Scalable Parallel Multigrid for Finite Element Computations

Minisymposium MS8 @ SIAM CS&E 2007

Large Scale Parallel Multigrid

B. Bergen (LSS Erlangen/ now Los Alamos)
T. Gradl, Chr. Freundl (LSS Erlangen)
U. Rüde (LSS Erlangen, ruede@cs.fau.de)

Lehrstuhl für Informatik 10 (Systemsimulation)
Universität Erlangen-Nürnberg
www10.informatik.uni-erlangen.de
SIAM CS&E, Feb 2007
Overview

- Motivation
  - Supercomputers
  - Example Applications of Large-Scale FE Simulation

- **Scalable** Finite Element Solvers
  - **Scalable Algorithms:** Multigrid
  - **Scalable Architectures:**
    - Hierarchical Hybrid Grids (HHG)
    - Parallel Expression Templates for PDE (ParExPDE)

- Results

- Outlook
  - Towards Peta-Scale FE Solvers
Part I

Motivation
HHG Motivation
Structured vs. Unstructured Grids
(on Hitachi SR 8000)

Pet Dinosaur HLRB-I:
Hitachi SR 8000
at the Leibniz-Rechenzentrum
der Bayerischen Akademie der
Wissenschaften
No. 5 in TOP-500 at time of
installation in 2000

- gridlib/HHG MFlops rates for matrix-vector
  multiplication on one node
- compared with *highly tuned* JDS results
  for sparse matrices
Our New Play Station: HLRB-II

HLRB-II: SGI Altix 4700 at the Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften No. 18 in in TOP-500 of Nov. 2006

- Shared memory architecture
  - 4096 CPUs
  - 17.5 TBytes RAM
  - NUMAlink network
  - 26.2 TFLOP/s Peak Performance
  - 24.5 TFLOP/s LINPACK performance
  - will be doubled after 2007 upgrade

- Test ground for scalability experiments
Motivation II: DiMe - Project

Data Local Iterative Methods for the Efficient Solution of Partial Differential Equations

- Cache-optimizations for sparse matrix codes
- High Performance Multigrid Solvers
- Efficient (free surface) LBM codes for CFD

- Collaboration project
  - funded 1996 - 20xy by DFG
Part II-a
Towards Scalable FE Software
Scalable Algorithms: Multigrid
Example: Flow Induced Noise

Flow around a small square obstacle

Relatively simple geometries, but fine resolution for resolving physics

Images by M. Escobar (Dept. of Sensor Technology + LSTM)
What is Multigrid?

- Has nothing to do with „grid computing“
- A general methodology
  - multi-scale (actually it is the „original“)
  - many different applications
  - developed in the 1970s - ...
- Useful e.g. for solving elliptic PDEs
  - large sparse systems of equations
  - iterative
  - convergence rate independent of problem size
  - asymptotically optimal complexity -> algorithmic scalability!
  - can solve e.g. 2D Poisson Problem in ~ 30 operations per gridpoint
  - efficient parallelization - if one knows how to do it
  - best basis for fully scalable FE solvers
Multigrid: V-Cycle

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

Relax on

Correct

Residual

Restrict

Interpolate

Solve

by recursion

...
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
Part II - b

Towards Scalable FE Software

Scalable Architecture

Hierarchical Hybrid Grids and Parallel Expression Templates for PDEs
Hierarchical Hybrid Grids (HHG)

- Unstructured input grid
  - Resolves geometry of problem domain
- Patch-wise regular refinement
  - generates nested grid hierarchies naturally suitable for geometric multigrid algorithms
- New:
  - Modify storage formats and operations on the grid to exploit the regular substructures
- Does an unstructured grid with 100,000,000,000 elements make sense?

HHG - Ultimate Parallel FE Performance!
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
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
HHG for Parallelization

SEND BUFFER

RECEIVE BUFFER

LOCAL BUFFER

Partition 0

MPI

RECEIVE BUFFER

SEND BUFFER

LOCAL BUFFER UNUSED

Partition 1
Parallel HHG - Framework

Design Goals

To achieve 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
ParExPDE

- A library for the user friendly, rapid development of numerical PDE solvers on parallel (super-)computers
- Provides a high level and intuitive user interface without compromising on efficiency
- Regularly refined hexahedral grids
- Support for deformed edges
Introduction to Expression Templates

- Encapsulation of arithmetic expressions in a tree-like structure by using C++ template constructs
- Evaluation of expression happens at compile time
- Avoidance of unnecessary copying and generation of temporary objects

Elegance & Performance
Introduction to Expression Templates

Evaluation of an expression:

```cpp
template <class T>
void Vector::operator=(Expr<T>& expr) {
    for (int i=0; i<_size; i++)
        _values[i] = expr.valueAt(i);
}
```
Part II - c

Towards Scalable FE Software

*Performance Results*
### Node Performance is Difficult!

*(B. Gropp)*

DiMe project: Cache-aware Multigrid (1996- ...)

<table>
<thead>
<tr>
<th>grid size</th>
<th>$17^3$</th>
<th>$33^3$</th>
<th>$65^3$</th>
<th>$129^3$</th>
<th>$257^3$</th>
<th>$513^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>standard</td>
<td>1072</td>
<td>1344</td>
<td>715</td>
<td>677</td>
<td>490</td>
<td>579</td>
</tr>
<tr>
<td>no blocking</td>
<td>2445</td>
<td>1417</td>
<td>995</td>
<td>1065</td>
<td>849</td>
<td>819</td>
</tr>
<tr>
<td>2x blocking</td>
<td>2400</td>
<td>1913</td>
<td>1312</td>
<td>1319</td>
<td>1284</td>
<td>1282</td>
</tr>
<tr>
<td>3x blocking</td>
<td>2420</td>
<td>2389</td>
<td>2167</td>
<td>2140</td>
<td>2134</td>
<td>2049</td>
</tr>
</tbody>
</table>

- Performance of 3D-MG-Smoother for 7-pt stencil in Mflops on Itanium 1.4 GHz
- Array Padding
- Temporal blocking - in EPIC assembly language
- Software pipelining in the extreme (M. Stürmer - J. Treibig)

Node Performance is Possible!
Single Processor HHG Performance on Itanium for Relaxation of a Tetrahedral Finite Element Mesh

![Graph showing performance metrics for different refinement levels and processors.](image-url)
Multigrid Convergence Rates

The graph illustrates the convergence rates over the number of cycles for different methods. The x-axis represents the cycle number, while the y-axis shows the convergence rate. The methods compared are HHG (red solid line) and ParExPDE (blue dashed line).
Scaleup HHG and ParExPDE

![Graph showing scaleup of HHG and ParExPDE](image)
Speedup HHG and ParExPDE
### HHG: Parallel Scalability

<table>
<thead>
<tr>
<th>#Procs</th>
<th>#DOFS x $10^6$</th>
<th>#Els x $10^6$</th>
<th>#Input Els</th>
<th>GFLOP/s</th>
<th>Time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>2,144</td>
<td>12,884</td>
<td>6144</td>
<td>100/75</td>
<td>68</td>
</tr>
<tr>
<td>128</td>
<td>4,288</td>
<td>25,769</td>
<td>12288</td>
<td>200/147</td>
<td>69</td>
</tr>
<tr>
<td>256</td>
<td>8,577</td>
<td>51,539</td>
<td>24576</td>
<td>409/270</td>
<td>76</td>
</tr>
<tr>
<td>512</td>
<td>17,167</td>
<td>103,079</td>
<td>49152</td>
<td>762/545</td>
<td>75</td>
</tr>
<tr>
<td>1024</td>
<td>17,167</td>
<td>103,079</td>
<td>49152</td>
<td>1,456/964</td>
<td>43</td>
</tr>
</tbody>
</table>

Parallel scalability of Poisson problem
discretized by tetrahedral finite elements.
Times for six V(3,3) cycles on SGI Altix (Itanium-2 1.6 GHz).

Largest problem solved to date:
$1.28 \times 10^{11}$ DOFS (45900 Input Els) on 3825 Procs: 7 s per V(2,2) cycle
Why Multigrid?

- A direct elimination banded solver has complexity $O(N^{2.3})$ for a 3-D problem.
- This becomes

\[ 338869529114764631553758 = O(10^{23}) \]

operations for our problem size
- At one Teraflops this would result in a runtime of 10000 years which could be reduced to 10 years on a Petaflops system
Guest Editors-in-Chief:
Chris Johnson
David Keyes
Ulrich Ruede

Call for papers:
- Modeling techniques
- Simulation techniques
- Analysis techniques
- Tools for realistic problems

Deadline:
April 30, 2007
Part IV

Conclusions
Conclusions

Supercomputer Performance is Easy!

- If parallel efficiency is bad, choose a slower serial algorithm
  - it is probably easier to parallelize
  - and will make your speedups look much more impressive
- Introduce the “CrunchMe” variable for getting high Flops rates
  - advanced method: disguise CrunchMe by using an inefficient (but compute-intensive) algorithm from the start
- Introduce the “HitMe” variable to get good cache hit rates
  - advanced version: disguise HitMe within “clever data structures” that introduce a lot of overhead
- Never cite “time-to-solution”
  - who cares whether you solve a real life problem anyway
  - it is the MachoFlops that interest the people who pay for your research
- Never waste your time by trying to use a complicated algorithm in parallel (such as multigrid)
  - the more primitive the algorithm the easier to maximize your MachoFlops.
References


- **ISC Award:** 2006 for Application Scalability.

Acknowledgements

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

■ Dissertationen Projects
  - U. Fabricius (AMG-Verfahren and SW-Engineering for parallelization)
  - C. Freundl (Parelle Expression Templates for PDE-solver)
  - J. Härtelein (Expression Templates for FE-Applications)
  - N. Thürey (LBM, free surfaces)
  - T. Pohl (Parallel LBM)
  - ... and 6 more

■ 19 Diplom-/Master- Thesis
■ Studien- /Bachelor- Thesis
  - Especially for Performance-Analysis/ Optimization for LBM
    - J. Wilke, K. Iglberger, S. Donath
  - ... and 23 more

■ KONWIHR, DFG, NATO, BMBF
■ Elitenetzwerk Bayern
  - Bavarian Graduate School in Computational Engineering (with TUM, since 2004)
  - Special International PhD program: Identifikation, Optimierung und Steuerung für technische Anwendungen (with Bayreuth and Würzburg) since Jan. 2006.
Talk is Over
Questions?

Preview of presentation in MS 92

BGCE Student Paper Prize
MS 83, Thu 9:45 am, Laguna Beach I/II - B1
MS 92, Thu 3:00 pm, Pacific II-B