Is 2.44 trillion unknowns the largest finite element system that can be solved today?

U. Rüde (LSS Erlangen, ruede@cs.fau.de)

Lehrstuhl für Simulation
Universität Erlangen-Nürnberg

ESCO 2014
4th European Seminar on Computing
Pilsen, Czech Republic
June 15 - 20, 2014
Overview

- Motivation
  - How fast our computers are
  - How much is „exa“
  - Some aspects of computer architecture

- Scalable Parallel Multigrid
  - Matrix-Free Multigrid FE solver
  - Hierarchical Hybrid Grids (HHG)

- Multiphysics applications with multigrid and beyond
  - Geodynamics with multigrid: TerraNeo
  - Granular dynamics
  - Complex flows with Lattice Boltzmann Methods

- Conclusions
On the way to Exa-Flops

(High Performance Computer Systems)
How much will be ExaFlops?

- $10^6 = 1$ MegaFlops: Intel 486
  33MHz PC (~1989)
- $10^9 = 1$ GigaFlops: Intel Pentium III
  1GHz (~2000)
  - If every person on earth computes one operation every 7 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
  - 122 400 Cores (Roadrunner, 2008)
  - 294 912 Cores (Jugene, Jülich, 1 PFlops, 1.44 $10^{14}$ Bytes Memory)
  - 155 000 Cores (SuperMuc, 3 PFlops, 3.33 $10^{14}$ Bytes Memory)
  - If every person on earth runs a 486 PC, we all together have an aggregate Performance of 7 PetaFlops.
- ExaScale ($\sim 10^{18}$ Flops) around 2020?
Giga=$10^9$, Tera=$10^{12}$, Peta=$10^{15}$, Exa=$10^{18}$

- World has a population of $7 \times 10^9$ humans
- Earth is $4.6 \times 10^9$ years old
  - the oceans together have ca. $1.3 \times 10^9$ km$^3$
  - the mantle has $0.91 \times 10^{12}$ km$^3$
  - $10^{12}$ finite elements can resolve the volume of the mantle with ca. 1 km resolution

- Number of stars in the galaxy: $10^{11}$
- Avogadro's constant: $6 \times 10^{23}$ mol$^{-1}$
- The recirculatory system contains $2.5 \times 10^{13}$ red blood cells
- The brain has ca. $10^{11}$ Neurons

- Processor chip has
  - $5 \times 10^9$ transistors
  - 3 GHz clock rate: $3 \times 10^9$
  - can perform $10^{11}$ (= 100 Giga) Flops
# Computer Architecture is Hierarchical

## Core
- Vectorization (SSE, AVX), i.e. vectors of 2-8 floating point numbers must be treated in blocks:
- May have their own cache memories
- Access to local (cache) mem fast
- Access to remote mem slow
- Pipelining, superscalar execution
- Each core may need several threads to hide memory access latency

## Node
- Several CPU chips (2) may be combined with local memory to become a node
- Several cores (8) are on a CPU chip
- Within a node we can use „shared memory parallelism“ e.g. OpenMP
- Several cores may share second/third level caches
- Memory access bottlenecks may occur
- Sometimes nodes are equipped with accelerators (i.e. graphics cards)

## Cluster
- Thousands of nodes are connected by a fast network
- Different network topologies
- Between nodes message passing must be used e.g. MPI
- High latency
- Low bandwidth
## What Supercomputers are Like Today

<table>
<thead>
<tr>
<th></th>
<th>JUQUEEN</th>
<th>SuperMUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>IBM Blue Gene/Q</td>
<td>IBM System x iDataPlex</td>
</tr>
<tr>
<td>Processor</td>
<td>IBM PowerPC A2</td>
<td>Intel Xeon E5-2680 8C</td>
</tr>
<tr>
<td>SIMD</td>
<td>QPX (256bit)</td>
<td>AVX (256bit)</td>
</tr>
<tr>
<td>Peak</td>
<td>5 872.0 TFlop/s</td>
<td>3 185.1 TFlop/s</td>
</tr>
<tr>
<td>Clock</td>
<td>1.6 GHz</td>
<td>2.8 GHz</td>
</tr>
<tr>
<td>Nodes</td>
<td>28 672</td>
<td>9 216</td>
</tr>
<tr>
<td>Node peak</td>
<td>204.8 GFlops/s</td>
<td>358.4 GFlops/s</td>
</tr>
<tr>
<td>S/C/T per Node</td>
<td>1/16/64</td>
<td>2/16/32</td>
</tr>
<tr>
<td>GFlops per Watt</td>
<td>2.54</td>
<td>0.94</td>
</tr>
</tbody>
</table>
10^{15} \text{N Joule} \text{ correspond to} \\
277 \text{GWh or} \\
240 \text{Kilotons TNT} \\
\text{If} \ N=10^{12} \text{ algorithms with } O(N^2) \text{ complexity are impossible} \\
the \text{picture shows the Badger-Explosion of 1953 with} \ 23 \text{Kilotons TNT} \\
\text{Source: Wikipedia}
Scalable Algorithms:
Multigrid for FE on Peta-Scale Computers
Quantify the cost of FE solvers a priori and use as design goal

- $10^6$ unknowns
  - memory requirement:
  - solution vector: 8 M Bytes
  - plus 3 auxiliary vectors: 32 MBytes
  - stiffness & mass matrix, assume #nnz per row 15 (linear tet elements): 240 MBytes
  - can save $O(10)$ cost by matrix-free implementation

- assume asymptotically optimal solver (multigrid for scalar elliptic PDE)
  - 100 Flops/unknown
  - efficiency $\eta=0.1$

- machine with
  - 1 GFlops, 100 MByte, should solve: $3 \times 10^6$ unknowns in 3 seconds
  - 1 PFlops, 100 TByte, should solve: $3 \times 10^{12}$ unknowns in 3 seconds
Optimal complexity: Multigrid: V-Cycle

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

- Relax on $A^h u^h = f^h$
- Correct $u^h \leftarrow u^h + e^h$
- Residual $r^h = f^h - A^h u^h$
- Interpolate $e^H = I^H_h e^H$
- Restrict $r^H = I^H_h r^h$
- Solve $A^H e^H = r^H$

(by recursion)
Hierarchical Hybrid Grids (HHG)


- 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
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
Regular tetrahedral refinement

Structured refinement of tetrahedra

Use regular HHG patches for partitioning the domain (only 2D for simplification)

Coarse grid with 132k elements, as assigned to supercomputer

communication of ghost layers
HHG Parallel Update Algorithm

for each vertex do
  apply operation to vertex
  update vertex primary dependencies
for each edge do
  copy from vertex interior
  apply operation to edge
  copy to vertex halo
  update edge primary dependencies
for each element do
  copy from edge/vertex interiors
  apply operation to element
  copy to edge/vertex halos
  update secondary dependencies
Parallel Efficiency of HHG on Different Clusters

- JUGENE
- JUQUEEN
- SuperMUC

Problem Size

Parallel Efficiency
Towards quantitative performance prediction of multigrid for tetrahedra


- Goal: Maintain V-cycle convergence rates <0.2 uniformly by using optimally tuned smoothers in each macro-tetrahedron
- Local Mode Analysis (LFA) for the quantitative prediction of
  - smoothing rate
  - two grid convergence rate
  - V-cycle/ W-cycle performance
- Idea: analyse multigrid in (discrete) Fourier space
  - neglecting effects of boundary conditions
  - quantitative analysis of coupling between high and low frequency modes
  - technically complex in particular for multi-color smoothers
- Generalization of classical multigrid analysis technique to tetrahedral meshes
- Analysis as an algorithm design tool
HHG Pros and Cons

Pro:
- node performance
  - SIMD, superscalar execution, GPUs
- better accuracy through local superconvergence effects
- well suited for parallelization
- tau-extrapolation for higher order
- locally selected line/plane smoothers for better efficiency

Con:
- only restricted adaptivity possible
- only limited ability to handle complex shapes
- how to solve the coarse grid problem
- high implementation effort
- less flexible
Application in Geophysics
TERRA-NEO
Co-Design of an Exascale Earth Mantle Modeling Framework

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

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\mathbf{u})</td>
<td>velocity</td>
</tr>
<tr>
<td>(p)</td>
<td>dynamic pressure</td>
</tr>
<tr>
<td>(T)</td>
<td>temperature</td>
</tr>
<tr>
<td>(\nu)</td>
<td>viscosity of the material</td>
</tr>
<tr>
<td>(\varepsilon(u) = \frac{1}{2}(\nabla \mathbf{u} + (\nabla \mathbf{u})^T))</td>
<td>strain rate tensor</td>
</tr>
<tr>
<td>(\rho)</td>
<td>density</td>
</tr>
<tr>
<td>(\kappa, \gamma, \mathbf{g})</td>
<td>thermal conductivity, heat sources, gravity vector</td>
</tr>
</tbody>
</table>

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

DFG SPP 1648/1 - Software for Exascale Computing

Stokes equation:
\[-\text{div}(\nabla \mathbf{u} - p \mathbf{I}) = \mathbf{f}, \quad \text{div} \mathbf{u} = 0\]

FEM Discretization:
\[a(\mathbf{u}_l, \mathbf{v}_l) + b(\mathbf{v}_l, p_l) = L(\mathbf{v}_l) \quad \forall \mathbf{v}_l \in \mathbf{V}_l,\]
\[b(\mathbf{u}_l, q_l) - c(p_l, q_l) = 0 \quad \forall q_l \in \mathbf{Q}_l,\]
with:
\[a(\mathbf{u}, \mathbf{v}) := \int_{\Omega} \nabla \mathbf{u} : \nabla \mathbf{v} \, dx, \quad b(\mathbf{u}, q) := -\int_{\Omega} \text{div} \mathbf{u} \cdot q \, dx\]

Schur-complement formulation:
\[
\begin{bmatrix}
A_l & B_l^T \\
0 & C_l + B_l A_l^{-1} B_l^T
\end{bmatrix}
\begin{bmatrix}
\mathbf{u}_l \\
p_l
\end{bmatrix}
= \begin{bmatrix}
\mathbf{f}_l \\
B_l A_l^{-1} \mathbf{f}_l
\end{bmatrix}
\]
Geodynamics

Starting with an icosahedron and refining until 10^12 FE are reached.
## Scalability on JuQueen for Stokes

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Threads</th>
<th>Grid points</th>
<th>Resolution</th>
<th>Time: (A)</th>
<th>(B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30</td>
<td>(2.1 \cdot 10^7)</td>
<td>32 km</td>
<td>30 s</td>
<td>89 s</td>
</tr>
<tr>
<td>4</td>
<td>240</td>
<td>(1.6 \cdot 10^8)</td>
<td>16 km</td>
<td>38 s</td>
<td>114 s</td>
</tr>
<tr>
<td>30</td>
<td>1920</td>
<td>(1.3 \cdot 10^9)</td>
<td>8 km</td>
<td>40 s</td>
<td>121 s</td>
</tr>
<tr>
<td>240</td>
<td>15360</td>
<td>(1.1 \cdot 10^{10})</td>
<td>4 km</td>
<td>44 s</td>
<td>133 s</td>
</tr>
<tr>
<td>1920</td>
<td>122880</td>
<td>(8.5 \cdot 10^{10})</td>
<td>2 km</td>
<td>48 s</td>
<td>153 s</td>
</tr>
<tr>
<td>15360</td>
<td>983040</td>
<td>(6.9 \cdot 10^{11})</td>
<td>1 km</td>
<td>54 s</td>
<td>170 s</td>
</tr>
</tbody>
</table>

![Bar graph showing computation and communication](image)

### Computation and Communication

- **Pressure correction**
- **Solve momentum**
- **Multigrid, finest level**
- **Multigrid, remaining levels**
- **Multigrid, CG, residual**
- **Multigrid, CG, scalar product**

Ulrich Rüde - Lehrstuhl für Simulation
Granular Media Simulations

with the *Pe* physics engine

with T. Preclik and K. Iglberger
Parallel Rigid Body Dynamics

Tobias Preclik

Sharp-edged Granular Matter in a High Amplitude Horizontal Shaker

31.3.2014

Not just point masses, but volumetric, geometrically defined objects
Computational Granular Dynamics

- Alternative contact models
  - FFD - Fast Frictional Dynamics (D. Kaufman et. al. 2005)
  - smooth: Discrete Element Method (DEM)
  - nonsmooth: Linear (LCP)/ Nonlinear Complementarity (NCP)
- MPI parallelization scaling up to full size
- largest simulation up to date: 28,311,552,000 fully resolved particles (NCP solver)
- novel contact model (T. Preclik) based on
  - generalized maximum dissipation principle
  - large-scale MPCC = mathematical program with complementarity constraints

Collisions & Contacts between Geometric Objects
Computational Fluid Dynamics in waLBerla with the Lattice Boltzmann Method

Falling Drop with Turbulence Model (slow motion) with N. Thuerey
Simulation of Flow in Vocal Folds

walBerla framework for massively parallel LBM
static load balancing of adaptive mesh structures
with F. Schornbaum, C. Godenschwager, M. Bauer, E. Fattahi
Weak scaling (Lid Driven Cavity) TRT

JUQUEEN
16 processes per node
4 threads per process

SuperMUC
4 processes per node
4 threads per process

837 billion cell updates per second (GLups)

2.1 trillion cell updates per second (TLups)
Fluid-Structure Interaction for Particle Laden Flows

- Collision detection
- Frictional collision response
- Time integration
- Rigid bodies act as obstacles
- Fluid results in external forces
- Update of fluid nodes: stream/collide
- Calculation of hydrodynamic forces (momentum exchange)
Tumbling Fibers

with D. Bartuschat and K. Gustavsson (KTH Stockholm):
validation against integral eqn/slender body approximation in Stokes flow
Virtual Fluidized Bed

512 processors

Simulation Domain
Size: 180x198x360 cells of LBM

900 capsules and 1008 spheres
= 1908 objects

Number time steps:
252,000

Run Time:
07h 12 min
Weak Scaling

Jugene
Blue Gene/P

Largest simulation to date: 8 Trillion ($10^{12}$) variables per time step

Densely packed particles

264 331 905 rigid spherical objects
ESCO 2014

Monday session I-D: 14:20 - 14:40:
S. Bogner: Lattice Boltzmann Study of the Drag Correlation for Dilute and Moderately Dense Fluid-particle Systems

Monday session II-B: 15:40 - 17:00: Simulation and Coupling Strategies for Particle Dynamics and CFD (chairs P. Neumann, U. Ruede):

- H. Hasse: Molecular Dynamics Simulations in Engineering
- K. Gustavsson: Large Scale Simulations of Rigid Fiber Suspensions
- D. Bartuschat: Parallel Multiphysics Simulations of Charged Particles in Microfluidic Flows
- X. Bian: Multiscale Simulation Using Lagrangian Particles

Tuesday session I-A 10:50 - 11:50: (part II)

- P. Neumann: Adaptive Mesh Refinement in Lattice Boltzmann Simulations
- N. Tchipev: Molecular Dynamics on 146 000 Cores
- C. Rauh: Thermofluiddynamical Numerical Simulations of Particle-laden Flows in (Bio) Chemical Fluidized Bed Reactors

Wednesday session II-B: 10:40 - 11:00
R. Ammer: Simulating Additive Manufacturing Processes With a 3D Free Surface Lattice Boltzmann Method
Conclusions

- Progress in computer technology and carefully designed algorithms enable FE and flow simulations in excess of $10^{12}$ Dofs ... and it keeps growing
- All computer systems are parallel and we are not well prepared for this disruptive change
- We need a new algorithm engineering methodology, based on a better performance analysis and prediction
- Co-Design of Apps, Models, Discretization, Solver, Software, and Parallelization
Thank you for your attention!

Video generated by S. Bogner at LSS with the massively parallel waLberla Software framework for Lattice Boltzmann based multi-physics applications.

Slides, reports, thesis, animations available for download at:
www10.informatik.uni-erlangen.de
Some examples of Coupled Multi Physics Simulations

Ulrich Rüde - Lehrstuhl für Simulation