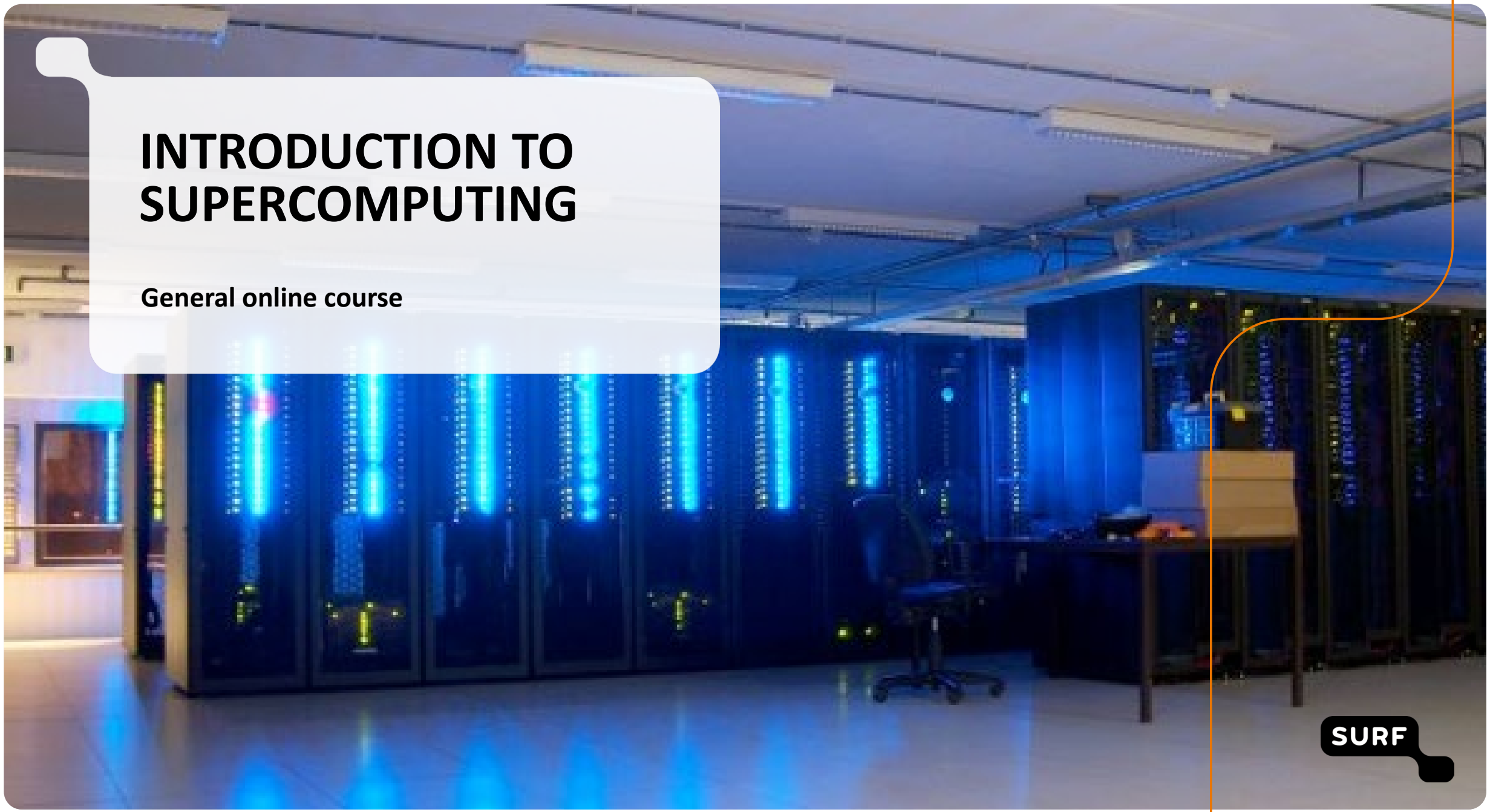


INTRODUCTION TO SUPERCOMPUTING

General online course

SURF

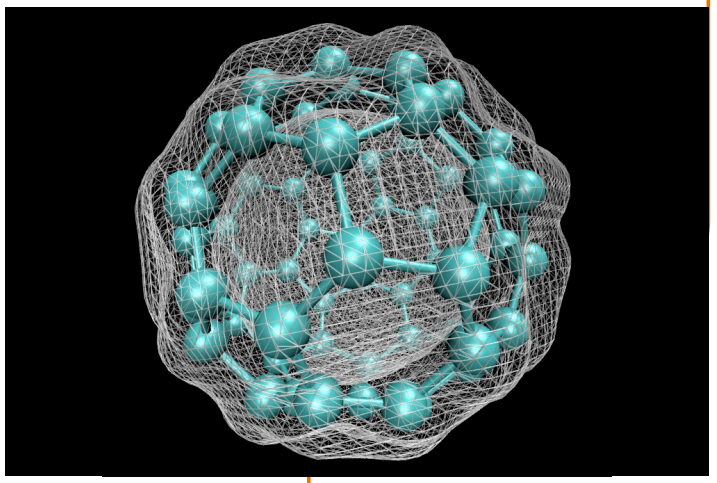
Outline

- Introduction to High Performance Computing
  - Definitions
  - Parallel programming
- SURF facilities
  - Presentation
  - Systems and specifications
- Running jobs
  - Definitions for jobs
  - Hands-on exercises with the files available in your home directories

Outline

- Introduction to High Performance Computing
  - Definitions
  - Parallel programming
- SURF facilities
  - Presentation
  - Systems and specifications
- Running jobs
  - Definitions for jobs
  - Hands-on exercises with the files available in your home directories

High-performance computing (HPC) is …



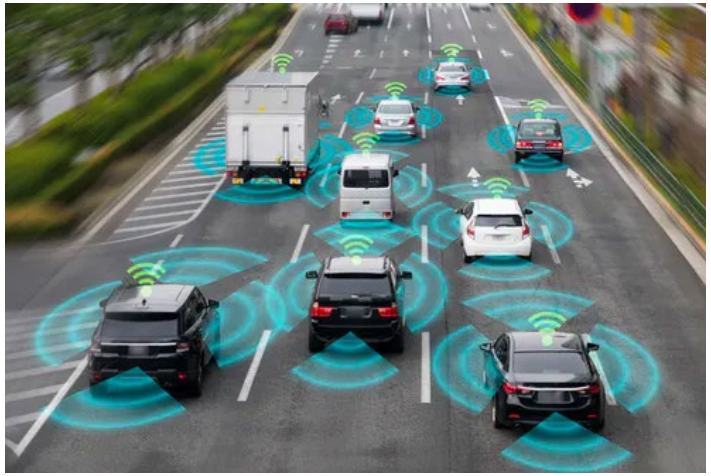


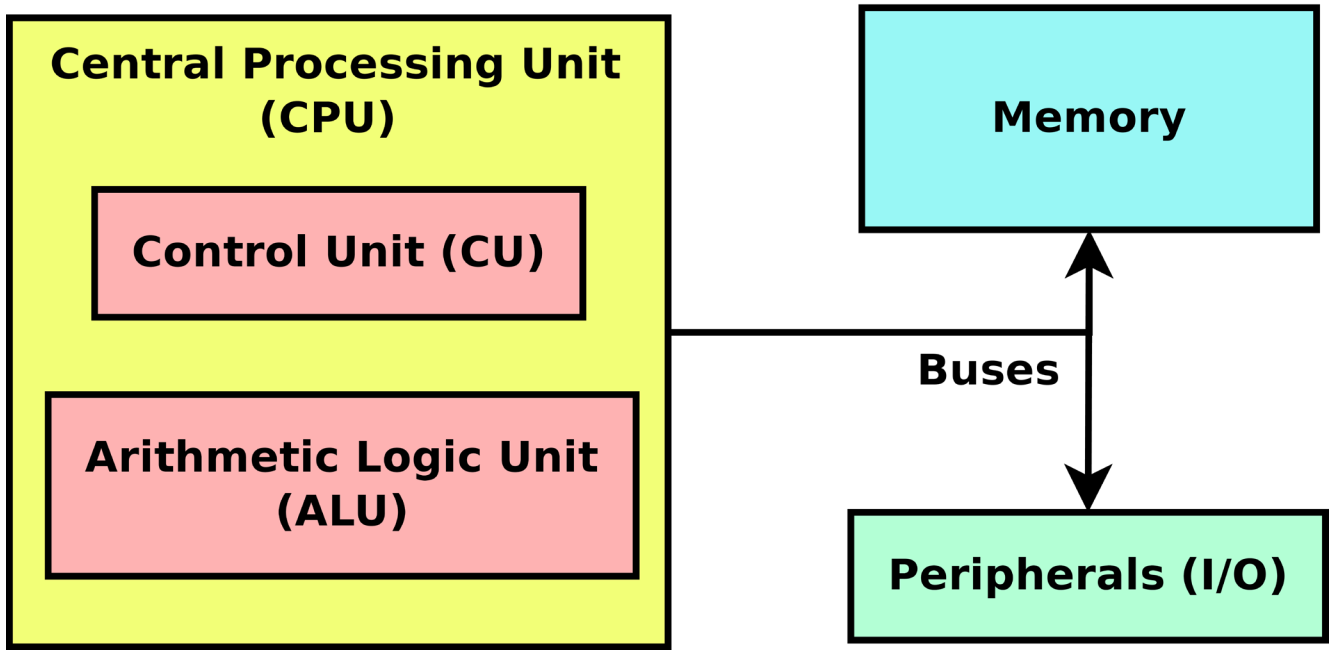
Image source: https://www.trustedreviews.com/news/gran-turismo-ps5-will-complete-form-series-3895762
Image source: http://www.militarysystems-tech.com/articles/l-3-link-achieves-world-s-first-boeing-787-8-full-flight-simulator-level-d-certifications
Image source: https://www.zdnet.com/article/canberra-stands-up-au9-7m-transport-office-to-prep-for-autonomous-vehicles
Image source: https://spacenews.com/astranis-lands-anchor-customer-for-its-first-small-geo-satellite
Image source: https://medium.com/@info_89535/applications-of-fullerene-in-medicine-20942944e41d
Image source: https://thestrategybridge.org/the-bridge/2016/8/16/a-new-plan-using-complexity-in-the-modern-world

High-performance computing (HPC) is …

- *… an area of computer-based computation. It includes all computing work that requires a high computing capacity or storage capacity.*
- *… the use of parallel processing for running advanced application programs efficiently, reliably and fast.*
- *… the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.*
- *… the use of super computers and parallel processing techniques for solving complex computational problems.*

A computer is …

A computer is …



… and Boolean logic (0's and 1's)

A central processing unit (CPU) is …

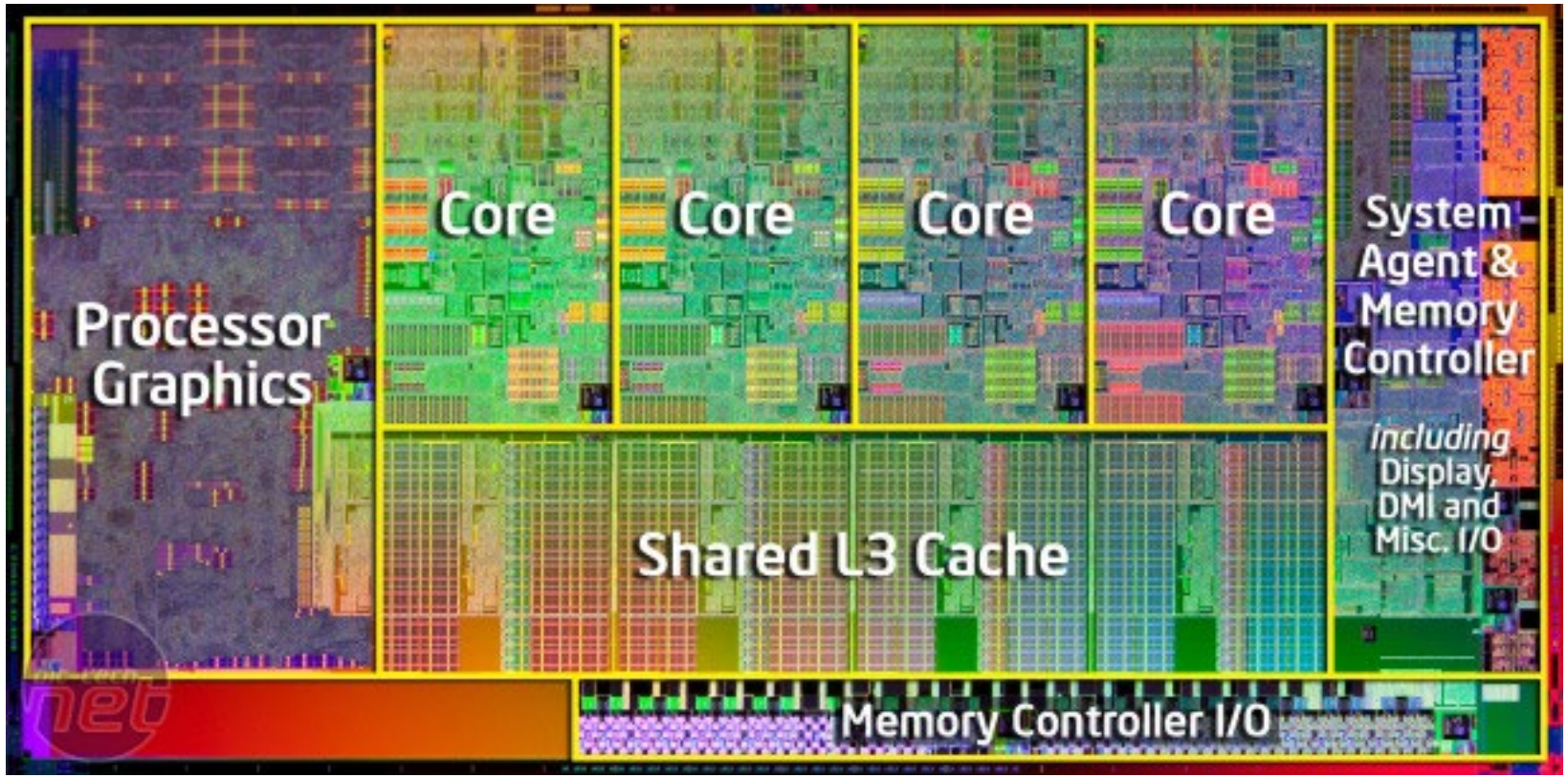
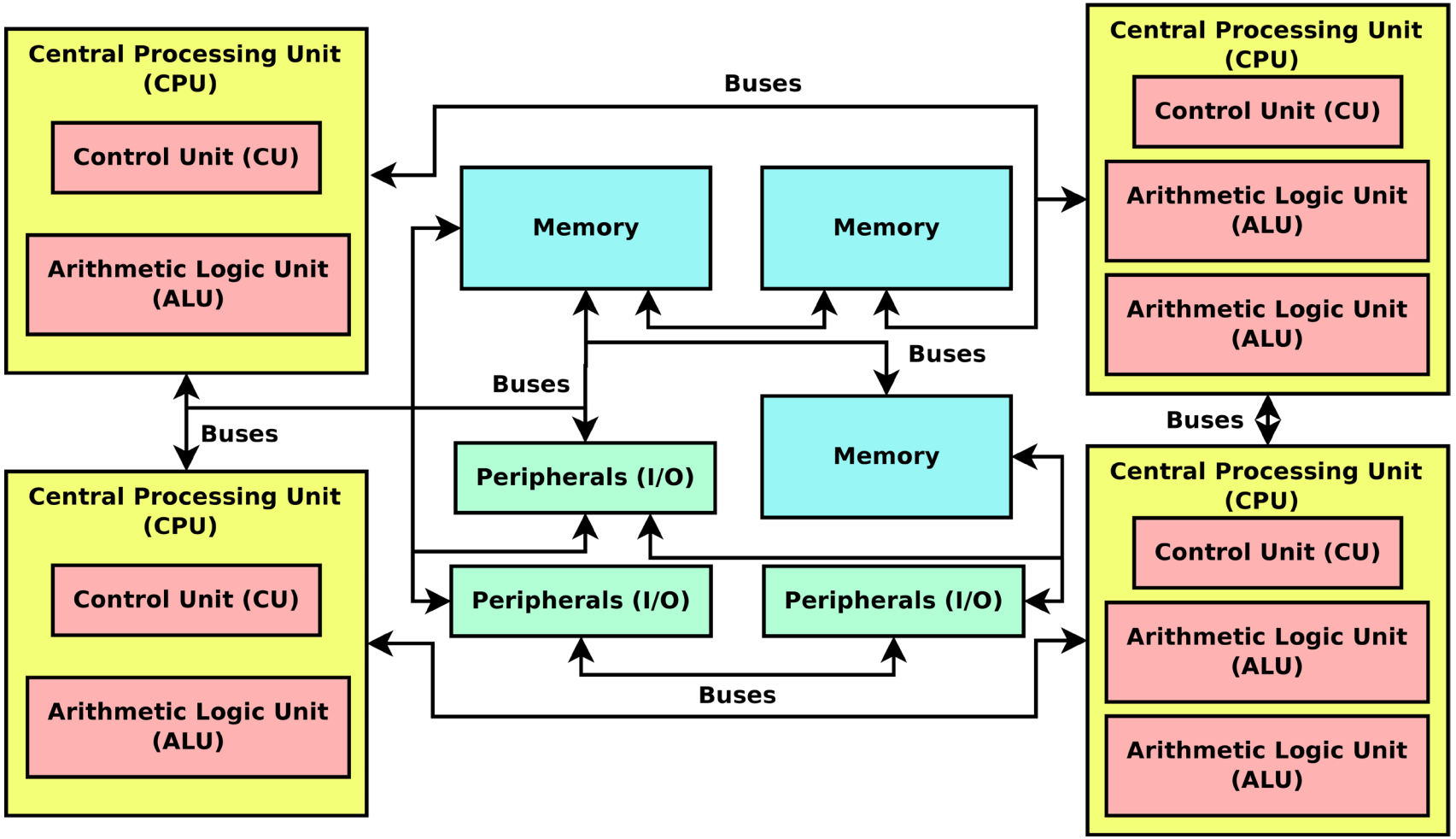


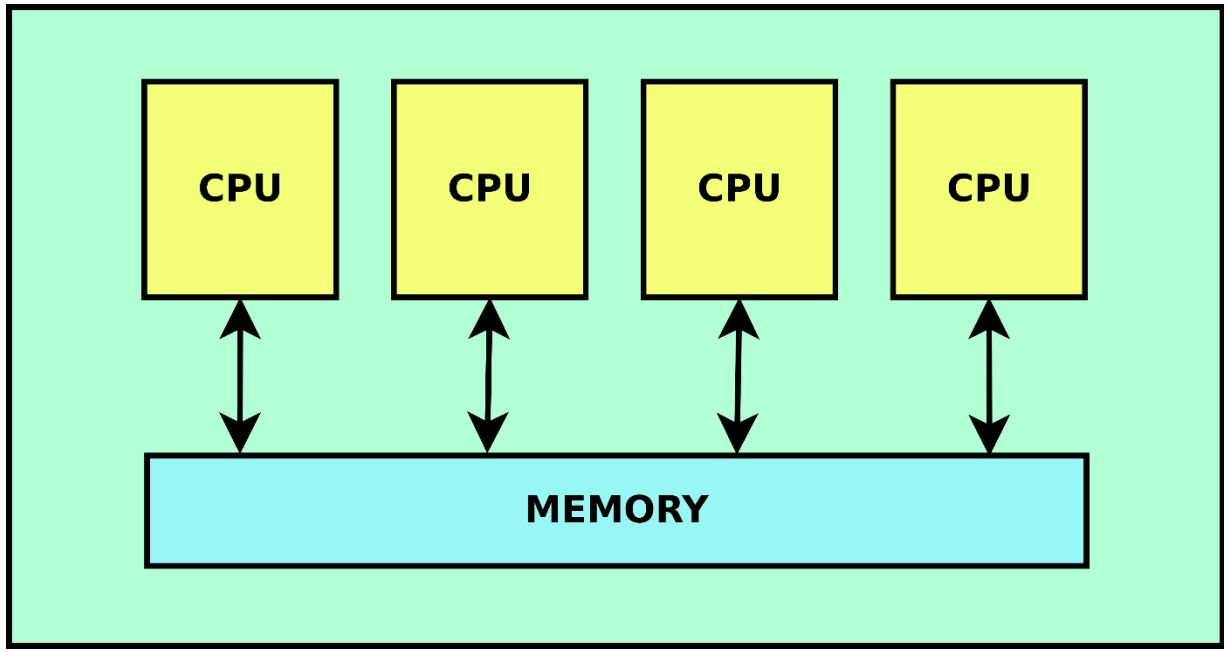
Image source: https://bit-tech.net/reviews/tech/cpus/intel-sandy-bridge-review/1/

Image source: https://www.computerhope.com/jargon/s/sandybri.htm

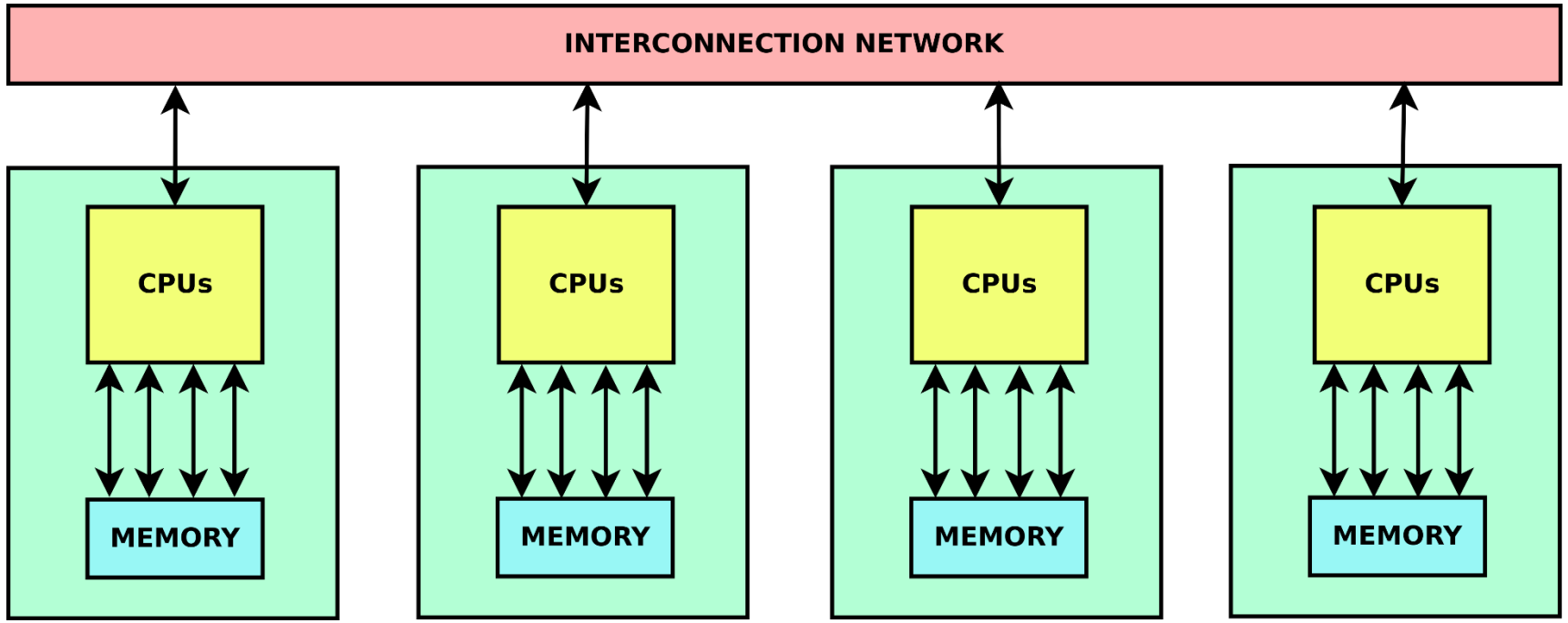
A larger computer could be …



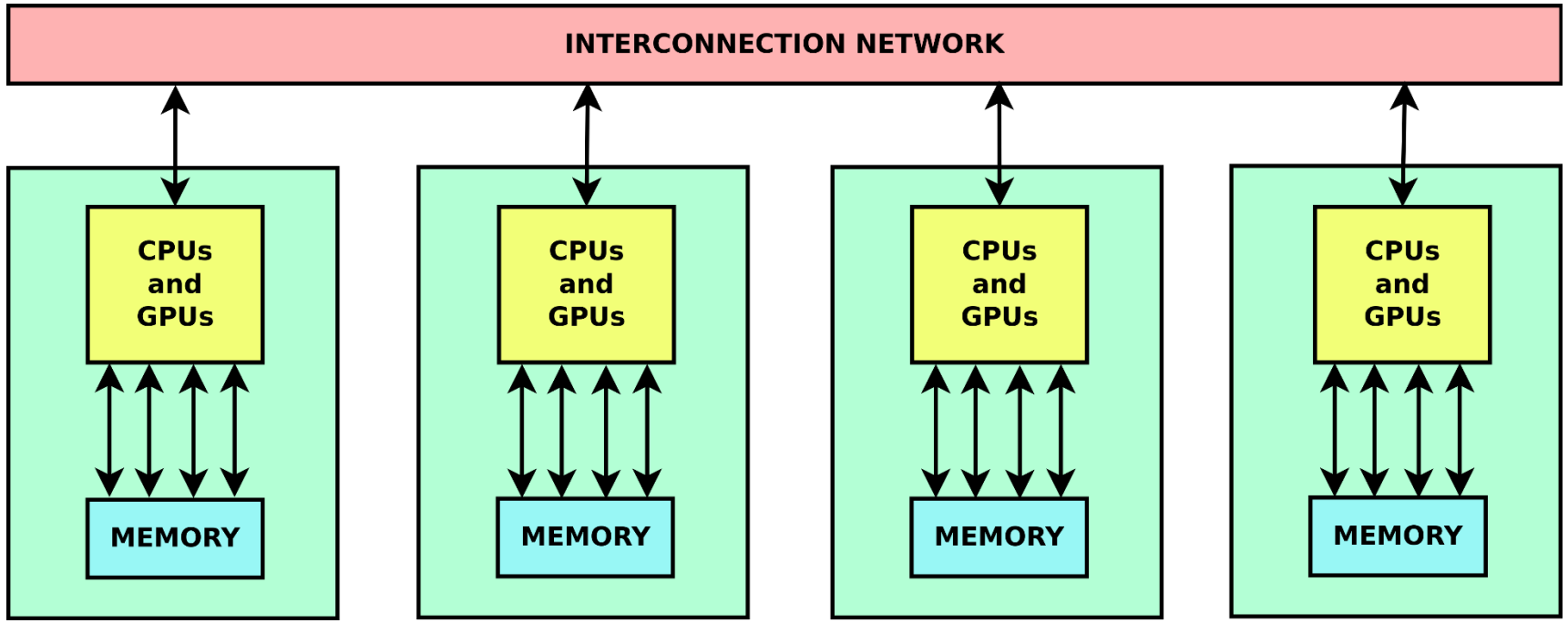
A larger computer actually is …



A larger computer actually is …



A larger computer actually is …



High-performance computing (HPC) …

- *… is an area of computer-based computation. It includes all computing work that requires a high computing capacity or storage capacity.*
- *… is the use of parallel processing for running advanced application programs efficiently, reliably and fast.*
- *… refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.*
- *… is the use of super computers and parallel processing techniques for solving complex computational problems.*

High-performance computing (HPC) …

- *… is an area of computer-based computation. It includes all computing work that requires a high computing capacity or storage capacity.*
- *… is the use of parallel processing for running advanced application programs efficiently, reliably and fast.*
- *… refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.*
- *… is the use of super computers and parallel processing techniques for solving complex computational problems.*
- *… is the part of computing focused on making computers collaborate efficiently up to very large scales*
- *… is optimized and scalable computer coordination (hardware and software)*

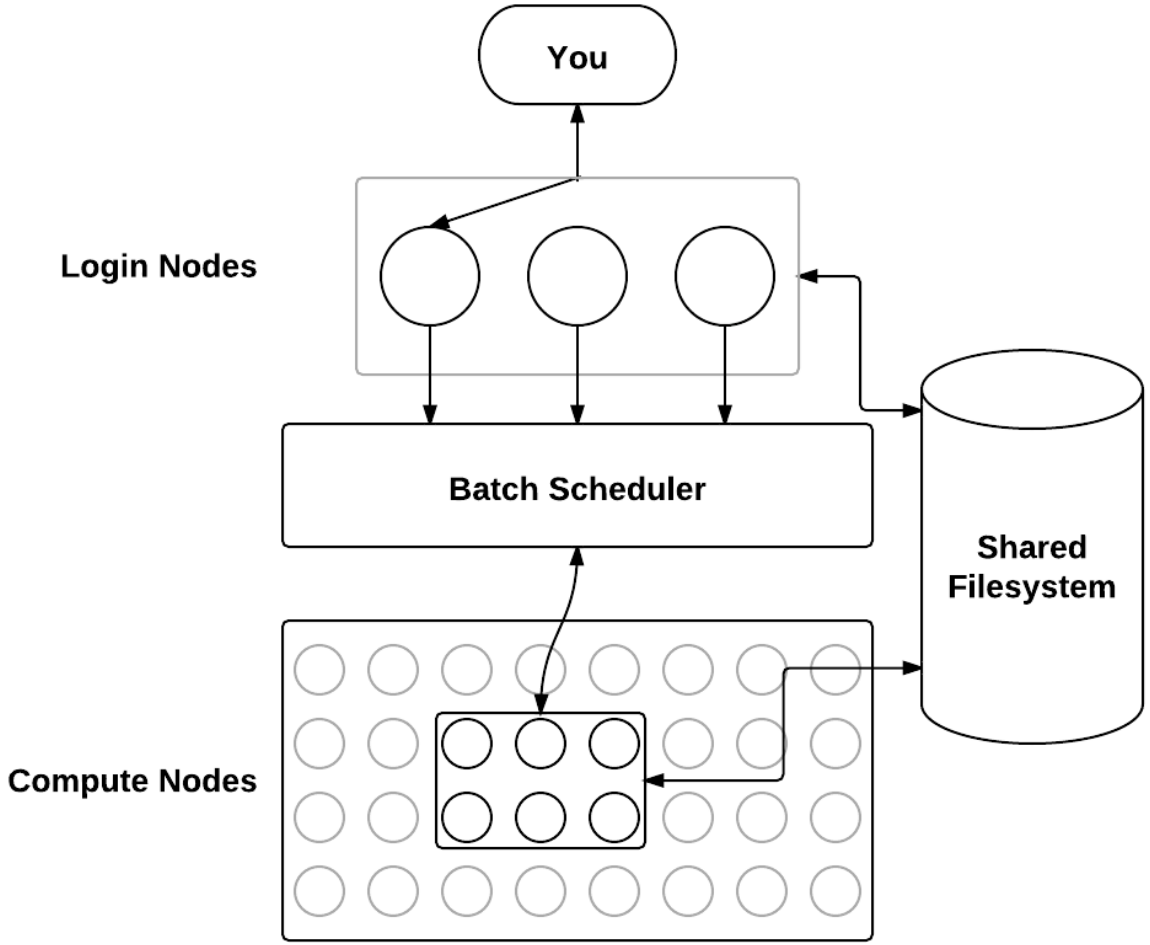
Outline

- Introduction to High Performance Computing
  - Definitions
  - Parallel programming
- SURF facilities
  - Presentation
  - Systems and specifications
- Running jobs
  - Definitions for jobs
  - Hands-on exercises with the files available in your home directories

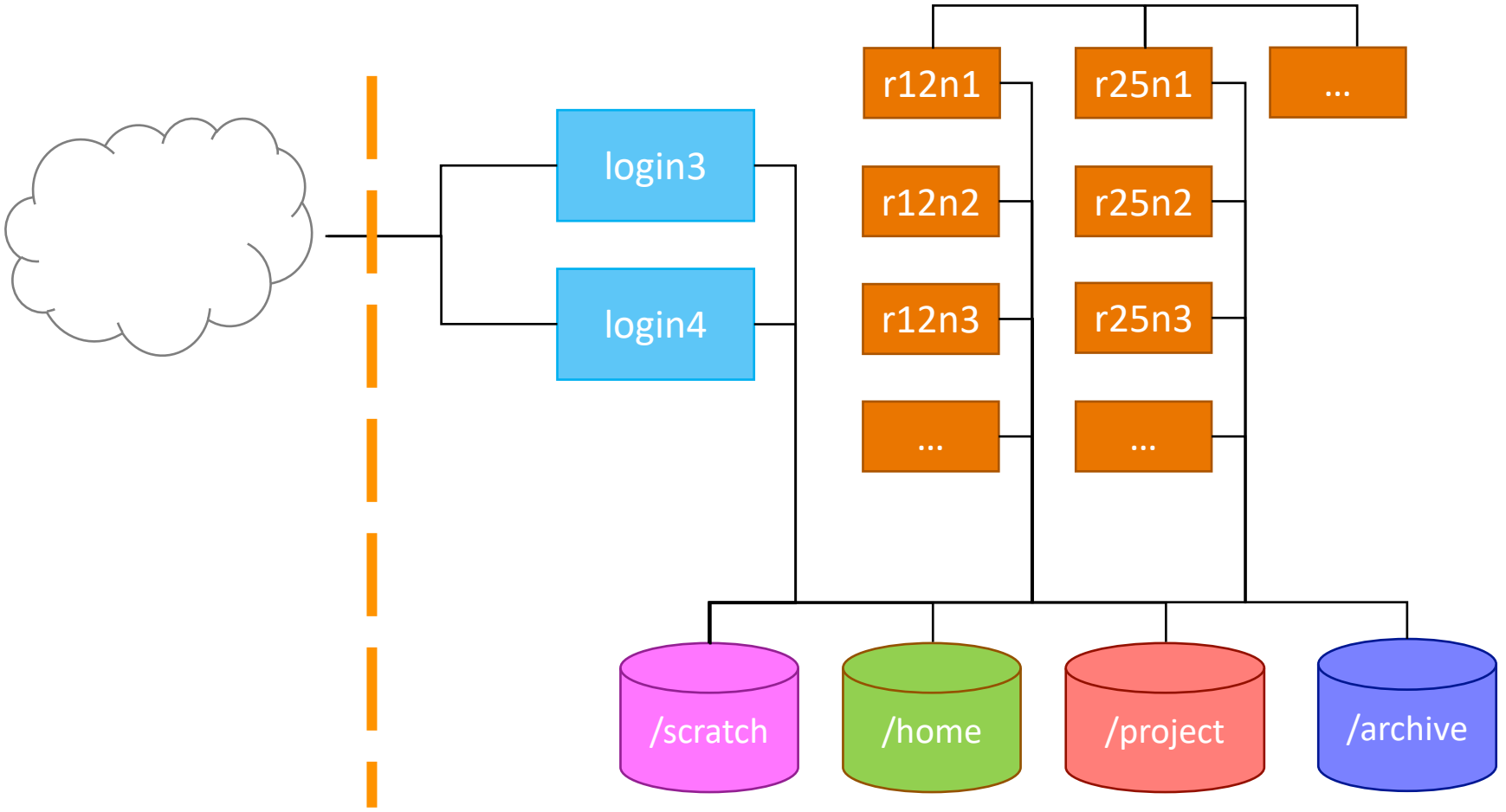
Dutch national supercomputers: performance increase

Year	Machine	R_{peak} (GFlop/s)	kW	GFlop/s/ kW
1984	CDC Cyber 205 1-pipe	0.1	250	0.0004
1988	CDC Cyber 205 2-pipe	0.2	250	0.0008
1991	Cray Y-MP/4128	1.33	200	0.0067
1994	Cray C98/4256	4	300	0.0133
1997	Cray C916/121024	12	500	0.024
2000	SGI Origin 3800	1,024	300	3.4
2004	SGI Origin 3800 +SGI Altix 3700	3,200	500	6.4
2007	IBM p575 Power5+	14,592	375	40
2008	IBM p575 Power6	62,566	540	116
2009	IBM p575 Power6	64,973	560	116
2013	Bull bullx DLC	250,000	260	962
2014	Bull bullx DLC	~ 1,000,000	520	1923
2017	Bull bullx DLC + KNL	~ 1,840,000	850	2168
2021	Lenovo AMD (1st phase)	~ 6,100,000	610	10000
2023	Lenovo AMD (2nd phase)	~ 14,900,000	1050	14190
2016	Raspberry PI 3 (35 euro)	0.44	0.004	110

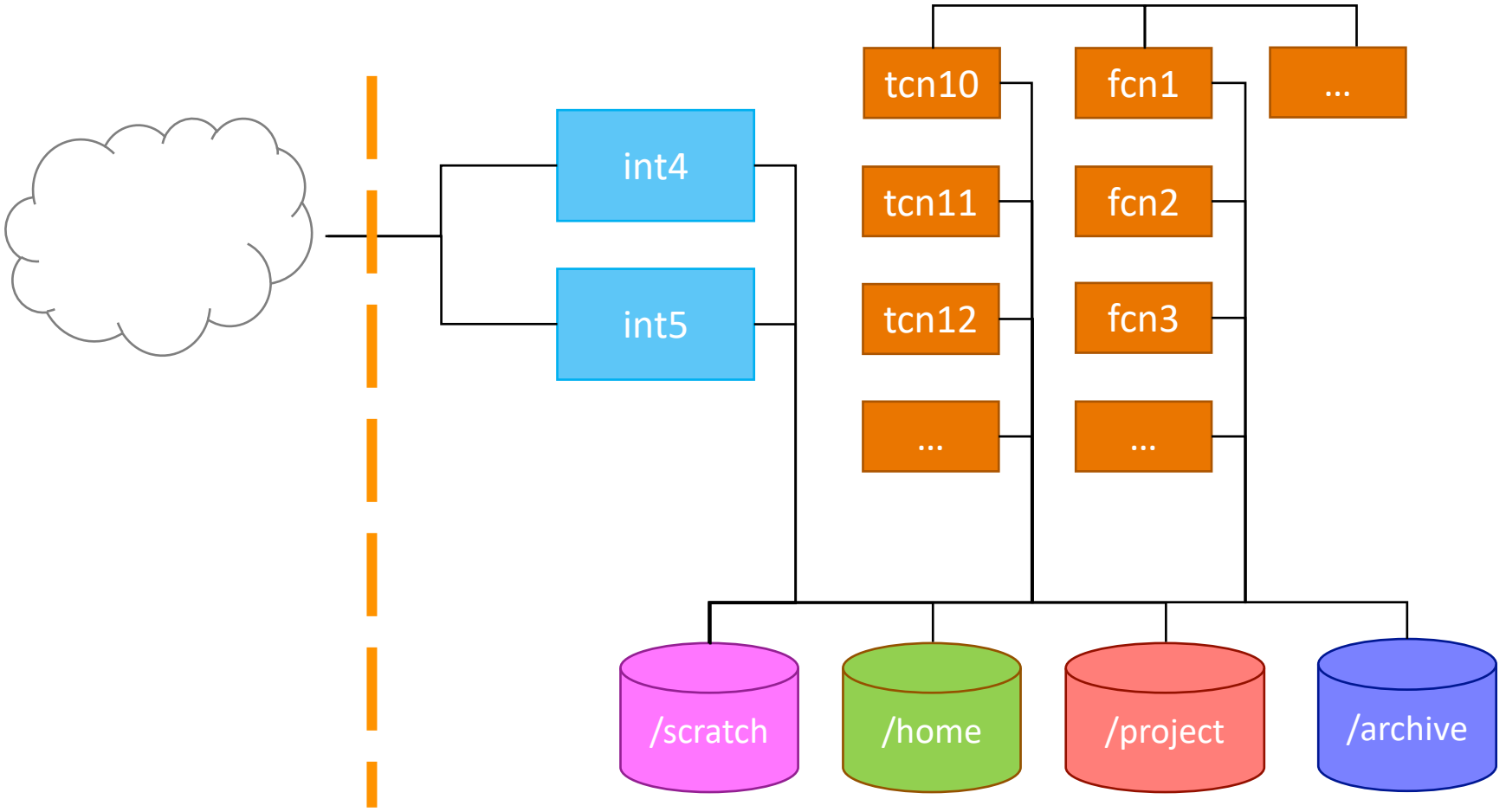
Schematic overview of a supercomputer



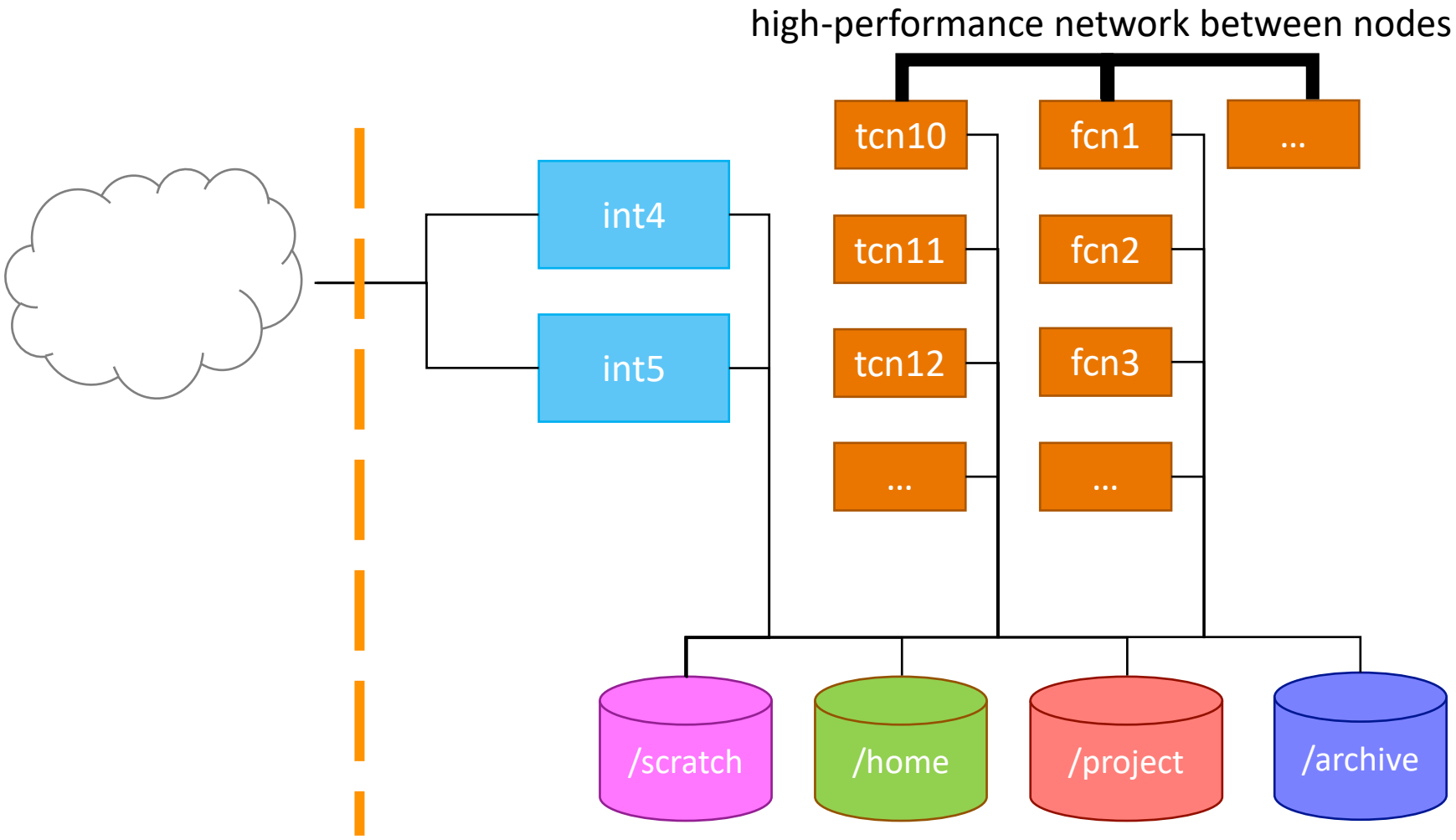
Specific example: architecture of our old Lisa cluster



Specific example: Snellius architecture



Specific example: Snellius architecture

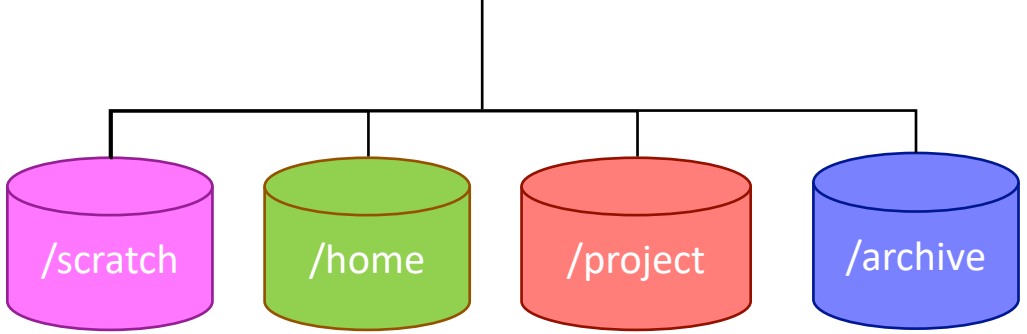


Compute power on Snellius (Phases 1 + 2)

Partition	Number of nodes	Memory per node	Sockets per node	Cores per socket	CPU model	CPU clock	Accelerator	Accelerator memory
rome	522	256 GB	2	64	AMD Rome 7H12	2.6 GHz		
genoa	785	384 GB	2	96	AMD Genoa 9654	2.4 GHz		
fat	72	1 TB	2	64	AMD Rome 7H12	2.6 GHz		
himem_4tb	2	4 TB	2	64	AMD Rome 7H12	2.6 GHz		
himem_8tb	2	8 TB	2	64	AMD Rome 7H12	2.6 GHz		
gpu	70	512 GB	2	36	Intel Xeon Platinum 8360Y	2.5 GHz	4x NVIDIA A100	4x 40 GB
staging	10	256 GB	2	8	AMD EPYC 7F32	3.2 GHz		
login	3	Login nodes may have different configurations.						

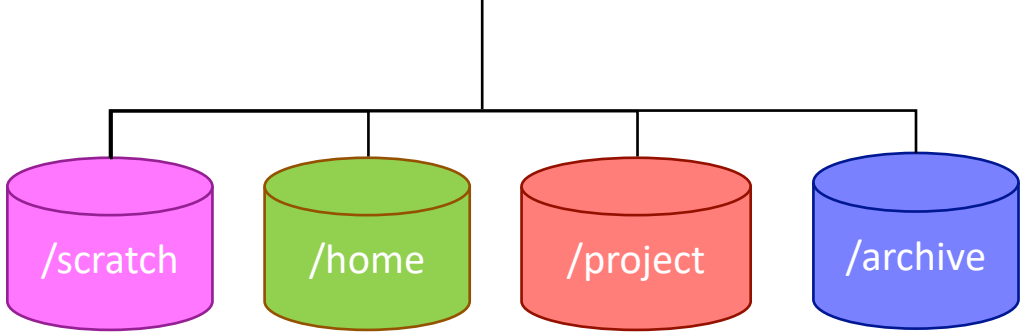
- 232,464 cores + 280 GPUs: **14.9 Pflop/s** (peak performance) + **1,421** TB memory.
- Low-latency interconnection network: InfiniBand HDR100 (100 Gb/s), fat tree.
- File systems: 720 TB for home directories and 12.4 PB for scratch and project spaces (GPFS).
- Specific policy for software installation and maintenance.

File systems on Snellius



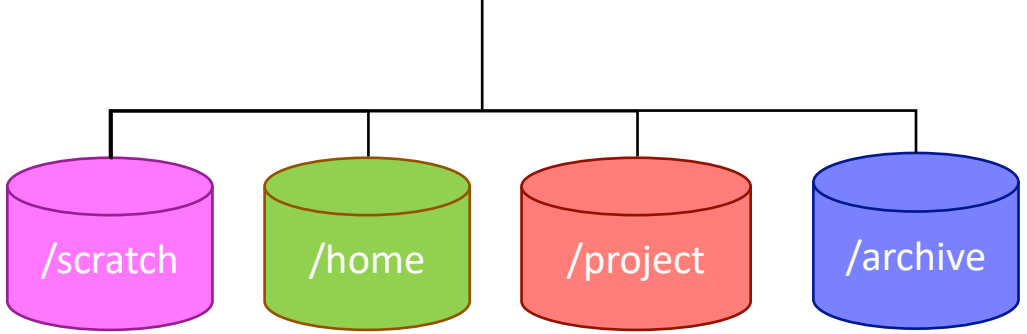
- **/home/user**
  - User home directory. Currently 200 GB.
  - Backed up.
  - Storage of important files: sources, scripts, input and output data.

File systems on Snellius



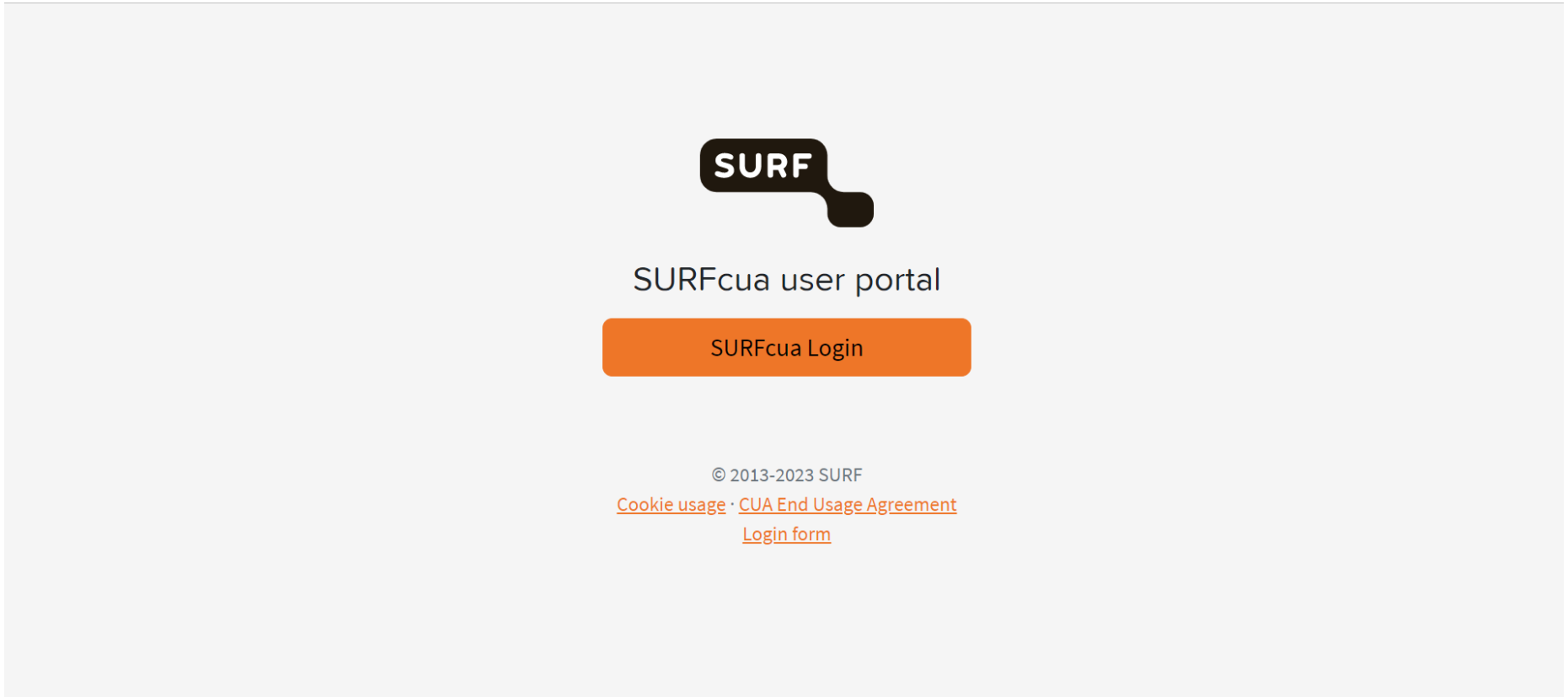
- **/scratch-***
  - **/scratch-local** & **/scratch-shared**
    - Variable quota depending on disk. Currently 8 TB guaranteed.
    - Not backed up.
    - Temporary storage. Data is removed after 6/14 days!
    - Based on GPFS: fast synchronization for parallel jobs.
  - **/scratch-node**
    - Truly node-local scratch. 6.4 TB available on some nodes (--constraint=scratch-node).
    - Not backed up.
    - Temporary storage. Data is deleted after the end of the job!
    - Based on NVMe SSD: fastest file system.

File systems on Snellius



- **/project**
  - Large and fast. Given upon request for projects requiring large, shared, permanent space.
  - Not backed up, but permanent until the end of the associated project.
  - Based on GPFS: comparable in speed with /scratch-local & /scratch-shared.
- **/archive**
  - Connected to the tape robot. Quota on demand, virtually unlimited.
  - Given upon request for long-term storage of files (in compressed format).
  - Backed up.
  - Slow—especially to retrieve “old” data—and not available on compute nodes.

Before using the system, connect to the user portal!



https://portal.cua.surf.nl

Connecting to Snellius

- Windows operating system
  - MobaXterm (recommended): https://mobaxterm.mobatek.net/
    - PLEASE DOWNLOAD THE PORTABLE EDITION !!!
  - Putty
- MacOS
  - Terminal (preinstalled)
  - XQuartz (http://www.xquartz.org)
- Linux
  - You are already well equipped!

Connecting to Snellius

- When you log in with *ssh*, you access the login nodes
- Alternative for untrusted connections: ssh <username>@doornode.surfsara.nl

```
user@local:~$ ssh scur0000@snellius.surf.nl
Password:
scur0000@int2:~$ ls
snellius-file.txt
```

- With *scp* you can transfer files to/from your local machine

```
user@local:~$ ls
local-file.txt
user@local:~$ scp local_file.txt scur0000@snellius.surf.nl:
user@local:~$ scp scur0000@snellius.surf.nl:snellius_file.txt .
user@local:~$ ls
snellius-file.txt    local-file.txt
user@local:~$ ssh scur0000@snellius.surf.nl
Password:
scur0000@int2:~$ ls
snellius-file.txt    local-file.txt
```

Outline

- Introduction to High Performance Computing
  - Definitions
  - Parallel programming
- SURF facilities
  - Presentation
  - Systems and specifications
- Running jobs
  - Definitions for jobs
  - Hands-on exercises with the files available in your home directories

Running jobs: how-to guide

- Schedulers distribute work to *batch nodes*
- Workflow:
  - 1. **You** upload your data from your computer to the cluster system
  - 2. **You** create a job script with the work steps
  - 3. **You** submit the job script to the scheduler
  - 4. **The scheduler** looks for available computers to run your work
  - 5. When a batch node with the requirements you specified becomes available, your work runs
  - 6. When the job is finished, **you** download the results to your computer
- Batch scheduler on Snellius: SLURM (http://slurm.schedmd.com)

Running jobs: useful commands of the SLURM scheduler

- sbatch <*jobscript*> - submit a job to the scheduler
- squeue -j <*job_id*> - inspect the status of job <*job_id*>
- squeue –u <*user_id*> - inspect all jobs of user <*user_id*>
- scancel <*job_id*> - cancel job <*job_id*>
- scontrol show job <*job_id*> - show estimated job start

Running jobs: first example

```
#!/bin/bash
#SBATCH --job-name="firsttest"
#SBATCH --nodes=1
#SBATCH --ntasks=10
#SBATCH --time=00:01:00
#SBATCH --partition=rome

echo "Who am I?"
whoami
echo

echo "Where ?"
srun hostname
echo

sleep 120

date
echo "DONE"
```

- Create a text file with *exactly* the first lines; name the file “job.sh”
- Submit this job with “**sbatch** job.sh” and look the status with “**squeue –u *login_id***”
- Use “**scontrol show job *job_id***” to find out when your job will run
- Look at your home-directory to see what happens there; look at the files (use ***ls*** command).
- Which files were created? Look at those files.
- Try to play with email notifications! Add the following two lines together:
  - #SBATCH --mail-type=BEGIN,END
  - #SBATCH --mail-user=<your_email_address>

Running jobs: best practices

- Give the scheduler a realistic *walltime* estimate.
- Your home directory is slow. Use $TMPDIR.
- Load software modules as part of your job script—this improves reproducibility.
- Run parallel versions of your programs (and use “srun” to ask SLURM to run multi-process applications).

Anatomy of a “real-world” job script

- Job scripts consist of:
  - the “shebang” line: `#!/bin/bash`
  - scheduler directives
  - command(s) that load software modules and set the environment
  - command(s) to prepare the input
  - command(s) that run your main task(s)
  - command(s) to save your output

```
#!/bin/bash

#SBATCH --job-name="firsttest"
#SBATCH --nodes=1
#SBATCH --ntasks=10
#SBATCH --time=00:01:00
#SBATCH --partition=rome

module load 2019
module load foss/2018b

cp -r <my_folder> $TMPDIR
cd $TMPDIR

srun a.out

cp -r $TMDPDIR/* ~/results
```

Module management: useful commands

- module avail - available modules in the system
- module load <mod> - load <mod> in the shell environment
- module list - show a list of all loaded modules
- module unload <mod> - remove <mod> from the environment
- module purge - unload all modules
- module whatis <mod> - show information about <mod>

Running jobs: second example

```
#!/bin/bash
#SBATCH --job-name="r_test"
#SBATCH --nodes=1
#SBATCH --ntasks=10
#SBATCH --time=00:01:00
#SBATCH --partition=rome

<< LOAD MODULES >>
echo "START"
<< CALL THE R SCRIPT >>
echo "DONE"
```

- Could you modify the initial script to prepare it as a template for the next session with R???
  - You will need to add the needed modules and also include the call to the command “Rscript”
    - Sample hint: Rscript -e "rnorm(1)"
- Explore the module environment to find the modules (module avail, module load…)
- Remember that you can submit this job with “**sbatch** job.sh” and look the status with “**squeue –u *login_id***”
- Can you see the output of the call?

Running jobs: third example

```
#!/bin/bash
#SBATCH --job-name="pi"
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=32
#SBATCH --time=00:10:00
#SBATCH --partition=rome

module purge
module load 2022
module load GCCcore-11.3.0

echo "OpenMP parallelism"

for ncores in `seq 8 4 48`
do
  export OMP_NUM_THREADS=$ncores
  echo "CPUS: " $OMP_NUM_THREADS
  echo "CPUS: " $OMP_NUM_THREADS >&2
  ./pi
  echo "DONE "
done
```

- Check the file “python.sh” in your home directory:
  - linux-cluster-computing/cluster/batch
- Submit this job with “**sbatch** python.sh” and look the status with “**squeue –u *login_id***”
- If you needed to use some input file or you would generate an output file… where would you put the copy commands for scratch?
- Now try the same with “pi.sh”… but first compile the code! (./compilepi)
- Can you play around with the variable ‘ncores’ and see some parallel efficiency?

Everything about jobs: service desk info pages

- https://servicedesk.surf.nl/wiki/display/WIKI/Snellius

- More courses by SURF (for research and more):
  - EuroCC Netherlands Agenda
    - https://eurocc-netherlands.nl/calendar/category/training-en/
  - SURF Agenda
    - https://www.surf.nl/en/agenda
  - SURF training mailing list
    - https://lists.surfsara.nl/listinfo/training-announce

THANK YOU FOR YOUR ATTENTION

SURF Service Desk

servicedesk.surfsara.nl

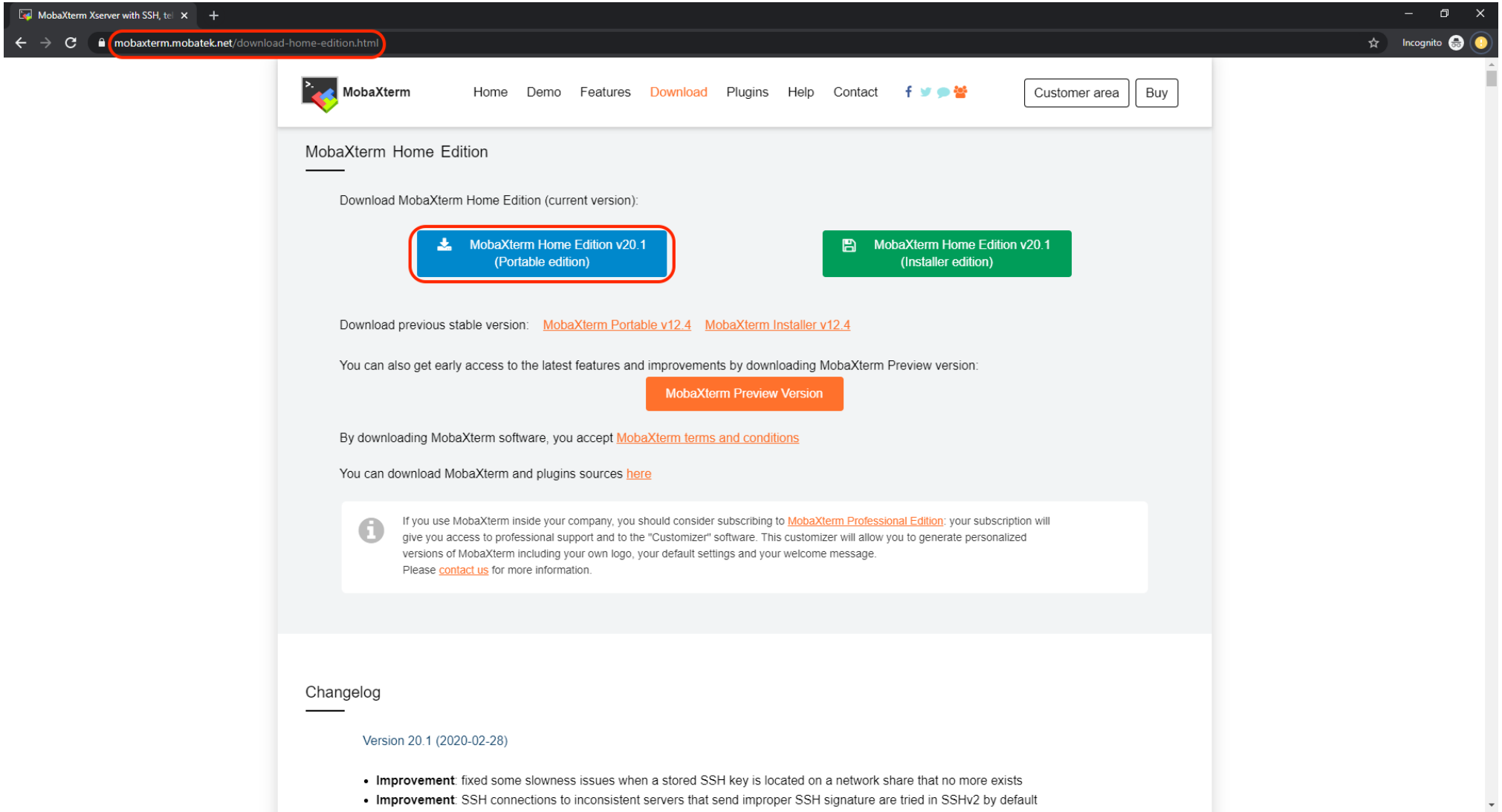
@SURF_onderzoek

Driving innovation together

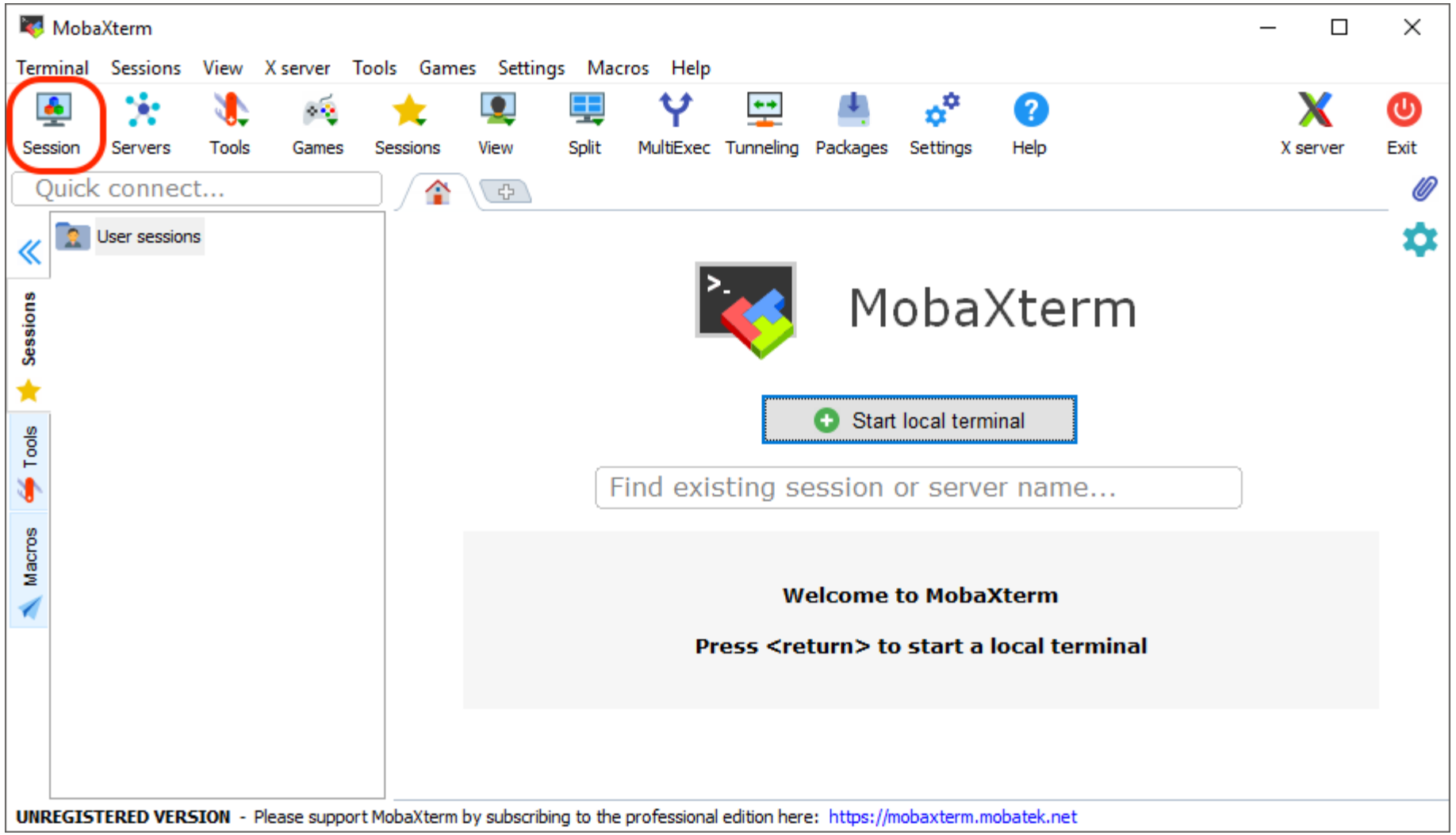
INTRODUCTION TO SUPERCOMPUTING

Backup slides for MobaXterm

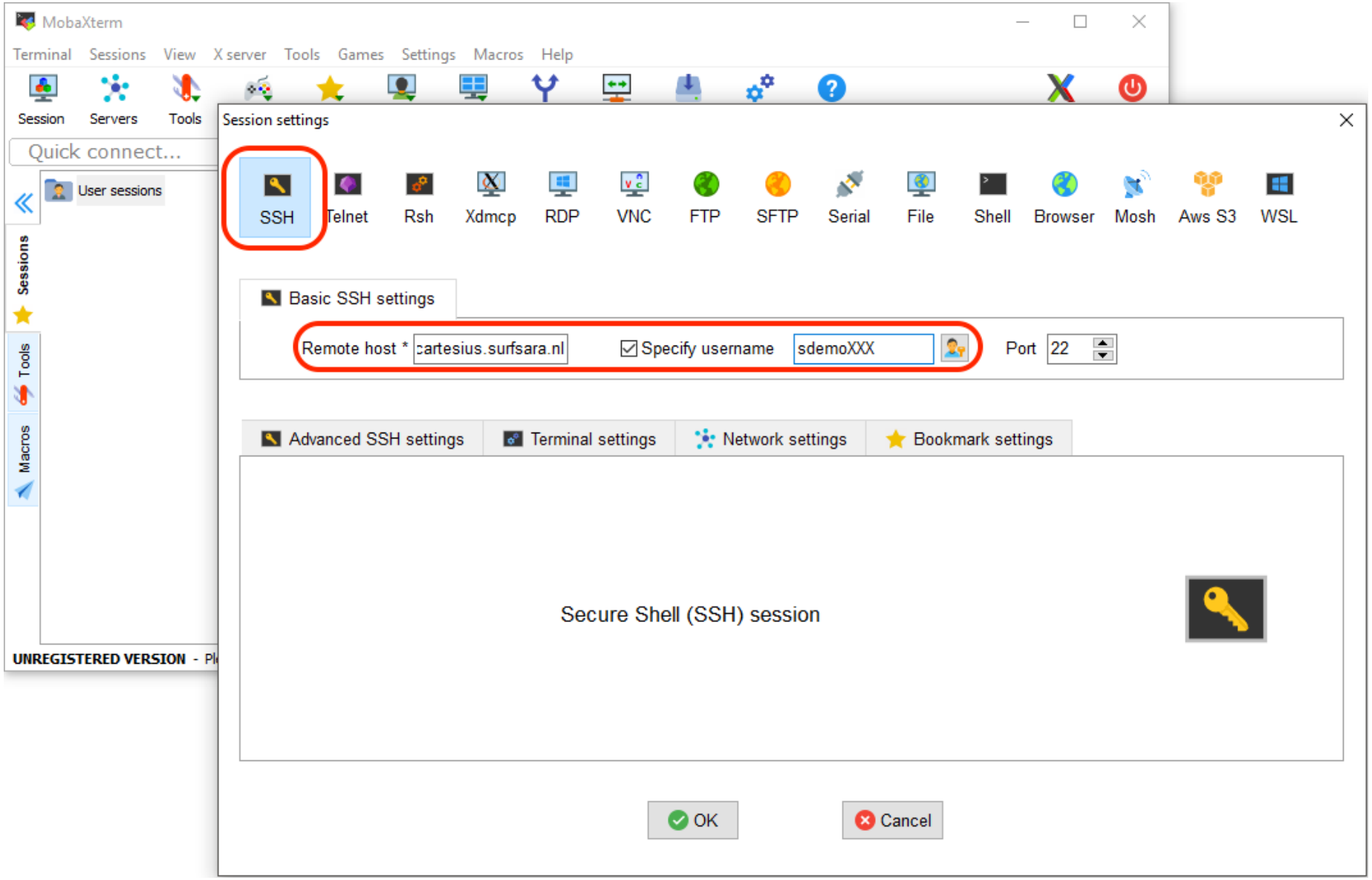
Download the portable edition of MobaXterm



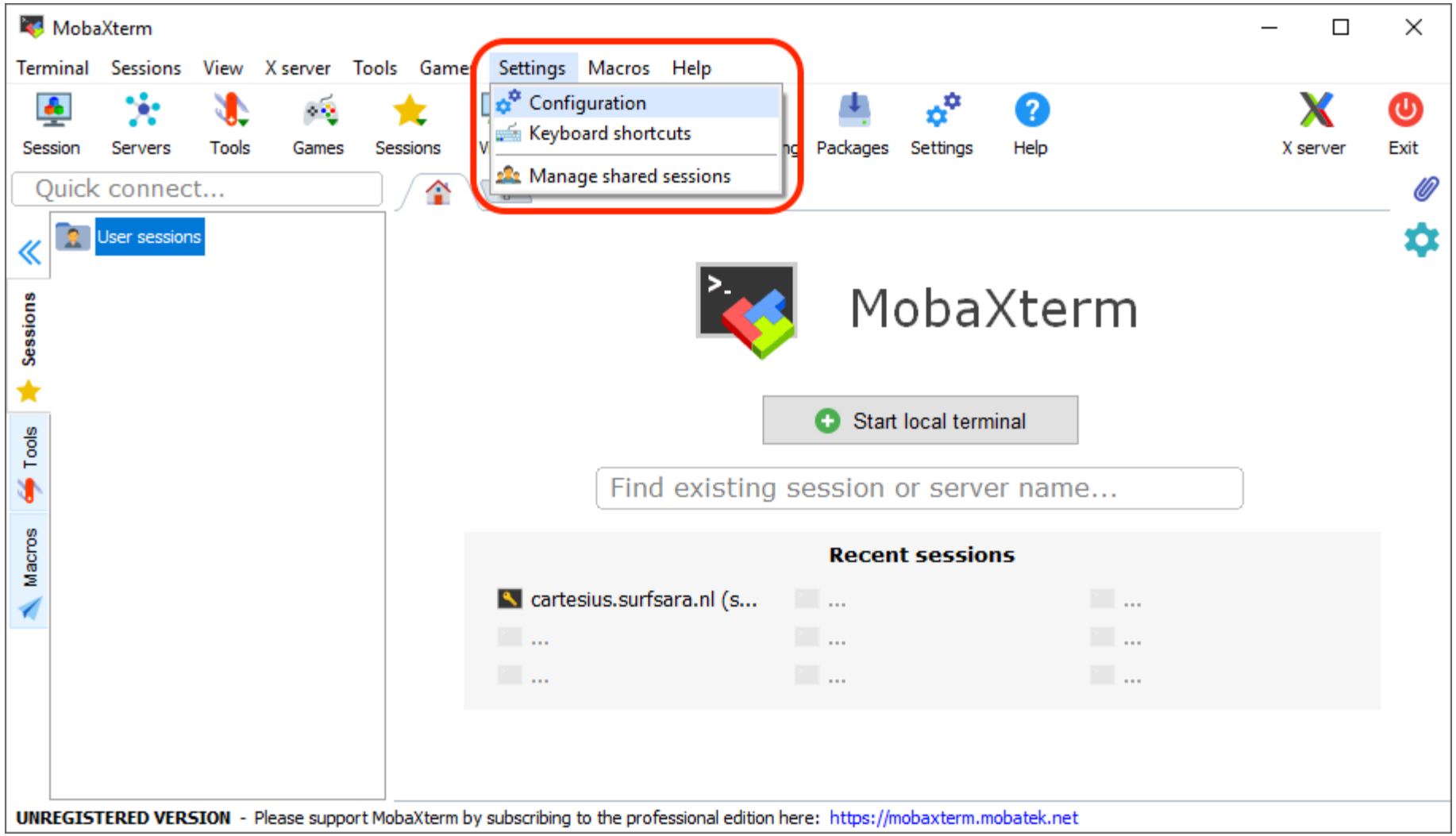
Open a new remote session



Open a new remote session



Change the root/home folders (via Settings->Configuration):



Change the root/home folders (via Settings->Configuration):