Adaptive Mesh Refinement (AMR)

Carsten Burstedde
Omar Ghattas, Georg Stadler, Lucas C. Wilcox

Institute for Computational Engineering and Sciences (ICES)
The University of Texas at Austin

Collaboration with
George Biros, Michael Gurnis, Shijie Zhong
under NSF PetaApps Program (OCI-0749334)
and
Wolfgang Bangerth

CIG meeting on
Opportunities and Challenges in Computational Geophysics
California Institute of Technology

March 30, 2009

Outline

- Motivation
- Types of adaptivity
- Parallel partitioning
- Space filling curves
- Forest of octrees
- Octree-based software
- Adaptivity and mantle convection
- Adaptivity and CIG

Promise of adaptivity

Invest computational work selectively
where it promises the highest gain in accuracy.

pros

- improve $\frac{\text{accuracy}}{\text{runtime}}$
- implement goal-oriented refinement
- mitigate curse of dimensionality
- turn intractable problems into tractable ones

cons

- non-trivial neighborhood relations
- non-trivial mesh partition
- non-trivial node ownership
- surplus of complexity costs development time
- surplus of topological operations costs runtime

Indications for adaptivity

Use adaptivity when error, energy, activity, ...
is distributed non-uniformly in space.

less likely

- when multiscale behaviour permeates the domain
- when activity spreads through the whole system

more likely

- when spatial resolution increases
- when physical heterogeneity increases
- when quantities of interest are localized

examples

– Turbulence
? Wave propagation
+ Mantle convection

Growth in computing power points towards the more likely regime?

Large scale adaptivity

static AMR (i.e., up-front adaptation)

- Mesh and parallel partition are known before program start
- Mesh setup can be precomputed and hand-optimized
- Cannot adapt to moving phenomena
- Change in setup can be costly

dynamic AMR (i.e., mesh changes over time)

- Additionally requires coarsening capability
- Mesh adaptation is integral part of the code
- Mesh adaptation occurs frequently
- Parallel repartition occurs frequently

> Adaptivity algorithms need to be at least as scalable as the numerical algorithms

Large scale adaptivity

mesh partitioning

- Each element is assigned to a unique processor core
- Distributed storage – no processor stores the whole mesh
- Numerical information exchanged between neighbor elements
- Connectivity information between elements known or stored

connectivity information – local

- Find neighbor elements
- Find owner processor of a neighbor
- Find degrees of freedom (DOF)

connectivity information – global

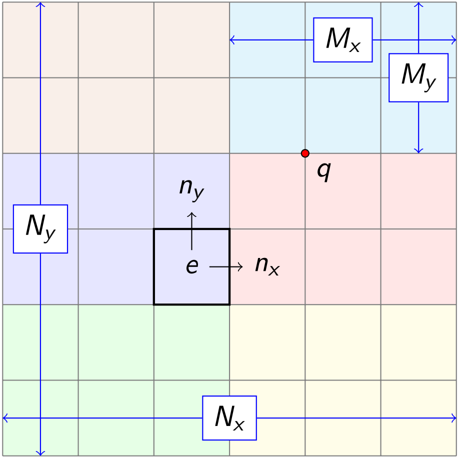
- Parallel partitioning – global redistribution of elements

Encode connectivity information with minimal storage

Types of adaptivity differ in choice of encoding

Types of adaptivity

local information (uniform mesh)



find neighbors

$$\#n_x = \#e + 1$$
$$\#n_y = \#e + N_x$$

find owner

$$P(e) = \left\lfloor \frac{e.x + N_x \lfloor \frac{e.y}{M_y} \rfloor}{M_x} \right\rfloor$$

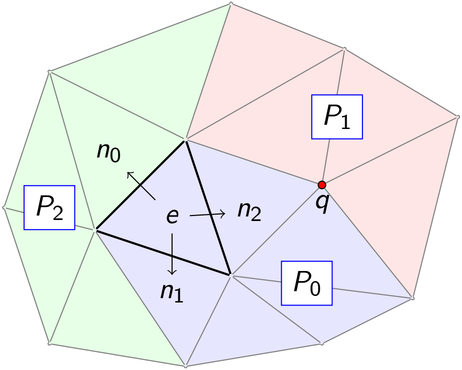
find DOF

$$\#q = q.x + (N_x + 1)q.y$$

global shared information: 4 integers N_x, N_y, M_x, M_y

Types of adaptivity

local information (unstructured mesh)



find neighbors

store $e.n[3]$

find owner

store $e.P$

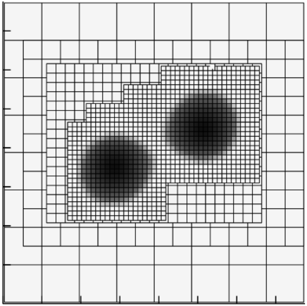
find DOF

store $\#q$

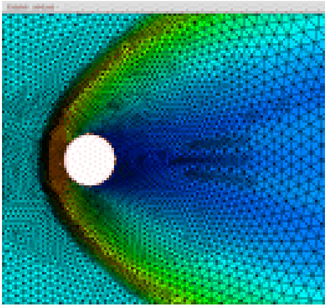
Loss of structure $\longrightarrow$ loss of information
Required information must be stored

Types of adaptivity

block-structured AMR [1,2]



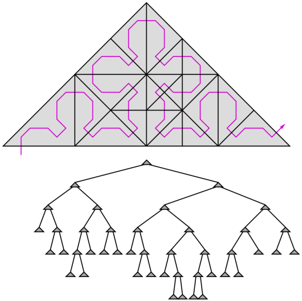
unstructured AMR [3]



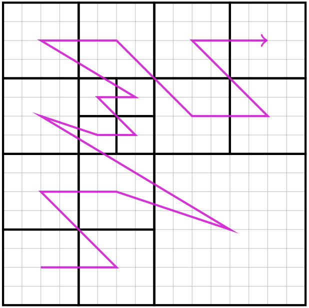
[1] E. Evans, S. Iyer, E. Schnetter et.al. 2005
[2] www.cactuscode.org
[3] www.openchannelfoundation.org/projects/Pyramid

Types of adaptivity

hierarchical triangles [1]



hierarchical hexahedra



[1] M. Bader, S. Schraufstetter, C.A. Vigh, J. Behrens 2008

Parallel partitioning

partitioning – global exchange of information

- Operation: redistribute elements among processors
- Objective: miminize overall run time

useful criteria?

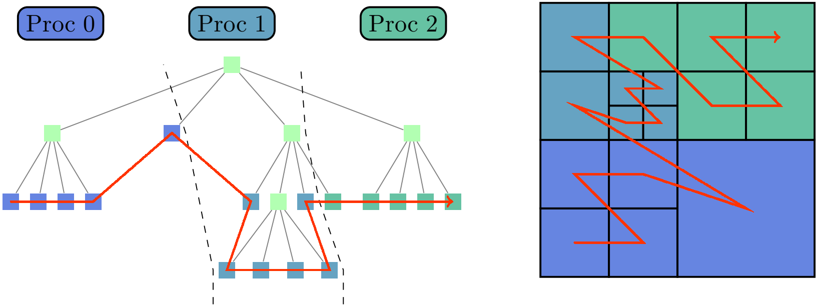
- Balance element counts between processors
- Balance node counts between processors
- Minimize number of neighbor processors
- Minimize number of elements on processor boundaries

tools available?

- Unstructured AMR: represent mesh connectivity as graph; use graph-partitioning heuristics (NP-hard), e.g. `Zoltan`
- Hierarchic AMR: space filling curves (SFC)

Parallel partitioning

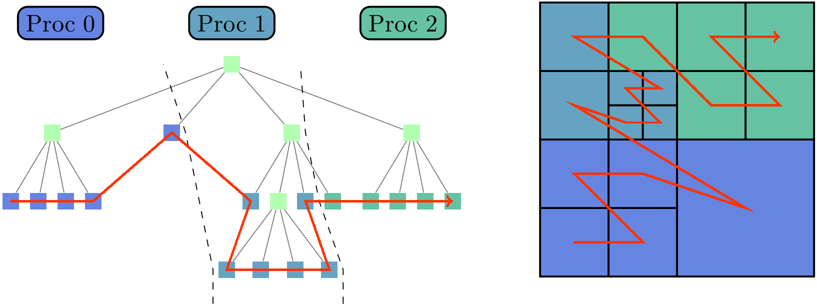
octrees and space filling curves (SFC)



- 1:1 relation between octree and SFC → efficient encoding
- Map a 1D curve into 2D or 3D space → total ordering
- Recursive self-similar structure → scale-free
- Tree leaf traversal → cache-friendly

Large scale adaptivity

octrees and space filling curves (SFC)

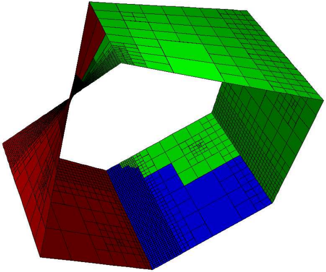


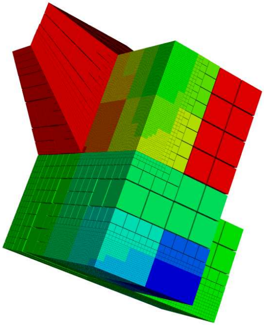
local information

- Find parent or children $\longrightarrow$ vertical tree step $\mathcal{O}(1)$
- Find on-processor neighbor $\longrightarrow$ tree search $\mathcal{O}(\log \frac{n}{p})$
- Find owner of off-processor neighbor $\longrightarrow$ binary search $\mathcal{O}(\log p)$

Forest of octrees

a conforming macro-mesh of adaptive octrees





- Connectivity between octrees is interpreted purely topological
- Any # of octrees (=≠ 4) can connect through common edge
- Any # of octrees (=≠ 8) can connect through common corner
- 2:1 balance condition across faces, edges and corners
  is honored within and between octrees (optional)

Octree-based parallel adaptive software

reinventing the wheel? (can be great fun! takes time though.)

- deal.II (W. Bangerth, R. Hartmann, G. Kanschat; general purpose)
- libMesh (G. Carey, D. Gaston, B. Kirk, J. Peterson, R. Stogner)
- AFEAPI (A. Patra et.al.)
- octor (T. Tu; closed source, pointer-based, ripple propagation)
- Dendro (R. Sampath; linear octree, insulation layers)
- `p4est` (C. Burstedde, L. C. Wilcox; forest of linear octrees)

Many of the headaches of parallel adaptivity
are happily encapsulated in a software library

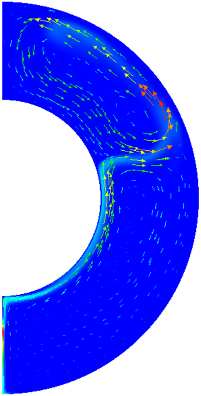
deal.II

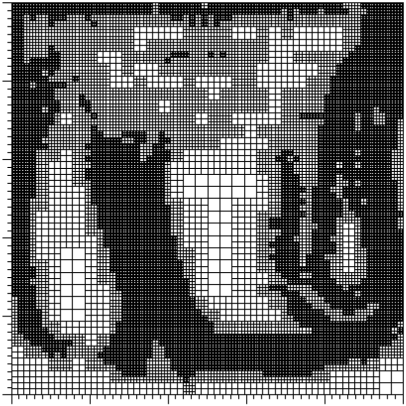
all-in-one finite element package

- Forest of octrees mesh topology
- Wealth of finite element spaces
- Problem assembly and linear algebra
- Direct and iterative numerical solvers
- Wealth of visualization formats
- Wealth of documentation
- Wealth of tutorials (including geodynamics!)
- Moderate parallelism ($\approx$ 100 processor cores)
- Directly available for download (`www.dealii.org`)

Get up and running quickly

fluid dynamics examples [1]





[1] W. Bangerth 2008

p4est

lightweight parallel adaptive mesh library (not a FE code!)

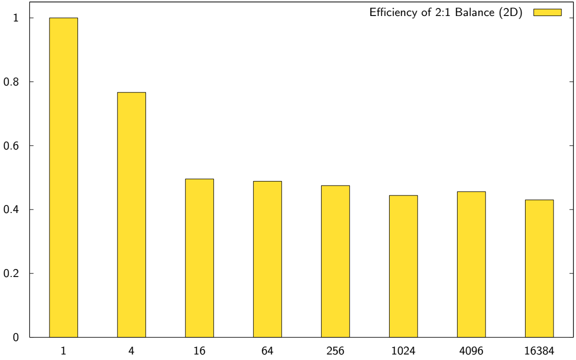
- Forest of octrees mesh topology
- Designed for uncompromised parallel scalability
- Almost arbitrary connectivity and periodicity of the domain
- Small memory footprint

local storage	24 bytes per element
global storage	32 bytes for each processor

- Ongoing work on integration into deal.II as mesh backend
- Ongoing work on generic FE mesh interface

2D scalings

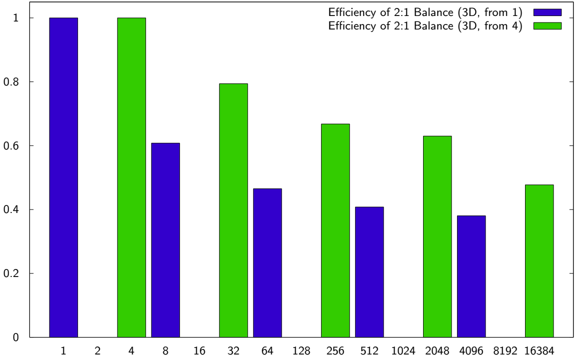
Star-shaped domain, 6 trees, parallel efficiency of 2:1 balance



Largest run: 6 trees, 32,768 cores, 74 billion elements

3D results

Spherical shell domain, 24 trees, parallel efficiency of 2:1 balance



Largest run: 24 trees, 62,464 cores, 256 billion elements

Adaptivity and mantle convection

the Rhea code [1,2]

- Global adaptive mantle convection simulation
- Continuous trilinear elements for both velocity and pressure
- AMG preconditioned MINRES iterations for Stokes
- SUPG predictor-corrector time integration
- Spherical shell resolved with 24 octrees by `p4est`
- Scaled up to 16,384 cores on TACC/Ranger

[1] C. Burstedde, O. Ghattas, M. Gurnis, E. Tan,
G. Stadler, T. Tu, L. C. Wilcox, Z. Zhong 2008.
Finalist paper for the '08 Gordon Bell Prize
[2] ongoing work

Adaptivity and mantle convection

the Rhea code – multigrid scalings [1]

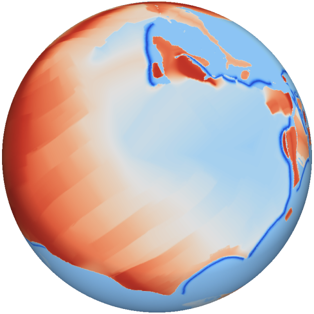
#cores	#dofs	MINRES #iter	AMG setup	AMG V-cycle
1	170K	66	1.45	18.06
8	1.1M	76	1.60	22.91
32	4.6M	88	2.22	33.20
128	17.9M	81	3.41	30.22
2,048	294M	63	15.12	70.53
16,384	2.35B	71	26.91	84.96

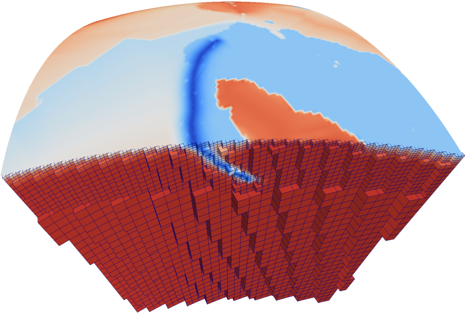
- Sum of setup and V-cycle times increase by a factor of 5.5
- All other FE computations scale roughly linearly

[1] With ML solver from Trilinos

Adaptivity and mantle convection

the Rhea code – present day slab dynamics [1]





[1] ongoing work with L. Alisic, O. Ghattas, M. Gurnis, G. Stadler, L. C. Wilcox

Adaptivity and CIG

high-level code

- deal.II works up to small computer clusters
- deal.II works for uniform and adaptive meshes
- Worth considering when starting a new high-level code

technology transfer

- `p4est` creates new scalable technology
- This technology propagates, e.g. into deal.II
- Whoever uses these codes will benefit without extra effort

special purpose codes

- Codes that bring their own FE logic could use `p4est`
- Mesh management would need to be separated
- Active collaboration with code development on both sides