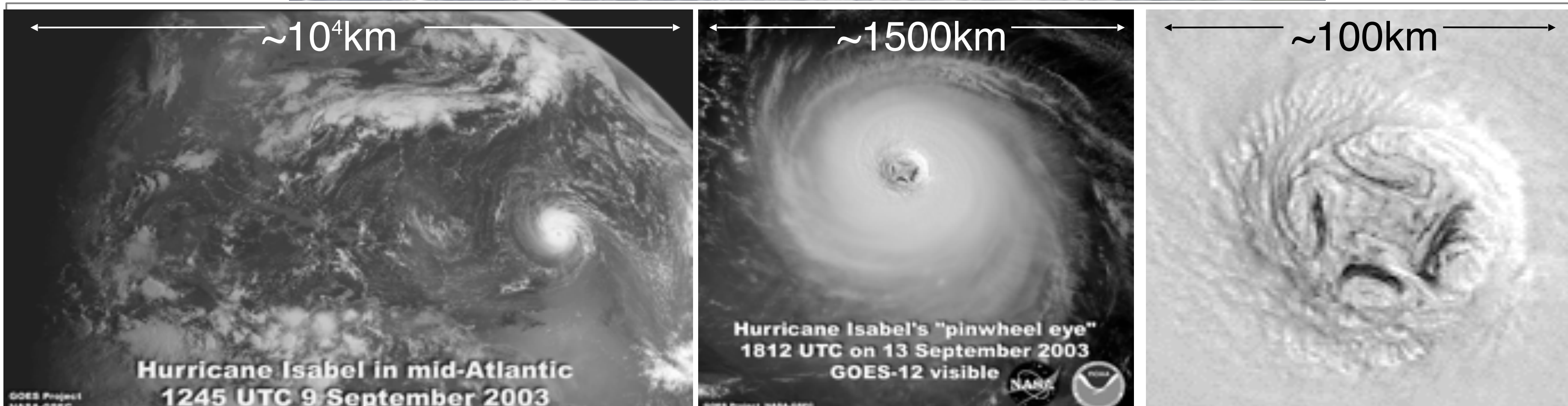
- Part 4 -
Multicore and Manycore Technology: Chances and Challenges

Vincent Heuveline

Numerical Simulation of Tropical Cyclones
“Goal oriented adaptivity for tropical cyclones”



Max-Planck-Institut
für Meteorologie

Numerical Simulation of Tropical Cyclones

Meteorology

Dynamics of tropical cyclones

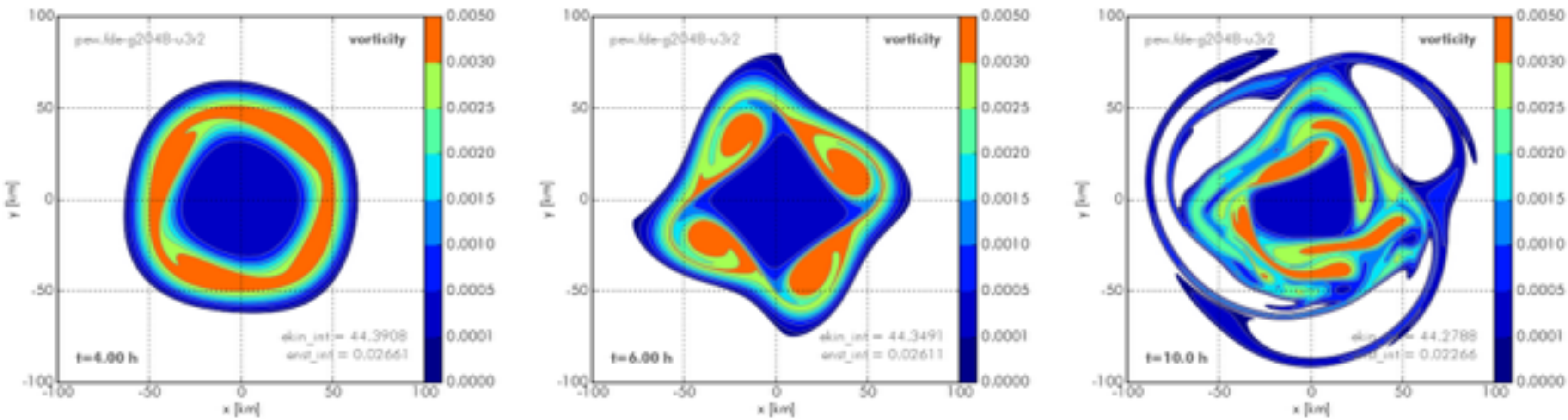
Numerical models of tropical cyclones

Mathematics

Numerical solution of PDEs, goal oriented adaptive FE methods

High performance computing

Adaptive techniques applied to TC problems

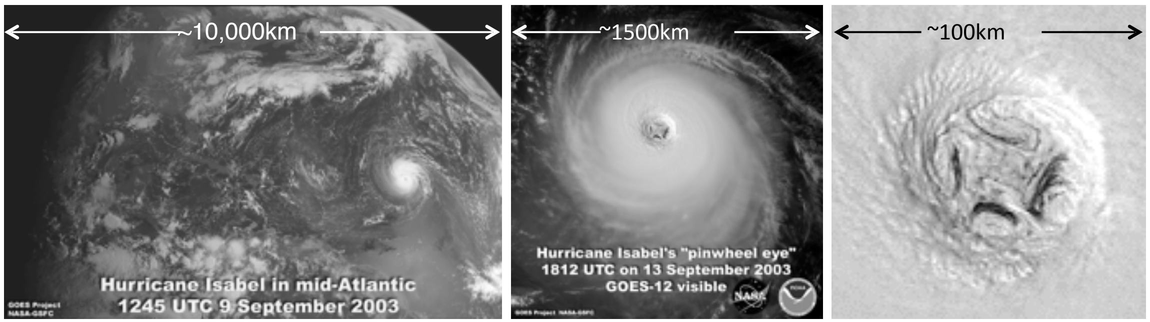


MetStröm DFG

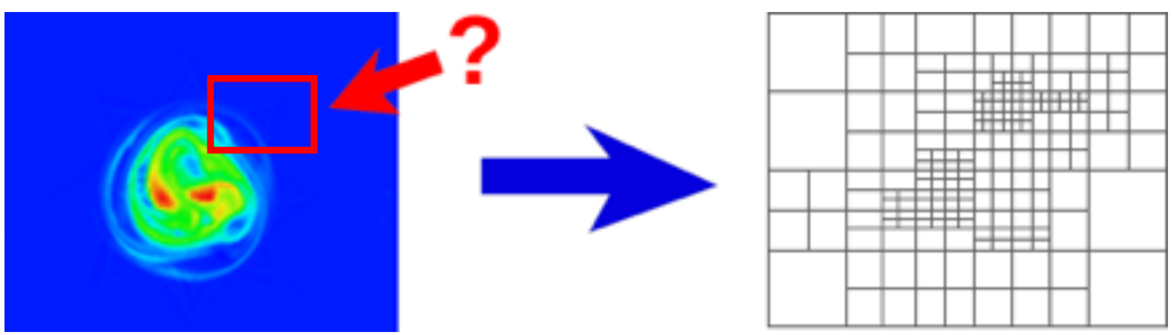
EMCL HITS

Numerical Simulation of Tropical Cyclones

– Multiple scales in space and time are relevant for the dynamics of the atmosphere



– In large domains, smallest scales cannot be resolved → adaptive methods
– Goal-oriented adaptivity: User-defined **features of interest** are in focus

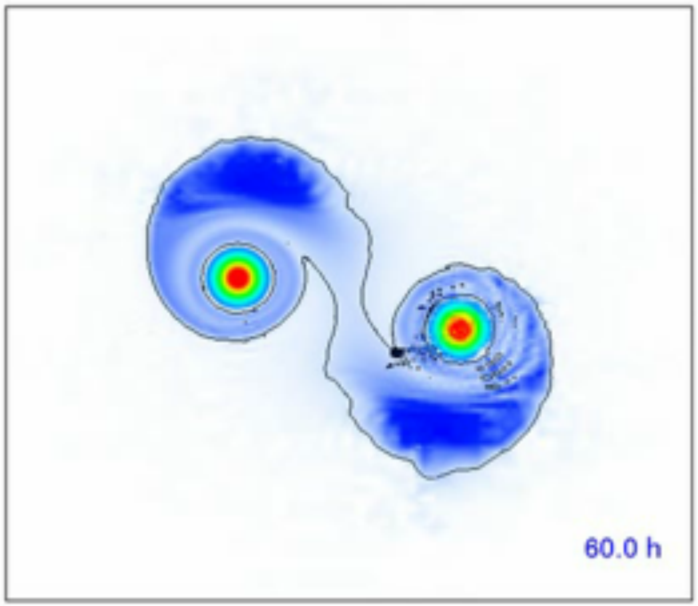


EMCL HITS

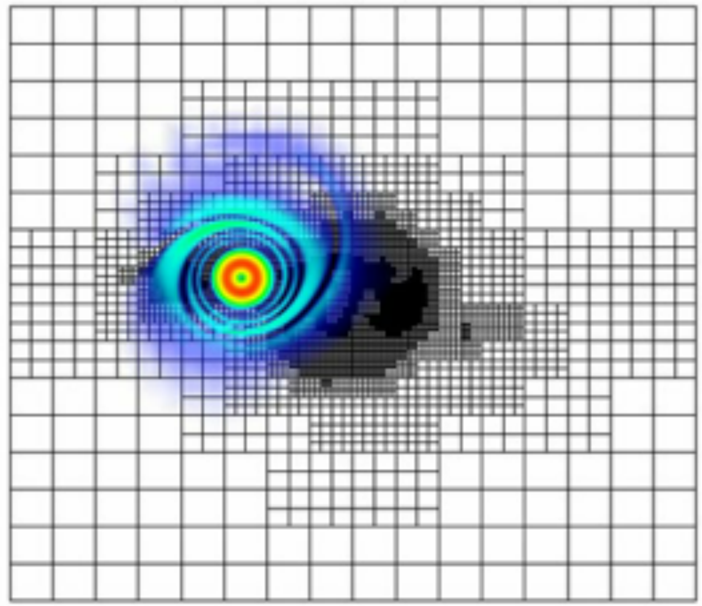
Numerical Simulation of Tropical Cyclones

– Goal-functional: $J(v) := \int_{B(x_0,r)} (\nabla \times v)(T,x)\, dx,\ T = 96\text{h}$

Primal Solution



Dual Solution

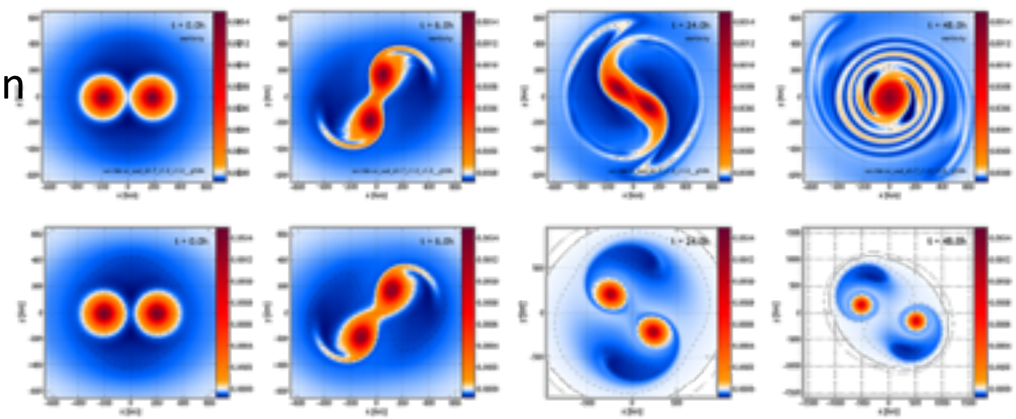


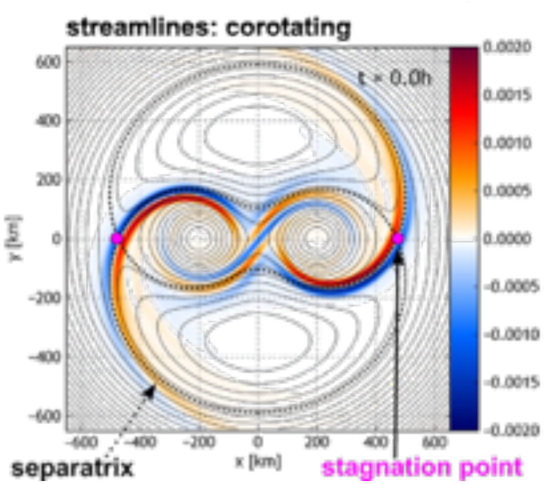
Vorticity

Velocity

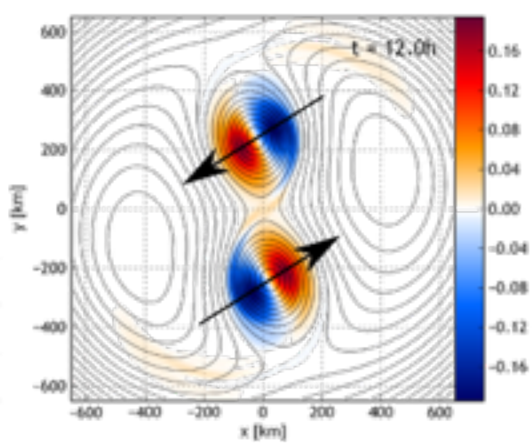
Numerical Simulation of Tropical Cyclones

- Small perturbations of the data
  → big perturbation of the solution
- Sensitivity
  - Linear approaches
  - Adjoint methods
  - Nonlinear methods





Perturbation of initial state (t=0h)



Developped perturbation (t=12h)

- Evolution of the optimal perturbation (i.e. adjoint solution)

MetStröm

L. Scheck (IMK-TRO)

EMCL HITS

Simple Performance Model

Run time : $T_R \geq \max \{ T_C , T_T \}$

- T_R … total run time of an algorithm
- T_C … compute time, $T_C \geq f / P$
- T_T … transfer time, $T_T \geq 8w / B$
- Classifies compute-bound or memory-bound algorithms

- Algorithm characteristics
  - f … number of floating point operations
  - w … number of memory transfers (words)
- Hardware characteristics
  - P … peak performance
  - B … memory bandwidth

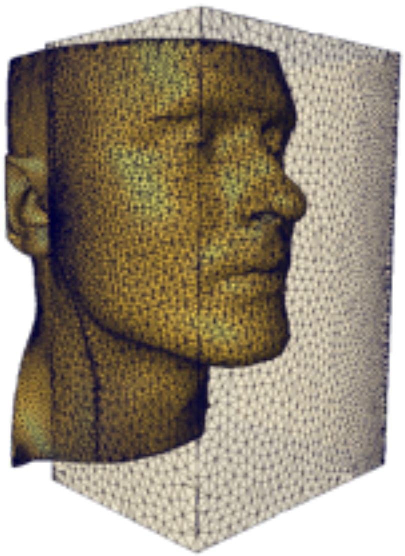
- We find: effective performance $P_{eff} \leq f / T_R \leq fB / (8w)$
  - Ratio f / w defines computational intensity

Computational intensity

Computational intensity =

 # floating point operations per transferred byte

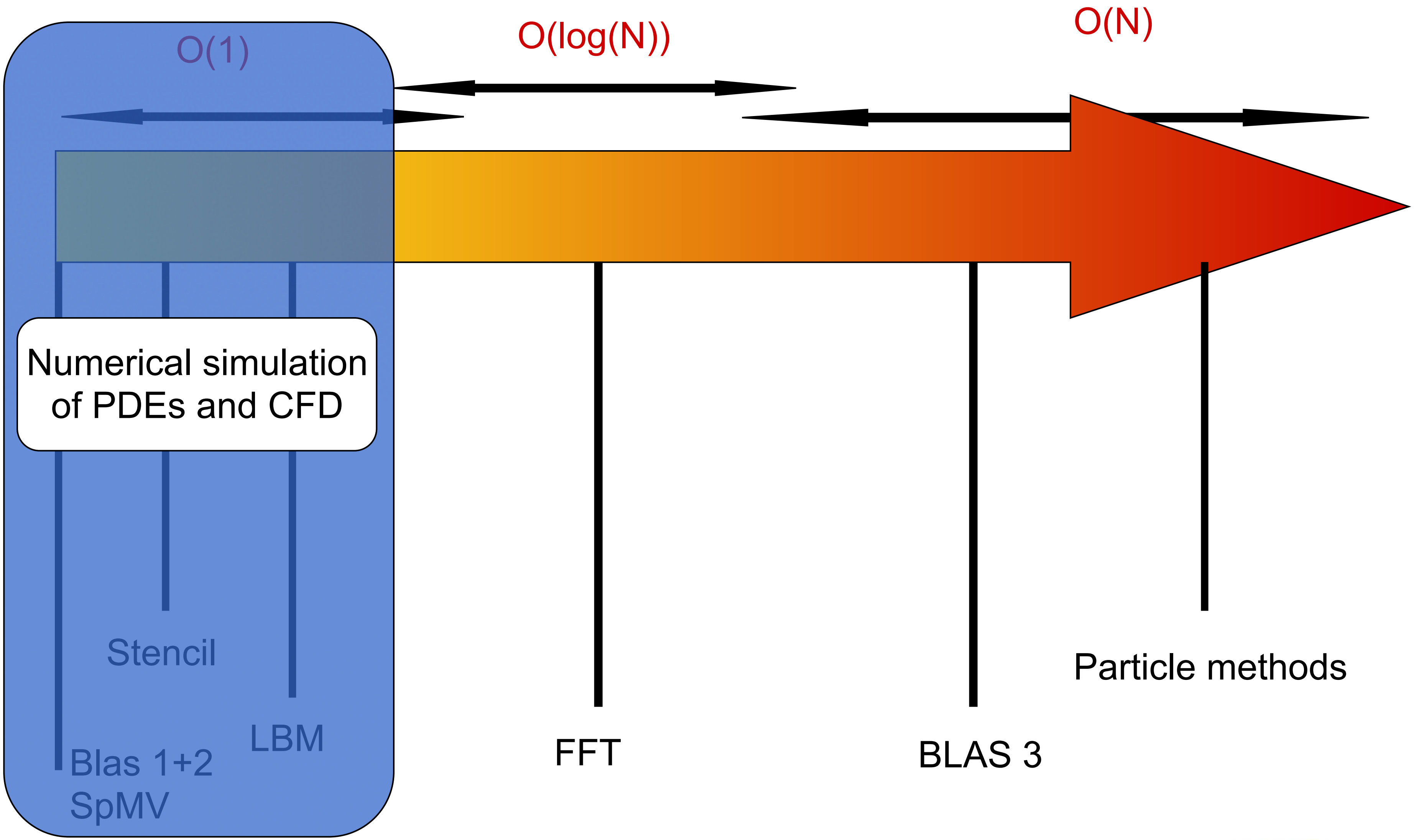
All basic operations have computational intensity of O(1)!



Consequences:

- Bandwidth limitations on many devices
- Only a small fraction of peak performance can be achieved

Computational intensity



Chorin-based Navier Stokes Solver

- Incompressible Navier-Stokes equations

$$\partial_t u - \nu \Delta u + (u \cdot \nabla) u + \nabla p = f \quad \text{in } \Omega$$

$$\nabla \cdot u = 0 \quad \text{in } \Omega$$

- Chorin-type projection method: iterative scheme, k=0, 1, …
  - Compute $\tilde{u}^{k+1}$: $$\frac{\tilde{u}^{k+1} - u^k}{\Delta t} + (u^k \cdot \nabla) u^k - \nu \Delta u^k = f^k \quad \text{in } \Omega$$
  - Compute p^{k+1} : $$\Delta p^{k+1} = \frac{1}{\Delta t} \nabla \tilde{u}^{k+1} \quad \text{in } \Omega$$
  - Compute u^{k+1} : $$\frac{u^{k+1} - \tilde{u}^{k+1}}{\Delta t} = -\nabla p^{k+1} \quad \text{in } \Omega$$

Chorin-based Navier Stokes Solver
Algorithm Characteristics

- Four basic operations (typical of solution of PDEs)
  - Sparse matrix-vector multiplication / stencil
  - (Non)-linear stencil operations
    - Nearest neighbor interaction
  - SAXPY / DAXPY vector updates
  - Dot products
- Encapsulated by iterative methods and time stepping schemes

- Characteristics
  - Huge memory requirements / fine mesh resolution
  - Low computational intensity of order O(1)
  - Bandwidth-bound algorithms
  - Non-uniform treatment of boundary conditions

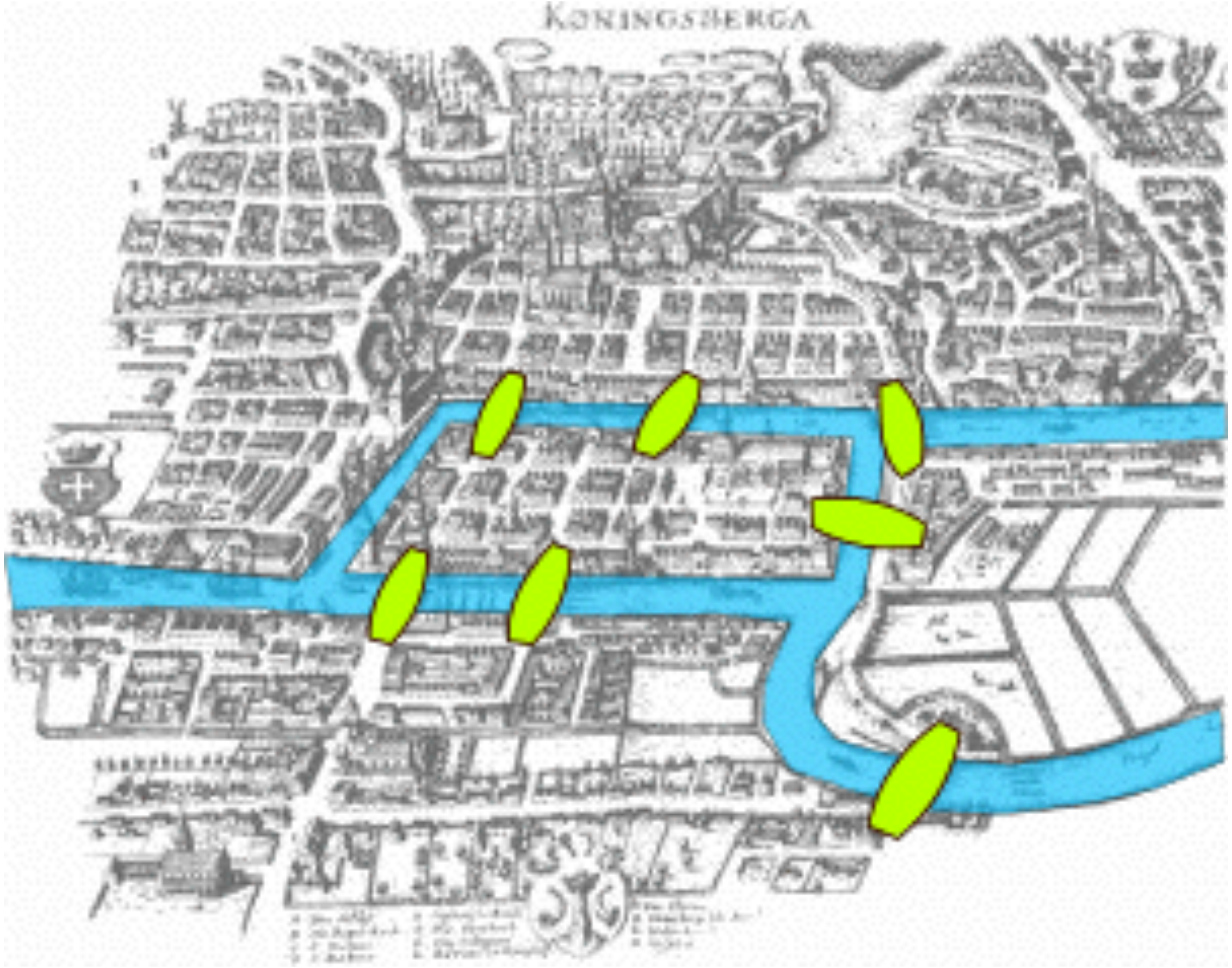
Elements of projection step

- LSE in projection step solved with conjugate gradient method (cg)
- Number of cg-steps per time step depends on $N^{1/3}$
- Each cg-step consists of

Function	Occ.	f [#flop]	w [#words]	f / w Comp. int.
Stencil operation	1	8N	2N	4.0
Vector norm	1	2N-1	N+1	2.0
Dot product	1	2N-1	2N+1	1.0
Normalization	4	2N	2N	1.0
DAXPY vector update	3	2N	3N+1	0.66

- Worst performance for DAXPY vector update
  - Easy routine with no data dependencies
  - Perfectly parallelizable on coarse and fine-grained platforms

Seven Bridges over the Gap?



Energy-efficient processor technology

ACPI – Advanced Configuration and Power Interface

- P-states (performance states)

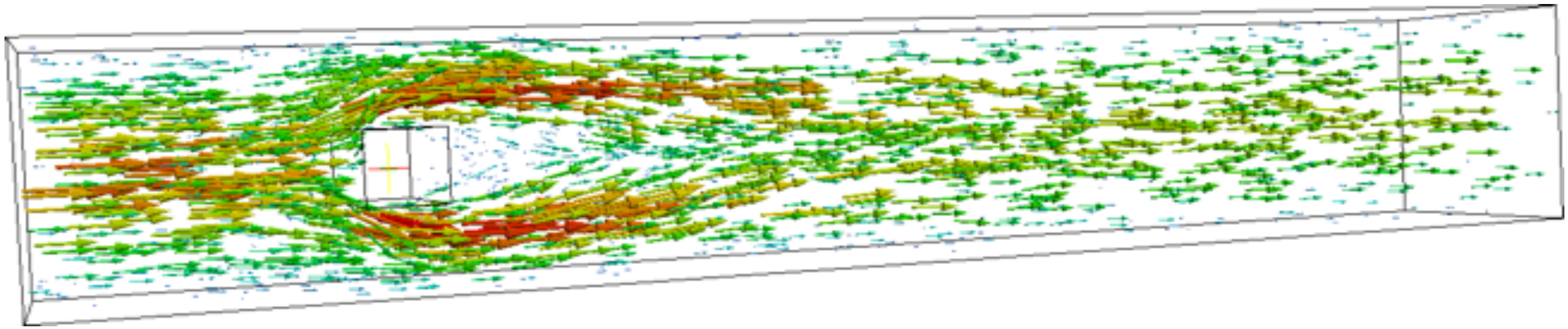
  DVFS – dynamic voltage and frequency scaling

- C-states (power states)

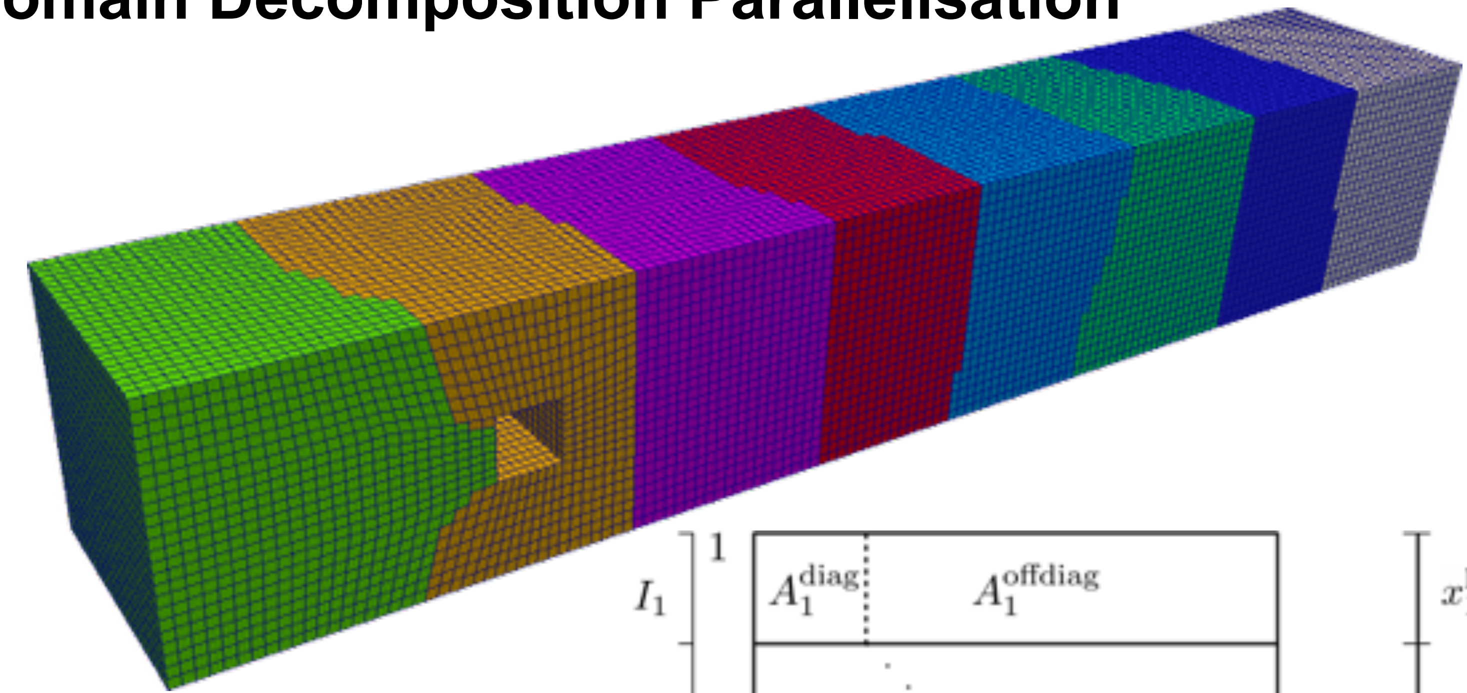
Intel E5504	voltage [V]	freq. [GHz]
P0	1.04	2.00
P1	1.01	1.87
P2	0.98	1.73
P3	0.95	1.60

AMD 6128	voltage [V]	freq. [GHz]
P0	1.23	2.00
P1	1.17	1.50
P2	1.12	1.20
P3	1.09	1.00
P4	1.06	0.80

Benchmark problem



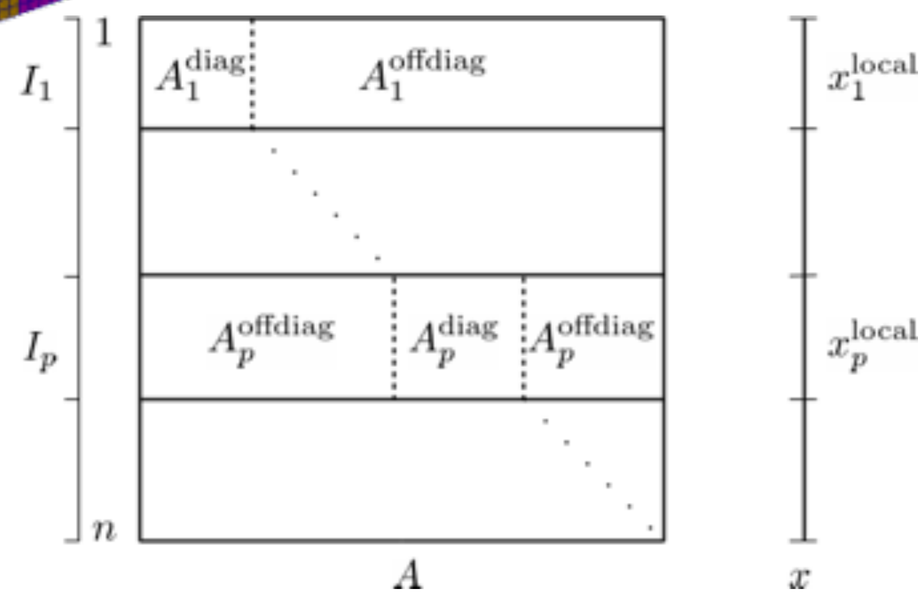
Domain Decomposition Parallelisation



$$Ax = b$$

$$F(x) = 0$$

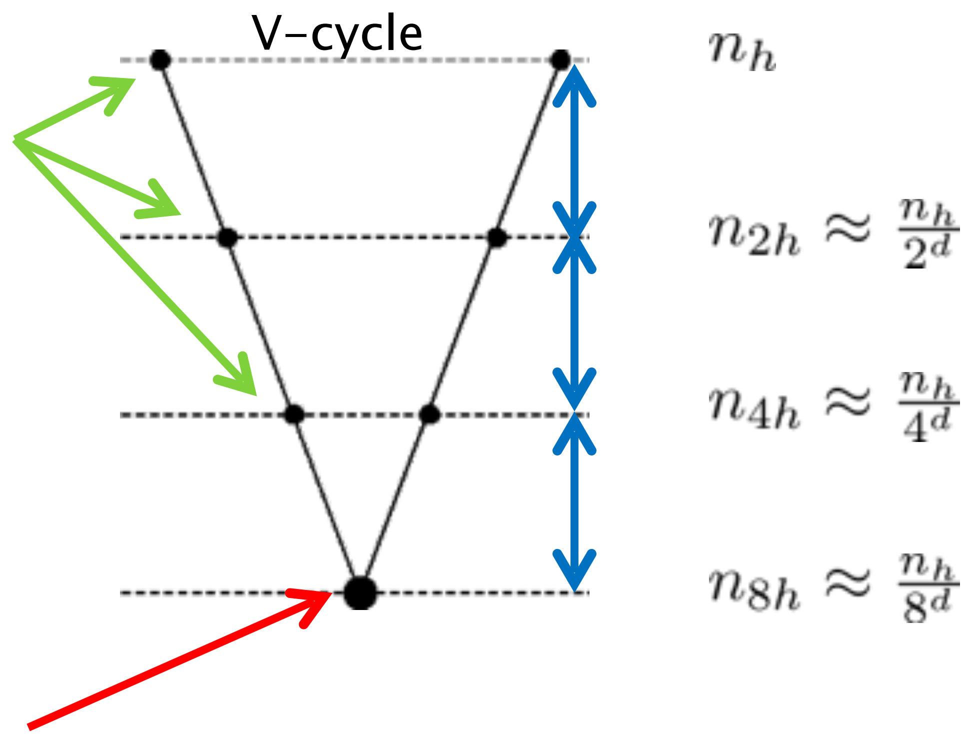
$$[\nabla F(\hat{x})]x = -F(\hat{x})$$



Multigrid Cycle

types of operation

problem sizes



fixed no. of smoother iterations:
vector scaling & addition, SpMV,
element–wise operations

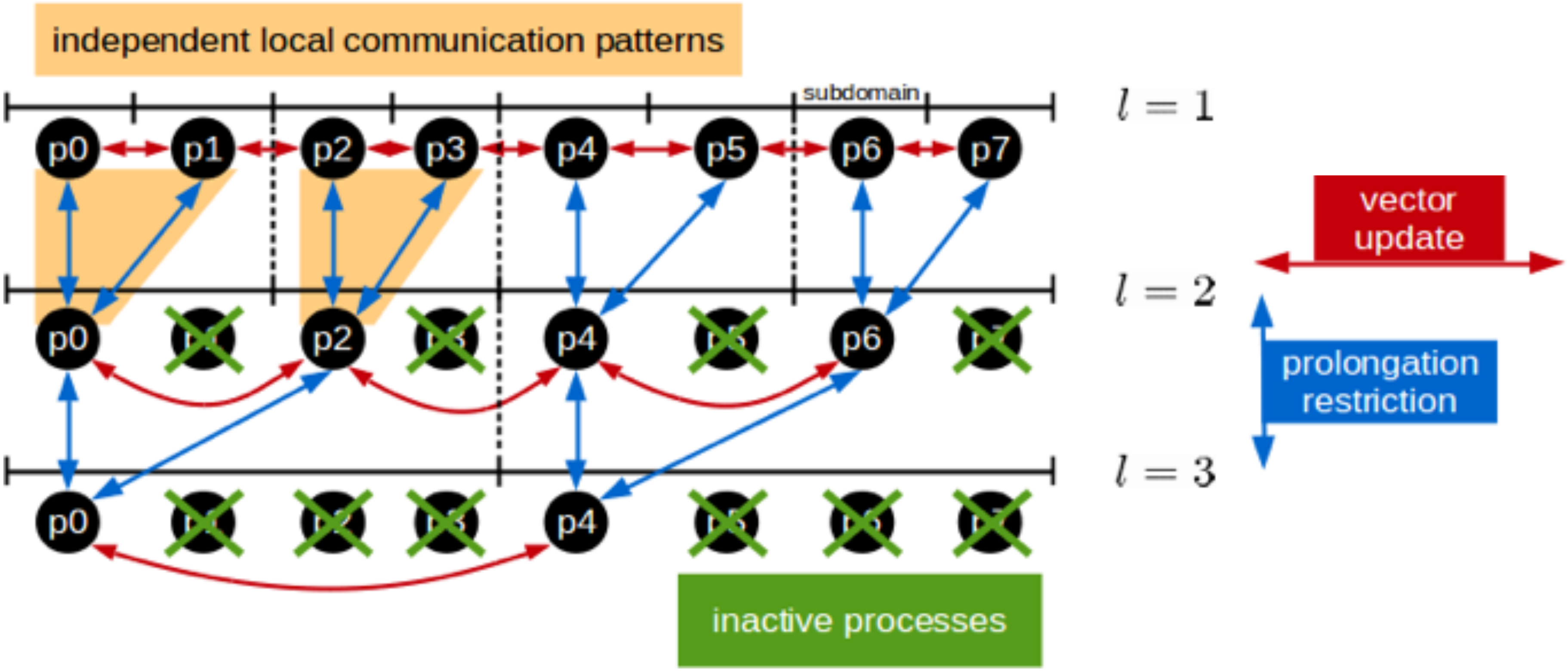
grid transfer:
SpMV, element–wise operations

solve error equation:
vector scaling & addition, SpMV,
scalar product

Multigrid Cycle

Dynamic adjustment of hardware activity

Adjust hardware activity according to solver needs



kill(pid, SIGUSR1); pause();

Results: Time and energy to solution

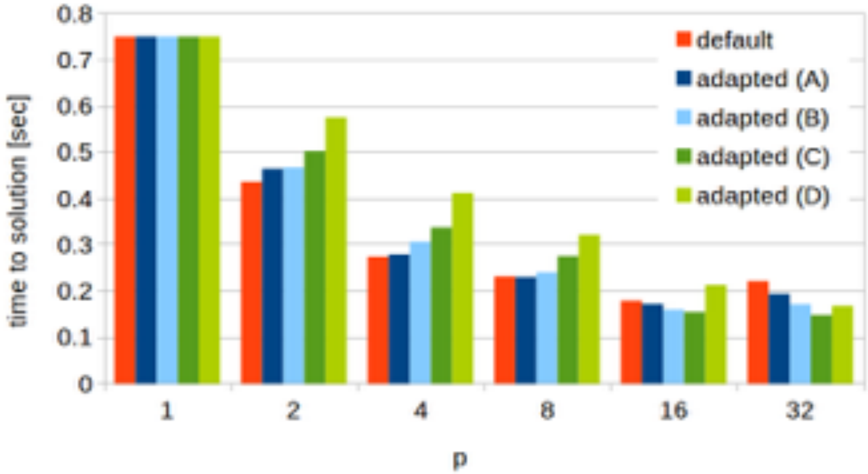
level 0   263 Thsd DoF

...

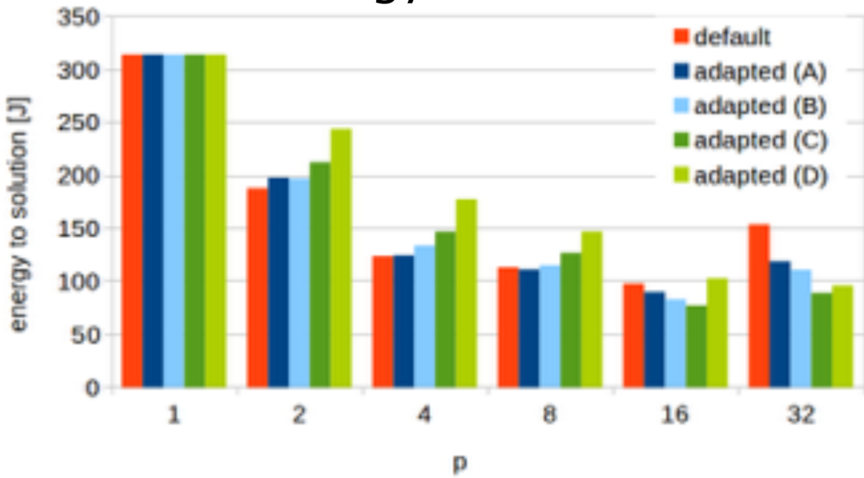
level 4   1 Thsd DoF

level	1	2	3	4	5
default	p	p	p	p	p
adapted (A)	p	p	p	p	$p/2$
adapted (B)	p	p	p	$p/2$	$p/4$
adapted (C)	p	p	$p/2$	$p/4$	$p/8$
adapted (D)	p	$p/2$	$p/4$	$p/8$	$p/16$

time to solution



energy to solution



global optimal time to solution adapted C 4 x 8-core CPU

saves 17%

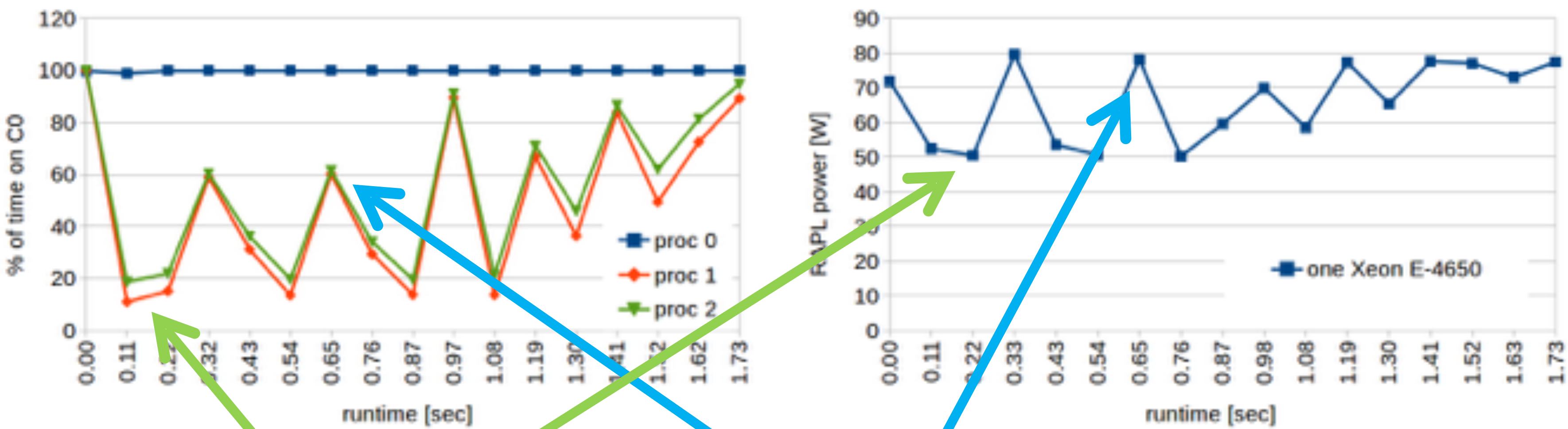
best default case

saves 21%

global optimal energy to solution adapted C 2 x 8-core CPU

Results: C-states and Intel RAPL

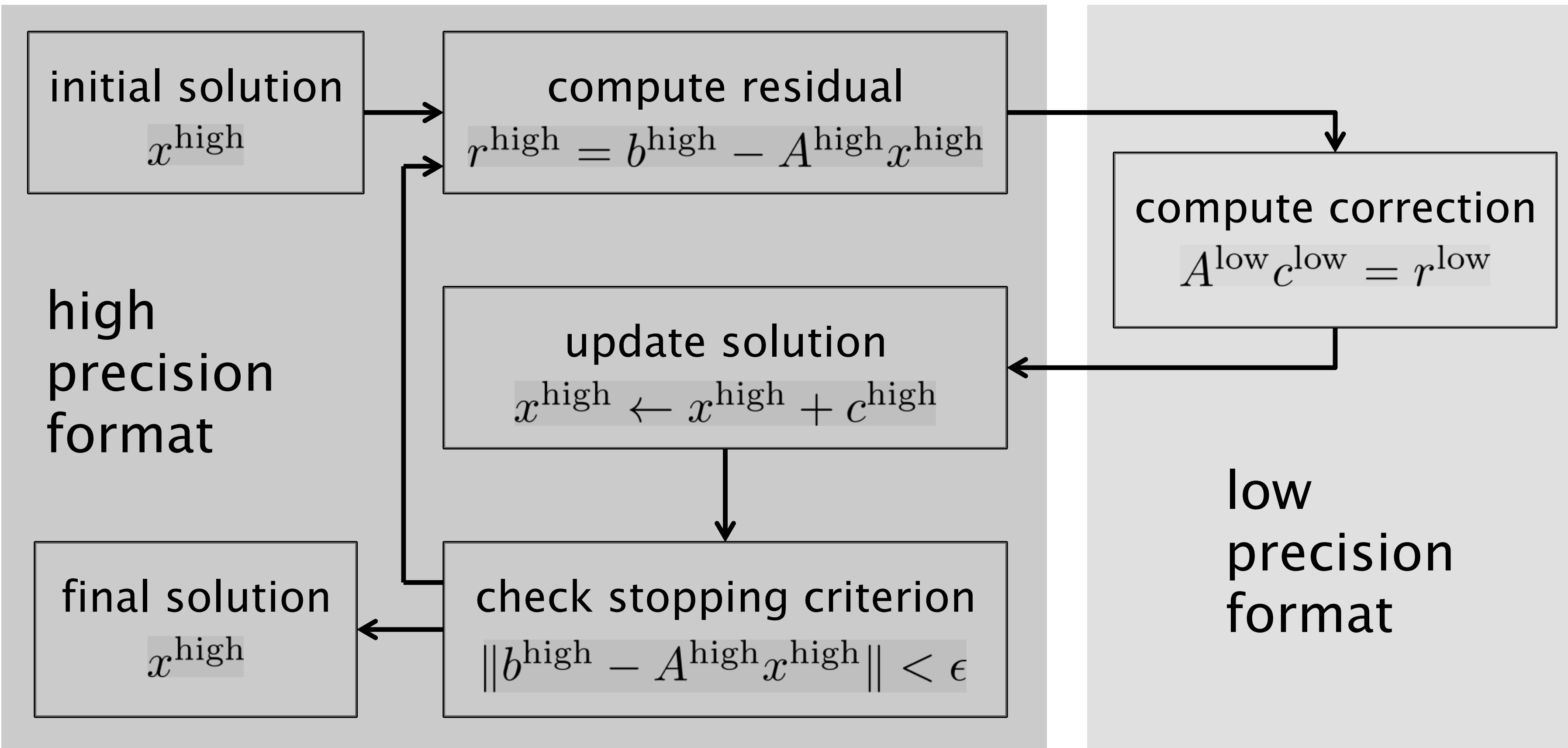
energy savings due to temporary CPU deactivation



multigrid cycle is on coarse grid

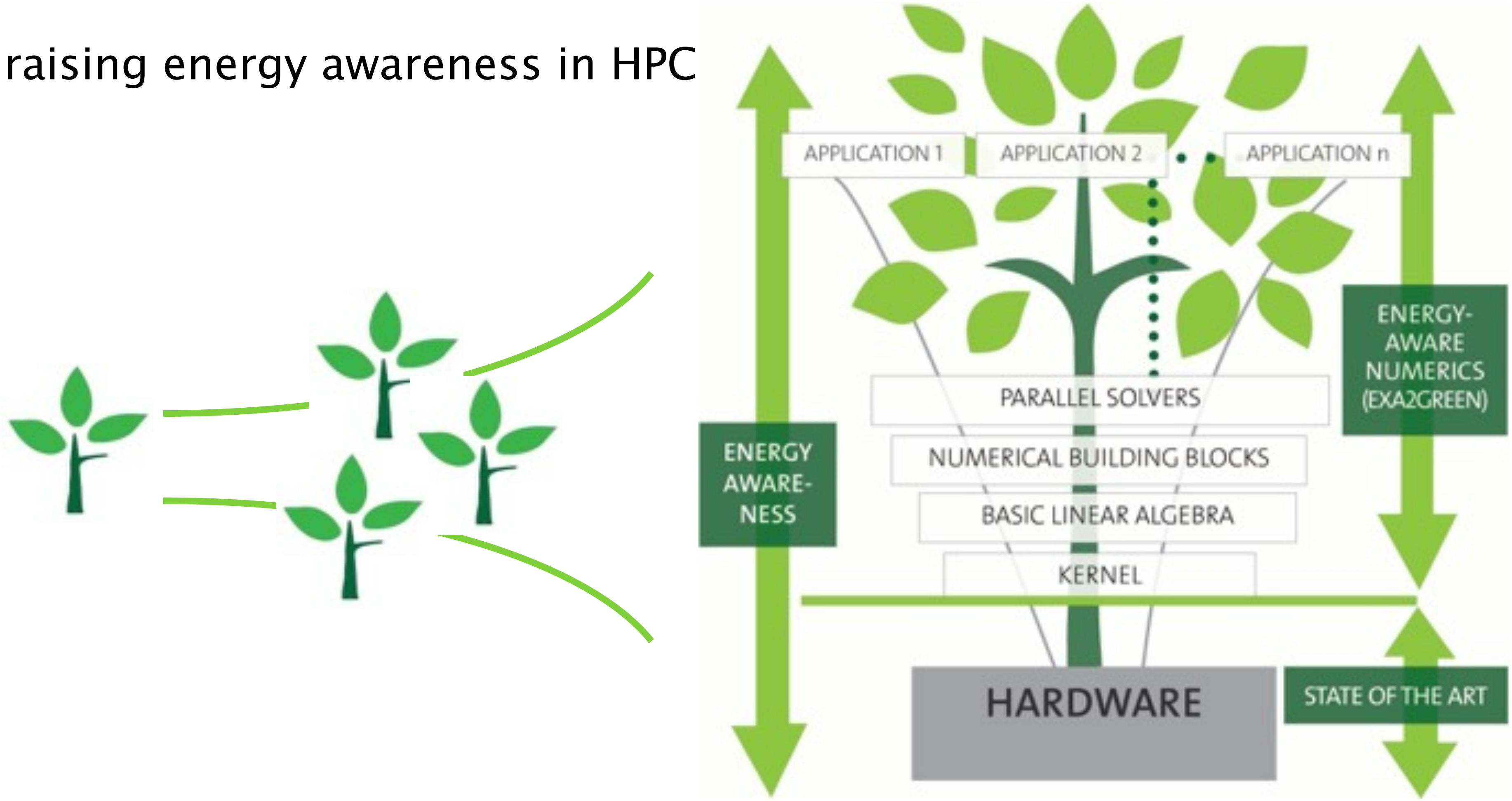
multigrid cycle is on fine grid

Mixed precision iterative refinement



On the road to exascale …

raising energy awareness in HPC



Thank you

Energy savings are possible while maintaining performance