ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

Application readiness at CSCS

Thomas C. Schulthess

Swiss Platform for High-Performance and High-Productivity Computing HP2C

Overarching goal

Prepare computational sciences to make effective use of next generation supercomputers

Specific goal

Emerge with several high-impact scientific applications that scale and run efficiently on leadership computing platforms in 2012/13 timeframe

Build on, and multiply the early science applications experience on Jaguar at ORNL in 2008

DCA++: simulate models of high-temperature superconductivity
first sustained petaflop/s in production runs (Gordon Bell Prize 2008)

WL-LSMS: simulate termodynamics properties in magnetic nanoparticles sustained petaflop/s in production runs (Gordon Bell Prize 2009)

Background HP2C

- Funded by the Swiss University Conference as a structuring project
  - strong networking component is a plus/requirement
- Approach:
  - fund HPC developers that are embedded in application development teams at the Universities
  - maintain future systems competence at CSCS as well as staff that can engage with domain science teams – avoid taking responsibility for codes
  - create interdisciplinary development projects with domain scientists, applied mathematicians, and computer scientists
- Open call for project proposals in summer 2009, peer review, project selection, and project kickoff in March 2010
- Projects will run through June 2013
  - a new platform that will succeed HP2C has been approved and will run through 2016

Projects of the HP2C platform (see www.hp2c.ch)

- Gyrokinetic Simulations of Turbulence in **Fusion Plasmas** (ORB5) – Laurent Villard, EPF Lausanne
- **Ab initio Molecular Dynamics** (CP2K) – Jürg Hutter, U. of Zurich
- Computational **Cosmology** on the Petascale – Geoge Lake, U. of Zurich
- Selectome, looking for **Darwinian evolution** in the tree of life – Marc Robinson-Rechavi, Univ. of Lausanne
- **Cardiovascular Systems Simulations** (LifeV) – Alfio Quarteroni, EPF Lausanne
- Modern Algorithms for **Quantum Interacting Systems** (MAQUIS) – Thierry Giamarchi, Univ. of Geneva
- Large-Scale Parallel Nonlinear Optimization for High Resolution **3D-Seismic Imaging** (Petaquacke) – Olaf Schenk, Univ. of Basel
- 3D Models of **Stellar Explosions** – Matthias Liebendörfer, Univ. of Basel
- Large Scale **Electronic Structure Calculations** (BigDFT) – Stefan Gödecker, Univ. of Basel
- Regional **Climate & Weather** Model (COSMO) – Isabelle Bey, ETH Zurich/C2SM
- Lattice-Boltzmann **Modeling of the Ear** – Bastien Chopard, U. of Geneva
- Modeling **humans under climate stress** – Christoph Zollikhofer, U. of Zurich

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

velocities

$$\frac{\partial u}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\frac{\partial E_h}{\partial\lambda} - vV_a\right\} - \dot{\zeta}\frac{\partial u}{\partial\zeta} - \frac{1}{\rho a\cos\varphi}\left(\frac{\partial p'}{\partial\lambda} - \frac{1}{\sqrt{\gamma}}\frac{\partial p_0}{\partial\lambda}\frac{\partial p'}{\partial\zeta}\right) + M_u$$

$$\frac{\partial v}{\partial t} = -\left\{\frac{1}{a}\frac{\partial E_h}{\partial\varphi} + uV_a\right\} - \dot{\zeta}\frac{\partial v}{\partial\zeta} - \frac{1}{\rho a}\left(\frac{\partial p'}{\partial\varphi} - \frac{1}{\sqrt{\gamma}}\frac{\partial p_0}{\partial\varphi}\frac{\partial p'}{\partial\zeta}\right) + M_v$$

$$\frac{\partial w}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial w}{\partial\lambda} + v\cos\varphi\frac{\partial w}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial w}{\partial\zeta} + \frac{g}{\sqrt{\gamma}}\frac{\rho_0}{\rho}\frac{\partial p'}{\partial\zeta} + M_w + g\frac{\rho_0}{\rho}\left\{\frac{(T-T_0)}{T} - \frac{T_0 p'}{T p_0} + \left(\frac{R_v}{R_d} - 1\right)q^v - q^l - q^f\right\}$$

pressure

$$\frac{\partial p'}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial p'}{\partial\lambda} + v\cos\varphi\frac{\partial p'}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial p'}{\partial\zeta} + g\rho_0 w - \frac{c_{pd}}{c_{vd}}pD$$

temperature

$$\frac{\partial T}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial T}{\partial\lambda} + v\cos\varphi\frac{\partial T}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial T}{\partial\zeta} - \frac{1}{\rho c_{vd}}pD + Q_T$$

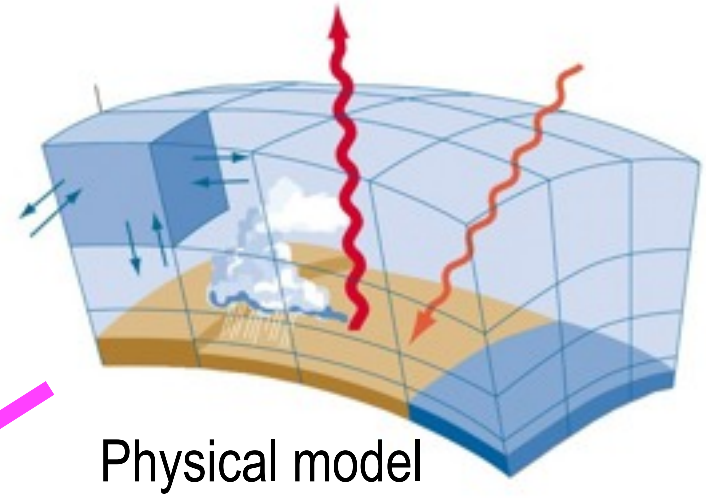
water

$$\frac{\partial q^v}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial q^v}{\partial\lambda} + v\cos\varphi\frac{\partial q^v}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial q^v}{\partial\zeta} - (S^l + S^f) + M_{q^v}$$

$$\frac{\partial q^{l,f}}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial q^{l,f}}{\partial\lambda} + v\cos\varphi\frac{\partial q^{l,f}}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial q^{l,f}}{\partial\zeta} - \frac{g}{\sqrt{\gamma}}\frac{\rho_0}{\rho}\frac{\partial P_{l,f}}{\partial\zeta} + S^{l,f} + M_{q^{l,f}}$$

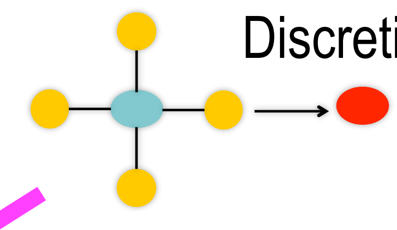
turbulence

$$\frac{\partial e_t}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial e_t}{\partial\lambda} + v\cos\varphi\frac{\partial e_t}{\partial\varphi}\right)\right\} - \dot{\zeta}\frac{\partial e_t}{\partial\zeta} + K_m^v\frac{g\rho_0}{\sqrt{\gamma}}\left\{\left(\frac{\partial u}{\partial\zeta}\right)^2 + \left(\frac{\partial v}{\partial\zeta}\right)^2\right\} + \frac{g}{\rho\theta_v}F^{\theta_v} - \frac{\sqrt{2}e_t^{3/2}}{\alpha_M l} + M_{e_t}$$



Physical model

Mathematical description



Discretization / algorithm

```
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);
```

Code / implementation

Code compilation

A given supercomputer

“Port” serial code to supercomputers
- **> vectorize**
- **> parallelize**
- **> petascaling**
- **> exascaling**
- **> ...**

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

velocities:

$$\frac{\partial u}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\frac{\partial E_h}{\partial \lambda} - vV_a\right\} - \dot\zeta\frac{\partial u}{\partial \zeta} - \frac{1}{\rho a\cos\varphi}\left(\frac{\partial p'}{\partial \lambda} - \frac{1}{\sqrt{\gamma}}\frac{\partial p_0}{\partial \lambda}\frac{\partial p'}{\partial \zeta}\right) + M_u$$

$$\frac{\partial v}{\partial t} = -\left\{\frac{1}{a}\frac{\partial E_h}{\partial \varphi} + uV_a\right\} - \dot\zeta\frac{\partial v}{\partial \zeta} - \frac{1}{\rho a}\left(\frac{\partial p'}{\partial \varphi} - \frac{1}{\sqrt{\gamma}}\frac{\partial p_0}{\partial \varphi}\frac{\partial p'}{\partial \zeta}\right) + M_v$$

$$\frac{\partial w}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial w}{\partial \lambda} + v\cos\varphi\frac{\partial w}{\partial \varphi}\right)\right\} - \dot\zeta\frac{\partial w}{\partial \zeta} + \frac{g}{\sqrt{\gamma}}\frac{\rho_0}{\rho}\frac{\partial p'}{\partial \zeta} + M_w + g\frac{\rho_0}{\rho}\left\{\frac{(T-T_0)}{T} - \frac{T_0 p'}{T p_0} + \left(\frac{R_v}{R_d} - 1\right)q^v - q^l - q^f\right\}$$

pressure:

$$\frac{\partial p'}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial p'}{\partial \lambda} + v\cos\varphi\frac{\partial p'}{\partial \varphi}\right)\right\} - \dot\zeta\frac{\partial p'}{\partial \zeta} + g\rho_0 w - \frac{c_{pd}}{c_{vd}}pD$$

temperature:

$$\frac{\partial T}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial T}{\partial \lambda} + v\cos\varphi\frac{\partial T}{\partial \varphi}\right)\right\} - \dot\zeta\frac{\partial T}{\partial \zeta} - \frac{1}{\rho c_{vd}}pD + Q_T$$

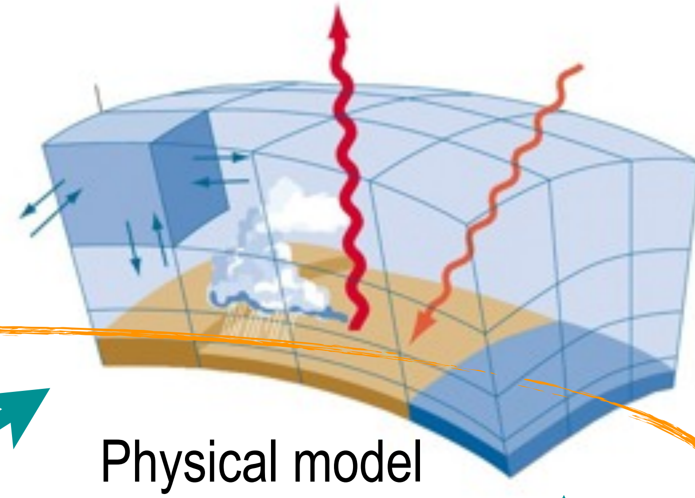
water:

$$\frac{\partial q^v}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial q^v}{\partial \lambda} + v\cos\varphi\frac{\partial q^v}{\partial \varphi}\right)\right\} - \dot\zeta\frac{\partial q^v}{\partial \zeta} - (S^l + S^f) + M_{q^v}$$

$$\frac{\partial q^{l,f}}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial q^{l,f}}{\partial \lambda} + v\cos\varphi\frac{\partial q^{l,f}}{\partial \varphi}\right)\right\} - \dot\zeta\frac{\partial q^{l,f}}{\partial \zeta} - \frac{g}{\sqrt{\gamma}}\frac{\rho_0}{\rho}\frac{\partial P_{l,f}}{\partial \zeta} + S^{l,f} + M_{q^{l,f}}$$

turbulence:

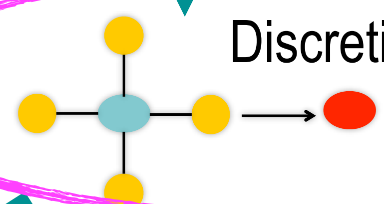
$$\frac{\partial e_t}{\partial t} = -\left\{\frac{1}{a\cos\varphi}\left(u\frac{\partial e_t}{\partial \lambda} + v\cos\varphi\frac{\partial e_t}{\partial \varphi}\right)\right\} - \dot\zeta\frac{\partial e_t}{\partial \zeta} + K_m^v\frac{g\rho_0}{\sqrt{\gamma}}\left\{\left(\frac{\partial u}{\partial \zeta}\right)^2 + \left(\frac{\partial v}{\partial \zeta}\right)^2\right\} + \frac{g}{\rho\theta_v}F^{\theta_v} - \frac{\sqrt{2}e_t^{3/2}}{\alpha_M l} + M_{e_t}$$



Physical model

Mathematical description

Driven by domain sicence



Discretization / algorithm

Needs to be properly embedded

```
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);
```

Code / implementation

Libraries
DSL / DSEL

Driven by vendors

Code compilation

Architectural options / design

Optimal algorithm
Auto tuning

WHILE IN THIS PRESENTATION OUR FOCUS IS ON RESULTS AND DEVELOPMENTS ON CRAY'S XE6/XK6, THE HP2C PROJECTS AND APPLICATION READINESS ACTIVITIES AT CSCS HAVE BEEN COVERING A BROAD RANGE OF PLATFORMS, INCLUDING IBM'S BG/Q SYSTEM AND SYSTEMS BUILT WITH INTEL'S XEON AND XEON-PHI PROCESSORS

CPU-GPU-Hybrid study

- After acceptance of Cray XK6 in fall 2012, start a study of application performance on Hybrid platforms (kickoff Dec. 1, 2012)
- Selected 6 HP2C teams and three additional teams
  - CP2K – chem./mat. sci.; U. of ZH + ETH-Z
  - Astro – astro./cosmology; U. of ZH
  - COSMO – climate/weather; ETH-Z + Meteo CH
  - MAQUIS – chem./mat. sci.; U of GE + EPFL + ETH-Z
  - Petaquake – Geophysics; U of BS + ETH-Z
  - BigDFT – chem./mat. sci.; Uof BS
  - MRAG – engineering; ETH-Z
  - GROMACS – life sci.; KTH (Sweden)
  - MAGMA eigenvalue solvers – all DFT codes; UTK/ETH-Z
- Timeline:
  - Report on porting status in January 2012
  - First report on performance of XK6 vs. XE6 at ACSS workshop in March 2012
  - Continue evaluation (including Xeon-Phi*) though Q1/2013

*NDA, so no results will be shown

9 projects and 15 applications: status Jan. 27, 2012

Project	Code	User community	Rosa (XE6)	Tödi (XK6)	HMC PE&Model
CP2K	CP2K	very large	runs	runs	CUDA
Cosmology	RAMSES	large	runs	runs	CUDA, OpenACC
Cosmology	PKDGRAV	medium	runs	runs	CUDA
Cosmology	GENGA	medium	runs	runs	CUDA

Remarks:

The XK6 was an entirely new architecture, new software stack, etc.

It took only 8 weeks, which included the Christmas holidays, to get apps. up an running

We were surprised by the level of adoption of CUDA
(HP2C funds application developers to become experts in HPC and they do what they do)

CFD	MRAG	small	runs	runs	CUDA, OpenCL
GROMACS	GROMACS	very large	runs	runs	CUDA
Comp. Materials	DCA++	small	runs	runs	CUDA
Comp. Materials	MAGMA solvers	very large	n.a.	runs	CUDA/CuBLAS
Comp. Materials	ELK/EXCITING	very large	runs	runs	CUDA

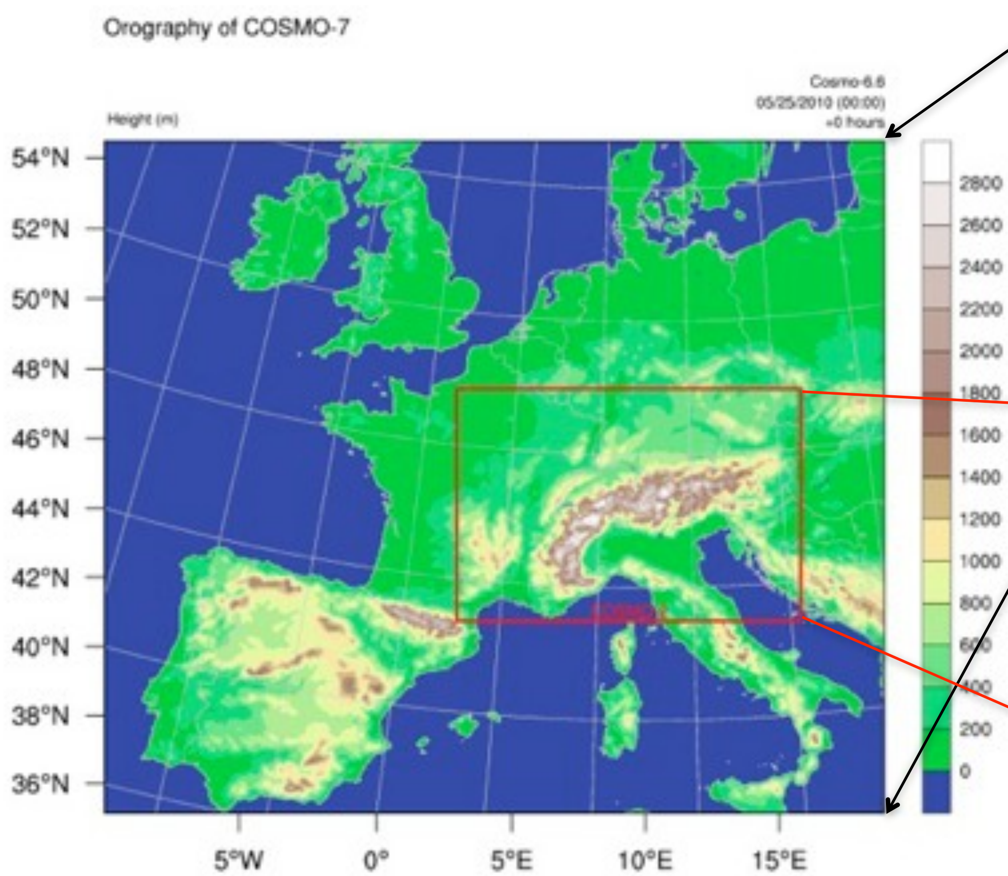
9 projects and 15 applications: status Jan. 27, 2012

Target apps on which we report performance in this presentation

Project	Code	User community	Rosa (XE6)	Tödi (XK6)	HMC PE&Model
CP2K	CP2K	very large	runs	runs	CUDA
Cosmology	RAMSES	large	runs	runs	CUDA, OpenACC
Cosmology	PKDGRAV	medium	runs	runs	CUDA
Cosmology	GENGA	medium	runs	runs	CUDA
C2SM	COSMO	large	runs	runs	CUDA, OpenACC
MAQUIS	ED	small	runs	no attempt	n.a.
MAQUIS	DMRG	small	no attempt	no attempt	n.a.
MAQUIS	SE (VLI)	small	runs	runs	CUDA
Petaquake	SPECFEM 3D	large	runs	runs	CUDA, OpenACC
BigDFT	BigDFT	large	runs	runs	CUDA, OpenCL
CFD	MRAG	small	runs	runs	CUDA, OpenCL
GROMACS	GROMACS	very large	runs	runs	CUDA
Comp. Materials	DCA++	small	runs	runs	CUDA
Comp. Materials	MAGMA solvers	very large	n.a.	runs	CUDA/CuBLAS
Comp. Materials	ELK/EXCITING	very large	runs	runs	CUDA

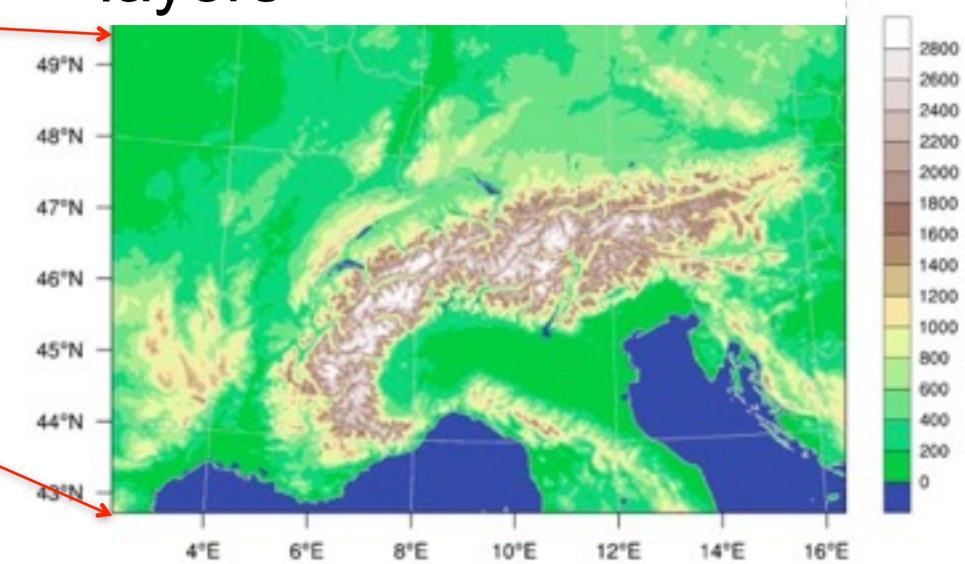
COSMO in production for Swiss weather prediction

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

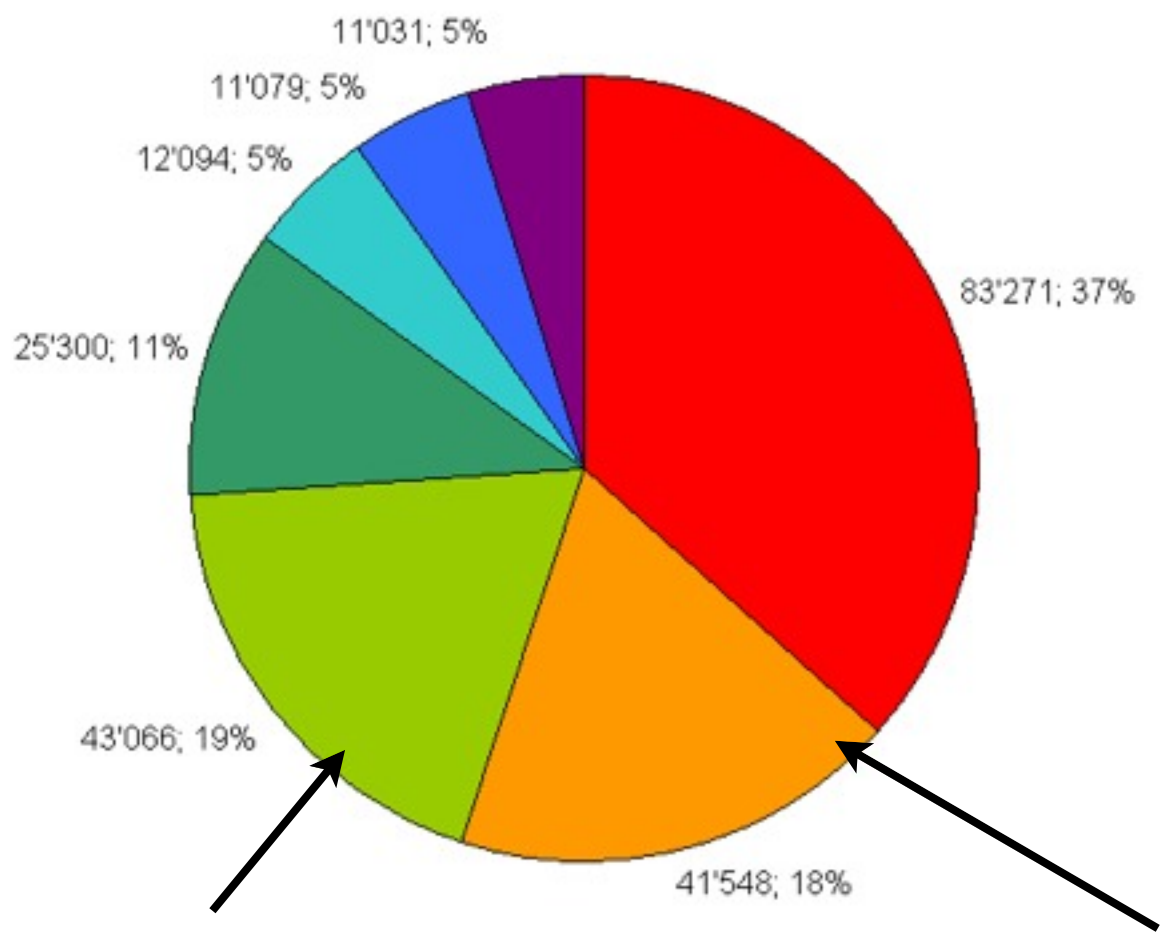


- Some of the products generated from these simulations
  - Daily weather forecast
  - Forecasting for air traffic control (Sky Gide)
  - Safety management in event of nuclear incidents

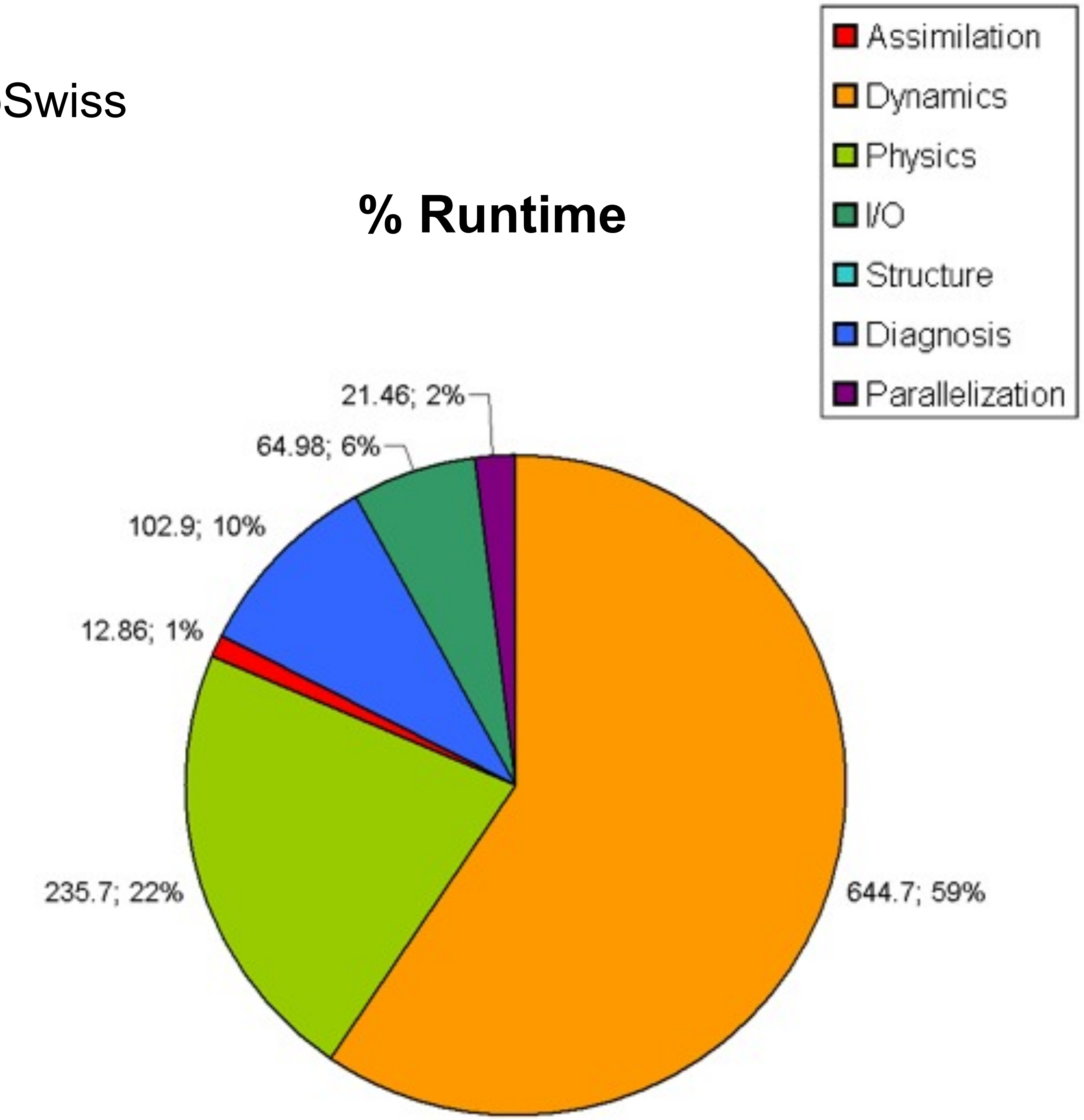
Performance profile of (original) COSMO-CCLM

Runtime based 2 km production model of MeteoSwiss

% Code Lines (F90)



% Runtime



Original code (with OpenACC for GPU)

Rewrite in C++ (with CUDA backend for GPU)

Analyzing the two examples – how are they different?

Physics

```
do j = 1, niter
  do i = 1, nwork
    c(i) = a(i)**b(i) + sin(b(i)) * log(a(i))
  end do
end do
```

3 memory accesses
136 FLOPs
➔ compute bound

Dynamics

```
do j = 1, niter
  do i = 1, nwork
    c(i) = a(i) + b(i) * ( a(i+1) - 2.0d0*a(i) + a(i-1) )
  end do
end do
```

3 memory accesses
5 FLOPs
➔ memory bound

- Arithmetic throughput is a **per core resource** that scale with number of cores and frequency
- Memory bandwidth is a **shared resource between cores** on a socket

Running the simple examples on the Cray XK6

Compute bound (physics) problem

Machine	Interlagos	Fermi (2090)	GPU+transfer
Time	1.31 s	0.17 s	1.9 s
Speedup	**1.0 (REF)**	**7.6**	**0.7**

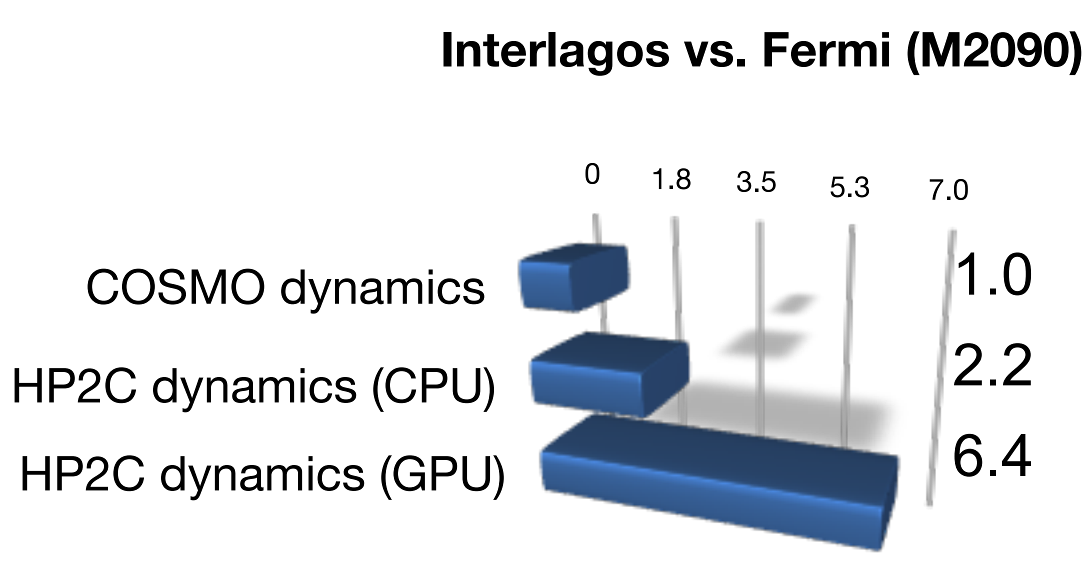
Memory bound (dynamics) problem

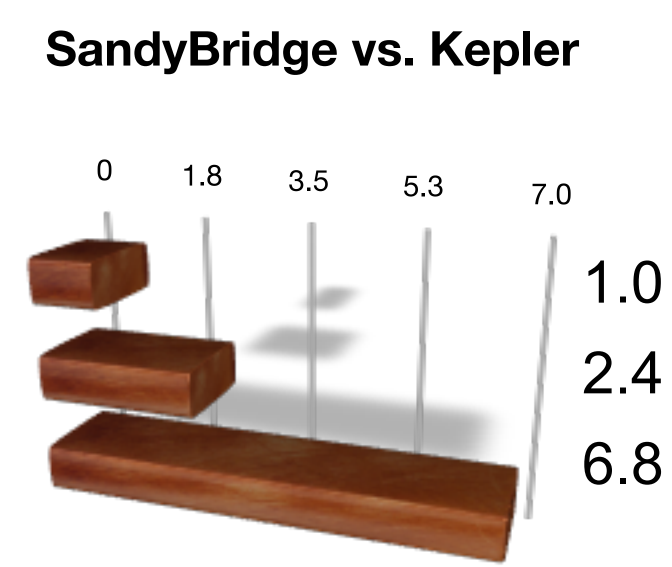
Machine	Interlagos	Fermi (2090)	GPU+transfer
Time	0.16 s	0.038 s	1.7 s
Speedup	**1.0 (REF)**	**4.2**	**0.1**

The simple lesson: leave data on the GPU!

Application performance of COSMO dynamical core (DyCore)

- The CPU backend is 2x-2.9x faster than standard COSMO DyCore
  - Note that we use a different storage layout in new code
  - 2.9x applied to smaller problem sizes, i.e. HPC mode (see later slide)
- The GPU backend is 2.8-4x faster than the CPU backend
- Speedup new DyCore & GPU vs. standard DyCore & CPU = 6x-7x





Solving Kohn-Sham equation is the bottleneck of 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 may not be orthogonal $$S_{\mu\nu} = \int \phi^*_\mu(\vec{r})\phi_\nu(\vec{r})d\vec{r}$$

Solve generalized eigenvalue problem $$(\mathbf{H} - \varepsilon_i \mathbf{S}) = 0$$

Solving Kohn-Sham equation is the bottleneck of most DFT based materials science codes

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

Remarks:

Typical matrix size is up to 10’000

Only a few methods require high accuracy and matrix size up to 100’000

Some methods try to avoid eigensolvers (orthogonalization will be important motif instead)

When lower accuracy is an option, order(N) methods are used instead
(this is covered by one of our HP2C projects see CP2K on hp2c.ch)

Dominant motif:

Solve generalized eigenvalue problem $(\mathbf{H} - \varepsilon_i\mathbf{S}) = 0$

We will need between 10% and 50% of the eigenvectors

Solving the generalized eigenvalue problem

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

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

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

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

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

Standard 1 stage solver

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

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

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

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

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

Solving the generalized eigenvalue problem with 2 stage solver

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

↓

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
Most time consuming step, but dominated by BLAS-3

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

↓

tri-diagonalize

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

↓

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

↓

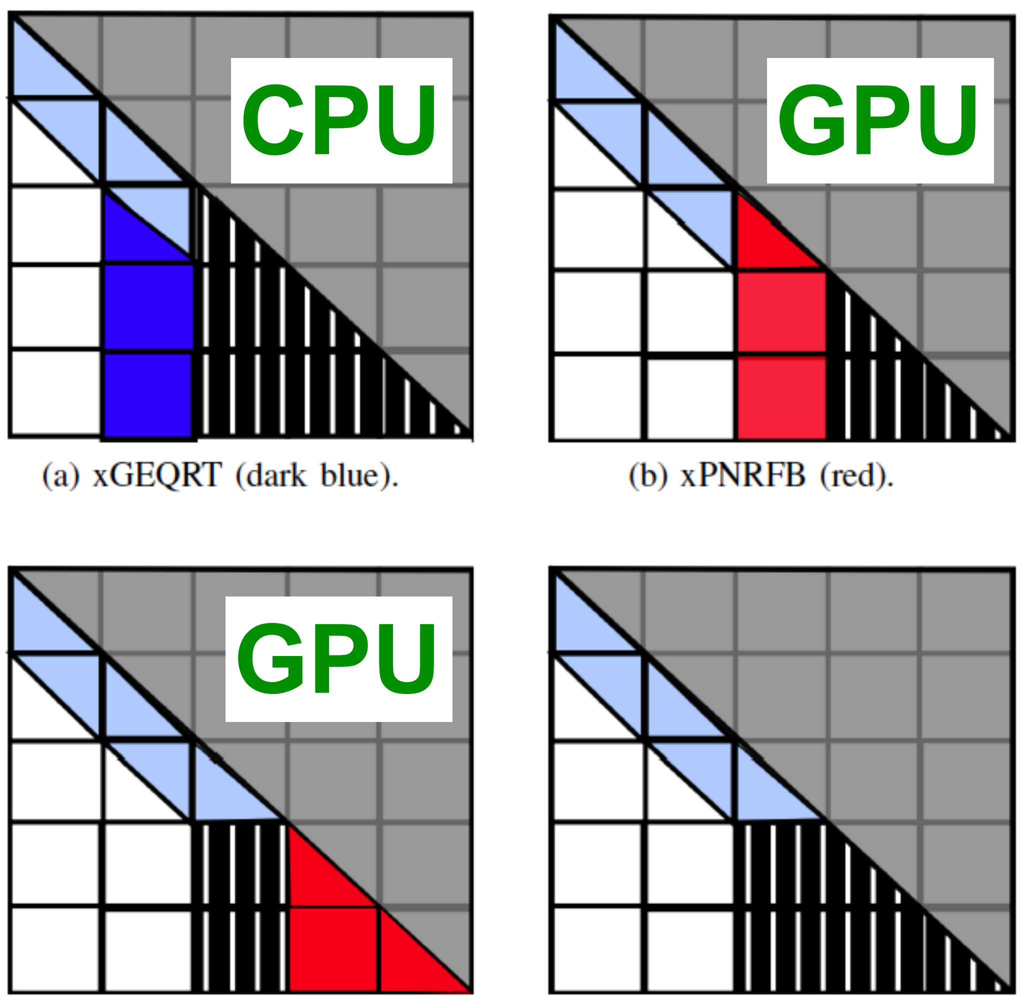
$$\mathbf{y}'' = \mathbf{Q_2 y}'$$

↓

needs two eigenvector transformations (but easy to parallelize)

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

2 stage solver maps best to hybrid CPU-GPU architecture



(a) xGEQRT (dark blue).
(b) xPNRFB (red).
(c) xHER2K (red)
(d) matrix structure.

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

$\downarrow$

reduction to banded
hybrid

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

$\downarrow$

tri-diagonalize
CPU

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

$\downarrow$

divide an conquer
CPU

$$\mathbf{T}\mathbf{y}' = \lambda \mathbf{y}'$$

$\downarrow$

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

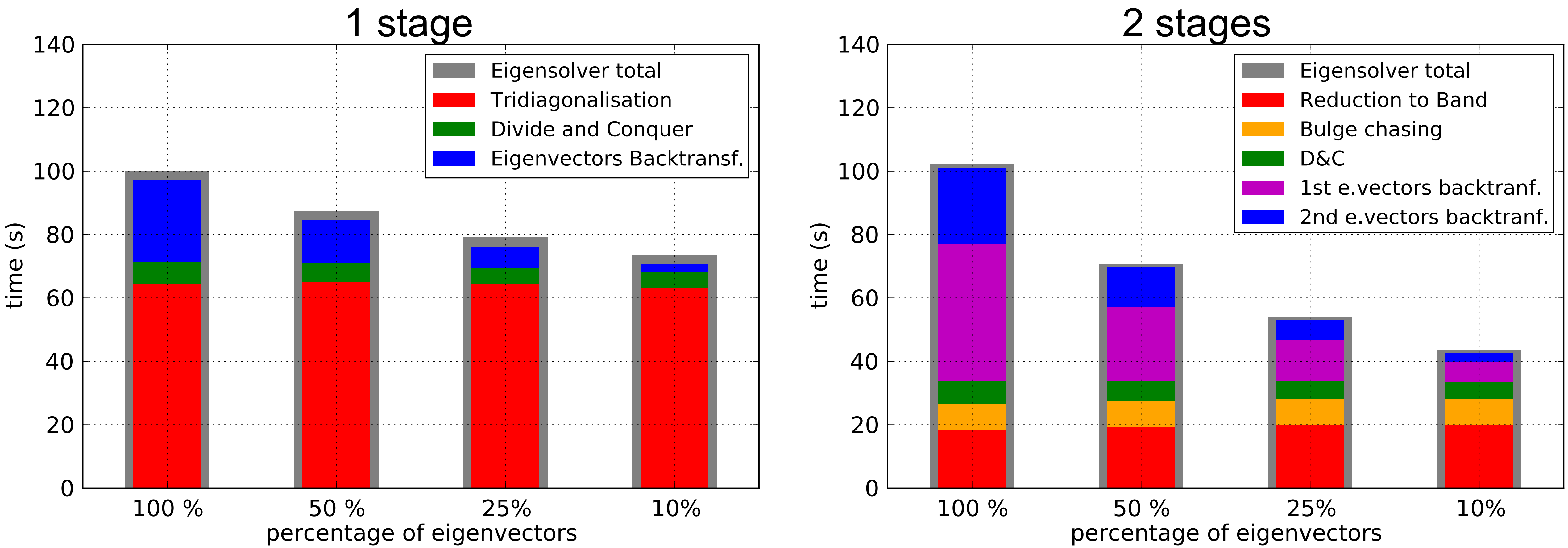
$\downarrow$

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

needs two eigenvector
transformations
(but easy to parallelize)
GPU

Profile and performance for relevant problem size

- Results of the standard eigensolvers computing partial eigenspace

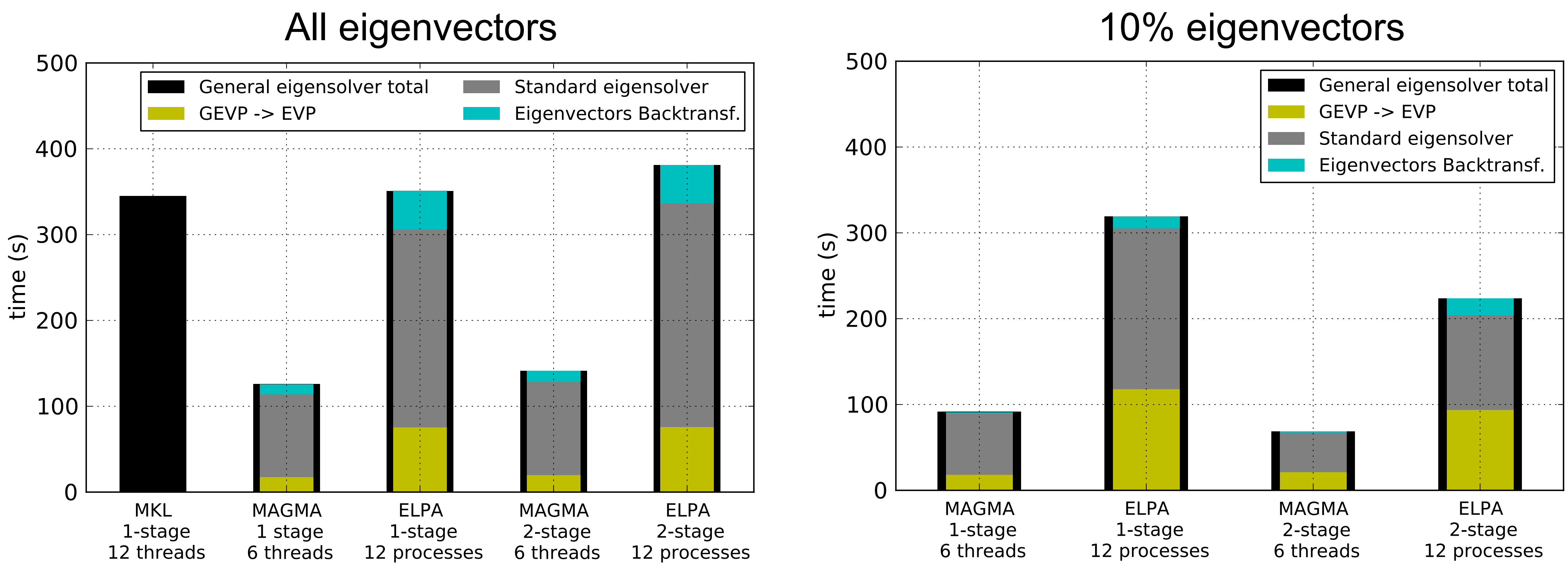


double complex, matrix size 10’000

6 threads on CPU + GPU (1x Intel X5650, 1x Nvidia M2090)

Comparison of general eigensolvers: multi-core vs. hybrid

double complex, matrix size 10000
hybrid with MAGMA (*): 6 threads + GPU (1x Intel X5650, 1x Nvidia M2090)
multi-core with MKL: 12 threads (2x Intel X5650)
multi-core with ELPA: 12 processes (2x Intel X5650)



(*) A novel hybrid CPU-GPU generalized eigensolver for electronic structure calculations based on fine grained memory aware tasks, A. Haidar, S. Tomov, J. Dongarra, R. Solcà, and TCS (under review)

(will be included in release of MAGMA late summer/fall 2012)

Hybrid performance of 5 Target Applications: status Oct. 2012

	measure	problem size	# nodes	IL-IL (Rosa)	IL-Fermi (Tödi)	Speedup
CP2K	time	864 H_2O	16	50	33	1.51
COSMO DyCore	time	128x96	1	2.15	1.19	1.81
Gromacs	ns/day	512k H_2O	1	0.659	1.025	1.56
Eigen Solver	time	8100x8100	1	288	94.3	3.05
SPECFEM3D	time	300k Eur.	16	20.78	21.13	0.98
			1	394.37	304.90	1.29

Remarks

- Snapshot for typical workloads of the codes (this is work in progress!)
  - SPECFEM3D 300k Europe is relatively small and reaches scaling limits on 16 GPUs
- Apples-to-apples comparison across several codes is hard and not always meaningful (need to consider codes individually)
- No Intel results shown here, since we don’t have SandyBridge+GPU with PCIe3 yet
  - note SandyBridge performs significantly better than AMD/Interlagos
- COSMO DyCore covers only about 60% of production run

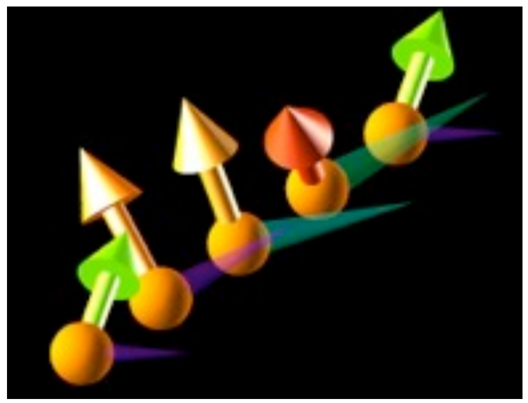
Computer performance and application performance increase ~10^3 every decade

~100 Kilowatts → ← ~5 Megawatts → ← 20-30 MW →

~1 Exaflop/s

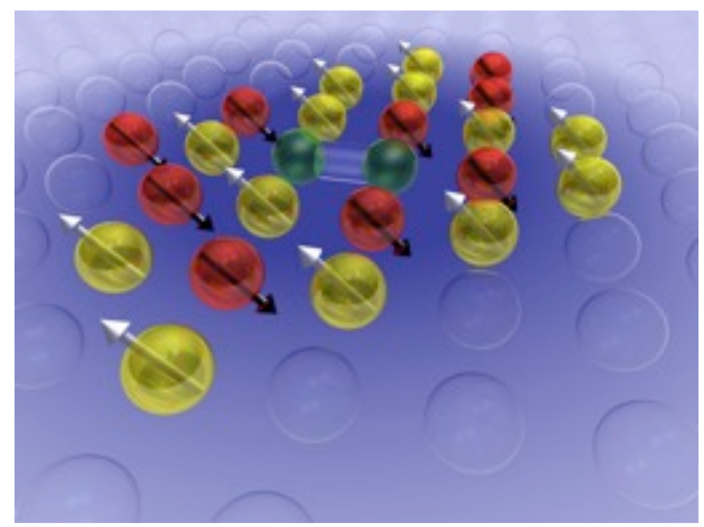
100 million or billion processing cores (!)

1 Gigaflop/s
Cray YMP
8 processors



1.02 Teraflop/s
Cray T_{3E}
1'500 processors

1.35 Petaflop/s
Cray XT5
150'000 processors



If HPC community continues to be in denial this NewPM will be CUDA

Vector	MPI MPP	MPI + OpenMP MPP + Multi Core	X + NewPM X + massive threading
1988	1998	2008	2018
First sustained GFlop/s Gordon Bell Prize 1988	First sustained TFlop/s Gordon Bell Prize 1998	First sustained PFlop/s Gordon Bell Prize 2008	Another 1,000x increase in sustained performance

Acknowledgements

- Team working on COSMO refactoring
  - **Oliver Fuhrer** (Meteo Swiss)
  - **Tobias Gysi** and **Daniel Müller** (Supercomputing Systems)
  - **Xavier Lapillone** and **Carlos Osuna** (C2SM@ETH)
  - **Will Sawyer**, **Mauro Bianco**, **Ugo Vareto**, **Ben Cumming**, and **TCS** (CSCS)
  - **Tim Schröder**, and **Peter Messmer** (NVIDIA)
  - **Ulli Schättler**, and **Michael Baldauf** (DWD)
- Team working on eigensolvers for hybrid architectures (MAGMA)
  - **Stan Tomov**, **Azam Haidar**, **Jack Dongara** (UTK), **Raffaele Solca**, and **TCS** (ETH)
- Future Systems team working on architecture evaluations at CSCS
  - **Sadaf Alam**, **Gilles Fourestey**, **Ben Cumming**, **Jeff Poznanovic**, **Ugo Vareto**, and **Mauro Bianco**
- Many researchers from the HP2C platform that develop the applications
- Funding for much of the HPC related work
  - Swiss University Conference through HP2C Platform (www.hp2c.ch)
  - PRACE 2IP WP8 and all institutions that had to provide matching funds
  - ETH Board through ETH Zurich

THANK YOU!