Introduction to Cartesius

Nijmegen – 13th April 2018

Jeroen Engelberts jeroen.engelberts@surfsara.nl
Consultant Supercomputing

SURF SARA

Outline

- **SURFsara**
  - Support
- **Parallel Computing**
  - Concept
  - Shared memory
  - Distributed memory
  - Hybrid (Shared + Distributed)
- **Cartesius**
  - Architectures and Specifications
  - File systems
  - Batch system
  - Module environment
  - Stopos
- **Hands on – Let's Play!**
  - Hands on can be found here: http://github.com/jjengelberts/cartesius-demo

About SURFsara

- 1971 – founded SARA
- 1984 – first national supercomputer
- 1995 – independent
- 2008 – split Vancis / SARA
- 2012 – new name SURFsara
- 2013 – merge to cooperation SURF

Support

- **Regular user support**
  - 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**
- **User training and workshops**
- **Please contact SURFsara at helpdesk@surfsara.nl**

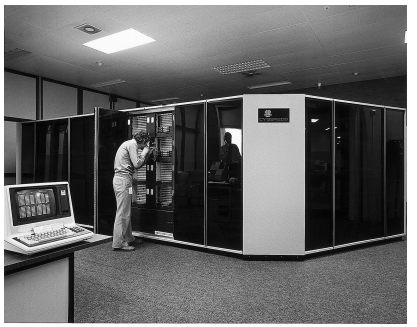
- **NB – Coaching for (master) students**

Supercomputer Cartesius

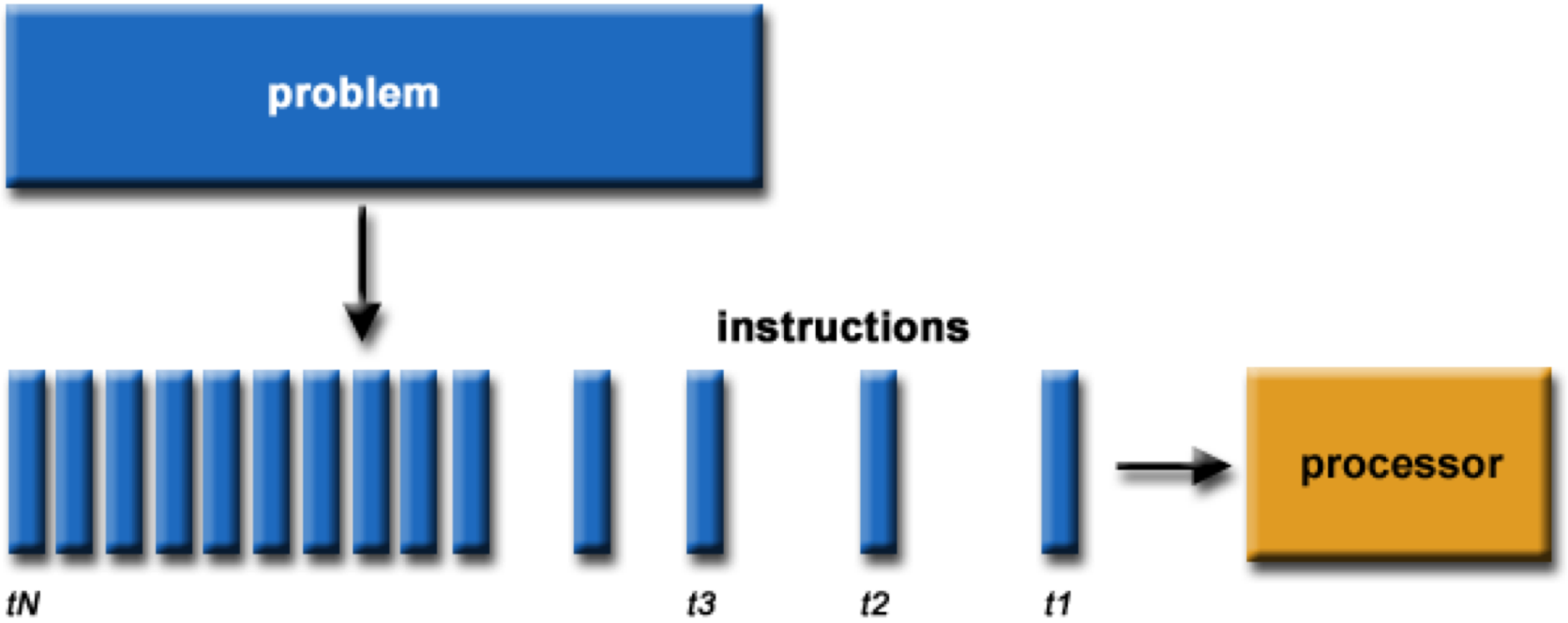
- **What is a Supercomputer?**
  - A fast computer
  - A large computer (memory/storage)
  - An expensive computer (millions of € for hardware, electricity and man power)
- **Why, or more, when do you need a Super?**
  - If your task would take months/years on a normal PC
  - If your task requires more space (memory/storage) than available in PC
- **What is the difference?**
  - Cartesius – larger “blocks” (capability computing – fewer large scale jobs)
  - LSG – smaller “blocks” (capacity computing – more small(er) scale jobs)

Performance Increase

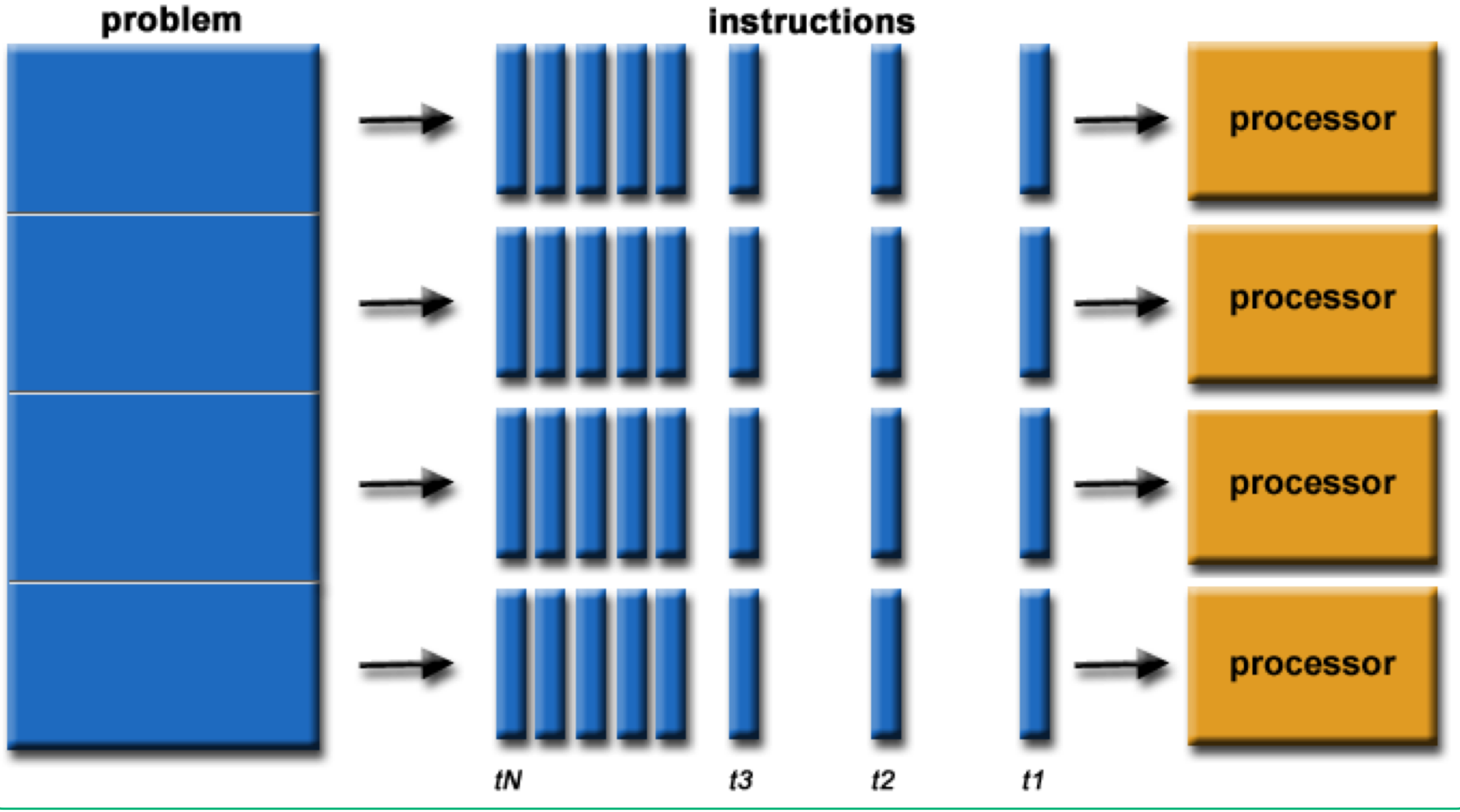
Year	Machine	R_{peak} GFlop/s	kW	GFlop/s / kW
1984	CDC Cyber 205 1-pipe	0.1	250	0.0004
1988	CDC Cyber 205 2-pipe	0.2	250	0.0008
1991	Cray Y-MP/4128	1.33	200	0.0067
1994	Cray C98/4256	4	300	0.0133
1997	Cray C916/121024	12	500	0.024
2000	SGI Origin 3800	1,024	300	3.4
2004	SGI Origin 3800 +SGI Altix 3700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



Serial

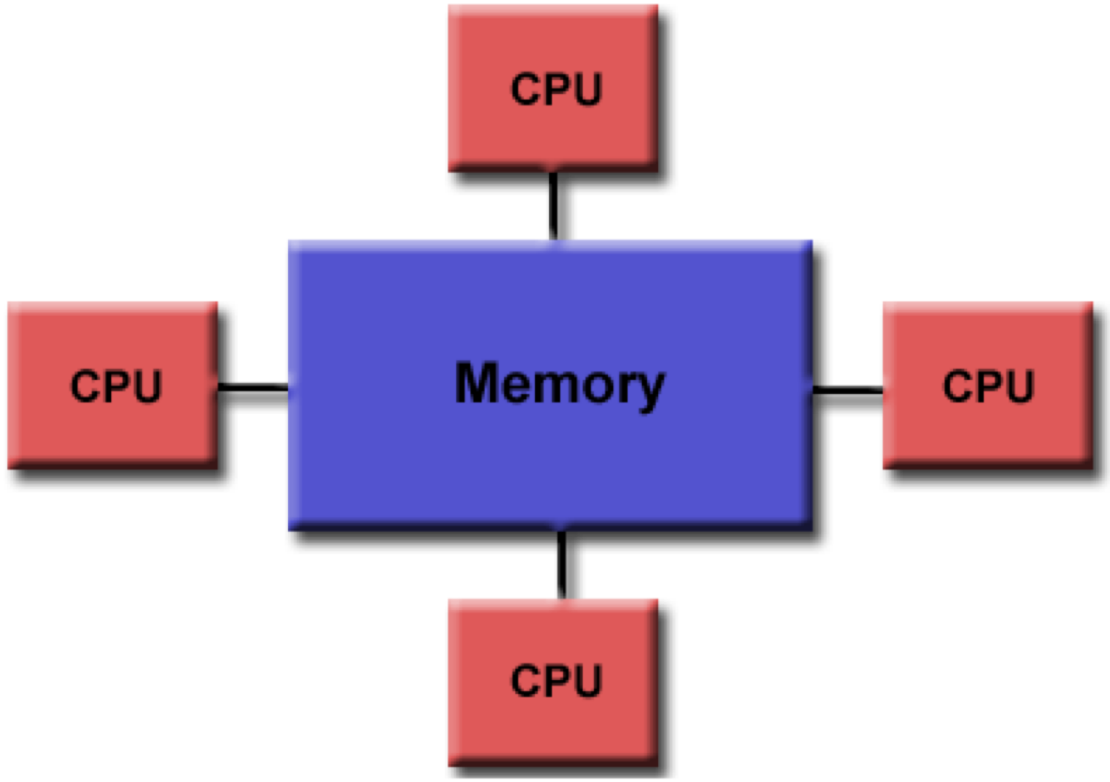


Parallel

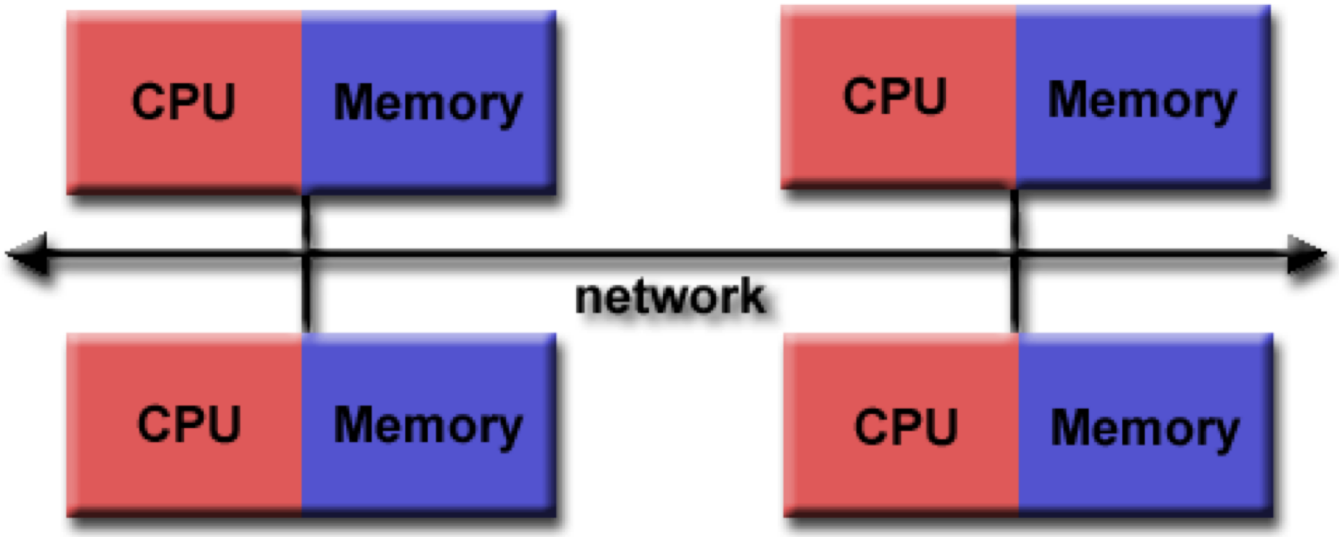


Author: Blaise Barney, Lawrence Livermore National Laboratory

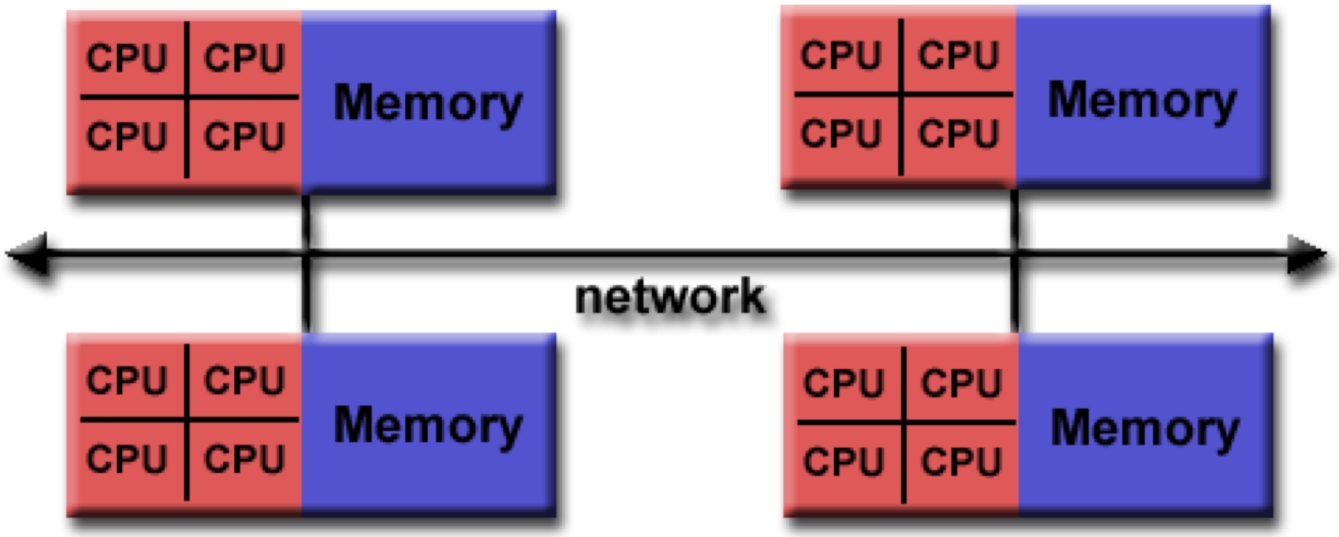
Shared Memory



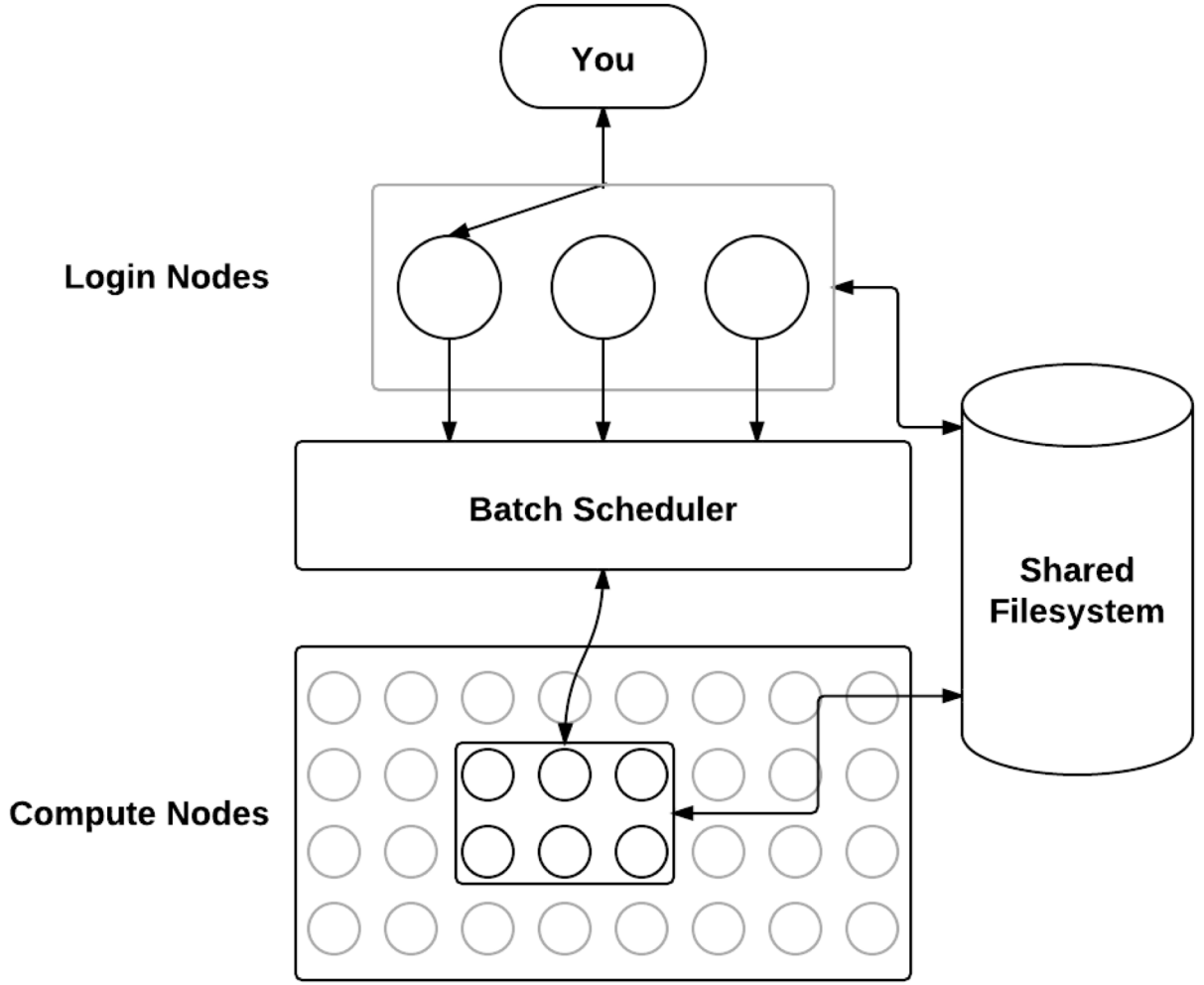
Distributed Memory



Hybrid – Shared & Distributed



Schematic overview of Cartesius



Cartesius – Login / Service / Fat & Thin

- **2 bullx R423-E3 interactive front end nodes (int1 and int2)**
  - 2 × 8-core 2.9 GHz Intel Xeon E5-2690 (Sandy Bridge) CPUs/node
  - 128 GB/node
- **5 bullx R423-E3 service nodes**
  - 2 × 8-core 2.9 GHz Intel Xeon E5-2690 (Sandy Bridge) CPUs/node
  - 32 GB/node
- **1 fat node island consisting of 32 bullx R428 E3 fat nodes**
  - 4 × 8-core 2.7 GHz Intel Xeon E5-4650 (Sandy Bridge) CPUs/node
  - 256 GB/node
  - 22 Tflop/s
- **2 thin node islands consisting of 540 bullx B710 thin nodes**
  - 2 × 12-core 2.4 GHz Intel Xeon E5-2695 v2 (Ivy Bridge) CPUs/node
  - 64 GB/node
  - water cooled
  - 249 Tflop/s

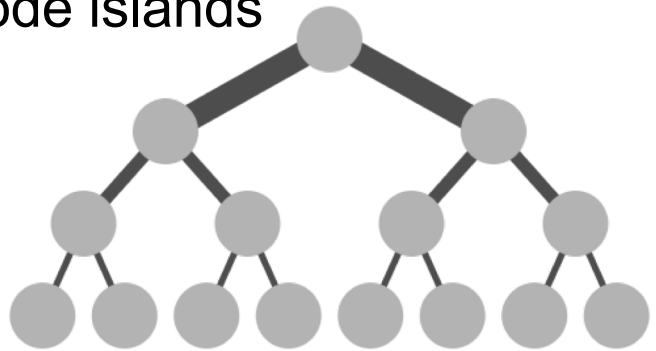
Cartesius – Thin / GPGPU

- **3 thin node islands consisting of 360 bullx B720 thin nodes**
  - 2 × 12-core 2.6 Ghz Intel Xeon E5-2690 v3 (Haswell) CPUs/node
  - 64 GB/node
  - water cooled
  - 1.078 Pflop/s
- **1 accelerator island consisting of 66 bullx B515 accelerated nodes**
  - 2 × 8-core 2.5 Ghz Intel Xeon E5-2450 v2 (Ivy Bridge) CPUs/node
  - 2 × NVIDIA Tesla K40m GPGPUs/node
  - 96 GB/node
  - 210 Tflop/s
- **1 thin node island consisting of 195 Bull sequana X1000 cells**
  - 177 sequana X1110: 2 × 16-core 2.6 GHz Intel Xeon E5-2697A v4 (Broadwell) CPUs
  - 64 GB/node | 236 Tflop/s
  - 18 sequana X1210: 1 × 64-core 1.3 GHz Intel Xeon Phi 7230 (Knights Landing) CPU
  - 96 GB/node | 48 Tflop/s
- **Total peak performance: 1843 Tflop/s**

Cartesius – other specs

Low-latency network: 4x FDR14 InfiniBand

- Non-blocking within fat node island and thin node islands
- 3.3 : 1 pruning factor among islands
- 56 Gbit/s inter-node bandwidth
- 2.4 µs inter-island latency



File systems and I/O

- 180 TB home file system
- Lustre file system for scratch and project space 7.7 PB

Operating system

- bullx Linux, compatible with Red Hat Enterprise Linux

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

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

LSG / Cartesius – batch commands

- **To submit a job**
  - LSG: qsub <jobscript>
  - Cartesius: sbatch <jobscript>

NB Submission returns a unique jobid

- **To view submitted jobs (of yourself)**
  - LSG: qstat –u <username>
  - Cartesius: squeue –u <username>

- **To cancel submitted jobs**
  - LSG: qdel <jobid>
  - Cartesius: scancel <jobid>

Modules – Why modules?

- **Why modules?**
- Environment variables are set for you, like:
  - PATH
  - LD_LIBRARY_PATH

- Multiple versions of software can coincide

Modules – Commands

Commands

- **module avail**
- **module load modulename**
- **module add modulename**
- **module display modulename**
- **module unload modulename**
- **module rm modulename**
- **module list**
- **module help**

Modules – module avail

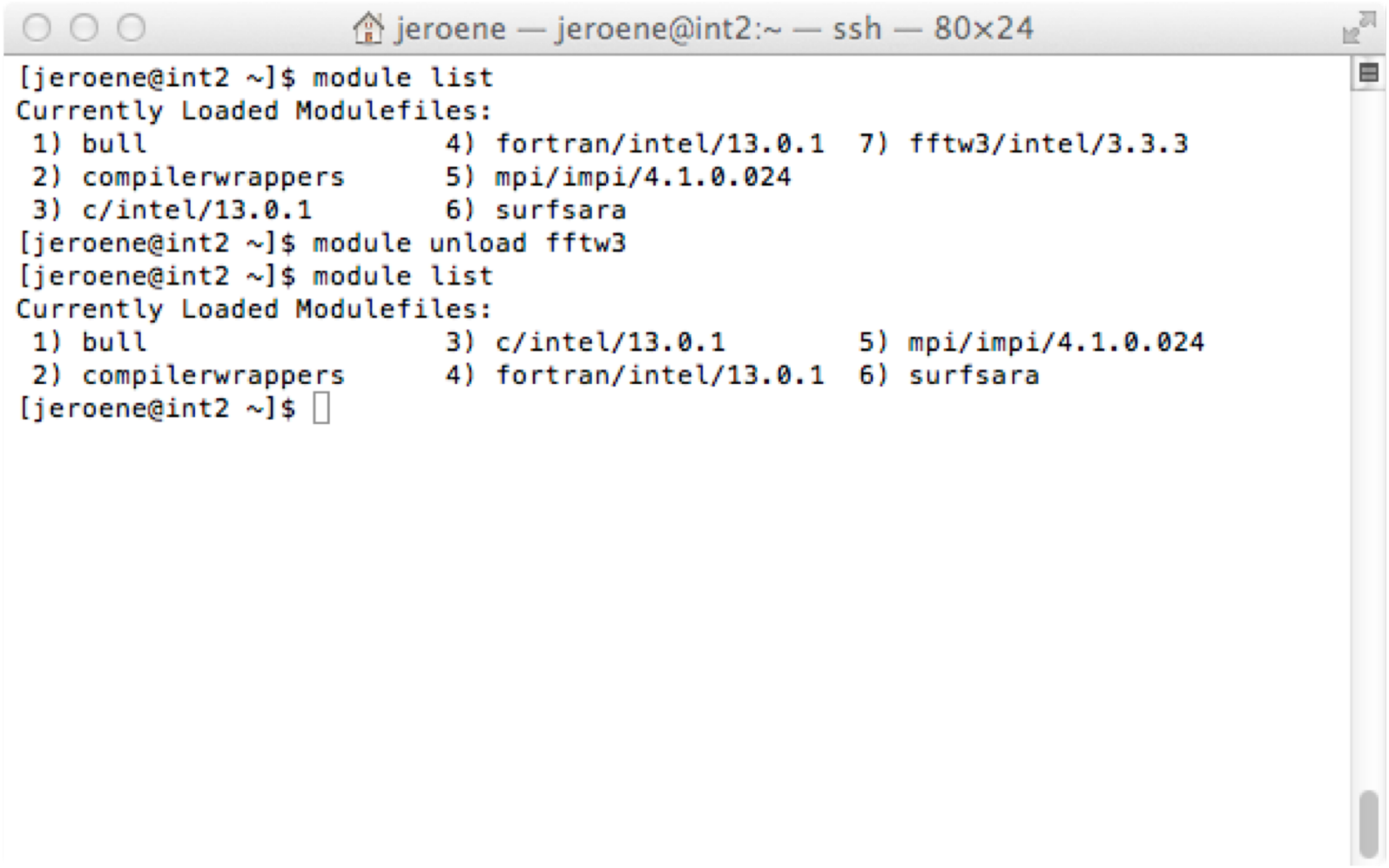
Modules – module load / display

jeroene — jeroene@int2:~ — ssh — 80×24

```
[jeroene@int2 ~]$ module load fftw3/intel
[jeroene@int2 ~]$ module display fftw3/intel
-------------------------------------------------------------------
/home/jeroene/modules/modulefiles/../modulefiles/libraries/fftw3/intel/3.3.3:

module-whatis   Activate fftw3 library
setenv  SURFSARA_FFTW3_ROOT     /hpc/sw/fftw3-3.3.3-intel-impi
setenv  SURFSARA_FFTW3_LIB      /hpc/sw/fftw3-3.3.3-intel-impi/lib
setenv  SURFSARA_FFTW3_INCLUDE  /hpc/sw/fftw3-3.3.3-intel-impi/include
prepend-path    SURFSARA_INCLUDE_PATH   /hpc/sw/fftw3-3.3.3-intel-impi/include :
prepend-path    SURFSARA_LIBRARY_PATH   /hpc/sw/fftw3-3.3.3-intel-impi/lib     :
append-path     PRACE_CFLAGS    -I/hpc/sw/fftw3-3.3.3-intel-impi/include
append-path     PRACE_FFLAGS    -I/hpc/sw/fftw3-3.3.3-intel-impi/include
append-path     PRACE_LDFLAGS   -L/hpc/sw/fftw3-3.3.3-intel-impi/lib -Wl,-R/hpc/
sw/fftw3-3.3.3-intel-impi/lib
-------------------------------------------------------------------
[jeroene@int2 ~]$
```

Modules – module list / unload



Cartesius – How to obtain Access

Take a look at the SURFsara website:
https://userinfo.surfsara.nl/systems/cartesius/account

1. Proposal to NWO
2. Filling in the forms in ISAAC
3. Peer review process
4. Approval from NWO, What next?
5. Granting letter (from NWO) → A copy to SURFsara
6. Acceptance letter (from NWO) → Fill it in and return it to NWO
7. User form (see website) → Fill in and send it to SURFsara
8. Usage agreement (see website) → each user should fill this in, sign it and send it to SURFsara

→ ***Access in two to three weeks for pilots, couple of months for regular projects***

- Pilot Cartesius: 500000 SBU
- Pilot Lisa: 100000 SBU

Stopos – Hands-On

Stopos – a tool to perform (large) parameter sweeps

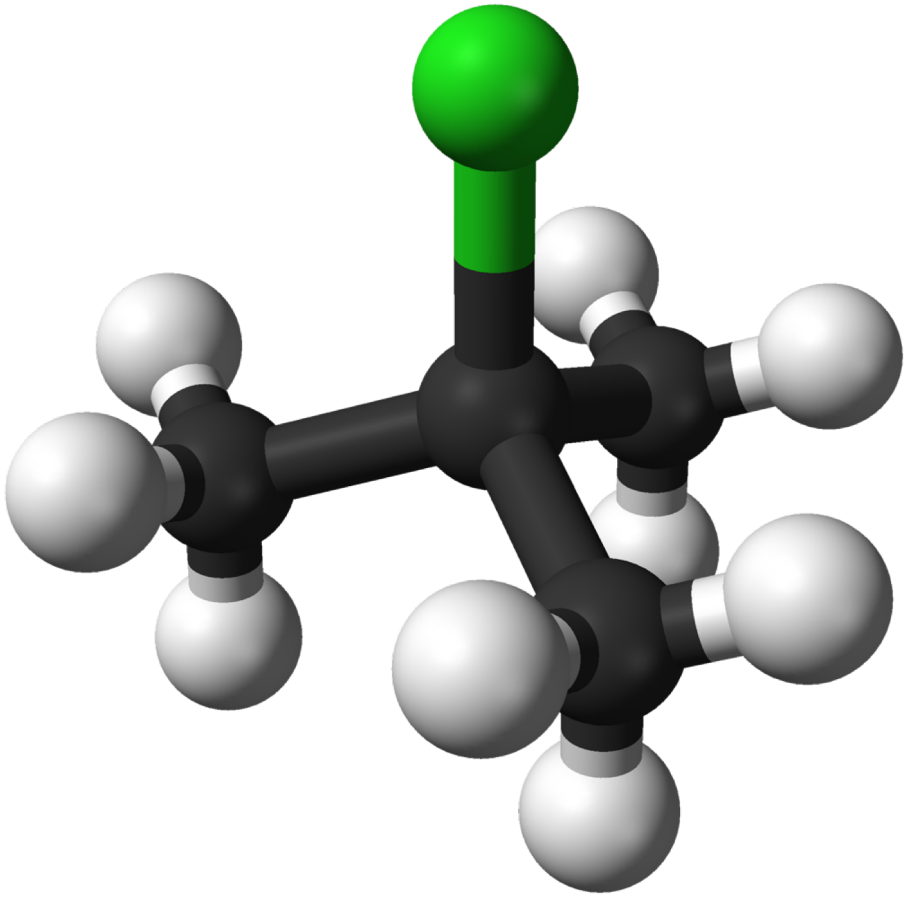
- Check https://userinfo.surfsara.nl/systems/lisa/software/stopos

Main commands:

- module load stopos – *load the Stopos module*
- stopos create -p pool1 – *create a pool called "pool1"*
- stopos -p pool1 parmset.txt – *add the set in parmset.txt to "pool1"*
- stopos pools – *view the current pools*
- stopos purge -p pool1 – *purge (trash) "pool1"*
- stopos next -p pool1 – *obtain next value of "pool1" in $STOPOS_VALUE*
- stopos remove -p pool1 – *remove used value ($STOPOS_VALUE) of "pool1"*
- stopos status -p pool1 – *show the status of "pool1"*

→ Check the example on the webpage and/or the demo presented today

Hands-On – *tert*-butylchloride – Stopos



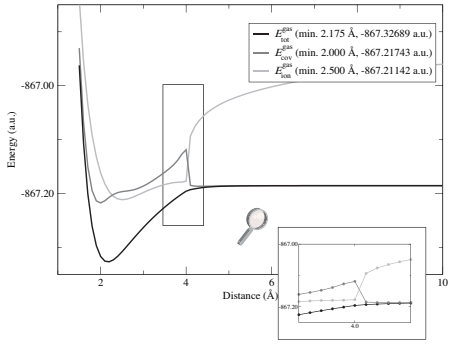


Figure 3.14: The dissociation curve of trimethylsilylchloride ($Si(CH_3)_3Cl$; **4**) in the gas phase. E_{tot}^{gas} is the total VB energy. E_{cov}^{gas} is the energy of structure **a** and E_{ion}^{gas} is the energy of structure **b**. The minima are shown in parenthesis.

distances. The bond character for *tert*-butylchloride (**2**; Figure 3.12) is also covalent, because the weight for the covalent structure is at least 0.66 over the whole range of the curve. In the chlorosilane (**3**) case (Figure 3.13) the bonding energy is 74.6 kJ/mol higher than in the chloromethane (**1**) case (Table 3.1), indicating that the Si–Cl bond is much stronger than the C–Cl bond.

For trimethylsilylchloride (**4**; Figure 3.14) the Coulombic character is more pronounced. For distances from 2.0 to 4.1 Å the ionic curve (E_{ion}^{gas}) lies below the covalent curve (E_{cov}^{gas}). The weight of the ionic structure is higher than that of the covalent structure between 2.3 and 4.1 Å, which is in line with the expectation that this bond possesses a higher ionic character. Still, the bonds in trimethylsilylchloride (**4**) and NaCl are quite different, because for the NaCl "molecule" the ionic and total curve almost coincide, while for **4** this is not the case, although the ionic curve lies lower in energy than the covalent curve near the equilibrium distance. At large distance this situation is reversed (compare Figure 3.8 with Figure 3.14). Regarding the shape of the curve, at the crossing of the ionic and covalent curves for **4** around 2.2 Å the geometry does not change drastically. Therefore, this crossing is not accompanied by any kind of discontinuity in contrast to the crossing around 4.0 Å (*vide infra*). At distances shorter than the Si–Cl equilibrium bond distance the negative charge cloud on the chlorine atom starts to overlap with the positive $Si(CH_3)_3$

Image from Wikipedia & Page 81 from PhD thesis Jeroen Engelberts

Thank you for listening!