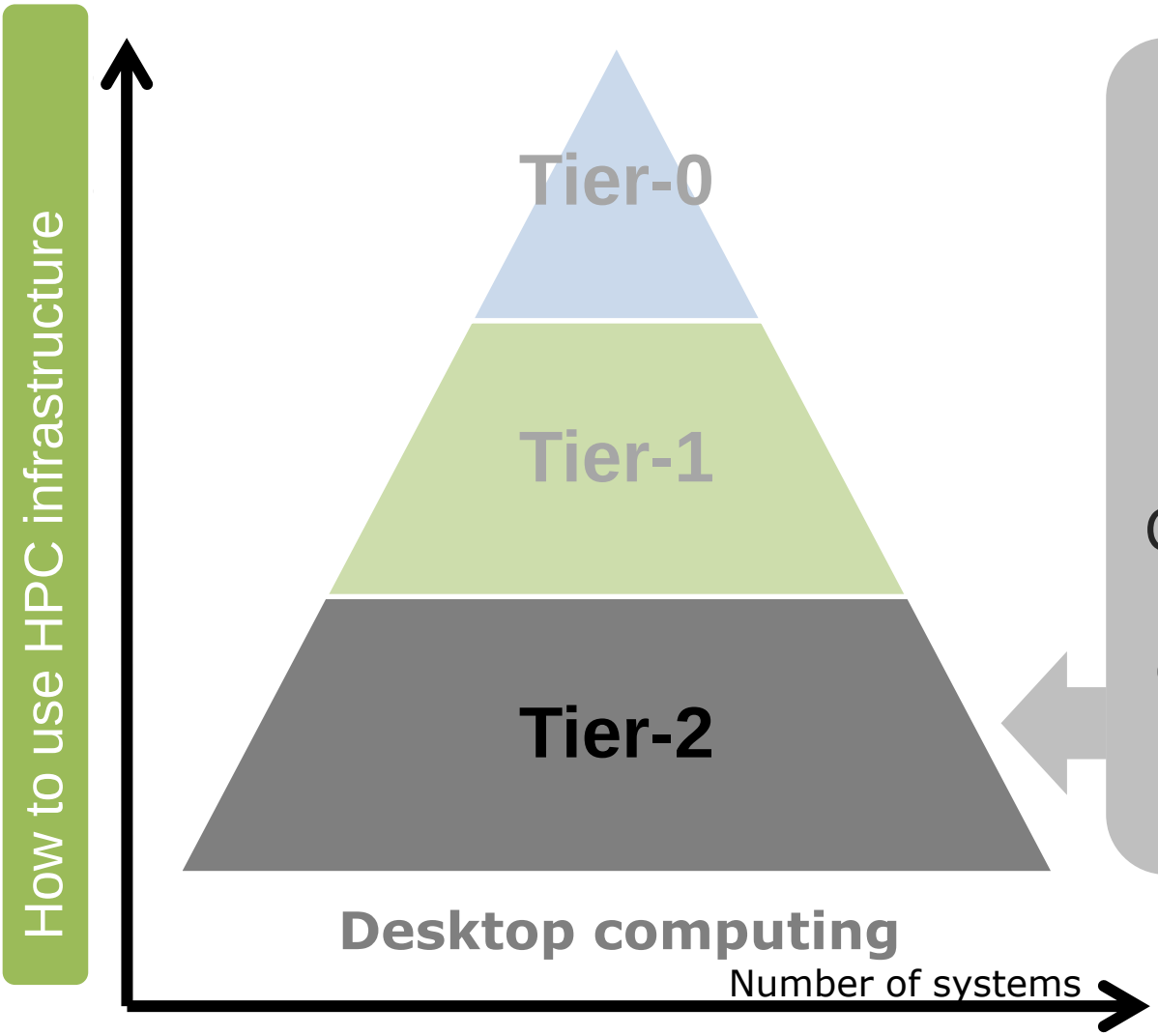
up to 4 TiB memory

200+ GPU

7+ PiB storage

Access through CÉCI
https://www.ceci-hpc.be/projetstier1.html

# L'écosystème du calcul intensif




C.E.C.I

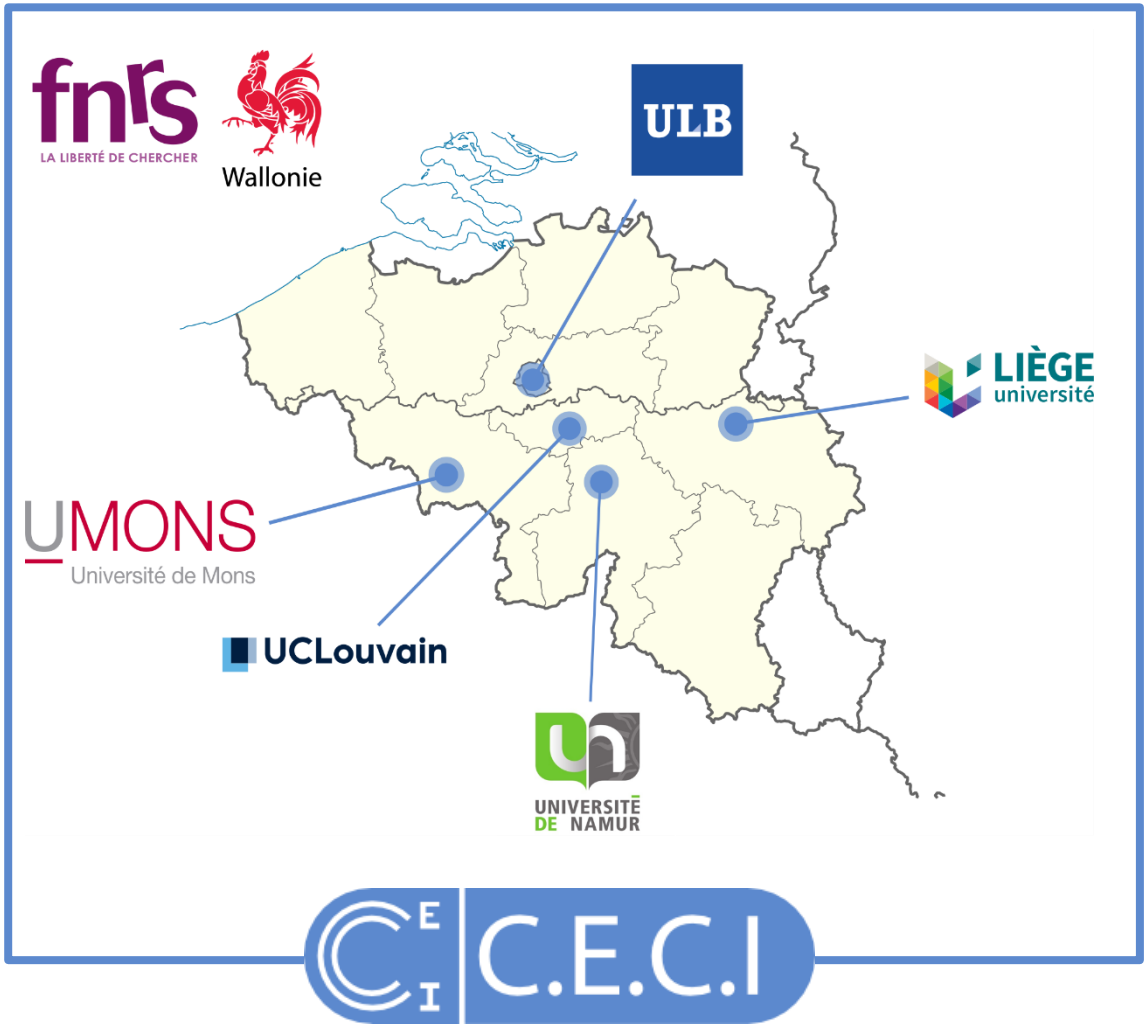
Interuniversity

Training
Common know-how

Optimization of the use of the resources

# Consortium des Équipements de Calcul Intensif





C.E.C.I

UMONS

UNIVERSITÉ DE NAMUR

LIÈGE université

UCLouvain Université

ULB

**Dragon 2**
**592 cores**

384 GiB RAM

4 GPU

100 TiB storage

**Q1 2019**

**Hercules 2**
**1152 cores**

2 TiB RAM

16 GPU

60 TiB storage

**Q3 2019**

**NIC5**
**4672 cores**

1 TiB RAM

250 TiB storage

**Q4 2020**

**Lemaitre 4**
**5120 cores**

768 GiB RAM

144 TiB storage

**Q1 2021**

**Lyra**
**1472 cores**

384 GiB RAM

46 GPU

3+ PiB storage

**Q4 2024**

**13008 cores total – 66 GPU**

C.E.C.I

LIÈGE université | UCLouvain Université | ULB | UNIVERSITÉ DE NAMUR | UMONS

**NIC5**
**4672 cores**

1 TiB RAM

250 TiB storage

**Q4 2020**

**Lemaitre 4**
**5120 cores**

768 GiB RAM

144 TiB storage

**Q1 2021**

**Lyra**
**1472 cores**

384 GiB RAM

46 GPU

3+ PiB storage

**Q4 2024**


**Hercules 3**
**2992 cores**

3 TiB RAM

16 GPU

256 TiB storage

**Q1 2025**


**Dragon 3**
**1000+ cores**

512 GiB RAM

100+ TiB storage

**Q3 2025**

**15000+ cores total – 62 GPU**

# Users

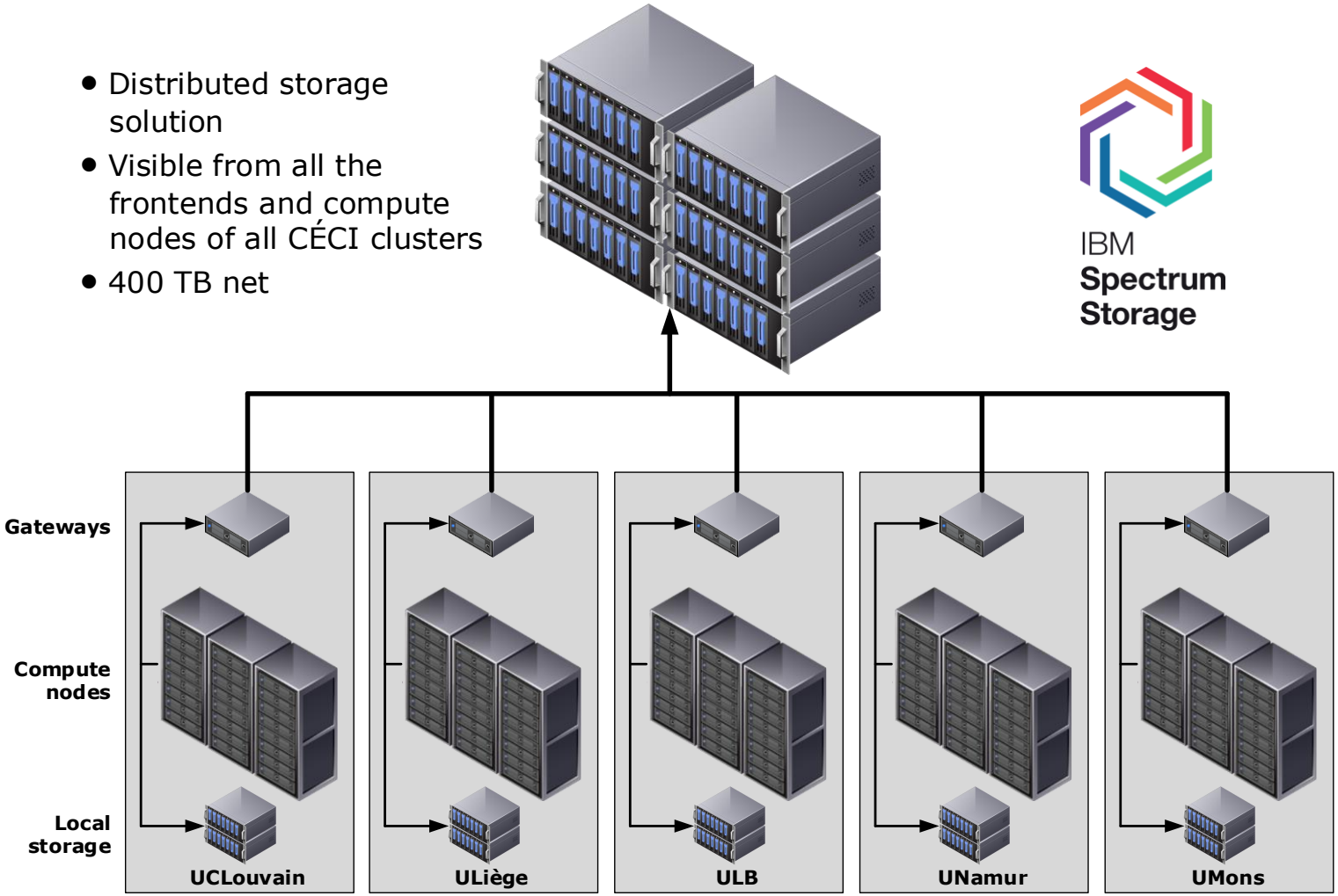
How to use HPC infrastructure

- 400+ CÉCI actives users

machine learning
sat/smt solving
astrophysics
physics
climatology
oceanography
particle physics
image processing
fluid mechanics
computational chemistry
bioinformatics
materials science
neurosciences
solid state physics
nuclear physics
electrical machines
number theory
electromechanical design
chemistry
security
mathematics

# CÉCI distributed storage

- Distributed storage solution
- Visible from all the frontends and compute nodes of all CÉCI clusters
- 400 TB net



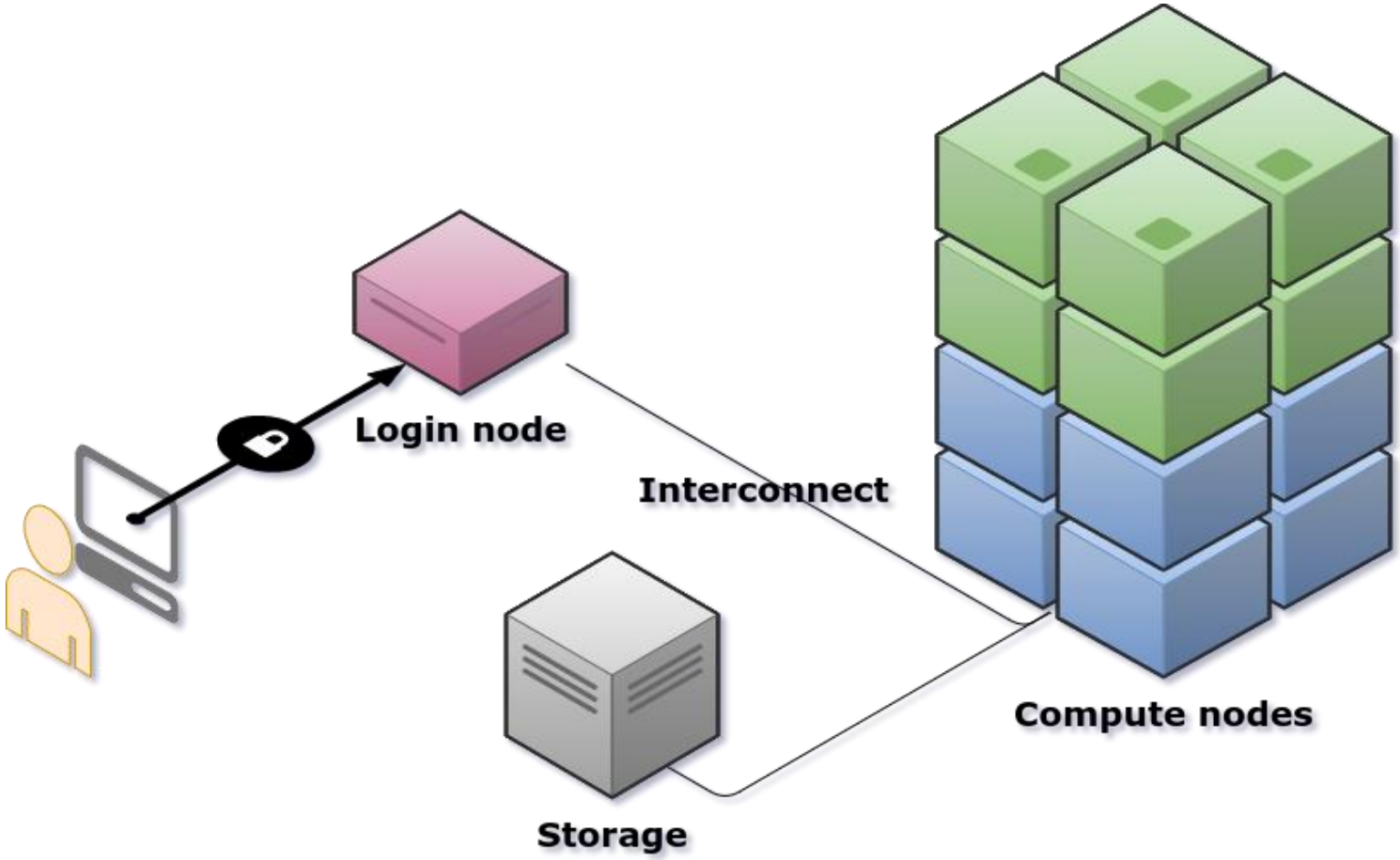
# CÉCI distributed storage



- Common storage directories for all CÉCI clusters
- No need to transfer data between clusters with scp
- Common software repository
- Almost all software installed on any cluster are available on all clusters

# A cluster in a nutshell



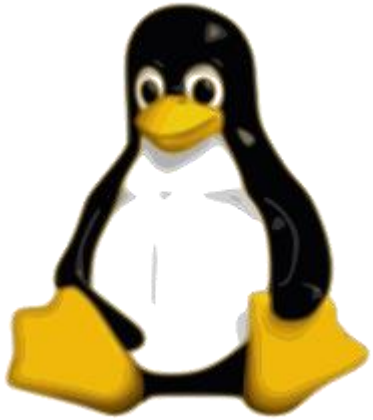


# Login node

- Submit jobs to batch system
- Manage your files
- Interactive work at small scale
- CÉCI login nodes
  - hercules2.ptci.unamur.be
  - dragon2.umons.ac.be
  - lemaitre3.cism.ucl.ac.be
  - nic5segi.ulg.ac.be

# Operating system





- All CÉCI cluster are running GNU/Linux

Rocky Linux

- Linux Rocky 8

# Clusters at CÉCI

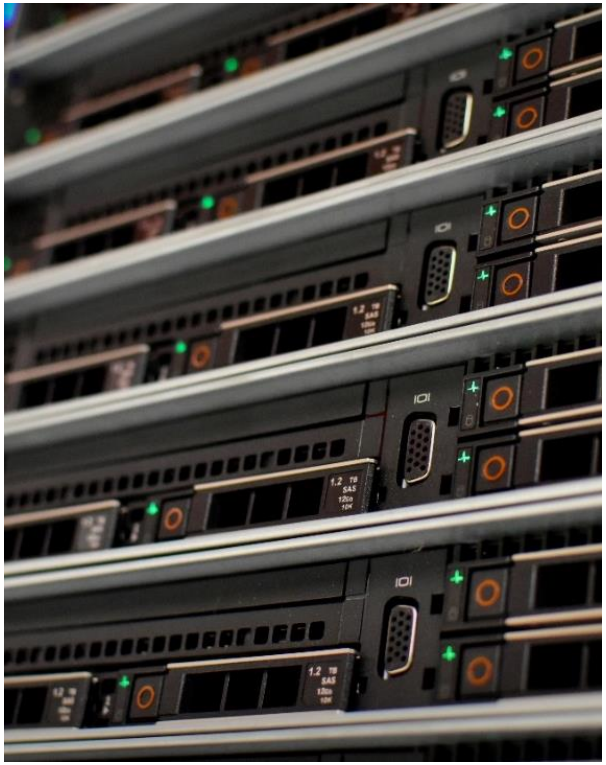
The aim of the Consortium is to provide researchers with access to powerful computing equipment (clusters). Clusters are installed and managed locally at the different sites of t
universities taking part in the Consortium, but they are accessible by all researchers from the member universities. A single login/passphrase is used to access all clusters through SS

All of them run Linux, and use Slurm as the job manager. Basic parallel computing libraries (OpenMP, MPI, etc) are installed, as well as the optimized computing subroutines (e
BLAS, LAPACK, etc.). Common interpreters such as R, Octave, Python, etc. are also installed. See each cluster's FAQ for more details.

| Cluster | Host | CPU type | CPU count* | RAM/node | Network | Filesystem** | Accelerator | Max time | Preferred jobs*** |
|---|---|---|---|---|---|---|---|---|---|
| **Lemaitre4** | UCLouvain | Genoa 2.4 GHz | 5120 (40 x 128) | 766GB | HDR Ib | BeeGFS 320 TB | None | 2 days | MPI |
| **NIC5** | ULiège | Rome 2.9 GHz | 4672 (73 x 64) | 256 GB..1 TB | HDR Ib | BeeGFS 520 TB | None | 2 days | MPI |
| **Hercules2** | UNamur | Naples 2 GHz<br>SandyBridge 2.20 GHz | 1024 (30 x 32 + 2 x 64)<br>512 (32 x 16) | 64 GB..2 TB | 10 GbE | NFS 20 TB | None | 15 days | serial / SMP |
| **Dragon2** | UMons | SkyLake 2.60 GHz | 592 (17 x 32 + 2 x 24) | 192..384 GB | 10 GbE | RAID0 3.3 TB | 4x Volta V100 | 21 days | serial / SMP |
| **Lemaitre3** | UCL | SkyLake 2.3 GHz<br>Haswell 2.6 GHz | 1872 (78 x 24)<br>112 (4 x 28) | 95 GB<br>64 GB | Omnipath | BeeGFS 440 TB | None | 2 days<br>6 hours | MPI |
| **Dragon1** | UMons | SandyBridge 2.60 GHz | 416 (26 x 16)<br>32 (2x16) | 128 GB | GbE | RAID0 1.1 TB | 4x Tesla C2075,<br>4x Tesla Kepler K20m | 41 days | serial / SMP |
| **NIC4*** | ULiège | SandyBridge 2.0 GHz<br>IvyBridge 2.0 GHz | 2048 (120 x 16 + 8 x 16) | 64 GB | QDR Ib | FHGFS 144 TB | None | 3 days | MPI |
| **Vega*** | ULB | Bulldozer 2.1 GHz | 896 (14 x 64) | 256 GB | QDR Ib | GPFS 70 TB | None | 14 days | serial / SMP / MPI |
| **Hercules*** | UNamur | SandyBridge 2.20 GHz | 512 (32 x 16) | 64..128 GB | GbE | NFS 20 TB | None | 63 days | serial / SMP |
| **Lemaitre2*** | UCL | Westmere 2.53 GHz | 1380 (115 x 12) | 48 GB | QDR Ib | Lustre 120 TB | 3x Quadro Q4000 | 3 days | MPI |

# Hercules 2

- 1,408 cores
- up to 2 TiB memory
- 16 GPU
- 60 TiB storage



©P. Beaujean

# HPC @ UNamur

UNIVERSITÉ DE NAMUR

- Local support :
  - Plateforme Technologique en Calcul Intensif (PTCI)
    - Juan CABRERA
    - Frédéric WAUTELET
  - ptci-support@unamur.be

- Other HPC resources
  - Hyades 2
    - 288 cores total
    - Up to 92 GB RAM per node

# Dragon 2 @ UMons

UMONS Université de Mons

How to use HPC infrastructure

- High performance SMP nodes
- Long duration job
  - 21 days
- GPU
  - 4x NVIDIA Volta V100
- No multi-node jobs

# HPC @ UMons

UMONS Université de Mons

How to use HPC infrastructure

- Local support
  - Sebastien.KOZLOWSKYJ@umons.ac.be
- Other HPC resources
  - Biovia Materials Studio cluster
    - 144 cores total
    - 192 GB RAM per node
  - HTC cluster
    - 512 cores total
    - Up to 256 GB RAM per node

# Lemaitre 4 @ UCLouvain

UCLouvain Université

How to use HPC infrastructure

- Massively parallel jobs
  - MPI
- I/O intensive jobs
- Short duration job
  - 2 days
- Fast parallel filesystem
  - $GLOBALSCRATCH

# HPC @ UCLouvain

UCLouvain Université

How to use HPC infrastructure

CISM

- Local support
  - Institut de Calcul Intensif et de Stockage de Masse (egs-cism@listes.uclouvain.be)

**Thomas Keutgen (Head)**

**Damien François**

**Olivier Mattelaer**

**Bernard Van Renterghem**

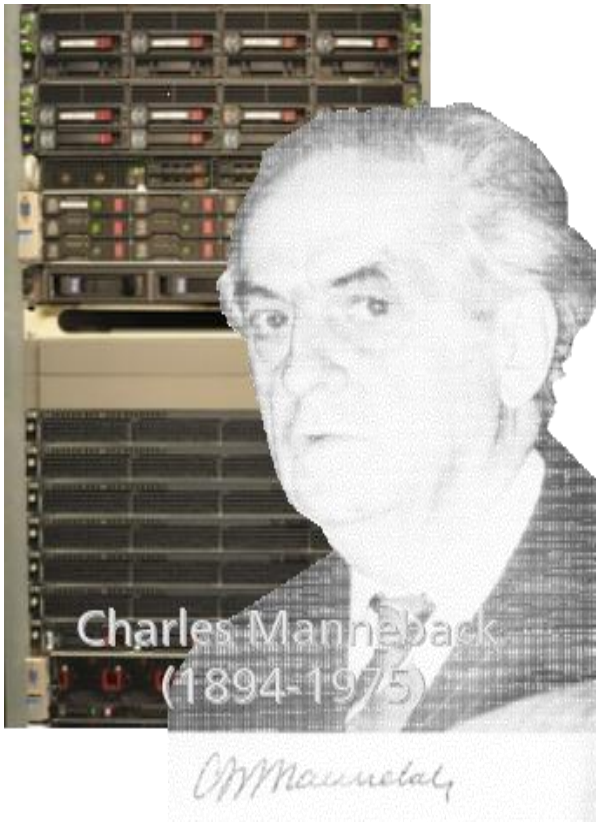
**Patrick Vranckx**

# HPC @ UCLouvain

UCLouvain
Université



- Other resources
  - Manneback HPC cluster
    - Heterogeneous hardware
    - +5700 cores
    - 82 Tflop/s
  - Mass storage
    - 317 TB storage total



# NIC5 @ ULiège



- Massively parallel jobs
  - MPI
- I/O intensive jobs
- Short duration jobs
  - 2 days
- Fast parallel filesystem
  - $GLOBALSCRATCH

# HPC @ ULiège

How to use HPC infrastructure

- Local support
  - David.Colignon@uliege.be
- More info
  - http://www.ulg.ac.be/nic4

# Lyra @ ULB

UNIVERSITÉ LIBRE DE BRUXELLES

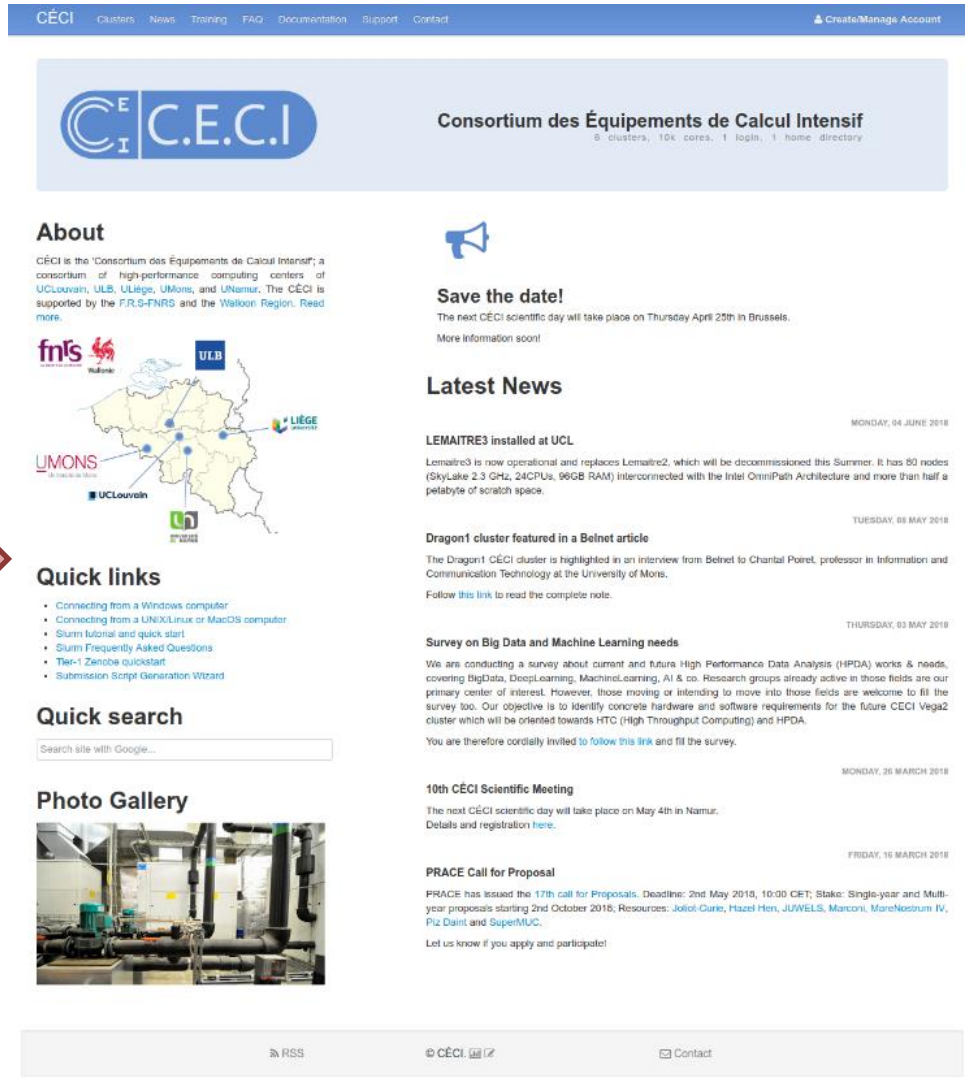


- For data-intensive jobs
  - BigData
  - Machine Learning
  - Deep Learning
  - AI
  - High Performance Data Analytics
- 46 NVIDIA GPUs
- Expected Q4 2024

# How to get a CÉCI account?

How to use HPC infrastructure

Go to **http://www.ceci-hpc.be**



# Create/Manage Account

How to use HPC infrastructure

CÉCI Clusters News Training FAQ Documentation Support Contact

Create/Manage Account

To create an account your computer must be connected to your university network

Ok

C.E.C.I

Consortium des Équipements de Calcul In

6 clusters, 10 cores, 1 login, 1 home

## About

CÉCI is the 'Consortium des Équipements de Calcul Intensif'; a consortium of high-performance computing centers of UCLouvain, ULB, ULiège, UMons, and UNamur. The CÉCI is supported by the F.R.S-FNRS and the Walloon Region. Read more.


## Save the date!

The next CÉCI scientific day will take place on Thursday April 25th in Brussels.

More information soon!

## Latest News

MONDAY, 04 JUNE 2018

### LEMAITRE3 installed at UCL

Lemaitre3 is now operational and replaces Lemaitre2, which will be decommissioned this Summer. It has 80 nodes (SkyLake 2.3 GHz, 24CPUs, 96GB RAM) interconnected with the Intel OmniPath Architecture and more than half a petabyte of scratch space.

TUESDAY, 08 MAY 2018

### Dragon1 cluster featured in a Belnet article

The Dragon1 CÉCI cluster is highlighted in an interview from Belnet to Chantal Poiret, professor in Information and Communication Technology at the University of Mons.

## Quick links

# I want to… create an account



CÉCI Login Management FAQ

## I want to…

create an account

You are about to request an account on the CÉCI clusters.

The first step is to enter your email address. You will recieve an email with a link to an online form which you will have to fill and submit.

Once your request has been approved, you will receive proper information on how to access the CÉCI clusters.

renew my account

join an existing project

reset my passphrase

retrieve my private key

change my email address

invite a guest or renew a guest account

## create an account

**My email address:**

frederic.wautelet @unamur.be

Send

# That’s it

- Click on the link sent to you by email.
- Fill-in the form and hit the “Submit” button
- Get your SSH private key from your email
- Configure your SSH client
- Connect and profit!

# SSH tools

- Windows
  - PuTTY
  - **MobaXterm**
  - X-Win32
  - OpenSSH on Windows (Windows 10)
- Linux/MacOS
  - ssh
  - scp

# MobaXterm



- Easy to use
- No installation needed
- Command line interface
- Allow use of graphical application remotely
- Files transfer

# Bash

- Shell is the interface between the user and the Linux system
- Interprets and run commands
- For Linux, “Bash” is the default
- Shell scripts

# Modules

- Modify user's environment
- Allow use of application with different versions
- Commands:
  - $ module load/unload
  - $ module list
  - $ module available
  - $ module spider

# module available

How to use HPC infrastructure

```
------------------------------------------------------------------------ Meta Modules ------------------------------------------------------------------------
   dot     releases/elic-2017b    releases/2016b            releases/2018a    tis/2018.01 (S,L)
   null    releases/2016a         releases/2017b (S,L,D)    releases/2018b    use.own

------------------------------------------------------- TIS: Toolchain Independent Software (2018.01) -------------------------------------------------------
   EasyBuild/3.5.1    MCR/R2013a    MCR/R2015a    MCR/R2017a                          crystal/17-v1.0.1        julia/0.6.3
   Java/1.8.0_31      MCR/R2013b    MCR/R2015b    MCR/R2017b                          crystal/17-v1.0.2 (D)    julia/1.0.0  (D)
   Java/1.8.0_92      MCR/R2014a    MCR/R2016a    MCR/R2018a                    (D)   freesurfer/6.0.0         xpress/xp850
   Java/1.8.0_121     MCR/R2014b    MCR/R2016b    NCBI-BLAST-database/20170306        gurobi/gurobi800

------------------------------------------------------------------------ Releases (2017b) ------------------------------------------------------------------------
   ABINIT/8.4.4-intel-2017b                      Python/2.7.14-GCCcore-6.4.0-bare
   ANTLR/2.7.7-intel-2017b                       Python/3.6.3-foss-2017b
   Boost/1.65.1-foss-2017b                       Python/3.6.3-intel-2017b                          (D)
   Boost/1.66.0-intel-2017b                (D)   Qhull/2015.2-foss-2017b
   CDO/1.9.2-intel-2017b                         Qt/4.8.7-foss-2017b
   CGAL/4.11-foss-2017b-Python-2.7.14            R/3.4.3-foss-2017b-X11-20171023
   CP2K/5.1-intel-2017b                          Ruby/2.5.0-intel-2017b
   Doxygen/1.8.13-GCCcore-6.4.0                  SCOTCH/6.0.4-foss-2017b
   Eigen/3.3.4                                   SCOTCH/6.0.4-intel-2017b                          (D)
   FFTW/3.3.6-gompi-2017b                        SQLite/3.20.1-GCCcore-6.4.0
   FFTW/3.3.6-intel-2017b                  (D)   SWIG/3.0.12-foss-2017b-Python-2.7.14
   FLUENT/14.0                                   SWIG/3.0.12-foss-2017b-Python-3.6.3
   FLUENT/18.2                             (D)   SWIG/3.0.12-intel-2017b-Python-3.6.3              (D)
   GCC/6.4.0-2.28                                ScaLAPACK/2.0.2-gompi-2017b-OpenBLAS-0.2.20
   GDAL/2.2.3-foss-2017b-Python-2.7.14           Singularity/2.5.2-foss-2017b
   GDAL/2.2.3-foss-2017b-Python-3.6.3      (D)   UDUNITS/2.2.25-intel-2017b
   GEOS/3.6.2-foss-2017b-Python-2.7.14           UDUNITS/2.2.26-intel-2017b                        (D)
   GEOS/3.6.2-foss-2017b-Python-3.6.3            X11/20171023-GCCcore-6.4.0
   GEOS/3.6.2-intel-2017b-Python-3.6.3     (D)   YAXT/0.5.1-intel-2017b
   GLib/2.53.5-GCCcore-6.4.0                     foss/2017b
   GMP/6.1.2-GCCcore-6.4.0                       gc/7.6.0-GCCcore-6.4.0
   GSL/2.4-GCCcore-6.4.0                         gflags/2.2.1-intel-2017b
   Guile/1.8.8-GCCcore-6.4.0                     gompi/2017b
   HDF5/1.8.19-intel-2017b                       grib_api/1.24.0-intel-2017b
lines 1-36
```

# Interactive or batch



- Interactive
  - Short tasks
  - Tasks that require frequent user interaction
  - Graphically intensive tasks
- Batch
  - Longer running processes
  - Parallel processes
  - Running large numbers of short jobs simultaneously
  - Submitted to a job scheduler

# Job scheduler



- Dispatch the batch jobs on compute nodes
- Parameters
  - Memory
  - Processor type
  - Execution time
  - Number of processors
  - Software license tokens
- Slurm workload manager

# Submit a batch job




- Connect to a login node

```
$  ssh hercules.ptci.unamur.be
```

# Job scripts

- Define resources to be reserved for your job:
  - CPU time
  - memory
  - platform
  - number of CPUs
  - List instructions to be executed

- Bash shell script

# Job scripts

- run.sh

```
#!/bin/bash
#SBATCH --job-name=hello
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=1:00:00
#SBATCH --mem-per-cpu=1000

echo "Hello World!"
```

# Submitting jobs



- Submit the job script

```
$  sbatch run.sh
Submitted batch job 3513668
```

  - Return the job id

- Job is running

```
$  squeue -u $USER
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST
           3513667       cpu    hello fwautele  R       0:12      1 n065
```

- Job is finished

```
$  squeue -u $USER
$
```

# Batch jobs



- Check output file

```
$  ls -altr
…
-rw-rw-r--  1 fwautele fwautele          13 Feb 26 11:16 slurm-3513668.out
```

- Hello world!

```
$  cat slurm-3513668.out
Hello World!
```

# Safeguards

- Slurm will automatically cancel jobs:
  - When the memory reserved is exceeded
  - When time is over

- Slurm constraint job in the number of core requested

# Delete a job

- scancel

```
$ scancel 2243523
```

- You can only delete your own jobs... (hopefully)

# Monitoring jobs

How to use HPC infrastructure

- squeue

```
$  squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           2619747       cpu PYV3_FBI jquertin  R   16:15:37      1 n076
           2619745       cpu PYV3_DHB jquertin  R 4-14:36:35      1 n020
           2620638       cpu PYV3_FA_ jquertin  R      43:33      1 n025
           2618213       cpu PYV3_SDP jquertin  R 9-19:40:43      1 n054
           2620635       cpu PYV3-CC2 jquertin  R      56:59      1 n020
           2620632       cpu PYV3-CC2 jquertin  R      59:22      1 n014
           2620633       cpu PYV3-CC2 jquertin  R      59:22      1 n014
           2620630       cpu PYV3-CC2 jquertin  R      59:52      1 n054
           2620631       cpu PYV3-CC2 jquertin  R      59:52      1 n064
           2620627       cpu PYV3-CC2 jquertin  R    1:01:24      1 n064
           2620628       cpu PYV3-CC2 jquertin  R    1:01:24      1 n064
           2620622       cpu PYV3-CC2 jquertin  R    1:18:17      1 n076
```

# Slurm Script Wizard

How to use HPC infrastructure

### 1. Describe your job

Email address: user@example.com

Job name: Some name

Project: Some project

Output file: (default)

**Parallelization paradigm(s)**

- [ ] Embarrassingly parallel / Job array
- [ ] Shared memory / OpenMP
- [ ] Message passing / MPI
- [ ] GPU / CUDA

**Job resources**

Duration : 0 days, 1 hour, 0 minutes.

Memory : 1000 MB

**Filesystem**

Filesystem: $HOME

Total CPUs: 1 | Total Memory: 1000 MB | Total CPU.Hours: 1

### 2. Choose a cluster

- (•) Lemaitre3
- ( ) Hercules2
- ( ) Dragon1
- ( ) Dragon2
- ( ) Nic5
- Lucia

### 3. Copy-paste your script

```
#!/bin/bash
# Submission script for Lemaitre3
#SBATCH --time=01:00:00 # hh:mm:ss
#
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=1000 # megabytes
#SBATCH --partition=batch,debug

module purge
module load LIST_THE_MODULES_YOU_NEED_HERE
```

http://www.ceci-hpc.be/scriptgen.html

# Array jobs

- Run several instances of the same program with different inputs

- Same allocation options
  - Memory size
  - Time limit
  - …

# --array options

```
# SBATCH --array=0-31
```

```
# SBATCH --array=1,3,5,7
```

```
# SBATCH -array=1-7:2
```

```
# SBATCH --array=1-15%4
```

# Example

How to use HPC infrastructure

```
$ sbatch --array=0-3 run.sh
Submitted batch job 3512681

$ squeue -u fwautele
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIS
         3512681_0       cpu   run.sh fwautele  R       0:12      1 n064
         3512681_1       cpu   run.sh fwautele  R       0:12      1 n077
         3512681_2       cpu   run.sh fwautele  R       0:12      1 n047
         3512681_3       cpu   run.sh fwautele  R       0:12      1 n047
```

# Job Dependencies

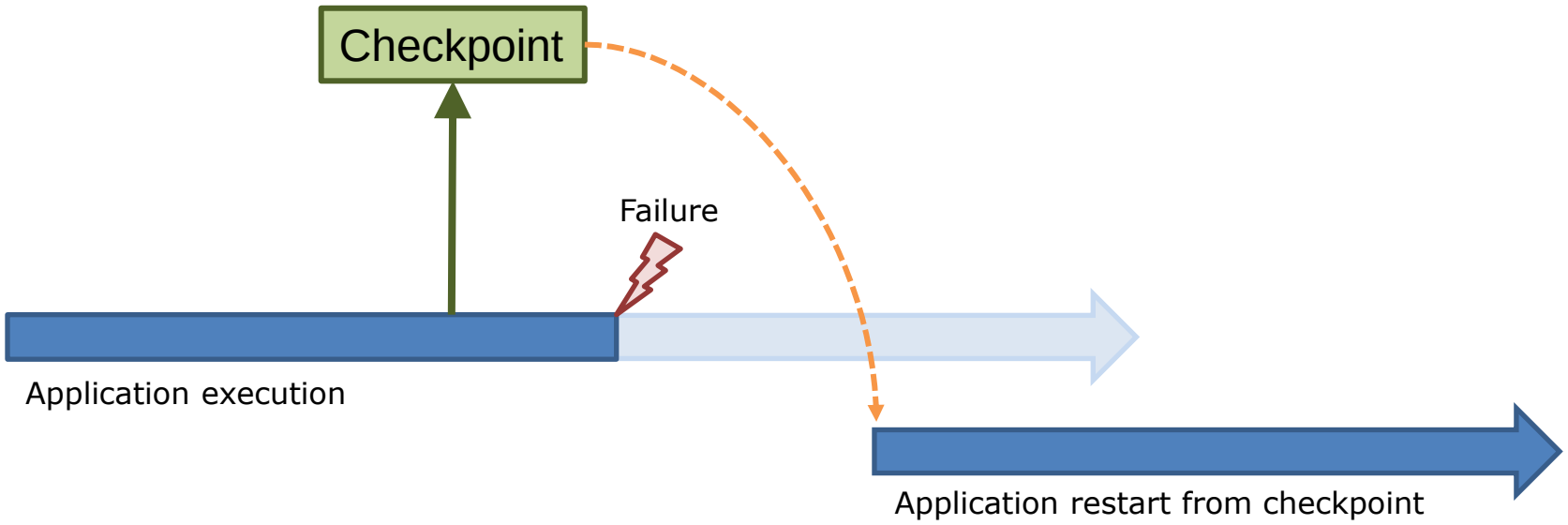


- A job can be dependent upon other job(s) status

- Dependency type:
  - after
    - after the specified jobs have started
  - afterany
    - after the specified jobs have terminated
  - afternotok
    - after the specified jobs have failed
  - afterok
    - after the specified jobs have terminated successfully

# Checkpointing

- To overcome job time limitation
- Allow rollback-recovery for long-running applications
- Enable job migration



# Code and Data Versioning



## Code versioning

Distributed source management system

## Data versioning

Large datasets versioning (based on git)

# Outline

- How to use HPC infrastructure
- How to program on HPC cluster
- Going Parallel
- Data Management

# Outline

## How to program on HPC cluster

I. Scientific software, interpreted languages, compilers, (c)make

II. C, Python

III. Python, Julia, Fortran

IV. Parallel computing, Debugging/Profiling, Advanced Python

# Objectives

- Building from source is preferred in an HPC environment
- Allow users to install applications
  - Link with numerical libraries
  - Built with optimized compiler
- Special case
  - Python
  - R
  - Perl

# Compilers available

- GNU Compiler Collection (GCC)
- Intel Parallel Studio XE Cluster Edition
- NVIDIA HPC SDK

# GNU Compiler Collection

- Alias « GCC »
- Open Source (GPL)
- Pretty good performance
- Compiler suite
  - gcc: C compiler
  - g++: C++ compiler
  - gfortran: Fortran compiler
- module load foss

# Intel Parallel Studio XE Cluster Edition



- Commercial
- High performance compiler
- Compiler suite
  - icc: C compiler
  - icpc: C++ compiler
  - ifort: Fortran compiler
- module load intel

# NVIDIA HPC SDK

- Previously “PGI” CDK
- Commercial
- Offloading on GPU
- Compiler suite
  - pgcc: C compiler
  - pgCC: C++ compiler
  - pgf77: Fortran 77 compiler
  - pgf90: Fortran 90 compiler
- module load PGI (for older versions)
- module load NVHPC (for recent versions)

# Optimized libraries



- Do not reinvent the wheel
- Use multicore-tuned libraries.
- Use optimized libraries
  - Boost
  - FFTW
  - GMP
  - GSL
  - HDF5
  - …

# Compiler Toolchains



- Compiler toolchain =
  - Compiler
  - + MPI library
  - + BLAS/LAPACK library
    - linear algebra routines
  - + FFT library
    - Fast Fourier Transforms

- Examples
  - foss/2022b
  - intel/2022b

# Compiler Toolchains



- Open Source compiler toolchain
  - foss/2022b
    - GCC 12.2.0
    - OpenMPI 4.1.4
    - OpenBLAS 0.3.21 (including LAPACK)
    - FlexiBLAS 3.2.1
    - ScaLAPACK 2.1.0
    - FFTW 3.3.10

# Compiler Toolchains



- Intel Parallel Studio XE Cluster Edition 2022
  - A toochain: intel/2022b
    - icc 2022.2.1 (C compiler)
    - icpc 2022.2.1 (C++ compiler)
    - ifort 2022.2.1 (Fortran compiler)
    - impi 2021.7.1 (Intel MPI)
    - MKL 2022.2.1 (Math Kernel Library)

# Python

- Python 2 (deprecated)
  - Python/2.7.16-GCCcore-8.3.0
  - Python/2.7.18-GCCcore-9.3.0
  - Python/2.7.18-GCCcore-10.2.0

- Python 3
  - Python/3.9.5-GCCcore-10.3.0
  - Python/3.9.6-GCCcore-11.2.0
  - Python/3.10.8-GCCcore-12.2.0

# Installing languages extensions



- Install with PIP
  - PIP is the easiest and recommended way to install Python packages

```
$ pip install --user example
```

- Install from source
  - If package not available on PIP
  - Steps:
    - Download the source and unpack it
    - Change to the source directory
    - python setup.py install --prefix=$HOME/.local

```
$ python setup.py install --prefix=$HOME/.local
```

# R

- Available versions
  - R/4.0.3-foss-2020b
  - R/4.1.0-foss-2021a
  - R/4.2.0-foss-2021b
- Already bundle with a set of libraries
  - Type "**`installed.packages()`**" to list them
- Additional libraries
  - R-bundle-Bioconductor/3.11-foss-2020a-R-4.0.0
  - R-bundle-Bioconductor/3.12-foss-2020b-R-4.0.3
  - R-bundle-Bioconductor/3.13-foss-2021a-R-4.1.0.eb

# Octave

- Interactive programming language
- Suited for numerical calculations
- Alternative to MATLAB

- Available version(s)
  - Octave/5.1.0-foss-2019b

# Julia

- General-purpose and high-level as Python
- Interactive as R
- But fast as C

- Available versions
  - Julia/1.6.7-linux-x86_64
  - Julia/1.8.2-linux-x86_64

# Profiling = finding hotspots



- Hotspot = Where in an application or system there is a significant amount of activity
  - Where: address in memory → line of source code
  - Significant: activity that occurs infrequently probably does not have much impact on system performance
  - Activity: time spent or other internal processor event

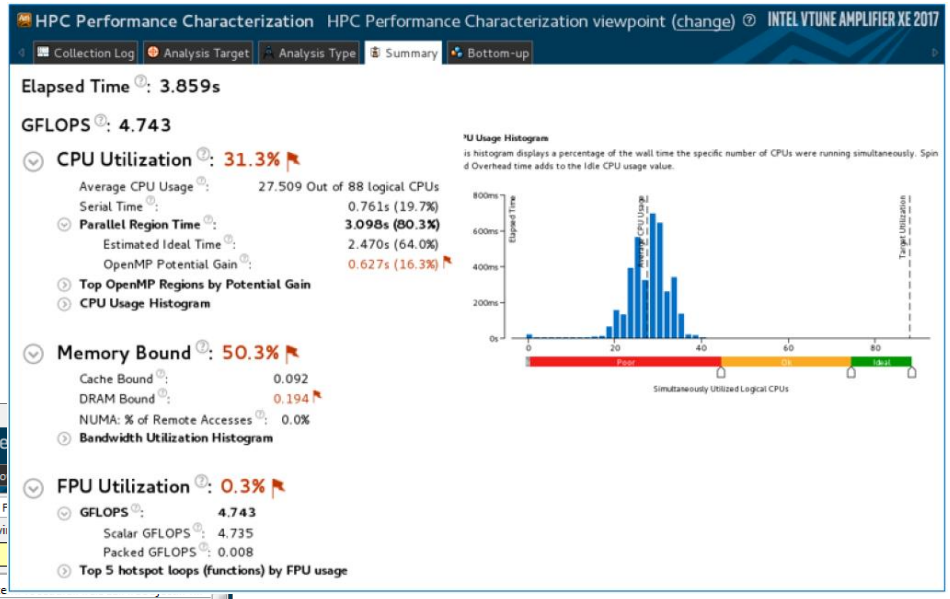
# Intel Vtune™ Amplifier

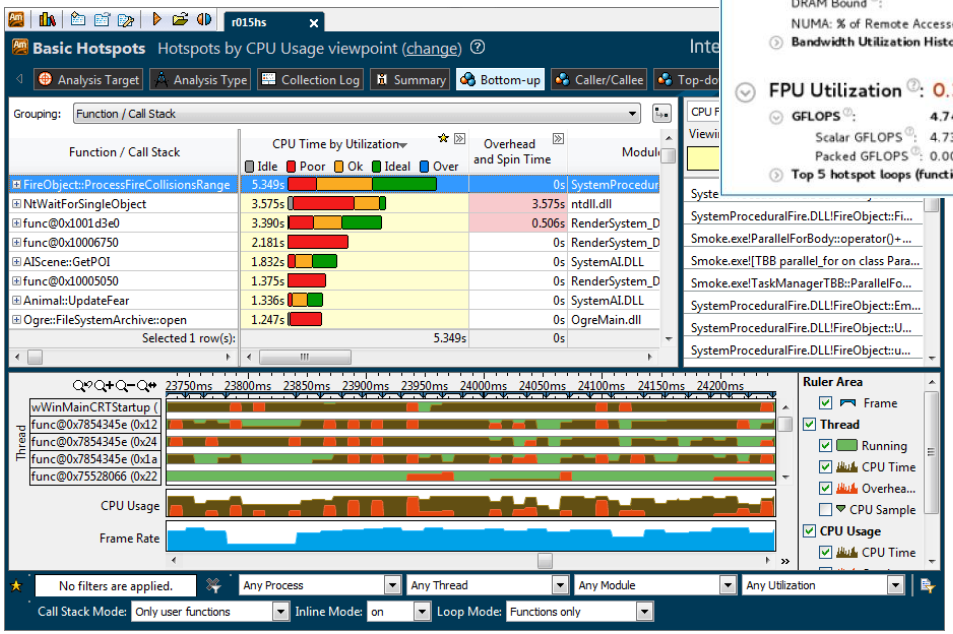


- What is the VTune™ Performance Analyzer?
  - Helps you identify and characterize performance issues by:
    - Collecting performance data
    - Organizing and displaying the data from system-wide down to source code or processor instruction
    - Identifying potential performance issues and suggesting improvements
  - Able to analyse serial, OpenMP and MPI application

```
$ ml load VTune
```

# Intel Vtune™ Amplifier




intel® Software

# Outline

- How to use HPC infrastructure
- How to program on HPC cluster
- Going Parallel
- Data Management

# Outline

## Going Parallel

I. Julia and OpenMPI

II. MPI, OpenMP

III. GPU

# Job type



- Sequential job
  - A single core on one node
- Threaded jobs
  - Several cores on one node
  - OpenMP
- MPI jobs
  - Several cores on several nodes
  - OpenMPI, MPICH, …

# Accelerators



- Hardware component with a specialized microprocessor
- Mostly General Purpose Graphical Processing Units (GPGPUs)
- Offer excellent floating point performance per Watt
- Parts of computation “offloaded” to accelerator

# GPGPUs resources at CÉCI

Going Parallel

| Cluster | Model | Cores | Memory | Float performance (FP32) | Double performance (FP64) |
|---|---|---|---|---|---|
| Dragon2 | 4 x NVIDIA Tesla V100 | 5120 | 16 GB | 14 TFLOPS | 7 TFLOPS |
| Hercules2 | 4 x NVIDIA RTX A6000 | 10752 | 48 GB | 40 TFLOPS | 1 TFLOPS |
| | 8 x NVIDIA Tesla A40 | 10752 | 48 GB | 38 TFLOPS | 600 GFLOPS |

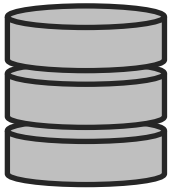
# Outline

## Data Management

I. Data storage and transfer, filesystems

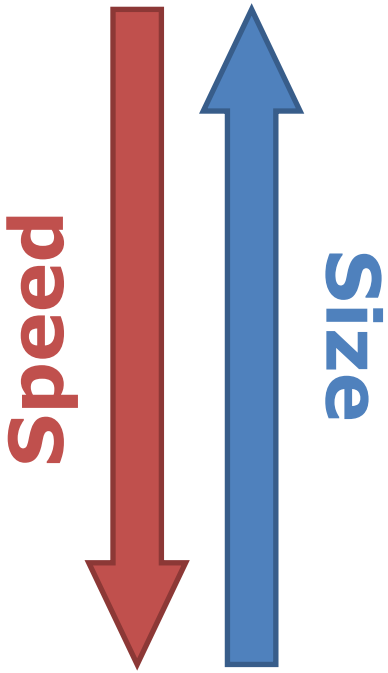
II. Data management plan, netCDF/HDF5 file format

# Four levels of storage



Data storage

- $CECIHOME
  - 400 TB
  - CÉCI distributed storage
- $HOME
  - Programs and scripts
- $WORKDIR
  - Input and output data
- $LOCALSCRATCH or $GLOBALSCRATCH
  - Job temporary data



# Scientific data: text or binary?

Data Management

**Table 5**

*Simulation results for using full data, CRs only, and proposed method under four missing mechanisms*

| Method | Bias[a] $(\hat{\beta}_W)$ | Bias[a] $(\hat{\beta}_X)$ | Variance[b] $(\hat{\beta}_W)$ | Variance[b] $(\hat{\beta}_X)$ | 95% CI[c] $(\hat{\beta}_W)$ | 95% CI[c] $(\hat{\beta}_X)$ |
|---|---|---|---|---|---|---|
| **(M.1)** $P(R=1)=0.66$ | | | | | | |
| Full | 0.01346 | 0.02229 | 0.04008 | 0.03685 | 0.955 | 0.950 |
| Comp | 0.03062 | −0.003561 | 0.1149 | 0.06732 | 0.960 | 0.955 |
| Impu | 0.01431 | 0.021 | 0.04088 | 0.05169 | 0.980 | 0.975 |
| **(M.2)** logit $P(R=1)=2Y$ | | | | | | |
| Full | 0.007908 | −0.02116 | 0.03838 | 0.03624 | 0.975 | 0.925 |
| Comp | 0.01945 | 0.07096 | 0.107 | 0.06581 | 0.960 | 0.950 |
| Impu | 0.006966 | 0.01597 | 0.04227 | 0.05226 | 0.975 | 0.985 |
| **(M.3)** logit $P(R=1)=2X$ | | | | | | |
| Full | 0.007908 | −0.02116 | 0.03838 | 0.03624 | 0.975 | 0.925 |
| Comp | 0.01225 | 0.0589 | 0.08856 | 0.06818 | 0.980 | 0.975 |
| Impu | 0.009563 | −0.04699 | 0.03865 | 0.04923 | 0.985 | 0.970 |
| **(M.4)** logit $P(R=1)=X+Y$ | | | | | | |
| Full | 0.01346 | 0.02229 | 0.04008 | 0.03685 | 0.955 | 0.950 |
| Comp | 0.02404 | 1.613 | 0.1102 | 0.08202 | 0.955 | 0.580 |
| Impu | 0.01814 | 0.08289 | 0.0578 | 0.06075 | 0.955 | 0.970 |

[a]Bias $= (\hat{\beta}-\beta_0)/\beta_0$.

[b]Simulation variance.

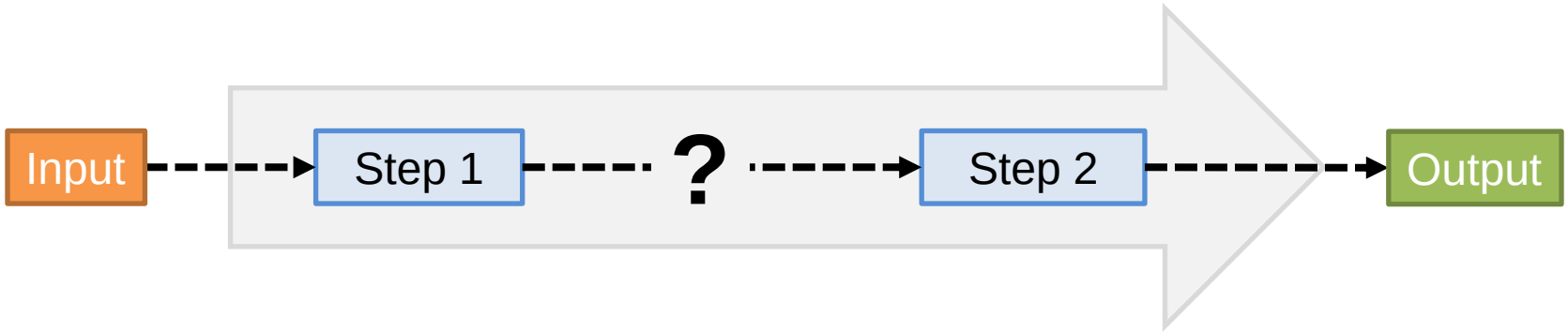
[c]Confidence interval using jackknife standard error.

# Scientific data

- What is scientific data?
- N-dimensional arrays + metadata:
  - Measurements at specific time, location, condition
  - Physics: temperature, pressure
  - Chemistry: reaction speed
  - Biology: type (species, cell types, nucleotides)
  - Economics: price
  - …

# Example

- Problem:




- Example:
  - A data crushing software written in Fortran generate results
  - A post-processing application written in Python read this results

# Solution 1: Text file



- Pro:
  - Human readable
  - Easy to write
  - Platform independent (Endianness)
  - Very flexible
  - Easy to add a variable
- Cons:
  - Sometime hard to parse
  - No accuracy
  - Performance problem
  - Data size

# Solution 2: NetCDF



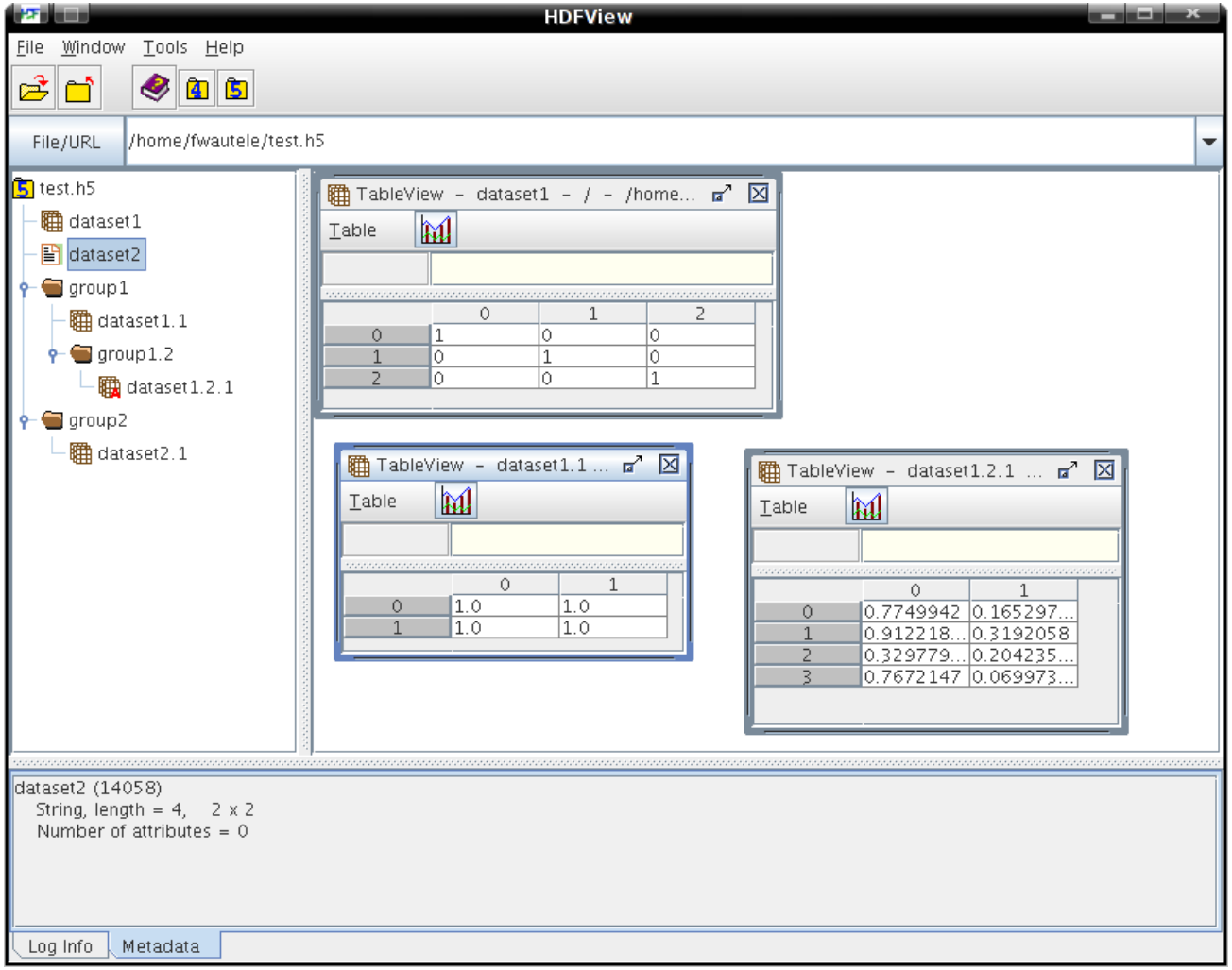
- Network Common Data Form
- For array oriented scientific data
- Available in many programming and scripting languages
  - C++, Java, Fortran, Perl, Python, R, …
- Emphasizes simplicity over power (unlike HDF5)

# Solution 3: HDF5



- Open file format
- Can represent very complex data objects
  - Like a files hierarchy
- No limit on the number or size of data objects
- Allow access time and storage space optimizations
- Many tools available

# HDFView




https://www.hdfgroup.org/downloads/hdfview/

# Green HPC

- Green500
  - Rank supercomputers in terms of energy efficiency
  - Performance per Watt (GFLOPS/W)
  - https://www.top500.org/lists/green500

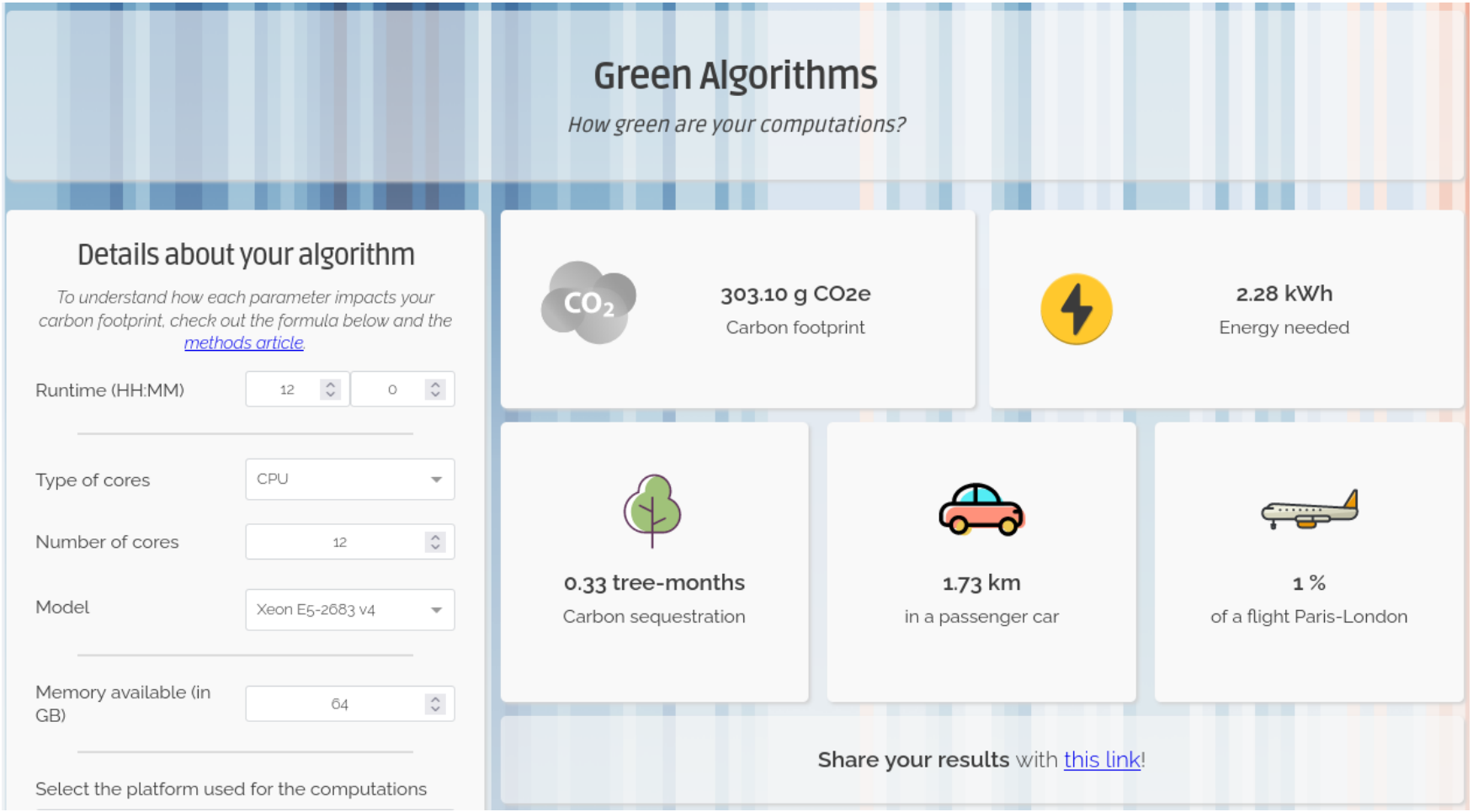

Jedi, EuroHPC, Juelich, Germany

# A green supercomputer: LUMI

- 200,000 cores
- Negative carbon footprint
- 100% renewable energy
- Wasted heat can be used by 20% of the houses of the surrounding city

# Carbon footprint of your computation

- http://www.green-algorithms.org



# Thanks you for your attention

and happy computing