Multigrid Algorithms on Heterogeneous Architectures

U. Rüde (FAU Erlangen, ulrich.ruede@fau.de)

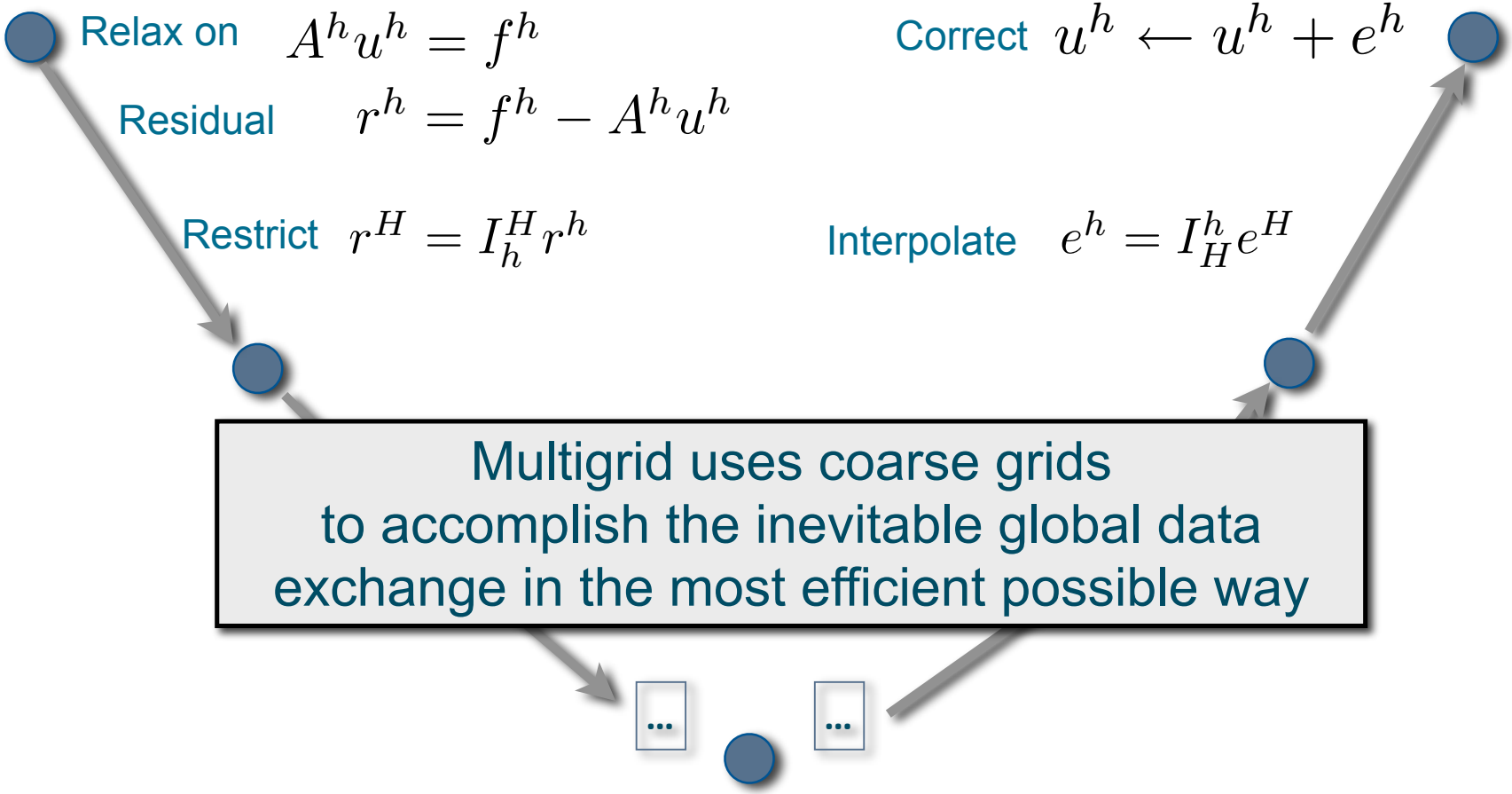
joint work with B. Gmeiner, Daniel Iuhasz, Sebastian Kuckuk, Markus Stuermer,

H. Koestler, H. Stengel (FAU),

M. Huber, C. Waluga, L. John, B. Wohlmuth (TUM)

Multigrid: Algorithms for 10^{12} unknowns

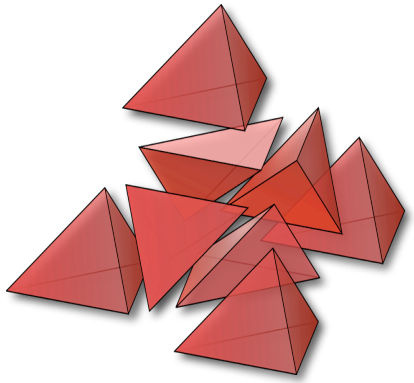
Goal: solve $A^h u^h = f^h$ using a hierarchy of grids



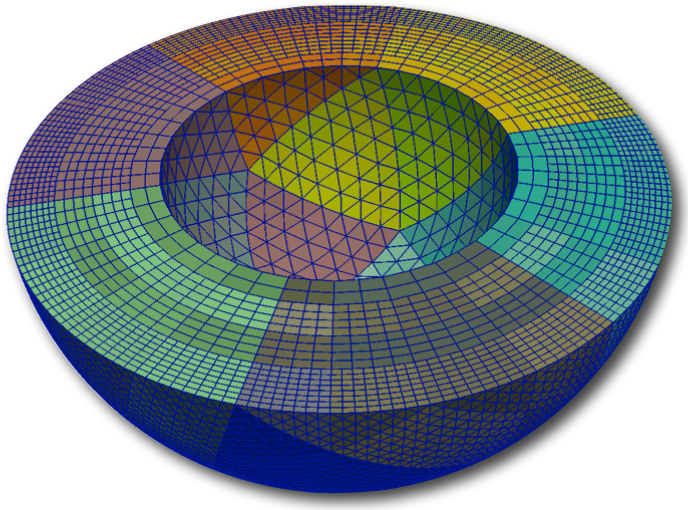
Hierarchical Hybrid Grids (HHG)

B. Bergen, F. Hülsemann, U. Rüde, G. Wellein: ISC Award 2006: „Is 1.7× 10^{10} unknowns the largest finite element system that can be solved today?“, SuperComputing, 2005.

- Parallelize „plain vanilla“ multigrid for tetrahedral finite elements
  - partition domain
  - parallelize all operations on all grids
  - **use clever data structures**
  - **matrix free** implementation
- Do not worry (so much) about Coarse Grids
  - idle processors?
  - short messages?
  - sequential dependency in grid hierarchy?
- Elliptic problems always require global communication. This cannot be accomplished by
  - local relaxation or
  - Krylov space acceleration or
  - domain decomposition without coarse grid

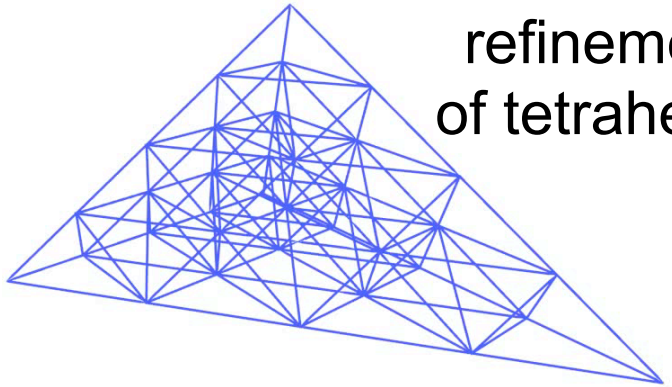


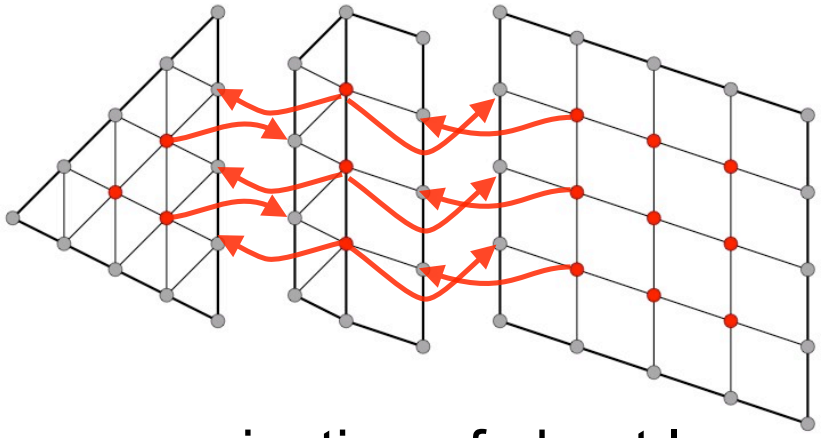
Bey‘s Tetrahedral Refinement



Regular tetrahedral refinement in HHG

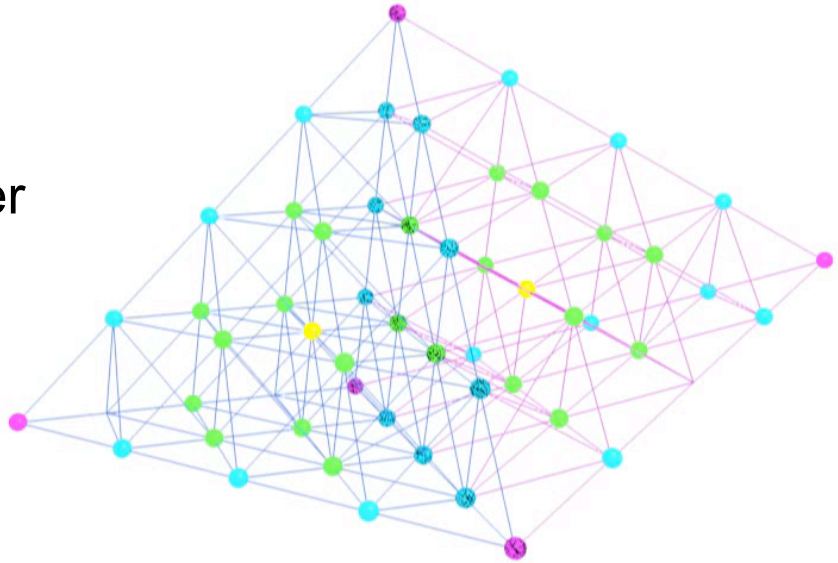
Structured refinement of tetrahedra



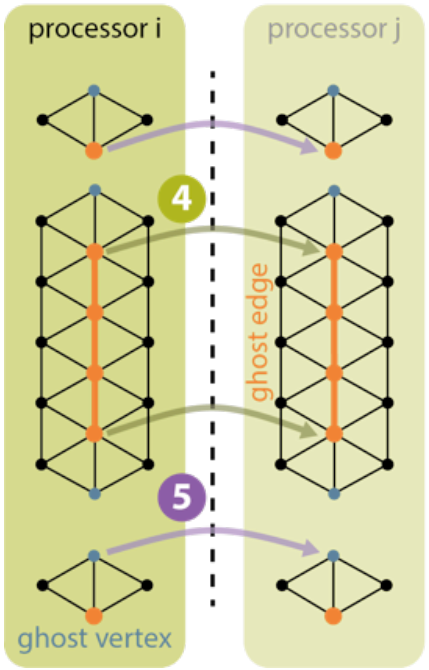


communication of ghost layers

- 3-D refinement hierarchy of tetrahedral patches
- Geometric hierarchy with 1-layer halos for
  - (volume) elements
  - faces
  - edges
  - vertices



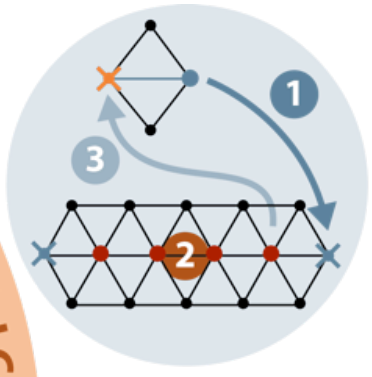
HHG Parallel Grid Traversal



for each **vertex** do
apply operation to vertex
update vertex **primary** dependencies
for each **edge** do
1 copy **from** vertex interior
2 apply operation to edge
3 copy **to** vertex **halo**
4 update edge **primary** dependencies
for each **element** do
copy from edge/vertex interiors
apply operation to element
copy to edge/vertex halos
5 update **secondary** dependencies

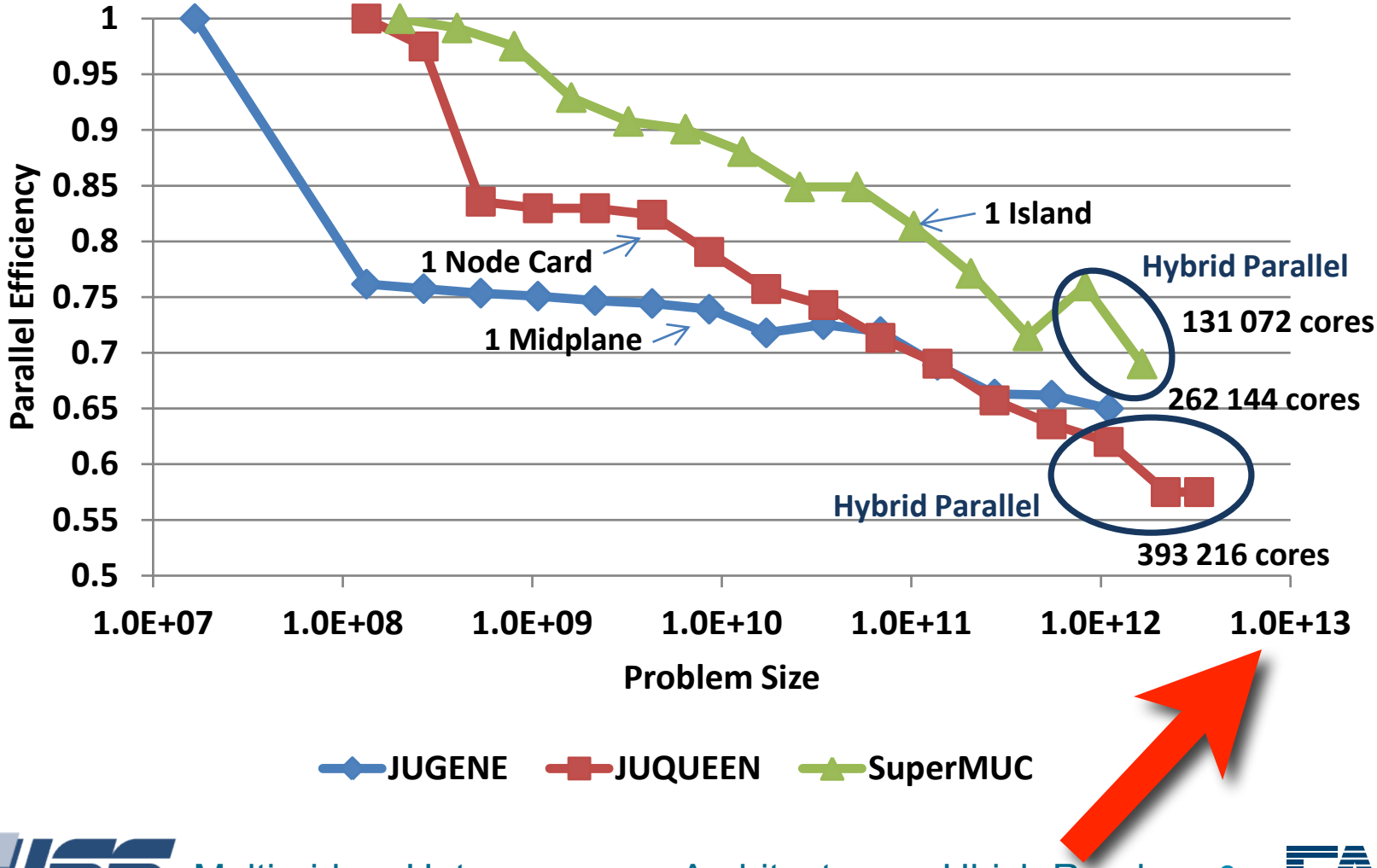
Communication

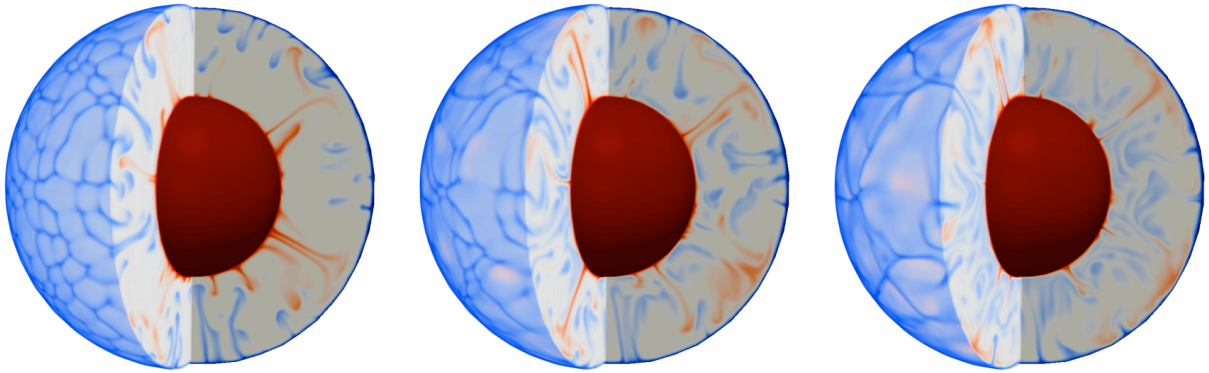
Parallel Computation



Parallel Efficiency of HHG

Gmeiner, Köstler, Stürmer, UR: *Parallel multigrid on hierarchical hybrid grids: a performance study on current high performance computing clusters,* Concurrency and Computation: Practice and Experience, 2014





work in progress:

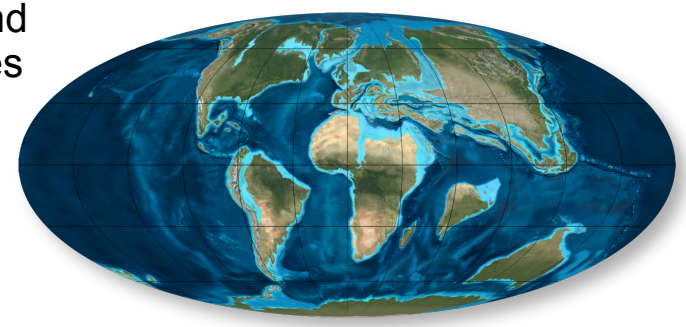
Application to Earth Mantle Convection Models

See also Christian Waluga‘s talk in MS 93:

From optimal algorithms to fast algorithms

HHG Solver for Stokes System
Motivated by Earth Mantle convection problem

Gmeiner, Waluga, Stengel, Wohlmuth, UR: Performance and Scalability of Hierarchical Hybrid Multigrid Solvers for Stokes Systems, to appear in SISC, 2015.



$$-\nabla \cdot (2\eta\epsilon(\mathbf{u})) + \nabla p = \rho(T)\mathbf{g},$$
$$\nabla \cdot \mathbf{u} = 0,$$
$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T - \nabla \cdot (\kappa \nabla T) = \gamma.$$

$\mathbf{u}$	velocity
p	dynamic pressure
T	temperature
ν	viscosity of the material
$\epsilon(\mathbf{u}) = \frac{1}{2}(\nabla\mathbf{u} + (\nabla\mathbf{u})^T)$	strain rate tensor
ρ	density
$\kappa, \gamma, \mathbf{g}$	thermal conductivity, heat sources, gravity vector

Scale up to ~10^{12} nodes/ DOFs
⇒ resolve the whole Earth Mantle globally with 1km resolution

Stokes equation: $-\operatorname{div}(\nabla\mathbf{u} - p\mathbf{I}) = \mathbf{f}$, $\operatorname{div}\mathbf{u} = 0$

FEM Discretization:

$$\mathbf{a}(\mathbf{u}_l, \mathbf{v}_l) + \mathbf{b}(\mathbf{v}_l, p_l) = \mathbf{L}(\mathbf{v}_l) \qquad \forall\, \mathbf{v}_l \in \mathbf{V}_l,$$
$$\mathbf{b}(\mathbf{u}_l, q_l) - \mathbf{c}(p_l, q_l) = 0 \qquad \forall\, q_l \in Q_l,$$

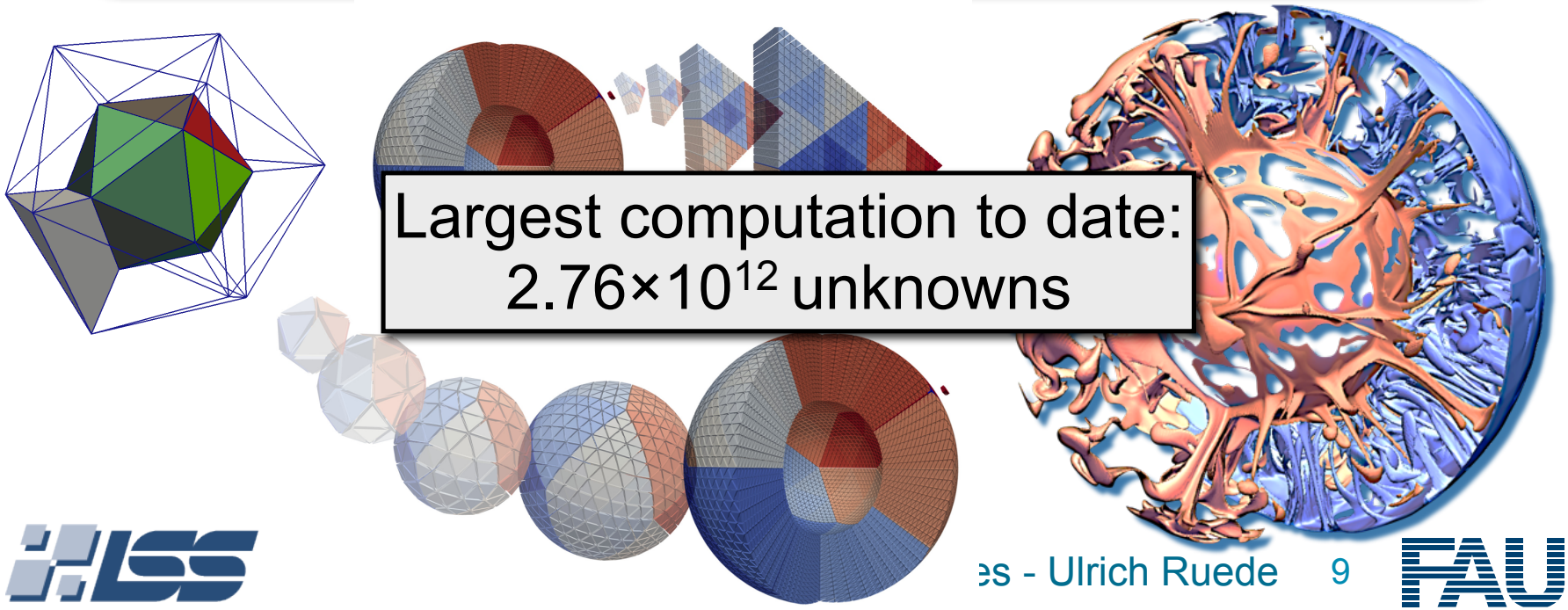
with: $\mathbf{a}(\mathbf{u},\mathbf{v}) := \int_\Omega \nabla\mathbf{u} : \nabla\mathbf{v}\,dx, \quad \mathbf{b}(\mathbf{u},q) := -\int_\Omega \operatorname{div}\mathbf{u} \cdot q\,dx$

Schur-complement formulation:

$$\begin{bmatrix} \mathbf{A}_l & \mathbf{B}_l^\top \\ \mathbf{0} & \mathbf{C}_l + \mathbf{B}_l\mathbf{A}_l^{-1}\mathbf{B}_l^\top \end{bmatrix} \begin{bmatrix} \underline{\mathbf{u}}_l \\ \underline{p}_l \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{f}}_l \\ \mathbf{B}_l\mathbf{A}_l^{-1}\underline{\mathbf{f}}_l \end{bmatrix}$$

HHG Weak Scalability on JuQueen for Stokes

Nodes	Threads	Grid points	Resolution	Time: (A)	(B)
1	30	$2.1 \cdot 10^{07}$	32 km	30 s	89 s
4	240	$1.6 \cdot 10^{08}$	16 km	38 s	114 s
30	1 920	$1.3 \cdot 10^{09}$	8 km	40 s	121 s
240	15 360	$1.1 \cdot 10^{10}$	4 km	44 s	133 s
1 920	122 880	$8.5 \cdot 10^{10}$	2 km	48 s	153 s
15 360	983 040	$6.9 \cdot 10^{11}$	1 km	54 s	170 s



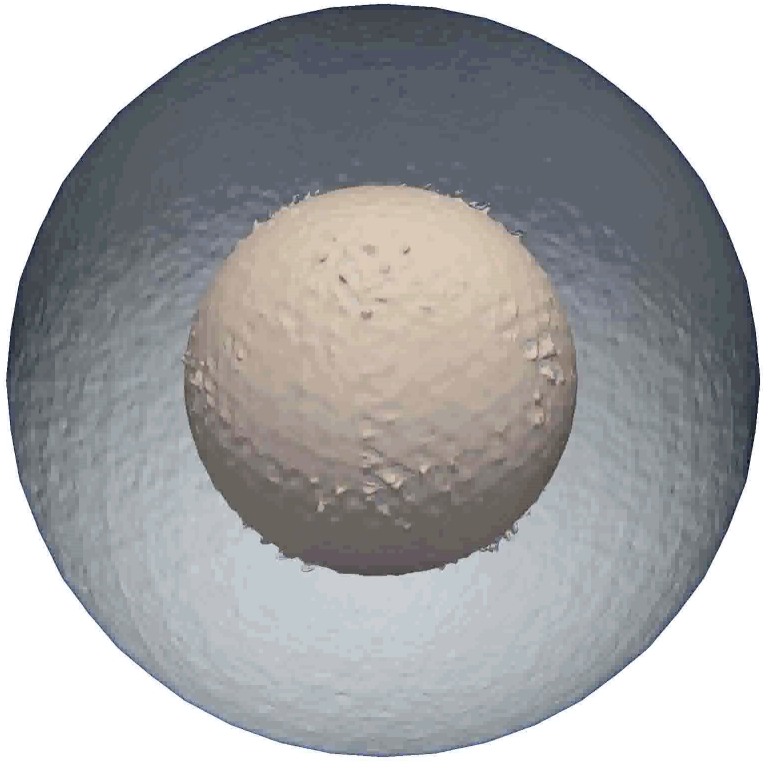
Largest computation to date:
2.76×10^{12} unknowns

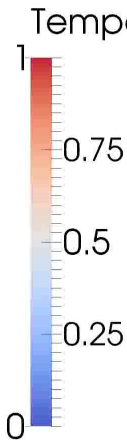
Instationary Computation

$$-\Delta \mathbf{u} + \nabla p = -\,\mathrm{Ra} T\,\hat{\mathbf{r}}$$

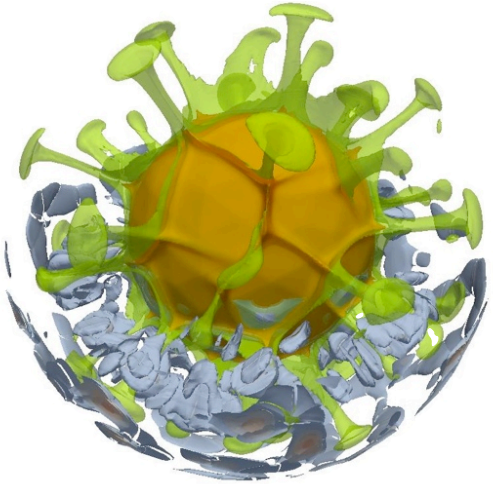
$$\operatorname{div} \mathbf{u} = 0$$

$$\partial_t T + \mathbf{u} \cdot \nabla T = \mathrm{Pe}^{-1} \Delta T$$





Simplified Model



6.5×10^9 DOF, 10000 time steps: run time 7 days

Mid-size cluster: 288 compute cores in 9 nodes of LSS at FAU

PeSOptimizing the Performance

Pessimize ←

with greetings from D. Bailey's: *Twelve Ways to Fool the Masses...*

- Bring loops in wrong order, ignore caches, randomize memory access, use many small MPI messages
  - **$10^{12} \rightarrow 10^{11}$ unknowns**
- Do not use a matrix-free imp… …eep in mind that a single multipl… …ass and stiffness matrix can … accesses per unknown):
  - **$10^{11} \rightarrow 10$…**
- Gain ad… …by using unoptimized uns… …direct mem access costs!)
  - **…knowns**
- Incre… …orithmic overhead, e.g. permanently check… convergence, use the most expensive error estimator, etc. etc.
  - **$10^{9} \rightarrow 10^{8}$ unknowns (... still a large system ...)**

An optimal algorithm is not necessarily fast

→ Optimize

Towards a holistic performance engineering methodology:

Parallel Textbook Multigrid Efficiency

Gmeiner, UR, Stengel, Waluga, Wohlmuth: Towards Textbook Efficiency for Parallel Multigrid, Numerical Mathematics: Theory, Methods and Applications 8, 2015, pp 22 - 46

Textbook Multigrid Efficiency (TME)

„Textbook multigrid efficiency means solving a discrete PDE problem with a computational effort that is only a small (less than 10) multiple of the operation count associated with the discretized equations itself." [Brandt, 98]

- work unit (**WU**) = single elementary relaxation
- classical ***algorithmic*** TME-factor:

  ops for solution/ ops for work unit
- new ***parallel*** TME-factor to quantify
  - algorithmic efficiency
  - combined with implementation scalability

Parallel TME

μ_{sm} # of elementary relaxation steps on single core/sec

U # cores

$U\,\mu_{\mathrm{sm}}$ aggregate relaxation performance

$T_{\mathrm{WU}}(N,U) = \dfrac{N}{U\,\mu_{\mathrm{sm}}}$ **idealized** time for a work unit

$T(N,U)$ time to solution for N unkowns on U cores

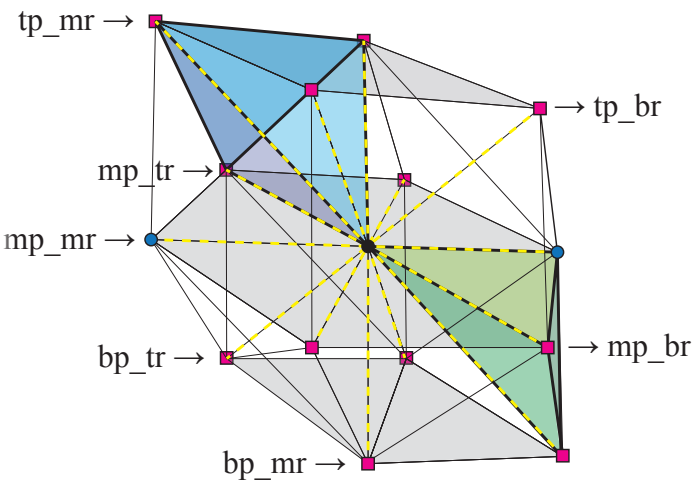
Parallel textbook efficiency factor

$$E_{\mathrm{ParTME}}(N,U) = \frac{T(N,U)}{T_{\mathrm{WU}}(N,U)} = T(N,U)\frac{U\,\mu_{\mathrm{sm}}}{N}$$

combines **algorithmic** and **implementation** efficiency.

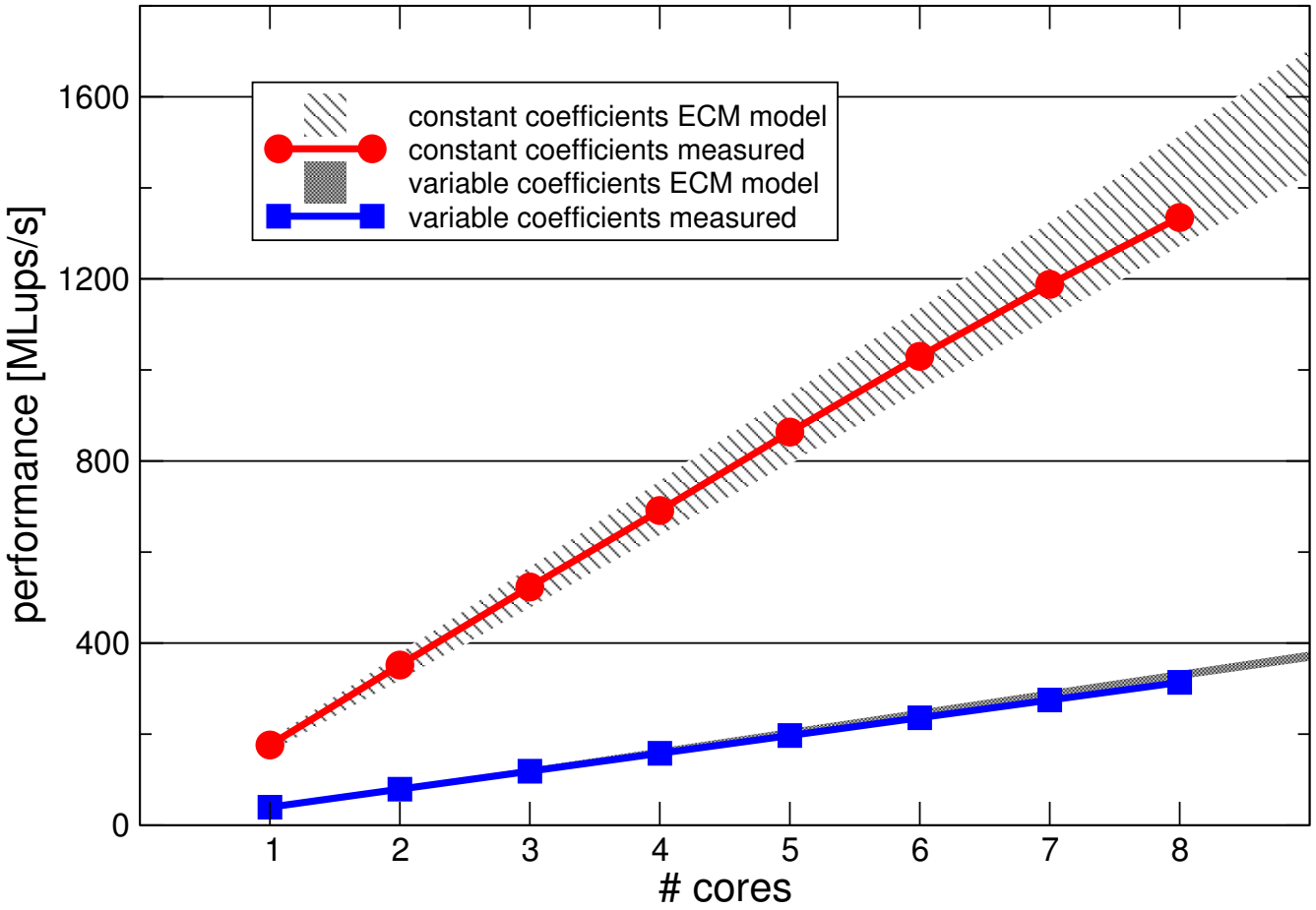
Analysing TME Efficiency: RB-GS Smoother

```
for (int i=1; i < (tsize-j-k-1); i=i+2) {
   u[mp_mr+i] = c[0] * (
    -c[1] *u[mp_mr+i+1] - c[2] *u[mp_tr+i-1] -
     c[3] *u[mp_tr+i]   - c[4] *u[tp_br+i]   -
     c[5] *u[tp_br+i+1] - c[6] *u[tp_mr+i-1] -
     c[7] *u[tp_mr+i]   - c[8] *u[bp_mr+i]   -
     c[9] *u[bp_mr+i+1] - c[10]*u[bp_tr+i-1] -
     c[11]*u[bp_tr+i]   - c[12]*u[mp_br+i]   -
     c[13]*u[mp_br+i+1] - c[14]*u[mp_mr+i-1] +
     f[mp_mr+i] );
```



- This loop should be executed on **single SuperMuc core** at
  - 720 M updates/sec (*in theory* - peak performance)
  - μ_{sm} = **176 M** updates/sec (*in practice* - memory access bottleneck; RB-ordering prohibits vector loads)
- Thus **whole SuperMuc** should perform
  - $U\,\mu_{\mathrm{sm}}$ = 147456*176M ≈ **26T** updates/sec

ECM: single-chip prediction vs. measurement



Sandy Bridge single-chip performance scaling of the stencil smoothers on 256^3 grid points. Measured data and ECM prediction ranges shown.

TME and Parallel TME results

Setting/Measure	E_{TME}	E_{SerTME}	E_{NodeTME}	E_{ParTME1}	E_{ParTME2}
Grid points	-	$2 \cdot 10^6$	$3 \cdot 10^7$	$9 \cdot 10^9$	$2 \cdot 10^{11}$
Processor cores U	-	1	16	4096	16384
(CC) - FMG(2,2)	6.5	15	22	26	22
(VC) - FMG(2,2)	6.5	11	13	15	13
(SF) - FMG(2,1)	31	64	100	118	-

- Three model problems:
  - scalar constant & scalar variable coefficients
  - Stokes solved via Schur complement
- Full multigrid with #iterations such that asymptotic optimality maintained
- TME = 6.5 (algorithmically for scalar cases)
- ParTME around 20 for scalar, and ≥100 for Stokes

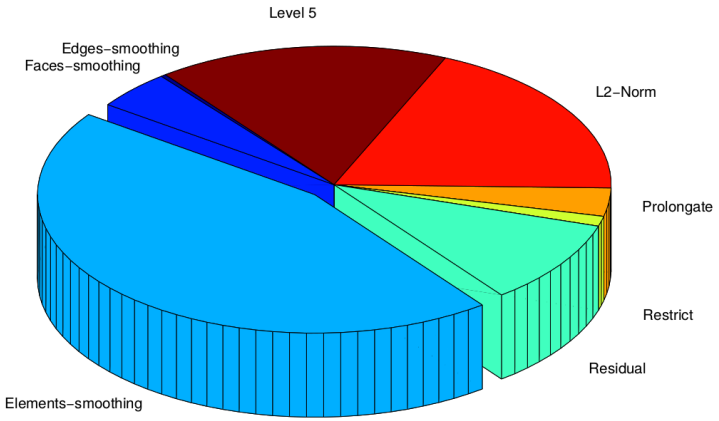
Using Accelerators:

GPU Implementation of Smoothers in HHG

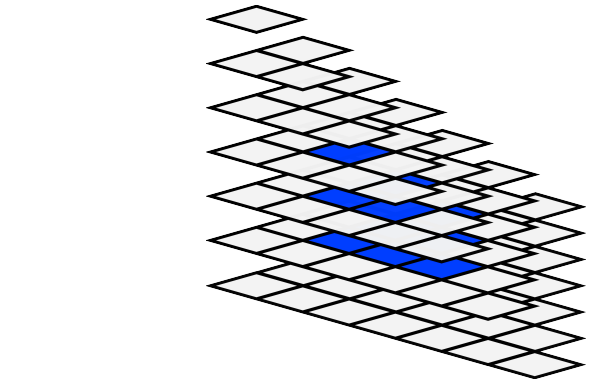
Daniel Iuhasz, Performance Analysis and Optimization of GPU Kernels for HHG, Master Thesis, FAU Erlangen, 2014

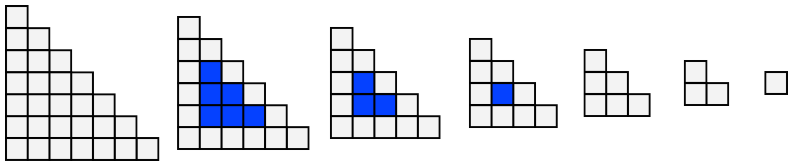
GPU for smoother in HHG Data Structure

- Basic HHG profiling

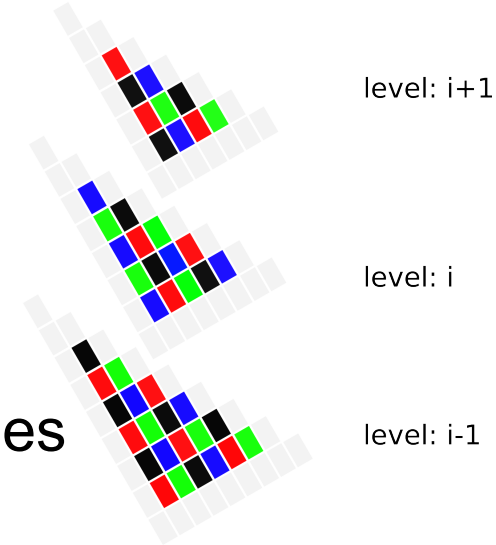


- Memory Layout





Coloring schemes to break data dependencies



- Difficulties:
  - index calculation
  - short loops

Architectures

Platform	Memory	Bandwidth	Cores	Peak
GTX580	1536	192.4	512	1581
Titan	6144	288.4	2688	4709

- Smoother will be memory bandwidth bound
- **bandwidth limited peak performance**

Line Size	IT GTX580(s)	IT TITAN(s)
9	2.75676e-08	1.83911e-08
17	3.24449e-07	2.16449e-07
33	3.18017e-06	2.12158e-06
65	2.8073e-05	1.87283e-05
129	0.000235653	0.000157211
257	0.00193053	0.00128792

GPU Smoother Performance Experiments

Platform	ideal band-width (s)	plane-wise parallel (s)	globally parallel red-back (s)	globally parallel six-color (s)	precomp. indirect address (s)
GTX580	0.0019	2.0894	0.02880	0.00632	0.003132
Titan	0.0013	3.2546	0.02817	0.00453	0.003211

- For HHG tetrahedron with side length 257 points
- Final GPU version uses precomputed indices and indirect addressing
- Comparison with highly optimized kernel on 8 core Sandy Bridge EP Intel Xeon E5-2680 8C (SuperMuc node): 0.0021s
- Best CPU: 1300 MLUPs vs. best GPU (Titan): 850 MLUPs
- GPU versions carefully analysed with NV-metrics (Nvidia profiler) ... see thesis by Daniel Iuhasz
- GPUs not yet attractive for HHG FE system

Conclusions

- 2.7×10^{12} unknowns in fully implicit solve with Peta-Scale
- But optimal algortithms are not necessarily fast
- Multigrid scales to Peta and beyond
- HHG: lean and mean implementation: excellent time to sol.
- but difficult to get accelerators up to good performance
- Next steps: Improve
  - geophysical models
  - discretization & solver for Stokes
  - resilience: ABFT

Thanks to:

DFG SPP Exa

KONWIHR

Elite Network of Bavaria