Adaptive Hierarchical Grids with a Trillion Tetrahedra

Tobias Gradl, Björn Gmeiner and U. Rüde
(LSS Erlangen, ruede@cs.fau.de)
in collaboration with many more

Lehrstuhl für Informatik 10 (Systemsimulation)
Universität Erlangen-Nürnberg
www10.informatik.uni-erlangen.de
Seattle, February 2010

SIAM Parallel Processing
Overview

- Motivation: towards Exa-Scale!
- Hierarchical Hybrid Grids (HHG)
- Parallel Performance
- Adaptive HHG
- τ - Extrapolation
- Conclusions
Towards ExaScale and Beyond
How much is a PetaFlops?

- $10^6 = 1$ MegaFlops: Intel 486
  - 33MHz PC (~1989)

- $10^9 = 1$ GigaFlops: Intel Pentium III
  - 1GHz (~2000)
  - If every person on earth computes one operation every 6 seconds, all humans together have ~1 GigaFlops performance (less than a current laptop)

- $10^{12} = 1$ TeraFlops: HLRB-I
  - 1344 Proc., ~2000

- $10^{15} = 1$ PetaFlops
  - 122400 Cores (Roadrunner, 2008)
  - If every person on earth runs a 486 PC, we all together have an aggregate Performance of 6 PetaFlops.

- ExaScale (~$10^{18}$ Flops) around 2018?
Trends in Computer Architecture

- On Chip Parallelism
  - instruction level
  - multicore
- Off Chip parallelism
- Limits to clock rate
- Limits to memory bandwidth
- Limits to memory latency
What are the consequences?

❖ For the application developers “the free lunch is over”
   ▪ Without explicitly parallel algorithms, the performance potential cannot be used any more

❖ For HPC
   ▪ CPUs will have 2, 4, 8, 16, ..., 128, ..., ??? cores - maybe sooner than we are ready for this
   ▪ We will have to deal with systems with millions of cores

❖ The memory wall grows higher
Multigrid: V-Cycle

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

Solve $A^H e^H = r^H$

by recursion

Correct $u^h \leftarrow u^h + e^h$

Interpolate $e^h = I^H_H e^H$

Restrict $r^H = I^H_h r^h$

Residual $r^h = f^h - A^h u^h$

Relax on $A^h u^h = f^h$

Goal: solve $A^h u^h = f^h$ using a hierarchy of grids
How fast are our algorithms (multigrid) on current CPUs

- **Assumptions:**
  - Multigrid requires 27.5 Ops/unknown to solve an elliptic PDE (Griebel ´89 for Poisson)
  - A modern laptop CPU delivers >10 GFlops peak

- **Consequence:**
  - We should solve one million unknowns in 0.00275 seconds
  - ~ 3 ns per unknown

- **Revised Assumptions:**
  - Multigrid takes 500 Ops/unknown to solve your favorite PDE
  - you can get 5% of 10 Gflops performance

- **Consequence:** On your laptop you should
  - solve one million unknowns in 1.0 second
  - ~ 1 microsecond per unknown

- Consider Banded Gaussian Elimination on the Play Station (Cell Processor), single Prec. 250 GFlops, for 1000 x 1000 grid unknowns
  - ~2 Tera-Operations for factorization - will need about 10 seconds to factor the system
  - requires 8 GB Mem.
  - Forward-backward substitution should run in about 0.01 second, except for bandwidth limitations
DiMe - Project

Data Local Iterative Methods (DFG 1996-2007) for the Efficient Solution of Partial Differential Equations

www10.informatik.uni-erlangen.de/de/Research/Projects/DiME/

Single core optimization

- Started jointly with Linda Stals (now ANU) in 1996
- Cache-optimizations for sparse matrix/stencil codes (1996-2007)
- Temporal Blocking Strategies
- Also used in Lattice-Boltzmann for CFD

Current Work: Multi-Core

- Temporal Blocking on Multi-Core Architectures:
- see e.g.
Data access optimizations
Loop blocking

- More complicated blocking schemes exist
- Illustration: 2D square blocking
**V(2,2) cycle - bottom line**
(old results)

<table>
<thead>
<tr>
<th>Mflops</th>
<th>For what</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Standard 5-pt. Operator</td>
</tr>
<tr>
<td>56</td>
<td>Cache optimized (loop orderings, data merging, simple blocking)</td>
</tr>
<tr>
<td>150</td>
<td>Constant coeff. + skewed blocking + padding</td>
</tr>
<tr>
<td>220</td>
<td>Eliminating rhs if 0 everywhere (except boundary)</td>
</tr>
</tbody>
</table>
Hierarchical Hybrid Grids (HHG)

Joint work with

Frank Hülsemann (now EDF, Paris), Ben Bergen (now Los Alamos), T. Gradl (Erlangen), B. Gmeiner (Erlangen)

HHG Goal: Ultimate Parallel FE Performance!

- unstructured adaptive refinement grids with
- regular substructures for
- efficiency
- superconvergence effects
Parallel High Performance FE Multigrid

- Parallelize „plain vanilla“ multigrid
  - partition domain
  - parallelize all operations on all grids
  - use clever data structures

- Do not worry (so much) about Coarse Grids
  - idle processors?
  - short messages?
  - sequential dependency in grid hierarchy?

- Why we do not use Domain Decomposition
  - DD without coarse grid does not scale (algorithmically) and is inefficient for large problems/ many processors
  - DD with coarse grids is still less efficient than multigrid and is as difficult to parallelize
HHG refinement example

Input Grid
HHG Refinement example

Refinement Level one
HHG Refinement example

Refinement Level Two
HHG Refinement example

Structured Interior
HHG Refinement example

Structured Interior
HHG Refinement example

Edge Interior
HHG Refinement example

Edge Interior
HHG for Parallelization

- Use regular HHG patches for partitioning the domain
Parallel HHG - Framework
Design Goals

To realize good parallel scalability:

- Minimize latency by reducing the number of messages that must be sent
- Optimize for high bandwidth interconnects → large messages
- Avoid local copying into MPI buffers
HHG Parallel Update Algorithm

for each vertex do
    apply operation to vertex
end for

update vertex primary dependencies

for each edge do
    copy from vertex interior
    apply operation to edge
    copy to vertex halo
end for

update edge primary dependencies

for each element do
    copy from edge/vertex interiors
    apply operation to element
    copy to edge/vertex halos
end for

update secondary dependencies
<table>
<thead>
<tr>
<th>#Proc</th>
<th>10^6 unknowns</th>
<th>Ph.1: sec</th>
<th>Ph.2: sec</th>
<th>Time to sol.</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>268.4</td>
<td>3.27</td>
<td>6.67*</td>
<td>39.3</td>
</tr>
<tr>
<td>16</td>
<td>536.9</td>
<td>3.35</td>
<td>6.75*</td>
<td>40.3</td>
</tr>
<tr>
<td>32</td>
<td>1,073.7</td>
<td>3.38</td>
<td>6.80*</td>
<td>40.6</td>
</tr>
<tr>
<td>64</td>
<td>2,147.5</td>
<td>3.53</td>
<td>4.92</td>
<td>42.3</td>
</tr>
<tr>
<td>128</td>
<td>4,295.0</td>
<td>3.60</td>
<td>7.06*</td>
<td>43.2</td>
</tr>
<tr>
<td>252</td>
<td>8,455.7</td>
<td>3.87</td>
<td>7.39*</td>
<td>46.4</td>
</tr>
<tr>
<td>504</td>
<td>16,911.4</td>
<td>3.96</td>
<td>5.44</td>
<td>47.6</td>
</tr>
<tr>
<td>2040</td>
<td>68,451.0</td>
<td>4.92</td>
<td>5.60</td>
<td>59.0</td>
</tr>
<tr>
<td>3825</td>
<td>128,345.7</td>
<td>6.90</td>
<td></td>
<td>82.8</td>
</tr>
<tr>
<td>4080</td>
<td>136,902.0</td>
<td></td>
<td>5.68</td>
<td></td>
</tr>
<tr>
<td>6102</td>
<td>205,353.1</td>
<td></td>
<td>6.33</td>
<td></td>
</tr>
<tr>
<td>8152</td>
<td>273,535.7</td>
<td></td>
<td>7.43*</td>
<td></td>
</tr>
<tr>
<td>9170</td>
<td>307,694.1</td>
<td></td>
<td>7.75*</td>
<td></td>
</tr>
</tbody>
</table>

Parallel scalability of scalar elliptic problem in 3D discretized by tetrahedral finite elements.

Times to solution on SGI Altix: Itanium-2 1.6 GHz.

Largest problem solved to date: 3.07 x 10^{11} DOFS (1.8 trillion tetrahedra) on 9170 Procs in roughly 90 secs.

HLRB-II

- Shared memory architecture
  - 9728 CPU Cores (Itanium2 Montecito Dual Core)
  - 39 TBytes RAM
  - NUMAlink network
  - 62.3 TFLOP/s Peak Performance

Our testing ground for scalability experiments

HLRB-II: SGI Altix 4700 at the Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften No. 10 in TOP-500 of June 2007
Multigrid Extrapolation

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

Relax on

$$A^h u^h = f^h$$

Restrict

$$r^H = I^H_h r^h$$

Residual

$$r^h = f^h - \frac{4}{3} A^h u^h + \frac{1}{3} A^H H I^H_h u^h$$

Correct

$$u^h \leftarrow u^h + e^h$$

Interpolate

$$e^h = I^h_H e^H$$

Solve

$$A^H e^H = r^H$$

by recursion

...
**τ-Extrapolation**

- τ-Extrapolation is a multigrid specific technique and works for both CS and FAS
- For CS the defects of two different grid levels are combined
- Higher accuracy is achieved by modifying the coarse grid correction at the finest grid level only

\[
\tilde{u}^{m+1}_h = u^m_h + I^H_h A^{-1}_H \left( \frac{4}{3} I^H_h (f_h - A_h u^m_h) - \frac{1}{3} (I^H_h f_h - A_H \hat{I}^H_h u^m_h) \right)
\]

- special care needed
  - when choosing the restriction operator and
  - the smoothing procedure in order not to destroy the higher accuracy

- In 2-D, for (unstructured) triangular meshes τ-Extrapolation is equivalent to using higher order (quadratic) elements. For 3D this is unknown. See:
Model Problem

\[-\Delta u = f \text{ in } \Omega\]

Boundary conditions (Dirichlet) and solution:

\[
u = e^{-20((2x-1+0.5\sin(2z\pi))^2 + (2y-1+0.5\cos(2z\pi))^2)} + e^{-20((2x-1+0.5\sin(2z\pi+\pi))^2 + (2y-1+0.5\cos(2z\pi+\pi))^2)}\]

Computational Domain \(\Omega\)

---

Lehrstuhl für Informatik 10 (Systemsimulation)
## Consistency Error

<table>
<thead>
<tr>
<th>Levels</th>
<th>Unknowns</th>
<th>Correction Scheme Discr. Error</th>
<th>Consistency</th>
<th>$\tau$-extrapolation Discr. Error</th>
<th>Consistency</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>55</td>
<td>$6.75 \cdot 10^{-1}$</td>
<td>-</td>
<td>$6.75 \cdot 10^{-1}$</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>623</td>
<td>$6.53 \cdot 10^{-2}$</td>
<td>3.37</td>
<td>$6.91 \cdot 10^{-2}$</td>
<td>3.29</td>
</tr>
<tr>
<td>5</td>
<td>5.855</td>
<td>$1.90 \cdot 10^{-2}$</td>
<td>1.78</td>
<td>$1.56 \cdot 10^{-2}$</td>
<td>2.14</td>
</tr>
<tr>
<td>6</td>
<td>50.623</td>
<td>$4.95 \cdot 10^{-3}$</td>
<td>1.94</td>
<td>$2.44 \cdot 10^{-3}$</td>
<td>2.68</td>
</tr>
<tr>
<td>7</td>
<td>420.735</td>
<td>$1.25 \cdot 10^{-3}$</td>
<td>1.99</td>
<td>$1.68 \cdot 10^{-4}$</td>
<td>3.86</td>
</tr>
<tr>
<td>8</td>
<td>$3.43 \cdot 10^{6}$</td>
<td>$3.12 \cdot 10^{-4}$</td>
<td>2.00</td>
<td>$1.06 \cdot 10^{-5}$</td>
<td>3.98</td>
</tr>
<tr>
<td>9</td>
<td>$2.80 \cdot 10^{7}$</td>
<td>$7.77 \cdot 10^{-5}$</td>
<td>2.00</td>
<td>$1.95 \cdot 10^{-6}$</td>
<td>2.45</td>
</tr>
</tbody>
</table>
Adaptive Refinement with HHG

- Hanging nodes → non-conforming grids
- Red-green → conforming grids
Adaptivity in HHG (with conforming meshes)
Refinement with Hanging Nodes in HHG

Coarse grid
Fine Grid
Adaptive Grid with Hanging Nodes
Smoothing operation
Residual computation
Restriction Computation

Treating Hanging Nodes as in FAC: see e.g.
- UR: Mathematical and Computational Techniques for Multilevel Adaptive Methods, SIAM, 1993
Conclusions and Outlook
What else do we do?

- Multi-Core-Aware Parallel Multigrid (on Cell, on GPU)
  - also for industrial applications
  - talk by UR this morning in MS „Iterative Solvers for MultiCore Architectures“
- Parallel Rigid Body Dynamics
  - Talk by Klaus Iglberger on Friday
  - Poster by Tobias Preclik
- Graduate Education in CS&E
  - Earlier talk by UR this afternoon in MS „Graduate Education for the Parallel Revolution“
- Parallel Lattice Boltzmann Methods for Complex Flows
  - no talk
- Performance Analysis:
  - Talk by Georg Hager (Erlangen Computing Center) in MS 45, Friday 9:50-11:50 am „Analysis of Hybrid Applications on Modern Architectures“
Acknowledgements

❖ Collaborators
- In Erlangen: WTM, LSE, LSTM, LGDV, RRZE, LME, Neurozentrum, Radiologie, etc.
- Especially for foams: C. Körner (WTM)
- International: Utah, Technion, Constanta, Ghent, Boulder, München, Zürich, Delhi, ...

❖ Dissertationen Projects
- N. Thürey, T. Pohl, S. Donath, S. Bogner (LBM, free surfaces, 2-phase flows)
- M. Kowarschik, J. Treibig, M. Stürmer, J. Habich (architecture aware algorithms)
- K. Iglberger, T. Preclik, K. Pickel (rigid body dynamics)
- J. Götz, C. Feichtinger (Massively parallel LBM software, suspensions)
- C. Mihoubi, D. Bartuschat (Complex geometries, parallel LBM)

❖ (Long Term) Guests in summer/fall 2009:
- Dr. S. Ganguly, IIT Kharagpur (Humboldt) - Electroosmotic Flows
- Prof. V. Buwa, IIT Delhi (Humboldt) - Gas-Fluid-Solid flows
- Felipe Aristizabal, McGill Univ., Canada (LBM with Brownian Motion)
- Prof. Popa, Constanta, Romania (DAAD) Numerical Linear Algebra
- Prof. N. Zakaria, Universiti Petronas, Malaysia
- Prof. Hanke, KTH Stockholm (DAAD), Mathematical Modelling
- several Indian student interns

❖ ~25 Diplom- /Master- Thesis, ~30 Bachelor Thesis
❖ Funding by KONWIHR, DFG, BMBF, EU, Elitenetzwerk Bayern
Thanks for your attention!

Questions?

Slides, reports, thesis, animations available for download at:
www10.informatik.uni-erlangen.de