Introduction to Cluster Computing

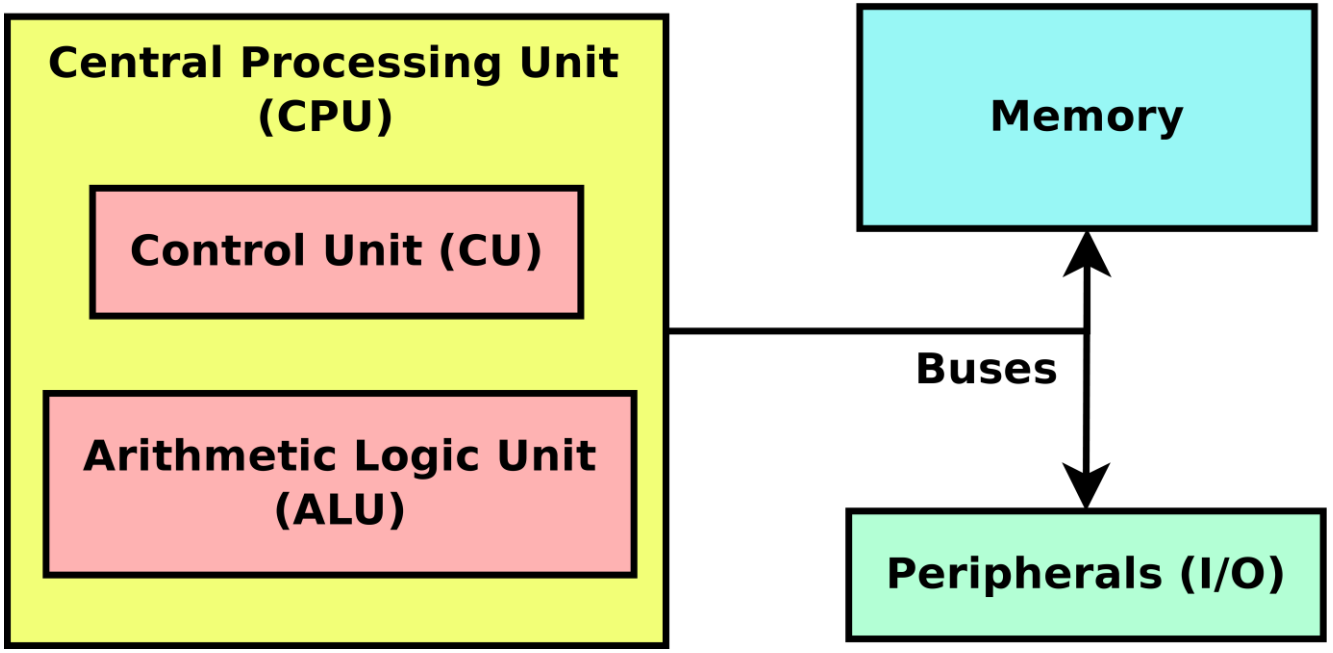
UVA – AMSTERDAM – 01.11.2019

Carlos Teijeiro Barjas

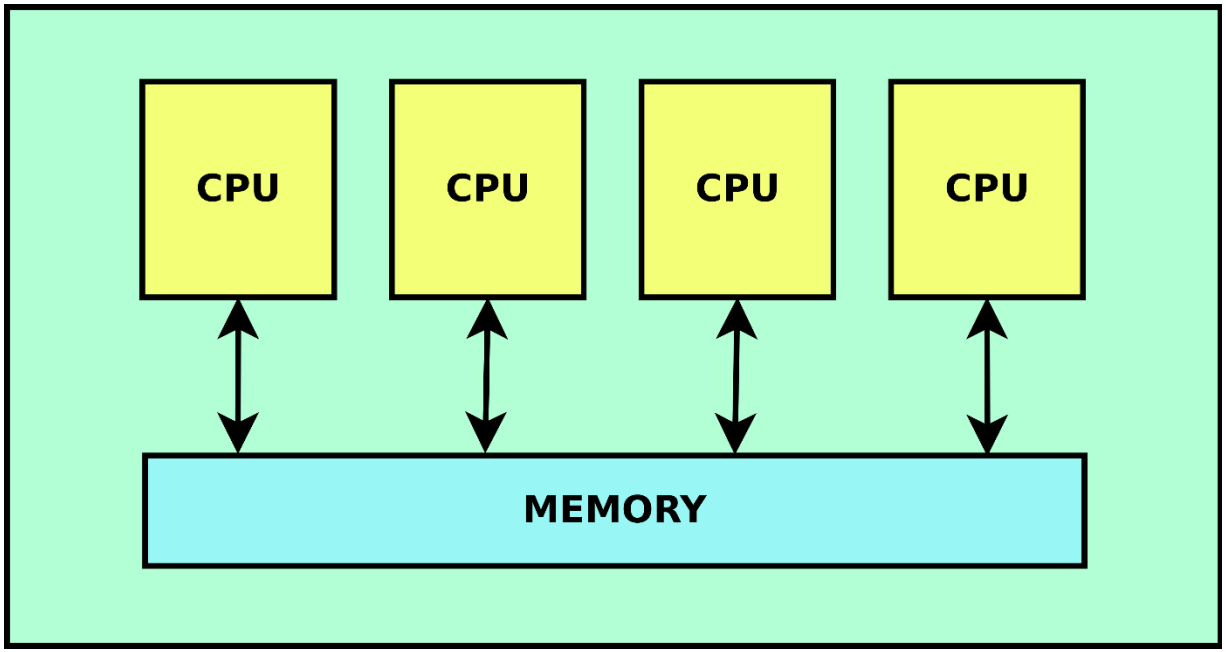
Advisor Cluster Computing carlos.teijeiro@surfsara.nl

SURF SARA

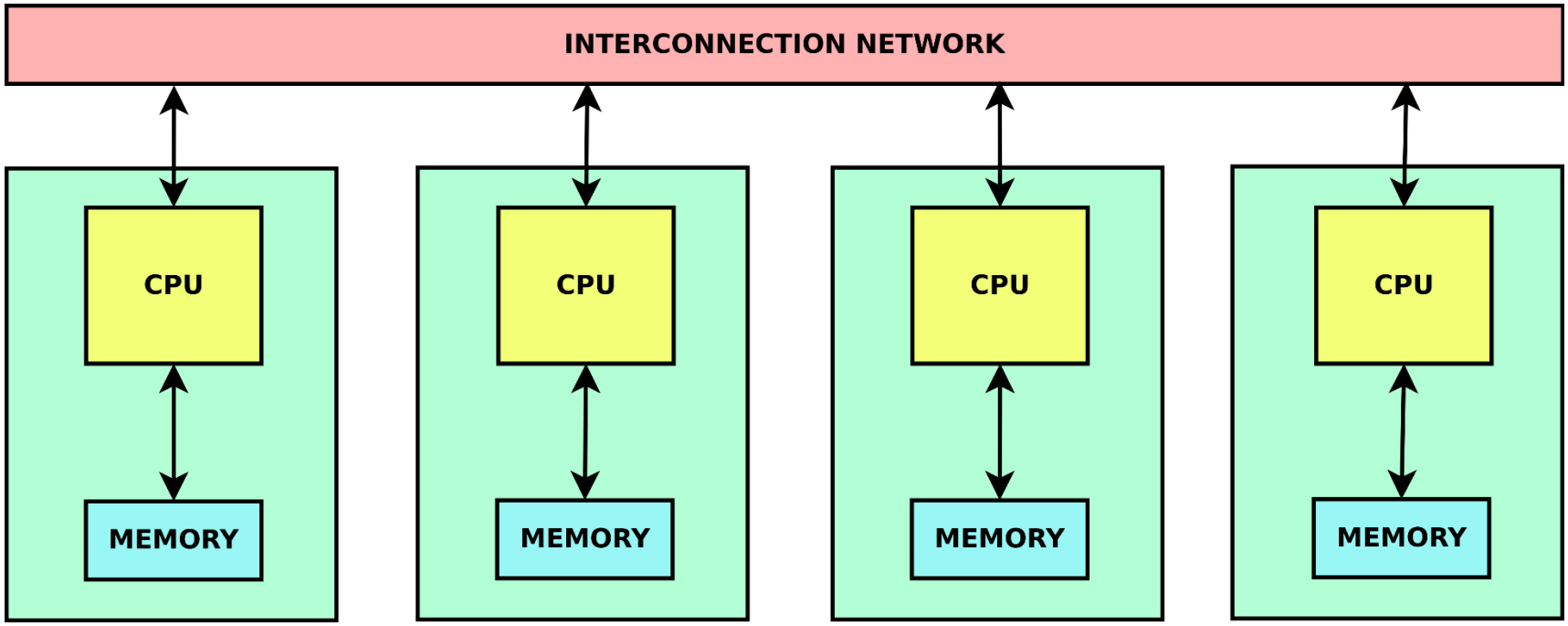
A computer is…



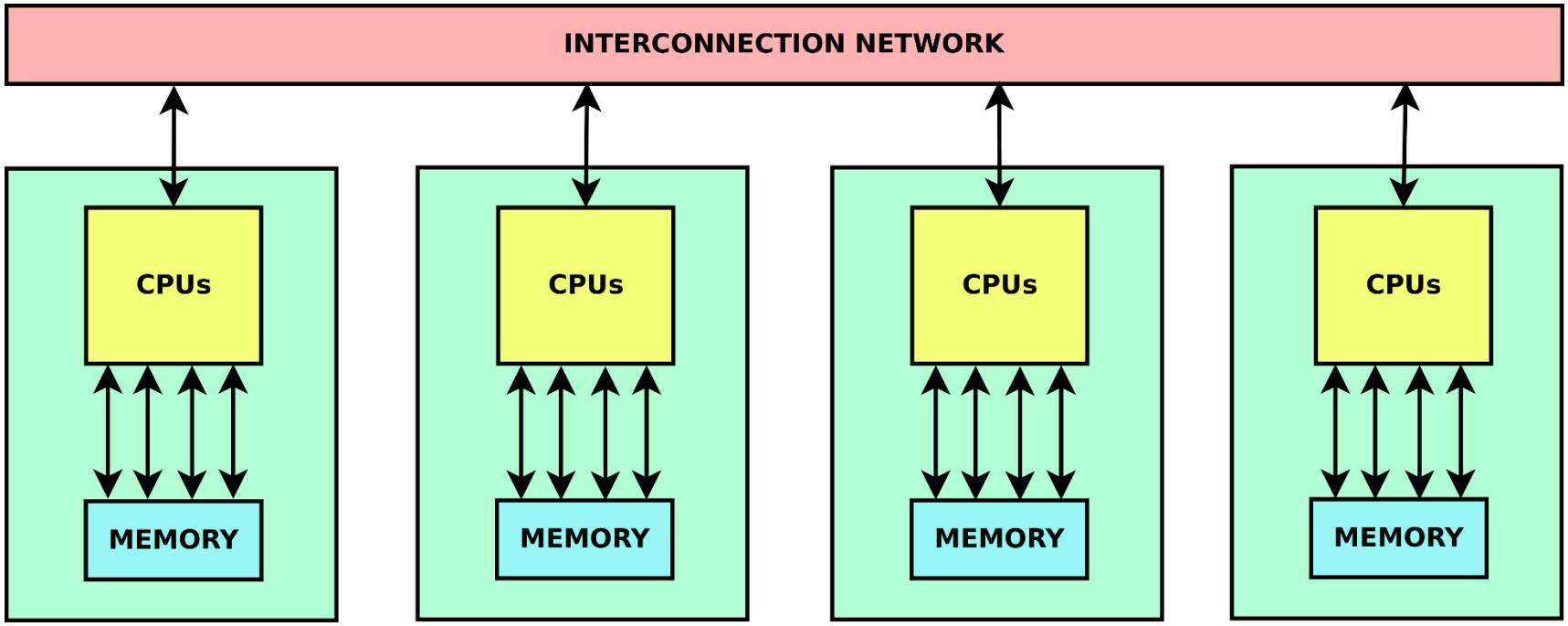
A larger computer usually is…



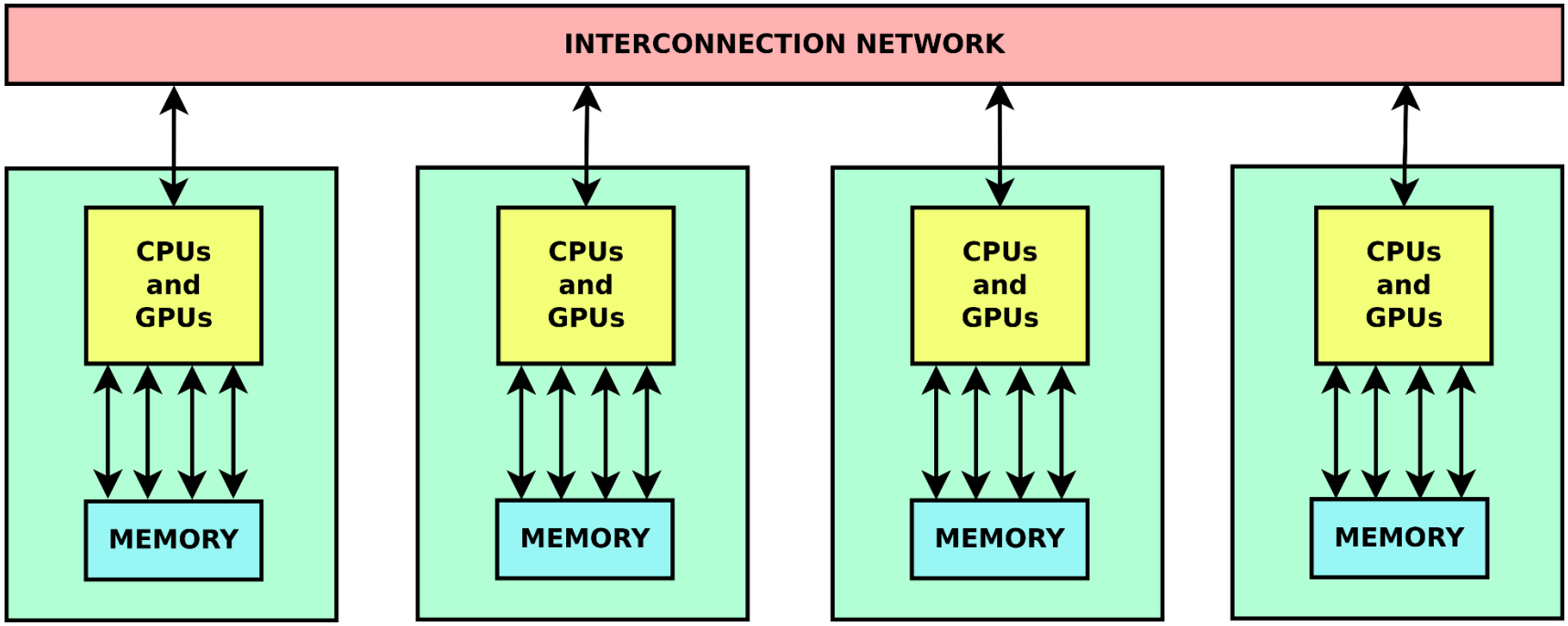
A larger computer usually is…



A larger computer usually is…



A larger computer usually is…



Outline

- **SURFsara facilities**
- **Running jobs with SLURM**
- **Hands-on exercises on the Lisa GPU island**

Outline

- **SURFsara facilities**
- **Running jobs with SLURM**
- **Hands-on exercises on the Lisa GPU island**

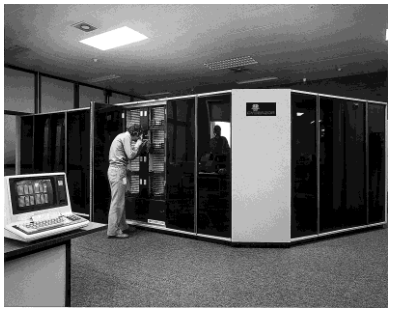
What do we do at SURFsara?

- **Regular user support for the supercomputers Cartesius and Lisa**
  - Typical effort: from a few minutes to a couple of days
- **Application enabling for Dutch Compute Challenge Projects**
  - Potential effort by SURFsara staff: 1 to 6 person months per project
- **Performance improvement of applications**
  - Typically meant for promising user applications
  - Potential effort by SURFsara staff: 3 to 6 person months per project
- **Support for PRACE applications**
  - PRACE offers access to European systems
  - SURFsara participates in PRACE support in application enabling
- **Visualization projects**
- **Training and workshops on demand**

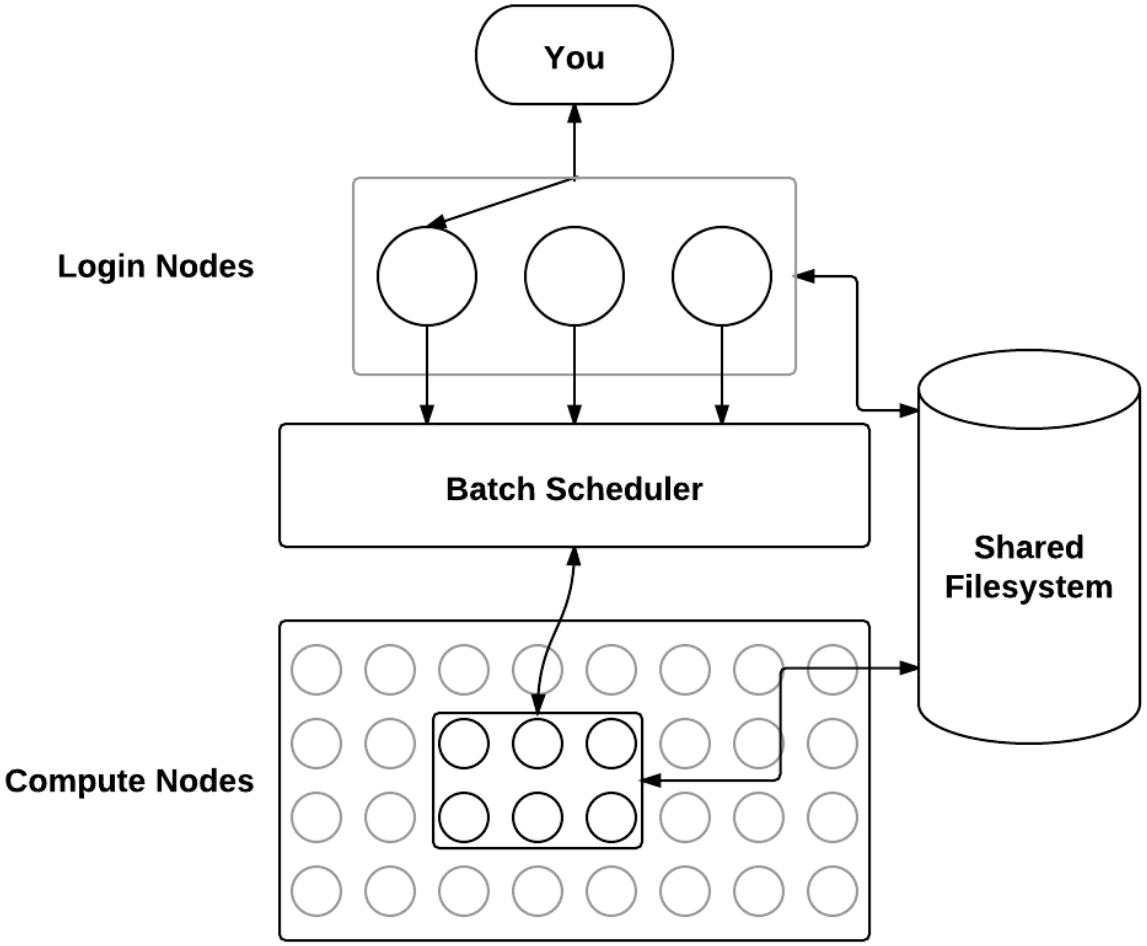
- **Please contact SURFsara at helpdesk@surfsara.nl**

Historical performance of Dutch supercomputers

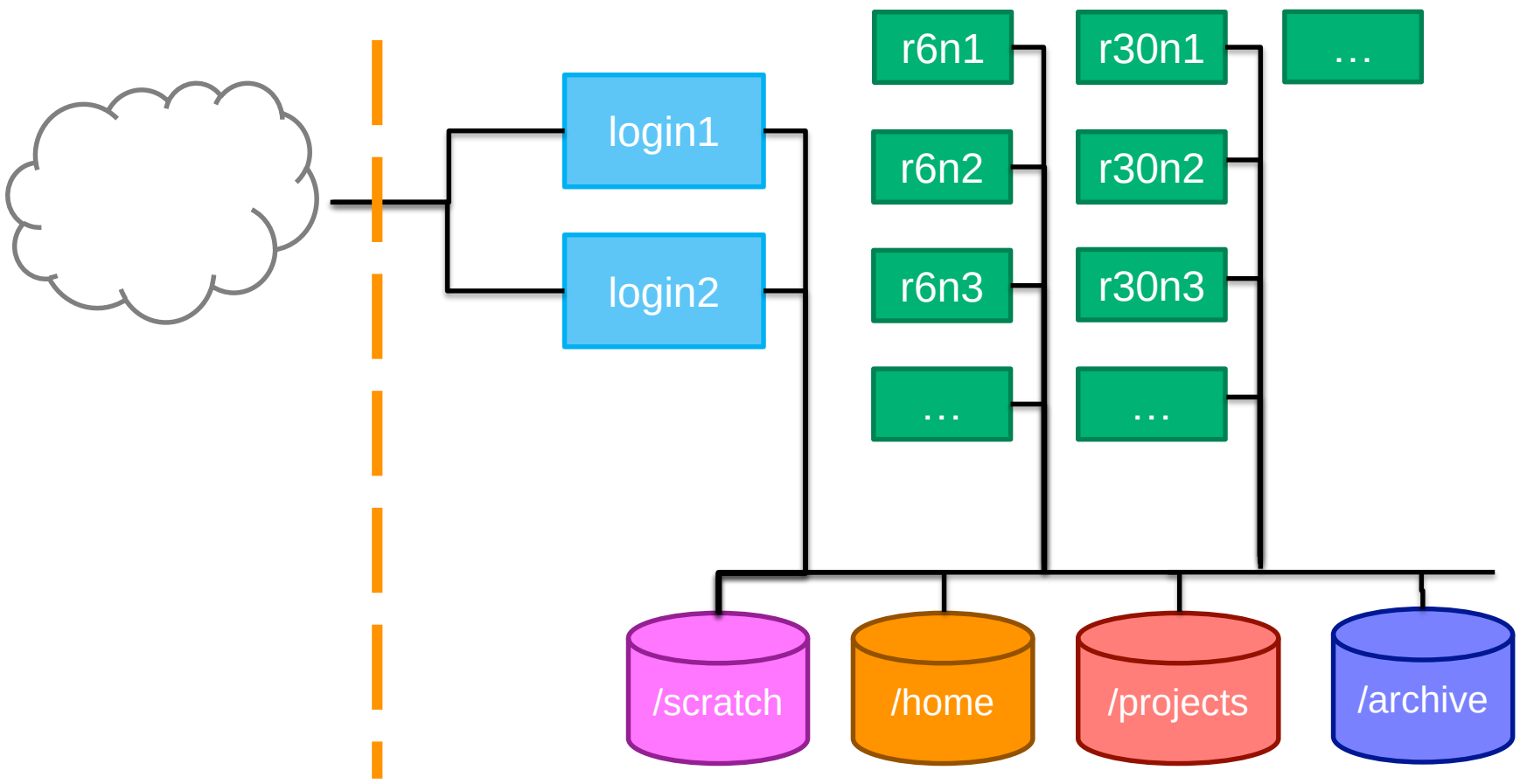
Year	Machine	R_{peak} GFlop/s	kW	GFlop/s / kW
1984	CDC Cyber 205 1-pipe	0.1	250	0.0004
1988	CDC Cyber 205 2-pipe	0.2	250	0.0008
1991	Cray Y-MP/4128	1.33	200	0.0067
1994	Cray C98/4256	4	300	0.0133
1997	Cray C916/121024	12	500	0.024
2000	SGI Origin 3800	1,024	300	3.4
2004	SGI Origin 3800 +SGI Altix 3700	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	> 1,800,000		
2016	Raspberry PI 3 (35 euro)	0.44	0.004	110



Schematic overview of Cartesius & Lisa



Specific example: Lisa architecture



Lisa – Nodes

- **The two login nodes are of type E5-2650 v2**

Number	Processor Type	Clock	Scratch	Memory	Cache	Cores	GPUs	Interconnect
32	E5-2650 v2	2.60 GHz	870 GB	32 GB QPI 8.00 GT/s	20 MB	16	-	10 Gbit/s ethernet
280	E5-2650 v2	2.60 GHz	870 GB	64 GB QPI 8.00 GT/s	20 MB	16	-	10 Gbit/s ethernet
32	E5-2650 v2	2.60 GHz	870 GB	64 GB QPI 8.00 GT/s	20 MB	16	-	Mellanox FDR
23	Bronze 3104	1.70 GHz	1.5 TB NVME	256 GB UPI 10.4 GT/s	8.25 MB	12	4 x GeForce 1080Ti, 11GB GDDR5X	40 Gbit/s ethernet
96	Silver 4110	2.10 GHz	1.8 TB	64 GB UPI 9.6 GT/s	11 MB	16	-	10 Gbit/s ethernet
1	E7-8857 v2	3.00 GHz	13 TB	1 TB QPI 8.00 GT/s	30 MB	48	-	10 Gbit/s ethernet
1	Gold 6126	2.60 GHz	11 TB	2 TB UPI 10.4 GT/s	19.25 MB	48	-	40 Gbit/s ethernet

Lisa – Nodes

CPU nodes
Total number of cores 7484
Total amount of memory 26 TB
Total peak performance 149 TFlop/sec
Disk space 400 TB for the home file systems
Mellanox InfiniBand network - FDR: 56 Gbit/sec Latency FDR: 1.3 µsec

GPU nodes
Total number of cores 276
Total amount of memory 5.9 TB
Total peak performance (SP) 1.026 TFlop/sec
Total peak performance (DP) 32 TFlop/sec
Disk space 400 TB for the home file systems
Interconnect 40 Gbit/s ethernet

Cartesius & Lisa – File systems

- **/home/user**
- User home directory (quota - currently 200GB)
- Backed up
- Meant for storage of important files (sources, scripts, input and output data)
- Not the fastest file system

- **/scratch**
- Cartesius: /scratch-local & /scratch-shared (quota – currently 8 TB)
- Lisa: /scratch (quota – depends on disk size)
- Not backed up
- Meant for temporary storage (during running of a job and shortly thereafter)
- The fastest file systems on Cartesius & Lisa

Cartesius & Lisa – File systems

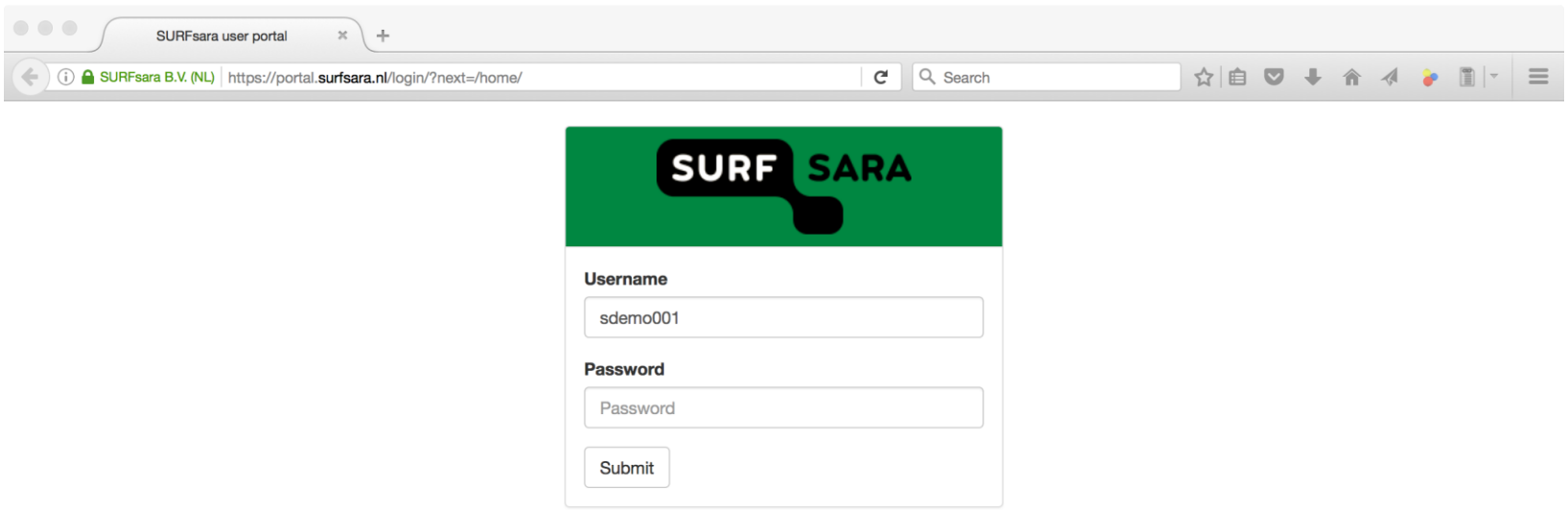
- **/archive**
- Connected to the tape robot (quota – virtually unlimited)
- Backed up
- Meant for long term storage of files, zipped, tarred, combined into small number of files
- Slow – especially when retrieving “old” data
- Not available to worker nodes

- **/project**
- On Cartesius: large and fast, On Lisa: large, but not so fast
- For special projects requiring lots of space (quota – as much as needed/possible)
- Not backed up
- Meant for special projects
- On Cartesius: comparable in speed with /scratch. On Lisa: comparable to /home.

Outline

- **SURFsara facilities**
- **Running jobs with SLURM**
- **Hands-on exercises on the Lisa GPU island**

Running jobs: first change your password



Running jobs: now connect to Lisa

- Connect to Lisa from MobaXterm or terminal

```
user@localmachine:~$ ssh lgpu0XXX@lisa.surfsara.nl
Password:
lgpu0XXX@login1:~$
```

- If you are not familiar with Linux commands, try to follow some tutorial. For example:

https://swcarpentry.github.io/shell-novice/

Running jobs: how-to

Remember: when you log in, you log in a *login node.*

The SLURM scheduler will 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 finishes, you can get an e-mail to inform you
7. When the job is finished, **you** download the results to your computer

Running jobs: first example

```
#!/bin/bash
#SBATCH -t 0:02:00
#SBATCH -n 1
#SBATCH -p gpu_shared_course

echo "Let's do some work"
sleep 20
echo "All work done."
```

- Create a text file with *exactly* the first lines; name the file "job.sh"
- Submit this job with "**sbatch** job.sh"
- Use "**squeue –u *username***" to look at the status
- Use "**scontrol show job *jobid***" to find out when your job will run
- Look at your home-directory to see what happens there; look at the files.
- Which files were created? Look at those files.

Running jobs: useful commands

sbatch <*jobscript*>	**- submit a job to the scheduler**
squeue <*jobid*>	**- inspect the status of job <*jobid*>**
squeue –u <*userid*>	**- inspect all jobs of user <*userid*>**
scancel <*jobid*>	**- cancel job <*jobid*> before it runs**
scontrol show job <*jobid*>	**- show estimated job start**

Running jobs: best practices

- **Give the scheduler a realistic *walltime* estimate**
- **Your home directory is slow. Use $TMPDIR: it is a temporary directory in /scratch created for your job (but then copy the results back to your /home !!!)**
- **Load software modules as part of your job script – this improves reproducibility**
- **Run parallel versions of your programs**

Anatomy of a job script

Job scripts consist of:

- the “shebang” line: `#!/bin/bash`
- scheduler directives: `#SBATCH …`
- loading software modules: `module load …`
- setting environment: `export VAR=…`
- preparing input
- running your program
- saving output

Example: a real job script

```
#!/bin/bash
#SBATCH -t 0:20:00
#SBATCH -N 1 -c 24

module load python/3.5.2

cp -r $HOME/run3 $TMPDIR

cd $TMPDIR/run3
python myscript.py input.dat

mkdir -p $HOME/run3/results
cp result.dat run3.log $HOME/run3/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>**

Everything about jobs: user info pages

Go to:

https://userinfo.surfsara.nl

Click on the corresponding system:

- **Cartesius: Usage → Batch Usage (jobs)**
- **Lisa: User guide → 4. Creating and running jobs**

Outline

- **SURFsara facilities**
- **Running jobs with SLURM**
- **Hands-on exercises on the Lisa GPU island**

Hands-on: downloading the dataset

- We are going to use the CIFAR10 dataset for machine learning tests on Lisa GPUs
  - Contains 50000 training and 10000 test images for classification
  - Web page: https://www.cs.toronto.edu/~kriz/cifar.html
  - Use **`wget`** to download the dataset directly to your /home directory on Lisa using the login node

```
lgpu0XXX@login1:~$ wget \
https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
```

Hands-on: installing PyTorch & TensorFlow

- Two local (user) installations are necessary
  - Load the required modules for Python and CUDA functions
  - Export the library path to load additional libraries
  - Use pip3 to perform the local installations

```
lgpu0XXX@login1:~$ module load Python/3.6.3-foss-2017b
lgpu0XXX@login1:~$ module load cuDNN/7.0.5-CUDA-9.0.176
lgpu0XXX@login1:~$ module load NCCL/2.0.5-CUDA-9.0.176
lgpu0XXX@login1:~$ export LD_LIBRARY_PATH=/hpc/eb/Debian9/
cuDNN/7.1-CUDA-8.0.44-GCCcore-5.4.0/lib64:$LD_LIBRARY_PATH
lgpu0XXX@login1:~$ pip3 install --user torch torchvision
lgpu0XXX@login1:~$ pip3 install --user tensorflow_gpu
```

Hands-on: connecting to login-gpu on Lisa

- Now we connect to the login-gpu node of Lisa !!!
  - NOTE: the previous step can be done on this login-gpu node too, but the main login nodes of Lisa are faster for **`wget`** and local installations with **`pip`**

```
lgpu0XXX@login1:~$ logout
user@localmachine:~$ ssh lgpu0XXX@login-gpu.lisa.surfsara.nl
Password:
lgpu0XXX@login-gpu1:~$
```

Example hands-on: PyTorch

- Download the examples directory for PyTorch
- Copy the CIFAR10 dataset to the dcgan folder to run a generative adversarial network (GAN)
- Use that folder as working directory and edit your batch script there

```
lgpu0XXX@login-gpu1:~$ git clone \
https://github.com/pytorch/examples.git pytorch_examples
lgpu0XXX@login-gpu1:~$ cp cifar-10-python.tar.gz \
pytorch_examples/dcgan/
lgpu0XXX@login-gpu1:~$ cd pytorch_examples/dcgan/
lgpu0XXX@login-gpu1:~$ nano pytorch.job
```

Example hands-on: PyTorch

```
#!/bin/bash
#SBATCH --job-name=example
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=3
#SBATCH --ntasks-per-node=1
#SBATCH --time=1:00:00
#SBATCH --mem=60000M
#SBATCH --partition=gpu_shared_course
#SBATCH --gres=gpu:1

module load pre2019
module load Python/3.6.3-foss-2017b
module load cuDNN/7.0.5-CUDA-9.0.176
module load NCCL/2.0.5-CUDA-9.0.176
export LD_LIBRARY_PATH=/hpc/eb/Debian9/cuDNN/7.1-CUDA-8.0.44-
GCCcore-5.4.0/lib64:$LD_LIBRARY_PATH

srun python3 -u main.py --dataset=cifar10 --dataroot=. --cuda
```

Example hands-on: PyTorch

- Submit the job…

```
lgpu0XXX@login-gpu1:~$ sbatch pytorch.job
```

- … and now you can also try the TensorFlow example on your own!

- (the guidelines are in your home folders)

Introduction to Cluster Computing

UVA – AMSTERDAM – 01.11.2019

Carlos Teijeiro Barjas

Advisor Cluster Computing carlos.teijeiro@surfsara.nl

SURF SARA