- Part 4 -
Multicore and Manycore Technology: Chances and Challenges

Vincent Heuveline
Numerical Simulation of Tropical Cyclones
“Goal oriented adaptivity for tropical cyclones”
Numerical Simulation of Tropical Cyclones

**Meteorology**
- Dynamics of tropical cyclones
- Numerical models of tropical cyclones

**Mathematics**
- Numerical solution of PDEs,
  - goal oriented adaptive FE methods
- High performance computing

Adaptive techniques applied to TC problems
Numerical Simulation of Tropical Cyclones

- Multiple scales in space and time are relevant for the dynamics of the atmosphere

- In large domains, smallest scales cannot be resolved → adaptive methods
- Goal-oriented adaptivity: User-defined features of interest are in focus
Numerical Simulation of Tropical Cyclones

- Goal-functional: \( J(v) := \int_{B(x_0, r)} (\nabla \times v)(T, x) \, dx, \quad T = 96h \)

Primal Solution

Dual Solution

Vorticity

Velocity
Numerical Simulation of Tropical Cyclones

- Small perturbations of the data → big perturbation of the solution

- Sensitivity
  - Linear approaches
  - Adjoint methods
  - Nonlinear methods

Perturbation of initial state (t=0h)  Developed perturbation (t=12h)

- Evolution of the optimal perturbation (i.e. adjoint solution)

MetStröm
L. Scheck (IMK-TRO)
A Simple Performance Model

Run time: \( T_R \geq \max \{ T_C, T_T \} \)
- \( T_R \) … total run time of an algorithm
- \( T_C \) … compute time, \( T_C \geq f / P \)
- \( T_T \) … transfer time, \( T_T \geq 8w / B \)

Classifies compute-bound or memory-bound algorithms

Algorithm characteristics
- \( f \) … number of floating point operations
- \( w \) … number of memory transfers (words)

Hardware characteristics
- \( P \) … peak performance
- \( B \) … memory bandwidth

We find: effective performance \( P_{eff} \leq f / T_R \leq fB / (8w) \)
- Ratio \( f / w \) defines computational intensity
Computational intensity

Computational intensity =

# floating point operations per transferred byte

All basic operations have computational intensity of $O(1)$!

Consequences:
- Bandwidth limitations on many devices
- Only a small fraction of peak performance can be achieved
### Computational intensity

- **O(1)**
  - Stencil
  - Blas 1+2
  - SpMV
  - LBM

- **O(log(N))**
  - FFT

- **O(N)**
  - Particle methods
  - BLAS 3

**Numerical simulation of PDEs and CFD**
Chorin-based Navier Stokes Solver

- Incompressible Navier-Stokes equations

\[ \partial_t u - \nu \Delta u + (u \cdot \nabla)u + \nabla p = f \quad \text{in} \quad \Omega \]

\[ \nabla \cdot u = 0 \quad \text{in} \quad \Omega \]

- Chorin-type projection method: iterative scheme, k=0, 1, ...

  - Compute \( \tilde{u}^{k+1} \):
    \[ \frac{\tilde{u}^{k+1} - u^k}{\Delta t} + (u^k \cdot \nabla)u^k - \nu \Delta u^k = f^k \quad \text{in} \quad \Omega \]

  - Compute \( p^{k+1} \):
    \[ \Delta p^{k+1} = \frac{1}{\Delta t} \nabla \tilde{u}^{k+1} \quad \text{in} \quad \Omega \]

  - Compute \( u^{k+1} \):
    \[ \frac{u^{k+1} - \tilde{u}^{k+1}}{\Delta t} = -\nabla p^{k+1} \quad \text{in} \quad \Omega \]
Chorin-based Navier Stokes Solver
Algorithm Characteristics

- Four basic operations (typical of solution of PDEs)
  - Sparse matrix-vector multiplication / stencil
  - (Non)-linear stencil operations
    - Nearest neighbor interaction
  - SAXPY / DAXPY vector updates
  - Dot products
- Encapsulated by iterative methods and time stepping schemes

- Characteristics
  - Huge memory requirements / fine mesh resolution
  - Low computational intensity of order O(1)
  - Bandwidth-bound algorithms
  - Non-uniform treatment of boundary conditions
Elements of projection step

- LSE in projection step solved with conjugate gradient method (cg)
- Number of cg-steps per time step depends on $N^{1/3}$
- Each cg-step consists of

<table>
<thead>
<tr>
<th>Function</th>
<th>Occ.</th>
<th>$f$ [#flop]</th>
<th>$w$ [#words]</th>
<th>$f / w$</th>
<th>Comp. int.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stencil operation</td>
<td>1</td>
<td>8N</td>
<td>2N</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td>Vector norm</td>
<td>1</td>
<td>2N-1</td>
<td>N+1</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>Dot product</td>
<td>1</td>
<td>2N-1</td>
<td>2N+1</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>Normalization</td>
<td>4</td>
<td>2N</td>
<td>2N</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>DAXPY vector update</td>
<td>3</td>
<td>2N</td>
<td>3N+1</td>
<td>0.66</td>
<td></td>
</tr>
</tbody>
</table>

- Worst performance for DAXPY vector update
  - Easy routine with no data dependencies
  - Perfectly parallelizable on coarse and fine-grained platforms
Seven Bridges over the Gap?
Energy-efficient processor technology

ACPI – Advanced Configuration and Power Interface

- P–states (performance states)
  - DVFS – dynamic voltage and frequency scaling

- C–states (power states)

<table>
<thead>
<tr>
<th>Intel E5504</th>
<th>voltage [V]</th>
<th>freq. [GHz]</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td>1.04</td>
<td>2.00</td>
</tr>
<tr>
<td>P1</td>
<td>1.01</td>
<td>1.87</td>
</tr>
<tr>
<td>P2</td>
<td>0.98</td>
<td>1.73</td>
</tr>
<tr>
<td>P3</td>
<td>0.95</td>
<td>1.60</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AMD 6128</th>
<th>voltage [V]</th>
<th>freq. [GHz]</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td>1.23</td>
<td>2.00</td>
</tr>
<tr>
<td>P1</td>
<td>1.17</td>
<td>1.50</td>
</tr>
<tr>
<td>P2</td>
<td>1.12</td>
<td>1.20</td>
</tr>
<tr>
<td>P3</td>
<td>1.09</td>
<td>1.00</td>
</tr>
<tr>
<td>P4</td>
<td>1.06</td>
<td>0.80</td>
</tr>
</tbody>
</table>
Benchmark problem
Domain Decomposition Parallelisation

\[ Ax = b \]
\[ F(x) = 0 \]
\[ [\nabla F(\hat{x})]x = -F(\hat{x}) \]
Multigrid Cycle

types of operation

fixed no. of smoother iterations:
vector scaling & addition, SpMV,
element-wise operations

grid transfer:
SpMV, element-wise operations

solve error equation:
vector scaling & addition, SpMV,
scalar product

V-cycle

problem sizes

\( n_h \)

\( n_{2h} \approx \frac{n_h}{2d} \)

\( n_{4h} \approx \frac{n_h}{4d} \)

\( n_{8h} \approx \frac{n_h}{8d} \)
Multigrid Cycle
Dynamic adjustment of hardware activity

Adjust hardware activity according to solver needs

kill(pid, SIGUSR1);

pause();
Results: Time and energy to solution

level 0  263 Thsd DoF
...
level 4  1 Thsd DoF

time to solution

global optimal time to solution
adapted C
4 x 8-core CPU

saves 17%

energy to solution

global optimal energy to solution
adapted C
2 x 8-core CPU

saves 21%

best default case
Results: C-states and Intel RAPL

energy savings due to temporary CPU deactivation

multigrid cycle is on coarse grid

multigrid cycle is on fine grid
Mixed precision iterative refinement

- **Initial solution**: $x_{\text{high}}$
- **Compute residual**: $r_{\text{high}} = b_{\text{high}} - A_{\text{high}} x_{\text{high}}$
- **Update solution**: $x_{\text{high}} \leftarrow x_{\text{high}} + c_{\text{high}}$
- **Check stopping criterion**: $\| b_{\text{high}} - A_{\text{high}} x_{\text{high}} \| < \epsilon$
- **Compute correction**: $A_{\text{low}} c_{\text{low}} = r_{\text{low}}$

**High precision format**

**Final solution**: $x_{\text{high}}$

**Low precision format**
On the road to exascale …

raising energy awareness in HPC
Thank you

Energy savings are possible while maintaining performance