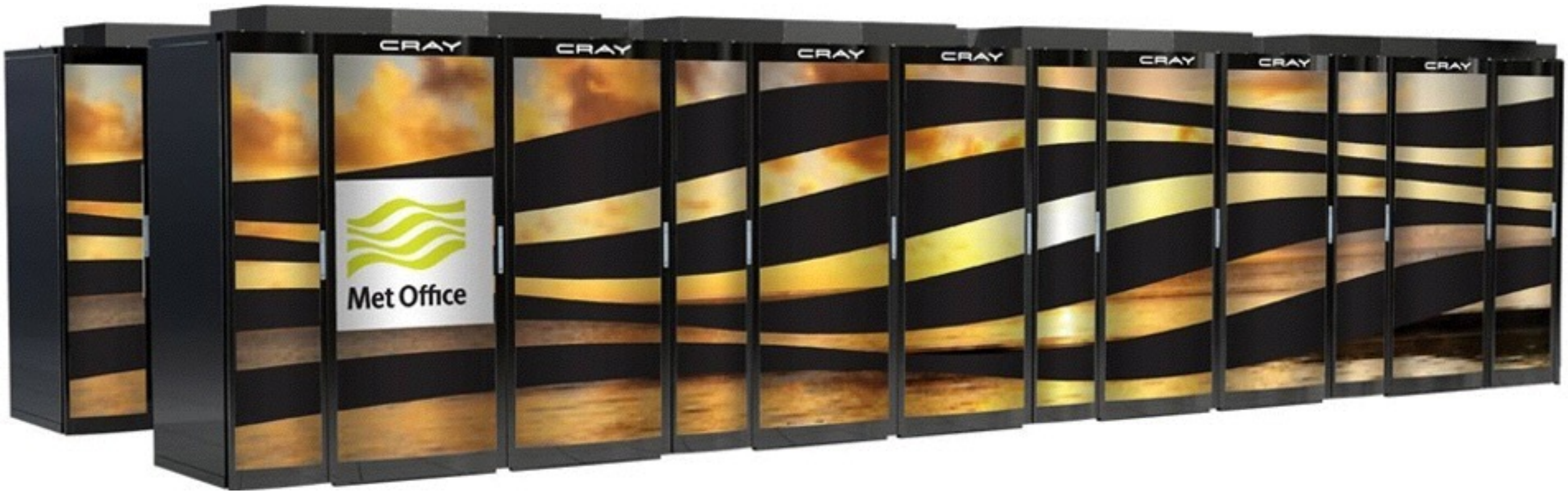
ETH zürich

Contemporary Supercomputing

Thomas C. Schulthess

Supercomputers – the most performant, general purpose HPC systems at any given time



Cray XC system – presently one of the best-selling supercomputing platforms – was funded by the DARPA HPCS* program

(*) HPCS stand for "High Productivity Computing Systems"

Hardware developments were successful, but none of the HPCS' high-productivity languages (Chapel and X10) have been widely adopted

Does this mean performance is important, but not productivity?

ETH zürich

Performance: floating point operations

www.top500.org

RANK	SITE	SYSTEM	CORES	RMAX (TFLOP/S)	RPEAK (TFLOP/S)	POWER (KW)
1	National Super Computer Center in Guangzhou China	Tianhe-2 (MilkyWay-2) - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
2	DOE/SC/Oak Ridge National Laboratory United States	Titan - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
3	DOE/NNSA/LLNL United States	Sequoia - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIII[illegible] 2.0[illegible], Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660
5	DOE/SC/Argonne National Laboratory United States	Mira - BlueGe[illegible] BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945
6	Swiss National Supercomputing Centre (CSCS) Switzerland	Piz [illegible]aint - [illegible]30, Xeon E5-2670 8C 2.[illegible], Ar[illegible] interconnect , NVIDIA K20x Cray I[illegible]	115,984	6,271.0	7,788.9	2,325
7	Texas Advanced Computing Center/Univ. of Texas United States	Stampede - PowerEdge C8220, Xeon E5-2680 8C 2.700GHz, Infiniband FDR, Intel Xeon Phi SE10P Dell	462,462	5,168.1	8,520.1	4,510
8	Forschungszentrum Juelich (FZJ) Germany	JUQUEEN - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM	458,752	5,008.9	5,872.0	2,301
9	DOE/NNSA/LLNL United States	Vulcan - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM	393,216	4,293.3	5,033.2	1,972
10	Government United States	Cray CS-Storm, Intel Xeon E5-2660v2 10C 2.2GHz, Infiniband FDR, Nvidia K40 Cray Inc.	72,800	3,577.0	6,131.8	1,499

FLOP/s

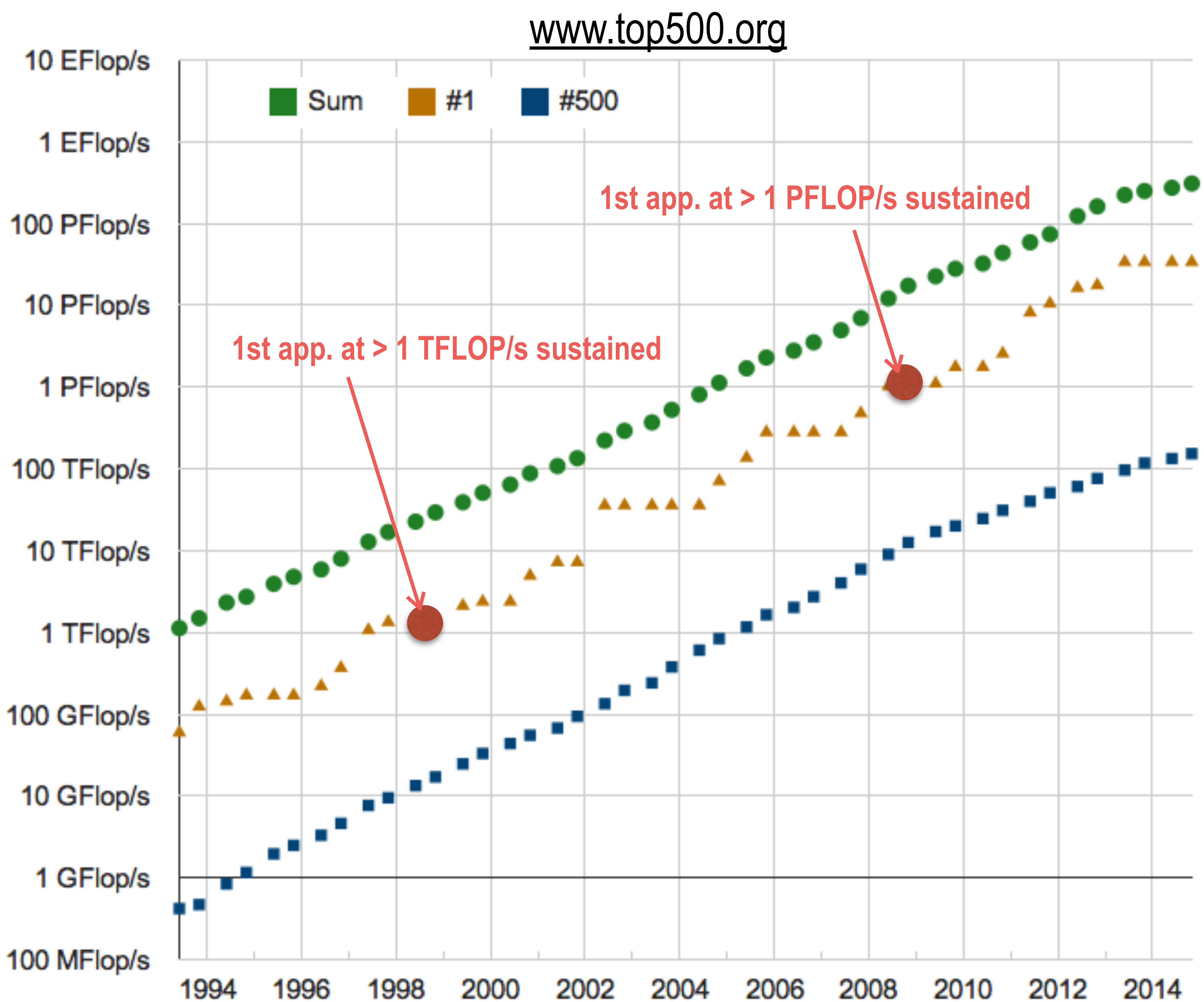
www.green500.org

Green500 Rank	MFLOPS/W	Site*	Computer*	Total Power (kW)
1	5,271.81	GSI Helmholtz Center	L-CSC - ASUS ESC4000 FDR/G2S, Intel Xeon E5-2690v2 10C 3GHz, Infiniband FDR, AMD FirePro S9150 Level 1 measurement data available	57.15
2	4,945.63	High Energy Accelerator Research Organization /KEK	Suiren - ExaScaler 32U256SC Cluster, Intel Xeon E5-2660v2 10C 2.2GHz, Infiniband FDR, PEZY-SC	37.83
3	4,447.58	GSIC Center, Tokyo Institute of Technology	TSUBAME-KFC - LX 1U-4GPU/104Re-1G Cluster, Intel Xeon E5-2620v2 6C 2.100GHz, Infiniband FDR, NVIDIA K20x	35.39
4	3,962.73	Cray Inc.	Storm1 - Cray CS[illegible], Intel Xeon E5-2660v2 10C 2.2GHz, Infiniband FD[illegible]m Level 3 meas[illegible] data available	44.54
5	3,631.70	Cambridge University	Wilkes - D[illegible] Cluster, Intel Xeon E5-2630v2 6C 2.600GHz, Infinib[illegible]VIDIA K20	52.62
6	3,543.32	Financial Institution	[illegible]DX360M4, Intel Xeon E5-2680v2 10C 2.800GHz, [illegible]band, NVIDIA K20x	54.60
7	3,517.84	Center for Computational Scie[illegible]es, University of Tsukuba	H[illegible]-PACS TCA - Cray CS300 Cluster, Intel Xeon E5-2680v2 10C [illegible]00GHz, Infiniband QDR, NVIDIA K20x	78.77
8	3,459.46	SURFsara	Cartesius Accelerator Island - Bullx B515 cluster, Intel Xeon E5-2450v2 8C 2.5GHz, InfiniBand 4× FDR, Nvidia K40m	44.40
9	3,185.91	Swiss National Supercomputing Centre (CSCS)	Piz Daint - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x Level 3 measurement data available	1,753.66
10	3,131.06	ROMEO HPC Center - Champagne-Ardenne	romeo - Bull R421-E3 Cluster, Intel Xeon E5-2650v2 8C 2.600GHz, Infiniband FDR, NVIDIA K20x	81.41

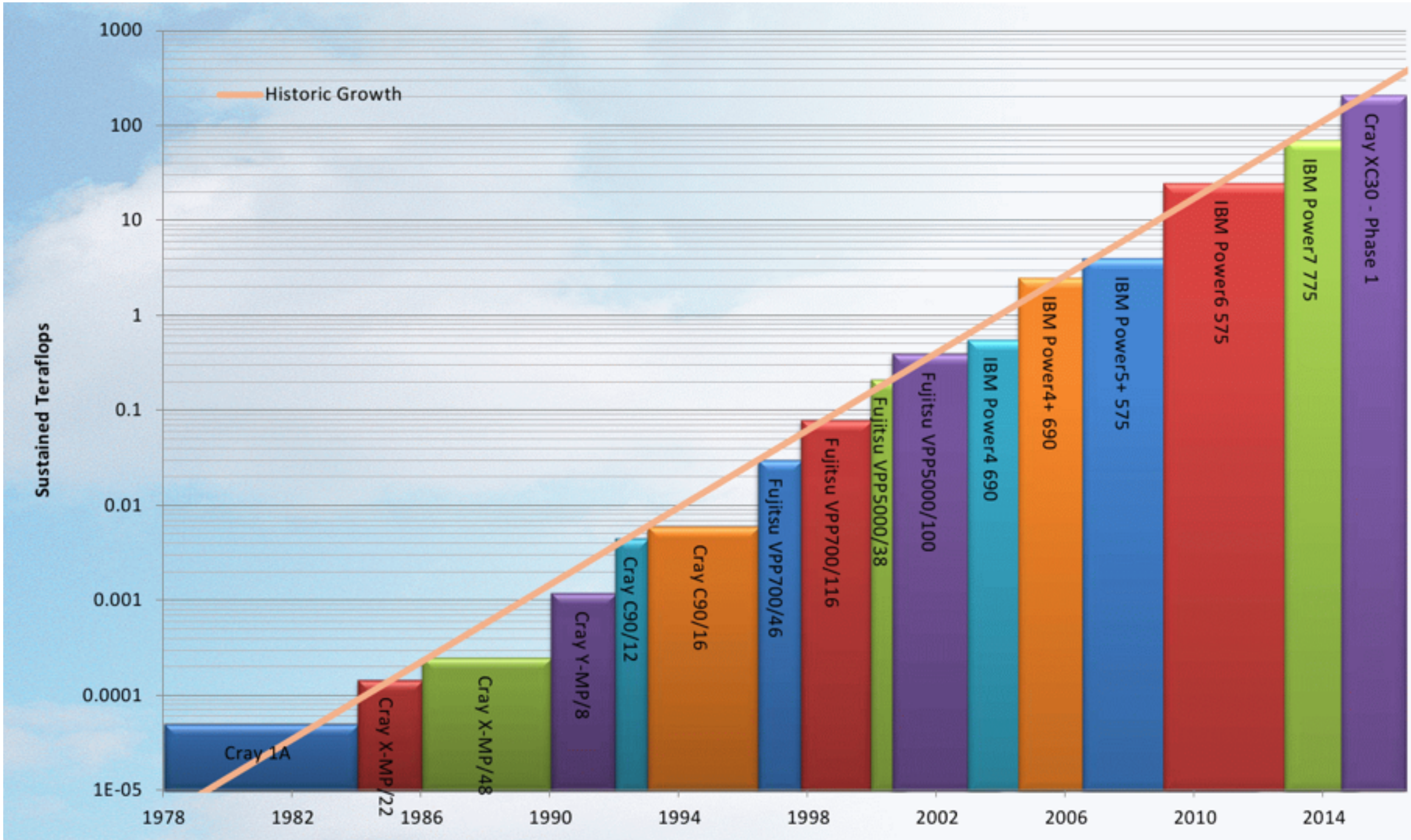
FLOP/s/W

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

1000-fold performance improvement per decade



“Only” 100-fold improvement for climate codes



Source: Peter Bauer, ECMWF

Has efficiency of climate codes dropped 10-fold every decade decade?

Revisiting the FLOP/s and GFLOP/s/W metrics

Metric for time to solution in High-Performance LINPACK benchmark:

(1) high arithmetic density increases with problem size: $\frac{\#\text{ of FLOP}}{\#\text{ of load-stores}} \propto O(N)$

(2) thus, it is reasonable to measure work in number of retired floating point operations (totFLOP),

(3) and to normalised the time to solution $\frac{\Delta t}{\text{totFLOP}}$ and performance $\frac{\text{totFLOP}}{\Delta t}$ $[\text{FLOP}/\text{s}]$ accordingly

… and a metric for energy to solution of HPL:

(1) normalised energy to solution E by simple measure of work $\frac{E}{\text{totFLOP}}$

(2) minimising energy to solution is equivalent to maximising $\frac{\text{totFLOP}}{E}\left[\frac{\text{FLOP}}{\text{Joule}}\right]$

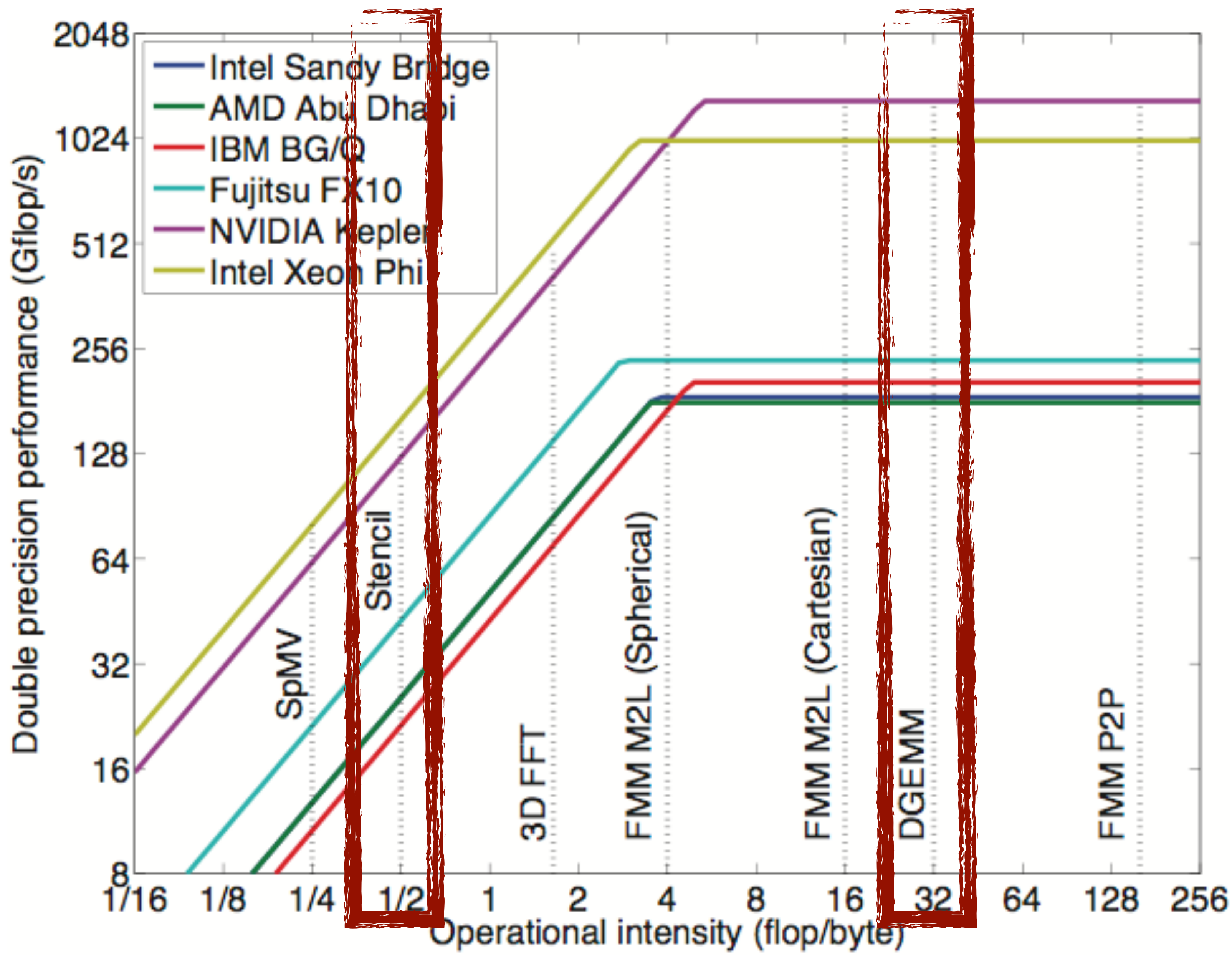
(3) ... and of course $\left[\frac{\text{FLOP}}{\text{Joule}}\right] = \left[\frac{\frac{\text{FLOP}}{\text{sec.}}}{\text{Watt}}\right]$

FLOP/s and GFLOP/s/W are good metrics for HPL, but is this true for all motifs?

Peak performance is algorithm dependent

Climate / HPCG

Materials science / HPL



source: lorena a. barba group (lorenabarba.com)

Peak performance varies with arithmetic density of algorithm / code / benchmark

Generic performance metrics in HPC

Energy & Time

Optimising Time and Energy to Solution

Time to solution (TTS):

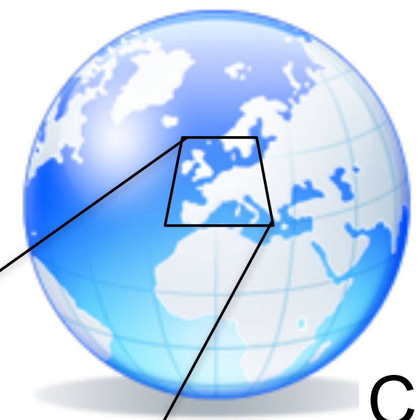
- do we have to minimise time to solution?
- no, it just needs to be good enough to meet operational constrains

Energy to solution (ETS):

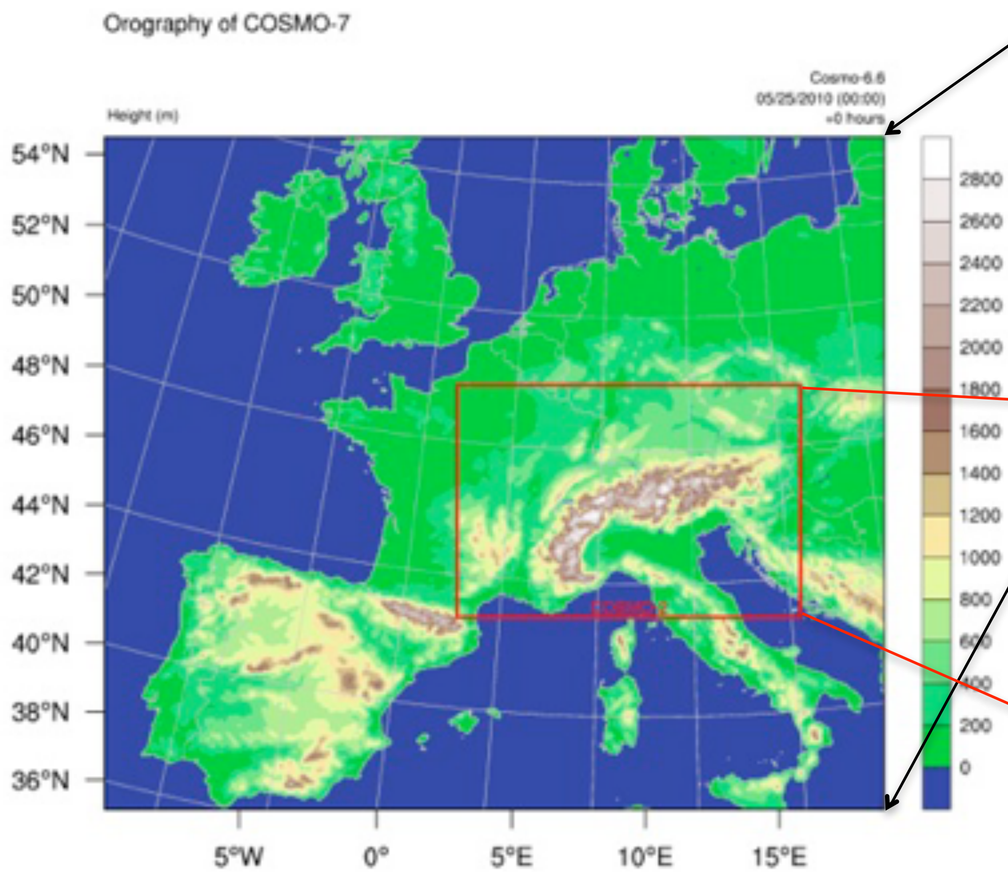
- energy is directly proportional to cost (energy = power x time)
- given all operational constraints, energy should be minimised

Today's (2015) production suite of Meteo Swiss

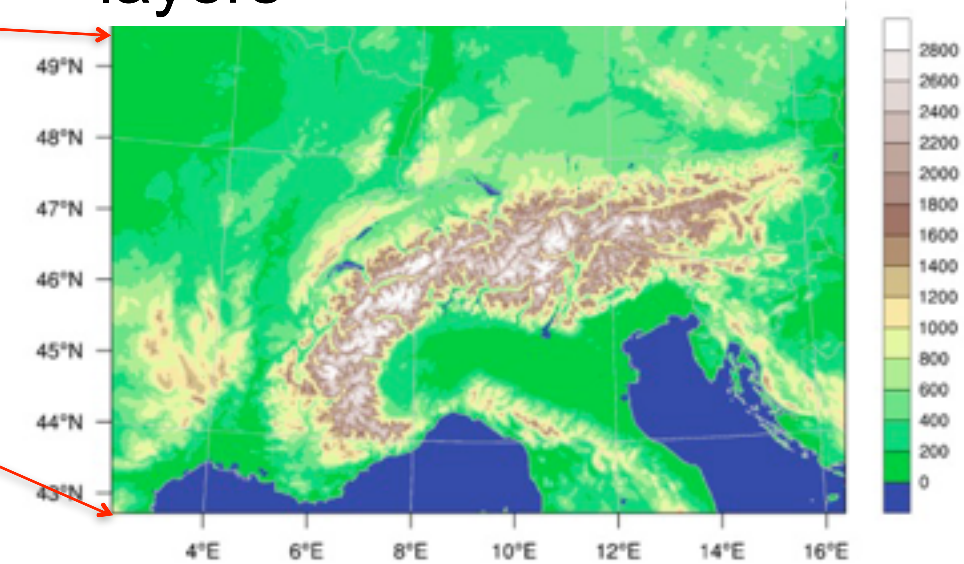
COSMO-7
3x per day 72h forecast
6.6 km lateral grid, 60 layers



ECMWF
2x per day
16 km lateral grid, 91 layers



COSMO-2
8x per day 24h forecast
2.2 km lateral grid, 60 layers



Some of the products generate from these simulations:

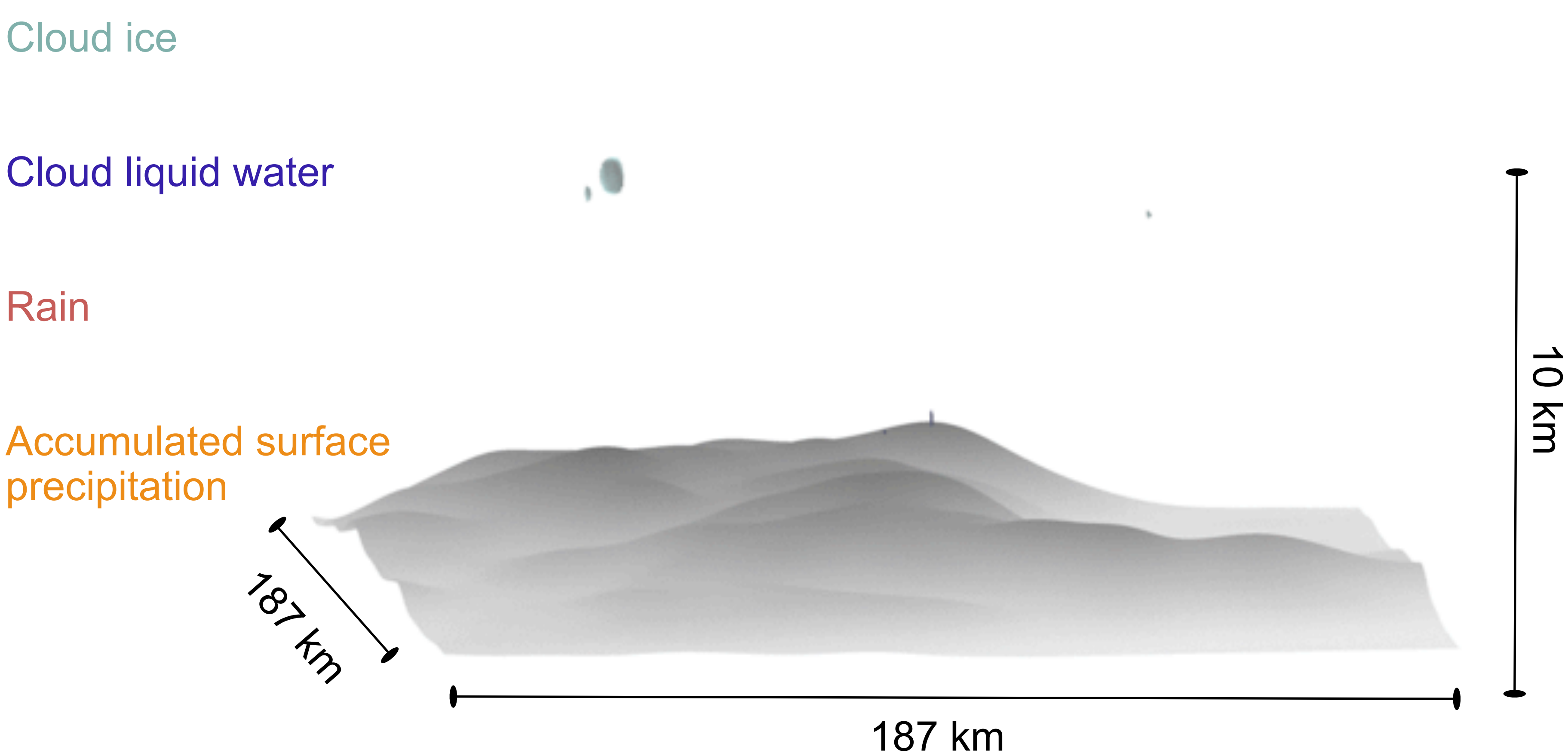
- Daily weather forecast on TV / radio
- Forecasting for air traffic control (Sky Guide)
- Safety management in event of nuclear incidents

"Albis" & "Lema", CSCS production systems for Meteo Swiss

Cray XE6 procured in spring 2012 based on 12-core AMD Opteron multi-core processors

Cloud resolving simulations

Institute for Atmospheric and Climate Science Study at ETH Zürich (Prof. Schär) demonstrates cloud resolving models converge at 1-2km resolution (at least for convective clouds over the alpine region)

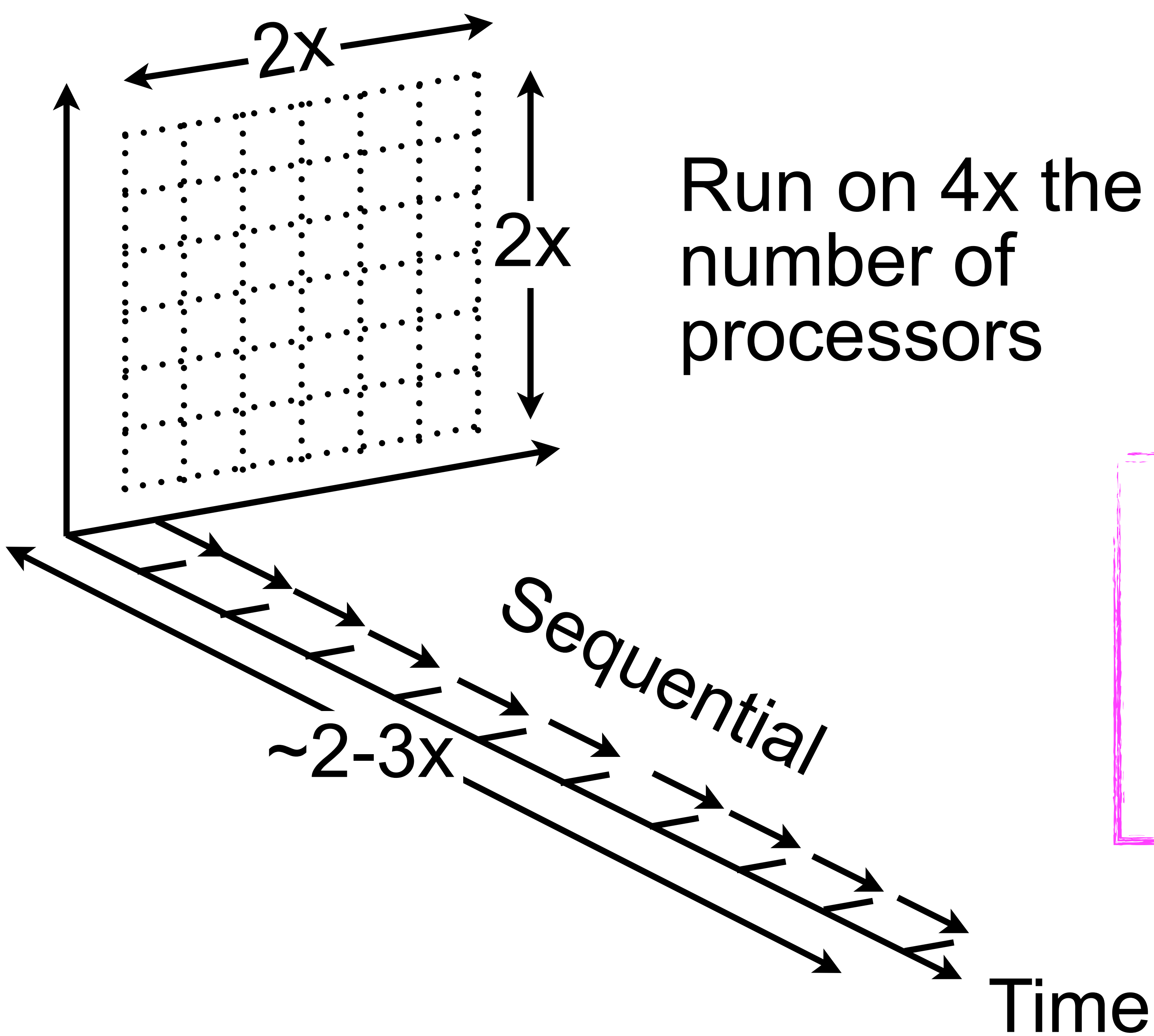


COSMO model setup: Δx=550 m, Δt=4 sec   Plots generated using INSIGHT

Orographic convection – simulation: 11-18 local time, 11 July 2006 (Δt_plot=4 min)

Source: Wolfgang Langhans and Christoph Schär, Institute for Atmospheric and Climate Science, ETH Zurich

Improve resolution of Meteo Swiss model from 2 to 1 km

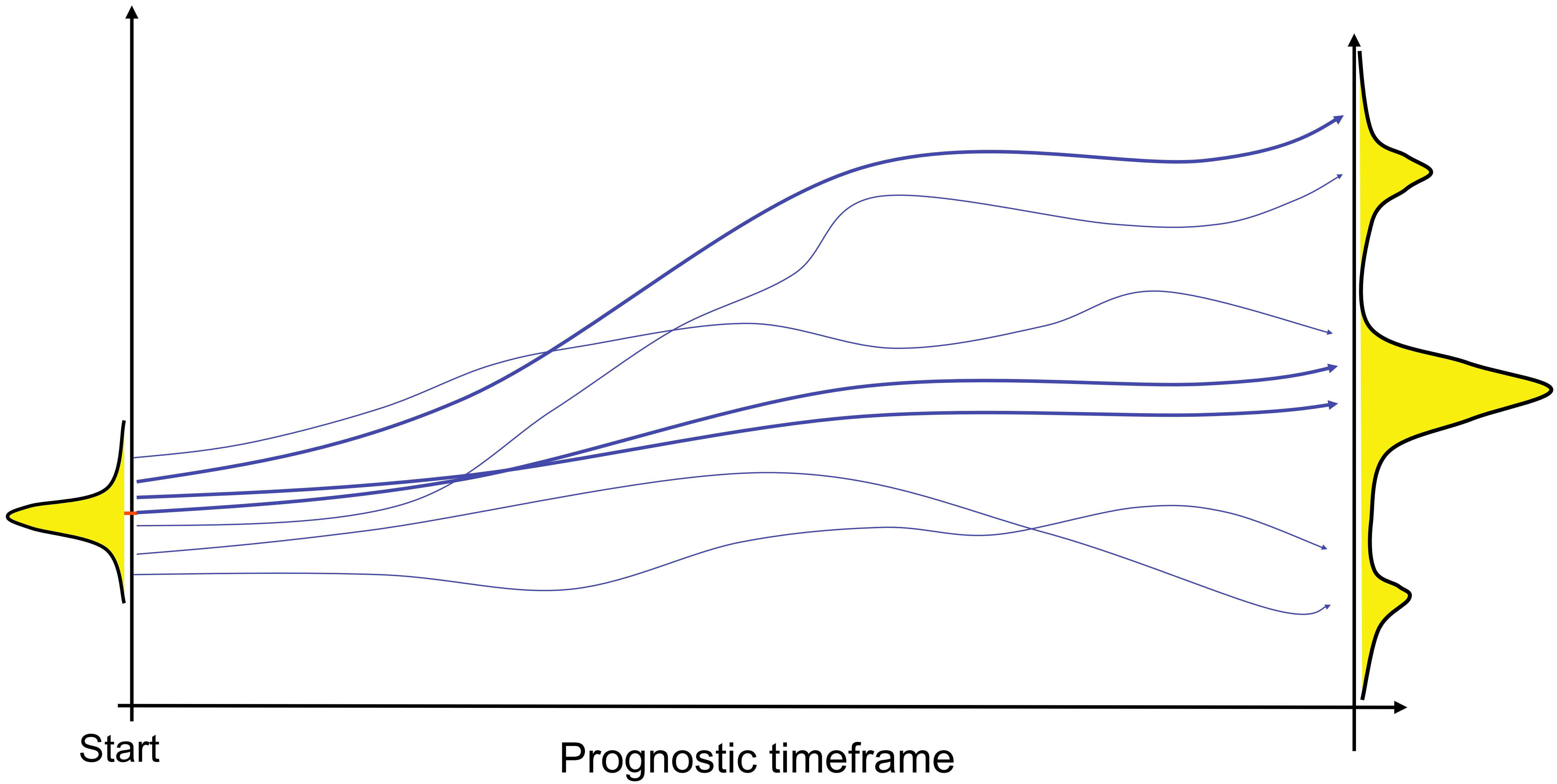


Run on 4x the number of processors

Doubling the resolution requires ~10x performance increase

Prognostic uncertainty

The weather system is chaotic
→ rapid growth of small perturbations (butterfly effect)



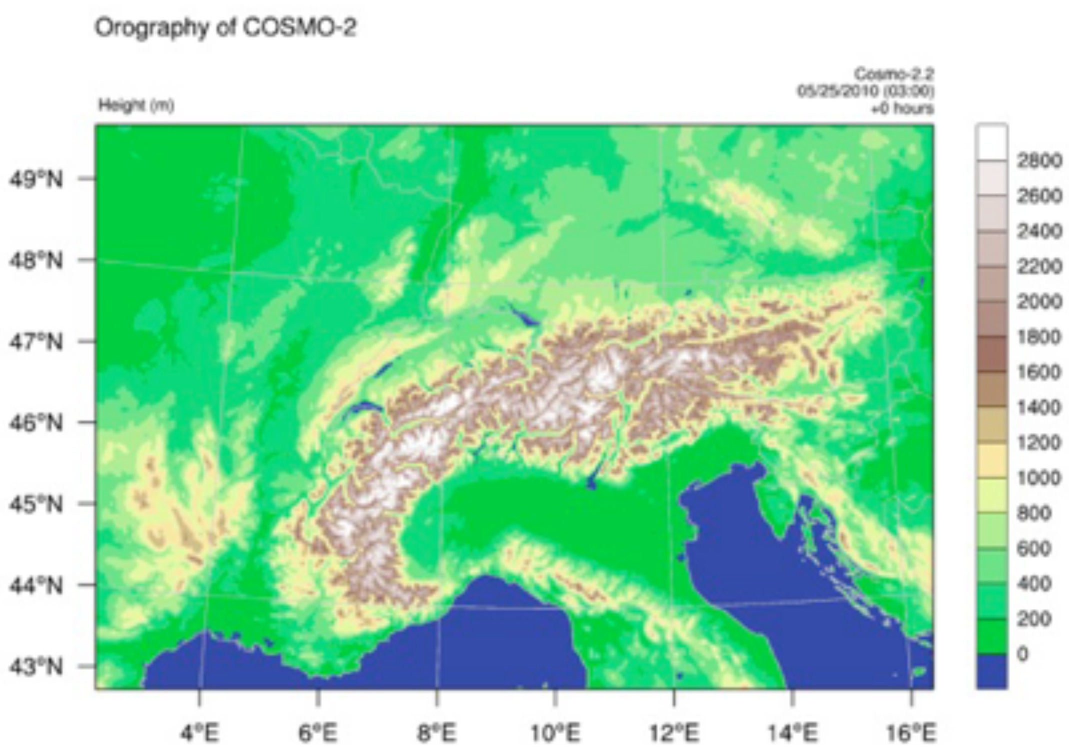
Ensemble method: compute distribution over many simulations

Improving simulation quality requires higher performance – what exactly and by how much?

Resource determining factors for Meteo Swiss' simulations

Current model running through mid 2016

COSMO-2: 24h forecast running in 30 min.
8x per day



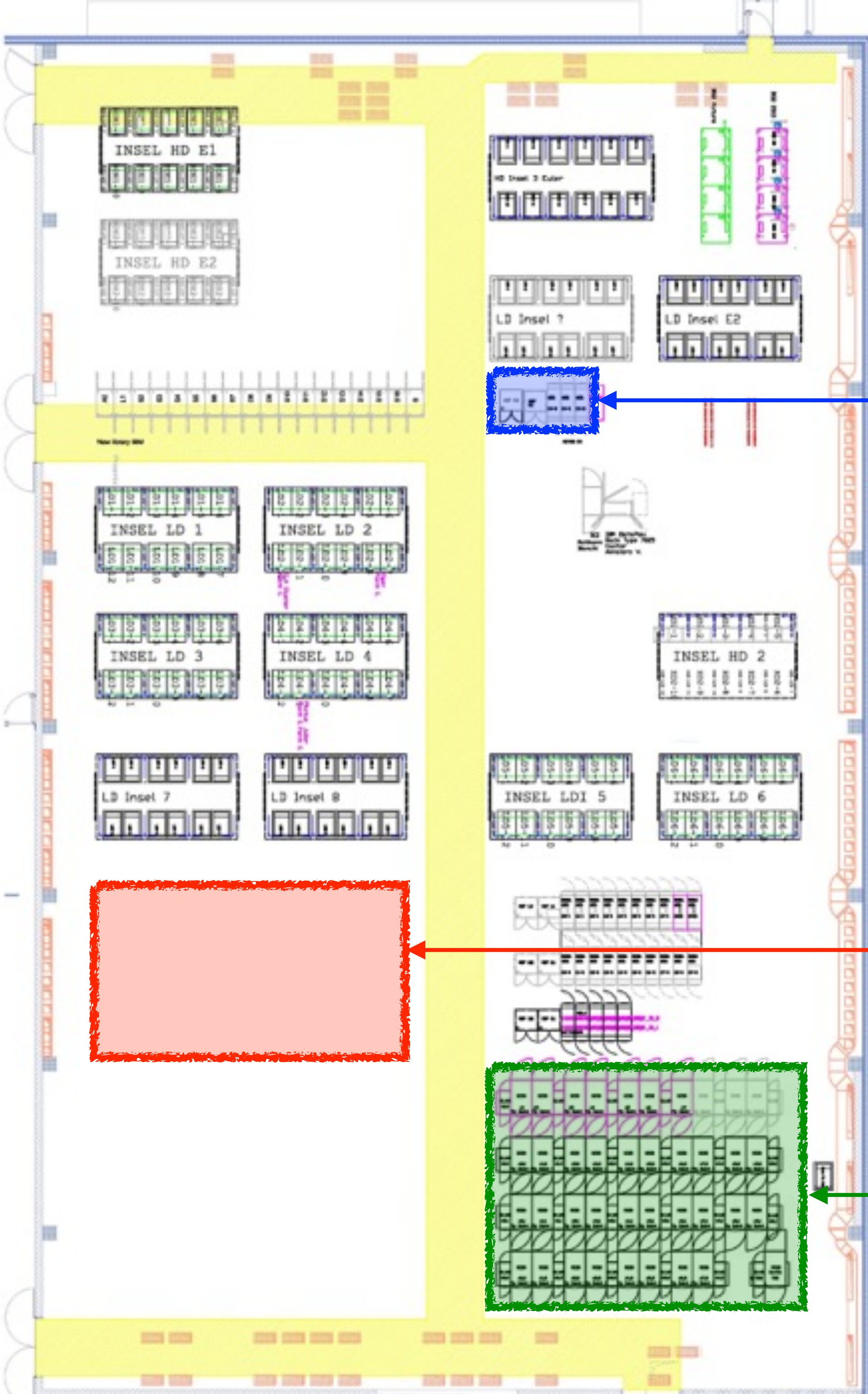
New model starting operation on in Jan. 2016

COSMO-1: 24h forecast running in 30 min.
8x per day (**~10x** COSMO-2)

COSMO-2E: 21-member ensemble,120h forecast
in 150 min., 2x per day (**~26x** COSMO-2)

KENDA: 40-member ensemble,1h forecast
in 15 min., 24x per day (**~5x** COSMO-2)

New production system must deliver
~40x the simulations performance
of "Albis" and "Lema"



CSCS machine room

State of the art implementation of new system for Meteo Swiss

Albis & Lema: 3 cabinets Cray XE6 installed Q2/2012

- New system need to be installed Q2/2015
- Assuming 2x improvement in per-socket performance: ~20x more X86 sockets would require 30 Cray XC cabinets

New system for Meteo Swiss if we build it like the German Weather Service (DWD) did theirs, or UK Met Office, or ECMWF … (30 racks XC)

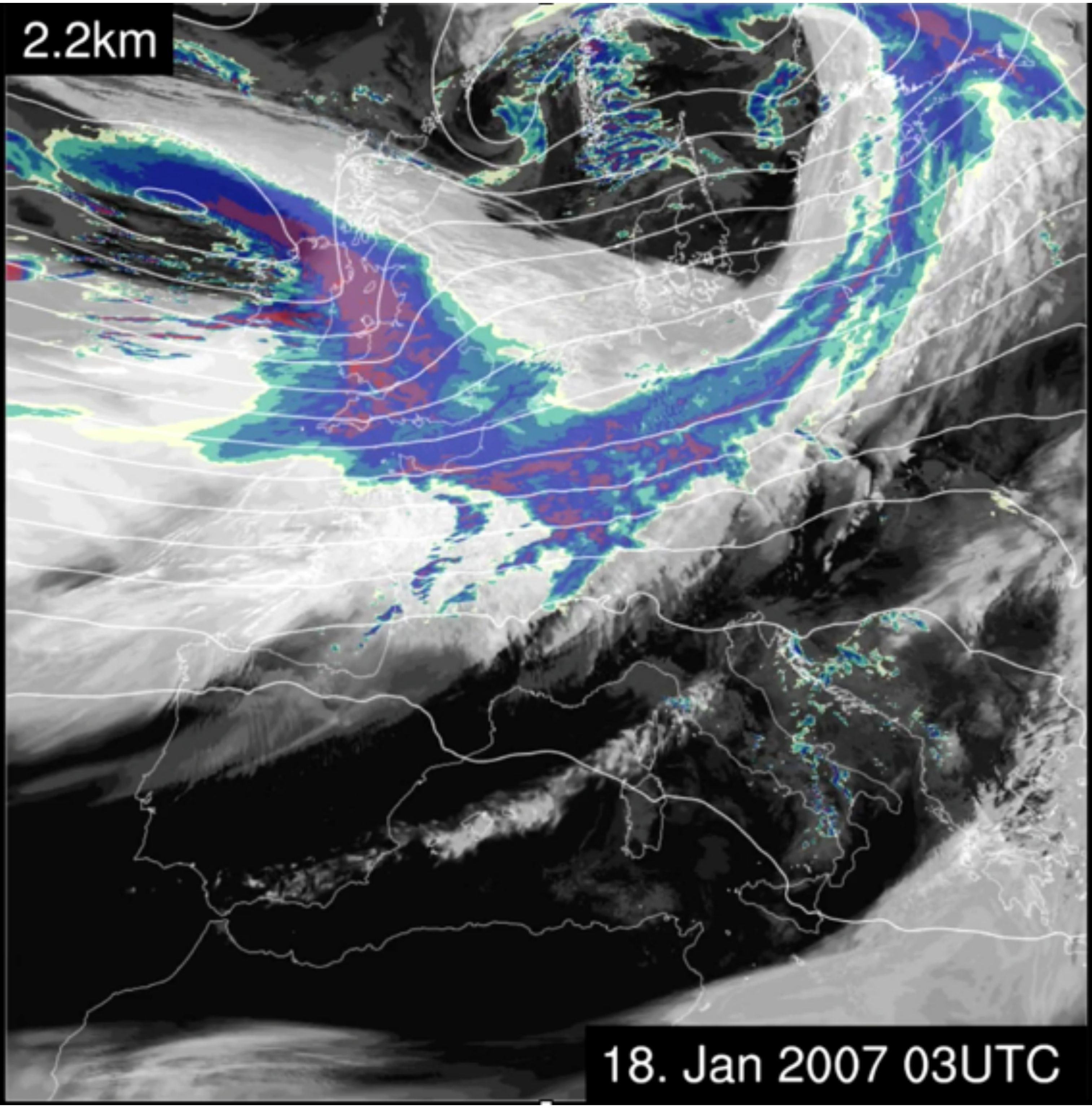
Current Cray XC30/XC40 platform (space for 40 racks XC)

Thinking inside the box is not a good option!

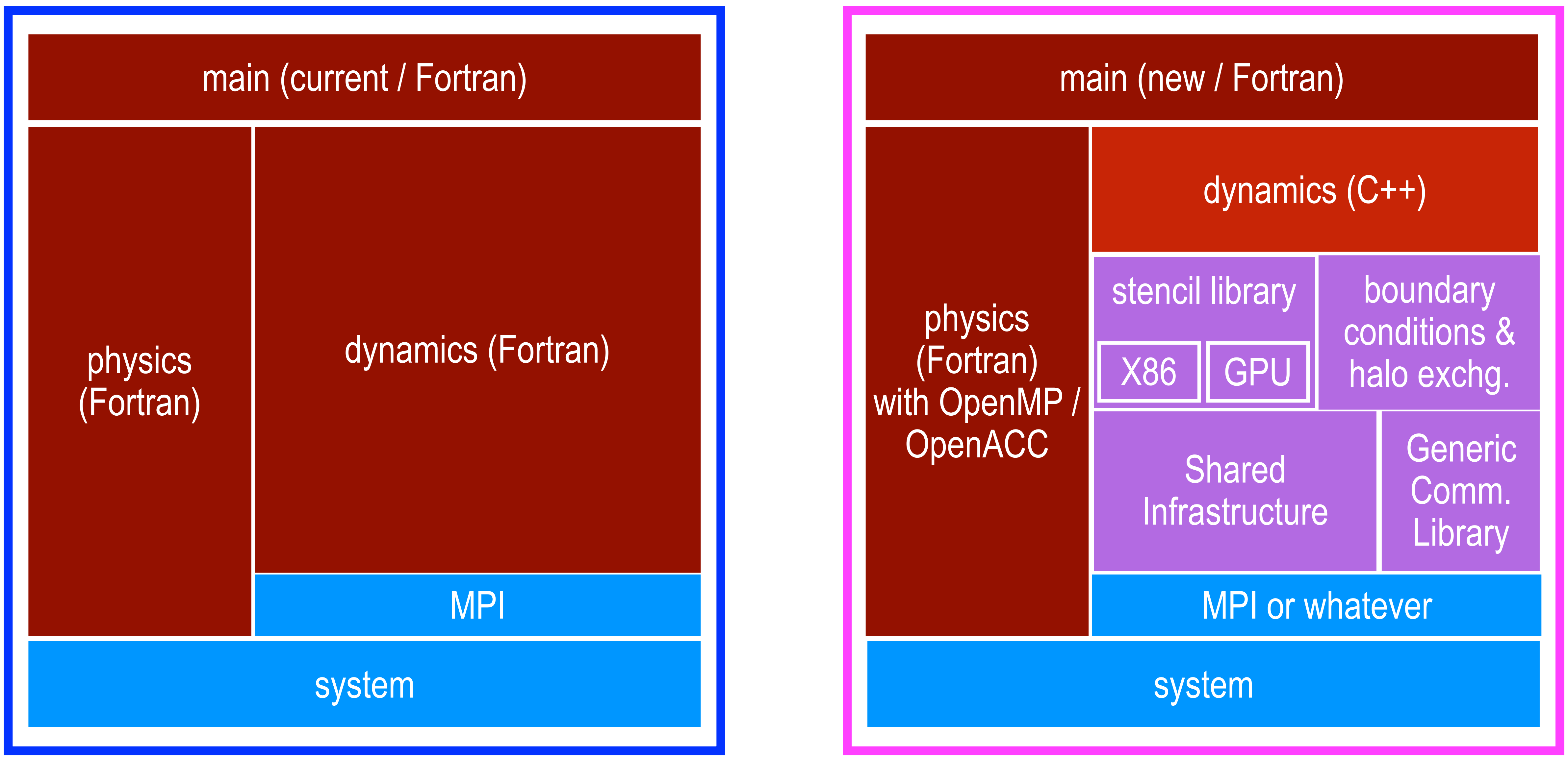
"Piz Daint," a productive supercomputer with CPU-GPU nodes

- Cray XC30 with 5272 compute nodes, each with one 8-core Xeon CPU and one K20X GPU
- Fully populated dragonfly: global bandwidth per node matches injection bandwidth
- Developed with application performance in mind: CP2K, COSMO, SPECFEM, GROMACS, Q.E.
- Co-designed with CP2K and **COSMO-OPCODE**
- Final upgrade 10/2013; accepted 12/2013; early science 01-03/2014; full operation since 04/2014

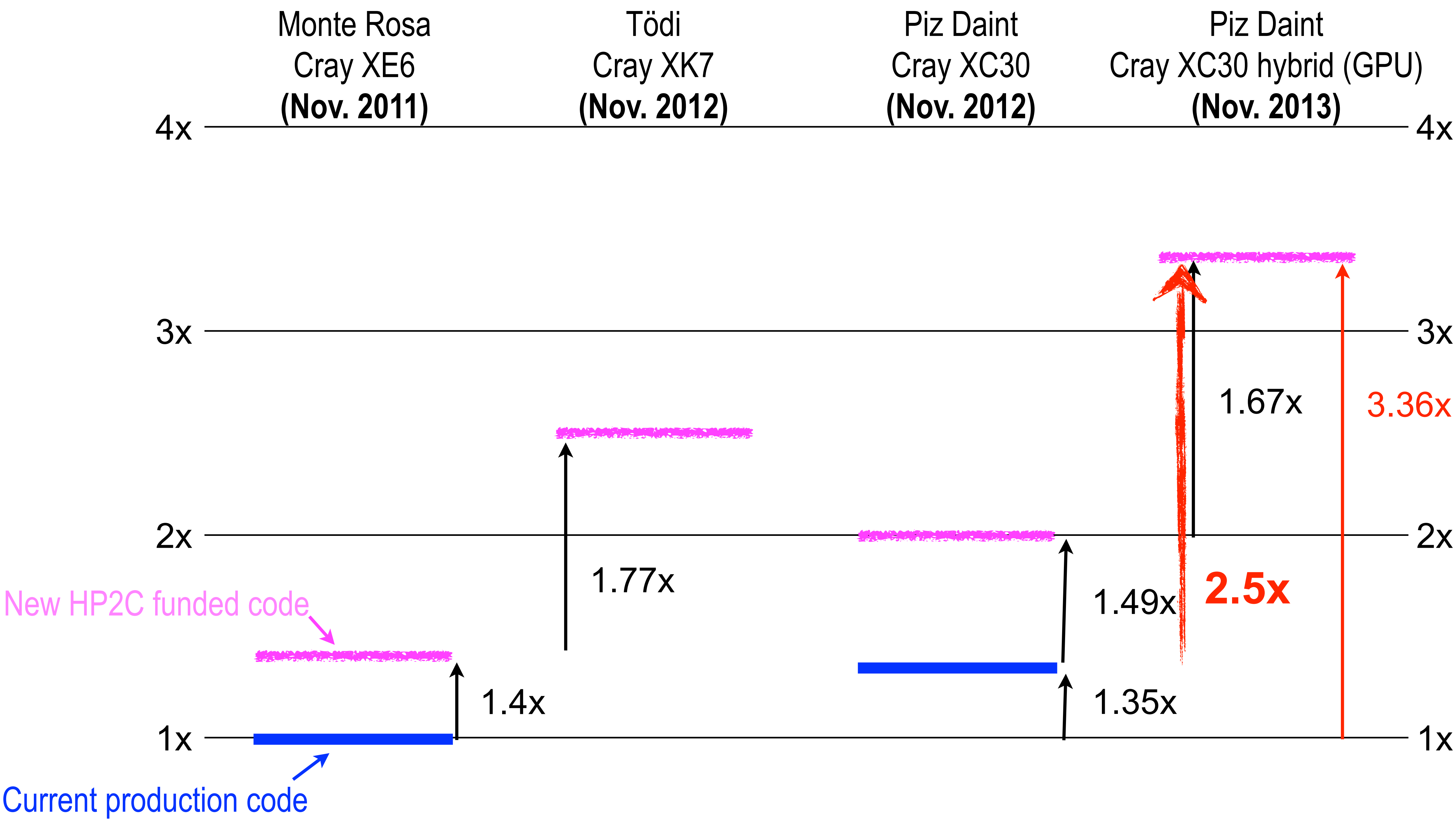
COSMO-2 running on the GPUs of "Piz Daint"



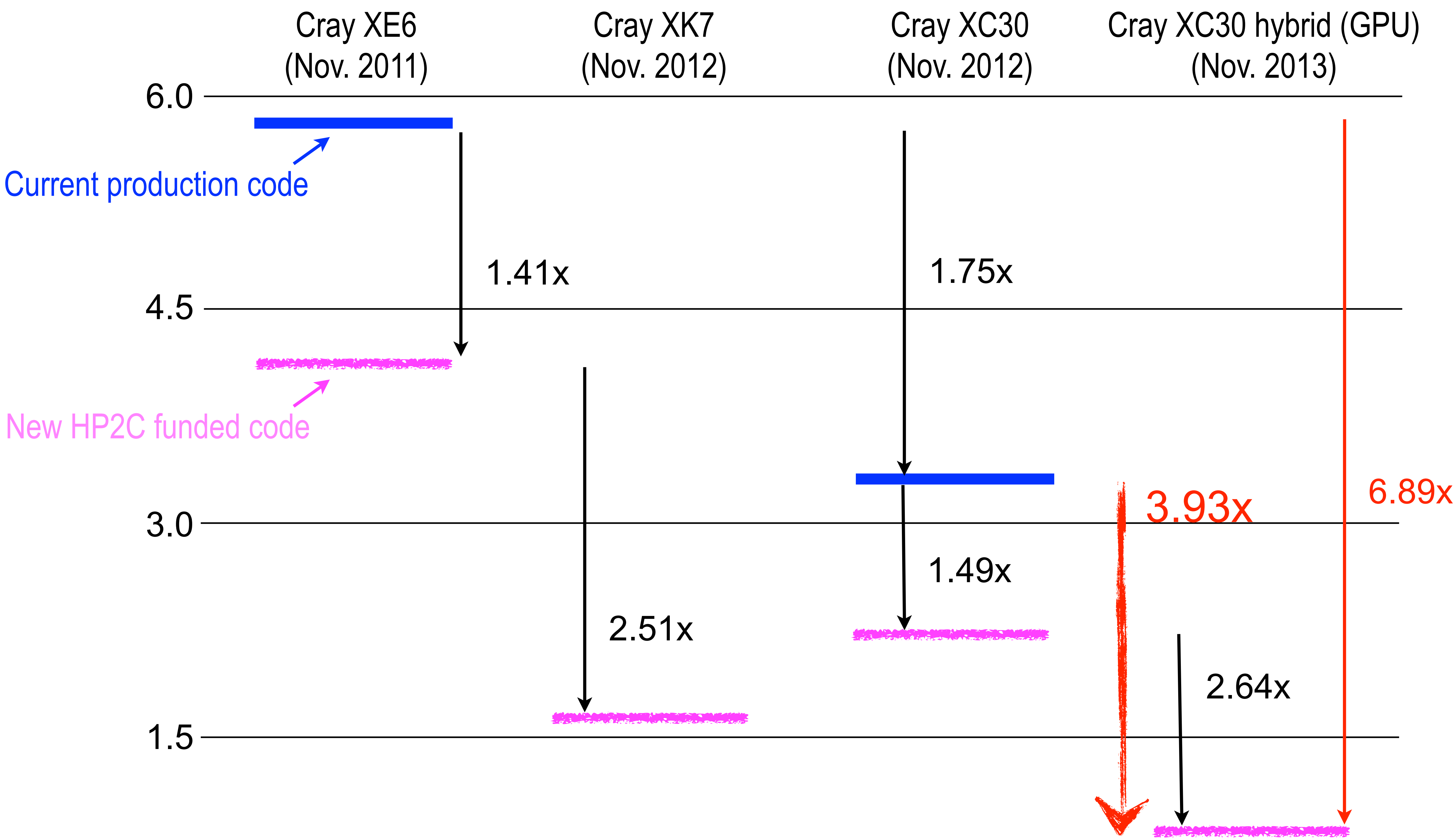
COSMO: current and new (HP2C developed) code



Speedup of COSMO-2 production problem – apples to apples comparison with 33h forecast of Meteo Swiss



Energy to solution (kWh / ensemble member)



ETHzürich

Unconventional implementation of new system for Meteo Swiss

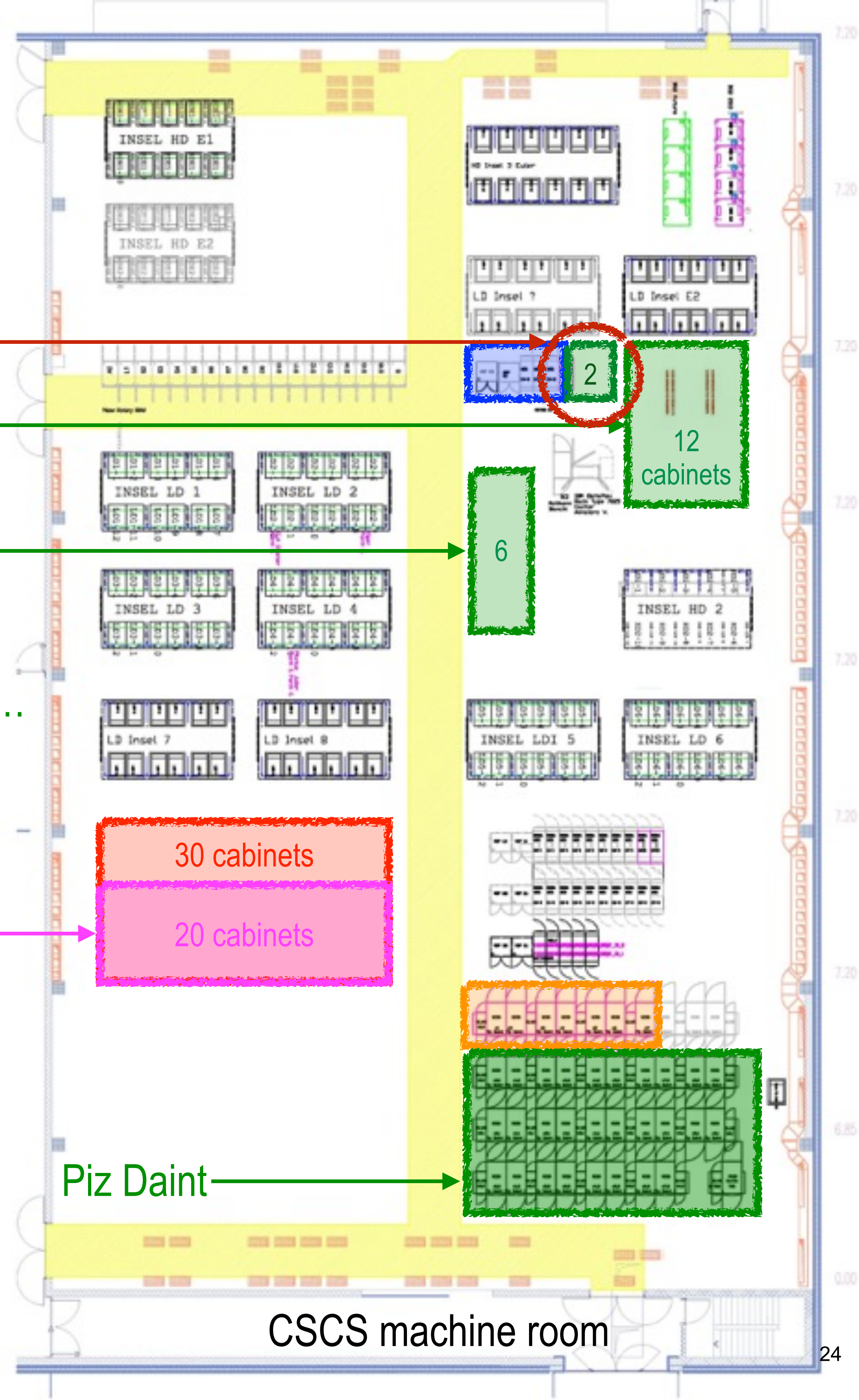
The new Meteo Swiss system “Piz Kesh”

Using same implementation as on “Piz Daint”

Modifying parts of the model to single precision

Further options: increase GPU density, use K40 or **K80** …

Using the refactored code on conventional X86




ECOphlex
ECOphlex
ECOphlex
CSCS
CRAY
CRAY

References and Collaborators

- Peter Messmer and his team at the NVIDIA co-design lab at ETH Zurich
- Teams at CSCS and Meteo Suisse
- O. Fuhrer, C. Osuna, X. Lapillonne, T. Gysi, B. Cumming, M. Bianco, A. Arteaga, T. C. Schulthess, "**Towards a performance portable, architecture agnostic implementation strategy for weather and climate models**", Supercomputing Frontiers and Innovations, vol. 1, no. 1 (2014), see superfri.org
- G. Fourestey, B. Cumming, L. Gilly, and T. C. Schulthess, "**First experience with validating and using the Cray power management database tool**", Proceedings of the Cray Users Group 2014 (CUG14) (see arxiv.org for preprint)
- B. Cumming, G. Fourestey, T. Gysi, O. Fuhrer, M. Fatica, and T. C. Schulthess, "**Application centric energy-efficiency study of distributed multi-core and hybrid CPU-GPU systems**", Proceedings of the International Conference on High-Performance Computing, Networking, Storage and Analysis, SC'14, New York, NY, USA (2014). ACM
- T. Gysi, C. Osuna, O. Fuhrer, M. Bianco and T. C. Schulthess, "**STELLA: A domain-specific tool for structure grid methods in weather and climate models**", to be published in Proceedings of the International Conference on High-Performance Computing, Networking, Storage and Analysis, SC'15, New York, NY, USA (2015). ACM

IP[y]: Notebook Gridtools4Py Last Checkpoint: Jun 30 12:29 (autosaved)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: Slideshow

In [1]:
```
#
# notebook setup
#
%matplotlib qt
import matplotlib.pyplot as plt
import numpy as np
```

::: Gridtools4Py :::

A Python interface for Gridtools

A copy stencil implemented in Python

In [2]:
```
from gridtools.stencil import MultiStageStencil


class CopyStencil (MultiStageStencil):
    """
    Definition of a simple copy stencil.-
    """
    def kernel (self, out_data, in_data):
        """
        The entry stage of this stencil.-
        """
        #
        # iterate over the interior data points
        #
        for p in self.get_interior_points (out_data):
            out_data[p] = in_data[p]
```

Use NumPy arrays as data fields

In [3]:
```
domain = (64, 64, 32)

source = np.random.rand (*domain)    # data field of size 'domain'
                                     # filled with random numbers
target = np.zeros (domain)           # data field of size 'domain'
                                     # filled with zeros
```

IP[y]: Notebook Gridtools4Py Last Checkpoint: Jun 30 12:29 (autosaved)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: Slideshow

Run the stencil in Python mode

In [4]:
```
copy          = CopyStencil ( )        # instance of the stencil defined above
copy.backend = "python"                # will run in Python only mode
                                       target) # execute it
```

1 loops, best of 10: 88.4 ms per loop

Run the *same* stencil in C++ mode

In [5]:
```
copy.backend = "c++"          # will run using Gridtools in C++
                              arget) # execute it
```

1 loops, best of 10: 83.7 µs per loop

The code has been translated, compiled and dynamically linked into the current session

In [6]:
```
copy.lib_obj
```

Out[6]:
```
<CDLL '/var/folders/qp/q9r_zwqjlqg7s2y6sj9hv3_80000gp/T/__gridtools_d_30unlc/libcopystenci
handle 7fa5653122d0 at 11d5a77b8>
```

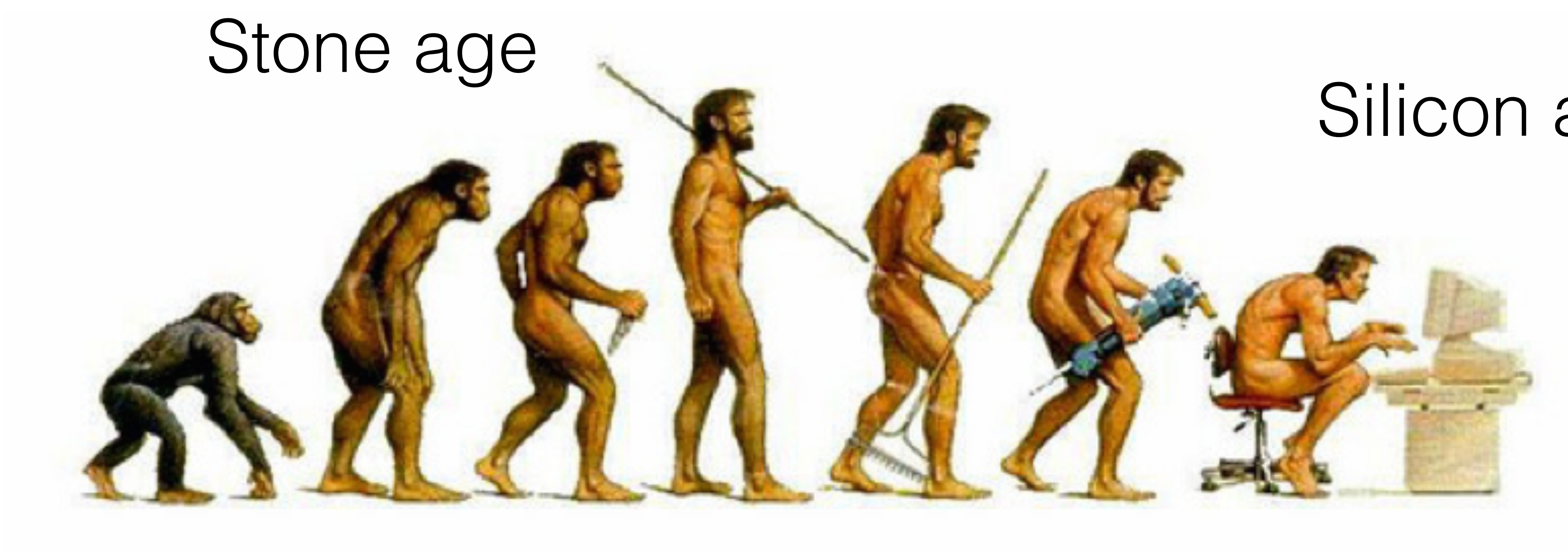
Example: the Laplace operator

In [7]:
```
class Laplace (MultiStageStencil):
    def kernel (self, out_data, in_data):
        """
        The user must always define a 'kernel' function.-
        """
        for p in self.get_interior_points (out_data):
            out_data[p] = -4.0 * in_data[p] - (
                                    in_data[p + (1,0,0)] + in_data[p + (0,1,0)] +
                                    in_data[p + (-1,0,0)] + in_data[p + (0,-1,0)])
```

Run it in Python, C++ and CUDA modes

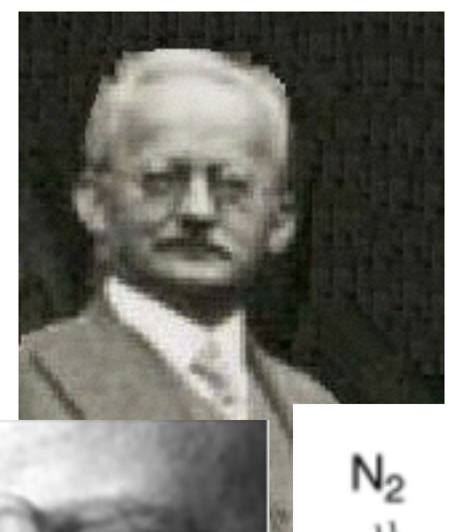
In [8]:
```
lap = Laplace ( )
```

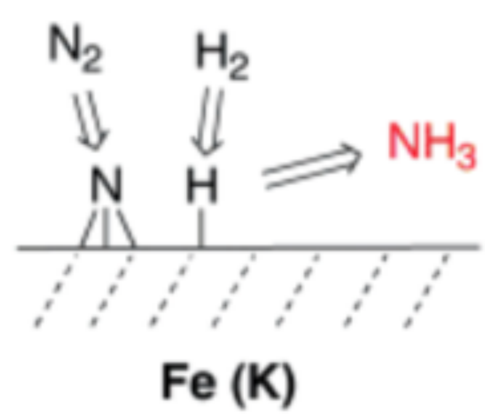
Materials and human evolution



Serendipitous discovery & Edisonian development

- Most new materials are discovered serendipitously (particularly true for complex materials)
- Or through very laborious searches, e.g.
  - Edison tested 3000 materials for his filament and settled on burned sewing thread
  - Haber-Bosh ammonia synthesis with osmium as a catalyst
    Mitasch (BASF) tested ~22,000 materials to find iron-based catalyst – still in use today
  - Norskov showed in 2009 that CoMo is a more efficient & inexpensive catalyst

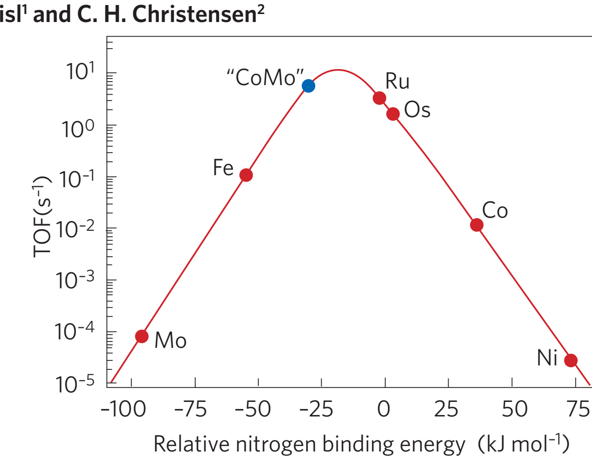





nature chemistry
REVIEW ARTICLE
PUBLISHED ONLINE: 19 MARCH 2009 | DOI: 10.1038/NCHEM.121

Towards the computational design of solid catalysts

J. K. Nørskov[1]*, T. Bligaard[1], J. Rossmeisl[1] and C. H. Christensen[2]



Nicola Marzari

Systematic searches with high-throughput & capability runs

- There are ~150,000 known inorganic materials with published structures
- Very basic properties computed with DFT-based quantum simulations take ~10 minutes on a powerful workstation (e.g. hybrid CPU-GPU)
- “Piz Daint” with 5272 hybrid CPU-GPU nodes could scan ~5000 structures / 10 minutes

But we want to study more complex, harder to compute properties – how complex?

Approaching the problem form the other end

Start with the most reliable (and expensive) approach to electronic structure …

Linearised Augmented Plane Wave Method (LAPW)

… and the largest problem that is reasonable* for materials searches …

~1000 atoms in a unit cell – the “1000-atom problem” **

… and bet on future improvements in extreme-scale computing

novel architectures and exa-scale computing

(*) Using W. Kohn’s arguments on nearsightedness of electronic matter

(**) proposed by Claudia Draxl at a PRACE project meeting in spring 2011

Solving the Kohn-Sham Equations is the bottleneck in most DFT-based materials science codes

Kohn-Sham Eqn.

$$\left(-\frac{\hbar^2}{2m}\nabla^2 + v_{\mathrm{LDA}}(\vec{r})\right)\psi_i(\vec{r}) = \epsilon_i \psi_i(\vec{r})$$

Ansatz

$$\psi_i(\vec{r}) = \sum_{\mu} c_{i\mu}\phi_\mu(\vec{r})$$

Hermitian matrix

$$H_{\mu\nu} = \int \phi_\mu^*(\vec{r})\left(-\frac{\hbar^2}{2m}\nabla^2 + v_{\mathrm{LDA}}(\vec{r})\right)\phi_\nu(\vec{r})d\vec{r}$$

Basis is not orthogonal

$$S_{\mu\nu} = \int \phi_\mu^*(\vec{r})\phi_\nu(\vec{r})d\vec{r}$$

Solve generalized eigenvalue problem

$$(\mathbf{H} - \varepsilon_i \mathbf{S}) = 0$$

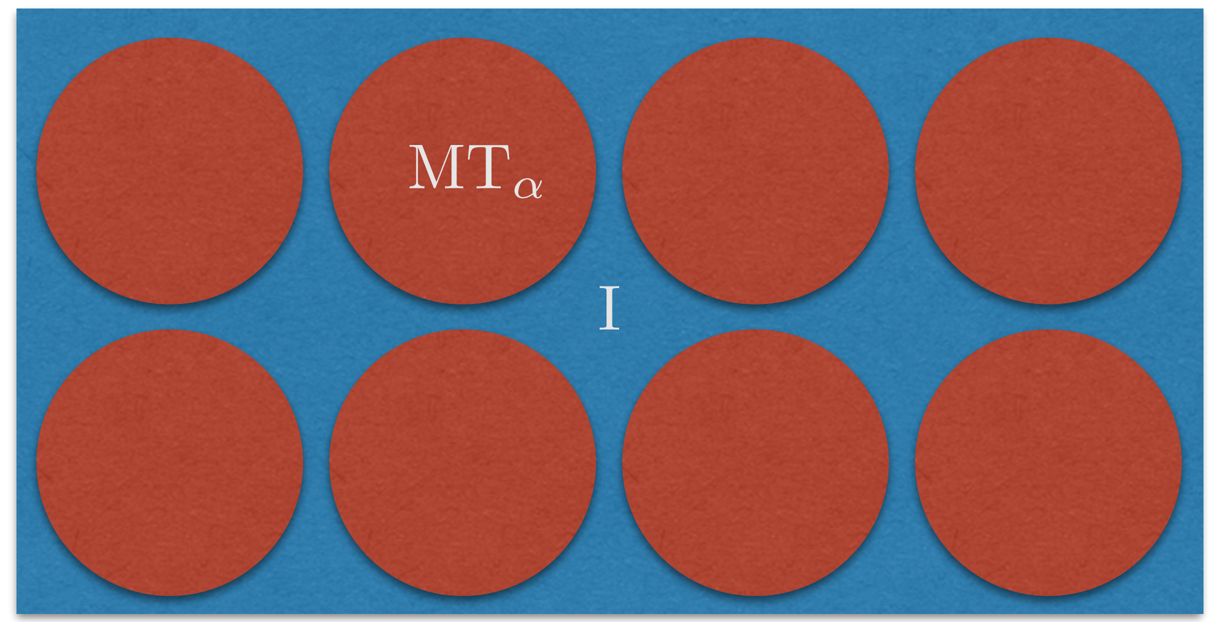
where we are usually interested in about 10-50% of spectrum

We need eigenvectors as well, to compute the density:

$$n(\xi) = \sum_{i=1}^{N} \psi_i^*(\xi)\psi_i(\xi)$$

Generalised eigenvalue problem in the LAPW

$$\sum_{G'} H^k_{GG'} C^{ik}_{G'} = \epsilon_{ik} \sum_{G'} O^k_{GG'} C^{ik}_{G'}$$



Overlap: $O^k_{GG'} = \langle \varphi_{G+k} | \varphi_{G'+k} \rangle$

Hamiltonian: $H^k_{GG'} = \langle \varphi_{G+k} | \hat{H} | \varphi_{G'+k} \rangle$

LAPW basis:

$$\varphi_{G+k}(r) = \begin{cases} \displaystyle\sum_L \sum_{\nu=1}^{O^\alpha_\ell} A^k_{\alpha L \nu}(G) u^\alpha_{\ell\nu}(r) Y_L(\hat{r}) & r \in \mathrm{MT}_\alpha \\ \displaystyle\frac{1}{\sqrt{\Omega}} e^{i(G+k)r} & r \in \mathrm{I} \end{cases}$$

Generalised eigenvalue problem in the LAPW (cont.)

$$\sum_{G'} H^k_{GG'} C^{ik}_{G'} = \epsilon_{ik} \sum_{G'} O^k_{GG'} C^{ik}_{G'}$$

Overlap:

$$O^k_{GG'} = \langle \varphi_{G+k} | \varphi_{G'+k} \rangle$$

$$= \sum_{\alpha L \nu} A^{k*}_{\alpha L \nu}(G) A^k_{\alpha L \nu}(G') + \Theta(G - G')$$

Hamiltonian:

$$H^k_{GG'} = \langle \varphi_{G+k} | \hat{H} | \varphi_{G'+k} \rangle$$

$$= \sum_{\alpha L \nu} A^{k*}_{\alpha L \nu}(G) B^k_{\alpha L \nu}(G') + \frac{1}{2}(G+k)(G'+k)\Theta(G - G') + \tilde{V}_s(G - G')$$

LAPW basis:

$$\varphi_{G+k}(r) = \left\{ \begin{array}{ll} \sum_{L} \sum_{\nu=1}^{O^\alpha_\ell} A^k_{\alpha L \nu}(G) u^\alpha_{\ell \nu}(r) Y_L(\hat{r}) & r \in \mathrm{MT}_\alpha \\ \frac{1}{\sqrt{\Omega}} e^{i(G+k)r} & r \in \mathrm{I} \end{array} \right\}$$

Generalised eigenvalue problem in the LAPW (cont.)

$$\sum_{G'} H^k_{GG'} C^{ik}_{G'} = \epsilon_{ik} \sum_{G'} O^k_{GG'} C^{ik}_{G'}$$

$O(N^3)$ complexity

Overlap: $$O^k_{GG'} = \langle \varphi_{G+k} | \varphi_{G'+k} \rangle$$

$$= \sum_{\alpha L \nu} A^{k*}_{\alpha L \nu}(G) A^k_{\alpha L \nu}(G') + \Theta(G - G')$$

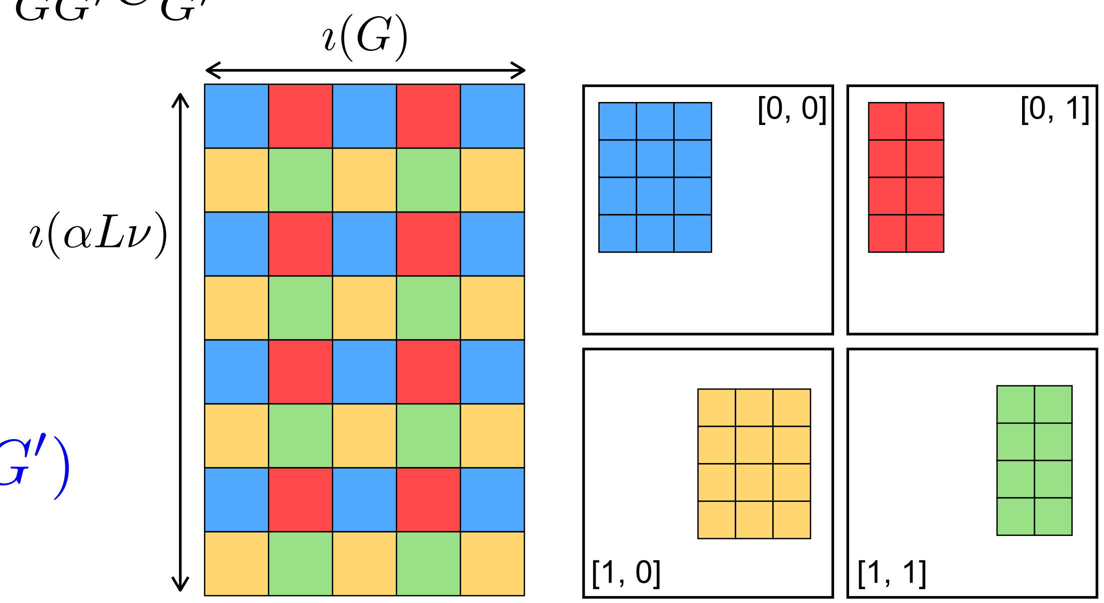
Hamiltonian: $$H^k_{GG'} = \langle \varphi_{G+k} | \hat{H} | \varphi_{G'+k} \rangle$$

$$= \sum_{\alpha L \nu} A^{k*}_{\alpha L \nu}(G) B^k_{\alpha L \nu}(G') + \frac{1}{2}(G+k)(G'+k)\Theta(G-G') + \tilde{V}_s(G-G')$$

$$B^k_{\alpha L \nu}(G) = \sum_{L_3 L_2 \nu_2} A^k_{\alpha L_2 \nu_2}(G) h^{\alpha l \nu}_{L_3 l_2 \nu_2} \langle Y_L | R_{L_3} | Y_{L_2} \rangle + \frac{1}{2} \sum_{\nu_2} A^k_{\alpha L \nu_2} u^{\alpha}_{l\nu}(R_\alpha) u'^{\alpha}_{l\nu_2}(R_\alpha) R^2_\alpha$$

Generalised eigenvalue problem in the LAPW (cont.)

$$\sum_{G'} H^k_{GG'} C^{ik}_{G'} = \epsilon_{ik} \sum_{G'} O^k_{GG'} C^{ik}_{G'}$$



Overlap: $O^k_{GG'} = \langle \varphi_{G+k} | \varphi_{G'+k} \rangle$

$$= \sum_{\alpha L\nu} A^{k*}_{\alpha L\nu}(G) A^k_{\alpha L\nu}(G') + \Theta(G - G')$$

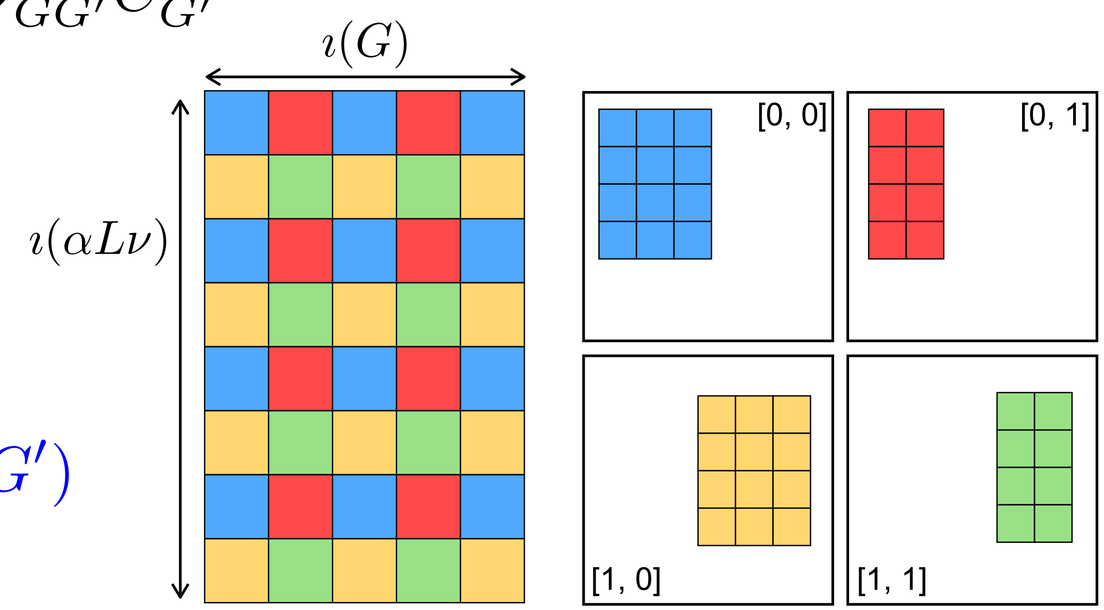
Hamiltonian: $H^k_{GG'} = \langle \varphi_{G+k} | \hat{H} | \varphi_{G'+k} \rangle$

$$= \sum_{\alpha L\nu} A^{k*}_{\alpha L\nu}(G) B^k_{\alpha L\nu}(G') + \frac{1}{2}(G+k)(G'+k)\Theta(G-G') + \tilde{V}_s(G-G')$$

$$B^k_{\alpha L\nu}(G) = \sum_{L_3 L_2 \nu_2} A^k_{\alpha L_2 \nu_2}(G) h^{\alpha l \nu}_{L_3 l_2 \nu_2} \langle Y_L | R_{L_3} | Y_{L_2} \rangle + \frac{1}{2} \sum_{\nu_2} A^k_{\alpha L \nu_2} u^{\alpha}_{l\nu}(R_\alpha) u'^{\alpha}_{l\nu_2}(R_\alpha) R^2_\alpha$$

Generalised eigenvalue problem in the LAPW (cont.)

$$\sum_{G'} H^k_{GG'} C^{ik}_{G'} = \epsilon_{ik} \sum_{G'} O^k_{GG'} C^{ik}_{G'}$$

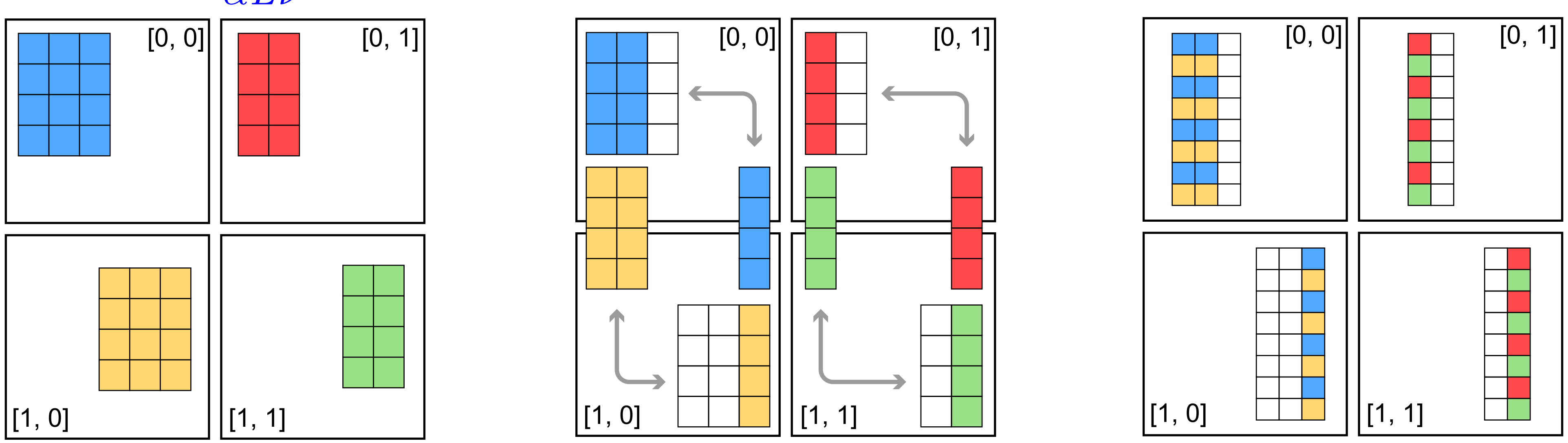


Overlap: $$O^k_{GG'} = \langle \varphi_{G+k} | \varphi_{G'+k} \rangle$$

$$= \sum_{\alpha L \nu} A^{k*}_{\alpha L \nu}(G) A^k_{\alpha L \nu}(G') + \Theta(G - G')$$

Hamiltonian: $$H^k_{GG'} = \langle \varphi_{G+k} | \hat{H} | \varphi_{G'+k} \rangle$$

$$= \sum_{\alpha L \nu} A^{k*}_{\alpha L \nu}(G) B^k_{\alpha L \nu}(G') + \frac{1}{2}(G+k)(G'+k)\Theta(G-G') + \tilde{V}_s(G-G')$$



Solving the generalised eigenvalue problem

$$\mathbf{Ax} = \lambda \mathbf{Bx}$$

xPOTRF $\quad \mathbf{B} = \mathbf{LL}^H$

xHEGST $\quad \mathbf{A}' = \mathbf{L}^{-1}\mathbf{AL}^{-H}$

xHEEVx $\quad \mathbf{A}'\mathbf{y} = \lambda \mathbf{y}$

xTRSM $\quad \mathbf{x} = \mathbf{L}^{-H}\mathbf{y}$

Standard 1 stage solver

$$\mathbf{A}'\mathbf{y} = \lambda \mathbf{y}$$

$\mathbf{T} = \mathbf{Q}^H\mathbf{A}'\mathbf{Q} \quad$ xHETRD

Most time consuming step, dominated by level 2 BLAS (memory bound)

$\mathbf{Ty}' = \lambda \mathbf{y}' \quad$ xSTExx

$\mathbf{y} = \mathbf{Qy}' \quad$ xUNMTR

Solving the generalised eigenvalue problem (cont.)

$$\mathbf{Ax} = \lambda \mathbf{Bx}$$

↓

xPOTRF $$\mathbf{B} = \mathbf{LL}^H$$

↓

xHEGST $$\mathbf{A}' = \mathbf{L}^{-1}\mathbf{AL}^{-H}$$

↓

xHEEVx $$\mathbf{A}'\mathbf{y} = \lambda \mathbf{y}$$

↓

xTRSM $$\mathbf{x} = \mathbf{L}^{-H}\mathbf{y}$$

$$\mathbf{A}'\mathbf{y} = \lambda \mathbf{y}$$

↓

reduction to banded

$$\mathbf{A}'' = \mathbf{Q_1}^H \mathbf{A}'\mathbf{Q_1}$$

Most time consuming step, but dominated by BLAS-3

↓

tri-diagonalize

$$\mathbf{T} = \mathbf{Q_2}^H \mathbf{A}''\mathbf{Q_2}$$

↓

$$\mathbf{Ty}' = \lambda \mathbf{y}'$$

↓

$$\mathbf{y}'' = \mathbf{Q_2 y}'$$

↓

$$\mathbf{y} = \mathbf{Q_1 y}''$$

needs two eigenvector transformations (but easy to parallelise)

Implementations of two-stage eigen solvers for our problem (i.e. with back transformation of eigenvectors)

For multi-cores systems: ELPA library

T. Auckenthaler et al., Parallel Comput. vol. 37, no. 12, pp. 783-794 (2011)

A. Marek et al., Psi-K Research Highlight, vol. 2014, no. 1, Jan. 2014

Remark: built on top of ScaLapack

For hybrid CPU-GPU systems: integrated into MAGMA library

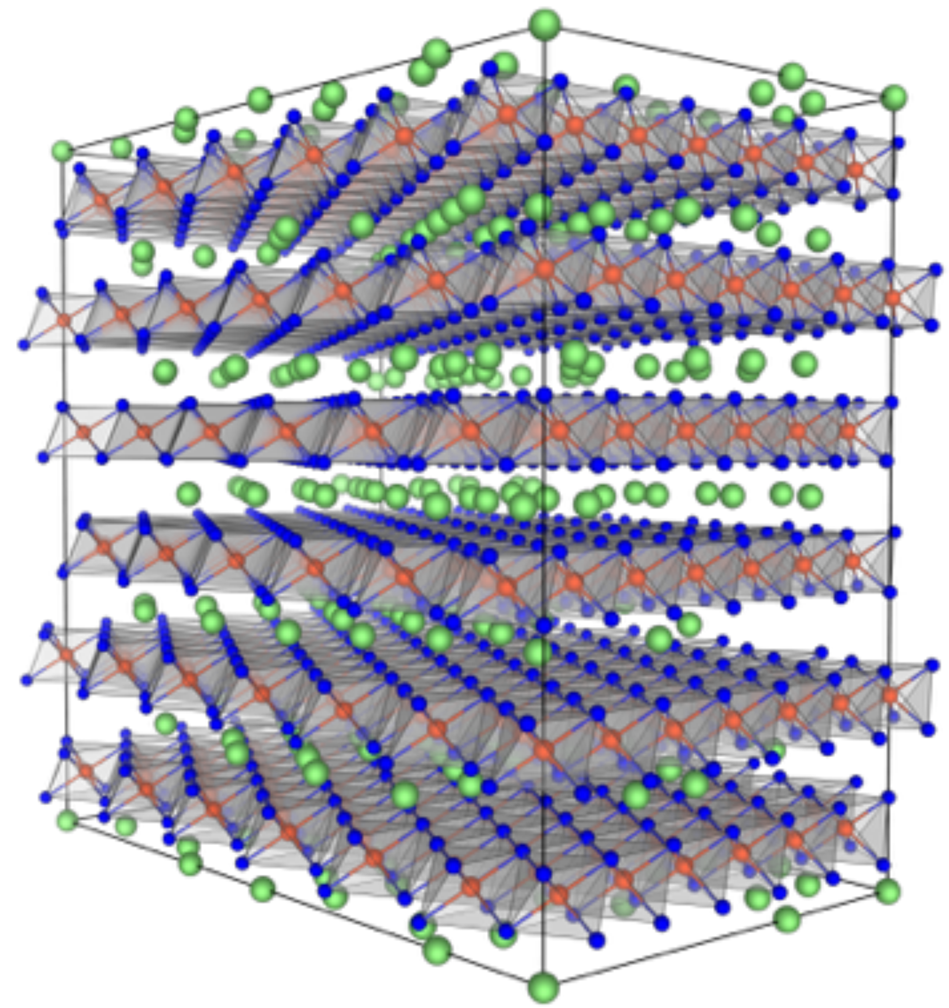
A. Haidar et al., Lecture Notes in Comp. Sci., 7905, 67-80 (2013)

A. Haidar et al., Int. J. of High Perf. Comp. App. 10.1177/1094342013502097 (2013)

R. Solcà et al., in preparation (2015)

Remark: distributed version built on top of a distributed implementation of libsciACC

1000-atom test problem



~115,000 basis functions (matrix size)

Running on Cray XC30:
> CPU runs on Xeon E5-2670 (Sandy Bridge)
> hybrid: same CPU + Nvidia K20X GPU

User comparable number of sockets

Li intercalated CoO_2:
- 432 formula units CoO_2
- 205 Li atoms
- **1501 atoms in total**

Results for the full runs (on SCF iteration)

MPI grid → MPI ranks / socket → OpenMP threads / rank

	active sockets	setup, O H (sec.)	solve (sec.)	rest (sec.)	total (sec.)	energy (kWh)
28x28 (2R:4T) ScaLAPACK	392	382.5	3166.8	69.2	3618.5	39.46
28x28 (2R:4T) ELPA2	392	383.2	705.3	63.6	**1152.1**	**17.40**
20x20 (1R:8T) ELPA2	400	374.0	720.5	61.1	**1155.6**	**16.9**
14x14 (1R:8T) hybrid	392	159.9	741.8	84.8	**986.5**	**8.27**
20x20 (1R:8T) hybrid	800	96.9	652.1	58.9	**807.9**	**12.49**

Resources used 1000-atom design problem

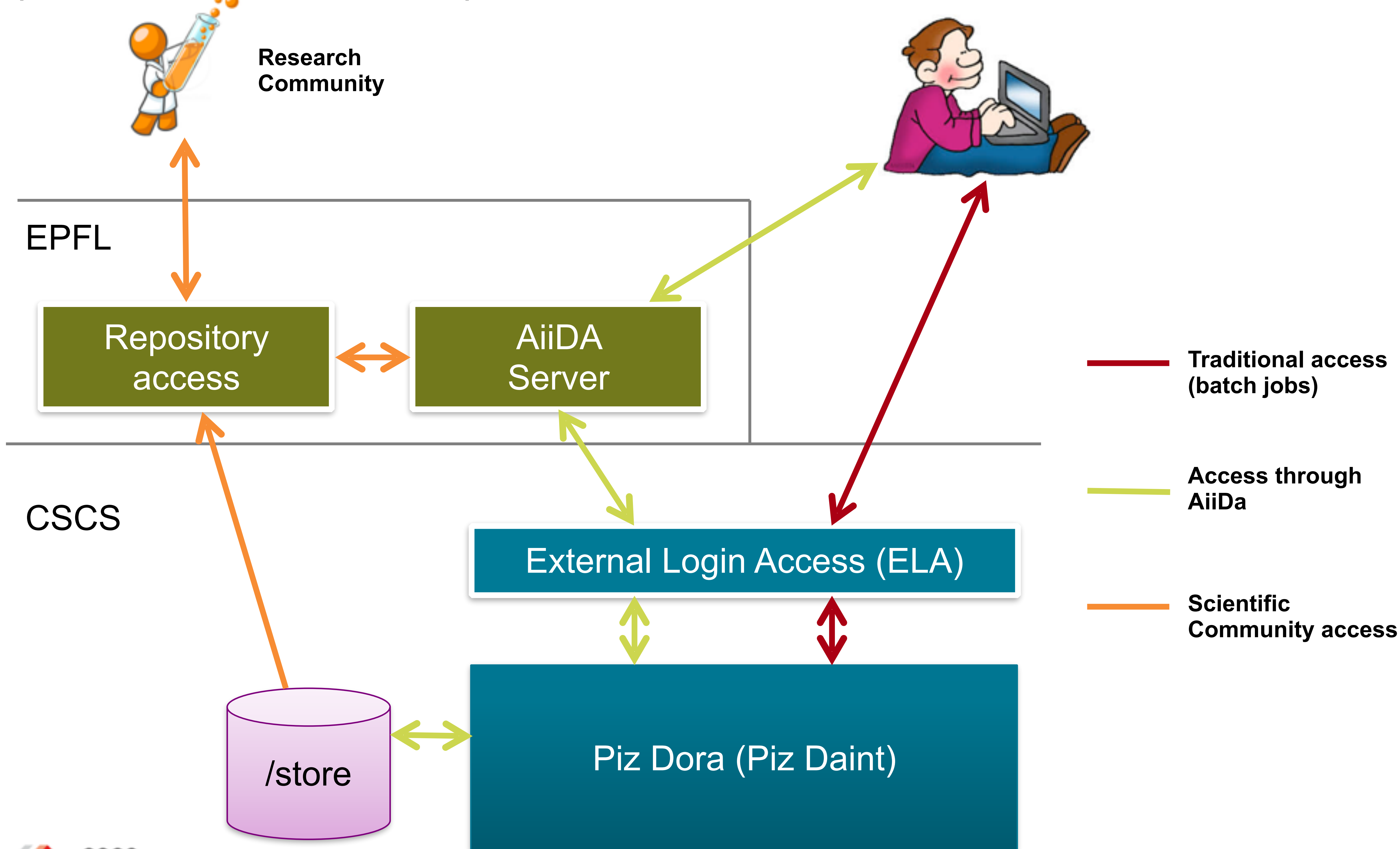
Time: ~15 minutes / iteration, i.e. 3 hours for ~10 iterations

Footprint: ~400 hybrid nodes on Cray XC30 (SandyBride+K20X)

Scan ~13 materials in 3 hours or 5,000 in ~16 days

(consider performance will improve 10-100x in by end of decade)

NCCR MARVEL: data science for materials design
(EPFL / CSCS collaboration)



Heterogenous Supercomputing Platform @ CSCS

"Piz Daint" with 28 cabinets Cray XC30 hybrid and 7 cabinets Cray XC40

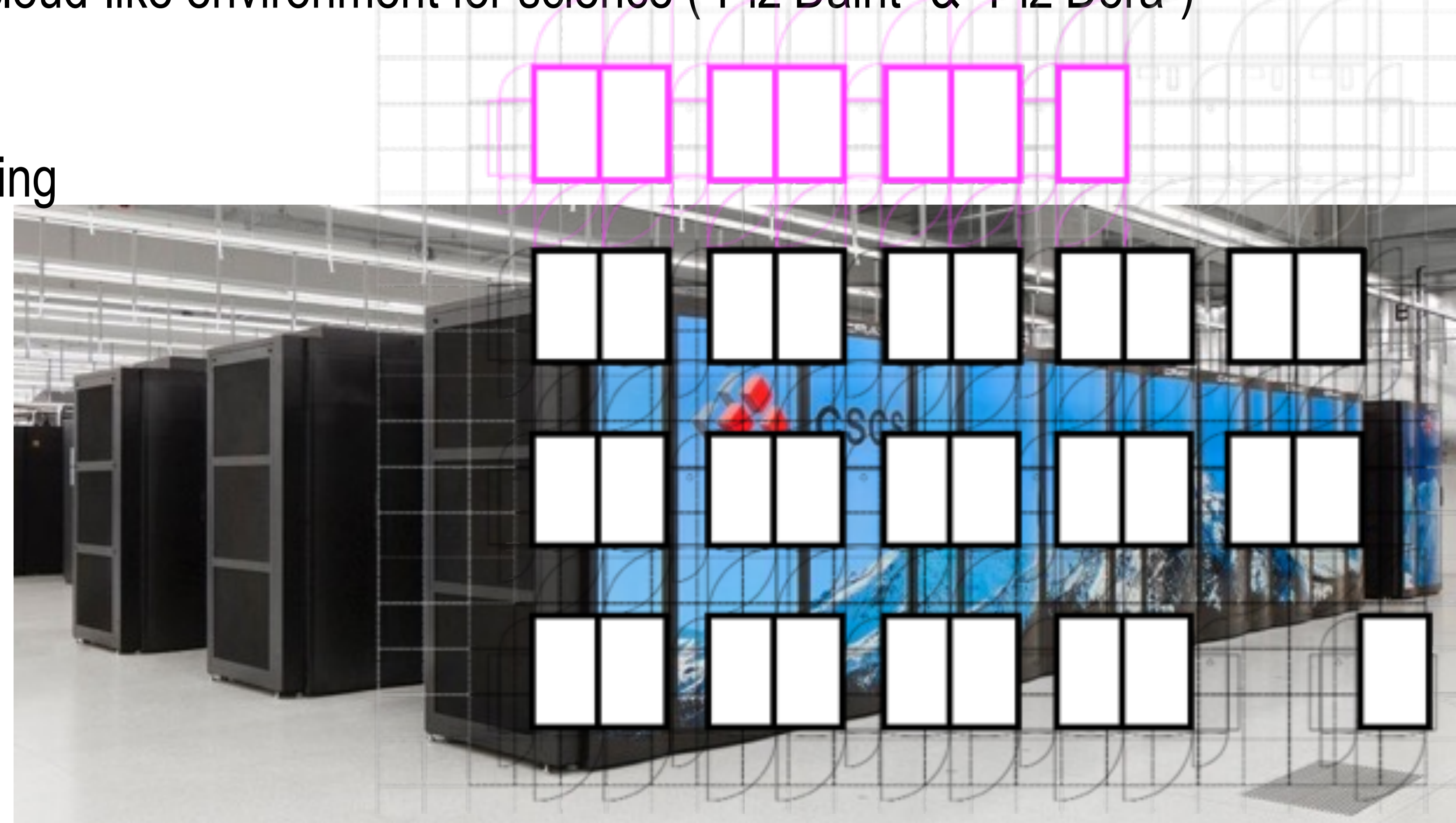
Cray XC40 is hosting:

– "Monte Rosa" replacement in the User Lab
– Replacement of pre- and post processing cluster of the user lab
– Successor of "Schrödinger" cluster for U. of ZH
– Cluster resources of U. of Lugano and PSI
– BigData analytics Cluster for ETH Zurich
– **Cluster and data resource for NCCR project MARVEL (materials design)**

Take pressure off infrastructure at Universities and Labs, to facilitate consolidation of their institution-wide computing infrastructure

Cray XC @ CSCS – a heterogeneous cloud-like environment for science ("Piz Daint" & "Piz Dora")
– hybrid CPU-CPU nodes (Piz Daint)
– CPU only nodes (Piz Dora)
– large memory nodes for data processing
– SSD-based I/O burst buffers
– very low latency network
– high bisection bandwidth

But isn't this expensive?
– no, it is much cheaper or we wouldn't do it this way!



Anton Kozhevnikov
with
Claudia Draxl,
Andris Gulans,
and Georg Huhs

SIRIUS: Domain Specific Library

Low-level LAPW (and PW) library that supports multiple codes

~30k lines of C++ code (incl. documentation) with F90 bindings

Exciting	Elk	other (e.g. QE)

SIRIUS C++ library

MPI + OpenMP parallel model with GPU acceleration

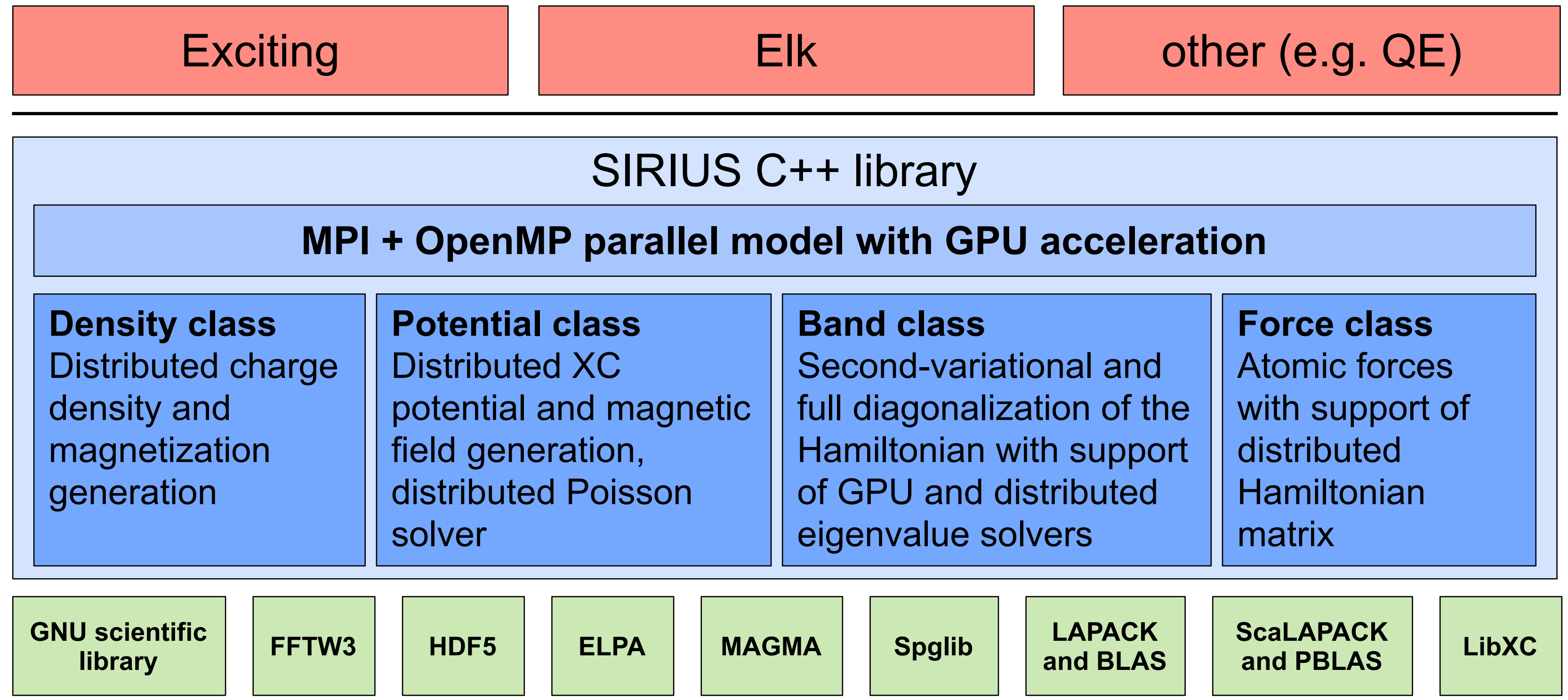
Density class	**Potential class**	**Band class**	**Force class**
Distributed charge density and magnetization generation	Distributed XC potential and magnetic field generation, distributed Poisson solver	Second-variational and full diagonalization of the Hamiltonian with support of GPU and distributed eigenvalue solvers	Atomic forces with support of distributed Hamiltonian matrix

GNU scientific library	**FFTW3**	**HDF5**	**ELPA**	**MAGMA**	**Spglib**	**LAPACK and BLAS**	**ScaLAPACK and PBLAS**	**LibXC**

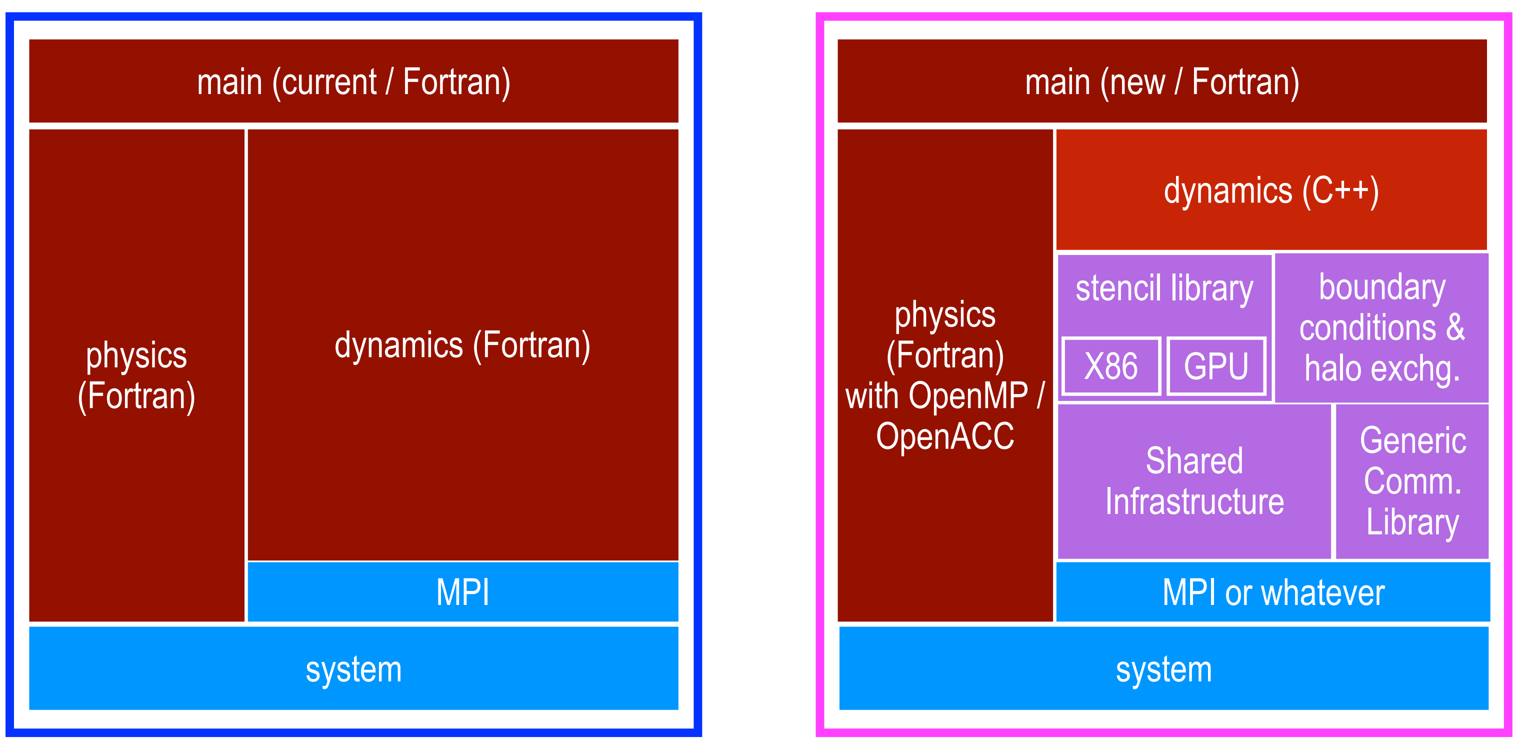
References and Collaborators

- Peter Messmer and his team at the NVIDIA co-design lab at ETH Zurich
- Teams at CSCS
- A. Haidar, R. Solcà, M. Gates, T. Tomov, T.C. Schulthess, J. Dongarra, "**Leading Edge Hybrid Multi-GPU Algorithms for Generalized Eigenproblems in Electronic Structure Calculations**", Supercomputing, pages 67-80 Springer Berlin, Heidelberg (2013)
- A. Haidar, S. Tomov, J. Dongarra, R. Solcà, T. C. Schulthess, "**A novel hybrid CPU-GPU generalised eigensolver for electronic structure calculations based on fine grained memory aware tasks**", International Journal of High Performance Computing Applications, August 2013
- R. Solcà, A. Kozhevnikov, A. Haidar, S. Tomov, J. Dongarra, T. C. Schulthess, "**Efficient implementation of quantum materials simulations on distributed CPU-GPU systems**", to be published in Proceedings of the International Conference on High-Performance Computing, Networking, Storage and Analysis, SC'15, New York, NY, USA (2015). ACM
- R. Solcà and T. C. Schulthess, "**Energy and Compute Resource Modelling in Complex Parallel Applications**", in preparation 2015

The real problem is software



COSMO: current and new (HP



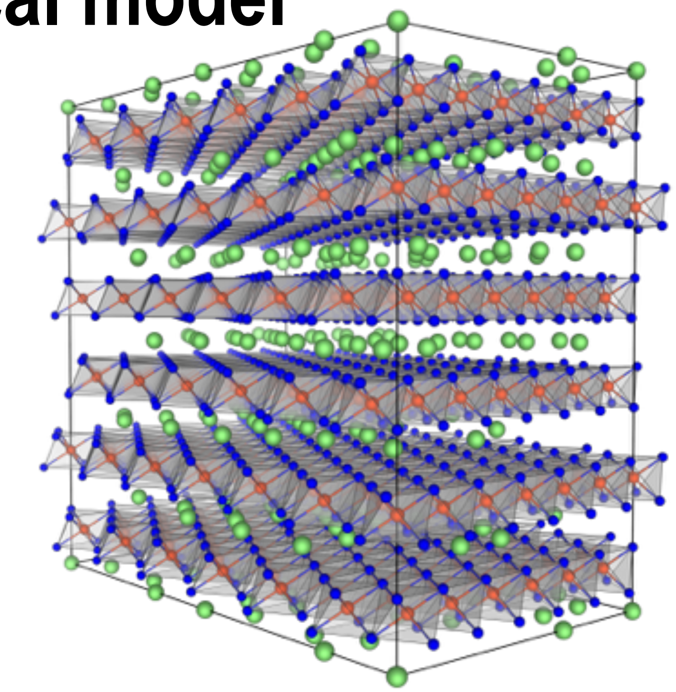
DFT ground state

Eigen-value problem $\hat{H}\Psi_{i\mathbf{k}}(\mathbf{r}) = \epsilon_{i\mathbf{k}}\Psi_{i\mathbf{k}}(\mathbf{r})$

Charge density $\rho_{\sigma\sigma'}(\mathbf{r}) = \sum_{i\mathbf{k}}^{occ} \Psi^{*\sigma'}_{i\mathbf{k}}(\mathbf{r})\Psi^{\sigma}_{i\mathbf{k}}(\mathbf{r})$

Effective potential $v^{eff}(\mathbf{r}) = v^{eff}[\rho_{\sigma\sigma'}(\mathbf{r})](\mathbf{r})$
and magnetic field $\mathbf{B}^{eff}(\mathbf{r}) = \mathbf{B}^{eff}[\rho_{\sigma\sigma'}(\mathbf{r})](\mathbf{r})$

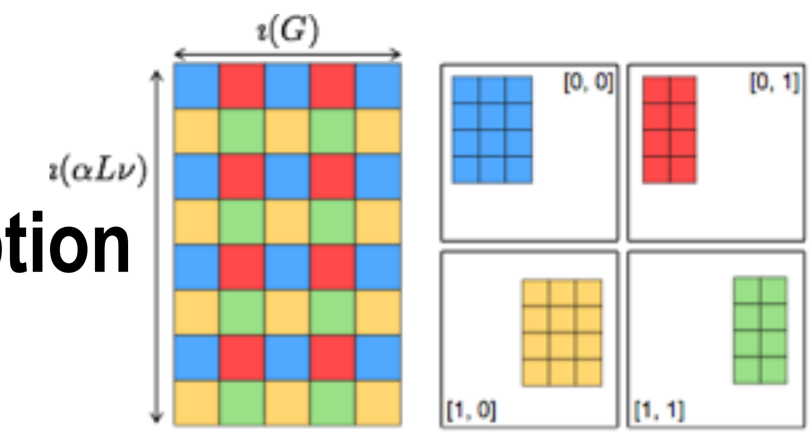
Physical model



Mathematical description

Domain science & applied mathematics

Algorithmic description



Imperative code

Compilation

Computer

Computer engineering

Schulthess, Nature Physics, May 2015

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

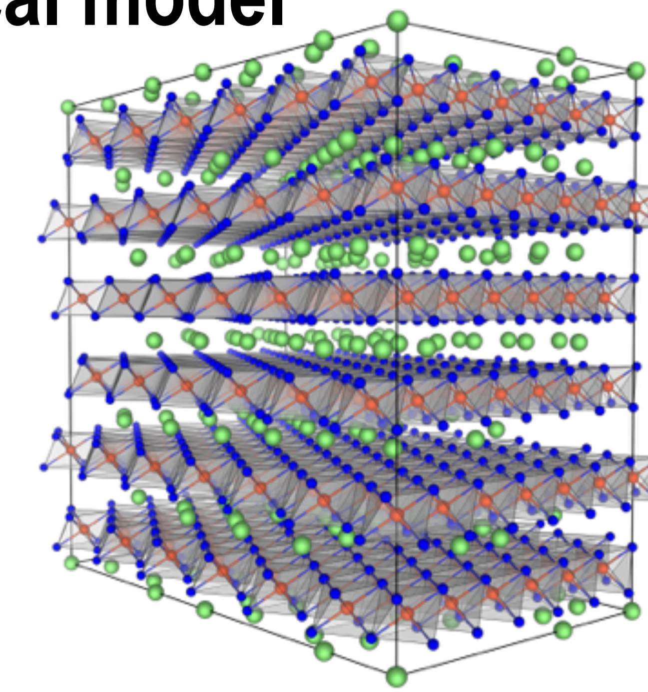
DFT ground state

Eigen-value problem $\hat{H}\Psi_{i\mathbf{k}}(\mathbf{r}) = \epsilon_{i\mathbf{k}}\Psi_{i\mathbf{k}}(\mathbf{r})$

Charge density $\rho_{\sigma\sigma'}(\mathbf{r}) = \sum_{i\mathbf{k}}^{occ} \Psi_{i\mathbf{k}}^{*\sigma'}(\mathbf{r})\Psi_{i\mathbf{k}}^{\sigma}(\mathbf{r})$

Effective potential and magnetic field $v^{eff}(\mathbf{r}) = v^{eff}[\rho_{\sigma\sigma'}(\mathbf{r})](\mathbf{r})$
$\mathbf{B}^{eff}(\mathbf{r}) = \mathbf{B}^{eff}[\rho_{\sigma\sigma'}(\mathbf{r})](\mathbf{r})$

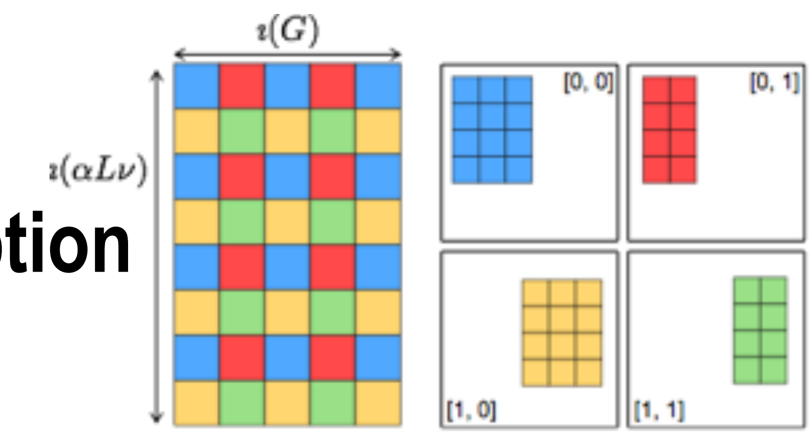
Physical model



Mathematical description

Domain science & applied mathematics

Algorithmic description



Imperative code

Compilation

Computer

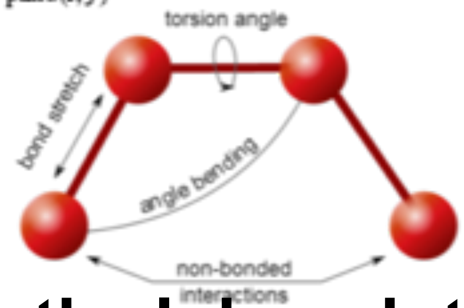
Computer engineering

Schulthess, Nature Physics, May 2015

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

$$V(r) = \sum_{\text{bonds}} k_b(b-b_0)^2 + \sum_{\text{angles}} k_\theta(\theta-\theta_0)^2 + \sum_{\text{dihedrals}} k_\phi(1+\cos(n\phi-\phi_0)) + \sum_{\text{impropers}} k_\psi(\psi-\psi_0)^2 + \sum_{\substack{\text{non-bonded}\\ \text{pairs}(i,j)}} 4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] + \sum_{\substack{\text{non-bonded}\\ \text{pairs}(i,j)}} \frac{q_i q_j}{\varepsilon_D r_{ij}}$$

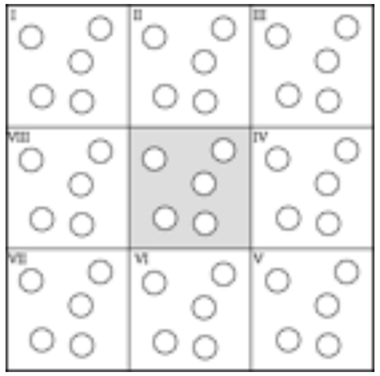


Physical model

Mathematical description

Domain science & applied mathematics

Algorithmic description




```
Cray PAT API "11 loop init mtran" in UKCA_CDAGWITHNUCL
      call pat_region_begin(11,'11 loop init mtran',pat_stat)
!
!!!        DO IMODE=1,NMODES
!!!         NDOLD(:,IMODE)=ND(:,IMODE)
!!!         DO ICP=1,NCP
!!!          MDOLD(:,IMODE,ICP)=MD(:,IMODE,ICP)
!!!          DO JMODE=1,NMODES
!!!           MTRAN(:,IMODE,JMODE,ICP)=0.0
!!!          ENDDO
!!!         ENDDO
!!!        ENDDO
!
! replace triple loops above with F90 array syntax (let
compiler decide)
      NDOLD=ND
      MDOLD=MD
      MTRAN=0.0
      call pat_region_end(11,pat_stat)
```

Imperative code

Compilation

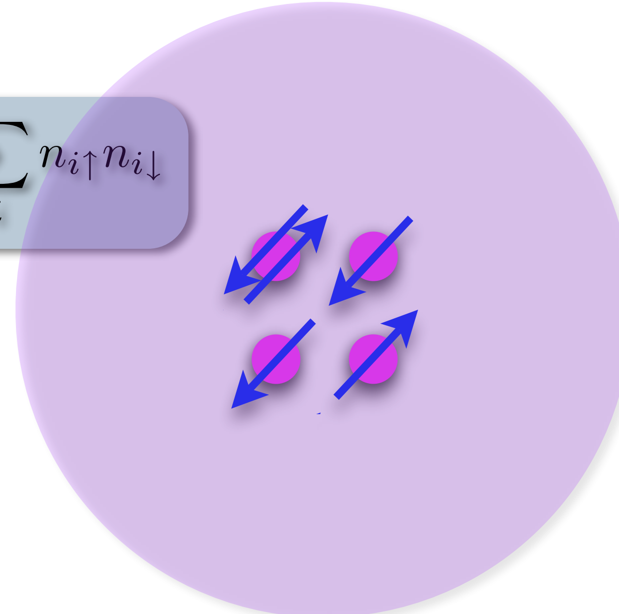
Computer

Computer engineering

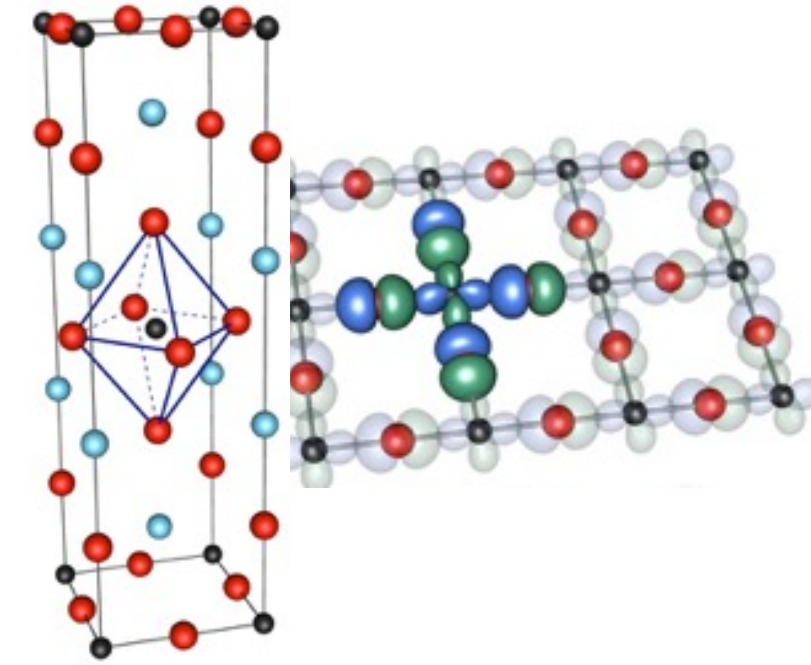
Schulthess, Nature Physics, May 2015

ETH zürich

$$\mathcal{H} = -t \sum_{\langle ij\rangle,\sigma} c^{\dagger}_{i\sigma} c_{j\sigma} + U \sum_{i} n_{i\uparrow} n_{i\downarrow}$$



Physical model

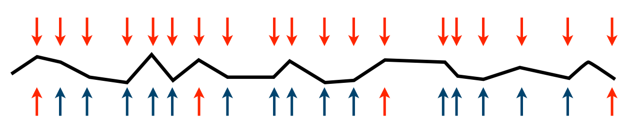


Mathematical description

Domain science & applied mathematics

Algorithmic description

$$\mathbf{G}_c(\{s_i,l\}_{k+1}) = \mathbf{G}_c(\{s_i,l\}_k) + \mathbf{a}_k \times \mathbf{b}_k^t$$



$$\mathbf{G}_c(\{s_i,l\}_{k+1}) = \mathbf{G}_c(\{s_i,l\}_0) + [\mathbf{a}_0|\mathbf{a}_1|...|\mathbf{a}_k] \times [\mathbf{b}_0|\mathbf{b}_1|...|\mathbf{b}_k]^t$$

Imperative code

Compilation

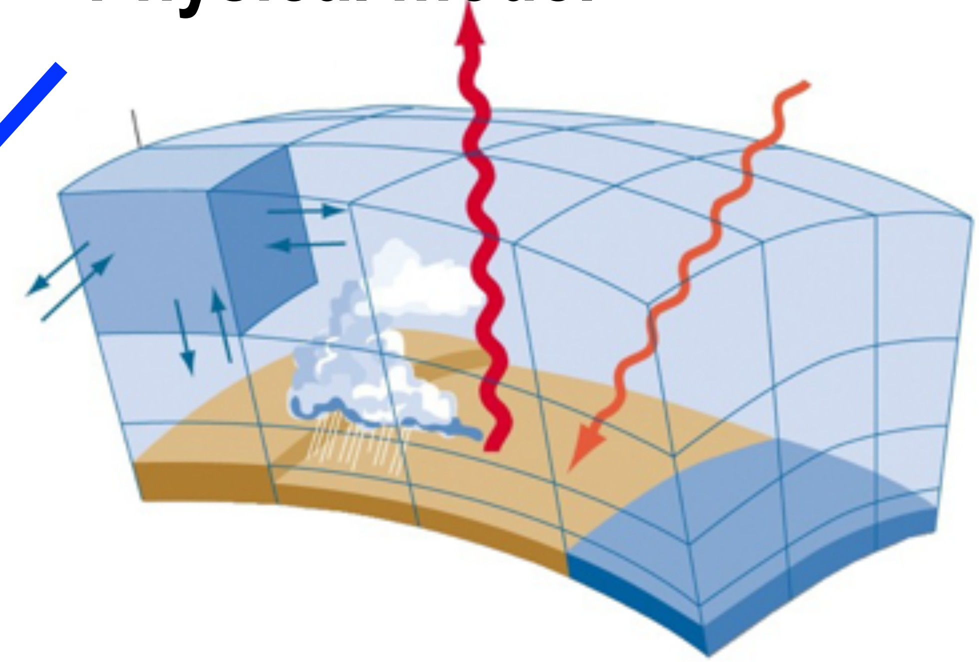
Computer

Computer engineering

Schulthess, Nature Physics, May 2015

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

Physical model



Wind $\rho\dot{\mathbf{v}} = -\nabla p + \rho\mathbf{g} - 2\Omega\times(\rho\mathbf{v}) + \mathbf{F}$

Pressure $\dot{p} = -(c_{pd}/c_{vd})\, p\nabla\cdot\mathbf{v} + (c_{pd}/c_{vd}-1)Q_h$

Temperature $\rho c_{pd}\dot{T} = \dot{p} + Q_h$

Water $\rho\dot{q}^v = -\nabla\cdot\mathbf{F}^v - (I^l + I^f)$

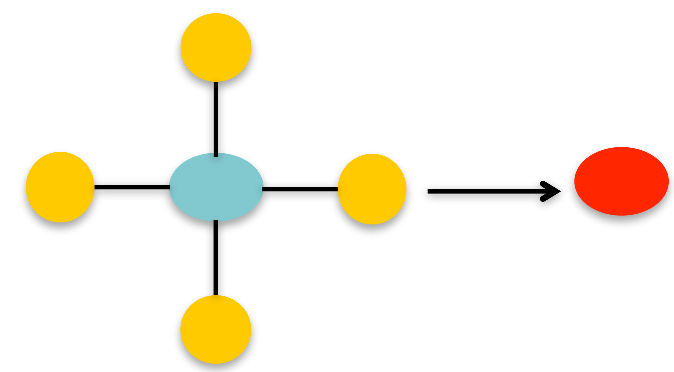
$\rho\dot{q}^{l,f} = \nabla\cdot(\mathbf{P}^{l,f} + \mathbf{F}^{l,f}) + I^{l,f}$

Density $\rho = p[R_d\,(1+(R_v/R_d-1)\,q^v - q^l - q^f)\,T]^{-1}$

Mathematical description

Domain science & applied mathematics

Algorithmic description



```
lap(i,j,k) = -4.0 * data(i,j,k) +
    data(i+1,j,k) + data(i-1,j,k) +
    data(i,j+1,k) + data(i,j-1,k);
```

Imperative code

Compilation

Computer

Computer engineering

Schulthess, Nature Physics, May 2015

ETH zürich

Wind $\rho\dot{\mathbf{v}} = -\nabla p + \rho\mathbf{g} - 2\Omega\times(\rho\mathbf{v}) + \mathbf{F}$

Pressure $\dot{p} = -(c_{pd}/c_{vd})\, p\nabla\cdot\mathbf{v} + (c_{pd}/c_{vd}-1)Q_h$

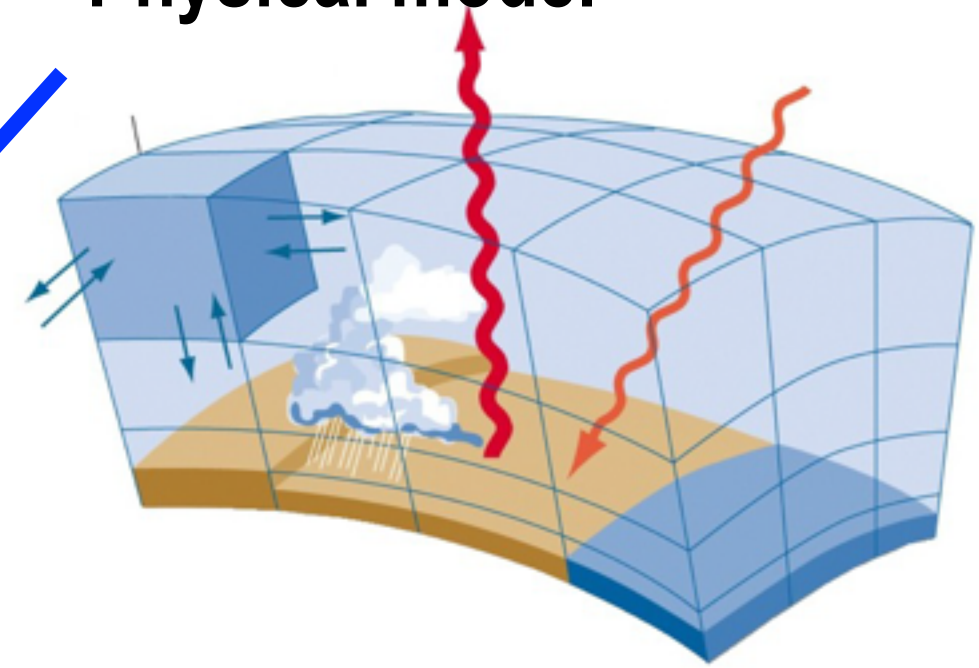
Temperature $\rho c_{pd}\dot{T} = \dot{p} + Q_h$

Water $\rho\dot{q}^{v} = -\nabla\cdot\mathbf{F}^{v} - (I^{l} + I^{f})$

$\rho\dot{q}^{l,f} = \nabla\cdot(\mathbf{P}^{l,f} + \mathbf{F}^{l,f}) + I^{l,f}$

Density $\rho = p[R_d(1+(R_v/R_d-1)q^{v} - q^{l} - q^{f})T]^{-1}$

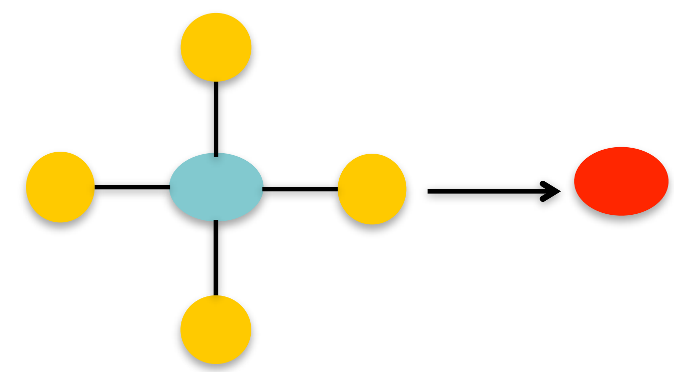
Physical model



Mathematical description

Domain science & applied mathematics

Algorithmic description



```
lap(i,j,k) = -4.0 * data(i,j,k) +
    data(i+1,j,k) + data(i-1,j,k) +
    data(i,j+1,k) + data(i,j-1,k);
```

Imperative code

Compilation

Computer engineering

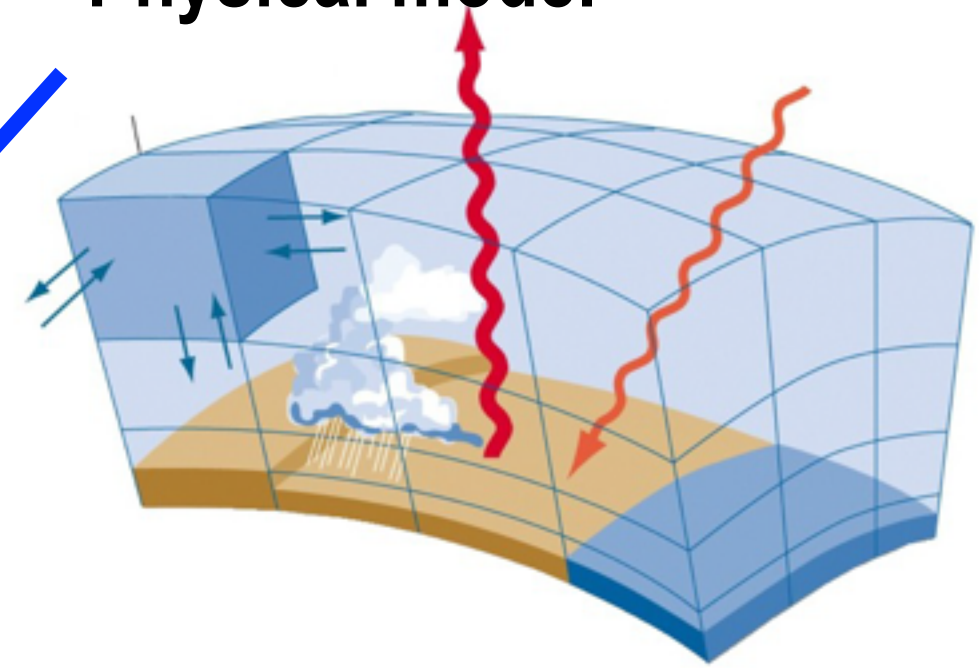
Computer architecture (X86 / Intel Xeon)

ulthess, Nature Physics, May 2015

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Physical model



Wind $\rho\dot{\mathbf{v}} = -\nabla p + \rho\mathbf{g} - 2\Omega\times(\rho\mathbf{v}) + \mathbf{F}$

Pressure $\dot{p} = -(c_{pd}/c_{vd})\, p\nabla\cdot\mathbf{v} + (c_{pd}/c_{vd}-1)Q_h$

Temperature $\rho c_{pd}\dot{T} = \dot{p} + Q_h$

Water $\rho\dot{q}^{v} = -\nabla\cdot\mathbf{F}^{v} - (I^{l} + I^{f})$

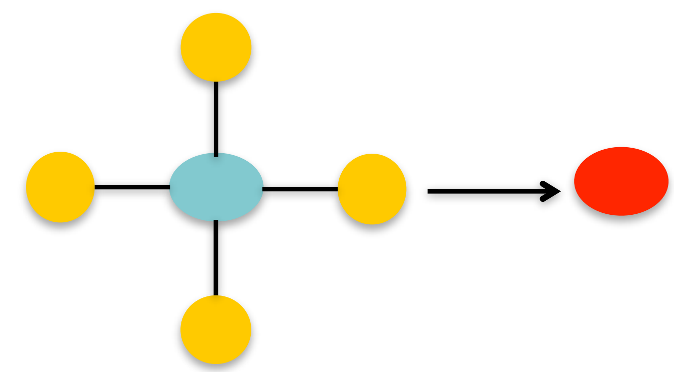
$\rho\dot{q}^{l,f} = \nabla\cdot(\mathbf{P}^{l,f} + \mathbf{F}^{l,f}) + I^{l,f}$

Density $\rho = p[R_d(1+(R_v/R_d-1)q^{v} - q^{l} - q^{f})T]^{-1}$

Mathematical description

Domain science & applied mathematics

Algorithmic description



```
lap(i,j,k) = -4.0 * data(i,j,k) +
    data(i+1,j,k) + data(i-1,j,k) +
    data(i,j+1,k) + data(i,j-1,k);
```

Imperative code

Compilation

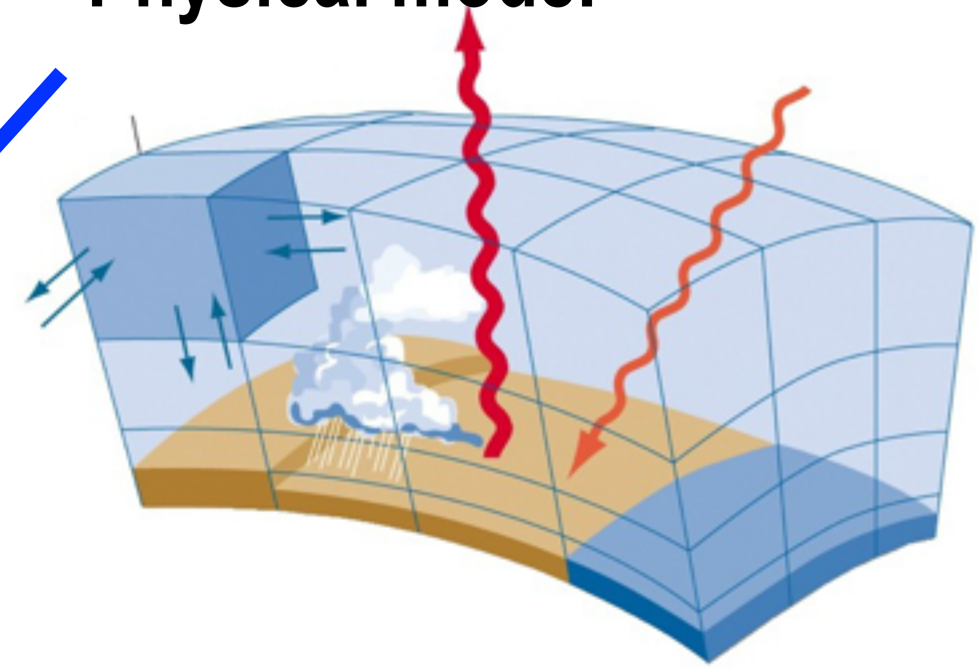
Computer engineering

Computer architecture (Intel Xeon-Phi)

chulthess, Nature Physics, May 2015

ETH zürich

Physical model



$$\text{Wind } \rho\dot{\mathbf{v}} = -\nabla p + \rho\mathbf{g} - 2\Omega\times(\rho\mathbf{v}) + \mathbf{F}$$

$$\text{Pressure } \dot{p} = -(c_{pd}/c_{vd})\, p\nabla\cdot\mathbf{v} + (c_{pd}/c_{vd}-1)Q_h$$

$$\text{Temperature } \rho c_{pd}\dot{T} = \dot{p} + Q_h$$

$$\text{Water } \rho\dot{q}^{v} = -\nabla\cdot\mathbf{F}^{v} - (I^{l} + I^{f})$$

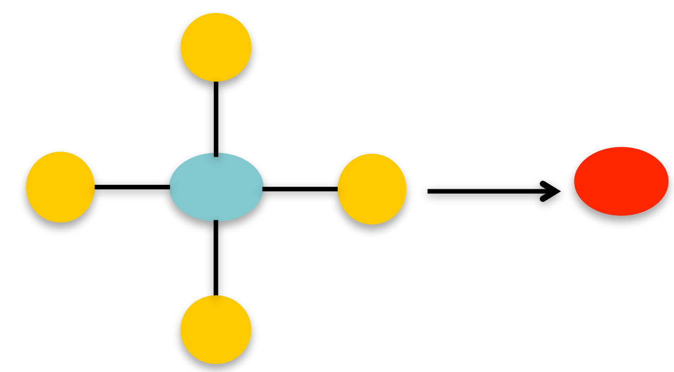
$$\rho\dot{q}^{l,f} = \nabla\cdot(\mathbf{P}^{l,f} + \mathbf{F}^{l,f}) + I^{l,f}$$

$$\text{Density } \rho = p[R_d\,(1+(R_v/R_d-1)\,q^{v}-q^{l}-q^{f})\,T]^{-1}$$

Mathematical description

Domain science & applied mathematics

Algorithmic description



```
lap(i,j,k) = -4.0 * data(i,j,k) +
    data(i+1,j,k) + data(i-1,j,k) +
    data(i,j+1,k) + data(i,j-1,k);
```

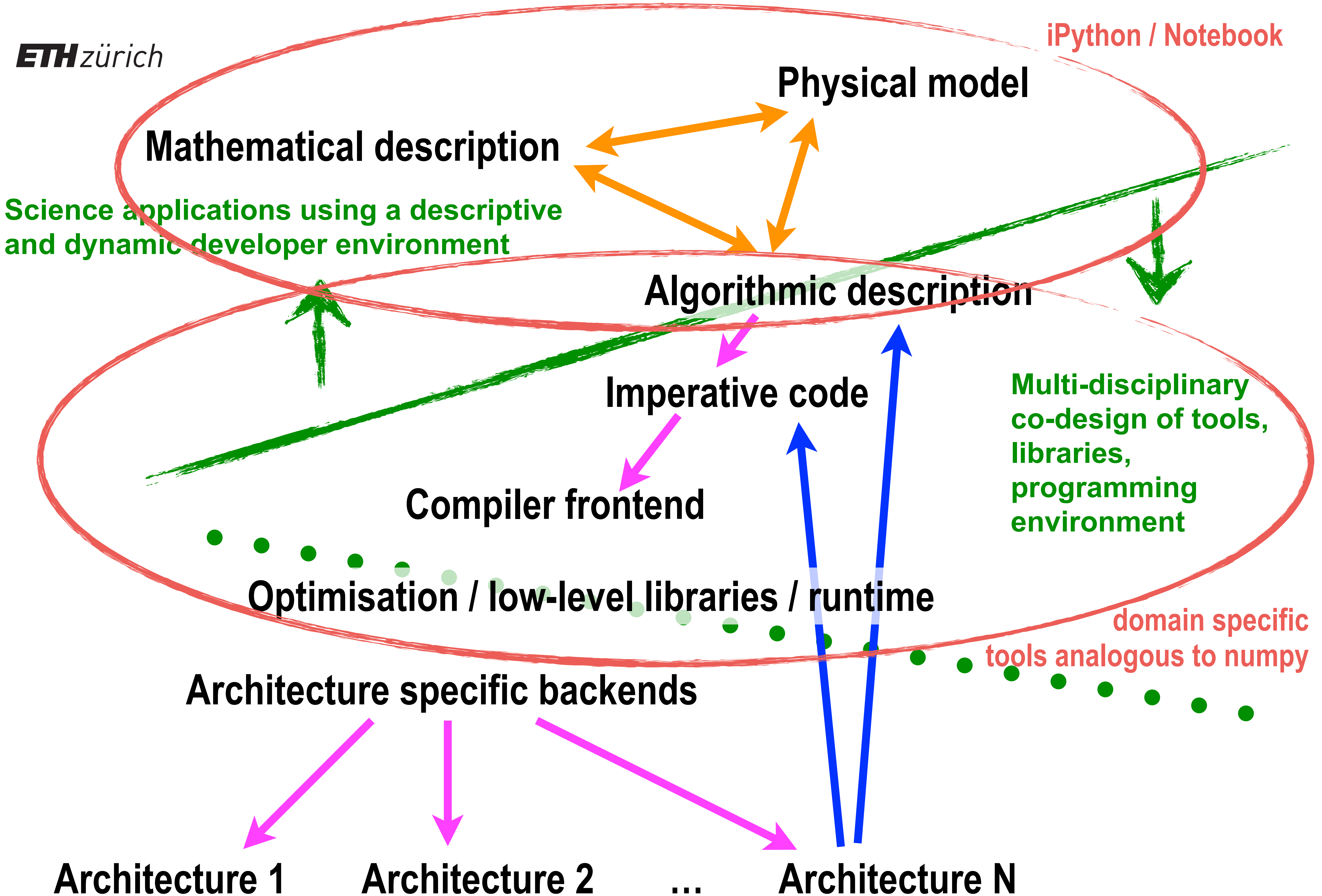
Imperative code

Compilation

Computer engineering

Computer architecture (NVIDA Tesla GPU)

…ess, Nature Physics, May 2015



Schulthess, Nature Physics, May 2015

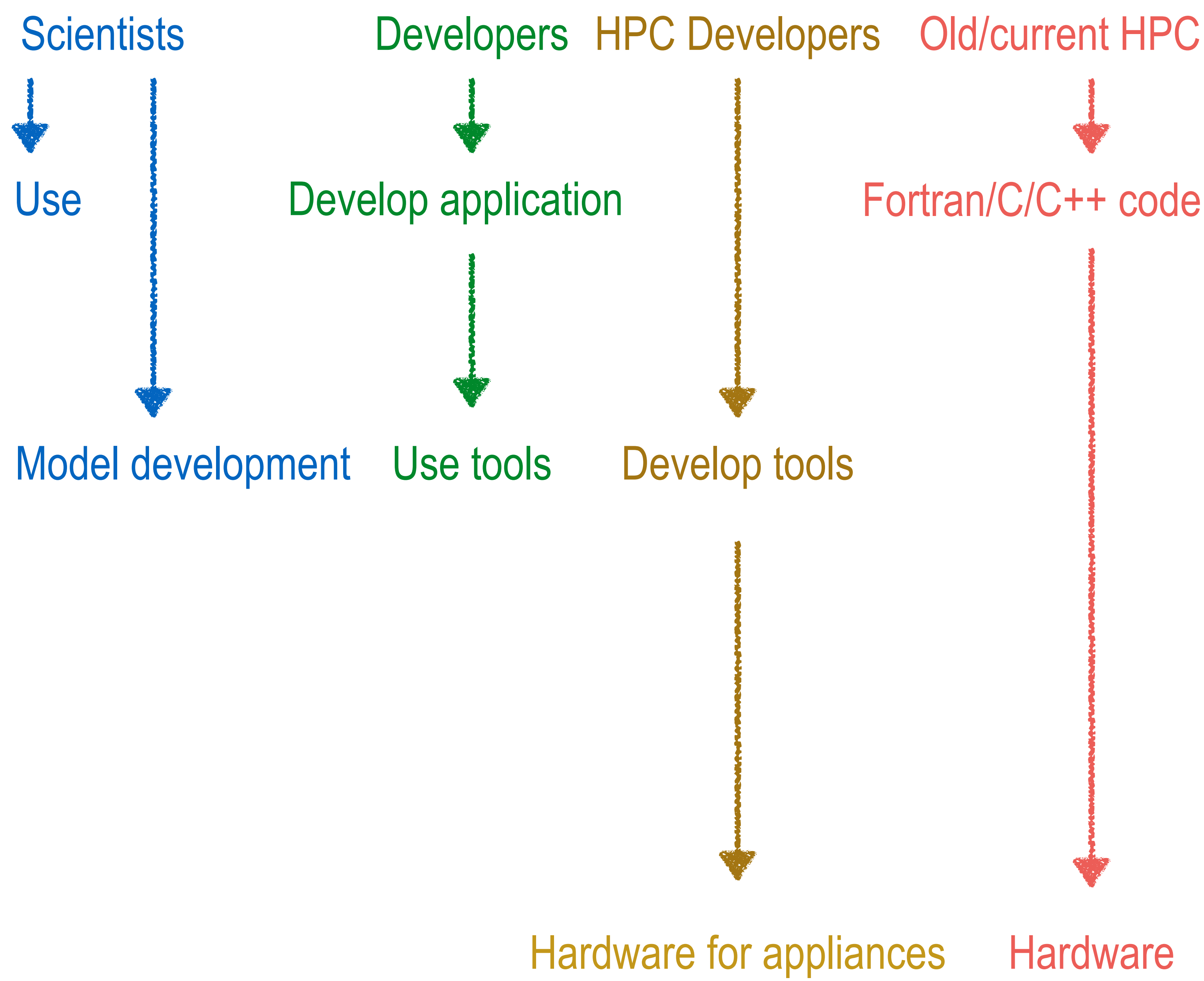
Scientific computing & data (HPC) as a service

Software as a Service (SaaS)
Modelling, searches, simulations …
e.g. weather forecast, materials design

Platform as a Service (PaaS)
Databases, algorithmic motifs
e.g. map/reduce, PDE solvers

Infrastructure as a Service (IaaS)
Data & compute services
e.g. through web services

Bare infrastructure
Compute / storage / networks /
identity mgt / security
Data centres (incl. power/cooling)



Scientific computing & data as a service

Software as a Service (SaaS)
Modelling, searches, simulations …
e.g. weather forecast, materials design

e.g. **NCCR MARVEL**, **CHIPP**, **HBP**, …

Collaborate with and support user communities
> in development of simulation / data analysis software
> support simulation / data services

Platform as a Service (PaaS)
Databases, algorithmic motifs
e.g. map/reduce, PDE solvers

e.g. **Cray**, **Nvidia**, **Intel**, …

e.g. **OLCF**, **TokyoTech**, …

Collaborate with vendors, other centres, developer communities
> develop HPC platform services
> in-situ and interactive data analysis tools

Infrastructure as a Service (IaaS)
Data & compute services
e.g. through web services

Bare infrastructure
Compute / storage / networks / identity mgt / security
Data centres (incl. power/cooling)

e.g. **JSC**, **CINECA**, **BSC**, …

CSCS' main business
> federate infrastructure with other centres
> collaborate with vendors on OpenStack, Docker, etc.
> scalable / elastic compute and storage
> networks and identity management

Join us @ the PASC16 Conference

PASC16 provides an opportunity for scientists and practitioners to discuss key issues in the use of High Performance Computing (HPC) in branches of science that require computer modelling and simulations. The scientific program will offer invited lectures, minisymposia, contributed talks and poster presentations. The active participation of graduate students and postdocs is strongly encouraged.

PASC16 Platform for Advanced Scientific Computing Conference

Lausanne Switzerland | **08–10 June 2016**

Conference information, registration and submission
www.pasc16.org

Queries may be addressed to
pasc16@pasc-ch.org

Venue
EPF Lausanne
Swiss Tech Convention Center
Lausanne
Switzerland

Contributions

Researchers from the academic and from the corporate world are invited to participate and present their research area in the form of minisymposia, contributed talks and/or poster presentations. PASC16 welcomes submissions in the following scientific fields:

- CLIMATE & WEATHER
- SOLID EARTH
- LIFE SCIENCE
- CHEMISTRY & MATERIALS
- PHYSICS
- COMPUTER SCIENCE & MATHEMATICS
- ENGINEERING
- EMERGING DOMAINS

Abstracts should describe original, interesting, and solid scientific content that is relevant to computational sciences and HPC. Cross-disciplinary approaches are highly encouraged.

Organization Committee

Maria Grazia Giuffreda, ETH Zurich / CSCS
Jan Hesthaven, EPFL
Torsten Hoefler, ETH Zurich
David Keyes, KAUST
Nicola Marzari, EPFL
Olaf Schenk, USI
Thomas Schulthess, ETH Zurich / CSCS

Thank you!